FLAMINGO

European Seventh Framework Network of Excellence

`http://www.fp7-flamingo.eu/`

WP1 — Joint Software and Labs

Deliverable D1.2 — Second Year Report on Joint Software and Labs

University of Twente, The Netherlands (UT)
Institut National de Recherche en Informatique et Automatique, France (INRIA)
University of Zurich, Switzerland (UZH)
Jacobs University Bremen, Germany (JUB)
Universität der Bundeswehr München, Germany (UniBwM)
University Politecnicà de Catalunia, Spain (UPC)
iMinds, Belgium (iMinds)
University College London, United Kingdom (UCL)

Project funded by the European Union under the
Information and Communication Technologies FP7 Cooperation Programme
Grant Agreement number ICT-FP7 318488

Document Control

Title:	D1.2 — Second Year Report on Joint Software and Labs
Type:	Public
Editor(s):	Anuj Sehgal, Nikolay Melnikov, Jürgen Schönwälder
E-mail:	{s.anuj, n.melnikov, j.schoenwaelder}@jacobs-university.de
Doc ID:	D1.2
Delivery Date:	2014-10-31
Author(s):	Aiko Pras, Anna Sperotto, Anuj Sehgal, Christos Tsiaras, Daphné Tuncer, Guilherme Sperb Machado, Jair Santanna, Jeroen Famaey, Jürgen Schönwälder, Maxim Claeys, Marinos Charalambides, Niels Bouten, Nikolay Melnikov, Olivier Festor, Rick Hofstede, Sebastian Seeber, Vaibhav Bajpai

For more information, please contact:

Dr. Aiko Pras
Design and Analysis of Communication Systems
University of Twente
P.O. BOX 217
7500 AE Enschede
The Netherlands
Phone: +31-53-4893778
Fax: +31-53-4894524
E-mail: <a.pras@utwente.nl>

Executive Summary

The work of this work package has progressed in the second year according to the description of work. Multiple FLAMINGO open source projects have been improved during the second year (Task 1.1). As such, it is believed that the objective “Within the first two years FLAMINGO will develop at least two open source software packages.” has been reached. Development on a new open source software jointly by multiple FLAMINGO partner institutions has also commenced. The objective “In the remaining years at least two additional open source packages sponsored by FLAMINGO will become available.” will be reached. An internal review of FLAMINGO open source software packages was performed by FLAMINGO PhD students, in order to enhance the likelihood of collaboration on these packages and also help improve them. Results from this review were compiled together and provided to the developers of the open source packages.

During the second year of the project, several infrastructure developments have taken place within the joint security lab (Task 1.2). The objective “After one and a half year, at least four FLAMINGO partners will participate within a joint security lab.” has been met. The facilities and data traces provided by the joint security lab have been utilized within the scope of multiple PhD student collaborations and these have resulted in several publications.

In year two, the University of Geneva joined EmanicsLab. Two nodes have been deployed at this new institution (Task 1.3). The objective “By the end of year three, the EMANICS lab will be extended with additional nodes to increase the processing / storage capabilities of this shared lab infrastructure.” is likely to be met. The EmanicsLab software was updated during the year, with the latest version now migrating to using Linux containers (LXC) instead of VServers for the slices. The move to LXC is advantageous since containers are now adopted into the mainline Linux kernel, unlike VServers. Initial support of IPv6 within EmanicsLab slices has been enabled in a manual way. Automatic and persistent support requires the development of new modules for the MyPLC distribution used by EmanicsLab. Once development of these modules is complete, in year three, these changes will be pushed upstream to the maintainers of MyPLC. The move to Linux containers introduced some bugs, which were resolved. Based on PhD student feedback, documentation for EmanicsLab has been improved and a new has been installed to host up-to-date information regarding how users can perform important tasks. A script was also setup to inform users about the impending expiration of their slices. Usage of EmanicsLab has been encouraged, with it seeing application in course work, research studies and deployment of FLAMINGO open source tools.

The usage of the Virtual Wall and related FIRE facilities (Task 1.4) has has seen growth over the second year, with multiple users running experiments on the infrastructure actively. To encourage usage of these facilities outside the scope of the project as well, a tutorial on deploying OpenFlow experiments on the Virtual Wall was presented at AIMS 2014.

Contents

1 Introduction

The primary objectives of this work package is to integrate the research of FLAMINGO partners by sharing software, tools and lab facilities. To achieve this objective, the FLAMINGO consortium committed itself to perform the following tasks.

- Creation and maintenance of open source software (Task 1.1).
- Creation and operation of a joint security lab (Task 1.2).
- Operation and evolution of the distributed EmanicsLab testbed (Task 1.3).
- Integration of existing infrastructures into virtual laboratories (Task 1.4).

The following SMART objectives (Specific, Measurable, Achievable, Relevant, Timely) relate to this work package.

- Integration of Labs: The High Security Laboratory at INRIA Nancy will become the starting point for an integrated security Lab. After one and a half year, at least four FLAMINGO partners will participate within a joint security lab. By the end of year three, the EMANICS lab will be extended with additional nodes to increase the processing / storage capabilities of this shared lab infrastructure.
- Open Source Software: Within the first two years FLAMINGO will develop at least two open source software packages. In the remaining years at least two additional open source packages sponsored by FLAMINGO will become available.

During the first year of the project, several open source software packages developed by various project partners were identified. To encourage collaborative developments, to increase familiarity with these software packages amongst partners and to identify possible improvements an internal review of FLAMINGO open source packages was performed by FLAMINGO PhD students at the beginning of the second year of the project. This review resulted in several updates to existing open source packages. The development on a new package was started as a collaboration between multiple project partners. To encourage adoption of these open source packages by the general community, dissemination activities were also carried out. Further details about the open source projects, their evaluation, improvements and developments made during the second year, and also an overview of the dissemination activities can be found in Section 2.

The infrastructure included in the joint security lab has received enhancements during the second year of the project since several partners updated and expanded their infrastructures. New traffic traces, which can be made available to project partners, are also being collected by some partner institutions. Several publications have resulted out of PhD student collaborations that have utilized joint security lab infrastructures and data traces. Section 3 documents the infrastructure enhancements, data traces collected and publications resulting from usage of joint security lab infrastructure.

The EmanicsLab has seen an expansion during the second year of the project, with University of Geneva joining and adding two nodes to the infrastructure. A migration to Linux containers was initiated, resulting in an upgrade to Fedora version 20 for the EmanicsLab nodes. Some bugs discovered after the migration have been fixed and support for IPv6 in the EmanicsLab slices has been enabled using a manual approach. Details regarding work performed on EmanicsLab and its usage during the second year of the project can be found in Section 4.

During the second year of the project, the usage of the Virtual Wall and FIRE facilities operated by the iMinds group has developed. These facilities are currently being used by several users to run experiments on. A tutorial was delivered at AIMS 2014 in order to attract new users. An overview of the activities related to the Virtual Wall are provided in Section 5.

2 Open Source Software

The activities related to the Open Source Software packages over the course of the second year of FLAMINGO project can be summarized as follows:

- Evaluation of software packages
- New version releases
- Software packaging for different systems
- User community questionnaire
- Dissemination through tutorials

An overview of the open source packages maintained by the consortium is provided in Section 2.1. This is followed by details of the evaluation procedure and its results in Section 2.2. Details regarding updates made to the open source packages during the second year of the project are presented in Section 2.3. Development activities related to BonaFide+, a new open source package developed collaboratively by the consortium, are discussed in Section 2.4. Dissemination activities related to the open source packages are discussed in Section 2.5. This section focuses on three open source packages (FNSS, SSHCure and BonaFide+). These packages were chosen since they have already reached a certain level of maturity and are seeing adoption by the larger community.

2.1 Overview of Open Source Projects

Package name	Origin	Description	Details In
SURFmap	UT	Goople Maps Flow Visualization Tool	D1.1
SSHCure	UT	SSH Intrusion Detection Plugin for NfSen	D1.1
tau-FST	UPC	Fuzzy logic-based finite-state transducers	D1.1
FNSS	UCL	Fast Network Simulation Setup	D1.1
TomP2P	UZH	P2P-based Key-value Pair Storage Library	D1.1
B-Tracker	UZH	P2P tracker with improved load balancing	D1.1
TraceMan	UZH	Traceroute-based Measurement and Management Tool	D1.1
DeRISC	UZH	Dispute rEsolution Recommender	D1.1
OTAGen	IMINDS	Generate Ontologies and Corresponding Queries	D1.1
NFQL	JUB	Network Flow Query Language	D1.1
ncclient	JUB	Python NETCONF API	D1.1
Contiki-snmp	JUB	SNMP agent running on Contiki	D1.1
Contiki-dtls	JUB	TLS/DTLS running on Contiki	D1.1
libsmi	JUB	Library for accessing SMI MIB module definitions	**D1.2**
BonaFide+	JUB	Mobile networks QoS and QoE mapping tool	**D1.2**

Table 1: List of the open source software packages of the project partners

The list of open source packages maintained by FLAMINGO partners is presented in Table 1. A short description of the FLAMINGO open source packages added during the second year can be found below. A similar short description of the other FLAMINGO open source packages can be found in deliverable D1.1.

2.1.1 BonaFide+

The BonaFide+ software package [1] is a mobile application developed for the Android operating system in order to perform active protocol based quality of service (QoS) measurements in mobile networks and to map inferred quality of experience (QoE) from these values. The application is able to measure and map QoE for HTTP, H.323, SIP, RTSP, BitTorrent and Flash Video Streaming. This application is an extension of BonaFide [2], which was originally designed and implemented by Jacobs University Bremen (JUB) for discovering traffic shaping in mobile networks. BonaFide+ can be downloaded from `http://www.bonafide.pw/`.

2.1.2 libsmi

The core of the libsmi software package is a library that allows network management applications to access SMI MIB module definitions. On top of this library, there are tools to check, analyze dump, convert, and compare MIB module definitions. The distribution also contains a steadily maintained and revised archive of all IETF and IANA maintained standard MIB and PIB modules. A link to the latest version of the software can be found via `http://cnds.eecs.jacobs-university.de/software/libsmi/`.

2.2 Evaluation of FLAMINGO Open Source Projects

The packages listed in Table 1 were evaluated[1]. Each package was assigned to evaluators who were asked to fill out a survey regarding the package being evaluated. The findings based on these evaluations were then presented to package developers at the FLAMINGO project meeting that took place at Jacobs University Bremen between 16 – 18 December 2013.

2.2.1 Software Package Assignment

PhD students and Postdoctoral researchers from the FLAMINGO consortium were used as evaluators. To avoid bias in evaluations, the software assigned to evaluators could not be from their affiliated institution. Due to the differences in lifecycle stages and priorities of the consortium, some packages received a greater number of reviews.

The evaluators were provided a list of all open source FLAMINGO packages. They were then requested to select four packages, based on their field of activity and expertise, that they would be willing to evaluate. In case there were not enough packages that fit the evaluators' primary area of work, they were required to choose at least two packages that interested them.

Based on the package choices returned by the evaluators, each one was assigned two packages to review. A list of the names of evaluators, their affiliation and the packages they evaluated is provided in Table 2.

2.2.2 Evaluation Parameters

Evaluations were submitted two-weeks after the package assignments. The distribution of reviews per package is shown in Table 3.

[1] BonaFide+ was excluded from this evaluation because development of the package started only after the evaluation began. No results for OTAGen are reported since a meaningful evaluation of the tool could not be completed by the assigned evaluator. Since libsmi was not updated before the review process, it has also not been evaluated.

Reviewer name	Affiliation	Packages assigned
Anna Sperotto	UT	tau-FST, B-Tracker
Antha Mayzaud	INRIA	Contiki-dtls, Contiki-snmp
Anuj Sehgal	JUB	TraceMan, SSHCure
Andri Lareida	UZH	FNSS, SURFmap
Christos Tsiaras	UZH	FNSS, SURFmap
Daphne Tuncer	UCL	DeRISC, TraceMan
Jair Santanna	UT	tau-FST, B-Tracker
Mario Flores	UPC	FNSS, NFQL
Maxim Claeys	IMINDS	NFQL, FNSS
Niels Bouten	IMINDS	TomP2P, ncclient
Nikolay Melnikov	JUB	SSHCure, tomp2p
Rashid Mijumbi	UPC	FNSS, OTAGen
Ricardo Schmidt	UT	NFQL, TraceMan
Rick Hofstede	UT	NFQL, TraceMan
Sebastian Seeber	UNIBWM	FNSS, SURFmap
Vaibhav Bajpai	JUB	TomP2P, TraceMan

Table 2: Reviewers of the open source software packages

A reference set of evaluation criteria questions was provided, and the evaluators were encouraged to document any other issues and suggestions that may not have been covered by the evaluation questions. The questionnaire dealt with the *installation*, *documentation*, *usage* and *possible extensions* of the software packages. The contents of these sections of the questionnaire are provided below:

Installation

I-1 What operating system did you use? What operating systems the tool claims to support?

I-2 Did you experience any problems during software download or installation (e.g. dependencies, etc.)?

I-3 How much time did it take you to install the package?

Documentation

D-1 Is on-line or off-line documentation available?

Usage

U-1 Did the package include example code or a quick start guide?

U-2 What were the frustrating aspects of using the package, e.g., complicated parameters, counter-intuitive inputs/outputs, prior related knowledge, etc.?

U-3 Would any PhD collaboration within FLAMINGO benefit from using this package in its current or extended form?

Possible Extensions

E-1 Is there a way to make the installation/usage of the package simpler? What would it be?

E-2 What extensions or features could benefit the tool?

E-3 Would you use this package if it was improved in a specific way? What would that improvement be?

Package name	Number of evaluations
B-Tracker	2
Contiki-dtls	1
Contiki-snmp	1
DeRISC	1
FNSS	6
ncclient	1
NFQL	4
SSHCure	2
SURFmap	3
tau-FST	2
TomP2P	3
TraceMan	5

Table 3: Number of reviewers per software package

2.2.3 Evaluation Findings

The results of the software evaluation were discussed during the December 2013 FLAMINGO project meeting in Bremen. Feedback from the software package maintainers indicated that they found the feedback useful because it highlighted issues that are often overlooked by authors themselves. A summary of the per package feedback can be found below.

B-Tracker The evaluators found it difficult to judge whether the package was functioning as required. While the software could be successfully downloaded, it could not be loaded successfully in Vuze.

No off-line and only sparse on-line documentation was cited as a reason for difficulties in successfully using the package. Evaluators requested for better documentation, including examples and test cases.

Contiki-dtls The evaluator indicated that the package did not come along with any documentation so it was hard to determine which embedded system platform it was actually targeting. Though sample code was found, lack of documentation and unclear coding style for the examples meant that the evaluator did not know what the purpose of the examples was, and could not decide which files to compile.

Contiki-snmp The target embedded system platform of the package was unclear for the evaluator, however, it could be identified by referring to webpages that mentioned the software. A lack of documentation was seen as causing a lot of difficulty towards usage of the package, since precise instructions on its usage were lacking. The evaluator also requested that information about dependencies be included since this was not clear either.

DeRISC The evaluator did not have any difficulty using the tool since it is web-based, however, it was felt that the documentation was not very clear. Specific feedback on what improvements should be made were provided and more information and guidelines regarding the tool were requested for the documentation.

FNSS None of the evaluators had any significant difficulties with the installation of the tool. However, they did request the following improvements:

- Detailed installation guide for each OS complete with screenshots of each step.
- Documentation regarding most common issues and their solutions.
- Simplify the process for installation of dependencies, or at least do not rely on 3rd party guides for the installation of the Python interpreter and other dependencies.

While some improvements to the documentation regarding installation were sought, all reviewers were able to locate the on-line documentation and were satisfied with it. However, a quick-start guide and a user manual detailing the usage of all components was requested. It was also indicated that while code examples were well written, they were not easy to locate since some were on the website and others only accessible in the software package.

ncclient The evaluators were able to install the package, but changes in dependencies meant that they had to find previous versions that would work. The documentation included was rated as being quite limited. Documents providing guidance on installing some NETCONF servers and test cases were requested.

NFQL The download link was initially found to be broken, but a fork from the Git repository could be obtained. Installation of the tool was not easy because it required changes to the makefile, which were not documented. It has been requested that documentation for Red Hat Linux distributions also be provided. The documentation related to the tool was found to be quite sparse, with little information regarding what the sample query is supposed to do. More detailed on-line and off-line API documentation was also requested. This documentation should focus on explaining more thoroughly how to use the engine and how to create queries.

While the query examples provided were useful in getting started, the evaluators felt that some of the code using JSON queries was not very intuitive and requires more comments. Explanation of parameters used would also be welcome.

SSHCure The reviewers did not have problems with the installation process and the documentation provided was also deemed quite useful. The evaluators commented upon the difficulty of installing NfSen, for which SSHCure is a plugin. Documentation related to the installation and usage of NfSen was requested.

SURFmap The reviewers did not have problems with the installation process and the documentation provided was also deemed quite useful. The evaluators commented upon the difficulty of installing NfSen, for which SSHCure is a plugin. Documentation related to the installation and usage of NfSen was requested.

tau-FST The evaluators found it quite difficult to successfully install the package since there were no instructions provided. Inspection of the source code and publications was required to determine dependencies. Since the sample source code also failed, the evaluators requested clear examples and documentation.

TomP2P The evaluators did not have any trouble with installation of the libraries, since it only needed including into the project. Online examples provided on the website were found to be quite useful. Expansion of the documentation has been requested since not all methods are documented.

TraceMan The evaluators found it relatively easy to install and use the software, but pointed out that target platforms were not specified and that dependence on a certain traceroute package means that it cannot be used on other systems easily. Documentation needs to be improved since it currently does not provide information on dependencies and the fact that a database engine might be needed. A simple quick-start documentation is also required since the current document focuses on the technical aspects of TraceMan. Some error messages related to the database engine were encountered and examination of the code was required to understand what they meant. Improvement of error documentation has been recommended.

2.3 Software Updates to Open Source Packages

Several open source packages developed and maintained by FLAMINGO partners received active updates. Some of these updates were based upon the feedback received during the first evaluation round. The updates performed for FNSS, libsmi, NFQL, SSHCure and TomP2P are highlighted here since these packages have already reached a certain level of maturity and adoption by the community.

2.3.1 FNSS

In the second year of the FLAMINGO project, the development of the Fast Network Simulation Setup (FNSS) proceeded further. The latest version is 0.5.1. The development effort took into account the outcome of the FLAMINGO evaluation process. The main development effort on FNSS during the second year can be summarized as follows:

- Documentation and ease of use
  - Improved the overall clarity and scope of the documentation.
  - Added several examples covering various use cases and modified the FNSS website to make examples easily accessible from the home page.
  - Increased degree of automation of the installation process by adding a script for installing FNSS and all dependencies in one step on Ubuntu operating systems and improving the build scripts of various components.
- Quality and reliability
  - Added a suite of test cases for the C++ library.
  - Added test cases to existing test suites.
  - Adopted Travis CI to automatically run all test cases each time a new commit is made to the codebase.
  - Fixed bugs discovered as a result of the addition of new test cases.
- Support for new simulators and emulators

  - Mininet
  - Omnet++
  - Autonetkit
  - jFed
- Miscellaneous
  - Added a generator for Chord DHT topologies.
  - Created FNSS virtual machine image and scripts to create such an image that allows users to run the FNSS code in the cloud or use FNSS in an isolated virtual machine.
  - Changed the templating engine for the ns-2 adapter from Templite++, which is released under GPL v2 license, to Mako, which is released under a BSD license. This change made it possible to release the ns-2 adapter under a BSD license, which gives more freedom to the final user.

2.3.2 libsmi

A new release of libsmi, version 0.5.0, was made during the second year of the project. libsmi is widely distributed and packaged for multiple platforms (Debian, Ubuntu, MacPorts, Homebrew, etc.). The SimpleWeb[2] also uses libsmi tools. The updates performed during this year can be summarized as follows:

- Tracking of MIB modules published by the RFC Editor or updated by IANA.
- Tracking YANG modules published by the RFC Editor and any errata published by the RFC Editor.
- Software updates to track changes in software packages the libsmi software depends on (in particular changes in bison).
- Minor updates to improve the quality and stability of the software.
- Production of a new official release (version 0.5.0), which involves updates of the test suite and the GNU auto configuration scripts.

2.3.3 NFQL

During the second year of the FLAMINGO project, the development work related to NFQL proceeded further. The latest NFQL release is 0.7.1. Taking into account the outcome of the evaluation process, the main development effort can be categorized as follows:

- Packaging and distribution
  - NFQL was packaged for the Macports[3] system, which is a package manager for Mac OS X. A Ports definition was created for NFQL, allowing the package to be built with a single step on Mac OS X systems.

[2]http://www.simpleweb.org/
[3]http://www.macports.org

  - Since NFQL uses `libfixbuf`, which is not packaged for Macports, a Ports definition was created for `libfixbuf`.
  - Created a Homebrew[4] formula for NFQL, which allows for it to be deployed on Mac OS X systems.
  - Since NFQL uses `json-c`, which is not packaged for Homebrew, a formula for this was also created.
  - A Debian package for `libfixbuf` was also created to ease installation of NFQL on Debian systems.

- New Releases
  - NFQL v0.7.1:
    - `cmake` properly handles binary and `man` page installation in standard locations.
    - Updated README.md with installation instructions. This was done in response to review comments.
  - NFQL v0.7:
    - Merged the NFQL IPFIX codebase with upstream.
    - Deprecated and removed flowy parser codebase.
    - Removed all references to flowy, f, fv2.
    - The executable binary is now called `nfql`. It was previously called engine.
    - Added a `nfql.1 man` page, updated README.md with reading instructions.
    - Fixed `asprintf(...)` unchecked return warnings.
    - Resolved several issues pointed out during the review process.
  - NFQL v0.6:
    - Added installation instructions for FreeBSD and Fedora. This was done in response to review comments.
    - Updated `bin/engine -h`, tracefile must be in flow-tools format. This was done in response to review comments.
    - Replaced all references of flowy with f.
    - Added installation instructions using MacPorts for Mac OS X.
    - Froze `json-c` library dependency to v0.10, since v0.11 uses a new name: `libjson-c`.
- Webpage updates
  - Added documentation on a list of NFQL query features.
  - Added a documentation on list of NFQL implementation features.
  - Added examples with common usage instructions on the webpage upfront.
  - Added history of NFQL to inhibit confusion around: flowy, f and nfql.

2.3.4 SSHCure

Development of SSHCure has continued during the second year of the project. The feedback received during the first review phase could not be taken into account because the comments were targeted at NfSen and the working/installation thereof, rather than at SSHCure itself. However, an

[4]`http://brew.sh`

effort was made several times to contact the developer of NfSen, aiming at making NfSen better, however, a reply was never received.

As such, the updates performed to SSHCure during the second year of the project are as follows:

- Updated detection algorithm based on the work described in [3].
- Fully overhauled user interface, based on conference feedback and end user interviews.

2.3.5 TomP2P

During the second year of the project, the development of TomP2P has continued with efforts towards releasing the new major version 5, which has a rewritten API. An overview of the updates performed can be seen below:

- Features related to relaying and rsync-like synchronization were improved.
- Support for automatically setting up NAT traversal has been improved. Documentation related to the feature was added.
- Security features of TomP2P have been improved in collaboration with INRIA.
- The API for release version 5 was rewritten.
- New major version packaging and bug fixes have been performed all through the second year, with latest version being 5.0 Alpha 24, released on 24 August 2014.

2.4 BonaFide+

The BonaFide+ software package was jointly developed by Jacobs University Bremen, University of Zürich and Universität der Bundeswehr München as an application for the Android mobile operating system to perform active protocol based QoS measurements in mobile networks [1]. These QoS measurements are then interpreted as QoE mean opinion score (MOS) values, which are associated with a color ranging from green to red (good to bad) and plotted on a map. This allows users to view the QoE they can expect from a provider in a specific area, based on historical measurements.

2.4.1 Architecture

BonaFide+ uses a client-server architecture to perform measurement tests. The client, i.e., the Android application, gathers network performance metrics by connecting to and sending data to the server. Since it is possible for mobile network operators (MNOs) to have protocol based traffic shaping policies applied and for the QoE related to a protocol change based on network conditions, just measuring throughput of a link using random data is not enough. As such, BonaFide+ emulates network flows of protocols that users wish to test. Currently, the application is able to measure and map QoE for HTTP, H.323, SIP, RTSP, BitTorrent and Flash Video Streaming. Measurement tests include the upload and download bandwidth achieved for random and emulated protocol data flows. Along with this, latency, signal strength, type of network, location and operator related data are also recorded.

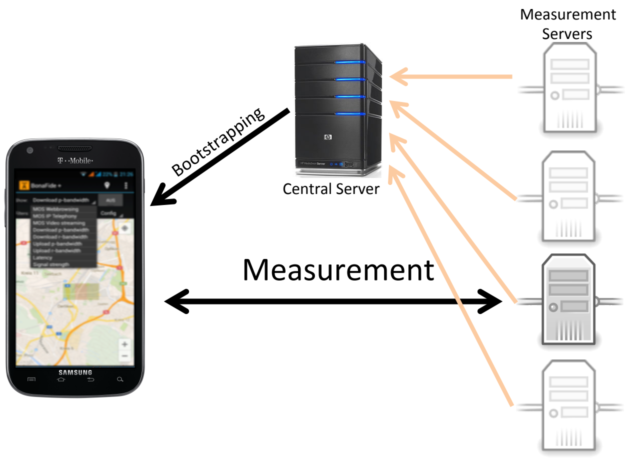

Figure 1: The architecture of the BonaFide+ measurement application.

The overall architecture of BonaFide+ can be seen in Figure 1. The central server is used to perform bootstrapping of the client application. Upon startup, the client application contacts the central server to obtain a list of currently active measurement endpoints. Based on this list of servers, the application automatically chooses the geographically nearest measurement server, however, the user can override this choice and select any measurement server manually as well. Before starting a measurement cycle, the application contacts the chosen measurement server for a list of supported protocols. Once users choose the protocols that they are interested in measuring, a measurement cycle starts.

During the measurement cycle the performance of random and protocol data flows is recorded, along with other network metrics. All obtained results are stored on the central measurement server.

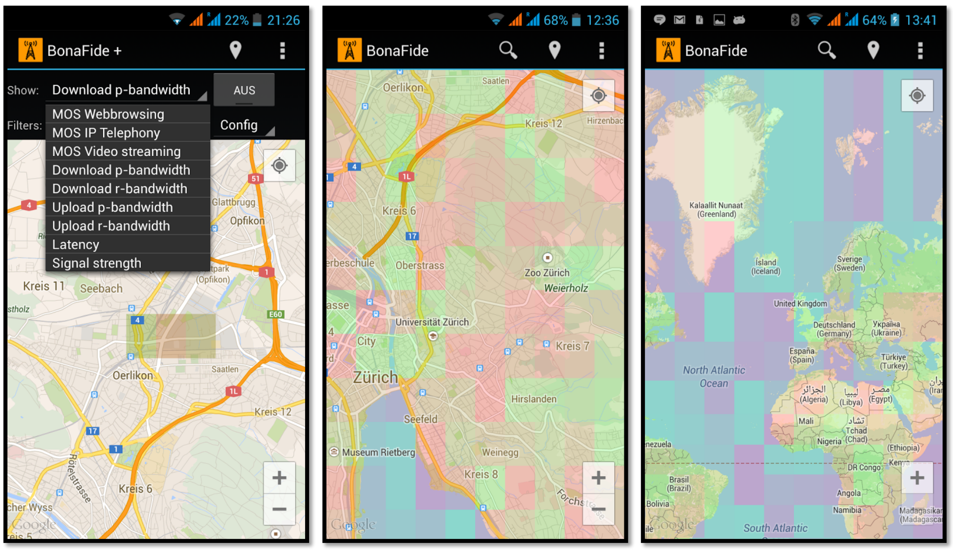

Figure 2: A sample view of the color coded and filtered MOS value results presented to the user (Green = excellent, red = poor).

Results stored on the measurement server can be displayed to the user within the mobile application. An overview of what the displayed results look like is shown in Figure 2. The user is able to filter results based on specific protocol type, protocol or random flow bandwidth, latency and signal strength in order to obtain an idea of what the network around them looks like. Colors corresponding to the MOS scores are displayed (green being excellent and red being bad performance) in squares that are bound to a specific region of the displayed map.

Protocol(s)	**Effect**	**QoS Parameter**	**min**	**max**	x_0	$MOS = 3$	$MOS = 5$	m^-	m^+	w_k
HTTP	increasing	downlink throughput	$0\ Mbps$	$50.1\ Mbps$	$1330\ Kbps$	-25%	$+100\%$	2.41	2.58	75%
	decreasing	latency	$1\ ms$	$2000\ ms$	$523\ ms$	$+15\%$	-50%	10.17	6.29	25%
Flash, RTSP	increasing	downlink throughput	$0\ Mbps$	$50.1\ Mbps$	$1.5\ Mbps$ 480p	-20%	$5\ Mbps$ 720p	3.11	1.49	100%
SIP, RTP	increasing	uplink throughput	$0\ Mbps$	$12.7\ Mbps$	$8\ Kbps^3$	$5.3\ Kbps^2$	$64\ Kbps^3$	1.67	0.86	25%
	increasing	downlink throughput	$0\ Mbps$	$50.1\ Mbps$	$8\ Kbps^3$	$5.3\ Kbps^2$	$64\ Kbps^3$	1.67	0.86	25%
	decreasing	latency	$1\ ms$	$2000\ ms$	$150\ ms^4$	$+50\%^3$	-50%	10.17	2.16	50%

Table 4: The relationship between QoS parameters and MOS values used by BonaFide+ to obtain overall QoE representation.

The calculation of the MOS values from recorded QoS data is based on the approach described in [1]. The relationship between MOS scores and QoS data used by the BonaFide+ application can be seen in Table 4. Here, *increasing* refers to those parameters that positively effect the QoE when their value increases; on the other hand *decreasing* refers to parameters that negatively effect the QoE when their value increases. *Min* and *Max* refer to the maximum and minimum values possible within the model used by BonaFide+. If the value exceeds the maximum provided in the table, then it represents the best or worst QoE, depending on whether the parameter has increasing or decreasing effect on QoE. x_0 refers to the performance required by a specific parameter in order to assign a MOS value of 4 to it. $MOS = 3$ and $MOS = 5$ refer to the amount of percentage increase or decrease required in the respective parameter's value so that the MOS of that particular protocol's QoE would correspond to a MOS of 3 and 5 respectively. m^- and m^+ refer to the rate of increase and decrease of that specific parameter in the mathematical model used for the MOS value calculation [1]. Finally, w_k, refers to the weight that each parameter contributes to obtaining a generic overall MOS for a specific protocol.

2.4.2 Development Activities

BonaFide+ has undergone significant development work over the course of the second year of the project. The main developments are:

- Complete overhaul of the previous BonaFide Android application user interface.
- Implementation of a map that displays QoE representation in colors ranging from red to green, based on equivalent MOS scores.
- Ability to filter protocols for displaying results on the map.
- Reorganization of the test methodology so that users can select which protocols to test for and the number of cycles they wish to contribute.
- Addition of the ability to run tests periodically in the background, i.e., even when the application is closed.
- Inclusion of QoE deduction algorithm from QoS measurements, based on the description presented in [1].
- Extended the Android application to support multiple measurement endpoints.
- Added feature to present users with available measurement servers, and allow them to pick one manually, or the geographically closest one automatically.

- Creation of a central server to track currently online measurement endpoints and also to act as a repository of collected QoS measurements.
- Packaging and debugging for release in the Google Play store.
- Creation of a website to provide information regarding BonaFide+, release all collected data and provide source code.

The ongoing work involves further development of the Android application so that data consumption while performing tests in cellular networks is further reduced. Development of the website and documentation is a priority so that a possible inclusion in the Measurement Lab (M-Lab) project [4] infrastructure can be requested. Presentation of collected results on the website is also an important activity that will be performed over the third year of the project.

All components of the project are released under the terms of a BSD license.

2.5 Open Source Packages Dissemination

2.5.1 BonaFide+

Besides the development of the BonaFide+ application (see Section 2.4.2), some dissemination activities have also taken place during the second year of the project.

Christos Tsiaras (one of the developers of BonaFide+) presented a paper titled “Towards Evaluating Type of Service Related Quality-of-Experience on Mobile Networks” at the 7th IFIP Wireless and Mobile Networking Conference (WMNC 2014) in Vilamoura, Portugal [1]. This paper discussed the protocol flow emulation approach adopted by BonaFide+ and the QoE related MOS value derivation from QoS measurements.

The first version of the developed application has also been packaged and released in the Google Play store so that it may be downloaded and installed by users. Following release, information about the application has been spread via social networks to prospective users. This initial phase has attracted almost 2000 measurement results, as of October 12, 2014.

A website describing the application, providing the source code and anonymized collected data was also setup[5] in order to attract additional users, developers and also target inclusion of the project in the Measurement Lab (M-Lab) [4] infrastructure.

An initial dialogue has also been opened with OpenSignal Inc.[6], a company specializing in creating a comprehensive database of cell phone towers, cell phone signal strength readings, and Wi-Fi access points around the world. The purpose of this dialogue is to investigate possible collaboration regarding usage of BonaFide+ QoE mapping approach in their application and also about sharing data gathered by the current OpenSignal application.

2.5.2 FNSS

Lorenzo Saino (the developer of FNSS) held a tutorial on “Fast Network Simulation Setup” at the 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2014) in Brno, Czech Republic.

[5]http://www.bonafide.pw/
[6]http://www.opensignal.com/

The lab was divided into three parts. In the first part, participants learned various models and datasets of networks topologies. They also learned the most commonly used models to assign link capacities, delays and buffer sizes and how to synthetically generate realistic traffic matrices. The second part provided an overview of the FNSS toolchain. Participants used a "walk through" manual to learn how to install and configure FNSS. The manual also provided information about its main features. The last part of the tutorial took participants through live coding examples, in order to easily generate complex simulation scenarios and deploy them on a number of different simulators or emulators.

The tutorial had the following structure:

- Modelling networks and traffic
  - Network topology models and datasets
  - Assignment of link capacities, delays, weights and buffer sizes
  - Modelling traffic matrices
- Overview of FNSS
  - Installation and configuration
  - Architecture and features overview
- Live coding examples
  - Create complete simulation scenarios
  - Deploy topologies on mininet platform
  - Deploy topologies and traffic matrices on custom-built simulators

We are aware that FNSS is being used internally by a commercial CDN provider to deploy scenarios for simulating complex configurations before deployment.

2.5.3 SSHCure

Efforts were undertaken during the second year of the project to disseminate information about SSHCure to the community at large. An article about SSHCure was published on the RIPE Labs blog [5]. This article was quite useful in attracting new users and it also generated interest in the tool on Reddit [6]. A poster regarding SSHCure was also presented at the TERENA TNC 2014 conference [7].

We are also aware of several SSHCure users that range from Web hosting companies to university campus networks and even nationwide backbone networks. Unfortunately, the identities of these users cannot be disclosed due to privacy concerns.

3 Joint Security Lab

The joint security lab consists of infrastructure such as multiple honeypots, malware analysis chains, measurement probes, and a number of traffic traces hosted by some consortium members that can be accessed by other project partners. The details of the facilities included in the joint security lab were provided during first year of the project. Since that time, some further infrastructure developments have taken place and the facilities of the joint security lab are actually being used by PhD student collaborations.

Institution	**Facilities**	**Details In**
University of Twente	Network Traces NetFlow and IPFIX Flow Export Honeypots	D1.1 **D1.2**
INRIA Lorraine	Network Telescope Malware Analysis Chains Honeypots Home-brewed Crawler	D1.1
University of Federal Armed Forces Munich	Unfiltered Research Network Network Telescope Honeypots	D1.1 **D1.2**
iMinds	Network Traces	**D1.2**

Table 5: List of project partners contributing to the joint security lab and the types of facilities they provide access to.

An overview of the partner institutions providing facilities for the joint security lab can be found in Table 5. With the integration of facilities provided by four FLAMINGO partners, we have met the milestone “After one and a half year, at least four FLAMINGO partners will participate within a joint security lab.” (Task 1.2).

Details regarding the infrastructure developments that took place in the second year of the project, and an overview of the publications resulting from PhD student collaborations that used joint security lab facilities, are provided below.

3.1 Infrastructure Developments

3.1.1 Facilities at the University of Twente

The University of Twente has been working on setting up an infrastructure to study Booters on the Internet, i.e., services providing DDoS on demand [8]. Towards this goal initially a packet capture devices was installed at the central University of Twente gateway, as shown in Figure 3. However, problems related to resource usage were encountered with this setup, thereby leading to incorrect observations. This setup is corrected by using a new setup that places an INVEA Tech 80 Gbps capture device at a large European Internet Exchange, as shown in Figure 4.

Although this setup is intended to capture traffic traces, it does not do anything unless a specific measurement is explicitly enabled. The machine is able to export both flow data and packet captures. As such, partners who wish to access such data from a large European Internet Exchange can request for a specific measurement to be executed.

In addition, during the second year of the project, the following infrastructure developments have taken place at the University of Twente:

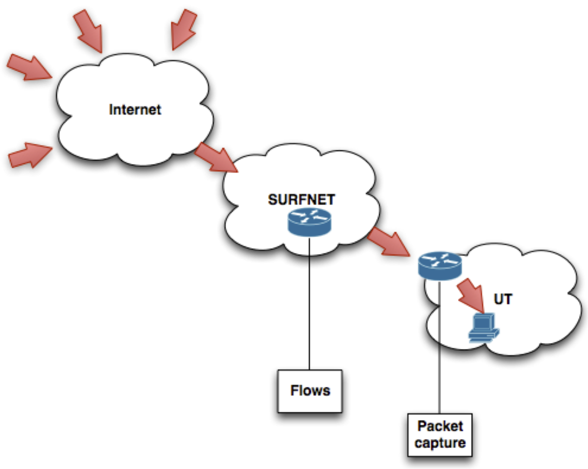

Figure 3: An overview of the initial network setup used to study the DDoS as a service phenomenon.

- An OpenFlow test-bed infrastructure has been deployed at the UT campus network, consisting of multiple OpenFlow-enabled switches. Consortium members can visit UT to get hands-on experience with the equipment.
- A 250 TB storage platform has been setup to store various datasets, ranging from the traces collected at UT and obtained from various third-parties.

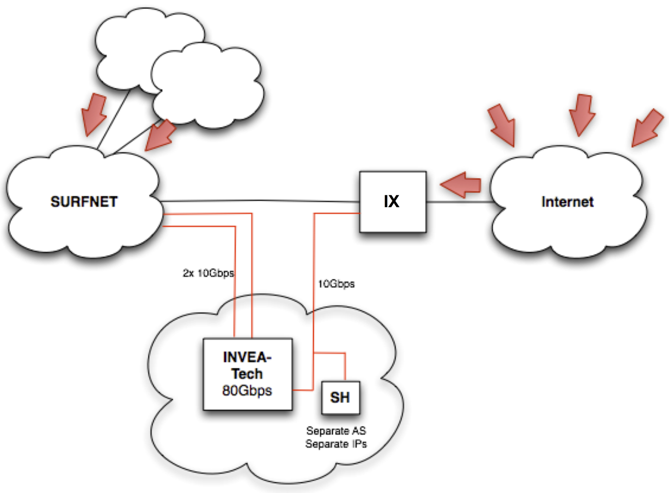

Figure 4: An overview of the network setup used to perform traffic measurements and captures at a large European Internet Exchange.

FLAMINGO partner institutions are able to request access to the infrastructure deployed at the University of Twente, or to the data that is collected and available there.

3.1.2 Facilities at the University of Federal Armed Forces Munich

The University of Federal Armed Forces Munich (UniBwM) operates a research network environment that is, in contrast to the other networks at the university, not filtered. One of UniBwM's joint security lab networks will host a network telescope. The other two networks will imitate a regular office-like environment and a production network. All three networks are protected by a firewall. NetFlow data collectors are placed on both sides of the firewall.

The following hardware and software components are used as part of UniBwM's joint security lab infrastructure:

- NetOptics WireTAP - Regeneration TAP
- 2 Cisco WS 6503E as Netflow v5 Exporters
- 2 HP 5304XL as sFlow Exporters; sampling rates of 1:1, 1:50, 1:100
- Cisco WS2966S 24 port switches - port mirroring
- Cisco IPS 4345
- Raritan Dominions SX - console server
- HP 3400CL Switch
- Suricata IDS v. 1.4.7
- Snorby
- nfdump and nfsen
- FlowMatrix

UniBwM will also purchase several additional hardware components usable for the joint security lab. These hardware components will be purchased before the end of 2014 and will be fully operational in 2015. The aim of purchasing this hardware is to analyze huge data sets derived from their local security lab. Within FLAMINGO this hardware can also be used by FLAMINGO partners for pattern matching or various big data analysis. Because only hardware components are purchased, no restrictions about deployable software components exist. The overall solution consists of two BladeCenter chassis, of which one of them is able to host 14 compute nodes. Each compute node is equipped with 256 GB RAM and two 500 GB SSDs.

All compute nodes will be connected within the blade center via a 10 Gbps switch to allow databases over multiple compute nodes with shared caches. Also both BladeCenter chassis will be connected with each other using two 40 Gbps ports. In addition, each chassis will be connected to two SAN storage arrays with an entire storage capacity of 108 TB.

FLAMINGO partners are able to access NetFlow data, logfiles from various IDSs (e.g., Suricata) and interactive honeypots (e.g., Sebek, Argos, and if possible, a hybrid approach) via this infrastructure.

3.2 Data Traces: Sources, Sharing

3.2.1 Facilities at the University of Twente

Several additional data traces are now being collected at the University of Twente. These can be summarized as follows:

- Data feeds are now collected on a periodic basis to allow for long-term analysis, including the following:
  - Alexa's list of the top million domain names (daily) - `http://s3.amazonaws.com/alexa-static/top-1m.csv.zip`.
  - MaxMind (daily) - `https://www.maxmind.com`.
  - Spamhaus: SBL, XBL, PBL, DBL (every 15 minutes) - `http://www.spamhaus.org`.
  - PSBL: spam (hourly) - `http://psbl.org`.
  - APEWS:spam (hourly) - `http://apews.org`.
  - OpenBL: SSH scanners (hourly) - `http://www.openbl.org`.
  - Passive DNS: DNSDB (`http://www.dnsdb.info`), CERT.at.
- Several anonymized datasets have been published on the SimpleWeb [9], including DNS and SSH datasets with log files included as well [10].

3.2.2 Facilities at iMinds

iMinds is able to provide access to traces collected from a major ISP's content delivery network. The trace contains information regarding requests to a Video on Demand service operated by a leading European telecom operator.

3.3 Usage of Data Traces

The data collected by the joint security lab has been utilized by several PhD student collaborations during the second year of the project. An overview of these collaborations can be found in deliverable D5.2 and D6.2.

The following publications result from PhD student collaborations that utilized facilities made available by the joint security lab during the second year of the project:

[1] Mario Golling, Rick Hofstede, and Robert Koch. Towards Multi-layered Intrusion Detection in High-Speed Backbone Networks. In *Proceedings of the NATO CCD COE 6th International Conference on Cyber Conflict (CyCon)*, Tallinn, Estonia, 2014.

[2] Mario Golling, Robert Koch, and Gabi Dreo Rodosek. From Just-in-Time Intrusion Detection to Pro-Active Response by Means of Collaborated Cross-Domain Multilayered Intrusion Detection. In *Proceedings of the 9th International Conference on Cyber Warfare and Security (ICCWS)*, West Lafayette, IN, USA, 2014.

[3] Maxim Claeys, Daphne Tuncer, Jeroen Famaey, Marinos Charalambides, Steven Latré, Filip De Turck, and George Pavlou. Proactive Multi-tenant Cache Management for Virtualized ISP Networks. In *Proceedings of the 10th International Conference on Network and Service Management (CNSM)*, Rio de Janeiro, Brazil, 2014.

[4] Maxim Claeys, Daphne Tuncer, Jeroen Famaey, Marinos Charalambides, Steven Latré, Filip De Turck, and George Pavlou. Towards Multi-tenant Cache Management for ISP Networks. In *Proceedings of the European Conference on Networks and Communications (EuCNC)*, Bologna, Italy, 2014.

[5] Rick Hofstede, Pavel Celeda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto, and Aiko Pras. Flow Monitoring Explained: From Packet Capture to Data Analysis with NetFlow and IPFIX. *IEEE Communications Surveys and Tutorials*, PP(99), 2011.

[6] Rick Hofstede, Luuk Hendriks, Anna Sperotto, and Aiko Pras. SSH Compromise Detection using NetFlow/IPFIX. *ACM SIGCOMM Computer Communication Review*, 44(5), 2014.

[7] Gijs van den Broek, Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras. DNSSEC meets real world: dealing with unreachability caused by fragmentation. *IEEE communications magazine*, 4(52), 2014.

[8] Roland van Rijswijk-Deij, Anna Sperotto, and Aiko Pras. DNSSEC and its potential for DDoS attacks. In *Proceedings of the ACM Internet Measurement Conference (IMC)*, Vancouver, BC, Canada, 2014.

4 EmanicsLab

Activities related to EmanicsLab infrastructure and its usage over the second year of the project can be summarised as follows;

- EmanicsLab policy changed with regard to slice sizes. According to the new policy, slices will have unlimited disk space.
- The EmanicsLab management site resides under a new IP address.
- Two new partner institutions added to EmanicsLab.
- New EmanicsLab distribution release, version MyPLC (based on LXC).
- Investigation of IPv6 support on EmanicsLab slices.

In the second year of FLAMINGO, two additional nodes were setup at the University of Geneva in Switzerland. Currently, the EmanicsLab network consists of 24 nodes at 12 sites across Europe, as shown in Figure 5.

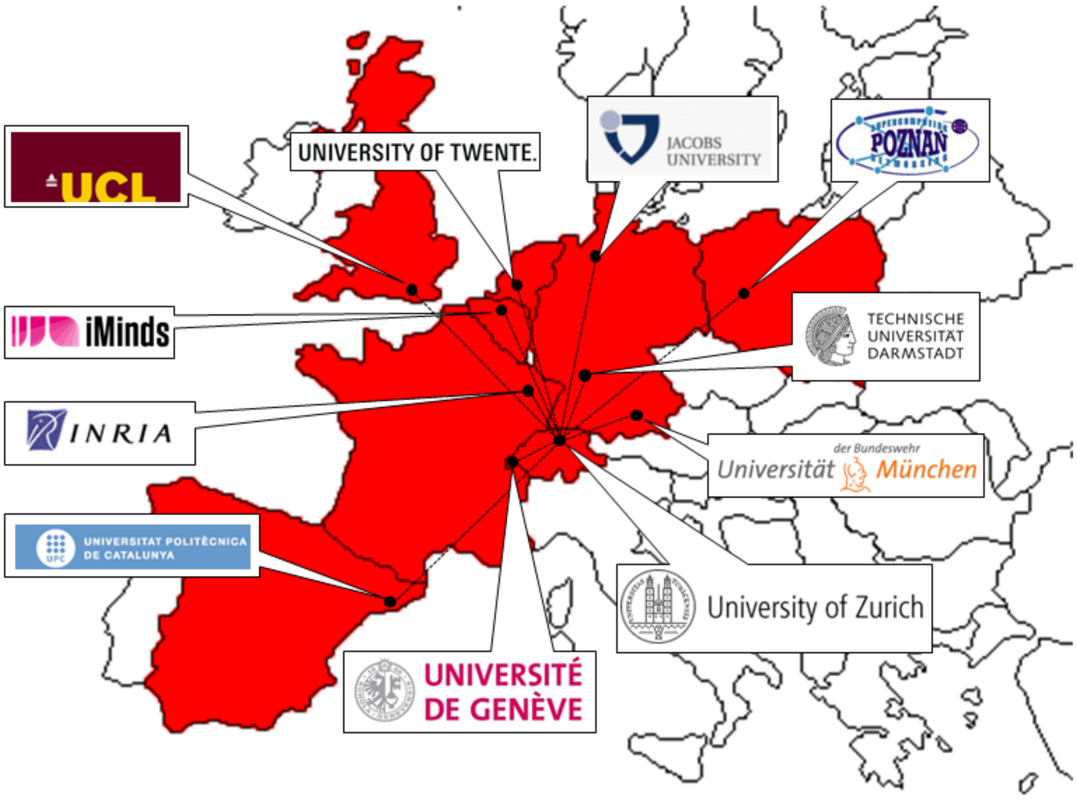
UCL
UNIVERSITY OF TWENTE.
JACOBS UNIVERSITY
POZNAN
iMinds
TECHNISCHE UNIVERSITÄT DARMSTADT
INRIA
Universität der Bundeswehr München
UNIVERSITAT POLITÈCNICA DE CATALUNYA
University of Zurich
UNIVERSITÉ DE GENÈVE

Figure 5: The EmanicsLab network.

Over the course of the last year, a number of improvements have been implemented as described in Section 4.1. Section 4.2 documents the usage of the infrastructure in the second year of the FLAMINGO project.

4.1 EmanicsLab Improvements

Besides maintenance of the EmanicsLab infrastructure itself, the following work has been completed during the first year of the project.

4.1.1 Migration to Linux Containers instead of Linux VServers

A new software release has been rolled out in order to support Linux Containers (LCX) instead of Linux VServers. The change to LXC was performed since LXC is now supported by the mainline Linux kernel, unlike VServers. The new release also updates the Fedora operating system to version 20 (Heisenbug). This also fixed a number of security problems (e.g., the NTP servers in the old Fedora version could be used in amplification attacks).

EmanicsLab migration to the new version of PLC was initiated on March 19, and was completed once all participating institutions installed a new image on dedicated EmanicsLab servers. The new PLC version offers more stable Slice Management within EmanicsLab. The slivers can now run within LXCs and also support a more recent Fedora version. Instead of sharing the address space with all slivers on a node, another benefit of the LXC approach is that each sliver will have its own (NATed) private IP address, while still being able to bind services to the node's public IP address. Since not all applications require access to a node's public IP address, this should allow for more applications to run on the EmanicsLab nodes.

The EmanicsLab website (`www.emanicslab.org`) migration occurred on March 14, 2014. On that day EmanicsLab operated in a limited mode, since users were not be able to create new slices or add new users. Only the existing slices continued to operate on the distributed EmanicsLab nodes.

This migration effort was led by Prof. Dr. David Hausheer of Technische Universität Darmstadt in collaboration with the Communication Systems Group (CSG) at the University of Zurich (UZH).

4.1.2 Bug fixes for Linux Containers

Following the migration of EmanicsLab from vservers to LXCs, a few bugs were discovered. The bugs and their resolutions are provided below.

100% CPU utilization. In some scenarios it was discovered that the container would cause the host machine to reach 100% CPU utilization. The bug was non-deterministic and could not be easily reproduced. Research into the problem finally discovered, however, that the problem was caused by the `systemd` process internal to slices. As such, nodes would reach a 100% CPU utilization only when a slice was running on them. Disabling the pre-scheduled cleanup of temporary files via the `systemd` process within the slice was discovered as an initial workaround to the problem.

A final solution to the problem was to upgrade the EmanicsLab server software to a release based on the Fedora 20 version. A new upgrade cycle for all EmanicsLab nodes was once again kicked off. This second migration was finally completed on August 03, 2014. The 100% CPU utilization problem has not been observed after this second upgrade.

Bugs in the `bind_public` module. Since each EmanicsLab sliver is now a Linux container, it does not receive access to the host server's public IP by default. In order to obtain access to the public IP, an application must specifically bind to the public IP address and port of its choice. This triggers a NATting between the private address of the container and the public IP address of the node. Internally, within the Linux kernel, the `bind_public` method is called in order to set this up.

Due to experimental IPv6 support being enabled in the EmanicsLab distribution, it was discovered that the `bind_public` module did not always function as desired. Applications that were attempting to bind to the public IP address would receive a socket, however, the appropriate NAT bindings would not be setup. As a result, access to the host server's public IP address was no longer available.

An initial workaround was found for this bug. To successfully bind with the server's public IP address, it was required that, while attempting to open a socket, the application either explicitly choose an IPv4 address to bind to or declare preference for IPv4. While this is a temporary workaround, a final fix for this bug is being currently investigated. When a solution is found, the fix for the `bind_public` module will be pushed upstream to the MyPLC project.

4.1.3 Support for IPv6 in Linux Containers

Initial support for IPv6 has been investigated and implemented for EmanicsLab. The mainline MyPLC distribution needs to be updated in order to enable IPv6 support automatically. As such, over the course of the second year of the project, development effort was focused towards initially enabling IPv6 support on nodes and slices manually.

The MyPLC `tags` feature is used to enable manual IPv6 support in EmanicsLab. Assuming that the IPv6 prefix `2001:67c:16dc:1301::/64` is used to achieve host node connectivity and management, the following special tags must be added in the node's configuration via the PLC web interface:

```
ipv6addr: 2001:67c:16dc:1301::2
ipv6_defaultgw: 2001:67c:16dc:1301::1
```

It must be noted that the MyPLC implementation assumes that a prefix assigned to the node is `/64`. The `ipv6addr` tag defines the node's management IPv6 address and the `ipv6_defaultgw` configures the default gateway.

Following this, the default `libvirt` network must be configured to assign an IPv6 prefix to the slivers, by using the `virsh` tool, which is an interface for managing virsh guest domains. Assuming that the prefix `2001:67c:16dc:1302::/64` is provided to the slivers, the following configuration must be added within the `<network></network>` tags.

```
<ip family='ipv6' address='2001:67c:16dc:1302::1' prefix='64'> </ip>
<ip family='ipv6' address='fe80:1302::1' prefix='64'> </ip>
```

Restarting `libvirt` networking at this point configures the host node's virtual bridge, i.e., `virbr0`, to obtain the appropriate IPv6 addresses. A routing daemon can now be configured to provide routing advertisements to slivers, via the `virbr0` interface.

Over the course of the next year of the project, this manual process will be automated so that changes can be pushed upstream to MyPLC and IPv6 enabled in slivers without the need to manage configuration files.

4.1.4 Improved documentation and user-friendliness

A short survey of EmanicsLab users from the project consortium was performed during the second year, in order to determine how EmanicsLab could be improved. The suggestions during this survey centered around the lack of documentation regarding the usage of EmanicsLab (or that it was outdated). A further complaint was that there was no system for users to know when their slices were expiring, and as such, sometimes these would be deleted without the user realizing that the expiration date was near.

To resolve these issues, the following steps have been taken:

- To improve the quality of documentation, a new EmanicsLab Wiki has been created (`http://wiki.emanicslab.org/`). This Wiki is organized into the following sections:
  - IPv6 Support – a tutorial on how to enable IPv6 manually in slivers is provided.
  - How To – this section of the Wiki provides information on how users can perform important tasks, e.g., binding ports to public IPs.
  - Scripts and Tools – information regarding tools that users of EmanicsLab find helpful.
  - Applications – information regarding applications that are installed on EmanicsLab servers.
- A link to this Wiki site has been added to the main navigation menu of the EmanicsLab website, so that users can locate information more easily.
- A new script was created and setup to send emails to users before their slice expires.

Documentation for EmanicsLab will be improved further over the course of the next year, and further feedback will be sought from users to understand what improvements can be implemented.

4.2 EmanicsLab Usage

Over the course of the second year of the project, EmanicsLab has been used for several purposes. As of October 12, 2014, 30 slices have been created during the year on EmanicsLab. Some of these slices were used for teaching activities, others for research and some for obtaining network measurements. Information regarding some of the more prominent EmanicsLab usage activities is provided in this section.

4.2.1 Jacobs University – Advanced Computer Networks

The Advanced Computer Networks course[7] taught at Jacobs University Bremen during the Spring 2014 semester made use of the EmanicsLab infrastructure for teaching purposes. For the section on Voice Over IP (VoIP), students were provided with individual slices. Within the slices, each student had to setup the Asterisk VoIP server and configured it such that VoIP clients could connect to their servers.

The slices were all located at geographically distributed host nodes of different EmanicsLab partner institutions. This setup was adopted so that the students could also study the effects of latency on VoIP communication, while they called each others' servers.

[7] `http://cnds.eecs.jacobs-university.de/courses/acn-2014/`

4.2.2 Study on DDoS

A study regarding DDoS being offered as a service by Booters [8] was performed by University of Twente (UT). The EmanicsLab infrastructure has been used by a Bachelor's student to study traffic originating at Booters.

The EmanicsLab infrastructure was also used to perform a study regarding the vulnerability of home networks, which use DSL as access technology, to DDoS networks. The EmanicsLab nodes were used to generate traffic that was targeted at the home network of a FLAMINGO researcher. This study also aimed at studying how easy it was to perform amplification attacks, i.e., to examine if ISP networks were deploying network ingress filtering [11] or not.

4.2.3 BonaFide+ usage

The BonaFide+ application was created by consortium members to map type of service based QoE, on the basis of QoS measurements performed in a mobile network. As described in section 2.5.1, the mobile application performs active QoS measurements against a measurement server. In order to ensure that the routing path to the chosen measurement server is as short as possible, the geographically nearest server to a user is chosen. The EmanicsLab infrastructure is utilized by this collaboration so as to achieve a wide geographic distribution of measurement servers in Europe.

5 Virtual Wall and FIRE Facilities

During the “First Year Project Review”, it was established that the Virtual Wall and FIRE Facilities infrastructure have been completely setup. It was also decided that the focus should be shifted towards the usage of these facilities within, as well as outside, the scope of the project.

5.1 Virtual Wall Usage

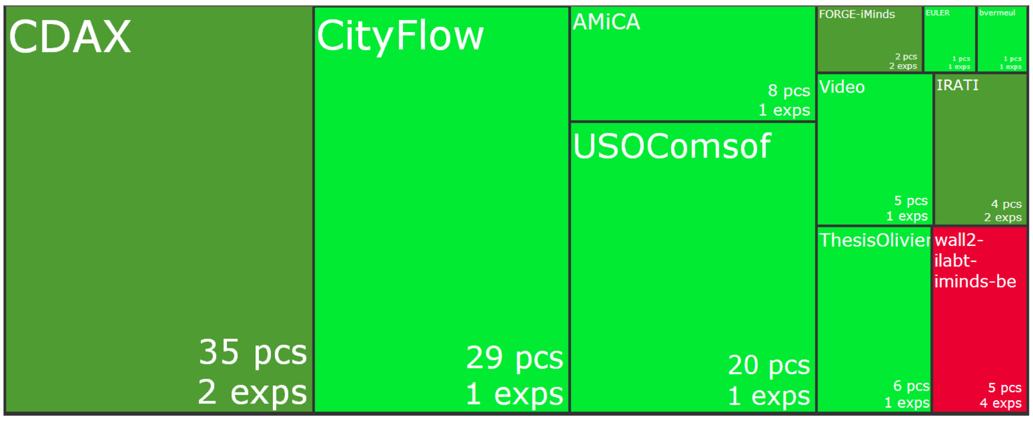

Figure 6: The usage of Virtual Wall facilities. Names of projects and number of nodes and experiments per project are shown.

As of October 12, 2014 there are around 100 users of the virtual wall. Typically about 30 experiments are active at any given time and on average these use about 250 nodes. Figure 6 provides an overview of the current usage on one of the two Virtual Walls.

5.2 Dissemination Activities

Niels Bouten, Maxim Claeys and Jeroen Famaey presented a tutorial on “Deploying OpenFlow Experiments on the Virtual Wall Testbed” at the 8th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2014) in Brno, Czech Republic.

The goal of this hands-on tutorial was to familiarize the participants with the concept of Software-Defined Networking (SDN) in general and with OpenFlow in particular. The tutorial explained OpenFlow’s capabilities to dynamically reroute traffic, guarantee bandwidth, and differentiate flows. Participants were given the opportunity to apply the acquired knowledge by setting up an OpenFlow-based experiment that guarantees the Quality of Service requirements of a networked video application. The experiment was run in a live network setting, facilitated by the Virtual Wall testbed. In the process, the Virtual Wall nodes were assigned different functionalities.

The tutorial covered the following aspects:

- SDN and OpenFlow
  - General introduction to the SDN concept
  - The OpenFlow protocol and architecture
  - Routing and differentiated-service capabilities of OpenFlow

- The Virtual Wall testbed
  - General introduction to large-scale network testbeds and Emulab
  - Overview of the Virtual Wall's functionality
- Hands-on OpenFlow experiment
  - Configuring a network topology
  - Installing and configuring OpenFlow
  - Running a QoS-differentiation experiment

6 Planned and Achieved Progress

Progress within the work package is being achieved as planned. An overview of the actions planned so far within each task, and their current status, can be found in Tables 6 (Task 1.1), 7 (Task 1.2), 8 (Task 1.3) and 9 (Task 1.4).

Action	Planned	Ongoing	Completed	Details In
Identify open source packages developed by project partners			✔	D1.1
Perform internal review and evaluation of open source packages			✔	D1.2
Undertake a second evaluation of open source packages	✔			
Develop at least two software packages within Y1 and Y2			✔	D1.1, D1.2
Make available two additional open-source packages within Y3 and Y4		✔		D1.2
Perform maintenance and updates to open-source packages		✔		D1.2
Promote joint development of open source packages			✔	D1.2
Disseminate information regarding open source packages		✔		D1.2

Table 6: Creation and maintenance of open source software (Task 1.1) - Planned and achieved progress.

Several open source software development initiatives (Task 1.1) are already taking place within FLAMINGO, as can be seen in Table 6. During the first year of the project, open source software packages maintained by the project partners were identified. In the second year of the project, these packages have undergone an internal review process by the FLAMINGO PhD students. This process has been useful in identifying improvements that can be made in the software packages. It has also helped more partners in the project aware about the functionality of the software packages developed within the project. A detailed description of the software evaluation can be found in Section 2.2. Since the package maintainers found feedback useful, a second round of open source package evaluations is currently planned for the third year of the project.

The open source packages, FNSS and NFQL have evolved significantly during the first two years of the project. As such, the project has met the SMART objective to develop two FLAMINGO open source software packages within the first two years. The project is also well on its way to meet the other SMART objective to make available two additional open source packages sponsored by FLAMINGO in the remaining years. Development work on a new open source package, BonaFide+, has already started towards this goal. Details regarding this new open source package can be found in Section 2.4. Since BonaFide+ is being developed jointly by Jacobs University Bremen, University of Zürich and Universität der Bundeswehr München, the goal to promote joint development of open source packages has also been met by the project.

Certain actions related to open source packages are classified as ongoing since it is expected that these actions will be performed continuously over the lifetime of the project. While open source package maintenance and update related tasks have been performed in the second year of the project (see Section 2.3), these activities are likely to continue over the course of the project's lifetime. As such, maintenance and updates to open source packages is an ongoing activity. During the second year of the project several dissemination activities related to the open source packages sponsored by FLAMINGO were performed (see Section 2.5). The consortium views this as an ongoing task since some activities are currently underway and additional activities might be undertaken if the appropriate avenue presents itself.

The creation and operation of a joint security lab (Task 1.2) is proceeding as planned. An overview of the status of current actions can be seen in Table 7. The initial SMART objective of using the high security lab infrastructure at INRIA Nancy as the starting point for the joint security lab was already met in the first year of the project. Three partners had provided infrastructure towards the joint security lab in the first year. With the inclusion of data traces available at iMinds, the joint

Action	Planned	Ongoing	Completed	Details In
The high security lab infrastructure at INRIA Nancy to become starting point for a joint security lab			✔	D1.1
After one and a half years, at least four FLAMINGO partners will participate within a joint security lab			✔	D1.1, D1.2
Perform improvements to the joint security lab infrastructure		✔		D1.2
Utilize data traces made available by the joint security lab within the FLAMINGO PhD student collaborations		✔		D1.2

Table 7: Creation and operation of a joint security lab (Task 1.2) - Planned and achieved progress.

security lab now has four partner institutions participating in it. As such, the SMART objective that after one and a half years at least four FLAMINGO partners will participate within a joint security lab has also been met. The goal of the joint security lab was to provide access to relevant data sources to the PhD student collaborations, where they could be utilized. As shown in Section 3.3, data traces from the joint security lab have already been used by PhD student collaborations and have also resulted in publications in the second year. However, since the data from the joint security lab will be available to PhD student collaborations in the remaining years of FLAMINGO as well, this action remains ongoing. The infrastructure available within the scope of the joint security lab at University of Twente has undergone improvements in the second year and several infrastructure improvements are currently ongoing at Universität der Bundeswehr München, as described in Sections 3.1 and 3.2. These infrastructure improvements are likely to be completed in the third year of the project.

Action	Planned	Ongoing	Completed	Details In
Additional nodes to be added by the end of Y3			✔	D1.1, D1.2
Establish new steering committee			✔	D1.1
Deploy open source tools on EmanicsLab			✔	D1.1
Promote usage of EmanicsLab in teaching and research			✔	D1.2
Improve EmanicsLab documentation		✔		D1.1, D1.2
Migrate to Linux containers from vservers			✔	D1.2
Implement initial IPv6 support in EmanicsLab			✔	D1.2
Include IPv6 support by pushing changes to the MyPLC distribution	✔			
Operation and maintenance of EmanicsLab		✔		D1.2

Table 8: Operation and evolution of the distributed EmanicsLab testbed (Task 1.3) - Planned and achieved progress.

The EmanicsLab (Task 1.3) has seen several improvements during the second year of the project. As can be seen from Table 8, the SMART objective of extending EmanicsLab with additional nodes by the end of year three has already been met. This is because a total of four new nodes from two new institutions were added during the first year of the project (see deliverable D1.1) and during the course of the second year two new nodes from the University of Geneva have also been added to EmanicsLab (see Section 4). A new steering committee was established and also open source tools were deployed on the EmanicsLab during the first year of the project. During the second year of the project, as described in Section 4.2, EmanicsLab was used by Jacobs University Bremen for teaching a graduate course, a study on DDoS attacks was performed at University of Twente and the BonaFide+ application jointly developed by project partners is using the EmanicsLab nodes as measurement endpoints. As such, the goal of promoting usage of EmanicsLab in teaching and research has been achieved. Implementation of IPv6 support in EmanicsLab was deferred to the second year because of the impending migration to Linux containers. This migration to Linux containers has been completed in the second year (Section 4.1.1) and initial support for IPv6 has also been included into EmanicsLab (Section 4.1.3).

The current method of enabling IPv6 in EmanicsLab requires multiple manual steps, as such, a new automatic approach is being developed that will be included in the MyPLC Linux distribution used by EmanicsLab. This work is planned to be completed during the third year of the project.

Due to the requests for improving EmanicsLab documentation, several steps have been taken towards achieving this goal, as described in Section 4.1.4. The work on the EmanicsLab Wiki is ongoing, with information regarding changes and new relevant tutorials still being added. It is expected that this documentation will keep evolving over the remaining course of the project and as such this is an ongoing action. The move to Linux containers introduced a few new bugs in EmanicsLab, some of which have been resolved (see Section 4.1.2). More work is needed to resolve some other bugs and the resolutions will then be pushed back to the MyPLC distribution. Due to the ongoing nature of this work, this action is expected to continue over the third year of the project as well. Other maintenance and operation issues related to EmanicsLab are also going to be performed as an ongoing task during the remaining lifetime of the project.

Action	Planned	Ongoing	Completed	Details In
Develop and operate the Virtual Wall and FIRE facilities			✔	D1.1
Encourage usage of the Virtual Wall and FIRE facilities within and outside of FLAMINGO			✔	D1.1, D1.2

Table 9: Integration of existing infrastructures into virtual laboratories (Task 1.4) - Planned and achieved progress.

The Virtual Wall (Task 1.4) operated by iMinds was opened to FLAMINGO researchers during the first year of the project. The goal of developing and operating the Virtual Wall and FIRE facilities has already been met during the first year of the project, as shown in Table 9. The goal for the second year of the project was to improve usage of these facilities within and outside of FLAMINGO. This goal has been achieved, since the Virtual Wall now experiences continuous usage (see Section 5.1) and a tutorial was also presented in the second year to encourage researchers to deploy OpenFlow experiments on the Virtual Wall (see Section 5.2).

Task	Action	Institution	External Funding	
			Percentage	*Source*
Open Source (Task 1.1)	Software updates to SSHCure	UT	25%	Dutch national funding
	FNSS related dissemination activities	UCL	25%	AIMS 2014 Conference
	SSHCure related dissemination activities	UT	25%	Dutch national funding EU FP7 UniverSelf
Joint Security Lab (Task 1.2)	Infrastructure acquisition and maintenance	UT	80%	University
		UniBW	100%	University
	Data traces acquisition and maintenance	iMinds	100%	EU FP7 OCEAN
Virtual Wall (Task 1.4)	Infrastructure setup and maintenance	iMinds	90%	EU FP7 Ophelia EU FP7 BonFIRE EU FP7 Fed4FIRE

Table 10: Actions that were funded jointly with other sources.

All the work reported in this deliverable, except for the actions listed in Table 10, was performed within the scope of the FLAMINGO project. Some amount of the work, percentage listed in Table 10, has been performed by partners in other projects and contexts. However, these actions are reported because the outcome of these actions is utilized within FLAMINGO as well. Within the scope of maintenance and development of open source packages (Task 1.1), 25% of the development of and dissemination activities related to SSHCure were shared with the Dutch national funding agency and also the FP7 UniverSelf project. The FNSS software package development this year has been exclusively within the scope of FLAMINGO. The travel budget for the tutorial presented at AIMS 2014 on FNSS was partly funded by the conference organizers, however, the tutorial preparation was performed within the context of the FLAMINGO project.

The infrastructure procured for the joint security lab (Task 1.2) by University of Twente and Universität der Bundeswehr München was mostly funded by the university where the infrastructure is hosted. Similarly, the data traces made available by iMinds were acquired via the FP7 OCEAN project. However, the usage of these facilities and infrastructure by PhD collaborations has been enabled within the context of the joint security lab. This sharing has also resulted in joint publications.

All work performed on the maintenance and operation of EmanicsLab (Task 1.3) during the second year of the project, except for the migration to Linux containers (LXC), was within the context of FLAMINGO. The migration to LXC involved EmanicsLab partner institutions that are not members of the FLAMINGO consortium. As such, some work was performed by them in coordinating and applying the update on their nodes.

Most of the Virtual Wall (Task 1.4) maintenance is performed by iMinds was within the context of the FP7 Ophelia, FP7 BonFIRE and FP7 Fed4FIRE projects. However, these facilities are made available to the FLAMINGO partners within this project. The dissemination and experimental setup related activities are performed solely within the context of FLAMINGO.

7 Conclusions

This deliverable reports the progress made during the second year on the creation and maintenance of open source software, the creation and operation of a joint security lab, the operation and evolution of the distributed EmanicsLab testbed, and the integration of the existing infrastructures into virtual laboratories.

As detailed in Section 6, the work package WP1 is progressing well regarding its objectives. The SMART objectives listed in Section 1 have almost been reached after the second year of the project. During the coming years, we plan to continue the evolution of the joint infrastructures and the open source projects with the goal to support research collaborations in the FLAMINGO project.

References

[1] Christos Tsiaras, Anuj Sehgal, Sebastian Seeber, Daniel Dönni, Burkhard Stiller, Jürgen Schönwälder, and Gabi Dreo Rodosek. Towards Evaluating Type of Service Related Quality-of-Experience on Mobile Networks. In *Proceedings of the 7th IFIP Wireless and Mobile Networking Conference (WMNC)*, Vilamoura, Portugal, 2014.

[2] Vitali Bashko, Nikolay Melnikov, Anuj Sehgal, and Jürgen Schönwälder. BonaFide - A Traffic Shaping Detection Tool for Mobile Networks. In *Proceedings of the 13th IFIP/IEEE International Symposium on Integrated Network Management (IM)*, Ghent, Belgium, 2013.

[3] Rick Hofstede, Luuk Hendriks, Anna Sperotto, and Aiko Pras. SSH Compromise Detection using NetFlow/IPFIX. *ACM Computer Communication Review*, 44(5), 2014 *(to appear)*.

[4] Measurement Lab. M-Lab: Open Internet Measurement. [Online]. Available: `http://www.measurementlab.net`. [Accessed: October 7, 2014].

[5] Luuk Hendriks. SSHCure: SSH Intrusion Detection Using NetFlow and IPFIX. [Online]. Available: `https://labs.ripe.net/Members/luuk_hendriks/sshcure-ssh-intrusion-detection-using-netflow-and-ipfix`. [Accessed: October 7, 2014].

[6] Rick Hofstede. SSHCure: SSH Intrusion Detection Using NetFlow and IPFIX (NfSen plugin). [Online]. Available: `http://www.reddit.com/r/netsec/comments/27dk6h/sshcure_networkbased_ssh_intrusion_detection/`. [Accessed: October 7, 2014].

[7] Luuk Hendriks. SSHCure: SSH Intrusion Detection using NetFlow and IPFIX. In *Proceedings of the TERENA Networking Conference (TNC)*, Dublin, Ireland, 2014.

[8] Jair Santanna, Anna Sperotto, and Aiko Pras. The DDoS as a Service Phenomenon. In *Proceedings of the Workshop on Management of Large Scale Virtualized Infrastructures: Smart Data Acquisition, Analysis and Network and Service Management in the Future Internet (EUCNC 2014)*, Bologna, Italy, 2014.

[9] Rick Hofstede. Simpleweb: Traces. [Online]. Available: `http://www.simpleweb.org/wiki/Traces`. [Accessed: October 7, 2014].

[10] Rick Hofstede. Simpleweb: SSH Datasets. [Online]. Available: `http://www.simpleweb.org/wiki/SSH_datasets`. [Accessed: October 7, 2014].

[11] P. Ferguson and D. Senie. Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. RFC 2827 (Proposed Standard), May 2000.