SEVENTH FRAMEWORK PROGRAMME

Information & Communication Technologies
Trustworthy ICT

NETWORK OF EXCELLENCE

A European Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet: *Europe for the World* [†]

Deliverable D2.3: 1st Project Workshop Proceedings

Abstract: This document contains the pre-proceedings of the *SysSec* 1st Project Workshop, which took place in Amsterdam on July the 6th, co-located with the DIMVA 2011 conference.

Responsible Partner	**Politecnico di Milano**
Contractual Date of Delivery	August 2011
Actual Date of Delivery	August 2011
Deliverable Dissemination Level	Public
Editor	Federico Maggi
QMC Reviewer	VU

The *SysSec* consortium consists of:

FORTH-ICS	Coordinator	Greece
Politecnico Di Milano	Principal Contractor	Italy
Vrije Universiteit Amsterdam	Principal Contractor	The Netherlands
Institut Eurécom	Principal Contractor	France
IICT-BAS	Principal Contractor	Bulgaria
Technical University of Vienna	Principal Contractor	Austria
Chalmers University	Principal Contractor	Sweden
TUBITAK-BILGEM	Principal Contractor	Turkey

[†] The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 257007.

In this document, the pre-proceedings are formatted according to the style of the *SysSec* deliverables, whereas the actual proceedings will be printed by the IEEE Computer Society according to their formatting rules.

This document is divided into three chapters:

- Chapter 1: Contains the pre-proceedings' frontmatter (preface, workshop organizers, program committee, publication chair and list of external reviewers).
- Chapter 2: Contains copies of the accepted student (research) papers as they will appear in the actual proceedings.
- Chapter 3: Contains copies of the accepted research roadmap (position) papers as they will appear in the actual proceedings.

Contents

1 Frontmatter

1.1 Preface

The first SysSec workshop aims to mobilize the Systems Security research community in Europe. Organized by SysSec, a Network of Excellence for Managing Threats and Vulnerabilities for the Future Internet, this workshop is a first step towards creating a virtual center of excellence which will consolidate the Systems Security research community in Europe and provide links to the rest of the world.

We are happy to report that the research community responded positively to the call of SysSec. Thus, we are proud to present a program consisting of 29 papers (23 position–vision papers and 6 research papers) written by 95 co-authors from 35 organizations—most of them from Europe.

The resulting program will spark lively discussions and will capture the vision of the Systems Security community in Europe in mid 2011. We believe that this program will serve as a time capsule: We hope that people who will look back at it after several years will admire the predicted threats, will smile at the threats which never became a reality, and will remember the words of Nobel prize laureate Niels Bohr: "Prediction is very difficult, especially about the future".

Amsterdam, 6 July 2011

Evangelos Markatos, co-chair
Stefano Zanero, co-chair

1.2 Program Chairs

Evangelos Markatos (FORTH-ICS)

Stefano Zanero (Politecnico di Milano)

1.3 Program Committee

Magnus Almgren, Chalmers

Michael Bailey, University of Michigan

Kiril Boyanov, IICT-BAS, Bulgaria

Marco Cova, University of Birmingham

Herv Debar, Tlcom SudParis/RST

Thorsten Holz, Ruhr University Bochum

Sotiris Ioannidis, FORTH, Greece

Grgoire Jacob, University of California, Santa Barbara

Andrea Lanzi, Institute Eurecom

Corrado Leita, Symantec Research Europe

Federico Maggi, Politecnico di Milano

Paolo Milani Comparetti, Technical University Vienna

Ali Rezaki, TUBITAK-UEKAE, Turkey

William Robertson, UC Berkeley

1.4 Publications Chair

Federico Maggi, Politecnico di Milano

1.5 List of External Reviewers

Dimitris Antoniadis

Elias Athanasopoulos

Leyla Bilge

Lorenzo Cavallaro

Eleni Gessiou

Daniel Hedin

Georgios Kontaxis

Andreas Larsson

Laertis Loutsis

Antonis Papadogiannakis

Athanasios Petsas

Michalis Polychronakis

Guido Salvaneschi

Luigi Sportiello

F.-X. Standaert

Gianluca Stringhini

Giorgos Vasiliadis

2 Student Papers

This chapter contains copies of the accepted student (research) papers as they will appear in the actual proceedings.

Unity in diversity: Phylogenetic-inspired techniques for reverse engineering and detection of malware families

Wei Ming Khoo
University of Cambridge
wmk26@cam.ac.uk

Pietro Lió
University of Cambridge
pl219@cam.ac.uk

***Abstract*—We developed a framework for abstracting, aligning and analysing malware execution traces and performed a preliminary exploration of state of the art phylogenetic methods, whose strengths lie in pattern recognition and visualisation, to derive the statistical relationships within two contemporary malware families. We made use of phylogenetic trees and networks, motifs, logos, composition biases, and tree topology comparison methods with the objective of identifying common functionality and studying sources of variation in related samples. Networks were more useful for visualising short `nop`-equivalent code metamorphism than trees; tree topology comparison was suited for studying variations in multiple sets of homologous procedures. We found logos could be used for code normalisation, which resulted in 33% to 62% reduction in the number of instructions. A motif search showed that API sequences related to the management of memory, I/O, libraries and threading do not change significantly amongst malware variants; composition bias provided an efficient way to distinguish between families. Using context-sensitive procedure analysis, we found that 100% of a set of memory management procedures used by the FakeAV-DO and "Skyhoo" malware families were uniquely identifiable. We discuss how phylogenetic techniques can aid the reverse engineering and detection of malware families and describe some related challenges.**

***Keywords*-malware analysis; phylogenetics;**

I. Introduction

Today's malware is written to be persistent. Financial incentives are the dominant motivation for writing and spreading malware, and making sure that the malware remains as long as possible on the victims' machines. As a result of this, malware exist in families, often numbering in the thousands, in order to constantly evade anti-virus products and operating systems defences. The task of analysing all these variants is resource-intensive and automating the process of reversing and classifying samples is inevitable as the current malware trend continues.

Like human viruses, computer-bourne malware has been co-evolving with the operating systems that they target as well as the external environment. Almost all of today's malware do not exist unobfuscated or unencrypted. Malware samples usually contain a packer which implement code compression and code entry point obfuscation. Other forms of obfuscation frequently employed include virtual machine detection, anti-debugging routines, and code metamorphism. However, malware is seldom written from scratch. Because new malware variants are usually inspired by previous ones, at some level they show a convergence of functionality. Analysis of the Internet worm Conficker [16] showed that there was a 35% overlap in functional prototypes based on hashes of subroutines and basic blocks, as well as similar domain generation algorithms between variants A and B.

In biology, phylogenetic methods help to extract statistical relationship information from multiple human viral sequences. Phylogeny differs from taxonomy in that taxonomy is a grouping based on shared characteristics, whereas phylogeny assumes a common ancestor and seeks to derive evolutionary relationships between the ancestor and evolved species. Sequences or structures that are conserved between distantly related samples are likely to be important or essential for the functioning of the virus. Conversely, regions that differ are equally interesting for studying evolutionary relationships between samples.

The key questions that we want to address are

1) What portions of malware families are most conserved?
2) Conversely, what portions of malware families are most diverse?
3) Since malware exists in families, can we leverage on malware variants for the task of reverse engineering and detection?

We make the following contributions.

1) We developed a tool, Chronicler, to perform full execution capture (Section II). Chronicler logs all instructions executed, memory modifications, register modifications so that post-hoc program analysis can be done using scripts.
2) We describe a framework to abstract and align instruction sequences in order to perform analysis at two levels—the API level and the procedure level (Sections III and IV).
3) We demonstrate how phylogenetic trees and networks, tree topology comparisons, motif searches, sequence biases can be used for reverse engineering and detecting two malware families. At the API level, we observe that system calls related to memory manage-

ment, I/O, dynamic library management and threading are most conserved amongst malware variants (Section V-D). At the instruction level, we show how logos can be used for identification of metamorphic code transformations (Section V-C).

4) Instead of classifying the whole malware binary, we propose context-sensitive procedure analysis (Section VI). We tested this method on a set of malicious and benign procedures and were able to uniquely identify 100% of the malicious ones.

II. Full execution capture

A tool, Chronicler, was written to track and log the execution using the PIN binary instrumentation framework [12]. The advantage of using dynamic binary instrumentation is it allows full control of the execution and developing analysis tools is easy. However, such frameworks can have a high performance overhead compared to native execution.

Because of the performance overhead, full execution capture is performed, that is, all memory read and writes, register read and writes, disassembly, instruction pointer address and debugging symbols are logged. In order to reduce the size of the resulting logs, they were compressed using the zlib compression library [11]. High compression ratios of about 0.5% were achieved and log sizes were about 10 megabytes per million instructions. Post-hoc analysis of the execution trace is subsequently done using scripts.

The advantage of full execution capture is several analyses can subsequently be performed in parallel. One disadvantage is that the resulting logs are large. Because of this, the file size was limited to 500 Mb per sample, equivalent to about 50 million instructions.

III. Program abstraction

Program abstraction is a process by which a program is defined with a representation of its semantics, while hiding its implementation details. Examples include n-grams [6], instruction mnemonics [18], API calls [1], and control flow graphs [5]. As an initial study, we made use of instruction mnemonics and API calls because they can be easily mapped to an alphabet and fed to existing phylogenetic tools.

For example, given an execution trace of instructions,

```
push ebp
mov ebp, esp
mov dword ptr [ebp-0x4]
jmp +0x14
```

it is abstracted as a sequence of mnemonics, i.e.

```
push, mov, mov, jmp
```

ignoring the operands. Each mnemonic is then mapped to a unique alphabet-pair, e.g. `mov = MO`, `push = PH`, `jmp = JM`. The resulting sequence is thus `PHMOMOJM`. We did not distinguish between addressing modes, e.g. `mov eax, ebx` versus `mov eax, [ebx]` and for instructions with prefixes such as `repeat`, the prefix was used. A similar mapping scheme was used for API calls. Each API was mapped to a unique alphabet-pair; arguments and return values were ignored.

IV. Sequence alignment

Sequence alignment is a process of arranging two or more sequences placed one below each other, for their full length (global alignment) or for short subsequences with highest similarity (local alignment). A scoring system rewards with a positive score those positions at which the sequences agree and with a negative score (a penalty) those positions where there is a disagreement ("mutation") and insertion of a blank ("gap"). Regions of similarity are called homologous regions. Regions that cannot be well aligned, i.e. their scores are similar to that of random sequences, are simply ignored according to this scheme [10]. A commonly used scoring system makes use of substitution matrices which assigns scores based on the evolutionary probability of the mutation. A substitution matrix is a 20 by 20 matrix, where each entry (i, j) is the probability of protein i being substituted with protein j. The point accepted mutation (PAM) substitution matrices were introduced by Dayhoff [2] and were based on observed mutations in 71 families of closely related proteins. Since we are modeling instructions and not proteins, the matrix can be determined empirically or by simply using an identity matrix.

In the context of malware analysis, sequence alignment can be used for tasks such as code normalisation via logos (Section V-C), studying malware evolution via trees and networks (Section V-A) and classification (Section VI), just to name a few.

Sequences can be aligned by leveraging on structural similarities, e.g. aligning along basic blocks. The main disadvantage of doing so is the number of aligned sequences can potentially be large. Secondly, malware writers may not use the standard call-ret calling convention.

Another method of sequence alignment is to align based on functional similarities, by looking at the program semantics, or indirectly by looking at the context, for example determining whether it is running in user space or in kernel space. In the latter, the main idea is that kernel space code, which consists of APIs, should be functionally similar across different malware samples. Furthermore, sequences preceding and following from the same API call should behave in a similar fashion.

The alignment strategy adopted is as follows. The execution trace is first divided into contiguous sequences which can be kernel sequences or user sequences. The boundaries of each sequence are code transitions either from user space to kernel space or from kernel to user space. Subsequently, we conduct analysis at two levels, the first is at the API-level. To perform API-level analysis, each kernel sequence

is mapped to an alphabet-pair and concatenated to form more succinct API sequences. Multiple API sequences are aligned using the CLUSTAL method [7] in which an identity substitution matrix is used in scoring, and the neighbour joining method is used in clustering.

The second level of analysis is performed at the malware procedure level. Procedures are continuous instruction sequences that are executed before and after API calls. For the purposes of analysis, this sequence is treated as a single procedure, which may be made up of more than one function. Procedure-level alignment is similarly performed using the CLUSTAL method. Code that is re-executed, e.g. in a for- or while-loop, appear as repeated sequences. To deal with the difficulty of aligning loops in several different sequences, the instruction pointer address is logged and only the first instance of the instruction is recorded.

A. Case study: Compiler options and optimisation

As an initial experiment, we investigated the use of sequence alignment to compare code generated by different compiler options and optimisation.

As an example, we used the well-known triangle program (Figure 1), which takes the three lengths of the sides of a triangle, and returns what kind of triangle it is. The program was compiled in Microsoft Visual Studio using three different compiler settings—with debugging symbols (`dbg`), default settings (`def`) and optimised for speed (`spd`). The objective of this experiment was to compare the effect of different substitution matrices on the alignment. The goal was to align the structural similarities, represented by the seven `cmp` instructions, and functional similarities, represented by the three `imul` instructions. From a reverse engineering perspective, the `cmp` instructions are important as they reveal the control flow of this program execution; the `imul` instructions are important as they hint at the fact that the program is computing the square of the three arguments. The more these instructions were aligned, the better the overall alignment.

The mnemonic sequences were extracted from the three execution traces from the start to the end of the `classify` subroutine. Figure 2 shows the three sequences before and after alignment for a right-angled triangle with sides 3, 4 and 5. When an identity matrix was used, the seven `cmp` instructions were aligned but only one out of three `imul` instructions were. However, when a PAM matrix was used, only four `cmp` instructions and one `imul` instructions were aligned. Thus the identity matrix out-performed the PAM matrix, which was to be expected as the PAM matrix was modeled after protein substitution probabilities. The identity matrix was used for the rest of our experiments.

```
1  int classify(int a, int b, int c){
2     int kind = UNKNOWN_TRIANGLE;
3     if(a+b<=c || b+c<=a || c+a<=b)
4        return INVALID_TRIANGLE;
5     if(a*a+b*b==c*c || b*b+c*c==a*a || c*c+a*a==b*b)
6        kind |= RIGHT_TRIANGLE;
7     else if (a*a+b*b>c*c || b*b+c*c>a*a || c*c+a*a>b*b)
8        kind |= ACUTE_TRIANGLE;
9     else
10       kind |= OBTUSE_TRIANGLE;
11
12    if(a==b || b==c|| c==a)
13       if(a==b && b==c)
14          kind |= EQUILATERAL_TRIANGLE;
15       else
16          kind |= ISOSCELES_TRIANGLE;
17    else
18       kind |= SCALENE_TRIANGLE;
19    return kind;
20 }
```

Figure 1. The triangle program

V. Phylogeny

Reconstruction of molecular phylogenetic relationships is typically done by building a mathematical model describing the evolution of the virus of interest in order to estimate the distance between each sequence pair. A model can be built empirically, using properties calculated through comparisons of observed virus strings, or parametrically, using logic properties of the instructions. An example of the former is to use the expected number of substitutions per position that have occurred on the evolutionary lineages between them and their most recent common ancestor (if any). The main disadvantage of this model is that sequences have to be aligned beforehand.

A. Trees and networks

Distances may be represented as branch lengths in a phylogenetic tree; the extant sequences form the tips of the tree, whereas the ancestral sequences form the internal nodes and are generally not known. Phylogenetic networks are a generalisation of trees for modeling uncertainty about which bifurcation in the tree comes first, thus allowing for different evolutionary paths to be possible. Networks are also useful for modeling horizontal gene transfer, that is, the passage of strings from one child to another instead of from parent to child.

To demonstrate the usefulness of networks over trees, we made use of two datasets. The first was a set of sequences from a procedure "f_1" found in the FakeAV-DO malware family. The FakeAV-DO malware family is a trojan that displays fake alerts that coax users into buying rogue antivirus products. Sequences of "f_1" contain multiple redundant code sequences mechanically inserted by a metamorphic engine described in more detail in Section V-C. To simulate a different model of evolution, a second dataset was generated comprising a set of execution traces from the triangle program, by running it with different arguments.

While some relationships in the trees (Figures 3a and 3c) are obviously closer, e.g. `t344` and `t345`, differences in the two datasets are much clearer by looking at the corresponding networks (Figures 3b and 3d). The branches of the

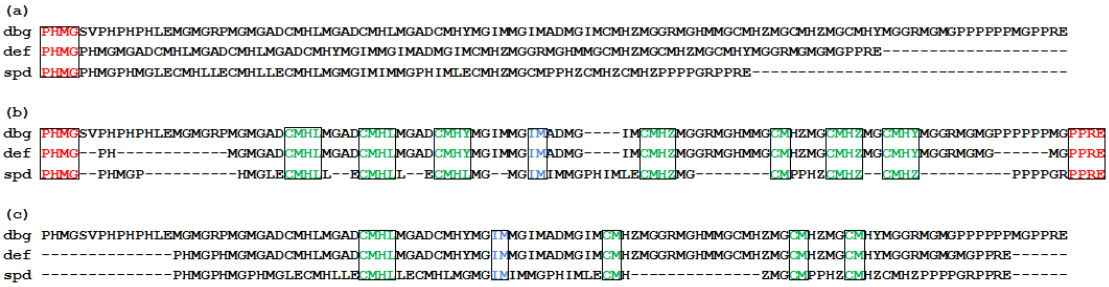


Figure 2. Sequence alignment (dbg: with debugging symbols, def: default settings, spd: optimised for speed). (a) Before alignment. (b) After alignment using an identity substitution matrix. (c) After alignment using a PAM substitution matrix.

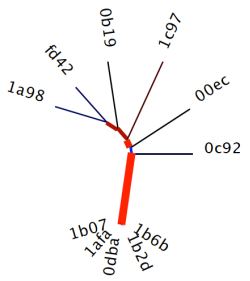


Figure 4. A neighbour joining tree of FakeAV-DO set of procedures F_1.

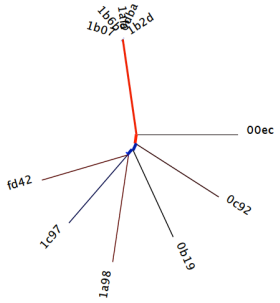


Figure 5. Neighbor joining tree of FakeAV-DO set of procedures F_2 from the same samples as Figure 4.

FakeAV-DO network are widely spread out implying that the evolution is more varied, and that the differences between samples are significant and not simple. Moreover, there are more "ancestor" nodes in the FakeAV-DO network compared to the triangle network, eight versus three, implying that the evolutionary path is more complex and uncertain. We note that the generation of trees and networks depends on the sequence alignment, and in particular the substitution matrix used. Depending on the matrix used, different alignments, and in turn different trees and networks, will result.

B. Comparing tree topologies

Trees constructed from different phylogenetic methods for the same sequences can be compared so as to highlight the parts of the trees that differ, both in terms of topology and branch length [15]. This method is useful for detection changes, e.g. code rearrangements or replacements, in different sets of procedures. Two sets of homologous procedures, $F_1 = \{f_{1,1}, f_{1,2}, \ldots, f_{1,10}\}$ and $F_2 = \{f_{2,1}, f_{2,2}, \ldots, f_{2,10}\}$, were extracted from ten FakeAV-DO execution traces. A comparison of the tree topology of F_1 and F_2 (Figures 4 and 5) shows a topological score of 65.4%. The thickness provides an estimate of the differences between the two topologies: Thick lines highlight statistically relevant branch length differences. The group including `1b07`, `1afa` etc. seem closer in F_2 than in F_1. Importantly, this method was able to correctly distinguish between two groups of samples despite the high amount of code metamorphism present.

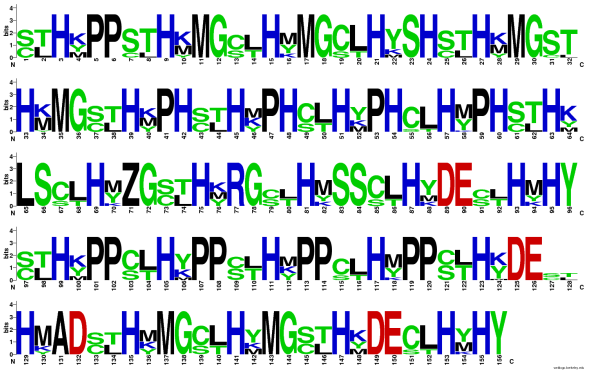


Figure 6. A sequence logo for the FakeAV-DO function "f_1". Positions with large characters indicate invariant parts of the function, while positions with small characters vary due to code metamorphism.

C. Sequence logos

A sequence logo [19] is a graphical method for identifying statistically significant patterns in a set of aligned sequences. The characters of the sequence are stacked on top of each other for each position in the sequence. The height of each letter is made proportional to its frequency, and letters are sorted so that the most common one is on top.

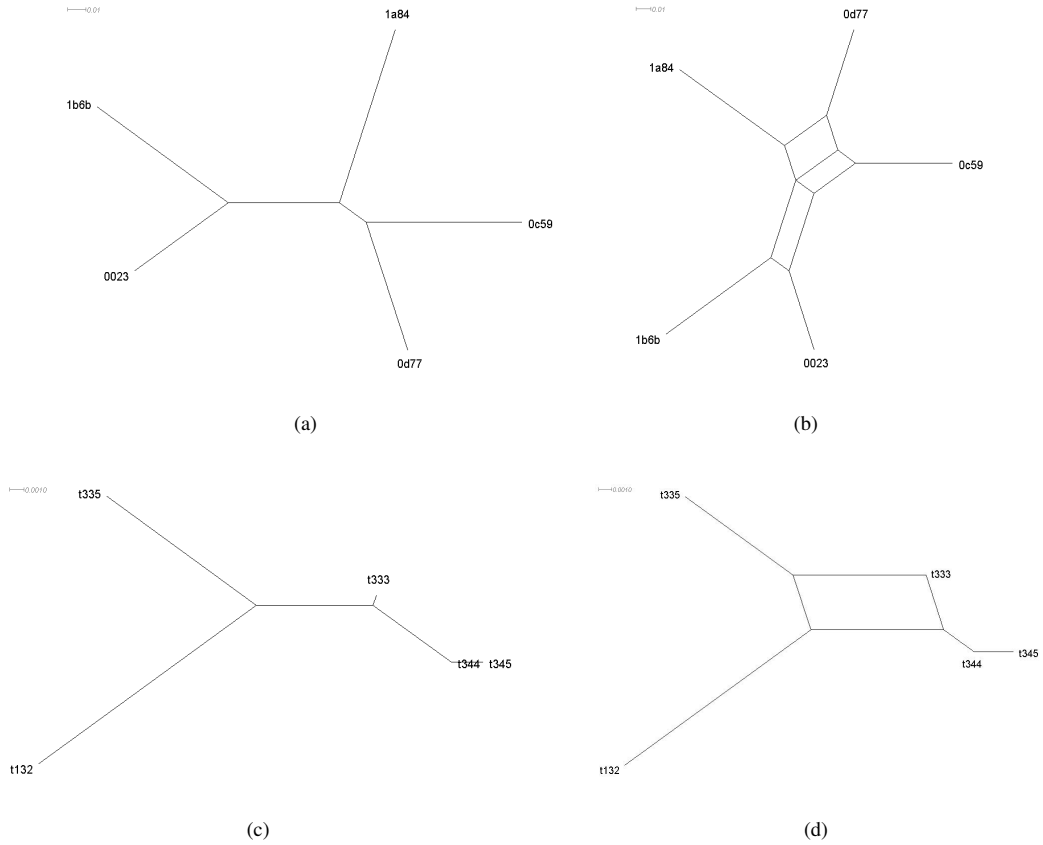


Figure 3. An unrooted phylogenetic (a) tree and (b) network of the FakeAV-DO function "f_1". An unrooted phylogenetic (c) tree and (d) network of different executions of the triangle program.

One use of a sequence logo is code normalisation. For example, the sequence logo constructed for f_1 (Figure 6) shows that positions 5, 6, 11, 12 and so on do not vary across all sequences, while there was a series of substitutions that were prominent, particularly between `ST` and `CL` at positions 1, 2, 7, 8 and so on. These sequences correspond to the instructions sequences `jmp`, `stc; jb` and `clc; jnb`. We observe that the `jb` conditional jump is always taken when preceded by `stc`, and similarly `jnb` always branches when preceded by `clc`. Therefore, code sequences `jmp`, `stc; jb` and `clc; jnb` are semantically equivalent and their presence is likely to be the result of a metamorphic engine. Code normalisation was performed by converting all `stc; jb` and `clc; jnb` sequences to simply `jmp` instructions since they are semantically equivalent. After normalisation, all 17 sequences were identical. This method of code normalisation reduced the number of instructions by about 62% in FakeAV-DO and by about 33% in Skyhoo.

D. Searching for motifs

A motif is a sequence pattern that occurs repeatedly in a group of related sequences. The purpose of a motif search is to identify consensus sequences which can be subsequently used for alignment. A common method is Gibbs sampling, which draws samples from a joint distribution based on the full conditional distributions of all possible sequences. The advantage of this approach is that sequences do not have to be aligned beforehand.

We used Gibbs sampling to identify motifs in samples from FakeAV-DO and "Skyhoo", a small but active botnet. The Skyhoo binaries were downloaded over the course of 5 months starting from November 2010. Figure 7 shows the motifs found in the API sequences of FakeAV-DO and Skyhoo samples. These motifs show the patterns formed by four main groups of APIs. The first group was concerned with memory allocation, for example `GlobalAlloc`, `GlobalFree`, `GlobalProtect`; the second group comprised I/O APIs such as `OutputDebugString`, `OpenSCManager`, `OpenService`, `CreateFile`; the third is made up of functions concerned with dynamic libraries such as `LoadLibrary`; the last group comprised threading API such as `GetProcessAddress`. The fact that these patterns are commonly found suggests that they

```
(a)
ILILILILHYHYQRTVHYTVNSILILILILNSILNSILNSILNSILNSILNSILNS

(b)
WCIGQPTMDMTMDMTMDMTMDMVFIWIWHQCAAI
IGIGITPEDKIMKIKEIKTVTVTVTVTVTVTVTVTV

AI kernel32.dll:CloseHandle
CA ntdll.dll:NtClose
DK kernel32.dll:GlobalLock
DM ADVAPI32.DLL:OpenServiceA
HQ kernel32.dll:DuplicateHandle
HY kernel32.dll:LocalAlloc
IG kernel32.dll:GetModuleHandleA
IK kernel32.dll:GlobalFree
IL kernel32.dll:GetProcAddress
IM kernel32.dll:GetModuleFileNameA
IT kernel32.dll:VirtualQuery
IW kernel32.dll:GetCurrentProcess
KE kernel32.dll:GlobalUnlock
KI kernel32.dll:GlobalHandle
NS kernel32.dll:LoadLibraryA
PE kernel32.dll:GlobalAlloc
QP ADVAPI32.DLL:OpenSCManagerA
QR kernel32.dll:LocalFree
TM kernel32.dll:CreateFileA
TV kernel32.dll:VirtualProtect
VF ADVAPI32.DLL:CloseServiceHandle
WC kernel32.dll:OutputDebugStringA
```

Figure 7. Common motifs among (a) FakeAV-DO and (b) Skyhoo API sequences.

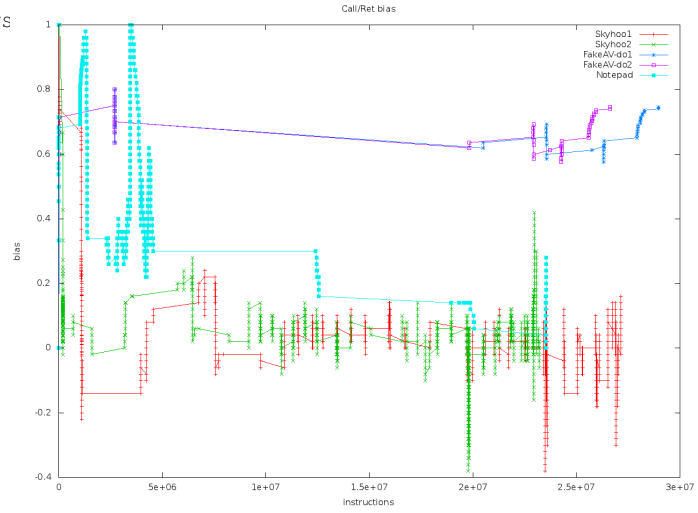


Figure 8. Call/Ret bias in FakeAV-DO, Skyhoo and notepad

are not significantly altered from generation to generation and are unique to the malware family.

E. Sequence base composition bias

Comparison of base composition, for example GC-content versus AT-content, in different species can be used to infer their phylogenetic relationships. The same can be true for symmetric or opposite instructions such as `call` and `ret`, or `push` and `pop`. Malware may not follow the standard call-ret procedure, but may instead use them as obfuscation routines. One example is the following code snippet found in Skyhoo.

```
call 0x46cae6
xchg dword ptr [esp], eax
pop eax
```

The stack pointer `esp` is not modified by the `call` because the `pop` restores it; `eax` gets "pushed" unto the stack by `xchg` but gets restored by the `pop`. These three instructions are semantically equivalent to

```
jmp 0x46cae6; nop; nop
```

Let q be the number of instruction Q in a window of n instructions and w the number of W, a symmetric or opposite instruction to Q. The bias can be calculated by the formula

$$bias = \frac{q - w}{q + w}$$

Figure 8 shows the bias in `call` and `ret` instructions for FakeAV-DO, Skyhoo and the Windows Notepad application for window size $n = 100$. The higher the bias, the higher the proportion of `call` to `ret` instructions. The bias is 1 when there are only `call` instructions and no `ret` instructions, and 0 when there is an equal number of each. The pattern of the bias is more distinctive in the two FakeAV-DO samples than for the Skyhoo samples. Nevertheless when compared to Notepad, we can see noticeable differences in the bias for the three groups of programs.

VI. Context-sensitive procedure analysis

The objective of this set of experiments is to investigate similarities between procedures used by FakeAV-DO and Skyhoo and those used in Notepad, 7zip, WinSCP and the triangle program. While the term "context" used in program analysis typically refers to a set of function pointers on the call stack, we use a different definition. In an execution trace of alternating kernel APIs and user procedures, we define the context of a user procedure to be APIs preceding or succeeding it. These procedures are expected to be similar within the same group of programs. Procedures preceding or succeeding the kernel32 GetProcAddress API were chosen as this API was the most commonly used. We tested 19 FakeAV-DO, 24 Skyhoo binaries, plus 7zip, WinSCP, Notepad and the 3 triangle programs. Code was normalised using methods described in Sections V-C and V-E. In total, experiments were performed using six datasets:

1) Sequences preceding the GetProcAddress API
2) Sequences preceding the GetProcAddress API, normalised
3) Sequences succeeding the GetProcAddress API
4) Sequences succeeding the GetProcAddress API, normalised
5) Sequences between two consecutive GetProcAddress APIs

Expt.	Num. of sequences			TP0	TP1	TP2	Total
	Benign	FakeAV	Skyhoo				
1	20	94	42	86.1%	99.3%	90.8%	95.0%
2	20	94	42	87.3%	79.0%	89.8%	83.1%
3	24	94	43	85.0%	100%	84.6%	93.8%
4	24	94	43	82.1%	100%	85.2%	93.8%
5	10	67	12	83.9%	100%	100%	98.3%
6	10	67	12	81.6%	100%	100%	97.9%

Table I
CONTEXT-SENSITIVE PROCEDURE DETECTION RATES. TP: TRUE POSITIVE RATE OF THE THREE CLASSES, BENIGN, FAKEAV-DO AND SKYHOO RESPECTIVELY.

6) Sequences between two consecutive GetProcAddress APIs, normalised

We extracted 156 sequences for experiments 1 and 2, 166 for experiments 3 and 4 and 89 for the last two. Half of the sequences were used for training and the remainder for testing. The sequences were aligned using the CLUSTAL algorithm using the identity substitution matrix for alignment with a gap penalty of 0.02 and the neighbour joining clustering algorithm. Test sequences are pairwise aligned with training sequences and classified based on the nearest neighbour classifier. Successful matches were matches where the correct class (benign, FakeAV-DO or Skyhoo) was assigned.

A. Results

Table I shows the results of the 6 experiments. Matches were highest when both contexts were taken into consideration (experiments 5 and 6). Code normalisation did not affect the results much, suggesting that context-sensitive analysis can cope with short `nop`-equivalent code metamorphism.

The high false positive rate for benign samples was because the number of available benign samples were small and, unlike the malware samples, belonged to a variety of programs and thus were under-represented. While this could be fixed with a more comprehensive evaluation, context-sensitive procedure analysis is perhaps more useful for exploratory analysis, providing clues to the malware's capabilities. For example, one mismatch was between a procedure in WinSCP and a Skyhoo one, implying that the Skyhoo procedure was more similar to WinSCP than it was to, say, 7zip. This may be seen as a false positive, but from the perspective of understanding Skyhoo better this is a useful data point.

VII. DISCUSSION

Our current study was limited to a small number of samples. In future, we intend to pursue this line of inquiry with a larger sample size, as well as to test the robustness of the alignment and clustering methods on other well-known metamorphic engines.

Secondly, we intend to choose a better mnemonic-alphabet mapping as the current simple scheme results in the same letter being used frequently, which affects alignment in many cases as it is based on single alphabets and not alphabet-pairs. Further work is also needed to consider how the program abstraction can be improved. One possibility is adding control and data flow information, but doing so efficiently. We also hope to investigate the use of substitution matrices other the identity matrix. As the matrix size is quite large (400 by 400), it is necessary to develop efficient techniques to generate them.

Another limitation is that the VMWare environment can be detected by malware through the use of "red pill" routines [17]. Red pills exploit imperfections in the virtual environment to detect it and are currently being used in the wild. A possible solution is to use hardware-assisted virtualisation as proposed by Dinaburg et al. [3].

Finally, Chronicler currently cannot handle multiple threads and only the parent thread was analysed. Support for multithreaded programs is left as future work.

VIII. RELATED WORK

Goldberg et al. [6] was perhaps the first to study malware phylogenies using suffix trees to construct "phyloDAGs", or directed acyclic graphs using occurance counts of 20-gram byte sequences. Erdélyi and Carrera [5] used phylogenetic trees to show the relationships between 6 distinct groups of malware using function call graphs. Karim et al. [8] used "n-perms" in addition to n-grams on opcode sequences to classify malware and to make comparisons with existing classification schemes. This comparison was visualised using phylogenetic trees. Ma et al. [13] studied the diversity of shellcode by looking at exedit distances of instruction byte sequences obtained from code emulation. Phylogenetic trees were constructed to show that shellcode was close-related to the type of vulnerability being exploited. Wehner [20] used phylogenetic trees to show how families of internet worms were related by looking at their normalized compression distance. Our contribution is in expanding the use of phylogenetic tools that, to our knowledge, have not been used in the context of malware analysis before.

Given the threat of malicious software, there has been significant effort invested in understanding malware functionality using static analysis, dynamic analysis or both ([22], [4], [16], [9]). Commonly used techniques are program slicing [21] and dynamic taint analysis [14]. Our objective is similar in spirit in that we also aim to extract functionality from malware, but taking a phylogenetic-inspired approach. We believe that phylogenetic methods can be applied to both static and dynamic analysis and complement existing methods for malware functionality extraction.

IX. CONCLUSIONS AND FUTURE WORK

We defined a framework for using phylogenetics methods to perform malware reverse engineering and detection. Although our results are still in preliminary stages, we have

shown that these techniques can be used to identify unique patterns in contemporary malware families. Moreover, we have shown that these patterns may be identifiable despite code metamorphism. In future, we intend to improve and adapt these techniques, as well as to conduct a more comprehensive study. Ultimately, malware does not exist in a vacuum and studying the unified diversity of malware has the potential to leverage on complexities of the evolving environment, something that malware writers cannot control or predict, to develop new analysis and detection techniques. Lastly, Chronicler is available at

`http://www.cl.cam.ac.uk/~wmk26/chronicler`

X. Acknowledgements

We would like to thank Sophos Plc. and Richard Clayton for kindly providing the malware samples, and also to our anonymous reviewers and our shepherd, Marco Cova, for providing extremely detailed constructive criticism.

References

[1] Bayer, U., Comparetti, P. M., Hlauschek, C., Krügel, C., and Kirda, E. Scalable, behavior-based malware clustering. In *Proceedings of the 16th annual network and distributed system security symposium (NDSS'09)* (2009).

[2] Dayhoff, M. O., and Schwartz, R. M. Chapter 22: A model of evolutionary change in proteins. In *in Atlas of Protein Sequence and Structure* (1978).

[3] Dinaburg, A., Royal, P., Sharif, M., and Lee, W. Ether: Malware analysis via hardware virtualization extensions. In *Proceedings of the 15th ACM conference on computer and communications security (CCS'08)* (2008).

[4] Egele, M., Kruegel, C., Kirda, E., Yin, H., and Song, D. Dynamic spyware analysis. In *2007 USENIX Annual Technical Conference* (2007).

[5] Erdélyi, G., and Carrera, E. Digital genome mapping: advanced binary malware analysis. In *Proceedings of the 15th Virus Bulletin International Conference* (2004), pp. 187–197.

[6] Goldberg, L., Goldberg, P., Phillips, C., and Sorkin, G. Constructing computer virus phylogenies. *J. Algorithms 26* (1998), 188–208.

[7] Higgins, D. G., Bleasby, A. J., and Fuchs, R. Clustal v: improved software for multiple sequence alignment. *Computer Applications in the Biosciences 8*, 2 (1992), 189–191.

[8] Karim, M. E., Walenstein, A., Lakhotia, A., and Parida, L. Malware phylogeny generation using permutations of code. *Journal in Computer Virology 1* (2005), 13–23.

[9] Kolbitsch, C., Holz, T., Kruegel, C., and Kirda, E. Inspector gadget: Automated extraction of proprietary gadgets from malware binaries. In *2010 IEEE Symposium on Security and Privacy* (2010).

[10] Lio, P., and Bishop, M. Modeling sequence evolution. *Methods in molecular biology (Clifton, N.J.)* (2008), 255–285.

[11] loup Gailly, J., and Adler, M. The zlib compression library. http://zlib.net.

[12] Luk, C., Cohn, R., Patil, R., Klauser, H., Lowney, A., Wallace, G., Reddi, S. V. J., and Hazelwood, K. Pin: Building customized program analysis tools with dynamic instrumentation. In *In Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation* (2005).

[13] Ma, J., Dunagan, J., Wang, H. J., Savage, S., and Voelker, G. Finding diversity in remote code injection exploits. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement (IMC'06)* (2006), pp. 53–64.

[14] Newsome, J., and Song, D. Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software. In *Proceedings of the 12th Network and Distributed System Security Symposium (NDSS'05)* (2005).

[15] Nye, T. M., Li, P., and Gilks, W. R. A novel algorithm and web-based tool for comparing two alternative phylogenetic trees. *Bioinformatics 22*, 1 (2005), 117–119.

[16] Porras, P., Saidi, H., and Yegneswaran, V. A foray into conficker's logic and rendezvous points. In *Proceedings of the 2nd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET'09)* (2009).

[17] Rutkowska, J. Red pill or how to detect VMM using (almost one) cpu instruction. http://invisiblethings.org/papers/redpill.html, 2004.

[18] Santamarta, R. Generic detection and classification of polymorphic malware using neural pattern recognition. http://www.reversemode.com, 2006.

[19] Schneider, T. D., and Stephens, R. M. Sequence logos: A new way to display consensus sequences. *Nucleic Acids Res. 18* (1990), 6097–6100.

[20] Wehner, S. Analyzing worms and network traffic using compression. *J. Comput. Secur. 15* (August 2007), 303–320.

[21] Weiser, M. *Program slices: formal, psychological, and practical investigations of an automatic program abstraction method*. PhD thesis, University of Michigan, Ann Arbor, 1979.

[22] Yin, H., Song, D., Egele, M., Kruegel, C., and Kirda, E. Panaroma: capturing system-wide information flow for malware detection and analysis. In *Proceedings of the 14th ACM conference on computer and communications security (CCS'07)* (2007).

Detecting Insufficient Access Control in Web Applications

George Noseevich, Andrew Petukhov
Computer Systems Lab, CS Dept.
Lomonosov Moscow State University
Email: {ngo, petand}@lvk.cs.msu.ru

***Abstract*—Web applications have become a de facto standard for delivering services on the internet. Often they contain sensitive data and provide functionality which should be protected from unauthorized access. Explicit access control policies can be leveraged for validating the access control, but, unfortunately, these policies are rarely defined in case of web applications. Previous research shows that access control flaws in web applications may be revealed with black-box analysis, but the existing "differential analysis" approach has certain limitations. We believe that taking the state of the web application into account could help to overcome the limitations of exiting approach. In this paper we propose a novel approach to black-box web application testing, which utilizes a use-case graph. The graph contains classes of actions within the web application and their dependencies. By traversing the graph and applying differential analysis at each step of the traversal, we were able to improve the accuracy of the method. The proposed method was implemented in the proof-of-concept tool AcCoRuTe. Evaluation with several real-world web applications demostrates better results in comparison to simple differential analysis.**

***Index Terms*—web applications; access control; vulnerability analysis; penetration testing;**

I. Introduction

Web applications are an increasingly popular way of providing services and data on the internet. Unfortunately, they are often implemented by developers who lack security skills, or security is neglected due to time or financial constraints. This has resulted in an increase of web application vulnerabilities that were reported recently.

According to the statistics, the amount of traditional input validation errors such as SQL injections and XSS is decreasing[1]. Besides, automated techniques for detecting such vulnerabilities have improved significantly[2].

Automated detection of certain other vulnerability types, however, is not so advanced. For example, access control flaws typicaly require manual detection[3].

Authorization protects data and operations from unauthorized access. Whenever a user can successfully perform actions that should have been prohibited for him (e.g., transfer money from the account that does not belong to him), or access data that he is not allowed to view, there is an access control flaw.

One of the possible approaches for broken access control detection is leveraging explicit access control policy [3], [4]. Unfortunately, such a document is rarely available for existing web applications.

In the absence of an explicit access control policy we need a way of extracting it from the web application behavior. We have chosen to regard a web application as a black box and to use the application's interface to derive access control rules. We define an access control flaw as a possibility for certain user to successfuly perform an HTTP request that could not have originated from his web interface.

Several authors have addressed access control flaws by applying black-box web application analysis and leveraging the web interface as an implicit access control policy. Differential analysis ([5], [6]) suffers from several limitations, which reduce its effectiveness in large web applications with complex logic. The main reason for this is its inability to take the web application state (the state is determined by web application database contents, filesystem, etc) into account.

In this paper, we extend the existing method by introducing a use case graph, which represents the web application logic and contains important information about actions that are possible within it. By traversing this graph in a specific order and applying modified differential analysis at each step, we were able to increase test coverage, which was confirmed by tests on a real-world application.

[1]The Web Application Security Consortium statistics for 2008 [1] states that the number of web sites containing vulnerabilities of these two widespread types has decreased by 13% and 20% respectively as compared to 2007.

[2]According to WASC statistics, 29% of XSS vulnerabilities and 46% of SQL injection vulnerabilities were detected by automatic scanning with default settings.

[3]According to the same statistics, only 14 (or 3%) of 615 insufficient authorization vulnerabilities were detected by automatic scanning. See also [2].

II. Understanding access control vulnerabilities

There are different types of authorization flaws. To be able to evaluate different techniques for their detection capabilities, we point out the following types, based on the root cause[4]:

A. Privileges under user control

When a web application makes assumptions about user privileges based on unvalidated input, it can be used by an attacker to escalate his privileges. The sample defect of this type is the ability to set `"editMode=true"` parameter in the request, resulting in the ability to edit any document.

Although in this case the vulnerable application typically contains many flawed pages, discovering any of them is sufficient to identify the root cause of the error. Thus, enumerating all vulnerable pages is not necessary.

B. Missing access control list entry

When application-wide access control is implemented using a blacklist approach, the missing blacklist entry usually results in an authorization vulnerability. An illustrative example is an Apache-based webserver that lacks .htaccess protection on some folders, leaving admininstrative functionality exposed to public viewers.

Flaws of this type are actually misconfigurations, which are introduced during the deployment phase. They are also identified by any instance, avoiding the need to enumerate all particular cases[5].

C. Insufficient access control on certain execution paths

When access control checks are distributed within the web application, some execution paths may not be covered by this checks. In this case, enumerating single instances makes sense, as different instances represent distinct execution paths.

III. Motivating Example

A. Sample web application

In order to illustrate the shortcomings of existing "differential analysis" approaches let us have a look at a typical Learning Management System (LMS). Table I describes user actions that are possible within the system ("login" and "logout" actions are omitted for brevity).

Let us suppose that this application contains the following authorization flaws:

1) Student can delete any unassigned course.
2) Manager can assign course to any student, not only to the student assigned to him.
3) The "view course" page is viewable by anyone.

Furthermore, we assume that the system is in the following state:

- There are four registered users, `root` of the role `Admin`, `instructor` of the role `Manager` and $\texttt{student}_{1,2}$ of the role `Student`;
- $\texttt{student}_1$ is assigned to `instructor`;
- there is one course uploaded; this course is assigned to $\texttt{student}_1$.

B. Existing approach

Existing methods for black-box access control testing are variations of the technique called *differential analysis*. This method suggests that, in order to discover authorization flaws, web application should be browsed on behalf of each user (including the "public" user, which means browsing without credentials) and faced URLs should be recorded (we will refer to this as the *crawling phase*). In our case this will result in five sitemaps: $\mathrm{URL}_{\texttt{root}}, \mathrm{URL}_{\texttt{instructor}}, \mathrm{URL}_{\texttt{student}_1}, \mathrm{URL}_{\texttt{student}_2}$ and $\mathrm{URL}_{\texttt{public}}$.

The next step is to try actions that were accessible to one user on behalf of the another. Considering our sample LMS, we should try to access ("$\setminus$" denotes set subtraction):

1) links from $\mathrm{URL}_{\texttt{root}} \setminus \mathrm{URL}_{\texttt{instructor}}$ on behalf of `instructor`;
2) links from $(\mathrm{URL}_{\texttt{root}} \setminus \mathrm{URL}_{\texttt{student}_1}) \cup (\mathrm{URL}_{\texttt{instructor}} \setminus \mathrm{URL}_{\texttt{student}_1})$ on behalf of $\texttt{student}_1$;
3) links from $(\mathrm{URL}_{\texttt{root}} \setminus \mathrm{URL}_{\texttt{public}}) \cup (\mathrm{URL}_{\texttt{instructor}} \setminus \mathrm{URL}_{\texttt{public}}) \cup (\mathrm{URL}_{\texttt{student}_1} \setminus \mathrm{URL}_{\texttt{public}})$ without credentials;
4) links from $\mathrm{URL}_{\texttt{student}_2} \setminus \mathrm{URL}_{\texttt{student}_1}$ on behalf of $\texttt{student}_1$;
5) links from $\mathrm{URL}_{\texttt{student}_1} \setminus \mathrm{URL}_{\texttt{student}_2}$ on behalf of $\texttt{student}_2$.

During steps 1–3, vertical privilege escalation will be detected, whereas steps 4 and 5 test for horizontal escalation.

C. Discussion of the existing approach

While the described method does very well in simple web applications, in complex systems with lots of interdependent actions it typically fails to detect a large amount of vulnerabilities for two major reasons.

[4]We do not claim to have built an exhaustive taxonomy of authorization flaws; these vulnerability types are informal and their mutual exclusion is in question. However, this grouping is useful to outline the capabilities of access control testing methods.

[5]The task of identifying all misconfiguration cases may be challenging as well, but it is more reasonable to review the configuration, once the black-box analysis discovers the root cause

User role	Available actions
Admin	Create manager, create student, delete student, delete user, assign a student to manager, revoke a student from manager.
Manager	Upload a course, assign a course to student, revoke a course from student, delete course;
Student	browse a course.

TABLE I
AVAILABLE ACTIONS FOR A SAMPLE LMS WEB APPLICATION

1) Incomplete web interface coverage: This problem arises because the described method does not specify the exact order in which these URL collections should be gathered. In our case, if we collect URLs for `root` first, we may occasionaly trigger the "revoke student from manager" action, which will prevent finding the "assign course to student" link in `instructor`'s web interface. Also, the web application state[6] in which we should start gathering the URL collections is not specified. In our example, while enumerating resources and actions visible to `instructor`, the crawler may overlook the "assign course to student" action, because $\texttt{student}_1$ is already assigned the only course that is uploaded at the moment.

2) Incorrect testing conditions: Even if the URL is successfully collected, failure to get unauthorized access to it does not necessarily mean it is not flawed. Suppose that we encountered the "assign course to student" link, but later, during browsing, triggered the "delete course" action. When we later try to assign the course on behalf of the student, it may fail not because the access is restricted, but simply because it is not possible to assign a course which is deleted.

Thus, applying differential analysis to our sample LMS will likely fail to detect vulnerabilities 1 and 2, identifying only the publicly-visible "view course" page.

In contrast to well-known difficulties which are encountered during automated or semi-automated black-box web application scanning (form filling, triggering javascript events, etc.), these are not technical restrictions, but rather limitations of the approach. While it is possible to get over the first limitation by making use of human-directed crawling, manually checking preconditions of each action during the testing phase dramatically increases complexity of the access control audit.

Thus, although typically well suited to discover vulnerabilities of types "A" and, in simplest cases, "B", this method is almost unusable for thorough access control testing in complex systems[7].

[6]Web application *state* is determined by its runtime environment, including database and filesystem contents, RAM, etc.

[7]This is confirmed by the aforementioned WASC statistics [1]: only 3% of applications containing access control flaws were identified during automated scanning

IV. PROPOSED APPROACH

The basic idea of our approach is to perform state-preserving differential analysis in a series of web application states.

As iterating through all possible states is not an option, we need to carefully select the states in which to perform testing, providing fair test coverage and keeping a reasonable degree of complexity at the same time.

To do this, we introduce the use case graph, which is traversed in a special way. After each step of the traverse the web application's state changes and modified differential analysis is applied. During the differential analysis we build *sitemaps* for each user and perform accessibility tests.

A. Selecting appropriate states and ensuring completeness

To get sensible results, we do not have the option of testing the possibly infinite number of valid HTTP requests. Instead, we divide all possible HTTP requests into classes, which aggregate requests with the same semantics. These classes will be called **actions**. For example, all HTTP requests that perform site search will belong to the "search" action whereas all requests that result in forum message creation will form the "create message" class.

Furthermore, we will utilize user roles instead of separate users. Indeed, the number of web application users is arbitrary, new users may be created on the fly and different users of the same role typically have similar privileges so testing all of them is not feasible. The variety of user roles, on the contrary, is typically determined during the web application design. We also introduce partial order on the set of roles in order to avoid testing the access to actions of lower-privileged users on behalf of the user with higher privileges.

The concepts of action and role may be combined into the notion of **use case**, which stands for *an action performed by a user of a given role*.

We can now switch from trying to cover all possible HTTP requests, which is unfeasible, to testing, at least once, all use cases of the web application. Given that access control is performed based on the use cases, as opposed to treating each request in its own way, we

#	Use Case Name	Depends On	Cancels
Public			
1	Browse public resources	-	-
Admin			
2	Login	1	-
3	Create Manager	2	-
4	Create Student	2	-
5	Assign Student to Manager	3, 4	-
6	Revoke Student from Manager	5	12, 13
7	Delete Manager	3	7, 10 - 15
8	Delete Student	4	8, 12, 13, 16 - 18
9	Logout	2	9, 3 - 8
Manager			
10	Login	3	-
11	Upload Course	10	-
12	Assign Course to Student	11, 5	-
13	Revoke Course from Student	12	12, 17
14	Delete Course	11, 13	12, 14
15	Logout	10	11 - 15
Student			
16	Login	4	-
17	Browse Course	12, 16	-
18	Logout	16	17

TABLE II
SAMPLE LMS: USE CASE DEPENDENCIES AND CANCELLATIONS

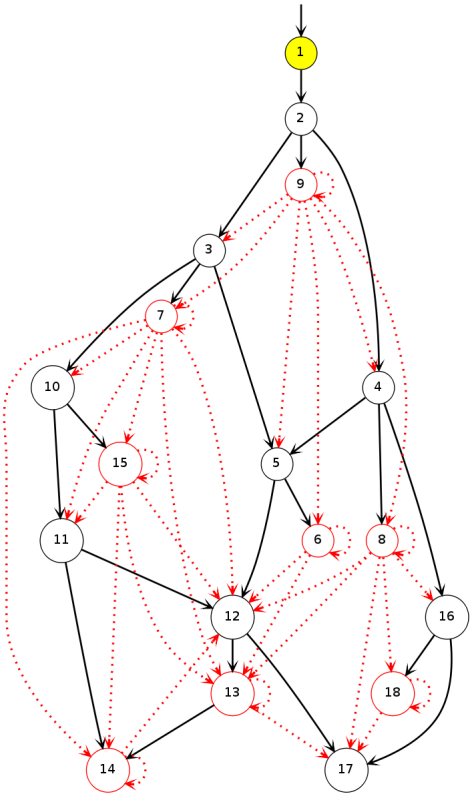


Fig. 1. Use case graph for sample application. Solid edges represent dependencies, dotted edges represent cancellations.

will discover all possible authorization flaws in this manner.

B. Use case dependencies and cancellations

In a typical web application with nontrivial logic, actions within the application are interdependent. We therefore introduce use case **dependencies**. Use case A is considered dependent on use case B when we must perform B in order to be able to perform A.

We also introduce use case **cancellations**. Use case A is said to cancel use case B, when, after performing A in the state where it is possible to perform B, we may lose this opportunity.

Table II enumerates use case dependencies and cancellations for the sample web application introduced above.

We are now able to build a use case graph, including dependencies and cancellations. The graph for the sample application is shown in Figure 1.

C. Traversing the use case graph

To build the desired sequence of states, we traverse the use case graph in the following way:

1) At the beginning of the traversal, use cases with no prerequisites are available for execution.
2) Once a use case is executed, more use cases may become available if their requirements are satisfied.
3) At each step, we execute an available, non-visited use case that cancels least amount of other use cases. Among these, we pick such a use case that satisfies the maximum number of unsatisfied dependencies. The executed use case is marked as visited.
4) We keep executing use cases until all of them are visited.

D. Complete testing algorithm

We propose the following procedure for testing the application's access control:

1: Enumerate use cases and roles in the application.
2: Specify use case dependencies and cancellations and build the use case graph.
3: Create two groups of users, each containing a user for every role.
4: Traverse the graph as specified in IV-C to get the use case execution list UCL.
5: **for all** $u = (action_i, role_i) \in UCL$ **do**
6: let $user_{1,i}$ be the user of the role $role_i$ from the first group;
7: execute $action_i$ on behalf of $user_{1,i}$;
8: perform crawling, enumerating accessible resources and operations for each user. While do-

ing this, execute only state-preserving actions.
9: **for all** $role_1, role_2$ **do**
10: **if** $role_1$ is not less privileged than $role_2$ **then**
11: let $user_1$ be the user of the role $role_1$ from the **first** group;
12: let $user_2$ be the user of the role $role_2$ from the **second** group;
13: let $sitemap_1, sitemap_2$ be the URL collections for $user_1$ and $user_2$, gathered during crawling;
14: try to perform actions from $sitemap_1 \setminus sitemap_2$ on behalf of $user_2$;
15: record any successfully performed action as a authorization flaw;
16: **end if**
17: **end for**
18: **end for**

V. Implementation details and discussion

We implemented the proposed approach in a proof-of-concept tool called AcCoRuTe (Access Control Rule Tester). The workflow is divided into two stages: information gathering and scanning.

Information gathering

During the first step, a self-developed Firefox extension assists the operator in building the use case graph. Actions are recorded as the operator performs them using the browser. The operator then uses them to build the use case graph. He also provides additional information that will be passed to the scanner (e.g. list of web application roles and users, the crawling scope, etc).

The process of use case graph creation cannot be performed in a fully automated way for a black-box scenario. We can nevertheless significantly reduce the complexity of this task. First of all, most of the state-preserving actions may be excluded from the graph. Indeed, these actions cannot be dependencies or cancel other actions. Furthermore, if the state-preserving action has only one dependency, we can safely assume that this dependency will be met at least once during the use case graph traversal.

We should also note that providing an incomplete use case graph may reduce test coverage (resulting in decreased numbers of discovered and tested actions), but will not influence the correctness of the analysis (i.e. the discovered actions will still be tested in proper states). Thus, when creating the use case graph of a very large web application, the operator may choose to enumerate use cases at the coarse-grained level (e.g. only enumerate dependencies and cancellations for the most important actions), possibly reducing the coverage. However, in order to correctly distinguish between state-changing and state-preserving actions during the scan, the operator should record a representative for each state-changing action.

Scanning

Gathered information is passed to the second part of the tool, the scanner. The scanner is a standalone Java application based on HtmlUnit [7].

According to the presented approach (see section IV), the scanner navigates the web application as follows:

1) recursively crawls the web application, building "sitemaps" for each user (only state-preserving actions are executed);
2) performs accessibility tests;
3) changes web application state by issuing one of the state-changing requests according to the use case graph or returns, if the traversal is complete;
4) proceeds with step 1 in the changed state.

Combining requests into actions: During the second and the third step the scanner needs a way to detect similar HTTP requests and combine them into actions.

In order to test requests for similarity, we represent each of them as a list of parameters. Every parameter is then assigned a meaning which is inferred from the web application interface: for example, user-modifiable form fields map into user-controllable parameters, whereas hidden fields are considered "automatic" (i.e. out of user control). The current implementation also allows the operator to manually map parameter name/location pairs to their meanings in order to identify session/CSRF tokens and volatile parameters (e.g. document identifiers)[8]. When testing for similarity, the scanner ignores user-controllable parameters and session identifiers because they do not contribute to the meaning of the request.

Preventing uncontrolled state changes: As we already mentioned, only state-preserving requests should be issued during the crawling step. To recognize state-changing requests the scanner tests them for similarity with actions recorded by the operator.

During the testing step the state of the web application may also change, which means that a state-changing action is vulnerable. In this case, the scanning process cannot be continued, because the state of the application after successful exploitation is unpredictable. Therefore, if the scanner detects vulnerable state-changing action, it stops execution. This action is then added to scanner "suppression" list (i.e. list

[8]It is possible to automate this process (e.g. by repeatedly fetching the index page of the application to discover session token name and location).

of actions that will not be tested), the application is reset to its initial state by the operator and the scan is launched again.

Determining successfully performed actions: During accessibility testing, we need a way to understand whether a request was successful or not. The usual approach here is to perform pattern matching in order to identify "access denied"-like HTTP responses based on operator input, which is prone to both false positives and false negatives. Instead, we compare received response with the valid response to the similar request that was collected during previous steps and use pattern matching as a fallback, if this information is not available. To test responses for similarity we compute normalized edit distance [8] and introduce a configurable similarity threshold. Although this approach does well in typical scenarios, it may fail to detect performed actions in some cases (e.g. when the web application responds as if the access was denied, while nevertheless performing the action).

VI. Evaluation

We evaluated our tool and our approach on a real-world web application (*Easy JSP Forum*) obtained from SourceForge repository, which is a typical message board written in JSP. This application was previously analyzed by Felmetsger et al [9], who discovered two access control vulnerabilities, namely that any authenticated user is able to delete and modify any post in the forum.

Using our tool we were able to find one known vulnerability[9] and three previously unknown vulnerabilities in this application: **any moderator can delete or modify any forum as well as assign it to any user**.

The application contains a management interface, which can be accessed by forum administrators and moderators. The interface allows assigning a forum to a moderator (corresponding form is shown only to administrative users) and contains links to the "forum details" page where a forum can be edited or deleted. Moderators are only allowed to edit or delete forums that belong to them, and for them only allowed "edit forum" links are shown. However, when a delete/modify request is issued, the forum application fails to check whether the moderator owns the forum he is about to modify or delete. The individual "edit forum" pages also lack this check.

Furthermore, the "assign forum to moderator" action, which should be available only to forum administrators is in fact accessible by moderators as well. The corresponding execution path fails to ensure that the session belongs to the administrator (although it checks whether the user has at least moderator privileges, therefore regular users cannot exploit this vulnerability)

Our tool reported this as four vulnerable actions: three state-changing actions (forum deletion, modification and assignment) and one state-preserving (the "edit forum" page itself).

Our tool also incorrectly reported the "view profile" action as vulnerable, because the "view profile" link for a user is shown to other users only if that user posted at least one message to the forum, whereas it is always shown to the user himself. This is the only false positive reported.

Also, AcCoRuTe failed to detect one of the previously-known vulnerabilities which allows post modification. That was due to limitations of its crawling engine (javascript-generated popups are not always correctly processed), and it will be fixed soon.

Some details about performed scan iterations are presented in Table III. All tests were performed on a PC with an Intel Xeon CPU (2.33 GHz) with 16 GB of RAM. The *Pages Fetched* and *Pages Parsed* columns represent the number of fetched URLs (including images, scripts and style sheets) and the number of actually parsed pages, respectively. The *Forms Filled* column denotes the number of forms which were automatically filled in during the crawl (note that each form was filled several times by different users in different states). The *Checks Performed* column shows the number of the accessibility tests carried out, the *Raw Alerts* column shows the number of successful tests, whereas *Unique Alerts* shows the number of distinct vulnerable actions as reported by the tool.

The amount of human work required to build the use case graph cannot be measured objectively. During our experiments it took about 30 minutes to record the needed use cases and specify their dependencies and cancellations for an operator familiar with forum functionality. The recorded use case graph for *Easy JSP Forum* contained 11 nodes and 22 edges (8 dependencies and 14 cancellations).

VII. Limitations

Both, presented approach and its implementation have certain limitations which are briefly enumerated in the current section.

A. Limitations of the approach

1) Vulnerability types: The suggested approach ensures that every use case will be discovered and tested at some stage of graph traversal. As mentioned above, this is enough to find all authorization defects only if we assume that the access control flaws that we are

[9]The tool discovered the vulnerable "delete post" action but failed to discover the "edit post" vulnerability.

Iteration[a]	Pages Fetched	Pages Parsed	Forms Filled	Checks Performed	Raw Alerts	Unique Alerts[b]	Vulnera-bilities	False Positives	Runtime (min:sec)
First	499	127	175	302	22	5	4	1	3:50
Second	944	254	315	512	25	1 (2)	1 (1)	0 (1)	7:50
Third	2566	709	833	1323	61	0 (2)	0 (1)	0 (1)	15:52
Total						6	5	1	26:52

TABLE III
EXPERIMENTS ON *Easy JSP Forum*

[a]Three scanner runs were required to completely test this web application: after discovering a state-changing vulnerable action the scanner stops, discovered vulnerabilities are added to the suppression list, web application is reset to its initial state and the scanner is launched again (possible ways to recover after discovering state-changing vulnerabilities is discussed in section V).

[b]For the second and third runs of the scanner the number of alerts that duplicate alerts reported in previous runs is shown in brackets

dealing with are of a certain type. More specifically, we assume that, if

- unauthorized access to a given use case $uc = (action, role_{victim})$ by a user of a given role $role_{attacker}$ is possible
- AND the web application in its current state allows one of the users of $role_{victim}$ to perform the $action$,

we will be able to reproduce unauthorized access in the current state by executing $action$ on behalf of any user of $role_{attacker}$.

If, for example, the vulnerability appears only when deleting the 20th blog post, or only if three or more customers are concurrently shopping, the presented approach will probably fail to detect them.

2) Hidden functionality: Another limitation arises from the test case selection: the proposed approach will not discover actions and data that are not explicitly linked from the user's web interface (e.g. database backups stored online). There are various solutions for this problem, including fuzzing (stateful or stateless), which can be combined with our approach to improve its completeness. This is, however, beyond the scope of this paper.

B. Limitations of the implementation

The current implementation has several technical limitations which are typical for automated black-box crawlers. Although the tool provides reasonable javascript support, it may still fail to detect some links, especially when non-trivial user interactions are needed to trigger respective requests (e.g. multiple mouse movements and clicks).

The accessibility test may also fail to detect that the tested action was actually carried out (e.g. when the "access denied" message was shown, but the action was nevertheless performed).

The implementation currently lacks AJAX support, which is left for future work.

The amount of manual operator work is still high especially considering very large web applications.

Finally, the performance of the scanner should be improved in order to analyse large-scale web applications.

VIII. RELATED WORK

First, our work is related to the research on inferring program specifications for vulnerability detection. One of the first techniques that use inferred specifications in order to find application-specific errors was developed in the work by Engler et al. [10]. Using a number of pre-defined templates, the authors infer specifications in the form of programmer "beliefs". The code is then checked for contradictions with these beliefs and statistical analysis is used to distinguish between valid beliefs and coincidences. Another work by Tan et al. [11] focused on detecting missing security checks using specifications automatically extracted from the source code. Kremenek et al. [12] also leveraged static source code analysis to extract program specifications and find bugs.

Various authors also used dynamic analysis to derive program specifications, which are then either used for intrusion detection ([13], [14]), or statically checked (as in the research by Nimmer and Ernst [15]).

The Waler tool (Felmetsger et al., [9]) also explores the combination of dynamic and static analysis in order to detect vulnerabilities. They infer specifications by dynamically analyzing the web application's "normal operation", and then they use static analysis to find execution paths that violate the specifications. This approach is similar to ours in the sense that both works try to detect web application vulnerabilities by checking the inferred specifications. Our work, however regards the web application as a black box, inferring specifications from its interface and uses accessibility tests to check the inferred specifications. Our approach requires a significant amount of operator work in order to express the web application logic as a use case graph, whereas the combination of dynamic and static analysis probably offers a higher level of automation. However, the completeness of the analysis performed

by Waler seems to depend significantly on how the normal behavior is covered during the invariant extraction. This is illustrated by *Easy JSP Forum* web application, which contains at least three vulnerabilities that were not reported by Waler, possibly because of an incomplete set of inferred invariants (see Section VI).

Our research is also related to several works that use black-box analysis to detect authorization flaws in web applications. The proposed approaches include forceful browsing (i.e. manual testing), differential analysis and fuzzing.

Segal [6] describes the aforementioned differential analysis technique. The shortcomings of this method, which prevent it from discovering complex access control flaws in large applications, are described in Section III. Book [16] by Stuttard and Pinto contains a complete overview of black-box web application security analysis. In respect to attacking authorization, the book suggests a variation of differential analysis, which is afflicted by the above mentioned limitations. Book [5] also overviews black-box analysis techniques and offers differential analysis as an alternative to manual access control verification.

The task of discovering hidden (not linked explicitly from the web interface) data and functionality of the web application is also related to our work because access to hidden content is typically not restricted ("security by obscurity"). This problem is typically approached by fuzzing (i.e. educated and optimized guesswork). For example, OWASP DirBuster tool [17] uses dictionaries to guess locations of hidden files and directories. Discovering hidden content, though, is out of scope of our approach.

Our work is also related to detection of execution after redirect (EAR), a web application vulnerability researched by Doupe and Boe [18], that may lead to broken access controls. In case of unauthorized access, the vulnerable application generates redirect but fails to stop further request processing. Assuming that the request processing generates output, which is sent back in the body of the redirect, and assuming that this output is similar to that of the authorized user, our approach remains applicable for this case.

IX. Conclusion

In this paper, we introduced a method to detect access control flaws in web applications by black-box analysis. We extend the widely-used "differential analysis" to take web application state into account. In our approach, we construct the web application use case graph with assistance of a human operator, traverse it in a specific order to get the sequence of use cases, iterate through this sequence and apply differential analysis at each step of the traverse.

We implemented the proposed approach in a tool that we named AcCoRuTe, and it was able to identify previously unknown access control vulnerabilities in a real-world web application.

X. Acknowledgments

We want to thank Christian Platzer, Dennis Gamajunov and a number of anonymous reviewers who gave us very useful feedback on a previous version of this paper.

References

[1] W. A. S. Consortium, "Web application security statistics 2008," http://projects.webappsec.org/f/WASS-SS-2008.pdf, Web Application Security Consortium.

[2] J. Grossman, "WhiteHat Website Security Statistics Report," http://www.whitehatsec.com/home/assets/WPStatsreport_100107.pdf, 2007.

[3] A. Masood, "Scalable and effective test generation for access control systems," 2006.

[4] E. Martin, "Automated test generation for access control policies," in *Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications*. ACM, 2006, pp. 752–753.

[5] J. Scambray and M. Shema, *Hacking exposed: Web applications*. McGraw-Hill Osborne Media, 2002.

[6] O. Segal, "Automated testing of privilege escalation in web applications," *Watchfire*, 2006.

[7] "HtmlUnit," http://htmlunit.sourceforge.net.

[8] A. Weigel and F. Fein, "Normalizing the weighted edit distance," in *Pattern Recognition, 1994. Vol. 2-Conference B: Computer Vision & Image Processing., Proceedings of the 12th IAPR International. Conference on*, vol. 2. IEEE, 1994, pp. 399–402.

[9] V. Felmetsger, L. Cavedon, C. Kruegel, and G. Vigna, "Toward automated detection of logic vulnerabilities in web applications," in *USENIX Security*, 2010.

[10] D. Engler, D. Chen, S. Hallem, A. Chou, and B. Chelf, *Bugs as deviant behavior: A general approach to inferring errors in systems code*. ACM, 2001, vol. 35, no. 5.

[11] L. Tan, X. Zhang, X. Ma, W. Xiong, and Y. Zhou, "Autoises: Automatically inferring security specifications and detecting violations."

[12] T. Kremenek, P. Twohey, G. Back, A. Ng, and D. Engler, "From uncertainty to belief: Inferring the specification within," in *Proceedings of the 7th symposium on Operating systems design and implementation*. USENIX Association, 2006, pp. 161–176.

[13] M. Bond, V. Srivastava, K. McKinley, and V. Shmatikov, "Efficient, context-sensitive detection of semantic attacks," Citeseer, Tech. Rep., 2009.

[14] A. Baliga, V. Ganapathy, and L. Iftode, "Automatic inference and enforcement of kernel data structure invariants," in *2008 Annual Computer Security Applications Conference*. IEEE, 2008, pp. 77–86.

[15] J. Nimmer and M. Ernst, "Static verification of dynamically detected program invariants:: Integrating daikon and esc/java," *Electronic Notes in Theoretical Computer Science*, vol. 55, no. 2, pp. 255–276, 2001.

[16] D. Stuttard and M. Pinto, *The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws*. Wiley Publishsing, 2007.

[17] "OWASP DirBuster," https://www.owasp.org/index.php/Category:OWASP_DirBuster_Project.

[18] A. Doupe, "Overview of execution after redirect web application vulnerabilities," http://adamdoupe.com/overview-of-execution-after-redirect-web-appl.

I/O Attacks in Intel PC-based Architectures and Countermeasures

Fernand Lone Sang*†, Vincent Nicomette*† and Yves Deswarte*†
*CNRS; LAAS; 7 Avenue du colonel Roche, F-31077 Toulouse Cedex 4, France
†Université de Toulouse; UPS, INSA, INP, ISAE; UT1, UTM, LAAS; F-31077 Toulouse Cedex 4, France

***Abstract*—For a few years now, attacks involving I/O controllers have been subject to a growing interest. Unlocking smartphones and game consoles through USB connections, or bypassing authentication through FireWire are examples of such attacks. Our study focuses on I/O-based attacks targeting Intel PC-based information systems such as laptop or desktop computers. This paper provides a survey of such attacks and proposes a characterization and a classification of these attacks. Then, an overview of various techniques which mitigate the risks related to I/O attacks are described and their respective limitations are discussed. Finally, several I/O attacks we are currently investigating are presented.**

***Index Terms*—Intel PC architecture, I/O controller, peripheral device, peer-to-peer, Direct Memory Access (DMA), PCI Express, operating system security**

I. Introduction

Nowadays, it is difficult to protect information systems in an efficient way. Because of their increasing complexity, the attack surface on those systems keeps on expanding and the successful attack rate increases, in spite of the numerous protection mechanisms implemented in those systems.

In an information system, a security loss can result from different kinds of malicious logic actions [1], that can be classified as follows:

- *Class 1* corresponds to corrupting the main memory, where the running software components' state (*i.e.*, data) and behavior (*i.e.*, code) are temporarily stored. Amongst these software components, the operating system kernel is a particularly attractive target for an attacker, since corrupting the kernel means subverting potentially all software components that run upon it.
- *Class 2* corresponds to corrupting the hardware components' state which software components depend on, namely registers of the CPU or of the chipset. These registers represent the execution environment memory.
- *Class 3* corresponds to corrupting the other hardware components, such as I/O controllers interconnected through the chipset.

These corruptions can be carried out either from the CPU, by exploiting software flaws (*e.g.*, buffer overflow, integer overflow, null pointer dereference), or they can be carried out from an I/O controller by misusing I/O features such as Direct Memory Access (DMA). DMA enables I/O controllers to transfer data directly (*i.e.*, without going through the CPU) to or from the main memory, using a dedicated DMA engine. This mechanism allows the processor to be relieved of I/O transfers, it is only involved in initiating and terminating them.

Malicious actions carried out from the CPU are well known nowadays. Protection mechanisms against such attacks [2] are mature and largely implemented in current operating systems. Conversely, few mechanisms exist to mitigate attacks originating from I/O controllers. This paper proposes a survey of such attacks in Intel PC-based platforms and discusses several solutions that can be used to counter them. First, we recall in Section II some technical background required to make this paper self-contained. Then, Section III intends to characterize and classify I/O attacks, and techniques to block such attacks are discussed, with their respective limitations. In Section IV, we describe several unexplored I/O attacks we are currently investigating. Finally, Section V concludes this article.

II. Technical background

For almost two decades now, Intel has a dominating position on the market of processors and chipsets. As a consequence, Intel PC architecture is nowadays the most widespread architecture in information systems. This technical background section recalls some specificities of this architecture.

A. Overview of the Intel PC architecture

Figure 1 depicts a simplified view of a typical Intel PC-based system[1]. It is composed by two main components: (1) the Central Processing Unit (CPU), upon which software components (applications, operating system, *etc.*) are running; and (2) the chipset, which interconnects the CPU, the main memory and the I/O controllers, such as peripheral device controllers, bus and network controllers (USB controller, Ethernet controllers, *etc.*) or I/O bus bridges (PCI Express bridge, PCI Express-to-PCI bridge, *etc.*).

Most of the Intel chipsets are composed of two separate chips, the northbridge and the southbridge, interconnected by a proprietary bus referred to as Direct Media Interface (DMI).

The northbridge[2] is in charge of interconnecting the CPU and the main memory. I/O controllers that require high bandwidth, such as graphic controllers or gigabit Ethernet controllers for instance, can sometimes be directly attached

[1]Note that Figure 1 depicts a simplified view of the former Intel PC architecture. Even though recent architectures, such as Intel Nehalem, are a bit different, the concepts discussed in this section remain applicable.

[2]The northbridge is sometimes referred to as the Memory Controller Hub (MCH) in former Intel PC architectures, or the I/O Hub (IOH) on some recent Intel PC architectures such as Intel Nehalem.

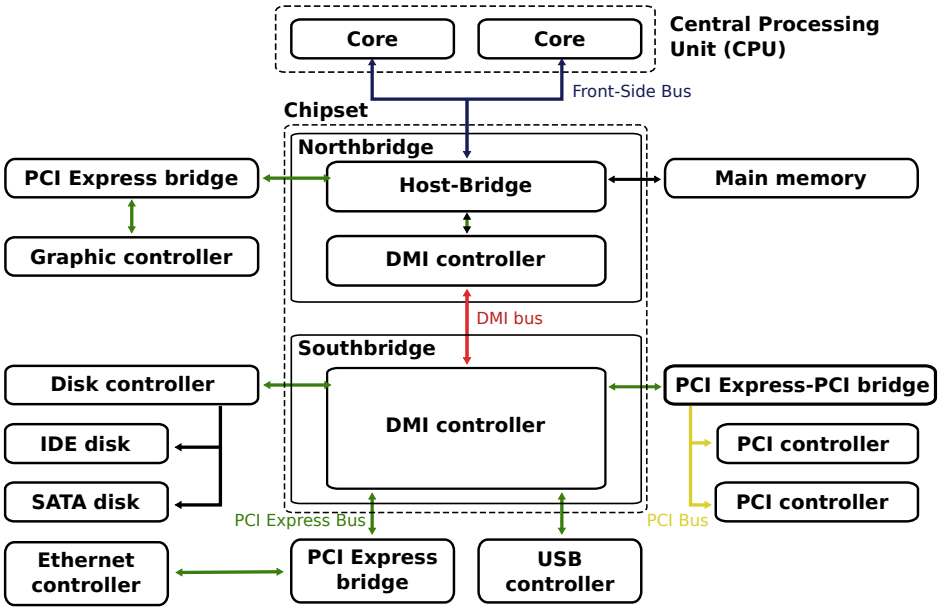


Figure 1. Example of a typical Intel PC architecture (simplified view)

to the northbridge through a PCI Express bridge. In the same way, the southbridge[3] is responsible for interconnecting the remaining I/O controllers (attached to PCI Express buses) to the northbridge.

B. I/O address spaces and related access mechanisms

This subsection outlines the different I/O address spaces defined in the Intel PC architecture, and describes the mechanisms that enable access to these spaces from the CPU and from the I/O controllers.

1) Programmed I/O space: in the Intel PC architecture, the legacy mechanism to communicate with I/O controllers from the CPU consists in executing special I/O instructions, namely `in` and `out` assembler instructions. These instructions enable the CPU to read from (respectively, write to) registers mapped into the Programmed I/O space (PIO), a 16-bit wide address space distinct, at least logically, from the memory space. At the I/O bus level (*cf.* Figure 1), while executing these instructions, the CPU activates dedicated lines on the Front-Side Bus (FSB). The host-bridge converts those signals into PCI Express I/O requests which are routed finally by the chipset to the concerned I/O controller. Let us note that many controllers, such as SATA controllers or keyboard controllers for instance, still map some of their registers to this address space for backward compatibility reasons.

2) Memory space: another way to communicate with I/O controllers is through the memory space, which maps either the main memory, or some I/O locations (*i.e.*, memory or registers physically located into an I/O controller) referred to as memory-mapped I/O (MMIO). Access to this space from the CPU are performed in the same way as regular memory accesses and no special I/O instructions are required. At the I/O bus level, the host-bridge either directly forwards the request to the main memory controller when the access targets the main memory, or else translates it into a PCI Express memory request, which is further routed by the chipset to the concerned I/O controller. Input and output to this space are also possible from the I/O controllers themselves through Direct Memory Access (DMA). Concretely, to carry out a Direct Memory Access, the I/O controller sends a PCI Express memory request in the same way as the host-bridge does when a memory access is performed from the CPU.

3) Configuration space: in the Intel PC architecture, each PCI [3], PCI-X and PCI Express [4] controller has a configuration space in addition to the MMIO and PIO spaces. This space contains several registers that enable the operating system to identify and configure an I/O controller bus interface. Knowing the controller I/O bus address, access to this space from the CPU is either possible through an aperture in the PIO space or through dedicated regions of the MMIO space. Both access mechanisms make the host-bridge generate configuration requests on I/O buses. Note that the I/O bus address used to access this space is a unique triplet (Figure 2) composed of:

- ***a bus number***: it corresponds to a number assigned by the BIOS to the bus which the I/O controller is physically connected to.
- ***a device number***: for a given bus number, it indicates the physical slot on the motherboard which the I/O controller is connected to.
- ***a function number***: it identifies a subsystem in the I/O controller. For instance, on multiple ports USB card, this number refers to a port.

The reader is referred to [3], [4], [5] for a full description of the configuration space and the related access mechanisms.

[3]The southbridge is sometimes referred to as the I/O Controller Hub (ICH).

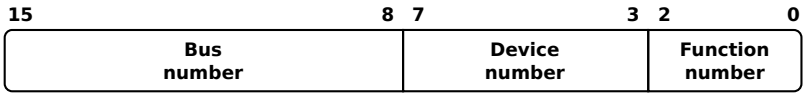


Figure 2. I/O bus address format to access the configuration space

III. Characterization of I/O attacks and countermeasures

Currently existing I/O attacks against Intel PC-based systems share a common feature: they all abuse the DMA capability of I/O controllers to access the memory space. First, we recall in this section different means that an attacker can use to set up a DMA attack. Then, we propose a characterization and classification of such attacks, illustrated by concrete attack examples. Finally, various mitigation techniques are detailed and we discuss some of their respective limitations.

A. Different means to perform DMA attacks

The first way to perform a DMA attack consists in abusing the DMA engine programming interface of an I/O controller in order to make this controller perform a malicious memory access. It is programmed to carry out a memory access either (1) with the CPU cooperation, by executing a piece of code that will set up several DMA related registers in the I/O controller; or (2) from the I/O controller itself, triggered by specific external stimulus. For instance, DMA engines in USB On-The-Go (OTG) [6] and FireWire [7] controllers can be driven by peripheral devices attached to them. Over the years, many examples of such feature abuse have been published [8], [9], [10], [11]. Some of them are detailed in the next subsection.

The second way to achieve this is by exploiting a flaw in an I/O controller (*e.g.*, a buffer overflow in its firmware), which enables the attacker to drive the DMA engine from the I/O controller itself. A concrete example of such a firmware exploitation has been published in [12].

Finally, an attacker can also manufacture his own malicious I/O controller using for instance programmable logic technologies such as FPGA. Thus, the attacker defines his own interface to program the I/O controller DMA engine. So far, several prototypes of such controllers have been published [13], [14].

B. Characterization and classification of DMA attacks

This subsection provides a survey of DMA attacks. We characterize such attacks and classify them in two categories: those targeting (1) the main memory and those aiming at (2) the I/O controllers memory-mapped I/O locations.

1) DMA attacks targeting the main memory: some I/O attacks abuse DMA to access the main memory. Such attacks aim at violating the confidentiality or integrity of software components (*i.e.*, their code and/or data) loaded in the main memory and fit, as a result, in malicious actions categorized in the *Class 1* (*cf.* Section I).

In some circumstances, it can be considered legitimate to use an I/O controller with DMA ability to read the main memory and perform live memory acquisition. For instance, B. Carrier and J. Grand developed Tribble [15], a dedicated PCI controller which uses DMA to capture forensic images of the main memory. A similar controller, Copilot [16], has been developed by Komoku for RAM acquisition and for online operating system kernel integrity monitoring.

In less legitimate cases, Direct Memory Access can be abused to bypass memory protection mechanisms implemented by the operating system. L. Duflot [17] showed for instance that a malicious code with reduced privileges can set up DMA transfers to escalate privileges. In his proof-of-concept, he programmed some malicious DMA transfers in a USB UHCI [18] controller in order modify several critical kernel data structures. As soon as a USB peripheral (*e.g.*, a USB flash drive) was plugged to the system, the programmed memory accesses were executed by the USB controller, allowing him to lower the `securelevel` of an OpenBSD [19] system. To perform such an attack, it is necessary for an attacker to (1) execute some piece of code in order to set up DMA transfers and to (2) have a physical access to the targeted system in order to trigger them. In some circumstances, a physical access to the system is enough to perform DMA attacks. M. Dornseif [9], [8] demonstrated that former FireWire drivers in Mac OS, BSD or Linux systems enable by default FireWire devices (*e.g.*, a modified iPod) to access the entire main memory through the FireWire controller which they are connected to. His work has been later extended by A. Boileau [11] who discovered a technique to trick FireWire device drivers in Windows and to carry out attacks through FireWire on these systems. His trick consisted in modifying the Configuration Status Register (CSR) of another FireWire device in order to usurp the identity of a device which is authorized to access the main memory. To illustrate his work, he performed a live demonstration of a Windows authentication mechanism bypass. Several proofs-of-concept followed then: D.R. Piegdon [20] presented a variety of techniques for compromising Linux systems while D. Aumaitre [10] focused on Windows systems. In this craze, D. Maynor [21] highlighted that USB On-The-Go (OTG) [6] controllers also present the same issues, namely DMA in these controllers can be abused by peripheral devices. Finally, Y.A. Perez *et al.* [12] proved recently that DMA can also be ordered remotely. By exploiting a vulnerability in the firmware of a network controller, they managed to modify its behavior and make it perform DMA on reception of specific frames. They used this facility to obtain a remote root-shell. G. Delugré [22] extended their work by developing tools to reverse engineer the firmware of the network controller and demonstrated that this firmware could be modified to integrate a rootkit hard to detect from the operating system.

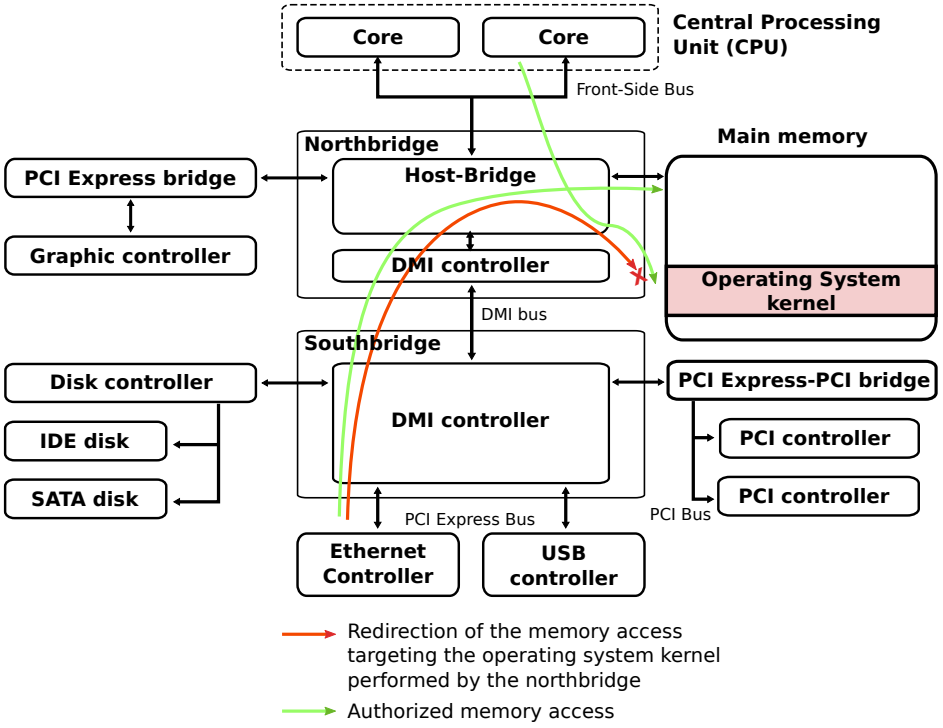


Figure 3. Redirecting the memory access from I/O controllers

2) Peer-to-peer DMA attacks: literature also mentions DMA attacks targeting memory-mapped I/O locations of other I/O controllers. In order to differentiate them from regular DMA attacks aiming at the main memory, these attacks will be referred to as *peer-to-peer DMA attacks* in the following. By carrying out an abnormal access to another I/O controller's internal memory or registers, these attacks try to abuse the controllers' resources and features (*e.g.*, memory or processing capability in graphic cards). Such malicious actions are categorized in the *Class 3* (*cf.* Section I). The major advantage of peer-to-peer DMA attacks remain in the fact that they are difficult to detect as they do not require transfers through the main memory. Fortunately, those DMA attacks cannot be implemented on all chipsets. So far, the feasibility of such an attack has been demonstrated [23], [24] on Intel X58 [25] chipsets (Nehalem architecture). Intel 5100 [26], Intel 5500 and Intel 5520 [27] chipsets present probably the same feature, whereas some restrictions have been set up in Intel Q45 and in Intel Q35 chipsets. The reader is referred to our study [23] for a detailed discussion on the possibility to carry out such attacks on current chipsets.

The first demonstration of peer-to-peer DMA attacks has been done by M. Dornseif in [9]. He developed a proof-of-concept in which he managed to modify the content showed on the screen of a laptop, by altering the framebuffer of its graphic controller by means of a FireWire device. In the same vein, we implemented a screen-grabbing attack through DMA to illustrate our study on peer-to-peer DMA attacks [23] on recent chipsets. We used DMA in FireWire controllers to dump the framebuffer of a graphic controller [24]. Finally, A. Triulzi implemented a more complete attack scenario [28], [29]: by loading a customized firmware in a network controller, he was able to communicate with a graphic controller in which was implanted a minimal secure shell server. This remote access enabled him to load a customized firmware in another network controller, beforehand downloaded from the Internet. This customized firmware instructed the second network controller to directly pass packets to the first network card, enabling the attacker to bypass any network packet filtering rules implemented by the operating system kernel.

C. Countermeasures

Some countermeasures can be set up on a system to mitigate such attacks. They are discussed in this subsection. Some of their respective limitations are also outlined.

1) Modifying the system address map to defeat DMA attacks: as described in Section II-B2, the memory space maps different kinds of memory (the main memory, memory-mapped I/O locations, *etc.*). Each of these memory types has its own general characteristics such as alignment, variability of starting address and size, or the system action on access to the area. The chipset relies on programmable registers to make transparent the access to these different types of memory. One can alter these registers in order to modify the system address map and defeat DMA attacks. Such technique has been presented by J. Rutkowska [30] to thwart DMA attacks targeting the main memory in AMD64 [31] systems (Figure 3). Her idea consisted in making the chipset believe

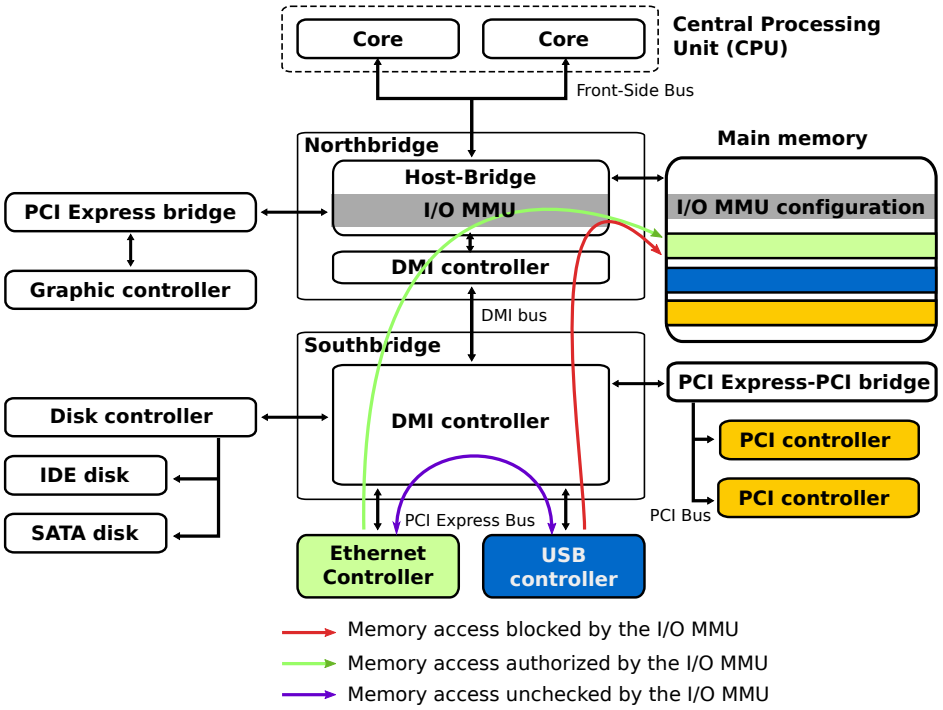


Figure 4. Authorized and blocked accesses when using an I/O MMU

that the critical region of the main memory to protect, where a hypervisor can have been loaded for instance, was an MMIO area. Thus, DMA requests targeting this memory region were not decoded by the main memory controller and were bounced to the I/O buses as if it was targeting an I/O controller.

Such technique is efficient to protect the main memory. However, applying this solution to protect memory-mapped I/O locations seems difficult as the routing of DMA requests depends on the chipset configuration. Let us note also that setting up such a countermeasure requests a lot of caution. The operating system kernel may not have been designed to support such a system memory mapping change, and there is a risk to render the system unstable or to hang it up.

2) Input/Output Memory Management Unit (I/O MMU): another way to block undesired memory requests from I/O controllers consists in using an I/O MMU such as Intel VT-d [32] on Intel chipsets or AMD IOMMU [33] on AMD ones.

An I/O MMU is a hardware component embedded in recent chipsets that enables the operating system to virtualize the main memory for I/O controllers and to verify their accesses to it according to a configuration loaded in the main memory itself and set by the operating system. It is usually embedded in the northbridge, and located between the block formed by the I/O controllers and the main memory (Figure 4).

Because of its location in the chipset, an I/O MMU controls efficiently any DMA requests transiting through the northbridge. However, our experiments on peer-to-peer DMA attacks [23] demonstrated that it is inefficient against certain attacks involving only I/O controllers attached to the southbridge. Actually, we still succeeded in carrying out the screen-grabbing attack [24] with a graphic controller connected to the southbridge while the I/O MMU was activated. The peer-to-peer DMA attack demonstrated by A. Triulzi in [29], in which two network cards (probably connected to the southbridge) cooperated to exchange Ethernet frames without involving the operating system, is likely to succeed as well.

It is well worth noting that literature mentions other I/O MMU limitations. The reader is referred to [34], [23], [35] for a full discussion of these limitations.

3) Access Control Service (ACS) extensions: to address the issues related to attacks coming from I/O controllers, the PCI Special Interest Group (PCI-SIG), consortium in charge of drafting the conventional PCI [3], the PCI-X and the PCI Express [4] specifications, has defined a set of security-oriented extensions for I/O bus bridges, referred to as Access Control Services (ACS). The components that implement the ACS capability can be configured to perform various access control oriented actions at the reception of an I/O request. An excerpt of the access control services specified by the PCI-SIG is presented in the following:

- ***ACS Source Validation.*** When enabled on an I/O bus bridge, this feature configures the component to check systematically the identity[4] advertised by the I/O controller issuing an I/O request. Such feature prevents, for instance, an I/O controller to spoof the identity of another I/O controller, and prevents, as a result, the exploitation of a limitation of the I/O MMU discussed in [34].

[4]When issuing a request, an I/O controller defines its identity through a unique triplet composed of the I/O controller bus, device and function identifiers referred to as its requester-id (*cf.* Section II-B3).

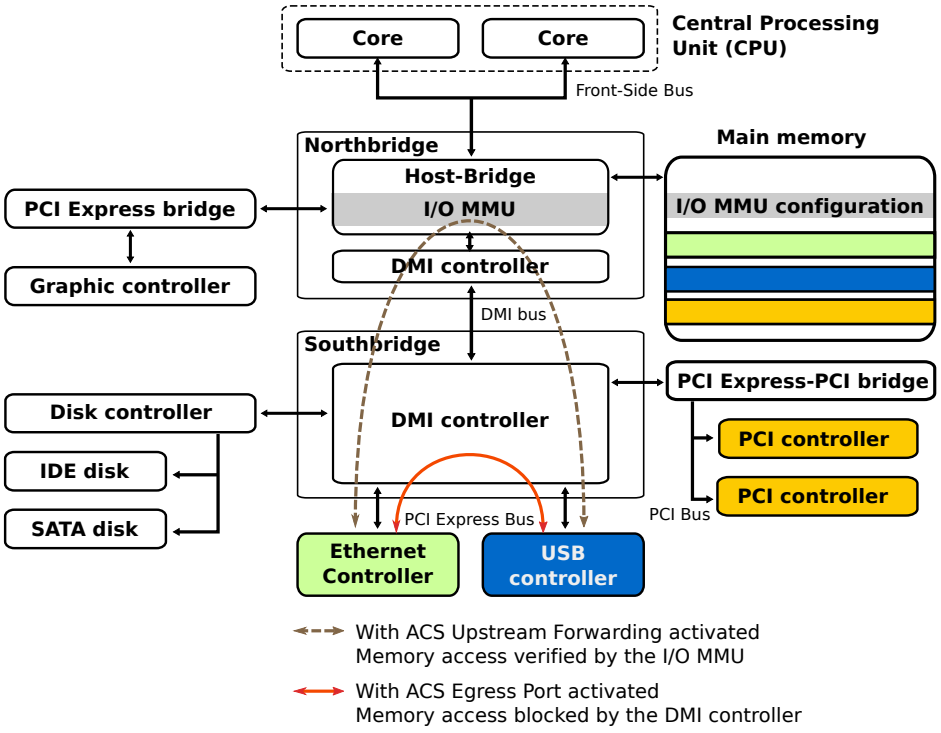


Figure 5. Behavior with the activation of ACS extensions in the southbridge

- ***ACS Upstream Forwarding.*** This feature configures the I/O bus bridge which implements this capability to redirect any requests from I/O controllers to an upstream component for validation. The upstream component can decide to authorize or to block the request, or it can, in turn, delegate the decision to another upstream component. In the example depicted in Figure 5, the DMI controller connected to the southbridge passes the request to the DMI controller located in the northbridge to decide whether the Ethernet controller can communicate directly with the USB controller. The latter further forwards it to the I/O MMU which decides finally whether to block or to authorize it by forwarding back the request downstream.
- ***ACS P2P Egress Control.*** This feature configures the I/O bus bridge which implements this capability to block any peer-to-peer communication going through it.

Implementing these extensions in the I/O bus bridges of the southbridge (Figure 5) can mitigate some uncontrolled peer-to-peer access in the southbridge. For instance, the *ACS Upstream Forwarding* feature forces the I/O bus bridges in the southbridge to forward all requests to the northbridge for validation. In the same vein, the *ACS P2P Egress Control* feature can be used to block any peer-to-peer communications in the southbridge. Unfortunately, none of the chipsets we analyzed [23] have Access Control Services extensions implemented yet in the southbridge.

IV. Possible future I/O attacks

Countermeasures against I/O attacks mainly focus on attacks through DMA. There are chances for some other I/O mechanisms to be still less strictly controlled than memory accesses. An attacker may take advantage of this situation to bypass protection mechanisms implemented in current systems [35]. In this section, we present various I/O attacks we are currently experimenting and we discuss their possible impact.

A. Peer-to-peer PIO attacks

Although PIO is a legacy mechanism to communicate with I/O controllers, many I/O controllers still provide PIO interfaces that enable to control them. It is still possible for instance to drive a USB controller or a disk controller using only `in` and `out` assembler instructions. One of the I/O attacks we are currently considering consists in sending directly PIO requests from an I/O controller to another I/O controller in order to abuse its resources. Such attacks are alternatives to peer-to-peer DMA attacks and also fall into *Class 3* (*cf.* Section I). Unfortunately, no countermeasure to these attacks is likely to be developed in the future as the PCI-SIG deprecates the use of PIO in controllers.

B. Peer-to-peer PCI configuration attacks

PCI and PCI Express specifications forbid I/O bridges to forward peer-to-peer configuration requests upstream. However, to enable manageability, Intel has implemented this forbidden capability in its chipsets. As the configuration registers of the chipset are mapped in the PCI configuration

space, this capability may be exploited by an attacker to alter for instance the state of the chipset from an I/O controller. Such malicious actions are categorized in the malicious logic actions in *Class 2* (*cf.* Section I). In that case, attacks which required the execution of some malicious piece of code on the CPU to modify the execution environment memory, such as [36], [37] for instance, could also be carried out from an I/O controller.

Note that few I/O controllers have the ability to generate such PCI requests on I/O buses. This fact likely explains why those attack vectors have not been exploited yet, and why chipset manufacturers have not considered these issues so far. Let us note that more and more I/O controllers are developed using programmable logic technologies. Such I/O controller could have been reconfigured for instance at manufacturing time or diverted by an attacker [38] to hide a backdoor. Thus, these cases should not be neglected. Within this perspective, we are currently developing a dedicated I/O controller using FPGA technologies to analyze the possible impact of their exploitation.

V. Conclusion and future work

This article presents a short survey on I/O attacks in Intel PC architectures. The main advantage (from the attacker's point of view) is that such attacks are difficult to detect by the CPU. But to perform them, an attacker must first take the control of an I/O controller, which has been shown easy in some circumstances. Within this perspective, we analyzed which malicious actions an attacker can carry out against an information system using an I/O controller and which countermeasures exist. We proposed a classification scheme for I/O attacks based on their purpose. Amongst the studied I/O attacks, some alter the main memory in order to subvert software components running upon the system while others abuse an I/O controller to attack another I/O controller in order to exploit its resources. We also presented different techniques to mitigate I/O attacks and we mentioned some of their limitations. Finally, we described several I/O attacks we are currently investigating. At the time of writing, we are developing an dedicated I/O controller using FPGA technologies. This I/O controller will enable us to perform peer-to-peer PIO attacks and peer-to-peer PCI configuration attacks with the objective of evaluating experimentally the impact of such I/O attacks. As future work, it would be interesting to use this I/O controller to analyze, for instance, the resilience of the chipset to malformed requests or to use it as an oracle for system integrity monitoring, both at software and hardware (or firmware) level.

References

[1] E. Lacombe, V. Nicomette, and Y. Deswarte, "Enforcing Kernel Constraints by Hardware-assisted Virtualization," vol. 7, no. 1. Springer, 2011, pp. 1–21. [Online]. Available: http://www.springerlink.com/content/v0w56774150764vr/

[2] K. Piromsopa1 and R. J. Enbody, "Survey of Protections from Buffer-Overflow Attacks," vol. 15, no. 2. Faculty of Engineering, Chulalongkorn University, 2011, pp. 31–52. [Online]. Available: http://www.ej.eng.chula.ac.th/eng/index.php/ej/article/view/112

[3] PCI Special Interest Group (PCI-SIG), *PCI Local Bus Specification – revision 2.3*, Mar. 2002. [Online]. Available: http://www.pcisig.com/specifications/conventional/conventional_pci_23/

[4] ——, *PCI Express™ Base Specification – Revision 1.1*, Mar. 2005. [Online]. Available: http://www.pcisig.com/specifications/pciexpress/specifications/

[5] ——, *PCI-to-PCI Bridge Architecture Specification – Revision 1.1*, Dec. 1998. [Online]. Available: http://www.pcisig.com/specifications/conventional/pci_to_pci_bridge_architecture/

[6] LSI Corporation, Hewlet-Packard Company, Intel Corporation, Microsoft Corporation, NEC Corporation, and ST-NXP Wireless Company, *On-The-Go and Embedded Host – Supplement to the USB Revision 2.0 Specification*, 8 May 2009. [Online]. Available: http://www.usb.org/developers/onthego/USB_OTG_and_EH_2-0.pdf

[7] Apple Corporation, Compaq Computer Corporation, Intel Corporation, Microsoft Corporation, National Semiconductor Corporation, Sun Microsystems, Inc., and Texas Instruments, Inc., *1394 Open Host Controller Interface Specification*, Jan. 2000. [Online]. Available: http://download.microsoft.com/download/1/6/1/161ba512-40e2-4cc9-843a-923143f3456c/ohci_11.pdf

[8] M. Becher, M. Dornseif, and C. N. Klein, "FireWire - all your memory are belong to us," in *CanSecWest/core05*, 4–5 May 2005. [Online]. Available: http://md.hudora.de/presentations/#firewire-cansecwest

[9] M. Dornseif, "0wned by an iPod - hacking by Firewire," in *PacSec/core04*, 11-12 Nov. 2004. [Online]. Available: http://md.hudora.de/presentations/#firewire-pacsec

[10] D. Aumaitre, "A Little Journey Inside Windows Memory," vol. 5, no. 2. Springer, 2009, pp. 105–117. [Online]. Available: http://www.springerlink.com/content/eu55858362752617/

[11] A. Boileau, "Hit by a Bus: Physical Access Attacks with FireWire," in *RUXCON 2006*, Oct. 2006. [Online]. Available: http://www.ruxcon.org.au/files/2006/firewire_attacks.pdf

[12] L. Duflot, Y.-A. Perez, G. Valadon, and O. Levillain, "Can you still trust your Network Card?" in *CanSecWest/core10*, 24-26 Mar. 2010. [Online]. Available: http://www.ssi.gouv.fr/IMG/pdf/csw-trustnetworkcard.pdf

[13] C. Devine and G. Vissian, "Compromission physique par le bus PCI," in *Proceedings of the 7th Symposium sur la Sécurité des Technologies de l'Information et des Communications (SSTIC 2009)*, Jun. 2009, pp. 169–193. [Online]. Available: http://actes.sstic.org/SSTIC09/Compromission_physique_par_le_bus_PCI/

[14] D. Aumaitre and C. Devine, "Subverting Windows 7 x64 Kernel with DMA attacks," in *HITBSecConf 2010 Amsterdam*, 29 Jun. - 2 Jul. 2010. [Online]. Available: http://conference.hackinthebox.org/hitbsecconf2010ams/materials/D2T2%20-%20Devine%20&%20Aumaitre%20-%20Subverting%20Windows%207%20x64%20Kernel%20with%20DMA%20Attacks.pdf

[15] B. Carrier and J. Grand, "A Hardware-based Memory Acquisition Procedure for Digital Investigations," *Digital Investigation*, vol. 1, no. 1, pp. 50–60, Feb. 2004. [Online]. Available: http://www.digital-evidence.org/papers/tribble-preprint.pdf

[16] J. Nick L. Petroni, T. Fraser, J. Molina, and W. A. Arbaugh, "Copilot - a Coprocessor-based Kernel Runtime Integrity Monitor," in *13th USENIX Security Symposium*, 9-13 Aug. 2004. [Online]. Available: http://www.usenix.org/events/sec04/tech/full_papers/petroni/petroni.pdf

[17] L. Duflot, "Contribution à la sécurité des systèmes d'exploitation et des microprocesseurs," Ph.D. dissertation, Université de Paris XI, Oct. 2007. [Online]. Available: http://www.ssi.gouv.fr/archive/fr/sciences/fichiers/lti/these-duflot.pdf

[18] Intel Corporation, *Universal Host Controller Interface (UHCI) Design Guide*, Intel Corporation, Mar. 1996. [Online]. Available: http://download.intel.com/technology/usb/UHCI11D.pdf

[19] OpenBSD core team, "The OpenBSD project," 2010. [Online]. Available: http://www.openbsd.org

[20] D. R. Piegdon, "Hacking in Physically Addressable Memory," in *Seminar of Advanced Exploitation Techniques, WS 2006/2007*, 12 Apr. 2007. [Online]. Available: http://david.piegdon.de/papers/rwth-SEAT1394-svn-r432-paper.pdf

[21] D. Maynor, "0wn3d by everything else - USB/PCMCIA Issues," in *CanSecWest/core05*, 4-5 May 2005. [Online]. Available: http://cansecwest.com/core05/DMA.ppt

[22] G. Delugré, "Closer to metal: reverse-engineering the Broadcom NetExtreme's firmware," in *Hack.lu*, Luxembourg, 27-29, Oct. 2010. [Online]. Available: http://esec-lab.sogeti.com/dotclear/public/publications/10-hack.lu-nicreverse_slides.pdf

[23] F. Lone Sang, V. Nicomette, Y. Deswarte, and L. Duflot, "Attaques DMA peer-to-peer et contremesures," in *Proceedings of the 9th Symposium sur la Sécurité des Technologies de l'Information et des Communications (SSTIC 2011)*, Jun. 2011, (to be published soon at: http://www.sstic.org/2011/actes/).

[24] F. Lone Sang, V. Nicomette, and Y. Deswarte, "Demonstration of a peer-to-peer DMA Attack against the Framebuffer of a Graphic Controller Through FireWire," Jan. 2011. [Online]. Available: http://homepages.laas.fr/nicomett/Videos/

[25] I. Corporation, *Intel X58 Chipset - Datasheet*, Nov. 2009, http://www.intel.com/assets/pdf/datasheet/320838.pdf.

[26] Intel Corporation, *Intel 5100 Memory Controller Hub Chipset – Datasheet*, Intel Corporation, Jun. 2009. [Online]. Available: http://www.intel.com/Assets/PDF/datasheet/318378.pdf

[27] ——, *Intel 5520 Chipset and Intel 5500 Chipset – Datasheet*, Intel Corporation, Mar. 2009. [Online]. Available: http://www.intel.com/assets/pdf/datasheet/321328.pdf

[28] A. Triulzi, "Project Maux Mk.II - "I 0wn the NIC, Now I want a Shell!"," in *PacSec/core08*, 12-13 Nov. 2008. [Online]. Available: http://www.alchemistowl.org/arrigo/Papers/Arrigo-Triulzi-PACSEC08-Project-Maux-II.pdf

[29] ——, "The Jedi Packet Trick takes over the Deathstar (or: "Taking NIC Backdoors to the Next Level")," in *CanSecWest/core10*, 24-26 Mar. 2010. [Online]. Available: http://www.alchemistowl.org/arrigo/Papers/Arrigo-Triulzi-CANSEC10-Project-Maux-III.pdf

[30] J. Rutkowska, "Beyond The CPU: Defeating Hardware Based RAM Acquisition," in *BlackHat DC*, 28 Feb. 2007. [Online]. Available: http://www.blackhat.com/presentations/bh-dc-07/Rutkowska/Presentation/bh-dc-07-Rutkowska-up.pdf

[31] Advanced Micro Devices (AMD), *BIOS and Kernel Developer's Guide for AMD Athlon 64 and AMD Opteron Processors*, 3rd ed., Feb. 2006. [Online]. Available: http://support.amd.com/us/Processor_TechDocs/26094.PDF

[32] Intel Corporation, *Intel Virtualization Technology for Directed I/O - Architecture Specification*, Intel Corporation, Sep. 2008. [Online]. Available: http://download.intel.com/technology/computing/vptech/Intel(r)_VT_for_Direct_IO.pdf

[33] Advanced Micro Devices (AMD), *AMD I/O Virtualization Technology (IOMMU) - Architectural specification*, Feb. 2009. [Online]. Available: http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/34434.pdf

[34] F. Lone Sang, E. Lacombe, V. Nicomette, and Y. Deswarte, "Exploiting an I/OMMU Vulnerability," in *Proceedings of the 5th IEEE International Conference on Malicious and Unwanted Software (MALWARE)*, 19-20 Oct. 2010, pp. 7–14. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5665798

[35] R. Wojtczuk and J. Rutkowska, "Following the White Rabbit: Software Attacks against Intel VT-d," Invisible Things Lab (ITL), Tech. Rep., May 2011. [Online]. Available: http://www.invisiblethingslab.com/resources/2011/Software%20Attacks%20on%20Intel%20VT-d.pdf

[36] J. Rutkowska and R. Wojtczuk, "Preventing and detecting xen hypervisor subversions," in *Blackhat USA, Las Vegas*. Invisible Things Labs, 7 Aug. 2008. [Online]. Available: http://invisiblethingslab.com/resources/bh08/part2-full.pdf

[37] A. Tereshkin and R. Wojtczuk, "Introducing ring -3 rootkits," in *Blackhat USA, Las Vegas*. Invisible Things Labs, 30 Jul. 2009. [Online]. Available: http://invisiblethingslab.com/resources/bh09usa/Ring%20-3%20Rootkits.pdf

[38] F. Domke, "Distributed FPGA Number Crunching For The Masses," in *27th Chaos Communication Congress (27C3)*, 28 Dec. 2010. [Online]. Available: http://https://events.ccc.de/congress/2010/Fahrplan/attachments/1801_27C3%20-%20Distributed%20FPGA%20Number%20Crunching%20for%20the%20Masses.pdf

CAPTCHuring Automated (Smart)Phone Attacks

Iasonas Polakis, Georgios Kontaxis and Sotiris Ioannidis
Institute of Computer Science,
Foundation for Research and Technology Hellas, Greece
email: {polakis, kondax, sotiris}@ics.forth.gr

***Abstract*—As the Internet has entered everyday life and become tightly bound to telephony, both in the form of Voice over IP technology as well as Internet-enabled cellular devices, several attacks have emerged that target both landline and mobile devices. We present a variation of an existing attack, that exploits smartphone devices to launch a DoS attack against a telephone device by issuing a large amount of missed calls. In that light, we conduct an excessive study of *Phone CAPTCHA* usage for preventing attacks that render telephone devices unusable, and provide information on the design and implementation of our system that protects landline devices. Subsequently, we propose the integration of Phone CAPTCHAs in smartphone software as a countermeasure against a series of attacks that target such devices. We also present various enhancements to strengthen CAPTCHAs against automated attacks. Finally, we conduct a user study to measure the applicability of our enhanced Phone CAPTCHAs.**

I. Introduction

The advancement of computers has also led to the evolution of telephone technology. From traditional PSTN networks and mobile devices we have moved on to the era of Voice over IP (VoIP) and smartphones. Integration of the Internet into everyday life has led to the demand for web access on-the-go and the widespread adoption of new generation mobile devices. Such Internet-enabled devices are becoming increasingly popular, with smartphone users expected to exceed 1 billion worldwide by 2014 [8]. Additionally, VoIP subscribers will reach almost half a billion worldwide by 2012 [6]. Thus, one can expect that in the near future, legacy telephony technologies will slowly become obsolete. Nonetheless, in this transitional period where such technologies co-exist, we can expect the emergence of new threats that exploit their interconnection. As demonstrated in our previous work [22], as well as in the wild [12], an attacker can leverage VoIP technology to flood traditional telephone devices with a large number of missed calls[1] and render them unusable. We demonstrated the feasibility of such an attack, which we refer to as DIAL attacks, by leveraging VoIP technology.

Additionally, by exploiting vulnerabilities in smartphone software, attackers have proven the feasibility of issuing phone calls towards arbitrary numbers [2]. Based on these two attack classes, we argue that one can perform a DIAL attack against a target telephone device using traditional cellular networks instead of VoIP technology. This variance of DIAL attacks preserves the key characteristics of its initial form and also presents a very important advantage for the attacker; it requires zero financial resources from the attacker in all cases. Even if the victim answers an incoming call, the compromised smartphone will be charged.

While smartphones have been targeted by a small number of malware to date, experts predict that they will attract a large number in the immediate future [7], [3], as attackers will be tempted by the built-in payment mechanism available in phones. Security vendors have already caught malware that targets smartphones, and issues a series of phone calls towards premium-rate numbers for profit.

In this work we expand the notion of Phone CAPTCHAs as a countermeasure against DIAL attacks. We explore several axes upon which they can be improved. We also propose their use as defense mechanisms against several recent attacks that target smartphones. Our key contributions are summarized as follows:

- As shown in our previous work, end telephone devices have little means to defend themselves from a DIAL attack. To mitigate this effect, we implemented a fully functional call center incorporating *Phone CAPTCHAs* for protecting telephone devices from such attacks. Furthermore, we propose a series of improvements to traditional audio CAPTCHAs to strengthen them against voice recognition attacks.
- We expand the idea of DIAL attacks and demonstrate that by exploiting a vulnerability in a smartphone, one can leverage cellular networks for flooding a target telephone device with calls.
- We propose the modification of smartphone operating system API calls to incorporate client-side Phone CAPTCHAs so as to prohibit compromised devices from issuing arbitrary calls.
- We conduct a user study that demonstrates the applicability of Phone CAPTCHAs, as first-time, non-native users managed to successfully solve the CAPTCHAs in 71% to 83% of the cases. We consider this to be very satisfactory for the newly introduced CAPTCHAs.

[1]The attacker initiates a large number of calls which are terminated immediatelly so the victim can not answer them.

II. Attacks targeting landline, mobile and smart phones

In this Section we present a series of attacks that target telephone devices. Initially we review attacks that have been presented in the past, and continue with a new attack scenario that combines chracteristics of previous attacks.

A. First generation DIAL attacks

These attacks leverage a series of characteristics inherent in VoIP technology to target legacy telephone devices (both landline and cellular). An adversary is able to incapacitate a telephone device or calling center and obstruct legitimate callers from getting through. As we demonstrated in [22], the attack is carried out by injecting a large amount of missed calls towards a target telephone number and, thus, rendering it unusable. The adversary uses the SIP [26] protocol to register and communicate with a VoIP provider for the routing of the attack calls. By carefully placing a large number of call initiation and termination pairs, the attacker can keep the target device continuously *busy* and hinder legitimate callers from accessing it. This attack aims to disrupt the normal operation of a telephone device and can indirectly lead to profit for the attacker [12].

B. "Smart" Dialers

With the wide adoption of smartphones, an old familiar attack has resurfaced. Dialers [10] used to infect computers with Dial-Up Internet access and reconfigure their modem to place calls towards premium-rate numbers. DSL connections, which operate over a virtual private circuit with the ISP, had rendered such attacks ineffective. However, smartphones combine telephone devices capable of "dialing-in", with a sophisticated environment capable of executing arbitrary code and, at the same time, offer a full-featured browser access to the Internet. Therefore, smartphones present a large attack surface as their users visit arbitrary sites on the web. In late 2008, a bug [9] in Apple's Safari web browser for the iPhone device, could be exploited by a malicious website to initiate a phone call, to a destination chosen by the attacker, without user interaction. Furthermore, in 2009, a security researcher demonstrated a technique [23], [5] for discovering software vulnerabilities in smartphones and also exploiting them, all via SMS. An exploitation of such a vulnerability can result in the malware infection of the phone. It is, therefore, evident that smartphones suffer from the same flaws as standard computers and are also appealing targets because of their dial-up capabilities. They can be exploited to deliver distributed DIAL attacks or commit financial fraud with unauthorized charges. These attacks take advantage of the built-in billing system that mobile phones have and result in direct profit for the perpetrator.

C. Second generation DIAL attacks

The first generation of DIAL attacks was a result of the inter-connection of traditional telephony networks and a relatively new technology, namely Voice Over IP. The second generation, emerges from the integration of a new set of capabilities, traditionaly found in computers, in mobile devices. This bundling of disjoint services allows the exploitation of one service to gain access to the other. Thus, building on the notion of DIAL attacks, an adversary can exploit vulnerabilities in smartphones to flood a target telephone device with a large amount of missed calls. Additionaly, the adversary can masquerade his attack by hiding it inside a seemingly innocent application. In both cases, the adversary instructs the smartphone device to issue calls towards an arbitrary telephone number. For this type of DIAL attacks, traditional mobile telephony networks are used as the attack medium as opposed to VoIP providers. In spite of the built-in billing system of smartphones, this attack does not aim to exploit it and lead to a monetary profit for the adversary, but rather uses it to carry out the attack.

III. Client Side Countermeasures

In this Section we present in detail the defense mechanism we implemented to protect landline devices from DIAL attacks. Telephone devices currently have no means of defence against the attack outlined in this paper. Our goal is to enable a potential target to defend against an attack utilizing IP telephony technology regardless of the countermeasures that Voip providers may, or may not, incorporate. We are, to the best of our knowledge, the first to implement a complete system that can hinder attackers from rendering a landline useless. Our solution is based on *Phone CAPTCHAs*. A CAPTCHA[30] is a challenge test that requires a response before an action can be performed, and is used in computing to ensure that the action is not automatically initiated by a computer. The goal is to prevent computer programs from performing certain actions that will lead to the degradation of quality of a certain service. Although there is much debate over the use of CAPTCHAs there is plenty of recent academic effort related to this area [17], [21], [33].

A. Architecture

The goal of our system is to protect landlines from the DIAL attack as described in Section II. Furthermore, it can be used to block automated SPAM over IP Telephony (SPIT) calls [25], the number of which will continue to increase as VoIP technology becomes cheaper and widely adopted. Here, we focus on how to defend against the attack. Nonetheless, our system needs no modifications to filter-out automated SPIT phone calls. We will first describe the components that comprise our system and then how we model *Phone CAPTCHAs*.

Software. The core component of our platform is the Asterisk PBX, an open-source software implementation of

a private branch exchange (PBX). Hardware private branch exchanges are used to make connections amongst the internal telephones of an organization. Asterisk can deliver voice over a data network and inter-operate with the Public Switched Telephone Network (PSTN) so as to create an automated call center. Asterisk also supports Interactive Voice Response (IVR) technology, and can detect touch tones, i.e. dual-tone multi-frequency (DTMF) signaling, and respond with pre-recorded messages or dynamically created sound files. For Asterisk to work as a PBX, dial plans have to be created to control all the devices and users connected to the system. Configuration files are used to register devices and users, and to define actions to be performed for incoming and outgoing calls.

A native language is used to define contexts, extensions and actions. Devices and users are assigned to a context that defines their dial plan and, thus, restricts the extensions they may access and the calls they can commence. This can be used to enforce organization policies regarding access permission for user groups. A context can contain several extensions, and is structured as a sequence of lines, each belonging to a specific extension. Extensions consist of several ordered lines, where each line performs actions on known variables or executes one of the many applications available to Asterisk. Each line has the following components: an extension, a priority and a command with its parameters.

Hardware. For Asterisk to handle landlines, the host machine must be equipped with specialized hardware that connects it to the PSTN circuit. Depending on the hardware used, several landlines can be connected to the host and handled by Asterisk. With the use of such specialized hardware and Phone CAPTCHAs, organizations and home users can defend against VoIP-based DoS attacks targeting landlines, as described next. Figure 1 is a diagram of a call center incorporating Phone CAPTCHA technology to be deployed as a defence measure against the attack outlined in this paper. The prototype we implemented, protects a single landline, as opposed to the diagram that depicts a call center protecting numerous landlines.

B. Phone CAPTCHAs

We use Phone CAPTCHAs as a countermeasure to the attack described in our previous work. Our Phone CAPTCHA is a type of CAPTCHA crafted for use with the Asterisk PBX, but that could easily be deployed by any software PBX that supports IVR technology and call queues. When an incoming call is received, Asterisk places the call in a call queue. The caller, then, receives a Phone CAPTCHA and has a limited amount of time to answer the CAPTCHA test using the phone's dial pad. The Phone CAPTCHA test requires the caller to press a sequence of keys based on the instructions presented to him by a recorded message. If the caller provides the correct answer, Asterisk forwards the call to its destination as determined by the dial plan. Otherwise the call is dropped. This mechanism prohibits automated calls from binding to the end device and consuming resources, which could prevent legitimate callers from reaching the destination number. Even if the attacker probes with high rates and terminates the calls immediately upon receiving a RINGING tone, our system is not affected since these calls never get past the Phone CAPTCHA to the final destination. With our defense mechanism, attackers must incorporate automatic speech recognition software in their effort to successfully launch an attack. On the other hand, it is trivial for legitimate callers to pass the phone CAPTCHA test.

A fundamental requirement for Phone CAPTCHAs to be effective against multiple attackers is the utilization of call queues. With the use of call queues, incoming calls are sent to a queue where they must pass a Phone CAPTCHA test before they are forwarded to the destination number. Without call queues, if the attackers can simultaneously issue more phone calls than the number of the target's available phone numbers, it will be nearly impossible for legitimate users to reach an available number.

The digital circuits of traditional PSTN lines are the basic granularity in telephone exchanges. That means that they have one channel and can only handle a single phone call. Higher capacity circuits such as the the T1 and E1 lines can multiplex 24 and 32 channels respectively. When a PSTN line is called, even though the call will be handled by Asterisk, and may never be forwarded to the end device, the line is occupied. Consequently, organizations with a limited number of PSTN lines cannot effectively utilize Phone CAPTCHAs against attackers with many resources, but can still deploy them as a filter against automated SPIT calls. With higher capacity circuits, multiple calls can be multiplexed through a single line and until the incoming calls match the number of available channels, the line will be available.

With the combined use of Phone CAPTCHAs and call queues, automated calls will not be forwarded to the end devices and high probing rates can be sustained. The critical infrastructure that is most likely to be targeted by attackers will be equipped with higher capacity circuits and multiple phone lines. It is common practice for organizations of this type, that have many available phone lines but only a limited number of personnel, to rely on traditional call queues to cope with multiple users. In these cases, Phone CAPTCHAs can be highly effective against this type of attack, since only legitimate callers will be forwarded to the personnel.

C. Limitations

In this Subsection we discuss a series of inherent limitations to our countermeasure platform that stem from the limitations of the system's individual components.

Attacking the infrastructure. Asterisk handles all incoming and outgoing calls and, thus, the system's effectiveness is bound by the maximum number of simultaneous

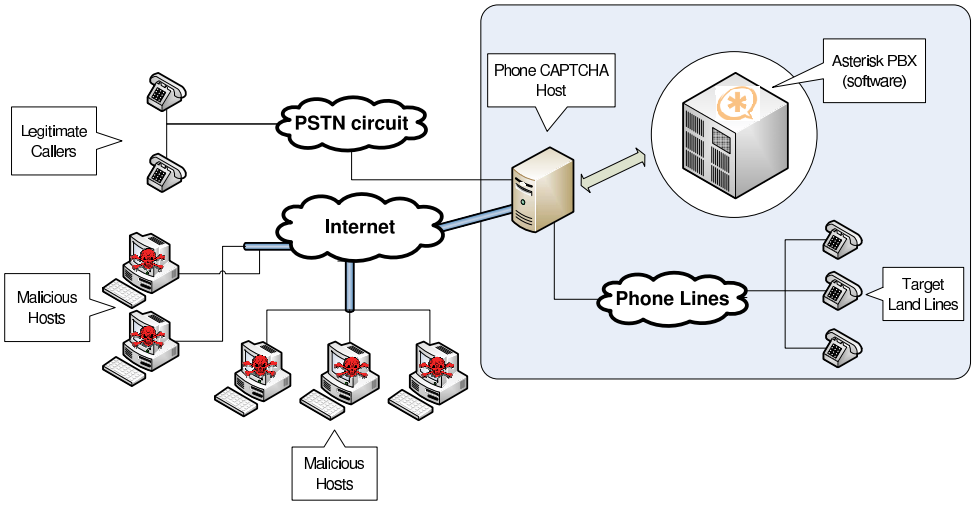


Figure 1. Diagram of a call center incorporating Phone CAPTCHA technology as a defence measure.

calls that can be handled by Asterisk, and its robustness against high calling rates. Attackers that don't have the ability to automatically solve the Phone CAPTCHAs may launch a DoS attack on the system by flooding it with such a large number of calls that Asterisk won't be able to handle. That will result in incoming calls being dropped or even the whole system crashing. For this attack to be successful, a much larger number and rate of incoming calls is demanded than in the case of an attack against a landline. Further investigation is needed to determine the threshold after which Asterisk is rendered ineffective, and whether a solution that relies on a cluster of Asterisk hosts and utilizes load balancing techniques can reinforce the infrastructure against such attacks. Even though this is a possibillity, measurements show that if Asterisk is deployed on a commodity desktop, it can easily handle more than 40 concurrent calls and can, thus, sustain the attack.

Breaking the Phone CAPTCHA. Phone CAPTCHAs are vulnerable to attacks that utilize automatic speech recognition (ASR) software to transcribe the audible information. For this type of attack [27], the adversary must first use sample data, preferably from the target speaker, to train the classifier. Even for a limited vocabulary, a large number of training samples is needed. After the training phase, the audio from the Phone CAPTCHA test can be input to these trained classifiers that will try to recognize and extract the information. Once the information has been extracted, the malicious script can send the corresponding DTMF signals to pass the test. The duration of the "deciphering" phase may vary depending on the presence of noise and distortion in the audio signal. For a successful attack, the model must decipher the message and answer the Phone CAPTCHA within the limited time before Asterisk terminates the call. In the Section V we present a series of enhancements for Phone CAPTCHAs that we implemented as well as some future directions.

Blocking legitimate callers. Even though the solution of a Phone CAPTCHA is a trivial task for a person under normal circumstances, under extreme conditions (e.g. panicked due to a fire, robbery etc.) people may not possess the mental lucidity to correctly answer the CAPTCHA. To overcome this, our defence mechanism can be utilized only when the infrastructure is receiving more calls than it can process and can use sampling to select only a number of calls to forward to the queues. That way, the phone line infrastructure can be offloaded since not all calls are presented with a Phone CAPTCHA. However, further investigation is needed to determine what types of heuristics could lead to effective sampling policies.

IV. Using Phone CAPTCHAs in smartphones

In this section we present the local use of Phone CAPTCHAs by smartphones, so as to prevent automated dial-out attacks. As mentioned in Section II-B, there have been successful exploits, misusing the phone's web browser, to dial arbitrary numbers without the user's interaction or knowledge. Example cases involve the abuse of browser URIs (e.g., tel:+123456789 or sms:+123456789, similar to http:), tapjacking [11] techniques and software (such as a valid game or application) that dials premium-rate numbers in the background [4]. Automatic dialing of arbitrary numbers could be leveraged, not only for reaching premium-rate destinations (financial fraud), but also for performing distributed DIAL attacks, similar to the ones described in Section II.

A. Preventing Automated Phone Actions

To prevent phone initiations we propose a modification to smartphones where local Phone CAPTCHAs are presented by the smartphone's operating system and lie in-line beneath the phone's API calls (i.e., Phone.Talk(number)). The user is presented with a CAPTCHA, which he must successfully

solve for the action to continue. It is undesirable for a user-level application to be able to circumvent the CAPTCHA challenge and directly dial the number. Therefore, considering that user level applications only have access to the API calls and not directly to system calls, our modifications are limited to the programmable interface. It is necessary to add the necessary logic, to all calls providing access to phone I/O capabilities, to present a CAPTCHA challenge and block the action until it is correctly solved. We propose to use a wrapper for each sensitive API call, for instance write a wrapper sTalk (or secureTalk) for the Talk() number-dialing API call. Such a wrapper will implement the challenge logic and either call the original Talk() or not. Of course, it is necessary for the original Talk() to be withdrawn from public API access. A straightforward way to do this, and not break current phone applications, is to rename Talk() to xTalk() and change the interface scope from public to private and at the same time, rename the sTalk() wrapper to Talk().

As requiring the user to solve such puzzles every time he initiates a call is frustrating, we define the following heuristic to detect cases where such input will be required:

(a) a number is about to be dialed, which is NOT present in the "recent outgoing calls" history
(b) Phone calls are issued towards emergency services (optionally such calls may be white-listed entirely or presented with a CAPTCHA only upon several consecutive calls in a short period of time).

B. Limitations

The use of local Phone CAPTCHAs, presented by the smartphone's operating system, can be bypassed if the OS itself is compromised by malware. The attacks described above employ techniques to access the phone's dial API for calling numbers or sending SMS. However, a smartphone rootkit that lives inside the system's core has direct access to the device I/O interface. Therefore, such malware can disable the CAPTCHA mechanism or bypass it completely, in which case the operating system is unaware of any calls taking place. While vendors are trying to hinder such actions with several techniques such as the Android permission system [2], there is no, currently available, technique to detect rootkits on smart phones [14]. Of course, CAPTCHA-solving can be enforced by the server, in which case it will not be possible to circumvent. However, such an implementation requires the server to keep a history of recent outgoing calls and present CAPTCHA challenges only for previously unseen ones, in accordance to the heuristics presented above.

Furthermore, if the attacker manages to forge the phone's history file, he could inject arbitrary destination numbers which would then be dialed without the need for CAPTCHA solving. However, current API calls either offer read-only access to the phone call history or none at all.

[2]http://developer.android.com/guide/topics/security/security.html

V. Phone CAPTCHA Enhancements

Simple phone CAPTCHAs may contain digits spoken by several speakers while other speakers that are audible in the background serve as noise that makes them more difficult for ASR software to break. However, as demonstrated by Tam *et al.* [27] this type of CAPTCHAs can be broken. Therefore it was vital to enhance phone CAPTCHAs in such ways that their solution remains trivial for a person, but are robust against existing software that can pass phone CAPTCHAs. In this Section we propose and implement enhancements that lead to the creation of more robust phone CAPTCHAs.

Expanding the vocabulary. The effectiveness and performance of speech recognition software is greatly affected by the training phase. A large number of labeled sample data is needed during this phase to correctly train the model to recognize a specific set of words. Even for a small set of words, a very large number of training samples is needed for the system to achieve high success percentage. Traditional audio CAPTCHAs rely on a very limited vocabulary, namely digits, for security. An example 4-digit phone CAPTCHA could be: "dial-eight-one-four-three".

A way to greatly extend the vocabulary, is to incorporate the use of words. Words have been widely used in the US to help people memorize telephone numbers by "translating" numbers into letters. Based on that principle, phone CAPTCHAs can randomly select a word and ask the caller to spell it. [3] This will exponentially increase the complexity of the ASR's model and the duration of the training phase needed so as to be able to recognize such an immense vocabulary. It is important to note, that selected words should meet certain criteria, such as having a relatively small length, and being easy to spell, so as not to inhibit legitimate users from passing the test.

We intend on exploring the following dimensions for the further enhancement of phone CAPTCHAs.

Speech distortion. Removing noise from audio signals has been an active research area exhibiting successful results[19], [15], [16]. The use of algorithms to distort the speech signal itself, and not merely add background noise, in a way that would not result in CAPTCHAs unintelligible for humans, might prove to be an effective way to render speech recognition software inefficient against phone CAPTCHAs. The software would be much less accurate and efficient as it would have a high word error rate (WER) and higher real time factor (RTF). However this requires a lot of research and testing before definitive conclusions can be drawn.

Incorporating semantics. The previous enhancements aim to prolong the duration of the ASR's training phase by increasing the vocabulary, or to render the system slow and inaccurate. These enhancements result in raising the bar for attackers trying to break phone CAPTCHAs. We propose

[3]For example, the phone CAPTCHA could say "spell chair". Then, based on the letter-to-dialpad mapping, the caller would have to dial "24247".

the incorporation of semantic information as a way of eliminating existing speech recognition software as a tool for breaking phone CAPTCHAs. For attackers to automatically solve CAPTCHA tests containing semantic information, their software must not only be able to recognize words from a vast vocabulary, but also use machine learning techniques and knowledge representation in order to correctly answer the tests. Even questions that are trivial for humans to answer, such as "Which animal hunts mice?" or "What color is a red car?", demand very sophisticated software. By issuing more elaborate questions, that are still answerable by people, would require the attackers to have software far more advanced than what is available today.

For our proof-of-concept countermeasure implementation we have implemented three types of phone CAPTCHAS. The first type vocalizes a series of digits. This type is the most trivial to be broken since it relies on a limited vocabulary and doesn't incorporate any noise, other than that of the recording and transmission media, i.e. the microphone and phone line. The second type requires a mathematical operation on two vocalized numbers, subsequently incorporating a larger vocabulary than the previous type. The final type requires users to use the dial pad to spell a vocalized word in order to pass the test. This type is the most robust against speech recognition attacks since it can vocalize any word from a a potentially huge vocabulary. Asterisk can utilize Festival[1] to dynamically synthesize the audio files for words from a dictionary. However, this would make it easier for an attacker to break the CAPTCHA since she could easily create a large number of training data. It is safer to use pre-recorded sound files and, preferably, from various speakers to create more robust CAPTCHAs. In our implementation, Asterisk randomly selects from a pool of words pre-recorded from a single speaker.

A. User case study

Here we present the results of the user case study we conducted using our proof-of-concept countermeasure implementation. The goal was to measure the usability of our client-side solution and utilize user-feedback to improve phone CAPTCHA design. The 14 test subjects that participated in the study were students and staff from our campus, between the ages of 22 and 32. They were randomly separated into two user groups, the Informed Group and the Uninformed Group. The members of the first group were fully informed of the nature of the experiment, while those of the second group were simply asked to dial a phone number. The users of the first group knew that there would be a succession of 15 phone CAPTCHA tests, separated into 3 sets of tests. The first 5 tests would ask the user to spell a word, the next 5 to type the result of a simple mathematical calculation, while the next 5 would be a random succession of tests of the first two types.

In Table I we see the results. As expected, the informed group achieved higher success rate (74-83%) than the uninformed one (63-74%) in the first two sets of tests, indicating that previous knowledge of the phone CAPTCHA type can lead to higher success rates. In our experiment both user groups had the same success rate (71%) in the final test set. The users of both groups scored worse in the case of mathematical calculations. Most users stated that after the first couple of tests, it was easier to solve them. The phone CAPTCHA tests contained a significant amount of noise which led users to mistakes because they couldn't always make out the words. Moreover, since the Phone CAPTCHAs were in English and the test subjects had varying degrees of familiarity with the English language, this deployment represents an international deployment. We expect the success rates to be higher for a national deployment (i.e., where the language of the Phone CAPTCHAs matches the native language of the users). Nonetheless, the informed group successfully solved the spelling CAPTCHA tests 83% of the time, which leads us to believe that native speakers will be able to solve phone CAPTCHAs (that don't incorporate additional noise) with extremely high probability. This indicates that the robustness of phone CAPTCHAs must stem from the vastness of the vocabulary used and not the incorporation of additional noise. Furthermore, while the calculation phone CAPTCHA type offers only a marginal improvement in robustness, relatively to the basic type, it actually resulted in lower success rates. On the other hand, the spelling type CAPTCHAs had a much higher success rate and can utilize an immense vocabulary, making them far more robust against automated voice recognition attacks.

User Group	Spelling Set	Calculation Set	Random Set
Informed Group	83	74	71
Uninformed Group	74	63	71

Table I
SUCCESS RATES(%) OF THE USER STUDY.

VI. RELATED WORK

Wang *et al.* [31] are able to identify and correlate VoIP calls anonymized by low latency networks, through a watermarking technique that makes the inter-packet timing of VoIP calls more distinctive. Wright *et al.* in [32], build a classifier that is able to identify with good probability the language used by callers, based on the VoIP packet sizes. Zhang *et al.* in [34] exploit the billing of VoIP systems that use the Session Initiation Protocol (SIP) and are able to bill users for calls that never happened or over-charge them for ones that did [26].

Research for attacks that target cellular telephony networks has been carried out in the past. Traynor *et al.* [28] argued that it is sufficient to reach a sending rate of 165 SMS messages per second, to incapacitate the GSM

networks all over Manhattan. They further explore such attacks in [29]. Enck *et al.* in [18] demonstrate the feasibility of using a simple cable modem to obstruct voice service. They claim that with the use of a medium-sized botnet it is possible to target the entire United States. They also present a series of countermeasures against SMS-based attacks. These include separating the voice and data transmission channels, provisioning for higher resource utilization, and rate limiting on-air interfaces. Nauman *et al.* [24] provide a policy enforcement framework for the Android that enables users to grant selective permissions to Android applications and prohibit the use of specific resources.

The research community has also investigated the possible implication of attacks against emergency services since emergency services base their operation on the telephony network. Aschenbruck *et al.* [13] report that it is possible to peer VoIP calls to public service answering points (PSAP). This peering can have grave implications because it makes it possible to carry out DoS attacks against emergency call centers. Thus, based on the technique presented in [22] it is possible for adversaries to target and take down emergency services by flooding ther call centers with a large amount of missed calls. Countermeasures that are to be deployed by VoIP providers were presented by the authors, and can lead to the mitigation of such attacks. Fuchs *et al.* in [20] investigate the applicability of intrusion detection in emergency call centers.

VII. Conclusion

In this paper we explore the use of Phone CAPTCHAs as defense mechanisms for a series of attacks targeting telephone devices. Initially, we build a fully functional call center to protect landlines from DIAL attacks. All incoming calls are placed in queues, where they are presented with a Phone CAPTCHA puzzle that the caller must solve before the call is forwarded to the telephone device. If the caller fails to answer correctly, the call is terminated. Therefore, automated calls never get through to the device which remains available for legitimate callers.

Next, based on the vulnerabilities that exploit smartphones to dial arbitrary numbers, we expand the notion of DIAL attacks and outline a new attack. By exploiting a smartphone, an adversary is able to issue a large number of missed calls towards a target telephone device. In this attack, the adversary no longer leverages VoIP technology, but leverages the advanced capabilities of smartphone devices.

To defend against this new type of attack, as well as those seen in the wild, we propose the incorporation of Phone CAPTCHAs in smartphone operating systems. By rendering the solution of a Phone CAPTCHA puzzle a prerequisite for issuing a phone call, we can successfully hinder adversaries from carrying out their attacks. Nonetheless, issuing a puzzle for every call would have a negative impact on the smartphone usability, and therefore we present a series of heuristics to be used by the smartphone operating system to decide whether a Phone CAPTCHA must be solved before the call is issued or not.

As shown in our previous work, however, traditional audio CAPTCHAs can be easily broken. Therefore, we propose a series of enhancements that harden CAPTCHAs against voice recognition attacks. By mapping words to numbers using a phone's dialpad and incorporating semantics in tests, an adversary will require far more sophisticated software than what is available today to break our CAPTCHAs.

Finally, we conduct a user study to measure the applicability of our enhancements for Phone CAPTCHAs. Our preliminary results show that users are able to solve the puzzles in 71% to 83% of the cases. Taking into account that the test subjects were not native English speakers, we consider these results to be very promising and demonstrate that our proposed enhancements can strengthen traditional audio CAPTCHAs against adversaries without hindering legitimate users from solving them.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement 257007. This work was supported in part by the Marie Curie Actions – Reintegration Grants project PASS. We thank the anonymous reviewers for their valuable comments. Iasonas Polakis, Georgios Kontaxis and Sotiris Ioannidis are also with the University of Crete.

References

[1] The Festival Speech Synthesis System. http://www.cstr.ed.ac.uk/projects/festival/.

[2] About the security content of the iPhone 1.1.1 Update. http://support.apple.com/kb/HT1571.

[3] dark READING - Smartphone Malware Multiplies. http://www.darkreading.com/insiderthreat/security/attacks/showArticle.jhtml?articleID=225402185.

[4] F-Secure - Trojanised Mobile Phone Game Makes Expensive Phone Calls. http://www.f-secure.com/weblog/archives/00001930.html.

[5] Forbes - How to HiJack 'Every iPhone In The World'. http://www.forbes.com/2009/07/28/hackers-iphone-apple-technology-security-hackers.html.

[6] IDC Predicts Almost Half a Billion Worldwide Personal IP Communications Subscribers by 2012. http://www.idc.com/getdoc.jsp?containerId=prUS21219408.

[7] Mobile malware will increase once crooks figure how to profit from it. http://www.rcrwireless.com/article/20101001/BSS_OSS/101009990/0.

[8] Number of Smartphone Users to Quadruple. http://www.marketwire.com/press-release/Number-of-Smartphone-Users-to-Quadruple-Exceeding-1-Billion-Worldwide-by-2014-1136308.htm.

[9] SecurityFocus - iPhone Safari phone-auto-dial vulnerability . http://www.securityfocus.com/archive/1/504403/30/0/threaded.

[10] Symantec - Glossary: Dialer. http://www.symantec.com/business/security_response/glossary.jsp#d.

[11] Tapjacking: owning smartphone browsers. https://media.blackhat.com/bh-us-10/whitepapers/Bursztein_Gourdin_Rydstedt/BlackHat-USA-2010-Bursztein-Bad-Memories-wp.pdf.

[12] Thieves Flood Victims Phone With Calls to Loot Bank Accounts. http://www.wired.com/threatlevel/2010/05/telephony-dos/.

[13] ASCHENBRUCK, N., FRANK, M., MARTINI, P., TOLLE, J., LEGAT, R., AND RICHMANN, H. Present and Future Challenges Concerning DoS-attacks against PSAPs in VoIP Networks. *Proceedings of the Fourth IEEE International Workshop on Information Assurance, April* (2006).

[14] BICKFORD, J., O'HARE, R., BALIGA, A., GANAPATHY, V., AND IFTODE, L. Rootkits on smart phones: attacks, implications and opportunities. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications*, HotMobile '10.

[15] DENG, L., DROPPO, J., AND ACERO, A. Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition. In *Speech and Audio Processing, IEEE Transactions on* (2003).

[16] DENG, L., DROPPO, J., AND ACERO, A. ALGONQUIN: Iterating Laplace's Method to Remove Multiple Types of Acoustic Distortion for Robust Speech Recognition. In *Speech and Audio Processing, IEEE Transactions on* (2005).

[17] ELSON, J., DOUCEUR, J., HOWELL, J., AND SAUL, J. Asirra: a CAPTCHA that exploits interest-aligned manual image categorization. In *Proceedings of the 2007 ACM Conference on Computer and Communications Security (CCS)*.

[18] ENCK, W., TRAYNOR, P., MCDANIEL, P., AND PORTA, T. L. Exploiting Open Functionality in SMS Capable Cellular Networks. In *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05)*.

[19] FREY, B. J., DENG, L., ACERO, A., AND KRISTJANSSON, T. ALGONQUIN: Iterating Laplace's Method to Remove Multiple Types of Acoustic Distortion for Robust Speech Recognition. In *7th European Conference on Speech Communication and Technology* (2001).

[20] FUCHS, C., ASCHENBRUCK, N., LEDER, F., AND MARTINI, P. Detecting voip based dos attacks at the public safety answering point. In *ASIACCS '08: Proceedings of the 2008 ACM symposium on Information, computer and communications security*.

[21] GOLLE, P. Machine learning attacks against the asirra captcha. In *CCS '08: Proceedings of the 15th ACM conference on Computer and Communications Security*.

[22] KAPRAVELOS, A., POLAKIS, I., ATHANASOPOULOS, E., IOANNIDIS, S., AND MARKATOS, E. P. D(e|i)aling with VoIP: Robust prevention of dial attacks. In *Proc. 15th European Symposium on Research in Computer Security (ESORICS 2010)*.

[23] MULLINER, C., AND MILLER, C. Fuzzing the phone in your phone. In *BlackHat USA 2009* (July 2009).

[24] NAUMAN, M., KHAN, S., AND ZHANG, X. Apex: extending android permission model and enforcement with user-defined runtime constraints. In *ASIACCS '10: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*.

[25] PHITHAKKITNUKOON, S., DANTU, R., AND BAATARJAV, E.-A. VoIP Security: Attacks and Solutions. *Inf. Sec. J.: A Global Perspective 17*, 3 (2008), 114–123.

[26] ROSENBERG, J., SCHULZRINNE, H., CAMARILLO, G., JOHNSTON, A., PETERSON, J., SPARKS, R., HANDLEY, M., AND SCHOOLER, E. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853, 4320, 4916.

[27] TAM, J., SIMSA, J., HUGGINS-DAINES, D., VON AHN, L., AND BLUM, M. Improving Audio CAPTCHAs. In *Symposium On Usable Privacy and Security (SOUPS) 2008*.

[28] TRAYNOR, P., ENCK, W., MCDANIEL, P., AND PORTA, T. L. Mitigating Attacks on Open Functionality in SMS-Capable Cellular Networks. *12th Annual International Conference on Mobile Computing and Networking* (2006).

[29] TRAYNOR, P., MCDANIEL, P., AND PORTA, T. L. On attack causality in internet-connected cellular networks. In *In USENIX Security Symposium (SECURITY)* (2007).

[30] VON AHN, L., BLUM, M., HOPPER, N. J., AND LANGFORD, J. *CAPTCHA: Using Hard AI Problems for Security*. Lecture Notes in Computer Science.

[31] WANG, X., CHEN, S., AND JAJODIA, S. Tracking anonymous peer-to-peer VoIP calls on the internet. In *CCS '05: Proceedings of the 12th ACM conference on Computer and Communications Security*.

[32] WRIGHT, C. V., BALLARD, L., MONROSE, F., AND MASSON, G. M. Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *SS'07: Proceedings of the 16th USENIX Security Symposium*.

[33] YAN, J., AND EL AHMAD, A. S. A low-cost attack on a microsoft captcha. In *CCS '08: Proceedings of the 15th ACM conference on Computer and Communications Security*.

[34] ZHANG, R., WANG, X., YANG, X., AND JIANG, X. Billing attacks on sip-based voip systems. In *WOOT '07: Proceedings of the first USENIX workshop on Offensive Technologies*.

Outsourcing Malicious Infrastructure to the Cloud

Georgios Kontaxis, Iasonas Polakis, Sotiris Ioannidis
Institute of Computer Science
Foundation for Research and Technology Hellas
{*kondax, polakis, sotiris*}*@ics.forth.gr*

***Abstract*—Malicious activities, such as running botnets, phishing sites or keyloggers, require an underlying infrastructure for carrying out vital operations like hosting coordination mechanisms or storing stolen information. In the past, attackers have used their own resources or compromised machines.**

In this paper, we discuss the emerging practice of attackers outsourcing their malicious infrastructure to the Cloud. We present our findings from the study of the first major keylogger that has employed Pastebin for storing stolen information. Furthermore, we outline the traits and features of Cloud services in facilitating malicious activities. Finally, we discuss how the nature of the Cloud may shape future security monitoring and enhance defenses against such practices.

I. Introduction

Malicious activities, such as running botnets, phishing sites or keyloggers, require an underlying infrastructure for carrying out vital operations like hosting coordination mechanisms or storing collected information. In the past, attackers have used their own resources or compromised machines to store stolen user information (from keyloggers or phishing schemes) or coordinate their activities (issue new orders, push updates, etc) as in the case of botnets. Both cases entail disadvantages for the attackers. In the first case, their actions can be traced back to them. In the second case, the infrastructure is not reliable as compromised servers can be identified and patched.

In this paper, we highlight the emerging practice of attackers outsourcing their malicious infrastructure to the Cloud. In other words, we discuss a new trend on the Internet where attackers switch from IRC channels to Twitter accounts for coordinating their botnets and from private FTP sites to public user-content hosting sites such as Pastebin. We present the findings from our study of the first major keylogger that has employed Pastebin to upload stolen information. Furthermore, we analyze the nature of Cloud services in respect to the needs of attackers. Finally, we discuss how such shift towards Cloud-based infrastructure may shape future security monitoring and defense mechanisms.

The contributions of this paper are the following:

- We present a study on the first major case of a keylogger using Pastebin to upload its stolen information. To the best of our knowledge, this is the first study concerning the use of public content-sharing sites as malicious dropzones.
- We discuss the nature of Pastebin-like sites and present scenarios where their features can be employed by malware to coordinate their actions and store information. We also provide proof that attackers are already discussing and developing techniques based on the, sometimes unique, features of such Cloud services.
- We discuss the future of Pastebin-enabled keyloggers, along with new ways that attackers can take advantage of the Pastebin service to enhance other nefarious activities, such as botnet coordination.
- We propose Cloud-oriented security methods for detecting and monitoring this new generation of malware.

II. Related

Security researchers at Kasperky Labs [1] report that there is an increasing trend for botnets to move their command and control channels away from IRC and into the Web. They claim that the number of HTTP-based coordination channels outnumbers the IRC servers by a factor of 10 to 1, as a result of monitoring and aggressive action against the latter.

Authors in [2] acknowledge and study the trend of botnet command and control services evolving from traditional IRC-based approaches to the Web and, specifically, to social-networking sites such as Twitter. Furthermore, they point out the irony of attackers hiding their infrastructure in plain sight in popular Web sites so as to improve the unobservability of their operations.

Symantec in [3] and also security researchers in [4] discuss the nature of a new type of malware that employs Twitter as part of its command and control infrastructure. In detail, the malware accesses the public timeline of messages (tweets) of a specific Twitter account that contains base64-encoded strings. These strings contain URLs to pastebin-like sites that in turn carry DLL and executable files also in base64-encoded form.

A technical report by Balatzar et al. [5] exposes the Web 2.0 orientation of a new generation of malware with social network propagation components and an infrastructure, from command and control services to malware repositories and storage of stolen information, entirely over HTTP applications.

In 2009, 10K Hotmail passwords were leaked [6] to the Web via public uploads to Pastebin.com. In 2010, 10K credit card numbers were also leaked [7] on the same service.

In 2010, the use of Pastebin and similar sites has been suggested [8] as a way for criminals to anonymously store, exchange or even advertise samples of stolen information, with little or no risk of liability.

Holz et al. in [9] perform dynamic analysis of malicious software (e.g. keyloggers) in an automated fashion. Their goal is to discover online repositories that each malware uses to upload stolen information. Furthermore, they analyze the nature and content of the stolen data, thus providing an insight into the back-end of such underground operations. Their investigation of popular keyloggers has led them to privately-owned or compromised Internet servers that execute attacker-provided scripts (e.g. PHP) to implement the necessary functionality for storing and managing the stolen information.

Our work focuses on the emerging trend of keyloggers and other malicious software employing inherent functionality of public Cloud services, such as user-content-hosting.

III. Background

In this section, we outline the generic behavior of key-logging malware (keyloggers) and also introduce the basic features of the Pastebin service.

A. Keyloggers

A keylogger is a piece of computer code, usually packaged as a stealth program, that records all keys struck on a keyboard. It falls in the category of malicious software as it is often employed for stealing sensitive user information, such as passwords or financial information, as the owner types them (e.g., while logging in a service or making an online purchase). Keyloggers may simply record all key strokes or be more sophisticated and capture keyboard activity right after a predefined sequence of keystrokes (e.g., after the user types mybank.com, presumably in the Web browser's address bar). Subsequently, keyloggers place the captured information in a log file and upload it to an online location (or dropzone) accessible by the attacker. Sophisticated keyloggers present data in a structured form, perhaps identifying URLs ("http" or "www" strings) and grouping keystrokes accordingly. Simpler tools provide access to a chaotic stream of keystrokes for the attacker to extract useful information from.

B. Pastebin

Pastebin sites were originally conceived as clipboard-like collaboration places where developers could conveniently share source code, logs and other text-based content without having to worry too much about the original formatting getting corrupted or other problems associated with trying to share structured text over e-mail or instant messaging applications. A registered account is not required. One may use the service to upload (or paste) arbitrary blocks of text online and receive a URL (pointer) to that content. He may then share that URL with their colleagues or friends or keep it private for personal use. Returned URLs are of the form http://pastebin.com/⟨ID⟩, where the ID appears to be random, perhaps the product of a hash function. The lifetime of such content is user-defined and may vary between 10 minutes to 1 hour, 1 day, 1 month or for ever. Due to the service's orientation towards developer, it offers syntax highlighting. By default, pasted content offers no syntax highlighting, never expires, is public and carries no identifying title or username (anonymous). All public pastes appear in a "recent posts" timeline and can therefore be accessed by anyone. Crawling the keyspace of Pastebin IDs is not an option as the space is quite large (7-8 characters long, [A-Za-z0-9]) and randomly populated. Finally, an interesting feature of Pastebin is the support for arbitrary subdomains. Any guest of the service may type http://⟨anything⟩.pastebin.com and will be presented with a valid view of the service. Any pastes created under that arbitrary subdomain are not referenced by http://pastebin.com/⟨ID⟩ but by http://⟨anything⟩.pastebin.com/⟨ID⟩ and for that matter do not appear in the public timeline. To access such content, one requires both the random ID and the domain prefix.

IV. The Pastebin Incident

In this section we present, to the best of our knowledge, the first major case of a keylogger using Pastebin (or any other Cloud service for that matter) to upload its stolen information. We introduce the timeline of events, provide technical details regarding our efforts to capture and study the incident and, finally, share the outcome of our analysis regarding the nature of the malware.

In May 2010 a large number of entries, containing raw streams of what appeared to be keystrokes, began appearing on Pastebin. The presence of lists of usernames and passwords is not new to the service, but this specific type of entries quickly became a trend while constantly increasing in volume and ended up dominating the content being uploaded to the service during that period of time.

Pastes that fell in this category carried the same set of characteristics: anonymous entries, with no syntax highlighting or specific internal structure, comprised of "[ ]" blocks carrying what appeared to be titles of Web browser windows (e.g., "Internet Explorer - Facebook.com" or "Mozilla Firefox - Hotmail.com"), followed by a stream of keystrokes. Their frequency was so high that they dominated, almost completely, the "recent posts" list in the service's homepage.

The overwhelming volume of such pastes caught our attention and, at the same time, emerged as an issue in security-related blogs [10], [11]. Their sudden appearance and sudden increase in volume, indicated the launching of a new keylogging tool that employed Pastebin to upload the stolen information. We started to monitor this incident and analyze its characteristics as it appeared to be the first

of its kind. The keylogging tool was later identified by BitDefender [1] as *Trojan.Keylogger.PBin.A*.

A. Dataset Collection

In this section we outline our collection methodology of keylogger data (or pastes) on Pastebin. Moreover, we elaborate on the completeness of our collected trace and discuss certain limitations of our approach.

Active Crawling. A straightforward crawl of all uploads on Pastebin was not an option, so we resolved to a more elegant solution. Pastebin uploads are assigned a random-looking ID of the form http://pastebin.com/[0-9A-Za-z]+. Since the ID distribution appears to be random, a simple iteration over the available namespace would be inefficient in terms of time. However, by default the last 8 pastes created appear in a "recent posts" timeline on the homepage of Pastebin. We periodically downloaded the list, parsed its entries and fetched any pastes that we could attribute to the output characteristics of the keylogger. By examining, fast enough, the "recent posts" timeline, we were able to gather all or almost all keylogger pastes as soon as they were created.

We developed an infrastructure that periodically downloaded the "recent posts" timeline, examined it for entries matching our search criteria and downloaded any pastes that were determined to be a match. In detail, we downloaded the homepage of Pastebin and parsed the entries present in the recent timeline. Entries in that list contain the username or title of the paste, along with a label describing its content type (later used for syntax highlighting, e.g. C, HTML, PHP). Furthermore, all entries are active hyperlinks leading to the page of each paste. The pastes containing keylogger output were created using default parameters, i.e., by an anonymous user and without a type specification. Our infrastructure followed the hyperlinks for pastes matching the default parameters, downloaded the respective pages and performed regular expression string matching to identify structural properties of the keylogger pastes. As mentioned earlier, each keylogger paste had its streams of keys grouped around lines of the format "[<browser> - <window title>]" (e.g., "Internet Explorer - Facebook.com"). If the downloaded pastes exhibited this kind of structure, they were considered valid matches and kept for further analysis. Otherwise, they were discarded.

To secure the completeness of our collection methodology we took steps to ensure that our polling rate (periodic download of the timeline) was fast enough to capture all interesting pastes before they disappeared from the timeline (during period of increased activity) and also not to frequent so as to avoid stressing the service when not necessary. In that light, each time we downloaded the timeline, we calculated the dice coefficient of its entries with the one downloaded previously. A dice coefficient above 0.75 meant that 75% of the entries in the timeline had not changed since the previous time we had downloaded the list; we considered this as an indication that we were polling the service too fast in a period where few uploads were made per second. In that case, we slowed down our polling rate. On the other hand, a dice coefficient of less than 0.25 meant that only 25% of the entries in the timeline remained the same, which was fine, meaning we had not missed any entries between our last poll and the current one. But if a sudden burst of activity occurred we could miss some entries. To avoid such a case, we increased our polling rate. Overall, the average value of the coefficient was 0.80, indicating a sufficient polling rate, able to provide a complete keylogger data trace.

Time-machine Crawling. Our previous crawling method allowed an efficient gathering of all subsequent pastes. However, we also wanted to check if similar pastes had been uploaded in the past, how long ago and plot their volume as a function of time. For that matter, we employed the advanced tools provided by the Google search engine. In detail, Google Search allows one to search for a keyword within a specific domain and also limit the query scope using time constraints. To perform our backwards search we placed a query similar to "site:pastebin.com <search heuristic>" and limited the scope to one month at a time.

B. Limitations

As mentioned earlier, by default all pastes are created as public and appear on the "recent posts" timeline. By examining that timeline with a fast-enough rate, one may compile a complete or near-complete set of the public pastes created. Here we describe two cases where such an approach may not be that effective.

A paste may be explicitly set to being private and therefore will never appear in the public timeline. Considering that all pastes, public or private, are assigned random-looking IDs, there is no way for a third party to discover that paste other than guessing character sequences from a very wide namespace.

Furthermore, the use of arbitrary subdomains also allows the creation of pastes that do not appear on the service's homepage. Posts on those subdomains may still be public but one would have to know the specific prefix before pastebin.com to access their respective timeline.

We believe that, in this case study, such limitations did not apply. However, as we discuss in Section V, future strings of keyloggers could employ such techniques to hide their presence from the public.

C. Analysis

Data Volume. Using our active (or forward) crawling technique, we gathered an almost-complete set of all new pastes that matched our heuristics regarding the content and structure of the keylogger pastes. Figure 1 plots the volume

[1] http://www.bitdefender.com/

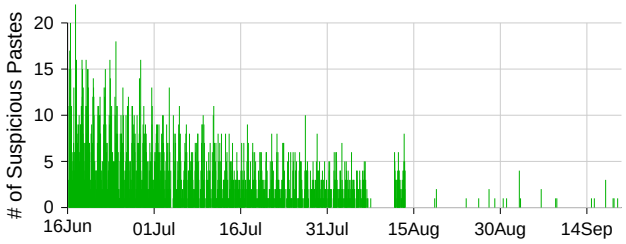


Figure 1: Volume of PBin.A stolen information on Pastebin, per hour. (Active Crawling)

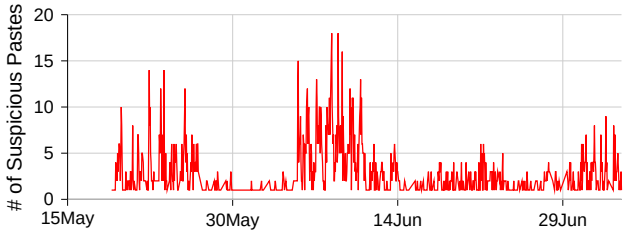


Figure 2: Indexed Volume of PBin.A stolen information on Pastebin, per hour. (Google Search)

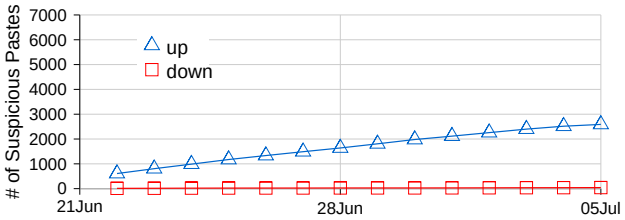


Figure 3: Cumulative Volume of available and taken-down PBin.A entries of stolen information on Pastebin, during the middle of the activity period.

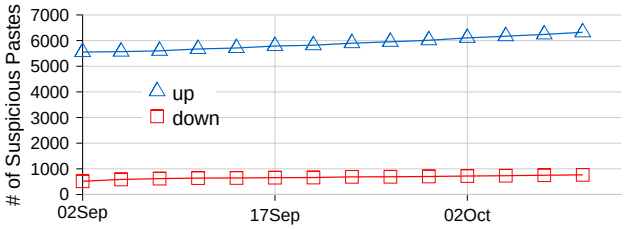


Figure 4: Cumulative plot of available and taken-down PBin.A entries of stolen information Pastebin, after the malware had almost ceased its operation.

of new pastes over a period of 4 months. A closer look reveals diurnal patterns, an expected phenomenon attributed to daylight cycles and user behavior which has been extensively documented in [12]. Moreover, there is a steady decrease in volume, indicating that the shrinking presence of this keylogger in infected computers.

Furthermore, we employed our backwards crawling technique and, using Google Search, calculated the volume per day up to 6 months before the time we started the active crawling. While we received search results for the entire search period, pastes matching our heuristics began appearing in May 2010, indicating that the appearance of keylogger data on Pastebin was a new trend and we had caught it almost as soon as it started. Figure 2 plots the volume of pastes over time. There are no pastes before May 19. Let there be noted that this figure presents the Google-search indexed volume of Pastebin entries which is essentially a sampled dataset. On the other hand, Figure 1 visualizes the almost complete set of pastes during our period of active crawling.

Takedown Rate. During the first few days since keylogger data started appearing in large volume on Pastebin, the service claimed [11] that efforts were being made to identify those entries and remove them. To verify that claim, we periodically checked the availability of pastes collected by our crawler. Figure 3 plots the cumulative volume of available pastes over the period of our active crawling, using a one day granularity. Since pastes were created using default options and were set to never expire, we can attribute any unavailability to takedown efforts. However, one may see that efforts were not sufficient in identifying existing data or keep up with the rate of new data being uploaded. Moreover, in September 2010, almost a month after the data uploading activities of the keylogger had nearly ceased, the stolen information still remained online and, apparently, takedown efforts had completely ceased. (Figure 4).

D. Strings Identified

Although our analysis heuristics were strict, ensuring that the set of pastes were indeed part of the keylogger output, our crawling heuristics were pretty relaxed, resulting in more than one different strings of keyloggers being identified by their output. Besides the primary keylogger string being analyzed here, the rest were also interesting in terms of data. One batch of entries contained a URL and clearly marked “username” and “password” fields indicating that the keylogger was context-aware and was able to extract only the necessary information. Also, some other batches were site-exclusive and contained usernames and passwords, each one for a specific site indicating the presence of very particular keyloggers.

E. Information Gathered

We closely examined the stolen information being uploaded to Pastebin by the keylogger. Figure 5 presents a screenshot of an example entry. As one may notice, while the content is structured to group recorded keys by the application window in which they were recorded, the malware does not attempt to identify or organize the stolen

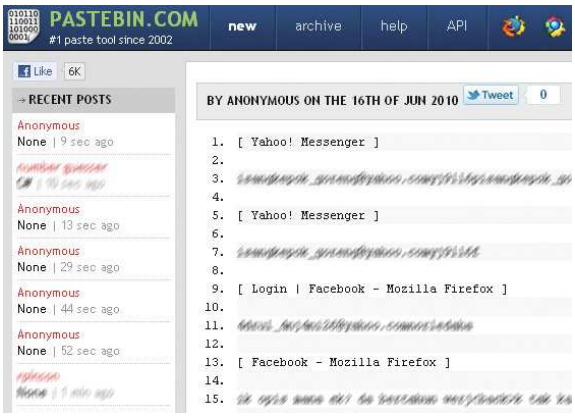


Figure 5: Screenshot of a Pastebin entry containing stolen information by Keylogger PBin.A.

information in any way. As a result, the uploaded entries were a chaotic stream of recorded keyboard keys. We noticed that most keys were recorded twice, indicating perhaps a bug in the keylogger's implementation or an excessive polling rate for keyboard events, such that it recorded almost each event twice. The next question was whether the keylogger uploaded stolen information in fixed-size batches. Thus, we studied the size distribution of the entries the malware uploaded on Pastebin. Figure 6 demonstrates that the malware does not operate on a fixed-number-of-bytes basis but rather employs a time frame after which any stolen information is uploaded. Let it be noted that the average size of an entry was 6 KBytes and the median was 1 KByte.

The next step was to attempt the reconstruction of the data using the following automated methodology: we treated each entry (paste) as standalone and grouped the streams of keys by their respective title in the application window, contained inside clearly marked blocks "[ ]". This resulted in a concatenated stream of keys per application window, for every uploaded entry. We removed duplicate keys, i.e. identical characters adjacent to each other but only in the cases where such combinations did not exist in the English language. We removed special keys such as <BACK> (backspace). We tokenized the stream using spaces as a delimiter and looked up the resulting words in a dictionary. For words not found, we looked them up in Google search and leveraged the "did you mean" or "search instead" feature of the search engine to resolve them. Finally, we ended up with a pretty much readable transcript of the user's input. Overall, we extracted hundreds of URLs, usernames, passwords, e-mail addresses and Instant Messenger conversations.

F. Correlation with Traditional Keyloggers

A subset of the stolen information entries contained the IP addresses of the respective infected computers. To investigate a possible correlation with hosts infected by traditional

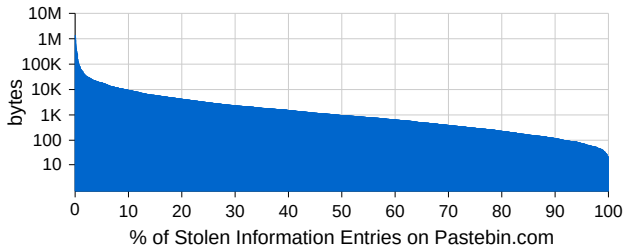


Figure 6: Distribution of stolen-data batches in terms of size.

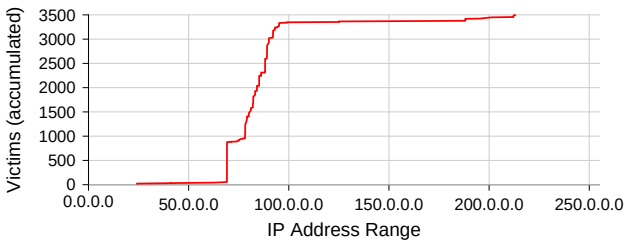


Figure 7: Cumulative distribution of keylogger victims across the IP address space.

keyloggers, we plotted (Figure 7) the cumulative distribution of victims across the IP address space. Interestingly, the most dense IP address range matches similar ranges found also by Holz et al. [9] when studying traditional keylogger data, which also match ranges of host computers known to be infected with malware such as the ZeuS botnet.

V. Outsourcing to the Cloud

In this section we discuss the trend of outsourcing malicious infrastructure to the Cloud. We present how attackers may benefit from the nature of the Cloud and detail the use of certain traits and features that may contribute to more sophisticated types of malware.

A. Economics

Public Cloud services are very cheap or free to use. One could consider examples from social networks such as Twitter, or services like Pastebin and Rapidshare. Traditionally, attackers had to maintain their own dedicated hardware and network connectivity.

B. Reliability

Cloud services aim to provide reliable uptime service for at least 99% of the time. Traditionally, attackers use compromised computers of victims which can be shut down or cleaned from infections. If that happens, the attacker will lose all stored information (e.g. keylogger data) and may also lose the command and control point which coordinates the activities.

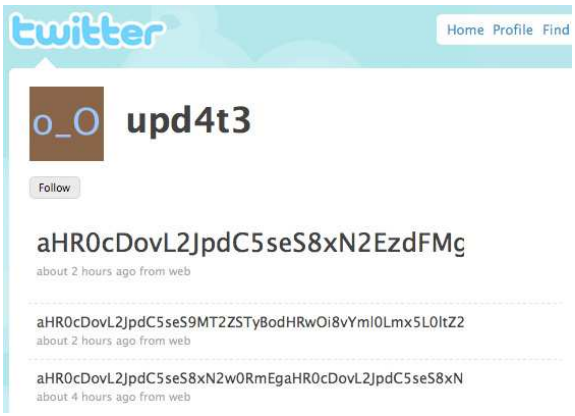


Figure 8: Twitter profile being used to command and control (C&C), pushing BASE64-encoded information.

C. Scalability

Cloud services are designed to scale in terms of storage, processing power or bandwidth. A private FTP site on a compromised server may run out of space, while an infected host acting as a command and control point may be overwhelmed by the number of network connections.

D. Unobservability

Cloud services offer practical anonymity. Traditionally, stolen information is transmitted directly back to the attacker or to a compromised system under his control. At the same time, malware in need of coordination (e.g. botnets) contacts an attacker-provided exchange point. An attacker employing his own resources to support the back-end infrastructure can be traced, identified and prosecuted. There is always the option of using a random compromised PC for that purpose but, as mentioned earlier, this may sacrifice reliability. On the other hand, infected victims employing the Cloud for malicious activities, does not differ from a large population of users using the Cloud for benign purposes. For instance, bots uploading stolen information on Pastebin blend in with a plethora of users sharing source code or arbitrary text on a daily basis. Moreover, in the Cloud it is much harder for a prosecuting authority to retrieve user records from an international service.

E. Plausible Deniability

An attacker contacting a compromised system or a private dropzone can be traced and charged with malicious activity. As no legitimate user will ever contact an infected workstation on a random TCP port bound by a backdoor, anyone who connects is probably malicious and his IP address can be used to identify him. For instance, a discovered dropzone employed by a keylogger may be kept online by security researchers to find out who will come and collect the stolen information. When this happens, it will be very hard for the attacker to deny his actions. On the other hand, an attacker using the Cloud to download stolen information does not differ from a plethora of other users using the Cloud for benign purposes. Moreover, he is able to create enough noise so that he cannot be tied beyond any doubt with malicious activity. For instance, one may visit all new entries on Pastebin, keep the ones uploaded by his keylogger and discard all the rest. Such activity will not be any different from, for example, our crawler presented in the previous section.

F. Unique Features and Flexibility

The Cloud offers a certain amount of flexibility in the use of the services it provides and a series of unique features that would otherwise be unavailable to the attacker. Here we discuss more sophisticated methods for storing keylogger data on Pastebin, along with other ways that the service's features can be leveraged for serving nefarious purposes, such as supporting botnets.

Sophisticated Keylogger Pastes. During our work on this paper, we have discovered discussions in underground forums of the security community, dating back to the beginning of 2010, exploring sophisticated ways of placing keylogger data on Pastebin and similar services. For instance, in a well known underground hacking forum, there is a discussion about using the arbitrary subdomain feature of Pastebin to create highly-dynamic private areas for uploading data. As mentioned earlier, any URL of the form http://foo.pastebin.com will produce a valid service page and any data uploaded through that page will not appear under the public home page of the service but under the new subdomain. Therefore one has to know "foo" in order to access that area and parse the "recent posts" list. The forum discussion suggests this technique as a way of uploading keylogger data without being noticed, contrary to the incident presented in this paper. Furthermore, they also suggest that this private subdomain should change frequently to avoid detection or prevent access to the entire set of uploaded data if the private URL is discovered. Their approach for the keylogger to discover the private subdomain, prior to uploading its data, is to poll an external Web page, that will act as a coordination point. However, one could implement a function that generates a subdomain string, using the current time and day or week or month, similar to traditional domain flux techniques employed by botnets [13]. This way no coordination point is required and the attacker remains in sync with the current Pastebin private area at all times.

Private C&C Pastes. In the same underground forum, one may read on employing Pastebin as a coordination point (Command and Control or C&C) for traditional botnets. The idea is to hide the address of the coordination point or change it continuously so to minimize damage if its current location is discovered. Furthermore, any pastes created will

```
Function UploadtoPastBin(ByVal text As String)
  Dim wc As New Net.WebClient
  Dim pl As String = "paste_code=" & text
  Return System.Text.Encoding.ASCII.GetString(
   wc.UploadData("http://pastebin.com/api_public.php",
    "POST", System.Text.Encoding.ASCII.GetBytes(pl)))
End Function
```

Listing 1: VB.NET Source Code for uploading to Pastebin.com Source

have a very small lifetime (minutes or hours) after which they will expire and be removed from the service. In detail, the discussion is about employing domain generation algorithms and applying their output to form private Pastebin subdomains. As mentioned in various posts, the creation of such a subdomain is instant, while traditional domain registration is expensive in terms of time, effort and money.

Overall, we believe that Pastebin is a very flexible service and can be leveraged for various malicious actions. As a matter of fact, in the same underground forum, Listing 1 was found in a Hacking forum titled "Post to PasteBin.com Source (Good for keyloggers)", a fact which indicates that attackers are actively developing and sharing modules of code to be used in malware.

VI. Discussion

Here we discuss how the security community can respond to this new generation of Cloud-supported malicious software. In other words, the public nature of the Cloud may present opportunities for security researchers to develop novel methods for identifying malware that employ such practices.

In general, free Cloud services are public. So far, we have discussed how attackers are able to, essentially, upload information to the Cloud in a practically anonymous fashion, and later access it to facilitate information exchange. The public nature of such services results in their content being indexed by search engines and cached. Therefore, it is easy for anyone, besides the attackers themselves, to locate and access information even if it has been removed from the original site, e.g. deleted by the attackers or taken down by the site's administrators. Moreover, malware, for instance botnets, usually targets a large number of victims. As a result, their large-volume output on the Cloud will be detected in a site like Twitter or Pastebin.

Traditional security practices inspect the *input of malware* to identify it and take measures against it. Such input includes inbound network behavior (e.g. network scanning or packet signatures) or malicious executables on PCs. However, lately a large portion of attacks has moved inside the user's Web browser. As a result, it is invisible at the network level, bypasses firewalls and intrusion detection systems. We propose the use of *malware output* in the Cloud (e.g. Twitter, Pastebin, etc.) as a heuristic for detecting and tracking new and emerging threats. For instance, we could monitor Twitter for a new botnet's command and control messages, and keylogger data or general information leakage on Pastebin.

Anomaly Detection on Cloud Services. Certain services, such as Twitter and Google search, employ heuristics to prevent abuse of their resources from automated scripts. Pastebin also intends to implement similar behavior. Such practice can be extended to form anomaly detection systems that not only block certain users from accessing the service, but also produce alerts or signatures for emerging automated activities. For instance, the appearance of a large number of IP and e-mail addresses in Pastebin entries, or other popular user-content hosting sites, can be detected and the victims be warned. Another example is the inspection of the indexed portions of malicious infrastructure. In detail, one can search the Web via popular search engines for certain keywords that will reveal lists of passwords or other sensitive information and perhaps investigate their replication across multiple sites on the Web.

Global View of the Attacks - Warning System. Traditional security practices require a plethora of distributed network monitors or sensors in order to acquire an accurate view of the attacks on a global scale. By monitoring the output of malware on the Cloud, one needs only a single point of observation in order to be able to inspect instances of malicious infrastructure on the Internet.

VII. Conclusion

In this paper we have focused on an emerging practice of attackers; outsourcing the infrastructure required to support their malicious acts to the Cloud. We present an empirical study on the first major usage of a Cloud service, Pastebin, by a keylogger for storing stolen information. We evaluate this trend by analyzing the nature of Cloud services in terms of facilitating the requirements of attackers and present the benefits of the Cloud along with unique features that provide new capabilities to the attackers and increase the efficiency of past practices. Finally, we discuss how a shift towards the Cloud may shape security monitoring practices, and propose countermeasures for such malicious activities.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement 257007. This work was supported in part by the Marie Curie Actions – Reintegration Grants project PASS. We thank the anonymous reviewers for their valuable comments. Georgios Kontaxis, Iasonas Polakis and Sotiris Ioannidis are also with the University of Crete.

REFERENCES

[1] "Kaspersky labs report: Irc botnets dying... but not dead," http://threatpost.com/en_us/blogs/report-irc-botnets-dyingbut-not-dead-111610.

[2] E. J. Kartaltepe, J. A. Morales, S. Xu, and R. Sandhu, "Social network-based botnet command-and-control: emerging threats and countermeasures," in *Proceedings of the 8th international conference on Applied cryptography and network security*, 2010.

[3] "Symantec - Official Blog: Twitter + Pastebin = Malware Update," http://www.symantec.com/connect/blogs/twitter-pastebin-malware-update.

[4] "Arbor Networks - Twitter-based Botnet Command Channel by Jose Nazario," http://asert.arbornetworks.com/2009/08/twitter-based-botnet-command-channel/.

[5] "The Real Face of KOOBFACE: The Largest Web 2.0 Botnet Explained," http://blog.trendmicro.com/the-real-face-of-koobface/.

[6] "10,000 Hotmail passwords mysteriously leaked to the web," http://www.theregister.co.uk/2009/10/05/hotmail_passwords_leaked/.

[7] "Mybanktracker - mastercard credit card numbers leaked by wikileaks supporters," http://www.mybanktracker.com/bank-news/2010/12/08/mastercard-credit-card-numbers/.

[8] "A Treasury of Dumps," http://blog.damballa.com/?p=695.

[9] T. Holz, M. Engelberth, and F. C. Freiling, "Learning more about the underground economy: A case-study of keyloggers and dropzones," in *ESORICS*, 2009.

[10] "Malware City Blog - Keyloggers Posting on Webpages," http://www.malwarecity.com/blog/keyloggers-posting-on-webpages-831.html.

[11] "Krebs on Security Blog - Cloud Keyloggers?" http://krebsonsecurity.com/2010/06/cloud-keyloggers/.

[12] D. Dagon, C. C. Zou, and W. Lee, "Modeling botnet propagation using time zones," in *NDSS*, 2006.

[13] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: analysis of a botnet takeover," in *Proceedings of the 16th ACM conference on Computer and communications security*, 2009.

Demarcation of Security in Authentication Protocols

Naveed Ahmed
Informatics and Mathematical Modeling
Technical University of Denmark
naah@imm.dtu.dk

Christian Damsgaard Jensen
Informatics and Mathematical Modeling
Technical University of Denmark
Christian.Jensen@imm.dtu.dk

***Abstract*—Security analysis of communication protocols is a slippery business; many "secure" protocols later turn out to be insecure. Among many, two complains are more frequent: inadequate definition of security and unstated assumptions in the security model. In our experience, one principal cause for such state of affairs is an apparent overlap of security and correctness, which may lead to many sloppy security definitions and security models.**

Although there is no inherent need to separate security and correctness requirements, practically, such separation is significant. It makes security analysis easier and enables us to define security goals with a fine granularity. We present one such separation, by introducing the notion of *binding sequence* as a security primitive. A *binding sequence*, roughly speaking, is the only required security property of an authentication protocol. All other authentication goals, the correctness requirements, can be derived from the *binding sequence*.

I. Introduction

In a cryptographic protocol, correctness is defined with respect to its functional requirements; namely, the output should follow a certain distribution (multi-party computation), some assurance on who participated in the protocol (authentication), and an assurance on who else may know the input or output of the protocol (secrecy). Security, on the other hand, is the assurance that the protocol remains correct in the presence of an adversary. At least conceptually, therefore, correctness and security are distinct requirements.

As soon as we start formulating the concrete specifications, correctness and security requirements start overlapping. After all, interpretation of adversarial behavior is with respect to the goals that a protocol achieves, i.e., whether or not an adversary can affect the correct behavior. For example, in an identification protocol, it seems natural to define security as the protocol's ability to only recognize a party A as A. On the other hand, such a protocol may not be able to do so even when there is no adversary, highlighting the non-security aspect of the definition.

We believe that demarcation of security has quite significant practical implications and is not merely a theoretical odyssey. Firstly, a new application normally has a different set of requirements, which may lead to unique security goals, at least in a naïve sense. This, however, is quite unsatisfying, and error prone [5], to reformulate security for each application separately. The job of a security analyst (human/automated tool) should be less strenuous if security requirements are fewer and pure, considering the security analysis is an undecidable problem in general [14]. Secondly, the analysis for security becomes independent from the analysis for correctness. This opens the door for adaptable security [11], because we can change security and correctness requirements quite independently, corresponding to the actual demands of an application.

To achieve separation between security and correctness, the challenge is to operationalize a seemingly simple concept —protection against adversary— in a tangible manner and independent of any non-security requirements. In a general setting, the demarcation of security appears to be quite hard. We, however, believe this may not be the case in the individual classes of protocols, as, in this paper, we present some promising results for (entity) authentication protocols.

Authentication is the basis of secure communication. Even a perfect encryption fails to provide privacy, due to notorious man-in-middle attacks. Besides secure communication, authentication on its own, without any subsequent communication, is also quite useful; the domain of RFID (Radio Frequency Identification) is a prime example.

The intuitive idea for the aforesaid separation is simple: define "appropriate" intermediate properties that are independent of protocol goals; and then, more importantly, only using these intermediate properties define the correctness (protocol goals). These intermediate properties are regarded as security properties. Clearly, if these properties can be validated then the correctness analysis essentially reduces to a non-security analysis, i.e., an analysis that does not depend on any adversary.

We are not the first ones to exploit this idea and define such separation for authentication protocols. We, however, do claim that the intermediate security property that is proposed here—the *binding sequence*—is the most suitable candidate for authentication protocols. We come back to this point after briefly going through the related work in § II.

Rest of the paper is arranged as follows. In § III, we refine the above idea of demarcation. In § IV, the *binding sequence* is introduced. In § V, the link between the *binding sequence* and authentication goals is established. The approach is exemplified in § VII, and § VIII presents the conclusion. Throughout the exposition, a common distinction between

conceptual and operational definitions should be observed[1].

II. Related Work

In the context of entity authentication, the subject matter came to light when formal methods, especially automated tools, find their way in the analysis of cryptographic protocols. This is because there is no agreed way of specifying abstract concepts of authentication, such as the belief of an entity that its peer entity is currently there and/or is willing to communicate.

In Lowe's work [6], authentication is achieved when an entity commits to a session with a peer entity if the peer entity has indeed taken part in the protocol run. Although this definition was sufficient for refuting the correctness, namely, discovering a subtle flaw in a Needham Shcroeder's protocol (after 17 years), it is just a security requirement without any formal link to the correctness. A null-protocol is also secure in this model, which, obviously, does not achieve any authentication goal.

The first explicit account on this subject is due to Roscoe [4], who coined the terms *intensional* and *extensional* specifications. An *intensional* specification is specific to a protocol, and is usually in form of assertions about the protocol messages and states. An *extensional* specification is independent of the details of a protocol and instead defines what this protocol is suppose to achieve, from the perspective of an external observer. In our context, an *intensional* specification corresponds to the security, while an *extensional* specification, roughly, corresponds to the correctness. Roscoe recommends *intensional* specifications to be the basis of security analysis, which, however, is an error-prone approach if one does not know how the correctness requirements are related to the *intensional* specifications.

In the area of provable security, the most notable notion, in this regard, is *matching conversation* [2], which essentially means that a protocol is secure if the runs of peer entities match. Clearly this definition is independent of what protocol achieves; we may even prove security for a protocol that is incorrect (with respect to its intended functionality). Therefore, protocol goals need to be defined over the protocol transcript, e.g., if encryption of a nonce is received we might consider it achieving the operativeness of the peer entity (§ V); but, a concrete method for doing so is not available in the art.

The above mentioned separation is also too strong. Firstly, it requires 100% matching over the transcripts of a protocol. Secondly, it is not clear whether or not such matching is really the right criteria [8]. It is in fact trivial to construct a protocol that is correct in a natural sense but fails to guarantee the *matching conversation*.

[1]Generally, a conceptual definition is in an abstract form capturing the natural use of a concept, while an operational definition is an algorithm providing the observability of the concept, e.g., is the concept of confidentiality, which is operationalized in the definition of indistinguishability.

Many authors, including Gollmann [7], Boyd and Mathuria [8], criticize the style of specifications that does not consider correctness requirements, namely what a protocol achieves in terms of authentication goals. We believe these observations are due to a missing link between the security (e.g., intensional specifications) and the correctness (authentication goals). Our proposal also settles this concern.

Lowe's hierarchy of authentication goals [5] is close to our work, in spirit. He formalized various forms of correctness requirements for authentication. In comparison to our work, there are two shortcomings: Lowe's definitions for authentication are not fine grained (cf. § V); and each Lowe's definition requires its own security analysis. In our case, security analysis is aimed at verifying one fundamental primitive, the *binding sequence*, and the correctness analysis is strictly a non-security analysis.

At last, we consider the positions of two recent formal methods of security analysis: LYSA calculus [12], which is based on static analysis[2]; and OFMC [13], which is based on model checking. Although LYSA and OFMC utilize different approaches, the security guarantees they provide are about the same security primitives: authenticity and confidentiality of individual messages. For entity authentication protocols, the process between these security guarantees and correctness is essentially left to the designers.

The work that we present in this paper is not entirely fresh; some of the ideas have already appeared in our previous work, although informally at some places [10] and incomplete at others [11]. Now, we aim at providing a complete picture of a *binding sequence* based security framework, while also improving on some previous results [9].

By no means our account for the related work is complete, due to the space constrain. Nevertheless, our proposal is distinctive, at least, in two aspects: we provide a rigorous link between the correctness and the security; and our requirement for the security, the binding sequence, is a considerably weaker requirement than corresponding security requirements in the related work.

III. Background

Let Π be an authentication protocol, α be an adversarial model, G be a Π's authentication requirement and Θ be a background theory. Intuitively, Θ stands for anything that is required for security analysis, e.g., environment model and setup assumptions (such as long term keys). An abstract security argument is in the following form.

$$\Theta, \Pi, \alpha \Rightarrow G \tag{1}$$

Here, the implication "$\Rightarrow$" stands for a security analysis. As we know, security analysis can be of different types: it may based on, e.g., complexity theory [2], modal logic [1], static analysis [12], model checking [13] or simply an

[2]LYSA calculus is similar to Spi calculus; see the reference for details.

informal analysis. In any case, Θ ,Π , α, $\Rightarrow$ and G are all fundamental parts of each meaningful security argument.

For our purpose, Eqn. 1 represents the security analysis as a process in which a security analyst tries to show that under a given security model, $\{\Theta, \alpha\}$, an authentication goal G can be achieved by running an instance of Π. Now, assuming G is not a pure security requirement, we can decompose Eqn. 1 in the following two arguments[3].

$$\Theta, \Pi, \beta \models G \tag{2}$$

$$\Theta, \Pi, \alpha \models \beta \tag{3}$$

In the above arguments, a new notion β is introduced, which is an intermediate property. If this property is valid then it suffices to analyze whether or not the original requirement G can be met in Π under a given security model. In fact, in the related work, we have already seen several constructions of β: Bellare's matching conversation; Roscoe's extensional specification; and confidentiality and authentication of messages in LYSA and OFMC. In rest of the paper, we interpret β as a *binding sequence*, whose construction is in the next section. Also, we refer to Eqn. 2 and Eqn. 3 as β-argument and α-argument respectively.

As the reader may have noticed, β-argument and α-argument define demarcation of security: the validation process of β-argument is independent of adversary and that of α-argument is independent of authentication requirements. Therefore, β-argument establishes the correctness, while α-argument guarantees the security of Π. In § V, we describe the validation process of β-argument, thus, formally establishing an (often missing) link between security and correctness. In § VI, we discuss the validation process of α-argument.

IV. Binding Sequence

Let X_c be a network entity with the identity c. A *binding sequence* is conceptually defined as follows.

Definition 1. *A binding sequence β_{X_c} is a list of selected messages from the protocol transcript of an entity X_c, such that any change (deletion, insertion and modification) in β_{X_c} generates an efficiently detectable event on X_c.*

Intuitively, a *binding sequence* is a list of selected messages that preserves its integrity. Of course, β_{X_c} can be replayed; only an unauthorized change in the list generates a detectable event. Note that the integrity of β_{X_c} is different from the integrity of the individual messages in it. A *binding sequence* can be constructed from completely unauthenticated messages, but, when these messages are part of a *binding sequence* they preserve their integrity, e.g., a random number on a communication channel cannot be an authentic message, but as a part of some challenge-response protocol, such as the first nonce in MAP1 [2], it can be considered an authentic message at the end of the protocol *run*.

For operational definition of β_{X_c}, first we consider a generic form of n-party m-round authentication protocol Π.

for k in $\{1, ..., m\}$ **loop**
  k. $R^k_{sender} \longrightarrow R^k_{receiver}$: M_k
end loop

Here, R^k_{sender} and $R^k_{receiver}$ represents two *roles*, out of n possible *roles*, who interact by sending and receiving an arbitrary message M_k in kth round. In fact, this style of specification is called *protocol narrations*, which commonly appears in the literature [8]. This specification, however, leaves out some of the crucial details required to analyze a protocol. Therefore, more than often, we also need to specify the execution model of Π, such as the extended protocol narrations [12]. An execution model specifies how an entity executes a particular role in an instance of Π.

A natural requirement in such execution models is to specify *grant* assertions; kth *grant* assertion corresponds to M_k and is encoded on the receiver of M_k, and M_k is accepted as a valid message only if the corresponding assertion succeeds. We denote these assertions by a meta notation $grant(\mathbf{T}^c, M_k)$, where $\mathbf{T}^c$ denotes the memory of X_c. For example, $grant(\mathbf{T}^c, M_k)$ may represent the correct decryption of M_k, verification of a time-stamp in M_k, verification of a signature in M_k or an always true assertion[4].

The concept of a *binding sequence* is operationalized using the *grant* assertions. The operational definition of β_{X_c} is as follows.

1. **initialize** $\chi \leftarrow \Pi_{X_c}$
   loop M_k **in** χ
2. $M'_k \overset{p}{\leftarrow} \mathcal{A}^\alpha$ **s.t.** $(grant(\mathbf{T}^c, M'_k) = true)$
3. **if** $(M'_k \neq M_k \quad \wedge \quad p \not\approx 0)$
   $\chi = \chi - M_k$
   end loop
4. $\beta_{X_c} \leftarrow \chi$

In Step-1, we initialize a set χ with the transcript of Π on X_c, denoted by Π_{X_c} . Next, we enter in a loop over χ. In Step-2, a malicious strategy $\mathcal{A}^\alpha$ computes a message M'_k with a probability p such that the computed message is a valid message for the receiving party, i.e., the corresponding *grant* assertion succeeds. Here, $\mathcal{A}^\alpha$ is an arbitrary strategy within the adversarial model α. In Step-3, if the probability p is not negligible and M'_k is not the same as the original message M_k then we remove M_k from χ. At last, the final form of χ is the binding sequence for X_c.

The probability p in the above definition is computed over a set of protocol runs with which a potential adversary

[3]Here, it is enough to accept our claim in a hypothetical sense; later, we show that this is indeed the case.

[4]For example, if an entity is expecting to receive a random challenge then the *grant* assertion is always true.

interacts. In cryptography, the size of this set is polynomial in a security parameter, and in formal methods, the size may be infinite, corresponding to unbounded number of sessions.

V. FLAGs

We refer to the most elementary authentication goals as Fine Level Authentication Goals (FLAGs). A set of FLAGs represents a possible set of correctness requirements for an authentication protocol. Two authentication protocols are functionally different for a calling routine (who may use an authentication protocol as a a service) if their sets of FLAGs are different. Of course, to achieve a certain FLAG, different protocols may employ different cryptographic techniques, e.g., public-key vs. symmetric-key ciphers, and nonces vs. time-stamps.

Next, we present a list of FLAGs; hierarchical relations between FLAGs that are valid (by definition) are shown in Fig. 1. The presented list is based on our experience; we do not claim, although we strongly believe, that the list is complete as far as the correctness of entity authentication is concerned[5].

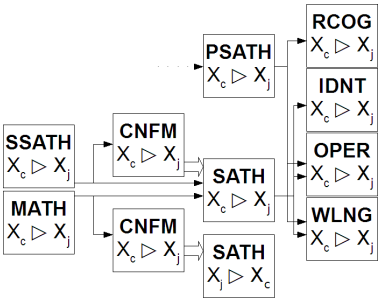


Figure 1. Relations among FLAGs

Definition 2. *Let X_c represents the local entity for which a FLAG is being defined, and X_j and X_l are two other network entities, s.t. $c \neq j$ and $c \neq l$. Let G be a variable on FLAGs. RCOG[$X_c \triangleright X_j$] $\stackrel{def}{=}$ If X_c verifies that X_j is the same entity that once existed then X_c is said to achieve the goal* recognition *for X_j.*
IDNT[$X_c \triangleright X_j$] $\stackrel{def}{=}$ If X_c verifies that X_j can be linked to a record in a pre-specified identification database then X_c is said to achieve the goal identification[6] for X_j.
OPER[$X_c \triangleright X_j$] $\stackrel{def}{=}$ If X_c verifies that X_j currently exists on the network then X_c is said to achieve the goal operativeness for X_j.
WLNG[$X_c \triangleright X_j$] $\stackrel{def}{=}$ If an entity X_c verifies that once X_j wanted to communicate to X_c then X_c is said to achieve willingness[7] for X_j.
PSATH[$X_c \triangleright X_j$] $\stackrel{def}{=}$ Pseudo single-sided authentication is achieved if an entity X_c verifies that a peer entity X_j, with a pseudonym $pid(X_j)$, is currently ready to communicate with X_c.
SATH[$X_c \triangleright X_j$] $\stackrel{def}{=}$ Single-sided authentication is achieved if an entity X_c verifies that a peer entity X_j, with the identification j, is currently ready to communicate with X_c.
CNFM[$X_c \triangleright X_j, G$] $\stackrel{def}{=}$ If an entity X_c verifies that the peer entity X_j knows that X_c has achieved a FLAG G for X_l then X_c is said to achieve the goal confirmation on G from X_j.
SSATH[$X_c \triangleright X_j$] $\stackrel{def}{=}$ Strong single-sided authentication is achieved by X_c for X_j if X_c has the confirmation on the single-sided authentication *for X_j from X_j.*
MATH[$X_c \triangleright X_j$] $\stackrel{def}{=}$ If an entity X_c verifies that both parties (X_c and the peer entity X_j) currently want to communicate with each other, then X_c is said to achieve mutual authentication.

A few remarks for the above definition are in order. *Identification*, *willingness* and *recognition* do not have any timeliness property. *Operativeness* and *willingness* do not require the knowledge of identity (or pseudo identity) of the peer entity. The goal *confirmation* can be applied to any other goal, e.g., a confirmation on MATH may be regarded as a stronger form of *mutual authentication*. *Identification* and *operativeness* are comparable to "aliveness" and "recent aliveness" [5]. Single-sided authentication is equivalent to the notion of "strong entity authentication" [8].

The FLAGs as presented above are concepts, which are independent of any security model (Θ and α) or protocol (Π). To validate the beta argument, we turn to the operational definitions of FLAGs, which provide procedures for verifying that a protocol satisfies a certain set of FLAGs.

Operationalizing a FLAG requires a bit of extra care, because, firstly, the link from a conceptual definition to its operational definition is somewhat intuitive[8], and secondly, operational definitions (procedures) depend on the interpretation of the implication in the beta argument. We choose to interpret the said implication in a complexity theoretic sense. We believe that, from these operational procedures, deriving corresponding operational procedures for other formalisms is not too hard.

In the following, if X_c can distinguish between two instances of a *binding sequence* then there exists a distinguisher $\mathcal{D}(C_b, \lambda)$ on X_c, which is a polynomial time

[5]The reader must note that we are not considering the security requirements here, e.g., forward-security, linkability or privacy.

[6]Further, if that record cannot be used to feasibly recover the identity j then it is qualified as anonymous identification. For brevity, we do not include the anonymity aspect in this exposition, but it is trivial to write the anonymous versions of FLAGs.

[7]Again, willingness have two versions: whether X_j is referring to X_c with a pseudonym or with the identity c. We, however, do not make this distinction any further.

[8]As the reader may have noted, conceptual definitions are abstract and informal, while operational definitions represent concrete procedures.

algorithm in the length of its input. Here, C_b is a challenge picked by X_c and is either C_0 or C_1; and λ is an auxiliary input, such as a decryption key. The distinguisher correctly outputs 0 or 1 corresponding to C_0 and C_1 with a high probability p_h, where $p_h = 1 - \varepsilon(|C_b| + |\lambda|)$, and $\varepsilon(.)$ is a negligible function.

- RCOG$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$ Let $\beta_{X_c}(i)$, $\beta_{X_c}(i')$ and $\beta_{X_c}(i'')$ be generated when X_c executes Π with X_j, X_l and X_j respectively, as shown in Fig. 2. Let the two challenges be $C_0 = (\beta_{X_c}(i), \beta_{X_c}(i'))$ and $C_1 = (\beta_{X_c}(i), \beta_{X_c}(i''))$. If there exists $\mathcal{D}^{rcog}(C_b, \lambda)$ on X_c for all choices of j and l then X_c is said to achieve the goal *recognition* of X_j from $\beta_{X_c}(i)$.
- IDNT$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$ Same as RCOG$(X_c \rhd X_j, \beta_{X_c}(i))$ except the distinguisher $\mathcal{D}^{idnt}(C_b, \lambda)$ gets a read-only access to an identification database containing the identification records of all network entities, as a part of its auxiliary input λ.
- OPER$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$ Let $\beta_{X_c}(i)$ and $\beta_{X_c}(i')$ be generated when X_c executes Π twice with X_j, as shown in Fig. 2. Let the two challenges be $C_0 = \beta_{X_c}(i)$ and $C_1 = \beta_{X_c}(i')$. If there exists $\mathcal{D}^{oper}(C_b, \lambda)$ on X_c for all runs with X_j then X_c is said to achieve the goal *operativeness* for X_j.
- WLNG$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$ If $\beta_{X_c}(i)$ is generated on X_c in a *run* involving X_c and X_j, as shown in Fig. 2, then IDNT$(X_j \rhd X_c, \beta_{X_j}(i)) \Rightarrow$ WLNG$(X_c \rhd X_j, \beta_{X_c}(i))$,
  where $\beta_{X_j}(i)$ consists of all those messages from $\beta_{X_c}(i)$ in which X_j is a peer entity.
- PSATH$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$ WLNG$(X_c \rhd X_j, \beta_{X_c}(i)) \wedge$ OPER$(X_c \rhd X_j, \beta_{X_c}(i)) \wedge$ RCOG$(X_c \rhd X_j, \beta_{X_c}(i))$
- SATH$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$
  WLNG$(X_c \rhd X_j, \beta_{X_c}(i)) \wedge$ OPER$(X_c \rhd X_j, \beta_{X_c}(i)) \wedge$ IDNT$(X_c \rhd X_j, \beta_{X_c}(i))$
- Let $\beta_{X_c}(i) = \beta'_{X_c}(i) || \beta''_{X_c}(i)$ ($||$ stands for concatenation). CNFM$(X_c \rhd X_j, \beta_{X_c}(i), G) \stackrel{\text{def}}{=}$ RCOG$(X_c \rhd X_j, \beta''_{X_c}(i)) \wedge$ OPER$(X_c \rhd X_j, \beta''_{X_c}(i)) \wedge$ G$(X_c \rhd X_j, \beta'_{X_c}(i))$
- SSATH$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=} G \wedge$ CNFM$(X_c \rhd X_j, \beta_{X_c}(i), G)$, where $G =$ SATH$(X_c \rhd X_j, \beta_{X_c}(i))$
- MATH$(X_c \rhd X_j, \beta_{X_c}(i)) \stackrel{\text{def}}{=}$ SATH$(X_c \rhd X_j, \beta_{X_c}(i)) \wedge$ CNFM$(X_c \rhd X_j, \beta_{X_c}(i), G)$, where $G =$ SATH$(X_j \rhd X_c, \beta_{X_j}(i))$

Next, we show that the operational definitions are equivalent ($\Leftrightarrow$) to the corresponding conceptual definitions.

Proposition 1. *The operational recognition is equivalent to the conceptual recognition.*

Proof: First we consider the forward implication: Oper. RCOG $\Rightarrow$ Con. RCOG. The main idea is to use the distinguisher to satisfy the conceptual requirement.

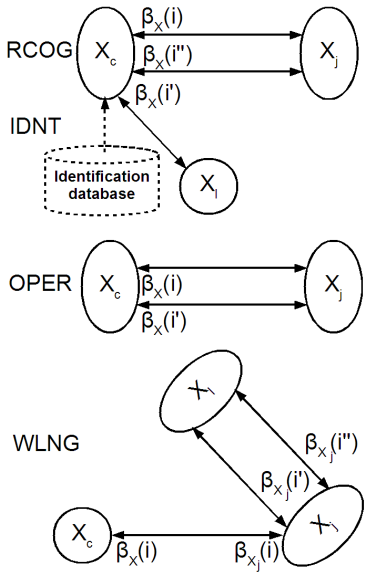


Figure 2. Distinguishability Setups for FLAGs

On X_c, create a local database: $db ::= \{id_n : id_n \in \{0,1\}^{|\beta_{X_c}|}\}$. The database db may grow to N records, where N is the total number of network entities. Initially, the database is empty: $db = \epsilon$. Each time X_c generates an instance $\beta_{X_c}(i)$ with X_j, do the following.
Go through each entry id_n in db and query the distinguisher $\mathcal{D}^{rcog}((id_n, \beta_{X_c}(i)), \lambda)$. If the distinguisher returns 1 then id_n is the pseudo identity of the peer entity, and it also shows that the peer entity once existed on the network. If the distinguisher returns 0 on all entries then $db \leftarrow db \cup \{\beta_{X_c}(i)\}$ and conclude that the peer entity never existed before.

Next, consider Oper. RCOG $\Leftarrow$ Con. RCOG. We use the contrapositive argument: if a distinguisher for RCOG does not exists then there exists an efficient attack against the conceptual recognition. Since there is no distinguisher on X_c, computationally the two pairs are equal:
$(\beta_{X_c}(i), \beta_{X_c}(i'')) \approx (\beta_{X_c}(i), \beta_{X_c}(i'))$
Therefore, $\beta_{X_c}(i'') \approx \beta_{X_c}(i')$. Hence, X_c is oblivious to the identities (or pseudo identities) of all peer entities, which is a clear violation of the conceptual definition. ■

Proposition 2. *The operational identification is equivalent to the conceptual identification.*

Proof: The proof is essentially same as in Proposition 1. In the forward implication, the database is already available in form of an identification database. Therefore, if no match is found we reject the claimed identity. For the reverse implication, an additional conclusion is that X_c does not make any effective use of the identification database. ■

Proposition 3. *The operational operativeness is equivalent*

to the conceptual operativeness.

Proof: Operational OPER $\Rightarrow$ Conceptual OPER:
Operational operativeness guarantees that each new instance of binding sequence is different from any previous instance. This is because if X_c can distinguish two instances of a binding sequence then it can use its distinguisher to distinguish p instances in $0.5p(p-1)$ steps[9], which is a polynomially bounded number. Therefore, if a *run* $\Pi(i)$ with X_j is accepted by X_c then corresponding *binding sequence* β_{X_c} must be freshly created by X_j, which implies that X_j is currently on the network.

Operational OPER $\Leftarrow$ Conceptual OPER:
We prove the corresponding contrapositive argument: if a distinguisher does not exists for OPER then there exists an efficient replay attack to violate conceptual operativeness. In the replay attack we consider two runs of a protocol at t_1 and t_2 respectively, where t_1 is the recent time and t_2 corresponds to some past instant. Since X_c cannot distinguish between the binding sequences at t_1 and t_2, replacing the *run* that is at t_1 with the old *run* that was at t_2 does not generate an event on X_c. Therefore, X_c cannot have any assurance that X_j is currently there (at t_1). Hence, X_c cannot verify that X_j currently exists. ■

Proposition 4. *The operational willingness is equivalent to the conceptual willingness.*

Proof: Operational WLNG $\Rightarrow$ Conceptual WLNG:
When X_c generates an instance of *binding sequence* $\beta_X(i)$ with X_j then operational willingness guarantees that X_j can achieve recognition (or identification) for X_c from its part of messages in $\beta_{X_c}(i)$. Therefore, the contribution of these messages by X_j implies the willingness of X_j.

Operational WLNG $\Leftarrow$ Conceptual WLNG:
Once again, we use the corresponding contrapositive argument. Clearly, if a distinguisher for recognition (or identification) does no exists on X_j then X_j is oblivious to the identity of a peer entity, as shown in Proposition 1; but, it is impossible to convey willingness without recognizing the peer entity first. Hence, X_c cannot get an assurance that X_j is willing to communicate with it.

■

We leave out the claims of equivalence for PSATH, SATH, CNFM and MATH. None of these FLAGs depends on the existence of any new distinguisher. In fact, each of these FLAGs uses the primitive FLAGs, namely RCOG, IDNT, OPER and WLNG, in a certain way. Therefore, it is trivial to construct the equivalences for these high-level FLAGs. A few more comments on these definitions are in the following.

The goal PSATH is achieved if X_c can get assurances on WLNG, OPER and RCOG in the same binding sequence. Intuitively, this means X_c has an assurance that X_j is currently ready (due to OPER) and wants to communicate with X_c (due to WLNG). The goal SATH is similar to PSATH except it depends on IDNT (identification) rather than RCOG (recognition). The operational definition of CNFM uses the fact that if X_j is engaged in a certain round of a *run* then X_j should have accepted the messages in all previous rounds. Therefore, if X_j has already achieved a FLAG then the next message from X_j to X_c guarantees the confirmation of the achieved FLAG to X_c. Similarly, the operational goals for SSATH and MATH can be interpreted in the same way.

Formally, authentication is any subset of FLAGs, e.g., bar-code (or RFID tag) based authentication of goods at a payment counter may only require IDNT, while the authentication used for on-line banking usually requires IDNT (user-name and password), OPER (user-specific challenge) and WLNG (website certificate). Therefore, the correctness of an authentication protocol corresponds to a set of FLAGs, and unless this set is derivable from the security guarantee (i.e., validity of its binding sequence), the protocol may well be functionally incorrect. The validity of a binding sequence is briefly discussed in the next section.

VI. Security Analysis

As we are able to formulate all correctness requirements of an authentication protocol using its *binding sequence*, security analysis essentially reduces to verifying the *binding sequence* itself, namely the alpha argument. Security analysis, however, is not the main focus of this paper. Here, we briefly discuss how the validation of *binding sequence* can be done in practice, with an aim of demonstrating a general procedure for doing so.

Recall from § IV the notion of *grant* assertion. The operational definition of *binding sequence* wholly depends on the grant assertion, namely, grant assertion must fail whenever any property of *binding sequence*, as per Definition 1, is violated. Since we are only considering protocols with fixed number of messages, any deletion or insertion in a *binding sequence* is easily detectable: if some message is deleted a time-out event occurs, while if a message is inserted then there is an extra message in the protocol, and the effect is same as the modification of messages.

Detection of a modified message needs some extra consideration. Since the *grant* assertions detect modified messages in a *binding sequence*, their computation cannot be public, otherwise an adversary can simply play the role of the peer entity. There has to be some secret input to create asymmetry between a legitimate entity and an adversary. A straight forward analysis is exemplified in the following, however, there might be other efficient ways to verify the validity of a *binding sequence*.

Let us consider a *binding sequence* consisting of two messages: $\beta_{X_c} = [M_1, M_2]$. There are three different ways in which this sequence can be modified: $[M_1', M_2]$, $[M_1, M_2']$

[9]The value can be calculated using arithmetic progression.

and $[M_1', M_2']$, where, e.g., M_1' represents an arbitrary modification in M_1. Now, for each of these modified sequence, we calculate an upper bound of accepting the modified sequence, namely, none of the grant assertions fails. In a valid *binding sequence* these upper bounds should represent a negligible probability. Generalizing this method results in a security analysis that involves verifying $2^{|\beta x_c|} - 1$ cases of modified sequences. Interested readers are referred to the appendixes of our technical report [11].

VII. An Example

In this section, we consider the authentication aspect of Needham-Schroeder Symmetric Key (NSSK) protocol [15][10]. We do not analyze its key establishment aspect. We only analyze it to discover a set of FLAGs assuming the binding sequences of the entities are valid. The protocol is as follows:

$$
\begin{array}{lll}
(M_1) & A \rightarrow S: & A, B, N_A \\
(M_2) & S \rightarrow A: & \{N_A, B, K_{AB}, \{K_{AB}, A\}K_{BS}\}K_{AS} \\
(M_3) & A \rightarrow B: & \{K_{AB}, A\}K_{BS} \\
(M_4) & B \rightarrow A: & \{N_B\}K_{AB} \\
(M_5) & A \rightarrow B: & \{N_B - 1\}K_{AB}
\end{array}
$$

In the protocol, A and B represent two roles that entities can take, and S is the role of trusted third-party (TTP). It is assumed that S knows the identities of all legitimate entities. Further, S shares long term keys, K_{AS} and K_{BS} with A and B respectively. The term K_{AB} denotes a session key, and when the session expires the session key is discarded.

In M_1, A sends a request to the TTP along with its own identity, the identity of peer entity and a nonce. The TTP generates a random session key from a uniform distribution and replies in M_2, which is a encrypted message with A's long-term key. The message M_2 includes K_{AB} and another encrypted message containing the same shared key but encrypted with B's long term key, which, in the next step, is sent to B. If B is able to correctly decrypt M_3 in a sense that it can recover its own identity, then B sends a nonce N_B encrypted with the session key K_{AB} in M_4. On receiving M_4, A decrypts it and sends $N_B - 1$ to B encrypted using the same session key. This completes the protocol.

The analysis for S is left out, as no binding sequence is possible for S unless some "artificial" assumptions are introduced. This is because S is an oracle and any party, legitimate or adversary alike, can query S. We assume symmetric encryption is non-malleable;Our technical report contains many practical attacks [17] if the encryption is malleable. We assume the following binding sequences are valid for A and B:
$\beta_A = [\{N_B\}K_{AB}]$,
$\beta_B = [\{K_{AB}, A\}K_{BS}, \{N_B - 1\}K_{AB}]$.
Note that the sent messages of an entity are not included in its binding sequence. This is because it is often not possible to provide a guarantee to a sender that a sent message has not been deleted or modified en route (as per the definition of binding sequence). The sent messages, however, can be provided to a distinguisher $\mathcal{D}(C_b, \lambda)$, as a part of its auxiliary input λ.

In the following analysis, A B and an arbitrary entity C take the roles of X_c, X_j and X_l in the definitions of FLAGs. We start with the identification of B for A: IDNT($A \rhd B, \beta_A$). We construct a distinguisher $\mathcal{D}_A^{idnt}(C_b, \lambda)$ for A as follows.

> Let $\lambda = \{K_{AB}, K'_{AB}, K_{AC}\}$. On receiving $C_b = (\beta_A^1, \beta_A^2)$, $\mathcal{D}_A^{idnt}$ tries to decrypt C_b with each key in the auxiliary input. If $\mathcal{D}_A^{idnt}$ succeeds with K_{AB}, K'_{AB} then it returns 0, otherwise it returns 1.

Clearly, the distinguisher $\mathcal{D}_A^{idnt}$ succeed with a high probability, therefore IDNT($A \rhd B, \beta_A$) is achieved.

Now we consider OPER($A \rhd B, \beta_A$). Let $\lambda = \{K_{AB}\}$. The distinguisher $\mathcal{D}_A^{oper}(C_b, \lambda)$ is constructed as follows.

> On receiving $C_b = \beta_A$, $\mathcal{D}_A^{oper}$ tries to decrypt C_b with K_{AB}. If $\mathcal{D}_A^{oper}$ succeeds then it returns 0, otherwise it returns 1.

As $\mathcal{D}_A^{oper}$ succeeds with a high probability, OPER($A \rhd B, \beta_A$) is achieved[11].

Now consider WLNG($A \rhd B, \beta_A$). In this case we need to construct a distinguisher on B: $\mathcal{D}_B^{wlng}(C_b, \lambda)$. Let $\lambda = \{K_{AB}, K'_{AB}, K_{AC}\}$. This auxiliary input is computable on B because B receives these values in M_3.

> On receiving $C_b = (\beta_A^1, \beta_A^2)$, $\mathcal{D}_B^{wlng}$ tries to decrypt C_b with each key in the auxiliary input. If $\mathcal{D}_B^{wlng}$ succeeds with K_{AB}, K'_{AB} then return $b = 0$, otherwise return $b = 1$.

Since the distinguisher $\mathcal{D}_B^{wlng}$ succeed with a high probability, therefore WLNG($A \rhd B, \beta_A$) is achieved.

The single-sided authentication (SATH) is also achieved for B: SATH($A \rhd B, \beta_A$) = IDNT($A \rhd B, \beta_A$) $\wedge$ OPER($A \rhd B, \beta_A$) $\wedge$ WLNG($A \rhd B, \beta_A$).
We cannot go any further because CNFM requires that β_A should be a concatenation of two binding sequences, which is not possible as there is only one message in β_A.

Now we turn to B and consider IDNT($B \rhd A, \beta_B$). Let $\lambda = \{K_{BS}\}$. We construct a distinguisher $\mathcal{D}_B^{idnt}(C_b, \lambda)$ for B as follows.

[10]Although the protocol is broken assuming an adversary can get hold of an old session key and the corresponding M_3 in the protocol [16], however, this does not affect the analysis for the correctness. The reader may assume that adversary cannot recover an old session key.

[11]Clearly, if we do not assume that the encryption is non-malleable then it is not possible to build this distinguisher; as an counter example, one can simply play any data in M_4 and A will be convinced that it is is sharing K_{AB} with B.

On receiving $C_b = (\beta_B^1, \beta_B^2)$, $\mathcal{D}_B^{idnt}$ decrypts C_b with K_{BS}. If the identities in the two plain texts match then return $b = 1$, otherwise return $b = 0$.

Clearly, $\mathcal{D}_B^{idnt}$ succeed with a high probability, therefore IDNT$(B \rhd A, \beta_B)$ is achieved.

Next we consider OPER$(B \rhd A, \beta_B)$. Let $\lambda = \{K_{BS}, N_B\}$. The distinguisher $\mathcal{D}_B^{oper}(C_b, \lambda)$ is constructed as follows.

On receiving $C_b = \beta_B$, $\mathcal{D}_B^{oper}$ decrypts the first message in β_B to compute K_{AB}. Then $\mathcal{D}_B^{oper}$ decrypts the second message in β_B. If the decrypted text is indeed $N_B - 1$ then it returns 0 otherwise it returns 1.

As $\mathcal{D}_B^{oper}$ succeed with an high probability, OPER$(B \rhd A, \beta_B)$ is achieved.

Now we turn to WLNG$(B \rhd A, \beta_B)$. In this case, we need to construct a distinguisher on A: $\mathcal{D}_A^{wlng}(C_b, \lambda)$. Let $\lambda = \{K_{AS}, M_2, M_2', M_2''\}$

On receiving $C_b = (\beta_B^1, \beta_B^2)$, $\mathcal{D}_A^{wlng}$ decrypts M_2, M_2' and M_2'', with K_{AS} read from the auxiliary input. Although $\mathcal{D}_A^{wlng}$ cannot decrypt β_B^1 or β_B^2, it can match them to the decrypted texts of M_2, M_2' and M_2''. If the match is with M_2 and M_2' then it returns 0, otherwise 1.

Clearly, $\mathcal{D}_A^{wlng}$ succeed with a high probability, therefore WLNG$(B \rhd A, \beta_B)$ is achieved.

Consequently, the single-sided authentication (SATH) is also achieved for A:
SATH$(B \rhd A, \beta_B)$ = IDNT$(B \rhd A, \beta_B) \wedge$ OPER$(B \rhd A, \beta_B) \wedge$ WLNG$(B \rhd A, \beta_B)$.
Although β_B contains two messages we cannot split β_B because otherwise OPER cannot be achieved from the first half. Therefore CNFM cannot be achieved, and we cannot proceed any further with the analysis.

VIII. Conclusion

We presented a *binding sequence* based framework that achieves demarcation of security for authentication protocols. In particular, we demonstrated that a *binding sequence* is the only required security property; all authentication properties of practical significance can be derived from the *binding sequence* of a protocol. In this way, we are able to define a clear boundary between security and correctness. In general, designing and verifying security protocols is a delicate matter. Many aspects of secure protocol design are close to the art than science. We hope that this theoretical contribution towards demarcation of security will help to bring the subject on sound footings.

References

[1] M. Burrows and M. Abadi, *A logic of Authentication*, DEC System Research Center, Report 39, revised Feb 22, 1990

[2] M. Bellare and P. Rogaway, *Entity authentication and key distribution*, Crypto'93, Pg.232–249, Springer-Verlag LNCS, Vol 773, 1993

[3] D. Gollmann, *What do we mean by entity authentication?*, IEEE Symposium on Security and Privacy, Pg.46–54, 1996

[4] A. W. Roscoe, *Intensional specifications of security protocols*, 9th IEEE workshop on Computer Security Foundations, Pg.28, 1996

[5] G. Lowe, *A Hierarchy of Authentication Specifications*, Proceedings of 10th Computer Security Foundations Workshop (CSFW '97), 1997

[6] G. Lowe, *Breaking and fixing the Needham-Schroeder public-key protocol using FDR*, Journal: Tools and Algorithms for the Construction and Analysis of Systems, pg.147–166,1996

[7] D. Gollmann, *Authentication — myths and misconception*, Progress in Computer Science and Applied Logic 20, Pg.203–225, 2001

[8] C. Boyd and A. Mathuria, *Protocols for Authentication and Key Establishment*, Springer Book, ISBN: 978-3-540-43107-7, 2003

[9] N. Ahmed and C.D. Jensen, *Definition of entity authentication*, International Workshop on Security and Communication Networks (IWSCN), Karlstad, Pg.1–7, 2010

[10] N. Ahmed and C.D. Jensen, *Entity authentication: analysis using structured intuition*, 4th Nordic Workshop on Dependability and Security, NODES10, 2010, Copenhagen, 2010

[11] N. Ahmed and C.D. Jensen, *Adaptable authentication model: exploring security with weaker attacker models*, In Proceedings of Engineering Secure Software and Systems, Madrid, (also in IMM Technical Report-2010-17), Pg.234–247, 2010

[12] C. Bodei, M. Buchholtz, P. Degano, F. Nielson and H.R. Nielson, *Static validation of security protocols*, Journal of Computer Security, Pg.347–390, 2005

[13] D. Basin,S. Mödersheim, L. Vigano, *OFMC: A symbolic model checker for security protocols*, International Journal of Information Security, Pg.181–208, 2005

[14] S. Even and O. Goldreich, *On the security of multi-party ping-pong protocols*, 24th Annual Symposium on Foundations of Computer Science, Pg.34–39, 1983

[15] R.M. Needham and M.D. Schroeder, *Using encryption for authentication in large networks of computers*, Journal: Communications of the ACM, Pg.993–999, 1978

[16] D.E. Denning and G.M. Sacco, *Timestamps in key distribution protocols*, Journal: Communications of the ACM, Pg.533–536, 1981

[17] N. Ahmed, C.D. Jensen and E. Zenner, *Refining Encryption in Formal Security Analysis*, IMM Technical Report-2011-10, 2011

3 Research Roadmap Papers

This chapter contains copies of the accepted research roadmap (position) papers as they will appear in the actual proceedings.

The MINESTRONE Architecture Combining Static and Dynamic Analysis Techniques for Software Security

Angelos D. Keromytis
Columbia U.
angelos@cs.columbia.edu

Salvatore J. Stolfo
Columbia U.
sal@cs.columbia.edu

Junfeng Yang
Columbia U.
junfeng@cs.columbia.edu

Angelos Stavrou
George Mason U.
astavrou@gmu.edu

Anup Ghosh
George Mason U.
aghosh@gmu.edu

Dawson Engler
Stanford U.
engler@csl.stanford.edu

Marc Dacier
Symantec Research Labs
marc_dacier@symantec.com

Matthew Elder
Symantec Research Labs
matthew_elder@symantec.com

Darrell Kienzle
Symantec Research Labs
darrell_kienzle@symantec.com

I. Problem Statement

We present MINESTRONE, a novel architecture that integrates static analysis, dynamic confinement, and code diversification techniques to enable the identification, mitigation and containment of a large class of software vulnerabilities in third-party software. Our initial focus is on software written in C and C++; however, many of our techniques are equally applicable to binary-only environments (but are not always as efficient or as effective) and for vulnerabilities that are not specific to these languages. Our system seeks to enable the *immediate* deployment of new software (*e.g.,* a new release of an open-source project) and the protection of already deployed (legacy) software by transparently inserting extensive security instrumentation, while leveraging concurrent program analysis, potentially aided by runtime data gleaned from profiling actual use of the software, to gradually reduce the performance cost of the instrumentation by allowing selective removal or refinement. Artificial diversification techniques are used both as confinement mechanisms and for fault-tolerance purposes. To minimize the performance impact, we are leveraging multicore hardware or (when unavailable) remote servers that enable quick identification of likely compromise. To cover the widest possible range of systems, we require no specific hardware or operating system features, although we intend to take advantage of such features where available to improve both runtime performance and vulnerability coverage.

The fundamental problem being addressed in this project – finding vulnerabilities in software — is being addressed in the commercial marketplace today by a combination of tools and expertise. Today, companies such as Coverity, Klocwork, Ounce Labs, and Fortify have developed sophisticated source code analyzers that analyze C/C++ and Java code for known vulnerabilities. Other tools such as ITS4, RATS, and cppcheck also provide vulnerabilities with varying degrees of effectiveness. Most of these products are state-of-the-art releases of software research and represent the best software vulnerability analysis has to offer today. One common attribute of these tools is that they produce a large number of false positives—warnings of potential vulnerabilities that often are not true. As such, they require software security expertise—a need that is met by commercial consulting offerings, some by the tool vendors themselves. Our ultimate goal is to take advantage of these and other analysis techniques without having to expose users, programmers or administrators to their output.

To address shortcomings in software vulnerability analysis, software fault isolation [1] or confinement techniques [2] can be used to limit the effects of residual software vulnerabilities. Software fault isolation techniques can be used to confine the bounds of program execution. Failure oblivious computing [3] is a set of software techniques to continue executing even in the presence of faults. Similarly, error virtualization uses a program's native error handling routines to mask faults while returning the program to a known safe state after an error [4]. Almost all confinement approaches require program instrumentation that imposes additional instruction execution time, which means increased (potentially substantial) overhead. The finer-grained the instrumentation, the finer the containment and the more overhead is required. The coarser the instrumentation, on the other hand, the lower the overhead, but the higher the likelihood for missed or unhandled faults. We will use code instrumentation to implement software fault isolation and process-level confinement to limit malicious software behavior. We will leverage our analysis to remove or refine this instrumentation so as to minimize or eliminate its performance impact when possible.

II. Research Direction

The overall MINESTRONE architecture and system workflow is shown in Figure 1. The intellectual core and novelty of our approach revolves around establishing and leveraging a feedback loop among a hardened production system, program analysis, and diversification. The key idea is that static analysis will allow us to target the instrumentation, while runtime data

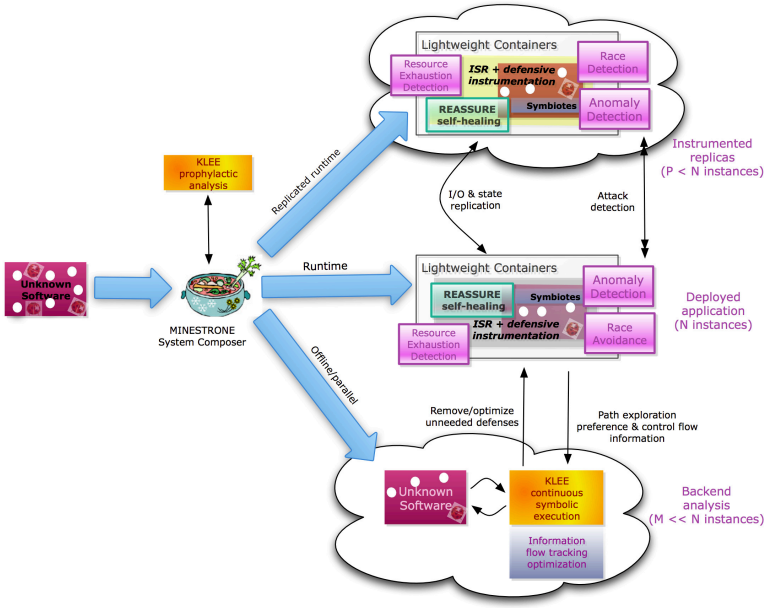


Fig. 1. **The MINESTRONE architecture and workflow. New software is processed by the MINESTRONE meta-compiler which, depending on configuration and environment capabilities, deploys several components. The production software (PS) is embedded within a lightweight OS-virtualization container which manages resource consumption and uses behavior anomaly detection. Inline reference monitors, in the form of binary- or source-rewriting are also embedded within the PS. A separate environment used for symbolic analysis is also created, with communication between PS and analysis. Diversified replicas (DRs) with similar instrumentation may also be set up, with I/O and state sharing performed with the PS. A given analysis instance may be used by one or more PS; likewise, each PS may be using one or more DRs.**

will allow us to focus further concurrent program analysis (through symbolic execution) to portions of the code that are more heavily exercised or are otherwise considered security-critical. We are using symbolic execution combined with static analysis to determine the safety properties of parts of software that is being deployed. Our symbolic execution framework [5], [6], [7] will explore possible execution paths of a program by analyzing it on unconstrained symbolic input and systematically following branches when the outcome depends on the symbolic input. Further, we will selectively integrate our previous static analysis and model checking frameworks [8], [9], [10], [11], [12], [13], [14], [15] into MINESTRONE to detect vulnerabilities such as number handling, error handling, concurrency handling, memory safety errors (*e.g.,* buffer overflows/underflows), null pointer errors, and tainted data/input validation errors. On top of these program analysis frameworks, we plan to build several novel analysis techniques to improve the coverage of error detection, the soundness of confinement and diversification, and the speed of the entire system. While the initial static analysis may be done either at a centralized repository from which the software is downloaded or at the end-user machine, the symbolic execution component is likely to be centralized (*e.g.,* at the department or enterprise level) to avoid unnecessary duplication of significant effort.

Since symbolic execution of non-trivial software is a time-consuming endeavor, we proactively and comprehensively (except as otherwise indicated by static analysis) instrument software with code that detects and confines vulnerabilities. The nature of the instrumentation depends on the type of vulnerability. When source code is available, we will insert the instrumentation through source-code transformations [16], [17]. Otherwise, we will inject our instrumentation in program binaries using the PIN binary rewriting tool (which is neither as efficient nor as effective, due to the various challenges in working with binaries) [18], [4]. The specific vulnerability types we are protecting against include number handling, error handling, concurrency handling, memory safety errors (*e.g.,* buffer overflows/underflows), null pointer and tainted data/input validation errors.

Multithreaded code is difficult to write because developers must reason about all possible ways the threads may interact with each other. Due to the same complexity, multithreaded code is also difficult to debug and fix. For example, a study has shown that a significant number of concurrency error fixes did not fix the corresponding errors and, worse, introduced new errors [19]. This situation may worsen as developers are writing more multithreaded programs driven by the high performance demand and the current multicore trend. We propose to automatically avoid races[1] using a number of novel techniques. Mechanically, our approach will work as follows: (1) analyze applications to detect likely data races, (2) merge adjacent data races into atomic regions that match developer atomicity or ordering intents of code, and (3) defensively insert synchronization operations to prevent these likely races. The key advantage of this approach is to relieve developers from fixing many races, thus improving the reliability of multithreaded software.

[1]We use the term *race* to denote all non-deadlock concurrency errors, including low-level data races (*i.e.,* concurrent accesses to a shared variable with at least one write access), atomicity errors, and order errors [19].

One important and novel element of our approach involves protecting the inline reference monitor itself from silent (unobserved) compromise. We will achieve this using what we call "In-Code Execution" (ICE). ICE is inserted into the program code either using existing "memory gaps" between code and string structures or specially crafted padding generated during the analysis phase. ICE creates an independent execution context from the native program context at runtime making sure that all the necessary state information is preserved. The reference monitor executes as an encrypted payload spread out throughout the code. The ICE insertion and payload encryption engines can be modified at each run of the binary offering a diversification of the protection mechanism that elevates the protection of the reference monitor. It is important to note that ICE does not use traditional virtualization techniques only standard CPU instructions. We will also use lightweight OS-level virtualization to provide an additional layer of process confinement, including anomaly detection. This will let us interact with the OS and enforce resource usage limits, protect the inline reference monitor, and capture/inspect/duplicate I/O (especially in conjunction with diversified replicas).

Artificial code diversification (ACD), in the form of ASLR and Instruction Set Randomization (ISR) [20], [21], will act as one form of containment. We have extended our previous work on ISR to minimize its performance impact by reducing the portion of the program that must be run in the ISR runtime. This is done by concentrating ISR to the parts of the code that static analysis indicates is more likely to contain bugs and by randomizing selective portions of the program that cannot be avoided by a successful compromise (thus causing a program fault) [22]. We are developing a framework that allows the integration of any type of localized ACD toward detecting attacks. When local resources are available (*e.g.,* fast system with multicore CPU), our system executes multiple versions of the diversified code and compare results. If local resources are insufficient, our instrumentation will transmit all process I/O to a remote system that runs diversified replicas of the application. While this will only allow *ex post facto* intrusion detection, its relatively low performance impact on the protected system allows its use on low-power devices such as smartphones and netbooks. Remote execution of instrumented programs replicas also enables investigation and implementation of additional diversification and fault detection approaches that would not otherwise be possible with deployed program instances.

The feedback loop will enable our system to gradually and selectively remove instrumentation checks as symbolic execution indicates that specific parts of the code are not susceptible to certain vulnerabilities. Thus, over time the performance of a deployed piece of software will improve. Furthermore, the injected instrumentation will gather usage data that will allow us to concentrate the symbolic execution to the actively used portions of the program. Specific input vectors may also be supplied to the symbolic execution engine to facilitate its exploration of the code paths and state space.

As a specific example of such interaction, we briefly discuss how analysis can improve the performance of our dynamic confinement and self-healing components. As part of handling a security violation, these components may roll back a faulty execution to a series of recent checkpoints for recovery. Since taking a checkpoint is expensive, we will avoid doing so when possible by developing a purity analysis that analyzes the side effects of functions [23]. There are three cases we can skip checkpointing a function: (1) when the function is *pure*: it does not modify any memory location outside the stacks, allocate or deallocate any resource, or perform any I/O; (2) when the function is *on-error pure*: it does not have any side-effect when it returns an error; and (3) when the function is *partially pure*: it has side effects only on some of its execution paths, so we only need to checkpoint for those paths.

While general purity analysis has been previously studied [23], [24], we are the first to propose on-error purity and partial purity. Further, a key novel feature of our analysis is *return-code-sensitive*, *i.e.,* it will use the results from our error-code analysis to compute different side-effect summaries for success returns and error returns. Such return-code-sensitivity strikes a good balance between precision and scalability and is valuable for other kinds of static analysis as well, as functions tend to do very different things on success and error. We plan to extend return-code sensitivity to other types of static analysis as well.

In addition to designing, prototyping and evaluating the overall `MINESTRONE` architecture, we seek to advance the state of the art in each of the component areas, by expanding the scope of program analysis techniques to new vulnerability classes, developing self-protecting confinement mechanisms, and creating diversification schemes that offer highly tunable performance-security tradeoffs.

III. Current Status

Our primary task in realizing this research vision is the development of an integrated architecture that combines static and program analysis, confinement, and diversification in a feedback system that allows for continuous improvement of the security and performance of the protected software. Therefore, our subtasks relate to the development of individual mechanisms within each of these areas, and the integration into a single system. We have been working on this project vision since August 2010. In that time, we have refined the architecture and have been working toward building the individual components:

- We have developed a binary IRM that implements selective ISR, Write Integrity, self-healing, and taint tracking with advanced performance optimizations to remove unnecessary instrumentation. For example, in some applications the overhead of ISR is less than 1%, while we have reduced the overhead of taint analysis by 40% to 60% over the best reported implementation [25], [26].
- We have developed techniques for analyzing programs to identify and mitigate concurrency bugs [27], [28]. Using static analysis and schedule memoization, we can force safe thread

scheduling with modest performance impact.

• We have developed versions of ICE that can be embedded in binaries for different architectures (ARM, MIPS, x86). We have demonstrated ICE for such diverse environments as Cisco routers and Android handsets. The ICE implementation is very efficiently executed utilizing the raw computational resource of the hardware platform, bypassing layers of overhead produced by operating systems or VMs that host an OS. One advantage of an ICE security payload over a reference monitor is better performance.

• We have augmented our initial lightweight container scheme with full-process logging/replay and system call monitoring. This capability, combined with the self-healing component in our IRM, will allow us to do fast rollback.

• We have scaled up the symbolic execution component, with respect to the state space explored. Our new techniques allow us to identify equivalent states (and thus avoid them), yielding a 9-fold performance speedup on average.

• We are developing a I/O redirection prototype that covers interactive applications. We are integrating this with our diversification IRM and the lightweight containers, allowing us to place and move replicas in any number of systems and to deploy the detection and mitigation techniques we develop.

The main challenge with our work will be managing the integration complexity, conducting realistic experiments, and developing a management framework that makes it easy to use `MINESTRONE`. This will be the focus of our future efforts.

Acknowledgements*:* This work was supported by the US Air Force through Contract AFRL-FA8650-10-C-7024. Any opinions, findings, conclusions or recommendations expressed herein are those of the authors, and do not necessarily reflect those of the US Government or the Air Force.

REFERENCES

[1] Wahbe, R., Lucco, S., Anderson, T.E., Graham, S.L.: Efficient software-based fault isolation. In: In Proceedings of the 14th ACM Symposium on Operating Systems Principles. (1993) 203–216

[2] Seward, J., Nethercote, N.: Valgrind, an open-source memory debugger for x86-linux. (http://developer.kde.org/~sewardj/)

[3] Rinard, M.C., Cadar, C., Dumitran, D., Roy, D.M., Leu, T., Beebee, W.S.: Enhancing server availability and security through failure-oblivious computing. In: OSDI. (2004) 303–316

[4] Sidiroglou, S., Laadan, O., Viennot, N., Perez, C.R., Keromytis, A.D., Nieh, J.: ASSURE: Automatic Software Self-healing Using REscue points. In: Proceedings of the 14^{th} International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS). (2009) 37–48

[5] Cadar, C., Dunbar, D., Engler, D.: KLEE: Unassisted and automatic generation of high-coverage tests for complex systems programs. In: Proceedings of the 8^{th} Symposium on Operating Systems Design and Implementation (OSDI). (2008) 209–224

[6] Yang, J., Sar, C., Twohey, P., Cadar, C., Engler, D.: Automatically generating malicious disks using symbolic execution. In: Proceedings of the 2006 IEEE Symposium on Security and Privacy (SP '06). (2006) 243–257

[7] Cadar, C., Ganesh, V., Pawlowski, P.M., Dill, D.L., Engler, D.R.: EXE: automatically generating inputs of death. In: Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06). (2006) 322–335

[8] Engler, D., Chelf, B., Chou, A., Hallem, S.: Checking system rules using system-specific, programmer-written compiler extensions. In: Proceedings of Operating Systems Design and Implementation (OSDI). (2000)

[9] Engler, D., Yu Chen, D., Hallem, S., Chou, A., Chelf, B.: Bugs as deviant behavior: A general approach to inferring errors in systems code. In: Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01). (2001)

[10] Engler, D., Ashcraft, K.: RacerX: Effective, Static Detection of Race Conditions and Deadlocks. In: Proceedings of ACM SOSP. (2003)

[11] Yang, J., Kremenek, T., Xie, Y., Engler, D.: MECA: an extensible, expressive system and language for statically checking security properties. In: Proceedings of the 10^{th} ACM Conference on Computer and Communications Security (CCS). (2003)

[12] Ashcraft, K., Engler, D.: Using programmer-written compiler extensions to catch security holes. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy (SP '02), Oakland, California (2002)

[13] Yang, J., Chen, T., Wu, M., Xu, Z., Liu, X., Lin, H., Yang, M., Long, F., Zhang, L., Zhou, L.: MODIST: Transparent model checking of unmodified distributed systems. In: Proceedings of the 6^{th} Symposium on Networked Systems Design and Implementation (NSDI). (2009)

[14] Yang, J., Sar, C., Engler, D.: Explode: a lightweight, general system for finding serious storage system errors. In: Proceedings of the Seventh Symposium on Operating Systems Design and Implementation (OSDI '06). (2006) 131–146

[15] Yang, J., Twohey, P., Engler, D., Musuvathi, M.: Using model checking to find serious file system errors. In: Proceedings of the 6^{th} Symposium on Operating Systems Design and Implementation (OSDI '04). (2004) 273–288

[16] Sidiroglou, S., Keromytis, A.D.: Execution Transactions for Defending Against Software Failures. International Journal of Information Security (IJIS) **5** (2006) 77–91

[17] Sidiroglou, S., Locasto, M.E., Boyd, S.W., Keromytis, A.D.: Building A Reactive Immune System for Software Services. In: Proceedings of the 11^{th} USENIX Annual Technical Conference. (2005) 149–161

[18] Kim, H.C., Keromytis, A.D.: On the Deployment of Dynamic Taint Analysis for Application Communities. IEICE Transactions **E92-D** (2009) 548–551

[19] Lu, S., Park, S., Seo, E., Zhou, Y.: Learning from mistakes: a comprehensive study on real world concurrency bug characteristics. In: ASPLOS XIII: Proceedings of the 13th international conference on Architectural support for programming languages and operating systems, New York, NY, USA, ACM (2008) 329–339

[20] Kc, G.S., Keromytis, A.D., Prevelakis, V.: Countering Code-Injection Attacks With Instruction-Set Randomization. In: Proceedings of the 10^{th} ACM Conference on Computer and Communications Security (CCS). (2003) 272–280

[21] Boyd, S.W., Keromytis, A.D.: SQLrand: Preventing SQL Injection Attacks. In: Proceedings of the 2^{nd} Applied Cryptography and Network Security Conference (ACNS). (2004) 292–302

[22] Locasto, M., Wang, K., Keromytis, A., Stolfo, S.: FLIPS: Hybrid Adaptive Intrusion Prevention. In: Proceedings of the 8^{th} Symposium on Recent Advances in Intrusion Detection (RAID). (2005) 82–101

[23] Landi, W., Ryder, B.G., Zhang, S.: Interprocedural side effect analysis with pointer aliasing. In: PLDI '93: Proceedings of the 1993 ACM SIGPLAN conference on Programming language design and implementation. (1993)

[24] Xu, H., Pickett, C.J.F., Verbrugge, C.: Dynamic purity analysis for java programs. In: PASTE '07: Proceedings of the 7th ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering. (2007)

[25] Portokalidis, G., Keromytis, A.D.: Fast and Practical Instruction-Set Randomization for Commodity Systems. In: Proceedings of ACSAC. (2010)

[26] O'Sullivan, P., Anand, K., Kothan, A., Smithon, M., Barua, R., Keromytis, A.D.: Retrofitting Security in COTS Software with Binary Rewriting. In: Proceedings of the 26^{th} IFIP International Information Security Conference (SEC). (2011)

[27] Cui, H., Wu, J., che Tsai, C., Yang, J.: Stable Deterministic Multithreading through Schedule Memoization. In: Proceedings of the 9^{th} Symposium on Operating Systems Design and Implementation (OSDI). (2010)

[28] Wu, J., Cui, H., Yang, J.: Bypassing Races in Live Applications with Execution Filters. In: Proceedings of the 9^{th} Symposium on Operating Systems Design and Implementation (OSDI). (2010)

The Free Secure Network Systems Group: Secure Peer-to-Peer Networking and Beyond

Christian Grothoff
Free Secure Network Systems Group
Department of Computer Science
Technische Universität München
Email: grothoff@net.in.tum.de

***Abstract*—This paper introduces the current research and future plans of the Free Secure Network Systems Group at the Technische Universität München. In particular, we provide some insight into the development process and architecture of the GNUnet P2P framework and the challenges we are currently working on.**

I. Introduction

The Free Secure Network Systems Group (FSNSG) was established in Fall 2009 by a grant from the Deutsche Forschungsgesellschaft (DFG) in the Emmy-Noether Program. It currently consists of four full-time researchers (including three PhD students) and five Master's students working on research projects or theses. The group is largely working in the area of secure peer-to-peer (P2P) networks, but is also looking at networking issues in general, including work on scaleable graph algorithms and distributed programming [1].

In terms of security, our focus is on secure network protocol design and implementation. Secure software engineering is a key component when building secure systems. Our software engineering practice is tool-centric; as part of our implementation work, we use, extend and sometimes develop software engineering tools, in particular static analysis tools, portability and regression testing tools. Since availability and performance often are closely related, we are currently developing a new tool for cross-platform performance regression analysis.

II. Current Research

The main focus of our group is the development of GNUnet[1], GNU's framework for secure P2P networking. One of the characteristics of the GNUnet system is that it uses a multi-process architecture for fault isolation; failures in individual components are isolated in their respective address spaces and rarely affect other parts of the system. While GNUnet is currently mostly written in C, the multi-process architecture also enables the development of extensions in other languages.

GNUnet uses a layered architecture (Figure 1). At the bottom layer, transport plugins enable P2P communication [2]. GNUnet can currently communicate using UDP, TCP, HTTP or HTTPS. Support for IP-less direct communication using WLAN is under development. Our next goal here is to formalize an effective strategy for efficiently selecting the best communication method while satisfying resource and security constraints. The transport layer also includes support for NAT traversal [3]. Above the transports, the GNUnet core provides link-encrypted peer-to-peer communication, bandwidth allocation, peer discovery and other general-purpose functions necessary for any secure P2P network. GNUnet applications (for example, anonymous file-sharing [4]) and services (for example, our DHT) then use the core to communicate with other peers.

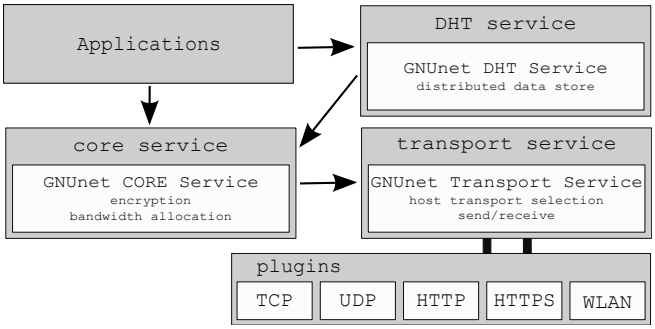


Fig. 1. GNUnet Architecture.

Our main research focus at this point is on demonstrating the security and scalability of a new randomized DHT design and its use for various applications. Our DHT has the special property that it can work on top of a restricted-route underlay; in other words, it does not make the assumption that any peer can directly communicate with any other peer. Using this DHT, we plan to develop a mesh routing abstraction for the construction of redundant tunnels between clients and network services integrated into the P2P overlay. As a first service, we plan to offer a virtual network interface (IP-VPN) that uses the P2P overlay to provide IP connectivity (for both IPv4 and IPv6).

For the evaluation of our designs, we emulate large P2P networks using the testing support library available in GNUnet. Using the GNUnet framework, we can easily setup thousands of peers on a single desktop system (or tens of thousands using a cluster), connect them using various topologies and run experiments. A transport plugin using UNIX-domain sockets can be used to avoid problems with the 64k port limitation of UDP or TCP.

[1] https://gnunet.org/

III. Teaching

Our teaching goals are to enable students to design and implement secure systems as well as to analyze existing designs for flaws. The GNUnet framework provides a starting point for the design and implementation of secure P2P network protocols. For example, we have in the past asked students in the "Peer-to-Peer Systems and Security" course to implement a range of proposed DHT designs in the framework. While some groups succeeded, it is clear that making the framework accessible from multiple languages would be a major improvement: many of our students are more proficient in Java or Python than in C. This need reinforces our belief that the multi-process architecture is the right design choice for a P2P framework. Currently, we are asking students to design and implement a distributed web search engine using the framework; however, it is too early to draw any conclusions from this. However, students at other universities have already created new applications using the GNUnet framework.[2]

On the analysis side, we regularly supervise students who, as part of their master's thesis, analyze the design of an existing free software P2P network and then devise, implement and evaluate an attack based on vulnerabilities caused by particular design choices. The goal is not to find simple bugs in the existing implementation but to find and exploit weaknesses that the developers build-in by design. For example, our attack on Tor [5] is based on source-routing and low-latency routing, two fundamental design choices for Tor. For Freenet, our attack [6] exploits a key step in their routing algorithm. Our recent work on I2P [7] builds on their use of unidirectional paths and performance-based peer selection. The resulting thesis is typically publishable work and the students are keenly aware of the security implications of certain design choices and have learned to understand complex software systems to a sufficient degree to find design flaws by studying documentation and source code.

Finally, we of course encourage all of our students to famlilarize themselves with the various software engineering tools that we have deployed. This generally improves their ability to write correct code quickly. Furthermore, we believe that knowing available tools is key for secure software engineering.

IV. Future Plans

We currently see an urgent need for an Internet architecture that is resilient to malicious participants and not under the control of cooperations or governments. With the widespread use of wireless networking equipment, a secure, scalable and most of all easy-to-use P2P network with support for DNS and an IP-VPN could solve the problem of three-strike-Internet-kill-switches and further the agenda of free software: user's freedom. Naturally, a large number of technical hurdles need to be overcome: secure routing, scaleable creation of virtual tunnels with TCP-like semantics, secure naming for DNS and design and integration of secure variants of important Internet applications into the P2P network. Finally, we hope to face the challenge of making the resulting system easy to use while maintaining security for ordinary users.

In the near term, we also plan to further extend on our tool suite for secure software engineering. In particular, we want to customize off-the-shelf tools to better support the idioms of a particular large software system. Furthermore, we are working a tool that can be deployed at end-user systems to help developers automate key steps in the diagnosis of problems, especially those that they cannot reproduce on their own systems. This could be particularly useful if the end-user experiences system-specific problems, such as an external attack, and the developer requires more than simple logs or heap images for the diagnosis.

V. Conclusion

Our group offers expertise in the areas of analysis, design and implementation of secure P2P networks. We will be happy to support other groups that want to build systems using the GNUnet P2P framework for teaching or research. Feedback on the various libraries, software-engineering and language tools maintained by our group is also always welcome. We would be interested in models or measurement data to help make our security and performance analyses more realistic.

Acknowledgements

This work was funded by the Deutsche Forschungsgemeinschaft (DFG) under ENP GR 3688/1-1.

References

[1] N. Evans, C. GauthierDickey, C. Grothoff, K. Grothoff, J. Keene, and M. J. Rutherford, "Simplifying parallel and distributed simulation with the DUP system," in *Proceedings 43rd Annual Simulation Symposium (ANSS-43 2010)*. Orlando, FL, USA: Society for Modeling & Simulation International, April 2010, pp. 208–215. [Online]. Available: http://dupsystem.org/anss2010

[2] R. A. Ferreira, C. Grothoff, and P. Ruth, "A Transport Layer Abstraction for Peer-to-Peer Networks," in *Proceedings of the 3rd International Symposium on Cluster Computing and the Grid (GRID 2003)*. IEEE Computer Society, 2003, pp. 398–403. [Online]. Available: https://gnunet.org/transports

[3] A. Müller, N. Evans, C. Grothoff, and S. Kamkar, "Autonomous nat traversal," in *10th IEEE International Conference on Peer-to-Peer Computing (IEEE P2P 2010)*. IEEE, 2010, pp. 61–64. [Online]. Available: https://gnunet.org/pwnat

[4] K. Bennett and C. Grothoff, "gap - Practical Anonymous Networking," in *Designing Privacy Enhancing Technologies*. Springer-Verlag, 2003, pp. 141–160. [Online]. Available: http://gnunet.org/gap

[5] N. S. Evans, R. Dingledine, and C. Grothoff, "A practical congestion attack on tor using long paths," in *18th USENIX Security Symposium*. USENIX, 2009, pp. 33–50. [Online]. Available: https://gnunet.org/torattack

[6] N. S. Evans, C. GauthierDickey, and C. Grothoff, "Routing in the dark: Pitch black," in *23rd Annual Computer Security Applications Conference (ACSAC 2007)*. IEEE Computer Society, December 2007, pp. 305–314. [Online]. Available: https://gnunet.org/pitchblack

[7] M. Herrmann and C. Grothoff, "Privacy-implications of performance-based peer selection by onion-routers: A real-world case study using i2p," in *Privacy Enhancing Technologies Symposium (PETS 2011)*, 2011. [Online]. Available: https://gnunet.org/i2p_2011_pet

[2] http://sourceforge.net/projects/s-n-a-g/

Adapting Econometric Models, Technical Analysis and Correlation Data to Computer Security Data

Spyros K. Kollias*, Vasileios Vlachos†, Alexandros Papanikolaou† and Vassilis Assimakopoulos*

*School of Electrical and Computer Engineering
National Technical University of Athens
Athens, Greece
Email: spyridon.kollias@gmail.com,
vassim@central.ntua.gr

†Dept of Computer Science and Telecommunications
Technological Educational Institute of Larissa
Larissa, Greece
Email: vsvlachos@gmail.com,
alpapanik@teilar.gr

***Abstract*—Guaranteeing the safety of computers connected to the Internet is a challenging task. Despite the efforts of contemporary security software, the threat remains due to the incapability of existing software to predict and prevent the variance of attacks. According to recent studies, new computer malware appears at an almost constant rate, making their confrontation a rather difficult task and therefore creating a need for a different approach that will increase the effectiveness of security software. This paper introduces forecasting models and techniques from the financial world. Some possible approaches are investigated, such as the correlation between computer malware incidents and extracted data from electronic social networks, that could possibly lead to effective forecasting, in an attempt to come up with new ways for preventing imminent computer epidemics.**

***Keywords*-security; forecasting; computer malware; time series;**

I. Introduction

"Prediction" and "forecasting" are words that rarely refer to computer security. Actions against a given threat usually take place after its existence has been detected and its nature has been determined, by applying trivial methods such as signature matching and observation of abnormal activities [1]–[3]. On the other hand, financial forecasting has been applied to objects of interest by both the industrial and academic world, since large amounts of past data have became available for processing [4], [5]. Moreover, the currently available computational power permits the processing of large such amounts of data in a short period of time. It is worth investigating whether these two fields can be combined by linking stocks data with appropriate security data, in order to forecast imminent cyber-attacks and prevent them from inflicting damage to organisations and personal computers. The first step is to define the relationship between stock data (price, volume, durations etc.) and data from the security world (number of attacks, category of attack, protocol used etc.). Consequently, some issues may arise from the diversity of the data, probably rendering some models totally inapplicable and others applicable only after they have undergone appropriate modifications.

Apart from financial models, there is a lot of data that can be correlated with computer security. For example, user comments in electronic social networks, queries in search engines or even the number of visited related sites can predict the future computer security trends.

Due to space limitations, research leads will mainly be presented, in a high-level form, without providing an in-depth analysis. The paper is organised as follows: Section II summarises the related work, Section III presents ideas and discusses the proposed models, while Section IV concludes by summarising the proposed scheme.

II. Related Work

Financial forecasting is mainly based on processing past data and since data gathering is a temporal process, measures of time series analysis can be used for investigating normality, possible correlation and linear trend in the data.

The various techniques and methodologies on financial forecasting have also been adopted by other fields. For instance, different implementations of the ARIMA time series model have been used for forecasting the wheat area and production in Pakistan [6], traffic flow [7], the reliability of computer systems [8] or even the price of electricity on the next day [9].

The GARCH model has been used for making assumptions on the influence of exchange rate volatility in U.S.

imports from Canada [10] and even for forecasting air pollution levels [11].

Apart from the financial models, data scattered across the Internet has also been exploited, in an attempt to correlate it with a particular topic of interest and hence forecast near-future trends. For instance, in [12] data from electronic social networks was exploited in order to predict the box office revenue for movies. In particular, the input data was from Twitter user posts and the number of "tweets" referring to a specific movie were counted. Furthermore, an appropriate mechanism was employed for determining whether the comments on a particular movie were positive or negative. Moreover, the author in [13] used search engine query data to predict and detect influenza epidemics, a technique that was used for predicting the spread of influenza in the US between the years 2008 and 2009. Last but not least, the work presented in [14] is able to forecast the US unemployment rate by exploiting data from Google's search index.

An example of threat-predicting system is the PROactive Threats Observatory System (PROTOS), proposed in [15], part of which is a kind of a predicting mechanism, called FORIT (FORecasting Imminent Threats). PROTOS is a system that can protect non-expert computer users from threats, according to recently-observed threats. FORIT predicts imminent attacks and gives feedback to PROTOS, so as to adjust the security level accordingly. In addition, PROTOS becomes an information and data source, to notify other systems. Should financial forecasting models prove to be successful when applied to security-related data, such a mechanism could be incorporated to the FORIT sub-system.

III. Discussion

The proposed research will be based on applying well-known financial models and techniques to data related to computer security. Although the success of such an action cannot be guaranteed, nevertheless such models have successfully been applied to other kinds of data (Section II), thus making our approach promising.

Initially, publicly available data from sources such as WildList [16] and information from organisations such as DShield will be used. Generally speaking, there are two main approaches: The first one applies most of the econometrics models (ARMA, ARIMA, GARCH, ACD) to past security-related data; the second one predicts the future of security using technical analysis patterns, by observing the charts and looking for similarities with past data (computer observation using genetic algorithms or artificial neural networks). The correlation matrices may prove to be rather helpful in understanding the data and the various relationships within it. There also seems to be a correlation between external data (such as that from security-related accounts in electronic social networks and search engines' query data) and the trends of electronic attacks. Should there be a significant correlation with a time lag, it will be possible to speculate the future of computer security.

Furthermore, one other kind of realistic data which will be processed for extracting additional information is the cost of stolen credit card numbers (in groups of 1000s, for example), the cost for every MB of e-mail addresses to be used for sending spam, the cost of malware kits (such as Zeus and SpyEye) and the cost for renting botnets. The reason for doing so is because the price of the available undetectable custom malware and that of given volumes of credit card numbers from underground communities and markets can be correlated to the security level as thus: if the current price for obtaining, say, credit card numbers is higher than "yesterday's price", it could mean that it is more difficult to retrieve credit card details, possibly because there is a higher level of security associated with the related transactions. Of course, this is an oversimplification but can create an index that can contribute to the overall effort. Such data can provide us with useful information and most importantly, since they involve real-life financial transactions, the econometric models are expected to produce really useful results.

An important issue is the amount of data that is required for such a purpose. The publication of financial data is quite well regulated, whereas security-related data is sparser. Hence, part of this research would be to determine whether the available data is sufficient or additional sources should be exploited (such as Tweeter posts and search engines' query data) and perhaps create a requirement for publicising such data in a systematic manner. In addition, the proposed research does not aim at creating a stand-alone tool, but rather a component of a larger system. Some attacks may evade detection, like econometric models sometimes fail to detect crises. Another goal would be to minimise the probability of such failures to the best attainable degree.

Another problem is deciding the way in which the incidents will be considered. A simplistic approach could be to consider all threats as a stock index, without distinguishing them on the basis of their nature. For example, the "price" (index) of security or could be the number of any observed attacks over time, which would also be a simple way for plotting the initial charts. In this case, volume information will be lost, thus making some of the models inapplicable. It is worth noting that volume is one of the main factors that can influence the predictions of a stock.

An alternative approach would be to categorise threats according to their nature (protocol or port used, malware family) and consider them as being different stocks. This is a better approach due to the extra information that will be produced/preserved. What is more, each kind of threat can be evaluated separately and compared to a global view of threats. In this way, a vector bearing the weights of each kind of threat can be produced. The objective would be to find similarities between security data and stock data to enable the use of appropriate models on it and also exploit any

patterns that may appear. The sum of the attacks will be regarded as the "volume" and the severity of the attack as the "price". For instance, in the case of viruses, a plausible approach would be to set the price according to the mutations of each virus. Namely, a virus with many mutations will be priced higher than a virus with no mutations at all.

Although the described model has significant advantages, there are some issues that will most probably impede the use of all applicable models and technical analysis interpretation. In particular, stock market data includes much more information than just the price of the stock. For instance, within a 5-minute-long window, a stock market provides information such as: highest/lowest price, actual (closing) price, volume of shares transacted, duration between transactions, duration for an amount of shares transacted and duration for a difference of price.

In addition, the model should be flexible and able to adapt any new parameters representing external factors that could influence the future of security, otherwise, serious mishaps may occur. For instance, regarding financial modelling, most of the hedge fund strategies weren't in position to measure the liquidity risk, which lead to all long/short strategies collapsing due to liquidity in August of 2007 [17].

Also, both financial and security time series may work under the Efficient Market Hypothesis (EMH), which states that the time series reflect the information about their object. There are three different levels of EMH:

1) Weak EMH, where the time series reflect only the past, publicly available information.
2) Semi-strong EMH, where the time series reflect the up-to-now information and the price changes accordingly.
3) Strong EMH, where the time series reflect all the information about their object, namely publicly and "insider" information.

The financial sector does not work under the strong EMH due to arbitragers who try to exploit insider information and their movements are not reflected by the time series. Due to the current crisis, there is evidence that neither weak EMH nor semi-strong EMH work in the financial sector. The security sector time series, however, follow the strong EMH because there is no "insider" information. All the threats are real and have already occurred.

As has been mentioned, the authors believe that computer security data shares some similarities with financial data. The key is to interpret them in a way that security data fit the econometric models.

A. Models

The ARMA model is a combination of two models [18]: The Auto Regressive (AR) and the Moving Average (MA) model. These models are capable of making forecasts on returns.

A more generalised model that can be used is the Auto-Regressive Integrated Moving Average (ARIMA) model, which uses one more parameter in order to handle non-stationary time series.

The Generalised Auto-regressive Conditional Heteroscedasticity (GARCH) model predicts the volatility of time series. It has been recorded that the volatility in time series appears in clusters over time [19], thus making it easier to forecast these clusters.

Auto-regressive Conditional Duration (ACD) models make forecasts based on durations [20], such as the duration between two successive incidents or the required time for a given number of incidents to occur. This may be rather difficult to implement on security-related data and future research could possibly deal with this issue, part of which could be to process the already-existing data set more extensively, in order to extract additional information, such as durations.

B. Data and Scenarios

In this section, the different ways are presented, in which the available data may be exploited. The data set primarily consists of data taken from WildList [16] and is coded as follows: A plus sign ($+$) before the name of a virus denotes its first occurrence in the lists, whereas an asterisk ($*$) means that the virus had been introduced in a previous list, was then removed and it has now been added to the list again. Viruses whose names are not preceded by any sign have been included in at least one of the previous lists. The number following the virus' name denotes the different mutations of the specific virus. The date column contains information on when the virus was firstly inserted to the lists (month, year) and the last column states whom it was reported by. At the end of this list another one follows, containing the viruses that have been removed from the lists (preceded by a minus sign).

The proposed research will investigate the possibility of a correspondence between malware and stocks and hence the applicability of existing stock market models to computer malware. A stock is classified according to the size of the capital, thus classifying a company/stock as Top-Cap, Mid-Cap or Bottom-Cap. Viruses can have a similar categorisation, according to their durability, for example. Durability could be measured by how many lists has the viruses appeared in and how long has it existed in these lists for, thus classifying them as top-durable, mid-durable and bottom-durable viruses.

A top-durable virus will be known to most related security applications (antivirus engines, Intrusion Detection Systems) and will most probably be prevented from inflicting a significant amount of damage, similarly to a top-cap stock that is not expected to make big returns (either positive or negative). A bottom-cap stock is usually characterised as a high-risk investment, due to the fact that an incident

(either positive or negative) or a piece of information about the market itself could have a significant impact on it. In addition, a freshly-introduced virus can cause significant damage, before appropriate signature updates are released. However, if it is not an "intelligent" virus, it may be possible to be successfully confronted with ease. Essentially, this leads to a volatile environment that can be predicted by an appropriate GARCH model.

IV. Conclusion

In this paper, a research direction was proposed for correlating the available security-related data with financial data, in an attempt to exploit the already-existing financial forecasting models for security-related purposes, so as to enable prediction of imminent threats like computer viruses.

An initial approach was attempted in Section III, where the main ideas behind the proposed framework were discussed briefly, as well as some related issues that arose in the process. Additional data needs to be collected and extra processing on already-existing data needs to be performed, in order to extract useful information that will be used in the proposed research. What is more, should the proposed technique prove to be sufficiently effective, it can be used for improving the forecasting ability of related systems.

References

[1] G. Bakos and V. Berk, "Early Detection of Internet Worm Activity by Metering ICMP Destination Unreachable Messages," in *Proceedings of the the SPIE Aerosense*, E. M. Carapezza, Ed., vol. 4708, 2002, pp. 33–42.

[2] F. Yu, R. Katz, and T. V. Lakshman, "Gigabit Rate Packet Pattern-Matching Using TCAM," in *Proceedings of the 12th IEEE International Conference on Network Protocols, 2004. ICNP 2004.*, October 2004, pp. 174–183.

[3] J. Canto, M. Dacier, E. Kirda, and C. Leita, "Large Scale Malware Collection: Lessons Learned," IEEE SRDS Workshop on Sharing Field Data and Experiment Measurements on Resilience of Distributed Computing Systems, October 2008, Naples, Italy.

[4] W. Leigh, R. Purvis, and J. M. Ragusa, "Forecasting the NYSE Composite Index with Technical Analysis, Pattern Recognizer, Neural Network, and Genetic Algorithm: A Case Study in Romantic Decision Support," *Decistion Support Sustems*, vol. 32, no. 4, pp. 361–377, 2002.

[5] M. Y. Zhang, J. R. Russell, and R. S. Tsay, "A Nonlinear Autoregressive Conditional Duration Model with Applications to Financial Transaction Data," *Journal of Economics*, vol. 104, pp. 179–207, 2001.

[6] N. Iqbal, K. Bakhsh, and A. S. Ahmad, "Use of the ARIMA Model for Forecasting Wheat Area and Production in Pakistan," *Journal of Agriculture & Social Sciences*, vol. 1, no. 2, pp. 120–122, 2005.

[7] G. Yu and C. Zhang, "Switching ARIMA Model-Based Forecasting for Traffic Flow," in *Acoustics, Speech, and Signal Processing*, vol. 2, 2004, pp. ii – 429–32.

[8] S. L. Ho and M. Xie, "The Use of ARIMA Models for Reliability Forecasting and Analysis," in *Proceedings of the 23rd International Conference on Computers and Industrial Engineering*, vol. 35, October 1998, pp. 213–216.

[9] J. Contreras, R. Espínola, F. J. Nogales, and A. J. Conejo, "ARIMA Models to Predict Next-Day Electricity Prices," in *IEEE Transactions on Power Systems*, 2003.

[10] T. Caporale and K. Doroodian, "Exchange Rate Variability and the Flow of International Trade," *Economics Letters*, vol. 46, pp. 49–54, 1994.

[11] Universidad Politécnica de Madrid, Facultad de Informática, "A computer System to Control and Forecast Air Pollution," [Online]. Available: http://www.fi.upm.es/?id=tablon&acciongt=consulta1&idet=501, 2010.

[12] S. Asur and B. A. Huberman, "Predicting the Future With Social Media," in *International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM*, Toronto, ON, 2010, pp. 492–499.

[13] J. Ginsberg, M. H. Mohebbi, R. S. Patel, L. Brammer, M. S. Smolinski, and L. Brilliant, "Detecting Influenza Epidemics Using Search Engine Query Data," *Nature*, vol. 457, pp. 1012–1014, 2009.

[14] F. D'Amuri and J. Marcucci, ""Google it!" Forecasting the US Unemployment Rate with a Google Job Search Index," Institute for Social and Economic Research, ISER working papers 2009-32, November 2009.

[15] P. G. Spirakis, V. Vlachos, V. Karakoidas, D. Liappis, D. Kalaitzis, E. Valeontis, S. Kollias, and G. Argyros, "Blueprints for a Large-Scale Early Warning System," in *2010 14th Panhellenic Conference on Informatics (PCI)*, Tripoli, Greece, Septempber 10–12 2010, pp. 7–11.

[16] The WildList Organization International, "WildList," [Online]. Available: http://www.wildlist.org/WildList/.

[17] A. E. Khandani and A. W. Lo, "What Happened to the Quants in August 2007?" [Online]. http://ssrn.com/abstract=1015987, November 4 2007, last revised: January 17, 2008.

[18] T. J. Watsham and K. Parramore, *Quantitative Methods in Finance*. International Thomson Business Press, 1997.

[19] R. Cont, "Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues," *Quantitative Finance*, vol. 1, pp. 223–236, October 2001.

[20] R. F. Engle and J. R. Russell, "Analysis of High Frequency Financial Data," in *Handbook of Financial Econometrics*, Y. Ait-Sahalia and L. Hansen, Eds. Amsterdam: North Holland, 2005, vol. 1.

[21] Microsoft Corporation, "A History of Windows," [Online]. Available: http://windows.microsoft.com/en-US/windows/history, 2010.

A Trustworthy Architecture for Wireless Industrial Sensor Networks

Research roadmap of EU TWISNet Trust & Security project

Markus Wehner, Sven Zeisberg
University of Applied Sciences Dresden
Dresden, Germany
{wehner, zeisberg}@htw-dresden.de

Alexis Olivereau, Nouha Oulha
CEA-LIST
Gif-sur-Yvette, France
{alexis.olivereau, nouha.oulha}@cea.fr

Laura Gheorghe, Emil Slusanschi
University Politehnica of Bucharest
Bucharest, Romania
{dan.tudose, emil.slusanschi}@cs.pub.ro

Basil Hess, Felix von Reischach
SAP
Walldorf, Germany
{basil.hess, felix.von.reischach}@sap.de

Mike Ludwig
Dresden Elektronik Ingenieurtechnik GmbH
Dresden, Germany
mike.ludwig@dresden-elektronik.de

David Bateman
Electricité de France
Paris, France
david.bateman@edf.fr

***Abstract*— Over the past years, the deployment of sensor networks in industrial environments has attracted much attention in several business domains. An increasing number of applications have been developed, ranging from defense, public security, energy management, traffic control to health care. Sensor networks are particularly interesting due to their ability to control and monitor physical environments. Nevertheless, several technical (e.g. remote management, deployment) and security (e.g. user's privacy, data confidentiality and reliability) challenges deter their integration in industrial processes. This paper presents initial research results on an architecture aiming at supporting and securing the integration of sensor networks into large scale industrial environments. This work is carried out in the recently started "TWISNet: Trustworthy Wireless Industrial Sensor Networks" project.**

Keywords-sensor networks, threat analysis, authentication, cybersecurity

I. INTRODUCTION

Over the past years, the deployment of sensor networks in industrial environments has attracted much attention in several business domains. An increasing number of applications have been developed, ranging from defense, public security, energy management, traffic control to health care. Sensor networks are particularly interesting due to their ability to control and monitor physical environments. However, it appears that many security concerns, raised by business applications, have not been properly and efficiently addressed, particularly as far as multi-owner or mobile networks are concerned. This paper presents initial research results of the recently started "TWISNet: Trustworthy Wireless Industrial Sensor Networks" project that aims to support and secure the integration of sensor networks into large scale industrial environments.

Our objective in TWISNet is therefore to develop a platform supporting the integration of sensor networks in an efficient, secure and reliable way, considering the strong technical constraints of sensor networks. For that purpose, we have identified four use cases in the area of nuclear plant facility management, supply and demand energy management, industrial process monitoring and control, and multi-owner environmental monitoring. With these identified use cases we address the major concerns of user's privacy, node authentication in multi-PAN environments, data confidentiality or reliability that we believe will occur in most of the scenarios that could be envisaged using sensor networks. All those security requirements must be fulfilled considering resource constraints on the nodes by means of efficient (e.g. battery, CPU, memory) security and trust mechanisms.

Starting from a brief description of the use cases and business processes that will be addressed in TWISNet, this paper then presents the initial research results on the identified technical enablers. Finally, a description of the expected project outcomes is provided.

II. USE CASES

A. Sensors attached to a person moving from PAN to PAN

In this mobile scenario it is typical that the field-operator might move over several kilometers and between several PANs. Therefore, unless all of the PANs have the same authentication key and the same channel a key concern with this scenario is the process of re-authentication. The remote key provisioning and the security of this provisioning is therefore a key challenge of this scenario. A second key challenge is making the data anonymous in order to preserve workers' privacy. This might be performed through smart

choices in the network architecture that would enable separation of security zones which protect the private information.

B. Sensor networks for supply and demand optimization

Supply and demand energy management appears to be the one of the first large scale (thousands to millions of nodes) sensor network deployment. A key challenge of this scenario is the consumers' privacy preservation, which also raises an additional issue: processing confidential information. Monitoring applications need to be able to make decisions derived from confidential information. Assuming that remote management of customer equipment is possible, this equipment is prone to remote attacks, in particular to DoS (Denial of Service) attacks. Therefore, authentication of the provenance of commands to customer equipment is another strong requirement. Another key challenge of this scenario is the remote management of the authentication key and security model of the customer devices.

C. Sensor networks for the monitoring and control of an industrial process

A key challenge to the deployment of wireless sensors in industrial processes is the assurance of the authenticity of the data from the sensors and to the command and control of the industrial process. It is important to manage and police the access rights of the various users of the sensor network. Another key challenge of this scenario is the remote management and update of the wireless sensor network's security mechanisms, and ensuring that this process cannot be itself used as a means of attacking the network.

D. Multi-owner sensor networks

This scenario uses wireless sensors from multiple different services to form a single network, where a separate network for each service is not viable due to the geographical coverage of the wireless nodes. The authentication to the shared network is a key challenge for this scenario. The users must consider that the wireless sensor network is a shared resource and treat the security requirements of this network accordingly. Another key challenge is devising appropriate means of isolating the services provided by the network for different owners. The rights of the various services to access the resources of the network are also needed to be managed. A third key challenge in this scenario is therefore the resource management, so as to ensure fair distribution of resources between the services.

III. INITIAL RESULTS ON TECHNICAL ENABLERS

The use cases presented in the previous section highlighted several key challenges for sensor networks for secure and trusted data processing. These can be categorized as follows: (1) remote credentials provisioning, (2) fast (re-)authentication, (3) privacy, (4) inter-operator connection sharing, (5) service availability & trustworthiness and (6) adaptive security.

The present section introduces the technical solutions that are being developed to address these identified security challenges. These solutions are organized into six corresponding security services, presented in detail in the following sub-sections.

A. Automatic Configuration and Reconfiguration

In order to ease sensor deployment, a simple system will be designed to allow the initial authentication step to be carried out the first time a sensor is connected to a given PAN, leading to a successful registration of the device. The SCEP [1] protocol is conceptually close, yet technically distant, to what TWISNet aims at designing. It allows a client to remotely acquire an X.509 certificate and may leverage on a long-term shared secret. It however does not handle intermittently connected devices and the X.509 infrastructure would not fit well to the WSN world. On the other hand, the Kerberos protocol distributes short-term authorizations credentials ("tickets") that are required to be based on a long-term security association. Sensor nodes need to be loaded with keys prior to deployment, through their physical or wireless interfaces.

Envisioned devices are likely to be widely spread and possibly in locations that are difficult to access. Furthermore, they are expected to be deployed in very large numbers. Therefore, manual administration should be reduced to a minimum. Instead, TWISNet will provide an interface allowing for secure and remote administration, so that mobile devices can be easily updated. Especially, this interface will allow for the refreshment of expired/compromised cryptographic material (e.g. encryption keys) or authentication parameters. The interface may rely for that on an enhanced version of the µTESLA [2] broadcast authentication protocol that for instance addresses denial of service attacks.

B. Identity Management, Authentication and Access Control

Authentication is an important security requirement that should be addressed to ensure that no malicious node can masquerade as an honest node. The resource-constrained nature of sensor nodes and their mobility emphasize the need for fast re-authentication as the sensors will have to periodically attach to a new PAN, and will be unable to transmit any contextual data. In TWISNet, additional keys, called cluster keys, can be used for authentication, as for LEAP [3]. The cluster key in TWISNet allows a mobile sensor node to pre-authenticate with multiple nodes at the same time. When the sensor node moves, it can rely on such key to establish key paths with new neighboring nodes.

The proposed authentication mechanism in TWISNet should not reveal the real identities of sensor nodes to an eavesdropper. Application-level virtual identities can be used instead. To produce virtual identities, the authentication mechanism may rely on a trusted entity that will be responsible for issuing certificates for nodes (e.g., [4]) before the deployment phase of nodes. Based on the certificate in their possession, nodes can issue multiple self-certified virtual identities. An alternative to this solution may rely on cryptographic algorithms [5] based on low-complexity operations (e.g., hash functions).

C. Shared Information and Ressources

Preserving user's privacy is a major requirement for user acceptance of new solutions being developed. Various sensitive data and processes are generally considered as belonging to the privacy framework, as is described in [6]. We therefore consider location privacy – where mathematical algorithms proposed in literature [7] do not match the reality of the sensor world; and privacy protection of context-related information – for which classical aggregation schemes [8] do not match with the considered industrial scenarios presented in Section 2. To alleviate these problems, we propose dynamic provisioning of virtual identities specifically adapted to wireless sensor networks. In certain situations, true anonymity can be attained. Otherwise, pseudonymity, as described in [9], is used in conjunction with WSN, such that intermittently-connected devices can obtain pseudonyms on time. Finally, software-assisted privacy modules are designed to prevent the user from inadvertently disclosing sensitive information.

Whenever using node resources like bandwidth or computing power to help other nodes, for example by routing packets, may lead to obvious security threats. To alleviate these problems payment-based [10] or reputation-based [6] methods are used as enablers for collaboration enforcements aimed at countering the nodes' selfishness. Therefore, a tradeoff needs to be established between sensor node constraints and the need for per-packet security mechanisms to unequivocally bind packets to its sender. Furthermore, nodes can dynamically play the role of authenticators with the assistance of remote servers which determine the way in which resources are accessed. A lightweight mechanism preventing WSN nodes from misbehaving is to be provided, together with co-management capabilities allowing for the control of nodes owned by more than one party.

D. Availability for Communications, Informations and Services

Trustworthy architectures have to be designed to guarantee certain levels of service quality in the presence of hardware and software damage. Therefore, a service quality assessment system is used to monitor the availability and the security of the network. When network components are accidentally or maliciously damaged, the system pinpoints unsecure nodes and the network availability is updated. Based on the monitoring process, a secure and trusted system ensuring failure anticipation, prevention and detection is to be designed. This covers techniques for predicting failures that are likely to happen, detecting failures, and taking appropriate preventive actions by using data replication, redundant logical entity and similar methods. Such techniques are used in secure and fault-tolerant routing protocols for real-time and critical sensor data.

The failure management system is based on trusted entities, which ensure effective failure detection and recovery, with explicit secure notifications from various trusted entities in the network when a failure is suspected. This system also includes a secure and trusted service recovery system. This is particularly needed in case of permanent component failure. In such a case, techniques for dynamic assignment of new roles to unaffected and trusted entities are investigated to ensure that a minimal service is assured. The service availability problem is also important for the scenario where the service information flows across heterogeneous administrative domains or from a private to a public domain and vice versa. The goal behind this is to ensure that a satisfactory trust level can be established among network components from different administrative domains to ensure an inter-domain end-to-end service availability. Techniques for setting up backup options are considered when service continuity cannot be ensured because of access limitations to some network infrastructures.

E. Adaptive Security

Security adaptation is an important concept in the framework of sensor networking, where scarce resources and quickly changing environments make adaptive mechanisms attractive. While [11] explicitly targets low-resource nodes in a machine to machine environment, it assumes a fixed security architecture with static security roles per entity. TWISNet enhances this by providing support for dynamicity and context dependency in the security services.

F. Secure and Trusted Mediation Layer

Erroneous sensor data can have far reaching consequences, e.g. in remote patient monitoring: a patient is equipped with a Body Sensor Network (BSN) [12] which monitors vital information and activities. Erroneous sensor data may lead to a wrong therapy or even death of this person. Ensuring that the data from sensor networks is trustworthy is a difficult task that can have significant consequences on the sensors battery life [13]. In the case of WSN, ensuring the trustworthiness of the sensors data might even be impossible as an attacker might have physical access to the WSN [14]. For this reason, we are investigating trustworthiness assessment of processed sensor data rather than ensure that the sensor data is implicitly trustworthy. Foreseen challenges beyond the state of the art include algorithms for the detection of misbehaving node [15] or malicious data from sensor nodes [16], trust and reputation systems [17] and development of a trust model from the capture of the data on the sensor to its use in a business process.

Confidentiality of sensor data typically needs end-to-end encryption of the data or at least that the decryption of the data will be handled in a middleware between the WSN and the business process [18]. An important energy saving mechanism within WSN is in-network processing [19], where data from the sensors is aggregated within the WSN itself to reduce the energy consuming data transmissions. This typically requires decryption of the data at the aggregating nodes of the network, a process that introduces a risk to the confidentiality of the data. Foreseen challenges beyond the state of the art include complex (beyond the existing secure statistical network processing [20]) sensor data processing and definition of appropriate security mechanisms for supporting end-to-end secure processing over sensor data at application-level.

IV. PROJECT BENEFITS

The TWISNet project will provide protocols and architectures for the deployment of trustworthy wireless sensor networks in various identified industrial environments and scenarios. For large networks of sensors (1000's of nodes) including mobile sensor networks, all five security aspects (integrity, confidentiality, traceability (and authentication), availability, privacy) will be addressed based on a threat analysis for the proposed scenarios. Privacy preserving identity management will be studied as well as investigations of assurance of security, integrity and availability of data in business processes involving measurements and data collection with wireless sensor networks will be done. Activities will address scenario definition and business process requirements, threat assessments, target security model and technical security solutions to be provided in the form of at least network protocols, architectures and middleware. Finally, testbeds will provide the proof-of-concept that effective security is ensured in wireless sensor networks operating in the above mentioned framework. Security solutions will be designed to accommodate the limited resources of sensors and will also be able to adapt to long periods of sleep (intermittency) and will be adapted to threat levels, and target security level, specific to the industrial applications. Dynamicity will be emphasized in the design of security solutions, so that components can adapt their behavior to their current capabilities and environment. The project will provide generic tools and network solutions for both IP-based and non-IP networks. Solutions capable of managing different trust levels and a mix of secured and non-secured services will be provided. The project will evaluate the performance/security trade-off and ensure that security does not jeopardize performance. Interactions and trade-offs between security and safety will be investigated.

V. CONCLUSION AND OUTLOOK

This paper presents the initial research results of the TWISNet project. Starting from industrial scenarios, security challenges relative to the deployment of sensor networks in this context have been identified. The solutions corresponding to these challenges, briefly presented in this paper, are currently being refined through a methodical threat analysis procedure carried out on each scenario.

These solutions will be further enhanced by implementing their algorithms and protocols on a sensor network prototype, tested in actual industrial conditions. Integrating commercial off-the-shelf or pre-standard devices, this platform will serve as a mediation layer between the sensor network and industrial applications. With a security architecture addressing the major business application security requirements (e.g. user's privacy, data confidentiality, reliability), its suitability will be validated based on the identified use cases. Finally, the scientific and technical outcome of TWISNet will be its contribution to standards, such as IETF 6LoWPAN or ETSI M2M architecture.

This work was financially supported by the EC under grant agreement FP7-ICT-258280 TWISNet project.

REFERENCES.

[1] A. Nourse et al., "Cisco Systems' Simple Certificate Enrollment Protocol", IETF Internet Draft (draft-nourse-scep-18.txt), work in progress, July 2009.

[2] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tygar, "SPINS: Security protocols for sensor networks", In Proc. of 7th Annual Intern. Conf. on Mobile Computing and Networks. 2001.

[3] S. Zhu, S. Setia, S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Sensor Networks", in Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03), Washington D.C., October, 2003.

[4] L. A. Martucci, M. Kohlweiss, C. Andersson, A. Panchenko, "Self-certified Sybil-Free Pseudonyms", in Proceedings of the 1st ACM Conference on Wireless Network Security, WiSec'08. April 2008. Alexandria, VA, USA. p.154-159.

[5] Y. Ouyang, Z. Le, Y. Xu, N. Triandopoulos, S. Zhang, J. Ford, and F. Makedon, "Providing Anonymity in Wireless Sensor Networks", in Proceedings of the IEEE International Conference on Pervasive Services (ICPS'07), pp. 145-148, Istanbul, Turkey, July 15-20, 2007.

[6] P. Michiardi and R. Molva, "Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks", IFIP Comm. and Multimedia Sec. Conference, pp. 107-121, 2002.

[7] P. Golle and K. Partridge, "On the Anonymity of Home/Work Location Pairs", Proceedings of the 7th International Conference on Pervasive Computing; 2009 May 11-14; Nara, Japan. Berlin: Springer; 2009; LNCS 5538: 390-397

[8] A. Kapadia, "AnonySense: Opportunistic and Privacy-Preserving Context Collection", proceedings of the 6th International Conference on Pervasive Computing, May 2008.

[9] Z. Ma et al., "Pseudonym-On-Demand: A New Pseudonym Refill Strategy for Vehicular Communications", Proceedings2nd IEEE Intern. Symposium on Wireless Vehicular Comm., Sept. 2008.

[10] Y. Xi and E. M. Yeh, "Pricing, competition, and routing for selfish and strategic nodes in multihop relay networks", IEEE Conference on Computer Communications, pp. 1463-1471, April 2008.

[11] N. R. Prasad and M. Ruggieri, "Adaptive Security for Low Data Rate Networks", Wireless Personal Comm. 29: pp. 323–350, 2004.

[12] Y. Guang-Zhong. Body Sensor Networks. Springer, 2006.

[13] P. Hamalainen, M. Kuorilehto, T. Alho, M. Hannikainen, and T. D. Hamalainen. Security in wireless sensor networks: Considerations and experiments. pages 167–177, 2006.

[14] W. Zhang, S. Das, and Y. Liu. A trust based framework for secure data aggregation in wireless sensor networks. in the Proceedings of the IEEE SECON, September 2006.

[15] A. Tanenbaum and M. V. Steen. Distributed Systems: Principles and Paradigms. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[16] Q. Zhang, T. Yu, and P. Ning. A framework for identifying compromised nodes in sensor networks. In Proceedings of the IEEE SecureComm, August 2006.

[17] W. Zhang, S. Das, and Y. Liu. A trust based framework for secure data aggregation on wireless sensor networks. in the Proceedings of the IEEE SECON, September 2006.

[18] A. Agostini, C. Bettini, and D. Riboni. Loosely coupling ontological reasoning with an efficient Middleware for contextawareness. The Second Annual International Conference Mobile and Ubiquitous Systems: Networking and Services, 2005.

[19] L. Gomez, L. Odorico, A. Sorniotti, and K. Wrona. Secure and trusted in-network data processing in wireless sensor networks. Journal of Information Assurance and Sec., 2007.

[20] Tassos Dimitriou, Dimitris Foteinakis, "Secure and Efficient In-Network Processing for Sensor Networks", Third International Conf. on Sensor Technologies and Applications, 2009

Mapping Systems Security Research at Chalmers

M. Almgren, Z. Fu, E. Jonsson, P. Kleberger, A. Larsson,
F. Moradi, T. Olovsson, M. Papatriantafilou, L. Pirzadeh, and P. Tsigas
Department of Computer Science and Engineering
Chalmers University of Technology
SE–412 96 Gothenburg, Sweden

***Abstract*—The department of Computer Science and Engineering at Chalmers University has a long tradition of research in systems security, including security metrics, attack detection and mitigation. We focus on security issues arising in four specific environments: (1) backbone links, (2) sensor networks, (3) the connected car, and (4) the smart grid. In this short summary we describe recent results as well as open research questions we are exploring.**

I. Introduction

At the department of Computer Science and Engineering at Chalmers University, there is a long tradition of research in systems security.[1] More than two decades ago, we started to look at *security metrics* and modeling and today the research include attack detection and alert correlation as well as mitigation of, for example, Denial of Service attacks. We also have on-going projects focusing on systems security issues in four *specific environments*: (1) backbone links, where both efficiency of the algorithms as well as user privacy is of concern, (2) sensor networks with each node being limited in its capabilities, (3) the connected car, and (4) the smart grid. These areas will be further described below.

II. Security Metrics and Modeling

It has been claimed that going from qualitative to quantitative aspects is the way of progress for a scientific discipline [1]. The ultimate conclusion of this should be that science is not *real* science until it can be assessed in a quantitative way, i.e. measured. In particular, for security-related areas we will not be able to evaluate scientific progress properly until we can find metrics for it, including giving proper definitions and a clear-cut terminology.

In this way, our research in security metrics establishes a foundation for other research efforts within the department. The research effort started over two decades ago [2–4] and one notable result is a classification of intrusions with respect to technique as well as to result [5], derived from the traditional decomposition of security into three main aspects ("CIA").

There exists a large number of suggestions for how to measure security, with different goals and objectives. In many cases the goal is to find a *single overall* metric of security. However, given that security is a complex and multi-faceted property, we believe that there are fundamental problems in finding such an overall metric. Thus, we are currently developing a framework for security metrics that is based on a number of system attributes taken from the security and the dependability disciplines [6]. Having metrics related to different types of attributes facilitates making quantitative assessment of the concept of combined security and dependability and improves our understanding of the underlying system properties.

[1] Other types of security research at the department, such as *language-based security*, are not included in this summary.

III. Attack Detection and Protection Mechanisms

A. Intrusion Detection and Logging

It is difficult to build secure systems and, sometimes, legacy or operational constraints do not even allow the systems to be run in a secure fashion. The goal of an intrusion detection system (IDS) is to detect active misuse and attempts, either by legitimate users ("insiders") or by external parties. Since the seminal paper by Denning [7], intrusion detection systems have seen a tremendous development in the type of data collected, the analysis, as well as the user interface [8]. However, these systems are still hampered by fundamental problems, such as the base-rate fallacy [9].

In our projects, we have looked at a range of issues relevant to intrusion detection systems. For example, it is critical to *log* the right type of data [10, 11], as well as being able to extract *attack manifestations* [12] in an efficient manner. An IDS needs data both to measure how well it is performing and to automatically learn to discriminate between attacks and normal behavior. For that reason we have presented a synthesized dataset for fraud detection systems [13] as well as investigated methods to reduce the amount of training data needed through the use of *active learning* [14]. We have also investigated complementary methods to collect data for an IDS [15] and proposed a multi-sensor model to improve the attack detection when using different types of sensors [16].

Currently, we are adapting some of the techniques described above to the *special environments* described below.

B. Mitigating Denial of Service Attacks

An important aspect of security for emergency preparedness is the availability of systems. We study methods to protect the network and applications against denial of service (DoS) attacks, i.e. attacks that overwhelm the system so that the normal requests cannot be answered. Attacks can be addressed in application-level and network-level.

Along the former type and by considering adversaries that can eavesdrop and launch directed DoS attacks to the

applications' open ports, solutions based on pseudo-random port-hopping have been suggested [17]. In [18] we proposed a general method that can also be used for a group of processes, and not just a client-server pair, as was the case in the earlier work. In addition, our proposed solution tolerates time differences (in particular clock-drifts) between the nodes, which was earlier not known how to achieve.

DDoS attacks, i.e. distributed DoS attacks, are challenging not only for the targets of the attacks, but also for the network, as large volume of illegitimate traffic share the same network resources as legitimate traffic and can furthermore cause congestion phenomena and performance degradation. To mitigate that, the unwanted traffic needs to be controlled as close to the source(s) as possible. By building on earlier work and improving on distribution of control aspects, we proposed a proactive cluster-based method, which we call CluB , to mitigate DDoS attacks [19]. The method balances the effectiveness-overhead tradeoff by addressing the issue of granularity of control in the network. CluB can collaborate with different routing policies in the network, including contemporary datagram options. We have also studied ways to improve methods that use tokens to distinguish legitimate traffic. Our algorithm [20] reduces the effect of a particular form of attack (denial-of-capability, which applies to token-based methods for mitigation) [21]. With this algorithm, the legitimate hosts can get service with guaranteed probability.

As the above methods are complementary, we plan to continue on both approaches and to also study methods for integrating them. In particular, we are working on methods to adapt the port-hopping solutions [18], which are application-centered, to overlay networks, which are specialized networks defined and maintained by distributed applications (e.g. online social networks), with access control [19, 20]. Taking the Internet perspective, such overlays can be defined among participating routers, which may cooperate to achieve secure routing and to mitigate DDoS attacks.

IV. Focus on Specific Environments

A. Large-Scale Internet Backbone Traffic Analysis

Access to real-life large-scale datasets is in many cases crucial for understanding the true characteristics of network traffic, application behavior, and malicious behavior. However, the collection and the subsequent analysis of these datasets pose some special requirements. For example, *user privacy* is very important, and thus the data needs to be desensitized before being analyzed, a process that may influence the type of analysis method that can be used. The scale of the data also affects the collection of data and the analysis.

We have collected several large-scale datasets in a number of passive measurement projects on an Internet backbone link belonging to a national university network. The datasets have been used in different studies as part of the following projects.[2]

[2]More information about the data collection process and the collected datasets can be found in [22].

As part of the *MonNet project*, Internet backbone traffic was investigated to find malicious traffic in order to see how and to what extent protocols are abused. Initial studies investigated protocol features of packet headers [23] and packet header anomalies in order to discuss potential security problems, such as incorrect use of IP fragmentation [24].

The objective of the *Malbone project* is to measure and understand larger communication patterns among hosts over a longer time period. This may include normal as well as malicious behavior. Analysis in [25] spans from simple attribute aggregates (such as top IP and port numbers) to advanced temporal analysis of communication patterns between normal and malicious hosts.

The final project, the *AntiSpam project*, is focused on the problem of spam or unsolicited email. Email is probably one of the most popular application on the Internet, but spam is an increasing problem and has been estimated to cost businesses significant amounts of money. Current antispam tools are limited in that they only hide the spam from users' mailboxes. Therefore, we want to move the defense against spam as close to the spammers as possible in order to reduce problems such as the amount of unwanted traffic and waste of mail server resources. We are currently investigating spam detection through a social network based analysis (first proposed in [26]). Using e-mail addresses as nodes and letting edges symbolize any e-mail exchange, we have generated "email networks" using anonymized collected email traffic [27]. By focusing on structural and temporal properties of such networks, we have found several properties that are statistically different for spam and legitimate traffic. Deployment of these distinguishing characteristics for detection of spammers at the network level without a need to consult email contents is the subject of our ongoing research.

B. Sensor Networks

There are many promising application areas for wireless sensor networks. The possibilities spans areas as civil security, health care, agriculture, research, environmental, commercial and military applications [28]. Security is critical for many applications of sensor networks, both due to sensitivity of data and the need to remain functional in presence of attacks. Wireless sensor networks come with additional security challenges, in large due to hardware limitations and the wireless communication medium [29, 30]. Malicious insider nodes are a serious threat due to the physical access of the nodes [31].

There are many services, several building upon each other, that are needed for wireless sensor network applications. Our aim is to provide such high level networking protocols for sensor networks and/or ad-hoc networks that are both secure and self-stabilizing. Self-stabilization lets nodes recover from arbitrary faults once conditions are back to normal. We take into account the serious threat of compromised nodes inside the network.

Accurate clock synchronization is imperative for many applications in sensor networks, such as mobile object tracking, detection of duplicates, and TDMA radio scheduling. In [32],

we presented the first secure and self-stabilizing algorithm for clock synchronization in sensor networks. Clustering organizes a network into groups that, e.g., can be used for forming backbones, for routing, for aggregating data, and for building hierarchies that allow for scaling. In [33], we present the first self-stabilizing (k, r)-clustering algorithm for ad-hoc networks providing k cluster heads within r communication hops. Multiple paths are used to improve security, availability and fault tolerance. In [34] we provided the first security module providing symmetric key cryptography for the Contiki wireless sensor network operating system. We have also looked at the areas of routing and public key cryptography.

Going forward, we aim to secure additional fundamental network services. Routing is needed in any sensor network application that does not merely store sensor readings locally. Thus, to set up a secure sensor network, secure routing is one such needed service. Combining different protocols together into a secure and fault tolerant package for increased efficiency and ease of use would be fruitful. Additionally, it could cut down costs if different services could share mechanisms with each other and thus reduce the total amount of needed calculations and/or messages.

C. Securing the Connected Car

An upcoming trend in the automotive industry is to equip the vehicle with a wireless network gateway, enabling the vehicle to connect to an external network (i.e. Internet). The benefits from introducing such a connection are many, not only will there be new applications introduced for the driver and passengers, but there will also be a new possibility of performing remote diagnostics and issuing remote firmware updates over the air (FOTA) to the vehicle.

Introducing the connected car, communication with the Electronic Control Units (ECUs) in the in-vehicle network will be possible through a wireless network gateway. This communication will no longer require physical access to the vehicle and may be performed at any time. The wireless gateway may also be used for taking part in the emerging Vehicle-to-Vehicle (V2V) communication networks, where vehicles can exchange information with each other to, for example, increase traffic safety. Since the vehicle is a safety-critical system, and to ensure that the new external network traffic introduced in the in-vehicle network will not be a threat to the safety nor the security of the vehicle, necessary security mechanisms need to be in place. It has recently been shown that such mechanisms are still lacking for the vehicle setting [35].

Our research is focused towards securing the in-vehicle network and the communication with the connected car, so that services to future vehicles can be provided in a secure and safe manner. One of the main research focus is to provide a secure infrastructure for remote diagnostics and software updates over a wireless link. In [36] we presented a set of guidelines for such a wireless infrastructure.

A defense-in-depth approach to address the security needs has also been proposed, where we look at methods for prevention, detection, deflection and forensics [37]. Furthermore, the Controller Area Network (CAN) and FlexRay-protocols used in the in-vehicle network has been evaluated with respect to a set of security properties [38, 39].

Some general challenges for applying security mechanisms to the connected car are the limited resources available in processing power and memory, cost sensitivity and the lifetime of the solution as the vehicle can be used for many years.

A complete security architecture for the connected car is still missing, and we intend to continue contributing in defining one.

D. Security Issues in the Smart Grid

The Smart Grid is being promoted on both sides of the Atlantic as the way to solve problems in energy production, distribution, and consumption in the future. The definition of what the smart grid exactly will entail varies depending on perspective, but its main idea is to allow two-way communication of both power and data between devices, thus allowing for a more adaptive and effective way to utilize energy. However, a documented consequence is that new vulnerabilities are appearing[3] and some "features" have large security implications [40]. Given that electricity is required for many other critical services in society, any security vulnerability within this software-intensive critical system will attract attention from hostile groups or organized crime.

We have investigated open security issues in a wide range of critical systems [41] and are currently looking especially at the issues within the smart grid [42]. Among the security challenges of the smart grid is the sheer scale of the deployment and that any vulnerability may have a very large impact on society as a whole. Among our recent work, we have considered the *optimal power flow* (OPF) problem as a minimum cost flow and applied a cost-scaling push-relabel algorithm in order to solve the OPF in a distributed agent environment [43]. We are also investigating issues related to the advanced metering infrastructure (AMI).

V. Conclusion

In this short summary, we have described the research related to systems security at the department of Computer Science and Engineering at Chalmers University. We have focused on current projects but also included a discussion of research topics we are actively exploring.

Acknowledgment

This work was supported by the Swedish Civil Contingencies Agency (MSB), the .SE The Internet Infrastructure Foundation, and the SIGYN II project (2009-01722) co-funded by VINNOVA, the Swedish Governmental Agency for Innovation Systems. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 257007.

[3] http://edition.cnn.com/2009/TECH/03/20/smartgrid.vulnerability/index.html

REFERENCES

[1] M. Bunge, *Scientific Research. Strategy and Philosophy.* Berlin: Springer-Verlag, 1967.

[2] E. Jonsson, M. Andersson, and S. Asmussen, "An attempt to quantitative modeling of behavioural security," in *Proc. 11th International Information Security Conference (IFIP/SEC'95), Cape Town, South Africa*, 1995.

[3] E. Jonsson, L. Strömberg, and S. Lindskog, "On the functional relation between security and dependbility impairments," in *ACM New Security Paradigms Workshop, (NSPW 1999), Caledon Hills, Canada*, Sep. 1999.

[4] S. Brocklehurst, B. Littlewood, T. Olovsson, and E. Jonsson, "On measurement of operational security," in *COMPASS '94, Proceedings of the Ninth Annual Conference on Computer Assurance*. Gaithersburg: IEEE Computer Society, 1994, pp. 257–266, ISBN 0-7803-1855-2.

[5] U. Lindqvist and E. Jonsson, "How to systematically classify computer security intrusions," in *Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on*, May 1997, pp. 154 –163.

[6] E. Jonsson, "Towards an integrated conceptual model of security and dependability," in *Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on*, Apr. 2006, p. 8 pp.

[7] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, Feb. 1987.

[8] M. Almgren and E. Jonsson, "Tuning an IDS – learning the user's preferences," in *11th Nordic Workshop on Secure IT Systems (NordSec 2006)*, V. Fåk, Ed. Linköping university, Sweden: Published by Linköping university, Sweden, Oct. 19–20, 2006, pp. 43–52.

[9] S. Axelsson, "The base-rate fallacy and its implications for the difficulty of intrusion detection," in *Proceedings of the 6th ACM Conference on Computer and Communications Security*, Kent Ridge Digital Labs, Singapore, November 1999.

[10] S. Axelsson, U. Lindqvist, U. Gustafson, and E. Jonsson, "An approach to UNIX security logging," in *Proceedings of the 21st National Information Systems Security Conference*. Arlington, Virginia: National Institute of Standards and Technology/National Computer Security Center, Oct. 5–8, 1998, pp. 62–75.

[11] E. Lundin Barse, "Logging for intrusion and fraud detection," Ph.D. dissertation, Chalmers University of Technology, 2004.

[12] U. Larson, E. Lundin Barse, and E. Jonsson, "METAL - a tool for extracting attack manifestations," in *Proceedings of Detection of Intrusions and Malware & Vulnerability Assessment workshop (DIMVA)*, Vienna, Austria, July 7-8 2005.

[13] E. Lunding Barse, H. Kvarnström, and E. Jonsson, "Synthesizing test data for fraud detection systems," in *19th Annual Computer Security Applications Conference (ACSAC '03)*. Published by the IEEE Computer Society, 2003.

[14] M. Almgren and E. Jonsson, "Using active learning in intrusion detection," in *17th IEEE Computer Security Foundations Workshop (CSFW 2004)*. Asilomar, USA: IEEE Computer Society, Jun. 28–30, 2004, pp. 88–98.

[15] M. Almgren and U. Lindqvist, "Application-integrated data collection for security monitoring," in *Recent Advances in Intrusion Detection (RAID 2001)*, ser. LNCS, W. Lee, L. Mé, and A. Wespi, Eds., vol. 2212. Davis, California: Springer-Verlag, Oct. 10–12, 2001, pp. 22–36.

[16] M. Almgren, U. Lindqvist, and E. Jonsson, "A multi-sensor model to improve automated attack detection," in *Recent Advances in Intrusion Detection (RAID 2008)*, ser. LNCS, R. Lippmann, E. Kirda, and A. Trachtenberg, Eds., vol. 5230. Cambridge, MA, USA: Springer-Verlag, Sep. 15–17, 2008, pp. 291–310.

[17] G. Badishi, A. Herzberg, and K. Idit, "Keeping denial-of-service attackers in the dark," *Dependable and Secure Computing, IEEE Transactions on*, vol. 4, no. 3, pp. 191 –204, 2007.

[18] Z. Fu, M. Papatriantafilou, and P. Tsigas, "Mitigating distributed denial of service attacks in multiparty applications in the presence of clock drifts," *Reliable Distributed Systems, 2008. SRDS'08. IEEE Symposium on*, pp. 63–72, 2008.

[19] Z. Fu, , M. Papatriantafilou, and P. Tsigas, "Club: A cluster based method for mitigating distributed denial of service attacks," in *26th Annual ACM Symposium On Applied Computing*. ACM Press, 2011.

[20] Z. Fu, M. Papatriantafilou, P. Tsigas, and W. Wei, "Mitigating denial of capability attacks using sink tree based quota allocation," *Proceedings of the 2010 ACM Symposium on Applied Computing*, pp. 713–718, 2010.

[21] K. Argyraki and D. Cheriton, "Network capabilities: The good, the bad and the ugly," *ACM HotNets-IV*, 2005.

[22] F. Moradi, M. Almgren, W. John, T. Olovsson, and P. Tsigas, "On collection of large-scale multi-purpose datasets on internet backbone links," in *Workshop on development of large scale security-related data collection and analysis initiatives (BADGERS 2011)*, 2011.

[23] W. John and S. Tafvelin, "Analysis of internet backbone traffic and header anomalies observed," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, ser. IMC '07, 2007, pp. 111–116.

[24] W. John and T. Olovsson, "Detection of malicious traffic on backbone links via packet header analysis," *Campus-Wide Information Systems*, vol. 25, no. 5, 2008.

[25] M. Almgren and W. John, "Tracking malicious hosts on a 10gbps backbone link," in *15th Nordic Conference in Secure IT Systems (NordSec 2010)*, 2010.

[26] P. O. Boykin and V. P. Roychowdhury, "Leveraging social networks to fight spam," *Computer*, vol. 38, no. 4, 2005.

[27] F. Moradi, T. Olovsson, and P. Tsigas, "Analyzing the social structure and dynamics of e-mail and spam in massive backbone internet traffic," Chalmers University of Technology, no. 2010-03, Tech. Rep., 2010.

[28] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102 – 114, Aug. 2002.

[29] E. Shi and A. Perrig, "Designing secure sensor networks," *Wireless Communications, IEEE*, vol. 11, no. 6, pp. 38 – 43, dec. 2004.

[30] X. Chen, K. Makki, K. Yen, and N. Pissinou, "Sensor network security: a survey," *Communications Surveys Tutorials, IEEE*, vol. 11, no. 2, pp. 52 –73, 2009.

[31] ——, "Node compromise modeling and its applications in sensor networks," in *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*, Jul. 2007, pp. 575 –582.

[32] J.-H. Hoepman, A. Larsson, E. M. Schiller, and P. Tsigas, "Secure and self-stabilizing clock synchronization in sensor networks," in *Prooceedings of the 9th Scandinavian Workshop on Wireless Ad-hoc Networks (Adhoc 2009)*, 05 2009, pp. 78 – 82.

[33] A. Larsson and P. Tsigas, "Self-stabilizing (k,r)-clustering in wireless ad-hoc networks with multiple paths," in *OPODIS'10, 14th International Conference On Principles Of Distributed Systems*, Tozeur, Tunisia, December 2010.

[34] L. Casado and P. Tsigas, "Contikisec: A secure network layer for wireless sensor networks under the contiki operating system," in *Proceedings of the 14th Nordic Conference on Secure IT Systems: Identity and Privacy in the Internet Age*, ser. NordSec '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 133–147.

[35] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental Security Analysis of a Modern Automobile," in *Proceedings of the 31st IEEE Symposium on Security and Privacy (SP)*, 2010, pp. 447–462.

[36] D. K. Nilsson, U. E. Larson, and E. Jonsson, "Creating a Secure Infrastructure for Wireless Diagnostics and Software Updates in Vehicles," in *Proceedings of the 27th International Conference on Computer Safety, Reliability, and Security (SAFECOMP)*, ser. LNCS, vol. 5219. Newcastle upon Tyne, UK: Springer-Verlag, Sep. 2008, pp. 207–220.

[37] D. K. Nilsson and U. E. Larson, "A Defense-in-Depth Approach to Securing the Wireless Vehicle Infrastructure," *Journal of Networks*, vol. 4, no. 7, pp. 552–564, Sep. 2009.

[38] ——, "Simulated Attacks on CAN Buses: Vehicle Virus," in *Proceedings of the 5th IASTED Conference on Communication Systems and Networks (ASIACSN)*. Langkawi, Malaysia: IASTED, Apr. 2-4 2008.

[39] D. K. Nilsson, U. E. Larson, F. Picasso, and E. Jonsson, "A First Simulation of Attacks in the Automotive Network Communications Protocol FlexRay," in *Proc. of the 1st International Workshop on Computational Intelligence in Security for Information Systems (CISIS)*. Springer, 2008.

[40] F. Cohen, "The smarter grid," in *Proceedings of IEEE Symposium on Security and Privacy*, vol. 8, 2010, pp. 60–63.

[41] FORWARD Consortium, "Forward white book: Emerging ict threats," http://www.ict-forward.eu/media/publications/forward-whitebook.pdf, Jan. 2010.

[42] SysSec Consortium, "Syssec: Managing threats and vulnerabilities in the future internet," http://www.syssec-project.eu, Mar. 2011.

[43] P. Nguyen, W. Kling, G. Georgiadis, M. Papatriantafilou, and L. Bertling, "Distributed routing algorithms to manage power flow in agent-based active distribution network," in *Innovative Smart Grid Technologies Conference Europe (ISGT Europe), 2010 IEEE PES*, 2010.

Exploring the Landscape of Cybercrime

Zinaida Benenson, Andreas Dewald, Hans-Georg Eßer, Felix C. Freiling, Tilo Müller, Christian Moch,
Stefan Vömel, Sebastian Schinzel, Michael Spreitzenbarth, Ben Stock, Johannes Stüttgen
Friedrich-Alexander-University Erlangen-Nuremberg, Germany
www1.informatik.uni-erlangen.de

***Abstract*—This document gives an overview over current research within the security group at Friedrich-Alexander-University Erlangen-Nuremberg, Germany, and attempts to describe the future research roadmap of the group. This roadmap is structured around the *landscape of cybercrime* with its three main groups of actors (attackers, users and investigators) and their main activities and deficits: attack and evasion for attackers, awareness and education for victims, evidence extraction and analysis for investigators.**

I. Introduction

Cybercrime is a growing phenomenon, however also one that still waits to be fully understood [1], [2]. Roughly speaking, cybercrime is crime in cyberspace, where cyberspace is a social space whose infrastructure is formed by digital internetworked computers.

The amount of crime involving digital systems is steadily increasing. This involves both more traditional crime in which digital systems are merely used as tools (e.g., different types of fraud, blackmailing, hidden communication) as well as new forms of crime in which digital systems are an enabling technology (e.g., computer abuses, malicious software, malicious remote control networks like botnets). Both forms of cybercrime correspond to two types that have been identified in the literature [2], [3]: crime that is more oriented towards people (e.g., cyberstalking) and crime that is more oriented towards computers. We focus on the latter type of cybercrime which is far from being well understood.

Much of cybercrime can be characterized by an economic motivation where cybercriminals "hack for profit", thereby forming an underground economy of considerable size [4], [5]. This, however, makes them behave in a rather rational manner and contrasts cybercrime from notions like cyberwar or cyberterrorism in which crimes either do not make a cost/benefit caluclation or are politically motivated.

This document aims at describing the current and future research roadmap of the security research group at Friedrich-Alexander-University Erlangen-Nuremberg, Germany. The major part of this group was affiliated previously with the University of Mannheim, Germany. The relocation to Erlangen gave us the possibility to re-focus our research agenda and form a new joint group vision which we wish to communicate to the systems security research community.

Our vision is structured around the *landscape of cybercrime* with its three main groups of actors (attackers, users and investigators) and their main activities and deficits:

1) attack and evasion for attackers,
2) awareness and education for users,
3) evidence extraction and analysis for investigators.

II. The Landscape of Cybercrime

As mentioned in the introduction, cybercrime is crime that happens in cyberspace. Cyberspace is here understood as the "digital world" in which many people spend a non-negligible part of their daily life. For simplicity, we visualize cyberspace as an unstructured but clearly distinguishable realm in the landscape of cybercrime (see Fig. 1).

A. Actors

Since crime always relates to the physical world, there is no crime that happens *entirely* in cyberspace. There are always humans that act or are acted upon. In this context, we identify three groups of actors in cyberspace:

1) Attackers: Humans that act in an offensive manner, i.e., practice attack and evasion techniques. Such humans can be cybercriminals or security researchers who impersonate the role of adversaries in order to test certain computer systems for weaknesses (penetration testing).
2) Users: Humans who are acted upon and suffer from the actions of the first group. People belonging to this group are often called victims.
3) Investigators: Humans who try to understand and investigate the activities of the two previous groups. These people can be thought of as security researchers from academia or investigators of law enforcement agencies.

There is no sharp distinction between these three groups. For example, security researchers can belong to all three groups, depending on what they are doing. Although we think that ethics are an important topic, note that in our distinction we try to avoid *malicious intent* as a defining attribute of any of these groups. Nevertheless, any scientific activity should be governed by ethical considerations. This is especially crucial if scientists play the role of attackers, as we explain below.

B. Activities

The classification of the groups described above implies a characteristic set of activities for each participant that we will outline in more detail in the following (see Fig. 1).

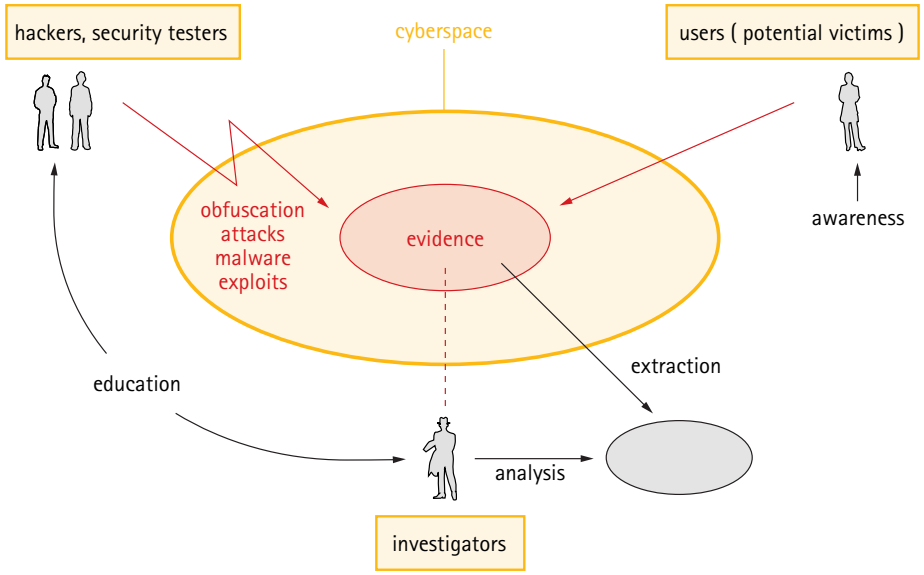


Figure 1. The Landscape of Cybercrime with its Actors.

1) Attackers: Attackers practice offensive thinking, i.e., they look at systems with the intent to break (into) the system. This can result in penetration or security testing, employing techniques like fuzzing or source code analysis, identifying vulnerabilities, and finally producing exploits that can be used to automate attacks with the help of malicious software. Such software must again be protected by evasion or obfuscation methods that may lead to attacks which are hard to detect and mitigate.

2) Users: Users "use" networked digital systems. Depending on their proficiency, they (should) practice reasonable conduct in cyberspace and employ attack detection and attack mitigation techniques (antivirus software, intrusion detection, cryptography, etc.). In case of security incidents, they should also practice basic incident response activities. All this depends on the education and awareness of the users.

3) Investigators: Investigators (pro)actively investigate security incidents in cyberspace. They prepare for incident response through training, they collect evidence created by attackers and users, the analyze said evidence and try to find out what actually happened. Typical activities of investigators are reverse engineering, logfile analysis, development of incident response and digital forensics tools, as well as documentation of their activities.

III. Current and Future Research Areas

We now present relevant research areas which are important to us in current and future research. We structure these areas according to the classes of actors described above.

A. Develop and Cultivate Offensive Technologies

This area corresponds to the class of attackers described above. We now describe current research we are currently exploring.

1) Current research activities: Penetration testing is a widely used approach to assess the security of real-world systems by attacking them. The penetration tester tries to actively push the system to an insecure state by accessing it in a way that was not foreseen by the system's designer. Interestingly, there is little work that passively monitors a system and infers secret information from the behavior of the system. These passive attacks are well known in cryptology under the name "side channnel attacks", but little known for large systems such as business applications. We investigate the following research questions [6]:

- How to detect side channel information leaks in large software systems?
- How to decode the information leaked through a side channel?
- How to prevent and mitigate information leakage through side channel vulnerabilities?

We also investigate anti-forensic techniques to protect data on disk or in memory. This includes our work on AESSE [7], a system for memory analysis resistant [8]–[10] disk encryption. In this context we also look at obfuscation techniques that extend the capabilities of current malware to withstand reverse engineering.

We also extend the global malware analysis system from the InMAS project [11] and make it accessible through the website mwanalysis.org. The website offers a dynamic malware analysis based on CWSandbox [12] and is a source for a large set of new and interesting malware.

2) Future research directions: Future research directions that interest us are the following:

- Speculate and validate future malware techniques, e.g.,

in the areas of obfuscation and anti-reverse-engineering. As examples we wish to investigate RAM encryption based on AESSE [7] to improve memory analysis resistance.
- Building systems that resist other specific attacks on disk encryption, e.g., bootkit attacks (stoned bootkit, evil maid attacks).
- Attacking non-standard hardware like real-world sensor network systems [13]. Wireless sensor networks are already being used in such critical domains as monitoring of offshore oil rigs and such networks often turn out to be quite sloppily specified and prorammed.

B. Awareness and Education

This area corresponds to the class of users described above.

1) Current research activities: We investigate issues of the psychology of security. Technical means for achieving IT security have been steadily improving over the decades. Therefore, the main weak point in securing computer systems is shifting from the technology to the psychology [14], [15]. We examine what people think and feel about computer security and why they think and feel this way. The main goals of this research is to understand why people cannot use the current systems in a secure way.

Another aspect of our current research is usable security. Currently, users perceive security mainly as an interruption of their primary tasks [16], [17]. The main goal of our research on usable security is to find out how to make security not a nuisance but a service. How can security goals and methods be communicated in an appealing and understandable way? How can people develop a feeling for security and insecurity in the digital world?

2) Future research directions: In future research we wish to improve research-orientation in training and education so that users (and investigators) are not so much restricted by their tools. We also develop a specifically offensive education curriculum for undergraduate and graduate students.

C. Foundations of Forensic Computing

Forensic computing (sometimes also called *digital forensics*, *computer forensics* or *IT forensics*) is a branch of forensic science pertaining to evidence in cyberspace. Forensic computing aims at identifying, preserving and analyzing digital evidence after a security incident has occurred. As in other forensic sciences, investigators attempt to establish hypotheses about previous actions and try to falsify them based on traces of actions left at the scene of the crime.

Like in other forensic sciences, the emergence of forensic computing was mainly driven by practitioners trying to satisfy immediate needs within concrete digital investigations. Now that many universities, mainly from North America, have started to establish degree programs and research labs in this area, forensic computing is increasingly profiting from research knowledge and the scientific methods developed in computer science, but there is still a lot of potential [18].

1) Current research activities: The current research topics of the group encompass the following activities.

An important benefit of science's participation in digital forensics is the insight that digital forensics has much more in common with traditional forensics than its pioneers assumed. From this perception, we form a foundation of forensic computing and develop new approaches based on the experience of traditional forensic science [19].

In this context we develop tools and techniques in evidence collection and evidence analysis, for example:

- We examine traces of volatile information in main memory. These approaches complement persistent data-oriented techniques and may be of indispensable help when dealing with encrypted disks or sophisticated types of malicious applications that solely reside in RAM. For this purpose, we evaluate existing frameworks for memory acquisition and analysis (e.g., Volatility [20]) and extend their functionality.
- We develop a tool called ADEL [21] for the analysis of smartphones with a major focus on Google's Android platform. ADEL is able to dump and analyze SQLite databases from a connected smartphone.
- We are also developing a technique called *selective imaging*, which is the creation of partial forensic images by selectively acquiring only relevant data from digital devices [22]. While selective imaging has already been researched on a per-file basis [23], we work on achieving arbitrary granularity of selection to enable the application of this technique, even in complicated cases. The resulting evidence containers require accurate provenance documentation [24] and precise verification procedures, which we are currently developing to achieve the same level of reliability as with common sector-wise images [25].

We also investigate techniques to better educate and train investigators, taking the foundations of forensic computing as the primary basis. For this, we are using the Forensic Image Generator Generator tool (Forensig^2) [26]. This is a tool to reduce the creation time of an artificial forensic image to a minimum without losing the "ground truth" of the image content. Therefore the author of the image has to write a script describing the artificial image. The scripting language is similar to Python.

The primary aim of this tool is for education purposes, generating artificial images for apprentice forensic investigators. However, the use of the tool is not limited to this scope, it is also a very handy tool for open research questions. The ability to generate huge amounts of similar but not identical images makes it possible to test the difficulty of a certain forensic problem, to quantify the knowledge of an

investigator, to evaluate different teaching approaches, and to answer many more unanswered research questions.

2) Future research directions: Future research directions that interest us are the following:

- Analysis of "non-standard" digital technologies (Solid State Disks, Flash Memory, SCADA systems, sensor networks, Firewire, Thunderbolt, etc.)
- Investigate fundamental tradeoffs in technically unavoidable evidence. The example of caches (in their many forms) shows that there is a tradeoff between performance (using a cache) and not creating evidence (not using a cache).
- The general question of quantification and empirical research of correlations between evidence and actions is still largely open.
- Developing new types of reverse engineering approaches for larger software systems like complex applications (e.g., relating certain actions to evidence) together with the software (re-)engineering community.

REFERENCES

[1] D. Wall, *Cybercrime*. Cambridge: Polity Press, 2007.

[2] S. W. Brenner, *Cybercrime: criminal threats from cyberspace*. Santa Barbara: Praeger, 2010.

[3] S. Gordon and R. Ford, "On the definition and classification of cybercrime," *Journal in Computer Virology*, vol. 2, no. 1, pp. 13–20, 2006.

[4] R. Thomas and J. Martin, "The underground economy: Priceless," *The USENIX Magazine*, vol. 31, no. 6, pp. 7–16, 2006.

[5] J. Spoenle, "Underground economy," in *Current Issues in IT Security*, M. Bellini, P. Brunst, and J. Jähnke, Eds. Berlin: Duncker & Humblot, 2010, pp. 67–79.

[6] S. Schinzel and F. C. Freiling, "Detecting hidden storage side channel vulnerabilities in networked applications," in *Proceedings of IFIP SEC 2011*, 2011.

[7] T. Müller, A. Dewald, and F. Freiling, "AESSE: A Cold-Boot Resistant Implementation of AES," in *Proceedings of the Third European Workshop on System Security (EUROSEC)*. Paris, France: ACM, Apr. 2010, pp. 42–47.

[8] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten, "Lest We Remember: Cold Boot Attacks on Encryptions Keys," in *Proceedings of the 17th USENIX Security Symposium*, Princeton University. San Jose, CA: USENIX Association, Aug. 2008, pp. 45–60.

[9] B. D. Carrier and J. Grand, "A Hardware-Based Memory Acquisition Procedure for Digital Investigations," *Digital Investigation* , vol. 1, no. 1, pp. 50–60, Feb. 2004.

[10] M. Becher, M. Dornseif, and C. N. Klein, "FireWire - All Your Memory Are Belong To Us," in *Proceedings of the Annual CanSecWest Applied Security Conference*. Vancouver, British Columbia, Canada, 2005.

[11] M. Engelberth, F. C. Freiling, J. Göbel, C. Gorecki, T. Holz, R. Hund, P. Trinius, and C. Willems, "The InMAS Approach," in *Proceedings 1st European Workshop on Internet Early Warning and Network Intelligence (EWNI)*, 2010.

[12] C. Willems, T. Holz, and F. C. Freiling, "Toward automated dynamic malware analysis using CWSandbox," *IEEE Security & Privacy*, vol. 5, no. 2, pp. 32–39, 2007.

[13] A. Becher, Z. Benenson, and M. Dornseif, "Tampering with motes: Real-world physical attacks on wireless sensor networks," in *Proceedings Security in Pervasive Computing*. Springer, 2006, pp. 104–118.

[14] R. West, "The psychology of security," *Commun. ACM*, vol. 51, pp. 34–40, April 2008. [Online]. Available: http://doi.acm.org/10.1145/1330311.1330320

[15] B. Schneier, "The psychology of security," http://www.schneier.com/essay-155.html, 2008.

[16] A. Adams and M. A. Sasse, "Users are not the enemy," *Commun. ACM*, vol. 42, no. 12, pp. 40–46, 1999.

[17] P. G. Inglesant and M. A. Sasse, "The true cost of unusable password policies: password use in the wild," in *Proceedings of the 28th international conference on Human factors in computing systems (CHI)*. New York, NY, USA: ACM, 2010, pp. 383–392.

[18] S. L. Garfinkel, "Digital forensics research: The next 10 years," in *Proceedings of the Digital Forensics Research Conferencs (DFRWS)*, 2010.

[19] K. Inman and N. Rudin, *Principles and Practice of Criminalistics: The Proefssion of Forensic Science*. Boca Raton: CRC, 2000.

[20] Volatile Systems, LLC, "The volatility framework: Volatile memory artifact extraction utility framework," 2008. [Online]. Available: https://www.volatilesystems.com/default/volatility

[21] F. C. Freiling, S. Schmitt, and M. Spreitzenbarth, "Forensic Analysis of Smartphones: The Android Data Extractor Lite (ADEL)," in *Conference on Digital Forensics, Security and Law*, 2011.

[22] P. Turner, "Selective and intelligent imaging using digital evidence bags," *Digital Investigation*, vol. 3, pp. 59–64, 2006.

[23] M. Bäcker, F. Freiling, and S. Schmitt, "Selektion vor der Sicherung," *Datenschutz und Datensicherheit*, vol. 34, no. 2, pp. 80–85, 2010.

[24] P. Turner, "Digital provenance-interpretation, verification and corroboration," *Digital Investigation*, vol. 2, no. 1, pp. 45–49, 2005.

[25] E. Kenneally and C. Brown, "Risk sensitive digital evidence collection," *Digital Investigation*, vol. 2, no. 2, pp. 101–119, 2005.

[26] C. Moch and F. C. Freiling, "The forensic image generator generator (forensig2)," in *Fifth International Conference on IT Security Incident Management and IT Forensics (IMF)*, IEEE Computer Society, 2009, pp. 78–93.

CLEARER: CrySyS Laboratory Security and Privacy Research Roadmap

Levente Buttyán Márk Félegyházi Boldizsár Bencsáth
Laboratory of Cryptography and System Security (CrySyS Laboratory)
Department of Telecommunications
Budapest University of Technology and Economics
www.crysys.hu

***Abstract*—The Laboratory of Cryptography and System Security (CrySyS) is dedicated to conduct research in the field of computer security and user privacy. This paper shows a research roadmap of the CrySyS Lab from its inception in 2003 until today. We will present the major achievements in the past with a particular emphasis on the research and teaching curriculum in security and privacy. We will discuss network- and wireless system security, the core competences of CrySyS. Building on the research foundation and competences in these areas, we will lead the laboratory into new territories of security and privacy in wireless embedded computing systems and the future Internet.**

***Keywords*-system security; network security; privacy enhancing technologies; economics of security; trust; wireless networks; embedded computing; Internet of Things; Future Internet**

I. Introduction

The Laboratory of Cryptography and System Security (CrySyS) was created with the purpose of conducting high-quality research on security and privacy of computer networks and systems, and to teach security and privacy at the Budapest University of Technology and Economics (BME). From its inception in 2003, the laboratory has gone a long way: our members have participated in international research efforts with widely-recognized research results in wireless network security, and in order to transfer the obtained research expertise, we bootstrapped a curriculum in the security of telecommunication systems that include courses on network and system security, privacy, cryptography, and the economics of security and privacy.

Our existing research results are centered around security and privacy problems in wireless embedded systems, notably in different types of sensor networks and vehicular communication systems. However, we are aware and keep track of the continuous evolution of other types of computing systems and communication networks too. We have witnessed such an evolution in recent years in terms of the core Internet infrastructure, end-devices and the software services relying on them. The core infrastructure is facing the major shift from legacy protocols to future Internet design. Naming and addressing, and the corresponding security mechanisms have to adapt to this change. For the end-devices, mobility and the integration of embedded systems bring a fundamental change in the way of communication and the capabilities of the communication devices. There is a significant paradigm shift at software services too; more and more functionality is concentrated at content providers who consequently started to develop infrastructure-based software services (e.g. collaborative software services).

Besides continuing our research in security and privacy in networked embedded systems, we are dedicated to contribute to the design of security and privacy solutions for the future Internet. The objective of this paper is to define a research roadmap to realize this goal.

II. Where we are?

The Laboratory of Cryptography and System Security, in its current form, was established in 2003. In terms of institutional organization, it is part of the Department of Telecommunications of the Budapest University of Technology and Economics. The lab currently has 4 faculty members, 1 post doc researcher, and 5 PhD students. This research team is currently led by Levente Buttyán, associate professor.

The research activities of the CrySyS Lab have been centered around security and privacy issues in wireless networked embedded systems. We successfully contributed to joint European research efforts by designing a security toolbox for sensor networks in the UbiSec&Sens Project (www.ist-ubisecsens.org), by applying sensor networks in the domain of critical infrastructure protection in the WSAN4CIP Project (www.wsan4cip.eu), by developing high-bandwidth and secure mesh networks in the EU-MESH Project (www.eu-mesh.eu), by designing a security architecture and location privacy enhancing mechanisms for communicating vehicles in the SeVeCom Project (www.sevecom.org), and recently, by developing secured e-health services based on body area sensor networks in the CHIRON Project (www.chiron-project.eu). We have published a number of papers on these topics in prominent journals and conferences (see our publication list on our web site at www.crysys.hu), and established a good reputation in the European ad hoc and sensor network research community.

In terms of teaching, our scope is broader, encompassing (i) an MSc level base course on Information Security, covering a wide range of security and privacy issues in IT systems and networks in general, and (ii) a special MSc program on Security in Telecommunication Systems, containing 3 mandatory courses and 1 elective course on Cryptography, Secure Protocols, Security in E-Commerce Systems, and Network Security in Practice, respectively, a number of laboratory exercises, and various offerings for semester and diploma level student projects.

III. Where we want to go?

Networked embedded systems (ubiquitous computing, Internet of Things) play an increasingly important role in IT, and therefore, their security and privacy issues remain an important research direction in the CrySyS Lab. However, besides the embedded world, we also intend to extend our research activities to the next generation Internet. In this section, we briefly summarize our research goals within the areas mentioned above.

A. Security and privacy in embedded systems and networks

1) Code attestation and code execution integrity for embedded devices: Similar to other computing systems, embedded devices are also controlled by software, which can contain vulnerabilities that can be exploited by malware. Indeed, some possible malware based attacks on sensor motes [1], RFID systems [2], implantable medical devices [3], and vehicles have been recently published, and more are expected to appear in the future. In addition, the recent incident caused by the Stuxnet worm shows, that malware targeting industrial embedded control systems exists already in practice. Smart phones will also be attractive targets for malware [4], as people store more and more sensitive information on them. At the same time, the resource limitations of embedded systems and their often specialized operating conditions make it difficult to protect them with existing anti-malware approaches known from the PC world.

In this context, remote code attestation and code execution integrity verification appear to be interesting security building blocks that help to increase the trustworthiness of embedded computing systems. Remote code attestation can assure a remote verifier that code loaded for execution is not tampered with, while code execution integrity verification in addition allows for checking that a given piece of code was executed as intended on the remote embedded execution platform. Both approaches help to detect the presence of malware.

While some hardware root of trust seems to be indispensable for remote code attestation and code execution verification, the general problem of hardware based approaches is that they cannot be applied to legacy and embedded systems that lack required hardware extensions. Purely software based solutions to attestation and verifiable execution of code running on legacy or embedded platforms have also been proposed in [5], [6] as good trade-offs between security and practical applicability. Unfortunately, some potential vulnerabilities in the most prominent software based code attestation solutions have recently been identified in [7]. This led to some debate among researchers [8], [9] that resulted in the conclusion that while software based code attestation is a useful security primitive, its design principles are not yet fully understood.

In our future research activities, we intend to develop a know-how on software based remote code attestation and code execution verification on various embedded platforms, and to design and analyze novel protocols for code attestation and code execution integrity protection. It is also our objective, to look into the possibility of applying formal modeling and reasoning for establishing some correctness proof for existing and future protocols. Our research in this area aims at increasing the trustworthiness of embedded computing systems and the services that they provide.

2) Privacy in cooperative networks of embedded devices: Cooperative networks based on the interconnection of various types of embedded devices with each other and to the Internet open the possibility to create new, highly-innovative services and applications. Embedded devices may be static, deployed at fix locations in the environment and in living spaces, or they may be mobile, carried by people or vehicles. In both cases, embedded devices can be used to sense and to control the environment, and to collect and provide various types of information about and to human users, respectively. Examples of such cooperative networks of embedded devices include wireless sensor and actuator networks, smart metering applications, vehicular networks, opportunistic ad hoc networks of personal mobile devices, or RFID based systems.

Mobile phones can also be considered as embedded computers, and they play a particularly important role, because they are usually equipped with different types of sensors (e.g., GPS based location sensors, accelerometers, cameras, microphones, thermometers, etc.), and at the same time, they have access to the Internet through WiFi or 3G connections. Hence, mobile phones are communication devices, end-user terminals, and they are also capable for continuous capturing of additional context information about the user (e.g., his physical location, the type of activity he is involved in).

While such global networks of embedded devices open new horizons in the domain of context aware services, they also create serious privacy problems. In particular, sensed data and the associated context information may leak private information about the individual sharing them.

Third party service providers who have access to such data may misuse them for tracking the location and the activities of the individual. Another aspect of the privacy problem concerns those users that want to access the shared information: their searching and downloading activities may reveal their personal interest profiles to other parties.

Therefore an important research objective of our lab is to design and analyze new privacy enhancing techniques for the sort of cooperative embedded networks described above, including both aspects of privacy, i.e., privacy enhancing techniques for information providers (who share sensed data and context information) and for information consumers (who search and download information).

B. Building trust in the Future Internet design

As the Internet transformed from a small computer network used by researchers to a global communication infrastructure the trust relations between the participants diminished. As a consequence of the disappearing trust relations, security became a fundamental issue. Unfortunately, Internet protocols were originally designed for the trusted networking environment and its rapid growth prevented computer scientists to redesign the networking protocols with security in mind. Instead, networking experts started to develop complementary solutions that could fix the original design weaknesses.

There exists a vast amount of Internet security protocols applied in practice to solve various problems and an even larger literature of academic security protocol design. There is however a substantial difference between the theoretical design of security protocols and their application in practice. The purpose of security solutions is often jeopardized by economic factors: the limited knowledge of users, the lack of incentives for users and companies to adopt appropriate security solutions or asymmetric information between participants in the security defense mechanisms. In recent years, the economic issues surrounding security problems received a significant amount of attention. Anderson and Moore [10] argued that the economic factors largely contribute to the inefficient application of security protocols. Following this observation, the field called economics of security emerged. We believe that the economics point of view of security and privacy, including the analysis of incentives using game theoretic tools, will be a key element in the design of new trust establishment mechanisms. Our research agenda covers the economics of security defense mechanisms as well as risk management issues as follows:

1) Economics of domain name registrations: Service providers play an important role in the online economy and their services are often misused by cyber-criminals. But, the service providers might not have the incentives to prevent such abuse especially if they do not suffer the consequences of misuse. In order to improve security incentives for service providers, it is important to understand how their existing infrastructure works. The online economy surrounding domain name registrations is quite complex and misused by criminals. Anecdotal evidence suggest that most of the new domain registrations serve malicious purposes [11]. Indeed, a recent study [12] has shown that a specific behavior called typo-squatting, where miscreants register domains that are reached by mistyping well-known domain names, gain momentum in domain registrations. To understand the incentives in this economy, one has to understand the purpose of domain names and the incentives of domain registrars. Our goal in this study is twofold. First, we aim at understanding how the current domain registration practices evolve and what type of domain names become popular. This is very relevant in the security context. Second, we will study how the practices of a competitive service provider market can be improved either by regulation or by designing an appropriate reputation mechanisms.

2) Trust and reputation in security defense mechanisms: Security defense relies on establishing trust between the communicating parties and blocking access of untrusted parties. Keeping usability is the main goal in mind, most existing defense mechanisms allow all unknown traffic to go through by default, observe the communication pattern and filter out communication that is deemed to be malicious. Reputation systems provide a key ingredient in defense mechanisms by sharing relevant observations with other defenders. This sharing component is crucial as it prevents the malicious traffic to penetrate widely on the Internet. Miscreants found a way to dynamically change their network identifiers and the defense mechanisms have answered with a more aggressive and reactive blocking of these identifiers resulting in an arms-race between attackers and defenders. We hypothesize that existing security mechanisms that are fundamentally based on exclusion will not be able to cope with the dynamically changing environment. Hence, there is a need to deploy more efficient reputation schemes and increasingly rely on greylisting techniques. Unfortunately, the situation is likely to escalate further with major architectural changes in the Internet architecture.

Our goal is to design reputation mechanisms that are able to keep up with the changing infrastructure while maintaining the convenient use of the protected services. We believe that this requires a transition from the traditional blocking-based security protocols to trust-building evaluation mechanisms. The main principle of the novel protocols is the incremental development of trust between Internet participants.

3) Cyber-insurance and risk management: Improving security practices on the Internet is difficult if the rules governing the interaction of participants are unclear or

weakly enforced. Cyber-insurance was proposed as a catalyst for improving security [13]. The assumption is that insurance companies in a future cyber-insurance market have a natural incentive to mitigate risks and motivate their clients to pay better attention to their computer systems. The cyber-insurance market has not yet taken off in large scale and there are several factors that hinder its widespread introduction [14]: interdependent security, correlated risks and asymmetric information jointly contribute to the difficulty of the problem. Because of these limiting factors insurers have a hard time to quantify risks and develop cyber-insurance products for complex IT systems.

We will focus on developing cyber-insurance mechanisms that fuel the improvement in network security practices. The recent example of University of California [15] has shown that clearly communicated security requirements enable an insurance company to develop an insurance products for seemingly uninsurable systems. Our research is to developing security mechanisms that provide clear guidelines for self-protection and serve as a basis of insurance products to mitigate risks that cannot be (or difficult to) protected against. We will integrate options to mitigate the adverse effects of interdependent security, information sharing and asymmetric information. We will pay a special attention to data management issues as these types of breaches have significantly increased in recent years.

IV. How we want to get there?

In order to reach our objectives, we will, on the one hand, leverage on our existing know-how and expertise in security and privacy in wireless embedded systems, and we will, on the other hand, take advantage of the background of our newly hired faculty members to progress on the new research domains described in this roadmap. More specifically, we will establish a new group with 2-3 PhD students, led by Márk Félegyházi, that will be dedicated to work on security and trust problems of the future Internet and that will follow an economics-based approach to tackle those problems. Besides that, we keep our group dedicated to security and privacy of embedded networked computing systems, and we will hire new PhD students to reach our goals in this domain. In order to obtain research funding, we are actively seeking opportunities to participate in related EU project proposals in the current and in the upcoming framework programs.

References

[1] A. Francillon and C. Castelluccia, "Code injection attacks on harvard-architecture devices," in *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2008.

[2] M. R. Rieback, B. Crispo, and A. S. Tanenbaum, "Is your cat infected with a computer virus?" in *Proceedings of IEEE Pervasive Computing and Communications (PERCOM)*, 2006.

[3] K. Rasmussen, C.Castelluccia, T. Heydt-Benjamin, and S. Capkun, "Proximity-based access control for implantable medical devices," in *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2009.

[4] M. Becher, F. C. Freiling, J. Hoffmann, T. Holz, S. Uellenbeck, and C. Wolf, "Mobile security catching up? – revealing the nuts and bolts of the security of mobile devices," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2011.

[5] A. Seshadri, A. Perrig, L. van Doorn, and P. Khosla, "SWATT: Software-based attestation for embedded devices," in *Proceedings of the IEEE Symposium on Security and Privacy*, May 2004.

[6] A. Seshadri, M. Luk, E. Shi, A. Perrig, L. van Doorn, and P. Khosla, "Pioneer: Verifying code integrity and enforcing untampered code execution on legacy systems," in *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, 2005.

[7] C. Castelluccia, A. Francillon, D. Perito, and C. Soriente, "On the difficulty of software-based attestation of embedded devices," in *Proceedings of ACM Conference on Computer and Communications Security (CCS)*, 2009.

[8] A. Perrig and L. van Doorn, "Refutation of On the difficulty of software-based attestation of embedded devices," http://sparrow.ece.cmu.edu/group/pub/ccs-refutation.pdf, 2010.

[9] A. Francillon, C. Castelluccia, D. Perito, and C. Soriente, "Comments on Refutation of On the difficulty of software-based attestation of embedded devices," http://planete.inrialpes.fr/~ccastel/PAPERS/2010_CCS_attestation_comments_on_rebutal.pdf, 2010.

[10] R. Anderson and T. Moore, "The economics of information security," *Science*, vol. 314, no. 5799, p. 610, 2006.

[11] P. Vixie, "Taking Back the DNS," http://www.isc.org/community/blog/201007/taking-back-dns-0, Jul 29 2010, retrieved on Feb 27, 2011.

[12] T. Moore and B. Edelman, "Measuring the perpetrators and funders of typosquatting," *Financial Cryptography and Data Security*, pp. 175–191, 2010.

[13] L. Gordon, M. Loeb, and T. Sohail, "A framework for using insurance for cyber-risk management," *Communications of the ACM*, vol. 46, no. 3, pp. 81–85, 2003.

[14] R. Böhme and G. Schwartz, "Modeling Cyber-Insurance: Towards A Unifying Framework," in *Proceedings of GameSec 2010*, 2010.

[15] K. Eft and A. Goldblatt, "New cyber insurance program at UC," http://inews.berkeley.edu/articles/Oct-Nov2010/cyberinsurance, Aug 9 2010, retrieved on May 31, 2011.

Towards malware-resistant networking environment

Dennis Gamayunov
Computer Systems Lab, CS Dept.
Lomonosov Moscow State University
Moscow, Russia
gamajun@cs.msu.su

***Abstract*—The modern cybercrime activities largely rely on malware-based infrastructure, i.e. botnets and backdoors in popular services for collecting private financial data, distributed denial of service and etc. A significant effort to develop better methods and tools for accurate malware detection and prevention is mounted both by the industry and academic community. With this paper we present current research roadmap for two adjacent fields: line-speed malware detection in modern network channels and privilege escalation prevention at host level by means of run-time monitoring of the networking applications normal behavior.**

***Keywords*-intrusion detection; network security; malware analysis; operating systems security**

I. Introduction

Recently malware has become a primary instrument of the cybercrime. Information security researchers in cooperation with industry have successfully shutdown several most famous botnets. Among them we could mention the Torpig botnet, deeply investigated by the UCSB research group [1], the Zeus botnet involved in FBI' investigations which ended in arrest of more than twenty people in September 2010 [2], and also the Rustock shutdown by Microsoft in early 2011.

We can designate the following stages of the botnets life cycle: propagation, privilege escalation on the infected computer, downloading trojan payload, linking to the botnet, executing commands from the botnet's C&C, removal from the botnet. Comparing the ease of botnet activity detection and differentiating it from normal Internet users activity, the propagation stage would be the most interesting as it involves computer attack, which is always an anomaly. The stages that follow successful infection - trojan extensions downloads, linking to botnet and receiving commands are usually made using ordinary application level protocols like HTTP or (rarely) IRC, different variations of P2P protocols, so that these communications are easily rendered to look as normal traffic. At the same time the propagation stage almost always involves malicious binary or JavaScript code transfer between attacker and victim, therefore it may be easier to detect then a botnet operation. Also during the privilege escalation phase the vulnerability within application or operating system is exploited, therefore it can be tracked too. Current research roadmap of our group focuses on two main directions: 1) detecting malware propagation phase in modern high-speed network channels, and 2) preventing privilege escalation at host level using run-time monitoring approach.

A. High-speed traffic analysis

When we try to detect malware propagation at network level, we come across hardware processing power limitation. The botnet growth phase is best observed on a large scale, i.e. at Tier-1 or Tier-2 ISP level backbone channels. At this level we observe an evolving gap between computational power of the modern processors and the throughput of the network channel. Therefore, we should pay special attention to the computational complexity of the algorithms used for malware propagation detection. In our work we try to formulate the task of the malicious shellcode detection in the high-speed network channels as a multi-criteria optimization problem: how to build a shellcode classifier topology using a given set of simple shellcode feature classifiers, where each simple classifier is capable of detecting one or more simple shellcode features with zero false negative rates, given computational complexity and false positive rates within it's shellcode classes, in order to provide the optimum aggregate false positive rates along with computational complexity. The resulting hybrid shellcode detector may be later mapped onto specific hardware architecture like FPGA to achieve optimal performance.

B. Host-level privilege escalation prevention

The privilege escalation prevention could help to block successful malware propagation at host level. There are numerous existing security mechanisms in modern operating systems that try to solve this problem. Whether malicious code runs on the target system processor or not, and whether this code gains privileged access to the operating system and user sensitive data or not - all the tasks mentioned depend on maturity of operating systems' access control mechanisms. Recently traditional UNIX discretionary access model was extended with mandatory

and role-based access control, and more specific mechanisms protecting application from memory corruption attacks - stack execution prevention, address space layout randomization and etc. There are also security subsystems in modern operating systems which try to control applications behavior, for example SELinux and AppArmor. In SELinux behavior control is based on description of subject and object types, and explicitly defined lists of allowed operations for each pair of subject and object. As a result, in case of successful exploitation of the remotely exploitable vulnerability, the malcode would not gain access to the whole user data.

One of the drawbacks of SELinux and AppArmor is that they implement only the basic principle of least privilege. Once the process is started under control of SELinux it has the same set of privileges for the whole run (except for the special cases when application is allowed to switch security contexts). We could possibly make least privilege principle even more restrictive by observing the actual states of the application and providing the least privileges necessary for the application right at a given moment. For example, we could allow to read/write config files only during the startup phase, if there are no other places in the application's CFG, where it requires access to the configuration. The motivating example of the application, where we could benefit from this more strict principle of least privilege, is an application with user authentication and authorization phase. In his case we could switch security context upon successful authentication depending on the role of authenticated user. This kind of context switching is implemented in OpenSSH server. The idea is to extend this kind of behavior control to the wider scope of networking applications, even to those which are unaware about SELinux and operating system security controls. In our work we're using automatically build application "models" in form of alternating security automata, which observe application behavior using software breakpoints and input/output data analysis and switch security context depending on the current application state.

II. Hybrid network-level malware classifier

Earlier we published a paper on Racewalk, a high-speed NOP-sled detection algorithm [4]. Racewalk was inspired by the Stride [3] algorithm and aimed at performance improvement and reducing false positive rates. Later we tried to extend these results to develop a hybrid shellcode detector, built as a data-flow graph combination of known classifiers for different types of malcode. We provide the idea of this hybrid classifier below.

Malicious executable code is characterized by a certain set of features that can divide entire set of malware into the classes. The problem of shellcode detection can be formulated in terms of recognition theory. Each shellcode detection method can be considered as a classifier which assigns the executable malicious code to one of the classes K_i of shellcode space. Each classifier has its own characteristics of shellcode space coverage, false negative and false positive rates, computational complexity.

Using the set of classifiers we can formulate the problem of automatic synthesis of such hybrid shellcode detector, which will cover all shellcode feature classes and reduce the false positive rates while reducing the computational complexity of the method compared with the simple linear combination of algorithms. The method should be synthesized in conformance with the profile of traffic channel data. In other words, the method should consider the probability of executable code transmission through the channel, etc. Let us consider the problem of algorithm synthesis as construction of a directed graph $G = (V, E)$ (see Fig. 1) with a specific topology, where $\{V\}$ is the set of nodes which are classifiers themselves, $\{E\}$ is the set of arcs. Each arc represents the route of flow data. We decided to include in the graph such classifiers (methods) that provide the most complete coverage of the shellcode classes $K_1, \ldots, K_l$. Each of the selected classifiers is assigned with two attributes: false positive rates and complexity. The attributes' values can be calculated through profiling, for example.

We assume that each node is associated with the type of the set $\{REDUCING, NON\_REDUCING\}$. If a node v_i has type $REDUCING$, and the classifier v_i concludes object S to be legitimate, the flow is not passed on. That implies the computational cost decreases and input flow is reduced. The reduced flow example is shown in Fig. 2

We associate each path in the graph G with its weight. The weight consists of two parameters combination: i) the total processing time, and ii) the false positive rates. It is necessary to include a classifier with lowest false positive rates to each path in G.

As part of the problem being solved it is necessary to propose a topology of graph G such that: i) the traffic profile will be taken into account; ii) all paths will be completed in the shortest time, and iii) the probability that the resulting information vector $\tilde{\alpha}(S)$ will be veritable is optimal. It implies that all paths will be completed with the lowest false positive rates. We will consider that problem in terms of multicriteria optimization theory.

III. Run-time monitoring for privilege escalation prevention

In addition to malcode detection at the network level we are developing a host-based privilege escalation prevention system, based on the run-time monitoring (or run-time verification) of application behavior. Run-time monitoring approach is relatively new and was first suggested in context of information security in early 2000s. For example, Martinelli and Matteucci in [5] proposed the theoretical

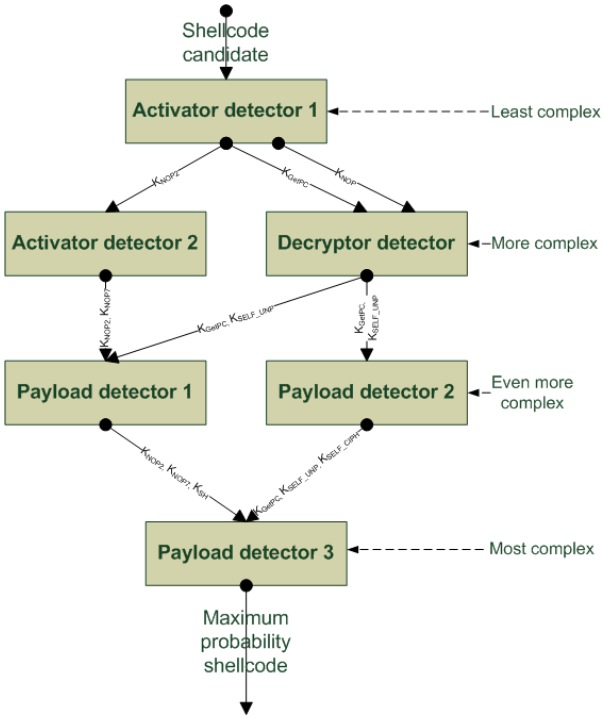


Figure 1: Graph example. Solid arrow represents the route of shellcode candidates. The arc (v_i, v_j) is marked with one of the classes K_x if v_i classifier checks whether shellcode candidate belongs to class K_x.

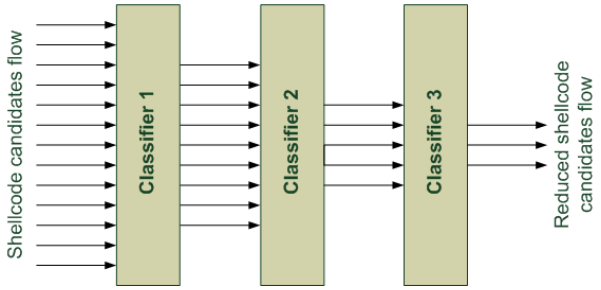


Figure 2: Example of flow reducing. Arrows represent the flow of shellcode candidates. The Classifiers 1, 2 and 3 consider part of the objects as legitimate, so they are not passed on.

foundation required to automatically construct controller operators - the special automata which can check and enforce given security polices for program. The security policies are formulas in the μ-calculus formalism (an extension of propositional modal logic) under sequences of security related actions. The mechanism was proposed to automatically create appropriate security automaton which can check the sequence of security related actions performed by the observed application and enforce policies in case of violation (for instance, terminate the execution or perform the contra-action). Later Bauer et al in [6] proposed a policy specification language Polymer and checker tool for Java programs. The Polymer can be used to specify the simple policies and its combinations which can interact one with another. Each policy provides a reaction to security relevant events (for example, method invocation). The special monitors are inserted into program to check and to enforce policies. There is also a significant ongoing research on fine-grained compartmentalization of applications, for example Wedge system by Bittau et al [7] and Capsicum by Watson et al [8]. Security compartmentalization is usually achieved by manual partitioning the given solid application, which requires significant code modifications. The proposed approach utilizes software breakpoints to achieve similar effect, and does not require any code modifications. We propose the extension of SELinux security subsystem with run-time application behavior monitoring, so that the operating system kernel could recognize inner states of the applications and switch security contexts, depending on the real observed application state. In addition to monitoring system and library calls performed by the observed application, we mark the application binary with checkpoints and observe each time when control flow passes the checkpoint. Checkpoints allow us to utilize knowledge about application CFG and use this information to attribute system and library calls. In order to minimize the privilege escalation effect of possible successful vulnerability exploitation, we need to split the whole set of resources needed by the application in several subsets, so that only one subset was available to the application at a time. To make the resulting application slicing more or less meaningful, an automated slicing process could be developed. The whole task of dynamic behavior control with privilege escalation prevention can be divided into three subtasks as follows:

A. Subtask 1 - Program slicing

The fist subtask is program slicing into a set of blocks and marking them with checkpoints. Let us have a program and corresponding SELinux profile for that program. We should reconstruct the program control flow graph (CFG) and split it into the set of non-overlapping blocks, so that the number of different privileges for each block was less than the overall number of privileges in the initial SELinux profile. For each block we should designate the entry point and all exit points and mark them as checkpoints.

B. Subtask 2 - Generating normal behavior model

The second subtask is to build normal program behavior model in terms of syscalls and passing checkpoints. Let us have a program marked with checkpoints. We should generate such recognition automaton, which accepts all

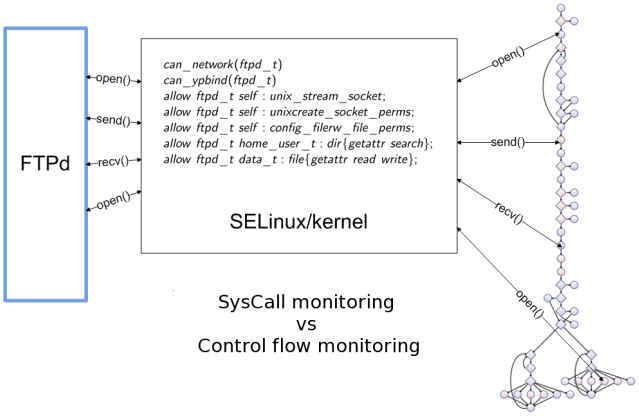


Figure 3: Example of the run-time behavior monitoring in terms of system calls and checkpoints within application CFG. The application example is FTP daemon.

normal traces of syscalls and sequences of checkpoints and rejects those traces, which do not belong to the normal behavior.

C. Subtask 3 - Run-time behavior monitoring

The third task is to control application behavior dynamically from within SELinux subsystem with the previously built normal behavior model. Let us have a program marked with checkpoints, a set of SELinux profiles each for separate program block and a normal behavior model in a form of recognition alternating automaton. We should implement SELinux module which would observe real program traces, pass them to the normal behavior model and effectively utilize the model output. Every time the subsystem observes exit from the block and entrance to the new block, the corresponding profile should be marked as active.

Currently we made a PoC-example of the run-time monitoring system with dynamic SELinux context switching for a simple FTP server (see Fig. 3)

IV. Conclusion

In this paper we described current and future research roadmap in the field of malware detection and prevention, both at network level and host level. The first research direction is focused on developing hybrid malware classifier capable of analyzing modern high-speed network channels with throughput of 1Gbps and higher. Hybrid classifier is build as a data-flow graph of elementary classifiers, each of them recognizes specific class of malicious code. The graph topology is chosen automatically so that the resulting throughput is optimal and error rates are minimal. The second research direction is focused on preventing privilege escalation at host level by means of run-time monitoring of networking applications behavior. In this approach an effective model of application normal behavior is built for each controlled application in a form of recognition alternating automaton, which monitors system and library calls performed by the application and checkpoints within application's address space. The monitoring system switches SELinux profile for the controlled application using information about its current state provided by the automaton. The latter reseach direction is currently in a PoC stage of development and has been tested with simple networking application. We believe that combining network level detection and filtering of malicious code with host level implementations of "hard" least privilege principle could bring us closer to the malware-resistant Internet infrastructure in the future, making the development of malware much more expensive and challenging task while minimizing possible profit of successful target host exploitations.

References

[1] B. Stone-Gross et al., *Your Botnet is My Botnet: Analysis of a Botnet Takeover*. Technical report, University of California, May 2009.

[2] FBI, *International Cooperation Disrupts Multi-Country Cyber Theft Ring*. Press Release, FBI National Press Office, Oct 2010.

[3] P. Akritidis, E. Markatos, M. Polychronakis, and K Anagnostakis, *Stride: Polymorphic sled detection through instruction sequence analysis*. In Proc. of the 20th IFIP International Information Security Conference (SEC'05), 2005.

[4] D. Gamayunov, N. T. Minh Quan, F. Sakharov, E. Toroshchin, *Racewalk: fast instruction frequency analysis and classification for shellcode detection in network flow* In: 2009 European Conference on Computer Network Defense. Milano, Italy, 2009

[5] Martinelli, F. and I. Matteucci, *Through modeling to synthesis of security automata*, In Proc. of ENTCS STM06, 2006

[6] L. Bauer, J. Ligatti, and D. Walker. *Composing expressive run-time security policies*, ACM Transactions on Software Engineering and Methodology. Vol. 18, No. 3, Article 9, 2009.

[7] Bittau, A., Marchenko, P., Handley, M., and Karp, B. *Wedge: Splitting Applications into Reduced-Privilege Compartments*, In Proc. of the 5th USENIX Symposium on Networked Systems Design and Implementation (2008), pp. 309322.

[8] R. N. M. Watson, J. Anderson, B. Laurie. *Capsicum: practical capabilities for UNIX*, In Proc. of the USENIX Security, 2010.

Research Roadmap on Security Measurements

Xenofontas Dimitropoulos
ETH Zurich
fontas@tik.ee.ethz.ch

***Abstract*—In the context of the SysSec Network of Excellence call for consolidating the European (and international) systems security research community, this position paper aims at summarizing the current research activities in the Communication Systems Group (CSG) of ETH Zurich relating to network security. Our research is aligned along three projects: 1) identifying, validating, and characterizing computer infections from intrusion alerts; 2) building privacy-preserving collaborative network security mechanisms based on efficient secure multi-party computation (MPC) primitives; and 3) dissecting Internet background radiation towards live networks with one-way flow classification. In addition, we highlight important directions for future research.**

I. Introduction

Our work is motivated by a number of inter-related problems. As modern malware rely increasingly on social engineering techniques, its becoming more important to develop reliable methods to detect with a small number of false positives systems within an organization that have been infected. Furthermore, to build better defenses its important to collaborate and share data about suspected attackers. Despite, its clear advantages, collaboration is presently largely avoided due to privacy concerns.

Motivated by these problems, in the following sections we discuss our current and future research on security measurements. In Section II we first describe security datasets we have accumulated and use in our studies. Next, in Section III we describe our inference, validation, and characterization of computer infections. Sections IV and V outline our research on aggregating sensitive data using MPC and on classifying one-way flows, respectively. Finally, in Section VI we outline future directions and we conclude in Section VII.

II. Collecting and Correlating Diverse Types of Data about a Target Network

A fundamental problem with understanding security failures and with evaluating proposed defenses is the lack of security datasets from real-world environments. Datasets, like intrusion detection alerts and traffic traces, are vital for scientific research. In CSG, we have been collecting unsampled NetFlow data from the border routers of SWITCH since 2003 and have accumulated a rich archive of more than 70 Gbytes of compressed flow records. In addition, since 2009 we have been collecting Snort alerts from a sensor next to the edge router of the ETH campus. Using the Snort signature ruleset and the Emerging Threats (ET) ruleset, we observe on average three million alerts per day and have accumulated a rich archive of several million Snort alerts. *The collection of different types of data from a production network is essential for building a more complete picture of security incidents and for correlating footprints from different observation instruments*. Members of CSG are actively working on extending our archives with additional types of data. In particular, on-going efforts examine the possibility of collecting vulnerability reports from end-hosts, reports from a Deep Packet Inspection (DPI) system, and DNS data.

III. Identifying, Validating, and Characterizing Computer Infections from IDS Alerts

In collaboration with Elias Raftopoulos we are working on identifying, validating, and characterizing computer infections in a large academic network infrastructure from intrusion alerts. In particular, our work addresses the following problems (more information can be found in our technical report [1]):

Extrusion Detection from IDS Alerts: A fundamental problem of network intrusion detection systems is that they generate a large number of false positives. For example, *our IDS sensor generates on average three million alerts per day*. Given an archive of IDS alerts, an important problem is how an administrator can filter out the noise to identify actual security incidents. In this work, we are particularly interested in identifying computer infection within a monitored infrastructure, i.e., extrusion detection. Annotating a rich trace of IDS alerts with inferred security incidents is useful both for forensics investigations as well as for evaluating network defenses with realistic data. To address this problem, in our current research we have developed a novel alert correlation technique tailored for extrusion detection.

Validating Inferred Infections in a Production Network: Having inferred a number of suspected infections, it is very useful to thoroughly validate the incidents. This is particularly challenging in production environments, where validation might interfere with regular employee work. In this work, *we are interested in the challenging problem of remotely validating suspected infections on un-managed hosts within a production network*. In our current research, we are conducting a complex experiment in which an expert

is assigned to manually validate live infections by collecting and analyzing data about the suspected host from a number of independent sources, including intrusion alerts, blacklists, vulnerability reports, host scanning, and search engine queries. The goal of manual assessment is to "connect the dots", i.e., correlate collected evidence and decide if they agree with the background knowledge about the suspected malware.

Characterizing Computer Infections: Infections are amongst the most critical events for computer administrators. Using our heuristic, we have identified several thousands infections that infected over a period of 9 months more than 11 thousand distinct hosts with static IP addresses. Given data about a large number of computer infections, it is important to systematically analyze infected hosts and derive insights for building better defenses. In our current research, we have characterized a number of different aspects of computer infections including infection times, types of infections, infected hosts, and observed spatio-temporal correlations. Among various interesting findings, we observe that the volume of inbound attacks to infected hosts increases rapidly after their infection and that infections exhibit significant spatial correlations, i.e., a new infection is more likely to occur close to already infected hosts.

IV. Building Privacy-Preserving Collaborative Network Security Mechanisms based on Secure Multi-party Computation

Internet security suffers from a fundamental imbalance: although attackers are globally spread and well coordinated, individual network domains are isolated to analyzing only local data, when in need to diagnose global security problems. If independent network domains collaborated, then it would be possible to design much more effective network monitoring and security mechanisms. For example, several collaborating networks can identify and blacklist spammers much faster and with higher accuracy than any single network alone. Despite its obvious advantages, collaboration is presently largely avoided because privacy laws, security policies, and competition prevent sharing sensitive data. To mitigate this problem we are working with Martin Burkhart on building efficient MPC primitives suited for collaborative network monitoring and security applications.

MPC appears as an ideal solution to the privacy-utility trade-off. On the one hand, any function can be turned into an MPC protocol and on the other hand the computation process provides strong privacy guarantees. Applying MPC on practical scenarios involving aggregation of network security data introduces the challenge of having to build very efficient protocols that can deal with voluminous input data. Our research spans a path starting from MPC theory, going to system design, performance evaluation, and ending with measurement. Along this path *a key contribution of our work is that we make an effort to bridge two very disparate worlds: MPC theory and network monitoring and security practices*.

In the theory front, we have designed optimized MPC comparison operations based on the observation that the performance of data-intensive MPC operations can be improved by not enforcing the widely-used constant-round paradigm. We have learned that constant-round protocols (at the cost of many multiplications) are not a panacea in MPC protocol design: *allowing many parallel invocations and removing the constant-round constraint enabled us to design protocols that substantially reduce the total run-time*.

In addition, we have designed four MPC protocols, namely the entropy, distinct count, event correlation, and top-k protocols. Our protocols have been inspired from specific network monitoring and security applications, but at the same time they are also general and can be used for other applications. For example, in our top-k protocol we have used sketches, an approximation data structure, to reduce expensive MPC comparison operations to much more efficient MPC additions and multiplications at the cost of a manageable approximation error [2].

We have implemented our basic operations and MPC protocols in the SEPIA library [3], [4], which we have made publicly-available. Our extensive performance evaluation shows that SEPIA operations are between 35 and several hundred times faster than those of existing comparable MPC frameworks. In addition, we have used our protocols with real-world traces from 17 customer networks of SWITCH to investigate the practical applicability of collaborative network monitoring and security based on MPC. We have investigated different ways the networks could have collaborated to troubleshoot an actual global network anomaly. This is the first work to apply MPC on real traffic traces and to demonstrate that *collaborative network monitoring and security based on MPC is both computationally feasible and useful for addressing real network problems*.

This work has led to one recent publication [3], while further research is carried out in the context of two projects. In the DEMONS FP7 collaborative European project, we are further optimizing the performance and extending the functionality of SEPIA, while in collaboration with IBM Research we have acquired support from the Swiss National Science Foundation to transfer the technology of the multi-party computation library to a privacy-preserving multi-domain traffic flow analyzer.

V. Beyond Network Telescopes: One-way Flow Classification

Network telescopes extract Internet background radiation to unused IP address blocks and have been very useful in characterizing malicious patterns and events. An alternative way to study background radiation in edge networks is to find one-way flows, i.e., flows that never receive a reply. One-way flows are an important fraction of Internet traffic.

In our traces from SWITCH, *in 2004 two out of every three flows were one-way, while in 2008 one out of every three flows were one-way*. One-way flows provide a number of advantages for monitoring Internet background radiation. They can be easily extracted from live networks, can be used even when large unused IP address blocks are not available, and require fewer resources for instrumentation. To enable administrators and researchers to extract and analyze background radiation from one-way flows, we have been working with Eduard Glatz on *novel techniques for classifying one-way flows in a set of malicious and benign classes*. Our classification associates each one-way flows with up to 17 different signs derived from flow-level data and uses a set of expert rules to determine the appropriate class of an one-way flow. It does not require training (plug & play), is easily extensible, and is based on comprehensible classification rules. *We have used our classification to analyze a massive dataset of flow records summarizing 7.73 petabytes of traffic between 2004 and 2010* that crossed the border routers of SWITCH. In addition, we work on characterizing how the volume of background radiation changed over time and what is the persistence of local port numbers as top targets. Among our findings, we observe that the relative volume of background radiation towards the target network decreased sharply between 2004 and 2007 by 73% and remains almost constant since then. In addition, filtering the top-5 or 10 port numbers, reduces the volume of background radiation by 35.6% and 45.0%, respectively. We believe that one-way flow classification opens a number of new directions on exploiting background radiation for building better intrusion detection systems. More information can be found in our technical report [5].

VI. Future Directions

Based on our experience with our current research, we believe that the following open problems deserve further attention in the future:

1) **Assessing the privacy risk of the MPC output:** Although MPC based on Shamir's secret sharing scheme guarantees that the computation provides information theoretic privacy, the output of the computation may still pose privacy risks especially if an adversary can correlate it with background knowledge. Intuitively, aggregating data from multiple parties obscures the sensitive input of any individual party. However, it is still an open question *how to assess the privacy risk of the output of an evaluated MPC function and if the output needs to be perturbed before publication* to provide some rigorous privacy guarantees.
2) **Automating security assessment:** Our experience with validating a number of inferred computer infections taught us that *security assessment in large organizations is a magic art rather than a scientific process*. It requires collecting relevant evidence, leveraging the background knowledge of domain experts about the infrastructure and the suspected malware, and relying on an ad-hoc cognitive process to diagnose a suspected security incident. In the future, the steps of a security assessment process need to be supported by automated techniques that expedite or even replace the manual work of a security administrator.
3) **Correlating flow data with intrusion alerts:** Past research has extensively studied how an administrator can use intrusion alerts or flow data to detect attacks and other security incidents. Although these data sources have been studied in isolation, an important question is *how to best combine flow data with IDS alerts to detect security incidents more accurately*. Flow data provide more information about the normal behavior of a host, while intrusion alerts provide more details about a suspected incident. Their proper combination can lead to reducing the false positive rate of IDSs and to enriching flow-based anomaly detectors with more information about a suspected anomaly.

VII. Conclusions

This short paper aims at summarizing the current research of part of the Communication Systems Group of ETH Zurich that relate to security measurements. In addition, we pinpoint important future research directions. Our focus areas are on: 1) identifying, validating, and characterizing computer infections from intrusion alerts; 2) MPC-based collaborative network monitoring and security applications; and 3) one-way flow classification. More details about the summarized works can be found in [1], [4], [3], [5].

References

[1] E. Raftopoulos and X. Dimitropoulos, "Detecting, validating and characterizing computer infections from IDS alerts," ETH Zurich, TIK-Report 337, June 2011.

[2] M. Burkhart and X. Dimitropoulos, "Fast privacy-preserving top-k queries using secret sharing," in *International Conference on Computer Communications and Networks (ICCCN)*, 2010.

[3] M. Burkhart, M. Strasser, D. Many, and X. Dimitropoulos, "SEPIA: Privacy-Preserving Aggregation of Multi-Domain Network Events and Statistics," in *USENIX Security Symposium*, 2010.

[4] M. Burkhart and X. Dimitropoulos, "Privacy-preserving distributed network troubleshooting – bridging the gap between theory and practice," *ACM Transactions on Information and Systems Security (under submission)*, 2011.

[5] E. Glatz and X. Dimitropoulos, "Beyond network telescopes: One-way traffic classification," ETH Zurich, TIK-Report 336, June 2011.

Towards a better understanding of the impact of emerging ICT on the safety and security of the Citizen

Digital Citizen Security - a programmatic approach

Ioannis Kounelis, Jan Löschner, Vincent Mahieu, Jean-Pierre Nordvik, Pasquale Striparo
Traceability and Vulnerability Assessment Unit
Joint Research Centre of the European Commission
Ispra, Italy
{name.surname}@jrc.ec.europa.eu

Sead Muftic
School of Information and Communication Technology Royal Institute of Technology (KTH)
Stockholm, Sweden
sead@kth.se

Abstract — **The Joint Research Centre (JRC) of the European Commission has taken initiative to investigate, assess and forecast issues of the exploitation of digitalized personal data of citizens in our forthcoming digital society. It responds that way to some of the key challenges put forward in the Communication from the Commission "Europe 2020" and to one of its flagship initiatives, the "A Digital Agenda for Europe". The issues addressed are namely Trust and Security, a vibrant digital single market through building digital confidence and ICT-enabled benefits for the EU society and Intelligent Transport Systems for safer, more secure and more efficient transport and better mobility in Europe. The paper describes the current organization and the research roadmap of the Traceability and Vulnerability Assessment Unit [1] of the JRC and its partners. It illustrates the unit's objectives for the coming years and into the European Unions 8th Research Framework program in respect to the digital security of citizens.**

Digital Agenda, Digital Trace, European Commission, Data Protection, Intelligent Transport Systems, Joint Research Centre, Vulnerability Analysis, Video Surveillance, Profiling, Security, Trust,

I. INTRODUCTION

Emerging Information and Communication Technology (ICT) is changing our society in a revolutionary dimension. As a result and part of the scientific aims and programs of the European Commission anchored amongst others in the Stockholm Program [2] for an open and secure Europe serving and protecting the citizens, the societal challenges arising from this development and the adoption of emerging technologies appeared as one of the central research topics during the last 10 years. The European Commission (EC) formulated as a response to the economic and financial crisis its strategy for smart, sustainable and inclusive growth [5], in which the "five measurable EU targets for 2020" are defined. Those targets include amongst technical ones, the social, ethical, institutional and legal implications of research and development of emerging technologies. The ETICA project states for example: "While the problems of evaluating ethics of ICT are significant for current ICTs, it becomes more pressing, but also more difficult, for emerging ICTs. If societies want to be proactive in addressing possible ethical issues, they need to have some reliable way of identifying these technologies." [4] Anticipation of long-term implications and on how societal concerns may frame publics' perceptions and acceptance are crucial for Europe in order to design socially robust policies. The review of the EU Data Protection Directive in these days does not come as a surprise. One of the main policy objectives for the European Commission is to modernize the EU legal system for the protection of personal data, in particular to meet the challenges resulting from globalization and the use of new technologies.

An important tool of the European Commission is its Joint Research Centre consisting of 7 institutes and working in 6 thematic areas. One of the 7 institutes is the Institute for the Protection and Security of the Citizen (IPSC) whose mission is to provide research results and to support EU policy-makers in their effort towards global security and towards protection of European citizens from accidents, deliberate attacks, fraud and illegal actions against EU policies.

In a globalized world, security and crisis management must address cross-border challenges in a highly efficient and automated way. JRC Thematic Area "Security and Crisis Management" therefore will put a strong emphasis on research in intelligent systems and interconnected solutions for security and crisis management. This way it will provide a solid scientific basis for JRC support to the Stockholm Program [2] and to the implementation of the Commission R&D ICT Work Program 2011 – 2012 [6]. Key areas of research will be:

- **Intelligent analysis and situation assessment** in crisis management and maritime security,
- **Modelling of complex interdependencies** for the protection of critical network and building infrastructures,
- **Interdisciplinary assessment of the impact of new ICT technologies** on the European Citizen.

The European citizen as an individual is put to the centre of the considerations of the restructured Traceability and Vulnerability Assessment Unit of the IPSC that will address issues such as the acceptance of new ICT, data protection and privacy concerns, security ethics, citizen profiling and electronic traces. The approach goes beyond a purely technical consideration and will also address social aspects (acceptance, awareness, ethics), and in the future potentially also legal aspects where new technologies might create a new demand for European regulation.

The IPSC complements research in this area in the time frame between the long term view of the Institute for Prospective Technological Studies (IPTS) of the JRC, which responses to policy challenges that have a socio-economic as well as a scientific/technological dimension and the immediate needs of the policy makers.

II. Organisation of the Traceability and Vulnerability Assessment Unit

The Traceability and Vulnerability Assessment Unit of the IPSC was reorganized at the beginning of 2011 and will focus on the citizen impact research, an interdisciplinary approach, between technology and social sciences, meant to produce scientific results and policy support that go beyond the technology perspective. The unit will enter in a number of iterations to assess the impact of emerging ICT on the safety and security of Citizen, it will derive from this guidelines and best practice recommendations for the European policy makers in the Parliament and in the client directorates of the European Commission and it will develop scenarios to anticipate potential risks and threats for the citizen to be used for the next iteration of impact assessment. To achieve this, the unit is organized since the beginning of 2011 in the following 3 actions:

- The Citizen Digital Footprint (CIDIPRINT) action which will develop and assess scenarios associated with information recorded when a citizen interacts in a digital smart environment, in particular with the internet of the future and with intelligent transport systems;
- The Security Aspects of the Digital Society (SIDSO) action which will detect and anticipate potential societal implications of emerging Information and Communication Technologies;
- The Surveillance Systems and the Citizen (SURCIT) action which will assess existing and emerging technologies and solutions for surveillance and monitoring.

The three Actions are complementary and contribute jointly to the emergence of a competence centre in the multidisciplinary assessment of the citizen impact of new Information and Communication Technologies. Findings of the SIDSO Action on Societal Impact will be used to assess the technologies and solutions in scope of the two other Actions on footprints and monitoring. Likewise, the scenarios and findings of these two Actions will feed into the societal impact assessment work of the first Action.

III. Objective toward a Protected and Secure Citizen

For the citizen impact research, an interdisciplinary approach will be taken, between technology and social sciences, meant at producing scientific publications that go beyond the technology perspective.

The main objectives of the Traceability and Vulnerability Assessment Unit are to:

- **Assess the implication** of emerging ICT on the citizen in a structured and substantiated (i.e. scientific) way;
- Focus on the citizen and his **perception** using ICT or interacting with the digital world;
- Broaden a purely technological approach to an **interdisciplinary** one including societal and legal issues;
- Identify and **establish partnerships** with the most competent stake holders and foster high level networking inside Europe and in the World to impact in a sustainable way European policy in the benefit of the Citizen;
- **Develop and assess scenarios** associated with information recorded when a citizen interacts intentionally or non intentionally with ICT;
- Detect, **anticipate** and prioritize potential society implications of emerging ICT's.

The scenarios associated with information recorded when a citizen interacts in the role of information consumer or of information provider will focus on citizen interactions in a digital smart environment, in a mobile world, in the Internet of the future, under video surveillance and with Intelligent Transport Systems (ITS). Key enabling technologies like smartcards with ultra high communication bit rate, Near Field Communication (NFC), Radio Frequency Identification, the Internet Protocol Version 6, Social Networks and the satellite based localization systems will be put in relation to citizen centric dimensions like trust, confidence and convenience.

With this three actions the Traceability and Vulnerability Assessment Unit plans to carry out coordinated activities with and complementing the work of the Information Society Unit of the IPTS.

IV. Strategy to Control the ICT Impact to Citizen of the Future

The Traceability and Vulnerability Assessment Unit has defined its strategy based on 3 pillars:

1 The **experience and expertise** gained developing, evaluating, and applying in the past methods for assessing vulnerabilities of complex systems and infrastructures to technological, manmade (voluntary or not) and natural hazards;

2 Closed **links with other Directorates General** of the European Commission to facilitate an optimal policy interaction.

3 A **strategic partnership** with key research partners and an active participation to relevant thematic expert networks;

An example for the 1st pillar, the existing expertise of the Traceability and Vulnerability Assessment Unit is the work performed [9], [10] to establish the Digital Tachograph as a trusted information source in professional vehicles with the potential to be further developed under the current 2011 Regulation Revision Process of the Council Regulation (EEC) No 3821/85. This expertise includes the routine operation of the European Root Certification Authority (ERCA) with the potential to be further developed as an anchor of trust for the European citizen in emerging ITS applications.

This example links as well to the 2nd pillar concerning the support of the Directorate General (DG) MOVE of the EC in regards to the already mentioned regulation. Other partners will be Information Society and Media Directorate General, the Directorate General for Home Affairs and the Directorate General for Justice, the Directorate General Enterprise and Industry and the European Network and Information Security Agency.

The 3rd pillar, which concerns the strategic partnership of the IPSC will be illustrated more in detail. A concrete partnership established is the research collaboration between the Royal Institute of Technology KTH Stockholm and the Institute for the Protection and Security of the Citizen under which 3 KTH students received a research grant from the JRC to study the impact that emerging mobile information and communication technologies have on the citizen, its security and privacy. Focus of the research collaboration is the user interaction via different applications such as the use of the Internet, the use of smart cards, public registration of identifiable objects, the free move with identifiable objects, moving as a person or the storage of personal data in repositories [8] during communication. High priority will be given to the most modern interactions of mobile nature. This will include communication via established networks or in ad hoc networks.

One grant holder will be dedicated to the security and privacy issues related to mobile devices communication protocols, mobile devices applications and mobile operative systems. Since ICT users are entering at a high pace in the "always connected" paradigm Era, where people are connected 24/7 to the Internet, to smart environments where they can interact with other devices, such as Internet of Things, with third entities like banks, postal offices, etc., from wherever they are, it is clear that this topic is very sensitive and that it may have a big impact in terms of number of people affected. Moreover, the trends towards communication without involving telecom operators can lead to kind of ad hoc mobile networks with web 2.0 functionalities, where large number of people can interact with each other.

More in details this work will elaborate on two concrete examples of Bluetooth and NFC. It will demonstrate on the example of Bluetooth threats and risks for citizen in motion by analyzing intentional and unintentional interactions between Bluetooth enabled devices carried around and potential attackers. This analysis will focus on surveillance by acquiring specific details about a Bluetooth device, traceability by identifying Bluetooth devices uniquely, sniffing and eavesdropping the Bluetooth broadcasts traffic and the Denial of Service in mission critical security applications. The arising risk of such networks being used as mobile Botnet will be considers and studied.

The two wireless communication possibilities are chosen as they are seen as the enabling technology where mobile payment systems will be based on. In the very next future, smart mobile devices are going to replace wallets and plastic credit cards enabling the

Citizen to pay his purchase by just swiping his mobile phone against another NFC enabled device. A citizen centric analysis of requirements such as robustness in case of loss or theft of the mobile device, in case of sniffing and eavesdropping of the radio signal during payments session or in case of unwanted and unsolicited payment triggered by a malicious person getting closed to the Citizen who is carrying a NFC enable mobile device to prevent harm from the citizen in the future will be carried out.

A second grant holder will work devoted to research taking advantage of the latest communicational and computational capabilities of mobile devices in terms of working together distributed. It will illustrate the approach towards distributed mobile applications. The goal is to design an architecture that will allow secure mobile distributed computing. The focus of this work will be put on secure mobile applications that are executed in such a distributed environment. It will link to challenges arising in this specific area such as trust, mobile clouds, mobile ad hoc networks and the optimized use of limited mobile resources.

To implement this, several authentication levels between nodes are proposed and associated with different trust levels. An example will be given, where authenticating will not be enough but user identification is required. In this case a link between the user/owner of the mobile device and the application could be established, which can be facilitated with the use of the Universal Integrated Circuit Cards (UICC). A possibility to design middleware architecture between the distributed applications and the operating system in order to enable interoperability between several platforms will be analyzed. The new challenge to be addressed in the architecture and discussed in the future will be the way the mobile devices connect with each other in mobile ad hoc networks. This will require a different approach from just having computers as nodes. Since mobile devices from several platforms might be used, it will be interesting to build ad hoc networks for isolated or internet connected personal area networks with the use of Bluetooth and Wi-Fi connections. Moreover, it will be beneficial to provide transparent fail-over between redundant network paths. This is especially relevant in case of the ad hoc network is used as an underlying infrastructure uses as a cloud. The planned work will lead to a detailed assessment of the cloud for execution and data storage security concerns, mostly with respect to the level of trust in the distributed application and with respect to the citizen's privacy.

REFERENCES

[1] http://ipsc.jrc.ec.europa.eu/research.php?unit=7

[2] EUROPE 2020, A European strategy for smart, sustainable and inclusive growth, COMMUNICATION FROM THE COMMISSION, Brussels 2010

[3] The Stockholm Programme – An open and secure Europe serving and protecting the citizens, Council of the European Union, 17024/09, 02.12.2009

[4] The Magazine of the European Innovation Exchnage, ETICA Ethical issues of emerging ICT applications, 2010, http://europeaninnovationexchange.net/docs/EIEX03ETICA.pdf

[5] COMMUNICATION FROM THE COMMISSION TO THE EUROPEAN PARLIAMENT, THE COUNCIL, THE EUROPEAN ECONOMIC AND SOCIAL COMMITTEE AND THE COMMITTEE OF THE REGIONS - A Digital Agenda for Europe, COM(2010) 245 final/2, Brussels, 2010 http://ec.europa.eu/information_society/digital-agenda/index_en.htm

[6] The Commission R&D ICT Work Programme 2011 – 2012, ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/ict-wp-2011-12_en.pdf

[7] Council conclusions on the commission communication "Action Plan for the Deployment of Intelligent Transport Systems in Europe" http://www.consilium.europa.eu/uedocs/cms_data/docs/pressdata/en/trans/106964.pdf

[8] DIGITAL FOOTPRINT, Part 1: State of the art, JRC Technical Note – Draft, Ispra 2011

[9] J. W. Bishop, J. - P. Nordvik, "Digital Tachograph System European Root Policy", Version 2.1, JRC Technical Note, Ispra, 2009

[10] J. W. Bishop, D. Landat, "Digital Tachograph Equipment Type Approval", Interoperability Test Specification Version 2.0, JRC Scientific and Technical Report EUR 23061 EN, Ispra, 2008

From SSIR to CIDre: a New Security Research Group in Rennes, France

Emmanuelle Anceaume, *CNRS*
Christophe Bidan, *SUPELEC*
Sébastien Gambs, *Univ. Rennes 1 and INRIA*
Guillaume Hiet, *SUPELEC*
Michel Hurfin, *INRIA*
Ludovic Mé, *SUPELEC*
Guillaume Piolle, *SUPELEC*
Nicolas Prigent, *SUPELEC*
Eric Totel, *SUPELEC*
Frédéric Tronel, *SUPELEC*
Valérie Viet Triem Tong, *SUPELEC*

***Abstract*—CIDre , which stands for "Confidentialy, Integrity, Availability, and repartition", is the name of a new research group created in Rennes, France, as a follow-up of the SSIR team (www.rennes.supelec.fr/ren/rd/ssir), which was, until 2011, a Supélec team whose work was mainly focused on intrusion detection and spontaneous network (ad hoc, P2P) security.**

The global research objective of this new CIDre research group is to study new security solutions for nodes and network of nodes, in particular through the use of classical but potentially revised approaches coming from the distributed computing field. More especially, we focus on three different aspects of security: privacy, trust, and intrusion detection.

***Keywords*-confidentialy, integrity, availability, repartition, intrusion detection, trust, privacy.**

I. Introduction and Team Characteristics

Two research communities traditionally address the concern of accidental and intentional failures: the fault tolerance and distributed computing community and the security community. While both these communities are interested in the construction of systems that are correct and secure, an ideological gap and a lack of communication exist between them that is often explained by the incompatibility of the assumptions each of them traditionally makes. Furthermore, in terms of objectives, the distributed computing community has favored systems availability while the security community has focused on integrity and confidentiality, and more recently on privacy. By contrast with this traditional conception, we are convinced that by looking at information systems as a combination of possibly revisited basic protocols, each one specified by a set of properties such as synchronization and agreement, security properties should emerge. This vision is shared by others and in particular by Myers et al [1], whose objectives are to "explore new methods for constructing distributed systems that are trustworthy in the aggregate even when some nodes in the system have been compromised by malicious attackers".

In accordance with this vision, the **first main characteristic** of the CIDre group is to gather researchers from the two aforementioned communities in order to address in a complementary manner both the concerns of accidental and intentional failures [2].

The **second main characteristic** of the CIDre group lies in the scope of the systems it considers. Indeed, during our research, we will consider three complementary levels of study: the *Node Level*, the *Group Level*, and the *Open Network Level*:

– *Node Level*: The term *node* either refers to a device that hosts a network client or service or to the process that runs this client or service. Node security management must be the focus of a particular attention, since from the user point of view, security of his own devices is crucial. Sensitive information and services must therefore be *locally* protected against various forms of attacks. This protection may take a dual form, namely prevention and detection.

– *Group Level*: Distributed applications often rely on the identification of sets of interacting entities. These subsets are either called groups, clusters, collections, neighborhoods, spheres, or communities according to the criteria that define the membership. Among others, the adopted criteria may reflect the fact that its members are administrated by a unique person, or that they share the same security policy. It can also be related to the localization of the physical entities, or that they need to be strongly synchronized, or even that they share mutual interests. Due to the vast number of possible contexts and terminologies, we refer to a single type of set of entities, that we call *set of nodes*. We assume that a node can locally and independently identify a set of nodes and modify the composition of this set at any time. The node that manages one set has to know the identity of each of its members and should be able to communicate directly with them without relying on a third party. Despite these two restrictions, this definition remains general enough to include as particular cases most of the examples mentioned above. Of course, more restrictive behaviors can be specified by adding other constraints. We are convinced that security can benefit from the existence and the identification of sets of nodes of limited size as they can help in improving the efficiency of the detection and prevention mechanisms.

– *Open Network Level*: In the context of large-scale

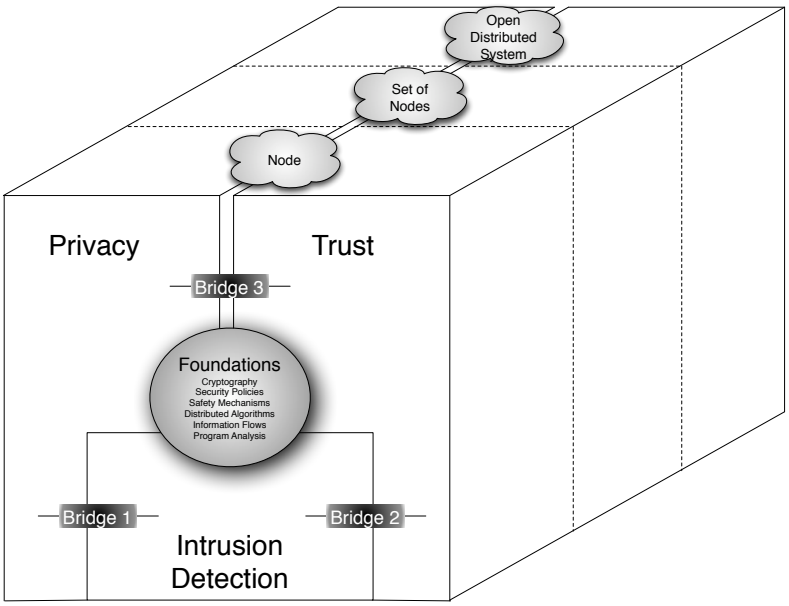


Figure 1. A synthetic view of the CIDre research group objectives

distributed and dynamic systems, interaction with unknown entities becomes an unavoidable habit despite the induced risk. For instance, consider a mobile user that connects his laptop to a public Wifi access point to interact with his company. At this point, data (regardless it is valuable or not) is updated and managed through non trusted undedicated entities (i.e., communication infrastructure and nodes) that provide multiple services to multiple parties during that user connection. In the same way, the same device (e.g., laptop, PDA, USB key) is often used for both professional and private activities, each activity accessing and manipulating decisive data.

The **third characteristic** of the CIDre group is to focus on three different aspects of security, *i.e.*, trust, intrusion detection, and privacy, and on the different bridges that exist between these aspects. Indeed, we believe that to study new security solutions for nodes, set of nodes and open network levels, one must take into account that it is now a necessity to interact with devices whose owners are unknown. To reduce the risk to rely on dishonest entities, a *trust mechanism* is an essential prevention tool that aims at measuring the capacity of a remote node to provide a service compliant with its specification. Such a mechanism should allow to overcome ill-founded suspicions and to be aware of established misbehaviors. To identify such misbehaviors, *intrusion detection systems* are necessary. Such systems aimed at detecting, by analyzing data flows, whether violations of the security policies have occurred. Finally, *Privacy Protection* which is now recognized as a basic user right, should be respected despite the presence of tools that continuously observe or even control users actions or behaviors. An overall description of our research objectives is depicted in Figure 1. In addition to the three aforementioned research topics, the three levels and the three bridges, a cylinder corresponding to the fundamental tools we rely on (distributed computing, cryptography, security policy, ...) appears at the center of the graph.

In the remaining of this paper we present a short introduction describing our vision of these three different aspects of security, and identify examples of concrete research goals we will work on.

II. Trust Management

While the distributed computing community relies on the trustworthiness of its algorithms to ensure systems availability, the security community historically makes the hypothesis of a Trusted Computing Base (TCB) that contains the security mechanisms (such as access controls, and cryptography) that implement the security policy. Unfortunately, as information systems get increasingly complex and open, the TCB management may itself get very complex, dynamic and error-prone.

From our point of view, an appealing approach is to distribute and manage the TCB on each node and to leverage the trustworthiness of the distributed algorithms in order to strengthen each node's TCB. Accordingly, the CIDre group proposes to study *automated trust management systems* at all the three identified levels:

– at the *node level*, such a system should allow each node to evaluate by itself the trustworthiness of its neighborhood and to self-configure the security mechanisms it implements;

– at the *group level*, such a system might rely on existing trust relations with other nodes of the group to enhance the significance and the reliability of the gathered information;

– at the *open network level*, such a system should rely on reputation mechanisms to estimate the trustworthiness of the peers the node interacts with. The system might also benefit from the information provided by *a priori* trusted peers that, for instance, would belong to the same group (see previous item).

For the last two items, the automated trust management system will *de facto* follow the distributed computing approach. As such, emphasis will be put on the trustworthiness of the designed distributed algorithms. Thus, the proposed approach will provide both the adequate security mechanisms and a trustworthy distributed way of managing them.

By way of examples of our research goals regarding the trust management field, we briefly list some of our short and long term objectives at node, group and open networks levels:

1) at node level, we are going to investigate how implicit trust relationships, identified and deduced by a node during its interactions with its neighborhood, could be explicitly used by the node (for instance by means of a series of rules) to locally evaluate the trustworthiness of its neighborhood. The impact of trust on the local security policy, and on its enforcement will be studied accordingly.

2) at the set of nodes level, we plan to take advantage of the pre-existing trust relationship among the set of nodes to design composition mechanisms that would guarantee that automatically configured security policies are consistent with each group member security policy.

3) at the open distributed system level, we are going to design reputation mechanisms to both defend the system against specific attacks (whitewashing, bad mouthing, ballot stuffing, isolation [3]) by relying on the properties guaranteed at nodes and set of nodes levels, and guaranteeing persistent and safe feedback [4], and for specific cases in guaranteeing the right to oblivion (*i.e.,* the right to data erasure).

III. Intrusion Detection

By exploiting vulnerabilities in operating systems, applications, or network services, an attacker can defeat the preventive security mechanisms and violate the security policy of the whole system. The goal of intrusion detection systems (IDS) is to be able to detect, by analyzing some data generated on a monitored system, violations of the security policy.

From our point of view, while useful in practice, misuse detection is intrinsically limited. Indeed, it requires to update the signatures database in real-time similarly to what has to be done for antivirus tools. Given that there are thousands of machines that are every day victims of malware, such an approach may appear as insufficient especially due to the incredible expansion of malware, drastically limiting the capabilities of human intervention and response. The CIDre group takes the alternative approach, *i.e.* the anomaly approach, which consists in detecting a deviation from a referenced behavior. Specifically, we propose to study two complementary methods:

– *Illegal Flow Detection*: This first method intends to detect information flows that violate the security policy [5], [6]. Our goal is here to detect information flows in the monitored system that are allowed by the access control mechanism, but are illegal from the security policy point of view.

– *Data Corruption Detection*: This second method aims at detecting intrusions that target specific applications, and make them execute illegal actions by using these applications incorrectly [7], [8]. This approach complements the previous one in the sense that the incorrect use of the application can possibly be legal from the point of view of the information flows and access control mechanisms, but is incorrect considering the security policy.

In both approaches, the access control mechanisms or the monitored applications can be either configured and executed on a single node, or distributed on a set of nodes. Thus, our approach must be studied at least at these first two levels.

Moreover, we plan to work on intrusion detection system evaluation methods. For that research, we set a priori aside no particular IDS approach or technique.

Here are some concrete examples of our research goals (both short term and long term objectives) in the intrusion detection field:

1) at node level, we are going to apply the defensive programming approach (coming from the dependability field) to data corruption detection. The challenge is to determine which invariant/properties must be and can be verified either at runtime or statically. Regarding illegal flow detection, we plan to extend this method to build anti-viruses and DBMS tools by determining viruses signatures.

2) at the set of nodes level, we are going to revisit the distributed problems such as clock synchronization, logical clocks, consensus, properties detection, to extend the solutions proposed at node levels to cope with distributed flow control checking mechanisms. Regarding illegal flow detection, one of the challenges is to enforce the collaboration and consistency at nodes and set of nodes levels to obtain a global intrusion detection mechanism. Regarding

the data corruption detection approach, the challenge is to identify local predicates/properties/invariants so that global predicates/properties/invariants would emerge at the system level.

3) Open distributed system level: no specific work identified yet.

IV. Privacy

In our world of ubiquitous technologies, each individual constantly leaves digital traces related to his activities and interests which can be linked to his identity. In forthcoming years, the *protection of privacy* is one of the greatest challenge that lies ahead and also an important condition for the development of the Information Society. Moreover, due to legality and confidentiality issues, problematics linked to privacy emerge naturally for applications working on sensitive data, such as medical records of patients or proprietary datasets of enterprises.

Privacy Enhancing Technologies (PETs) are generally designed to respect both the principles of *data minimization*[1] and *data sovereignty*[2]. In the CIDre project, we will investigate PETs that operate at the three different levels (node, set of nodes or open distributed system) and are generally based on a mix of different foundations such as cryptographic techniques, security policies and access control mechanisms just to name a few. Examples of domains where privacy and utility aspects collide and that will be studied within the context of CIDre include: identity and privacy, geo-privacy, distributed computing and privacy, privacy-preserving data mining and privacy issues in social networks.

Here are some concrete examples of our research goals in the privacy field:

1) at the node level, we aim at designing privacy preserving identification scheme, automated reasoning on privacy policies [9], and policy-based adaptive PETs.

2) at the set of nodes level, we plan to augment distributed algorithms (*i.e.*, consensus) with privacy properties such as anonymity, unlinkability, and unobservability.

3) at the open distributed system level, we plan to target both geo-privacy concerns (that typically occur in geolocalized systems) [10] and privacy issues in social networks [11]. In the former case, we will adopt a sanitization approach while in the latter one we plan to define privacy policies at user level, and their enforcement by all the intervening actors (*e.g*, at the social network sites providers).

[1]The data minimization principle states that only the information necessary to complete a particular application should be disclosed (and no more). This principle is a direct application of the legitimacy criteria defined by the European data protection directive (Article 7).

[2]The data sovereignty principle states that data related to an individual belong to him and that he should stay in control of how this data is used and for which purpose. This principle can be seen as an extension of many national legislations on medical data that consider that a patient record belongs to the patient, and not to the doctors that create or update it, nor to the hospital that stores it.

V. Bridging our Research Topics

To bridge the gap between the previously described activities, we have identified interests for each pair of research topics. We aim at working on the following topics:

– Trust/Intrusion Detection: attacks against trust detection, trust modulation according to attack detected;

– Intrusion Detection/Privacy: attacks against privacy detection, privacy preserving intrusion detection and alert fusion;

– Privacy/Trust: privacy preserving distributed computation of trust, anonymous crediential.

VI. Conclusion

The CIDre team whose research objectives have been presented in this paper will be created by the end of 2011.

References

[1] A. Myers, F. Schneider, and K. Birman, "Nsf project security and fault tolerance, nsf cybertrust grant 0430161."

[2] M. Hurfin, J. Le Narzul, F. Majorczyk, L. Mé, A. Saidane, E. Totel, and F. Tronel, "A dependable intrusion detection architecture based on agreement services," in *Proc. of the 8th Int. Symposium on Stabilization Safety and Security*, 2006.

[3] E. Anceaume, F. Brasileiro, R. Ludinard, B. Sericola, , and F. Tronel, "Modeling and evaluating targeted attacks in dynamic systems," in *Proceedings of DSN*, 2011.

[4] E. Anceaume, F. Brasiliero, R. Ludinard, and A. Ravoaja, "Peercube: an hypercube-based p2p overlay robust against collusion and churn," in *Proceedings of the International Conference on Self- Autonomous and Self-Organizing Systems (SASO)*, 2008.

[5] J. Zimmermann, L. Mé, and C. Bidan, "An improved reference flow control model for policy-based intrusion detection," in *Proceedings of the 8th European Symposium on Research in Computer Security (ESORICS)*, October 2003.

[6] G. Hiet, V. Viet Triem Tong, L. Mé, and B. Morin, "Policy-based intrusion detection in web applications by monitoring java information flows," in *3nd International Conference on Risks and Security of Internet and Systems (CRiSIS 2008)*, 2008.

[7] J.-C. Demay, E. Totel, and F. Tronel, "Detecting illegal system calls using a data oriented detection model," in *Proceedings of the 26th IFIP International Conference (IFIP SEC 2011)*, 2011.

[8] O. Sarrouy, E. Totel, and B. Jouga, "Application data consistency checking for anomaly based intrusion detection," in *The 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS 2009)*, 2009.

[9] G. Piolle and Y. Demazeau, "Obligations with deadlines and maintained interdictions in privacy regulation frameworks," in *8th IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'08)*, 2008.

[10] S. Gambs, M.-O. Killijian, and M. N. del Prado, "Show me how you move and i will tell you who you are," in *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS (SPRINGL'10)*, 2010.

[11] E. Aïmeur, S. Gambs, and A. T. Ho, "Towards a privacy-enhanced social networking site," in *Proceedings of 5th International Conference on Availability, Reliability, and Security (ARES 2010)*, 2010.

System Security Research at Newcastle

Jeff Yan
School of Computing Science
Newcastle University, UK
Jeff.Yan@ncl.ac.uk

Abstract—***We describe current system security efforts and future research roadmap of Lab of Security Engineering at Newcastle University, England.***

I. INTRODUCTION

Our research in the Laboratory of Security Engineering at Newcastle University, England (http://homepages.cs.ncl.ac.uk/jeff.yan/lab.html) focuses on real-world security with the aims of 1) understanding what goes wrong with security in the real world, and 2) developing deployable security solutions that are not only scientifically sound, but also practically useful and usable.

We are interested in most aspects of computer and network security, both theoretical and practical. Our previous contributions illustrate both our view of security and research methodology. Namely, security fails not only because of the lack or failure of technical mechanisms, but also because of failures of other issues such as usability and motivation, and therefore an interdisciplinary approach is needed to tackle (many) security problems.

Our recent works are primarily in the areas of usable security, system security, applied cryptography, and novel data structure and algorithm (in particular those that are relevant to computer security problems). Our work aims to improve real-world security by

1) inventing or discovering novel attacks to inform better security design,

2) designing novel security mechanisms, including both systems and primitives, and

3) understanding human aspects of security, so that security design is well-informed with human issues.

Most of our projects concern with some human elements in one way or another, and our long-term vision is to improve real-world security by tackling its weakest link, i.e. human issues.

II. RECENT AND CURRENT WORK

A number of our recent and ongoing projects are briefly introduced as follows.

CAPTCHA robustness and usability. CAPTCHA was invented to tell apart humans and computers, and its robustness research was dominated by computer vision, pattern recognition and machine learning researchers before our contribution. Inspired by classic cryptanalysis thoughts, we have established a new method [3] for the study of CAPTCHA robustness. This method systematically searches for exploitable invariants that CAPTCHA generators fail to remove. Applying our own methodology, we have broken all major designs, including the text schemes deployed by Microsoft, Yahoo, Google as well as reCAPTCHA [1][2]. Another fundamental contribution we have made in this area is to establish some basic principles for CAPTCHA design [1][6].

Collaborated with Microsoft Research Asia, we systematically studied the design of image recognition CAPTCHAs, with the aim of developing a new scheme for large-scale real-life applications such as Gmail and Hotmail – the latter consumes 100 millions of CAPTCHAs on a daily basis. We performed a security analysis of representative schemes. For the schemes that remain not broken yet, we developed novel attacks. For the schemes that known attacks are available, we proposed a theoretical explanation why those schemes have failed. More importantly, we have defined a simple theoretical framework for guiding the design of better image recognition CAPTCHAs. This topic area has so far had very little structure; our framework is the first attempt to bring in some order.

For both text and image recognition CAPTCHAs, we are keen to study and improve their usability as well **Error! Reference source not found.**[6][7]. We have also studied the robustness of some other schemes such as [4][11].

This ongoing project also contributes to a better understanding of the difference in computational capabilities between humans and machines, and to advancing the understanding of the AI problems (e.g. image segmentation) that underpin common CAPTCHAs.

Graphical passwords. Our work has focused on recall-based graphical passwords for mobile systems. We proposed, implemented and tested a novel scheme, Background Draw a Secret [8]. We also designed and rigorously evaluated a number of novel techniques to defend BDAS (and other graphical password systems) against shoulder surfing attacks [9].

Online game security. We were among the earliest researchers examining security of online games, and still

active in this area with the latest paper being [10]. An active project is funded by Microsoft Research to discover novel cheats and mitigating cheat in online games.

We have also done some work in designing human computation games [13], and addressing security issues in these games [12].

Phishing and Spam. We have designed a novel anti-phishing solution, and investigated novel extensions to the elegant data structure of Bloom filter, applying them to improve the performance of collaborative spam detection systems such as Razor and Distributed Checksum Clearinghouse.

III. FUTURE DIRECTIONS

We never intend to work only on domains or topics that we are familiar with. Instead, we are open-minded, and always keep our eyes open for security problems that are interesting, relevant, challenging, and potentially can lead to a significant impact. Topics that on our radar for future investigation include cloud security, smart grid security, banking security, cybercrime and forensics, among others.

We are keen to exchange ideas with and develop collaboration with researchers from both inside and outside of EU, and expect that the SysSec workshop will be an ideal venue for such purposes.

IV. ACKNOWLEDGEMENTS

We thank the anonymous reviewers, whose comments and suggestions helped to improve this position paper.

REFERENCES

[1] Ahmad El Ahmad, Jeff Yan, Mohamad Tayara. “The robustness of Google’s CAPTCHAs”. Submitted.

[2] J Yan and A S El Ahmad. “A Low-cost Attack on a Microsoft CAPTCHA”, 15th ACM Conference on Computer and Communications Security (CCS’08). Virginia, USA, Oct 27-31, 2008. ACM Press. pp. 543-554.

[3] Jeff Yan, Ahmad Salah El Ahmad, "Captcha Robustness: A Security Engineering Perspective," IEEE Computer, pp. 54-60, February, 2011.

[4] Ahmad El Ahmad, Jeff Yan, Lindsay Marshall. “The Robustness of a New CAPTCHA”. European Workshop on System Security (EuroSec 2010), April, 2010, Paris, France. Associated with ACM SIGOPS EuroSys conference. ACM Press. Pp.36-41.

[5] J Yan and AS El Ahmad. “Usability of CAPTCHAs - Or ‘usability issues in CAPTCHA design’”. The 4th Symposium on Usable Privacy and Security (SOUPS’08), Carnegie Mellon University, July 23-25, 2008. ACM Press. pp 44-52.

[6] B. B. Zhu, J. Yan, Q. Li, C. Yang, J. Liu, N. Xu, M. Yi, K. Cai, “Attacks and Design of Image Recognition CAPTCHAs,” ACM CCS 2010, pp. 187-200.

[7] Ahmad Salah El Ahmad and Jeff Yan. “Colour, usability and security: a case study”,Technical Report CS-TR-1203, School of Computing Science, Newcastle University, England. 2010. To appear at IEEE *Internet Computing* magazine.

[8] P Dunphy and J Yan, “Do background images improve Draw a Secret graphical passwords?”, 14th ACM Conference on Computer and Communications Security (CCS '07), Virginia, USA, Oct 29 – Nov 2, 2007. ACM Press. pp 36-47.

[9] Haryani Zakaria, David Griffiths, Sacha Brostoff, Jeff Yan. “Shoulder Surfing Defense for Graphical Passwords”. Technical Report CS-TR-1194, School of Computing Science, Newcastle University, England, 2010. To appear at SOUPS’11.

[10] J Yan. “Collusion Detection in Online Bridge”. Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10), Atlanta, Georgia, USA, July 11–15, 2010. pp.1510-1515.

[11] Jeff Yan, Ahmad Salah El Ahmad, "CAPTCHA Security: A Case Study," IEEE Security & Privacy, vol. 7, no. 4, July/Aug. 2009. pp. 22-28.

[12] Su-Yang Yu and Jeff Yan. “Security Design in Human Computation Games”, 18th International Workshop on Security Protocols, Cambridge, UK, March 24-26, 2010. Lecture Notes in Computer Science xxxx Springer. pp xx-xx (in press).

[13] Jeff Yan, Su-Yang Yu. “Streamlining Attacks on CAPTCHAs with a Computer Game”. Proc. of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-09), Pasadena, California, USA, July 11-17, 2009. pp. 2095-2100.

Security Research at NASK

Supporting the operational needs of a CERT team and more

Piotr Kijewski, Adam Kozakiewicz
NASK – Research and Academic Computer Network
Warsaw, Poland
piotr.kijewski@cert.pl, adam.kozakiewicz@nask.pl

***Abstract*—The paper presents the current and future research in the area of network and computer security at NASK, focusing on activities of CERT Polska and NISM teams. A large part of the activities are driven by the operational needs of CERT Polska, but independent research themes are also pursued. Most of the work concentrates either on threat detection methods (server and client honeypots) and threat intelligence or on distribution of security-related information. Other research areas are briefly described as well.**

Keywords—threat detection, threat intelligence, botnets, honeypots, client honeypots, early warning, virtualization, multilevel security, trust management

I. Introduction

Network and computer security is a very active research topic at NASK. Several projects are under development at all times, involving several teams. This paper will concentrate on activities of two teams:

- CERT Polska is a Computer Emergency Response Team operated by NASK that handles incidents related to the .pl namespace. A part of the team's work is focused on researching new detection and analysis methods and developing tools to aid this process.
- Network and Information Security Methods Team (NISM) is a part of the NASK Research Division dealing with security problems.

Both teams cooperate in most (but not all) of their projects, offering different views of the same problem. Many of these projects are supported by programmers from the Software Development Department. Security-related research conducted by other teams at NASK is omitted here due to space constraints.

The research conducted by both teams is strongly correlated, but different. The CERT Polska team is focused on topics relevant to their everyday operations – incident handling, propagation of information, data collection, correlation and analysis. NISM, as part of the Research Division, is interested in a wider range of topics, including theoretical research.

The projects and research activities at NASK can be broadly grouped into three main categories. In section II we present threat detection and analysis efforts – from honeypots to threat intelligence and attack attribution. Section III focuses on propagation of security-related information. Section IV summarizes NASK's interests in data protection, including handling of classified information and access control techniques. Section V lists the themes which are not directly connected to any of the above groups. Finally, a short summary is available in section VI, followed by a list of references.

II. Threat detection and Threat Intelligence

Threat detection and threat intelligence are two areas of security research driven by the operational needs of CERT Polska. The CERT's primary mission is to handle incidents for the Polish .pl namespace, provide watch and warning services and improve security awareness. As security incident handlers, CERT teams are often among the first to learn of new threats on the Internet, gaining invaluable hands-on experience. At the same time, they require tools to better handle and keep up with the constantly evolving threat landscape. The CERT's needs in terms of detection, analysis and threat attribution are under constant review, and are the primary drivers of research in those areas. Much of this research involves detection of compromised IPs, malicious URLs, analysis of malicious code through the use of honeypots and sandbox technologies.

A. HoneySpider Network Project

The HoneySpider Network [1] project (carried out together with GOVCERT.NL and SURFnet) is an effort to build a platform for the bulk processing and detection of malicious URLs that infect through drive-by download attacks. For this purpose both low and high interaction client honeypots are used. Low interaction client honeypots are essentially browser emulators that attempt to use various heuristics to detect signs of malicious activity by a web site. We employ machine learning techniques to assess whether a site is malicious or suspicious. For the high interaction component we use a modified version of the Capture-HPC [2] client honeypot. Capture-HPC drives a real browser inside a virtual machine running Windows and logs system

calls responsible for registry, files and process changes. It enables the capture of malware. The project has resulted in a working system, operational at CERT Polska and a few other CERTs worldwide.

Future work for HoneySpider involves designing and implementing a second version of the system. This is necessary to better keep up pace with the changing client side threat environment. We envision a more elastic, pluggable and scalable open source framework that will enable the hooking of externally developed solutions, for example, such as PhoneyC [3] as well as chaining various analyzers using different analysis techniques. This should allow other developers to contribute to the framework. We are currently carrying out research into detection of malicious PDF and Flash files. The detection methods developed as part of this research will be implemented as plugins into the new solution.

B.ARAKIS Project

The goal of the ARAKIS Project [4] was to develop an Internet "Early Warning" system aimed at detecting and characterizing threats that spread through active means, for example worms. The system was first designed when worms were at their height, back in 2004. It is based primarily – but not only – on server side honeypots. It employs a unique detection mechanism based on clusterization of packet payload using similarity of content measures as a metric. The assumption is that the fact that packets are sufficiently different from previous ones to form a new cluster may be a sign of a new attack. In 2007, an implementation of ARAKIS called ARAKIS-GOV was deployed together with the Polish Internal Security Agency. The system successfully detected many worms and botnet activity, including the spread of the Conficker worm.

While ARAKIS-GOV is still operational and successful at detecting threats spreading through active means, we are planning to start work on a new version of the ARAKIS system focusing more on detecting Web based attacks.

C.WOMBAT Project

The WOMBAT (Worldwide Observatory of Malicious Behaviors and Attack Threats) Project [5] is an EU 7th Framework Programme initiative that aims at providing new means to understand the existing and emerging threats to the Internet infrastructure and the services this infrastructure supports. This is motivated by the transformation of the type of threats found on the Internet into well organized, professional endeavors. Organization's behind these threats are quick to discover new attack vectors and design malware to defeat current security best practices.

As part of the project, NASK contributed the HoneySpider Network system (see II.A) and integrated the solution with other systems under WOMBAT through the WOMBAT API. Furthermore, it is developing a mechanism for the reduction of false positives in Capture-HPC through a machine learning technique. Work is also being carried out to better visualize relationships between malicious URLs detected thus getting a better picture of the hidden malware distribution network.

D.SOPAS Project

Automated response to threats detected by honeypots or other threat detection tools is the aim of project SOPAS – a system for defense against network attacks, developed under a grant from the Polish Ministry of Science and Higher Education by a consortium of three research institutions. The protected network may consist of several domains controlled by different organizational units. Each domain contains a SOPAS decision module, which collects the data generated by sensors, correlates them and sends instructions (e.g. firewall rules) to reaction elements. The decision modules form a peer-to-peer network, coordinating the operation of the entire network protection infrastructure.

As the main contribution to this project, NASK will create the interface allowing SOPAS to connect with existing ARAKIS and/or HoneySpider Network installations in the monitored domains, using them as additional sensors. NASK is also currently researching visualization techniques which will be used to present collected data and network situation.

E.Other Research

Apart from the above formalized projects, there is a wealth of activity going on in terms of threat detection and threat intelligence research.

Our CERT team receives tens of millions incidents yearly from multiple external and internal systems. We recently published a report analyzing over 12 million cases relating just to Polish networks [6]. These incidents were categorized into spam, scanning, phishing, bot, malicious URLs, C&C servers, fastflux, DDoS, sandbox (adresses contacted by malware running in sandboxes), and other categories. We are planning to expand analysis of such incidents by utilizing data mining techniques to better identify relationships between incident reports.

NASK operates the .pl registry and .pl primary nameserver. Thanks to this it has a unique view of DNS queries for .pl domains. These queries can be analyzed for anomalies leading to the identification of malicious or C&C domains. CERT Polska is currently developing techniques for automated detection of such cases. At the same time, NISM is analyzing the statistical properties and similarities of domain names treated as character strings, both in the .pl domain and in publicly available collections such as alexa.com or malwaredomains.com. The analysis is focused

on the differences in structure between typical malicious and benign domains, hoping to provide useful heuristics for picking out potentially suspicious domains in huge collections of mostly benign addresses.

Botnet monitoring and malware evolution is another topic of interest at CERT Polska. In particular, due to the character of our services, we are interested in information stealing Trojans that focus on banks such as Zeus or SpyEye. We are actively researching and developing our capabilities at analyzing such cases. Current work involves amongst others, automated extraction of configuration files to identify banks being targeted – this is expected to be part of our early warning services.

Although malware on smartphones has been heralded as a threat for many years, it appears to be materializing only now. While it may be too early yet for the development of mobile honeypots, it does appear that we are in need of technologies that can help distinguish between benign and malicious software on smartphones. Furthermore, the recent evolution of banking Trojans such as Zeus to the mobile platform (Zitmo [7]) in order to defeat SMS based two-factor authentication means that we must improve our analysis capabilities in order to provide improved early warning services. This area is a subject of future research.

III. Distribution of Security-Related Information

Effective sharing of security-related information is essential to stem the tide of Internet threats. Distribution of operational information, e.g. alerts and data collections, is a necessary part of threat detection and threat intelligence projects described in the previous section, like WOMBAT and SOPAS. However, other kinds of information need to be distributed as well, reaching end users with messages they can understand and use to their advantage.

A. FISHA and NISHA Projects

The recently finished FISHA (Framework for Information Sharing and Alerting) project aimed to solve the problem of reaching home users and SMEs with the information they need to keep their systems secure. A large part of the project was devoted to identification of the different user groups, their typical activity on the Internet and the kind of information most useful to them.

The technical part of the project resulted in development of a prototype of a European Information Sharing and Alert System (EISAS) consisting of a network of portals providing security-related information joined by a peer-to-peer network assisting the exchange of information ranging from security advisories and warnings to best practices documents, internet safety movies, etc. Technical contribution of NASK was the design and implementation of this underlying peer-to-peer network and its information structure.

To reach the end users it is crucial to convert the available information to a form most suitable for a given target group (e.g. the utility of providing detailed, technical advisories to young kids is minimal). The network is designed to enable cooperation between very different users. Security companies, researchers and software vendors can act as sources of highly technical security information. Editors, journalists and translators use this information, creating documents for different target groups. CERTs can act as members of both of these groups. Finally, portal owners act as information consumers, including the available documents in their portals, which may be either directly accessible to end users, or serve as information sources for traditional media.

There are still improvements to be made in the FISHA network. Also, while information sources and consumers are relatively easy to find, the selection of best documents, translation and adaptation to different groups is likely to be a bottleneck of the system and requires organizational support. Some basic considerations were presented as part of FISHA, but much remains to be done. These aspects are the focus of the NISHA (Network for Information Sharing and Alerting) project, which – if funding will be granted – will start in the last quarter of this year.

IV. Access Control and Classified Information

Since its beginnings, the interests of NISM included the subject of access restrictions. Most of the theoretical work concerns access control models, mainly trust management languages. On the practical side, secure processing of classified data offers a wide spectrum of issues to analyze.

A. Trust Management

Traditional access control models (discretionary, mandatory or role-based) share a common problem – they base the decisions on the identity of the principal requesting access. This approach works well in local systems, where all users are known a priori, but not in a large distributed system without a central authority assigning rights to users.

The most promising solution is the trust management approach, where access decisions are based on credentials assigned to the principal. Delegation of authority by trusting credentials signed by some other parties makes it possible to decentralize access control. The system can be made more even more powerful by the addition of an inference system, making access decisions based on available credentials and inference rules, eliminating the need for direct specification of each allowed action.

This functionality is essential in many applications, but formal analysis is necessary to show that the approach is not inherently flawed. The formal model is available in the literature as a family of Role-based Trust management languages (RT) [8]. NISM works on extensions to this

model. Current work focuses on providing time restrictions on the validity of credentials, making the model more suited for practical use, where rights are rarely granted forever. A proof of soundness and completeness of an inference system for the RT^T language with these extensions was recently developed and will soon be published [9, 10]. Semantics of the modified language are being defined. Further research will focus on other practical aspects of the model, including access scenarios.

B.BSDZS Project

Due to legal restrictions, classified data must currently be processed on separate physical machines, resulting in unnecessary costs. BSDZS (acronym of the Polish name of the project: Secure Workstation for Special Applications) project's aim is to verify whether virtualization technology, supported with the trusted computing infrastructure, can provide sufficient isolation of different virtual systems to enable multilevel secure processing of information. Trusted computing is seen here as the way to guarantee physical separation of data wherever possible. The expected outcome of the project is a working technology demonstrator of a secure workstation. Each of the virtual machines running on the workstation will be assigned to a different security domain.

The scope of the project naturally includes work on secure configurations of the host system and the guest systems, modifications to the virtualization software and integration of all of these parts and the necessary hardware to form a complete workstation. As added value, the consortium will develop additional security measures: cryptographic protection of removable and non-removable volumes, a new access control model based on trust management and new user authentication mechanisms, both biometric and non-biometric. Since the product of the project is supposed to be certifiable, or at least as close to it as possible, formal models will be extensively used and advanced audit mechanisms will be implemented.

The project is developed by a consortium of four polish research units under a grant from the Ministry of Science and Higher Education and is currently in early stages of development. NASK is involved in almost all tasks of the project, but its main input will be in the area of access control and user authentication.

V.Other Research Topics

The research goals of CERT Polska and NISM are not limited to the currently pursued themes. Some of the areas for future investigation include:

- Security of critical ICT infrastructure. NASK has already been involved in a project dealing with this problem [11]. The main focus was on simulation of threats to the critical infrastructure, seen as a tool for training of crisis management teams, verification of scenarios for contingency planning and source of prognoses aiding action during actual crisis.
- Security of sensor networks. The Network Control Team in the Research Division is already active in the field of sensor networks. The problem of security of this type of systems cannot be avoided and will probably result in cooperation with NISM.
- Security of energy distribution monitoring networks. The Network Control Team is involved in research regarding networks of intelligent meters. As in case of sensor networks, security of this type of solutions is an interesting topic for future research at NISM.

VI.Summary

As our lives are getting increasingly dependent on the Internet and information systems, network threats have huge potential for impact as well as being more widespread, necessitating constant progress in security-related research. NASK, as a research institute and network provider, is well prepared to respond to this threat. Experienced teams taking part in several internal, national and international projects in this field and performing independent internal research, guarantee interesting results for years to come.

References

[1]Piotr Kijewski, Carol Overes, Rogier Spoor, "The HoneySpider Network – fighting client side threats" 20th Annual, FIRST Conference on Computer Security Incident Handling, Vancouver, Canada, June 2008.

[2]Capture-HPC Client Honeypot, The Honeynet Project, https://projects.honeynet.org/capture-hpc

[3]Jose Nazario, "PhoneyC: A Virtual Client Honeypot", In Proceedings of the USENIX Workshop on Large-Scale Exploits and Emergent Threats, 2009

[4]ARAKIS – An Internet Early Warning System – Public Dashboard, http://www.arakis.pl/en/index.html

[5]WOMBAT – Worldwide Observatory of Malicious Behavior and Attack Threats, http://www.wombat-project.eu

[6]CERT Polska Annual Report for 2010, http://www.cert.pl/PDF/Raport_CP_2010.pdf

[7]CERT Polska blog entry, "ZITMO: The new mobile threat", http://www.cert.pl/news/3193/langswitch_lang/en

[8]Li N., Mitchell J.C.: RT: A Role-Based Trust Management Framework. In: 3rd DARPA In-formation Survivability Conference and Exposition (DISCEX III), pp. 201-212, 2003

[9]A. Felkner, A. Kozakiewicz, „Czasowa ważność poświadczeń języka RTT+", Konferencja Sieci Komputerowe, in press (in Polish)

[10]A. Felkner, A. Kozakiewicz, "Time Validity in Role-Based Trust Management Inference System", unpublished

[11]A. Kozakiewicz, A. Felkner, T.J. Kruk, C. Rzewuski, K. Fabjański, "Simulation analysis of threats to ICT infrastructure" ("Symulacyjna analiza zagrożeń dla infrastruktury teleinformatycznej") (chapter in book: "Modele zagrożeń aglomeracji miejskiej wraz z systemem zarządzania kryzysowego na przykładzie miasta stołecznego Warszawy", WAT, Warsaw, ISBN 978-83-61486-22-0), ss. 679-699, 2009 (in Polish)

The security aspects of the research activities in IICT-BAS

Acad. Kiril Boyanov
Institute of Information and Communication Technologies
Bulgarian Academy of Sciences
Acad. G. Bonchev St., Block 25A, 1113 - Sofia, BULGARIA
E-mail: boyanov@acad.bg

***Abstract* - The paper presents information on past, present and future research activities in IICT-BAS in the field of ICT security. The main directions of these activities are: critical infrastructure cyberattacks protection, security of distributed systems, security of social networks and dependability of distributed systems.**

Keywords:ICT security, critical infrastructure security, distributed systems security, dependability, social networks security.

I. INTRODUCTION

The Institute of Information and Communication Technologies (IICT) at the Bulgarian Academy of Sciences (BAS) was created in 2010 as a successor of the Institute for Parallel Processing (IPP), the Institute of Information Technologies and the Institute of Computer and Communication Systems. The strategic objective for this act was to consolidate the research fragmentariness in the field of ICT in the academy that has more than half of a century history. This objective was also supported by the fact that IPP has been promoted twice for a "Center of Excellence" (2001-2004 and 2005-2007) by the European Commission and during the last ten years was a major national and significant regional player in the field of computer science.

In the rest of this paper a short description of some of the research activities in IICT-BAS will be outlined.

II. IICT-BAS RESEARCH ACTIVITIES

Generally, the mission of IICT-BAS is to carry out a fundamental and applied research in the field of computer science and ICT as well as to develop innovative interdisciplinary applications that are directly related to the main national and international priorities.

Some of the guiding lines in IICT-BAS research activities facing different security aspects are:

A. Computer networks and architectures

This activity is mainly referring to development and application of modern network technologies, distributed systems and facilities for network security, monitoring and control.

B. IT developments for emerging new security challenges

An interdisciplinary research team explores advances and applies methodologies and tools for IT governance and change management, design and analysis of architectures and capabilities, modeling and simulation for the security sector, including information security management. The activities are strongly supported and accepted both within NATO & EU integrated security sector governing level.

C. Development and maintenance of scientific infrastructure

The activity includes the development, monitoring and maintenance of Bulgarian Research and Education Network (BREN) and Grid clusters that are the biggest part of the National Grid infrastructure of the country. It is also connected with the scientific infrastructure of national significance and refers to research and support of important national infrastructure with key meaning for scientific and education institutes in the country.

D. Super computer applications

This activity is aimed at ICT driven contributions to science, technology, health, environmental protection, etc. It includes large scale computer simulation, high performance computer architectures and algorithms, computational linear algebra. The only one super computer in Bulgaria (BlueGene/P) is under the scientific maintenance of IICT-BAS.

Evidently, these research guiding lines demonstrate an environment with solid background and modern facilities for successful multi-aspect research in the field of ICT security problems.

III. PAST ACTIVITIES IN ICT SECURITY AREA

The working experience of IICT-BAS researchers on security problems of computer networks and distributed computer system is related to a solid background. General security problems in computer systems are considered in [1], [2] and [3]. The problems related to the information encoding and ciphers are object of publications [4], [5] and [6].

General network security problems are treated in [7], [8] and [9]. Network management and monitoring policy and related tools with relevance to the monitoring of the security issues are considered in [10], [11] and [12]. In these publications are described details of systems for monitoring the state of the services from the network infrastructure perfSONAR Multi-Domain Monitoring (MDM) which are used in GEANT network.

The publication [13] treats the realization of the Certification Authority (CA) for user authorization and identification in Bulgarian network and in GRID infrastructure.

The problems of user identification in distributed systems like GRIDs are discussed in [14] and [15].

A substantial work was done in the area of Information Technologies application for the Security Sector which includes:

- Methodologies and Tools for IT Governance and Change Management.
- Design and analysis of system architectures.
- Modeling and simulation for the security sector.
- Support to capabilities planning and security sector transformation.

These results [20] were supported by a number of national and international projects. A worth noting fact are the efforts for building a national Basic low-cost Environment for Simulation and Training (BEST) that will become a part of NATO Exercise Toolbox. This environment includes CIMIC communication aspects related to EDXL standard and a unique biological cryptographic solution [21].

During the EU/FP7 project ICT-FORWARD (http://www.ict-forward.eu/) IICT-BAS, as a partner in the project, was involved in the efforts to identify the emerging and future cyber threats [16]. The ultimate goal of the work was to identify the areas in which cyber threats could occur and cause serious and undesirable consequences. The research group from IICT-BAS was focused particularly on the threats to critical systems. Based on the specifics of these systems several areas were identified where security threats might grow in the future and where new solutions should be sought for [17]. The identified threats to critical systems (CSs) summarize the views of many experts both from information security, industrial automation and critical infrastructures. They reflect the general vision that critical systems can become an attractive target to cyber attacks and the cross-area of ICT and CS is an open field for security research.

IV. FUTURE DIRECTIONS IN ICT SECURITY

The main future directions of the research plans in IICT-BAS are related to several aspects:

A). To summarize and analyze possible cyberattack scenarios against Critical infrastructures (CI). The research of cyberattacks on CI will be based on modeling with different concepts, software environments and scenarios that allow both static/dynamic behavior and nature exploration. In these areas IICT-BAS researchers have significant experience. The recent Stuxnet attacks on SCADA show an evident necessity for effective and specific countermeasures in this domain. The role of the human factor and its analysis (HFA) is extremely important in these environments where the human-system interaction affects safety and could have serious consequences for the society.

Within HFA, IICT-BAS is already working in cooperation with subject matter experts and modern lab facilities in the framework of a research project funded by the National Science Fund (http://cleverstance.com).

B). We plan to work for further extension of the set of emerging cyber threats using as an initial base the White book classification resulting from the FORWARD project [16]. Additionally, we plan to implement a detailed questionnaires based survey and sensitivity analysis, which will result in an improved classification and scenarios that should allow to better foreseen the different emerging cyber threats following the methodology presented in [20].

C). In the field of distributed systems security we plan to summarize and analyze some of the security issues with possible cyber attacks on parallel systems like Grid environments. We have experience with the exploitation of Grid clusters located in the IICT-BAS. The core of the CSIRT (Computer Security Incident Response Team) for Bulgarian Grid infrastructure is located in the institute. This group of security officers also actively gathers and analyzes detailed information related to security incidents, potential threats and precautions from a huge archive of logs collected from many Grid nodes in the last few years. Our future plans and interests in this area are to do research on:

- Security issues specific to parallel processing applications which are typically executed on a distributed environments like Grid.
- Analyzing and developing additional software tools for automating and monitoring the security activities and administration.

The highly possible and desired evolution of this IICT-BAS activity is towards security problems in Cloud Computing environments.

IICT-BAS was a participant in one of the research projects related to developing future Internet technologies – PSIRP (Publish-Subscribe Internet Routing Paradigm) which is funded by the European Commission 7th Framework Program (http://www.psirp.org/). The main goal of this project was the research and development of a brand new future Internet architecture based on publish/subscribe paradigm. In this model, security instruments and mobile devices support are integrated in the model in principle [18]. Our plans are to analyze some of the possible security issues of PSIRP.

D). In the Social networks security area the plans are to implement a survey [19], case study of Facebook and Twitter and also some analysis about psycho-social and ICT aspects of the intrusions' motivation.

E). Our recent research interests are also related to cybersecurity in heterogeneous networks and application of

dependability mechanisms in network security. Based on our experience in dependability of distributed systems and networks there are plans for work on approaches to apply fault-tolerance mechanisms and techniques for improving IT security. We have developed algorithms for fault-tolerant clock synchronization in distributed real-time process control systems that, with the necessary modifications, could be implemented in sensor networks where the efficient use of system resources is a critical task.

V. CONCLUSIONS

The presented research activities of IICT-BAS will be realized from an international and interdisciplinary team within the next 3 years since 2011. The work will be performed in collaboration with the integrated security sector and in the framework of EU FP7 projects like SysSec "Network of Excellence in Managing Threats and Vulnerabilities in the Future Internet: Europe for the World" (www.syssec-project.eu/).

REFERENCES

[1] Ville E., N. Sinyagina, P. Borovska. Deploying Trusted Computing. Information technologies and Controls, 2009, pp 28-32, ISSN 1312-2622-1.

[2] Dobrinkova N., N. Sinyagina. Information security – bell la Padula model. Problems of Engineering Cybernetics and Robotics, 2009 62, pp. 15-20, ISSN 0204-98.

[3] Kolev A., N. Sinyagina. Discover of the Critical Directories in the Computer System. Collection of scientific works "Military-scientific forum", National military university Vasil Levski, pp. 76-82, 2006 (in Bulgarian).

[4] Sinyagina, N., B. Aleksandrov. 3-D Structure of Block Cryptografic Ciphers. Proceedings of the Fourth Bulgarian-Greek Scientific Conference "COMPUTER SCIENCE 2008", Kavala, Greece, pp. 791-797, 2008.

[5] Sinyagina, N., M. Yordanova. Distributed Secret on the Basis of the Linear Correcting Codes. Proceedings of the International Scientific Conference UNITECH-2008, Gabrovo, pp I-436-441, 2008.

[6] Aleksandrov B., N. Sinyagina. Modification of Block Cryptfgraphic Ciphers. Proceedings of the International Scientific Conference UNITECH-2008, Gabrovo, pp. I-474-480, 2008.

[7] Iliev, L., H. Turlakov. Current Problems in Network Security. Proceedings of the International Workshop on Network and GRID Infrastructure, Sofia, pp. 125-139, 2007.

[8] Boyanov K., D. Todorov, H. Turlakov. ICT, democracy, Internet treats and ethics. "Automation and Information" journal, pp. 7-12, 2008, (in Bulgarian).

[9] Sinyagina, N., S. Ruseva. Defense mechanisms against computer attacks 'Distributed denial of service' type. UNWE International conference "Management of secure related RMD research in support of defense industrial transformation", pp. 85-91, 2007.

[10] Hanemann A., V. Jeliazkov, O. Kvittem, L. Marta, J. Metzger, I. Velimirovic. Complementary Visualization of perfSONAR Network Performance Measurements. International Conference on Internet Surveillance and Protection (ICISP'06), IARIA/IEEE, Cap Esterel, France, pp. 6-6, 2006. (best paper award).

[11] Gajin, S., V. Jeliazkov, C. Kotsokalis, Y. Mitsos. Seamless Integration of Network Management Tools in a Multi-Domain Environment. 10th IFIP/IEEE International Symposium on Integrated Network Management, Munich, Germany, pp. 745-748, 2007.

[12] Jeliazkova, N., L. Iliev. Extending and Monitoring the Prefsonar Infrastructure. Proceedings of the International Workshop on Network and GRID Infrastructure, Sofia, pp. 36-41, 2007.

[13] Dimitrov, V., L. Iliev, L. Boyanov, H. Turlakov. Bulgarian Academic Certification Authority. Proceedings of the International Workshop on Network and GRID Infrastructure, Sofia, pp. 23-28, 2007.

[14] Weigold T., P. Buhler, J. Thiyagalingam, A. Basukoski, V. Getov. Advanced Grid Programming with Components: A Biometric Identification Case Study. Proc. IEEE COMPSAC, IEEE CS Press, pp. 401-408, 2008.

[15] Naydenova I., Kaloyanova K., Ivanov S. Multi-Source Customer Identification. Information Systems & GRID Technologies, 28-29 May 2009, Sofia, Bulgaria, pp.77-85.

[16] ICT FORWARD, White Book: Emerging ICT Threats, January 2010, http://www.ict-forward.eu/media/publications/forward-whitebook.pdf.

[17] E. Djambazova, M. Almgren, K. Dimitrov, E. Jonsson, "Emerging and Future Cyber Threats to Critical Systems", in J. Camenisch, V. Kisimov, and M. Dubovitskaya (Eds.): iNetSec 2010, LNCS 6555: Open Research Problems in Network Security, pp. 29 – 46, 2011.

[18] Dimitrov V., V. Koptchev. PSIRP project – Publish-Subscribe Internet Routing Paradigm. New ideas for future Internet. International conference CompSysTech'2010, Sofia, 17-18.06.2010. Published in "ACM International Conference Proceeding Series (ICPS)", ISBN:978-1-4503-0243-2, Vol. 471, pp. 167-171, 2010.

[19] Minchev Zl., M. Petkova "Information Processes and Threats in Social Networks", A Case Study. At Conjoint Scientific Seminar "Modeling and Control of Information Processes", 22 November 2010, Organized by College of Telecommunications, Institute of ICT - Bulgarian Academy of Sciences, Institute of Mathematics and Informatics - Bulgarian Academy of Sciences, Sofia, Bulgaria, 2010

[20] Bulgarian Knowledge Portal on OA & CAX, Available at: http://www.gcmarshall.bg/KP/Bulgarian_CAX_OA_Knowledge_Portal.htm

[21] Oscar, H., Z. Minchev, and D. Popivanov "Non-linear System for Digital Information Transmission", Patent 107414/20.12.02, BG 6, 2004 (published on 13.12.2006, BG 840 Y1).

Less is More – A Secure Microkernel-Based Operating System

Adam Lackorzynski, Alexander Warg
Operating Systems Group
Technische Universität Dresden, Germany
{adam,warg}@os.inf.tu-dresden.de

***Abstract*—Microkernel-based systems have gone through a steady development and current implementations have reached a new level of functionality. While the first systems started with the fundamental idea, latest systems offer a wide range of features. Experience showed that the most important feature, a secure system architecture, cannot be retrofitted into the system at a later stage but must be the core of it. A recent redesign of the architecture introduced capability-based access control on objects as the core mechanism upon which any functionality is built.**

Features of current systems include support for multi-cores, portability across different architectures, real-time execution and virtualization. Microkernels are built with the goal of being sufficiently generic to host multiple subsystems with differing isolation and security requirements. Although putting functionality into many different components sounds appealing, it is a severe burden on the implementation side. It must be possible to reuse existing software, and with the help of virtualization techniques it is possible to find a better split of components. This way systems with a small trusted computing base can be built without reimplementing existing functionality. One of the open questions is how such a split must be designed and can be implemented and offered in a generic way, given all the options current modern systems offer.

In this paper we report on the current state of the operating system developed at TU Dresden, focusing on its security mechanisms, and possible future direction that we envision with the ongoing changes in the hardware and software world.

***Keywords*-operating systems; design, security; virtualization**

I. Introduction

People use computer systems in their everyday life, be it desktop, laptops or smartphones. Those systems have to handle a broad range of different use-cases, such as web-browsing for information retrieval and entertainment but they are also handling privacy-sensitive data or are used for doing confidential corporate work. The increasing compute power of flexibility of modern mobile devices leads to situations where a single device shall support all usage scenarios, ranging from openess for entertainment purposes up to trustworthy protection of confidential information.

This requires that each subsystem must be isolated in a way that one subsystem cannot harm others. This includes protection of unwilling modification of data of an application as well as that one malicious subsystem cannot monopolize on a single resource, for example CPU time. Yet it must be possible that the subsystem can communicate with others in a well defined and controlled manner.

More generally not only communication relations between subsystems must be defined and controlled but access to all resources and functionality in the system. In this paper we will give an overview on the operating system developed at TU Dresden that is designed based on that key idea. We will continue with a short description of the history of the system and then continue with the system architecture of the current incarnation.

II. History

Our group has been working on microkernel-based systems since the early 90ies, all starting with the L4 microkernel designed and developed by Jochen Liedtke [1]. The initial focus has been on performance of the microkernel as this has been a major niggle for such systems. L4 showed that microkernels can be fast, if properly designed [2]. The first sophisticated system running on top of L4 has been a modified Linux kernel and benchmarks showed that such a system has comparable performance to a natively running system [2]. At that point the system still lacked a proper layer that arbitrated resources for multiple clients and abstracted from the pure kernel interface which only offers basic core functionality. This consideration led to the development of the L4 Environment that provides several system services split into multiple components as well as runtime support to ease development of applications. The L4 Environment provides the foundation for our research on microkernel-based systems [3]–[10].

Opposed to the Mach kernel the original L4 kernel family did not provide explicit communication channels and allowed direct addressing of any thread. There were several ideas to restrict the allowed communication. However, all of them kept the global naming scheme for threads. This decision attributed to the rule of achieving best IPC performance. As those solutions for restrictions were slipped over the existing system rather than tightly integrated they were never adopted beyond specific setups.

The growing importance of multi-core systems and the security drawbacks implied by the global naming of threads, as well as from the missing management APIs for kernel resources, led us to investigate a major redesign of the microkernel API. The design and implementation shall allow

efficient use of multi-core systems as well as a uniform access control mechanism to communication and kernel services.

III. Architecture

Modern software designs usually follow a set of patterns to achieve goals, such as maintainability, extensibility, robustness, security and real-time properties. From the maintainability and extensibility point of view, component-based design, and separation of concerns are of importance. For robustness and security reasons a strong spatial isolation is required. Information flow security can be provided for example by using reference monitors. For real-time properties timing requirements temporal predictability and isolation are required.

We propose an architecture that supports this modern software design patterns, and removes the shortcomings we learned from our previous development. As traditional microkernel-based system we make heavy use of spatial isolation via virtual address spaces. However, this isolation is not sufficient for our security goals. So we build on a generalized form of address spaces: protection domains (PDs). A PD shall provide, besides virtual memory, a local (virtual) view of all resources in the system, including communication channels and kernel services.

Our solution for this is to provide a fully object-oriented model, with object capabilities as references to objects in a different protection domain, including the microkernel's domain. Consequently, all invocations crossing a protection boundary are mitigated by a capability and are modeled as message passing to the object. The capabilities by themselves are kernel-protected references that are managed in a special address space per PD. Access rights to objects can be delegated by transferring capabilities via message passing.

The use of this kind of capabilities solves two problems, namely, the allocation channel caused by the global thread IDs, and the missing communication control, as well as the management of access rights to kernel services. The locally managed capabilities are also the key feature to transparently interpose any communication channel or kernel service access, the key feature for implementing a reference monitor.

Interestingly, the introduction of capabilities as a sole mechanism to reference kernel objects also provides an optimal solution for multi-core synchronization of the kernel-object life cycle. Accessing a kernel object can be done without any locking overhead.

The architecture is completed by a runtime environment on top of the functionality provided by the microkernel. The services in the runtime environment build upon the same object-oriented paradigm as described above.

In the following we will (shortly) describe core functionality of the system that enables further research and evaluation with and of microkernel-based systems.

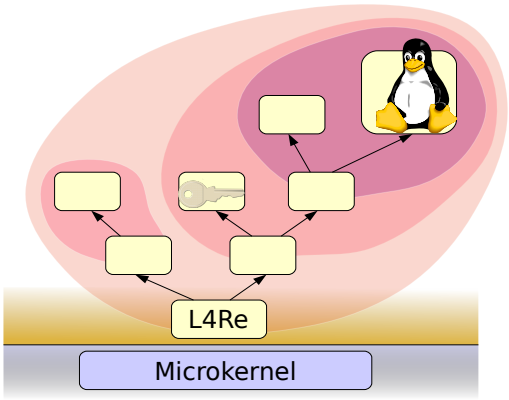


Figure 1. System architecture showing the recursive nature resulting from our propose architecture. The yellow rectangles depict individual components that form a tree structure. This structure results from the pattern how components launch other components. The elliptical shapes indicate that the environment that is visible to each component is controlled by the launching component. As shown the virtual machine running the Linux OS is a normal component and can appear anywhere in such a hierarchy.

A. Kernel

Being a microkernel, the kernel offers fundamental functionality only. The primary goal of the kernel is concerned with security, as to provide isolation between subsystems, secure naming, and secure communication. As such PDs and execution contexts are implemented in the kernel as well as low-level interrupt and exception handling.

B. L4 Runtime Environment

The runtime environment, L4Re, is the layer that provides the core operating-system functionality and abstracts from the pure kernel interface. It consists of services and libraries that provide functionality such as memory allocators, virtual memory management, and application launching.

C. Platform Management

The platform is managed by a central service that is responsible for platform initialization (e.g., PCI bus management). The service provides access to platform and device resources based on a given configuration, such that different applications can access hardware directly. The management of peripheral and platform resources is based on virtual system buses.

D. Multi-Processor

Systems have increasingly more than one core, which need to be handled by the operating system. From the applications point of view multiple processors are accessible through the scheduler interface. The communication mechanism is kept MP agnostic, apart from the performance, to enable transparently managed CPU assignment. However, the microkernel implementation of the scheduler interface does not do any automatic load balancing among

the available CPUs. Additionally, the interface also allows to express CPU-affinity requirements of an application. Scheduler services on user-level are then free to do any dynamic management of processing resources, based on the feature of transparent interposition.

E. Virtualization

With virtualization we mean running general-purpose operating systems, including their applications, on top of the microkernel-based system, as a user-level application. This kind of virtualization has gotten a ubiquitous feature in any modern operating system. The particular implementation of virtualization forms a gradation ranging from full hardware-assisted virtualization to OS re-hosting. In any case a proper microkernel support is either necessary or at least beneficial for the performance and the implementation effort of the virtualization solution [11]–[14].

In our research projects we mostly use a re-hosted version of the Linux kernel, L4Linux. The use cases reach from front-end user OS to the use as a provider for device drivers [15].

F. Scheduling, Real-Time and Security

Scheduling is often deemed to be a concern for real-time systems only. Nevertheless, modern desktop and embedded systems already have to fulfill real-time constraints. And most important security properties are also dependent on the scheduling of execution contexts on a system.

The requirements reach from apparent things, such as providing guaranteed shares of CPU time to different applications and virtual machines, and preventing abuse of CPU time, up to the prevention of timing channels caused by scheduling effects [16].

Current research investigates possibilities to combine real-time execution of diverse subsystems with the security aspects required in the system.

IV. Virtual Private Filesystem – An Example

The example of a privacy-enhanced file system that is constructed around the previously introduced principles shows the potential of this architecture. As the original proposals were built upon the assumption of communication control and a restricted view of an applications environment. The current design can use these features.

The Virtual Private Filesystem (VPFS) corporately uses traditional approaches comparable to privilege separation and virtualization techniques to create a file system with greatly enhanced confidentiality and integrity properties for persistent information [9]. Current research (jVPFS) focuses on enhancing robustness and availability properties of the stored data [17].

The general architecture of VPFS is a split of the functionality into a storage component and a security critical component. The storage component in the example is provided by an isolated virtual machine, running a Linux operating system. This allows the use of the greatly optimized file-system implementations and a rich set of block-device drivers, available in Linux. The security critical part is implemented in an isolated component, directly using the features of the runtime environment and the microkernel. This component is orders of magnitude smaller than the Linux-based storage component and uses cryptography to ensure the confidentiality and integrity of the data and meta-data stored via the storage component.

With a look at availability, we have to consider the storage component, because either accidental or malicious modification of the stored data can render the whole data unaccessible. To restrict the consequences of accidental corruption jVPFS provides improvements. To prevent malicious corruption we would propose to use techniques, such as intrusion detection for the Linux operating system, this techniques can be even refined by using the additional virtualization layer.

V. Future Research Directions

High-Level-Software—One of the main concerns of the runtime-environment layer is the management of low-level shared resources, such as physical memory, CPU time, and peripheral devices. The current situation for highly-secure and real-time systems is often to do a fully static partitioning of those resources. On the other end of the spectrum general-purpose operating systems provide the illusion of unlimited resources to their applications and do this with a fully dynamic management of those. We propose to develop a model where both worlds can be brought together and to move towards a more application and user centric management of resources. To achieve this goal the illusion of unlimited resources must be dropped. However, the goal to provide a good utilization of the computing resources prohibits a static partitioning of the system's resources in most cases. An application-centric resource management could likely also improve the quality of service provided by soft-real-time applications, such as video players.

Low-Level-Software—The ongoing development in the hardware area requires the operating system to adapt to the new directions and technologies and reflect them in the view of security in a microkernel-based system. Continuous updates are required to adopt the system to added features of the platform, for example when virtualization features are introduced.

Power management is a topic that is getting increasingly popular across all systems, either for cost and environmental reasons in server systems or for a practical battery lifetime in mobile devices. Using microkernel-based systems, power management is a challenging task as all components must be considered which requires appropriate interfaces and security considerations. Eventually the goal is to handle energy in the same way as CPU and memory today.

The growing number of cores available on one chip may lead to fundamental changes to the known hardware system architectures. There are hints that the traditionally provided coherency of caches will be relaxed for the sake of scalability. The SCC (Single-chip Cloud Computer) research system from Intel [18], which offers 48 non-coherent cores connected via a network, provides a forecast of how future CPUs may look like. Our goal is to explore the features of such a system with respect to virtualization techniques, as well as for robustness and error resilience in the context of transient errors.

In general, the size reduction of chips possible with the ongoing semiconductor development increases the error rate produced by this chips. This is caused by the shifting signal-to-noise ratio with smaller chip structures. A set of new technologies, such as software fault tolerance and hardware extensions for fault detection, are under ongoing development and we aim to explore the applicability these techniques to operating-system software with a special focus on the microkernel-based architectures [19].

Special purpose cores, such as graphics processors, are being integrated onto the same chip as the main processing unit. Today those graphics cores are still handled as an external device, although they are getting more general purpose and tighter integrated with every revision. However, a tighter integration of such functionality into the operating system and securely providing it user programs is still open to investigations.

Overall the future in operating-system research is manifold. Reacting on new hardware developments is required to have a usable system. However, an ever existing goal is to positively influence new hardware designs.

References

[1] J. Liedtke, "On μ-kernel construction," in *Proceedings of the 15th ACM Symposium on Operating System Principles (SOSP)*, Copper Mountain Resort, CO, Dec. 1995, pp. 237–250.

[2] H. Härtig, M. Hohmuth, J. Liedtke, S. Schönberg, and J. Wolter, "The performance of μ-kernel-based systems," in *Proceedings of the 16th ACM Symposium on Operating System Principles (SOSP)*, Saint-Malo, France, Oct. 1997, pp. 66–77.

[3] H. Härtig, R. Baumgartl, M. Borriss, C.-J. Hamann, M. Hohmuth, F. Mehnert, L. Reuther, S. Schönberg, and J. Wolter, "DROPS: OS support for distributed multimedia applications," in *Proceedings of the Eighth ACM SIGOPS European Workshop*, Sintra, Portugal, Sep. 1998.

[4] C. Helmuth, A. Westfeld, and M. Sobirey, "μSINA - Eine mikrokernbasierte Systemarchitektur für sichere Systemkomponenten," in *Deutscher IT-Sicherheitskongress des BSI*, ser. IT-Sicherheit im verteilten Chaos, vol. 8. Secumedia-Verlag Ingelsheim, May 2003, pp. 439–453.

[5] N. Feske and H. Härtig, "Demonstration of DOpE — a Window Server for Real-Time and Embedded Systems," in *24th IEEE Real-Time Systems Symposium (RTSS)*, Cancun, Mexico, Dec. 2003, pp. 74–77.

[6] J. Loeser and H. Härtig, "Low-latency Hard Real-Time Communication over Switched Ethernet," in *Proceedings of the 16th Euromicro Conference on Real-Time Systems (ECRTS)*, Catania, Italy, Jun. 2004, pp. 13–22.

[7] L. Reuther, "Disk Storage and File Systems with Quality-of-Service Guarantees," Ph.D. dissertation, TU Dresden, Fakultät Informatik, Nov. 2005.

[8] J. Brakensiek, A. Dröge, H. Härtig, A. Lackorzynski, and M. Botteck, "Virtualization as an Enabler for Security in Mobile Devices," in *Proceedings of the First Workshop on Isolation and Integration in Embedded Systems (IIES 2008), EuroSys 2008 Affiliated Workshop*, Glasgow, Scotland, UK, April 2008, pp. 17–22.

[9] C. Weinhold and H. Härtig, "VPFS: Building a virtual private file system with a small trusted computing base," *SIGOPS Oper. Syst. Rev.*, vol. 42, no. 4, pp. 81–93, 2008.

[10] L. Singaravelu, C. Pu, H. Härtig, and C. Helmuth, "Reducing TCB complexity for security-sensitive applications: three case studies," *SIGOPS Oper. Syst. Rev.*, vol. 40, no. 4, pp. 161–174, 2006.

[11] A. Lackorzynski and A. Warg, "Taming subsystems: Capabilities as Universal Resource Access Control in L4," in *Proceedings of the Second Workshop on Isolation and Integration in Embedded Systems, Eurosys affiliated workshop*, ser. IIES '09. ACM, March 2009, pp. 25–30. [Online]. Available: http://doi.acm.org/10.1145/1519130.1519135

[12] H. Schild, A. Lackorzynski, and A. Warg, "Faithful Virtualization on a Real-Time Operating System," in *Proceedings of the Eleventh Real-Time Linux Workshop*, Dresden, Germany, 2009, pp. 237–243.

[13] T. Frenzel, A. Lackorzynski, A. Warg, and H. Härtig, "ARM TrustZone as a Virtualization Technique in Embedded Systems," in *Proceedings of Twelfth Real-Time Linux Workshop*, Nairobi, Kenya, October 2010.

[14] S. Liebergeld, M. Peter, and A. Lackorzynski, "Towards Modular Security-Conscious Virtual Machines," in *Proceedings of Twelfth Real-Time Linux Workshop*, Nairobi, Kenya, October 2010.

[15] J. LeVasseur, V. Uhlig, J. Stoess, and S. Götz, "Unmodified Device Driver Reuse and Improved System Dependability via Virtual Machines," in *Proceedings of the 6th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, San Francisco, CA, Dec. 2004, pp. 17–30.

[16] M. Völp, "Provable protection of confidential data in microkernel-based systems," Ph.D. dissertation, TU Dresden, Aug. 2010.

[17] C. Weinhold and H. Härtig, "jVPFS: Adding Robustness to a Secure Stacked File System with Untrusted Local Storage Components," in *Proceedings of USENIX Annual Technical Conference. USENIX ATC '11*, Portland, OR, USA, June 2011.

[18] Intel, "The Single-chip Cloud Computer," http://techresearch.intel.com/ResearchAreaDetails.aspx?Id=27#scc, 2011.

[19] "DFG SPP 1500 - Dependable Embedded Systems," http://spp1500.itec.kit.edu/24.php.

Computer Security and Machine Learning: Worst Enemies or Best Friends?

Konrad Rieck
Technische Universität Berlin
Germany

***Abstract*—Computer systems linked to the Internet are confronted with a plethora of security threats, ranging from classic computer worms to involved drive-by downloads and bot networks. In the last years these threats have reached a new quality of automatization and sophistication, rendering most defenses ineffective. Conventional security measures that rely on the manual analysis of security incidents and attack development inherently fail to provide a timely protection from these threats. As a consequence, computer systems often remain unprotected over longer periods of time.**

The field of machine learning has been considered an ideal match for this problem, as learning methods provide the ability to automatically analyze data and support early detection of threats. However, only few research has produced practical results so far and there is notable skepticism in the community about learning-based defenses. In this paper, we reconsider the problems, challenges and advantages of combining machine learning and computer security. We identify factors that are critical for the efficacy and acceptance of learning methods in security. We present directions and perspectives for successfully linking both fields and aim at fostering research on intelligent security methods.

I. Introduction

The amount and diversity of security threats in the Internet has drastically increased. While only few years ago most attacks have been developed for fun rather than profit, we are now faced with a plethora of professional security threats, ranging from stealthy drive-by downloads to massive bot networks. These threats are employed by an underground economy for illegal activities, such as theft of credit card data, distribution of spam messages and denial-of-service attacks [see 1, 2]. As part of this development, attacks and malicious software have been systematically advanced in automatization and sophistication. Today's attack tools comprise a wide range of functionality, including various techniques for propagation, infection and evasion.

This change in the threat landscape confronts computer security with new challenges. Basically, computer security can be viewed as a cyclic process, which starts with the discovery of novel threats, continues with their analysis and finally leads to the development of prevention measures (Figure 1). This process builds on manual processing of data, that is, security practitioners take care of updating detection patterns, analyzing threats and crafting appropriate defenses. With the growing automatization of attacks, however, this cycle increasingly gets out of balance. The amount and complexity of threats renders manual inspection time-consuming and often futile. Thus, only a minor fraction of novel security threats is sufficiently analyzed for protecting computer systems in the future (arrows in Figure 1).

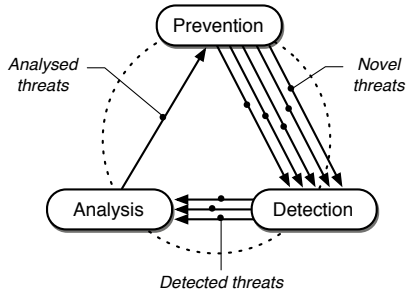


Figure 1: Computer security as a cyclic process.

Clearly, there is a need for techniques that help to analyze and fend off novel threats more quickly. If the attackers are systematically automatizing their instruments, why not try the same in the context of defense? The field of machine learning has been considered an ideal match for this problem, as learning methods are able to automatically analyze data and provide timely decisions, for example when identifying attacks against services [3, 4] or web browsers [5, 6]. Unfortunately, many researchers have exploited security solely as a playground for benchmarking learning methods, rather than striving for practical solutions. Despite a large body of work, only few research has produced practical results and there is notable skepticism in the security community about machine learning [7, 8].

In view of the possible advantages of learning-based defenses and the demand for alternative security measures, its worth reconsidering the combination of computer security and machine learning. In this paper, we study the problems, challenges and perspectives of linking the two fields. We identify key factors that contribute to the efficacy and acceptance of learning methods in security. While previous work has largely focused on making learning effective, we also emphasize the need for transparent and controllable

methods that can assist a human expert during analysis. Based on these observations, we present directions for future work on combining learning and security, where we point out new perspectives in detecting, analyzing and preventing security threats.

II. Problems and Challenges

Computer security fundamentally differs from other application domains of machine learning. The sound application of a learning method requires carefully addressing various constraints that are crucial for operating a security system in practice. While the performance of machine learning in other areas is often determined by a single quality, such as the classification accuracy, security involves several factors that require attention. Sommer and Paxson [8] have studied some of these factors for network intrusion detection. We extend this work to the generic application of machine learning and identify five key factors that impact the efficacy of learning-based security systems.

(a) *Effectivity:* First, any learning method applied in the context of security needs to be effective—either in detecting, analyzing or preventing threats. In contrast to other areas, this effectivity is highly problem-specific and may involve several quality metrics. For example, an intrusion detection system must accurately identify attacks as well as attain a reasonable low false alarm rate, as otherwise it is inapplicable in practice.

(b) *Efficiency:* A second important factor is efficiency. The main motivation for using learning methods in security is their ability to automatically provide results. Thus, learning needs to be fast to achieve a benefit over conventional security techniques. A good example is the work of Bayer et al. [9] which systematically improves the run-time performance of a clustering method for malicious software [10].

The majority of previous research has focused on these two factors when considering learning in security applications. Operating a system in practice, however, also requires addressing demands of practitioners. A main reason for the lack of machine learning in practical security is that effectivity and efficiency alone are not sufficient for designing successful security systems.

(c) *Transparency:* One central aspect is transparency. No practitioner is willing to operate a black-box system, which fails to provide explainable decisions. Fortunately, machine learning is not per se opaque and there exist several approaches for explaining the decisions of learning methods. One example is the visualization developed by Rieck and Laskov [11] which enables explaining the decisions of several learning-based intrusion detection systems.

(d) *Controllability:* Many security experts are deterred by the idea of handing over control to a learning method. This concern reflects a relevant problem of machine learning in security: learning-based systems must retain control of the operator, such that false decisions can be immediately corrected and the system adapted to dynamics in the environment. A key to this problem is changing the role of machine learning from operating totally autonomously to being actively supervised by a human expert.

(e) *Robustness:* Finally, any extension to a security system will become a target of attacks itself. Hence, machine learning must also deal with the problem of being attacked, for example, if an adversary tampers with the learning process or tries to evade detection and analysis [12, 13]. If considered during the design however, learning methods can be constructed in a robust manner and withstand different attack types, for example by randomization [14] and diversification [15] of the learning process.

We need to note here that none of these factors is new in the field of computer security and actually any practical security system should address these key factors—whether it applies machine learning or not. It thus comes as no surprise that even many conventional security instruments fail to satisfy all factors equally well. For example, many tools for attack detection suffer from false alarms and analysis systems for malicious software are often vulnerable to evasion. Nevertheless, it is a pity that a substantial body of previous work on learning for security has ignored these factors and there is a clear demand for research that brings the promising capabilities of machine learning to practical security solutions.

III. Perspectives and Applications

Based on these observation, we are ready to explore perspectives for machine learning in computer security. In view of the presented constraints and problems, this research is quite challenging and demanding. Sommer and Paxson [8] thus suggest to apply machine learning solely as a tool for preprocessing data. However, the ability of machine learning to provide protection from novel threats, only comes into effect if learning methods are deployed in the first front of defenses. Consequently, we herein argue that machine learning resembles a tool for directly strengthening the full cycle of computer security (Figure 1)—provided practical constraints and factors are carefully addressed.

In the following, we give a brief description of some promising applications of machine learning, including the detection of unknown network attacks, the automatic analysis of malicious software and the assisted search for vulnerabilities in software.

Proactive Detection of Attacks: One of the main advantages of machine learning over regular security techniques is its ability to detect anomalous events and identify novel attacks. Starting with the seminal work of Denning [16], learning methods for anomaly detection have been applied in different contexts of security. In particular, for network intrusion detection several methods have been developed which attain remarkable effectivity in practice [3, 4, 15]. However, all of these methods operate as black-box systems and do not provide interfaces for controlling and amending the detection process.

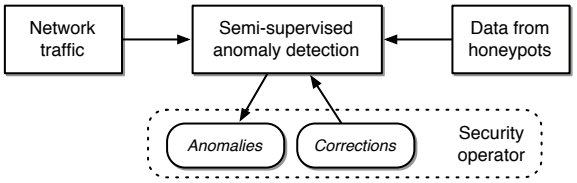


Figure 2: Schematic depiction of proactive threat detection.

A first step towards improving the practicability is thus the development of transparent anomaly detection methods which enable understanding and adapting their detection models during operation. One direction for addressing this problem is linking learned models back to their underlying features, for example, by automatically transforming statistical models into equivalent string patterns and rules. In contrast to numbers and vectors, strings and rules can be easily adapted and thereby allow an operator to carefully refine learning models in practice.

A further addition is the combination of anomaly detection and proactive techniques, such as server-based and client-based honeypots. Honeypots allow to automatically monitor malicious activity and provide a valuable source for training and calibrating learning methods. If combined with techniques for semi-supervised anomaly detection, these information can be directly fed into the learning process. For example, malicious web sites detected using honeypots and sandboxes [5, 17] can be transferred to a learning-based web proxy [e.g., 6] to create a dynamic defense against the threat of drive-by-download attacks. A corresponding detection system is illustrated in Figure 2.

Automatic Analysis of Threats: Another promising area for the application of learning in security is the analysis of threats. Security researchers are swamped by the amount of malicious activity in the Internet. Whether analyzing malicious programs, faked profiles in social networks or web pages of spam campaigns, in many settings there are thousands of data instances per day that need to be analyzed and fit into a global picture of threats. Machine learning can greatly assist in this process and provide a valuable instrument for accelerating threat analysis.

In particular, the automatic analysis of malware has proved to be a fruitful ground for learning. In the last years techniques for automatic classification and clustering of malware have been developed [9, 10, 18] which allow to identify malware variants as well as discover new families of malicious software. However, grouping malware into classes is only a one step in defending against malicious code. What is needed are analysis techniques that extract relevant information from these groups and propose patterns for signature generation to the analyst. Hence, novel learning systems need to be developed that automatically extract discriminative patterns from malicious code and guide the construction of anti-malware signatures.

While clustering has been studied for analysis of malicious programs, malicious web pages and malicious network flows, none of these approaches provides the ability to selectively correct the learned grouping. Often however, a security expert can clearly indicate some instances of a sample that need to be grouped into the same cluster and identify pairs that should to be placed in different groups. Currently, this information is lost. A possible direction of research hence lies in semi-supervised clustering methods that group data instances into clusters while at the same time satisfying the constraints given by a human expert. A corresponding system is illustrated in Figure 3.

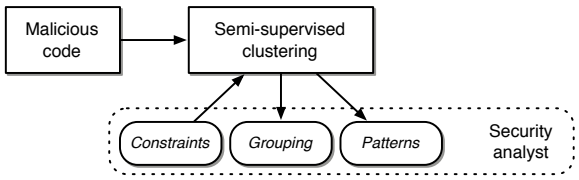


Figure 3: Schematic depiction of automatic threat analysis.

Assisted Discovery of Vulnerabilities: A third area for application of machine learning that has received almost no attention so far is the discovery of vulnerabilities. Security ultimately aims at eliminating threats and thus finding vulnerabilities is a crucial step for protecting computer systems. The search for security flaws is usually carried out in one of two extremes: on the hand vulnerabilities are often discovered in a brute-force manner using fuzzing techniques, whereas on the other hand security researchers devote considerable time into manually tracking down software vulnerabilities in program code.

Machine learning can help in establishing a link between these contrasting workflows. Instead of blindly scanning for possible vulnerabilities, the search may be actively guided by learning methods that incorporate knowledge about problematic programming constructs and known vulnerabilities of similar software. For example, known flaws in a web browser may be used to discover similar though not identical

vulnerabilities in the code of another browser. Similarly, an expert may actively control the search of a learning-based auditing tool, once positive or negative results are reported. A schematic depiction of this concept is illustrated in Figure 4.

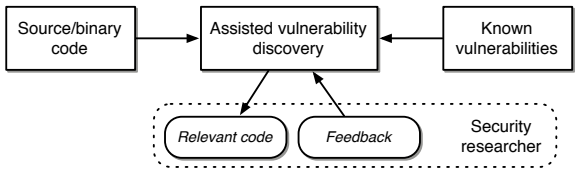


Figure 4: Schematic depiction of vulnerability search.

IV. Conclusions

In conclusion, we can note that computer security and machine learning are far from being "worst enemies". Instead, there is good hope to make them "best friends" in the near future. To this end, the proposed directions and perspectives for linking the two fields are currently explored by a mixed group of security and learning researchers at Technische Universität Berlin (and soon also at the University of Göttingen).

While this paper can not generally rule out the difficulties of applying machine learning in the field of security, it pinpoints the relevant challenges and advantages of linking the two and aims at fostering interesting security research to keep abreast of future attack developments.

Acknowledgments

The author acknowledges funding from the *Bundesministerium für Bildung und Forschung* under the project PROSEC (FKZ 01BY1145).

References

[1] J. Franklin, V. Paxson, A. Perrig, and S. Savage, "An Inquiry Into the Nature and Causes of the Wealth of Internet Miscreants," in *Proc. of Conference on Computer and Communications Security (CCS)*, 2007, pp. 375–388.

[2] "Symantec Global Internet Security Threat Report: Trends for 2009," Vol. XIV, Symantec, Inc., 2010.

[3] Y. Song, A. Keromytis, and S. Stolfo, "Spectrogram: A Mixture-of-Markov-Chains Model for Anomaly Detection in Web Traffic," in *Proc. of Network and Distributed System Security Symposium (NDSS)*, 2009.

[4] W. Robertson, F. Maggi, C. Kruegel, and G. Vigna, "Effective anomaly detection with scarce training data," in *Proc. of Network and Distributed System Security Symposium (NDSS)*, 2010.

[5] M. Cova, C. Kruegel, and G. Vigna, "Detection and analysis of drive-by-download attacks and malicious JavaScript code," in *Proc. of the International World Wide Web Conference (WWW)*, 2010.

[6] K. Rieck, T. Krueger, and A. Dewald, "Cujo: Efficient detection and prevention of drive-by-download attacks," in *Proc. of 26th Annual Computer Security Applications Conference (ACSAC)*, 2010.

[7] C. Gates and C. Taylor, "Challenging the anomaly detection paradigm: A provocative discussion," in *Proc. of New Security Paradigms Workshop (NSPW)*, 2006, pp. 21–29.

[8] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. of IEEE Symposium on Security and Privacy*, 2010, pp. 305–316.

[9] U. Bayer, P. Comparetti, C. Hlauschek, C. Kruegel, and E. Kirda, "Scalable, behavior-based malware clustering," in *Proc. of Network and Distributed System Security Symposium (NDSS)*, 2009.

[10] M. Bailey, J. Oberheide, J. Andersen, Z. M. Mao, F. Jahanian, and J. Nazario, "Automated classification and analysis of internet malware," in *Recent Adances in Intrusion Detection (RAID)*, 2007, pp. 178–197.

[11] K. Rieck and P. Laskov, "Visualization and explanation of payload-based anomaly detection," in *Proc. of European Conference on Computer Network Defense (EC2ND)*, November 2009.

[12] Y. Song, M. Locasto, A. Stavrou, A. D. Keromytis, and S. J. Stolfo, "On the infeasibility of modeling polymorphic shellcode: Re-thinking the role of learning in intrusion detection systems," *Machine Learning*, 2009.

[13] R. Perdisci, D. Dagon, W. Lee, P. Fogla, and M. Sharif, "Misleading worm signature generators using deliberate noise injection," in *Proc. of IEEE Symposium on Security and Privacy*, 2006, pp. 17–31.

[14] G. Cretu, A. Stavrou, M. Locasto, S. Stolfo, and A. Keromytis, "Casting out demons: Sanitizing training data for anomaly sensors," in *Proc. of IEEE Symposium on Security and Privacy*, 2008.

[15] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, "McPAD: A multiple classifier system for accurate payload-based anomaly detection," *Computer Networks*, vol. 5, no. 6, pp. 864–881, 2009.

[16] D. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. 13, pp. 222–232, 1987.

[17] J. Nazario, "A virtual client honeypot," in *Proc. of USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2009.

[18] K. Rieck, P. Trinius, C. Willems, and T. Holz, "Automatic analysis of malware behavior using machine learning," *Journal of Computer Security*, 2011, to appear.

Systems Security at VU University Amsterdam

Herbert Bos, Lorenzo Cavallaro, and Andrew S. Tanenbaum
Systems & Security Group
Department of Computer Science
Vrije Universiteit Amsterdam

***Abstract*—At VU University Amsterdam, research in system security occurs in two groups: the systems and network security group, and the secure and reliable systems group. Historically, the former group is rooted in attack detection, dynamic analysis, and reverse engineering of software, while the focus in the secure and reliable systems group is on studying and guaranteeing dependability properties of systems. However, the research in the two groups overlaps and the researchers work on joint activities and blend their knowledge and expertise to contribute to the security of networks and systems.**

I. Scope of research

With three full professors, four associate professors, and two assistant professors, the Computer Systems section at the Vrije Universiteit Amsterdam (VU University) is one of the largest in the Netherlands. Moreover, the last research assessment of Dutch universities in Computer Science over the period 2002-2008 (published in 2010), gave the Computer Systems section of the VU a top ranking (with maximum scores on all evaluation criteria).

The broad area of security at systems level features prominently in the section's research interests. At VU University, the area is covered by two separate, but closely collaborating groups:

1) a group headed by Herbert Bos that works on Systems and Network Security;
2) a group headed by Andrew S. Tanenbaum that works on Secure and Reliable Systems.

The focus on Tanenbaum's group leans towards reliability, but also covers aspects of security, as witnessed by security researchers like Lorenzo Cavallaro and Bruno Crispo that make up the group. The focus in Bos' group is mostly on system-level security both at the host and in the network. Both groups work primarily in low-level systems security. So, while the groups do much at the VM, OS and assembly level, there is less work on, say, Web security.

In the remainder of this document, we describe both research thrusts in more detail. Examples of research topics in the two groups are shown in Figure 1.

II. Research in the Systems and Network Security Group

The Systems and Network Security Group is very active in the area of attack detection, dynamic analysis, and reverse engineering.

Reverse engineering and security of legacy binaries: In terms of resources, the main research area in this group is that of reverse engineering of complex software. Funded by an ERC Starting Grant as well as several smaller grants, the group aims to tackle a fundamental problem in computer science: to revert low-level assembly to high-level source code. To do so, we use both static and dynamic analysis.

Most of the commercial software industry assumes that compilation (the translation of source code to binary code), is irreversible in practice for real applications. The research question for our group is whether this irreversibility assumption is reasonable. Specifically, we aim to demonstrate that the assumption is false.

The application domains for this research direction are two-fold. First, we want to be able to analyse what software is doing. For instance, if we buy a program, we would like to verify that it does what it is advertised to do (and not what we would not like it to do). Second, we would like to fix bugs in binary software. Specifically, we aim to protect legacy binaries from memory corruption attacks.

In addition, the reverse engineering techniques that we develop will be interesting for malware analysis. While malware may use obfuscation techniques, it is difficult (or at least expensive) to hide the use of certain data structures. In general, code obfuscation will be an important research area in our group. Specifically, we will work on detection, circumvention and improvement of code and data obfuscation techniques.

Dynamic analysis: The research team developed a variety of taint analysis solutions, of which the Argos full system emulator is probably best known. Argos is used by many organisations in many projects, mostly as a honeypot or malware analysis engine. In general, we work on a variety of high-interaction honeypot solutions—both for the client-side and for server applications. In other projects, taint tracking also features prominently. For instance, we work on solutions for attack detection, (decoupled) protection of mobile devices, intrusion recovery, that all depend on dynamic taint analysis. In addition, we have developed our own, very fast, dynamic binary translator with support for taint tracking. We expect to build on this

Attack detection and analysis: Systems-level attack detection and analysis permeates the research group. We already mentioned that we work on techniques to detect

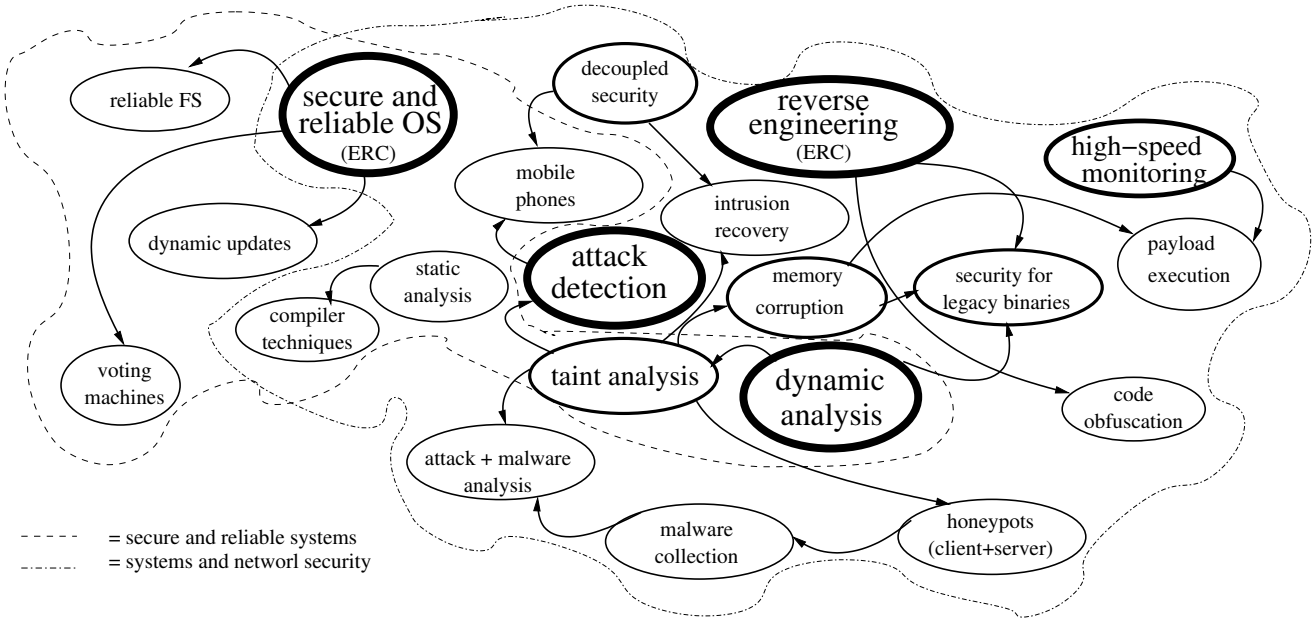


Figure 1. Examples of research topics in the system security groups at VU University

attacks by means of taint tracking. However, we also work on efficient solutions to detect attacks in the network. Part of our effort here is in building network monitoring tools, another part in abstract payload execution. Also, we collect, run and analyse large traces of malware. Finally, we conduct research in detecting attacks in constrained environments, such as mobile phones and other ultraportables.

III. Research in the Secure and Reliable Systems Group

The research interests of the Secure and Reliable Systems Group broadly fall in the area of software dependability. Supported by an ERC Advanced Grant, the research team currently places particular emphasis on making operating systems (OSes) more reliable and secure.

Monolithic operating systems (OSes) are in fact complex pieces of software that usually offer very little reliability and security guarantees. Faulty user-space applications can generally be restarted without affecting the existing concurrent communications but those involving the faulty processes. On the other hand, in a monolithic OS design, the kernel and all its components share a common address space and any component can potentially invoke arbitrary kernel functions. In this scenario, it becomes extremely complicated–if not impossible–to isolate and restart faulty kernel components as it is generally hard to define their boundaries and interactions (e.g., what kernel control paths are executed and how information are shared). Unfortunately, run-time bugs are not the only security threats an OS must deal with. For instance, malicious components may undermine the security of the whole system from its root: kernel rootkits can be installed on the system to replace or modify the legitimate behavior of arbitrary subsystems of the OS to fulfill criminals will.

To withstand such threats, the group is exploring approaches to combine a careful OS design with automated compiler-based instrumentation techniques. This spins off a number of interesting research directions. For instance, combining a modular operating system design with compiler-based techniques enables low-overhead runtime address space randomization (ASR) and fine-grained live update support for arbitrary OS components. This ultimately shows that is possible to build a polymorphic OS that constantly re-randomizes itself, while keeping the overhead to a minimum. Similarly, the same carefully planned OS design does not expose any recovery infrastructure to the OS programmer and drastically reduces the complexity of the problem space considered. This allows effective crash recovery using automatic instrumentation in a nonintrusive way, achieving transparent and component-agnostic recovery from crashes occurring anywhere during the execution of the component's task.

Automated compiler-based instrumentation techniques on their own enable a number of other interesting research directions. For instance, program analysis can be leveraged to identify interesting programs invariants that can be made available at runtime. Such properties can then be asynchronously and dynamically checked by idle cores, improving the dependability requirements of the whole system, while keeping the runtime overhead to a minimum. Moreover, compiler-based instrumentation enables to experiment with generic fine-grained program transforma-

tions, e.g., taint-tracking, obfuscation, and address space randomization, which improve the overall security of the transformed application.

IV. Overlap

As indicated by Figure 1, the two research groups collaborate closely on a variety of projects. The heads of the groups co-supervise Ph.D. students in the design of secure and reliable operating systems, while members of the groups also work together on several projects.

V. Research environment

The Computer Systems section at VU University is one of the largest Systems departments in the country with 3 full professors and a strong reputation in operating systems and security (Minix, CVS, Argos, and Amoeba all started here). The work on Minix sparked the development of Linux. In recent years, prof. Tanenbaum was awarded a prestigious Dutch Royal Society of Science (KNAW) Professorship, and an equally prestigious ERC Advanced Grant. Similarly, there was an ERC Starting Grant for prof. Herbert Bos, bringing the current total to two—more than any other CS department in the Netherlands. Two of the full professors are among the top 10 most cited computer scientists in The Netherlands (at nr. 1 and nr. 9, respectively), indicating the excellence of the research environment.

Many of the section's former Ph.D. students rank among the top researchers in the world. Examples include Frans Kaashoek (MIT), Robert van Renesse (Cornell), Leendert van Doorn (AMD), Sape Mullender (Bell Labs), and Werner Vogels (Amazon). One important quantitative measure of academic reputation is citation impact. The Report on Science and Technology Indicators issued by the Netherlands Observatory for Science and Technology (NOWT, 2008), shows that the CS Department at the VU ranks highest of all CS departments in the Netherlands in impact score. The high score is corroborated by other, independent studies. The last assessment of research of Dutch universities in Computer Science in over the period 2002-2008 (published in 2010), gave the Computer Systems section of the VU the top ranking (with a maximum score on all evaluation criteria).

VI. Outlook and future research directions

The research directions currently pursued by the system security groups are relatively stable. Both groups are partially funded by ERC grants that provide clout to the research efforts. The group of Secure and Reliable systems has a strong tradition of incorporating the fruits of the research in a working system, centered on Minix-3. Unlike most other operating systems the design of Minix-3 is centered first on reliability, and second on performance. Meanwhile, a lot of the effort in the Systems and Network Security group in the next five years will be invested in reverse engineering—with the goal of increasing the security of systems. At the same time, attack detection, prevention, and analysis remain very much on our radar too.

References

[1] Jorrit N. Herder Andrew S. Tanenbaum and Herbert Bos. Can We Make Operating Systems Reliable and Secure? *IEEE Computer, ISSN 0018-9162*, 39(4):44–51, May 2006.

[2] Herbert Bos, Willem de Bruijn, Mihai Cristea, Trung Nguyen, and Georgios Portokalidis. FFPF: Fairly Fast Packet Filters. In *Proceedings of OSDI'04*, San Francisco, CA, December 2004.

[3] Herbert Bos and Kaiming Huang. Towards software-based signature detection for intrusion prevention on the network card. In *Proceedings of Eighth International Symposium on Recent Advances in Intrusion Detection (RAID2005)*, Seattle, WA, September 2005.

[4] Herbert Bos and Bart Samwel. Safe kernel programming in the OKE. In *Proceedings of the Fifth IEEE Conference on Open Architectures and Network Programming (OPENARCH'02)*, pages 141–152, New York, USA, June 2002.

[5] Willem de Bruijn, Herbert Bos, and Henri Bal. Application-tailored I/O with Streamline. *ACM Transactions on Computer Systems (TOCS)*, 2011.

[6] Willem de Bruijn, Asia Slowinska, Kees van Reeuwijk, Tomas Hruby, Li Xu, and Herbert Bos. SafeCard: a Gigabit IPS on the network card. In *Proceedings of 9th International Symposium on Recent Advances in Intrusion Detection (RAID'06)*, Hamburg, Germany, September 2006.

[7] Cristiano Giuffrida, Lorenzo Cavallaro, and Andrew S. Tanenbaum. We Crashed, Now What? In *Proceedings of the 6th Workshop on Hot Topics in System Dependability (HotDep'10)*, Oct 2010.

[8] Lorenzo Martignoni, Aristide Fattori, Roberto Paleari, and Lorenzo Cavallaro. Live and Trustworthy Forensic Analysis of Commodity Production Systems. In *13th International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2010.

[9] Srijith Nair. *Remote Policy Enforcement Using Java Virtual Machine*. PhD thesis, VU University Amsterdam, 2010.

[10] N. Paul and A. S. Tanenbaum. Trustworthy Voting: From Machine to System. *IEEE Computer*, pages 23–29, May 2009.

[11] Georgios Portokalidis and Herbert Bos. Eudaemon: Involuntary and On-Demand Emulation Against Zero-Day Exploits. In *Proceedings of ACM SIGOPS EUROSYS'08*, pages 287–299, Glasgow, Scotland, UK, April 2008. ACM SIGOPS.

[12] Georgios Portokalidis, Philip Homburg, Kostas Anagnostakis, and Herbert Bos. Paranoid Android: Versatile Protection For Smartphones. In *Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC)*, Austin, Texas, December 2010.

[13] Georgios Portokalidis, Asia Slowinska, and Herbert Bos. Argos: an Emulator for Fingerprinting Zero-Day Attacks. In *Proc. ACM SIGOPS EUROSYS'2006*, Leuven, Belgium, April 2006.

[14] Christian Rossow, Christian J. Dietrich, Herbert Bos, Lorenzo Cavallaro, Marteen van Steen, Felix C. Freiling, and Norbert Pohlmann. Sandnet: Network Traffic Analysis of Malicious Software. In *1st Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*, April 2011.

[15] Asia Slowinska and Herbert Bos. The age of data: pinpointing guilty bytes in polymorphic buffer overflows on heap or stack. In *23rd Annual Computer Security Applications Conference (ACSAC'07)*, Miami, FLA, December 2007.

[16] Asia Slowinska and Herbert Bos. Pointless Tainting? Evaluating the Practicality of Pointer Tainting . In *Proceedings of ACM SIGOPS EUROSYS*, Nuremberg, Germany, March-April 2009.

[17] Asia Slowinska, Traian Stancescu, and Herbert Bos. Howard: a dynamic excavator for reverse engineering data structures. In *Proceedings of NDSS 2011*, San Diego, CA, 2011.

[18] Cristiano Giuffrida Stefano Ortolani and Bruno Crispo. Bait your Hook: a Novel Detection Technique for Keyloggers. In *Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection*, pages 200–217, 2010.

[19] Yves Younan, Pieter Philippaerts, Lorenzo Cavallaro, R. Sekar, Frank Piessens, and Wouter Joosen. PAriCheck: an Efficient Pointer Arithmetic Checker for C Programs. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 145–156, Beijing, China, 2010. ACM.

System Security Research at Birmingham: Current Status and Some Future Work

Marco Cova
University of Birmingham, UK
m.cova@cs.bham.ac.uk

***Abstract*—In this paper, we will describe the system security research at the University of Birmingham, UK, and briefly sketch directions of future work.**

I. Security Group at Birmingham

The "Formal Verification and Security" group is a well-established research group in the School of Computer Science of the University of Birmingham. Its original composition and strength is in the application of formal verification methods to systems and mechanisms of real-world complexity. Highlights of this line work include the formalization and verification of security properties of systems, such as the Trusted Platform Module (TPM) [13] and electronic voting systems [17]; the identification of flaws in security protocols, such as those used in electronic passports [8]; and the development or extension of tools to support and automate this kind of analyses [1].

The group has recently expanded to also cover system security (the author leads this line of work). In this area, our focus is on protecting computer systems from malicious activity. In this paper, we will summarize our system security work[1] and briefly overview our plans for future work.

II. Current and Recent Research

Much of our research has focused on web-based threats, which have become prevalent in the last few years. Here, we summarize our efforts to characterize these threats and to provide defenses for web applications and web clients.

A. Protecting web applications from attacks

Web applications have become the standard means to provide services to online users. Unfortunately, they have also frequently become the targets of compromises. In fact, since web applications are typically publicly available, they are exposed to attacks from anywhere in the world. Web applications also manage sensitive data, which can be easily monetized on the underground market. Finally, developers (and the programming languages and tools they use) often focus more on powerful features and usability rather than on security. While considerable research has been performed in the past to protect web applications from attacks, several challenges remain open. We tackled three significant open problems, namely, the modeling of sanitization routines, the detection of multi-step attacks, and the detection of application-specific errors.

In [4], we used a combination of static analysis and testing techniques to precisely detect taint-style vulnerabilities that are commonly found in web applications (e.g., SQL injection and cross-site scripting). More precisely, we extended a source code analysis tool to statically approximate the sets of string values that variables can hold at certain program points. We used this novel string analysis technique to conservatively identify incorrect input validation checks, that is, cases in which an untrusted input string has been checked with a sanitization routine, but it still violates a security policy (i.e., it may be used to perform a SQL injection attack). If a faulty sanitization is found, we automatically generate (and feed to the application) inputs that are likely to trigger the vulnerability. We use this additional analysis step to prune false positives generated from our static analysis techniques and to provide concrete inputs that demonstrate the presence of the vulnerability. Our approach provides a considerable improvement over previous techniques, which typically used predetermined, coarse models for input validation checks (e.g., they always considered the checks completely effective or completely ineffective).

In [5], we presented a tool that uses static analysis techniques to automatically detect "multi-step vulnerabilities," i.e., vulnerabilities that can be exposed only through multi-step attacks (e.g., stored SQL injection, stored cross-site scripting, and forceful browsing attacks). Such vulnerabilities have been traditionally difficult to discover via manual analysis or testing because they are "deep" in the application's logic, and they are typically missed by other static analysis tools that analyze the individual modules of an application in isolation (e.g., a file at a time) rather than the application as a whole. Our tool analyzes the source code of an application and builds a model of the application's state and of how it changes as a consequence of the requests that users can issue. We then use model checking techniques to identify sequences of user requests that could lead the application into an insecure state, for example, one in which user input that was stored in the application's database is used without proper sanitization in a SQL query (i.e., a stored SQL injection vulnerability scenario).

While previous research (including our own [4], [5]) focused on detecting violations of generic security policies

[1]Most of the work described here was performed when the author was with the Computer Security Lab at the University of California, Santa Barbara.

in web applications (e.g., "no user input should be used unsanitized in critical program operations"), little work existed for detecting attacks that violate application-specific policies (e.g., "no unauthenticated user should access the private section of this site"). In [9], we proposed an approach to automatically detect violations of a class of such application-specific policies. More precisely, we focused on workflow policies, which specify how users are supposed to navigate an application. Our approach was based on the use of anomaly-based and likely invariant detection techniques to learn "normal" and invariant values for the application's state. Then, during the detection phase, we monitor the application's execution to identify requests that bring the application in a state that deviates from the established models of normality or that violates the learned invariants. By using these techniques, we were able to detect authentication and authorization bypasses, parameter tampering, and code inclusion attacks in real-world applications.

B. Protecting clients from malicious web applications

Besides being the target of attacks, web applications have recently become the vector through which client machines are compromised. This situation is the result of the emergence of a new form of malware, *malicious web code*, which is written in dynamic languages, such as JavaScript and ActionScript, rather than binary code, and targets web clients, as opposed to infrastructure or server-side components. In our research, we tackled two prevalent types of malicious web code: web pages that perform drive-by-download attacks and malicious Flash advertisements.

In a drive-by-download attack, a victim is lured to a malicious web page. The page contains code that exploits vulnerabilities in the user's browser or in the browser's plugins. If successful, the exploit downloads malware on the victim machine, which, as a consequence, often becomes a member of a botnet. Existing techniques for the detection and analysis of drive-by attacks include signature matching and the execution of malicious samples in honeyclients. Unfortunately, these approaches are less than ideal. In fact, signatures need to be updated whenever an unknown sample is found and can be easily evaded by simple modifications to the malicious code. Furthermore, honeyclients can be difficult to configure correctly and offer little insight into how an attack works. To detect web pages that contain drive-by-download attacks, we developed a tool, called WEPAWET, that interprets HTML pages and JavaScript code in an instrumented browser, and records events that occur during their execution (e.g., the instantiation of an ActiveX control or the fetching of external resources). For each event, the tool extracts one or more features, whose values are assessed using anomaly detection techniques in order to identify malicious content. We validated WEPAWET on an extensive dataset of web pages and we found it to be effective in detecting known and novel attacks while generating few false positives [11]. Related work has focused on detecting different forms of these threats (for example, malicious Flash-based advertisements [15]), and on scaling up the analysis capability of this system [7].

We have made WEPAWET publicly available as an online service (http://wepawet.cs.ucsb.edu) where users can submit URLs or files for analysis. For each sample, a report is generated that shows the classification of the page and the results of the deobfuscation, exploit classification, and other analyses. The reports generated by our tools are routinely used to investigate incidents and new exploits, as supportive evidence in take-down requests, and to complement existing blacklists.

C. Measuring the malicious web

A third thrust of our research has been directed at better characterizing web-based threats. While these are, by their very nature, moving targets (attackers often change their techniques, strategies, and "business plans"), we focused on understanding fundamental characteristics of web attacks by measuring specific threats, learning about the tools used in attacks, understanding the attackers' *modus operandi*, and uncovering the structure of the associated criminal activities. We believe these are necessary steps to assess the effectiveness of current mitigation techniques and to develop novel and better approaches to combat these threats.

Botnets: A botnet is a network of malware-infected machines that are controlled by a single entity (often called the "botmaster"). Botnets have become the root cause of a large number of security problems on the Internet, such as denial of service attacks, spam distribution, and stealing of sensitive data. In [18], we described our experience in sinkholing, for ten days, the Command and Control (C&C) servers of the botnet created by the Torpig malware. We found that this botnet comprised over 180,000 machines that were compromised through drive-by-download attacks. We leveraged our sinkholing to present a comprehensive study of this large, successful, and long-lasting botnet. In particular, we provided an accurate measurement of the botnet size, estimated the value of the financial information exfiltrated from infected machines, and discussed the threats to the victims' privacy. We also presented our experience in cooperating with affected ISPs, financial institutions, and law enforcement agencies to mitigate the damage suffered by the victims of the botnet.

Phishing: Phishing is a form of identity theft in which an attacker (a "phisher") attempts to elicit confidential information from unsuspecting victims by luring them to web sites that look like a trusted web site, e.g., a well-known online banking or government site. Phishers often rely on *phishing kits*, packages that contain complete phishing web sites in an easy-to-deploy format. In [10], we performed a measurement study of the kits used by phishers: we dissected their structure and functionality, and analyzed the habits of

their users. In particular, we found that phishing kits often contain hidden backdoors that send the stolen information to third parties.

Rogue security software: A rogue anti-virus program ("rogue AV" in short) is a type of misleading application that pretends to be legitimate security software, such as an anti-virus scanner, but which actually provides the user with little or no protection, and, in some cases, facilitates the installation of the very malicious code that it claims to detect and eradicate. In [12], we performed a large-scale study of web sites involved in the distribution of rogue AV programs. We first identified thousands of such sites and then proceeded to identify any emerging patterns in the way these servers are created, managed, and inter-connected with each other. In particular, for each site, we extracted a number of "network observables" (such as IP addresses, registration data, and domain name structure), and used them as input to a multi-criteria clustering algorithm to determine groupings of server components with similar characteristics. Manual validation confirmed with strong confidence that the derived clusters actually represented distinct campaigns, i.e., coordinated efforts of (likely) distinct criminal groups distributing rogue AV programs.

D. Additional research areas

Our interests include other system areas, such as combating malicious programs and hardening critical systems.

Malware analysis: Malware generically refers to code that fulfills the (malicious) intent of an attacker, such as viruses, worms, and spyware. The state-of-the-art approach for the analysis of malware consists of running malicious programs in an analysis environment and monitoring their behavior to uncover, for example, the specific type of malicious activity they perform (e.g., keylogging, phishing, spamming) or the external sites they contact.

A current challenge in malware analysis is detecting "split-personality" malware, i.e., malicious programs that, when run in an emulated or virtualized analysis environment, behave differently than on a real system. In [6], we presented a novel approach, based on record-and-replay techniques, to efficiently and robustly detecting such malware.

Electronic voting security: The security of electronic voting systems has gained increasing interest in recent years. In particular, in response to concerns about their reliability and integrity, the State of California sponsored a top-to-bottom review of the electronic voting systems certified for use in the state. A similar evaluation was later commissioned by the State of Ohio. These studies were unprecedented for the level of access to the software, hardware, and documentation that they granted. We participated in both studies as a Red Team (penetration testers). We documented our experience, the testing methodologies we developed, and our findings in [2], [3]. We discovered a number of serious design and implementation flaws that could be leveraged to completely compromise the integrity of the election, and implemented realistic attacks. As a consequence, both California and Ohio enforced strict limitations on the use of electronic voting systems in their states and proposed significant changes to their election procedures.

III. Future Research

In the last few years, security threats have expanded in volume (e.g., breach reports are nowadays daily events), scope (e.g., targeting critical infrastructure systems), and sophistication (e.g., state-level attacks). This situation highlights the need for more effective system security research and for its transition to real-world tools and approaches. This will require an effort in several directions: focusing on techniques that prevent attacks from being successful in the first place, rather than detecting them ex-post; facilitating the collaboration between groups (both in academia, industry, and operational community) with complementary skill sets, approaches, and access to datasets; and incorporating lessons learned in other disciplines (e.g., economics and psychology) to design more effective security mechanisms. Here we describe more concretely some our future plans.

Malicious Web: We believe malicious activity on the Web will continue to be a major source of concern in the next years. We intend to tackle it in several ways.

An important first step is understanding the limitations (and strengths) of existing approaches and tools. As a preliminary example of this activity, we have explored the effectiveness of black-box testing tools for web applications [14] and the robustness to evasion attempts of high-interaction honeyclients [16]. This kind of "offensive research" is invaluable in indicating problem areas that need additional work.

Second, we intend to improve existing tools and approaches for *detecting* malicious web-based activity. For example, we intend to extend the classes of malicious content that we can reliably detect, improve the techniques that we use to identify potentially malicious content (e.g., locating particularly bad neighborhoods on the Web), and make the detection techniques more effective against evasion attempts. We envision that this will require sophisticated static and dynamic analysis techniques, such as program analysis and multi-path exploration techniques for JavaScript.

Third, we need more sophisticated techniques to *analyze* a threat after it has been detected. For example, such techniques will allow us to identify patterns in the actual attack code and traffic (e.g., to identify the individual exploit kits used by attackers), to characterize the way malicious activity is organized and monetized (e.g., to identify and understand the hosting infrastructure used in an attack), and to understand the techniques of the individuals responsible for it (e.g., their habits). These capabilities are necessary to enable proactive defenses (e.g., predicting future attacks, attributing attacks) as opposed to reactive ones.

We will also need better *prevention* techniques. For example, we envision programming languages and development frameworks that guarantee the absence of certain vulnerabilities in web applications. Similarly, we intend to work on a formalization of security-relevant properties of web clients that can help to identifying possible security issues early on and to informing the design of future web technologies. While approaches have been proposed in this direction, it is critical that these solutions be simple to use and practical.

Malware: Malware is a prevalent and, seemingly, inevitable presence in today's computing environment. In current systems, a malware infection completely compromises the security guarantees of the system. Unfortunately, malware-infected systems are commonly used to perform sensitive operations, e.g., accessing online banking accounts. We plan to investigate techniques that enable systems to maintain some of their security properties even in the presence of malware infection. For example, secure data flow and data encapsulation could be used to ensure that online banking credentials are only sent to the banking web site.

Threats to privacy: Web-based attacks are not the only issue on the Web. Privacy, and more generally, access rights to user's data on the Web are also a major concern, given the explosive growth of social networking sites, online advertisements, and cloud computing. We are interested in developing solutions that protect the interests of users while maintaining the usability of today's Web.

REFERENCES

[1] M. Arapinis, E. Ritter, and M. Ryan. StatVerif: Verification of Stateful Processes. In *Proceedings of the IEEE Computer Security Foundations Symposium (CSF)*, June 2011.

[2] D. Balzarotti, G. Banks, M. Cova, V. Felmetsger, R. Kemmerer, W. Robertson, F. Valeur, and G. Vigna. Are Your Votes *Really* Counted? Testing the Security of Real-world Electronic Voting Systems. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA)*, pages 237–248, July 2008.

[3] D. Balzarotti, G. Banks, M. Cova, V. Felmetsger, R. Kemmerer, W. Robertson, F. Valeur, and G. Vigna. An Experience in Testing the Security of Real-World Electronic Voting Systems. *IEEE Transactions on Software Engineering*, 36(4), 2010.

[4] D. Balzarotti, M. Cova, V. Felmetsger, N. Jovanovic, E. Kirda, C. Kruegel, and G. Vigna. Saner: Composing Static and Dynamic Analysis to Validate Sanitization in Web Applications. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 387–401, Oakland, CA, USA, May 2008.

[5] D. Balzarotti, M. Cova, V. Felmetsger, and G. Vigna. Multi-Module Vulnerability Analysis of Web-based Applications. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 25–35, Alexandria, VA, USA, Oct. 2007.

[6] D. Balzarotti, M. Cova, C. Karlberger, E. Kirda, C. Kruegel, and G. Vigna. Efficient Detection of Split Personalities in Malware. In *Proceedings of the Symposium on Network and Distributed System Security (NDSS)*, Feb. 2010.

[7] D. Canali, M. Cova, C. Kruegel, and G. Vigna. Prophiler: a Fast Filter for the Large-Scale Detection of Malicious Web Pages. In *Proceedings of the International World Wide Web Conference (WWW)*, Mar. 2011.

[8] T. Chothia and V. Smirnov. A Traceability Attack against e-Passports. In *Proceedings of the Financial Cryptography Conference*, pages 20–34, Jan. 2010.

[9] M. Cova, D. Balzarotti, V. Felmetsger, and G. Vigna. Swaddler: An Approach for the Anomaly-based Detection of State Violations in Web Applications. In *Proceedings of the Symposium on Recent Advances in Intrusion Detection (RAID)*, pages 63–86, Gold Coast, Queensland, Australia, Sept. 2007.

[10] M. Cova, C. Kruegel, and G. Vigna. There Is No Free Phish: An Analysis of "Free" and Live Phishing Kits. In *Proceedings of the USENIX Workshop on Offensive Technologies (WOOT)*, San Jose, CA, USA, July 2008.

[11] M. Cova, C. Kruegel, and G. Vigna. Detection and Analysis of Drive-by-Download Attacks and Malicious JavaScript Code. In *Proceedings of the International World Wide Web Conference (WWW)*, Apr. 2010.

[12] M. Cova, C. Leita, O. Thonnard, A. Keromytis, and M. Dacier. An Analysis of Rogue AV Campaigns. In *Proceedings of the Symposium on Recent Advances in Intrusion Detection (RAID)*, Sept. 2010.

[13] S. Delaune, S. Kremer, M. D. Ryan, and G. Steel. A formal analysis of authentication in the TPM. In *Proceedings of International Workshop on Formal Aspects in Security and Trust (FAST)*, Sept. 2010.

[14] A. Doupé, M. Cova, and G. Vigna. Why Johnny Can't Pentest: An Analysis of Black-box Web Vulnerability Scanners. In *Proceedings of the Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, July 2010.

[15] S. Ford, M. Cova, C. Kruegel, and G. Vigna. Analyzing and Detecting Malicious Flash Advertisements. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, Dec. 2009.

[16] A. Kapravelos, M. Cova, C. Kruegel, and G. Vigna. Escape from monkey island: Evading high-interaction honeyclients. In *Proceedings of the Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, July 2011.

[17] S. Kremer, M. D. Ryan, and B. Smyth. Election verifiability in electronic voting protocols. In *Proceedings of the fifteenth European Symposium on Research in Computer Security (ESORICS)*, Sept. 2010.

[18] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna. Your Botnet is My Botnet: Analysis of a Botnet Takeover. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 635–647, Nov. 2009.

The SPARCHS Project

Hardware Support for Software Security

Simha Sethumadhavan, Salvatore J. Stolfo,
Angelos Keromytis, Junfeng Yang
Department of Computer Science
Columbia University
New York, NY
(simha,sal,angelos,junfeng)@cs.columbia.edu

David August
Department of Computer Science
Princeton University
Princeton, NJ
august@cs.princeton.edu

I. Problem Statement

Current security research is largely top-down, where the most exposed layers --- the network/ application layers --- are first secured, and the lower layers are secured as and when threats appear. Security, thus, has become an arms race to bottom. For every software mitigation strategy today, vulnerabilities in the software layer below it can be used to attack and weaken the mitigation strategy. There are many examples of such attacks in the literature including those that attack anti-virus, libraries, operating systems, hypervisors, and BIOS routines.

A solution to the above problem is to push the security mechanisms down to hardware, which is typically immutable. Growing on-chip transistor budgets provide the opportunity to explore this possibility. In addition to offering immutable security, there are two further advantages to implementing security mechanisms in hardware. First, hardware supported security mechanisms can be much more energy-efficient compared to software only mechanisms. Given that energy- and power-efficiency significantly influence computing today, hardware support could very well be necessary for security mechanisms to gain traction in many real world settings. Second, implementing security mechanisms can provide unmatched visibility into execution. This provides an opportunity for new security techniques.

The SPARCHS project is considering a new computer systems design methodology that considers security as a first-order design requirement at all levels, starting from hardware, in addition to the usual design requirements such as programmability, usability, speed, and power/energy-efficiency. The rest of the paper describes the proposed hardware security mechanisms and the current status of the project.

II. Research Directions

Directly implementing security mechanisms in hardware poses a significant problem. First, since hardware is finite not all known security mechanisms can be implemented in hardware. Second, hardware is less flexible than software, so it cannot be easily updated when new attacks are discovered[1]. Thus, ideally, hardware mechanisms should also be able to cover attacks that are not yet discovered. This begs the question what hardware mechanisms can cover a wide variety of known and unknown attacks?

Instead of trying to discover unknown attacks, which is hard, and develop specific defenses, which is also hard, our strategy is to mimic the defenses from the biological world where a fantastic number of defenses have evolved over many thousands of years to survive constantly attacking predators. Our goal is to find counterparts to successful biological protection mechanisms and implement them in hardware.

The benefits of bio-inspired approaches to security have been mentioned as a promising direction in several reports[1,2]. The idea of applying bio inspired security principles to the hardware level is the key novel contribution of the SPARCHS project. Next we touch upon some biological protection mechanisms and then describe their hardware/software formulations that can mimic these mechanisms.

A. Biological defenses

Defensive strategy is pervasive at all levels in the animal and plant kingdom where existence is constantly threatened due to predators and environmental vagaries. At the molecular level, our genetic code is suspected to contain

[1] FPGAs offer an opportunity to update software in the field but currently have limited utility in general purpose computing.

high-level of redundancy, at the cellular level lymphocytes offer innate protection against viruses and microbes, at the organ level, redundancy (e.g., two kidneys) and regeneration (e.g., skin cuts, lizards dropping tails under attacks) allow continuous function and recovery under attack, and organisms have amazing ability to learn from past attacks (e.g., vaccination.) In many cases, multiple organisms co-operate (e.g., microbiomes) from symbiotic relationship to provide immunity over and above innate and adaptive immunity. Innate immunity mechanisms is typically a first, generic response to attacks from foreign organisms. Typical functions of innate immunity include capturing cellular debris, foreign particles and invading microorganisms. The adaptive immune response provides facilities to recognize and remember specific attack vectors, and provide stronger protection as more attacks are encountered in future.

In the biological world, the attackers have also evolved many sophisticated techniques to thwart existing defenses. The most notorious of the attackers attack the immune system itself (e.g., HIV) and is difficult to destroy because it constantly changes its tertiary structure (polymorphism), which guarantees the virus a safe harbor in the host. To provide these amazing security features organisms spend nearly 30% of their energy in defense. Given the success of flora and fauna, the defensive strategies used in biological systems are certainly worth emulating.

To summarize, the biological techniques, we aim to provide hardware support to mimic the following biological primitives: (1) Innate Immunity for detection and isolation, (2) Diversity and polymorphism for prevention, (3) Symbiotic Immunity for implementing protection and detection techniques, (4) Adaptive Immunity for prevention, (5) Optimized redundant execution for continued execution, (6) Autotomy to contain damage when all else fails.

B. Hardware Analogues

Innate Immunity One basic function of innate immunity is to identify and contain foreign particles. Translated to the computer systems, this translates to ensuring untrusted data does not reach confidential code, and trusted data is not sent to untrusted code. Information flow tracking (IFT) essentially provides the above functionality. While IFT is no means a new technique, it has been difficult to implement correctly without hardware support. Verifying the data flow of program is insufficient to verify that no illegal information flows occur in the program. Current solution is to allow implicit flows by converting all data dependences to control dependences. In the SPARCHS project we are considering how the implicit flows, or the flow of information through control dependences, must be determined with static analysis as information can flow through segments of code that do not execute. This information will be conveyed to the hardware to track these flows.

Diversity and Polymorphism The key idea in digital defensive polymorphism is to change the execution of a program dynamically to thwart attackers. One of the simplest ways to change execution is to change the hardware each time a program executes. We call this type of shape-shifting hardware as polymorphic hardware. Polymorphic hardware succeeds against attackers by purposely injecting randomness into program execution. This method could have three very useful impacts for security at different levels of execution:

1) At the hardware level this architecture would make would make side-channel attacks very difficult, because it is always harder to attack a moving or unpredictable target. Conceptually each execution of a program happens on a different hardware, and with this type of uncertainty the attacker cannot reliably interpret of the side-channel data. This resilience to side-channels is leveraged by symbiotes to avoid detection.

2) Polymorphism can provide resilience against semantic attacks: Consider a code-injection attack. The attacker takes advantage of knowledge of the programs ABI and the instruction's semantics in the program to carry out the attack. With polymorphism at the instruction-set level — Instruction-Set Randomization (ISR) — this attack can be thwarted because the attacker can no longer know the semantics of each instruction.

3) Polymorphism can provide resilience against program logic bugs. While most security attacks to date exploit bugs in serial programs, more parallel programs are being produced because of adoption of multicore programs. It is well known that reliable parallel programming is harder than sequential programming, and it is likely that attackers will take advantage of concurrency bugs in the near future.

The polymorphic architecture can decrease the chance of security attacks on emerging parallel programs by reducing the chance of race conditions because of the diversified, random execution substrate. Additionally, polymorphism may have a side benefit of improving program performance by reducing unintentional contention on shared resources, and also enable better testing of programs through automatic fuzzing of program execution. The SPARCHS project is investigating how these shape-shifting features can be implemented in the simplest way into existing processors without undue performance impact.

Symbiotic Immunity The idea of how symbiotes can be adapted to computer systems was first proposed in the Minestrone project at Columbia. At a high-level, the

symbiote is a small program that is embedded in a host program. The symbiote can reside within any arbitrary body of software, regardless of its place within the system stack. While symbiotes share some commonalities to reference monitors in terms of benefits they offer, a key difference is that symbiote cannot survive without the host program and the host program cannot survive without the symbiote. This interdependency is not required for reference monitors. The SPARCHS project aims to provide hardware support that will allow symbiotes to have this property.

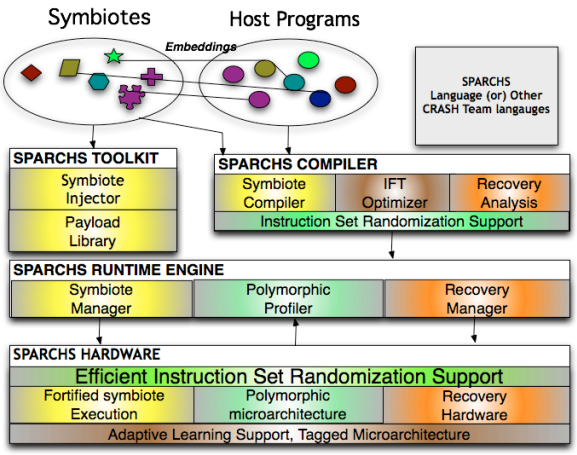
Symbiotes can be supported in hardware through three distinct ways that have different easy of implementation vs. benefits trade-offs. First, one or few cores in a multicore processor may be hidden from all system software by modifying the BIOS, and having the symbiotes run on a hidden core. Since system software cannot see the disabled core, the symbiotes functions cannot be monitored. This solution assumes that the hidden core has access to all of the on-chip memory, which can be easily architected. The second solution is not to disable the cores (thus not reduce throughput) but use the shape-shifting polymorphic architecture such that no side-channels are possible. Finally, the most efficient option is to build a special hardware unit that guarantees physical and execution isolation for the symbiotes.

Adaptive Immunity Adaptive immunity requires methods to learn about normal and abnormal program behaviors. We are developing better hardware support to identify anomalous execution points by using fine grained measurements from on-chip performance counters. We are also working on newer performance counter architectures as on-chip performance monitors today are tuned for collecting information on the common case; in security (and software engineering) we are interested more in the uncommon case. Other methods for program characterization include learning about control and data flow execution graphs. The SPARCHS project is investigating hardware primitives that can be securely used to learn about program behavior.

Optimized Redundant Execution Mimicking biological redundancy in digital systems is fairly obvious. N-versioning is already a very common approach but it is also terribly impractical approach for many modern execution environments such as mobile and server environments. We are planning to make N-versioning better with compiler and hardware optimizations. We plan to use values produced from one redundant copy in another to improve the speed of the diversified replica or use special purpose microarchitectures to quickly communicate values between the N-versions.

Autotomy/Apotasis SPARCHS will bring autotomy to computing systems by detecting attacks and faults in a software subcomponent and responding to it by removing the component from the critical system. The goal is to heal the programs, and make them available as much as possible. SPARCHS Autotomy will employ Rescue Points which are locations in the application code in which error handling is performed with respect to a given set of foreseen. Rescue points basically create a mapping between the set of errors that could occur during a program's execution and the limited set of errors that have been explicitly handled in the program code. Thus, a failure that would cause the program to crash is translated into a return with an error.

C. Integrated System



While the project's main focus is on discovering efficient hardware primitives for security another important goal is to demonstrate how the proposed primitives can be used in software. Towards this goal we are developing hardware and software for the SPARCHS system. Figure 1 illustrates the different facets of the SPARCHS system and how it interfaces with existing and proposed research techniques. At the hardware level, SPARCHS includes support for microarchitecture-level polymorphism, support for memory versioning and checkpointing to support roll back and recovery, and specially fortified hardware to support symbiote execution. These techniques are orthogonal to and can be integrated with information flow tracking and strong instruction set randomization. SPARCHS guarantees that an outside attacker cannot simply turn off the protection mechanism. SPARCHS includes a simple management layer that provides safe storage of keys and feeds program profile information to the SPARCHS compiler. The management layer can also provide simple recovery services.

SPARCHS is full-system effort and includes many software aspects. The SPARCHS compilation suite serves three main

purposes: first, it combines the application, the symbiote, and the symbiote policy into a single binary, and applies instruction-set randomization to the binary; second, it provides static analysis techniques for managing recovery and repair. Finally, it provides analysis to enforce correct dynamic information flow in hardware. The SPARCHS environment includes toolkits for mining static, dynamic and program information to help programmers specify policies, symbiote payload libraries, and also standalone injection of symbiotes in binaries if necessary.

III. CURRENT STATUS

We have been working on this project for three quarters now. We have made significant progress on several fronts. First, we have created software symbiote infrastructures to understand how they should be protected in hardware[3]. We are planning to create symbiotes in x86 and ARM to demonstrate feasibility in a wide variety of architectures. The software port is likely to be completed in the next quarter and detailed hardware analysis is going to begin in the following years. We have made significant progress on hardware support for learning/adaptivity[4]. Existing methods for accessing performance counters on x86 machines seem terribly out-of-date. In fact, popular tools like Vtune, PAPI and Oprofile use heavyweight kernel calls which perturb hardware measurements. We have created new tools that will allow to precisely read the performance counters and have about 70x lower overhead compared to PAPI. The tool is available for download from: http://castl.cs.columbia.edu/limit. This tool is currently being used to learn normative execution characteristics of programs. As a stepping stone to ISR, we have developed a full-system ISR mechanism[5], not just covering single program program binaries but including support for DLLs, shared libraries, key management etc. Hardware modifications and full system prototypes are underway. To test our systems we are working on creating new attacks (concurrency based[6]) and also demonstration of defense mechanisms against such attacks. There is much exciting work to be done in this area. One major open question is what further primitives can be added to hardware. Our experience with SPARCHS could help answer this question.

[1] A. Somayaji, S.Hofmeyr, S. Forrest, "Principles of a Computer Immune System," NSPW '97 Proceedings of the 1997 workshop on New security Paradigms, 1997.

[2] S. E. Goodman, H. S. Lin, "Toward a Safer and More Secure CyberSpace," Committee on Improving Cybersecurity Research in the United States, The National Academies Press, Washington D.C., 2007

[3] A. Cui, S.J. Stolfo, "Defending Legacy Embedded Systems with Software Symbiotes," Proceedings of the 14th International Symposium on Recent Advances in Intrusion Detection. RAID 2011.

[4] J. Demme, S. Sethumadhavan, "Rapid Characterization of Architectural Bottlenecks via Precise Event Counting," Proceedings of the 39th ACM/IEEE International Symposium on Computer Architecture, June 2011.

[5] G. Portokalidis, A.D. Keromytis: Fast and practical instruction-set randomization for commodity systems. ACSAC 2010: 41-48

[6] J. Yang, A. Cui, J. Gallagher, S. Stolfo, S.Sethumadhavan, "Concurrency Attacks,", Department of Computer Science Technical Report, CUCS-028-11, 2011.

Malicious Website Detection: Effectiveness and Efficiency Issues

Birhanu Eshete, Adolfo Villafiorita, Komminist Weldemariam
Center for Information Technology (FBK-IRST)
Fondazione Bruno Kessler
via Sommarive 14, 38123 Trento, Italy
Email: (eshete,adolfo, sisai)@fbk.eu

***Abstract*—Malicious websites, when visited by an unsuspecting victim infect her machine to steal invaluable information, redirect her to malicious targets or compromise her system to mount future attacks. While the existing approaches have promising prospects in detecting malicious websites, there are still open issues in effectively and efficiently addressing: filtering of web pages from the wild, coverage of wide range of malicious characteristics to capture the big picture, continuous evolution of web page features, systematic combination of features, semantic implications of feature values on characterizing web pages, ease and cost of flexibility and scalability of analysis and detection techniques with respect to inevitable changes to the threat landscape. In this position paper, we highlight our ongoing efforts towards effective and efficient analysis and detection of malicious websites with a particular emphasis on broader feature space and attack-payloads, flexibility of techniques with changes in malicious characteristics and web pages and above all real-life usability of techniques in defending users against malicious websites.**

***Keywords*-Malicious Websites, Detection, Efficiency, Effectiveness**

I. Introduction

Attackers lure an unsuspecting victim to visit malicious websites and they steal important credentials from the victim or install malware on the victim's machine to use it as a springboard for future exploits [1], [2], [3], [4]. When the victim visits a malicious website, the attack is initiated and up on finding evidences of exploitable vulnerabilities (e.g., of browser components [5], of browser extensions [6]), the attack payload is executed.

To defend Web users against malicious websites, several automated analysis and detection techniques and have been proposed. However, given the alarming prevalence of malicious websites and the ever-changing techniques in crafting attack payloads combined with emerging threats, current approaches to tackle the problem have common and specific limitations in effectively and efficiently: characterizing the malicious payloads using a more complete feature set; incorporating inevitable evolution of web page features; systematic methods of selecting and composing web page features; ensuring the flexibility and scalability of feature extraction, model construction and model training. In this position paper, we highlight critical issues in this regard and propose a research roadmap in improving effectiveness and efficiency in automated analysis and detection of malicious websites.

II. Malicious Websites: Scope of the Problem

To combat the impacts of malicious websites, approaches proposed fall into two complementary categories: static and dynamic analysis. The former rely mainly on the source code and some static features such as URL structure, host-based information, and web page content to perform analysis and construct characterizations of malicious payload. The latter focus on capturing "behaviors" manifested when the page is rendered in a controlled environment. A strategy common to both approaches is that they extract features of some type for further analysis to get patterns of malicious payloads, based on which a classification algorithm is trained using machine learning techniques.

A widely used protection technique is based on blacklisting of known malicious URLs and IP addresses collected via manual reporting, honeypots, and custom analysis techniques. While lightweight to deploy and easy to use, blacklisting is effective only if one can exhaustively identify malicious websites and timely update the blacklist. In practice, doing so is infeasible due to: fresh websites are too new to be blacklisted even if they are malicious, some websites could escape from blacklisting due to incorrect analysis (e.g., due to "cloaking"), and the attackers may frequently change where the malicious websites are hosted. In effect, the URLs and IP addresses may also change accordingly [1], [3].

Lexical aspects in the URL (e.g. URL length, domain name length, query length, path length) and host-based information (e.g. WHOIS information, DNS records) have been demonstrated to be successful in economically characterizing malicious web pages in [2] and [7] and partly in [1]. The major assumption in such approaches is the tendency of URL tokens and host-based values of malicious URLs to differ from that of the benign ones. The strength of such approaches is the speed of feature extraction without executing the URL. However, if we consider the WHOIS information of websites registered recently, by registrars with low reputation, such websites are likely to be classified as malicious due to low reputation scores. In effect, there is a high risk of false positives. Conversely, false negatives may arise–as old and well-reputed registrars may host malicious

websites. Another potential source of false negatives could be when there are websites that use free hosting services, already compromised sites with benign-looking URL and host information. Above all, the feature space considered is way limited to capture the most devastating attacks such as injection of malicious code, drive-by downloads, and social engineering tricks.

Web page content extraction and the use of machine learning techniques for classification of web pages is another proposal relying on features such as text content, HTML, native JavaScript functions and objects, ActiveX objects, and iFrame size [8], [9]. Such approaches are also able to quickly extract content features to learn classifiers for web pages and are particularly effective in detecting spams and phishing tricks. In [1], a fast pre-filtering technique combining URL structure, host-based information and page content is proposed and is demonstrated to significantly reduce the execution load of a dynamic analysis technique. The general limitation of considering only page content is the high risk of obfuscated content (e.g., multilevel obfuscation of JavaScript code) and overlooking web pages that necessarily need execution to launch attacks (e.g., malicious JavaScript that exploits vulnerabilities of browser plug-ins).

Execution monitoring approaches have also been shown to be effective in analysis and detection of malicious websites [3], [4], [10], [11], [12], [13], [14], [15]. Among such techniques, honey-clients are systems that mimic a human visitor using a dedicated sandbox environment (e.g. virtual machine) to visit a web page. When a page is being rendered, the execution dynamics is captured and analyzed to infer evidences for attack payloads. Honey-clients are of three types, namely: low-interaction (use simulated browser and minimal OS features), high-interaction (use real browser and full OS features) and hybrid (combine qualities of both). Low-interaction honey-clients such as $HoneyC$ are typically limited to monitoring the traces of activities during the interaction against pre-defined signatures – as a result are not able to detect zero-day exploits due to the static nature of the reference signatures. On the contrary, high-interaction honey-clients such as $Capture - HPC$ and $HoneyMonkey$ [16] check integrity changes in system states which requires monitoring file system, registry entries, processes, network connection, and physical resources like memory and CPU consumption anomalies. The advantage of honey-clients, specially high-interaction ones is the deep insight they provide as to the internal details of attack payloads embedded in malicious websites. However, they are resource-intensive as they need to load and execute individual pages under analysis and modern web pages are usually stuffed with rich client-side code and multimedia taking longer analysis time per a web page. Moreover, not all web pages are likely to launch attacks upon visiting. There are web pages which demand user interaction or wait for time-bombs to take action, which makes honey-clients inflexible for such websites. From evasion perspective, IP addresses of honey-clients are likely to be black-listed by malicious servers, their virtual machines be detected through advanced fingerprint identification techniques and they may also be victims of CAPTCHA verification that necessarily involve human visitors.

Given the common and specific limitations of existing approaches, an effective and efficient method to analyze a web page and alert web users prior to visiting a potentially malicious website is still far from reliably available to protect web users from attacks. In addition, a problem that is not yet well-addressed is a fine-grained characterization of rich set of features pertinent to the ever-changing malicious payloads and multiple artifacts of malicious websites. In what follows, we pose research questions with emphasis on effectiveness and efficiency issues that challenge existing approaches with respect to common challenges, web page feature issues, and analysis and detection techniques.

Common Challenges.
The vast majority of existing approaches dealing with analysis and detection of malicious websites base the foundation of their techniques on a prominent attack. However, an attacker can craft virtually any variation of an existing or newly devised attack and embed it into web pages. As a result, the majority of the techniques not only ignore a different type of attack than the one they are primarily devised for, but also are likely to miss the fine-grained features characterizing a malicious web page since the techniques, by design, are based on limited set of features. As a result, the techniques suffer from giving valuable clues about a comprehensive snapshot of attack payloads based on which the analysis and detection techniques could be improved.

No matter how effective and efficient a security technique is, it is as strong as the extent to which it stands resistant to possible countermeasures by attackers. To this end, techniques to analyze and detect malicious websites are not exception. Honey-clients, for instance, by their very design nature, are vulnerable to fingerprinting. Even approaches that leverage page content are subject to evasion due to sophisticated obfuscation and cloaking.

Feature Completeness: The number and the semantics of feature values in the state-of-the-art are not complete enough to capture a comprehensive snapshot of characteristics of malicious web pages.

Feature Type: While identifying feature types is useful in defining effective and efficient feature extraction and selection schemes, in the existing approaches, none is incorporated or assumed about recurring features(e.g. WHOIS information, URL tokens) as compared to features that change very often (e.g., page content).

Feature Selection, Composition and Values: Given the

different magnitude of contribution by feature values in characterizing malicious web pages, there is no systematic way to prioritize and define an affective and efficient method for computing, selecting, and combining features of different type and semantics in a manner that enhances existing feature extraction techniques. Moreover, feature selection, value computation and, weight assigning are subjective across approaches.

Feature Evolution: Existing web page features may change (become obsolete) and new features may emerge as web pages evolve due to inevitable changes such as change in: page content, functionality, protocols, configuration, browser components, browser extensions, and usage policies. The major bottleneck with most of the existing techniques is lack of flexibility to (semi)automatically and quickly revise and accordingly upgrade the extraction, analysis, and detection techniques, which rely on evolving features.

Analysis and Detection Strategies
Although machine learning techniques are potentially effective in the analysis and detection of malicious websites, there are yet unaddressed questions. First of all, different approaches use partial snapshot of the web page to evaluate different machine learning techniques (e.g. classifier builders). Secondly, the dataset used even for the same feature set is so diverse. Third, the relationship between the web page features used and the machine learning techniques applied is not well-traced experimentally and not in large-scale context except in few recent works such as [1]. As a result, the answer to the question "which machine learning technique is effective and efficient for malicious website detection and why?" is still subjective.

Concerning training of models to build classifiers, whether training is conducted after combining all the distinct feature classes into a single thick feature vector or it is separately conducted for each feature class, remains a question partially-answered. Recently, in [1], it is suggested to combine the models learned for each feature set (URL tokens, host-based information and page content in this case) to perform a collaboration-oriented classification. However, the detail of the composition technique and why it is effective needs further investigation.

III. The Roadmap

In response to the problems and questions posed in the previous section, we present our proposal to address the general problem of effectively and efficiently detecting malicious websites and the specific questions about the common challenges shared by current approaches, issues about web page features, and analysis and detection techniques. The core mission in our proposal is not only to detect known malicious websites in the wild but also to uncover not yet known malicious websites with optimized resource consumption.

We envision a more comprehensive, more effective and more efficient approach that takes into account : common challenges, web page feature-related issues and analysis and detection techniques. To tackle the common challenge of focusing on a prominent attack, we propose an aggregation of different types of attack payloads and investigating the relationship among each attack type to look for patterns of convergence towards capturing the big snapshot of an ideal malicious web page.

To achieve a more complete set of features, based on the existing feature set in the state of the art, we are currently building a richer set of features and a feature extraction engine to enhance both feature quantity and feature granularity. Rather than considering restricted set of features, limited to capturing part of the big picture, we are incorporating all the relevant features of web page identity, URL tokens, web page content and web page execution trace. In line with this, we are also investigating the effective methods to combine such features to achieve a more complete characterization of attack payloads. To make the features more useful, we are also refactoring some of them to have a more fine-grained feature values. For example, current approaches consider only presence of remote links on a page. Part of our proposal is to split links into local and remote. In addition, identifying the final targets of links (e.g., other pages, executable files, PDF documents, images) is useful to capture fine-grained values of features.

Another investigation we are undertaking is to draw a solid line between stable features and those changing frequently. Such an identification is a basis for estimating the resource consumption of analyzing a single web page based on which an optimization strategy could be devised. In this regard, one of the possible ways forward is to identify features that are based on stable feature sources and monitor their resource consumption during extraction from the web page. For example, WHOIS information features are relatively stable as opposed to sequence and number of functions called when the browser invokes a plugin.

By devising an algorithm that identifies the most visible contributors in the feature scores and validating these score values using historical profile of feature extraction in addition to domain knowledge, we can filter only the features for which acceptable values are extracted. To this end, we plan to apply state-of-the-art feature selection techniques that minimize redundancy and maximize relevance to get the best candidate feature set with respect to improving effectiveness and efficiency of analysis and detection.

With an ultimate goal of identifying methods to map set of features to set of algorithms, we can incorporate features that are subject to change and as a result define a frame of reference for training on the fly using live feeds of URLs from the real world. In this respect, online learning algorithms [7], [17] instead of the classical ones are demonstrated to be more effective and much faster with

URL tokens and host-based features. We plan to extend the usage of online learning techniques by introducing page content and execution trace features. Building up on the suggestions in [1] about combining models after separate training, another interesting line of investigation we are planning is to experimentally verify the merits and demerits of training each feature class versus training for the union of the feature classes with respect to efficiency and detection accuracy. Another equally important issue to investigate concerning training is the frequency with which the models are updated since the set and values of features extracted from a web page analyzed at time $t1$ may not be the same at time $t2$, and the time frame $t2$-$t1$ has implications both on the accuracy and efficiency of the analysis and detection.

IV. Conclusion

Current approaches in automated analysis and detection of malicious websites have concrete limitations in considering the holistic snapshot of an ideal web page and resilience to possible evasion. Moreover, there are important questions that are yet to be addressed through effective and efficient techniques of feature type identification, maintaining feature evolution, feature composition, and feature value computation. In fact, the analysis and detection techniques which mostly rely on machine learning algorithms also need enhancements in terms of dealing with evolving features, different feature types, and evasion attempts by attackers.

In this position paper, we proposed a holistic approach to effectively and efficiently analyze and detect malicious websites. In particular, we are currently working on enhancement of feature set and feature extraction technique for characterizing malicious payloads by combining page identity, URL tokens, page content, and execution trace so as to capture a complete snapshot of a malicious web page.

Extending proposals by past research, we are also investigating effective incorporation of feature evolution, feature types and systematic composition of feature classes to improve analysis and detection of malicious websites. Improving the effectiveness and efficiency of training strategies (e.g., training frequency) using online learning techniques and large-scale experimental validation of our approach, using industry and research benchmarks, with live feed of real-life websites is also within the scope of our future plan.

References

[1] D. Canali, M. Cova, G. Vigna, and C. Kruegel, "Prophiler: a fast filter for the large-scale detection of malicious web pages," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 197–206.

[2] J. Ma, S. L. K., S. Stefan, and V. G. M., "Beyond blacklists: learning to detect malicious web sites from suspicious urls," in *Proceedings of the 15th international conference on Knowledge discovery and data mining*, 2009, pp. 1245–1254.

[3] M. Qassrawi and H. Zhang, "Detecting malicious web servers with honeyclients," *Journal of Networks*, vol. 6, no. 1, 2011.

[4] A. Dewald, T. Holz, and F. C. Freiling, "Adsandbox: sandboxing javascript to fight malicious websites," in *ACM Symposium on Applied Computing*, 2010, pp. 1859–1864.

[5] G. Aggarwal, B. E., J. C., and D. Boneh, "An analysis of private browsing modes in modern browsers." in *Proceedings of the 19th USENIX conference on Security*, 2010, pp. 6–6.

[6] B. S., K. S.T., M. P., and W. M., "Vex: vetting browser extensions for security vulnerabilities," in *Proceedings of the 19th USENIX conference on Security*, 2010, pp. 22–22.

[7] M. Justin, S. L. K., S. Stefan, and V. G. M., "Identifying suspicious urls: an application of large-scale online learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09, 2009, pp. 681–688.

[8] H. Yung-Tsung, C. Yimeng, C. Tsuhan, L. Chi-Sung, and C. Chia-Mei, "Malicious web content detection by machine learning," *Expert Syst. Appl.*, vol. 37, pp. 55–60, 2010.

[9] C. Seifert, I. Welch, and P. Komisarczuk, "Identification of malicious web pages with static heuristics," in *Proceedings of the Australasian Telecommunication Networks and Applications Conference*, 2008.

[10] C. Marco, K. Christopher, and V. Giovanni, "Detection and analysis of drive-by-download attacks and malicious javascript code," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 281–290.

[11] M. Alexander, B. Tanya, D. Damien, G. S. D., and L. H. M., "Spyproxy: execution-based detection of malicious web content," in *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, 2007, pp. 3:1–3:16.

[12] S. Ford, M. Cova, C. Kruegel, and G. Vigna, "Analyzing and Detecting Malicious Flash Advertisements," in *Proceedings of the Annual Computer Security Applications Conference*, 2009.

[13] A. Ikinci, T. Holz, and F. Freiling, "Monkey-spider: Detecting malicious websites with low-interaction honeyclients," in *Proceedings of Sicherheit, Schutz und ZuverlŁssigkeit*, 2008, pp. 407–421.

[14] B.-I. K., C.-T. I., and H.-C. J., "Suspicious malicious web site detection with strength analysis of a javascript obfuscation," in *International Journal of Advanced Science and Technology*, 2011, pp. 19–32.

[15] R. K., K. T., and D. A., "Cujo: efficient detection and prevention of drive-by-download attacks." in *Proceedings of the 26th Annual Computer Security Applications Conference*, 2010, pp. 31–39.

[16] Y.-M. W., X. J. Doug B., C. V. Roussi R., and S. T. K. Shuo C., "Automated web patrol with strider honeymonkeys: Finding web sites that exploit browser vulnerabilities." in *Proceedings of the Network and Distributed System Security Symposium*, 2006.

[17] J. T. Ma, "Learning to detect malicious urls." in *PhD Dissertation: University of California, San Diego*, 2010.

Systems Security research at Politecnico di Milano

Federico Maggi
Dipartimento di Elettronica e Informazione
Politecnico di Milano
Milano, Italy
fmaggi@elet.polimi.it

Stefano Zanero
Dipartimento di Elettronica e Informazione
Politecnico di Milano
Milano, Italy
zanero@elet.polimi.it

***Abstract*—This paper summarizes the past, present and future lines of research in the systems security area pursued by the Performance Evaluation Lab (VPLab) of Politecnico di Milano. We describe our past research in the area of learning algorithms applied to intrusion detection, our current work in the area of malware analysis, and our future research outlook, oriented to the cloud, to mobile device security, and to cyber-physical systems.**

***Keywords*-intrusion detection; malware analysis; computer virology; cloud security; systems security**

I. Introduction

This position paper describes the main research lines we are currently pursuing with our research group at Politecnico di Milano [1], the largest school of engineering in Italy. With over 35,000 students and over 1,400 faculty members, PoliMi has a long tradition of research and teaching in all the domains of technology.

Our small research group works within the *Performance Evaluation Lab* (VPLab) [2], part of the Systems Architecture research area[3], within the department of computer science, *Dipartimento di Elettronica e Informazione*[4] (DEI), which comprises all of the ICT related research areas, with 185 faculty members and slightly short of 230 PhD students and post-doc researchers. Our group was founded about six years ago, and now includes one Assistant Professor (Stefano Zanero) and one post-doctorate research fellow (Federico Maggi), and has a steady-size of 3-4 research assistants and a yearly average of 6-7 BSc and MSc students. Our research group participates actively in research projects, including the FP7 STREP project WOMBAT[5], and the NoE SysSec[6], as well as the NATO SfP project SCADA-NG.

Our research has started from the application of unsupervised learning techniques to security issues, particularly in the field of anomaly-based network intrusion detection. It now encompasses several topics, including malware analysis and virology.

[1]http://www.polimi.it
[2]http://www.vplab.elet.polimi.it
[3]http://sagroup.ws.dei.polimi.it
[4]http://www.dei.polimi.it
[5]http://wombat-project.eu
[6]http://syssec-project.eu

In the following we will briefly outline our historical background in system security (Section II), describe our current research interests in malware analysis (Section III) and finally outline our most recent interests in smart devices, the cloud and cyber-physical systems (Section IV).

II. The past: Learning in Intrusion Detection

Our original interests lied in applying unsupervised learning algorithms to intrusion detection tasks [1]. Over the years, this evolved into several different projects:

- ULISSE, a network based unsupervised learning IDS based on self-organizing maps and outlier detection [1], [2]. Interestingly, ULISSE was one of the first NIDS that proposed to apply learning to the payloads of network packets, and also one of the first IDS to apply a double tier of learning. The architecture of ULISSE is shown in Figure 1
- S^2A^2DE, a host based IDS based on the analysis of the sequence and the arguments of system calls on Linux [3], [4], an evolution of SyscallAnomaly [5]. The architecture of S^2A^2DE is shown in Figure 2
- Masibty, a web application IPS based on the analysis of the sequence, the parameters and the contents of HTTP messages, correlated with SQL queries and results to detect anomalies [6], [7].

Over the years, our research explored two interesting concepts: the possibility of using multiple layers of algorithms to analyze complex interactions (this is well outlined in [1], [3], [6]), and the conversely important issue of coordinating multiple models and visions of a phenomenon into a coherent view. The latter problem was explored both in the area of a posteriori aggregation of data coming from different sources [8], and in the area of combination of multiple models inside a single intrusion detector. In this specific area, we were the first to propose to exploit cooperative negotiation among agents as a model [9], [7].

III. The present: malware analysis and computer virology

Our original interest in this area was triggered by virus propagation models and their mathematical expression [10],

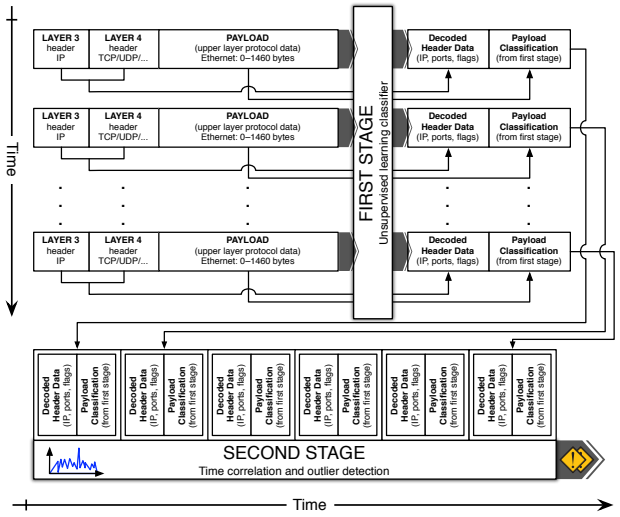


Figure 1. The architecture of the NIDS ULISSE

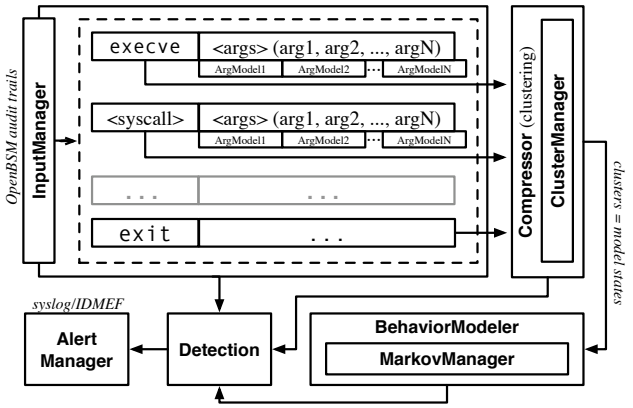


Figure 2. The architecture of the HIDS S^2A^2DE

[11]. However, our focus quickly shifted to the issue of propagation of bluetooth and wireless malware, with a seminal experiment known as the Bluebag which received worldwide media attention [12]. More recently, our BlueBat bluetooth honeypots [13] led us to express serious doubts about the actual dangerousness of wireless-enabled malware [14]. Our ongoing research is now trying to say a final word on the subject of viability of wireless-spread infections.

Another issue we explored, jointly with colleagues from Technical University of Vienna and University of California, Santa Barbara, is automation of malware analysis. Today, each newly discovered malware binary must be analyzed mostly by hand, to understand its capabilities, its level of threat and its potential impact. We developed an hybrid approach mixing dynamic and static analysis, to overcome their symmetric limitations. Dynamic analysis is unlikely to explore all of the malware capabilities, i.e., to execute all of the reachable code, as most modern malware includes triggers that execute certain functions only if some conditions are verified. On the other hand, static analysis is very difficult to automatize. We proposed a system called Reanimator [15] that exploits similarities in the code base among different malware samples, specifically by identifying interesting behaviors, mapping them back to the code implementing them, and creating a resilient set of fingerprints based on the Control Flow Graph (CFG) of said genotypes. A limitation of Reanimator is that the analyst needs to define manually the behaviors of interest. For this reason, we are working to exploit clustering on both the structural features of a malware collection and the dynamic features. Our objectives are threefold:

1) since dynamic clustering necessarily works on an incomplete set of features (because of the inherent incompleteness of dynamic analysis), comparison between the dynamic and static analysis may reveal the presence of dormant behaviors in malware samples, thus helping to improve the clustering in malware families.
2) we can use dynamic clustering to automatically derive "interesting behavior" sets from the malware dataset itself, thus further automating the approach laid out by Reanimator and removing one manual step.
3) since arguably a behavior can be reimplemented from scratch, thus "blinding" the techniques used by Reanimator (but not the dynamic behavioral clustering), we can use the matching between statically and dynamically derived clusters to expand the set of signatures for a behavior, in order to improve the ability to detect it in other dormant samples.

Another research problem we have been devoting attention to is the analysis malware naming inconsistencies. In particular, we recently came up with a demonstration that major inconsistencies plague the naming convention and malware taxonomy employed by different vendors. This creates an obvious issue for researchers focusing on integrating and systematizing threats, for instance to create ground truths for automated analysis approaches.

In addition to topics strictly related to the malware domain, we have been focusing on two aspects of the current threat scenario. First, we have been conducting the largest and most realistic data collection experiment on the World Wide Web that features more than 5,300 users. The goal of this experiment is to determine accurately the extent to which short URLs, one of the most revolutionary technologies in Web 2.0, masquerade significant threats, by acting as "amplifiers" of the attack surface (i.e., web clients) with respect to attack vectors such as phishing, drive-by download and spam URLs. The first phase of this large experiment is concluded and its results, which will be soon submitted for review, represent a significant improvement

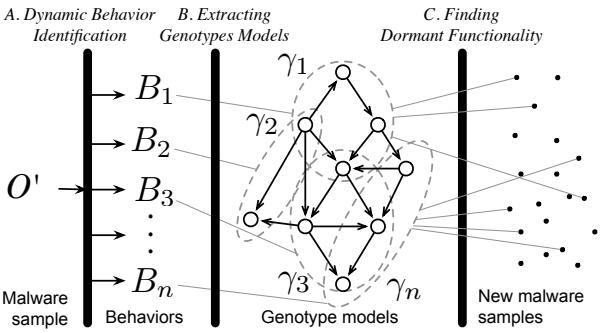


Figure 3. The architecture of Reanimator

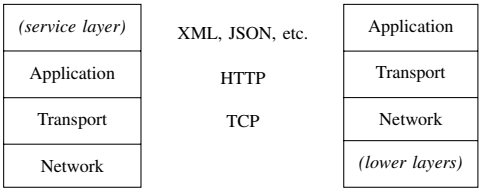


Figure 4. The change in the networking stack is noticeable from the traditional application layer (left) that, in the case of HTTP, is playing the role of a transport protocol (right) to encapsulate upper layer protocols (e.g., SOAP, JSON, XML), typical of modern web services.

over the state-of-the-art work [16]. Last, we are investigating underground economy and the usage of captured credentials is an issue we are currently devoting some research efforts. In this context, besides some unpublished work which we cannot yet disclose, we have explored the phenomenon of reverse vishing. We have built an architecture, described in Figure 5, to capture relevant data for analyzing this growing phenomenon.

IV. The future: smart devices, the cloud, and cyber-physical systems

Arguably, the future of ICT is characterized by a pervasive access to the Internet through smart devices, by the extension of the cloud computing paradigm, and by the increasing interaction between the digital and the physical world. We have already mentioned our work on bluetooth-enabled smartphones [12], [13]. Building upon this expertise, we are currently working on the vulnerabilities of smartphone user interfaces and input systems.

We have some past experience on analyzing the grid computing paradigm [17], arguably one of the ancestors of the upcoming cloud computing revolution. In this area, we are working on the basis of the observation that there is a strong parallel between the emerging paradigm of cloud computing and the traditional time-sharing era [18]. Clouds are the modern reincarnation of mainframes, available on a pay-per-use basis, and equipped with virtual, elastic, disks-as-a-service that replace the old physical disks with quotas. This comparison, beyond being fascinating in its own self, prepares the ground for a constructive critique regarding the security of such a computing paradigm and, especially, of one of its key components: web services. Along this line, we concentrate on a few, critical hypotheses that demand particular attention. Although in this emerging landscape only a minority of threats qualify as *novel*, they could be difficult to recognize with the current countermeasures, given the change that the new computing paradigm has induced in the use of the network stack (see Figure 4), and thus can expose web services to new attacks. Our current research works by analyzing the traditional countermeasures such as intrusion detection systems, developed to mitigate well-known web security threats, and by trying to explore the affinities and differences when trying to use them within the cloud computing paradigm.

The final new trend that we wish to study is the emerging class of security issues that arise in the interstitial layer between safety-critical, physical systems and digital, pervasive systems (a typical example is a SCADA-controlled industrial process, but in the near future we can foresee more and more such interactions).

The increasing interconnection between such systems creates new attack surfaces that are neither physical nor digital, and which cannot be identified if such systems are studied and secured separately as is customarily done nowadays. As such systems are prevalent in critical infrastructures such as power grids or water-distribution plants, they are primary target for cyber-terrorism and cyber-warfare attacks.

We are beginning to investigate this emerging class of vulnerabilities with an empirical, bottom-up methodology. We will start from devising real-world attacks against both the digital and the physical side of carefully selected target systems, and strive to unveil recurring vulnerability patterns that can be generalized to a (possibly novel) class of vulnerabilities. To ensure their real-world applicability, we will validate, step by step, assumptions, results and countermeasures on replicas, models or simulated systems in controlled environments, with the help of industrial partners.

As a first result, we aim to generate actionable assess-

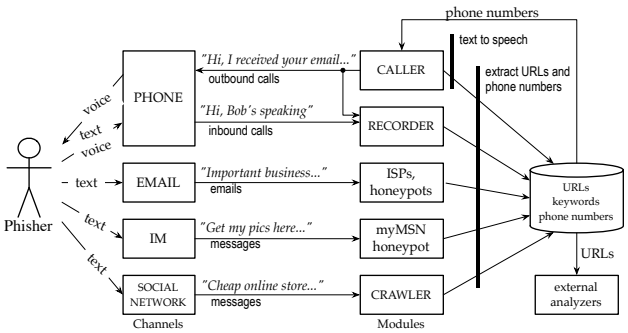


Figure 5. Our social-engineering centric data collection architecture.

ments of the security of representative, high-profile systems, structured in a series of novel attack papers, and correspondingly produce targeted countermeasures. However, our more ambitious long term goal is to formalize the general problem and its root causes, by attempting to develop a theory, a taxonomy and a methodology for describing, identifying and assessing this novel class of vulnerabilities. Leveraging this general theory, and systematizing our observations, we will also try to produce preemptive secure design patterns and general solutions for this class of problems.

ACKNOWLEDGMENT

The research described in this position paper has received funding from the European Union, under the 7th Framework Programme (FP7/2007-2013) under grant agreements nr. 257007 (SysSec) and 216026 (WOMBAT). The opinions expressed in this paper are those of the authors and do not necessarily reflect the views of the European Commission. Further funding for our research has been received from NATO under the SfP programme, contract nr. 983805 "SCADA-NG".

REFERENCES

[1] S. Zanero and S. M. Savaresi, "Unsupervised learning techniques for an intrusion detection system," in *Proceedings of the 2004 ACM Symposium on Applied Computing (SAC), Nicosia, Cyprus, March 14-17, 2004*, H. Haddad, A. Omicini, R. L. Wainwright, and L. M. Liebrock, Eds. ACM, 2004, pp. 412–419.

[2] S. Zanero, "Ulisse, a network intrusion detection system," in *Proceedings of the 4th annual workshop on Cyber security and information intelligence research: developing strategies to meet the cyber security and information intelligence challenges ahead*, ser. CSIIRW '08. New York, NY, USA: ACM, 2008, pp. 20:1–20:4.

[3] F. Maggi, M. Matteucci, and S. Zanero, "Detecting intrusions through system call sequence and argument analysis," *Dependable and Secure Computing, IEEE Transactions on*, vol. 7, no. 4, pp. 381 –395, Oct-Dec 2010.

[4] A. Frossi, F. Maggi, G. L. Rizzo, and S. Zanero, "Selecting and improving system call models for anomaly detection," in *Detection of Intrusions and Malware, and Vulnerability Assessment, 6th International Conference, DIMVA 2009, Como, Italy, July 9-10, 2009. Proceedings*, ser. Lecture Notes in Computer Science, U. Flegel and D. Bruschi, Eds., vol. 5587. Springer, 2009, pp. 206–223.

[5] D. Mutz, F. Valeur, G. Vigna, and C. Kruegel, "Anomalous system call detection," *ACM Trans. Inf. Syst. Secur.*, vol. 9, pp. 61–93, February 2006.

[6] C. Criscione, G. Salvaneschi, F. Maggi, and S. Zanero, "Integrated detection of attacks against browsers, web applications and databases," in *Proceedings of the 2009 European Conference on Computer Network Defense*, ser. EC2ND '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 37–45.

[7] A. Volpatto, F. Maggi, and S. Zanero, "Effective multimodel anomaly detection using cooperative negotiation," in *Decision and Game Theory for Security - First International Conference, GameSec 2010, Berlin, Germany, November 22-23, 2010. Proceedings*, ser. Lecture Notes in Computer Science, T. Alpcan, L. Buttyán, and J. S. Baras, Eds., vol. 6442. Springer, 2010, pp. 180–191.

[8] F. Maggi, M. Matteucci, and S. Zanero, "Reducing false positives in anomaly detectors through fuzzy alert aggregation," *Information Fusion*, vol. 10, no. 4, pp. 300–311, 2009.

[9] F. Amigoni, F. Basilico, N. Basilico, and S. Zanero, "Integrating partial models of network normality via cooperative negotiation: An approach to development of multi-agent intrusion detection systems," in *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Intelligent Agent Technology, Sydney, NSW, Australia, December 9-12, 2008*. IEEE, 2008, pp. 531–537.

[10] G. Serazzi and S. Zanero, "Computer virus propagation models," in *Performance Tools and Applications to Networked Systems*, ser. Lecture Notes in Computer Science, M. Calzarossa and E. Gelenbe, Eds. Springer Berlin Heidelberg, 2004, vol. 2965, pp. 26–50.

[11] E. Filiol, M. Helenius, and S. Zanero, "Open problems in computer virology," *Journal in Computer Virology*, vol. 1, no. 3-4, pp. 55–66, 2006.

[12] L. Carettoni, C. Merloni, and S. Zanero, "Studying bluetooth malware propagation: The bluebag project," *IEEE Security & Privacy*, vol. 5, no. 2, pp. 17–25, 2007.

[13] A. Galante, A. Kokos, and S. Zanero, "Bluebat: Towards practical bluetooth honeypots," in *Proceedings of IEEE International Conference on Communications, ICC 2009, Dresden, Germany, 14-18 June 2009*. IEEE, 2009, pp. 1–6.

[14] S. Zanero, "Wireless malware propagation: A reality check," *IEEE Security & Privacy*, vol. 7, no. 5, pp. 70–74, 2009.

[15] P. M. Comparetti, G. Salvaneschi, E. Kirda, C. Kolbitsch, C. Kruegel, and S. Zanero, "Identifying dormant functionality in malware programs," in *31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berleley/Oakland, California, USA*. IEEE Computer Society, 2010, pp. 61–76.

[16] D. Antoniades, I. Polakis, G. Kontaxis, E. Athanasopoulos, S. Ioannidis, E. Markatos, and T. Karagiannis, "we.b: The web of short URLs," in *WWW 2011*, 2011.

[17] S. Zanero and G. Casale, "Givs: integrity validation for grid security," *IJCIS*, vol. 4, no. 3, pp. 319–333, 2008.

[18] F. Maggi and S. Zanero, "Rethinking security in a cloudy world," Politecnico di Milano, Tech. Rep. TR-2010-11, 2010, http://home.dei.polimi.it/fmaggi/downloads/publications/2010_maggi_zanero_cloud_security.pdf.

Systems Security Research at Ruhr-University Bochum

Thorsten Holz
Research Group "Embedded Malware"
Ruhr-University Bochum
Email: thorsten.holz@rub.de

***Abstract*—The Horst Görtz Institute (HGI) located at Ruhr-University Bochum, Germany, is one of Europe's largest university-based institution for interdisciplinary research in the field of IT security. Ruhr-University Bochum offers both Bachelor and Master degree programs in IT security and more than 500 students are enrolled in these study courses.**

The research at RUB focusses on cryptography, embedded security, and network security. Recently, the research group "Embedded Malware" was established, which focusses on systems security. In this position paper, we provide a brief overview of this group and discuss previous research and future research topics.

I. Introduction

The Horst Görtz Institute (HGI) located at Ruhr-University Bochum, Germany, is one of Europe's largest university-based institution for interdisciplinary research in the field of IT security. The HGI consists of research groups from the Departments of Electrical Engineering & Information Sciences, Mathematics, Economics, and Law. Ruhr-University Bochum offers both Bachelor and Master degree programs in IT security since more than 10 years. At this point, more than 500 students are enrolled in these study courses and about 10 research groups perform research in the area of IT security.

Recently, the research group "Embedded Malware" was established. The research activity of the group focusses on different topics from the areas *systems security* and *network security*. Specifically, researchers in the group are interested in *applied IT security*, a research field that deals with the security of deployed systems and the analysis of real-world phenomena observed in the wild. The research topics include honeypots and honeynets, botnet detection and mitigation, binary analysis, smartphone security, and similar aspects from the area applied IT security. In the following, we discuss these different areas and provide both an overview of previous research and an outlook for future research plans.

More information about the HGI and the research group "Embedded Malware" is available at www.hgi.rub.de and www.emma.rub.de, respectively.

II. Honeypots and Honeynets

Honeypots are electronic bait (i.e., network resources such as computers, routers, or switches) that are deployed to be probed, attacked, and compromised. Honeypots run special software which permanently collects data about the system behavior and greatly aids in post-incident computer and network forensics. Several honeypots can be assembled into networks of honeypots called *honeynets*. Through the wealth of data collected through them, honeynets are considered a useful tool to learn more about attack patterns and attacker behavior in real environments.

Together with Niels Provos from Google, Thorsten Holz wrote the book "*Virtual Honeypots: From Botnet Tracking to Intrusion Detection*" [22] which became the standard reference in the area of honeypots and honeynets. In the past few years, members of the group were involved in several honeypot tools that can be used to study current attacks against computer systems [1, 5, 21, 32].

In the future, we plan to continue this line of work and develop novel honeypot tools that can be used to study current attack vectors. For example, we would like to understand how the concept of honeypots can be applied in the area of embedded systems, especially for mobile phones. Another area of future work is developing honeypots that can be used to protect critical infrastructures (e.g., control systems or the smart grid). Recent attacks such as *Stuxnet* that attacked a specific type of control systems demonstrated that this threat is real and we believe that honeypots offer a valuable tool to study this kind of attacks.

III. Botnet Detection and Mitigation

The term *bot* is derived from the word *robot* and refers to a program which can, to some degree, act in an autonomous manner. A computer system that can be remotely controlled by an attacker is called a bot or *zombie*. A *botnet* is a network of compromised machines under the control of an attacker. Botnets are responsible for many of today's abuses on the Internet, e. g., an attacker can use the compromised machines to send large amount of spam mails or to perform Distributed Denial-of-Service (DDoS) attacks. To control a large group of bots, a *Command and Control* (C&C) mechanism is used by the attacker. The easiest form are botnets with a *central* C&C server. Such botnets can use two different communication mechanisms to distribute the attacker's commands: on the one hand, botnets can implement a *push* mechanism in which the attacker pushes the command to each infected machine (e.g., this is the common setup for IRC-based botnets). On the other hand, botnets can

implement a *pull* mechanism like for example HTTP-based botnets do: periodically, each bot requests a specific URL from the C&C server, in which it encodes status information. As a reply, the server sends to the infected machine the command it should execute. Botnets with a central server have a single point of failure: once the central server is offline, the whole botnet is non-functional since the attacker cannot send commands to the infected machines anymore. From an attacker's point of view, it is thus desirable to have a more robust communication mechanism within the botnet. As a result, new botnet structures emerged that use *peer-to-peer-based* communication protocols: each infected machine is a peer that can relay messages and act as a client and server. Such botnets are harder to mitigate since no central C&C server can be taken offline.

In the area of botnet research, we published one of the first systematic studies on botnets and introduced a methodology to detect botnets [7]. In this paper, we showed that it is possible to identify, infiltrate, and analyze the remote control mechanism used within botnets with a central C&C server in an automated way. Later on, we extended this methodology to botnets with a peer-to-peer-based communication channel [14]. To demonstrate the practical feasibility of the proposed methodology, we studied several different kinds of botnets and showed how the method can be applied to each of them. In a first example, we studied botnets that use a central C&C server [7, 9, 11] and then extended the study to botnets with peer-to-peer-based communication [14, 27]. As a third example, we studied so called *fast-flux service networks* (FFSNs), a different kind of botnet. The idea behind these networks is that the attacker does not directly abuse the compromised machines, but uses them to establish a proxy network on top of these machines to enable a robust hosting infrastructure. Our method can be applied to this kind of botnets as well and we presented empirical results supporting this claim [13, 19].

Besides studying botnets, we also developed several network-level techniques to detect botnets [8, 24, 31]. All these approaches focussed on vertical correlation, i.e., the tools aim at detecting tokens or signatures within the network traffic that indicate infected machines. While *Rishi* [8] focussed on detecting artifacts of the communication channel of IRC-based botnets based on manual analysis, recent work focussed on automatically generating network signatures: we developed a system to generate signatures that model the behavior of bots [31] and another system that can detect bots in high-speed networks [24] based on invariant patterns in the communication channel between a bot and the controller.

In the future, we want to continue the work in this area and develop novel techniques to detect botnets. The main limitation of current approaches is that the generated signatures have *no contextual information*, i.e., it is not clear what information is encoded within a signature and what meaning an alert has (e.g., is the alert generated due to the download of a config file or because information is exfiltrated from a compromised machine?). Furthermore, the signatures use content-based analysis to detect command tokens. Thus, these systems have problems with encrypted or obfuscated command channels. This is a limitation that our approach shares with *all previous techniques* that aim to detect single bots. As part of our future work, we plan to address these limitations by combining network-level analysis with information extracted at the host level. As a first step, we studied how to leverage host-based information that provides insights into which data is sent over each network connection as well as the ways in which a bot processes the information that it receives [16]. Furthermore, we aim at detecting the *effects* of bot infections since these artifacts should always be visible either at the host- or network level.

IV. Binary Analysis

With the help of honeypots and other techniques, we can collect a binary copy of autonomous spreading malware without any human interaction. In order to learn more about the remote control structure behind such malware, we also need to automatically analyze a collected binary: we want to extract all important information from the given binary program in some kind of automated way. The automated analysis should focus on (1) *automation*, (2) *effectiveness* and (3) *correctness*. Automation means that the analysis tool should create a detailed analysis report of a malware sample quickly and without user intervention. A machine readable report can in turn be used to initiate automated response procedures like updating signatures in an intrusion detection system. Effectiveness of a tool means that all relevant behavior of the malware should be logged, no executed functionality of the malware should be overlooked. This is important to realistically assess the threat posed by the malware sample. Finally, a tool should produce a correct analysis of the malware, i.e., every logged action should in fact have been initiated by the malware sample to avoid false claims about it.

We utilize two different tools to perform the automated analysis: *Anubis* and *CWSandbox*. The basic idea of both approaches is to execute the binary in an instrumented environment and observe during runtime the behavior of the sample. As a result, we obtain a detailed report that summarizes the observed behavior, e.g., changes to the filesystem or network activity. Developing such an analysis system is challenging since malicious software employs many tricks to defeat analysis, thus many obstacles had to be solved. *CWSandbox* was developed at the University of Mannheim, while *Anubis* was developed at the Vienna University of Technology and there are close research relationships to both universities [18, 29].

The analysis reports generated by such tools can be used for classification [23] and clustering [25] of malicious soft-

ware. Furthermore, we developed a binary analysis technique based on program slicing to extract from a given binary exactly the instructions related to a specific behavior [18]. Based on this preparatory work, we plan to develop methods to understand a given binary executable in detail. The extracted information will then be used together with the observed network behavior to generate enhanced signatures to detect botnet communication, even if the attackers try to hide the communication or blend it with benign-looking communication. Developing better binary analysis techniques for example based on symbolic execution or information flow analysis are other topics that we are interested in. Again, we will study how to analyze malware on embedded systems, especially smartphones, in the future.

V. Smartphone Security

With more than four billion GSM users world-wide, there is a need to study the security aspects of mobile phones. A recent trend in this area are *smartphones*: numerous new "smart" devices such as BlackBerries, iPhones and Android-based phones have been introduced that revolutionized the market. Many researchers and practitioners are expecting a major security incident with mobile phones ever since these devices began to become more powerful: with increased processing power and memory, increased data transmission capabilities of the mobile phone networks, and with open and third-party extensible operating systems, they became an interesting target for attackers.

We recently saw the first real attacks against smartphones: In March 2010, Iozzo and Weinmann demonstrated a drive-by download attack against an iPhone 3GS that enabled an attacker to steal the SMS database from the phone [20]. In November 2010, one of the first public exploits to perform an attack against the mobile browser shipped with Android was released [17]. Recently, Weinmann introduced the first over-the-air exploitation of memory corruptions in GSM software stacks [28] and Oberheide and Lanier identified several attack vectors against the iTunes App Store [10].

Thus, there is a need to study security aspects of smartphones and develop tools and techniques to protect these devices. We argue that smartphones have some unique properties such as limited battery, a different instruction set architecture (e.g., smartphones typically use ARM processors), and an inherent billing system [3]. We try to transfer some of the research results obtained in other areas to the embedded world. Specifically, we are working on a project called *MobWorm* in which we examine both attack and defense strategies for smartphones. On the one hand, we study how smartphones can be attacked both from the user side and over-the-air via rogue base stations. On the other hand, we also develop protection techniques such as a binary instrumentation framework that embeds control-flow integrity mechanisms into a binary to protect against runtime attacks on software. This enables us to prevent advanced attacks (e.g., return-oriented programming [15, 26]). Another topic is light-weight protection mechanisms for mobile browsers in order to protect them against heap spraying and similar attack vectors.

VI. Underground Economy

With the growing digital economy, it comes as no surprise that criminal activities in digital business have lead to a *digital underground economy*. Because it is such a fast-moving field, tracking and understanding this underground economy is difficult. Typically, the only observable evidence of this economy refers to *indirect effects* of underground markets, such as announcements of trading and offers of stolen credentials in public IRC channels [6]. We investigated keylogging attacks and provided a detailed analysis of the collected data, giving a first-hand insight into the underground economy of Internet criminals from a unique and novel viewpoint [12]. We believe that our method can be generalized to many other forms of credential-stealing attacks, and that it helps to get a better understanding of how the underground markets work.

This type of research complements the technical aspects of our work: while we can achieve a lot of security based on technical means, we think that we also need to study the problem from a different angle, for example from an economical point of view. This enables us to analyze the financial impact of security breaches and we can develop additional countermeasures that make cybercrime less interesting for attackers. We studied for example the financial motives of attackers [33] and the effects of spam on stock markets [4]. In the future, we plan to continue work in this area and analyze other ways to study the financial motives of attackers and possible countermeasures from an economical point of view. Especially spam is an interesting research subject since botnets are the root cause behind a lot of these unsolicited bulk email messages. Thus, studying spam links different research subjects such as honeypots, botnets, and also binary analysis.

VII. Social Networks

A complementary field of research deals with security of social networking sites such as *Facebook*, *LinkedIn*, *Twitter*, and *Xing*. These networks have been increasingly gaining in popularity. In fact, Facebook has been reporting growth rates as high as 3% per week, with more than 500 million registered users as of March 2011. Furthermore, people spend over 700 billion minutes per month on Facebook, and it is reported to be one of the largest photo storage site on the web with over one billion uploaded photos. Clearly, popular social networking sites are critical with respect to security and especially privacy due to their very large user base.

We studied different security aspects of popular social networks. On the one hand, we introduced a practical de-anonymization attack that makes use of the group informa-

tion in social networking sites [30]. Using empirical, real-world experiments, we showed that the group membership of a user in a social network (i.e., the groups within a social network in which a user is a member), may reveal enough information about an individual user to identify her when visiting web pages from third parties.

On the other hand, we studied how an attacker can profile users by taking advantage of a common weakness of these networks, namely the fact that an attacker can query popular social networks for registered e-mail addresses on a large scale [2]. Starting with a list of about 10.4 million email addresses, we were able to automatically identify more than 1.2 million user profiles associated with these addresses. By automatically crawling and correlating these profiles, we were able to collect detailed personal information about each user, which we used for automated profiling (i.e., to enrich the information available from each user). In the future, we plan to continue this line of work and study other aspects of social network security, especially attack aspects and how to improve the privacy of users.

VIII. Summary

In this position paper, we provided a brief overview of the research group "Embedded Malware" located at Ruhr-University Bochum. The group focusses on different topics from the areas systems and network security and we feel that there is a large overlap with the topics identified by the SysSec Network of Excellence.

References

[1] P. Baecher, M. Koetter, T. Holz, M. Dornseif, and F. C. Freiling. The Nepenthes Platform: An Efficient Approach to Collect Malware. In *International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2006.

[2] Marco Balduzzi, Christian Platzer, Thorsten Holz, Engin Kirda, Davide Balzarotti, and Christopher Kruegel. Abusing social networks for automated user profiling. In *Recent Advances in Intrusion Detection (RAID)*, 2010.

[3] Michael Becher, Felix C. Freiling, Johannes Hoffmann, Thorsten Holz, Sebastian Uellenbeck, and Christopher Wolf. Mobile Security Catching Up? Revealing the Nuts and Bolts of the Security of Mobile Devices. In *IEEE Symposium on Security and Privacy*, 2011.

[4] R. Boehme and T. Holz. The Effect of Stock Spam on Financial Markets. In *Workshop on the Economics of Information Security (WEIS)*, June 2006.

[5] M. Dornseif, T. Holz, and C.N. Klein. NoSEBrEaK - Attacking Honeynets. In *IEEE Information Assurance Workshop*, June 2004.

[6] J. Franklin, V. Paxson, A. Perrig, and S. Savage. An inquiry into the nature and causes of the wealth of internet miscreants. In *ACM Conference on Computer and Communications Security (CCS)*, 2007.

[7] F. Freiling, T. Holz, and G. Wicherski. Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks. In *European Symposium On Research In Computer Security (ESORICS)*, 2005.

[8] J. Göbel and T. Holz. Rishi: Identify Bot Contaminated Hosts by IRC Nickname Evaluation. In *Usenix Workshop on Hot Topics in Understanding Botnets (HotBots)*, 2007.

[9] J. Göbel, T. Holz, and C. Willems. Measurement and Analysis of Autonomous Spreading Malware in a University Environment. In *Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA)*, July 2007.

[10] A. Greenberg. Google pulls app that revealed Android flaw, issues fix, 2010. "http://news.cnet.com/8301-27080_3-20022545-245.html".

[11] T. Holz. A Short Visit to the Bot Zoo. *IEEE Security & Privacy*, 3(3):76–79, 2005.

[12] T. Holz, M. Engelberth, and F. Freiling. Learning More About the Underground Economy: A Case-Study of Keyloggers and Dropzones. In *European Symposium on Research in Computer Security (ESORICS)*, 2009.

[13] T. Holz, C. Gorecki, K. Rieck, and F. Freiling. Measuring and Detecting Fast-Flux Service Networks. In *Proceedings of 15th Annual Network & Distributed System Security Symposium (NDSS)*, 2008.

[14] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling. Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm. In *Usenix Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, April 2008.

[15] R. Hund, T. Holz, and F. Freiling. Return-Oriented Rootkits: Bypassing Kernel Code Integrity Protection Mechanisms. In *USENIX Security Symposium*, 2009.

[16] Gregoire Jacob, Ralf Hund, Christopher Kruegel, and Thorsten Holz. Jackstraws: Picking Command and Control Connections from Bot Traffic. In *USENIX Security Symposium*, 2011.

[17] MJ Keith. Android 2.0-2.1 Reverse Shell Exploit, 2010. "http://www.exploit-db.com/exploits/15423/".

[18] C. Kolbitsch, T. Holz, E. Kirda, and C. Kruegel. Inspector Gadget: Automated Extraction of Proprietary Gadgets from Malware Binaries. In *IEEE Symposium on Security and Privacy*, 2010.

[19] J. Nazario and T. Holz. As the Net Churns: Fast-Flux Botnet Observations. In *International Conference on Malicious and Unwanted Software*, October 2008.

[20] A. Portnoy. Pwn2Own 2010, 2010. "http://dvlabs.tippingpoint.com/blog/2010/02/15/pwn2own-2010".

[21] F. Pouget and T. Holz. A pointillist approach for comparing honeypots. In *Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, July 2005.

[22] N. Provos and T. Holz. *Virtual Honeypots: From Botnet Tracking to Intrusion Detection*. Addison-Wesley, 2007.

[23] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov. Learning and Classification of Malware Behavior. In *Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA)*, 2008.

[24] K. Rieck, G. Schwenk, T. Limmer, T. Holz, and P. Laskov. Botzilla: Detecting the "Phoning Home" of Malicious Software. In *ACM Symposium on Applied Computing (SAC)*, 2010.

[25] K. Rieck, P. Trinius, C. Willems, and T. Holz. Automatic Analysis of Malware Behavior using Machine Learning. Technical Report Technical Report 18-2009, Berlin Institute of Technology, December 2009.

[26] H. Shacham. The Geometry of Innocent Flesh on the Bone: Return-into-libc Without Function Calls (on the x86). In *ACM Conference on Computer and Communications Security (CCS)*, 2007.

[27] B. Stock, J. Göbel, M. Engelberth, F. Freiling, and T. Holz. Walowdac – Analysis of a Peer-to-Peer Botnet . In *European Conference on Computer Network Defense (EC2ND)*, December 2009.

[28] R.-P. Weinmann. All Your Baseband Are Belong To Us. hack.lu security convention, 2010. http://2010.hack.lu/archive/2010/Weinmann-All-Your-Baseband-Are-Belong-To-Us-slides.pdf.

[29] C. Willems, T. Holz, and F. Freiling. CWSandbox: Towards automated dynamic binary analysis. *IEEE Security and Privacy*, 5(2), 2007.

[30] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A Practical Attack to De-anonymize Social Network Users. In *IEEE Symposium on Security and Privacy*, 2010.

[31] P. Wurzinger, L. Bilge, T. Holz, J. Göbel, C. Kruegel, and E. Kirda. Automatically Generating Models for Botnet Detection. In *European Symposium On Research In Computer Security (ESORICS)*, 2009.

[32] J. Zhuge, T. Holz, X. Han, C. Song, and W. Zou. Collecting autonomous spreading malware using high-interaction honeypots. *Proceedings of ICICS*, 7, 2007.

[33] J. Zhuge, T. Holz, C. Song, J. Guo, X. Han, and W. Zou. Studying Malicious Websites and the Underground Economy on the Chinese Web . In *Workshop on the Economics of Information Security (WEIS)*, June 2008.