

(Vital Infrastructure, Networks, Information
and Control System Management)

REPORT D.2.2

Threats and vulnerabilities, final report

PROJECT TITLE:	Vital Infrastructure, Networks, Information and Control Systems Management			
PROJECT ACRONYM:	VIKING			
GRANT AGREEMENT NUMBER:	225643			
PROJECT START DATE:	01.11.2008			
DURATION:	36 MONTHS			
PROJECT CO-ORDINATOR:	GUNNAR BJÖRKMAN, ABB	(ABB)	DE	
PROJECT MEMBERS:	ABB	(1)	(ABB)	DE
	EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE ZÜRICH	(2)	(ETHZ)	CH
	E.ON	(4)	(E.ON)	DE
	ASTRON INFORMATICS LTD.	(5)	(Astron)	HU
	KUNGLIGA TEKNISKA HÖGSKOLAN	(6)	(KTH)	SE
	UNIVERSITY OF MARYLAND FOUNDATION	(8)	(USMF)	US
	MML ANALYS & STRATEGI	(8)	(MML)	SE

DOCUMENT IDENTIFIER:	D3.1
ISSUE:	1.0
ISSUE DATE:	2011-11-19
PREPARED:	INDUSTRIAL INFORMATION AND CONTROL SYSTEMS, KTH
APPROVED:	2011-11-19
DISSEMINATION STATUS:	PUBLIC

History Chart

Issue	Date	Changed Page (s)	Cause of Change	Implemented by
0.1	2011-09-06	All	Outline	Mathias Ekstedt
1.0	2011-11-19	All	Finalization	Mathias Ekstedt

Authorization

No.	Action	Name	Signature	Date
1	Prepared	Teodor Sommestad (KTH) Hannes Holm (KTH) Mathias Ekstedt (KTH)		2011-11-19
2	Approved	Technical Lead		2011-11-19

The information in this document is subject to change without notice.

All rights reserved.

The document is proprietary of the VIKING consortium members listed on the front page of this document. The document is supplied on the express understanding that it is to be treated as confidential and may not be used or disclosed to others in whole or in part for any purpose except as expressly authorized in terms of Grant Agreement number 225643.

Company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies.

Distribution List

This document is distributed as below. Additional copies held by unnamed recipients will not be updated.

Electronic Copy Number	Name	Address
1	European Commission	EU, Brussels
2	Gunnar Björkman	ABB AG, Mannheim
3	Pontus Johnson	KTH, Stockholm
4 - 9	VIKING consortium members	

Table of Contents

1	Introduction	6
1.1	About the VIKING project.....	6
1.2	Summary of VIKING	6
1.3	Description of deliverable 3.1	8
1.3.1	Objectives.....	9
1.3.2	A brief overview of the Cyber Security Modelling Language (CySeMoL)	9
1.4	Outline of the document.....	12
2	An example SCADA architecture.....	13
2.1	Model	13
2.1.1	External DMZ network	13
2.1.2	Office network	14
2.1.3	Internal DMZ network.....	15
2.1.4	SCADA network	16
2.1.5	Process.....	17
2.1.6	Data flows.....	18
2.2	Attack analysis	19
2.2.1	Compromise PCU Communication as an outsider	20
2.2.2	Write to the Historian database as an outsider	24
3	Real-world case studies.....	28
3.1	Utility 1	28
3.1.1	Aim, scope and delimitations	28
3.1.2	Models created and data collected	28
3.1.3	Analysis performed.....	28
3.1.4	Lessons learned	29
3.2	Utility 2	29
3.2.1	Aim, scope and delimitations	29
3.2.2	Models created and data collected	29
3.2.3	Analysis performed.....	30
3.2.4	Lessons learned	30
3.3	Utility 3	31
3.3.1	Aim, scope and delimitations	31
3.3.2	Models created and data collected	31
3.3.3	Analysis performed.....	31
3.3.4	Lessons learned	31
4	Appendix	33

4.1	States of instantiated AccessControlPoints.	34
4.2	States of instantiated ApplicationClients.	34
4.3	States of instantiated Dataflows.	35
4.4	States of instantiated DataStores.	35
4.5	States of instantiated DeepPacketInspections.....	35
4.6	States of instantiated Firewalls.....	35
4.7	States of instantiated NetworkInterfaces.	36
4.8	States of instantiated NetworkZones.....	36
4.9	States of instantiated OperatingSystems.	37
4.10	States of instantiated PasswordAccounts.	37
4.11	States of instantiated PasswordAuthenticationMechanisms.	38
4.12	States of instantiated Persons.....	38
4.13	States of instantiated PhysicalZones.	38
4.14	States of instantiated Protocols.	39
4.15	States of instantiated SecurityAwarenessPrograms.	39
4.16	States of instantiated Services.....	40
4.17	States of instantiated SoftwareProducts.	41
4.18	States of instantiated ZoneManagementProcesses.	41

Figures

Figure 1 – A typical SCADA system architecture.	7
Figure 2 – Overall structure of VIKING.	8
Figure 3. High-level description of network architecture of the example, divided into parts a), leftmost part, and b), rightmost part.	13
Figure 4. Systems in the external DMZ network.	14
Figure 5. Systems in the office network.	15
Figure 6. Systems in the internal DMZ network.	16
Figure 7. Systems in the SCADA network.	17
Figure 8. Systems in the process network.	17
Figure 9. Data flows between and inside the networks. a) Model as a whole. b) Leftmost part of the model. c) Middle part of the model. d) Rightmost part of the model.	19
Figure 10. Visual description of the calculated weakest attack path.	21
Figure 11. Visualisation of attack path in SCADA network view. (This is the same view as the one in Figure 7)	22
Figure 12. Mitigation through complete software updates	22

Figure 13. Mitigation through a network-layer deep packet inspection filter	23
Figure 14. Attack with IPS as mitigation. Illustrated in SCADA network View. Note the IPS now added to the SMB service.	24
Figure 15. Visual description of the calculated weakest attack path.	25
Figure 16. Mitigation through password policies.	26
Figure 17. Mitigation through security awareness training.	26
Figure 18. Mitigation through software patching	27

1 **Introduction**

1.1 About the VIKING project

The VIKING (*Vital Infrastructure, Networks, Information and Control Systems Management*) project is an EU financed Framework 7 Collaborative STREP Project and is part of themes 4, ICT, and 10, Security. It will be executed between November 1, 2008 and October 31, 2011 by a consortium of industrial and academic partners.

1.2 Summary of VIKING

The main objectives of VIKING are:

- To investigate the vulnerability of SCADA (Supervisory Control And Data Acquisition) systems operating electric power systems and the cost of cyber attacks on society
- To propose and test strategies and technologies to mitigate these weaknesses
- To increase the awareness for the importance of critical infrastructures and the need to protect them

Society is increasingly dependent on the proper functioning of the electric power system, which in turn supports most other critical infrastructures: water and sewage systems; telecommunications, internet and computing services; air traffic, railroads and other transportation. Many of these other infrastructures are able to operate without power for shorter periods of time, but larger power outages may be difficult and time consuming to restore. Such outages might thus lead to situations of non-functioning societies with devastating economical and humanitarian consequences. For this reason, this consortium has decided to concentrate its research on the systems for transmission and distribution of electric power. We anticipate that many of the results will be applicable to the protection of other critical infrastructures.

The operation and management of the electric power system depend on computerized SCADA systems. Keeping these systems secure and resilient to external attacks as well as to internal operational errors is thus vital for uninterrupted service. However, this is challenging since the control systems are extremely complex. Yet, the systems are operating under stringent requirements on availability and performance: If control and supervision are not done in real-time, the power network may come to a collapse.

SCADA systems include functions for remote collection of vast amounts of data from measurements placed in strategic points, e.g. power stations, in the geographically widely spread processes and for the remote control of process devices. Many SCADA systems include computerized models of the process, which enables simulation of alternatives process states and of optimization. Due to legal and environmental constraints, e.g. for building of new high voltage power lines or power stations, the primary process itself is difficult to expand which in its turn leads to higher and higher utilization of the existing transmission and generation resources. The process is, in other words, operated closer to its physical limits. Thus, the SCADA systems are

becoming increasingly critical for achieving availability and security of the supervised infrastructure.

The objective of the VIKING project is to develop, test and evaluate methodologies for the analysis, design and operation of resilient and secure industrial control systems for critical infrastructures. As mentioned, the focus is on power transmission and distribution networks. The project combines a holistic management perspective—in order to counteract sub-optimization in the design—with in-depth analysis and development of security solutions adapted to the specific requirements of networked control systems.

The VIKING project will study the whole SCADA system architecture from the measurement points in the process itself over the communication network to the central computer system, and as illustrated in Figure 1 by yellow exclamation marks potential targets for cyber attacks may be distributed over the whole system architecture.

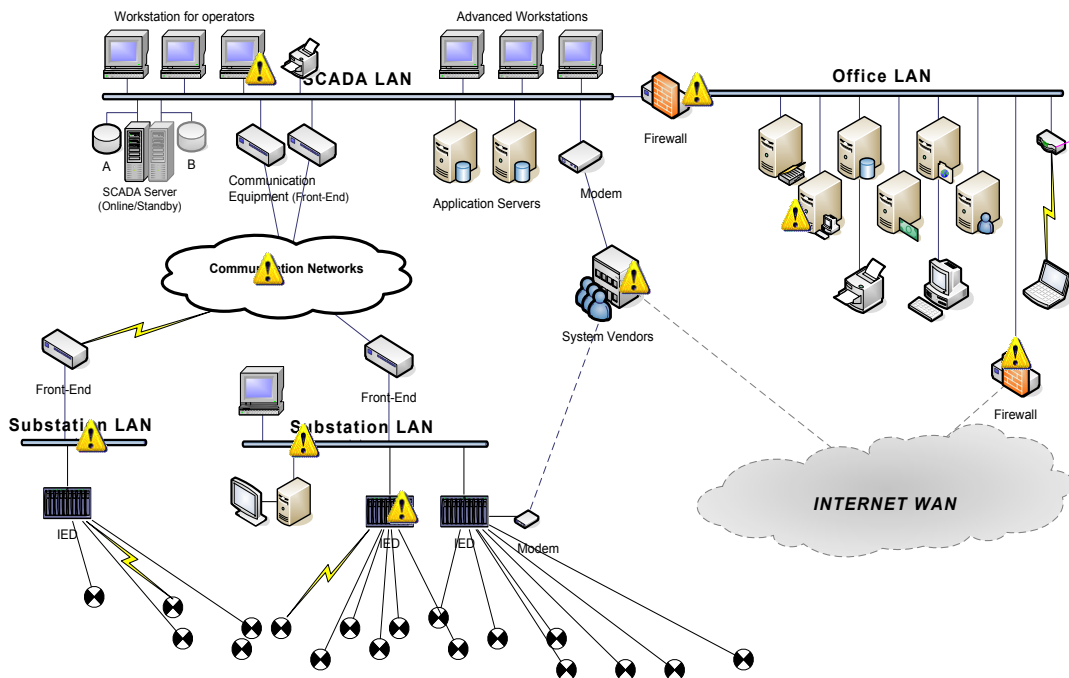


Figure 1 – A typical SCADA system architecture.

The project takes a model-based approach to investigating SCADA system vulnerabilities. Models are defined for the SCADA system, for the electrical process as well as of for the society that is dependent on the electricity supply, cf Figure 2. The society models are used to evaluate the economic consequences coming from disturbances in the electricity supply. The power system models are in turn used to evaluate the effects on the electricity supply by SCADA system misbehavior. Finally, SCADA system models are employed to assess the effect on SCADA system behavior by cyber attacks. Based on analysis performed on these models, VIKING will propose mitigation actions to be taken to decrease or to eliminate these risks. The results of the project will be evaluated and demonstrated on a test-bed that

can be configured to simulate the critical infrastructure of a power network and a range of attacks.

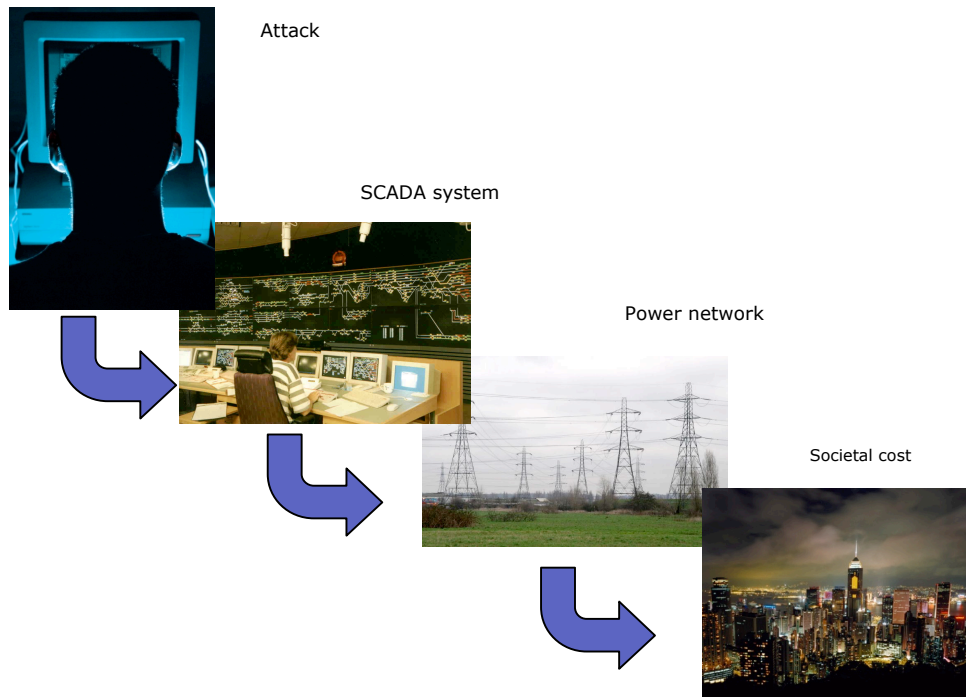


Figure 2 – Overall structure of VIKING.

Analyses made in different kinds of models are combined to provide a larger risk and cost based result of cyber attacks. With this approach the project hopes to achieve the following research results.

- Estimates of the security risk and consequences (in terms of monetary loss for the society)
- A method for calculating comparable, quantitative results for cyber security for different control system architecture solutions that is an environment for performing what-if analyses and help with identifying "weak spots" and how to mitigate them. *This deliverable, D2.2, is a central piece of this result.*
- Use of existing model based application as application level intrusion detection systems to detect manipulation of data
- Use of innovative and existing communication solutions to secure power system communication

1.3 Description of deliverable 3.1

This document is deliverable 3.1 of VIKING. It is a continuation of deliverable 2.2 that exemplifies how the Cyber Security Modeling Language (CySeMoL) can be used for analysing the likelihood that attacks targeting a system architecture are successful. The Purpose of the report is only to demonstrate the capability and intended use of CySeMoL, not to suggest optimal security solutions for different example architectures. Moreover, the practical feasibility or the costs related to the presented example architectures described herein is not considered.

1.3.1 Objectives

The objective of this work as a whole is to develop models for predicting and estimating the cyber security level of industrial control- and SCADA systems. The purpose of such an estimate is to support industrial decision makers of industrial control- and SCADA systems design and management to make cost-efficient design solutions. Moreover, as illustrated in Figure 2, in the VIKING project the output of this cyber security estimate could serve as input to for estimating the risks of malicious events in the cyber world in terms of events in the power system and eventually in a society. This whole chain is illustrated in delivery 5.3a, not in this report.

1.3.2 A brief overview of the Cyber Security Modelling Language (CySeMoL)

In summary, CySeMoL is a language (or meta model) in which system architectures are described. The language contains general-purpose entities such as *services*, *data flows*, *operating systems*, as well as security specific entities such as *intrusion detection systems*, *firewalls* and *patch management processes*. The language also defines how these concepts can be related to each other as well as some important properties (from a security perspective) of the entities, such as for instance if an operating system is using non-executable memory or if services have known vulnerabilities. With the language, users of CySeMoL are able to describe their system architectures. In principle they will draw figures looking like the architecture in Figure 1. In addition to this purely descriptive part of CySeMoL, a mechanism for calculating value that roughly could be considered a security index is also included in the language. In essence, this mechanism is an attack/defense graph, which describes how different attacks and attack steps could be performed in the system architecture and its different components. So, depending on the exact configuration of the architecture, different attack processes will be possible for an attacker to accomplish. For all those attack processes, CySeMoL provides numerical estimates for how likely it is that all the different attack steps are possible to accomplish. These estimates are given as conditional probabilities (specified in Bayesian networks). As an example, the table below illustrates numerical figures of conditional probabilities of how likely it is to succeed with a semantic (rather than brute force) denial of service attack on a computer given the following two parameters of the system architecture, i.e. if the targeted host has a known vulnerability and if the attacker has access credentials to the targeted host.

Software Vulnerability Present	Attacker has Access Credentials	Expected Likelihood of Attack Success
Yes	Yes	0.72
Yes	No	0.53
No	Yes	0.60
No	No	0.38

CySeMoL then combines a great number of such conditional probabilities of attack steps and associated countermeasures into an aggregated expected likelihood of success for the whole attack

process. This aggregated value is then finally what was labeled as the security index above. A more detailed discussion about these values is provided in the next section.

Data for the CySeMoL calculation mechanism

Of course, at the core of the CySeMoL lie the conditional probabilities used for the calculations. These figures have to a large extent been collected by asking security experts in surveys on their opinions of the impact of different countermeasures on different attacks, such as the DoS example above. For all questions the explicitly stated assumption to the respondent has been that the attacker is a professional penetration tester with one week of preparation. Some of the figures are also deterministically derived, and some have been derived from previously published studies. In total four surveys has been conducted on various parts of the CySeMoL with answers from 165 respondents as maximum and a handful of respondents as minimum. In order to identify qualified respondents (identifying which experts that really are experts) Cooke's classical method has been used. This method essentially weight different respondents depending on how good they are at answering some test questions relevant to the area of the survey questions (that the CySeMoL developers have known the answers to). This means that only a few of the for instance 165 respondents mentioned above performed good enough to be called experts. All answers, i.e. conditional probabilities, have been collected also including the respondents' opinion on the uncertainty of the answer (expressed as a three point estimation). For instance, for the first estimate in the table above the average answer (of the respondents selected as the "true experts") is 72% as indicated in the Table. However, there is a 5% chance that the value is below 32% (again, on average) and a 95% chance that the value is below 95% (on average). Another way of expressing it would be that there is a 90% chance that the attack success value is between 32% and 95%. As we can see from this example the figure 72% here is acquainted with a large share of uncertainty. However, in the calculations made in this report only the expected mean value is used, i.e. 72% in the example above.

Intended usage of CySeMoL

The intended usage of the CySeMoL is to support security analyses of SCADA and control system architectures. It should support users that are not necessarily security experts themselves. If the user provides a system architecture, the CySeMoL can provide a security estimate in terms of attack probabilities. So, by analysing different architectures and different attack processes the user can get a better understanding of available weak spots in the architecture. In addition, it also provides a clue on how effective different mitigation strategies (probably) are. As described above, the figures provided are often acquainted with quite big uncertainty. This imposes that the calculated percentage value figure should be treated with care. The results should be seen as a support for reasoning about different alternative scenarios or mitigations. On average a scenario with attack success probability of 10% is more resilient towards the analysed attack process than one

with 30%, even included the uncertainties (that in general are the same magnitude for scenarios). Essentially the user needs to define two things: 1) the system architecture (including a number of properties), and 2) which targets they would like to analyse as well as starting points for the attacks. I.e., CySeMoL delivers results for (the most probable attack process between) pairs of a single starting point and a single target. But, in order to get a more complete and holistic understanding of the whole architecture many such pairs needs to be considered. Again, comparing scenarios without analysing the complete set of potential pairs will provide an indication of their relative security.

Since the CySeMoL is quite large and complex, it is extremely time consuming to do the calculations by hand. In this report all examples used the Enterprise Architecture Analysis Tool (EAAT)¹ to calculate the results and visualize the models. The EAAT is not developed within the VIKING project.

Notes on the objectivity of the CySeMoL.

The correctness of results of the CySeMoL is to a large extent determined by the empirically collected conditional probabilities. A wide range of experts, both academic and practitioners, have been involved in the validation process. In the survey, Cooke's classical method has been used which is the state of the art method for expert knowledge elicitation. The method does not accept anyone as an "expert" but test all respondents with a test questions. Before, the surveys were sent out the parameters that were included in the survey was also validated with respect to their importance in a qualitative way by cyber security professionals. Both of these actions ensure the correctness in the details of the CySeMoL. Furthermore, the aggregated results (the attack path estimations) of the CySeMoL has also been validated through a Turing test. In the Turing test, both cyber security professionals and the CySeMoL were asked to assess a number of scenarios. All these assessments were then given to another group of cyber security professionals to judge if they believed that the results seemed reasonable (without knowing who produced the assessment). CySeMoL performed equally well in comparison to the cyber security professionals (it was neither best nor worst and everyone was performing in the same "magnitude").

Nevertheless, it is important to realize that the figures are "soft facts" rather than "hard" statistical data. This is due to the fact that no statistical data is available for the scope of CySeMoL and it is not feasible, given the limited time of the VIKING project, to produce a sufficient amount of new data with better quality than the surveys. This means that we have ended up with much uncertainty in the results. This uncertainty is likely to reflect both that more detailed information about the system architecture is needed in order to make better predictions, but also that the answers to the questions that CySeMoL addresses are simply not known. (It could of course also be that the wrong people were asked. But this has at least carefully been tried to avoid by using a well-known scientific methods.) In many aspects CySeMoL needs to be enhanced and complemented in future

¹ KTH - Royal Institute of Technology, Enterprise Architecture Analysis Tool
homepage, www.ics.kth.se/eat

works. By removing the following currently used assumptions CySeMoL could be improved:

- The attacker has been simplified to one type of attacker with a lot of variance related to it. So how well these figures represent advance persistent threats or script kiddies needs to be further studied.
- The focus of CySeMoL is on addressing the availability and integrity objectives of the system rather than protecting the confidentiality (due to the SCADA and control system focus of the VIKING project). Thus, in the current version, countermeasures related to deterrence and recovery is not yet considered.
- The focus of CySeMoL is mainly on technical parts of security. Thus, other type of cyber security attacks that involves non-technical aspect, e.g. social engineering, information security governance and physical attacks are treated in a very rudimentary in this version.
- The conditional probabilities that produce the CySeMoL results needs to be updated from time to time as knowledge of both attackers and defenders change over time. CySeMoL reflects a snapshot of knowledge dated approximately to 2010.

Despite the abovementioned limitations, we believe that CySeMoL provides significant contribution in providing a systematic approach for analyzing the cyber security of SCADA and control systems. From an academic point of view this delivery should not be considered the end of CySeMoL but rather the beginning.

1.4 Outline of the document

This chapter, chapter 1, has described the background of VIKING and a summary of CySeMoL. In chapter 2 a fictitious, but realistic, example is elaborated. Firstly, the system architecture is described and secondly some potential attack paths are analysed including some mitigations for these attacks. In chapter 3, three real case studies are described. Due to confidentiality reasons no models or results are presented, instead the case studies are described in a qualitative manner and lessons learned, from a CySeMoL perspective, are reported.

2 An example SCADA architecture

This chapter provides an instantiated model of a SCADA system architecture. It is fictitious in the sense that it is not depicting any specific real-world system architecture. However, the example is intended to be realistic so that it can serve as a representative reference case. Even though realistic in terms of architecture and components, the example is not realistically complete. We only model a limited number of systems and services to make the example readable, graspable and possible to fit into a report. All specific brand names or specific products in the examples are just serving as examples of a larger set of products, and the results calculated on properties of the larger set of components (as defined by CySeMoL). The purpose of the chapter is to illustrate CySeMoL from a user's point of view; what kinds of models are to be modeled by the user as well as what kind of analysis results that can be delivered in return by CySeMoL. The complete set of parameters for the examples are found in the report Appendix.

2.1 Model

A high-level viewpoint of the example model can be seen in Figure 3. The depicted power utility comprises five different network zones (six when viewing the whole of Internet as one abstract zone). Each network zone has a specific zone management process and each zone is separated into different physical zones. These zones are: *Process*, *SCADA*, *Internal DMZ*, *Office* and *External DMZ* (and *Internet*). Each zone is separated by a firewall. Also, a network intrusion detection system (*Cisco IDS*) is present between the *Internal DMZ* and *Office*.

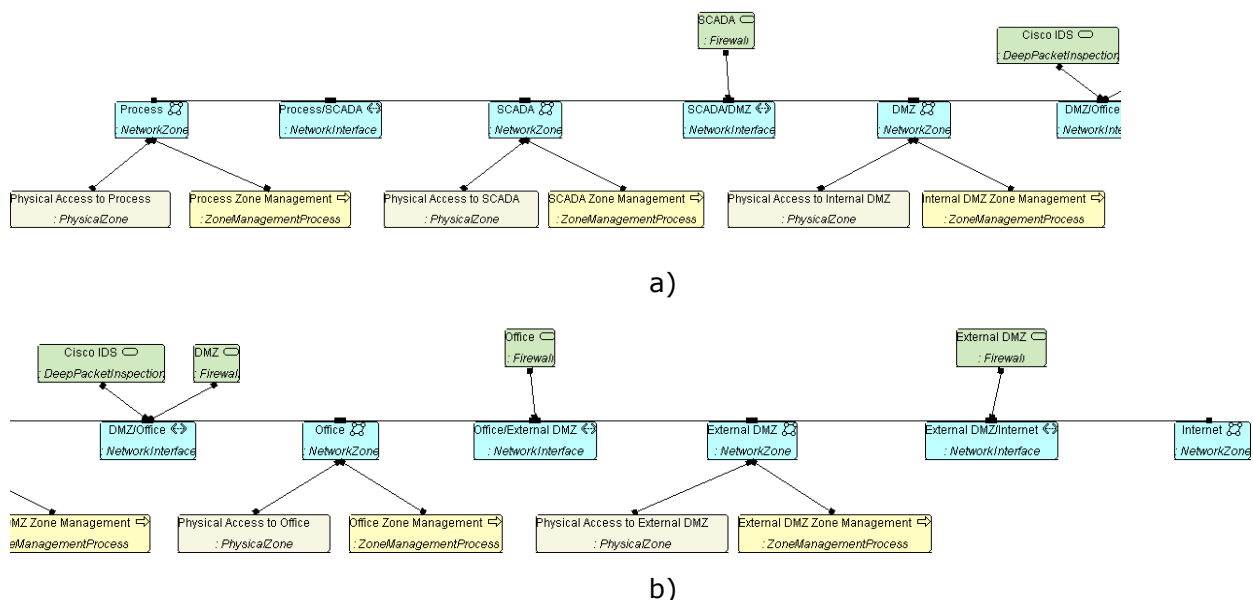


Figure 3. High-level description of network architecture of the example, divided into parts a), leftmost part, and b), rightmost part.

2.1.1 External DMZ network

An overview of the *External DMZ* can be seen in Figure 4. Two systems are modelled in this zone: a *Mail Server* and a *Web Server*. Both nodes are run on the *Windows Server 2008 (R2)* operating system (OS), with

only a single user with credentials to the systems. Only the most critical services are modelled (*SMTP*, *HTTP* and *HTTPS*). All of these services are available through functionality in the *Windows Server 2008* OS. (To simplify the model, we have chosen not to model the mail and web server *application* explicitly, only the operating systems that hosts these applications. However, if needed applications can be modelled separately.)

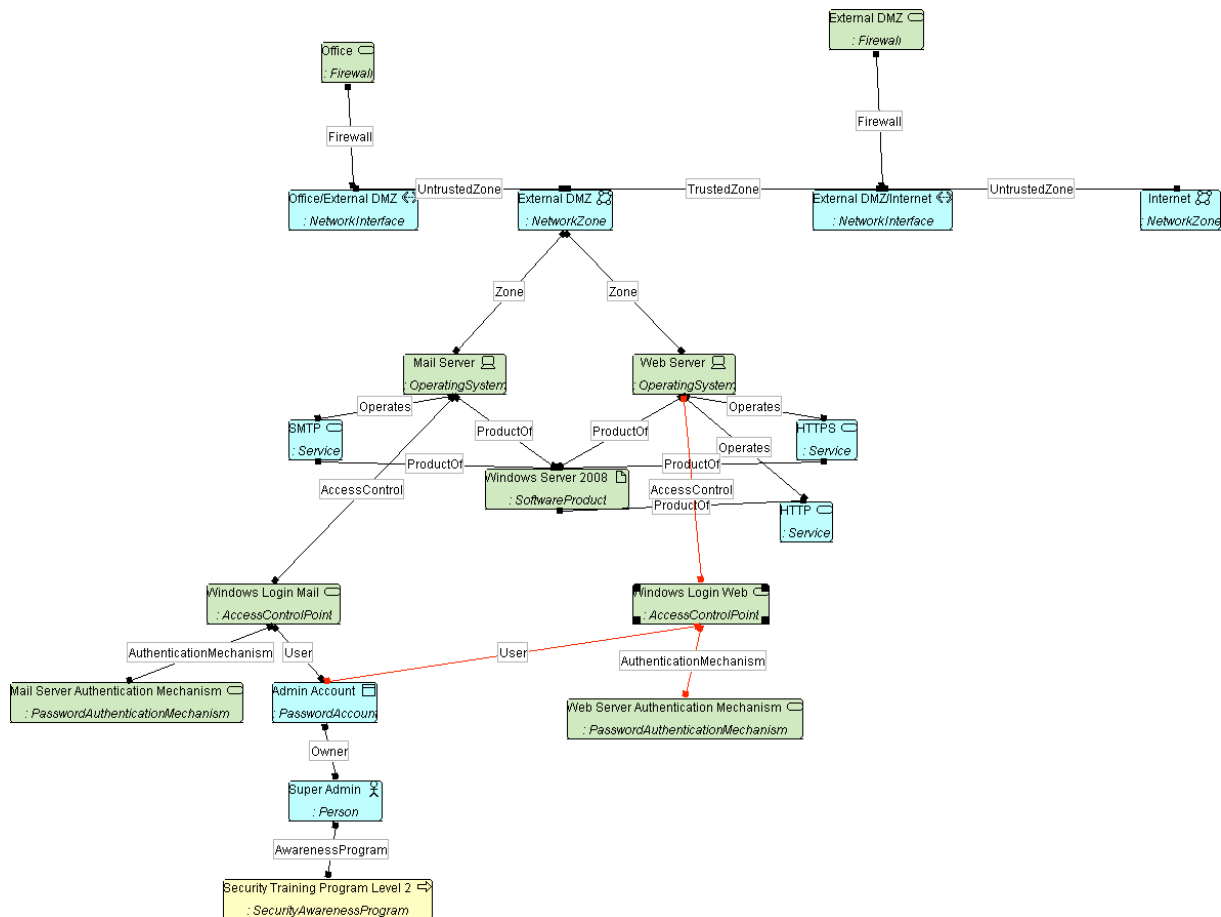


Figure 4. Systems in the external DMZ network.

2.1.2 Office network

Figure 5 depicts an overview of the *Office* network zone. This zone includes a single modelled *Office Client* (as all office user clients have the same properties there is often no need to model more) and an *Office Domain Controller* to keep track of all connected nodes and accounts. These systems employ *Windows 7* and *Windows Server 2008* (R2). The domain controller also has RDP (Remote Desktop Protocol) enabled. Three types of users are modelled: three regular office *Users*, one *SCADA engineer* and one *Super Admin*. The *Super Admin* have credentials to all systems in the *Office*, while the *SCADA Engineer* and *Users* only have access to the *Office Clients*. Also, these users have undergone two different security training programs. Finally, the *Office Client* has two application clients: a *Browser* (Microsoft Internet Explorer 8) and an *Email Client* (Outlook 2007). There is no policy restricting browsing the *Internet*.

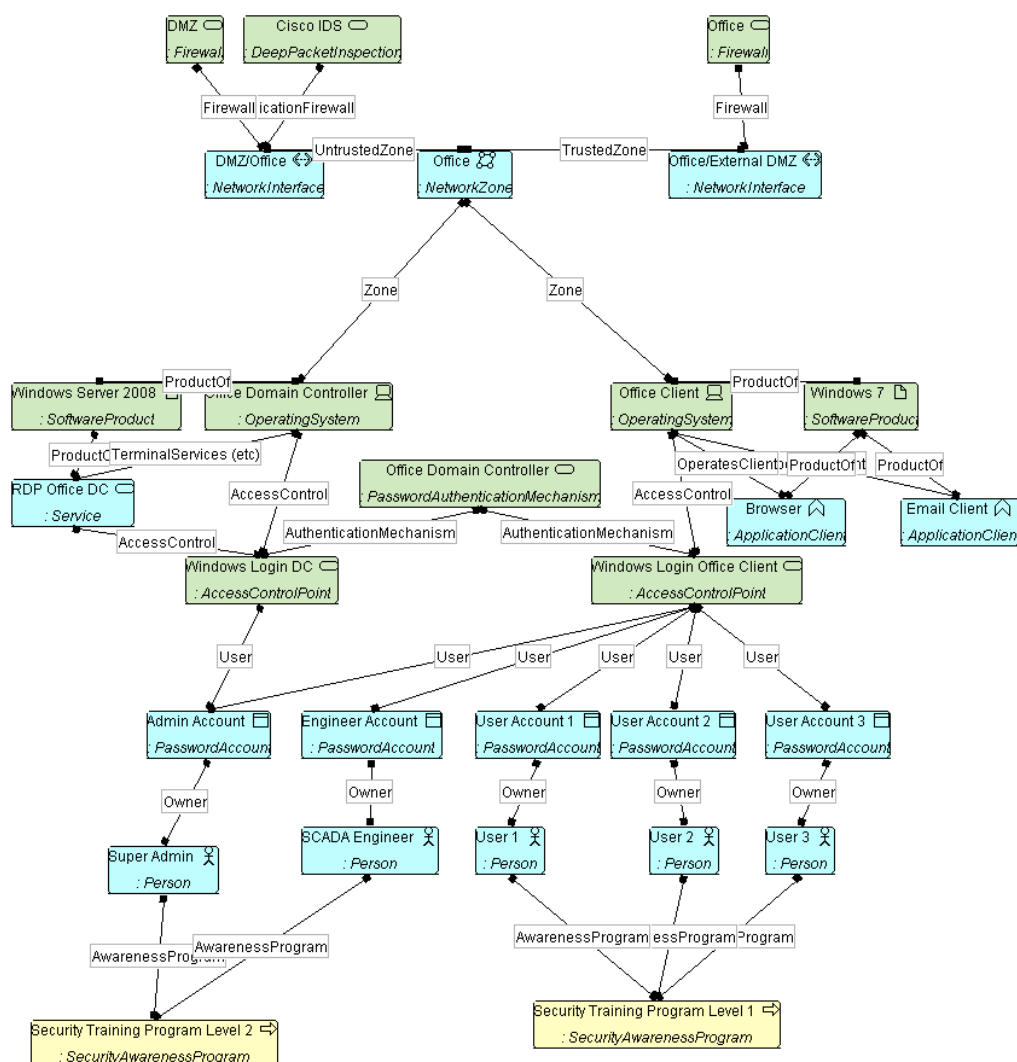


Figure 5. Systems in the office network.

2.1.3 Internal DMZ network

The Internal DMZ, cf. Figure 6, includes two systems: a database server named *Historian* and a *Terminal Server*. The purpose of the *Historian* is to store and provide data to users in the *SCADA* and *Office* network zones. The purpose of the *Terminal Server* is to allow *SCADA Engineers* in the *Office* zone to access resources in the *SCADA* zone (the firewalls prohibits all information flowing between *SCADA* and *Office*, as such the operators need to “jump” using this node. Both systems employ the OS Windows Server 2003. The zone has two users: *SCADA Engineer* and *Super Admin*. Notably, *SCADA Engineer* only has access to the *Terminal Server's Terminal Service*.

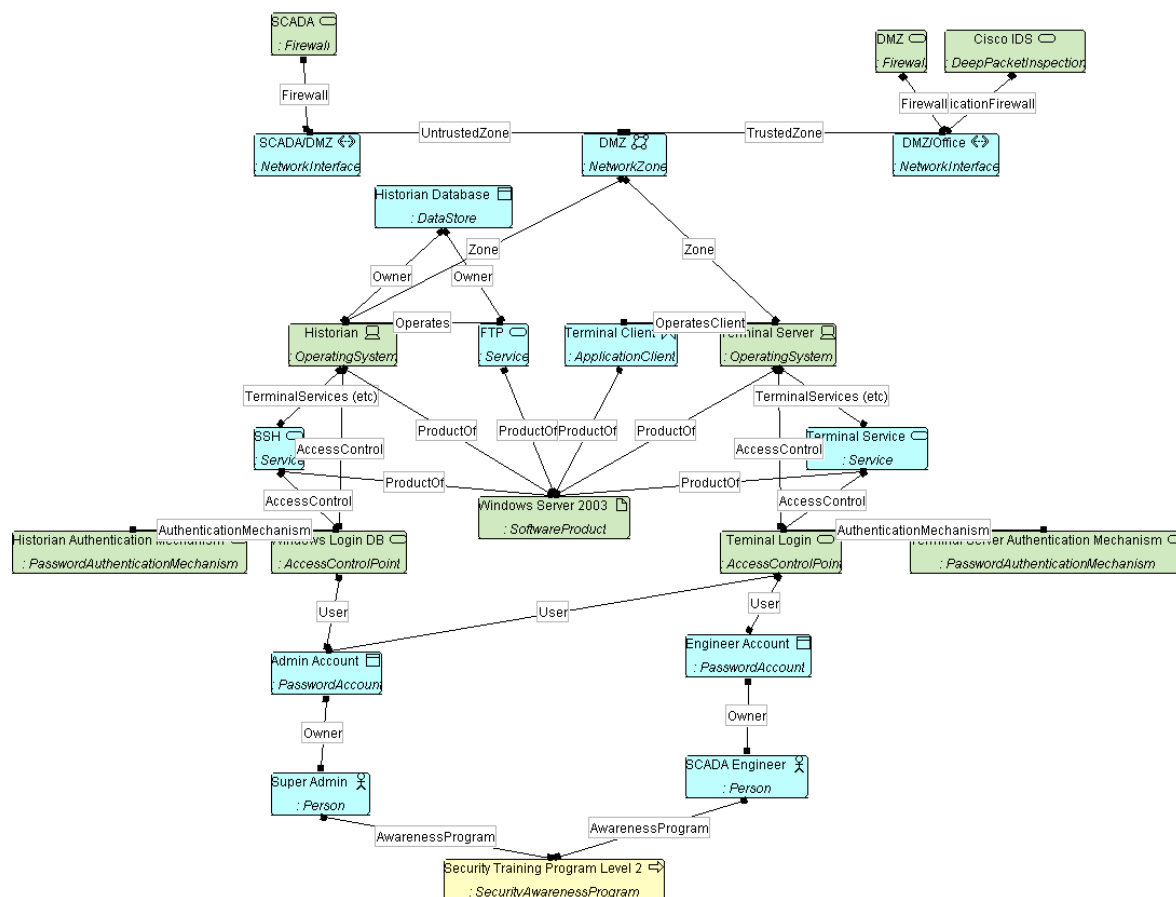


Figure 6. Systems in the internal DMZ network.

2.1.4 SCADA network

The SCADA network zone is illustrated in Figure 7. This zone contains four different systems: *PCU Communication*, *HMI*, *SCADA Domain Controller* and *Application Server*. The persons *SCADA Engineer* and *Super Admin* have credentials to operate the systems. Notably however, the SCADA Engineer has access to resources employed on the OSs that the *Super Admin* does not (e.g. the *RTU Comm Software* on the *PCU Communication*). The OSs *Windows XP*, *Windows Server 2003* and *Red Hat 4.0* are employed on the nodes in the zone. There is also a specific SCADA product; the *SCADA Network Manager*.

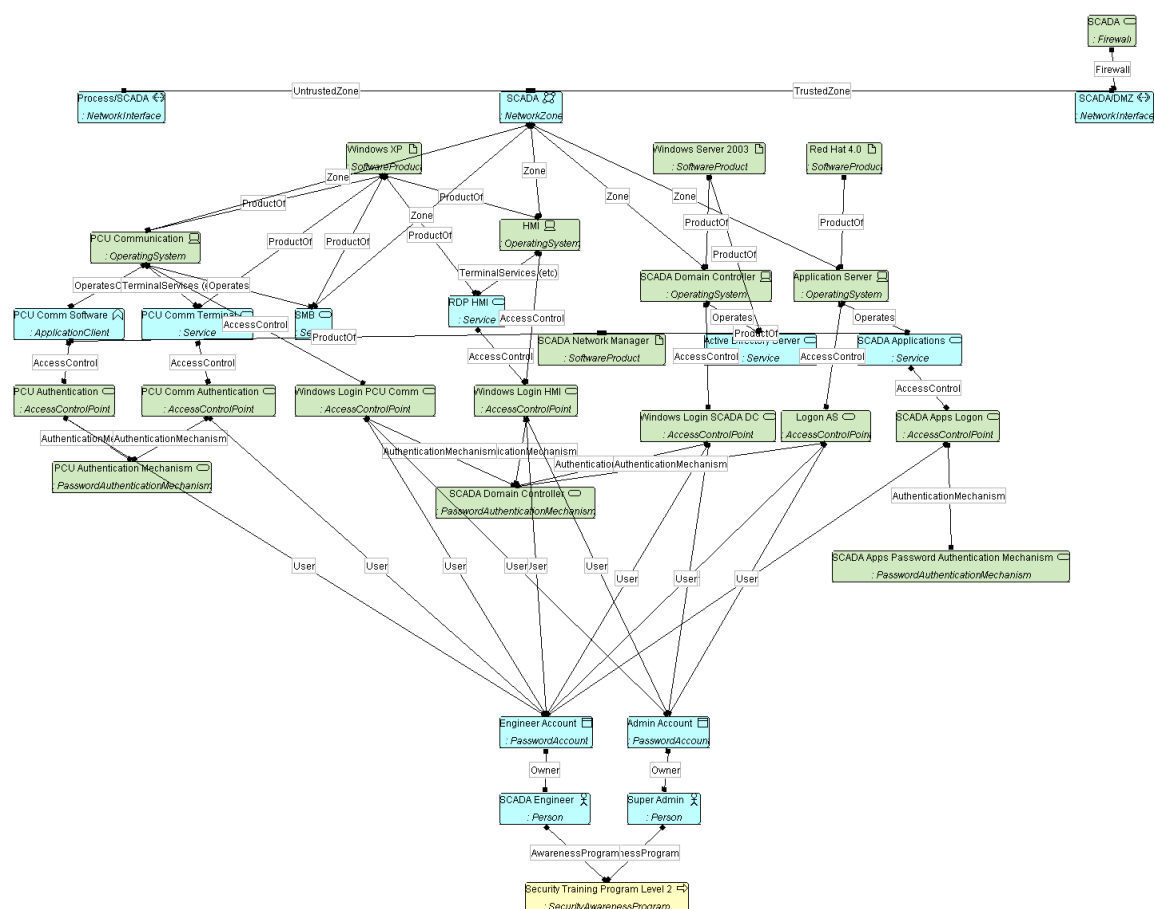


Figure 7. Systems in the SCADA network.

2.1.5 Process

The Process network zone can be seen in Figure 8. This zone comprises only one type of system: an *RTU* (with firmware loosely specified as *RTU Product*).

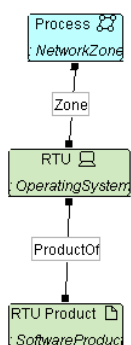


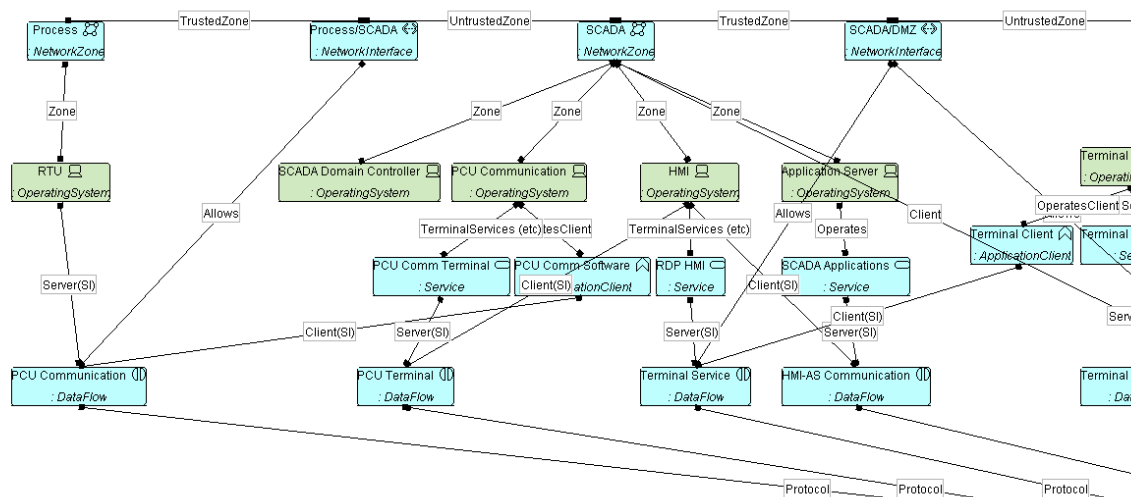
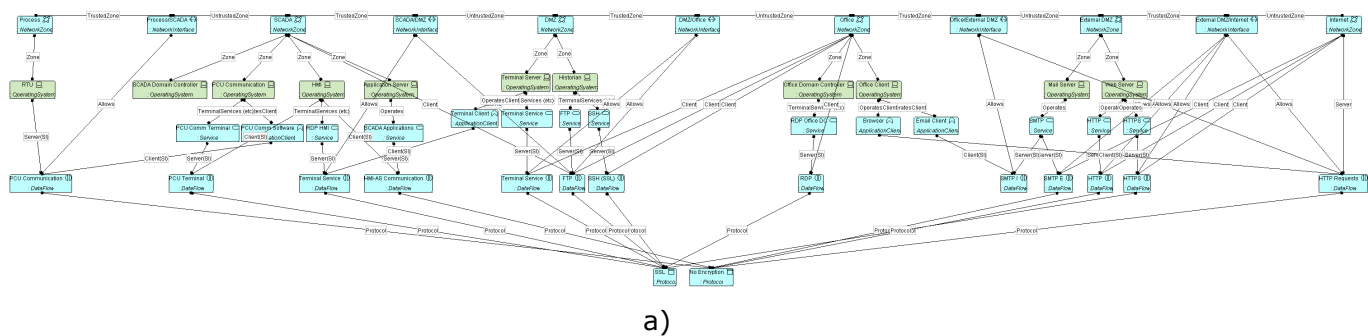
Figure 8. Systems in the process network.

2.1.6 Data flows

Finally, the viewpoint depicted in Figure 9 shows all the data flows inside and between the different network zones of the model.

The contents of the data flows are either obfuscated (SSL) or visible (No *Encryption*). The rest of this section depicts the data flows in the example model, beginning from the *Process* network zone and working towards the *Internet*.

The *RTU (Process)* is connected to the *PCU Software (SCADA)*. The *PCU Comm Service (SCADA)* can be accessed from the *HMI (SCADA)*. The *SCADA Applications (SCADA)* can be accessed from the *HMI (SCADA)*. The *HMI (SCADA)* can be accessed from the *Office* network zone through the “jump” using the *Terminal Server* in the *Internal DMZ*. *FTP* on the *Historian (Internal DMZ)* can be accessed from *Internal DMZ*, *SCADA* and *Office*. *SSH* on the *Historian (Internal DMZ)* can be accessed from the *Internal DMZ* and *Office*. The *Office Domain Controller* can be accessed from anywhere on the *Office* network zone. The *Browser* of the *Office Client* can access content on the *Internet*. The *Mail Client* of the *Office Client* can access *SMTP* on the *Mail Server* on the *External DMZ* for outgoing mail.



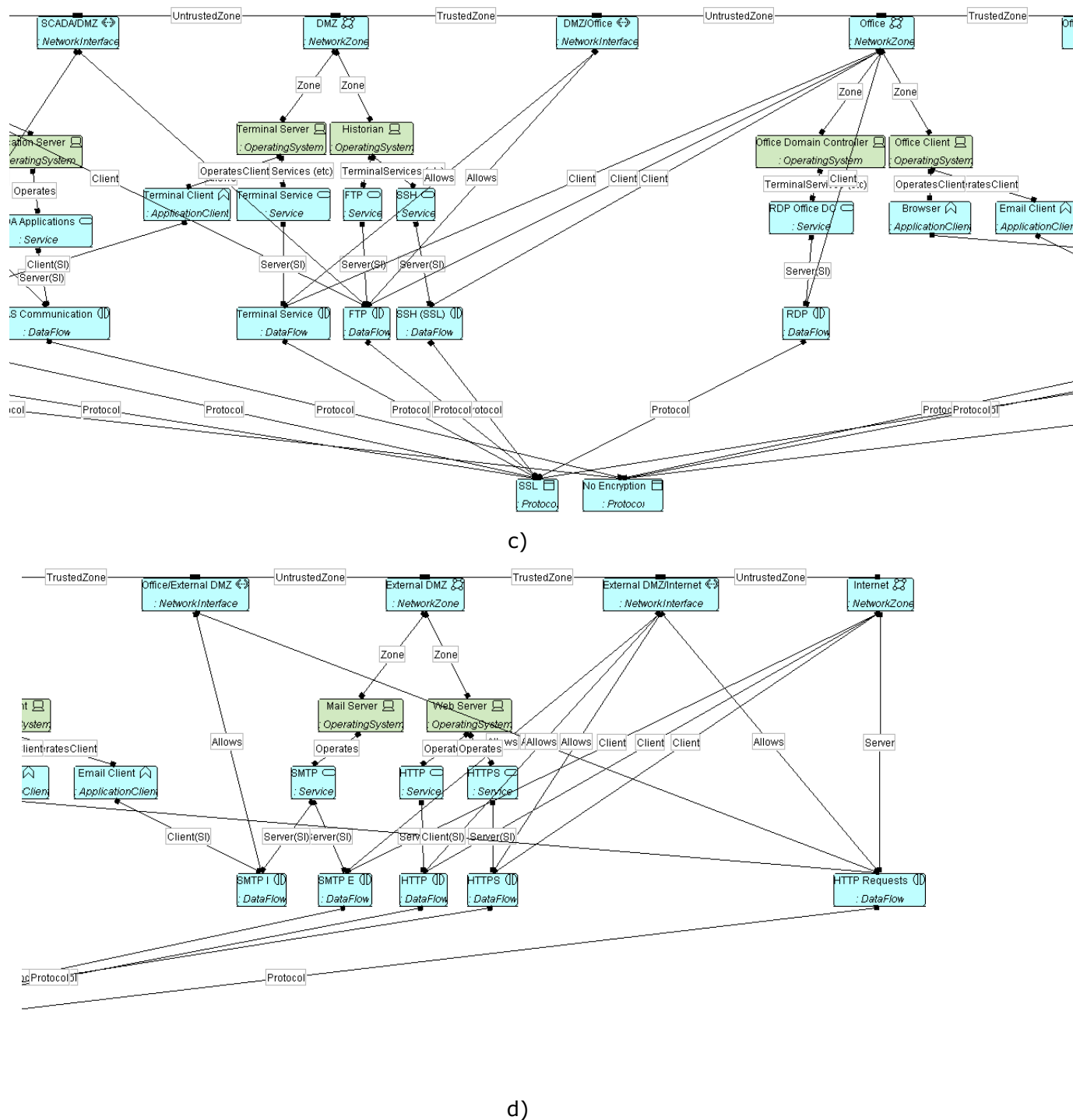


Figure 9. Data flows between and inside the networks. a) Model as a whole. b) Leftmost part of the model. c) Middle part of the model. d) Rightmost part of the model.

2.2 Attack analysis

In this section it is illustrated how different kinds of attacks can be analysed in the model by CySeMoL. For each attack the target node as well as entry point needs to be specified in the model, by the user. Again, calculations are done according to the Bayesian network

populated with empirical data that are reported in D2.2. The figures state probability estimates of the risk that a professional penetration tester would succeed in with the attack after one week of preparation. For each entry point – target combination there of course exists a large number of potential attack paths. In our examples we have chosen to only talk about the attack path that CySeMoL has identified as the most probable. A further note is that all specific brand names or specific products in the models are just serving as examples of a larger set of products when it comes to the attack success estimates. CySeMoL does not differentiate between different products in general, it only cares about some specific properties. For an operating system CySeMoL looks at for instance if it is using non-executable memory, address space layout randomization, and if it is fully patched. It does not have predefined estimate of e.g. Windows XP. Moreover when looking at mitigations the examples only provide the results from CySeMoL. The examples do not claim to show the most efficient mitigation and we have neither considered how difficult, feasible or costly the countermeasures are. The purpose of these examples is to illustrate CySeMoL from a user's point of view; what kinds of models are to be modeled by the user as well as what kind of analysis results that can be delivered in return by CySeMoL. Below, two attacks are elaborated.

2.2.1 Compromise PCU Communication as an outsider

This scenario involves an attacker in the form of an outsider without any credentials to any parts of the power utility infrastructure. The attacker places several malicious USB drives outside of the power utility. One of these USB drives is picked up by a *SCADA Engineer* who brings it into the SCADA network zone and connects it to the *HMI* system. The HMI station is thus the starting point for the attack analysis. The chosen target is the PCU communication station. In short the identified "weakest path" between the two stations can be described as the malware on the USB drive activates without alarming the operator and compromises the system with low-level privileges. From this point the malware probes the network zone for other vulnerable Windows machines. It finds the *PCU Communication* and exploits a known vulnerability in SMB.

2.2.1.1 Base scenario

This scenario, identified and calculated by CySeMoL, is illustrated in Figure 10. The attacker gains remote access to the HMI through a payload on an inserted USB stick (step 0, step 1). The HMI has an address on the SCADA network zone (step 2). The attacker connects to the Shared Message Block (SMB) service on the PCU Communication (step 3). As the attacker is able to connect to the PCU Communication and the SMB service (s)he can probe it to determine what type of product (and version) it is (i.e. that it is SMB and Windows XP) (step 4). The attacker uses this knowledge to identify a vulnerability in the software (in this case a publicly known vulnerability which can be patched) (step 5). As there is no patch mitigating this vulnerability deployed the attacker has found a functioning high severity exploit (step 6). The attacker executes this vulnerability in an attempt to run arbitrary code on the PCU Communication OS (step 7). After successful code injection the attacker is able to open a back door to the PCU

Communication and control it with low level privileges. The probability of this attack being successful is 24%.

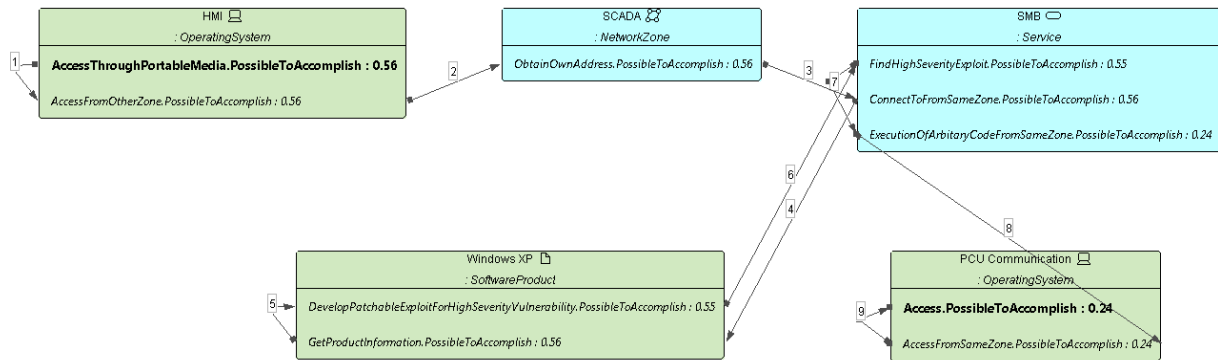


Figure 10. Visual description of the calculated weakest attack path.

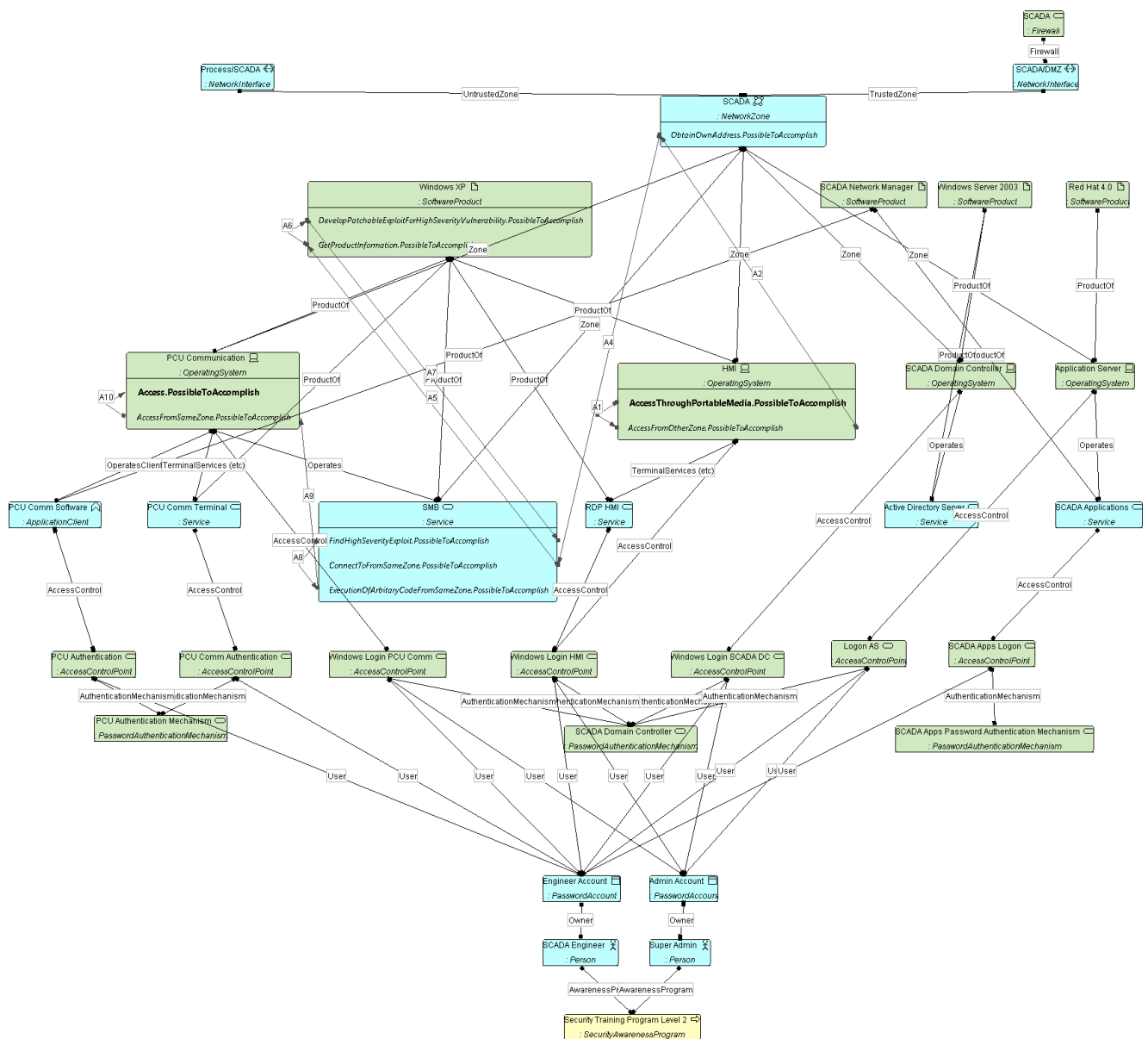


Figure 11. Visualisation of attack path in SCADA network view. (This is the same view as the one in Figure 7)

The most obvious mitigation strategy is to create a policy that prevents users from using USB sticks in the SCADA network zone. Another mitigation strategy is to educate personnel on IT security risks. Assuming that it is not possible to prevent the compromise of the HMI through the USB carried payload there are still things that can be done to decrease the likelihood of successful attack:

2.2.1.2 Mitigation example 1: Software patching

One mitigation technique is to update the software of the systems within the SCADA network zone. Assuming that the chosen mitigation strategy would be to update software deployed on the PCU Communication to their latest version (and no product has reached the end of its lifecycle, such as e.g. Windows 2000) the attacker would be forced to find a new vulnerability in SMB for the latest version of Windows XP (i.e. a 0-day) Assuming that there is no such vulnerability publicly available the attacker needs to discover a new one through either black box or white box testing (or a combination of them). Given such a scenario, the probability of successful compromise would degrade to 5% (see Figure 12). The decrease of probability compared to the original scenario is evident in step 5 (DevelopUnpatchableExploitForHighSeverityVulnerability.PossibleToAccomplish).

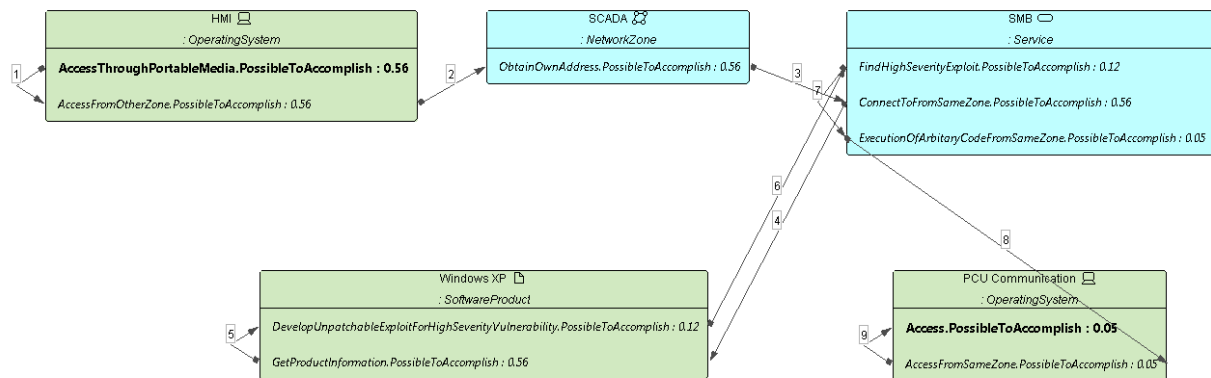


Figure 12. Mitigation through complete software updates

2.2.1.3 Mitigation example 2: An IPS

Another type of mitigation could be to apply a network-layer deep packet inspection filter to monitor the traffic directed for the PCU Communication. This mitigation would decrease the chance of successful attack to 17% (see Figure 13 and Figure 14). The added deep packet inspection filter influences the likelihood of successfully executing arbitrary code on the OS through the service (ExecutionOfArbitraryCodeFromSameZone.PossibleToAccomplish).

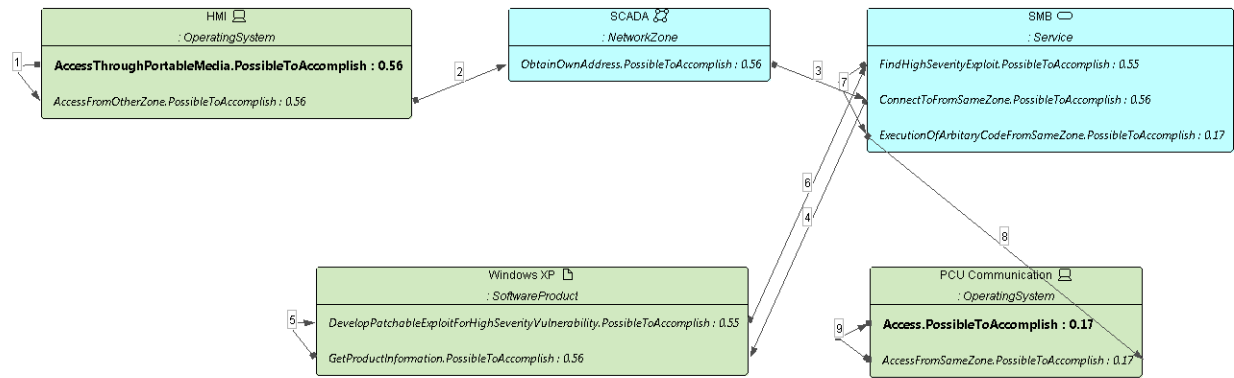


Figure 13. Mitigation through a network-layer deep packet inspection filter

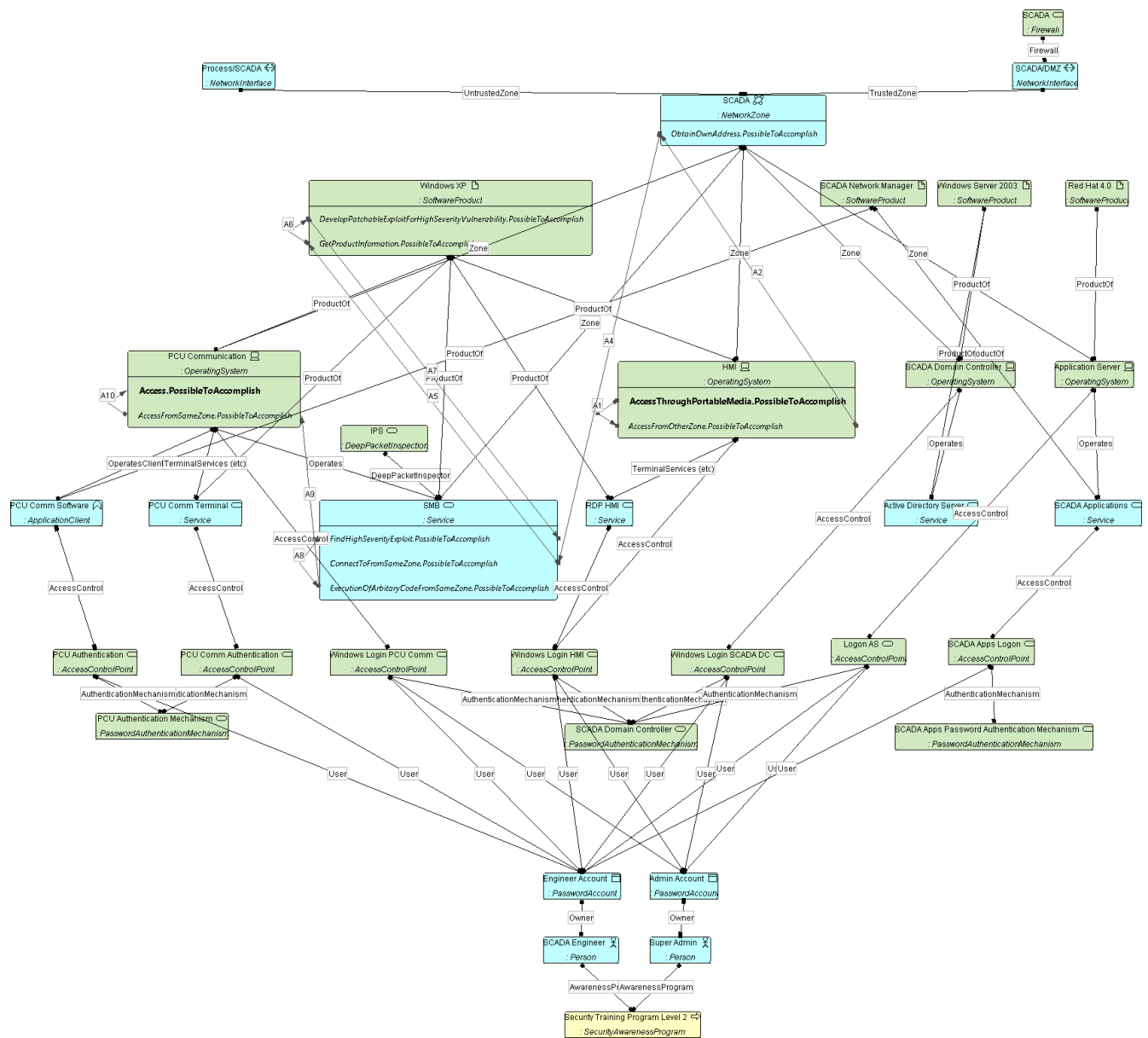


Figure 14. Attack with IPS as mitigation. Illustrated in SCADA network View. Note the IPS now added to the SMB service.

2.2.1.4 Summary of scenario 1

A summary of the attack scenario and its mitigation suggestions can be seen in Table 2. It seems that the best mitigation option for this threat is to keep all system software fully updated. There are naturally many delimitations in these mitigation suggestions which can be addressed. For example, CySeMoL does not include any analysis regarding costs and what solutions which are feasible in practice. To update software in a SCADA environment can for be a difficult task for several reasons such as: i) a SCADA network zone is generally fully disconnected from the internet, ii) SCADA software is not always interoperable with updated versions of some products.

Table 2. Summary of scenario 1

Scenario	Probability
Base scenario	24%
Mitigation 1: Software patching	5%
Mitigation 2: An intrusion prevention system	17%

2.2.2 Write to the Historian database as an outsider

This scenario involves an attacker in the form of an outsider without any credentials to any parts of the power utility infrastructure. The starting point of the analysis is a malicious server on the Internet and the target is the Historian Database located in the internal DMZ. In short the weakest attack path identified by CySeMoL could be described as follows. The attacker sets up a malicious server on the *Internet* and fools a *SCADA Engineer* into accessing its content through a *Browser* (Internet Explorer). The attacker then exploits a vulnerability in the Browser to allow execution of malicious code. The malicious payload is specified to give the attacker access to the *Office Client* OS and install malicious software. As such, it allows the attacker to discover any hosts reachable from the Office Client. The attacker probes the surrounding network and finds that the *FTP* server service on the *Historian* system is reachable. The attacker uses the default password on the FTP to be able to read and write to the database.

2.2.2.1 Base scenario

The base scenario is illustrated in Figure 15. The attacker has set up a *Malicious Server* on the *Internet* (step 0, source). The attacker lures the unsuspecting *SCADA Engineer* to browse the content of the *Malicious Server* (step 1, 2). The exploit kit on the Malicious Server probes the *Browser* and its resources (e.g. Java RE) to determine product types and versions (step 3). The root kit finds a publicly available vulnerability that there exists a patch in the *Browser* (step 4). As there is an automatic patch management process deployed within the Office network zone there is a 79% chance that the Browser and its resources are fully patched. Nevertheless, there is as such still 21% chance that the exploit is applicable (step 5). The attacker tries to utilise the exploit to execute arbitrary code on the Office Client OS

through the Browser (step 6). The attacker manages to gain access to the Office Client OS (step 7, 8). The attacker uses the Office Client to produce a request to the FTP server service (step 9). The attacker is as such able to connect to the FTP (step 10). The attacker realizes that there is an authentication mechanism for the FTP service – the user is required to input his or her credentials to access the contents of the server (step 11). The attacker guesses the password to the authentication mechanism, much due to it being a default password for the service (step 12). The authentication mechanism is bypassed by the attacker (step 13, 14). The attacker as such has access to the FTP service and its resources (step 15, 16). The attacker uses the FTP service to write to the Historian Database as a legitimate user (step 17). The probability of this scenario being successful is 12%.

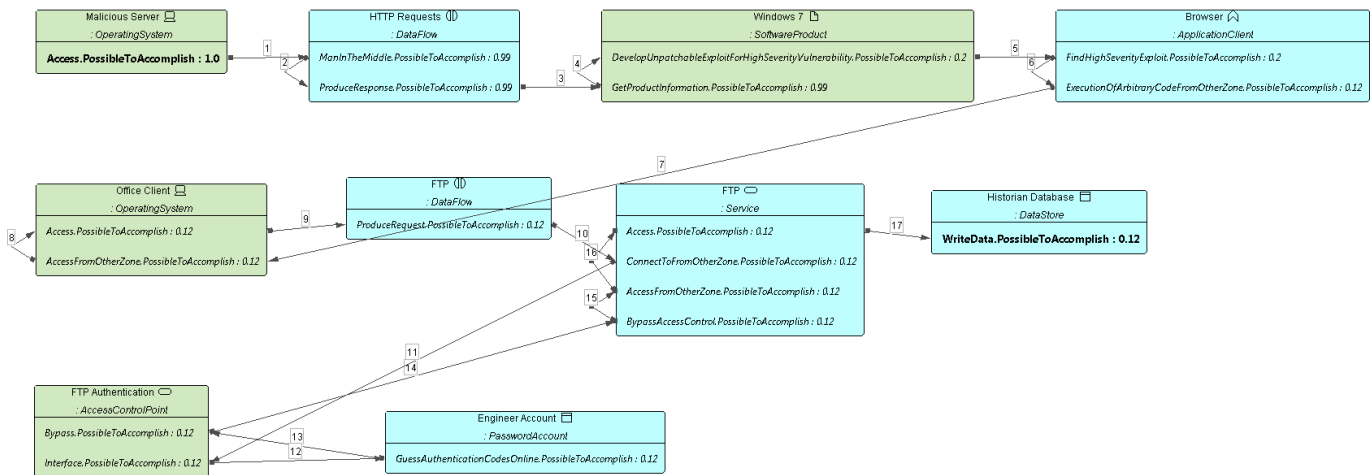


Figure 15. Visual description of the calculated weakest attack path.

2.2.2.2 Mitigation example 1: Implement password policies

This mitigation example can be seen in Figure 16. One very simple mitigation option is to update the password policies of the Historian authentication mechanism; to implement a back-off technique and a proactive password checker. This scenario would make it significantly more difficult to guess the password online, even if the attacker has access to applicable dictionaries. The most significant difference in the attack scenario is that the attacker would try to social engineer the credentials of the SCADA Engineer, rather than try to guess it (step 12). The likelihood of this scenario being successful is 9%.

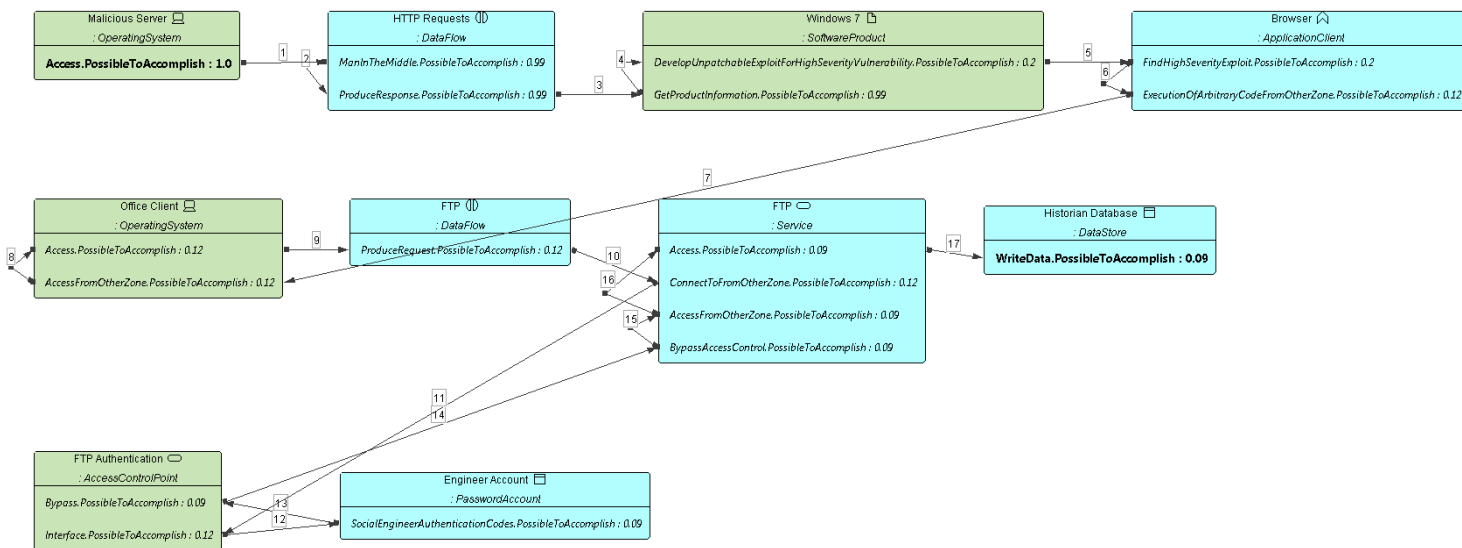


Figure 16. Mitigation through password policies.

2.2.2.3 Mitigation example 2: Security awareness training

Expanding on the mitigation suggestion above; one could also improve the security awareness program and thus make personnel (e.g. the SCADA Engineer) more resistant to social engineering-attacks (cf. Figure 17). If the SCADA Engineer would be security trained and both a back-off technique and a proactive password checker were used then the likelihood of successful attack would decrease to 3% (notice the decreased probability at step 12).

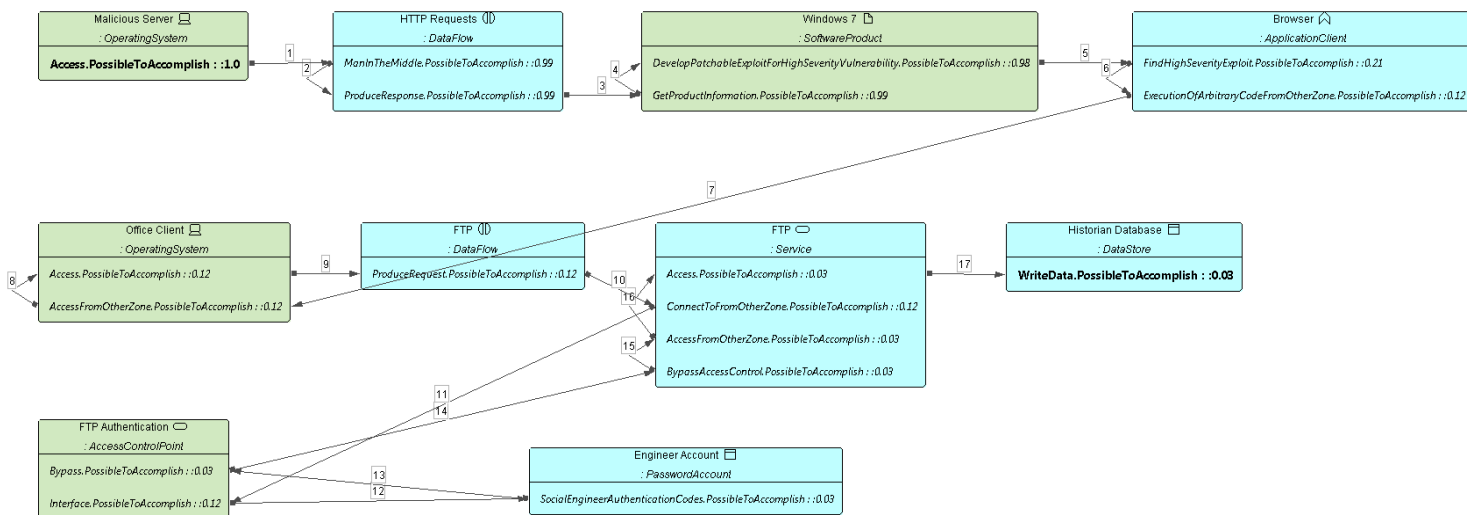


Figure 17. Mitigation through security awareness training.

2.2.2.4 Mitigation example 3: Software patching

A third mitigation option is to fully update the Browser and its resources, and as such delimit the possibility to use older public exploits. Given such as scenario the attacker would be forced to discover a vulnerability and exploit for which there is not patch available as of yet. This is illustrated in Figure 18. As can be seen, step

4 involves developing an unpatchable exploit for a high severity vulnerability, rather than using a known public exploit. The difficulty of conceiving such an exploit is evident in the decreased probability described in this attack step. The probability of succeeding with the attack given this scenario is 11% (given that previous mitigation suggestions are disregarded).

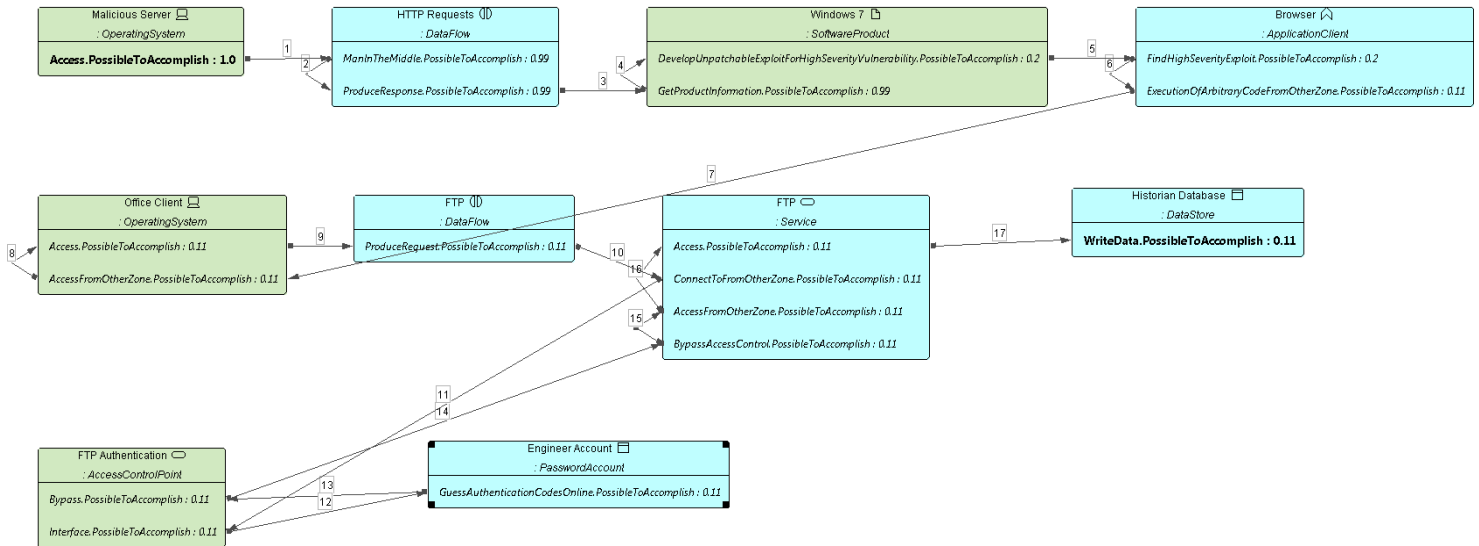


Figure 18. Mitigation through software patching

2.2.2.5 Summary of scenario 2.

A summary of the attack scenario and its mitigation suggestions can be seen in Table 4. It seems that the best mitigation option for this threat is to implement password policies, and then to train personnel regarding security awareness. As for the first attack scenario (cf. Section 2.2.1.4), there are naturally numerous factors that are not considered in this simple example. However, it clearly displays the usefulness of the CySeMoL analysis engine.

Table 4: Summary of scenario 2

Scenario	Probability
Base scenario	12%
Mitigation 1: Password policies	9%
Mitigation 2: Security awareness training and password policies	3%
Mitigation 3: Software patching	11%

3 Real-world case studies

3.1 Utility 1

This case study was carried out in two parts distributed over a year. The model created covered in the first part was of typical substations within the utility; the model created in the second party was of the wide area network that connected these to each other and to the control center.

3.1.1 Aim, scope and delimitations

This case study aimed to testing the modelling tool in practice and analyzes vulnerabilities in a subset of the utility's SCADA system. The analysis was broad and also aimed at suggesting improvements to architectures used today. The analysis was primarily quantitative and CySeMoL was used to assess vulnerabilities. However, manual analysis was also performed on aspects not covered in CySeMoL. For instance, patching procedures were discussed in-depth, and plans for handling malware attacks were discussed. Since CySeMoL requires that security-relevant information is collected and compiled, such analysis is straightforward to perform. For instance, the possibility to isolate networks and to move operations to uncompromised can be assessed using the data collected about networks, the services in them, and dataflows that create dependencies.

3.1.2 Models created and data collected

The study was performed in two parts. In the first part substations were modelled and analyzed; in the second part the wide area network was modelled and the complete architecture was analyzed.

All substations within the utility followed two solution patterns. In the first part models were created of these standard architectures. Hence, models were not created for all individual substations since they are all more or less identical. In the second part, models of the wide area network were completed. Since CySeMoL does not represent details of network implementation (e.g., wires and level-2 switches) this model is of modest size.

Documentation served as a basis for models in both parts of the case study. However, interviews with system owners were needed to complete the models. Approximately 10 hours was spent on preparations in each part, and approximately 6 hours of interviews. In the first part these six hours also included a study visit and ocular inspection of the modelled substations' IT systems.

3.1.3 Analysis performed

The analysis was not delimited to any specific attack forms. However, since the connection to external systems was not modelled, it required assumptions about from where attacks could be initiated. The assumptions used were that attacks were performed from within a substation engineer's office computer or from a forced entry to a substation. At the time of this report's completion only result from the first analysis has been presented to the system owner. The system owner did not disapprove with the analysis, but was somewhat surprised by its result who asked for several clarifications and the

rationale for the number. In addition to this analysis, the data collection (which CySeMoL requires) identified several possible issues outside the scope of CySeMoL. For instance, the difference between physical separation and conceptual/logical separation of computer networks in substations.

3.1.4 Lessons learned

The first part of this project was the first application of CySeMoL to an installed SCADA system. It demonstrated that the entities and relationships required by CySeMoL are practically feasible to collect data on. It also helped to identify issues of scale and redundancy. For instance, substations often contain a number of entities (e.g., Programmable Logic Controllers, PLCs) that are identical to each other. Support to enter such data (e.g., to copy-paste) was missing at the time of modelling.

3.2 Utility 2

This case study is the most comprehensive of the three. In this study the control center of a distribution utility supplying approximately one million customers. The analysis focused technical attacks on the SCADA system from the internet or office network.

3.2.1 Aim, scope and delimitations

The aim of this study was to perform a broad analysis of vulnerabilities present in the utility's IT environment and their impact on the SCADA system's vulnerability. The primary focus was on attacks of technical nature (i.e., not social engineering attacks). The analysis focused on attacks where the attacker would gain administrative privileges on machines that were: HMI workstations, SCADA servers, front-end servers, or data historians. The attacker was assumed to start the attack from the office network. Hence, the attacker would first have to gain access to this network (e.g., by compromising a machine or physically connect to the network).

3.2.2 Models created and data collected

The instance model was produced from system documentation and from interviews with system administrators. The utility was visited for about 12 hours, and their staff spent about 3 hours on providing data.

About 70 per cent of the entities and relationships in the final model were produced directly from existing documentation. This documentation consisted of spreadsheets over installed machines in networks and a diagram over services, system integrations and VLANs. A simplification made was to not model redundant networks (where machines are identical with each other). This simplification has no direct impact on the analysis results, but reduced the modelling effort greatly.

Interviews with system administrators concerned the details of entities modelled based on documentation and completeness checks. The details did for instance include information on operating system protection mechanisms and management practices. The completeness checks ensured that the collected and compiled was complete, and that no essential entities or relationships were missing.

After the initial data collection the model was instantiated in EAAT and its entities were ordered in viewpoints. This process was the most time-consuming, requiring approximately 40 hours from an analyst.

3.2.3 Analysis performed

The analysis focused on attacks where the attacker would gain administrative privileges on HMI workstations, SCADA servers, front-end servers, or data historians. The analysis was made given different scenarios:

- the current architecture
- an architecture where the firewalls' configurations would be 100% correct
- an architecture with an intrusion prevention system monitoring network interfaces
- if regular reviews would be performed of network logs
- if web interfaces to services (e.g., historians) in the DMZ would be removed

The result showed a number of possible ways that an attacker can achieve the abovementioned goals and it differed between the scenarios. However, most of these attacks were estimated to have a low probability of success; multiple hops would be required in all cases. When the result was presented to the utility emphasis was placed on attacks with comparably high probability of success. The presentation also described the factors that would reduce the probability of success. While no attack-paths found in this analysis was surprising the quantification and prioritization was appreciated by the utility.

3.2.4 Lessons learned

This case study showed that the data required by CySeMoL often is straightforward to collect. It is either documented already, or is known by the system owners. This is encouraging since an instance model then can serve more meaningful purposes than just analysis, e.g., as an asset inventory. The study also helped to identify further work that would be required if CySeMoL and its implementation in EAAT should be practically usable. In particular, two aspects needs to be improved:

- Support for model instantiation, including import of data from other formats, easier user interface for manual input, and support for portraying instance models graphically, without manually creating the graphical part.
- To produce analysis results that are easier to comprehend for a practitioner (i.e., someone else than the researcher).

The first bullet has partly been implemented since this case study. EAAT now supports import of data from XML-files. However, manual input is still limited to graphical drawings and no support is available for automatically producing graphical model from imported data or manually created data. No work has been done on the second bullet, analysis results today come in comma-separated files.

3.3 Utility 3

This case study focused on communication solutions between substations and the control center at a large distribution company.

3.3.1 Aim, scope and delimitations

The aim of this study was open ended in the sense that it was interested in identifying and analysing any type of vulnerabilities. From a research point of view the focus of this case study was to identify improvement suggestions for modelling tool that was used (EAAT), rather than CySeMoL. However, feedback on CySeMoL was also obtained.

3.3.2 Models created and data collected

The models created during this study are moderate in size, entities in the substation and the control center has been kept to a bare minimum to highlight the data flow between these. This was done to prevent the study from spending too much time on understanding a large and detailed model. Instead focus was spent on the tool support for modelling and analysis. The model created and analysed provide information about to which extent it is possible to accomplish attacks in the network such as: eavesdropping, replay messages, produce requests/responses and performing a man in the middle attack. The data used to create the model was mainly collected through system documentation. Two short interviews were conducted, one as a starting point to get familiar with the overall design of the SCADA system and the second as a check-up to validate the content of the created model. Approximately four hours were spent on creating the model after the information was collected.

3.3.3 Analysis performed

The created models were used to evaluate the possible success rates of the different attacks on the network mentioned above. Two scenarios were created using different communication protocols in the otherwise same model. The results could then be compared to see which protocol of the two that is the most secure to use when communicating from substation LAN to SCADA LAN. The functionality in the tool was based on two separate parts:

- Creating a model and performing an analysis on the model
- Tool demonstration and testing with system owners

System owners have been included in discussions on how to improve the tool. In total system owners have spent about 10 hours providing their opinions on the tool, its functionality and the modelling process. Some other conclusions about the modelling process could be drawn when creating the model.

3.3.4 Lessons learned

Many of the problems found during the study have been improved. The major things found are:

- It is fairly time-consuming to create a whole new model. To solve this, the tool could support preconfigured entities (or even sub-models) to use

- It is hard to find weak spots in the system. This has been improved by adding the possibility to show visual notifications in the model based on the analysis result in the tool. Finding weak spots is naturally a way to suggest mitigations. Thus, this functionality is essential for providing a support for an iterative analysis procedure. To automate this procedure could be interesting.
- Lack of collaboration support. It would be good if system owners and other stakeholders could model, use and maintain their part of the system while contributing to a comprehensive enterprise-wide model for complete analysis.
- The system owners involved have had a positive attitude and could see the great benefits of having a model that provides both system documentation and security analysis capabilities.

4 Appendix

This appendix specifies all the defined states of variables in the illustrated example discussed in this report. Three different colors are used to differentiate between the three main variable types in CySeMoL, from a data collection viewpoint. These semantics are illustrated in Table 5.

Table 5: Semantics when reading appendix.

Needs to be determined by the user (if not modeled the state is set to the default value FALSE).
Can be easily determined, but is not needed to be collected as it is calculated by CySeMoL.
Not recommended to be determined by the user as it generally is difficult to collect data for. The values are calculated by CySeMoL.

The rest of the appendix describes the attributes and states for all entities present in the example application of CySeMoL in this report. A state defined by the user in this report is either TRUE (T) or FALSE (F), or not defined (in which case the corresponding cell is empty and the value is calculated by CySeMoL).

4.1 States of instantiated AccessControlPoints.

AccessControlPoint	State												
	A	B	C	D	E	F	G	H	I	J	K	L	M
Functioning													
Bypass.PossibleToAccomplish													
Interface.PossibleToAccomplish													

A: Logon AS
 B: PCU Comm Authentication
 C: SCADA Apps Logon
 D: Terminal Login
 E: Windows Login DC
 F: Windows Login Office Client
 G: Windows Login Mail
 H: Windows Login Web
 I: Windows Login DB
 J: Windows Login HMI
 K: Windows Login PCU Comm
 L: Windows Login SCADA DC
 M: PCU Authentication

4.2 States of instantiated ApplicationClients.

ApplicationClient	State			
	A	B	C	D
HasAllHighSeverityPatches.Functioning			F	
Access.PossibleToAccomplish				
DenialOfService.PossibleToAccomplish				
FindHighSeverityExploit.PossibleToAccomplish				
ExecutionOfArbitraryCodeFromOtherZone.PossibleToAccomplish				
ExecutionOfArbitraryCodeFromSameZone.PossibleToAccomplish				

A: Browser
 B: Email client
 C: PCU Comm Software
 D: Terminal Client

4.3 States of instantiated Dataflows.

Dataflow	State												
	A	B	C	D	E	F	G	H	I	J	K	L	M
Disrupt.PossibleToAccomplish													
Replay.PossibleToAccomplish													
Eavesdrop.PossibleToAccomplish													
ManInTheMiddle.PossibleToAccomplish													
ProduceRequest.PossibleToAccomplish													
ProduceResponse.PossibleToAccomplish													

A: FTP
 B: HMI-AS Communication
 C: HTTP
 D: HTTP Requests
 E: HTTPS
 F: PCU Communication
 G: PCU Terminal
 H: RDP (HMI)
 I: RDP (Office DC)
 J: SSH (SSL)
 K: SMTP E
 L: SMTP I
 M: Terminal Service

4.4 States of instantiated DataStores.

DataStore	State Historian Database
CryptographicObfuscation.Functioning	T
ReadData.PossibleToAccomplish	
WriteData.PossibleToAccomplish	
DeleteData.PossibleToAccomplish	

4.5 States of instantiated DeepPacketInspections.

DeepPacketInspection	State	
	Cisco IDS	IPS (Scenario 1, Mitigation 2)
Functioning	T	T

4.6 States of instantiated Firewalls.

Firewall	State			
	DMZ	External DMZ	Office	SCADA
Functioning	F	T	F	F

4.7 States of instantiated NetworkInterfaces.

NetworkInterface	State				
	A	B	C	D	E
StaticARPTables.Functioning	F	T	F	F	F
ARPSpoof.PossibleToAccomplish					
DenialOfService.PossibleToAccomplish					

A: DMZ/Office

B: External DMZ/Internet

C: Office/External DMZ

D: Process/SCADA

E: SCADA/DMZ

4.8 States of instantiated NetworkZones.

NetworkZone	State					
	A	B	C	D	E	F
DNSSEC.Functioning		T				
PortSecurity.Functioning		T				
DNSSpoofing.PossibleToAccomplish						
FindUnknownEntryPoint.PossibleToAccomplish						
ObtainOwnAddress.PossibleToAccomplish						

A: DMZ

B: External DMZ

C: Internet

D: Office

E: Process

F: SCADA

4.9 States of instantiated OperatingSystems.

OperatingSystem	State										
	A	B	C	D	E	F	G	H	I	J	K
StaticARPTables.Functioning	F	F	F	F	F	F	F	F	F	F	F
HostFirewall.Functioning	F	F	F	T	F	F	F	F	F	F	T
ExecutableSpaceProtection.Functioning	F	F	T	T	T	T	F	F	T	T	T
AddressSpaceLayoutRandomization.Functioning	F	F	F	T	T	T	F	F	F	F	T
HasAllHighSeverityPatches.Functioning	F	F		F			F		F		T
Access.PossibleToAccomplish											
DenialOfService.PossibleToAccomplish											
FindHighSeverityExploit.PossibleToAccomplish											
ExecutionOfArbitraryCodeInUnknownServices											
FromSameZone.PossibleToAccomplish											
ExecutionOfArbitraryCodeInUnknownServices											
FromOtherZone.PossibleToAccomplish											
ARPSpoof.PossibleToAccomplish											
AccessThroughPortableMedia.PossibleToAccomplish											
FindUnknownServiceFromSameZone.PossibleToAccomplish											
FindUnknownServiceFromOtherZone.PossibleToAccomplish											
AccessFromOtherZone.PossibleToAccomplish											
AccessFromSameZone.PossibleToAccomplish											
AccessThroughUIFromSameZone.PossibleToAccomplish											
AccessThroughUIFromOtherZone.PossibleToAccomplish											
Access control non-functioning or bypassed											
A: Application Server B: HMI C: Historian D: Mail Server E: Office Client F: Office Domain Controller G: RTU H: PCU Communication I: SCADA Domain Controller J: Terminal Server K: Web Server											

4.10 States of instantiated PasswordAccounts.

PasswordAccount
GuessCredentialsOffline.PossibleToAccomplish
SocialEngineerAuthenticationCodes.PossibleToAccomplish
GuessAuthenticationCodeOnline.PossibleToAccomplish
Admin Account
Engineer Account
User Account 1
User Account 2
User Account 3

4.11 States of instantiated PasswordAuthenticationMechanisms.

PasswordAuthenticationMechanism	State								
	A	B	C	D	E	F	G	H	I
ProactivePasswordChecker.Functioning	F	T	T	T	T	T	F	T	F
HashedRepository.Functioning	T	T	T	T	T	T	T	T	F
HashedRepositorySalted.Functioning	F	F	F	F	F	F	T	T	F
DefaultPasswordsRemoved.Functioning	T	T	T	T	T	T	F	T	T
BackoffTechnique.Functioning	T	T	T	T	T	T	F	T	F
Functioning	T	T	T	T	T	T	T	T	T
ExtractPasswordRepository.PossibleToAccomplish									

A: Mail Server Authentication Mechanism
 B: Office Domain Controller
 C: PCU Authentication Mechanism
 D: SCADA Apps Authentication Mechanism
 E: SCADA Domain Controller
 F: Web Server Authentication Mechanism
 G: Historian Authentication Mechanism
 H: Historian Authentication Mechanism
 (Scenario 2, Mitigation 1 and 2)
 I: Terminal Server Authentication Mechanism

4.12 States of instantiated Persons.

Person
SCADA Engineer
Super Admin
User 1
User 2
User 3

4.13 States of instantiated PhysicalZones.

PhysicalZone	State				
	A	B	C	D	E
Access.PossibleToAccomplish	F	F	F	F	F

A: Physical Access to External DMZ
 B: Physical Access to Internal DMZ
 C: Physical Access to Office
 D: Physical Access to Process
 E: Physical Access to SCADA

4.14 States of instantiated Protocols.

Protocol	State	
	SSL	No Encryption
FreshnessIndicator.Functioning	T	F
CryptographicObfuscation.Functioning	T	F
CryptographicAuthentication.Functioning	F	F

4.15 States of instantiated SecurityAwarenessPrograms.

SecurityAwarenessProgram	State		
	Security Training Program Level 1	Security Training Program Level 2	Security Training Program Level 2 (Scenario 2, Mitigation 2)
Functioning	F	F	T

4.16 States of instantiated Services.

Service	State												
	A	B	C	D	E	F	G	H	I	J	K	L	M
LoadBalancer.Functioning	F	T	T	F	F	F	F	F	F	F	F	F	F
SecretRoaming.Functioning	F	F	F	F	F	F	F	F	F	F	F	F	F
HasAllHighSeverityPatches.Functioning		F	F	F			F	F	F	T			
Access.PossibleToAccomplish													
DenialOfService.PossibleToAccomplish													
FindHighSeverityExploit.PossibleToAccomplish													
ConnectToFromSameZone.PossibleToAccomplish													
ExecutionOfArbitraryCodeFromSameZone.PossibleToAccomplish													
ConnectToFromOtherZone.PossibleToAccomplish													
ExecutionOfArbitraryCodeFromOtherZone.PossibleToAccomplish													
DoSFlood.PossibleToAccomplish													
SemanticDoS.PossibleToAccomplish													
AccessFromSameZone.PossibleToAccomplish													
AccessFromOtherZone.PossibleToAccomplish													
Access control non-functioning or bypassed													
A: FTP													
B: HTTP													
C: HTTPS													
D: PCU Comm Terminal													
E: RDP HMI													
F: RDP Office DC													
G: SCADA Applications													
H: SMB													
I: SMB (Scenario 1, Mitigation 1)													
J: SMTP													
K: SSH													
L: Terminal Service													
M: Active Directory Server													

4.17 States of instantiated SoftwareProducts.

SoftwareProduct	State						
	A	B	C	D	E	F	G
OnlyUseSafeLanguages.Functioning	F	F	F	F	F	F	F
HasNoPublicUnpatchableHighSeverityVulnerability.Functioning	T	T	F	T	T	T	T
HasNoPublicPatchableHighSeverityVulnerability.Functioning	F	T	F	F	F	F	F
Scrutinized.Functioning	T	F	T	T	T	T	T
ImprovedWithStaticCodeAnalysis.Functioning	T	T	F	T	T	T	T
SourceCodeClosed.Functioning	F	T	F	T	T	T	T
BinarySecret.Functioning	F	T	F	F	F	F	F
ObtainSourceCode.PossibleToAccomplish							
ObtainBinaryCode.PossibleToAccomplish							
DevelopPublicUnpatchableExploitForHighSeverity							
Vulnerability.PossibleToAccomplish							
DevelopPublicPatchableExploitForHighSeverity							
Vulnerability.PossibleToAccomplish							
GetProductInformation.PossibleToAccomplish							

A: Red Hat 4.0
 B: SCADA Network Manager
 C: SCADA Solution
 D: Windows 7
 E: Windows Server 2003
 F: Windows Server 2008
 G: Windows XP

4.18 States of instantiated ZoneManagementProcesses.

ZoneManagementProcess	State				
	A	B	C	D	E
HostHardeningProcedures.Functioning	T	T	F	F	F
AutomatedPatchAndUpdatingProcess.Functioning	T	T	T	F	F
RegularLogReviews.Functioning	T	F	F	F	F
FormalChangeControlProcess.Functioning	T	F	F	F	F
RegularSecurityAudits.Functioning	T	T	T	F	F

A: External DMZ Zone Management
 B: Internal DMZ Zone Management
 C: Office Zone Management
 D: Process Zone Management
 E: SCADA Zone Management