NETWORK FIREWALLS DYNAMIC PERFORMANCE EVALUATION AND FORMALISATION

A thesis submitted in partial fulfilment of
the requirements of Edinburgh Napier University,
for the award of Doctor of Philosophy
in the Faculty of Engineering, Computing & Creative Industries

Lionel Saliou

March 2009

“ We are all equal in front of Knowledge, sadly we are unequal in front of Education.”

“ Nous sommes tous égaux face au Savoir, malheureusement nous ne le sommes pas face à l’Éducation.”

— Marie-Renée Saliou Fontaine

“ The only Way for Evils to Triumph is for Good Men to Do Nothing.”

“ La seule condition au triomphe du mal, c’est l’inaction des gens de bien. ”

— Edmund Burke (1729 - 1797)

“ Losing an illusion makes you wiser than finding a truth.”

“ Perdre une illusion nous rend plus sage que de trouver une vérité. ”

— Ludwig Börne (1786 - 1837)

“ The most secure computer is the one that is turned off, and unplugged from the network. ”

“ L’ordinateur le plus sûr est celui-ci qui est éteint, et déconnecté du réseau. ”

— Unknown Author.

Acknowledgements

This Thesis is dedicated to Alexandra Diane Bastin, my partner and keeper of my sanity. Research is definitively a solitary intellectual journey, however I would have never been able to complete it without the support Alexandra provides. Being reminded that there is more to life than research goes a long way. Indeed, during this Thesis write-up we were blessed with the birth of our daughter Anaïs Yukino. On the topic of family, I also want to thank my parents, Marie-Renée and Jean-Claude, for their encouragements throughout the years.

On many accounts this research is a success. It yielded several publications with good reviews, it is often used as basis for undergraduate and postgraduate projects, and is disseminated in several taught programs at Edinburgh Napier University School of Computing. Needless to say that it is all thanks to the guidance from my Director of Studies, Professor William J. Buchanan, and second supervisor, Doctor Josè M. Munoz. This Thesis is testimony to their dedication to scientific progress.

Research students are often stereotyped. Apparently, they spent many years in poorly lighted rooms, and in tiny cubicles. My experience, though, says otherwise, as I was very fortunate to have an office without internal walls. This allowed my colleagues and I to discuss ideas, teach each others about our research fields, and share skills. Hence, this is the opportunity to thank for this Sarah Clayton, Robert J. Irvine, Kevin F. Chalmers, Jamie R. Graves, and Zbigniew Kwecka.

This Thesis marks the end of my journey through the British academic system. This would have never been possible without the help from the lecturers at the Institut Universitaire Technologique Jules Raimu (Nîmes, France), and Ms Christine Julien in particular.

Thanks to you all,

Lionel Saliou

Remerciements

Cette Thèse est dédiée à Alexandra Diane Bastin ma partenaire et gardienne de ma santé mentale. Faire de la recherche est certainement un voyage intellectuel en solitaire, cependant je n'aurai pas pu terminer ce voyage sans le soutient qu' Alexandra m'apporte. Se rappeler qu'il y a bien plus dans la vie que faire des recherches aide beaucoup. En effet, pendant la rédaction de cette Thèse nous avons été bénis avec la naissance de notre fille Anaïs Yukino. En parlant de famille, j'aimerai aussi remercier mes parents, Marie-Renée et Jean-Claude, pour leurs encouragements tout au long de ces années.

Sur bien des points, les recherches effectuées pour cette Thèse sont un succès. Après tout, il y a eu plusieurs publications avec de bonnes critiques, beaucoup de projets de fin d'année sont basés dessus, et enfin, les différentes trouvailles sont disséminées dans plusieurs unités de valeur enseignées à Edinburgh Napier University School of Computing. Bien entendu, rien de tout cela n'aurai été possible sans l'expertise de mon Directeur de Recherche, Professeur William J. Buchanan, ainsi que de mon superviseur en second, Docteur Josè M. Munoz. Cette Thèse est un témoignage de leur passion pour le progrès scientifique.

Les étudiants chercheurs sont souvent engouffrés par les stéréotypes. Apparemment, ceux-ci passent des années dans des salles sombres et leur tout petits cubicules. Je n'ai rien vécu de similaire. Au contraire, j'ai eu la chance de travailler dans une pièce sans mûrs intérieurs. Cela nous a permis, à mes collègues et moi-même, de discuter des idées, nous informer sur les champs de recherche de chacun, ainsi que partager nos compétences. Je prends donc l'opportunité de remercier Sarah Clayton, Robert J. Irvine, Kevin F. Chalmers, Jamie R. Graves, and Zbigniew Kwecka pour cela.

Cette Thèse marque donc la fin de mon périple à travers le système académique Britannique. Cela n'aurai pas été possible sans l'aide de beaucoup d'enseignants à l'Institut Universitaire de Jules Raimu (Nîmes, France), et en particulier Madame Christine Julien.

Merci à toutes et à tous,

Lionel Saliou

Copyright

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institution of learning.

This Thesis has been examined by:
Prof. Steven M. Furnell, Plymouth University
as External Examiner,
and,
Dr. Imed Romdhani
as Internal Examiner.

This research has been conducted under the direction of:
Prof. William J. Buchanan
as of Director of Studies,
and,
Dr. Josè Munoz
as Second Supervisor.

Contents

List of Tables

List of Figures

Abstract

Computer network security is key to the daily operations of an organisation, its growth and its future. It is unrealistic for an organisation to devote all of its resources to computer network security, but equally an organisation must be able to determine whether its security policy is achievable and under which criteria. Yet, it is not often possible for an organisation: to define its security policy, especially to fully comply with the laws of the land; ensure the actual implementation on network devices; and finally audit the overall system for compliance. This thesis argues that one of the obstacles to the complete realisation of such an **I**ntegrated **S**ecurity **F**ramework is the lack of deep understanding, in particular in terms of dynamic performance, of the network devices on which the security policy will be deployed.

Thus, one novelty of this research is a **D**ynamic **E**valuation **E**nvironment for **N**etwork **S**ecurity that allows the identification of the strengths and weaknesses of networked security devices, such as in network firewalls. In turn, it enables organisations to model the dynamic performance impact of security policies deployed on these devices, as well as identifying the benefit of various implementation choices, or prioritisations. Hence, this novel evaluation environment allows the creation of instances of a network firewall dynamic performance model, and this modelling is part of the **I**ntegrated **S**ecurity **F**ramework, thus enabling it to highlight when particular security requirements cannot be met by the underlying systems, or how best to achieve the objectives. More importantly, perhaps, the evaluation environment enables organisations to comply with up-coming legislation that increases an organisation's legal cover, which demands consistent and scientific evidence of fitness *prior* to security incidents.

Dynamic evaluations produce a large amount of raw data and this often does not allow for a comprehensive analysis and interpretation of the results obtained. Along with this, it is necessary to relate the data collected to a dynamic firewall performance model. To overcome this, this research proposes a unique formalisation of the inputs and outputs of the proposed model, and this, in turn, allows for performance analysis from multiple view-points, such as: the increase security requirements in the form of larger rule-set sizes; effects of changes in terms of the underlying network equipment; or the complexity of filtering. These view-points are considered as evaluation scenarios and also have unique formalisations.

Evaluations focused on two types of network firewalls and key findings include

the fact that strong security policy overhead can be kept acceptable on embedded firewalls provided that out-going filtering is used. Along with this, dynamic evaluation allows the identification of the additional performance impact of unoptimised configurations, and such findings complement work that focuses on the logical properties of network firewalls. Also, these evaluations demonstrate the need for scientific rigour as the data show that the embedded and software network firewalls evaluated have different areas of strengths and weaknesses. Indeed, it appears that software firewalls are not as affected as embedded firewalls by the complexity of filtering. On the other hand, the number of rules software firewalls enforce is the main performance factor, especially for high network speeds.

Acronyms

ACL	**A**ccess **C**ontrol **L**ist
AN	**A**ctive **N**etwork
CEOs	**C**hief **E**xecutive **O**fficers
CMA	**C**omputer **M**is-use **A**ct 1990 (UK)
DDoS	**D**istributed **D**enial-**o**f-**S**ervice
DEENS	**D**ynamic **E**valuation **E**nvironment for **N**etwork **S**ecurity
d-Feral	**d**ynamic **F**irewall **E**valuation **R**esults **A**na**L**yser
DMCA	**D**igital **M**illennium **C**opyright **A**ct 2000 (USA)
DMZ	**D**e**M**ilitarised **Z**one
DNS	**D**omain **N**ame **S**ystem
DoS	**D**enial-**o**f-**S**ervice
DPA	**D**ata **P**rotection **A**ct 1998 (UK)
DUT	**D**evice **U**nder **T**est
FPGAs	**F**ield **P**rogrammable **G**ate **A**rrays
FTP	**F**ile **T**ransfer **P**rotocol
GUI	**G**raphical **U**ser **I**nterface
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
IDS	**I**ntrusion **D**etection **S**ystem
IP	**I**nternet **P**rotocol
ISF	**I**ntegrated **S**ecurity **F**ramework
ISPs	**I**nternet **S**ervice **P**roviders
LAN	**L**ocal **A**rea **N**etwork
MAS	**M**ulti-**A**gent **S**ystem

NAT	**N**etwork **A**ddress **T**ranslation
NP	**N**etwork **P**rocessor
ns-2	**T**he **N**etwork **S**imulator - 2
OPNET	**Op**timized **N**etwork **E**ngineering **T**ools
OSI	**O**pen **S**ystems **I**nterconnection
PAT	**P**ort **A**ddress **T**ranslation
PCI-DSS	**P**ayment **C**ard **I**ndustry **D**ata **S**ecurity **S**tandard
PCTCG	**P**rincipal-subordinate **C**onsequence **T**agging **C**ase **G**rammar
PDAs	**P**ersonal **D**igital **A**ssistants
RIPA	**R**egulation of **I**nvestigatory **P**owers **A**ct 2000 (UK)
RoI	**R**eturn-**o**n-**I**nvestment
RPC	**R**emote **P**rocedure **C**alls
RTT	**R**ound **T**rip **T**ime
SAFE	**S**ecure **A** blueprint **F**or **E**nterprise networks
SMEs	**s**mall and **m**edium size **e**nterprises
SNMP	**S**imple **N**etwork **M**anagement **P**rotocol
STP	**S**panning-**T**ree **P**rotocol
STOPE	**S**tandard, **T**echnology, **O**rganisations, **P**eople, and **E**nvironment
TCP	**T**ransmission **C**ontrol **P**rotocol
UDP	**U**ser **D**atagram **P**rotocol
vLAN	**v**irtual **L**ocal **A**rea **N**etwork
VoIP	**V**oice-**o**ver-**IP**
VPNs	**V**irtual **P**rivate **N**etworks
WAN	**W**ide **A**rea **N**etwork
XSLT	e**X**tensible **S**tylesheet **L**anguage **T**ransformations
XSL-FO	XSL Formatting Objects
XML	e**X**tensible **M**arkup **L**anguage
yans	**Y**et **A**nother **N**etwork **S**imulator

Chapter 1

Introduction

1.1 Background

Computer network security incidents are making the news headlines more often, from the theft of customers private and banking details [1, 2], through computer worm outbreaks [3, 4], to politically motivated network outages [5, 6, 7]. An important aspect of a computer network security incident is that they often have global side-effects. Indeed, the conflict between the Russia and Estonia did not only illustrate a modern information warfare tactic; the Russian attack disabled the Estonian computer network infrastructure temporarily, and this disturbed the Internet activities and connectivity around the world. This demonstrates that attacks performed by means of computer network systems can be indiscriminate and that it is unwise for computer network users to rely solely on others to achieve adequate security standards [8].

Many researchers realise the central role that computer network systems play in the modern economy and thus security breaches should be prevented as much as possible. Securing computer network systems against attacks, intrusions, or misuse is a non-trivial task. Thus, many experts advocate addressing this challenge as a process. However, a number of research projects, such as Bakry [9], Danchev [10], Rees et al. [11], and Rodgers [12], among others, do not present evidence that their approaches are applicable outside the area of focus which is often limited to managerial concerns, or operational matters of networked systems. In addition, they do not demonstrate the effect of design decisions on actual devices, or how to verify them. Rather, it would appear that the concept of security as a process applies only within the area of expertise. Furthermore, research that focuses on improving the usability of technologies involved in securing networks often requires additional knowledge, and the methodologies employed are often limited in terms of their scope of application. Finally, key factors are often ignored and this includes the fact that a networked system's mission might change over time and, thus, the type of threats associated. This typically translates into the modification of computer network systems configuration [13, 14] with little regards to the suitability, in terms of security, of the new configuration. Therefore, there is little possibility to identify all the required stages of a security process.

1.1.1 Internet-centered Approach

Experts are also concerned about the lack of adoption of improved techniques, methodologies, or approaches in security. This consensus is the driving force behind projects such as DETER [15] which aims to provide the research community with a large scale test environment, which can emulate the Internet. In turn, experiments and conclusions drawn from these are likely to relate to issues that **I**nternet **S**ervice **P**roviders (ISPs) face [15], and thus enhance the prospect of improved methods being more rapidly adopted. Thus, the findings from the DETER project enhance security at an Internet level, and, evidently, collaboration between organisations plays an important role in the success of such an approach [16, 17, 18]. On the other hand, some researchers are of the opinion that applying increased security at this level could hinder competitiveness, and cause privacy concerns [19, 20].

In addition, some researchers argue that hardening systems [16, 15] is not sufficient. Indeed, there is a need to adapt to real-time situations or on-going threats [21], as it is often demonstrated that damage could be done so quickly that implementing a mitigation procedure manually is often not possible or realistic [17, 22]. To overcome this problem, Zou et al. [22] demonstrated that it is feasible to accurately detect a quickly propagating threat, such as a computer worm. Despite the intrinsic benefit of such a system, Zou et al. argue that it is not sufficient, and they suggests that the output should be acted upon with a relevant counter-measure. Therefore, applying this methodology at the level that the researchers from the DETER project suggest that it requires close collaboration [15], and thus organisations may have to share the administrative burden. This would be highly time-consuming and in contradiction with the following analysis:

> "*... [the] defender will not have sufficient resources to protect every node in a vast network*" — Denning [23, p. 18].

This is not the only shortcoming of this type of approach as Staniford et al. [20] demonstrated that some type of attacks, such as defuse intrusions, are likely to void the benefit of collaborative defence approach. Indeed, the tactic of these attacks involves producing large amount of false information to cover the real targets and intentions of the intrusion.

1.1.2 Challenges from Organisations' Point-of-view

Security is thus a multi-dimensional concept, and those dimensions include among other things: privacy, physical access restriction, application availability, network confidentiality, content integrity, and access policy [15]. Thus, computer network security cannot be achieved with one single methodology, approach or technology, as organisations must consider several, and often conflicting, objectives simultaneously. Designing, implementing and verifying adequate security solutions is challenging,

and it appears that security decisions are often based on assumptions. More importantly, perhaps, security in organisations is typically static and reactive as security issues are often addressed after a successful breach [3, 8]. In other words, there is not enough scientific rigour applied to computer network security [24]. Arguably, this is due to the fact that organisations do not often have the possibility to evaluate the impact of their choices on computer networks, and, in particular, when these are in use; in other words, in terms of dynamic performance.

There is also strong evidence that security threats, such as malware or **D**enial-**o**f-**S**ervice (DoS) attacks, can rapidly affect an organisation, and human intervention is typically ineffective in these circumstances. The time sensitive nature of computer network security can be addressed with the deployment of an automated security system, such as based on **M**ulti-**A**gent **S**ystem technology [25]. Arguably, such an automated system will have the same shortcomings as current approaches which view security as an add-on to the system as opposed to a process. Hence, unless the decision making is improved, it becomes difficult to develop such an automated system that is effective. This enhanced decision making thus requires a deep understanding of the impact of security policy on the devices which are selected to enforce it, such as with network firewalls, for example.

1.2 Research Focus

1.2.1 Research Aim

This thesis focuses on the interest of organisations, and supports the view that, in a networked environment, the security of elements has an impact on neighbouring ones [26]. In return, improving security of one element is likely to contribute to the improvement of security for the overall environment. This thesis further argues that focusing on the improvement of security at organisational level allows for an improved integration of legal requirements, and reduces the obstacles to actual realisation, deployment and implementation of security onto computer network systems. Indeed, organisations often are responsible for the usage that it is made of their networked systems [10, 27].

There can be multiple possibilities to address security challenges, and this often results in variations in the manner security devices, such as network firewalls, are configured. In turn, it is necessary to assess the impact of security policy on security devices while these are in use and for a range of implementation choices. Hence:

> This research aims to develop a novel model which allows for **dynamic performance analysis from multiple view points**.

1.2.2 Objectives

In order to achieve the research aim (Section 1.2.1), this thesis focuses on the following objectives:

1. **Define and construct an efficient dynamic evaluation environment that supports multiple analysis view-points**: the proposed model reflects as much as possible the conditions a security device experiences in a production network. Along with this, there are several configuration parameters to control and many metrics that can be measured, and thus manual data collection is not a viable solution.
2. **Define a methodology to process the data collected during evaluations**: evaluations typically result in a large amount of raw data that need to be analysed in a context of the experimental criteria. One of the key challenges relates to the fact that one evaluation might be relevant to multiple view-points.
3. **Observe network firewall performance and its reaction to different security requirements or evaluation scenarios**: this involves defining typical implementation scenarios whose outcomes are significant within an enhanced security process. These scenarios rely on multiple dynamic performance models to build a consensus on the possible repercussions of these implementations within a computer network system.
4. **Integrate the results from the evaluation into an overall framework**: it is essential to highlight at which stage of the security process that dynamic performance information is relevant. In addition, network firewalls represent only one component of an organisation's security, and thus it is essential to empower organisations with the means to take all relevant security factors into consideration.

1.3 Contribution

This thesis shows that concerns over the performance impact of security policies on network security devices plays a major role in the lack of rigour in the deployment of these security policies. Performance concerns are typically used as a justification for lower security standards [28, 3, 29], or simply ruled out as an issue [30, 31]. Either way, the manner in which such a conclusion is reached is often not rigorous, and to address this shortcoming relevant data, particularly related to when the underlying system is in use, needs to be included as part of the decision [32]. However, to ensure that performance data retains an organisational context, the data must obtained through controlled evaluation conditions [24]. The main drawback of such requirements is that the data gathering process is often lengthy and difficult to be realised manually. The absence of automated security systems strongly supports the hypothesis that most evaluations, within organisations, are carried out manually, or

are narrow in terms of scope [33]. Hence, the knowledge gained is typically limited and thus insufficient to identify the weaknesses that intruders might exploit. This issue applies to the cornerstone of organisations' security: network firewalls. In order to address this issue, this thesis presents an automated **D**ynamic **E**valuation **E**nvironment for **N**etwork **S**ecurity (DEENS) (Section 4.3) and, in turn, DEENS allows for collecting data which serve to create a model for the dynamic performance of network firewalls (Section 4.4).

One of the main challenges of dynamic performance evaluation is the large amount of data that is generated (Sections 4.3.2, and 4.5.2). To solve this, Chapter 5 presents unique formalisations of the inputs and outputs of the proposed model for network firewalls dynamic performance (Section 4.4), and of evaluation scenarios (Section 5.2, and Section 5.3.3). This, in turn, allows for the creation of a software application, named **d**ynamic **F**irewall **E**valuation **R**esults **A**na**L**yser (d-FERAL) (Section 5.4), that identifies, processes, and presents the relevant collected data. It then becomes possible to assess the effect of a network firewall configuration on dynamic performance, such as when the filtering direction is changed, for example. In effect, DEENS creates instances of the dynamic performance model for network firewalls. Hence, this provides a contribution and a new approach to assessing the impact of security policies onto network firewalls that is not present in the literature.

To illustrate the benefits of such a methodology, Chapter 6 presents an analysis of results obtained for evaluation scenarios, such as: the increased security requirements in the form of: larger rule-set sizes; effects of changes in terms of the underlying network equipment; prioritisation of firewall rules; and the complexity of filtering. Key observations include the fact that the rule-set size is a key factor in the dynamic performance of network firewalls (Section 6.2), and that it is possible to mitigate the effects by re-ordering the rules within the firewall rule-set (Section 6.3).

The evaluations are carried on two types of network firewalls: embedded (Cisco packet filter IOS firewall), and software-based (Linux Netfilter firewall). Overall, it is not possible to deploy as many rules on the software network firewall compared to the embedded one, however the embedded firewall is not without weaknesses. Indeed, the filtering complexity has a significant impact on the dynamic performance of the embedded firewall, whereas this aspect does not affect the software firewall (Section 6.4.2). Results show that, most times, there is little measurable performance overhead when evaluations are carried out for slow network speeds, however the filtering complexity impacts the dynamic performance of the embedded firewall even for relatively slow network speeds (Section 6.4). In addition, the advantages of dynamic evaluation include the fact that it is possible to identify the fail-over conditions for different configurations. Hence, dynamic performance models can help prevent the selection of a device that would cease to operate properly once deployed in the organisational network (Section 6.4.1).

The data produced with the evaluation environment and the information produced, and thus made available through the formalisation, requires a context in which it can be exploited. Along with this, network firewalls are not the only devices affected by performance issues (Section 3.10). Thus, considering the fact that security is a multi-faced concept, Chapter 8 outlines a design of an **I**ntegrated **S**ecurity **F**ramework which relies on the deep understanding of the equipment available, and uses a model of real-life devices. Hence, this framework uses the model for the dynamic performance of network firewalls as part of the decision process, and hence integrates the evaluation environment. This is motivated by the fact that most studies focus on the logical properties of these devices, and performance information typically lacks the required context for organisation to adopt enhanced solutions [15]. Arguably, organisations also need to identify the current capabilities of their system, if adequate security decisions are to be made [14, 34]. In turn, the actual impact of security policies can be measured, at the design phase, and thus contributes to the understanding of the impact of security policies on the overall system. Thus, it allows the identification of the strengths and weaknesses of existing network devices.

1.4 Thesis Structure

The remaining elements of this thesis are organised as follows:

- **Chapter 2 - Theory:** Securing computer network system involves employing multiple approaches and devices, and thus this chapter outlines the manner in which these elements operate. Overall, it gives the background necessary to understand the themes covered in the Literature Review.
- **Chapter 3 - Literature Review:** This chapter provides an analysis of the main challenges in the domain of computer network security using cross-references to current research. Hence, it provides arguments on why it is essential that organisations employ integrated security systems and establishes a framework that promotes computer network security as a process.
- **Chapter 4 - Network Firewall Dynamic Performance Model:** This chapter presents a model for network firewall dynamic performance in which the criteria that influence network firewall configuration is used as inputs and a basis for dynamic evaluation. The outcomes and results of these evaluations can then serve to build an improved formal model for network firewall dynamic performance. This can, in turn, allow for establishing the feasibility of a particular security policy within the means of the organisation.
- **Chapter 5 - Evaluation Scenarios and Associated Analysis Tool:** This chapter describes three key network firewall implementation scenarios. These scenarios require multiple unique instances of the network firewall dynamic performance model to build a consensus on the possible repercussions of these deployments within a computer network system. To this end, the inputs and

outputs of the dynamic firewall performance model are formalised, and this, in turn, allows for the formalisation of the scenarios themselves. The scenario formalisations define the criteria that model instances match when they are considered relevant to an analysis. Hence, it describes an analysis software, **d**ynamic **F**irewall **E**valuation **R**esults **A**na**L**yser (d-FERAL), that implements the necessary functionalities, both in terms of search abilities, and charting. In other words, d-FERAL allows for the analysis of dynamic performance data from multiple viewpoints.

- **Chapter 6 - Scenario-based Evaluations:** This chapter presents an analysis of the evaluation scenarios described in Chapter 5. It provides evidence that it is necessary to know the full extent of network firewall device capabilities as part of an enhanced security process.
- **Chapter 7 - Conclusions and Future Work:** This chapter reflects on the achievements, and limitations of the thesis. It also outlines avenues for future work.
- **Chapter 8 - Integrated Security Framework:** This chapter outlines an **I**ntegrated **S**ecurity **F**ramework which aims to integrate computer network security, from its specification, to its verification once deployed, onto the actual live devices, and then provides formal verification. The design of this security framework relies on the concept of a prescriptive approach that integrates each main areas of expertise, hence promoting security as a process. A key aspect of the **I**ntegrated **S**ecurity **F**ramework is to provide a context for a dynamic evaluation environment in order to obtain performance data on the strengths and weaknesses of network firewalls.

1.5 Peer-reviewed Publications

The work conducted in this research has yielded several international publications, where each publication represents a milestone in the research process, such as with the publication of a design for an **I**ntegrated **S**ecurity **F**ramework, or the presentation of some results that lay the foundations for the dynamic evaluation environment. The details of these publications are as follows:

- **L. Saliou**, W.J. Buchanan, J. Graves, and J. Munoz, "*Scenario Analysis using Out-of-line Firewall*", published in the proceedings of the 6^{th} European Conference on Information Warfare and Security, Shrivenham, United Kingdom, pp 227 – 235, July 2 – 3, **2007**.
- **L. Saliou**, W.J. Buchanan, J. Graves, and J. Munoz, "*Analysis of Firewall Performance Variation to Identify the Limits of Automated Network Reconfigurations*", published in the proceedings of the 5^{th} European Conference on Information Warfare and Security, Helsinki, Finland, pp 205 – 214, June 1 – 2, **2006**.
- **L. Saliou**, W.J. Buchanan, J. Graves, and J. Munoz, "*Novel Framework for Automated Security Abstraction, Modelling, Implementation and Verification*", published

in the proceedings of the 4^{th} European Conference on Information Warfare and Security, Glamorgan, United Kingdom, pp 303 – 311, July 11 – 12, **2005**.

- Z. Kwecka, W.J. Buchanan, D. Spiers, and **L. Saliou**, "*Validation of* $1 - N$ *OT Algorithms in Privacy-Preserving Investigations*", published in the proceedings of the 7^{th} European Conference on Information Warfare and Security, Plymouth, United Kingdom, pp 119 – 127, June 30 – July 1, **2008**.
- S. Doughan and **L. Saliou**, "*An information economy built on sand*", Expert Witness Institute Summer Newsletter, pp 14 – 15, **2008**.
- J. Graves, W.J. Buchanan, **L. Saliou**, and J. Old, "*Towards a Framework For Evaluating System Call Data as a Source of Digital Forensic Evidence*", published in the proceedings of the 2^{nd} Conference on Advances in Computer Security and Forensics, Liverpool, United Kingdom, pp 90 – 96, July 12 – 13, **2007**.
- J. Graves, W.J. Buchanan, **L. Saliou**, and J. Old, "*Performance Analysis of Network Based Forensic Systems for In-line and Out-of-line Detection and Logging*", published in the proceedings of the 5^{th} European Conference on Information Warfare and Security, Helsinki, Finland, pp 41 – 50, June 1 – 2, **2006**.
- W.J. Buchanan, J. Graves, **L. Saliou**, H. Al Sebea, and N. Migas, "*Agent-based Forensic Investigations with an Integrated Framework*", published in the proceedings of the 4^{th} European Conference on Information Warfare and Security, Glamorgan, United Kingdom, pp 47 – 52, July 11 – 12, **2005**.
- W.J. Buchanan, and **L. Saliou**, "*Enhanced Methods of Coursework Provision in Computer Networks*", presented at the IEEE International Conference on Information Technology: Research and Education, London Metropolitan University, London, United Kingdom, 28 June – 3 July, **2004**.

Chapter 2

Theory

2.1 Introduction

Computer networks are often essential to organisational prosperity, and often link many organisational assets, such as public presence and data, together. A universal solution to computer network security is not achievable, and organisations must thus often produce their own. This task, though, is non-trivial, and requires the deployment and use of an extensive array of approaches, methodologies, and techniques. Along with this, the principles of computer network security include prevention, detection, and correction. Thus, this chapter describes the manner in which key security devices, such as network firewalls or intrusion detection systems, operate.

The coordination of the configurations and activities of these security systems is paramount, as discrepancies may give intruders the opportunity to compromise systems. Unfortunately, there is no *de-facto* standard to achieve this coordination. Nevertheless, there exists some common approaches that this chapter outlines. Finally, with respect to the fast evolving nature of computer threats, it is essential to enable computer network systems with the ability to evolve rapidly, and this requires technologies that can address the rule-based and static nature of computer networks. **M**ulti-**A**gent **S**ystem, or **A**ctive **N**etwork technologies can thus provide such functionalities and thus their fundamental operation is also presented.

2.2 Organisation Assets

Organisations have many assets and these can be tangible, such as a range of products and services, as well as intangible ones, such as corporate image. Such examples are considered internal assets as an organisation has direct control over them. There are also external assets that emerge as by-products of the organisation operations or sector of activity. External assets include public presence, as well as legal responsibilities. All these assets are linked to each another through computer network systems [35, 36] (Figure 2.1).

Indeed, these allow effective communication, reduce costs in sharing data, enable always-on public presence, and so on [37]. Arguably, without computer network systems, and the extensive inter-linkage of these, many organisations would not operate

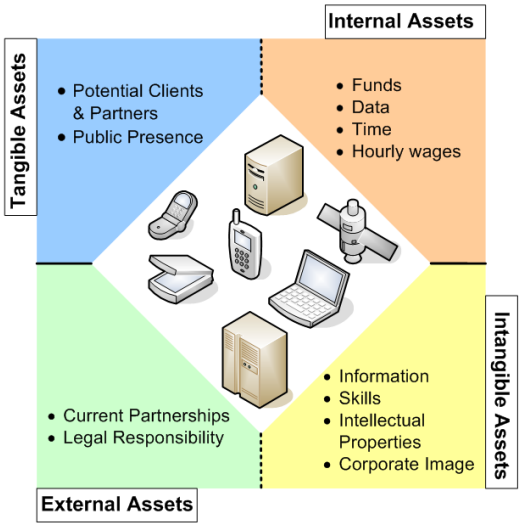

Figure 2.1 – Abstraction of organisational assets

effectively and prosper [38, 39]. Thus, computer network related incidents are events that could damage an organisation, yet, despite the central role of these systems, computer network security is a *relatively new* concern [12, 40]. This is particularly true in the upper ranks of an organisation's hierarchy, such as with senior management, or even **C**hief **E**xecutive **O**fficers (CEOs) [3, 8]. Moreover, it is often difficult to secure computer network systems, and, by the same token, other assets (Figure 2.1), without the support for security from senior management [41, 42, 12]. Part of this challenge resides in the fact that it is a multi-dimensional concept which encompasses issues, such as privacy, application availability, and access policy, amongst others [43]. Along with this, the importance of one dimension over another depends on the organisation's sector of activity [3, 8, 40].

All these factors prevent the creation of universal solutions and organisations can fail to produce proper security standards because they can lack methodologies to assess security-related investments [44, 45]. This is despite the fact that a successful circumvention of computer systems can lead to more than just financial losses [10, 36, 46, 47]. A public image tarnished as a result of a computer network incidents can take years to recover from [47].

2.3 Computer Network Security

2.3.1 Principles

The core principles of computer network security are: prevention, detection, and correction. Figure 2.2 illustrates the location of an asset within these principles, and Table 2.1 provides some examples of technologies that are involved in security. Hence, the purpose of networked system security is to safeguard data from deletion, corruption, and fraudulent access, amongst others. It is also about ensuring that services

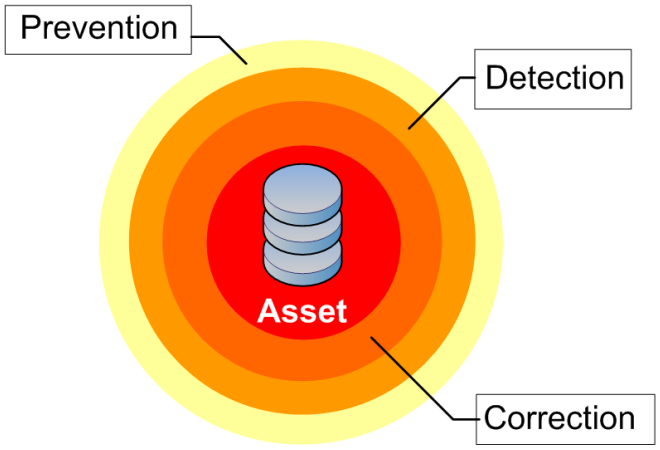

Figure 2.2 – Security layers

offered by the networked system remain available to legitimate users, and should limit the impact of mistakes. There is no one single fix to the computer network security challenge, and thus organisations typically use a whole array of techniques to protect their assets. In terms of mechanisms, protection is typically achieved using firewall devices, whereas detection can be achieved with an **I**ntrusion **D**etection **S**ystem (IDS), and correction could involve distributing backed-up files to replace corrupted ones. Finally, all the elements required to fulfil the need to be coordinated and this is typically achieved with security policies.

Table 2.1 – Example of controls as presented by Howie [48]

	Prevention	**Detection**	**Correction**
Physical	Locked Doors	Camera Surveillance	Fire Suppressant Systems
Administrative	Call back on Password Requests	Review of Sign-in Sheets	Adjustment of Data Classification
Technical	Firewall	Intrusion Detection System	Raid Array Disk

2.3.2 Interdependence

Computer network security is highly interdependent [26], and, thus, the security of one network node has an impact on the rest of the network [23]. Hence, security shortcomings outside an organisation are potential threats to organisation operations and also increase the likelihood of internal compromises [49]. Since computer network systems are typically interconnected to one another via the Internet, this openness is a factor that intruders can exploit. Figure 2.3 illustrates the various intrusion vectors that intruders can exploit in order to target an organisation. Indeed, the home becomes an extension of the organisation with the increased usage of **V**irtual **P**rivate **N**etworks (VPNs) or remote connections to corporate networks [50]. Similarly, the organisation might have dedicated network connections to remote offices, with limited

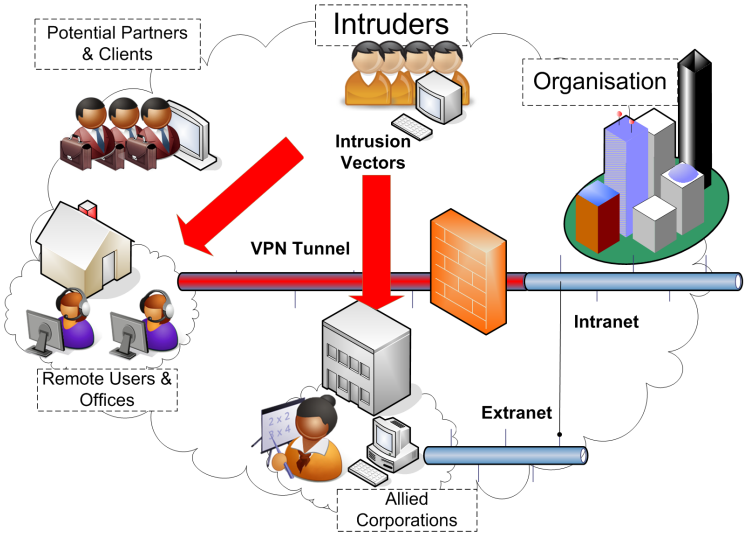

Figure 2.3 – Access to the organisation via VPN tunnel and Extranet

budgets allocated to security [8], as well as connections to allied corporations that may not have the same security standards [9]. All these means of communication often bypass firewall's scrutiny. Thus, these external locations typically constitute *targets of choice* for intruders, as they often protect themselves from being tracked down by circumventing weakly secured systems, and have these perform their deeds [18, 51]. Interestingly, researchers such as Guttman [52] or Uribe and Cheung [53] approach the challenge of computer network security using the same abstraction as in Figure 2.3.

Along with this, new technologies and devices typically are threats, whether these are used outside, or within, the organisation. Indeed, these often possess some computing or networking capabilities that can be subverted [54]. The sources, or tool-kits, to create the necessary software programs are often publicly available on the Internet, and bringing the functionalities together does not require an understanding of the underlying technical exploits [16, 55, 56]. Subverted devices can then be used to perform an network outage, for instance, which is often referred as DoS attack [26, 18]. Also, subverting network devices is facilitated by the fact that these are often left powered-on, remotely accessible, such as through the Internet, and are left unattended [54]. This leaves intruders the luxury of time to discover, and compromise network hosts. Along with this, one intrusion may only be one stepping stone within a larger-scale attack [57, 58]. Arguably, computer network systems are deployed within organisations because of the functionalities they offer, and that security is often secondary [59, 28]. Hence, a key issue in computer networks is that new technologies or devices could be deployed without adequate vulnerability, or security assessment [60].

Since one compromised node is enough to provoke damage [58, 55], achieving and maintaining adequate security standards requires to meet many criteria throughout the life-span of a system. These include consistency, for instance, whereby all the

networked devices must be configured properly. Network resilience is also another criterion and relates to ability of the network to handle traffic as well as having limited sensitivity to congestions, which might result in the hindrance of the organisation operations [50]. Consequently, there is a substantial upkeep in terms of finance and staffing, for instance, and this is particularly challenging to cope with in organisations where budgets are limited, such as for **s**mall and **m**edium size **e**nterprises (SMEs) or within home working [61, 3, 8].

Some research work aims at solving the computer network security challenge on a large scale [15], such as with ensuring that the Internet becomes less permissive to malware outbreaks, such as with computer worms [62, 63, 59], or DoS attacks [18, 16]. Arguably, these projects address part of the security challenge and therefore it is essential to enable organisations with the possibility to secure their computer networks in a consistent and rigorous manner. This should be achieved within the means of the organisation, such as in terms of what their various networked systems can handle [21].

2.4 Security Policies

Computer networks often represent a significant investment in financial terms as well as time and human resources, and thus need to be secured before damages. The manner in which this can be achieved includes developing security policies that aim to coordinate security mechanisms. At present, there are no standard definitions for a security policy *per se*, nevertheless, it is possible to distinguish three distinct forms within the literature.

First of all, from a management perspective, a security policy is expressed in a written document using a natural language, such as in English [64, 65, 34, 8]. Hence, it is a **Managerial Document** that describes the aims and objectives of the organisation, along with the assets associated to fulfil these, and includes the methods employed to protect these assets. Hence, a security policy is meant to coordinate the configuration of mechanisms used to defend the organisation's assets and resources. It also often outlines what is permitted or denied in terms of activities, applications, services, and so on [47]. Furthermore, a security policy might also incorporate instructions regarding data that should be logged, in order to provide evidences of problems in configuration, intrusions, or damages [10, 66, 27, 67].

A security policy can also be expressed using **mathematical representations and models** [52, 68]. This type of approach considers the security policy as a set of constraints for the system under scrutiny, and these constraints are then applied against the system functionalities. Hence, such a method is often suited to identify conflicts in the policy, however it requires logical models of individual elements of the system being analysed to operate properly. Along with this, such an approach is heavily reliant upon the expertise of personnel in charge of deploying computer

network systems.

A security policy could be a configuration syntax for a particular security equipment, such as Cisco **A**ccess **C**ontrol **L**ist (ACL) syntax for network firewalls [37], or rule syntax for an IDS, like Snort [69, 70]. These are thus **Intermediate Languages**, or *scripts*, and allow system administrators to create goals, or objectives, for the device without the drawback of having to understand the underlying implementation complexity. It is unlikely that a device can fulfill multiple security principles, and thus the translation between requirements and configuration has to happen as many times as there are devices needed to ensure computer network security.

2.5 Firewalls

Typically, organisations only have one gateway to the Internet, thus they often rely on a perimeter defence mechanism to keep-out unwanted, or malicious, traffic. This is often achieved by deploying a network firewall that resides between the organisational network and the outside. These devices can also be distributed throughout the organisation network infrastructure to filter communications between the different sub-networks [71, 52]. This typically allows better control over the applications, services, and protocols in use within the organisation.

Indeed, network firewalls can examine the information contained within the network traffic or data flow (Figure 2.4). Figure 2.5 shows that when the firewall receives a network packet on one of its interfaces, this packet has to meet multiple criteria before it is released towards its destination. If one of the criteria is not met, the packet is discarded [72, 73, 74, 31, 52, 75].

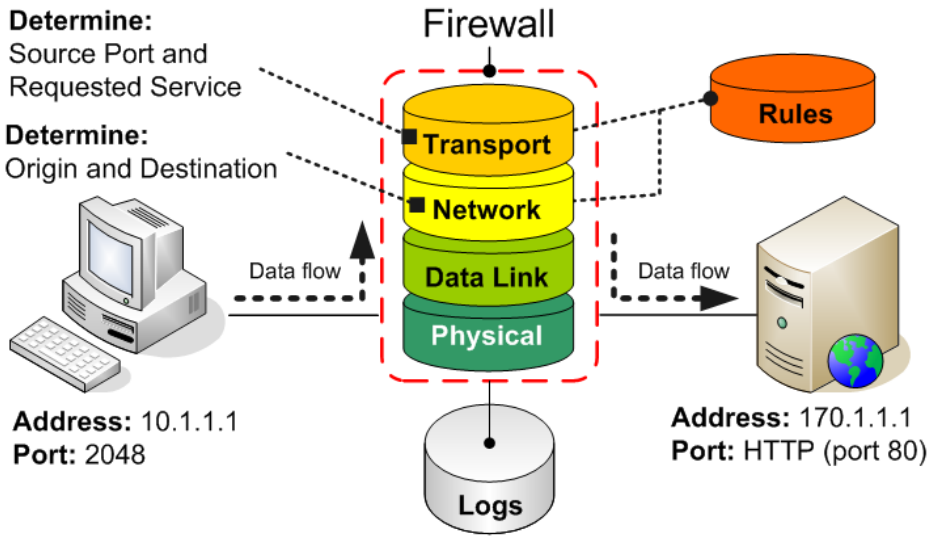

Figure 2.4 – Network firewall fundamentals

From a security point-of-view, it is the rules that apply to network information,

such as the origin and destination of the communication, or the transport information, such as the requested service (Figure 2.4) that are of interest. Table 2.2 lists examples of rules that can be deployed on network firewalls. These rule-sets should only permit communications that the organisation deems authorised, and is thus expressed in organisational security policies. For instance, a firewall may authorise users from an allied corporation (Figure 2.3) to consult a database located within a department of the organisation, however deny these users access to the intranet website. The deployment of such policies is possible without using network firewalls, however it would require the implementation of complex authentication schemes on both the data-base application, and intranet website.

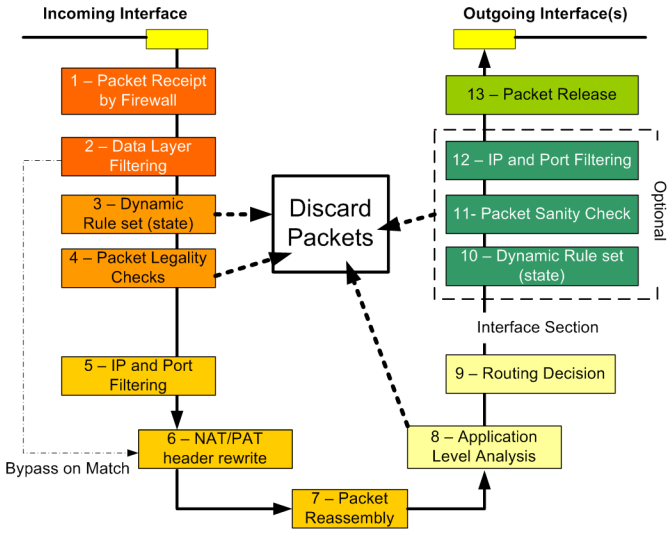

Figure 2.5 – Firewall's inner working as presented by Kamara et al. [31]

Table 2.2 – Example of network firewall rules

Source Address	Source Port	Destination Address	Requested Service	Permission
10.1.1.1	1024	170.1.1.1	HTTP (80)	Permit
10.1.1.2	any	170.1.1.1	any	Deny

A key characteristic of firewalls is that they typically evaluate the information contained in the network traffic against the statements which make-up the rule-set(s) in a sequential manner [52]. There are two types of filtering methods for this [76]:

- **Stateless filtering**: The firewall treats each network packet in isolation, and thus the device has to examine the rule-set every time a packet is received until a match is found. Otherwise, the packet is discarded.
- **Stateful filtering**: The firewall keeps track of the state of the network connections. New connections are treated in the same fashion as with Stateless filtering, however, the fact that a network packet belongs to a communication

that is *already* established matters [77]. This filtering method thus requires the device to maintain a connection state table.

Depending on the level of control the organisation requires over the extensive number of protocols, services, and applications, the resulting firewall rule-sets can be very large. Thus, it is paramount that the deployment of security policies on network firewalls has a manageable impact on key network characteristics, such as the available bandwidth and network latency [78, 32].

The policies that a network firewall enforces applies to all networked nodes connected to the firewall, and their centralised nature offers a single point-of-control [72, 73]. Along with this, it is also possible to install a firewall directly on end-hosts [79] (Figure 2.6), where the main difference is that the rules often apply to the specific host only, as Figure 2.6 and Table 2.3 show. This is often a software component that performs verifications on the data going back and forth from an application to the network interface, such as when using a web-browser. Such a component ensures that only authorised applications access the networked resources, or accept incoming connections [79]. This, in turn, limits the likelihood of intruders remotely controlling the host.

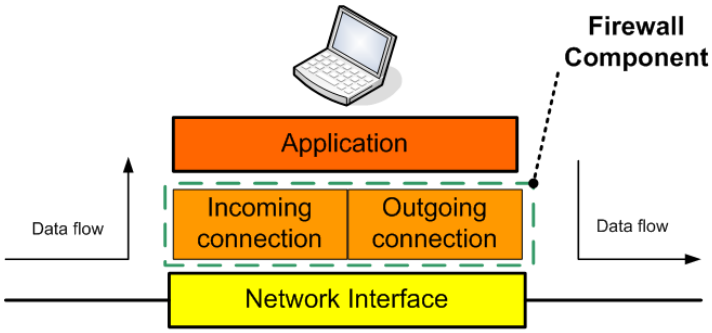

Figure 2.6 – Software firewall fundamentals

Table 2.3 – Example of software firewall rules

Application type	Incoming connection	Outgoing connection
Web client	Deny	Allow
Web server	Deny	Deny

Application Proxies

Application proxies are an example of technology for which the balance between functionality, security and flexibility is challenging to achieve for organisations. An application proxy acts on the client host's behalf, and, as they typically operate at the application layer of the **O**pen **S**ystems **I**nterconnection (OSI) model, they are

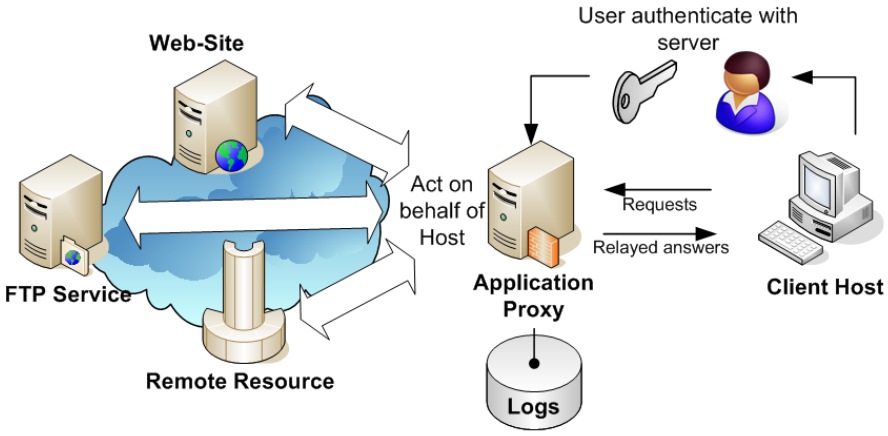

Figure 2.7 – Proxy server

often defined as Application Gateways [73, 80]. Along with this, proxies offer the same anonymity and isolationism as **N**etwork **A**ddress **T**ranslation (NAT) and **P**ort **A**ddress **T**ranslation (PAT) technology (Appendix D). Indeed, remote nodes only see a single entity connected to themselves despite the fact that multiple nodes have to establish communication with the proxy before reaching their required resources (Figure 2.7). However, unlike traditional firewalls, the implementation of proxies is not transparent, as they do not offer their services in a *de-facto* manner [73]. In other words, applications must be pre-configured to use the proxy.

Application proxies exercise a tighter control on authorised applications, and can keep track of established communications [80]. Indeed, network firewalls identify which applications or services are being used based on several network information, such as the **T**ransmission **C**ontrol **P**rotocol (TCP) port number, whereas application proxies can determine what is going on within the communication, and thus intervene accordingly. This typically results in computation intensive operations, and can lead to undesired performance overhead [73]. To further increase security, organisations can implement an authentication scheme before services are granted to requesters [73] (Figure 2.7). Hence, like with network firewalls, application proxies often adopt a configuration derived from the organisation objectives.

2.6 Intelligence Gathering and Intrusion Detection

Security issues, or traces of past, current, and, sometimes, future incidents can be found in various places on a computer network system [57, 81]. These traces can be collected with **S**imple **N**etwork **M**anagement **P**rotocol (SNMP) [82, 83], from the logging facilities available on some devices using operating systems logs [84], or from network devices facilities [85]. It is also possible to observe directly the network traffic with an IDS, such as Snort [69]. The purpose of deploying observation and data gathering facilities throughout a computer network system is to use the information

as a basis to produce and deploy mitigation procedures [86]. This section outlines the methods employed to gather information from network nodes contained in operating systems logs, gives an overview of SNMP, and it introduces the principles of IDSs.

Network Node Information

Most networked devices possess logging facilities which are present to help administrators assess if the system is running as intended, or if there are any malfunctions. There are few standards that dictate the manner in which networked devices must produce logs. Thus, the format of the information is likely to differ from one device to the next, as well as from one manufacturer to the next. Nevertheless, the meaning of the information remains the same [86]. Computer networks can include a large number of nodes; thus managing the overall system in such manner could be highly time consuming as the analysis of the system status occurs locally [84]. Arguably, these logs also contain an overwhelming amount of information [85], and thus, the type of information required can be outlined in the organisation security policy to ensure relevance of data. Another important factor is that many logs, such as provided by Unix/Linux or Microsoft Windows operating systems, are generally unsecured. Any person using the system could be able to access them, delete them, or even tamper with them, which could be misleading in the context of a forensic investigation, for instance [87, 88, 89].

Simple Network Management Protocol

Traditional network management relies on SNMP. In order to efficiently use this protocol, network administrators must know in details the type of information required and implement procedures to overcome the fact that each networked device provides a unique set of information [82, 83]. Unlike with operating systems logs, SNMP data can be sent to a central node for processing [82], and in this respect it eases administrative tasks. Nevertheless, the volume of information produced with this method can be large. This is due to the fact that SNMP only provides raw the information to the management node, or nodes, which the task is then: to process messages; interpret gathered data; and take appropriate actions, such as notifying administrators. Arguably, the main drawback of SNMP is the fact that information can be incomplete as there is no guarantee that data reaches the management nodes, since SNMP operates over the **U**ser **D**atagram **P**rotocol (UDP).

Intrusion Detection System

An IDS is passive mechanism which captures network traffic, and then verifies observed traffic against a pre-established set of rules. Unlike firewalls, an IDS does not intervene on its own if a violation is uncovered. Instead, any violation in the rules is reported with alerts [90] (Figure 2.8). In order to avoid writing complex rules to

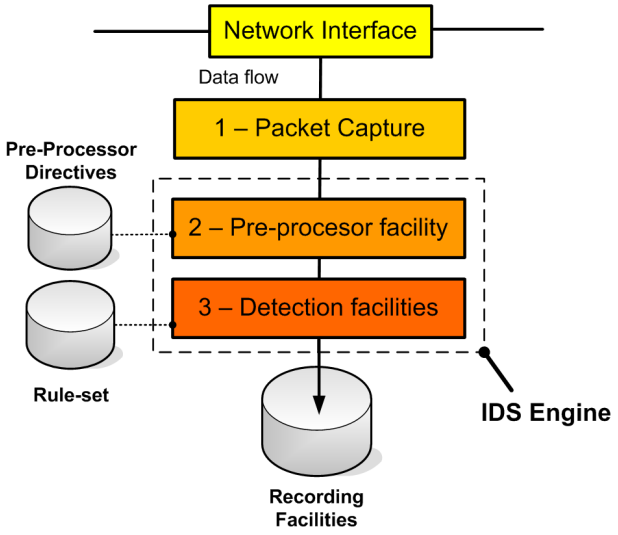

Figure 2.8 – Intrusion Detection System's core functionalities

detect traffic problems, such as in detecting fragmented packets, it is possible to use a pre-processor before the rule engine is called. Using pre-processor directives is an enhanced method to detection TCP port scans [69] which is a technique intruders favour to uncover vulnerable network nodes, for instance [56]. Along with this, IDSs are suited to uncover malicious activities carried out by legitimate means, such as in an end-user which executes a remote script on a Web browser.

The quality of an IDS is often determined by the ratio between false positives (incorrect alert), and true positives (correct alerts). This ratio is dependent on the type of engine used to analyse the network traffic, and experts often distinguish between two types of engines:

- **Signature-based Engine.** Signature matching is the method of parsing received packets, and verifying if they contain a signature known to the system. For example, an IDS system would uncover evidences of a remote user attempting to gain administrator's privileges on a system by looking up the packet's content for the string "`user root`". This type of engine is widely used because many exploits have known signatures, or patterns, and organisation can also develop their own.
- **Anomaly-based Engine.** On the other hand, it is a rare occurrence that intrusions happen in a single packet thus, it becomes necessary to interpret the whole communication instead. This is the principle of anomaly- based IDS [44]. This type of IDS requires to be trained before deployment so they are able to compare on going traffic against the patterns they have learned.

IDS deployment is often non-trivial, and more importantly, organisations must use a network design where the deployment of an IDS is feasible. Indeed, modern networks no longer use shared buses on which any nodes connected to could capture all communications. Instead, networks have evolved towards switched technologies that

allow for segmentation, as well as better bandwidth usage. Consequently, eavesdropping communications requires additional hardware which is capable of forwarding the required communications to the IDS, as well as forwarding it to the destination. The amount traffic then received by the IDS could thus be high, and hence IDSs require dedicated hosts with high computation capabilities [85, 70].

2.7 Dynamic Networks

Computer network systems are typically rule-based, reactive and difficult to make evolve. As threats become increasingly sophisticated and quick to propagate [58], it becomes paramount to enable computer network systems with the capability to evolve, or adapt, more rapidly. The use of modern software technologies can address these drawbacks, as they can streamline traditional administrative tasks, such as reconfiguring network firewalls [14], for instance. Nevertheless, these technologies are seldom deployed in organisations, however this thesis argues that their capabilities can be harnessed for security purposes, such as to create a real-time mitigation system [21]. The **M**ulti-**A**gent **S**ystem (MAS) or **A**ctive **N**etwork (AN) paradigms can fulfil this purpose, and be part of an improved security approach [21].

2.7.1 Multi-Agents Systems

In a distributed environment, experts argue that software agents represent a better alternative to traditional network administrative operations, such as remote script dispatch, as these agents can typically achieve complex tasks without relying on weakly secured technologies, such as with **R**emote **P**rocedure **C**alls (RPC) [91]. Perdikeas et al. [91] defines the properties of software agents, and these include:

- **Social ability**: agents are typically capable of communicating with each other, and exchange data in an autonomous manner.
- **Responsiveness**: the goal-oriented nature of MAS results in agents being smaller than traditional computing applications, and, in turn, enable the agents to react quickly to events.
- **Robustness**: the failure of one agents does not necessarily impede the activities of other agents [25].
- **Adaptability**: the agent intelligence is defined in a different way than traditional applications, and MAS platforms often allows up-dating the knowledge base without requiring the re-deployment of the agents themselves [83].

It is recommended, though, to chose a particular agent platform according to the objectives rather than the number of features offered by the technology [91], as indeed not all properties, such as listed in the above, are available in all agent platforms.

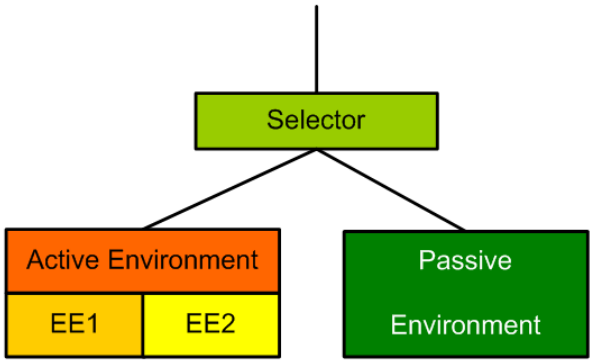

Figure 2.9 – Active Node network interface architecture by Li and Wolf [93]

2.7.2 Active Networks

An AN is a network space where all participating nodes can accept two distinct types of network packets. The traditional ones are dealt with in the passive environment. The Active Environment, on the other hand, processes the **switchlets** which are network packets embedded with instruction for the AN to perform [92, 93]. Figure 2.9 illustrates the network interface architecture of an AN node that Li and Wolf [93] presented. The Active Environment contains two areas, where the first area serves for the execution of the instructions, and the second allows building functionalities progressively. Typically, a series of switchlets are necessary to create a complex function, such as determining the best route to a destination across the network, and also permits the retrieval of information from a remote location [93]. Evidently, this might be regarded as a threat, and hence, similarly to MAS, the design of ANs address the drawbacks of RPC, and traditional programming. Thus, the array of possibilities of switchlets is limited. For example, a switchlet, or a created function, cannot access a node's kernel or files. In addition, in order to prevent intrusions based on buffer overflow exploits [94], switchlets are typically written in a strongly-typed language, such as Java or Caml [92]. A key aspect of ANs is that they are designed to facilitate the tasks of administration and the deployment of network services. These possibilities are essential in addressing the static nature of computer network systems. More importantly perhaps, AN technology does not aim to compete with traditional programming for networks, such as done in C++ or Object Pascal, nevertheless it makes ANs more effective in achieving their design objectives.

2.8 Conclusion

This chapter demonstrated that computer network systems are a key asset for organisation and thus outlined the basic concepts involved in securing them. It also showed that there are multiple technologies, approaches, and methodologies, that can be utilised to that end. Often, each mechanism addresses only one security aspect, and thus these methods have to be used in coordination. Such coordination

can be achieved with organisational security policies. Arguably, the rule-based, and often static, nature of computer network systems posses problems as intruders often use lapses in configuration to compromise network systems. This chapter introduced **A**ctive **N**etwork, or **M**ulti-**A**gent **S**ystem technologies as possible solution to address this shortcoming. Hence, it has provided the theory and context necessary to understand the themes covered in the literature review.

Indeed, the following chapter provides a critical appraisal of the concepts introduced in this chapter. The analysis of the relationship between the organisational objectives and computer network systems demonstrates that state-of-the-art approaches, or methods, are often developed in isolation. The Literature Review will highlight that one of the obstacles in a rigorous decision making process is the lack of dynamic performance data. Hence, an evaluation environment, that allows for collecting such key data, will be central to an enhanced security process. Thus one approach is to integrate these achievements together, this will provide the basis for the design of an **I**ntegrated **S**ecurity **F**ramework (Chapter 8).

Chapter 3

Literature Review

3.1 Introduction

Organisations have a duty to carry out their operations within legal boundaries, and this chapter reports on legislations determining the means organisations can employ to ensure the continuity of their operations. These legislations also offer routes to deter, pursue, and sanction individuals, or groups, who use computer systems to attack organisations. As newer technologies offer improved means of monitoring, there are concerns, such as breach of privacy, about legislations which are unable to keep up with the usage of these extended possibilities. Hence, the discrepancies between what the law requires, and their actual implementation within organisations is a key issue, as it could lead to the infringement of fundamental human rights. These laws also require that organisations meet some criteria, such as evidence that their computer network system is properly configured and performance known *prior* to alleged security incidents.

As security is a multi-dimensional concept, it is essential to establish policies that encompass the requirements and needs of an organisation. A review of published work indicates that it is challenging to establish such policies at organisational level. In the situation where policies are created from an abstraction of a particular system, the policies are often tied to a specific technology. Most importantly, these have to be interpreted prior to deployment onto network systems, and thus are often of limited benefit. Along with this, one of the major drawback in computer network security is the substantial upkeep required in terms of dedication, finance and staffing.

Arguably, the task of securing computer networks is made more challenging because of the number of systems that must be used to that end. Network firewalls, for instance, are the most suited technology to prevent unauthorised network access, however these are not suited to uncover potential intrusions. This is better achieved with the deployment of an IDS. Overall, security solutions do not often adapt well to the usage of new technology, and that their efficiency is typically linked to the skills of the operators in charge. Thus, changes are a time-consuming activity, and this contrasts sharply with the fact that most network threats can be re-engineered in a matter of hours.

There are technologies that could address this limitation, however, their adoption

is rare as they do not offer the same level of performances as traditional computer network systems. Most importantly, these require proper policing prior to deployment, and this thesis argues that an **I**ntegrated **S**ecurity **F**ramework and network firewall dynamic performance models can provide the necessary basis. Thus, this chapter provides arguments on why it is essential that organisations employ integrated security systems and which establishes a framework that promotes computer network security as a process. Indeed, concerns over the impact of security deployment on performance, for instance, is often used as justifications for low security standards in organisations. Yet, such a choice is seldom based on repeatable and verifiable evidence. Hence, this chapter demonstrates that an evaluation environment for network security devices, which allows for relevant data to be collected, needs to be part of an **I**ntegrated **S**ecurity **F**ramework. This, in turn, will permit the study of possible security policy implementation scenarios and the assessment of their impacts in terms of dynamic performance on network security devices.

3.2 Computer Networks and Legislation

The major objective of any organisation is to carry out activities within the boundaries of the law. By the same token, organisations must protect their assets, and this, in turn, implies deploying security mechanisms. Thus, there are legislations that regulate the usage of computer network systems, as technology typically offers increased monitoring and surveillance capabilities [95]. The challenge resides in limiting the impact on the fundamental rights of the end-users, such as for privacy, and aim to provide legal recourse to organisations in the event of security incidents. Hence, these regulations have an impact on the manner in which computer network systems are deployed and configured. This section presents the findings of published works that focus on the relationship between laws, computer network systems, and the interests of both organisations and their users.

3.2.1 Traditional Approach

Many nations realise that computer networks are central to their economy and growth. Thus, they put in place laws that typically seek to deter malicious activities which targeting these important infrastructures. Such laws include the **C**omputer **M**isuse **A**ct 1990 (UK) (CMA) [27, 96], and the **D**igital **M**illennium **C**opyright **A**ct 2000 (USA) (DMCA) [95] and their application can result in heavy fines and lengthy jail sentences [27, 97, 98].

One of the problems in this domain is that technology evolves rapidly and, often, lawmakers cannot keep up with new capabilities, as well as possible impacts. In the past, *computer crime* related mainly to activities happening directly on a local host, and laws did not include the concepts of networks, or network-based threats, such as DoS or **D**istributed **D**enial-**o**f-**S**ervice (DDoS) [99]. Indeed, Worthy and Fanning [99]

note that the CMA has shortcoming with respect to dealing with DoS type attacks as the disruption is *delivered* via authorised acts, where data is not always modified. Still, Kon and Church [97] report that it is possible for organisations to obtain justice and be awarded reparation through, arguably, outdated laws. Unfortunately, it requires tenacity and a key factor was the actual evidence of the case [97], thus such an outcome is more of an exception rather than the norm.

The increase in the number of security breaches, such as discussed in Timms et al. [3] and Alun et al. [8], triggered reviews of these laws, and thus key regulations are set to incorporate network concepts [99, 100]. Arguably, these changes focus mainly on the issues of authorship and the supply of tools, and the facilities that permit intrusions or attacks [99]. In this respect, a key factor is the international nature of intrusions and computer crime [99]. When authorship is established, it can then serve as a basis for extradition, or subpoena, of the people involved. Despite these changes the judicial process is still not perfect. For example, in the event where a British organisation is involved, if there is no extradition agreement between the United Kingdom and the country where the perpetrator, such as where a virus engineer lives, the proceedings cannot continue [27].

On a more technical matter, many threats are engineered by assembling elements from tool-kits [58, 16, 4, 101], and these tool-kits do not often include clear mentions of authorship [99]. More importantly, perhaps, lawmakers are ignoring the possibility of intrusions being carried out using legitimate means [99], such as in commercial e-mail software applications [97]. Indeed, this type of application does not modify the system in a malicious way, however it would be viewed as being faulty. However, Cusumano [102] shows that software manufacturers are unlikely to be blamed, and, Worthy and Fanning [99] argue that this type of event can potentially increase software manufacturer's liability, and thus stifle the industry.

Barton and Nissanka [27] note that public bodies often have funding problems that hinders drafting regulations, and the enforcing computer networks on a large scale. Arguably, there are several technical implementation problems too [100]. Thus, with respect to the potential losses security breaches create, Barton and Nissanka [27] advise that organisations should not rely solely on the laws to provide protection, prefer a more pro-active stance. Unfortunately, this places organisations on an uneven footing, where, decisions cannot be made on forecast costs alone, as these are typically difficult to estimate [27, 45], and that a tarnished public image cannot always be recovered [103, 47].

3.2.2 Monitoring Internal Activities

Organisations are liable for the actions that originate from their computer domain, and are responsible for the behaviour of their personnel using these resources [103, 104]. Indeed, the inter-linkage of networks increases the likelihood of local exploits becoming global. Moreover, organisations face the challenge of preventing malicious

activities that target local installations. In this domain, problems typically occur because several conflicting aims have to be fulfilled:

- Auditing of the system for maintenance purposes.
- Data collection and analysis for intrusion prevention.
- Data gathering for the assessment of the compliance with policies.

Thus, the data collected could include elements which can be attributed to an individual, and be of a personal nature, as well. This is often prohibited, though, as it is typically considered as an infringement on privacy [95, 103, 46, 104]. Unlike with traditional telecommunication networks, the delineation between computer networks *traffic data* and *content data* is problematic to establish. Traffic data is useful to analyse for maintenance purposes, and content data is key in preventing intrusions [105]. For instance, it is acceptable to record the queries which have been issued to a search engine, however combining this data with other items, such as time, date and the **I**nternet **P**rotocol (IP) address, could constitute personal data [100].

Thus, the type of data recorded, as well as the extent of access rights, need to be clearly specified, otherwise, administrators will simply record data along with some sensitive information for time intervals that are illegal. Mares [98] shows that poor policing can be detrimental to organisations, and reports on a case where a trial established that two individuals diverted data for commercial as well as personal gain. The central argument put forward by the accused is that their hierarchical positions within the organisation made their activities lawful. However, the judge argued that the manner in which the interception was accomplished did not fall inline with the relevant laws on this matter, namely the CMA, the **D**ata **P**rotection **A**ct 1998 (UK) (DPA), and the Human Rights Act 1998 [98]. Thus, organisations need to reflect on the role of operators, and administrative personnel, as, indeed, occupying such roles does not automatically grant permission to intercept communications [98].

The case highlighted in Mares [98] shows that a key aspect is consent, and Rogers [104] points out that there is often a signed agreement between users and organisations that stipulates communication monitoring. In addition, courts have often sided with organisations on cases related to privacy infringement, for such as in [49]. Several articles [98, 97] show that organisations could significantly lower the likelihood of internal sabotage if a compromise is reached between the expectation of the end-users and security requirements [47, 103]. Indeed, some argue that fear and misunderstanding typically drive up security costs unnecessarily [104, 55]. Baker and Wallace [55] explain that many security solutions are deployed without a real understanding of the threats they can mitigate against, or where there are overlaps. It is often acknowledged that personal data could be lawfully analysed, however, there are criticisms regarding the discrepancy between what the law suggests, and the manner in which systems are subsequently configured, or operated [103, 46, 104]. For instance, Rogers [104] provides the following organisational policy which applies to both local and remote users:

> *" The Company reserves the right to access any information contained in company computing and communication systems, and the company may, at its sole discretion, disclose this information to third parties."* — Rogers [104, p. 6]

This policy raises concerns over [103]:

- The type of data recorded.
- The time period for which the records are kept.
- The circumstances and criteria regarding the data that is internally, or externally, shared.

All these elements require detailed and precise policies as these, in turn, make collected data relevant [100]. Without detailed policies there is a tendency to assume that what is not written is permitted [98], and then it becomes difficult to establish, or prove, an offence. Indeed, digital evidence, such as from operating system event logs [67], cannot be the sole determinant factor for such a situation. Therefore, possible intrusions, or misbehaviours, should always be investigated from both a technology prospective and a human point-of-view [95, 89, 42].

While Barton and Nissanka [27] argue that laws are often confusing, some experts see problems beyond what legislations say, and Escudero-Pascual and Hosein [95] criticise the manner in which they are written. In most cases, the law regarding newer technologies are drawn from experiences gained with older, and, from what appears to be similar, technologies. For instance, electronic mail is regarded as similar to postal mail, and that communications over the Internet is regarded as similar to a phone call. However, findings could vary greatly depending on the monitoring mechanism involved, such as in the case of recording web traffic [95]. Each website has an IP address that is unique, and as this addressing scheme is not user-friendly, instead websites are found using domain names that normally reflects the content they provide. Thus, recording solely the IP address could be seen as non-descriptive or non-incriminating, while recording the domain name of the web-site possibily is. This problem arises due to the fact that chosen policies are written in a non-technologic specific way [95], such as argued by Kosta and Valcke [100] regarding Instant Messaging communications.

3.2.3 Legal Impact on Networked Systems

New regulations seeks to enhance organisation legal cover, and ease legal recourse. Compliance with these regulations have, in turn, consequences on the underlying computer network systems. Organisations, for example, keep temporarily records of some network traffic for billing purposes [100], however, as argued in the Section 3.2.2, this data could contain information which can be of interest to police forces. Thus, they could be requested to provide this information, and Kosta and Valcke [100] demonstrate that this activity can incur a substantial upkeep. Often, these data can become too voluminous to store, and is thus difficult to search through [100].

Moreover, the DPA stipulates that mechanisms have to be deployed to protect data from: unauthorised or unlawful processing; accidental loss; destruction; or damage [27].

Worthy and Fanning [99] also show that organisations could be required to keep large volume of data for an extended time period. Indeed, attacks, such as DoS or DDoS, typically exhaust network resources, such as available bandwidth, or exceed network devices capabilities [4]. Hence, in order to benefit from the improved legal coverage, organisations may have to provide consistent and verifiable audit data that allow the distinction of an attack from poor network performance, or as an error in configuration. This can only be achieved with increased logging, and rigorous surveillance of the system for long time periods. Furthermore, there is doubt on the verifiability of equipment specifications, such as demonstrated in Cremonini and Martini [45], and thus organisations need to employ suitable approaches [99]. This shift in emphasis addresses the typical post-incident focus of previous legislation [27], and thus a more pro-active vision is promoted.

Organisations can prevent many attacks from originating from within their domain by filtering outbound network traffic, and this can be achieved with network firewalls, for instance [106, 107]. This shows that regulations can influence the configuration of the computer network, and, in turn, this configuration affects the extend of the liability of the organisation. Still, the actual deployment is left to the discretion of organisations, and to a certain extend, also the operators [21, 14]. Indeed, Wool [107] shows that outbound network filtering, for instance, is a functionality that is not often used, typically due to network characteristics, such as increased network latency to a point where it impedes real-time applications, such as **V**oice-**o**ver-**IP** (VoIP) communications [108]. Such reticence represents a key problem, as there is strong evidence that most incidents originate from within the organisation [29, 47, 97, 109]. Indeed, Woloch [47] reports that several surveys estimated that 70% of security incidents are pertrated by insiders. Mitnick and Simon [64] or Winkler [110] show how costly these incidents could be to organisations as they could dead to the theft of trade-secrets, for instance. This is possible because a disgruntled employee, for example, will not be deterred by written rules, will probably carry out the necessary steps over a long period of time [109, 81], and will maybe abuse their position too [98], especially when the motivation has a potential pay-off [111]. Furthermore, the large amount of information contained in networked systems creates the situation whereby a disgruntled employee could commit fraudulent activities without being detected [49, 87]. A weakness of networked system is therefore the lack of verifiable enforcement.

3.3 Security Policies

The technologies introduced in Chapter 2 offer adequate means to address security threats, however, it would seem that practitioners are unable to make the best of defence mechanisms [112, 113, 47]. That might not be the case on an individual basis, however problems often appear when systems are combined to address the complexity of computer network security [14]. Along with this, organisations have to ensure that their overall system complies with legal requirements [98, 99, 100], and security policies are often considered the most appropriate method for this. Arguably, Mares [98] best illustrates this in the context of the **R**egulation of **I**nvestigatory **P**owers **A**ct 2000 (UK) (RIPA), and explains that organisations need to monitor their computer network in order to implement relevant changes to the system. This activity can easily lead to the acquisition, or interception, of private communications, such as e-mail exchanges [104], and hence there is a need to ensure that the purpose of the monitoring system is not subverted.

Ideally, all computer network elements should be configured according to the organisation's security policy, which incorporates requirements such as: privacy; physical access restriction; application availability; network confidentiality; data integrity; access policy; and so on [43]. This section presents an analysis of the various types of security policies as defined in the literature, and identifies their respective strengths and weaknesses.

3.3.1 Managerial Document

A security policy at a management level is typically a formal document that includes the aims and objectives of the organisation, and should define the best practices and internal rules, or regulations [10, 44, 114]. As computer networks constitute a medium to fulfil these objectives, such a document thus dictates the configuration that networked systems should adopt, as well as the solutions that should be employed to limit the risk of incident. It should also include the penalties involved with not respecting the rules [66, 10, 64, 115, 47]. Hence, the system must be configured to gather relevant evidence of intrusion, misuse, and so on. Defining security policies, though, is a non-trivial task, and a factor in the rise of successful intrusions resides in the fact that many organisations, regardless of their sector of activity, do not possess one [3, 103]. Or, when it exists, it might not convey enough meaningful information to effectively contribute to enhancing security [55].

Policies and systems are often regarded as separate entities, possibly due to the fact that computer network systems have been progressively deployed. Thus, an important factor in creating the document is prioritising it with respect to system capabilities, as it is often not possible to dedicate the same level of resource, such as budget or personnel, to all assets [8, 55]. There are few standards that can help

defining security policies, such as the ISO-17799 which provides detailed list of recommendations [3, 65, 34, 8]. Still, the management overhead that this often entails is often regarded as a barrier to the enhancement of organisational security standards [3]. Alternatively, a sub-set of the recommendations can be selected according to the area of activities of the organisation [34]. Eloff and Eloff [65], for instance, define an approach that matches objectives, as expressed in the ISO-17799 standard, with technical solutions, and their approach relies on the presence of certified products as a measure of compliance [65]. Arguably, this standard requires keeping up with the presence, or absence, of systems and procedures, however some experts are critical of this approach, as it does not allow to establish the extend of compliance [55, 47].

Consequently, the content of security policy is typically non-trivial, and personnel in charge of configuring the system are likely to interpret these requirements [10, 115, 66]. This is perhaps due to a lack of understanding, however, it is also likely that personnel adapt security policies to the: resources, such as in terms of hardware or software available to them [21]; and to their own skills [14]. The drawback of this approach is that security policies are often deployed in isolation, and this could result, for instance, in network firewalls being configured in such way that automated anti-virus updates are blocked [116]. Saliou et al. [21] and Woloch [47] thus highlight that there is no guarantee that demands are actually met by any given system.

A major issue with security policies is, thus, the discrepancy between the requirements and deployment, as it can lead to security policies becoming obstacles to current practices [10]. Danchev [10] argues that security policies at management level can only be successful if a dialogue is established between personnel liable for security within the organisation, and the users who participate in the fulfilment of the organisation's objectives. Similarly, Wadlow [111] also underlines that this would work as an incentive for users to find ways around them, thus creating more security breaches. Hence, this dialogue is essential to uncover the technologies used in each department of the organisation, and how they are being used [10]. It can also act as a leverage to the fact that managerial level security policies do not often take into account the technical realities [47].

Defining high-level security policies, though, is just the first step toward enhancing security, as security directives must then be followed through, and verified [65, 34]. Woloch [47] stresses that the traditional bi-annual basis of security reviews is inadequate to deal with threats which can evolve in a matter of hours. More importantly, perhaps, system configuration and policies tend to evolve separately, leading to inaccurate views of security capabilities [14, 111]. Indeed, Wadlow [111] investigated the computer networks of two organisations that merged, and identified that branches located in another country from the main office had their firewalls configured in such way that it was possible to connect directly to the corporate network. This created an intrusion vector such as shown previously in Figure 2.3. Wadlow

highlights that the management personnel of this particular organisation did not understand the need to re-evaluate the security of the organisation after its expansion. Therefore, there is a missing feedback element in security that would allow changes in computer network architecture, or set-up, to become visible in high-level security policies, or in a compliance document [21].

3.3.2 Modelling

For security policies as an approach, this this establishes two distinctions. On the one hand, research that attempts to map managerial-level security policy into system configurations using formalisms, such as with mathematical formulae. On the other hand, research that advocates practitioners to address computer network security from various point-of-views, not solely from a technological prospective.

Macro-level policy models

This type of model includes the work of Stajano and Anderson [117], who realise that computing is becoming ubiquitous and stress that an equally ubiquitous approach to computer security is not desirable. Other researchers, such as Viega and Messier [29], for instance, have already criticised the importance that organisations give to encryption. Stajano and Anderson [117], however, argue that a technology can be unsuitable depending on the system context of usage. In order to prove their point, they concentrate on the issue of authentication in mobile computing, and establish that encryption is computation intensive, and consumes, in turn, increased battery power. Hence, mobile devices, such as **P**ersonal **D**igital **A**ssistants (PDAs), have a reduced up-time capabilities when used in such context. Stajano and Anderson believe that it is possible to address security concerns by first considering the environment in which security has to be established, as opposed to add security once the system is operational.

Management level security policy

Research focusing on converting managerial level security policy to system configuration, include Al-Shaer and Hamed [75], Guttman [52], and Schneider [68]. The earliest research found on that topic is that of Guttman [52]'s, which focuses on ensuring that filtering policies are enforced by using multiple filtering devices. In order to achieve strong filtering, their resulting framework takes into account the logical system arrangement, which included: five distinct computer networks; the decided relationship between any two areas; and the underlying structure of the communications. In this instance, the various elements of information that can be extracted from the TCP/IP protocol suite, such as source and destination of the traffic, as well as the service requested, amongst other items. To that end, Guttman's model employs a description language that is based on several mathematical paradigms, and this

allows for the rigorous demonstration of whether the communication between two network locations is possible.

Also using mathematical formulae to support his theories is Schneider [68]. The difference between Guttman [52]'s work and Schneider's, is that Schneider considers policies on a *per-system basis*. His approach is thus more detailed and leads to the finding that policies might not always be feasible from a logical perspective. Furthermore, Schneider believes that security policies should be multiple and simple, as it is not often possible to implement one security policy across multiple systems that is monolithic. For example, a policy which seeks to prevent users from accessing websites typically requires network firewalls that are distributed across the organisation network to be configured accordingly. The challenge is thus for the configurations to be consistent throughout the computer network in that regard, and also take into account that the policy may hinder the activities of some users. Hence, the need for the multiple and simple requirements. Nevertheless, a key requirement in both works is that correct syntax is used, otherwise it leads to an approximation of what is actually wanted [52, 68].

The merit of Guttman [52]'s and Schneider [68]'s work is that it is possible to gain an insight of the security level which a change in configuration of the system would produce. However, considering the fact that most organisations already run a computer network system, it is equally desirable to be able to assess current attributes against initial requirements. Al-Shaer and Hamed [75] propose to address this issue for firewalls by modelling the manner in which their rules are written. Their model is able to uncover defects, such as inconsistency and contradiction. Unlike Guttman [52], however, their model is not capable to carry out this assessment for distributed elements. Indeed, Al-Shaer and Hamed [75]'s approach only allows the modelling of the logical properties of one network firewall at a time.

The key advantage of security modelling is that it allows a balance of functionalities against security requirements. However, from an organisational point-of-view, these approaches are often not efficient in their current form, as they typically require specialised knowledge in order to successfully use them. Along with this, their application are, at times, linked to a specific security mechanism, and, perhaps, the product of these approaches will still have to be translated into some form of configuration syntax for networked devices. Hence, these modelling techniques should be embedded into an expert system, itself part of an **I**ntegrated **S**ecurity **F**ramework, so that infeasible objectives are immediately identified, or configurations generated based on best practices, thus lowering the burden on administrators, without the need for interpretation [21]. Indeed, Nolann [118] highlights that there are several solutions to security issues, and thus administrators often deploy security in a manner that mirrors their own understanding of the situation [14].

3.3.3 Security Script

Security policy scripting typically allows for the focus on security parameters and requirements, as opposed to the intricacies of the underlying technology, such as for the Cisco ACL syntax in some firewall devices [107]. Thus, security policy scripts are written in a particular language, or syntax, and then interpreted or incorporated, into a security device, such as a firewall, or an IDS. Such methods are utilised by Paxson [119] for his intrusion detection system, and by Bertino et al. [120] in their secured web-document access system. Arguably, this approach reflects the sequential and rule-based nature of most security devices.

Paxson [119] presents an IDS capable of uncovering attacks in a real-time manner, using *Bro*, which relies on *policy scripts* to gain an understanding what traffic is legitimate, and what traffic is not. In essence, these scripts are high-level definitions of what nodes can, or cannot, do in terms of communication with parts of the organisation network or external computer networks. To that end, Paxson provides key examples of services and protocols, which constitute the core of given attacks. Since the scripts are not written in natural language, such as in English, the success of the presented system relies both the mastering of the scripting language, and the script writer's in-depth knowledge of intrusions.

The realm of application of a security policy is not limited to internal system and thus could be used to specify the manner in which communications with third parties is done. Bertino et al. [120] argue that organisations might have to distribute multiple versions of a same document to clients, thus they recommend that this should be done as a function of the relationship between the client and the organisation. Creating these versions manually often constitutes an important administrative overhead. Thus, Bertino et al. propose a model where the relationship is embedded into the requested document, and this, in turn, offers fine grained control. Unfortunately, it is only applicable to web-content and requires a designated security officer to write each of the relationships that the organisation has with its partners. Hence, it does not address the shortcomings found in Paxson [119], such as a heavy reliance in the understanding of the policies by the operators and technology-specific methodology. In other words, this methodology illustrates where security policies are interpreted and can result in discrepancies between the objectives expressed in the policies and technical implementations.

3.4 Current Security Practices

Since computer systems are intended to be operated by humans, it is appropriate to make users participate in securing these assets. This can be achieved by raising awareness, as all types of users, experienced or not, make mistakes from time-to-time [41, 8]. This section critically evaluates the procedures employed within organisations to prevent security breaches, or respond to security incidents, and highlights that user

involvement can be of real benefit to the overall security of the organisation.

3.4.1 Mapping Organisational Objectives

Computer systems are regarded as tools which accomplish objectives. As organisations become increasingly paperless, information and data are often not perceived in the same light as they were prior to the modern era. Hence, it is important to develop means to carry out daily activities, while limiting the exposure of the organisation to security threats. A key challenge in this domain is the definition of these objectives, and there are only a few available standards which can support this. However, it is often expensive to obtain the related documentation as well as obtaining the relevant certifications, such as with ISO-17799 for example [3, 65, 34, 8]. This problem is further exacerbated by the fact that these standards are often difficult to understand, and, more importantly, perhaps, challenging to adapt within the organisation [65, 24].

Thus, some researchers focus on a more bespoke approach. Bakry [9] is of the opinion that a successful development of security requirements is not possible without involving both the personnel who will have to respect it, and the personnel who have to verify its compliance. In order to motivate organisations towards establishing regulations, experts often highlight, or list, the various consequences that computer incidents might have [9, 10, 41]. Bakry focuses primarily on providing management personnel with a sufficient insight, in order for them to choose technological solutions with respect to a particular situation. Other researchers advocate the involvement of all the members in an organisation. This is to avoid creating, deploying, and enforcing security policies that are in conflict with the manner in which daily activities are performed [111, 87]. Otherwise, it gives incentives for users to bypass security measures [39].

3.4.2 Management Personnel Involvement

An organisation's executives can participate towards an improved security by employing qualified personnel [8, 14], and acknowledging the consequences of computer security incidents [9]. Caldwell et al. [14] underline that qualified personnel is often expensive, and, thus, relying solely on personnel to ensure the organisation security is often not a good strategy. Thus, managers must balance technical expertise with adequate investment in security devices. Still, this choice is often non-trivial as managers will normally rely on a **R**eturn-**o**n-**I**nvestment (RoI) index as the metric to determine investment [44]. Cremonini and Martini [45] demonstrates that the RoI index is often not appropriate for security, as it would typically favour the choice of one technology or device, whereas security mechanisms should be combined to offer adequate security. In addition, these investments have to occur regularly [3, 8, 44].

3.4.3 Lack of Leadership

Methods exist to help decision makers to take computer network security on-board, such as discussed in Rees et al. [11] and Rodgers [12]. Rees et al. focuses on the delivery of software applications, whereas Rodgers focuses on the deployment of network-based security in a corporate network. However, both research areas are similar in the principles they employ as both address the problem from the decision maker's point-of-view, and they identify the stages where security becomes important and the stages that defines security. Rodgers [12] is more descriptive and example-driven, whereas Rees et al. [11] formalise their approach. Nevertheless, it is an engineering-approach to the problem that is promoted. In other words, these researchers promote the planning, implementation and evaluation security, and demonstrate the need to iterate through these stages throughout the life span of the project. Both Rees et al. [11] and Rodgers [12] concur that having a member of the project responsible for the security elements ensures that the issue does not disappear along the life-cycle.

3.4.4 Administrator Lack of Security Standards

Within organisations, the task of ensuring the continuity of networked services is often viewed as the role of system and network administrators. Whilst this assignment often is true, it does not necessarily mean that administrators have to enforce security standards, or, for that matter, know about these [3, 8]. Administrators may not have an adequate security attitude themselves, such as: locking servers in ventilated cabinets; logging off while away from the keyboard, and so on [41]. Therefore, there might be doubt in their abilities to enforce a proper security standard within an organisation. In addition, organisations themselves might not have any policies in that domain, either, and this is confirmed, for the United Kingdom at least, in governmental surveys [3, 8].

System administrators are also responsible for implementing new services, such as an Intranet web-server. As there can be a high number of probes for vulnerabilities carried out each day [113], it is important that the deployment methodology for new devices, services, or applications is rigorous. This should include, for instance, ensuring that only legitimate users have access to these resources [73]. However, it is often not the case, and systems are often not properly audited throughout their life span [107]. This concern is also echoed in other research, such as of Ioannidis et al. [50] who reveal that intruders may use an error of judgement to bypass security mechanisms.

Administrators also rely too heavily on tools and security mechanisms [41, 121, 55]. The deployment of anti-virus, firewalls, and so on, do not necessarily prevent incidents [121, 58], just as biometric authentication does not prevent industrial spying [122, 110, 64]. Best practices recommend avoiding running unnecessary computer services as these increase the number of loopholes intruders can exploit [41, 16, 123,

124]. Nevertheless, there are often many applications, services, or protocols enabled within the network, and administrators struggle to keep up with their presence and capabilities [123], whilst allowing users to work efficiently.

3.4.5 End-user Issues

The purpose of security includes safeguarding data from deletion, corruption and fraudulent access. It is often the view of security experts [54, 28] that end-users are the weakest link in this domain. Yet, it is paradox that they are the ones who are often accessing sensitive data on a regular basis. For example, they are at risk of disclosing clients' information to competitors, or unauthorised people [104, 64]. This can be the consequence of users not carrying out their activities with regards to the organisation's best practices [10]. This might include, for example: forwarding work document to their home; leaving written notes about computer system access; and visiting untrusted Web sites [58].

3.4.6 Staff Training

Training users in the ways of security allow defending against one of the most efficient forms of attack against organisations: social engineering [110]. As Mitnick and Simon [64] demonstrate in the book "*The Art of Deception*", a few *know-hows* and *jargon* are enough to bypass the most elaborate systems, such as time-based token, and two-ways authentication schemes [64, *cf.* page 85]. Hence, proper training could prevent staff helping intruders to gain physical assess to resources, as this typically voids technical safeguards, such as password encryption [125, 126].

3.5 Pre-emptive Measures

The Section 3.2, an 3.3 covered the challenges of computer network security from a managerial as well as administrative point-of-view. Section 3.4, showed that raising awareness is an effective method for lowering the likelyhood of data leakage, for instance. Due to the multi-faced nature of computer network security, the various procedures discussed in Sections 3.2, 3.3, and 3.4, need to complemented with technological measures, and thus this section begins with the critical appraisal of these security measures involved in ensuring the continuity of computer network services.

The current model of computer network communication pushes the intelligence to the edges of the communication path. Hence, it is up to the communicating parties to deploy adequate software applications to ensure robust and reliable transport or security [26].

The complexity of the interaction among the processes within the host, and the presence of lapses in the applications' source code, make software applications a target of choice for intruders. Due to their complex nature, software applications

often have flaws [102, 58], thus, when successfully compromising a piece of software, intruders can work their way into a workstation's system and eventually access all the resources connected to that computer, in a way that a legitimate user would. This includes access to database systems, files, printers, and so on. Intruders can also use the node as a foothold to pursue the other stages of an intrusion [57].

3.5.1 Contributing Factors

Software compromises are often successful due to several factors. First of all, software capable of exploiting other applications' weaknesses are often publicly available [16, 58]. Also, computer network systems are centred around two-way communication, and, whenever one of the parties misbehave, the devices involved in delivering the messages will not intervene as it is not part of their initial requirement [26]. Furthermore, it might be challenging to uncover the type of activity depending on the intruder's skill to mask their presence [127, 128, 121]. Usually, automated malicious software, or malware, such as viruses, worms and Trojan horses, do not exhibit the same profile as compromises conducted by a human intruder. A worm will typically only perform a limited range of actions and then move on to a network-wide propagation, whereas a human intruder could use sophisticated steps and target a limited range of hosts [22].

Patching a software application plugs glitches that attackers might exploit. The spread of the Code-Red and MS-Blaster worms could be explained by the fact that patches were not applied [113, 112, 10]. However, patching is often an ad-hoc procedure and Morrison [112] argues that users would benefit from a system that would guarantee that patches were applied. Unfortunately, software patching requires a sustained upkeep. Indeed, administrators must monitor updates from all the software vendors whose products are deployed.

Another type of problem is malware which is designed to exploit flaws present on a specific computer platform. The spread of a worm can be viewed as similar to the propagation of cancer in body's cells. Indeed, like cancer, worms tend to compromise cells of their host so that their victims will attempt to compromise other cells [129]. An explanation for the rapid spread of malware is thus found in the fact that there are only a limited number of computer platforms. This observation might lead organisations to use software, or platforms, of limited popularity, however this does not enhance security [60], and organisations often choose to deploy the same type of equipment throughout their infrastructure. Their motivation is not improved specifications, but lower costs in maintenance and support [21, 87]. Furthermore, the arrival of the `Witty` worm [62, 59, 63] indicates that there is no complete immunity in using less-used software applications [60, 130]. Indeed, the exploit on which the `Witty` worm relied, for instance, applied to 12,000 internet hosts only, and this entire population was compromised within 45 minutes of the worm being released [59].

The level of maliciousness that the `Witty` worm exhibited particularly interested

researchers [59, 63]. Indeed, this worm was released within two days after the vulnerability it exploited was made public. It also randomly destroyed chunks of data, and because it uses UDP, it clogged networks producing large-scale DoS attacks. Moore and Shannon [59] see in the `Witty` worm all the elements of a high-profile worm. Experts in this domain have established that malware writers often exchange design ideas, or even compete on this basis [58], and thus it is only a matter of time before there are other worms based on the `Witty` model [63].

3.5.2 Administration and Management Overheads

Maintaining software applications represent a significant administrative and management overhead, as there are many applications, protocols, services and so on, to keep track off. Along with this, administrators have to establish if a new patch is suitable for the environment in which it is to be deployed, and the resulting computer network configuration needs to be assessed as well. The same often applies to software manufacturers where they have to stop updates from creating more vulnerabilities, and this then hinders the speed at which they can deliver remedies. Setting aside rapidly released worm such as `Witty`, there is strong evidence that proper practices are not employed in this domain [113]. Indeed, MS-Slammer and MS-Blaster were released hundreds of days after the patch publication, however, were still widely successful in their missions [112]. For instance, the objective of MS-Blaster was to create a DoS attack on Microsoft's WindowsUpdate website. Despite bugs within the source code which eventually slowed the malware down, Microsoft decided to remove the site's address from its **D**omain **N**ame **S**ystem (DNS) servers, thus, the worm's author achieved the required DoS attack objective [16]. The factors illustrate that when it comes to security, time is key issue [22].

Furthermore, patching is not a transparent process as it often stops users from using the computer host while it updates and reboots. Therefore, administrators have limited periods of time where they can perform system updates without causing major disturbances to the daily activities of an organisation. In addition, there is evidence where intruders will not wait for administrators to update their systems nor for them to be at their desk [131]. Indeed, Moore et al. [4] show that network intrusions, such as DDoS, occur mostly at night and on non-business days. Consistency is thus paramount in thwarting malcode outbreaks [58, 55]. Researchers do not argue against the benefit of patching [125], however, they agree that too many situations showed that security should not rely on end-user intervention [59]. Evidence of end-user unreliability can be found in Hughes and DeLone [58]'s analysis of malware outbreaks, as well as Baker and Wallace [55]'s survey of information security management.

Yegneswaran et al. [113] uncovered the reasons behind the persistence of some malware in the wild, months after the main outbreak, in what could be argued as enough time for practitioners to properly mend their systems. Their analysis of the

Code-Red worm reveals that the malware is programmed to propagate on a specific date, and then destroy the original copy. Hence, they showed that the cyclic rise in numbers of nodes compromised by Code-Red is due to the clock offset of the hosts, and the fact that administrators do not always re-apply patches. This represents a failure in management, as the necessary steps are not performed for each system being deployed, or re-deployed. This shortcoming is not limited to patch management, though, as Wool [107] makes the same observation regarding network firewall management.

3.6 Defence Mechanisms

This section continues the critical appraisal of the various mechanisms involved in securing computer networks. Systems that can implement or enforce security policies, such as network firewalls, are the focus.

When the Internet emerged as being a suitable way to conduct business, it was assumed that the internal network was trustworthy, and the outside world was untrusted [72, 73]. Thus, organisations with Internet access often relied on a perimeter defence system to block unwanted and malicious traffic. This is best achieved with firewalling technologies, and these are now discussed.

3.6.1 Proxies

Application proxies often offer better security than traditional network firewalls [80]. Yet, organisations do not often deploy proxy servers into their infrastructure [80]. Oppliger [73] predicted reduced adoption of this technology because it often requires the development of a specific application-layer gateway for each application that requires access to remote resources. This could, in turn, result in computation-intensive operations, and hence drastically affect performance. With respect to network's interoperability and functionality, Oppliger regards proxy servers as barriers to the implementation of new services and, in some respects, to network growth itself. Furthermore, organisations are often dependant on vendors to provide a more extensive range of supported applications, and it is unlikely that organisations are licensed to modify the proxy source code, or create add-on for application proxies. These are intermediate devices, and, like routers or firewalls, are known to be *closed* technologies [19]. Thus, for organisations it is essential to balance functionality and security requirements [65, 34, 21].

3.6.2 Trusted *vs.* non-trusted

Avolio [72] reports on the days when organisations were connected to the Internet without proper protection. This situation might be explained by the relative small

number of organisations connected, at the time, and the restricted services the Internet offered, such as file sharing, terminal access, and electronic mail. Due to the restricted community, there was thus a sense of mutual trust [73]. However, with the arrival of the World Wide Web, video streaming and multi-media, came more sophisticated attacks that exploited the openness of the Internet. Hence, in the early 1990's, organisations adopted firewalls to allow corporate users access to remote resources, without permitting intruders accessing to the corporate network. Publishing at the same period as Avolio, Oppliger [73] argued that the security of the Internet can only realised if access control, authentication, data confidentiality, and integrity are guaranteed to legitimate parties. In Oppliger's view, a firewall is a device that separates a trusted network from a less trusted one. This view is challenged by Ioannidis et al. [50]'s observation. With respect to current network's design and capabilities, Ioannidis et al. [50] outline that wireless and dial-up access are often examples of connections to corporate networks that avoid the firewall's scrutiny. For the time of the connection, nodes using these means are part of the network, but paradoxically may not be totally under the control of the organisation. Hence, the probability of hosts being compromised increases, and malware can exploit this type of connection to penetrate the network. Consequently, the paradigm of inside networks being trustworthy, and outside networks not to be trusted, is no longer relevant. This misconception often leads organisations to focus on the incorrect origin of most threats [87]. Indeed, the most successful intrusions often originate from within the organisation [47], and several legal cases illustrate this [97, 98]. Internal intrusions are more likely because insiders can investigate the network, and establish its limitations [49, 111]. Along with this, high traffic volumes can give a sense of anonymity which further increases the probabilities of intrusion [49].

3.6.3 Limits of Operations

On many counts, firewalls are an appropriate technique of defence, as its centralised model typically offers a single point-of-control, and facilitates the auditing of services [72, 73]. Firewalls are also best suited to deal with access control, as they can granularly enforce access to remote services, or resources. Still, there are threats for which firewalls are not so effective. Malware, for instance, often bypasses firewalls as they intrude the internal network with the help of an internal user. In many occasions, the trust of the user has been abused by bogus emails containing malicious payloads [73, 58, 132]. Indeed, firewalls often sit on the edges of a network, and they have a lesser effect in the event of misbehaviour from internal users [50, 73]. However, firewalls are effective against malcode propagation provided that strong filtering occurs on traffic going from one work-group, or **v**irtual **L**ocal **A**rea **N**etwork (vLAN), to another [112, 16, 71]. Weaver et al. [71] point out that this security strategy is only seldom used. This shortcoming's root causes include concerns regarding performance, and the level of expertise required to properly manage these devices [14].

3.6.4 Configuration Issues

As corporate networks grow, lapses in configuration can appear which, in turn, create loopholes that intruders can exploit [50, 14, 76]. This is probably due to the fact that network firewall configuration requires specialised knowledge [14]. Thus, it is often difficult to produce consistent, and error-free, rules for network firewalls [75, 16, 14, 76, 106, 107]. Furthermore, firewall configuration philosophy is diametrically opposed to that of hosts, and this often leads to rule-sets being deployed in the incorrect direction [107]. Indeed, what is normally considered as outbound traffic from a network node is typically treated as both inbound and outbound traffic by the firewall. Figure 3.1 illustrates this. When communication crosses the firewall's first interface, it is viewed as inbound, and then considered as outbound when it leaves the firewall's second interface towards the destination (Section 2.5). Whilst this is a simple example, firewalls are typically located at the crossroad of multiple network segments, and thus the distinction between origin and destination is much more challenging to make. In the end, such mistakes can result in situations where the firewall can be considered as being absent.

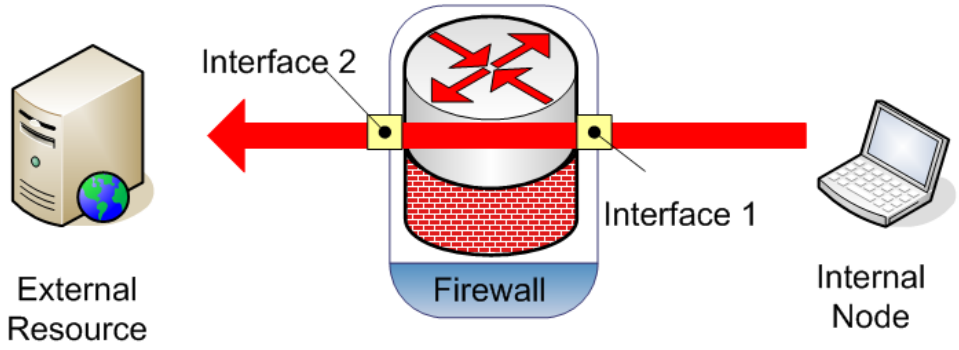

Figure 3.1 – Opposite point-of-view between network node and network firewall by Wool [106]. The arrow represents a request sent to an external resource. The firewall sees it as incoming when this request reaches Interface 1, however, the request is considered as outgoing once it leaves the firewall towards its destination. From the client host's point-of-view, the request is simply outgoing.

In addition, there are often limitations within the configuration syntaxes that aggravate the problem [107]. Firewalls are meant to protect end-hosts from compromise. Errors in configuration can have consequences beyond an increased number of compromised end-hosts, as these errors can lead to intruders gaining control of firewalls too [125, 71, 107]. No quick remedy exists for configuration problems as there is a limited number of suitable tools [77]. Hence, defects are typically difficult to diagnose, and can be time consuming to fix [16, 14], as, generally, network devices do not include facilities to revert back to previous suitable configuration, unlike with active network devices [19, 92].

Arguably, one solution is to validate the rule-set prior to its deployment on the device. Al-Shaer and Hamed [75] propose an algorithm that determines the sequence

of the rules that is capable of identifying some logical defects. In order to achieve this, they define a generic network firewall rule model, which can be translated later into platform specific syntax. This is different from Guttman [52]'s approach which relies heavily on Cisco ACL syntax. Al-Shaer and Hamed's rule model can granularly define communications, such as web requests, sent from one particular host in the network, and it requires that such communication only matches one item within the firewall rule-set, though [75]. Firewalls, however, can filter traffic according to many criteria including: protocol; source and destination addresses; source and destination ports (Figure 2.4, page 14). As firewalls examine rules in a sequential manner, if a packet matches multiple statements, Al-Shaer and Hamed [75]'s model considers that the rule-set contains defects. For example, the rule-set could start with denying communication from a host, however, later contains a statement permitting it. Along with this, rule-sets often contain inter-related statements which make such defects difficult to uncover [75]. Caldwell et al. [14] add that the management console typically offered by firewall devices limits the number of statements an administrator can examine at any one time. This observation is in agreement with Wool [106] who also identifies this issue as a source for misconfiguration. Thus, Al-Shaer and Hamed's algorithm iterates through the rule-set, where the analysis is performed using two distinct statements as inputs, and, if any defects are identified they are reported. A defect can then be of the one of the following types: Correlation; Generalisation; Redundancy; and Shadowing. The key objective of this approach is to ease firewall management [75]. While it is true that rule-sets produced in such a way contain fewer mistakes, administrators still have to contrast rule-sets with the requirement or deployment context. An error free rule-set could still be implemented on the wrong filtering direction [106].

Al-Shaer and Hamed [75]'s approach best applies to devices about to be deployed, nevertheless organisations also need to establish the security readiness of their existing devices. Administrators typically use vulnerability scanning tools in this situation, however this assessment method is often not sufficient [77]. Indeed, these evaluation methodologies put an emphasis on device weaknesses, such as being prone to a particular software exploit, however they do not allow the uncovering of problems within the device configuration. Arguably, a complete assessment would link firewall defects with assets that are made vulnerable because of these. This, in turn, allows the establishment of priorities, unfortunately though, there are very few tools available to uncover these defects. Since most network traffic is connection-oriented, Hoffman and Yoo [77] underline that most evaluation tools implement one side of the connection model, and they provide a test methodology which addresses this shortcoming. In turn, they provide evidence that the industry-standard Cisco PIX firewall allows malformed network packets through once the connection across the device is established, allowing intruders to use this malfunction to transfer malicious payload. Arguably, Hoffman and Yoo [77]'s framework has two shortcomings. First

of all, it requires the evaluation to be performed in an isolated environment, hence the device must be removed from the production network for a period of time, which can be considered counter-productive. Second of all, the administrators still have to contrast gathered information with organisational requirements. Nevertheless, the same evaluation carried out on a different system, a Linux Netfilter [133] set-up, shows that this filtering defect is not universal. Thus, organisations should always thoroughly evaluate security equipment, and this is regardless of prior knowledge or the manufacturer.

Caldwell et al. [14] introduce a solution which combines live activities and off-line analysis, thus providing a bottom-up approach to network devices management. This system typically reverse-engineers the configuration of the device, copies the content of the configuration console, and then parses and organises information into a database [14]. They argue that most administrators intended to deploy identical settings across the network. However, it is likely that the network is maintained by several individuals with different approaches or styles, implementations might differ. The main aim is to address the discrepancies between the requirement document and deployment reports, as these seldom evolve together. This usually leads to problems when modifications are required [14]. Thus, the database can identify recurrent statements, and in Caldwell et al.'s view, these recurrent statements embody the intended requirements. The key advantage of Caldwell et al.'s approach is that it is capable of contrasting the intelligence gathered, with data regarding the surrounding of the device, such as its position within the network topology. Eventually, the database can be used to create configurations based on best-practice templates. Arguably, the enhancement of firewall rule-sets prior to deployment is a suitable approach as most network firewalls do not offer on-board rule-sets improvement facilities [134].

3.6.5 Distributed Firewalling

Rigorous network control requires firewalls to be distributed, which is often a more complex task than verifying local enforcement. Arguably, this problem stems from the limitations within the firewall configuration method. Typically, firewalls provide syntax verification for the rule-sets, however, they are unable to perform semantic checks [52]. For example, it is possible for an operator to write statements, such as shown in Listing 3.1, where all traffic is denied, despite the fact that, at least, all Web traffic should be permitted. Al-Shaer and Hamed [75]'s algorithm will typically qualify this as a *shadowing* defect, as the first rule has a larger application scope than the second one. By the same token, Caldwell et al. [14]'s system would warn of network reachability issues. Hence, a major issue with firewalls is that their configuration is governed by the security policies, but not strictly linked to them.

Listing 3.1 – Cisco firewall ACL rules snippet

```
access-list 101 deny ip any any
access-list 101 permit tcp any any eq WWW
```

Caldwell et al. [14] argue that all devices should have the same configuration, and this is an argument that Yuan et al. [76] dispute. They stress that enhanced security policies should be tailored to department, in order to adapt to local work practices, or the type of applications used. In an organisational context, a department is typically regarded as a network segment, sub-network, or collection of sub-networks, dedicated to support an activity of the organisation, such as finance [64, 52, 10]. Thus, it is paramount that firewall rule-sets strictly enforce relevant policy, hence configurations are unlikely to be similar throughout the network. Essentially, Yuan et al.'s system operates in the same manner as Caldwell et al. [14]'s, and is capable of identifying a large array of logical defects in firewall rule-sets. However, Yuan et al. extend the system from uncovering errors within one device to assess the network as a whole.

Another method is to assess the network as a whole, from an operational point-of-view. Guttman [52] proposes to model the interconnection between networks and employs a methodology that describes the flow of communications between any two networks. In the situation where firewalls are not coordinated, an organisation might decide to deny a specific type of traffic coming from, say, the outside world, such as the Internet. However, this traffic could be an essential part of the collaboration with an allied corporation. Bakry [9] points out that when it comes to inter-organisation network connection, responsibilities are shared and advice can only be given, not enforced. Thus, under these circumstances, there is no guarantee that the unwanted traffic will not traverse the allied network to reach the organisation's network. With Guttman [52]'s approach, however, it is possible to cater for this type of issue. This challenge is abstracted with Figure 2.3, and is normally depicted by Guttman or Uribe and Cheung [53], among others, with Figure 3.2. Guttman underlines that some security requirements are better addressed with changes in the router configuration, as opposed to the deployment of large and complex firewall rule-sets. However, this often relies in the operator being knowledgeable enough to produce an adequate solution. Unfortunately, studies show that rule-sets and firewall configurations often contain errors because of human intervention [107]. Thus, relying solely on personnel is not sustainable [14].

3.6.6 Internal Subversions

One of the drawbacks of Guttman [52]'s method is that it cannot address the issue where unwanted hosts are connected to the network. These unwanted nodes benefit

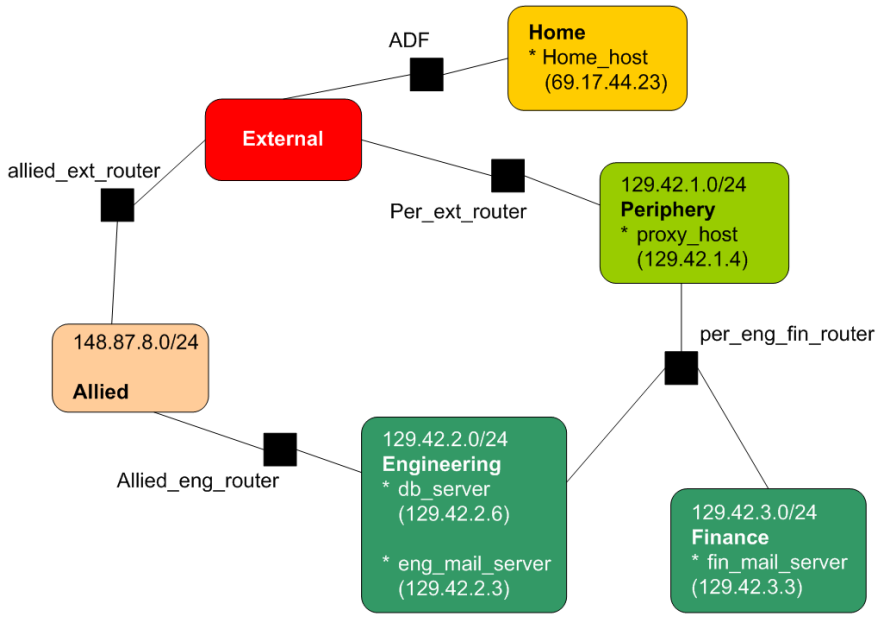

Figure 3.2 – Challenge of distributed firewalling as illustrated by Guttman [52].

from the firewall's protection, as they also gain connectivity to the nodes of the network this firewall manages, and even to the outside world. Also, there are several technologies that would escape the firewall's scrutiny, such as encrypted traffic [50]. Thus, Ioannidis et al. [50] propose a model where the filtering procedure is created as both hardware-based firewalls, and software host-based ones. The model is based on the concept of distributed trust management. Thus, in order to participate to the scheme, hosts must be enabled with relevant pieces of software to decode, interpret, and authenticate, according to the rules set at higher hierarchical levels. Hence, with an extensive number of criteria to fulfil before any communication is allowed, intruders will have to do much more than just physically tap into the network before they obtain connectivity. Indeed, Mitnick and Simon [64] show that this is an effective intrusion strategy. The model is particularly suited to deal with encrypted communication as only the source and destination are in a position to fully understand it [54]. Furthermore, the delegation capabilities allow for the mitigation of mistakes, such as giving away passwords, as access-rights can be revoked remotely. Since the filtering procedure is enforced down to the end-host, no process can access the network without the user's, or relevant authority's, approval [50]. As the emphasis of Ioannidis et al. [50]'s model is on host-to-host communication security, it should be used in combination with traditional firewalls, as these provide better means to thwart infrastructure-based attacks, and provide basic protection for devices that might not support the *distributed firewall* approach [50]. This illustrates Stajano and Anderson [117]'s argument regarding tried and tested security methodologies not being universally applicable (Section 3.3.2).

3.6.7 Inner Limitations

Along with the challenge of configuring network firewalls properly, organisations must also realise that these are not perfect devices, either [31]. Assuming a situation where correct configurations are in-place, intruders could still harm the network with a single packet [80, 16]. Airamo and Virtanen [135] show that choosing a network firewall requires more than a thorough study of technical specifications, and demonstrate that limitations may not appear until the device is actually deployed. Indeed, the network firewall they evaluated is unable to deal effectively with small network packets. Hence, organisations cannot rely on the *prestige* of some manufacturers as a measure of quality, or suitability, to their requirements.

Kamara et al. [31] propose an alternative approach to the selection of firewall solutions. Each of the vulnerabilities that a firewall, either implemented in hardware or software, might have is weighted according to the damage that exploiting the vulnerabilities can cause [31]. Thus, the quality of a firewall is determined according to the assets that are to be defended [31]. For instance, the down-time for a website might be acceptable, whereas a change of content might not, and these two objectives often do not require the same type of firewall. Since, as no two organisations have the same needs, or operate the same, it is likely that generalised solutions are not always adequate. Kamara et al. stress that a firewall's vulnerabilities are often easy to uncover [136, 123, 124, 137], however, their work is based on publicly-available vulnerability databases. Hence the benefit of their approach is limited to vendors willing to publicly acknowledge faults in their products.

Arguably, network security standards are inadequate due to the simplistic manner in which network firewalls are deployed. Wool [107] studied several firewall products, and their capabilities to address firewall management mistakes. It appears that manufacturers eventually enable their devices, and associated software applications, with features that correct some firewall implementation issues, such as filtering spoofed IP addresses on the incorrect network interfaces [107]. These findings are contrasted against some firewall configurations used in production networks over several months [107]. Organisations do acquire new devices, however the survey shows that administrators often reapply the previous configurations to new devices. The past mistakes are then usually carried over, and hence the value of the investment is diminished, and organisations, thus, do not benefit from enhanced security solutions. Hence, Saliou et al. [21] identify these as a major obstacle in computer network security which is the lack of accountability between the decision making-process, the subsequent implementation on live devices, and in the verification of compliance.

3.6.8 Performance Issues

Network firewalls are often combined with other key network functionalities, such as PAT/NAT translation [138, 139] (Appendix D). Moreover, firewalls can be subjected

to congestion, and thus considered as a single point-of-failure. These devices will typically have to cope with increased demand in bandwidth, without being seen as an obstacle to performance. There are typically three types of performance characteristics when it comes to firewalls [138, 32]:

- The efficient use of the resource available on the device, such as in avoiding CPU overload, and on-board memory exhaustion.
- The rules-to-match ratio of the rule-sets implemented on the device, this relates to the number of firewalling statements the device enforces compared to the number of statements triggered.
- The repercussions on the network characteristics, such as communication latency, available network throughput, number of packets dropped and so on.

Hence, organisations need to establish the limit of operation of their network devices.

One approach is to study the characteristic of the filtering algorithm the device employs [140]. This is an important piece of information as Gupta and McKeown [140] identify that some types of rules may not be as efficient as operators anticipate. Their focus is not solely on embedded systems, it also includes a detailed study of filtering algorithms, as well as hardware specific implementations [140]. Unfortunately, as with Kamara et al. [31], Yan et al.'s analysis is limited to documentation which is public available, and does not provide additional insight for closed platforms, such as Juniper or Cisco devices, which are often industry standards [77].

Firewalls are often embedded devices, and performance is ensured by distributing tasks across specifically designed chips [138]. **N**etwork **P**rocessor (NP) technology offers a middle ground between integration and flexibility, as the inner working of the device can be modified [141, 134]. When NPs are employed as network firewalls, it is typically the filtering algorithm that is altered, and the implementation methodology directly can impact performance [141]. Srinivasan and Feng [141] studied a filtering algorithm which can be deployed using either a pipelined, or parallel implementation, and the results demonstrate that the parallel implementation offers better performance. However, the complexity of the packet filter algorithms influence which implementation method can be used [141].

Piyachon and Luo [134] argue that NPs are seldom used to their full potential. Usually, NPs contain several hardware elements which are designed to perform a set of tasks efficiently. In order to prove this point, a parallel implementation of a filtering algorithm is mapped to the hardware features offered by an NP, and this results in improved memory usage [134]. Overall, the benefit of NP relies directly on studies completed before the device is deployed in the production environment. As with the modelling methodologies discussed in Section 3.3.2, this approach is one that only few organisations can afford.

A determinant factor in this domain is the number of rules a firewall can enforce [141, 138, 140]. Hamed and Al-Shaer [142] explain that network firewalls will typically include only a few complex statements that only match a limited amount of

packets from the network traffic in comparison to the overall rule-set size. Arguably, enhanced filtering devices must be scalable, and offer a trade-off between capabilities, and performance [142]. However, performance for Hamed and Al-Shaer [142] does not relate to network performance, such as poor network latency [107, 143], it relates to the usage of the resources available on the device, such as memory space, CPU usage, and so on. Therefore, their main aim is to enhance the rules-to-match ratio in the network firewall. To that end, they propose to re-order statements within the firewall rule-set according to the type of traffic the device is subjected to. This is possible because Hamed and Al-Shaer [142] combine the sequential nature of firewall filtering with an improved verification algorithm that they originally presented in [75]. Hence, a large number of network packets are matched with statements that appear early in the rule-set, and the number of statements that the device has to examine, or range lookup, is reduced.

Most studies focus on the effect that network packet size has on firewall performance [144, 135], due to the fact that the firewall treats small and large packet equally, and there are a limited number of packets the firewall can manage at any given time. Unfortunately, intruders often exploit this limitation to launch DDoS attacks [4, 139], may require organisations to know precisely the content and nature of the network traffic in which the device will evolve, which is unlikely to be easily determined. This lack of *context* could lead to network firewalls hindering the operations of some applications [145]. It is thus more appropriate to evaluate network firewalls under conditions as close as possible to live environment situation [78, 143], and such an approach allows organisations to determine the repercussions their device platform configurations have on network characteristics. Their configurations can then be assessed in terms of the number of rules network firewalls enforced, and the impact on key network characteristics, such as latency [78]. As shown in Hoffman and Yoo [77], it is essential that this environment is device independent in order to permit comparisons between various manufacturers, such as Saliou et al. [143] illustrate.

3.7 Intrusion Detection and Intelligence Gathering

Section 3.6 highlighted the key strengths and weaknesses of network firewalls, and even the best firewall deployment may not be sufficient to counter all the attack strategies that intruders have at their disposal. Thus, network firewalls must often be used with other security mechanisms [146, 16, 53] that are capable of understanding network activities. In other words, this section focuses on systems that embody the detection principle of security (Section 2.3.1, and Table 2.1).

Computer network security can be akin to traditional warfare [23], as such how current, accurate, and relevant intelligence information plays a crucial role in mitigating problems. For host-based IDS, data gathering is done on devices enabled with logging capabilities, and such implementation requires the collected data to be

sent to a central processing node in order to gain an understanding of the situation. For network-based IDS, observation can be carried out directly on the wire with an IDS. Accurate data and intelligence gathering is thus an important factor for the success of any mitigation procedure based upon the information collected. Paxson [119] demonstrated that the difference between malicious traffic and legitimate traffic is minimal, hence, proper data gathering and analysis is essential. This section analyses the various methods involved in inspection and intelligence gathering in computer networks.

3.7.1 Intrusion Detection Systems

As two networks are never identical, it is important to underline that IDSs should be chosen accordingly to traffic characteristics. Huseyin et al. [44] summarise that a signature-based IDS is effective for common form of attacks, whereas, according to Lippmann et al. [105], anomaly-based systems are suited to uncover new, unseen intrusion, however, their rate of successful detection is often low. Thus, the aim of an IDS is to extract and analyse information, or traces, from network communications. There are limitations in this process, such as with encrypted communications, as only the communicating parties can interpret the content [54].

Regardless of their engine type, the number of alerts IDS systems generate is a major drawback to their deployment. Huseyin et al. [44] stress that ultimately it would be the duty of an operator to make sense of the logs produced. Thus, this could lead to a situation whereby the benefit of an IDS is tied to the operator's skills [84, 14], and, by the same token, too many alerts creates a certain anonymity that intruders could exploit [49, 55].

Enhancing Detection Efficiency

Julisch [90] proposes to enhance the quality of generated alerts by applying an analysis method capable of uncovering the **root cause** of a particular alarm. Indeed, in most medium-to-large scale networks, devices could be mis-configured [14, 107], and, in turn, these would generate traffic that would be seen as intrusion without, in fact, being a danger [90]. Also, large amounts of such traffic could then mask genuine intrusion. Their study reveals that up to 90% of the alert logs are constituted by wrongly configured devices, and as a computer network's mission might change over time [147, 13], so could the type of intrusions. Furthermore, to achieve the results presented, an analysis of monthly logs could be needed, which shows how time consuming this activity might be. Moreover, in order to accurately establish causes, it is necessary to deploy multiple IDSs throughout the network, and hence overcome issues, such as consistency, correctness, and completeness of the distributed rule-sets [148]. Arguably, this observation applies to firewalls as they are also rule-based

systems. It is thus a sustained effort to keep the knowledge-base, or intelligence, up-to-date with current threats, however Julisch [90] advises to audit IDS rules regularly, so that outdated rules can be removed. This, in turn, allows the system to be more efficient.

Alternatively, organisations could use Ning et al. [57]'s approach to separate *false positives* from *true positives*. This approach is based on the fact that intruders go through stages to perpetrate their intrusion. Consequently, it is unlikely that an isolated alert represents an intrusion. Thus, operators only need to focus on strongly-correlated alerts. This approach enables operators to create adequate responses to threats, and is particularly suited to uncover intrusion stages spread across long period-of-time [57]. More importantly, intrusions with missing stepping stones can also be uncovered, which is typically when the data-set on which the analysis is based is incomplete. Indeed, Graves et al. [85] show that IDSs cannot always cope with high traffic volumes. The data-set can also be affected by the manner in which the data is collected. This is the case for Moore et al. [4] where the set is incomplete as many networks use load-balancing technologies, and thus sections of the intrusions are missing. As it stands Ning et al.'s approach is off-line based, and thus does not allow thwarting on-going, or imminent, intrusions. This is unlike Zou et al. [22]'s approach that generates a signal which could be acted upon, such as triggering the reconfiguration of network firewalls [21].

Arguably, anomalies should not be mistaken for intrusions attempts or actual attacks, hence data regarding intrusions should be gathered from multiple vantage points in the network [86]. It is likely that intrusion information originates from several systems which employ their own unique syntax for rule-sets and alerts. This, in turn, places an additional burden on administrators in terms of depth of knowledge and interpretation of the various log formats. Thus, Yan et al. [86] propose to use a multi-layered approach where the specifics of each IDSs are abstracted into an unique format: **P**rincipal-subordinate **C**onsequence **T**agging **C**ase **G**rammar (PCTCG). This new repository is then used for tasks such as attack correlation. This approach allows for the focusing on *semantic meanings*, as opposed to *syntactic details* [86]. In addition, this system keeps track of events that have already occurred using context details, such as similar time blocks, network IP addresses, and so on. The resulting information then serves as a basis to correlate new events as they occur. Like Ning et al. [57], the system aims to enable administrators with the opportunity to deploy a mitigation solution before intrusions reach a critical stage [86]. Unfortunately, this still relies on administrators knowing the consequences of attack scenarios, and determining the next phase of the intrusion. Without these two elements, it is not possible to develop an adequate solution.

Performance Issues

Like with network firewalls [138], researchers are often concerned with the effect of higher link speed and increasing number of networked application have over IDSs' performance [146, 119]. Hence, there are research efforts on lowering the sensitivity of IDSs. Antonatos et al. [70] investigates the characteristics of algorithms currently used in IDSs, such as Snort [69], and show that the poor scalability of IDS is partly due to the detection algorithms themselves. They formulated a hypothesis that the rest of the problem might be due to poor memory management performed by the system. Based on these observations, they propose an alternative algorithm called Exclusion-Based String Matching [70]. To evaluate the benefit of this algorithm, they replaced the Snort's engine with their own engine and results confirm that the improved algorithm led to improved performance, in particular with high network speed and larger packet sizes. However, they underline that a limited amount of traces were available to them, and therefore their system still needs to be evaluated in a more realistic set-up. Sommers et al. [149] reinforce that this aspect is crucial as attacks could produce traffic patterns that most security devices cannot cope with. This is correlated by Moore et al. [4] who provide evidence that for some intrusion the success factor relies entirely on the poor resilience of network devices. Hence, realistic evaluations are essential as Sommers et al. [149] show that some security devices are unable to operate properly, even at relatively low malicious traffic levels.

3.7.2 Network-Based Observation

SNMP allows for distributed data gathering, and also permits the collection of information from intermediate devices, such as for network switches or firewalls. Rayan et al. [82] argue that SNMP is no longer suitable for modern computer networks, as it could consume a lot of CPU cycles on the devices on which it is activated, and more importantly SNMP only provides raw information. Thus, it is often necessary to process the data before any kind of decisions can be formulated, as it requires a dedicated node with high capacity computational resources [82]. Furthermore, as networks grow, the volume of data transiting to the management node also increases. Hence, SNMP increased network utilisation can impede other operations carried out on the network [82].

SNMP also represents a challenge in terms of configuration and maintenance [82, 83], and this is perhaps due to the system-centric nature of the protocol. In other words, it requires each node of the network to be configured individually in order to operate. Lopez de Vergara et al. [83] created an management ontology for SNMP to address this aspect, and, in turns, it empowers administrators to make informed decisions regarding network management. Their approach is particularly suited to uncovering areas where system information overlaps, such as when two organisations merge. The merger of multiple separate networks under one management method

is feasible because it is possible to define behaviour and constrain properties for the software agents [83]. Hence, Lopez de Vergara et al. [83] argue that the ontology repository transforms these agents into *reasoning processes*, and that their system is thus an example of an informed decision-making paradigm.

3.7.3 Node-Based Observation

The end-host is the final element that accesses data, thus it is important to enable it with sufficient abilities to thwart intrusions. This is typically achieved with anti-virus software [128, 58, 55]. While organisations can build their own signatures, or rules, for an IDS as a function of the assets they wish to protect [90]; when it comes to anti-virus software they are typically dependant on the efficiency of the anti-virus vendor [150], as anti-virus frameworks are typically closed [128]. In addition, modern work practices increase the difficulty of keeping all the hosts up-to-date, such as with roaming users, in particular [55, 58]. Baker and Wallace [55] underline that an organisation may define such a policy, however their survey reveals that compliance with this is often not verifiable to a reasonable standard. This is a major drawback, as malware outbreak prevention typically requires at least 90% of the network hosts to be immunised and equipped with the lastest threat databases [55]. Normally, anti-virus databases are refreshed automatically, however Whitman [116] warns that the anti-virus policy could be developed seperately from the network access policy, such as that enforced by firewalls, for instance. Hence, automated updates may be stopped by the wrong configuration in firewalls, and thus the benefit of anti-viruses can be severely lowered [116].

One characteristic of malware is that it can propagate quickly [22, 59, 151, 58]. Indeed, a worm, such as `Witty`, would only need about one hour to traverse the entire Internet, and most anti-virus vendors generally produce accurate signature within two hours of the first incident being reported [151]. This estimate does not factor-in the deployment of the new signatures to the end-users though. This evidence leads to the conclusion that vendors have reached the limit of what is humanly feasible in this situation [151, 58]. Usually, anti-virus software which does not have their threat database up-to-date, will not block a process, even if it exhibits malicious behaviour, such as randomly deleting core system files [150]. Thus, in the light of the vendor's responsiveness being inadequate to properly thwart malcode, there are calls for behaviour-blocking technology to be developed, and for anti-virus software to be used as a corrective tool, rather than as a detective one [150]. A key concern is thus that organisations rely on third parties for the security of their systems. Arguably, malware outbreak can be thwarted by analysing the data collected from multiple sources [121]. Such an approach is necessary for threats which are designed to evade detection, such as metamorphic viruses [128].

Hughes and DeLone [58] point out that malware propagation might only constitute part of an attack. Since anti-viruses software only focus on thwarting malcode,

it is essential to use additional information sources to uncover threats. There is a wealth of information available on each node of a computer network, and not using data contained in operating systems logs can be a waste of resource [84]. Indeed, logs are often suited to discover attacks or compromised assets, as well as present evidence of failed intrusions [88]. Unfortunately, these logs can generate large volume of information, and are also often unsecured [88], however, centralising the logs can minimise the administrative overhead for system administrators, and encrypting the aggregated data improves the integrity of logging [84]. Thus, analysis can be based on the newly created repository. As there is no standard for the information stored in these logs, Chuvakin [84] proposes to organise the information into a database, and issuing simple queries, which is a position opposite from Laurie [88], who argues that the usefulness of logs is tied to administrator interpretation skills, and this limitation also occurs in other security domains, such as in network firewall configuration [14]. In addition, he points out that logs are unsuitable as a basis to uncover malicious activities through automated analysis tools. Indeed, logs were often not designed for this purpose, and intruders can go to quite a length to erase their trail, and he proposes that assets, such as web servers, are enabled with dedicated forensic modules designed to uncover intrusions, including the integration to the web service, itself. Hence, the module is transparent to users and intruders, alike, and thus less likely to be tampered with, or compromised. The specification also includes output data which is suitable for automated analysis, and does not actively look for intrusions, however seeks to ease forensic analysis. Unlike Chuvakin [84], Laurie [88] only partially addresses the data centralisation issue. Laurie advises to record log data from the Web server on a separate node, however the forensic module then becomes more vulnerable to DoS logging attacks. These could be the consequences of an intrusion targeting the logging node, or due to the volume of data the module itself produces. Incomplete data ultimately reduce the possibilities of uncovering problems [81, 85, 67].

3.8 Dynamic Networks

Section 3.5, 3.6, and 3.7 have demonstrated that security in computer networks is typically passive, static and reactive. Ultimately, the social and hierarchical structure of an organisation should be visible within the configuration of networks. Hence, it is desirable for a distributed system to be capable of reconfiguring itself in a timely-manner to reflect changes in policy, in practices, and in social hierarchy, such as the promotion of a member of staff, or in the face of security threats, such as in malware propagation. Hence, this thesis makes the hypothesis that either **M**ulti-**A**gent **S**ystem (MAS), or **A**ctive **N**etwork (AN) could be appropriate to create dynamically reconfigurable computer networks. This section reviews these technologies, and shows

that the key issues behind their limited adoption, which include performance concerns, and requirements for strong policing.

3.8.1 Agent-based Systems

Security is often carried-out in a distributed manner, as networks and users also tend to be distributed. Thus, a MAS designed to adapt computer networks to threats would integrate, or manipulate, functionalities such as with firewalling and IDS, with the functionality to both detect threats and deploy mitigation procedures. The information extracted from these sources can then be exchanged, and acted upon, accordingly. However, it is essential to define the boundaries of such activity. Software agents are goal-oriented, and this attribute can be used to divide complex tasks into small manageable objectives [25]. For instance, such a task could be defined as stopping the policy circumvention of a file server. Arguably, this would involve [21]:

- Identifying the elements of this mis-use, such as with the use of IDS signatures.
- Correlating these alerts with, for instance, information available in network device logs.
- Establishing the location of malicious hosts by means of their IP addresses.
- Reconfiguring network devices to stop access from malicious hosts, such as deploying the adequate rule-sets on network firewalls.

In MAS, each agent is tasked with a simple issue to solve. Santana Torrellas and Villa Vargas [25] underline that sub-problems have a casual relationship with one another, and thus they designed a MAS based on this property. Consequently, agents within their system will have an understanding of their specific tasks, as well as the impact on the overall system, such as consuming more resources than the organisation has allocated [18, 43]. In other words, implementing a change that results in a network latency of more than 52 ms will be detrimental to VoIP [108], and this could be significant if the organisation sees VoIP as criticial to its operations. Therefore, agents will not undertake actions that could compromise the aims of the organisation, or that no longer fit the organisation balance of needs [85]. Arguably, this particular feature addresses concerns from other researchers, such as of Staniford et al. [20]. Indeed, agents are autonomous software programs and, thus, could exhibit the same characteristics as malcode. Hence, their activity has to be constrained, and monitored.

There are a number of challenges in deploying a MAS in production environment. Saliou et al. [21] argue that MAS principally face difficulties in terms of interoperability with networked systems and applications. Thwarting threats, such as malware propagation, will typically involve modifying the configuration of devices, such as network firewalls, or vLAN switches, however these are often *closed* devices [19] and they might be physically inaccessible; and run proprietary operating systems that cannot be altered. Thus, these devices can not integrate the run-time environment

necessary for most MAS to operate and fully exploit the advantages of agent technology [91]. Instead, MAS will have to incorporate facilities to adapt to devices, and conform with remote configuration methods [21]. This could constitute a substantial overhead depending on the number of devices to support. Moreover, such MAS will have to understand the complete network topology and suitable mitigation strategies. Hence, a key issue is the definition and maintenance of the intelligence, or knowledge base, of the MAS [21].

Lopez de Vergara et al. [83] argue that an ontology approach is most suitable to define MAS intelligence, as this is regarded as an efficient method to share knowledge amongst distributed elements [152]. It allows assembling meanings and values from relevant domains, and, in turn, agents do not need to be pre-programmed [82], as they extract the necessary information from the ontology database [82]. This allows for the adaptation to environments that contain verbose information, such as with DDoS types [152, 26], or from data with SNMP [82, 83].

Rayan et al. [82] investigated the use of MAS as a mean to replace traditional management methods, such as with SNMP. One of the key advantages of agents is that intensive dialogue with a managed node can happen locally [91, 82]. Normally, SNMP data must be processed at a central location, however with Rayan et al. [82]'s system ,only relevant information is sent back to the central management node. Arguably, this approach can potentially reduce the network bandwidth consumption dedicated to network management. Indeed, if management traffic is too high, it could impede other network activities [82]. Rayan et al.'s approach is thus not always suitable, in particular for small networks, and it is more suited for growing networks because of lower management overhead, and the fact that with large infrastructure, the impact on network activities is minimal.

3.8.2 Active Networks

Example of capabilities

Alexander et al. [92] best demonstrate the capabilities of ANs. In traditional networking environment, it could take years to design, refine, and ratify a new network service, or protocol. They demonstrated that an active node can be taught, or remotely programmed, to interface between two networks that are not running the same version of a protocol, and this is accomplished in a total transparent manner to the network user. Along with this, these experiments highlight an important advantage of ANs over traditional networks. In traditional networks, if a new configuration is applied and generates more faults, or errors, than before hand, the system is often not capable to adopt the previous suitable configuration without the intervention of an operator. This could be complex, time consuming, and often requires in-depth knowledge [16, 14]. Alexander et al. [92] presented evidence of an automated fallback when the active node determines that the newest protocol is proven not to work.

Their work is a milestone in the creation of flexible and secure networks, and more importantly, their experiments show that it is possible to build up *functionalities* in network devices. In this instance, a simple repeater becomes an Ethernet bridge able to run complicated tasks, such as for the **S**panning-**T**ree **P**rotocol (STP), according to the specification provided in the switchlets (*cf.* Section 2.7.2).

Enhanced Network Management and Security

ANs can also be used to solve modern network's issues, such as in management, congest control, multi-casting, filtering, and so on [93]. Conversely, the challenge is to retrieve the relevant information from intermediate nodes and vantage points around the network [93]. In Li and Wolf [93]'s view, there is benefit in using the per-application and per-user property of ANs. Thus, their model allows the user to specify the content, the format, and the origin of the required data. Hence, this represents an enhanced method of data collection.

Wang et al. [51] use the characteristics of ANs to enhance network security, and propose to use the capability of active nodes to trace back intruders. Indeed, many intruders target end-hosts and camouflage themselves by using multiple compromised hosts to carry out their intrusion. Hence, Wang et al. created a framework where servers, running an IDS, will work hand-in-hand with Internet gateways. Whenever an intrusion is uncovered, the node will notify the nearest gateway by inserting a *watermark* into network frames and, upon reception, the gateway will relay this information to other collaborating devices. The combination of this information will then pin-point the intruder to the nearest trustworthy host. This model can help towards the implementation of a dynamic intrusion blocking, or containment, system [51]. Actually, Briesemeister et al. [13] show that active nodes enhance the survivability of networks in the face of a threat. Furthermore, since the watermarking circulates using an Active Environment (Section 2.7.2), it does not allow the intruder to notice that the node is reacting to the intrusion [51]. Weaver et al. [71] concur that adapting the network to threats, such as closing communication ports to hinder malware propagation, is a solution that should be explored [22].

By the same token, this kind of capability could lead to a situation where active nodes could try to disable remote malicious hosts [153]. This might raise the attention of the intruders and increase their determination [111], and retaliation is probably not an advisable approach [99]. Indeed, intruders often have access to a large number of comprised computers, as well as other networked resources [45], and these resources can be large enough to completely disable an organisation and its networked infrastructure [99].

Performance Issues

Just as mentioned in Alexander et al. [92], Campbell et al. [19] are of the opinion that AN have a great potential for future networks. This is because they: permit deployment of new services without requiring physical access; employ technologies which are more open than in traditional networking; and do not incur an outage of access, or usage, for users when changes in configuration, or offered services, are applied. Li and Wolf [93] summarise the lack of adoption of the technology by organisations down to the fact that ANs offer lower performance, such as in terms of *throughput* and *latency* compared to traditional networks. Alexander et al. [92] provide extensive evidence of this drawback, and Li and Wolf observe that traditional networks are so widely deployed it leaves no doubt that ANs will never replace them. Nevertheless, some researchers are working on addressing the performance issue of ANs.

Whilst Alexander et al. [92] propose to investigate and enhance the operating system which manages the Active Environment (Section 2.7.2), such as through the Java™ Virtual Machine [154], Lockwood et al. [155] apply the principles of AN to **F**ield **P**rogrammable **G**ate **A**rrays (FPGAs). Usually, microprocessors are designed in such way that permits the running of multiple types of applications or tasks. In turn, each of these tasks is not always performed at the best of the hardware's ability. On the other hand with FPGAs, it is possible to modify the internal structure of the microprocessor [155]. Therefore, Lockwood et al. [155]'s improved hardware device has the capacity to receive, understand, and adapt to instructions sent dynamically; more importantly the device produces an architecture that best suits the actions demanded, and demonstrated that the integration of FPGAs technology can enhance both ANs' performance and flexibility. In ANs, FPGAs offer great capabilities, however their resources, such as in terms of the available transistors, is a limiting factor. Thus, an evaluation of the target system capability is required prior to the deployment of new features, or functions [155]. Arguably, this limitation is less apparent in ANs that use an Active Environment which rely on software platforms, such as the Java™ JVM [154].

3.8.3 Enhanced Network Resilience

The benefits of dynamic networks can be seen in Briesemeister et al. [13] or Wang and Wang [147]. Briesemeister et al. [13] studied malware propagation within network with a high number of nodes, using simulations. They argue that networks are built to fulfil objectives, and thus they aim to help administrators to select adequate network topologies which ensure system survivability with respect to this. Under these circumstances, it appears that some network architectures still offer adequate connectivity while worms are propagating, and others foster the faster spread of the malware while still providing improved reliability in comparison [13].

Wang and Wang [147] also aim at preserving the network's mission, and propose to take into account parameters, such as *infection delay* and the *user vigilance*. They propose to integrate a system that would delay, or quarantine, traffic using active network elements. This quarantining restricts traffic not seen previously by the system. However, a balance must be found in the usage of such method, as significant delays could lead to problems, such as in real-time systems, or VoIP applications [147, 108].

3.9 Summary

The security of networked system is often a time sensitive issue, both in terms of deployment of initial requirements, and the adoption of new techniques, or configurations, to thwart security threats. Indeed, intruders often use the loopholes in the interaction, and interlinkage of, network elements to achieve their objectives. Computer network security owes its complexity to several factors ranging from legal requirements, to human involvement and technological limitations.

This chapter demonstrated that legislations do have an impact on the manner in which organisations can operate their computer network systems. These laws are typically two-way systems, whereby compliance facilitates justice proceedings for organisations in the case of security incidents. Moreover, balancing technical capabilities with regulation appears to be a major challenge in this area. This can be attributed to the fact that there are few adequate standards that can guide organisations through such a procedure. Thus, it further reinforces the need for rigour in the definition of security requirements, otherwise it becomes more challenging to prove that a security incident occurred, for example.

The legal contexts should thus be embedded into security policies, and one of the drawbacks with this is that they are often difficult to define, and, might also include requirements which the underlying system cannot meet. Along with this, administrators typically have to interpret requirements to adapt them to the capabilities of the equipment. Thus, a weakness of security policies is their strong emphasis on the top-down approach, where decisions are made, and expected to be thoroughly implemented on networked systems. Hence, a major shortcoming with security policies is that they often do not contain both the organisational objectives and the associated verification methods [21]. This is the case for patch deployment, for instance, which typically gives the illusion of security, however gives no guaranty of permanent immunity from threats, such as computer worms, as patches are not re-applied when required [113].

Many security issues can be associated with the involvement of human operators, and there is strong evidence that technical methods, alone, cannot solve computer network security in its entirety. Proper training and awareness, for instance, can thwart intrusions that use bogus emails, and represent the most effective method in preventing information leakage, thus limiting damage to the organisation's public

image [47, 3, 8, 132].

Security equipments and techniques are continually improving, however the adoption of enhanced items is typically slow [15]. This can be attributed to the fact that there are very few approaches available to assess the suitability of new methods. This is particularly true of organisations with existing, or large, computer network systems [21]. Along with this, the over-emphasis on some technologies leaves organisations greatly exposed to emerging threats, such as personal data theft through web browsers [8].

Ideally, all computer network elements should be configured according to the organisation's security policy, and, network firewalls could represent the device type that most closely relates to the organisation's requirements. Thus, firewalls are often the cornerstone of organisation security, as they can effectively enforce requirements in terms of authorised protocols, and applications. On the other hand, firewalls are typically transparent to most network activities, and this leads a reduction of resources, such as in terms of staffing, to administer and maintain these devices. Along with this, administration and maintenance are made more challenging in terms of uncovering configuration defects as well as concerns over network congestion and performance. One issue with firewalls is that they can be deployed in live production environment free of configuration defects, and so on, without a strict and verifiable compliance with the organisation requirements.

It is also challenging to secure all nodes against intrusions [23], and, developing adequate mitigation procedures requires information to be gathered from several vantage points within the network. There are also issues with the ability of these systems to obtain data [149] and for the quality of evidence [85, 67], as the aggregated information is non-trivial to process. This procedure, though, can be enhanced with a clear definition of the data required in the security policy. Along with this, a major drawback of IDSs is the amount of information generated as this can overwhelm administrators and thus, in turn, prevent the timely deployment of adequate mitigation solutions. Arguably, such tasks should no longer rely on network administrators, and instead the analysis should be performed by dedicated systems whose results can be used as an input for an automated mitigation system [22, 21, 20, 71].

Such functionality will thus require addressing the rule-based and static nature of computer network systems. This chapter analysed technologies that would permit enhanced network adaptability, and hence these can permit deploying mitigation procedures in a timely manner. Evidently, these new capabilities should not be subverted, and thus this chapter highlighted the need for strong policing. In addition, if these technologies are to modify the network's configuration to mitigate problems, they need data input that includes the organisational context.

3.10 Conclusion

While there is a strong body of work when it comes to the logical aspects of network security, these repercussions need to be measured when the network devices are in use, thus for their dynamic performance. Indeed, this chapter demonstrated that security requirements may not be fully deployed due to performance concerns [143, 28]. This chapter highlighted strong shortcomings when it comes making performance properties relevant to organisations [15], and this is especially true for the cornerstones of organisation's security: network firewalls (Section 3.6). The impact of security policies onto network devices is not well understood, and this stems from the fact that organisations do not possess, or have access to, facilities that can measure the performance footprint of security implementation on key resources, such as for available network bandwidth [18, 32]. More importantly, perhaps, this shortcoming would prevent organisations from complying with new legislation that requires strong evidence of computer network fitness *prior* to alleged security incidents. Such information is especially relevant for threats that seek to exhaust network resources, such as DDoS attacks [99, 100]. This is important in terms of understanding how networks are likely to cope with security breaches, such as DDoS. Indeed, these attacks often rely on the exhaustion of resources, such as available network bandwidth, to succeed [4, 18]. Hence, it is paramount to measure the impact different security policy requirements have on network resilience, and, to that end, the following chapter presents an automated **D**ynamic **E**valuation **E**nvironment for **N**etwork **S**ecurity that recreates the production environment. It also outlines a novel network firewall dynamic performance model, and describes the manner in which the environment controls most of the network characteristics, hence providing a repeatable and rigorous modelling of network firewall devices.

Chapter 4

Network Firewall Dynamic Performance Model

4.1 Introduction

This chapter presents a model for network firewall dynamic performance in which the criteria that influence network firewall configuration are used as inputs for dynamic evaluations. The outcomes and results of these evaluations can then serve to build a more complete model for the dynamic performance of network firewalls. This can, in turn, allow for establishing the feasibility of a particular security policy within the means of the organisation.

The challenge of creating network firewall dynamic performance models resides in the fact that network simulators tend to focus more on end-to-end communication modelling [156, 157] than security, and thus their security device models are not always accurate [156]. Thus, it is an improvement to build these models with data obtained from actual experimentations [32, 78, 143]. Unfortunately, these tests often cannot be carried out in live or production networks, and thus an important aspect of such tests is that they are carried out in an environment that retains organisational properties. This results in a novel DEENS that is subsequently employed in the following chapters to explore important firewall scenarios, such as for: increased security requirements; change in underlying networked equipment; prioritisation of rules; and the security policy complexity level. It thus presents a dynamic firewall performance model based on DEENS capabilities, and also details, from both technical and operational point-of-views, the complete evaluation procedure that is required to build one instance of the network firewall dynamic performance model.

4.2 Rationale

In order to satisfy the balance between functionality, security, and performance, the capabilities of the devices the organisation has at its disposal must be known [65, 34]. This, in turn, allows for establishing that the security requirements are achievable within the means of the organisation or operation constraints.

4.2.1 Lack of relevant performance information

Chapter 3 demonstrated that the logical aspects of network firewalls are well researched. Nevertheless, organisations do not often deploy enhanced solutions because the outcomes, or results, are often not relevant to the organisational context [15]. Typical examples include studies that focused on the effect on network packet sizes on network firewall performance. While this is relevant when assessing the processing efficiency of filtering engines [140], it does not reveal if the device is suitable for the activities that the organisation perform on the network. An improved approach is to measure the effect of security policies on organisation operations, such as in the transfer of large data file, real-time communications, like VoIP [108], and so on [143]. Indeed, concerns over reduced performance, such as increased delays to access services, and reduced network bandwidth, are often used as justifications for incomplete deployment of security policies [3, 78, 12, 71].

Section 3.6 presented a detailed analysis of research items that can enhance the security provided through the use of network firewalls, such as in terms of rigorous access-control [21].

Security policies enforced by network firewalls play a major role in the overall security of the organisation as these can ensure adequate protection against security incidents, such as malware outbreaks. For example, worms, such as MS-Blaster, SQL-Sapphire, and `Witty`, would have been effectively thwarted if network firewalls had been properly configured [112, 76, 59, 63]. Indeed, these malicious applications traversed network firewalls because these left several communication ports open. Along with this, network firewalls are often not configured to filter traffic in between internal sub-networks [71]. Therefore, these observations provide a strong basis for the creation of methodologies that complement research on logical firewall properties (Section 3.6.4), with relevant organisational data regarding performance, such as on the stress of intermediate devices, or key network characteristics, like latency [78, 158].

4.2.2 Related Challenges

Zhang et al. [32] best summarise the challenge of achieving a balance between performance and security as they highlight that demand for greater available network bandwidth is growing, and yet security policies are *thought* to drastically affect this network characteristic [50], along with others such characteristics, such as latency and network throughput. Thus organisations must deploy security policies with an acceptable impact on these elements [32]. In other words, organisations need to be able to measure the security policy overhead when their computer network system is live, and hence in terms of dynamic performance. This allows for the matching of an important security principle which stipulates that security deployments, such as rule-sets enforced by network firewalls (Section 3.6.8), are considered acceptable,

only if they are achievable within the resources that the organisation has allocated [18]. These resources include financial assets, staffing resources, as well as technical elements, such as for network characteristics, including: latency; available network throughput; on-board CPU usage; and so on. Arguably, decisions can be based on the results of simulations, or measurements performed in live environments.

Issues with Simulations

Arguably, one of the main advantages of employing network simulators, such as **T**he **N**etwork **S**imulator - 2 (*ns-2*) [159] or **Op**timized **N**etwork **E**ngineering **T**ools (OPNET) [160], is that they allow for detailed analysis of large computer networks. Thus, they overcome issues in terms of simplistic test, such as Al-Tawil and Al-Kaltham [74] highlighted, as well as the cost and time consuming aspect of using real devices. Arguably, from a network management perspective, the key objective is to assess the benefit of a new topology, or services from a network user point-of-view. Nevertheless, employing the correct simulator, that fits the purpose at hand, requires a good working knowledge of network simulators and their possible shortcomings [156, 157].

Network simulators are best suited for the study and modelling of end-to-end communications, and not network security [156]. Indeed, these software applications typically rely on mathematical models of key network protocols to operate, such as for TCP. For example, a key focus of network simulators is wireless communications as there are multiple methods of accessing the radio wave medium, and this aspect matters in the deployment of wireless networks [157]. Along with this, these access methods have mathematical representations that fit well within network simulators mode of operation [157]. Thus, researchers can simulate multiple radio carriers models, for example, so they can gain an understanding of the possible outcomes of a real-life implementation.

Lacage and Henderson [157] propose an improved network simulator in order to address key shortcomings found in OPNET, and *ns-2*. The closed source nature of the OPNET modeller [160] does not allow Lacage and Henderson [157] to go into details in terms of the specific weaknesses of OPNET. Flores Lucio et al. [156] argue that OPNET is best suited for network operators as opposed to researchers. Lacage and Henderson [157] argue that errors, such as in terms of programming or faulty models in one area can cause inaccurate results, and they point out that these errors are difficult to isolate from the simulation specifications. Indeed, Lacage and Henderson [157] suggest that this is due to the fact that the elements which compose *ns-2* are interdependent and that basic object orientation programming paradigms have been ignored.

One of the motivation behind the proposal of Lacage and Henderson [157] for **Y**et **A**nother **N**etwork **S**imulator (*yans*) is that traffic models in *ns-2*, for example, are limited. Indeed, these solely rely on mathematical equations. In other words,

it is not possible to perform an evaluation of traffic captured from a live environment. Arguably, modelled traffic may not fully represent the use that is made of the computer network and the behaviour of its users. Along with this, modelled traffic might be bias towards a particular outcome in an hypothesis, and this compromises the scientific quality of experiments conducted in such a manner [24]. Consequently, this increases the burden on administrators who then have to interpret the results for their own organisational context.

Arguably, Flores Lucio et al. [156] best illustrate the key weakness in network simulators: the lack of accurate and bespoke intermediate device models. Indeed, they try to demonstrate how to match live experiment results with the appropriate configuration settings in both OPNET and *ns-2* [156]. Flores Lucio et al. [156] show that there are numerous options that need to be altered in order to match the live experiment outcome. In particular, both network simulators assumed that the network router would treat the traffic between networks evenly whereas the real device did not. Hence when assessing the impact of a firewall network, simulators will provide the results of **a generic network firewall** and not necessarily the impact of the firewall device the organisation has chosen.

Obstacles with Live Evaluations

Computer networks are probably as unique as the organisation that owns them. Bajcsy et al. [15] argues that it is the absence of relevance to current network infrastructure, the lack of scientific rigour, and the non-repeatability of experiments, which constitute an obstacle to the adoption of new defence techniques by most organisations. Along with this, some evaluation can be destructive [15]. Nevertheless, Zhang et al. [32] argue that models for the performance of intermediate devices should be created from data collected in a live environment.

One of the key issues with live evaluation is that these are often simplistic. Al-Tawil and Al-Kaltham [74], for example, aim to compare the effects of network firewall's position within the network, and focus on evaluation scenarios that are easy to deploy. Consequently, the results are limited in scope as these do not provide information regarding load sensitivity, for instance [50]. The same shortcomings apply to the experiments of Lyu and Lau [30].

Allman [158] outlines a method to conduct performance evaluation in a production environment, with an aim to compare performance data obtained *from behind* the organisation Internet gateway against data collected *in front* of this gateway [158]. The benefits of this methodology include the pin-pointing of faults more rapidly as well as determining the performance of route caching the intermediate devices use. Arguably, Allman [158] evaluates all the devices that are situated between the two evaluation measurements locations, and, in this instance, these include load-balancing and proxy devices. Thus, this does not allow for the modelling of one component on its own, though.

Self-contained evaluation environments [145, 143] can address this limitation as these often combine the ability to create contexts for the evaluations, such as with network traffic load generators, and detailed evaluation by combining software tools [77] and automating the process [145]. Hence, Roedig and Schmitt [145], for example, have full control over the parameters of an evaluation, such as the number of concurrent sessions, and are free from concerns over privacy that Moitra and Konda [161] raised. Arguably, one of the drawbacks of this approach is that it is not always transferable to the production network. Roedig and Schmitt [145] use a modified version of a software firewall in order to incorporate the necessary performance monitoring facilities, and hence this limits the number of devices organisations can evaluate in this way. Likewise, Allman [158] and Hoffman and Yoo [77] modify networking tools to meet some specific criteria and thus these bespoke versions are tied to a particular evaluation environment.

Thus based on the observations, the network firewall dynamic performance proposed in this chapter uses as inputs: the configuration that a network firewall supports, as well as support for dynamic performance evaluation. Thus, the outcomes and results can serve to build a more complete model for firewalls. Indeed, the models are produced from data gathered with out-of-line experiments [78, 143, 32], a key aspect of the evaluation environment (DEENS) in which these are created with, is that it is free from vendor ties, such as seen in [139, 138, 141] for example. Along with this, DEENS allows for a strong emphasis on a scientific approach that, in turn, provides rigour in the decision making process for network security, as Cremonini and Martini [45] or Peisert and Bishop [24] suggest, for example.

4.3 Dynamic Evaluation Environment for Network Security

The main aim of the DEENS is to accurately measure the effect of varying firewall settings, such as varying rule-set sizes, on network performance. This, in turn, allows developing knowledge of the device from a component point-of-view [143], and provide data that allow for the analysis from multiple view-points. Indeed, individual security devices are seldom evaluated in isolation, whereas in most other engineering or scientific disciplines, a system is typically split into subcomponents, each of which are evaluated independently from the others, and verified from a component point-of-view. Thus, a model of each component is created, and evaluated to be able to build a complete understanding of the whole system, and how the components fit together.

4.3.1 Background

It is often recommended to evaluate systems before full-scale deployment, as it can be costly to roll-back a configuration in computer networks [16, 14, 162]. Generic solutions are often not suitable because computer networks are probably as unique as

the organisation that owns and operates them (Section 2.2). Ensuring the continuity of the computer network system services is often a complex task as methodologies and technologies have to be adapted to organisation requirements. Many experts, such as Bajcsy et al. [15] or Peisert and Bishop [24], among others, argue that it is the absence of the relevance to current network infrastructure, the lack of scientific rigour and the non-repeatability of experiments that constitute an obstacle to the adoption of improved defence techniques in most organisations. When it comes to performance evaluations, the major issue is that these cannot often be carried out in a production network as they can be destructive, and likely to disturb user activity on the network [143, 15].

Based on this, DEENS is an isolated computer network, and thus includes facilities to:

- **Configure network firewalls** according to specifications.
- **Generate network traffic**.
- **Collect** performance metrics.

Each of these core functions exists in software form, and therefore allow the procedure of the evaluations to be carried out in an automated manner [33, 143]. Hence, DEENS allows creation of multiple **unique** instances of the proposed model (Section 4.4) as well as the detailed investigations of network firewall capabilities. The combination of multiple instances of the model allows for the identification of the **D**evice **U**nder **T**est (DUT)'s failure conditions, and this is particularly relevant in the mitigation of DoS and DDoS attacks [113, 99, 4]. Indeed, there are multiple types of DoS and DDoS attacks, and some of these could simply exploit software vulnerabilities on a web-server [113] to take corporate services off-line. Another solution would consist in overloading the target network, and Paxson [18] demonstrates that this could be destructive. However, if the network firewalls were to fail because of a surge in network traffic, as opposed to a software exploit, the remedies are very different.

More importantly, perhaps, computer-related laws are being reviewed as current legislation does not often include networks, and network-based exploits [97]. Consequently, the increase in protection within the law against crimes such as DoS, and DDoS, could require organisations to demonstrate their network infrastructure fitness, prior to incidents [99]. This thesis argues that DEENS produces the data that can support this.

4.3.2 Internal Structure

In order to satisfy the goal of creating a component based model for network firewalls, DEENS is built around the DUT itself. As Figure 4.1 shows, the background traffic generator is connected to either end of the DUT, and each interface of the DUT is connected to a sub-network. The **source** sub-network includes the source measurement node whose function consists of initiating individual measurements.

The measurement sink node is located on the other end of the DUT in the **sink** sub-network.

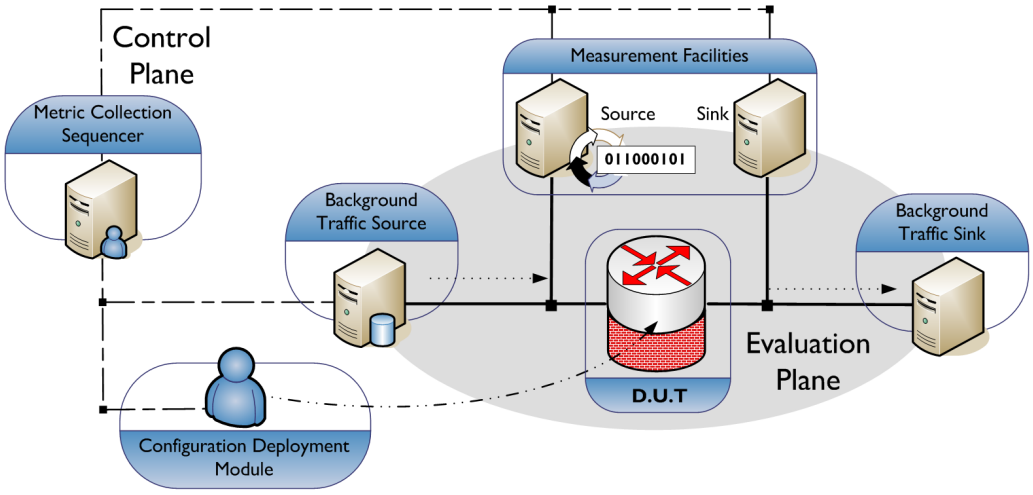

Figure 4.1 – Dynamic Evaluation Environment Network Layout [143].

All the network nodes located in the *evaluation plane* [78] are in fact dual-homed (Figure 4.2). In other words, they are both connected to the evaluation plane and *control plane* [143]. This enables each host with a software application that awaits instructions in terms of what measurements to perform, or traffic to generate. Then, the host carries out the assigned tasks and returns the measurement readings. The generated traffic is confined to the evaluation plane as the transfer of network traffic between the network interfaces of the hosts is not permitted.

The orchestration of events [33] that composes the evaluation procedure is achieved with the **Metric Collection Sequencer**. It is also responsible for storing and archiving the measurement readings (Figure 4.2). This software application distributes the configuration onto the DUT and the network switches, via the **Configuration Deployment Module**. The need for configurations of the various equipment to be changed regularly requires the evaluation environment to include facilities that permit remote interaction. As demonstrated by Saliou et al. [78], this can be achieved directly within the evaluation plane with terminal services or remoting protocols, such as Telnet or **H**yper**T**ext **T**ransfer **P**rotocol (HTTP). However, this requires the DUT to hold a minimal configuration to begin with. Instead, DEENS relies on Console Servers (Figure 4.2) for this functionality.

The network gateway completes the isolation of DEENS from other networks, or sub-networks. This node enforces strict access control to DEENS, as recommended in Paxson et al. [163] for a similar project, and also permits the control plane to use a different addressing scheme (Appendix D) from both the evaluation plane and the main corporate network. Hence, the overall environment can be extended easily. For instance, evaluating two network firewalls in parallel only requires an additional set of measurement nodes and traffic generation nodes.

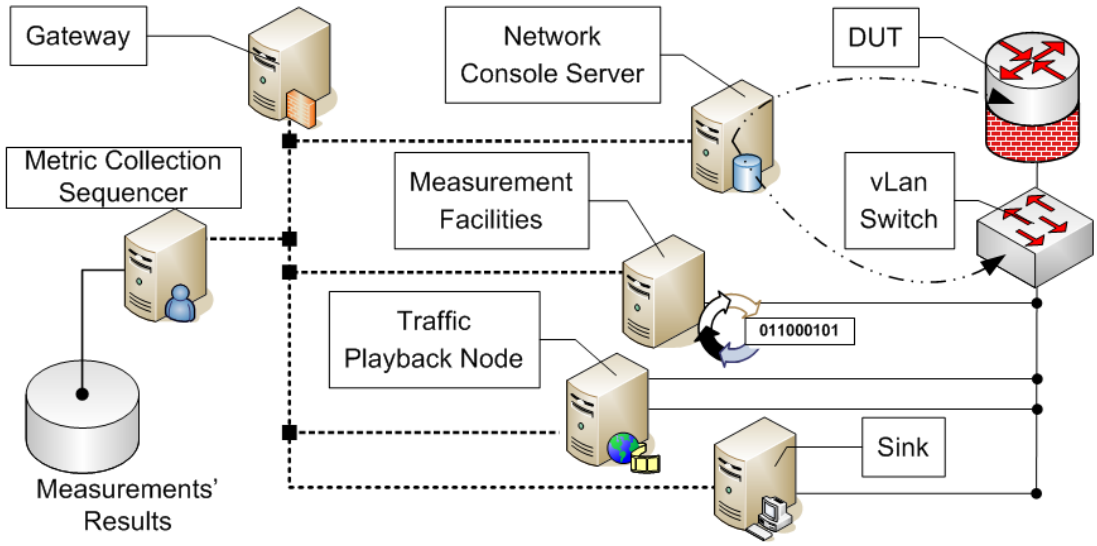

Figure 4.2 – Evaluation Environment Logical Layout.

Component-based System

The design for DEENS favours component reuse over bespoke systems, and this allows for the necessary elements to be changed as required. Indeed, bespoke systems typically merge the network conditions control mechanisms with the metric collection capabilities, as illustrated in Hoffman and Yoo [77] for the logical attributes of firewalls, and Roedig and Schmitt [145] for the performance of multi-media firewalls. Arguably, these approaches ease the overall evaluation procedure, however, they do not address Bajcsy et al. [15]'s point about the fact that researchers do not address, or present solutions, to problems in a manner that relate to organisational needs. Indeed, such integrated evaluation systems [77, 145] do not present an immediate benefit unless organisations deploy, use, or evaluate them first. In addition, organisations would no longer be in control of the procedures they rely on to administer, or monitor, their systems. At best, it would require organisations to build parallels between the specified tool chains [77, 145], and the tools already in-place within the organisation. Hence, this evaluation environment reuses well-established components such as HPing [164], and Netperf [165], to collect network metrics, and integrates SNMP modules [166], for instance. This component-based approach also allows for the adaptation to the collection metric requirement of an organisation, in particular. Tools can then be added or removed, as required.

Automated Environment

Scientific rigour requires experiments to be repeatable [24], and for the number of samples for each metric to be large enough to provide reasonable accuracy. Hickman et al. [144] specifies that the measurements should last at least 30 seconds in order to fully capture the effects of the evaluation conditions. With DEENS measurement time

is typically 60 seconds (T_{Sample}) as this ensures that all the metrics are gathered within a common time frame and this also meets Hickman et al. [144]'s recommendations. Hence, the time requirement for dynamic evaluation makes the exercise unattractive to be performed manually, and hence DEENS automates the evaluation process. Indeed, Equation 4.1 shows the parameters that influence the time requirement and Table 4.1 gives an estimate in minutes of the time it takes to complete performance evaluations on fast and slow network settings.

$$T_{Evaluation} = \left(Nb_{Samples} \times Nb_{SnapShots} \times Nb_{Metrics} \times T_{Sample}\right) + T_{Reconfiguration} \quad (4.1)$$

Where:

- $Nb_{Samples}$: This is the number of times measurements are carried out for the chosen network conditions.
- $Nb_{SnapShots}$: This specifies the number of times metrics are collected, and it typically depends on the network speed settings. Based on the findings of Saliou et al. [143], fast network settings use six snapshots, in other words from no traffic, 0%, to 50% of maximum network throughput using 10% increment, and slow network settings use eight snapshots, in other words from no traffic to 70% of maximum network throughput using 10% increments.
- $Nb_{Metrics}$: This relates to the number of metrics DEENS supports.
- T_{Sample}: This is the measurement time for each sample.
- $T_{Reconfiguration}$: The time it takes to reconfigure the DUT. Indeed, Saliou et al. [78] show that the reconfiguration time depends on both the protocol employed for this task, and the size of the configuration to deploy.

Table 4.1 – Time estimations of performance evaluations

	Fast network settings	Slow network settings
$Nb_{Samples}$	10	10
$Nb_{SnapShots}$	6	8
$Nb_{Metrics}$	4	4
T_{Sample}	1	1
Shortest times (minutes)	240	320

4.4 Proposed Model

This section presents a novel model for network firewall dynamic performance in terms of the parameters that are considered as inputs, and the data it provides. Such information can, in turn, be used to assess the feasibility and suitability, in terms of performance, of security policies for these devices. Along with this, this data plays

an essential role in determining if a device can cope with dynamic threats that rely on resource exhaustion, such as with DDoS attacks [4], or side effects of malware outbreaks (Section 3.5.1). The model possesses three categories (Figure 4.3), and these are the: *chosen device*; *firewall rule-set* characteristics (both of which relate to network firewall configuration parameters); and *network conditions*. The model associates the input parameters with impact on the network firewall component in terms of device-oriented metrics such as on the CPU usage or available memory [142, 141], as these devices can have limited tolerance to network traffic [78], and network-oriented metrics, such as available throughput and latency [78, 143, 30, 135] (Figure 4.3). In addition, performance data can be challenging to obtain [145, 78], and thus this model also incorporates the concept of an error-rate for each metric to reflect this aspect of the evaluation.

A key aspect of DEENS is that it takes into account the environment in which the device is to be deployed, hence it is linked to the type of usage the organisation makes of its computer network. Evidently, there is an almost unlimited number of firewall devices and configuration combinations [78] (Figure 4.3), hence each evaluation is performed for defined values of the parameters and is an **instance of the model** that this section describes.

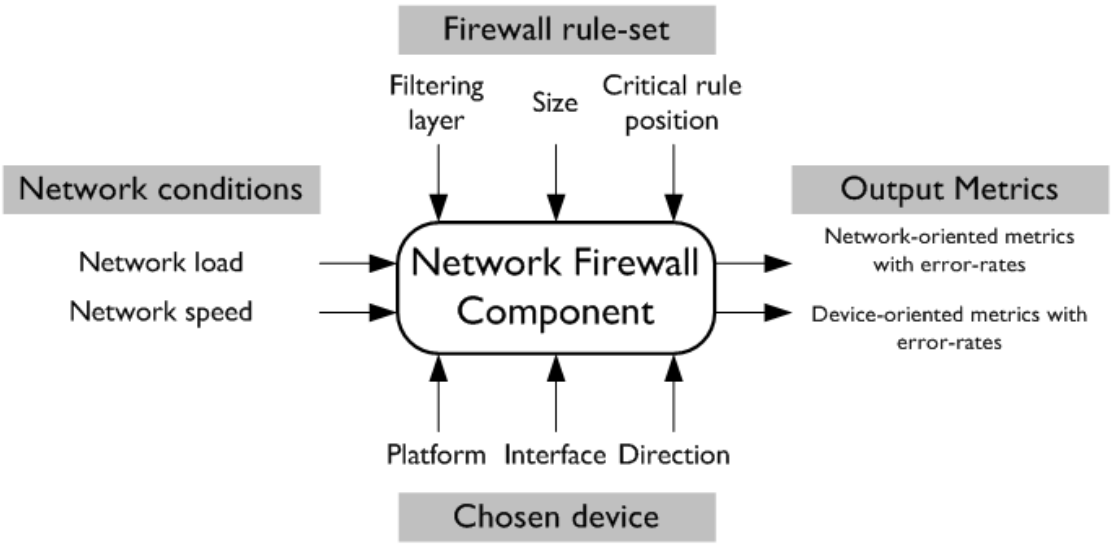

Figure 4.3 – Network Firewall Dynamic Performance Model

4.4.1 Input Parameters

Figure 4.3 shows the inputs of the evaluation environment and this section details its parameters. It also discusses the inputs related to the chosen device category first, then the firewall rule-set characteristics and finally the network conditions.

Platform

The platform parameter defines the type of equipment for the evaluations. In this thesis, the two platforms employed are the Cisco IOS hardware, and Linux NetFilter, whose technical specifications are listed in Appendix C. These platforms have been chosen as they are often used as part of research studies [77, 139], and these studies often employ these platforms to demonstrate a particular point, such as the more rigorous evaluation of filtering of Linux Netfilter firewalls in [77]. Overall, though, there is seldom an emphasis on organisation requirements, thus the evaluations on which the evaluations are built upon are akin to black-box testing. In other words, the capabilities of the DUT are not known in advance, and thus evaluations can identify these. This, in turn, allows the selection the firewall device according to organisation requirements. Some researchers, such as Airamo and Virtanen [135], or Cremonini and Martini [45] for example, highlight the danger of relying solely on manufacturer (Section 3.6.4), and thus void security policy implementations, such as of Eloff and Eloff [65] that rely heavily on this parameter.

Filtering Direction

This input parameter represents the combination of two firewall rule-set deployment criteria: the interface on which the rule-set is deployed; and the traffic direction to which it applies to. Network firewalls typically possess multiple network interfaces that support firewalling functions, and these can either be physical or logical. Logical network interfaces allow the network firewall to filter the traffic from sub-networks, or vLANs, than the firewall has physical interfaces. This, in turn, give flexibility to the network design, and also permits network growth [37, 167]. The devices employed in this thesis have two physical network interfaces, and support the use of vLAN technology.

Figure 4.4 shows that a firewall rule-set can be applied in two distinct locations within the device (Section 2.5). For the first location, a network packet is compared against the statements contained within the rule-set as soon as it arrives at the interface of the device, in other words the rule-set is applied on the **incoming direction**. Alternatively, the packet's content can be examined after the routing decisions are made and prior to its release on the interface towards its destination, in other words the rule-set is applied on the **outgoing direction**.

As mentioned in Section 3.6.4, this parameter has a different meaning depending on the view-point used during the configuration. In these evaluations, the configuration view-point is from the DUT itself. Hence, for the Cisco DUT, the incoming filtering typically refers to a rule-set applied on the first interface, and that is used to filter traffic as soon as it arrives on the device. Conversely, the outgoing filtering direction refers to a rule-set applied on the second interface and used to filter traffic when it leaves the DUT towards its destination. Table 4.2 is the lookup table that

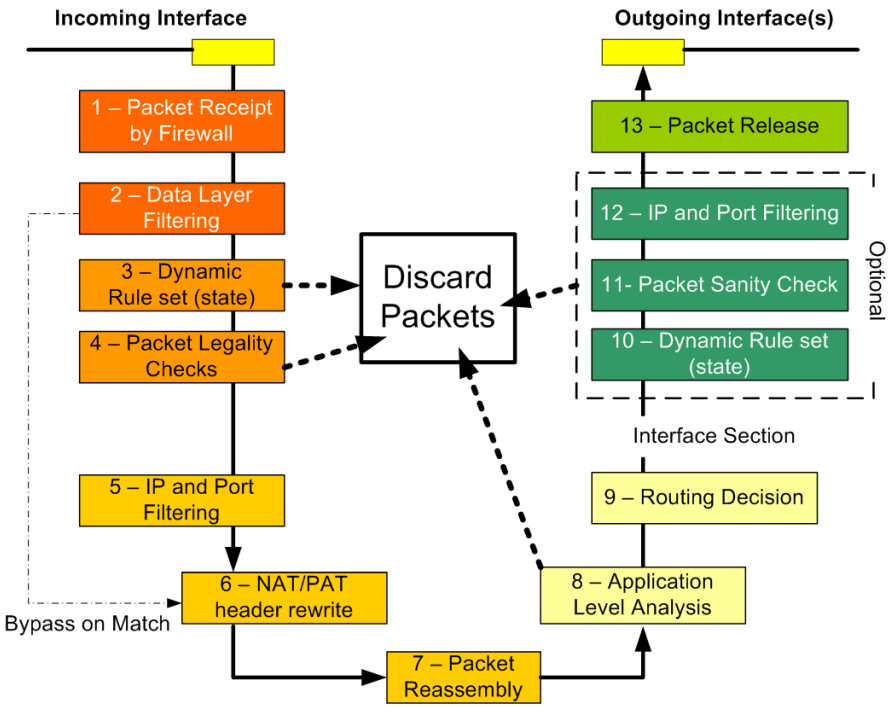

Figure 4.4 – Firewall's inner operations as described by Kamara et al. [31].

associates the technical aspect of the the Cisco DUT to the possible values supported by the model and the resulting filtering direction. Listing 4.1 shows a script snippet that deploys one ACL (`access-list 1`) in the incoming direction for a Cisco DUT.

Listing 4.1 – Applying an ACL in the incoming direction

```
! content of the rule-set
access-list 1 deny ip 177.129.1.56
access-list 1 deny ip 118.226.112.42
access-list 1 deny ip 124.84.62.221
access-list 1 permit any
! accessing interface sub-menu
interface fastethernet 0/0.10
ip access-group 1 in
no shutdown
! the end
```

Table 4.2 – Filtering directions for the Cisco DUT

Technical terms		Model instance value	Effective filtering
Interface name	Applied direction		
FastEthernet 0/0.10	in	CiscoInt1	incoming
FastEthernet 0/1	out	CiscoInt2	outgoing

For the DUT that uses the Netfilter firewall engine [133], on the other hand, the

directional filtering option is being phased out. In turn, the firewalling engine can define what statements to compare the network packet with, depending on the network interface it originates from. Hence, rule-sets that are applied in the incoming direction are linked to `LinuxInterface1`, and thus rule-sets that are applied in outgoing directions are linked to `LinuxInterface2` (Table 4.3, Listing 4.2). Typically, when the rule-set relies on the Netfilter engine to determine the filtering direction, the interface is defined simply as `LinuxInterface`. Arguably, this implementation method reduces the firewall administrative burden as a Linux Netfilter DUT no longer has one rule-set per interface, and per direction, like a Cisco DUT can. Along with this, it makes systems such as of Caldwell et al. [14] easier to deploy, as there is no longer the need to associate firewall rule-sets with a network interface, or segment (Section 3.6.4).

Listing 4.2 – Linux NetFilter snippet implementing outgoing filtering

```
iptables -A FORWARD --src 177.129.1.56 --out-interface eth1 -j DROP
iptables -A FORWARD --src 118.226.112.42 --out-interface eth1  -j DROP
iptables -A FORWARD --src 124.84.62.221 --out-interface eth1  -j DROP
iptables -A FORWARD --src 189.119.171.195 --out-interface eth1  -j DROP
...
iptables -A FORWARD --src 10.0.*.* -j ACCEPT
iptables -P FORWARD DROP
```

Table 4.3 – Filtering directions for the Linux Netfilter DUT

Technical terms	Model instance value	Effective filtering
eth0:10	LinuxInterface1	incoming
eth1	LinuxInterface2	outgoing

Rule-set size

There is a consensus among researchers that strong security policies need to be detailed [65], and, since network firewalls are rule-based systems (Section 2.5), security policy implementation typically creates large rule-sets [14, 107]. In addition, security policies are not unchanging, and thus likely to change over time. Hence, this parameter relates to the total number of firewall statements that compose the rule-set deployed on the DUT. This allows the assessing of the effect of security policies on network firewalls performance [78].

Critical Rule position

Unlike logical evaluations, performance evaluations only exercise, or trigger, a restricted number of statements within the rule-set. Since network firewalls typically process filtering statements in a sequential manner [75], this attribute allows the measurement of the effect of the position of these statements has on security performance.

As shown in Figure 4.5, network firewalls are often at the crossroad of multiple networks, or sub-networks. Hence, it is unlikely that the rules concerning the filtering of the traffic between the Engineering and Finance departments are in the same position as the rules concerning the filtering of the traffic between the Engineering department and the Periphery (Figure 4.5). The evaluations, that this thesis presents, focus mainly on a critical rule being placed at the **top**, in the **middle**, or at the end, of the rule-sets.

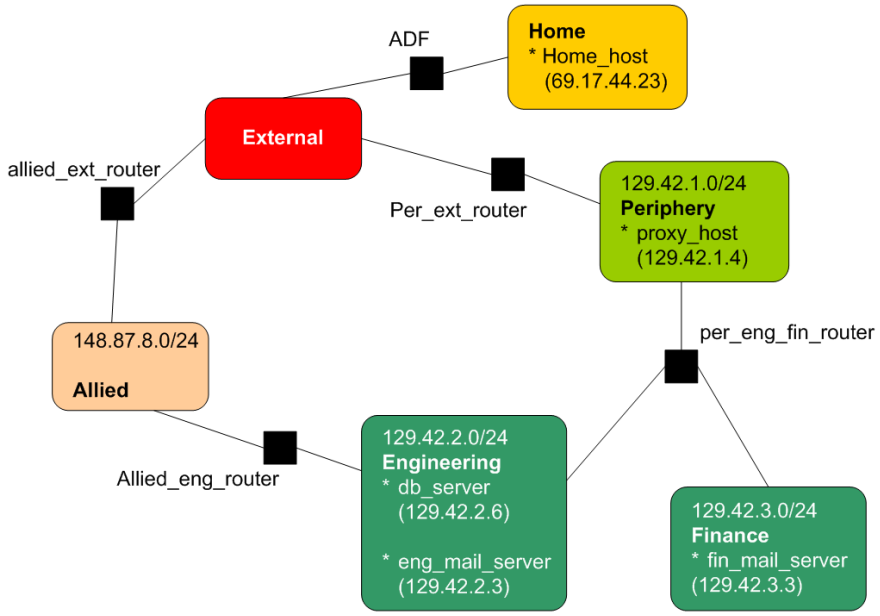

Figure 4.5 – Network firewalls, crossroad of multiple networks [52].

Filtering Layer

Network packets carry information regarding the specifics of communications, such as the origin of a particular stream of network traffic. A network firewall is capable of filtering communications on the basis of one or more pieces of information included in network packets (Section 2.5). Hence, a filtering statement can be simple, such as that only one item is scrutinised, or complex, when several items are taken into consideration. This aspect typically influences the OSI layer at which a firewall statement operates. A simple statement will typically only require information from the network layer (Layer 3), whereas a more complex statement will require information from both the network and transport layers (Layer 3 and Layer 4). Most studies focus on the fitness and logical attributes of firewalling engines, and to that end, specific network packets are employed, such as in terms of size and content [142, 138]. There is, though, little work on the outcome that the firewall rule-set statement must achieve and how this aspect impacts on the device, itself, and on network performance.

Network Speed

This parameter relates to the maximum throughput the DUT can support. For the purpose of this thesis, a maximum network throughput of **10 Mbps** is referred as **slow network** throughput, whereas a maximum network throughput of **100 Mbps** is referred as **fast network** throughput.

Zhang et al. [32] highlight that demand for greater available network bandwidth is growing, and this, in turn, increases the likelihood of network congestion at network firewalls [50]. Along with this, a network firewall device typically supports multiple speed settings (Appendix C), and this parameter plays a key role in the resilience of networks against DDoS [113, 13, 78, 143].

4.4.2 Environmental Conditions

Often, network firewalls are evaluated with the traffic that only exercises the logical attributes of the device, and thus evaluations do not often contain any other background traffic [30, 136, 123, 124, 137]. This is unfortunate, as once the network firewall is deployed in the production network, it must cope with both background, and filtered traffic [74]. Hence, **Environmental Conditions** relate to the typical use of the networked system. This includes the applications that are used over the network, their frequency of use, the network resources they consume, such as in terms of network bandwidth, and so on. Arguably, this aspect represents the operations that are carried out to fulfil the organisation's objectives and are dependant on the organisation sector of activity. The evaluations presented in this thesis uses network traces that Lippmann et al. [105] discuss. This enhances the repeatability, as well as the scientific rigour of the presented evaluations, and thus provides the organisational context of the measurements obtained with such background traffic. Other researchers, such as Antonatos et al. [146], employ similar network traces for their evaluations, as did Hamed and Al-Shaer [142], although their traces are not publicly available. Also, due to the nature of the evaluations, the results are not universal, as they are bounded to the traces employed, and hence specific to the organisation itself.

Repeatability can be ensured with the use of modelled network traffic, however the outcome of any evaluation must then be examined with care [24]. Indeed, this approach typically causes issues when the resulting network device models, for instance, are applied to production, or live, networks. This is best illustrated in the work of Roedig and Schmitt [145], as they conclude that the enhanced capabilities of their multi-media firewall cannot be guaranteed if used in production networks, where the traffic is unlikely to be solely composed of video or sound streams, for example. Arguably, this shortcoming illustrates Peisert and Bishop [24] argument regarding miss-leading outcomes because of biased input data-sets. Other examples include Piyachon and Luo [134]'s system which performs best when network user

activities fit a specific probabilistic distribution.

The dynamic aspect of the evaluations relates to the fact that the metrics are measured under defined and controlled traffic conditions. Henceforth, the **network load** refers to the network bandwidth usage across the DUT during each measurement, and this network loading is expressed in percentage of the maximum network throughput [78, 143]. This approach is fundamentally different from **static methodologies** which focus on the measurement of the difference in terms of performance between *pre* and *post* configurations [30, 76].

4.4.3 Output Metrics

The metrics DEENS provides are divided in two parts: the DUT monitoring aspect, and key network characteristics. Along with this, DEENS also includes an indication of how credible measurements are from a dynamic point-of-view with the inclusion of **Error-rates** for each metric (Figure 4.3).

Device Performance

CPU usage and available memory space are finite and precious commodities on network firewalls. Indeed, network firewalls have other tasks to perform in addition to filtering traffic, and these operations utilise these two resources [139]. These additional tasks include updating routing tables, and making routing decisions (Figure 4.4), as well as participating to management tasks, such as answering SNMP queries [86].

The effect of security policies on CPU usage and available memory is measured during the evaluations, in order to determine the effect of security policies have on the device. Such data is essential to the proper planning of security deployment, as well as for possible changes in network design, such as in terms of equipment upgrade, and load balancing.

Network Performance

Network firewalls, unlike IDS, are in-line devices [85] thus any delay they incur on network traffic could impact other nodes of the network. This aspect is measured with two distinct metrics. The first metric is communication latency which is derived from the **R**ound **T**rip **T**ime (RTT), and is the length of time it takes a packet to reach a remote node across the DUT and back. This metric is relevant to real-time applications, such as VoIP [108].

TCP stream measurements [165] relate to the rate at which the transmission of data between a network node and a remote node across the DUT occurs. Many essential applications rely on the TCP protocol, such as web-servers with HTTP, or **F**ile **T**ransfer **P**rotocol (FTP). When a TCP connection detects network congestions, it typically reduces its transfer rate, in other words it reduces its network bandwidth

consumption [168]. Such an event thus delay the completion of the task at hand, and it is also likely that user will notice this effect [32]. Therefore, the TCP stream [165] measurements allows an assessment of the impact of security policies on this metric, as network firewalls typically scrutinise each network packet independently.

Error-rates

This new concept relates to Saliou et al. [78]'s observation that depending on the parameters of evaluations, some measurements might not be possible or fail, thus indicating that the device is under stress. This, however, does not apply to all the metrics simultaneously [78]. With respect to the fact that the evaluations are carried out in an environment similar to production computer networks, such data is essential to the assessment of the network devices monitoring quality, as well as for the decision making process. Lack of credibility can lead to the wrong conclusion and decisions. Hence, this element defines the **window of credibility**, and thus is associated to each of the measured metrics. It is computed as the ratio of failed measurements over the number of samples the evaluation pattern specified, and results in a value between zero and unity. In other words, this element allows to identify whether the DUT reaches fail-over conditions progressively, or suddenly.

4.5 Evaluation Procedure

This section presents the manner in which an evaluation is performed. It details the creation of one instance of the network firewall dynamic performance model as well as the technical details of each measurement.

4.5.1 Definition of the Baseline

The experiments can only be completed if the DUT is able to route packets between the source and sink sub-networks [78, 143] (Figure 4.1, Section 4.3.2, and Section 4.5.4), and thus, for each DUT (Section 4.4.1), the configuration which satisfy this criteria represents the baseline. In both cases, the devices have the same number of physical interfaces, implement the same number of virtual interfaces, and rely on the same routing protocol (Appendix C). Appendix B contains all the baseline readings of the DUTs.

4.5.2 Orchestration details

The Metric Collection Sequencer (Section 4.3.2) can perform multiple evaluations in a sequence, where a number of evaluations, along with their respective specifications, are read and analysed when the application begins. The specifications are the values of each of the parameters described in Section 4.4, along with a unique identifier for the evaluation, henceforth referred as **model instance identifier** (ID). The value

of each parameter is analysed in order to generate and to apply, via the Configuration Deployment Module (Figure 4.1): the relevant settings on the DUT; and vLAN switches (Figure 4.2). The Metric Collection Sequencer does not generate the firewall rule-sets though, as these are read-in from pre-formatted files.

The first task of the Metric Collection Sequencer is the deployment of the necessary configuration onto the DUT and vLAN switches according to the evaluation specifications. It then moves onto making contact with the traffic generator node, and issues it with the characteristic of the traffic to generate (Section 4.4.2). This sequence is skipped if it is the beginning of the evaluation, as the first measurements are always carried out without any background traffic.

Once the traffic conditions are set, the Metric Collection Sequencer pauses for 30 seconds, which is necessary as modifying device configuration typically incurs some computation intensive activities on the DUT [78]. The measurement nodes dedicated to the device metrics (Section 4.4.3) gathering are then contacted. The CPU usage is typically measured first and then the memory usage. The measurement nodes then attempt to gather a range of samples before returning individual readings to the Metric Collection Sequencer, which archives these into the folder assigned to the current evaluation.

Subsequently, the Metric Collection Sequencer contacts the measurement dedicated to network performance metrics (Section 4.4.3), where the same pattern is employed to gather the readings for the network latency and available network bandwidth. Upon the completion of this task, the Traffic Generator node is instructed to stop. The Metric Collection Sequencer then waits for confirmation from the Traffic Generator before proceeding to the next set of measurements.

4.5.3 Design of the Rule-sets

In order to promote a scientific approach in these evaluations, it is essential to make the evaluation methodology repeatable, and mitigate, as much as possible, the effects of enhancements that manufacturers might enable their devices with. Possible enhancements include, for instance, algorithms that are able to summarise [75], or optimise, rule-sets once loaded into memory, otherwise possible comparisons with other equipment, or devices from other manufacturers, might not be possible. Consequently, the DUT has to evaluate each packet against every single statement of the rule-set. To that end, the evaluation-plane network (Section 4.3.2, Figure 4.1) employs unroutable IP addresses (Appendix D), and this is combined with rule-set statements which test IP addresses that will never appear in the network traffic, such as Listing 4.3 shows. Hence, this permits the measurement of the full impact that a rule-set has on the device's performance.

Listing 4.3 – Snippet of a Cisco ACL using exclusively routable IP addresses

```
access-list 65 deny host 177.129.1.56
access-list 65 deny host 118.226.112.42
access-list 65 deny host 124.84.62.221
access-list 65 deny host 189.119.171.195
access-list 65 deny host 33.83.183.49
...
access-list 65 permit any
```

4.5.4 Metric Collection

This section provides technical details of the software components employed for the individual metric measurement.

CPU Usage

The collection of this metric relies on SNMP, where the Metric Collection Sequencer indicates which SNMP object to query according to the type of network firewall being evaluated. Arguably, this metric is simpler to gather on Cisco equipment [169] compared to Linux devices, for instance, as Cisco devices support the SNMP object `cpmCPUTotal1minRev` [170] that holds the busy percentage of the last 1 minute period.

For Linux devices, this task requires each reading of the `ssCPURawIdle` SNMP object to be time-stamped. Indeed, `ssCPURawIdle` returns the number of CPU ticks, typically 1 millisecond units, that are unused. Hence, the CPU usage, in percentage terms, of a Linux network firewall, CPU_{Linux}, is computed using Equation 4.2 where n is the sample number.

$$CPU_{Linux} = 100 - \left[\frac{Tick_{n+1} - Tick_n}{(Time_{n+1} - Time_n) \times 100} \right] \tag{4.2}$$

Memory Usage

This metric too relies on SNMP, and, once again, the type of the DUT needs to be known. For the Cisco device, the SNMP object `ciscoMemoryPoolFree.1` [170] indicates the number of bytes which are not being used at the moment where the information is requested. For Linux devices, this data is contained in the `memAvailReal` object [166]. In both cases, the readings are collected every minute as these SNMP objects do not include the concept of time.

SNMP Technical errata

Due to the fact that it is challenging to collect SNMP readings when the network load is high, particularly for fast network throughput, the SNMP measurements are

carried out through a dedicated interface of the DUT. This is achieved by dividing one physical network interface into two virtual interfaces with the use of vLANs. The sub-network resulting from this configuration is isolated, and therefore, cannot receive communication emanating from other sub-networks and vice-versa. Hence, the data collected with SNMP can constitute the basis for, or against, a network design where monitoring and maintenance activities have their dedicated network within the overall computer network system of an organisation.

Latency

Section 4.3.2 defined that measurements last for 60 seconds, however network latency measurements are completed within a much shorter time window, and often results are given in milliseconds. This evaluation environment uses HPing to measure the RTT (Section 4.4.3), and typically performs measurements at one second intervals. Thus, the experimental time length requirement is met by requesting 60 individual samples, and considering the average of these as one measurement. While HPing [164] possesses facilities to compute the minimum, maximum, and overall average, of the samples, the measurement node returns each individual measurement to the Metric Sequencer application, otherwise it is not possible to compute the error-rate for this metric (Section 4.4.3).

If HPing was used on its own, Listing 4.4 shows the command and required arguments that would perform the task described in this section. The IP address 10.0.20.11 is the address of the sink node (Figure 4.1).

Listing 4.4 – HPing command example

```
hping -c 60 10.0.20.11
```

Available Network Bandwidth

A key focus is to evaluate performance using network metrics, such as latency and available network bandwidth across the DUT, as opposed to evaluating the application layer protocol performance, such as HTTP or FTP [30]. Indeed, programs implementing these protocols typically introduce overheads, and different program versions can thus influence evaluation results [143]. This evaluation environment uses Netperf [165] to measure the bulk data transfer performance. By default, measurements carried out with Netperf only lasts for 10 seconds, however it is possible to specify the length of the measurement [165].

If Netperf was used on its own, Listing 4.5 shows the command and required arguments that would perform the task described in this section. The IP address 10.0.20.11 is the address of the sink node (Figure 4.1). The `-P 0` option prevents the display of measurement headers, which typically includes the size of the network socket, and so on, thus eases the parsing and capturing of the relevant data.

Listing 4.5 – NetPerf command example

```
netperf -H 10.0.20.11 -l 60 -P 0
```

4.6 Conclusion

This chapter presented a novel model for network firewall dynamic performance as well as the environment, DEENS, with which instance of this model can be created. These models can thus inform a formal model. The application of the network firewall dynamic performance model is not limited to planning purposes, it can also be employed as benchmarks during the monitoring of the live computer network system. Indeed, this expert knowledge can help distinguish between threats and normal device behaviour.

The scheduling of the various elements involved in: configuring the DUT; controlling network conditions; and collecting network metrics; is based on the approach described in [33]. A by-product of using software components to perform the evaluation is that they require the configurations of the DUT to be abstracted, as opposed to hard-coded. Thus, DEENS allows the gathering of performance data for multiple network firewall configurations, and thus this rigorous evaluation methodology allows the identification of the strengths and weaknesses of several network firewall devices. Hence, it enables DEENS to be free from vendor ties, and thus allows direct comparison of results between two platforms which have been studied in the same evaluation conditions (Section 4.4.2). Arguably, this type of information is valuable, especially in the organisational decision-making process, as it removes vendor bias which is often detrimental to organisation security [45].

Arguably, such data is an essential part a risk assessment, such as to determine the vulnerability to a particular intrusion and its possible repercussions. More importantly perhaps, the data provided in the network firewall dynamic performance enable organisations to comply with up-coming legislations [99]. Indeed, these new regulations now include network-based attacks, and seek to enhance organisational legal cover in this area; in exchange organisation must be able to prove the fitness of their computer network system *prior* to alleged incidents (Section 3.2.3).

The following chapter relies on DEENS to collect performance data on typical network firewall implementations. These constitute the basis for evaluation scenarios, such as the increasing the security requirements in: the form of larger rule-set sizes; effects of changes in terms of the underlying network equipment, prioritisation, and complexity of filtering.

Sections 4.3.2, and 4.5.2 highlighted that DEENS generates a large amount of raw performance data, and thus this poses the same challenges in terms of analysing the data from multiple view-points. Thus, the following chapter details the evaluation scenarios, formalises the inputs and outputs of the proposed model (Section 4.4)

as well as the scenarios themselves. This formalisation allows for the creation of a software tool capable of selecting relevant performance data according to scenario criteria and automatically process them as well as present them in a manner suitable for analysis.

Chapter 5

Evaluation Scenarios and Associated Analysis Tool

5.1 Introduction

This chapter describes the three main implementation scenarios, and possible variations. In order to facilitate analysis, the inputs and outputs of the dynamic firewall performance model (Section 4.4) are formalised, and this, in turn, allows for the formalisation of the scenarios themselves. These scenario formalisations define the criteria that model instances match when they are considered relevant to an analysis. Hence, this chapter describes an analysis software, **d**ynamic **F**irewall **E**valuation **R**esults **A**na**L**yser (d-FERAL), that implements the necessary functionalities both in terms of search abilities, and charting. In other words, d-FERAL allows for the analysis of performance dynamic data from multiple view-points.

5.2 Evaluation Scenarios

This section defines three evaluation scenarios that are investigated with DEENS (Section 4.3). These scenarios require multiple unique instances of the network firewall dynamic performance model (Section 4.4) to build a consensus on the possible repercussions of these deployments within a computer network system. Hence, for each scenario, the key criteria for the model instances are explained. Thus, the aims include finding performance metrics, and establishing the DUT failure conditions, such as in terms of network loads or the security policy the DUT enforces. Indeed, the failure conditions are valuable information when defending assets [171, 172], as these also represent knowledge that intruders are likely to exploit to their advantage [4, 171, 172]. Thus, the main evaluation scenarios include:

- The increase in security requirements deployed onto the DUT.
- Prioritisation of the traffic filtering, such as discussed in Section 4.4.1.
- The level of fine-grained control over the network traffic (Section 2.5).

The possible variations of these evaluations scenarios include: the various firewall platform on which the security policy is deployed; the manner in which such policy is enforced by the device, such as in terms of filtering direction (Section 4.4.1); and

the effect of maximum network throughput as devices typically support multiple network speed settings (Section 4.4.1). Along with this, the error-rates (Section 4.4.3) of each metric will be used to outline the credibility of readings, and thus inform on the quality of monitoring that administrators, or agents, surveying the system, can attain for of the varying network conditions (Section 4.4.2).

5.2.1 Increased security requirements

There is a consensus among experts that strong security policies need to be detailed [65], and since network firewalls are rule-based systems, security policies implementation typically create large rule-sets [14, 107]. Lyu and Lau [30] conclude that strong security, typically requiring larger rule-sets, does not impede network performance, such as for available network bandwidth. Their findings are reused in other work, such as of Kamara et al. [31], to justify stronger emphasis on security provided by firewalls, and dismiss possible performance issues. The issue here is that Lyu and Lau [30]'s experiments only include up to 50 items in the rule-set, which, at best, enables organisations to predict performance overhead for the complete Cisco recommended rule-set [76]. Along with this, Lyu and Lau [30]'s studies do not include any background network traffic during the experiments. Hence, there is no indication of the outcomes when the DUT is subjected to bandwidth demanding applications, such as for FTP transfer. Most importantly perhaps, Kamara et al. [31] suggest that the outcome of Lyu and Lau [30]'s experiments are universal, whereas rigour demands the experiments to be repeated for multiple platforms, especially if a formal model for network firewall dynamic performance is to be created. Consequently, for each of the two DUTs, the model instances suitable for this scenario are those where the only changing parameter is the rule-set size (Section 4.4.1). For example, Ciscoo033 and Ciscoo041 (Section 5.4.6, and Table 5.3) match this criteria as the only difference between the two is the number of items contained in the rule-sets.

5.2.2 Filtering Direction

Few organisations filter their outgoing traffic [106], despite the fact that these organisations may be legally liable if one of their network nodes is identified as attacking network nodes belonging to other network domains [10]. Hence, there is every justification for filtering traffic that leaves the organisation's computer network system towards external networks [106].

The required filtering can be achieved in a number of ways. Figure 5.1 shows the possible locations of a rule-set that will filter the network traffic transiting from one node within the internal network to a remote resource located in an external network. As discussed in Section 4.4.1, the traffic can be controlled through the incoming filtering interface or the outgoing one.

Thus, this issue is investigated with instances that use the same rule-set sizes

with different filtering directions. Arguably, this scenario can be part of the increased security implementation in order to establish if the DUT can extend its security capabilities in function of the filtering direction employed.

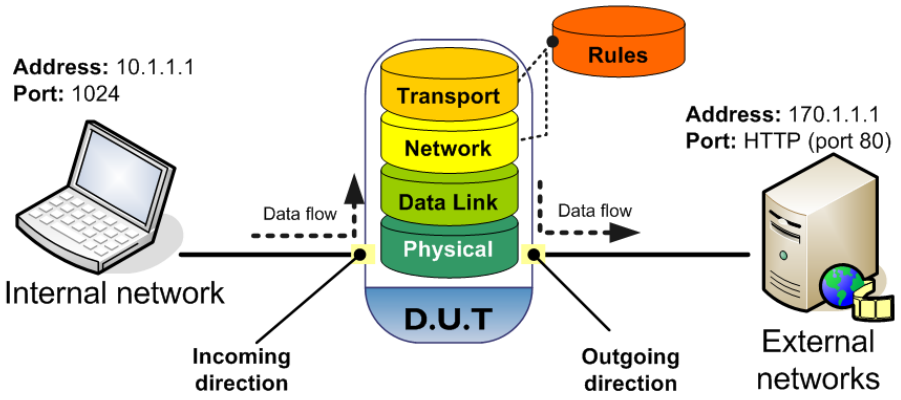

Figure 5.1 – Possible location of a firewall rule-set for the purpose of filtering traffic leaving an organisation internal network.

5.2.3 Critical Rule Positioning

Network firewalls typically join multiple network segments together [50], also firewall rule-sets are sequential in nature, thus rules regarding the protection of a particular segment often reside at different points in the rule-set. Thus, evaluating rule-sets in this manner [30, 78, 143] only provides results for the worse-case scenario, as the rules that are triggered are placed at the bottom of the rule-set. Arguably, this also applies to the scenarios described in Section 5.2.1, and Section 5.2.2.

Srinivasan and Feng [141] argues that the combination of both the traffic, the network firewall filters, and the rule-set it implements, matters to the overall impact on the network. Hamed and Al-Shaer [142] thus propose to re-order the rules in order to improve the match-ratio of network firewalls, and focus on rule-set efficiency, with respect to traffic content. This approach ensures that the number of statements the firewall has to examine is limited, thus this might reduce the impact on network performance. Thus, it is key to measure the benefit in terms of network performance of re-ordering rule-sets. Unfortunately, Hamed and Al-Shaer [142] solely focus on the performance of their network packet filtering algorithm. Arguably, this can help to determine if network firewall performance is solely a function of the rule-set size **or** if it is a function of the number of statements to examine. This can be of benefit to examine problems from various location within the computer networks. For example, if a user reports a performance bottleneck, the administrator who investigates it may not be located on the same sub-network as the user. Hence, this is achieved by analysing the model instances that employ a rule-set of a certain size and comparing performance readings against those for which the critical rules are located elsewhere

in the rule-set.

5.2.4 Layer of operation

Al-Shaer and Hamed [75] show that there are many types of firewall rules. These can be simple, such as denying a single IP host access to a network, or more complex, such as for verifying the state of a connection before reaching a filtering decision [139] (Section 2.5, and Section 3.6). The complexity of a rule is also related to the layer at which it operates. A security policy can thus be extensive, and the resulting firewall rule-set(s) can include a large number of simple and complex items. Network firewall rule-sets can be enhanced, and made smaller [75, 142], however, it might not be always possible to apply the methods Hamed and Al-Shaer [142] outline. This last aspect, however, demands that the effect of firewall rule complexity on network performance to be measured. Thus, it is necessary to to measure the difference in terms of dynamic performance between simple firewall statements that operate at Layer 3 (the network layer), against more complex rules that typically operate at Layer 4 (the transport layer) [173].

All the evaluations in this thesis use the same source rule-set using the Al-Shaer and Hamed [75]'s firewall statement model (Listing 5.1) which can then be converted into platform-specific syntax [143]. Thus, Listing 5.2 and Listing 5.3 illustrate simple and complex statements, respectively. Hence, this rule complexity is taken into account to quantify the overhead in terms of processing that each type of statement has. In addition, researchers have shown that rule-set sizes can be reduced by merging statements together [142]. Typically, this would involve taking several simple statements and merging them into fewer statements that are more complex. Rule complexity evaluations can thus help determine if it is a better approach to deploy many simple statements, as opposed to fewer more complex ones.

Arguably, Hamed and Al-Shaer [142]'s technique is advanced, and despite its advantages it is essential to establish if its use is always warranted, or if it is best used in specific situations. For example, Saliou et al. [78] show that a rule set of 65,000 items can have a detrimental effect on the resilience of network firewalls, especially at high network speeds, and thus recommend that such rule-set should be optimised. This optimisation should aim at reducing the total number of items while still matching the security policy. Wool [107] argues that simple firewall statements are far less error-prone, however Hamed and Al-Shaer [142]'s algorithm is likely to create statements with several parameters to examine, thus increasing overall complexity of the firewall rule-sets. Hence, this particular aspect has two consequences in terms of network firewall management: performance; and maintenance time by administrators. Thus, the instances that qualify for this investigation are those that employ a defined rule-set using the same deployment criteria (Section 4.4.1), however the main difference between the instances is that their rule-sets do not operate at the same OSI layer.

Listing 5.1 – Firewall rule model by Al-Shaer and Hamed [75]

```
<order>    <protocol> <s_ip> <s_port> <d_ip> <d_port>    <action>
```

Listing 5.2 – Simple statements example using Al-Shaer and Hamed [75]'s model

```
  1   ip   177.129.1.56    *    *   *   deny
  2   ip   118.226.112.42  *    *   *   deny
  3   ip   124.84.62.221   *    *   *   deny
  4   ip   189.119.171.195 *    *   *   deny
...
  n     ip   *   *   *   *  accept
```

Listing 5.3 – Complex statements example using Al-Shaer and Hamed [75]'s model

```
  1   tcp    177.129.1.56    47594   1.35.32.34     50533   deny
  2   tcp    118.226.112.42  16297   58.18.30.68    19236   deny
  3   tcp    124.84.62.221   50533   115.1.27.102   53473   deny
  4   tcp    189.119.171.195 19236   172.239.25.136 22175   deny
...
  n   tcp    any   any    any   any    accept
```

5.2.5 Network speeds and platforms

Many researchers have highlighted that security equipment does not often cope well with increasing network speeds [174, 80, 139]. With respect to the results shown in [30], it can be assumed that performance is related to the dimensioning of network equipment. A possible hypothesis would be that a device designed to work up to a certain network throughput will have no problem coping with strong security requirements on a network which has a throughput that is much lower. Many researchers present results for high-speed networks [134, 138, 139], however, they do not provide evidence that the issue of security against performance only applies to the settings they have chosen in terms of experimental conditions. It is also relevant to understand the feasibility of security policies depending on the target platform (Section 4.4.1, Appendix C), and therefore the majority of the experiments have been carried out on both platforms, Cisco and Linux firewalls, and for all the network speed settings they support (Appendix C). Evidently, organisations aim to use their investment, and thus their equipment, to their fullest; a possible outcome of this hypothesis verification could be the deployment of a load-balancing system, for instance.

5.3 Formalisations

DEENS produces a large amount of raw data (Section 4.3.2) and this does not allow for a comprehensive analysis and interpretation of the results obtained. Along with

this, it is necessary to relate the data collected to the dynamic firewall performance model (Section 4.4). Hence, in order to facilitate analysis, the evaluation scenarios (Section 5.2) are formalised along with the inputs DEENS supports (Section 4.4.1), and the metrics it measures (Section 4.4.3). This formalisation, in turn, allows for a search of relevant data in an automated manner.

5.3.1 Inputs Formalisation

Section 4.4.1 specified the values that each input of the parameters for the dynamic firewall performance model can take, and this section presents how these are organised into sets in order to allow model instance IDs to be selected based on set operations, such as an union operation [175].

Definition 1 (Sets of Input Values)
The set of supported platforms is:

$$S_{Plat} = \{Cisco, Linux\} \tag{5.1}$$

The set of possible logical implementations related to the filtering of traffic between the source and sink sub-networks (Section 4.4.1, Table 4.2, and Table 4.3) is:

$$S_{ImpLog} = \Big\{ \{In_{dir}\}, \{Out_{dir}\} \Big\} \tag{5.2}$$

The set members of S_{ImpLog} contain the following elements:

$$In_{dir} = \{(Cisco_{Int1} \wedge Filter_{in}), Linux_{Int1}, Linux_{INF}\} \tag{5.3}$$

$$Out_{dir} = \{(Cisco_{Int2} \wedge Filter_{out}), Linux_{Int2}, Linux_{INF}\} \tag{5.4}$$

The set of supported speeds is:

$$S_{Speed} = \{Slow, Fast\} \tag{5.5}$$

The set of supported OSI layers is:

$$S_{Layer} = \{Layer3, Layer4\} \tag{5.6}$$

The set of possible critical rule positions is:

$$S_{Pos} = \{Top, Middle, Bottom\} \tag{5.7}$$

The set of rule-set sizes is:

$$S_{Size} = \{500, 1000, 2000, 4000, 8000, 16000, 32000, 64000\} \tag{5.8}$$

5.3.2 Outputs Formalisation

The results can be represented as a sequence of metrics, and defined as follows.

Definition 2 (Sets of Output Metrics)

$$S_{Results} = \left\langle \{M_{CPU}\}, \{M_{Memory}\}, \{M_{Latency}\}, \{M_{Throughput}\} \right\rangle \tag{5.9}$$

where:

$$M_{CPU} = \{CPU_{av}, CPU_{pDiff}, CPU_{er}\} \tag{5.10}$$

$$M_{Memory} = \{Mem_{av}, Mem_{pDiff}, Mem_{er}\} \tag{5.11}$$

$$M_{Latency} = \{Lat_{av}, Lat_{pDiff}, Lat_{er}\} \tag{5.12}$$

$$M_{Throughput} = \{Throughput_{av}, Throughput_{pDiff}, Throughput_{er}\} \tag{5.13}$$

The annotations M_{av}, M_{pDiff}, and M_{er}, relate to the average of the readings, the percentage difference with the baseline results, and the error-rate for the metric M respectively.

Therefore, a search is defined as the intersection of the inputs sets (Definition 1):

Definition 3 (Search formal definition)

$$S_{Search} = \bigcap \{S_{Plat}, S_{ImpLog}, S_{Speed}, S_{Layer}, S_{Position}, S_{Size}\} \tag{5.14}$$

5.3.3 Evaluation Scenarios Formalisation

Scenario 1

The first scenario focuses on the effects on dynamic performance (Section 4.4) of increased security requirements which typically translates into larger firewall rule-sets (Section 5.2.1) on the DUT. Section 4.4.1 outlined that the direction filtering functionality of the Netfilter firewall engine is being phased out, and thus thus the analysis for this scenario needs to reflect aspect as well, as both DUTs are under scrutiny. Thus, this scenario is divided in three parts that reflects this requirement.

Definition 4 (Scenario 1)
This scenario is composed of three elements:

$$S_{Sc1} = \Big\{ \{S_{Sc1p1}\}, \{S_{Sc1p2}\}, \{S_{Sc1p1}\} \Big\} \tag{5.15}$$

where:

$$S_{Sc1p1} = \begin{array}{c} Cisco \wedge (In_{dir} \vee Out_{dir}) \wedge \\ Fast \wedge Layer3 \wedge Bottom \end{array} \tag{5.16}$$

$$S_{Sc1p2} = \quad Linux \wedge Fast \wedge Layer3 \wedge Bottom \wedge 1000 \tag{5.17}$$

$$S_{Sc1p3} = Linux \wedge Fast \wedge Layer3 \wedge Bottom \tag{5.18}$$

The analysis part for S_{Sc1p3} incorporates the findings for S_{Sc1p2}; in other words whether or not the filtering direction matters in terms of dynamic performance.

Scenario 2

The second scenario relates to the fact that network firewalls typically inter-connect multiple networks together, and thus, due to their rule-based nature, the critical rule position (Section 4.4.1) varies from one network to the next. As it is time consuming to create model instances, it is essential to establish if dynamic performance is related solely to the size of the rule-set or the number of rules the firewall examines before reaching a decision (Section 5.2.3). Hence, the effect of the critical rule position is formalised as follows.

Definition 5 (Scenario 2)
This scenario is composed of two elements:

$$S_{Sc2} = \Big\{ \{S_{Sc2p1}\}, \{S_{Sc2p2}\} \Big\} \tag{5.19}$$

Where:

$$S_{Sc2p1} = \begin{array}{c} \Big(Cisco \wedge (In_{dir} \vee Out_{dir}) \wedge Fast \wedge Layer3 \wedge Middle \wedge 32000\Big) \\ \cup \\ \Big(Cisco \wedge (In_{dir} \vee Out_{dir}) \wedge Fast \wedge Layer3 \wedge Bottom \wedge 16000\Big) \end{array} \tag{5.20}$$

$$S_{Sc2p2} = \begin{array}{c} \Big(Linux \wedge Fast \wedge Layer3 \wedge Middle \wedge 2000\Big) \\ \cup \\ \Big(Linux \wedge Fast \wedge Layer3 \wedge Bottom \wedge 1000\Big) \end{array} \tag{5.21}$$

Scenario 3

The third scenario relates to assessing the effects of rule-sets that exercise fine-grained control over several network protocols, applications, or services (Section 2.5, and Section 3.6).

Definition 6 (Scenario 3)
This scenario is composed of two elements:

$$S_{Sc3} = \Big\{ \{S_{Sc3p1}\}, \{S_{Sc3p2}\} \Big\} \tag{5.22}$$

where:

$$S_{Sc3p1} = Cisco \wedge (In_{dir} \vee Out_{dir}) \wedge Bottom \wedge 1000 \tag{5.23}$$

$$S_{Sc3p2} = Linux \wedge Fast \wedge Bottom \wedge \{500, 1000\} \tag{5.24}$$

5.4 dynamic Firewall Evaluation Results AnaLyser

This section introduces **d**ynamic **F**irewall **E**valuation **R**esults **A**na**L**yser (d-FERAL), an application designed to bring together data relevant to each scenario described in Section 5.2, and formalised in Section 5.3. It also presents the type and format of outputs produced by d-FERAL from the data collected with DEENS.

5.4.1 Purpose

d-FERAL is a software tool capable of identifying model instances that meet analysis criteria, such as the critical rule position. Figure 5.2 shows the **G**raphical **U**ser **I**nterface (GUI) for d-FERAL. The GUI provides access to the search criteria described in Definition 1. d-FERAL also automates data processing tasks, such as: computing the readings average; computing the percentage difference with the baseline results (Appendix B); and establishing the error-rate for each metric. Finally, d-FERAL converts the processed data into a format suitable producing charts and figures, such as with the GnuPlot engine [176].

Indeed, DEENS provides raw results because the Metric Collection Sequencer (Section 4.3.2) simply archives and stores the readings for each snapshot (Section 4.3.2), and does not perform operations, such as computing the readings average or preparing the data for plotting. In addition, the Metric Collection Sequencer keeps the readings separated into files according to the network load, and these are stored on a per model instance basis. This approach separates the collection of data from its processing, which Porter [109] also recommends, and ensures the portability of the Metric Collection Sequencer application (Section 4.3.2). Along with this, the measurement apparatus (Section 4.5.4) can be modified without necessarily altering the

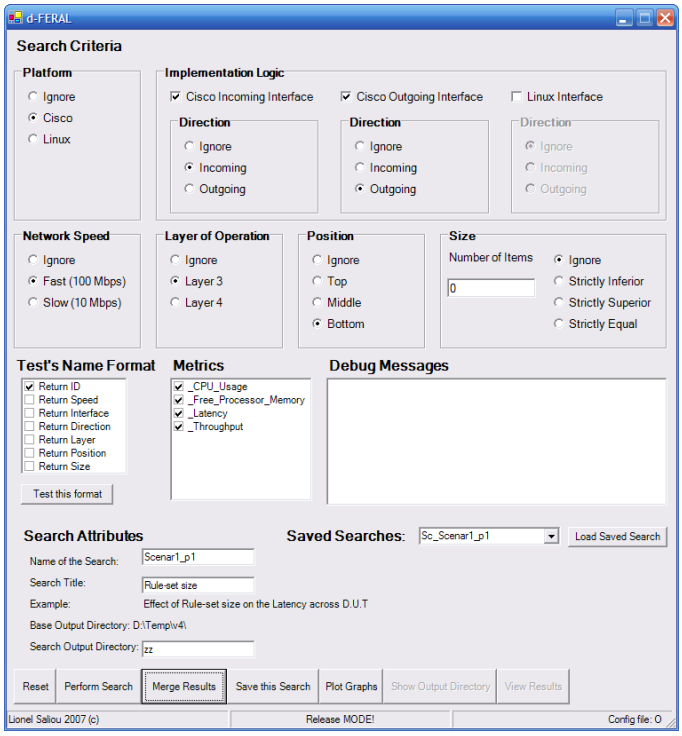

Figure 5.2 – Graphical User Interface for d-FERAL.

manner in which data is processed and vice versa. Unfortunately, this does not allow for a comprehensive interpretation or understanding of the results obtained.

Section 4.3.2 showed that carrying out one evaluation is highly time consuming, even with automation. The evaluations are scheduled so that the configuration time, between any two evaluations, is kept to a minimum as opposed to organising these according to the scenarios they relate to. Indeed, it is less time consuming to change the direction on which the rule-set is applied than performing all the evaluation requiring different rule-set sizes, such as necessary for Section 5.2.1, before moving onto the filtering direction (Section 5.2.2). Hence, the data relevant to the analysis of each scenario is typically distributed across multiple files and folders. There is also the possibility that the data might be relevant to multiple scenarios, and a per-scenario classification could produce duplicate data.

5.4.2 Formalisation of dynamic Firewall Evaluation Results AnaLyser Functionality

d-FERAL maps the results to the characteristics with which the DUT was evaluated. Thus using Definition 1, and Definition 2, this functionality is formalised as follows:

Definition 7 (d-Feral formal definition)

$$S_{Plat} \times S_{ImpLog} \times S_{Speed} \times S_{Layer} \times S_{Position} \times S_{Size} \mapsto \{y|y :\in S_{Results}\} \tag{5.25}$$

For instance, if an analysis focuses on the effect of the network speed (Section 4.4.1) on a given configuration of the Cisco DUT, the corresponding search equations include:

$$S = Cisco \wedge In_{dir} \wedge Layer3 \wedge Bottom \wedge 1000$$

5.4.3 Mode of Operation

This section describes d-Feral's mode of operation (Figure 5.3).

Stage 1: Specify

The selection criteria for the model instance IDs are defined through d-Feral's GUI (Figure 5.2) which is organised according to the definition of the input sets (Definition 1), and lists the possible values for these.

Stage 2: Search

The chosen values can then serve as input for the search parameters (Figure 5.3). For each input value d-Feral supports there exists a processing element that determines if a given model instance, such as represented in Listing 5.4, matches it. Hence for each criterion d-Feral creates a set [177], and thus the results of the intersection between all the sets (Definition 3). One of the advantages of using sets and binary operations is that there is no duplicates in the result set (Definition 3).

Listing 5.4 – Characteristics Hashtable

```
Cisco0041, 100|CiscoInt1|in|Layer3|bottom|32000
```

Stage 3: Compute

At start-up d-Feral loads two sorted lists into memory [178]. The first maps the model instance identifiers to the evaluation characteristics used, and Listing 5.4 shows the format utilised to that end. The second maps the model instance identifiers to the location where the raw data is stored. d-Feral reads-in the data collected during the evaluation, and computes the following:

- The average of the readings.
- The error-rate for each metric (Section 4.4.3).
- The percentage difference between the readings and the baseline results (Appendix B).

Finally, the data is transposed so that effect of the DUT's configuration on dynamic performance can be analysed. As d-Feral only processes data that are relevant to the specified search, it is not limited by the volume of data DEENS produces (Section 4.5). Section 5.4.4 to Section 5.4.7 describe these operations further.

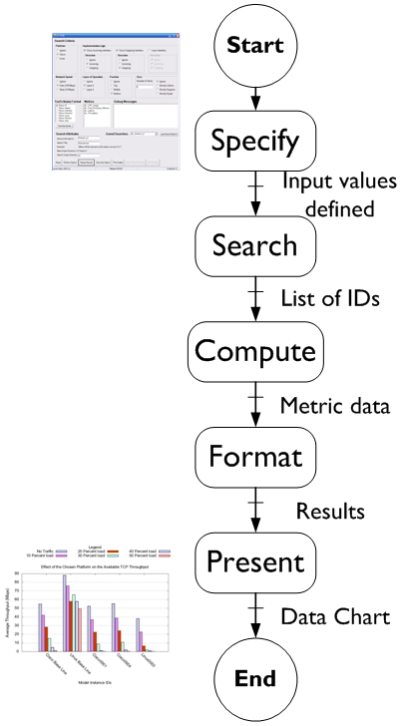

Figure 5.3 – d-Feral data flow.

Stage 4: Format

GnuPlot is a scientific data and function plotting utility [176], and it requires the plot data to be organised according to a pre-defined format before data can be visualised. d-Feral prepares the metric data (Figure 5.3) to match GnuPlot specifications, and also generates the scripts necessary to render the data in a graphical form.

The number of operations that are supported by GnuPlot data files is limited. In particular, it is not possible to perform a matrix transposition by simply changing the commands included in the script file. The source plot data thus needs to be organised differently. For this, d-Feral prepares two distinct sets of data plot files and visualisation scripts to overcome this limitation.

Stage 5: Present

This stage involves calling the GnuPlot utility and have it process a top level script that defines global options, such as line style, so that for a given search a model instance ID is always represented in the same manner. This top level script then activates the scripts created during Stage 4.

5.4.4 Average of actual readings

As described in Section 4.4, each evaluation represents one instance of the model, and each instance refers to defined values of the input parameters (Definition 1). Table 5.1 shows the details of one such instance for a Cisco 2600 XM (Appendix B) as the

DUT. Figure 5.4 illustrates the first type of output d-Feral produces, and this figure shows for the available network throughput, in other words $Throughput_{av}$ (Definition 2, and Equation 5.13), once the configuration described in Table 5.1 is deployed onto the DUT, along with the baseline results.

Table 5.1 – Example of an experiment's details

Model instance ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Cisco00041	100	CiscoInt1	in	3	bottom	32000

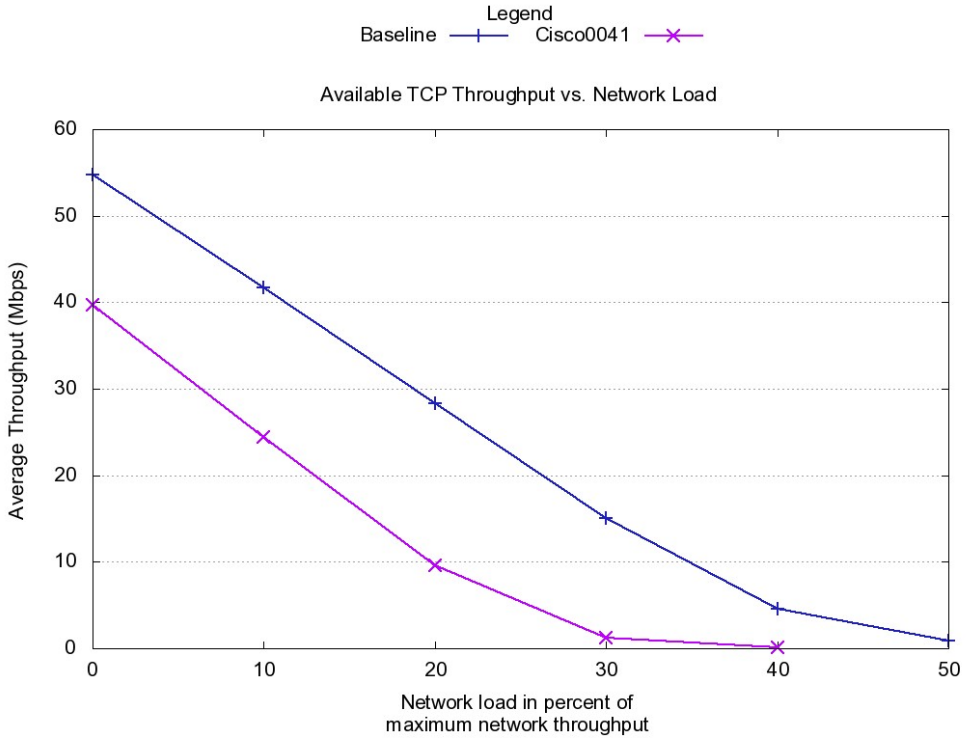

Figure 5.4 – Available network bandwidth across the DUT for the model instance ID Cisco00041.

5.4.5 Error-rate computation

The storage format of the DEENS data allows the d-Feral's file parser to establish the number of attempts that were made to measure a given metric (Section 4.5.4, and Section 4.4.3). This information, $m_{Expected}$, is compared against the number of values indeed obtained, $n_{Obtained}$. Hence, d-Feral implements Equation 5.26 and the results, based on Cisco00041 data (Table 5.1), are listed in Table 5.2.

$$E_{Rate} = 1 - \frac{n_{Obtained}}{m_{Expected}} \tag{5.26}$$

The error-rate defines the window of interest (Section 4.4.3) for the metric it is

Table 5.2 – Example of an error-rate table for available network bandwidth.

Model instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Cisco0041	0	0	0	0	0	1

associated with (Definition 2). Thus, from an analysis point-of-view, Figure 5.4 shows that the size of the rule-set greatly diminishes the DUT capability to handle bulk data when the network traffic level reaches 30%, and beyond. Combining the error-rate data with the metric average data indicates that the windows of interest (Section 4.4.3) applies to measurements carried out between 0% and 40% of network load. Indeed, these measurements are considered credible because the error-rate is zero as Table 5.2 shows. Along with this, Table 5.2 underlines the fact that the available network bandwidth for 50% of maximum network thoughput is not 0 Mbps but unmeasurable for these evaluation conditions.

5.4.6 Percentage difference

Saliou et al. [78] highlight that with some performance metrics, such as CPU usage, the key data is the relative difference between readings, and add that the actual value of the measurements can be misleading, such as when these are unexpectedly high. Hence, the experimental readings can also be assessed using the percentage difference, $\Delta_{Percentage}$, with the baseline results. Each baseline result is considered as the benchmark against which the readings obtained for measurements performed for the same evaluation conditions is compared. Hence, d-Feral implements Equation 5.27 whose results, based on Cisco0041 (Table 5.1), are plotted in Figure 5.5.

$$\Delta_{Percentage} = \left(\frac{Value_{Experiment}}{Value_{Baseline}} - 1 \right) \times 100 \tag{5.27}$$

5.4.7 Transposed data

Arguably, the manner in which the data is collected (Section 4.5) favours plotting the results with a focus on the effect of the network load (Section 4.4.2), whereas it can be relevant to assess the effect of the DUT's configuration on the dynamic performance, such as when the filtering direction is changed (Section 4.4.1, Section 5.2.2, Definition 1, and Equation 5.2). Hence, d-Feral also performs a matrix transposition of the processed data so that the x-axis is changed from the network load to another input variable. Consequently, the x-axis is composed by the values undefined for the search criteria.

This section illustrates this functionality based on the search shown in Equation 5.28, and Table 5.3 lists a sample of the model instance IDs that match these criteria. Thus, Figure 5.6 and Figure 5.7 illustrate the effect of the choice in terms of firewall

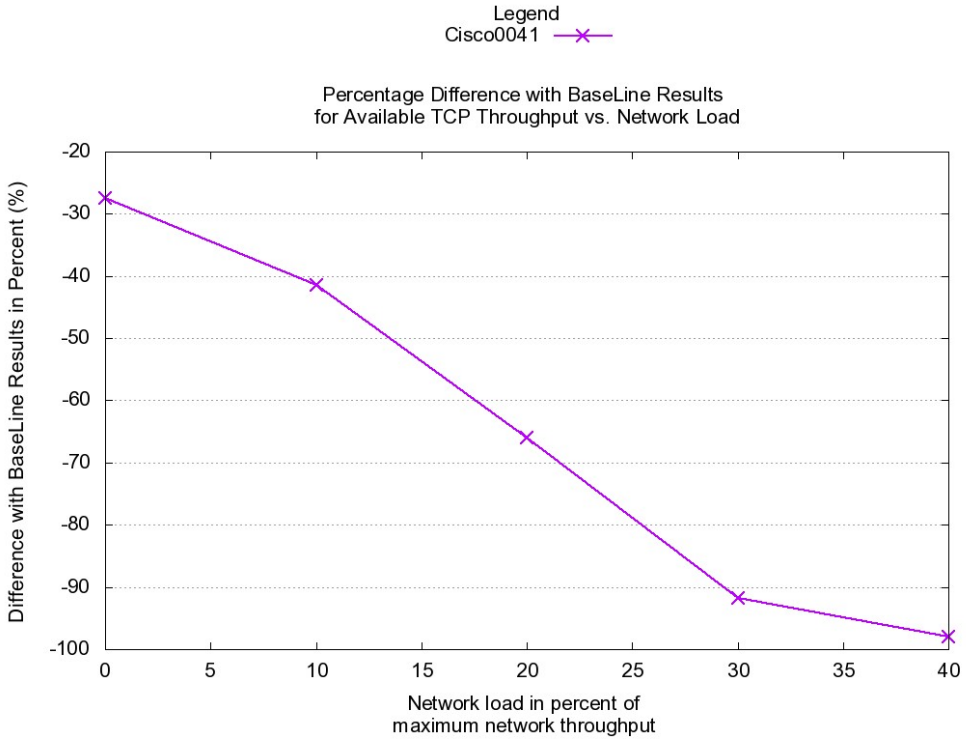

Figure 5.5 – Example of the percentage difference representation.

platform (Section 4.4.1) for a given rule-set.

$$S = Fast \wedge Layer3 \wedge Bottom \wedge 1000 \tag{5.28}$$

Table 5.3 – Example of selected instances for data transposition with emphasis on the variables parameters added.

Model instance ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Cisco0001	100	CiscoInt1	in	3	bottom	1000
Cisco0004	100	CiscoInt2	out	3	bottom	1000
Linux0003	100	n/a	n/a	3	bottom	1000

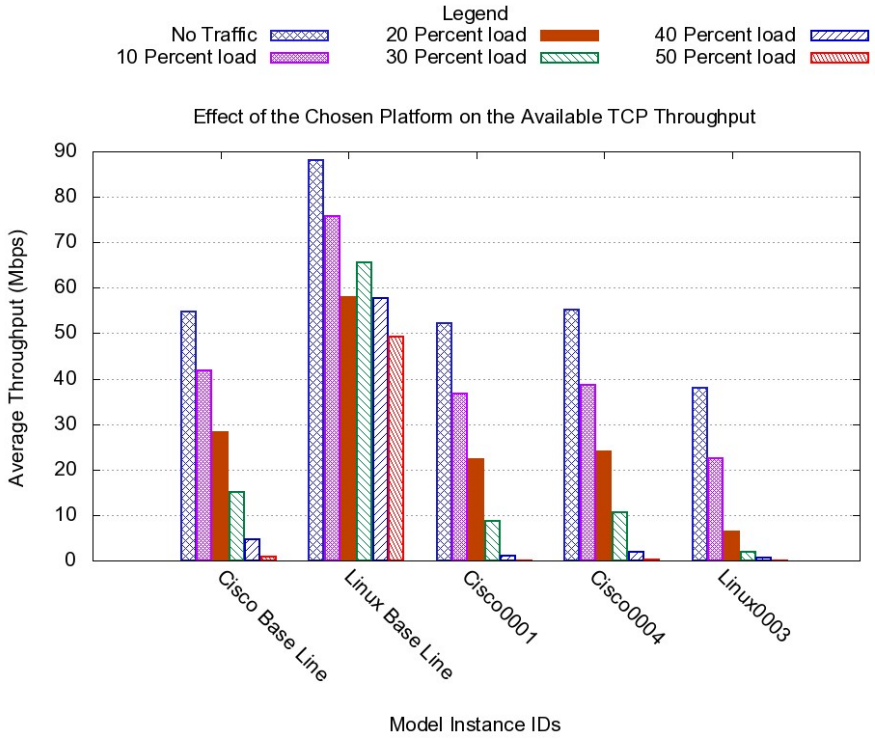

Figure 5.6 – Example of data transposition representation for actual readings.

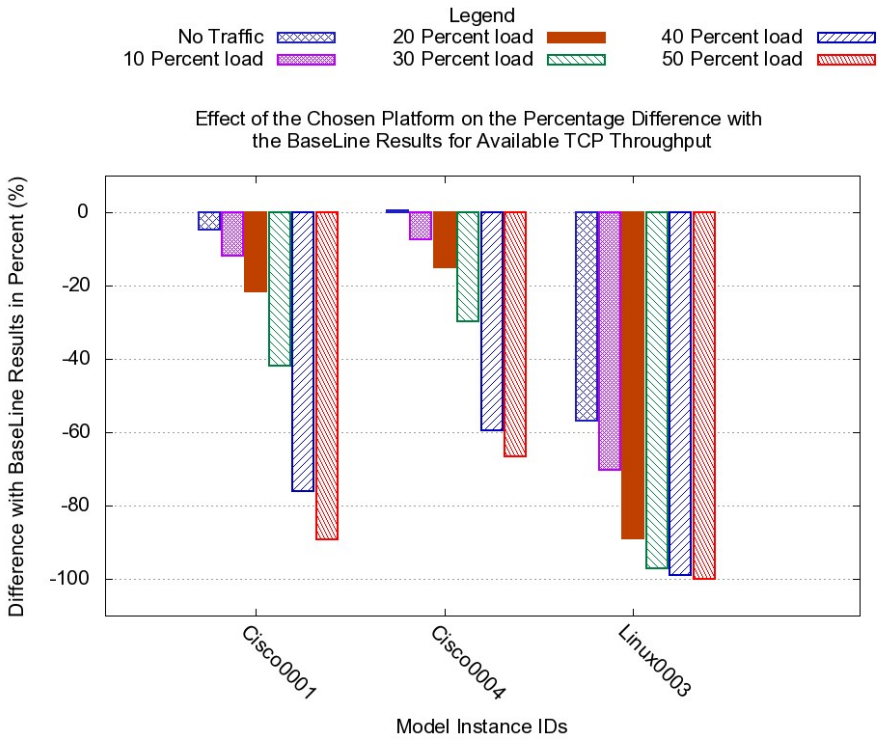

Figure 5.7 – Example of data transposition representation for percentage difference.

5.5 Conclusion

This chapter identified several issues related to the configuration of network firewalls, and the deployment of security policies onto them. These issues include:

- The rule-set size that results from the security policy.
- The position of the critical rule, such as when the network firewall is located at the cross-road between several sub-networks.
- The complexity of the items that compose the rule-sets, which also typically results from the security policy.
- The device that is chosen to enforce the security policy, as well as the manner in which it is configured.

If these issues are not investigated then the deployment of security policy becomes unscientific and difficult to verify. Along with this, the network might become more vulnerable to DDoS attacks [143].

In order to build a consensus on these issues, multiple instances of the network firewall dynamic performance model are necessary. In order to facilitate analysis, it is required to formalise the inputs and outputs of the dynamic firewall performance model (Section 4.4, and Section 5.3). This, in turn, permits the formalisation of the above scenarios, as well as the creation of an analysis tool (d-Feral) to deal with the large amount of data DEENS produces. Hence, d-Feral allows taking raw performance data, select relevant model instance IDs according to investigation criteria [21, 179], and present the results in a form suitable for analysis in an automated manner. Most importantly, d-Feral is not limited to the analysis of the effect of network load (Section 4.4.2). Indeed, d-Feral, through matrix transposition, extends the scope of analysis as it is capable of showing the effect of DUT's configuration on dynamic performance (Section 5.4.7).

Chapter 6

Scenario-based Evaluations

6.1 Introduction

This chapter presents an analysis of the evaluation scenarios described in Section 5.2. The outcome of these evaluation scenarios are part of the knowledge base employed to establish the feasibility, and thus suitability and verification of security policies. To that end, these scenarios follow the evaluation methodology that Section 4.5 describes, rely on the use of DEENS (Section 4.3) to create network firewalls performance model instances (Section 4.4), and d-Feral then compiles the relevant data from these evaluations (Section 5.4). To aid reading, it is recommended to have a copy of Appendix A for easier reference to the figures and tables.

6.2 Scenario 1: Performance Evaluation of Increasingly Large Rule-sets on Stateless Firewalls

One of the major obstacles to the deployment of security policies is there is concern that these will hinder functionalities and reduce the performance of the computer network systems on which they are applied. Due to their rule-based nature (Sections 2.5, and 3.6.7) network firewalls with strong security requirements typically translate into larger firewall rule-sets (Section 3.6.8). Chapter 3 showed the central role of these devices within an organisation, and also highlighted the above concerns. Still, there is a need to measure the impact of increasingly large rule-sets on network firewalls, while they are in use, and thus for their dynamic performance (Section 4.4). This expert knowledge provides a basis to establish whether a chosen security policy is achievable within the means of the organisation [18], and thus whether it is acceptable.

This scenario is analysed with data collected from network firewall dynamic performance model instances (Section 4.4) obtained with DEENS (Sections 4.3, and 4.5). Such an approach allows taking into account the various possible technical parameters involved in deploying security policies (Section 4.4.1), and all these combinations and their analysis are essential in choosing the appropriate configuration in function of the security requirements. Hence, the issue of increasingly large rule-sets is investigated for two devices and in two different directions. First of all, a Cisco device is used (Appendix C) where both incoming and outgoing filtering implementations

(Section 4.4.1, and Figure 6.1) are evaluated. For the Linux DUT (Appendix C), since the direction capabilities are being phased out, it is essential to establish whether there are any performance drawback or benefits in continuing to use direction-based configurations before focusing solely on increasingly large rule-sets.

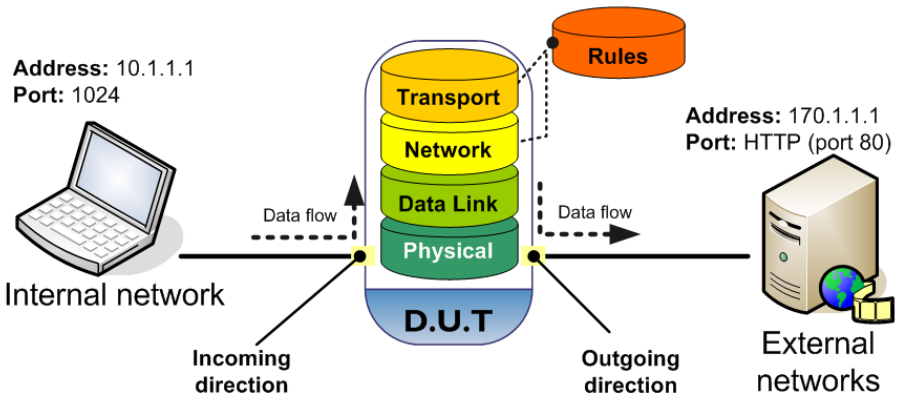

Figure 6.1 – Network firewall's possible filtering directions.

6.2.1 Effects on a Cisco Device

This section focuses on the performance impact of increasingly large firewall rule-sets on a Cisco 2600 XM router firewall (Appendix C). With the use of DEENS several unique instances of the network firewall dynamic performance model are created. The size of the rule-set the DUT enforces is gradually increased using rule-sets of 1000, 2000, 4000, 8000, 16000, 32000, and 64000 statements. The process of gathering the readings for the performance metrics (Section 4.4.3, and Section 4.5.4) was repeated for both incoming and outgoing filtering directions, as well as for all the network speeds (Section 4.4.1) that the DUT supports. The data collected for slow network speed showed that there is a limited dynamic performance overhead introduced by increasingly large rule-sets, even for model instances that enforce 64,000 statements. Thus, the analysis presented in this part of the scenario solely uses model instances based on fast network settings. Table 6.1 lists the model instances that match Definition 4 (Equation 5.16), and provides their respective implementation details.

Table 6.1 – List of Model Instances selected for this analysis.

Scenario ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Cisco00017	100	CiscoInt1	in	3	bottom	4000
Cisco00020	100	CiscoInt2	out	3	bottom	4000
Cisco00033	100	CiscoInt1	in	3	bottom	16000
Cisco00036	100	CiscoInt2	out	3	bottom	16000
Cisco00049	100	CiscoInt1	in	3	bottom	64000
Cisco00052	100	CiscoInt2	out	3	bottom	64000

Analysis

Direction The results (Section A.2.1) show that the filtering direction has an effect on the network metrics as well as on the window of credibility for all the metrics, and the outgoing filtering allows for improved dynamic performance. Indeed, for the readings obtained in the absence of network traffic, it is challenging to distinguish between the incoming filtering configuration and the outgoing one. However, the benefit of the outgoing filtering is more pronounced with background traffic. At 20% of network load, for example, the latency for Cisco00036 increases by 40.069% (Table A.7), in other words 4.3 ms (Table A.6), with respect to a baseline of 3.081 ms (Section 5.4.6, Equation 5.27), whereas for Cisco00033, the increase is 266.085%, in other words 11.279ms, for the same network conditions.

Figure A.6 further demonstrates the benefit of using the outgoing filtering implementation over the incoming one, with respect to the fact that larger rule-set can be deployed with smaller overhead as long as these rule-sets use the outgoing filtering. Figure A.6 shows that the overhead for Cisco00036, for instance, is greater than the one incurred by Cisco00017 (Table A.9). Arguably, this can be expected as the rule-set in Cisco00036 is much larger with 16000 items compared to Cisco00017 with 4000 (Table 6.1). However, as the network load increases, the impact of 4000 items applied in the incoming direction becomes greater than the one introduced by 16000 items applied in the outgoing direction once the network load reaches 30%, for instance (Figure A.6). Along with this, Figure A.6 illustrates that, with the exception of Cisco00049, the percentage difference graphs for the model instances almost have the same shape as long as they represent the same implemention, such as for Cisco00017 shown in and Cisco00033 shown in . Hence, for the purpose of building a formal model for network firewall dynamic performance, one of the key required elements would be the offset at which the graph starts, in other words the *static* footprint incured by the rule-set size (Section 4.4.2).

The CPU usage and available memory measurements, as well as the readings for the latency, also show the benefit of using the outgoing filtering direction. Considering, for instance, the error-rates for the latency when the network load is at 50%, Cisco00017 and Cisco00020 are close, 0.38 and 0.35 respectively (Table A.8), however the error-rate for Cisco00020 and Cisco00036 remains stable at 0.35 (Table A.8) while

Ciscoo0036 enforces four times as many rules (Table 6.1). On the other hand, the error-rate for Ciscoo0033 is 0.53. Moreover, the error-rate for Cisoo0052 remains 0.35 at 50% of network load, whereas no measurements could be made at all for Ciscoo0049 for the same network load.

Available network throughput Table A.10 show that rule-set sizes incur a distinct footprint from the moment the rule-sets are deployed onto the DUT. It is noteworthy to underline that, when readings are collected with no background traffic, it is challenging to distinguish the incoming filtering from the outgoing one, whereas, the dynamic approach allows differentiation between them. For example at 20% of network load (Table A.10), Ciscoo0049 has a footprint of 90.81% whereas Ciscoo0042 which enforces the same size rule-set (Table 6.1) only incurs a footprint of 48.33%. Their intial footprint, in other words for 0% of network load, are 38.81% and 38.22% respectively.

Figure A.6 highlights that performance does not decrease linearly, and from 40% of network load, the available throughput does not significantly decrease further. This phenomenon can arise from two factors. On the one hand, both DEENS and the Cisco DUT might have reached their limits in terms of hardware and software capabilities, or, the measurement methods employed in DEENS has side effects on the evaluation.

Latency The observations from the available network throughput readings correlates the evidence that Table A.6 and Table A.7 show: the rule-set sizes incur distinct overhead. Although, this phenomenon for the latency is more apparent once network traffic is introduced. The latency readings across the DUT for the baseline show their most notable change when the network load goes above 20% where the latency increases from an average of 3.081 ms to 13.04 ms (Table A.6). The same observation applies to the model instances selected. For example, the latency for Ciscoo0036 goes from 4.316 ms, at 20% of network load, to 20.35 ms at 30% network load, whereas the initial measurements (0% network load) averaged at 0.706 ms and reached 1.181 ms for 10% of network load.

Ciscoo0049 shows a rapid increase in latency across the DUT, and reaches its maximum of 314.5 ms for 30% of network load, after which point no readings could be obtained. This further constitutes evidence that the DUT has a fail-over point, and for the Ciscoo0049 model instance, it happened over 30% of network load. This also highlights a limitation in terms of the number of firewalling statements the DUT can support for this filtering direciton.

CPU Usage The CPU usage readings show that the manner in which the evaluation is orchestrated has an effect on the results for this metric. Indeed, Table A.2, the CPU usage percentage difference (Section 5.4.6), suggests that the effect of the

rule-set sizes on the DUT CPU usage is not consistent across all selected model instances. For example, Ciscoo020 shows a substantial overhead (123.08%) when there is no background traffic (network load of 0%) that is followed with better than baseline performance once background traffic is introduced, -7.16% at 10% network load (Table A.2). In addition, the difference between the baseline results and the model instances reading is at its highest at the beginnning of the measurement procedure (Table A.2). This phenomeon can be explained by the fact that the evaluation procedure starts with the CPU usage measurements. Indeed, this measurement occurs after the DUT is configured to the specifications of the evaluation and a pause of 30 seconds (Section 4.5.2). This pause is due to the computation intensive nature of the reconfiguration activity as demonstrated by Saliou et al. [78], and, from Table A.2, it could be argued that this *cool down* period needs extending.

Hence, the CPU usage readings are most credible for network loads of 10% and 20%, and, for the readings concerned, this suggests that there is an overhead introduced by increasingly large rule-sets when these are deployed on the incoming filtering direction. In some respects, this observation also applies to readings obtained for a network load of 30%, however there are missing measurements for evaluation instances that employ the incoming filtering direction (Ciscoo017, Ciscoo003, and Ciscoo049), and this factor needs to be taken into consideration. Figure A.1, and Table A.1 are evidence that the network load is the principal contributor to this metric.

High CPU usage can be caused by a variety of security exploits, such as routing table poisoning, or security breaches, such as computer worm outbreak [180]. In this respect, the CPU usage can help identify configurations that are not optimal for the DUT, as, for example, the CPU usage for Ciscoo049 is very high from the moment network traffic is introduced, and reaches 92.40% at a network load of 20% (Table A.1). Therefore, the dynamic CPU readings can help distinguish between normal operational state and routing poising exploits as the model instances can serve as a benchmark for comparison during day-to-day monitoring.

Processor Memory The results suggest that the amount of memory available to the processor (Section 4.5.4) does not have any dynamic characteristics. Indeed, this DUT has a total of 32 MB of memory space available (Appendix C) with which to run the operating system, and all networking tasks, such as input and output buffers servicing [78]. This amount of memory space is far less than the Linux DUT (Appendix C), and thus could constitute a contributing factor to the device not being able to cope with increasingly large rule-set sizes, as the DUT has to manage this resource efficiently. Yet, Table A.4 shows that the readings hardly vary from one network load to the next compared to the measurement obtained at the beginning of the evaluation procedure. More importantly, the main contributing factor appears to the size of the rule-set deployed onto the DUT (Figure A.7). Indeed, the baseline results are drawn from a minimal functional configuration for the DUT (Section 4.5.1),

and pairs of instances that enforce the same size rule-set typically have a similar memory footprint. For instance, Cisco00017 and Cisco00020 enforce a rule-set of 4000 items and incur a footprint of 0.93%. Similarly, Cisco00033 and Cisco00036 enforce a rule-set of 16000 items that results in a footprint of 3.46%. Arguably, these two sets of data could help project a footprint of 13.84% for a rule-set of 64000 items, and indeed Cisco00049 and Cisco00042 incur, on average, a footprint of 13.67% (Table A.4). In addition, the knowledge obtained can be used for planning or evaluation purposes when implementing a security policy as well as in uncovering attacks that exhaust memory to succeed.

Error-rate Arguably, this metric provides an evaluation of the stress conditions the DUT is under. Indeed, measuring the CPU usage and the available amount memory to the processor is a challenging task, even with a dedicated sub-network for this task (Section 4.5.4), as Tables A.3 and A.5 show that baseline results have themselves missing measurements (Section 4.4.3). Tables A.3 and A.5 demonstrate the effect that the deployment of firewall rule-set has on the monitoring capability of the Cisco DUT, indeed the first measurement errors often appear before a network load of 40% and are also typically higher than the one reported for the baseline. Hence, the deployment of larger rule-set diminshes the window of credibility of both the CPU usage and available memory measurements (Section 4.4.3, Figure A.2, and Figure A.3). Interestingly, Cisco00020, Cisco00030, and Cisco00052 (Table 6.1) are evaluation instances where the rule-set is applied in the outgoing direction (Figure 6.1, Section 4.4.1, Section 5.2.2, and Section 3.6.4) and this implementation choice allows for accurate results for measurements carried out up to 30% of network of load, as no errors were reported for these instances up until this point. For the model instances that employ the incoming filtering (Table 6.1) missing measurements occur at an earlier stage, and in the case of the instance Cisco00049 it is as soon as 20% of network load. Arguably, Table A.3 and Table A.5 provide a measure of how well this DUT copes with monitoring or management tasks, in combination with increased security requirements, as the error-rates for each model instances for both the CPU usage and the available memory measurements are close.

Table A.8 shows that the readings for the latency across the DUT have, overall, more missing measurements than other metrics (Tables A.3, A.5, and A.11), and this is perhaps due to the fact that a large number of individual readings are collected (Section 4.5.4). Nevertheless, the error-rate for the latency across the DUT is relatively low; Cisco00033 has the highest latency measurement error-rate with 0.53 at 50% of network load, for example. The exception to the low error-rate is Cisco00049 for which it is not possible to obtain latency readings when the network load is above 30%.

Perhaps not surprisingly the available network throughput measurements have the overall lowest error-rate, and for all model instances that use outgoing filtering direction there are no missing measurements (Table A.11). In all the metrics,

Ciscoo049 has the highest error-rate when the network load reaches 30%, and beyond that point typically no readings are obtained. This indicates that the choice of filtering direction, combined with the size of the rule-set and the network load, lower drastically, the dynamic performance of the Cisco DUT. The comparaison with the model instance Ciscoo052, however, shows that this mostly down to the implementation choice as in both model instances the rule-set size is of 64000 items (Table 6.1).

6.2.2 Effects of Direction filtering on the Linux Device

This section investigates if there are any impacts in using the direction flag option in Netfilter [133]. Indeed, in the version of Netfilter employed in these evaluations, filtering direction does not need to be specified for the firewall device and its rule-set to operate properly. This is a notable difference with previous version of this firewalling technology, as well as other similar devices [106], and such as reported in Section 6.2.1. The absence of a specific filtering direction leads to two outcomes. On the one hand, the firewalling process determines on its own when it is appropriate to compare network packets against the statements in the rule-set. Alternatively, network packets are always compared against firewall statements regardless of origin or destination. With Netfilter such direction filtering is implemented by examining the name of the physical interface from where network packets are captured, or meant to be forwarded to (Section 4.4.1).

There is an intuitive belief that specifying filtering direction, particularly incoming [78], such as required for Cisco ACL, improves the firewall device performance, however Section 6.2.1 demonstrates otherwise. Hence, it is essential to establish if the same observations as for the Cisco DUT also apply to the Linux DUT (Appendix C) prior to discussing the effects of increasingly large rule-sets. To this end, model instances for which only the direction of filtering differs are taken into account, and thus Table 6.2 lists the selected model Definition 4 (Equation 5.17), and provides the main implementation details.

Table 6.2 – List of selected Model Instances for this part of the analysis.

Scenario ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Linux0105	100	Eth1	out	3	bottom	1000
Linux0003	100	n/a	n/a	3	bottom	1000
Linux0103	100	Etho:1	in	3	bottom	1000

Analysis

CPU usage From the outset the CPU usage readings are high, even for the baseline configuration (Table A.12), and interestingly, this is not coupled with high error-rates. Indeed, the error-rates are low until the network load reaches 40% (Table A.14). Hence, up to 40% of network load the measurements can be considered credible, and these measurements thus suggest that the deployment of the firewall rule-sets only incurs a small overhead for the DUT in terms of CPU usage (Table A.13); Table A.13 shows an overhead of 0.36% at 10% of network load, for example. Thus, the measurements do not allow distinguishing between the selected model instances.

Table A.12 indicates that there is no readings available for the tested configuration at 50% of network load, while on the other hand, Table A.14 reports that at least one sample for each configuration was collected, as the error-rate is 0.91. This discrepency is explained by the fact that the CPU usage is computed using a minimum of two samples, and the Linux DUT's operating system does not maintain a per minute average of the CPU usage natively unlike the Cisco DUT (Section 4.5.4).

Available Memory Unlike with the Cisco DUT, the error-rates for the device-oriented metrics are not consistent. However, the two DUTs share a common trait when it comes to their available memory to the processor dynamic measurements, as the amount recorded at the beginning of the evaluation procedure does not change significantly throughout (Table A.15). Nevertheless, Figure A.8 suggests that employing the direction option in Netfilter dramatically reduces the available memory, with average footprint of 89.04% for Linux0105 and 87.31% for Linux0103 respectively. Arguably, a noteable impact on the available memory space could be expected as these rule-sets are typically stored in memory to allow fast access to their content. In contrast, Linux0003 relies on the firewalling engine to decide of the filtering direction (Table 6.2), and has more memory available than the baseline configuration (Table A.15). Perhaps, a large amount of this resource is utilised, for the baseline configuration, to enable the Linux DUT with superior network performance capabilities compared to the Cisco DUT (Table A.17, Table A.20, Appendix B).

Latency The maximum latency across the DUT for the Linux baseline is 1.65ms at 50% of network traffic (Table A.17), whereas, under the same network conditions, the Cisco DUT has a network latency of 146.83 ms (Table A.6). Hence, any change on the Linux DUT, such as the deployment of firewall rule-sets, is more readily noticeable as Table A.18 illustrates. Figure A.9 shows that the overhead incurred by the deployments is most visible from a network load of 20%, shown with  column. Despite the closeness of the readings for the selected model instances, Table A.18 and Table A.17 show that Linux0003, which does not specify the filtering direction (Table 6.2), performs better than Linux0105, which implements outgoing filtering, and Linux0103, which implements the incoming filtering configuration (Table 6.2). For

example, at 10% of network load, the latency for Linux0003 is 3.2 ms compared to 3.376 ms for Linux0105 and 3.946 ms for Linux0103.

Available network throughput The baseline available network throughput of the Linux DUT is far greater than the Cisco DUT. This, perhaps, reflects a lack optimisation for firewalling tasks over network tasks. Nevertheless, rule-set sizes have a significant impact of the DUT network throughput dynamic performance. Along with this, the deployment of the rule-sets affect the network throughput performance more than it does for the Cisco DUT. Indeed, the rule-sets for all selected model instances are composed of 1000 items which, in turn, incur a footprint of around 55% on average the moment these are deployed onto the DUT (Table A.21). Such footprint is unseen with the Cisco DUT even with rule-sets of 64000 items which typically incur a footprint of 38% (Table A.10).

Unlike with the network latency and available memory metrics, the available network throughput does not permit establishing clearly which model instance offers improved performance (Figure A.11, Figure A.10, Table A.21, and Table A.20). The error-rate for the network throughput only increases when the network is at its maximum, and only model instances that specify the filtering direction are concerned. This observation, however, needs to be contrasted with the fact that the baseline at this point has a network throughput of 49.34 Mbps, whereas the readings of the selected evaluation instances are nil (Table A.20).

Direction This section shows that the latency across the DUT and the available memory are improved when the DUT lets the Netfilter engine decide of the filtering direction. Along with this, from a administration and maintenance point of view, each evaluation instance has a similar reduction in terms of window of credibility for each metric, except for the available network throughput. Also, the CPU usage and available network throughput readings suggest that the three model instances incur the same dynamic performance overhead. Therefore, the evidence provided for this section suffice to justify the usage of this functionality over the direction options in the remaining parts of this chapter.

6.2.3 Effects of Increasingly Large Rule-sets on a Netfilter Device

This section builds on the findings from Section 6.2.2 in which it is established that the direction option of the Netfilter firewalling engine [133] does provide little gain, or does not incur any major footprint, in terms of dynamic performance. Thus, the model instances selected to identify the effect of increasingly large rule-sets on the Linux DUT (Appendix C) dynamic performance do not employ this option.

As in Section 6.2.1, the rule-sets deployed on the DUT follow the same design as the one outlined in Section 4.5.3, and the size of these rule-sets was also gradually

increased. An important observation is that this DUT cannot enforce as many firewalling statements as the Cisco DUT can, and hence the model instances scrutinised in this section enforce smaller rule-sets than the ones discussed in Section 6.2.1. Table 6.3 lists the selected model instances that match Definition 4 (Equation 5.18), and provides their respective implementation details.

Table 6.3 – List of Model Instances selected for this analysis.

Scenario ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Linux0013	100	n/a	n/a	3	bottom	500
Linux0003	100	n/a	n/a	3	bottom	1000
Linux0005	100	n/a	n/a	3	bottom	2000
Linux0007	100	n/a	n/a	3	bottom	4000

Analysis

Available network throughput The Linux DUT allows for relatively good network performance, such as with a latency of 1.65 ms and an available throughput of 49.34 Mbps when sustaining a network load of 50% (Table A.28, and Table A.31). This suggests that the DUT has spare capacity to handle firewalling operations. However, Table A.32 shows that the deployment of a rule-set of 4000 items, the footprint immediately incured is 96.78%, down from 87.99 Mbps to 2.83 Mbps (Table A.31). In addition, early evidence of the device difficulties to cope, with the combination of the configuration employed and network load, cannot be identified with device-oriented metrics (Section 4.4.3), for example.

Table A.28 and Table A.31 show that the Linux0003 model instance offer network performance in part with the Cisco DUT baseline results (Table A.6, Table A.9, and Appendix C), and thus it is likely that for implementations that require less than 500 items in the firewall rule-set the Linux platform represents a better choice.

Latency Linux0013 and Linux0003 incur, respectively, an overhead of 93.144% and 111.081% (Table A.29, Table A.28), however it is noteworthy to underline that their latency is still below the 1 ms mark once their respective rule-sets are deployed. Hence, the network latency is still better than the Cisco DUT baseline readings. The deployment of a rule-set of 2000 items sets the latency at 1.907ms, and 4000 items sets it at 5.475 ms. Arguably, the overhead introduced by Linux0013 and Linux0003 remains reasonable from a network point-of-view, such as for the purpose of VoIP communications [108], with readings below 4.405 ms for network load below 20% and 10%, respectively. An important aspect of the dynamic approach to the measurments is that, after these loadings, the latency across the DUT increases sharply; for example, the latency increases from 3.2ms at 20% of network load to 72.485 ms at 30% of network load for Linux0003 (Table A.28). Such a large increase occurs as soon as the

network traffic is introduced for the model instances Linux0005 and Linux0007.

Device Oriented Metrics The data provided in Section 6.2.2 highlight that the DUT does not have much room for maneuver in terms of CPU usage, as, even for the baseline, the readings are high (Table A.23). Nevertheless, it is possible to establish how increasingly large rule-sets affect the monitoring capabilities of the DUT. Table A.25, Table A.27, Table A.30, and Table A.33 show that the larger the rule-set the smaller the window-of-credibility becomes, for all four metrics.

Both the CPU usage and the available memory measurements rely on SNMP (Section 4.5.4), and unlike with the Cisco DUT, the window-of-credibility for these two metrics is not consistent with each other. Arguably, this has implications in terms of network management as monitoring accuracy findings cannot be used for other possible SNMP objects.

Along with this, results in Section 6.2.2 suggested that the absence of the filtering direction option improved, significantly, the memory usage as the model instance Linux0003 consumes less of this resource. Table A.26, however, shows that model instances that rely on the Netfilter firewalling engine to decide of the filtering direction, such as as Linux0003 (Table 6.3), can either consume more memory space, such as for Linux0013 and Linux0007, or free some of this resource, such as for Linux0005. More memory consumption is understandable as the rule-set is typically stored in a memory space that is quickly accessible [139], however, this does not explain why an evaluation instance that enforces a rule-set of 500 items requires more memory than an evaluation instance that enforces 4000 items. Arguably, the available memory to the processor metric raises issues regarding credibility, and accuracy, as well as the possibility which it is affected by parameters that are not part of the proposed model for the dynamic performance of network firewalls (Section 4.4).

Error-rate In most cases the latency readings are relatively high (Table A.28), however it is note worthy to underline that the error-rate never reaches unity (Figure A.12). This makes the DUT accountable even in difficult conditions, such as the combination of large rule-set and high network load. From a network operation point-of-view, it is important to know if such a high latency is still suitable. Indeed, the fact that the DUT can transmit packets under heavy burden is a key information as long as the delays do not prevent network applications, or protocols, to operate properly. Such expert knowledge is valuable for the purpose of monitoring and mitigating of threats.

Table A.33 shows that the accountability seen latency measurements extends to the measurements of the available network throughput. Table A.32 shows the footprints that increasingly large rule-sets have on the DUT ability to transmit bulk data (Section 4.5.4). With the Cisco DUT (Section 6.2.1), available network throughput readings typically need to be assessed by taking into account the error-rate. Table

A.32 shows that, despite the severe reduction in capability, the measurements are still possible, and arguably this fact correlates the observation made for the latency metric.

6.3 Scenario 2: Performance Evaluation of Critical Rule Positioning

Section 6.2 puts a strong emphasis on measuring the full extent of the effects on the dynamic performance of network firewalls of increasingly large rule-sets. One of the key findings is that the number of firewall statements a network device can enforce is finite, and this influences the type of security policy that can be deployed. Arguably, one of the options to ensure the dynamic performance of the device is to re-organise the statements within the rule-sets [143, 142]. Along with this, organisations are likely to use network firewalls as a connection point of several sub-networks (Figure 6.2), thus it is relevant to establish if there are any benefits in re-organising the statements.

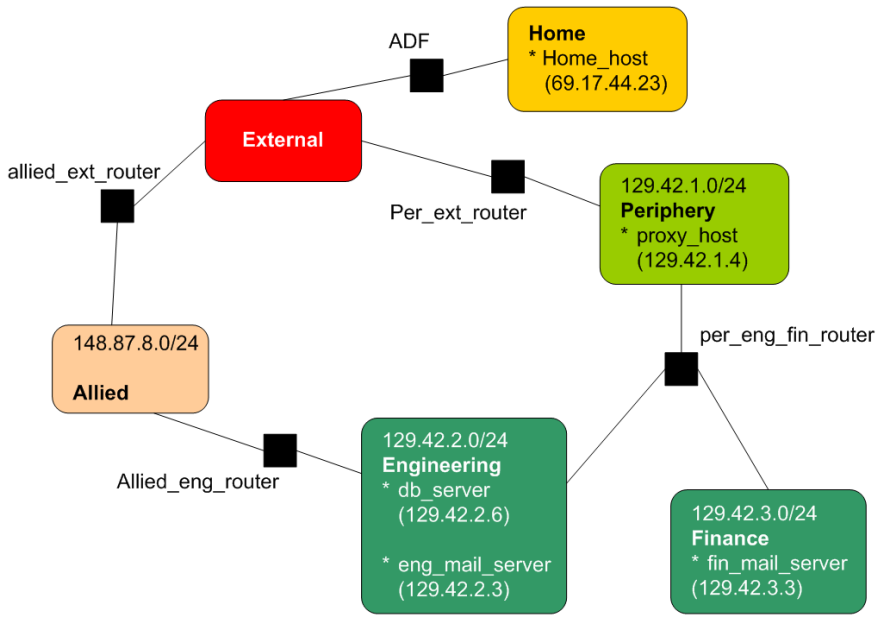

Figure 6.2 – Firewalls connection points of multiple sub-networks [52]. The rules that filter the traffic between the Engineering and Finance departments are unlikely to be in the same position as the rules that filter the traffic between the Engineering department and the Periphery network domain.

This second scenario uses DEENS to build model instances of a defined rule-set size and compare them with model instances which enforce twice as many firewall statements, however for which critical rule is placed in the middle (Section 4.4.1). For the Linux model instances and the Cisco model instances that utilise the outgoing filtering implementations, the results demonstrate that a major factor in the dynamic performance is the number of statements that must be examined before a firewalling

direction is reached, as each pair of model instances analysed incur similar performance overhead. Interestingly, there is a performance gain for network-oriented metrics in using large rule-sets for which the critical rule is repositioned for Cisco model instances that rely on the incoming filtering implementation (Section 4.4.1).

6.3.1 Effects on a Cisco Device

This part of the scenario (Section 5.2.3) utilises the Cisco DUT (Appendix C), and focuses on the performance difference between deploying a rule-set of 16000 items with the critical rule at the bottom and deploying a rule-set of 32000 items with the critical in the middle of the rule-set. In other words, this scenario establishes whether the dynamic performance of a network firewall is solely affected by the actual size of the rule-set, or by the number of items the device has to examine before reaching a firewalling decision (Section 3.6.8). Table 6.4 lists the selected model instances that match Definition 5 Equation 5.20, and provides their respective implementation details.

Table 6.4 – List of Model Instances selected for this analysis.

Model instance ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Cisco0033	100	CiscoInt1	in	3	bottom	16000
Cisco0036	100	CiscoInt2	out	3	bottom	16000
Cisco0125	100	CiscoInt1	in	3	middle	32000
Cisco0128	100	CiscoInt2	out	3	middle	32000

Analysis

Latency The readings obtained for the network-oriented metrics show a benefit in the repositioning of the critical rule, especially for Cisco0033 and Cisco0125, the two model instances that employ incoming direction filtering. Table A.38 shows that the overhead Cisco0033 and Cisco0125 incur, respectively, for the latency across the DUT reduces from 81.7% to 4.971% (Figure A.15). Along with this, the maximum latency for Cisco0125 is 185.247 ms compared to Cisco0033 with 197.385 ms. Results for Cisco0036 and Cisco0128 demonstrate that the number of rules the network device must examine is the major factor, as the difference between the two model instance is not as pronounced. Indeed, for a network load of 20%, Cisco0036 incurs an overhead of 40.069% and Cisco0128 at 43.095%. Hence, the model instances Cisco0036 and Cisco0128 can be considered similar.

Available network throughput Figure A.16 shows that Cisco0036 and Cisco0128 generate a footprint of respectively 25% and 29% at 10% of network load. This is the only stage of the evaluation where the readings between these model instances differ significantly. For the remainder of the evaluation, the readings correlate the findings

from the latency, in terms of the model instances being similar. Table A.41 further confirms the findings from the latency readings when it comes to Cisco00033 and Cisco00125. Indeed, the fact that the critical rule is located half way through the rule-set in Cisco00125 produces a lower footprint than Cisco00033. Cisco00125 represents a gain in throughput of 3.5 Mbps at 0% of network load, and 2.77 Mbps at 30% network (Table A.40). For the purpose of creating a formal model for the dynamic performance of network firewall, this offset could be built-in. The distinction between the two model instances is more challenging for higher network loads, and this due to the fact that, at this stage, the capabilites of the DUT to handle bulk data transfer is already limited (*cf.* baseline, Table A.40).

CPU Usage As established in Section 6.2.1, the readings for the CPU usage obtained when there is no background traffic are influenced greatly by the aftermath of the DUT reconfiguration (Section 4.5.2). Thus, these readings are not taken into account in this section. The focus is to establish whether pairs of model instances which have the same number of statements to examine incur the same performance overhead despite the fact that the overall rule-set sizes are different (Table 6.4).

For the pair Cisco00033 (critical rule at the bottom) and Cisco00125 (critical rule in the middle) (Table 6.4), the results suggest that the reposition of the critical rule has benefit for both the actual CPU usage readings as well as the window-of-credibility, such as seen for 10% and 30% of network load in Table A.35 and Table A.34. For example, at 20% of network load the overhead for the CPU lowers from 13.72% (Cisco00033, Table A.34) to 2.47% with Cisco00125, and there are no missing measurements when the network load reaches 30%. The advantages of repositioning the criticial also applies to model instances that use the outgoing filtering configuration as Cisco00128 improves the CPU usage by 14% instead of 5% for Cisco00036 (Table A.34).

Processor Memory Arguably, these improvements are also visible in terms of memory usage. Indeed, Cisco00033 and Cisco00036 have a combined average footprint of 3.46%, whereas the same combined average for Cisco00125 and Cisco00128 is only 0.77% (Table A.36). Along with this, the window of accuracy for incoming direction filtering is also improved (*cf.* Cisco00125, Table A.37). For example, Cisco00125 has an error-rate of 0.4 at 40% of network load, whereas Cisco00033 has an error-rate of 0.9. In addition, missing measurements occur from 40% of network load for Cisco00125, compared to 30% for Cisco00033.

6.3.2 Effects on a Linux Device

This part of the evaluation scenario benefits from the Section 6.2.2's findings as, unlike in Section 6.3.1, there is no longer the need to take into consideration direction

filtering parameters since there is no dynamic performance benefit. Moreover, Section 6.2.3 showed that the deployment of firewall rule-sets severely hinders the Linux DUT dynamic performance. Thus, any similarities between the model instances discussed in this section (Table 6.5) will provide evidence of the benefit of repositioning the critical rule within the Linux Netfilter rule-set.

Table 6.5 – List of Model Instances selected for this analysis.

Model instance ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Linux0003	100	n/a	n/a	3	bottom	1000
Linux0211	100	n/a	n/a	3	middle	2000

Analysis

The positioning of the critical rule does not significantly improve the device-oriented metrics (Section 4.4.3) window of credibility compared to the results previously shown in Table A.25, or Table A.27 for instance. Along with this, Table A.42 further demonstrates that the available memory to the processor does not have dynamic properties (Section 6.2.2. Indeed, dynamic measurements hardly differ from the values obtained when there is no background traffic.

Both model instances have similar network performance, though (Table A.44, and Table A.45) which highlight the benefit of repositioning the critical rule within large firewall rule-sets. It is noteworthy to underline that, for readings collected for up to 40% of network load, Linux0003 offers slightly improved dynamic network performance than Linux0211. Indeed, Linux0003 incurs less overhead in terms of latency (Table A.44, Table A.43). For example, at 20% of network load the latency across the DUT is 72.485 ms for Linux0003 and 76.154 ms for Linux0211. Arguably, there is also slightly more available network throughput (Table A.46, and Table A.45).

The rule-set for Linux0003 and Linux02111 are of 1000 items and 2000 items in size, respectively, and they examine the same number of rules before reaching a filtering decision as Linux02111 critical rule is placed in the middle (Table 6.5). Thus, the difference in terms of dynamic performance is linked to the total number of rules currently deployed onto the DUT. In other words, the size of the rule-set needs to be taken into account even if the critical rule is placed nearer the top of a rule-set. Hence, in terms of considering the deployment of security policies onto the Linux Netfilter firewall, it is important to keep a balance between the critical rule location and the overall size of the rule-set. This difference can be explained by the fact that the model instance Linux0211 has a larger rule-set (Table 6.5) stored into memory. As the critical rule is located in the middle of it, the DUT has to both interrupt the firewall operations as well as discard or manage the remain part of the rule-set, and this is likely to result more computations. For Linux0003 on the other hand, triggering the

critical rule is the last firewalling operation, and less memory management operations need to take place afterwards.

6.4 Scenario 3: Performance Evaluation of Fine-grained Filtering on Stateless Firewalls

The previous two scenarios employed rule-sets composed solely of simple firewalling statements (Section 2.5, Section 4.4.1, Section 5.2.4), and thus the evaluation outcomes are not directly applicable for the purpose of assessing the feasibility of security policies that exercise fine-grained control over multiple network protocols, applications, or services (Section 2.5, Section 3.6). Section 6.2 and Section 6.3 demonstrated that rule-set size play an important role in the dynamic performance of network firewalls. Section 6.3 showed that the critical rule (Section 4.4.1) can be relocated within the rule-set to maintain dynamic performance. Moreover, it is possible to reduce rule-set size by merging firewall rules together [142], and this, typically, involves taking several simple rules and merging them into fewer rules that are more complex (Section 5.2.4, Listing 5.2 and Listing 5.3). Hence, the focus of this third scenario is to determine the difference in terms of dynamic performance between evaluation instances that employ simple firewall rule-sets, and those that employ complex statements. This expert knowledge can help establish if there is dynamic performance gain in using multiple simple statements, or fewer more complex ones, such as after applying Hamed and Al-Shaer [142]'s algorithm to the security policy.

For security policies that translate into rule-sets of above 500 items and above, the results thus far (Section 6.2.3) indicate that the Cisco DUT offers better dynamic performance than its Linux Netfilter counterpart. The results shown in this section demonstrate that the complexity of the firewall statements plays a more important role in the dynamic performance of the Cisco DUT than it does for the Linux DUT. For example, the previous scenarios only focussed on model instances that operate on fast network settings (Section 4.4.1) and, as mentioned in Section 6.2.1, for model instances that represent slow network settings, there is typically an insignificant dynamic performance impact. Model instances based on slow network settings that employ complex firewall rule-sets do incur an overhead, and unoptimised configuration, such as using the incoming filtering direction instead of the outgoing one [106, 107], severely degrades dynamic performance. For the Linux DUT on the other hand, the dynamic performance is more closely linked to the number of statements to examine during the firewalling operations. In other words, it is feasible to employ complex firewall statements at fast network settings with the Linux DUT whereas this type of firewall rule-set significantly reduces the available network throughput for the Cisco DUT, for example.

6.4.1 Effects on a Cisco Device

This part of the scenario utilises the Cisco DUT (Appendix C), and focuses on the performance difference between deploying a rule-set of 1000 simple firewalling statements and deploying a rule-set of the same size that employ complex firewalling statements instead (Section 5.2.4). This should establish the benefit in terms of dynamic performance of applying algorithms, such as of Hamed and Al-Shaer [142]'s, to security policies, as these typically result in fewer but more complex firewalling statements (Section 3.6).

An important aspect is that, in previous scenarios (Section 6.2, and Section 6.3) the rule-sets did not have an impact on dynamic performance for slow network throughput. However, this scenario shows that the complexity of the firewalling statements does have a significant impact on dynamic performance even for model instances based on slow network settings. Hence, this section investigates the dynamic performance for the Cisco DUT for all the network speeds the device supports. Thus, Table 6.6 lists the selected model instances that match Definition 6 (Equation 5.23), and provides their respective implementation details.

Table 6.6 – List of Model Instances selected for this analysis.

Model instance ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Cisco1001	100	CiscoInt1	in	4	bottom	1000
Cisco1002	100	CiscoInt2	out	4	bottom	1000
Cisco00001	100	CiscoInt1	in	3	bottom	1000
Cisco00004	100	CiscoInt2	out	3	bottom	1000
Cisco00005	10	CiscoInt1	in	3	bottom	1000
Cisco00008	10	CiscoInt2	out	3	bottom	1000
Cisco1003	10	CiscoInt1	in	4	bottom	1000
Cisco1004	10	CiscoInt2	out	4	bottom	1000

Analysis

Available network throughput The measurements for this metrics provide a compelling example of the effect of the firewalling statement complexity on the DUT dynamic performance. Cisco1002 and Cisco00004 both use the outgoing filtering direction. Cisco00004 incurs a footprint on the available network throughput of 7.43% at 10% of network load. In contrast, Cisco1002 incurs a footprint of 89.45% for the same network conditions (Table A.49). With respect to the fact that most network applications are connection-oriented [32], the deployment of complex firewalling statements on the Cisco DUT can severely hinder network operations.

The almost consistent footprint Cisco1004 incurs for the latency also applies to the available network throughput (Figure A.19). Table A.53 indicates that the deployment of 1000 complex firewall statements has a footprint of 27.98%, on average. In addition,

Figure A.19 illustrates that rule-sets that use simple firewalling statements do not have a significant impact on the available network throughput.

Arguably, Table A.53 is further proof that incoming filtering direction drastically impacts performance. Indeed, at 30% of network load, the ability of the DUT to handle bulk data transfer is reduced by 89.88%. This is perhaps due network firewall's need to buffer packets before processing them [78]. This is expensive, in terms of computation, to move packets from the incoming buffer to the outgoing one. Kamara et al. [31] propose that it is preferable to apply rule-set to incoming packets, so that the device discards unwanted communications as soon as possible. The results presented in this chapter dispute the benefit of Kamara et al. [31]'s proposal. Moreover, Saliou et al. [78] suggested that, at slow network settings, the DUT would have enough time to service the incoming buffer, and Figure A.19 shows that this does not hold true when it comes to complex firewalling statements.

Latency Table A.48 further demonstrates the advantage of a dynamic evaluation over a static one, as the static readings (0% of network load) could not uncover the fact that DUT would cease to operate properly once deployed in the production environment (Section 4.4.2). Indeed, it is possible to obtain readings when there is no background network traffic, however once the traffic is introduced no measurements are possible. Along with this, for a network load of 0%, the latency overhead Cisco1001 and Cisco1002 incur is similar (Table A.48). This means that the DUT is functional at this point. Arguably, Cisco1002 highlights that the most advantageous filtering direction is the outgoing one (Section 4.4.1, and Section 6.2), as latency measurements are still possible.

For network loads between 0% and 20%, Cisco1002 has a more pronounced effect on the latency than Cisco0001 and Cisco0004. Indeed, Cisco1002 latency readings for these network loadings are nearly twice as high as the readings for the baseline (Table A.48). In Section 6.2.1, the results show that the latency typically increases significantly for network loads beyond 20%. Table A.48 correlates this observation, and shows that for network load above 20% Cisco0001, Cisco0004, and Cisco1002 produce a similar overhead.

The fact that Cisco1003 does not represent an optimal configuration for the deployment of complex statements is further illustrated in the latency measurements. Cisco1004 incurs an overhead of 109.8% on average (Table A.52), whereas, Cisco1003 incurs a far greater overhead, such as of 1135.891% at 10% of network load. In addition, this overhead increases futher with higher network load.

CPU Usage The impact of firewall statements complexity is also visible in model instances based on slow network settings. CPU usage is typically low for model instances Cisco0005, Cisco0008, and Cisco1004, which corresponds to the findings in Section 6.2.1 relating to the fact that the CPU usage is linked to the amount of data

transiting across the DUT. In addition, the CPU usage reading will become unusually high in that regard if the DUT's configuration is not optimal. Hence, Table A.50 shows that Cisco1003 is not an optimal configuration as for this evaluation instance as the CPU usage reaches a maximum of 80% for 40% of network load. Moreover, Table A.51 shows that the Cisco1003 CPU usage readings has a low accuracy, and this could explain the sharp rise as well as fall, after 40% of network load, of the measurement values. This contrasts with Cisco1002 which does not report any missing measurements.

Error-rate The combination of complex statements in rule-sets and fast network settings is highly detrimental to the dynamic performance of the Cisco DUT. This is particularly true for model instances that employ incoming direction filtering. For example, Table A.47 shows that, initially, the CPU usage measurements are possible, however the window-of-credibility diminishes rapidly for Cisco1001. Cisco1002, on the other hand, has an error-rate *in par* with model instances that employ simple firewalling statements.

6.4.2 Effects on a Linux Device

This part of the scenario utilises the Linux DUT (Appendix C), and focuses on the difference between deploying rule-sets composed with simple firewalling statements, and deploying rule-sets composed of complex firewalling statements (Section 5.2.4). Unlike with the Cisco DUT (Section 6.4.1), the use of complex firewalling statements does not have a significant impact of Linux model instance based on slow network settings, and thus this section only discusses evaluation instances based on fast network settings. Table 6.7 lists the selected model instances that match Definition 6 (Equation 5.24), and provides their respective implementation details.

Table 6.7 – List of Model Instances selected for this analysis.

Model instance ID	Speed (Mbps)	Interface	Direction	Layer	Position	Size (Nb. of items)
Linux0013	100	n/a	n/a	3	bottom	500
Linux1011	100	n/a	n/a	4	bottom	500
Linux0003	100	n/a	n/a	3	bottom	1000
Linux1001	100	n/a	n/a	4	bottom	1000

Analysis

The two previous scenarios (Section 6.2, and Section 6.3) demonstrate that deploying firewall rule-sets onto the Linux DUT has a minimal impact on the CPU usage. In addition, these scenarios also highlight that the amount of memory available to the processor is influenced by elements that are not part of the proposed model for the dynamic performance of network firewalls (Section 4.4). Hence, the network-oriented

metrics (Section 4.4.3) are used to establish the effect of complex firewall statements on the dynamic performance of the Linux DUT.

The pairs Linux0013 and Linux1011, and Linux0003 and Linux1001 enforce rule-sets of 500 and 1000 items, respectively. Linux1011 and Linux1001 employ simple firewalling statements for their rule-set, whereas Linux0013 and Linux0003 do not (Table 6.7). Table A.54 shows that each pair, with respect to their rule-set size, has similar results. Figure A.20 illustrates also that the available network throughput is not effected by the complexity of the rule-sets. Therefore, the complexity of the statement that make up the rule-set is not as significant a factor as it is for the Cisco DUT (Section 6.4.1).

This can be explained by the fact that the Netfiler firewalling engine is purely a software application. It relies on the operating system to capture network packets and make then available for comparison against the content of a firewall rule-set [139]. Hence, it is likely that all the information contained in the network packet is made available to the firewalling process from the outset. The Cisco DUT, on the other hand, is more akin of a **N**etwork **P**rocessor. Thus it is likely that it processes the packet information on a per-OSI layer basis, and therefore it only analyses the packet's content, if required. Arguably, it is this approach to the firewalling task that allows the Cisco DUT to enforce rule-sets of up to 64000 simple statements (Section 6.2.1).

6.5 Findings

This section summaries the main findings of the three scenarios analysed in this chapter, and shows the benefit of using the formalisations defined in Chapter 5.

6.5.1 Scenario 1

This scenario focuses on the effect of increasingly large rule-sets on the dynamic performance of the two DUTs (Appendix C), and it is divided into three distinct parts to accommodate for the technical specificities of each DUT.

The first part of this scenario (Section 6.2.1) utilised the Cisco DUT, and it established that increasingly large rule-sets typically reduce the window-of-credibility of the device-oriented metrics (Section 4.4.3). The impact was also more pronounced when the incoming filtering direction was used. Results suggested that the CPU usage is not suited to measure the effects of increasingly large rule-sets. Nevertheless, the CPU usage will be unusually high for configurations that are not optimal for the device, such as seen in the readings for Cisco0049 across all the metrics (Section 6.2.1). This information can thus help distinguish between normal operations, and attacks that seeks to overload the CPU [180].

A notable observation which emerged from this part of the scenario is that the available memory did not display any dynamic characteristic, and this suggest that

the DUT does not adjust this resource to cope with high network load, for example. Despite the lack of dynamic characteristics, the available memory metric allows for establishing the memory requirements of rule-sets, and this information can be linked to other parameters that consume memory on the DUT.

The results also highlighted a discrepancy between the window-of-credibility of the device-oriented (Section 4.4.3), and network-oriented metrics (Section 4.4.3). At times, the device oriented metrics suggest that a resource is at its full capacity, such as with the CPU usage, or readings impossible to obtain. Yet, for the same configuration and network conditions, network-oriented measurement will still be possible. This suggests that, to achieve this, the Cisco DUT prioritises network tasks, such as routing network packets and firewalling, over administrative and management tasks. This is an important piece of information as it could influence the type of mitigation that can be employed to thwart threats. In addition, these findings correspond to Campbell et al. [19]'s arguments who argue that this type of device is meant to be configured and deployed onto production networks and then left *as is* for as long as possible.

The most noticeable impacts are on the available network throughput as each rule-set size introduces a distinct footprint. This contrasts with latency readings where the effect of increasingly large rule-sets is more pronounced once the measurements are carried out dynamically. Along with this, both the latency and the available network throughput measurements demonstrate that there is a benefit in using the outgoing filtering direction instead of the incoming one. More importantly the dynamic evaluation demonstrate that this benefit extend also to the increased number of rules the DUT can then enforce. This information only comes to light through the dynamic evaluation approach that DEENS implements.

The second part of this scenario scrutinised the usage of the direction filtering option in Netfilter [133] in terms of dynamic performance. The three possible implementations, or instance of the network firewall dynamic performance model (Section 4.4), for a given rule-set size is not distinguishable when it comes to the CPU usage or the available network throughput, in which case all three model instances incur a similar footprint on the resources concerned (Section 6.2.2). This finding led to the rest of the scenarios to use model instances of the Linux DUT that rely on the firewalling engine to decide on the filtering direction. Arguably, the evaluation reported in this chapter cannot disprove the claim that removing the direction filtering improves performance [133], however results do not show a significant negative impact.

The two parts of the first scenario dedicated to the Linux DUT (Section 6.2.2, and Section 6.2.3) provides strong evidence that the deployment of firewall rule-set on the Linux DUT does not have a strong impact on the CPU usage, as this resource is heavily utilised from the outset. Arguably, the results for this metric correlates the findings of Accardi et al. [139] who noted that when using a Linux Netfilter firewall, the CPU spends most of its cycles dealing with transmission of data between the different memory buses.

The Linux DUT typically offers adequate accountability when it comes to the measurements of network-oriented metrics, in other words, the window of accuracy does not diminishes too much because of the security requirements. This attribute needs to be contrasted with the fact that the Linux DUT cannot enforce as many items as its Cisco counterpart. The reduction in the available network throughput is sharp though, as illustrated with a footprint of 37.29% when a rule-set of 500 items is deployed (Table A.32). A similar footprint on the Cisco DUT relates to a rule-set of 64000 items (Table A.10). In addition, at high network loads, it is likely that many network applications will not operate properly, as the latency accross the DUT is too high (Section 6.2.3).

6.5.2 Scenario 2

The second scenario (Section 5.2.3, and Section 6.3) focused on establishing the performance benefits or drawbacks of changing the position of the critical rule within the rule-set (section 4.4.1). The Linux DUT showed the strongest evidence that the number of *inspected* items was the major factor (Section 6.2.3, Section 6.3.2). This observation also applies to evaluation instances for the Cisco DUT that employ the outgoing filtering direction, whereas, for the evaluation instances that use the incoming filtering direction, results showed a performance gain in changing the position of the critical rule, especially for the network metrics.

Hence, the results demonstrate that the number of rules a device has to examine for the purpose of filtering network traffic between two locations of the computer networks (Figure 6.2), instead of dynamic performance being solely linked to the total number of statements contained in the firewall rule-set. Arguably, the critical rule position parameter can be associated to the priority an organisation gives to part of its computer network infrastructure, such as in deciding which department, or network segment (Section D.3.1), requires the most network throughput capabilities (Section 5.2.3).

6.5.3 Scenario 3

The third scenario focuses on determining the difference in terms of dynamic performance between model instances that employ simple firewall statements rule-sets, and those that employ complex statements. In the first part of this scenario, results further showed that the choice of filtering direction is an important parameter in terms of dynamic performance. Indeed, the latency readings (Table A.48) indicate that both Cisco1001 and Cisco1002 initially incur a similar overhead. This suggests that once the rule-set is deployed in the incoming filtering direction, the device can fulfil its functions. However, Table A.48 shows that once the background traffic is introduced,

all measurements fail, indicating that the device is no longer operational. This information only comes to light through the dynamic evaluation approach DEENS implements. Arguably, the most compelling evidence is found in the available network throughput measurements. For example, the deployment of a rule-set composed of 1000 simple firewalling statements incurs an initial footprint of 4.78% (Table A.49), whereas the deployment of a rule-set composed of 1000 complex statemetn incurs an initial overhead of 88.07% (Table A.49). In the meantime, a rule-set of 1000 items incurs a footprint of 56.74% for the Linux DUT (Figure A.20).

6.6 Conclusion

The results from the three scenarios show that dynamic performance can be severely affected depending on the type of firewall equipment employed to protect the network, and the manner in which that firewall is configured. Thus, there is an balance between: the desired security; the device chosen to enforce it; and, the environment in which the device will operate.

This chapter demonstrated that, with the use of DEENS to build unique instances of the network firewall model (Section 4.4), it is possible to measure the impact of security policies on the dynamic performance of network firewalls. The expert knowledge obtained is invaluable in terms of management as it allows for planning for this impact, by comparing model instance results against the security policy requirements, or mitigate it, such as changing the target network firewall platform, or its configuration. Indeed, the first Scenario (Section 5.2.1, and Section 6.2) demonstrate the impact of increased security requirements on network firewall dynamic performance. Along with this, the second scenario (Section 5.2.3, and Section 6.3) showed that the impact of increasingly large rule-set can be mitigated by changing the position of the critical rule within the rule-set. The third scenario (Section 5.2.4, Section 6.4) completed these findings with the fact that the level of control which the security policies exercise on network applications, protocols, and services plays a crucial role in terms dynamic performance when using the Cisco device.

The scenarios illustrated that a deep understanding of the device on which the security policies are going to be deployed on is important within an enhanced security process. For example, a device that offers excellent network performance, such as with the Linux DUT (Appendix C), may have its dynamic performance severely hindered once the firewalling functions are active. This needs to be contrasted with the fact that there is no need to take into consideration problematic the filtering direction (Section 3.6.4, and Section 5.2.2), and this lowers the complexity of administration and management of the device (Section 3.6.4).

Interestingly, for the Cisco DUT (Section 6.2.1), the outgoing filtering offers the possibility to deploy larger rule-sets than with the incoming filtering configuration, and this with a manageable performance overhead, in particular for high network

load. Indeed, Section 6.2.1 showed that the model instance Cisco0036 that enforces a rule-set of 16000 items has a footprint of 37.9% on the available network throughput at 30% of network load, whereas the model instance Cisco0017 that enforces a rule-set of 4000 items has a footprint of 44.94% for the same network conditions (Table A.10). Most importantly, this information is only identifiable with the dynamic evaluation approach DEENS implements.

This chapter demonstrated that the impact on dynamic performance of changing security requirements is best measured using a dynamic approach. Indeed, it is not always possible to distinguish between various configurations evaluated by simply measuring the difference between *pre* and *post* configuration performance. The dynamic approach to the evaluations allows a better understanding the impact of security policies on the network firewall resilience.

Performance evaluation and measurements only trigger a limited number of rules within the rule-sets. Results show that re-arranging the rule-sets lowers the impact of the rule-set size on dynamic performance. Hence, the second scenario (Section 6.3) demonstrated that performance issues must not be assessed from the perspective of the number of rules enforced by the network firewall device, alone. This artefact can be related to the priority organisations want to assign to their individual assets. In other words, organisations need to establish priorities in terms of their performance requirements, and only then the data provided by the dynamic performance model instances will be conclusive. Arguably, the situations for which dynamic performance data is not needed are when static measurements do not satisfy the organisation's requirements, as this typically means that the chosen network device will not perform as required once deployed in the production environment, such as Section 6.4.1 illustrated with the model instance Cisco1001.

This chapter presented results for both simple and complex firewalling statements. The detailed evaluations demonstrated a detrimental impact on performance for the Cisco DUT. This is even for model instances that employ outgoing direction filtering, which is a configuration choice that typically ensures improved dynamic performance as well as for model instance based on slow network settings (Section 6.2.1). Yet, this outcome would not apply to the Linux Netfilter DUT. Hence, a formal dynamic performance model of the Cisco device will have to include a complexity parameter, whereas a similar model for Linux Netfilter does not need to. Nevertheless, both models will include as a key factor the number of statements that need to be examined before a firewalling decision is made.

Finally, this chapter shows that using the outgoing filtering direction on the Cisco DUT allows enforcing large rule-sets, whereas significant performance overheads are the consequences of incorrect configuration. In that respect, the argument that the security policy will incur too much of an overhead is severely disputed, and hence this chapter shows the benefit of having dynamic performance models of network firewalls as part of the security process.

Chapter 7

Conclusions and Future Work

This research set out to develop a model that allows for the analysis of dynamic performance from multiple point-of-view (Section 1.2.1). Thus, this research investigated the dynamic performance of network firewalls because of their central role in computer network security (Section 3.10). The research aim is met with the formalisation of the inputs and outputs of the proposed model for network firewalls dynamic performance (Section 4.4, Section 5.3.1, and Section 5.3.2), which, in turn, allowed for the formalisation of key implementation scenarios (Section 5.3.3). These require dynamic performance data from several model instances (Section 4.4) to be assessed from multiple view-points in order to build a consensus on the impact of security policies on network firewalls.

Collecting the necessary dynamic performance data is a non-trivial and a lengthy process, thus this thesis proposes DEENS (Section 4.3) which is an automated facility capable of deploying the configurations to be evaluated on network firewalls, controlling the network conditions, and orchestrating the necessary measurements (Section 4.3). The design of DEENS follows Peisert and Bishop [24]'s recommendations in terms of network security related experiments, such as in having background traffic that provides a context to the results, changing only a limited set of input variables from one experiment to the next, and focussing on the repeatability of experiments. Along with this, DEENS supports the concept of an error-rate (Section 4.4.3) in order to address, for example, concerns of Roughan [168] and Saliou et al. [78] over measurements credibility. Indeed, the results discussed in Chapter 6 make it clear that the measurements are not always credible, and thus it is essential part in the decision-making process of an automated mitigation system, as the results for this metric can affect the type of mitigation that is employed (Section 6.6). Arguably, there may be some legal ramifications (Section 3.2.3), as new legislations require rigorous proof of network fitness prior to alleged incidents, if organisations want to benefit from improved legal cover [99]. Hence, DEENS and the proposed dynamic performance model for network firewalls (Section 4.4), address a key shortcoming in traditional computer network security: security policies are *thought* to diminish networked system capabilities. DEENS allows the **measurement** of the impacts of security policies and thus enable organisations to make informed decisions.

One of the main obstacles with dynamic performance analysis include the large amount of data on which the analysis is based. To address this, this thesis presents

d-Feral (Section 5.4), a software tool that takes advantage of the formalisations of both the network firewall dynamic performance model and the evaluation scenarios. d-Feral is then capable of identifying the model instances that are relevant to an analysis (Section 5.4.1); it then processes the data and presents it in an automated manner (Section 5.4.3). Arguably, the manner in which the data is collected (Section 4.5) favours presenting the results with a focus on the effect of the network load, however d-Feral, through matrix transposition, extends the scope of analysis as it can show, for example, the effect of network firewall configuration on dynamic performance (Section 5.4.7).

The data DEENS provides and the subsequent analysis of the three implementation scenarios with d-Feral resulted in several key findings. Chapter 6 demonstrated that the feasibility of security policies is linked to the device chosen to enforce them, the criteria these policies are meant to achieve, and the network environment in which the firewalls are deployed. Scenario 1 (Section 5.2.1, Section 5.3.3, and Section 6.2) focusses on the effects of increasingly large rule-sets on stateless firewalls, and established that the software based firewall could not support as many rules as its hardware counterpart (Section 6.2.3). However, the software firewall is easier to manage and administer as it is not always necessary to deal with the direction filtering (Section 3.6.4). Indeed, results (Section 6.2.2) show that allowing the software decide of the filtering direction does have a higher footprint in terms of on-board memory usage, however network performance is relatively unaffected. For the hardware firewall, on the other hand, choosing to implement the incoming direction filtering (Section 4.4.1, and Table 4.2) could severely limit the device resilience to high network load and the number of rules it can supported. Along with this, it is noteworthy to underline that this phenomenon can only be identified through the dynamic evaluations DEENS implements, as readings with no background traffic do not allow distinguishing between the filtering direction (Section 6.2.1).

Arguably, the first scenario allows for the planning of the worst-case scenario, as the critical rule (Section 4.4.1) was placed at the bottom of the rule-set, and thus the second scenario (Section 5.2.3, Section 5.3.3, and Section 6.3) investigated the effects of repositioning the critical rule within firewall rule-sets. Results show that the repositioning improves dynamic performance for both devices, especially for the software-based firewall, as it appears that the key factor is the number of rules it has to evaluate before reaching a filtering decision. For the hardware firewall, the total number of rules deployed plays a role. Another interpretation of this scenario is that its results can be related to the priority an organisation gives to parts of its infrastructure, or operations represented by, for instance, network segments.

The third scenario focussed on the effect of complex filtering on dynamic performance. With this, the results reinforce the fact that the main factor in the dynamic performance of the software-based firewall is the number of rules it must examine prior to reaching a decision, as the evaluations for simple and complex filtering are

similar (Section 6.4.2). More importantly, perhaps, this scenario highlighted that the hardware firewall does not cope well with complex rules, as dynamic performance is severely affected even for a small number of rules in comparison to the previous scenarios. Interestingly, this observation extends to evaluations carried out at slow network speeds (Section 6.4.1) which is not the case for the software-based firewall (Section 6.4.2). Hence, the third scenario shows that the software firewall is a better choice if the security policy translates into a rule-set of 500 items or less, or requires the deployment of complex rules.

Therefore, the results demonstrate that concerns in terms of requirements and performance have to be extended into the parameters they include. The extensive range of experimentations this research performed demonstrates the need to acquire detailed understanding of the devices chosen to enforce security policies. Indeed, security policies typically translate into extensive configurations and often large rule-sets for network firewall because of the rule based nature of network devices, however the results show that discarding a security policy based on its firewall rule-set implementation size is unwise (Section 6.5). Along with this, Schneier [171] argues that it is essential to understand system weaknesses since this plays an important part in the strategies of intruders [4]. Thus, the data that DEENS and d-Feral provide, give an **insight into the dynamic performance behaviour** of network firewall (Section 1.2.1), and hence complement works that focus on the logical properties of network firewalls [77, 76, 142]. Along with this, many models do not take into account that a networked system's mission might change over time [111, 113] and, thus, the type of threats associated. Hence, network firewall configuration will change and this change will result into different dynamic performance characteristics that need to be analysed so that it can be determined if these still meet the organisation requirements.

This research work has some implications in the literature. Indeed, Wool [107] stresses that outgoing filtering is hardly ever used in organisation and this is due to performance overhead. This is despite the fact that such security measure would lower the probability of an internal intrusion resulting in an attack on other organisations, and thus reduce legal liabilities [10, 3, 8, 47]. This research has demonstrated that indeed if administrators deploy the required rule-sets in the incoming direction (Section 3.6.4, and Section 4.4.1) and then it is likely that performance is severely hindered (Section 6.2.1). In this instance, systems such as of Caldwell et al. [14] or Yan et al. [86] could be employed however these are unlikely to highlight problems since the configuration of the device, although not optimal from a dynamic performance perspective, is logically sound. Where the evaluation environment (Section 4.3) and the analysis approach (Section 5.4) proposed in this thesis contribute is in the fact that organisation can identify the discrepancies between the anticipated dynamic performance impact and the measured impact once the device is deployed (Section 3.6.4). Thus, the network firewall dynamic performance model complements research works that focus on the logical properties of network firewalls [75], and thus permits better

informed decisions.

Furthermore, Wool [107] demonstrates that equipment upgrade can address key network firewall management issues, such as statement contradictions, however this requires the security process to be re-initiated once the device is chosen, or deployed. Wool [107] shows evidence that it is typically not the case. From the survey's findings, Wool [107] strongly advises to use a distributed network architecture (Section 3.6.5) for the network firewalls when the organisation's network is extended, as opposed to adding more firewalling statements to existing network devices, or modify their configuration settings. Unlike Ioannidis et al. [50], Wool's concern is not the single point-of-failure that could result in adopting a centralised architecture, it is simply a mean to force the security process into another iteration. Thus, DEENS (Section 4.3) and d-Feral (Section 5.4) can further inform on the outcomes of choosing a centralised architecture over a distributed one. Indeed, DEENS allows building multiple unique instances of the network firewall dynamic performance model which, in fact, represent configurations that can be employed, or that would results when changes are made (Section 4.4). Arguably, DEENS and d-Feral also enable organisations to avoid deploying new equipment unnecessarily, or not use devices to their full potential.

Another contribution of this research is that its dynamic evaluation environment (DEENS, Section 4.3) is capable of identifying network firewalls' failure conditions. This is particularly relevant in the mitigation of DoS and DDoS attacks, as some of these attacks consist of overloading the targeted network system [18, 26]. For example, Sommers et al. [149] stress that computer worms typically consume a large amount of network bandwidth, and thus the use of DEENS can thus help determining at which point, such as in terms of network utilisation, and security policy overhead, the network infrastructure becomes vulnerable [4]. This can result into the justification for the deployment of a network load balancing system, or new hardware.

This research work may change the manner in which security deployed is carried out for computer network systems. Administrators need to understand that security requirements extend beyond simply checking that a function is correctly added. They also have to understand the sustainability of their implementation choices; in other words, while the system is in use, thus in terms of dynamic performance.

Future Work

This work contains several areas suitable for improvements, such as with the analysis method, evaluation approach, and application of the collected data into a wider context.

The analysis of the scenarios rely on all the dynamic performance data to be available before any conclusion could be reached on the suitability of the configurations

evaluated. Arguably, this is similar to the manner in which administrators proceed. However, decision-making could be improved if d-Feral was able to identify evaluation instances based on dynamic performance criteria, such as ensuring that the network throughput is always at least 1 Mbps or more at 50% of network load, as well as configuration criteria. One method to achieve this would be to re-use the formalisation approach Chapter 5 outlines and thus derive a search formula, such as presented in Section 5.4.2. Nevertheless, this will likely require d-Feral to both process data as well as reach a conlusion, and this may be challenging depending on the amount of data to handle and it is likely to be time consuming. Hence, there would be limitations in using d-Feral as part of a system that mitigate threats in real-time.

At present, the impact of security policies on dynamic performance is assessed with respect to the configuration of the DUT that offers minimal functionalities (Section 4.5.1, and Section 5.4.6). One of the possible improvements includes the ability for d-Feral to compare model instances against one another instead. This can be achieved with the creation of a unified dynamic performance metric inspired by traditional routing metrics. This, in turn, will allow organisations to assess the different parameters according to the requirements, and thus establish the suitability of the network firewall configuration.

It will also be useful to incorporate topology information within the search and selection features d-Feral implements. Indeed, as it stands the user must know how the dynamic performance data relates to position of the network firewall within the computer network. Hence, it is possible that a search identifies network firewall configurations that in fact cannot be compared, such as for the filtering on one interface of the firewall in both directions (Figure 5.1, and Section 5.2.2).

One of the difficulties with the analysis for the results is that dynamic performance data and error-rate data are presented separately (Section 5.4.4, and Section 5.4.5). The analysis then requires both set of data to be understood and interpreted as a whole, thus a more meaningful way of representing the combined information needs to be identified. One solution would be to limit the output for the plots (Section 5.4.3) to a region that meets an error-rate criterion, such as being below 0.3 (Section 4.4.3, and Section 5.4.5). This also represents an opportunity to interpret the error-rate differently with fuzzy logic, such as in terms of a confidence factor, or stress-rate. This can then be extended with levels, such as:

- Between 0.0 and 0.2: the device is lightly loaded.
- Between 0.2 and 0.4: the device is lightly stressed.
- Between 0.4 and 0.6: the device is mildly stressed.
- Between 0.6 and 0.8: the device is near fail-over point.
- Between 0.8 and 1: the device is in a fail-over conditions.

Another area for improvement is the investigation of the dynamic performance of stateful firewalls. Indded, thus far, instances of the dynamic performance model are only based on network firewall devices that enforce stateless rule-sets (Section

2.5). Many network firewalls allow for stateful filtering, which keeps track of the state of the network connections. New connections are treated in the same fashion as with stateless filtering, however, the fact that a network packets belongs to a communication that is already established matters [77]. Hence, it is necessary to develop instances for such devices as well. It is noteworthy to underline that, to achieve this, the procedure DEENS follows to create instances of the model will have to be modified. Indeed, the Metric Collection Sequencer will have to keep track of the number of established connections at any one time, for example. The results suggest that, for both the hardware based firewall and the software-base ones, the on-board memory usage does not have any dynamic characteristics (Section 6.5). However, due to the fact that, for stateful firewalling, devices have to keep track of communication status, such as the number of connections currently established, it is thus likely that this metric is going to show dynamic characteristics. Arguably, this metric was challenging to collect in a reliable manner (Section 4.5.4), and thus this is an opportunity to investigate how this particular aspect can be addressed. One method would be to employ approaches that require full control over the DUT, such as in the work of Roedig and Schmitt [145], however it is important to underline that this would prevent dynamic performance data to be gathered once the DUT is deployed within the corporate network (Section 4.2.2).

Several publication have highlighted the challenges in terms of management when it comes to deploying SNMP (Section 3.7.3), and this research, indeed, faced some of these in the design and implementation of DEENS. In particular, the fact that the CPU and memory usage had their own specifications for each of the DUT (Section 4.5.4), and that limited the number of devices that could be evaluated. This can be addressed with the implementation of works such as with Rayan et al. [82] as it would allow for increasing the number of devices that can be evaluated by abstracting SNMP objects, and would also lower the management overhead (Section 3.7.3).

Thus far, the proposed model provides four metrics accompanied with error-rates (Section 4.4), and results show that most of the time CPU usage is not suited for the analysis of dynamic performance, hence the inclusion in DEENS of more relevant metrics needs to be investigated. Such metrics can include, for example, the number of packet errors, the number of times rules are triggered, or network jitter. One aim for such addition would be correlate the trend observed so far, as well as a mandate to limit the impact on network users activities. One key aspect would be to employ an adequate approach to measure the new metrics, as, network jitter for example, data can be collected from most firewalling devices, and many test tools allow it to be measured in-line.

Despite DEENS's automation, creating an instance of the dynamic firewall performance model is a lengthy procedure (Section 4.3.2, and Table 4.1), and this is an area can be improved upon. Indeed, the Metric Collection Sequencer (Section 4.3.2,

and Section 4.5.2) carries the entire evaluation procedure without analysing the readings as it receives them. One of drawback of this approach is that, when evaluation conditions exceed the DUT capabilities, measurements typically takes longer and the error-rate increases as well. For example, the latency measurements presented in Section 6.2.3 showed that these can be as high as 4000 ms, and this, in turn, indicate that this stage of the measurement procedure lasted more than 10 minutes. Hence, in order to avoid spending time on building instances such as Cisco1001 (Section 6.4.1), an improved approach would be to apply constraints on the Metric Collection Sequencer, such as cancelling the remaining stages of the evaluation whenever dynamic performance no longer meets the organisation's criteria. Alternatively, the error-rate can serve as a basis to determine if the next series of measurements is worth while, or to reduce the network traffic increment, accordingly.

Another weakness in the evaluation methodology is the fact that the experiments were carried out with an ever increasing number of rules, and an increasing network load (Section 4.5) in order to save time (Section 5.4.1). Hence, it is essential to establish whether the results obtained thus far are the same when evaluations are performed with an ever decreasing number of rules, or with a decreasing network load. Arguably, in order to further assess the repeatability of the experiments, and by the same token establish the accuracy of the reading obtained, the next step of the work would be to evaluate network firewalls with random input values and traffic conditions.

This research has collected far more data than what can be presented in this thesis, hence a possible use of this repository is the creation of a prediction system and compare the anticipated results with data obtained with DEENS. Success in this area can further reduce the time requirement involved in creating a formal model of network firewalls. Along with this, not all types of firewall devices can be thoroughly tested as there are an almost unlimited number of different infrastructure, however, with enough devices sampled key attributes, such as interface speed, firewalling engine, and so on, can be identified and, in turn, help determine a best fit, for an unknown device. This can lead to an ontology based [152] around firewall platform types, such as hardware firewalls, **N**etwork **P**rocessor, and software firewalls.

The results (Section 6.2.2) allow for concluding that the removal of direction option does not have much of an impact. The developers [133] argue that this design choice enhance performance. Arguably, the enhanced performance relates to the programming of the firewalling engine, and thus for the purpose of the scientific rigour, it would interesting to investigate if these changes affect the device dynamic performance. In addition, this could be the start of a repository for the benchmark of the various version of Netfilter.

During the evaluations the DUTs only enforced one rule-set whereas the Cisco device, for example, can actually support multiple rule-sets. Hence, it is necessary to extend the creation of dynamic performance model in that direction. In addition,

the evaluations only permitted for the rule-sets to employ one type of firewalling statements at a time (Section 4.5.3). Hence, an improved model for network firewall dynamic performance will allow for more configuration criteria.

The current performance evaluation only focuses on a single DUT, thus further work will also focus on investigating the dynamic performance properties and behaviour of distributed firewalls, such as to establish the performance impact of implementing rigorous inter-vLAN filtering that Weaver et al. [71] suggested. One method to achieve this would be to assess whether performance data from individual DUTs can be combined, or if the combination results in single point of enforcement [181] that has its own dynamic performance characteristics.

Denning [182] argues that for IT professionals it is challenging to keep up with innovations in the domain of computing, and often times the important aspect of these are uncovered during hands-on experience. Unfortunately, due to the possibly negative side effects of mis-configurations [14] or security testing [101], such expertise in the field of networking is difficult to obtain. Thus, the results presented in this thesis have their place in the training and education of network engineers. Indeed, Buchanan and Saliou [183] showed that the use of network device emulation packages allows network engineers to gain the confidence and the expertise necessary to operate devices proficiently. These emulators then allow constructing scenarios that familiarise users with possible network issues. This approach is reminiscent of what is done for airline pilots, for example. With this in mind, dynamic network firewall performance data can be incorporated into such emulators, and thus allow trainees to learn about the dynamic performance of network firewall as well as understand the impact of security mitigation solutions.

Finally, Chapter 3 analysed some technologies that could defend computer network systems, and accurately uncover malicious activities. It reviewed systems able to deploy security policy changes in a distributed manner, it is thus argued that it is possible to create pro-active security systems. Hence, an improved framework for security would incorporate this pro-active approach, along with the capability of integrating security issues, with attributes expected of the network. Moreover, this approach would empower designers, or decision-makers, to evaluate the impact of their choice on actual devices, and this without hindering the organisation's operations. As network firewalls represent just one component of the organisation security, a framework would eable organisations to take all relevant security factors into consideration. Thus, an avenue for future work would be create an **I**ntegrated **S**ecurity **F**ramework (ISF), and the next chapter outlines an ISF founded on these observations, which relies on a deep understanding of the repercussions of security policies on network devices. To that end, the proposed ISF uses metrics, such as proposed in Section 4.4.3, that serve to assess the feasibilities, or suitablitily, of a chosen security policy.

Chapter 8

Integrated Security Framework

8.1 Introduction

Chapter 3 showed that the multi-faced nature of computer network security results in a large array of solutions, and that organisations often do not take into account their strengths and weaknesses when addressing security issues. Chapter 6 illustrated the possible repercussions of such a lack of rigour when it comes to network firewall dynamic performance. Arguably, such an assessment is best carried out as part of a framework, however there are few frameworks that integrate fully computer network security with organisational aims and objectives. This is most likely due to the fact that creating such an **I**ntegrated **S**ecurity **F**ramework represents a major challenge.

Thus, this chapter highlights that one of the main shortcomings is that the strengths and weaknesses of security devices are not often known. Arguably, without such information it is not possible to ensure the feasibility of security policies within the means of the organisation. Hence, this chapter outlines a framework which builds upon and extends on the design Saliou et al. [21] propose. This framework has an emphasis on understanding the capabilities of the networked equipment that will enforce it, and which are deploying the security policy. It thus includes the facilities necessary to create that knowledge base. This **I**ntegrated **S**ecurity **F**ramework (ISF) is built around the concept of a formal approach that integrates key areas of expertise, hence promoting security as a process. This chapter also describes the ISF phases and the components these are made of.

8.2 Security Frameworks

A framework typically represents a method to address an issue as opposed to providing solutions. Thus, this section presents a review of frameworks that either focus on the management of computer network security or the technical implementation of security. As there are few security frameworks, it highlights some of the shortcomings that should be addressed as part of an **I**ntegrated **S**ecurity **F**ramework.

8.2.1 Management Frameworks

Management frameworks are suited in helping decision makers to take security on-board (Section 3.4.3), and this is shown, for example, in works such as of Eloff and Eloff [65] or Saleh et al. [34]. In both cases, the researchers aim to facilitate the process for organisations to meet standards, such as ISO-17799, and thus develop organisational security policies (Section 3.3). The ISO-17799 standard, for instance, provides an extensive list of security issues and threats that decision makers need to consider [3, 8]. One of the challenges includes that the standard suggests, for example, to secure e-mail communication, however does provide solutions for it, such as highlighting possible encryption facilities and their associated strengths or weaknesses.

Eloff and Eloff [65] propose that organisations start their compliance task by defining which type of organisation they are, such as academia or financial, and thereby identify key objectives and associated assets (Section 2.2). This allows organisation to better focus their efforts and maximise RoI. In other words, Eloff and Eloff [65]'s approach enhances the efficiency of the ISO-17799 standard. Arguably, Saleh et al. [34]'s **S**tandard, **T**echnology, **O**rganisations, **P**eople, and **E**nvironment (STOPE) approach is similar to Eloff and Eloff [65]'s work. However, there is a stronger emphasis on management personnel, including CEOs, vice-presidents, and IT administrators, as well as integrating regular training (Section 3.4). They provide an extensive list of recommendations, however many items in the lists solely require a Yes or No answer, and Baker and Wallace [55] showed that this approach seldom produces an accurate security assessment. One of the key differences with Eloff and Eloff [65] is the role of employees in security breaches. Like Woloch [47]'s proposal, it is suggested to outline in the security policy possible sanctions against users in the case of security breaches. STOPE, thought, does not involve end-users in the definition of the security policy, whereas Danchev [10] suggests that such an involvement would improve the security policy. He stresses that this could create situations that encourage security breaches [39], such as when policies are seen as an hindrance to work-flow [10]. More importantly, perhaps, Saleh et al. [34] highlight that detailed knowledge of existing systems and their capability is crucial to their framework, and they demonstrate the need for rigorous procedure enforcement, such as when it comes to users leaving the organisation, then all access rights should be removed from all relevant systems.

Decision makers have to define security solutions, and ensure that these are achievable without infringing laws [101]. Unfortunately, management frameworks seldom include mechanisms that can support organisations in integrating legal requirements in their security policies. This is a significant shortcoming as Kwecka et al. [96] show that, depending on the laws organisations need to comply with, the resulting implementation might create conflicts. Similarly, Furnell and Papadaki [101] point out that legislation might out-law some necessary security testing and evaluation methods. Indeed, many of the tools that intruders use are often similar to

those employed to identify computer system weaknesses [74], or to defend against the intrusions [101]. Alun et al. [8] note that such shortcomings often hinder the adoption of security standards in organisations.

Overall, management frameworks rely heavily on personnel to understand requirement, standards, legal requirements, outline solutions, and finally deploy and verify these. Hence, management frameworks are often slow to evolve and to benefit organisations [47]. Indeed, Woloch [47] stresses that bi-annual reviews are not suitable due to the fast evolving nature of computer threats. Along with this, these frameworks do not often have knowledge of the technical implications [179], and hence there is a need for approaches that relate to technical implementations.

8.2.2 Vendor Frameworks

Vendor frameworks harness the capabilities of security devices, such as for network firewalls or IDSs, however these frameworks cannot act as replacements for organisational security policies, and they often rely on these policies to provide the actual implementation [167].

The main advantage of vendor frameworks is that they ensure that objectives are feasible and that policies are not an hindrance. CheckPoint [184]'s framework, for example, operates in such a manner that the policy designer will not allow for the creation of policy statements that the target network equipment does not support. Similarly, Hinrichs [181] underlines that the Cisco Security Policy Manager will not permit the design and deployment of a filtering policy that disable network protocols utilised to manage network security devices. Indeed, some network protocols are viewed as not secure, however often they support key administrative tasks, such as remotely configuring routers with Telnet [181, 78]. Hence, management software that are part of vendor frameworks will bring these issues to the attention of designers.

Vendor frameworks will then typically highlight that security requirements cannot be met because these have knowledge of the target devices. Unfortunately, this can often translate in to a limited number of supported devices. For example, the Cisco Security Policy Manager [181] only works in combination with Cisco PIX firewalls, and for CheckPoint [184] the target deployment device must be compatible with the management software, such as in terms of supported operating systems. Cisco's **S**ecure **A** blueprint **F**or **E**nterprise networks (SAFE) [167] supports several type of devices from Cisco Systems and its partners. This framework thus allows for creating networks where the circumvention of one defence system does not necessarily result in a successful security breach. Hence, there are strong ties between these frameworks and the technical implementation. Nevertheless, multiple vendor frameworks may be required to achieve optimal security, and consequently organisations often need to manage the overall configuration, as well as possible overlaps in terms of functionality.

Often, policies are defined with functionalities that take precedence over security

[181, 14, 21], the Smart Defence framework by CheckPoint [184], on the other hand, distinguishes itself by focusing on the management of threats. Smart Defence [184] allows for selecting threats types, such as DDoS, and the framework will identify a suitable mitigation technique that will then be deployed to the network firewall [184]. A key aspect is that this framework does not permit organisations to design mitigation solutions, and hence CheckPoint [184] suffers from the same shortcomings as the Anti-virus vendors (Section 3.5.2) whereby the mitigation solutions can only be applied as soon as Checkpoint [184] can design and provide them.

The concept of infeasibility in vendor frameworks typically relate to functionalities, such as support for encryption not being present on chosen devices. These frameworks often do not often provide indications regarding the performance impact of policies on network security devices while these are in use within the production network; in other words, in terms of dynamic performance. Thus, it is up to organisations to determine the capabilities of their devices in this regard, and then establish, for example, whether there needs to be an equipment upgrade, or the deployment of network load-balancing.

8.3 Integrated Security Framework Overview

This thesis argues that the objectives of the organisation should dictate what resources are necessary to achieve these, along with the relevant constraints. Thus, the proposed ISF includes the legal requirements which are part of the organisation's guidelines for operation, and then included into security policies (Section 3.3). There is strong evidence that a key issue in computer network security is the discrepancies between the objectives and the implementation onto network devices (Section 3.9). Conversely, this extends to inconsistencies in configurations from one device to another. Arguably, this is a by-product of several shortcomings, such as managers, or decision makers, having to focus on the statutory legal requirements, however they may lack an understanding of the technical implementation and operation of computer network systems [111]. Similarly, the personnel in charge of deploying security policies are often in the opposite situation, where their comprehension of the law on these matters is limited and they also have to adjust, in a non-transparent manner [10, 66] (Section 3.3.2), the policies to the systems. This point also highlights the fact that organisations seldom know the extend of their computer network capabilities [34], which can lead to incorrect choices in terms of maintenance and upgrades [14, 45], and often contributes to the high up-keep associated with security [45, 44].

8.3.1 Design Principle

The ISF is composed of six phases which, together, allow for the accomplishment of both the implementation and verification of organisational security policies (Figure 8.1), in accordance with legislation. These phases typically work in pairs, as there is

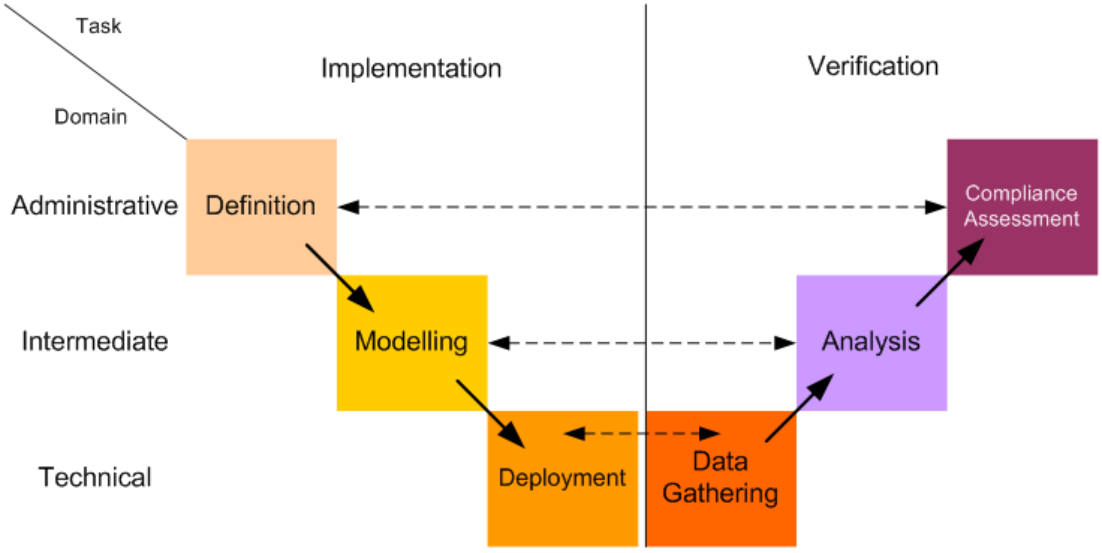

Figure 8.1 – Phases of the **I**ntegrated **S**ecurity **F**ramework without explicit feedback shown.

reciprocity in the data the phases process or require, and are distributed across three domains:

- **Administrative**: where the security policies are defined (Section 8.4), or the resulting implementation is assessed for compliance (Section 8.9).
- **Intermediate**: where the policies are modelled in preparation for deployment onto network devices (Section 8.5), or data collected from the computer network is analysed (Section 8.8).
- **Technical**: where the policies are deployed onto network devices (Section 8.6, or data is collected (Section 8.7).

The benefits of such a design include: consistency in terms of specification; early identification of possible limitations; and an increase in the use of best implementation practices. Hence, each phase has a precise goal, and this enhances the ISF flexibility and adaptability, as the procedure that is used to meet achieve a goal can be changed to incorporate a more suitable item. For example, for the purpose of consistent network filtering, Yuan et al. [76]'s system can be employed in order to benefit from both the identification of network firewall anomalies and the assessment of the impact on the overall network, as opposed to using Al-Shaer and Hamed [75]'s and Guttman [52]'s methodologies separately (Section 3.6.5).

8.3.2 Purpose

The ISF aims to empower decision makers, typically those legally responsible for the deployment of security in organisation [3, 8, 47], by including a feedback procedure that will highlight the security requirements which cannot be met because the necessary devices are not available. This will thus allow decision makers to assess

and refine security policies [111, 82, 87] and thus foster a more pro-active attitude to security.

Along with this, the ISF aims to limit the reliance on administrators [14] for the deployment and auditing of security. For this, the ISF uses agents for these tasks (Section 3.8), and this can support an automated mitigation system [21]. Such system can only be of benefit if they behave in the interest of the organisation, in other words requires strong policing (Section 3.8) that the ISF can provide [21].

The ISF relies on a deep understanding of the capabilities of the underlying devices, and this ensures that security policies are feasible. Conversely, this also permits the use of the framework in organisations with existing computer networks. Overall, this framework embodies an improved decision making approach as it addresses computer network security as a process rather than a technology.

Creating a complete security framework represent a major challenge, thus the remaining sections of this chapter outline the main functionalities of each of the phases, and highlight some of the research works, methodologies, or technologies that can be employed to realise them.

8.4 Definition Phase

The Definition Phase should include in a **Formal Security Policy Definition** (Figure 8.2) the legal and statutory requirements that the organisation needs to comply with its aims and its objectives. Such concepts are not often directly applicable to the configuration of computer network systems. One of the methods to address this complex task is to ensure that there is someone in charge of the gathering the information, as well as accountable for the subsequent deployment [11, 12] (Section 3.4.3). This approach is often not sustainable as it relies on the expertise of the personnel in charge [14, 8]. Thus, the ISF could rely on a **Template Repository**, like Guttman [52] suggests and advocated in the **P**ayment **C**ard **I**ndustry **D**ata **S**ecurity **S**tandard (PCI-DSS) standard [118, 185]. This provides decision makers with a starting point for the policy, ensuring that the policy is created from tried and tested elements, such as for legal requirements in terms of privacy preservation [96], and it also promotes best practices [52, 14].

It is unlikely that two organisations require the same security policy, and, along with this, the deployment of homogeneous solutions should be avoided as this could improve the efficiency of intrusions (Section 3.5.1). Hence, the Definition Phase allows for decision makers to tailor the templates in order for these to better match with the organisation's needs.

An important input for this phase is the definition of **Constraints** [68]. Indeed, it is unrealistic for an organisation to devote all of its resources (Section 2.2, and Section 3.4.2) solely to computer network security, nor should the security policy hinder computer network functionalities [21]. Hence, decision makers must define

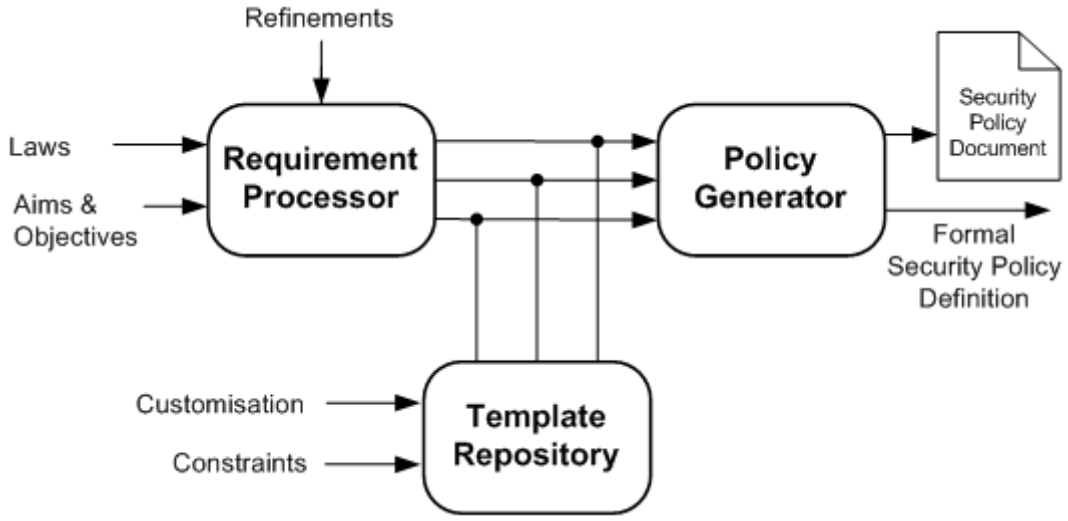

Figure 8.2 – ISF Definition phase.

the criteria against which the security policy is evaluated. This can include ensuring that the policy does not result in a significant increase in network latency [143], otherwise the quality of VoIP communications might be degraded [108]. Arguably, organisations have many aims and objectives and thus decision makers will have to define prioritises in terms of resources [116].

The **Policy Generator** should create the two outputs for this phase, which are a formal policy definition and a **Security Policy Document** (Figure 8.3). On the one hand, the security policy document is a human readable format so that it can be understood by all users, and thus lower the likelihood of breaches. Indeed, users are less likely to bypass security mechanisms if they understand why these are in-place [10, 104, 95]. On the other hand, the formal policy definition is achieved with the e**X**tensible **M**arkup **L**anguage (XML). This ensures the portability of the information contained in this form [178, 186], and compatibility with many modelling techniques, such as with Sirajuddin and Sqalli [187] for their distributed network management system.

Arguably, the fact that some requirements cannot be met might not be apparent until the deployment of the security policy onto security devices. Hence, the Definition Phase receives feedback from the Modelling Phase and the **Refinements** input (Figure 8.2) highlights the need for decision makers to appreciate, as well as understand, the risks, associated to their choice of security policy, and thus act accordingly, such as modifying it.

8.5 Modelling Phase

The main purpose of the Modelling Phase should be to transform the Formal Security Policy Definition into a series of network device configurations (Figure 8.3), and thus

it must establish the feasibility of the chosen policy. Hence, it relies on knowledge bases for the necessary technologies, protocols, or equipment, necessary to meet the security requirements, and the organisation structure which relates to the personnel hierarchy that defines users' roles and rights on workstations [188], for instance, as well as the operational structure, such as in the number of departments (Section 3.6.5), remote sites and so on [111]. In the event where some requirements cannot be met, such as the absence of a particular device, or unsatisfied performance demands, this information is brought to the attention of decision makers through the refinement feedback loop. This can include, for example, the conflict between a requirement that stipulates the need for network sharing with an allied organisation [9], and the security policy which bars the concerned units, or user groups, to make a network connection to remote resources.

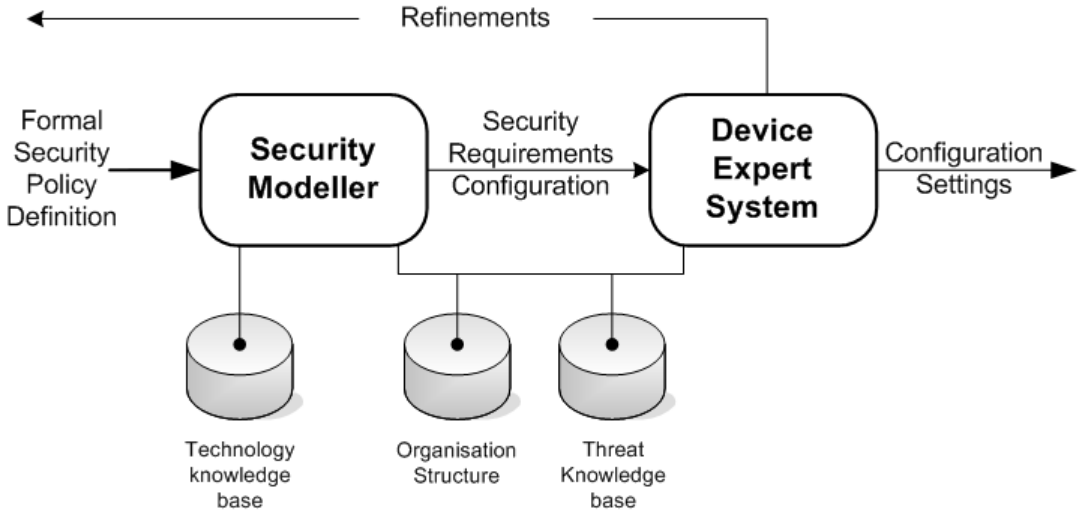

Figure 8.3 – ISF Modelling Phase

In order to accomplish those aims, the **Security Modeller** should check the limitations over functionality, or from hindering the completion of objectives, and that, overall, the combined inputs for the policy (Section 8.4) still achieve the aims and objectives of the organisation. Thus, the Security Modeller applies mathematical models to the Formal Definition. For example, Schneider [68] model defines that a given system has a set of functionalities, and that security policies represent, in fact, a set of constraints. Schneider [68]'s model evaluates the combination of functionalities and constraints against objectives, and thus determines the enforceability of a policy. Arguably, Guttman [52]'s approach accomplishes a similar outcome for distributed network firewalls. This procedure is likely to encounter feasibility issues depending on the size and scope of the policy, hence the Security Modeller should be capable of evaluating multiple deployment scenarios. Jones [179] argues that such an approach allows for the better understanding of an organisation capabilities and identification of possible risks. Indeed, it is noteworthy to highlight that one security

requirement typically results in several implementation requirements. For instance, anti-virus software is essential to thwart malware outbreak [58, 55] (Section 3.7.3), however, the efficiency of such technology depends on the signature databases being updated daily [58]. This, in turn, requires end-hosts and network firewalls to be configured accordingly to allow for updates. Auditing the level of coverage requires mobile, flexible, distributed and well policed data gathering agents, such as proposed by Buchanan et al. [36].

The Security Modeller should provide an XML document whose content is organised into key security categories, such as integrity, availability, confidentiality, and assurance [189], as well as management tasks, such as computer network system monitoring, data gathering and threat detection. Thus, the role of the **Device Expert System** (Figure 8.3) is to evaluate these requirements against the capabilities of the devices currently available in the production network, and, hence, this is where the deep understanding of the underlying networked systems intervene. In other words, the Device Expert System requires dynamic performance data, such as in the number of firewalling statements a particular network firewall equipment can enforce without consuming more bandwidth than specified in the requirements. Arguably, this type of information represents the balance that must be achieved between requirements, performance, and functionality [32]. Along with this, such data is paramount to establish the risks in terms of resilience against DDoS [78].

The **Device Expert System** outputs can use generic syntaxes for the specifications, such as of Yan et al. [86]'s for IDS, or Al-Shaer and Hamed [75]'s for network firewalls. Along with this, the relationships between network segments can be established in a similar fashion to Bertino et al. [120] definition (Section 3.3.3), so that they can be mapped to firewall filtering between vLANs, for instance. Such syntaxes allow starting from the same basis if the underlying equipments are changed, or upgraded. Along with this, syntactic conversion to platform specific syntax is easily achievable [14] and less error-prone than having human operators interpreting the security policies directly into device specific configuration scripts (Section 3.3.2).

8.6 Deployment Phase

The Deployment Phase should convert the generic syntaxes created with the Device Expert System (Section 8.5) into a format which can be implemented on computer network devices (Figure 8.4), such as for Cisco ACLs for network firewalls, or Snort rules for IDS [69, 70], as well as user rights on workstations [158, 190]. One of the drawbacks to overcome includes the fact that there are many equipment manufacturers which typically use their own syntax [14], and whose configuration procedures varies greatly. Thus, these details need to be known for the syntactic conversion to be feasible [78], and for configurations to remain consistent [14].

Administrators typically do not match security policies with devices capabilities

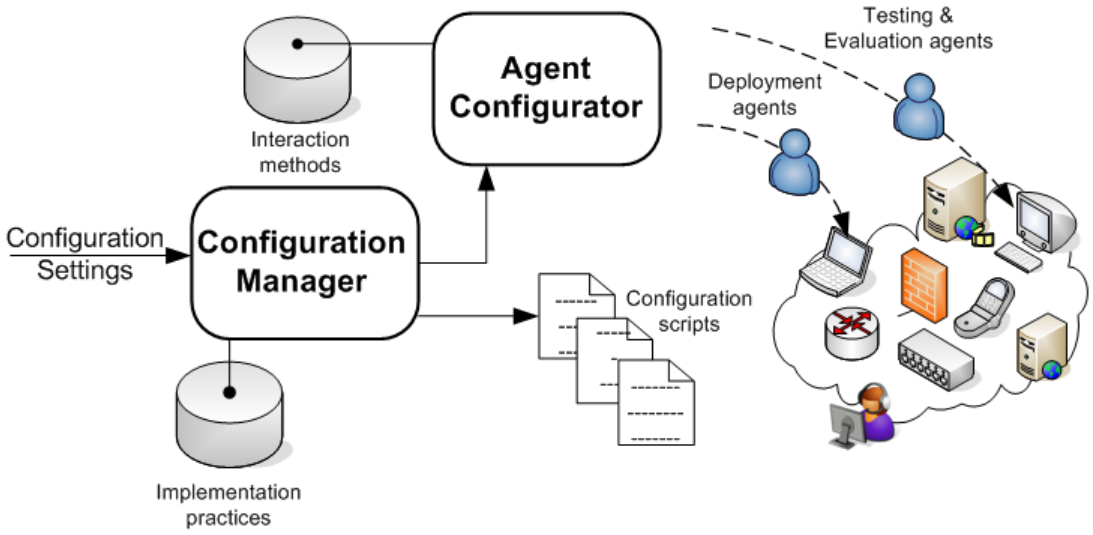

Figure 8.4 – ISF Deployment Phase

[107] (Section 3.6.4 and Section 3.6.7), thus a key aspect of the **Configuration Manager** (Figure 8.4) should be that it bases its outputs on best implementation practices that are extracted from the networked system knowledge base, which is updated as new equipment, or technologies, become available. Arguably, this procedure is the missing link Wool [107] identified, and whose absence results in mistakes being carried over from old devices to their replacements.

This phase of the framework should aim at limiting human interventions as these typically introduce inconsistencies into computer networks [14], which intruders then exploit [50]. Thus, the traditional manual configurations are replaced with a software agents system, such as AN or MAS (Section 3.8) and the **Agent Configurator** ensures that capabilities of the agents are restrained in accordance with the specifications obtained from the Modelling Phase. These agents then read-in the configuration scripts and deployed through standard application layer protocols, such as HTTP, or Telnet [78]. Indeed, Saliou et al. [21] highlight that one of the key challenges for such an agent system is the support for heterogeneous environments, and thus agents must use vendor-free deployment procedures.

Along with the creation of networked devices configuration, this phase should also produce verification and evaluation criteria for out-of-line and in-line monitoring [85]. The aim is to generate relevant data that can then be processed to assess compliance [47]. This can take the form of test scripts, as Paxson [119]'s did for his IDS. These tests should be created with an understanding of how the security policy is implemented, within its limits of operation, such as following scenarios or scripts often used by intruders or malware [21].

The functionalities of the Deployment Agents have can be re-utilised to thwart threats, such as with deploying pre-established configuration to combat known intrusions [21]. The main requirement is that these agents must be goal-oriented, so

that they can act quickly, to address the nature of most threats, as human intervention is typically too slow. Furthermore, Moore et al. [4] noted that most intrusions occur when human resources are limited, such as at night or at the week-end [21] (Section 3.5.2), hence it is paramount to offer mitigation capabilities. This could be achieved by basing the assessment of the overall computer network status on the data **Test and Evaluation Agents** produce (Figure 8.4), and feeding this data into a threat knowledge-base (Figure 8.3) in order for the Device Expert System to specify the adequate counter measures (Section 8.5). Thus, depending on the nature of the threat, agents could intervene, however they should be able to create device configurations that have limited impact on the overall system [78, 25] (Section 3.8).

8.7 Data Collection Phase

The aim of the Data Collection Phase should be to monitor the live computer network. This is achieved with the gathering of key security metrics, such as identifying the network services that are blocked or not, and measuring the available network throughput. This allows for the assessment of impact of the security policy deployment on network devices during system audit (Section 8.8, and Section 8.9). This phase can be viewed as an extension of the Deployment Phase, as it relies on the **Agent Configurator** (Figure 8.4) to ensure that agents uncover key threat characteristics, for example.

This phase is likely to produce a large amount of data (Section 3.7.1), thus a focus on relevance is needed as well as on the ability to obtain data from multiple sources. Thus, the agents must be able to carry out their tasks in a heterogeneous environment. Indeed, Porter [109] highlights that evidence of fraud, for example, is typically scattered across many devices, such as in servers, or work-stations logs [88, 127, 42]. This requirement thus has repercussions on devices configuration. Graves et al. [85] stress that network-based evidence is often used to correlate evidence obtained from end-hosts as such evidence is less likely to be tampered with. However, the configuration that needs to be deployed to permit the data collection can have a detrimental effect on performance [85]. Arguably, the device could miss evidence, such as when a network firewall drops packets because of congestion (Section 3.6.8). Thus, Graves et al. [85] recommends that network devices, such as network routers, should provide monitoring data about themselves, such as CPU usage, as opposed to the traffic they filter.

Evidently, collecting data from work-stations raises many issues, such as breach of privacy [36] and possible impact on user's activities (Section 3.5.2). Hence, Buchanan et al. [36] propose a design for a forensic mobile agent system where agents will: travel to end-hosts; intelligently gather data; reach a conclusion; and travel back to source. As Buchanan et al. [36]'s system relies on the mobile agent paradigm it is less likely that data would be lost in transit, thus partly addressing concerns over data

completeness [85], and also this allows for the creation of a central repository for analysis (Section 3.7.3). Along with this, such a centralised repository eases subsequent analysis [88, 84] and ensures that data, such as e-mail records, are only stored for period of times are permitted by legal requirements [103, 100]. Nevertheless, Buchanan et al. [36] stress that these agents need to be policed accordingly and argue that this is best achieved within an integrated framework which address the security requirements of an organisation and also caters for forensic investigation criteria. Indeed, agents can be recalled and re-issued with new objectives accordingly [36].

8.8 Analysis Phase

The aim of the Analysis Phase should be to process the collected System Metrics (Figure 8.5) in order to obtain an overview of the system computer network status, identify threats, and raise alerts if required. The output of this phase should use an XML format, as this allows for the system and threat data to be presented, merged, or further investigated in a flexible manner [191, 187, 186]. To that end, the **Interpreter** should identify the data that is relevant to the functions that the **Asserssor** and **Threat Analyser** perform (Figure 8.5), and thus the Interpretor should format the relevant data accordingly. Indeed, without the Interpreter both the Assessor and Threat Analyser would have to filter the System Metrics as this data typically originates from several sources (Section 8.7) and these often have their own format and meaning [86] (Section 3.7.3). Arguably, the Interpreter complements the storage facility time constraints as it also ensures that the output data-sets comply with relevant regulations [95, 103, 104, 46, 98, 100], by, for example, sanitising the data so that it does not include identifiable and private information (Section 3.2.2).

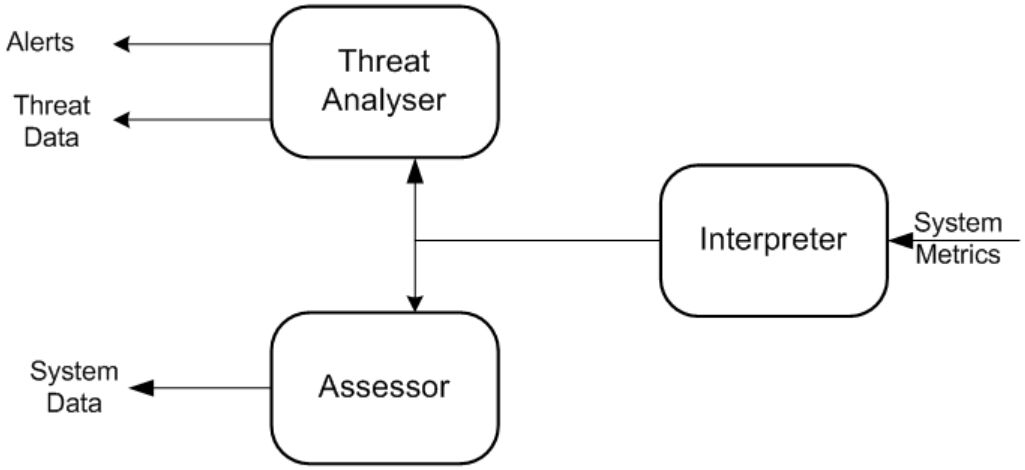

Figure 8.5 – ISF Analysis Phase

The **Threat Analyser** (Figure 8.5) should uncover threats by correlating the various security information made available, where a possible intrusion vector only matters if the corresponding vulnerable devices are present within the computer network [31]. Hence, the Threat Analyser should support multiple analysis engines, such as [22, 14, 86, 57, 81, 90]. For example, Ning and Xu [81]'s method is suited to identify intrusion that are spread over long period of time and that have several milestones, such as subverted user permissions on a workstation. One the other hand, for real-time detection, a system such as of Zou et al. [22] is required as it is capable of identifying fast propagating threats, such as computer worms, with a limited number of vulnerable hosts. Along with this, Julisch [90]'s approach can be employed to identify devices that are not configured properly and generate *false negatives*. If required, the Threat Analyser raises an alert that feeds into the Threat Knowledge Base (Figure 8.3) thus triggering the Security Modeller and Device Expert System (Section 8.5) into action to close loopholes accordingly, such as reapplying the intended workstation configuration. Arguably, the Threat Analyser cannot fully replace an administrator, such as in terms of being able to identify new pattern of activities [36], nevertheless the Threat Analyser off-loads some of the burden for administrators.

The **System Status Verifier** should produce compliance data, such as from routing information, and from network throughput measurements. The parameters of this analysis are based on the elements stipulated during the Modelling Phase (Section 8.5). Arguably, the structure of the ISF allows for this procedure to be used for the day-to-day monitoring of the overall computer network system.

This phase also represents the opportunity to refresh the **Networked System Knowledge Base** (Figure 8.3, and Section 8.5). Indeed, it is unlikely that all the weaknesses can be identified and addressed immediately with the Security Modeller where new data can emerge from updated evaluation methodologies. For example, Saliou et al. [78] show that there is often a negative impact on network latency in employing incoming interface filtering on network firewalls. Thus, the System Status Verifier could update the knowledge base accordingly. This particular functionality has consequences that radiates throughout the framework, such as outlined in Section 8.6.

8.9 Compliance Assessment

The purpose of this phase is to tailor the System Data and Threat Data (Figure 8.6) to specific audiences, such as with technical reports for administrators or **Formal Management Documents** for decision makers. Since the input for this phase could be based on XML, these documents can be generated from a common data source [186]. To this end, the **Distiller** relies on a repository of e**X**tensible **S**tylesheet **L**anguage

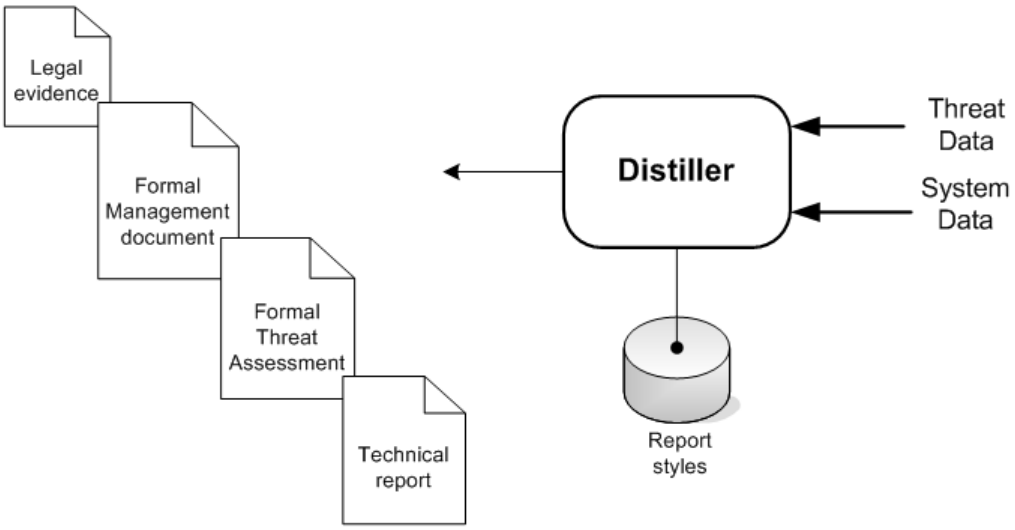

Figure 8.6 – ISF Compliance Phase

Transformations (XSLT), and XSL Formatting Objects (XSL-FO) templates. XSLT typically allows for the extraction and presentation of data contained in an XML document [191, 178]. XSL-FO is better suited for preparing XML data within a view to create printer-ready documents [192]. One of the advantages of separating content from presentation include the flexibility of updating one element without necessarily impacting on the other [193]. Hence, the information obtained from the computer network can be presented in a style that is suitable for legal purposes, such as network performance data as compared against evidence of DDoS attacks [99] (Section 3.2.3).

The Formal Management Document and **Threat Assessment Document** should be free from unecessary technical details, and could use a style that is similar to the Security Policy itself [21] (Figure 8.2, and Section 8.4). This, in turn, allows for the contrasting of the gathered data against the security policy specifications, and thus trigger a response, such as a change in policy, or the deployment of a new security device.

This phase also represents the opportunity to demonstrates the benefits of securing computer networks to decision makers (section 3.4.2). Thus, a Formal Management Document could contain a summary of the number of intrusions or faults detected over a given period, and provide a break-down of the critical incidents that administrators needed to investigate, for example. Investigating an intrusion is typically costly [90, 3, 8], thus this analysis could allow for deriving the amount of money saved in terms of man-hours required, by focusing only the most important ones. This, in turn, gives decision makers an estimate of the RoI for the organisation. Indeed, a "*tick box*" approach needs to be avoided as this is not sufficient to enable a comprehensive understanding of security posture, or readiness. Indeed, several researchers, such as Rodgers [12] or Baker and Wallace [55], independently showed

that decision makers often wrongly assume that owning a license for a system or a technology, such as an IDS, or anti-virus software, equates to enhanced security. Such an approach can be detrimental to organisations when a security technology does not address a particular threat. Alun et al. [8] highlight that many organisations assume that anti-viruses are effective against spyware, or information phishing [132]. Hence, the process described thus far can highlight the discrepancy between purchasing such an item and the fact that it has not been deployed, or that some security objectives are not met.

8.10 Conclusion

This chapter presented an ISF that would produce security policies which match the aims and objectives of the organisation, and then would ensure that these can be accomplished. To this end, ISF would understand the possible repercussions on the underlying networked systems, enabling it to inform decision makers when objectives cannot be met. Thus, the ISF methodology allows for addressing computer network security as a process rather than an add-on to a network computer system, or a collection of technologies. Another emphasis is on limiting manual interventions. Thus, the analysis of the security requirements, and the design of subsequent device configurations, employ modelling techniques, instead of relying on the interpretation of administrators when they configure networked devices (Section 3.3.2). Along with this, the deployment and data gathering phase employ dedicated software agents that, in turn, allow faster response to security threats. Hence, the ISF methodology improves upon the current manual and unaccountable approach, and thus possible benefits would be:

- **Optimisation**: security policies can be refined based on the information gathered and analysed (Section 8.4, Section 8.5, and Section 8.9).
- **Improved network security**: deployment is rigorous, consistent and based upon best implementation practices (Section 8.6).
- **Risk assessment**: the possible defects are identified early, and the adequate information is made available to the decision makers (Section 8.4 and Section 8.9).
- **Self-healing**: most of the procedure can be implemented in a software form, and thus deployed using MAS [25, 21] (Section 3.8, Section 8.6, and Section 8.8).

A key component of the ISF is the availability of dynamic performance models of security devices (Section 8.5), and thus it provides a context in which DEENS (Section 4.3) can exist. In other words, this chapter illustrated where the data DEENS provides, and d-Feral analyses through formalisation (Section 5.3 and Section 5.4.2), can be utilised.

References

[1] L. Greenemeier, "The TJX Effect: Details of the largest breach of customer data are starting to come to light." August 17 2007. [Online]. Available: http://www.informationweek.com/shared/printableArticle.jhtml?articleID=201400171

[2] D. McCullagh, "University suffers massive ID data theft," January, 11 2005. [Online]. Available: http://news.zdnet.co.uk/internet/security/0,39020375,39183592,00.htm

[3] S. Timms, C. Potter, and A. Beard, "Information Security Breaches Survey 2004," UK Government, Technical, 2004.

[4] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring Internet denial-of-service activity," *ACM Transactions on Computer Systems*, vol. 24, no. 2, pp. 115–139, 2006.

[5] TheEconomist, "A cyber-riot," February 7th 2007. [Online]. Available: http://www.economist.com/world/europe/PrinterFriendly.cfm?story_id=9163598

[6] J. Davis, "World War 2.0," June 10 2008. [Online]. Available: http://www.pbs.org/kcet/wiredscience/video/83-world_war_2_0.html

[7] CNN, "CNN Web site targeted," April 19th 2008. [Online]. Available: http://edition.cnn.com/2008/TECH/04/18/cnn.websites/index.html

[8] M. Alun, C. Potter, and A. Beard, "Information Security Breaches Survey 2006," PriceWaterHouseCooper, Tech. Rep., April 25th 2006.

[9] S. H. Bakry, "Development of security policies for private networks," *International Journal of Network Management*, vol. 13, no. 3, pp. 203–210, 2003.

[10] D. Danchev, "Building and Implementing a Successful Information Security Policy," Window Security, PDF, June 19th 2003.

[11] J. Rees, S. Bandyopadhyay, and E. H. Spafford, "PFIRES: a policy framework for information security," *Communications of the ACM*, vol. 46, no. 7, pp. 101–106, 2003.

[12] D. H. Rodgers, "Implementing a Project Security Review Process within Project Management Methodology," SANS, Practical Assignment, November 21, 2002 2003.

[13] L. Briesemeister, P. Lincoln, and P. Porras, "Epidemic profiles and defense of scale-free networks," in *ACM workshop on Rapid malcode*. Washington, DC, USA: ACM Press New York, NY, USA, 2003, pp. 67–75.

[14] D. Caldwell, A. Gilbert, J. Gottlieb, A. Greenberg, G. Hjalmtysson, and J. Rexford, "The cutting EDGE of IP router configuration," *SIGCOMM Computer Communication Review*, vol. 34, no. 1, pp. 21–26, 2004.

[15] R. Bajcsy, T. Benzel, M. Bishop, B. Braden, C. Brodley, S. Fahmy, S. Floyd, W. Hardaker, A. Joseph, G. Kesidis, K. Levitt, B. Lindell, P. Liu, D. Miller, R. Mundy, C. Neuman, R. Ostrenga, V. Paxson, P. Porras, C. Rosenberg, J. D. Tygar, S. Sastry, D. Sterne, and S. F. Wu, "Cyber defense technology networking and evaluation," *Communications of the ACM*, vol. 47, no. 3, pp. 58–61, 2004.

[16] M. Glenn, "A summary of DoS/DDoS Prevention, monitoring and Mitigation Techniques in a Service Provider Environment," SysAdmin, Audit, Network, Security Institute, White Paper, August 2003.

[17] D. Moore, C. Shannon, G. M. Voelker, and S. Savage, "Internet Quarantine: Requirements for Containing Self-Propagating Code," in *IEEE INFOCOM 2003*, IEEE, Ed., San Francisco, California, USA, April, 2003 2003.

[18] V. Paxson, "An Analysis of Using Reflectors for Distributed Denial-Of-Service Attacks," *Computer Communication Review*, vol. 31, no. 3, pp. 38 – 47, 2001.

[19] A. T. Campbell, H. G. De Meer, M. E. Kounavis, K. Miki, J. B. Vicente, and D. Villela, "A survey of programmable networks," *ACM SIGCOMM Computer Communication Review*, vol. 29, no. 2, pp. 7 – 23, 1999.

[20] S. Staniford, V. Paxson, and N. Weaver, "How to Own the Internet in Your Spare Time," in *Proceedings of the 11th USENIX Security Symposium*. USENIX Association, 2002, pp. 149–167.

[21] L. Saliou, W. J. Buchanan, J. Graves, and J. Munoz, "Novel Framework for Automated Security Abstraction, Modelling, Implementation and Verification," in *4th European Conference on Information Warfare and Security*, W. Hutchinson, Ed. Glamorgan, United Kingdom: Academic Conferences Limited, July 11-12 2005, pp. 303–311.

[22] C. C. Zou, L. Gao, W. Gong, and D. Towsley, "Monitoring and early warning for internet worms," in *10th ACM Conference on Computer and Communications Security*, Washington D.C., USA, 2003, pp. 190–199.

[23] P. J. Denning, "Network laws," *Communications of the ACM*, vol. 47, no. 11, pp. 15–20, 2004.

[24] S. Peisert and M. Bishop, "How to Design Computer Security Experiments," in *Fifth World Conference on Information Security Education (WISE)*, West Point, NY, 2007.

[25] G. A. Santana Torrellas and L. A. Villa Vargas, "Modelling a flexible network security systems using multi-agents systems: security assessment considerations," in *1st international symposium on Information and communication technologies*. Dublin, Ireland: Trinity College Dublin, 2003, pp. 365 – 371.

[26] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[27] P. Barton and V. Nissanka, "Cyber-crime - Criminal offence or civil wrong?" *Computer Law and Security Report*, vol. 19, no. 5, pp. 401–405, 2003.

[28] D. K. Smetters and R. E. Grinter, "Moving from the design of usable security technologies to the design of useful secure applications," in *2002 workshop on New security paradigms*. Virginia Beach, Virginia, USA: ACM Press New York, NY, USA, September, 23-26 2002, pp. 82 – 89.

[29] J. Viega and M. Messier, "Security is Harder than You Think," *Queue*, vol. 2, no. 5, pp. 60–65, 2004.

[30] M. R. Lyu and L. K. Y. Lau, "Firewall Security: Policies, Testing and Performance Evaluation," in *24th International Computer Software and Applications Conference*. IEEE Computer Society, 2000, pp. 116–121.

[31] S. Kamara, S. Fahmy, E. Schultz, F. Kerschbaum, and M. Frantzen, "Analysis of Vulnerabilities in Internet Firewalls," *Computers and Security*, vol. 22, no. 3, pp. 214 – 232, 2003.

[32] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the characteristics and origins of internet flow rates," in *the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*. Pittsburgh, Pennsylvania, USA: ACM Press, 2002, pp. 309–322.

[33] M. Donat, "Orchestrating an automated test lab," *Queue*, vol. 3, no. 1, pp. 46–53, 2005.

[34] M. S. Saleh, A. Alrabiah, and S. H. Bakry, "Using ISO 17799: 2005 information security management: a STOPE view with six sigma approach," *International Journal of Network Management*, vol. 17, no. 1, pp. 85–97, 2007.

[35] J. M. Myerson, "Identifying enterprise network vulnerabilities," *International Journal of Network Management*, vol. 12, no. 3, pp. 135–144, 2002.

[36] W. J. Buchanan, J. Graves, L. Saliou, H. Al Sebea, and N. Migas, "Agent-based Forensic Investigations with an Integrated Framework," in *4th European Conference on Information Warfare and Security*, W. Hutchinson, Ed. Glamorgan, United Kingdom: Academic Conferences Limited, July 11-12 2005, pp. 47–52.

[37] W. Odom, *CCNA Self-Study Exam Certification Guides*. Cisco Press, 2004.

[38] K. Doughty, "Implementing enterprise security: a case study," *Computers & Security*, vol. 22, no. 2, pp. 99 – 114, 2003.

[39] J. Viega, "Security—problem solved?" *Queue*, vol. 3, no. 5, pp. 40–50, 2005.

[40] Financial Services Authorities, "Data Security in Financial Services: Firms' controls to prevent data loss by their employees and third-party suppliers," FSA, Tech. Rep., April 2008.

[41] D. Danchev, "Reducing "Human Factor" Mistakes," November, 2nd 2003. [Online]. Available: http://windowsecurity.com/pages/article_p.asp?id=1261

[42] W. Harrison, G. Heuston, S. Mocas, M. Morrissey, and J. Richardson, "High-tech forensics," *Communications of the ACM*, vol. 47, no. 7, pp. 48–52, 2004.

[43] I. Bashir, E. Serafini, and K. Wall, "Securing network software applications: introduction," *Communications of the ACM*, vol. 44, no. 2, pp. 28 – 30, 2001.

[44] C. Huseyin, B. Mishra, and S. Raghunathan, "A model for evaluating IT security investments," *Communications of the ACM*, vol. 47, no. 7, pp. 87 – 92, 2004.

[45] M. Cremonini and P. Martini, "Evaluating information security investments from attackers perspective: the Return-On-Attack (ROA)," in *Fourth Workshop on the Economics of Information Security*, Kennedy School of Government Harvard University, Cambridge, Massachusetts, USA, 2 - 3 June 2005.

[46] S. Kierkegaard, "Privacy in electronic communication: Watch your e-mail: Your boss is snooping!" *Computer Law and Security Report*, vol. 21, no. 3, pp. 226–236, 2005.

[47] B. Woloch, "New dynamic threats requires new thinking - "Moving beyond compliance"," *Computer Law and Security Report*, vol. 22, no. 2, pp. 150–156, 2006.

[48] J. Howie, "Microsoft's Vision of "defence in Depth"," December, 15th 2003 2003.

[49] K. Curran, S. McIntyre, H. Meenan, F. McCloy, and C. Heaney, "Civil Liberties and Computer Monitoring," *American Journal of Applied Sciences*, vol. 1, no. 3, pp. 225–229, 2004.

[50] S. Ioannidis, A. D. Keromytis, S. M. Bellovin, and J. M. Smith, "Implementing a distributed firewall," in *7th ACM conference on Computer and communications security*, Athens, Greece, 2000, pp. 190–199.

[51] X. Wang, D. S. Reeves, S. F. Wu, and J. Yuill, "Sleepy watermark tracing: an active network-based intrusion response framework," in *Proceedings of the 16th international conference on Information security: Trusted information: the new decade challenge*, Paris, France, 2001, pp. 369–384.

[52] J. D. Guttman, "Filtering postures: local enforcement for global policies," in *Proceedings of the 1997 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 1997, pp. 120 –131.

[53] T. E. Uribe and S. Cheung, "Automatic analysis of firewall and network intrusion detection system configurations," in *Proceedings of the 2004 ACM workshop on Formal methods in security engineering*. Washington DC, USA: ACM Press, 2004, pp. 66–74.

[54] D. Bruschi and E. Rosti, "Disarming offense to facilitate defense," in *2000 workshop on New security paradigms*, ser. New Security Paradigms Workshop. Ballycotton, County Cork, Ireland: ACM Press New York, NY, USA, 2001, pp. 69–75.

[55] W. H. Baker and L. Wallace, "Is Information Security Under Control?: Investigating Quality in Information Security Management," *IEEE Security and Privacy*, vol. 5, no. 1, pp. 36–44, 2007.

[56] Metasploit-LLC, "The MetaSploit Project," July, 1 2006. [Online]. Available: http://www.metasploit.com

[57] P. Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *9th ACM conference on Computer and communications security*. Washington, DC, USA: ACM Press New York, NY, USA, 2002, pp. 245 – 254.

[58] L. A. Hughes and G. J. DeLone, "Viruses, Worms, and Trojan Horses: Serious Crimes, Nuisance, or Both?" *Social Science Computer Review*, vol. 25, no. 1, pp. 78–98, 2007.

[59] D. Moore and C. Shannon, "The Spread of the Witty Worm," *IEEE Security & Privacy*, vol. 2, no. 4, 2004.

[60] B. Potter, "New threat of Apple Mac OS X," *Network Security*, vol. 2006, no. 2, pp. 4–5, 2006.

[61] D. L. Hoffman, T. P. Novak, and A. Venkatesh, "Has the Internet become indispensable?" *Communications of the ACM*, vol. 47, no. 7, pp. 37–42, 2004.

[62] G. Groos, "'Witty' worm exploits hole in BlackIce security product," October, 22nd 2004. [Online]. Available: http://www.computerworld.com/printthis/2004/0,4814,91528,00.html

[63] B. Schneier, "The Witty worm: A new chapter in malware," October, 22nd 2004. [Online]. Available: http://www.computerworld.com/printthis/2004/0,4814,93584,00.html

[64] K. D. Mitnick and W. L. Simon, *The Art of Deception: Controlling the Human Element of Security*. John Wiley & Sons, Inc., 2002.

[65] J. Eloff and M. Eloff, "Information security management: a new paradigm," in *2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology*. Gauteng, South Africa: South African Institute for Computer Scientists and Information Technologists Republic of South Africa, September, 17-19 2003, pp. 130–136.

[66] T. Corbitt, "Protect you computer system with a security policy," *Management Services*, vol. 46, no. 5, pp. 20–21, May 2002.

[67] J. Graves, W. J. Buchanan, L. Saliou, and J. Old, "Towards a Framework For Evaluating System Call Data as a Source of Digital Forensic Evidence," in *The 2nd Conference on Advances in Computer Security and Forensics*, Liverpool, UK, July 12-13 2007, pp. 90–96.

[68] F. B. Schneider, "Enforceable security policies," *ACM Transactions on Information and System Security*, vol. 3, no. 1, pp. 30–50, 2000.

[69] M. Roesch, "Snort - the de facto standard for Intrusion detection / prevention," February 2005. [Online]. Available: http://www.snort.org/

[70] S. Antonatos, K. G. Anagnostakis, E. P. Markatos, and M. Polychronakis, "ExB: Exclusion-based signature matching for intrusion detection," in *IASTED International Conference on Communications and Computer Networks (CCN)*, vol. 1, Cambridge, USA, November 2002, pp. 146–152.

[71] N. Weaver, D. Ellis, S. Staniford, and V. Paxson, "Worms vs. perimeters: the case for hard-LANs," in *12th Annual IEEE Symposium on High Performance Interconnects, 25-27 Aug. 2004*, ser. Proceedings. 12th Annual IEEE Symposium on High Performance Interconnects. Stanford, CA, USA: IEEE Computer Society, 2004 2004, pp. 70 – 76.

[72] F. M. Avolio, "Putting it together a multi-dimensional approach to Internet security," *netWorker*, vol. 2, no. 2, pp. 15–22, 1998.

[73] R. Oppliger, "Internet Security: Firewalls and Beyond," *Communications of the ACM*, vol. 40, no. 5, pp. 92–102, 1997.

[74] K. Al-Tawil and I. A. Al-Kaltham, "Evaluation and testing of internet firewalls," *International Journal of Network Management*, vol. 9, no. 3, pp. 135 – 149, 1999.

[75] E. S. Al-Shaer and H. H. Hamed, "Modelling and Management of Firewall Policies," *IEEE Transactions on Network and Service Management*, vol. 1, no. 1, pp. 3 – 13, 2004.

[76] L. Yuan, H. Chen, J. Mai, C.-N. Chuah, Z. Su, and P. Mohapatra, "FIREMAN: A toolkit for firewall modeling and analysis," in *IEEE Symposium on Security and Privacy*, ser. Proceedings - IEEE Symposium on Security and Privacy, vol. 2006. Berkeley/Oakland, California, United States of America: Institute of Electrical and Electronics Engineers Inc., New York, NY 10016-5997, United States, May 21-24 2006, pp. 199–213.

[77] D. Hoffman and K. Yoo, "Blowtorch: a framework for firewall test automation," in *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*. Long Beach, CA, USA: ACM Press, 2005, pp. 96–103.

[78] L. Saliou, W. J. Buchanan, J. Graves, and J. Munoz, "Analysis of Firewall Performance Variation to Identify the Limits of Automated Network Reconfigurations," in *5th European Conference on Information Warfare and Security*, W. Hutchinson, Ed. Helsinki, Finland: Academic Conferences Limited, June 1-2 2006, pp. 205–214.

[79] I. G. Lugo and D. Parker, "Software Firewalls: Made of Straw? Part 1," 15 June 2005. [Online]. Available: http://www.security-focus.com/print/infocus/1839

[80] T. Porter, "The Perils of Deep Packet Inspection," January 27 2005. [Online]. Available: http://www.securityfocus.com/print/infocus/1817

[81] P. Ning and D. Xu, "Learning attack strategies from intrusion alerts," in *Conference on Computer and Communications Security*, Washington D.C., USA, 2003, pp. 200–209.

[82] S. Rayan, R. Pradeep, and N. Paramewaran, "Network management platform based on mobile agents," *International Journal of Network Management*, vol. 14, no. 1, pp. 59–73, 2004.

[83] J. E. Lopez de Vergara, V. A. Villagra, J. I. Asensio, and J. Berrocal, "Ontologies: giving semantics to network management models," *IEEE Network*, vol. 17, no. 3, pp. 15 – 21, 2003.

[84] A. Chuvakin, "Advanced Log Processing," november, 2nd 2002. [Online]. Available: http://www.securityfocus.com/print/infocus/1613

[85] J. Graves, W. J. Buchanan, L. Saliou, and J. Old, "Performance Analysis of Network Based Forensic Systems for In-line and Out-of-line Detection

and Logging," in *5th European Conference on Information Warfare and Security*, W. Hutchinson, Ed. Helsinki, Finland: Academic Conferences Limited, June 1-2 2006, pp. 41–50.

[86] W. Yan, E. Hou, and N. Ansari, "Extracting Attack Knowledge Using Principal-Subordinate Consequence Tagging Case Grammar and Alerts Semantic Networks," in *the 29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*. IEEE Computer Society, 2004, pp. 110–117.

[87] H. H. Thompson and R. Ford, "Perfect Storm: The Insider, Naivety, and Hostility," *Queue*, vol. 2, no. 4, pp. 58–65, 2004.

[88] B. Laurie, "Network Forensics," *Queue*, vol. 2, no. 4, pp. 50–56, 2004.

[89] M. Carney and M. Rogers, "The Trojan Made Me Do It: A First Step in Statistical Based Computer Forensics Event Reconstruction," *International Journal of Digital Evidence*, vol. 2, no. 4, pp. 1 – 11, 2004.

[90] K. Julisch, "Clustering intrusion detection alarms to support root cause analysis," *ACM Transactions on Information and System Security (TISSEC)*, vol. 6, no. 4, pp. 443 – 471, 2003.

[91] M. Perdikeas, F. Chatzipapadopoulos, I. Venieris, and G. Marino, "Mobile agent standards and available platforms," *Computer Networks*, vol. 31, no. 19, pp. 1999–2016, 1999.

[92] D. S. Alexander, M. Shaw, S. M. Nettles, and J. M. Smith, "Active bridging," in *Proceedings of the ACM SIGCOMM '97 conference on Applications, technologies, architectures, and protocols for computer communication*. Cannes, France: ACM Press, 1997, pp. 101–111.

[93] Y. Li and L. Wolf, "Collection of network information in active networks," *ACM SIGOPS Operating Systems Review*, vol. 35, no. 4, pp. 39 – 49, 2001.

[94] M. J. Ranum, "Security: The root of the Problem," *Queue*, vol. 2, no. 4, pp. 44–49, 2004.

[95] A. Escudero-Pascual and I. Hosein, "Questioning lawful access to traffic data," *Communications of the ACM*, vol. 47, no. 3, pp. 77–82, 2004.

[96] Z. Kwecka, W. J. Buchanan, D. Spiers, and L. Saliou, "Validation of 1-N OT Algorithms in Privacy-Preserving Investigations," in *7th European Conference on Information Warfare and Security*, B. Hutchinson, Ed. Plymouth, United Kingdom: Academic Conferences Ltd, June 30 - July 1 2008, pp. 119–127.

[97] G. Kon and P. Church, "A denial of service but not a denial of justice," *Computer Law and Security Report*, vol. 22, no. 5, pp. 416–417, 2006.

[98] F. Mares, "The Regulation of Investigatory Powers Act 2000: Overview of the case of R v. Clifford Stanford (CA (Crim Div) 1 February 2006) and the offence of unlawfully intercepting telecommunications on a private system (section 1(2) offence)," *Computer Law and Security Report*, vol. 22, no. 3, pp. 254–256, 2006.

[99] J. Worthy and M. Fanning, "Denial-of-Service: Plugging the legal loopholes?" *Computer Law and Security Report*, vol. 23, no. 2, pp. 194–198, 2007.

[100] E. Kosta and P. Valcke, "Retaining the data retention directive," *Computer Law and Security Report*, vol. 22, no. 5, pp. 370–380, 2006.

[101] S. Furnell and M. Papadaki, "Testing our defences or defending our tests: the obstacles to performing security assessment references," *Computer Fraud & Security*, vol. 2008, no. 5, pp. 8–12, 2008.

[102] M. A. Cusumano, "Who is liable for bugs and security flaws in software?" *Communications of the ACM*, vol. 47, no. 3, pp. 25–27, 2004.

[103] R. Dixon, "With Nowhere to Hide: Workers are Scrambling for Privacy in the Digital Age," *Journal of Technology Law and Policy*, vol. 4, no. 1, pp. 1 – 27, 1999.

[104] A. Rogers, "You Got Mail But Your Employer Does Too: Electronic Communication and Privacy in the 21st Century Workplace," *Journal of Technology Law and Policy*, vol. 5, no. 1, pp. 1 – 10, 2000.

[105] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 DARPA Off-Line Intrusion Detection Evaluation," *Computer Networks*, vol. 34, no. 4, pp. 579–595, 2000.

[106] A. Wool, "The use and usability of direction-based filtering in firewalls," *Computers and Security*, vol. 23, no. 1, pp. 459–468, 2004.

[107] A. Wool, "A Quantitative Study of Firewall Configuration Errors ," *IEEE Computer*, vol. 37, no. 6, pp. 62–67, 2004.

[108] X. Wang, S. Chen, and S. Jajodia, "Tracking anonymous peer-to-peer VoIP calls on the internet," in *the 12th ACM conference on Computer and Communications Security*. Alexandria, VA, USA: ACM, November 7-11 2005, pp. 81–91.

[109] D. Porter, "Insider Fraud: Spotting The Wolf In Sheep's Clothing," *Computer Fraud & Security*, vol. 2003, no. 4, pp. 12–15, 2003.

[110] I. S. Winkler, "Case study of industrial espionage through social engineering," in *national Information Systems Security*, Baltimore, MD, USA, 1996.

[111] T. A. Wadlow, "The answer is 42 of course," *Queue*, vol. 3, no. 5, pp. 34–39, 2005.

[112] J. Morrison, "Blaster Revisited," *Queue*, vol. 2, no. 4, pp. 34 – 43, 2004.

[113] V. Yegneswaran, P. Barford, and J. Ullrich, "Internet intrusions: global characteristics and prevalence," *ACM SIGMETRICS Performance Evaluation Review*, vol. 31, no. 1, pp. 138 – 147, 2003.

[114] W. Knight, "How to second-guess the next hack attack," *New Scientist*, vol. 181, no. 2431, p. 19, 2004.

[115] R. J. Shimonski, "Threats and your Assets - What is really at Risk?" November, 2nd 2004. [Online]. Available: http://windowsecurity.com/pages/article_p.asp?id=1354

[116] M. E. Whitman, "Enemy at the gate: threats to information security," *Communications of the ACM*, vol. 46, no. 8, pp. 91–95, 2003.

[117] F. Stajano and R. Anderson, "The Resurrecting Duckling: Security Issues for Ubiquitous Computing," January, 26 2002. [Online]. Available: http://www.computer.org/security/supplement1/sta/print.htm

[118] G. A. Nolann, "Seeking the Compliance Nirvana," *Queue*, vol. 4, no. 7, pp. 72 – ff, 2006.

[119] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time," *Computer Networks*, vol. 31, no. 23-24, pp. 2435–2463, 1999.

[120] E. Bertino, S. Castano, and E. Ferrari, "On specifying security policies for web documents with an XML-based language," in *Proceedings of the sixth ACM symposium on Access control models and technologies*. Chantilly, Virginia, United States: ACM Press, 2001, pp. 57–65.

[121] C. Merchant and J. Stewart, "Detecting and Containing IRC-Controlled Trojans: When Firewalls, AV, and IDS Are Not Enough," October, 11th 2002. [Online]. Available: http://www.securityfocus.com/print/infocus/1605

[122] F. Tari, A. A. Ozok, and S. H. Holden, "A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords," in *Proceedings of the second symposium on Usable privacy and security*. Pittsburgh, Pennsylvania: ACM Press, 2006, pp. 56–66.

[123] E. E. Schultz, "When firewalls fail: Lessons learned from firewall testing," *Network Security*, vol. 1997, no. 2, pp. 8–11, 1997.

[124] P. R. Moyer, "Enhanced firewall infrastructure testing methodology," *Network Security*, vol. 1997, no. 4, pp. 9–15, 1997.

[125] G. Rosamond, "Building a more Secure Network," SANS, Tech. Rep., March, 8 2004.

[126] A. J. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten, "Lest We Remember: Cold Boot Attacks on Encryption Keys," in *17th USENIX Security Symposium (Sec '08)*, San Jose, California, USA, July 28 - August 1 2008.

[127] A. Chuvakin, "Issues Discovering Compromised machines," November, 2nd 2004. [Online]. Available: http://www.securityfocus.com/print/infocus/1808

[128] P. Ferrie and F. Perriot, "Detecting Complex Viruses," December, 13 2004. [Online]. Available: http://www.securityfocus.com/print/infocus/1813

[129] S. P. Gorman, R. G. Kulkarni, L. A. Schintler, and R. R. Stough, "A predator prey approach to the network structure of cyberspace," in *winter international symposium on Information and communication technologies*. Cancun, Mexico: Trinity College Dublin, 2004, pp. 1–6.

[130] P. Wood, "The hacker's top five routes into the network (and how to block them)," *Network Security*, vol. 2006, no. 2, pp. 5–9, 2006.

[131] D. Hanson, B. Kostanecki, R. Jagodzinski, and J. V. Miller, "A Comparison Study of Three Worm Families and Their Propagation in a Network," 11th October 2003. [Online]. Available: http://www.securityfocus.com/print/infocus/1752

[132] A. Karakasiliotis, S. M. Furnell, and M. Papadaki, "An assessment of end-user vulnerability to phishing attacks," *Journal of Information Warfare*, vol. 6, no. 1, pp. 17 – 28, 2007.

[133] Netfilter Core Team, "The Netfilter.org Project," July, 17 2006. [Online]. Available: http://www.netfilter.org/

[134] P. Piyachon and Y. Luo, "Efficient memory utilization on network processors for deep packet inspection," in *the 2006 ACM/IEEE symposium on Architecture for networking and communications systems*. San Jose, California, USA: ACM Press, 2006, pp. 71–80.

[135] O. Airamo and T. Virtanen, "Enterprise IPv6 Firewalling," in *5th European Conference on Informaiton Warfare and Security*, W. Hutchinson, Ed., vol. 1. Helsinki, Finland: Academic Conferences Limited, 1 - 2 June 2006, pp. 1 – 8.

[136] P. R. Moyer and E. E. Schultz, "A systematic methodology for firewall penetration testing," *Network Security*, vol. 1996, no. 3, pp. 11–18, 1996.

[137] D. Morgan, "Low-level network assessment: Firewalls," *Network Security*, vol. 2005, no. 1, pp. 14–16, 2005.

[138] T. Verdickt, W. Van de Meerssche, and K. Vlaeminck, "Modeling the performance of a NAT/firewall network service for the IXP2400," in *the 5th international workshop on Software and performance*. Palma, Illes Balears, Spain: ACM Press, 2005, pp. 137–144.

[139] K. Accardi, T. Bock, F. Hady, and J. Krueger, "Network processor acceleration for a Linux* netfilter firewall," in *the 2005 symposium on Architecture for networking and communications systems*. Princeton, NJ, USA: ACM Press, 2005, pp. 115–123.

[140] P. Gupta and N. McKeown, "Algorithms for packet classification," *IEEE Network*, vol. 15, no. 2, pp. 24–32, 2001.

[141] D. Srinivasan and W.-c. Feng, "Performance Analysis of Multi-dimensional Packet Classification on Programmable Network Processors," in *29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*. IEEE Computer Society, November 16-18 2004, pp. 360–367.

[142] H. H. Hamed and E. S. Al-Shaer, "Dynamic rule-ordering optimization for high-speed firewall filtering," in *The 2006 ACM Symposium on Information, computer and communications security*. Taipei, Taiwan: ACM Press, 2006, pp. 332–342.

[143] L. Saliou, W. J. Buchanan, J. Graves, and J. Munoz, "Scenario Analysis using Out-of-line Firewall Evaluation Framework," in *6th European Conference on Information Warfare, and Security*. Shrivenham, UK: Academic Conferences Limited, July 2-3 2007, pp. 205–214.

[144] B. Hickman, D. Newman, S. Tadjudin, and T. Martin, "RFC 3511 - Benchmarking Methodology for Firewall Performance," February, 7 2003. [Online]. Available: http://www.faqs.org/rfcs/rfc3511.html

[145] U. Roedig and J. Schmitt, "Multimedia and firewalls: a performance perspective," *Multimedia Systems*, vol. 11, no. 1, pp. 19–33, 2005.

[146] S. Antonatos, K. G. Anagnostakis, and E. P. Markatos, "Generating realistic workloads for network intrusion detection systems," in *fourth international workshop on Software and performance*. Redwood Shores, California, USA: ACM Press New York, NY, USA, 2004, pp. 207–215.

[147] Y. Wang and C. Wang, "Modeling the effects of timing parameters on virus propagation," in *ACM Conference on Computer and Communications Security*, Washington, DC, USA, 2003, pp. 61–66.

[148] Y.-F. Hwang and D. C. Rine, "Algorithms to detect chained-inference faults in information distribution systems," in *2001 ACM symposium on Applied computing*. Las Vegas, Nevada, United States: ACM Press New York, NY, USA, 2001, pp. 679 – 685.

[149] J. Sommers, V. Yegneswaran, and P. Barford, "A framework for malicious workload generation," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. Taormina, Sicily, Italy: ACM Press, 2004, pp. 82–87.

[150] J. Leyden, "Why Sobig is bad for privacy and AV vendors," October 11th 2003. [Online]. Available: http://www.securityfocus.com/print/news/6810

[151] G. Szappanos, "Models of Viral Propagation," October 11th 2000. [Online]. Available: http://www.securityfocus.com/print/infocus/1265

[152] J. L. Undercoffer, A. Joshi, T. Finin, and J. Pinkston, "A Target-Centric Ontology for Intrusion Detection," in *18th International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, August 11 2003.

[153] A. Piszcz, N. Orlans, Z. Eyler-Walker, and D. Moore, "Engineering Issues for an Adaptive Defense Network," MITRE, Technical MTR 01W0000103, June 2001.

[154] T. Lindholm and F. Yellin, *The Java (TM) Virtual Machine Specification*, 2nd ed. Addison-Wesley Professional, 1999.

[155] J. W. Lockwood, C. Neely, C. Zuver, and D. Lim, "Automated tools to implement and test Internet systems in reconfigurable hardware," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 103 – 110, 2003.

[156] G. Flores Lucio, M. Paredes-farrera, E. Jammeh, M. Fleury, and M. J. Reed, "OPNET modeler and Ns-2: Comparing the accuracy of network simulators for packet-level analysis using a network testbed," in *3rd WEAS International Conference on Simulation, Modelling and Optimization (ICOSMO)*, 2003, pp. 700 – 707.

[157] M. Lacage and T. R. Henderson, "Yet another network simulator," in *2006 workshop on ns-2 (WNS2): the IP network simulator*, vol. 202. Pisa, Italy: ACM, October 10 2006.

[158] M. Allman, "On the performance of middleboxes," in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*. Miami Beach, FL, USA: ACM Press, 2003, pp. 307–312.

[159] ns 2, "The Network Simulator 2 ns-2," 2006. [Online]. Available: http://nsnam.isi.edu/nsnam/index.php/Main_Page

[160] OPNET Technologies Inc, "Optimized Network Engineering Tools." July 1st 2006. [Online]. Available: http://www.opnet.com/

[161] S. D. Moitra and S. L. Konda, "An empirical investigation of network attacks on computer systems," *Computer and Security*, vol. 23, no. 1, pp. 43 – 51, 2004.

[162] J. Qian, S. Hinriichs, and K. Nahrstedt, "ACLA: A framework for Access Control List (ACL) Analysis and Optimization," in *Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security Issues of the New Century*, Darmstadt, Germany, May 21-22 2001.

[163] V. Paxson, J. Mahdavi, A. Adams, and M. Mathis, "Architecture for large-scale Internet measurement," *IEEE Communications Magazine*, vol. 36, no. 8, pp. 48–54, 1998.

[164] S. Sanfilippo, "Hping Security Tool," January 2005. [Online]. Available: http://www.hping.org/

[165] Hewlett-Packard, "Netperf," January 2005. [Online]. Available: http://www.netperf.org/netperf/NetperfPage.html

[166] Net-SNMP, "Net-SNMP," July, 17 2006. [Online]. Available: http://net-snmp.sourceforge.net/

[167] S. Convery and B. Trudel, "SAFE: A Security Blueprint for Enterprise Networks," Cisco Systems, Tech. Rep., 2008.

[168] M. Roughan, "Fundamental bounds on the accuracy of network performance measurements," in *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. Banff, Alberta, Canada: ACM Press, 2005, pp. 253–264.

[169] Cisco Systems, "Cisco 2600 Series Multiservice Platforms," July 1st 2006. [Online]. Available: http://www.cisco.com/en/US/products/hw/routers/ps259/index.html

[170] Cisco Systems, "Cisco SNMP Object Navigator," July 1st 2006. [Online]. Available: http://tools.cisco.com/Support/SNMP/do/BrowseMIB.do?local=en&step=2

[171] B. Schneier, "Inside the Twisted Mind of the Security Professional," April, 16 2008. [Online]. Available: http://www.wired.com/print/politics/security/commentary/securitymatters/2008/03/securitymatters_0320

[172] B. Schneier, *Beyond Fear: Thinking Sensibly About Security in an Uncertain World*. Springer, 2003.

[173] W. J. Buchanan, *The Complete Handbook of the Internet*. London: Kluwer Academic, 2002, vol. 2.

[174] H. Yan and R. Frenz, "An adaptive and scalable architecture for rule-based real-time analysis of network traffic," october 2004. [Online]. Available: www.ece.cmu.edu/~rfrenz/papers/yan_frenz2003.pdf

[175] N. Dean, *The Essence of Discrete Mathematics*. Prentice Hall PTR, 1996.

[176] GnuPlot, "GnuPlot version 4.2 patchlevel 2," September 2007. [Online]. Available: http://www.gnuplot.info/

[177] R. Bothne and J. Showalter, "A C# sets class," June 2002. [Online]. Available: http://www.codeproject.com/KB/recipes/set.aspx

[178] S. Fraser and S. Livingstone, *C# XML: Essential XML Skills for C# Programmers*, 1st ed. Birmingham, UK: Wrox Press, 2002.

[179] A. Jones, "A framework for the management of information security risks," *British Telecom Technology Journal*, vol. 25, no. 1, pp. 30–36, 2007.

[180] Cisco Systems, "Troubleshooting High CPU Utilization on Cisco Routers," July 1st 2006. [Online]. Available: http://www.cisco.com/en/US/products/hw/routers/ps133/products_tech_note09186a00800a70f2.shtml

[181] S. Hinrichs, "Policy-Based Management: Bridging the Gap," in *15th Annual Computer Security Applications Conference*. Scottsdale, Arizona, USA: IEEE Computer Society, December 6-10 1999, pp. 209 – 218.

[182] P. J. Denning, "IT Professional: Who Are We?" *Communications of the ACM*, vol. 44, no. 2, pp. 15 – 19, 2001.

[183] W. J. Buchanan and L. Saliou, "Enhanced Methods of Coursework Provision in Computer Networks," in *IEEE International Conference on Information Technology: Research and Education*, London Metropolitan University, London, United Kingdom, 28 June - 3 July 2004.

[184] CheckPoint, "Smart Defense Technical White Paper," CheckPoint Software Technologies Ltd, Tech. Rep., 2004.

[185] Qualys, "Operationalizing Security & Policy Compliance: A Unified Approach for IT, audit and Operation Teams," Qualys, Tech. Rep., August 2008.

[186] A. Bernstein, Philip and M. Haas, Laura, "Information integration in the enterprise," *Communications of the ACM*, vol. 51, no. 9, pp. 72–79, 2008.

[187] S. Sirajuddin and H. Sqalli, Mohammed, "Distributed XML-based network management using JPVM," *International Journal of Network Management*, vol. 16, no. 4, pp. 263–277, 2006.

[188] E. A. Larsson, "A case study: implementing Novell identity management at Drew University," in *33rd annual ACM SIGUCCS conference on User services*. Monterey, CA, USA: ACM, November 6-9 2005, pp. 165–170.

[189] M. T. Siponen and H. Oinas-Kukkonen, "A review of information security issues and respective research contributions," *SIGMIS Database*, vol. 38, no. 1, pp. 60–80, 2007.

[190] M. Bishop and S. Peisert, "Your Security Policy is What??" University California Davis, Technical CSE-2006-20, March 2006.

[191] L. Feng, E. Chang, and T. Dillon, "A semantic network-based design methodology for XML documents," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 390–421, 2002.

[192] B. E. Campbell, "Automating the Process of Network Documentation," Master's thesis, Edinburgh Napier University, 2007.

[193] J. Lumley, R. Gimson, and O. Rees, "A framework for structure, layout & function in documents," in *2005 ACM symposium on Document engineering*. Bristol, United Kingdom: ACM, November 2-4 2005, pp. 32–41.

List of Novelties

List of Observations

Appendix A

Evidence

A.1 Synopsis

This appendix contains all the tables and figures that form the basis of the analysis for the evaluation scenarios (Section 5.2).

A.2 Scenario 1: Performance Evaluation of Increasingly Large Rule-sets on Stateless Firewalls

This section contains the tables and figures related to the first evaluation scenario (Section 5.2.1, Section 5.2.2, and Section 6.2). This section is further divided amongst the parts that compose the scenario (Section 5.3.3).

A.2.1 Data related to the first part of Scenario 1

This section contains all the figures and tables on which the analysis of the impact on dynamic performance of increasingly large rule-sets on the Cisco DUT is based.

Table A.1 – Actual CPU usage, in %, readings of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	2.6	32.53	59.53	83.87	97.55	99
Cisco0017	0.4	32.2	62.1	86.44	88	-
Cisco0020	5.8	30.2	58.9	82.1	90.8	73
Cisco0036	5.9	30.8	58.7	82.3	87.5	85
Cisco0033	0.3	35.3	67.7	88.17	89.75	-
Cisco0049	0.9	52.5	92.43	-	-	-
Cisco0052	6.5	30.8	59.2	82.2	89.5	73

Table A.2 – Percentage difference between the model instances CPU usage readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Cisco0017	-84.62	-1.01	4.32	3.07	-9.79	-
Cisco0020	123.08	-7.16	-1.06	-2.11	-6.92	-26.26
Cisco0036	126.92	-5.32	-1.39	-1.87	-10.3	-14.14
Cisco0033	-88.46	8.52	13.72	5.12	-8	-
Cisco0049	-65.38	61.39	55.26	-	-	-
Cisco0052	150	-5.32	-0.55	-1.99	-8.25	-26.26

Table A.3 – CPU usage error-rate for each individual model instance.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0.27	0.67
Cisco0017	0	0	0	0.1	0.6	1
Cisco0020	0	0	0	0	0.5	0.9
Cisco0036	0	0	0	0	0.6	0.8
Cisco0033	0	0	0	0.4	0.6	1
Cisco0049	0	0	0.3	1	1	1
Cisco0052	0	0	0	0	0.6	0.9

Table A.4 – Percentage difference between the model instances Processor Memory readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Cisco0017	-0.69	-0.81	-0.99	-1.01	-0.95	-0.88
Cisco0020	-1.02	-1.08	-1.03	-1.04	-0.93	-0.87
Cisco0036	-3.57	-3.62	-3.55	-3.55	-3.48	-3.43
Cisco0033	-3.12	-3.35	-3.6	-3.58	-3.5	-
Cisco0049	-13.32	-13.57	-13.8	-13.8	-	-
Cisco0052	-13.76	-13.79	-13.75	-13.79	-13.69	-13.64

Table A.5 – Available memory measurements error-rate for each individual model instance

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0.27	0.67
Cisco0017	0	0	0	0	0.8	0.9
Cisco0020	0	0	0	0	0.8	0.7
Cisco0036	0	0	0	0	0.5	0.7
Cisco0033	0	0	0	0.3	0.9	1
Cisco0049	0	0	0.1	0.9	1	1
Cisco0052	0	0	0	0	0.6	0.7

Table A.6 – Actual latency readings, in milliseconds (ms), of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0.659	1.112	3.081	13.011	60.344	146.83
Cisco0017	0.69	1.33	5.293	31.486	136.432	174.959
Cisco0020	0.679	0.983	3.945	21.795	122.116	162.379
Cisco0036	0.706	1.181	4.316	20.35	121.459	197.385
Cisco0033	0.731	2.021	11.279	67.609	170.845	212.176
Cisco0049	0.816	10.792	127.935	315.456	-	-
Cisco0052	0.803	1.331	4.443	20.608	120.887	176.93

Table A.7 – Percentage difference between the model instances latency readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Cisco0017	4.732	19.621	71.81	141.999	126.091	19.158
Cisco0020	3.104	-11.642	28.052	67.514	102.367	10.59
Cisco0036	7.16	6.197	40.069	56.409	101.277	34.431
Cisco0033	10.939	81.753	266.085	419.629	183.119	44.504
Cisco0049	23.881	870.536	4052.381	2324.535	-	-
Cisco0052	21.837	19.702	44.19	58.385	100.33	20.5

Table A.8 – Latency across the DUT measurement error-rate.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0.056	0	0.002	0.006	0.035	0.281
Cisco0017	0	0	0	0.02	0.16	0.38
Cisco0020	0	0	0	0	0.12	0.35
Cisco0036	0	0	0	0	0.17	0.35
Cisco0033	0	0	0	0.02	0.28	0.53
Cisco0049	0	0	0.03	0.42	-	-
Cisco0052	0	0	0	0.01	0.21	0.35

Table A.9 – Actual available network throughput readings, in Mbps, of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	54.75	41.71	28.29	15.02	4.61	0.84
Cisco0017	52.58	36.62	21.63	8.27	0.95	0.19
Cisco0020	53.42	37.8	23.55	10.27	1.7	0.22
Cisco0036	46.71	33.83	20.89	9.33	1.49	0.16
Cisco0033	46.43	31.07	16.03	4.69	0.37	0.06
Cisco0049	33.5	16.14	2.6	0.04	-	-
Cisco0052	33.82	24.53	14.62	6.54	1.09	0.12

Table A.10 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Cisco0017	-3.97	-12.21	-23.53	-44.94	-79.33	-77.02
Cisco0020	-2.43	-9.38	-16.74	-31.6	-63.12	-73.93
Cisco0036	-14.68	-18.89	-26.15	-37.9	-67.79	-80.48
Cisco0033	-15.2	-25.5	-43.35	-68.8	-91.89	-93.25
Cisco0049	-38.81	-61.3	-90.81	-99.77	-	-
Cisco0052	-38.22	-41.19	-48.33	-56.48	-76.46	-86.31

Table A.11 – Available network throughput measurement error-rate

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0.13
Cisco0017	0	0	0	0	0	0
Cisco0020	0	0	0	0	0	0
Cisco0036	0	0	0	0	0	0
Cisco0033	0	0	0	0	0	0.1
Cisco0049	0	0	0	0.6	1	1
Cisco0052	0	0	0	0	0	0

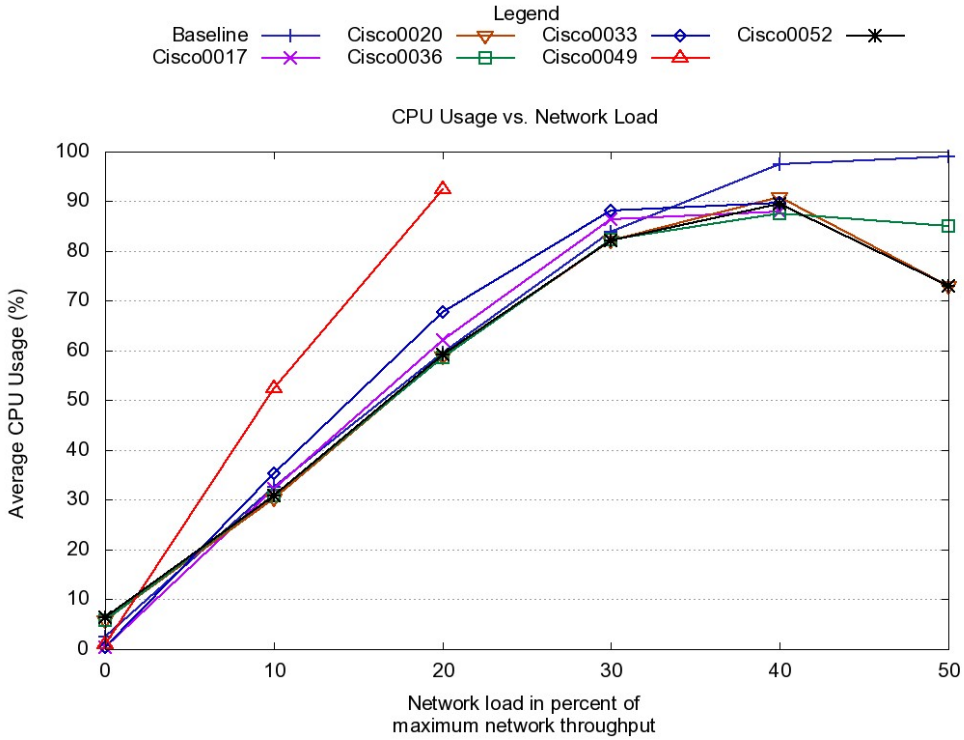

Figure A.1 – Effect of rule-set sizes on the DUT's CPU usage.

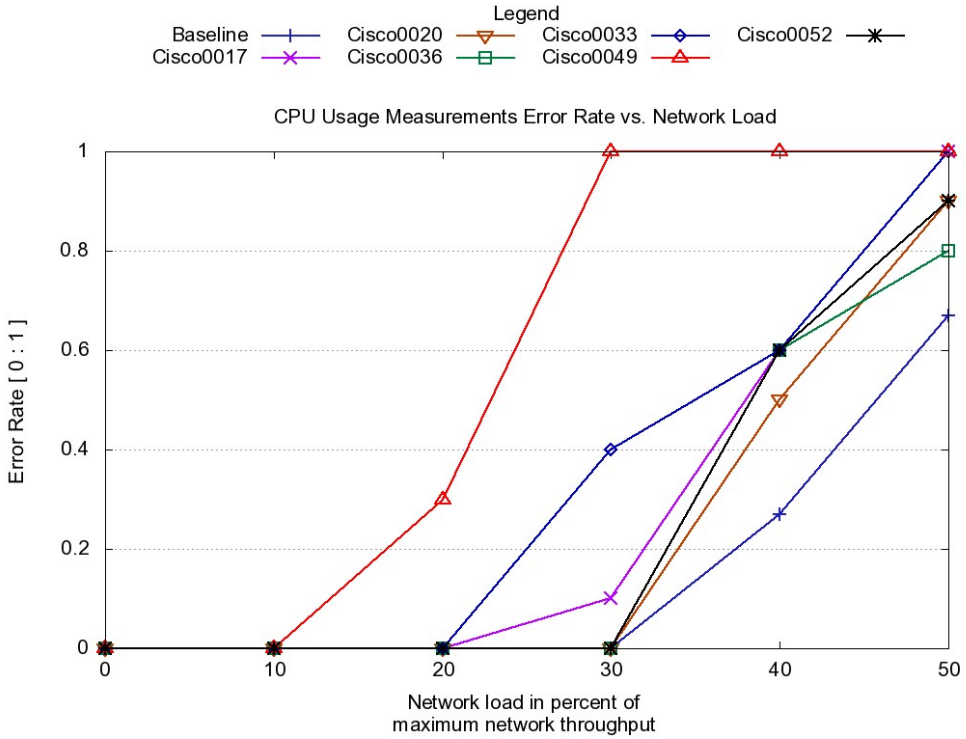

Figure A.2 – Effect of rule-set sizes on the CPU usage measurement error-rate.

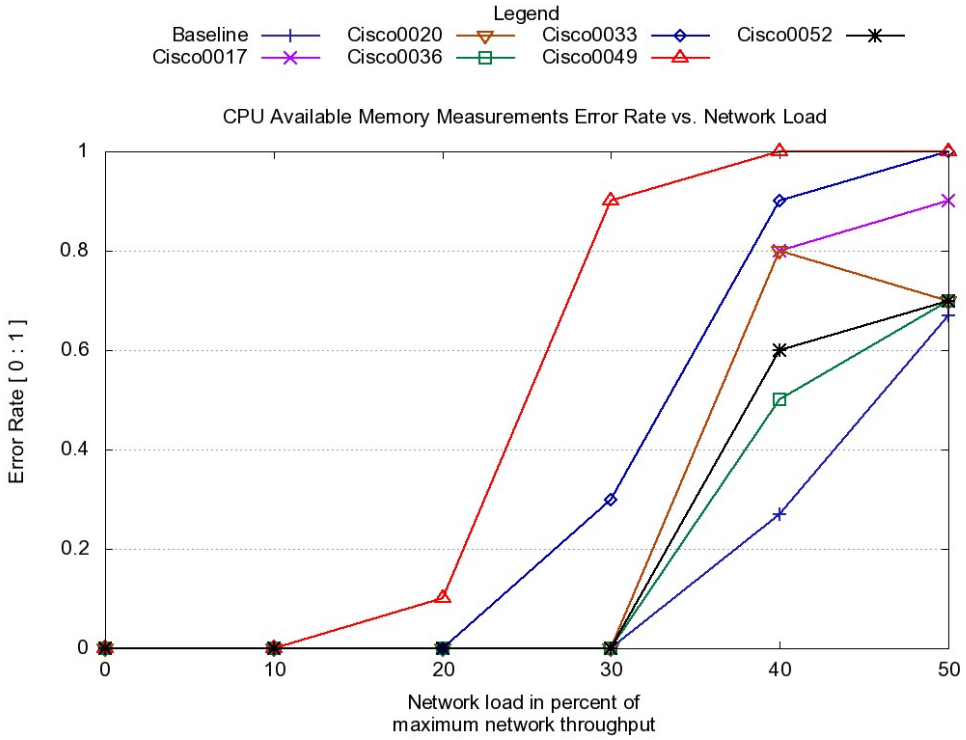

Figure A.3 – Effect of rule-set sizes on the Processor Memory measurements error-rate.

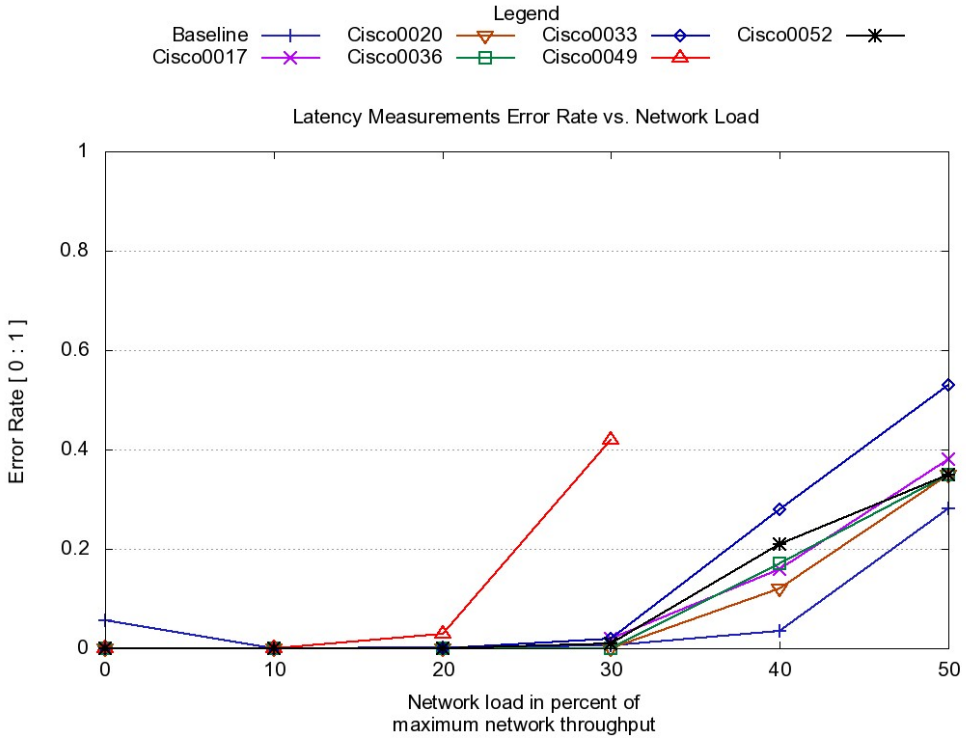

Figure A.4 – Effect of rule-set sizes on the latency measurements error-rate

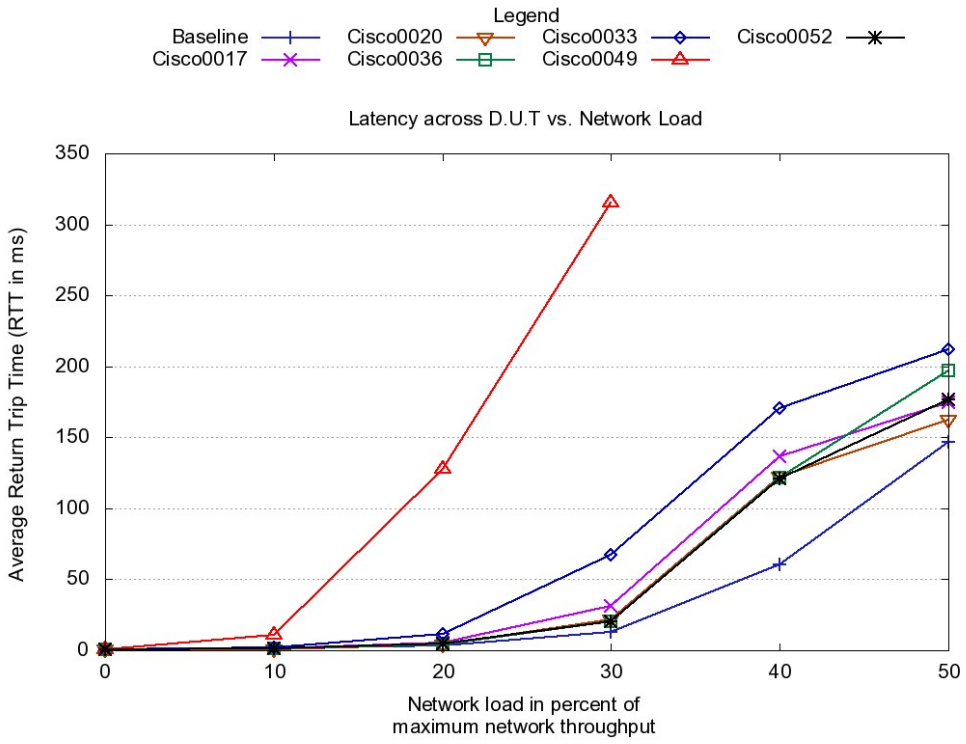

Figure A.5 – Effect of rule-set sizes on the latency across the DUT.

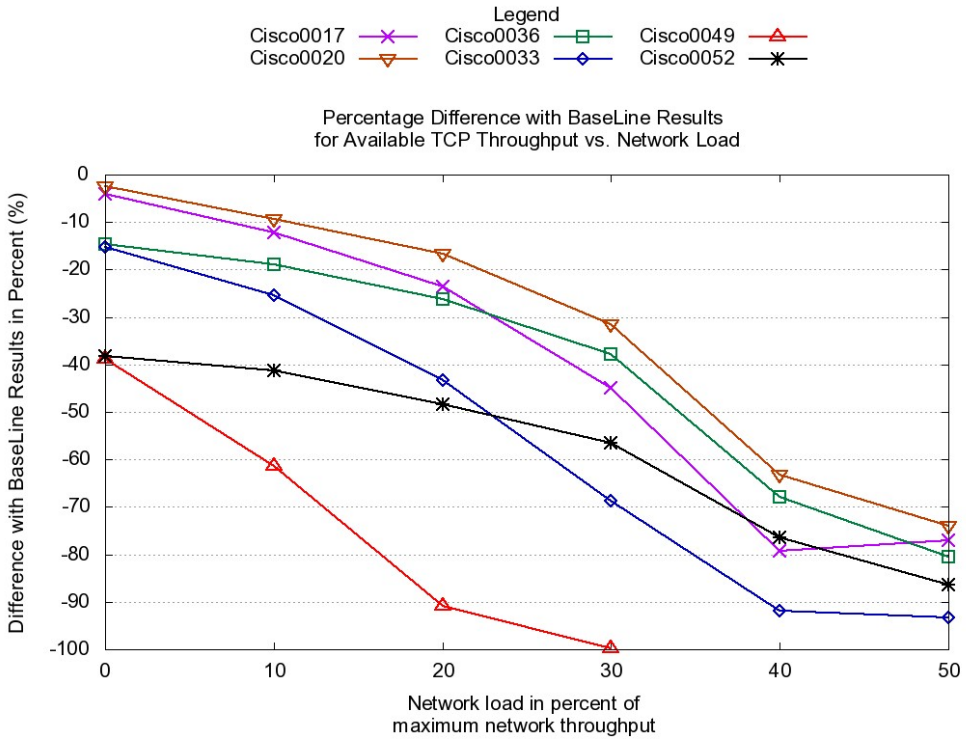

Figure A.6 – Effect of rule-set sizes on the available network throughput percentage difference between the measurements of the selected model instances and the baseline results.

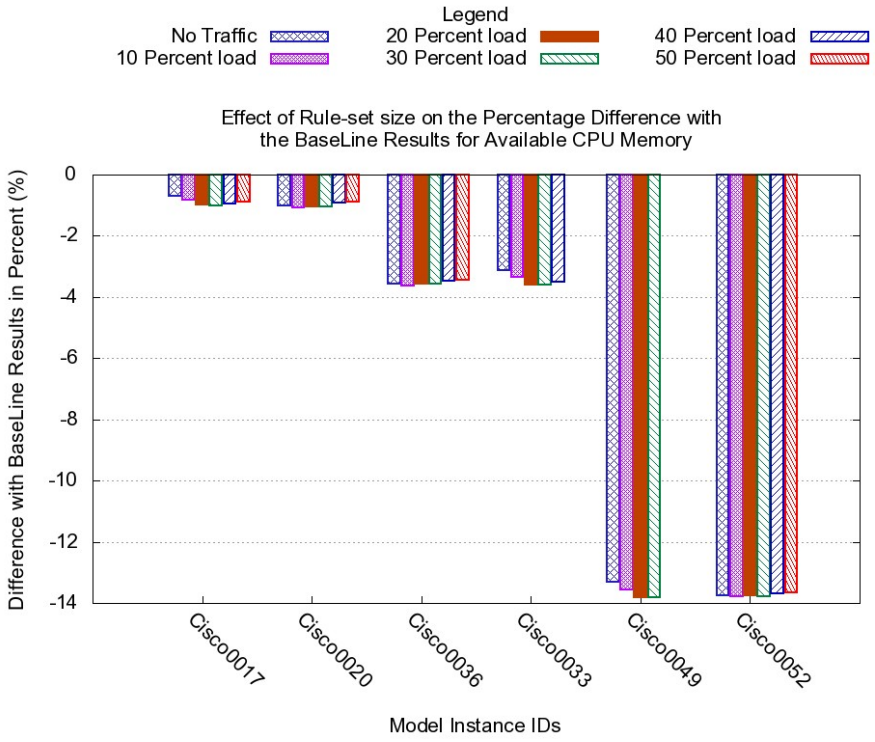

Figure A.7 – Transposition of the Process Memory readings for the selected model instances.

A.2.2 Data related to the second part of Scenario 1

This section contains all the figures and tables that are used for the analysis of the effect on dynamic performance of the direction parameter when using the Netfilter firewalling engine on a Linux device.

Table A.12 – Actual CPU usage readings for the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	99	99.09	99.18	99.26	99.36	99.45
Linux0105	99	99.44	99.86	99.97	99.99	-
Linux0003	99.01	99.44	99.85	99.97	99.99	-
Linux0103	99	99.44	99.86	99.97	99.99	-

Table A.13 – Percentage difference between the model instances CPU usage readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0105	0	0.36	0.69	0.71	0.63	-
Linux0003	0.01	0.36	0.68	0.71	0.63	-
Linux0103	0	0.36	0.68	0.71	0.63	-

Table A.14 – CPU usage error-rate for each individual model instance.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0
Linux0105	0	0	0	0.09	0.18	0.91
Linux0003	0	0	0	0.09	0.18	0.91
Linux0103	0	0	0	0	0.36	0.91

Table A.15 – Percentage difference between the model instances Processor Memory readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0105	-89.96	-89.5	-88.56	-89.16	-88.01	-
Linux0003	210.9	219.77	225.82	226.63	224.08	-
Linux0103	-83.3	-86.14	-87.13	-89.62	-90.38	-

Table A.16 – Available memory measurements error-rate for each individual model instance.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0
Linux0105	0	0	0	0	0.3	1
Linux0003	0	0	0	0	0.2	1
Linux0103	0	0	0	0	0.3	1

Table A.17 – Actual latency readings, in milliseconds (ms), of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0.297	0.32	0.473	0.695	0.913	1.65
Linux0105	0.88	3.376	77.263	228.923	255.311	279.431
Linux0003	0.627	3.2	72.485	226.775	247.187	277.512
Linux0103	0.778	3.946	77.951	228.251	257.018	279.253

Table A.18 – Percentage difference between the model instances latency readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0105	196.419	955.031	16234.714	32838.567	27863.975	16835.19
Linux0003	111.081	899.85	15224.528	32529.473	26974.175	16718.922
Linux0103	161.83	1133.251	16380.116	32741.812	28050.935	16824.422

Table A.19 – Latency across the DUT measurement error-rate.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0.002	0.005	0.013	0.013
Linux0105	0	0	0.01	0.14	0.29	0.45
Linux0003	0	0	0.01	0.14	0.31	0.42
Linux0103	0	0	0.01	0.15	0.27	0.45

Table A.20 – Actual available network throughput readings, in megabits per second (Mbps), of the selected model instances.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	87.99	75.7	57.95	65.55	57.67	49.34
Linux0105	40.33	20.76	6.14	1.84	0.59	0
Linux0003	38.06	22.5	6.39	1.88	0.63	0
Linux0103	37.57	22.13	6.16	1.84	0.58	0

Table A.21 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Linux0105	-54.16	-72.57	-89.4	-97.19	-98.98	-99.99
Linux0003	-56.74	-70.27	-88.97	-97.14	-98.9	-100
Linux0103	-57.3	-70.76	-89.38	-97.19	-99	-99.99

Table A.22 – Available network throughput measurement error-rate

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0
Linux0105	0	0	0	0	0	0.2
Linux0003	0	0	0	0	0	0
Linux0103	0	0	0	0	0	0.1

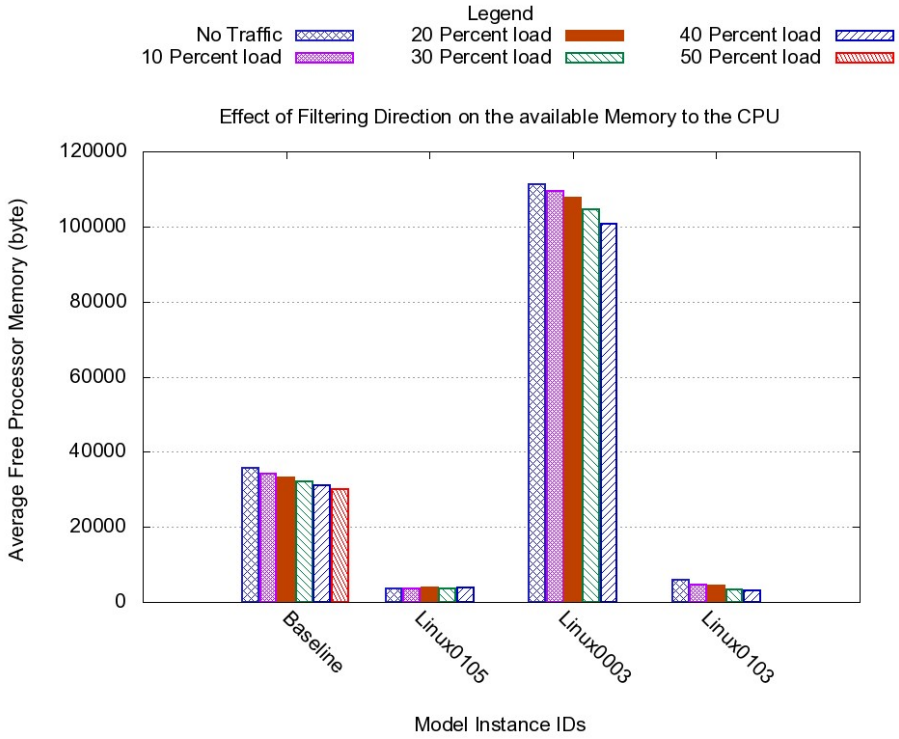

Figure A.8 – Effect of the direction option on the amount of memory available to the processor

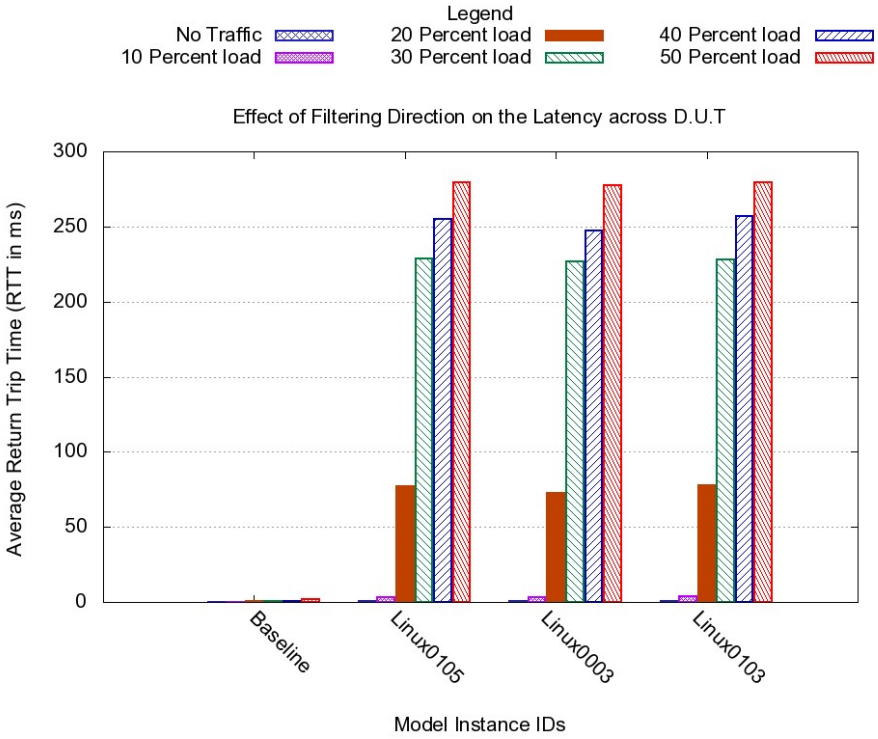

Figure A.9 – Effect on the direction option on the latency across the DUT.

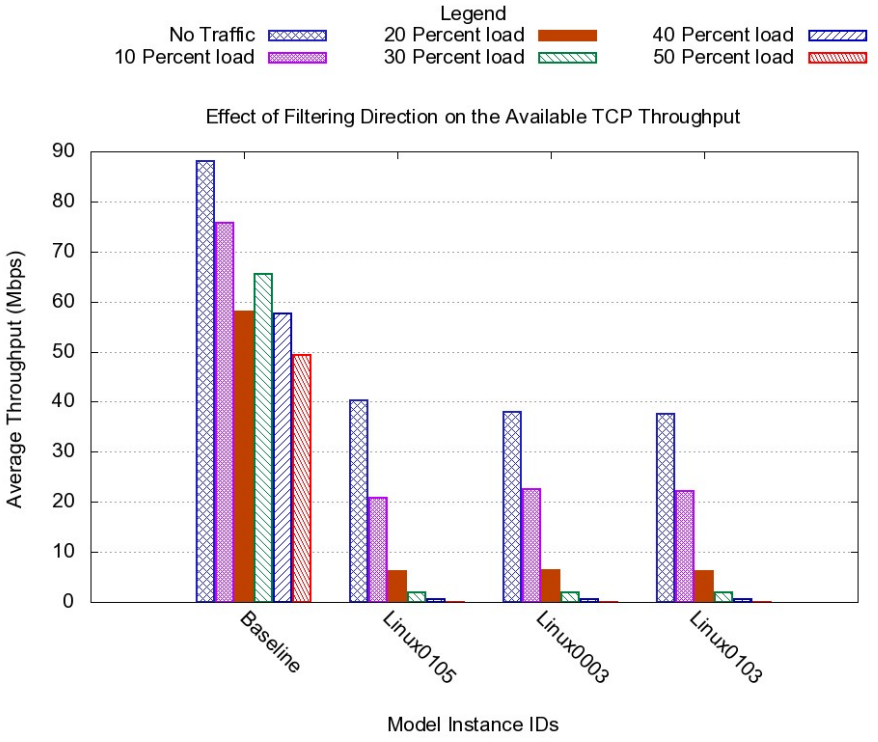

Figure A.10 – Effect of the direction option on available network throughput.

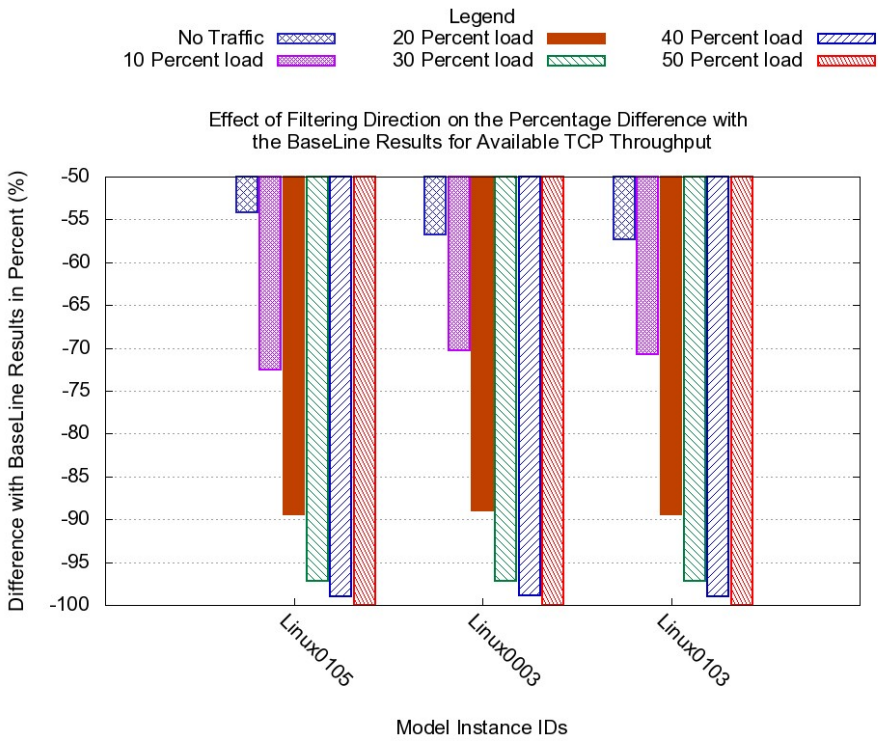

Figure A.11 – Effect of the direction option on the available network throughput percentage difference between the measurements of the selected model instances and the baseline results.

A.2.3 Data related to the third part of Scenario 1

This section contains all the figures and tables that are used for the analysis of the impact on dynamic performance of increasingly large rule-sets on a Linux device.

Table A.23 – Actual CPU usage, in %, readings of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	99	99.09	99.18	99.26	99.36	99.45
Linux0013	99	99.26	99.54	99.79	99.94	99.96
Linux0003	99.01	99.44	99.85	99.97	99.99	-
Linux0005	99	99.98	100	-	100	-
Linux0007	99	100	100	100	100	-

Table A.24 – Percentage difference between the model instances CPU usage readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0013	0	0.18	0.36	0.54	0.59	0.52
Linux0003	0.01	0.36	0.68	0.71	0.63	-
Linux0005	0	0.9	0.83	-	0.64	-
Linux0007	0	0.92	0.83	0.75	0.64	-

Table A.25 – CPU usage error-rate for each individual model instance.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0
Linux0013	0	0	0	0	0	0.27
Linux0003	0	0	0	0.09	0.18	0.91
Linux0005	0	0.09	0.64	0.82	0.73	0.91
Linux0007	0	0.45	0.36	0.64	0.73	0.82

Table A.26 – Percentage difference between the model instances Processor Memory readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0013	-86.11	-87.01	-87.9	-85.96	-87.24	-86.28
Linux0003	210.9	219.77	225.82	226.63	224.08	-
Linux0005	76.25	77.72	78.42	-	-	-
Linux0007	-55.52	-56.91	-55.68	-54.44	-53.59	-

Table A.27 – Available memory measurements error-rate for each individual model instance

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0
Linux0013	0	0	0	0	0	0.2
Linux0003	0	0	0	0	0.2	1
Linux0005	0	0.1	0.7	1	1	1
Linux0007	0	0.6	0.5	0.7	0.8	1

Table A.28 – Actual latency readings, in milliseconds (ms), of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0.297	0.32	0.473	0.695	0.913	1.65
Linux0013	0.574	0.979	4.405	29.221	109.99	148.108
Linux0003	0.627	3.2	72.485	226.775	247.187	277.512
Linux0005	1.907	811.667	899.771	963.433	1039.209	1166.107
Linux0007	5.475	3962.45	4133.921	4563.316	4875.956	5403.529

Table A.29 – Percentage difference between the model instances latency readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0013	93.144	205.852	831.306	4104.471	11947.081	8876.238
Linux0003	111.081	899.85	15224.528	32529.473	26974.175	16718.922
Linux0005	542.057	253545.932	190126.344	138523.494	113723.545	70573.176
Linux0007	1743.343	1238165.506	873879.149	656492.286	533958.671	327386.583

Table A.30 – Latency across the DUT measurement error-rate.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0.002	0.005	0.013	0.013
Linux0013	0	0	0	0.01	0.09	0.2
Linux0003	0	0	0.01	0.14	0.31	0.42
Linux0005	0	0.06	0.25	0.43	0.54	0.67
Linux0007	0	0.34	0.54	0.65	0.73	0.82

Table A.31 – Actual available network throughput readings, in megabits per second (Mbps), of the selected model instances.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	87.99	75.7	57.95	65.55	57.67	49.34
Linux0013	55.18	39.67	24.07	11.94	4.41	2.22
Linux0003	38.06	22.5	6.39	1.88	0.63	0
Linux0005	11.92	0.46	0.02	0	0	-
Linux0007	2.83	0	0	0	-	0

Table A.32 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Linux0013	-37.29	-47.6	-58.47	-81.79	-92.35	-95.51
Linux0003	-56.74	-70.27	-88.97	-97.14	-98.9	-100
Linux0005	-86.46	-99.39	-99.97	-	-	-
Linux0007	-96.78	-	-	-	-	-

Table A.33 – Available network throughput measurement error-rate.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0	0
Linux0003	0	0	0	0	0	0
Linux0013	0	0	0	0	0	0
Linux0005	0	0	0	0.1	0.1	1
Linux0007	0	0.2	0.4	0.8	1	.9

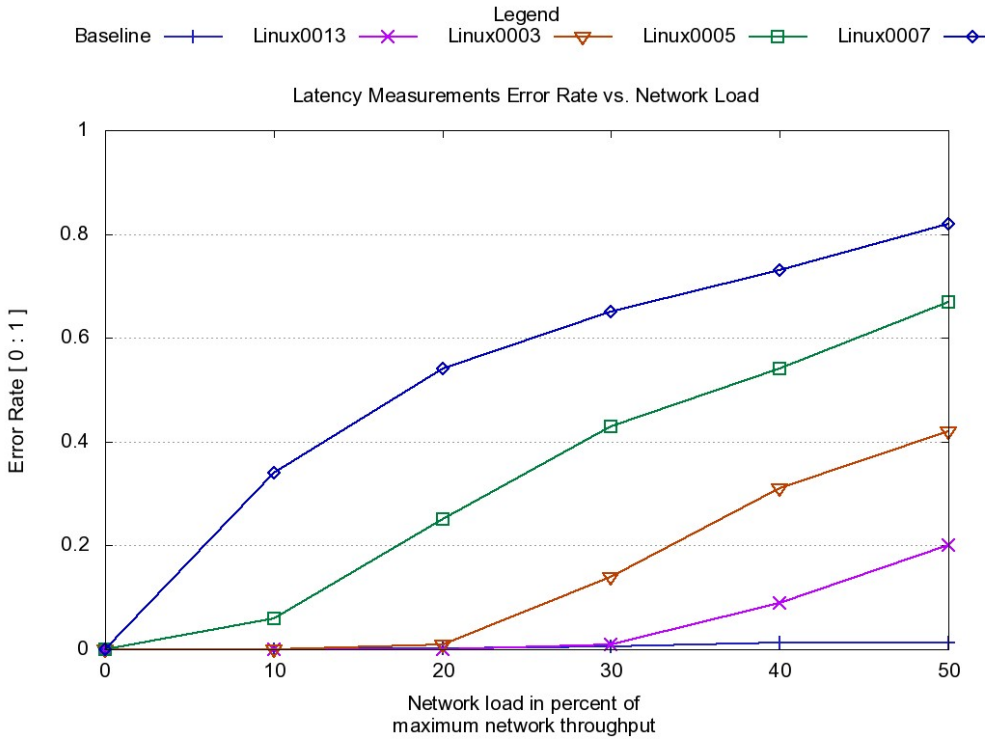

Figure A.12 – Effect of increasingly large rule-sets on the latency measurements error-rate.

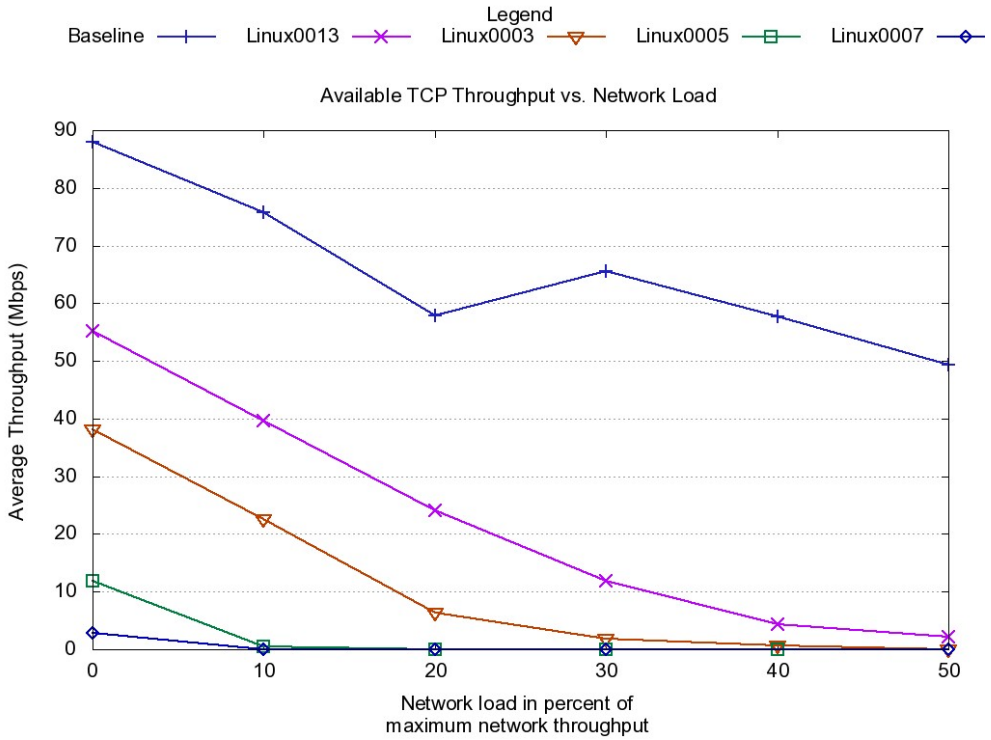

Figure A.13 – Effect of increasingly large rule-sets on the available network throughput.

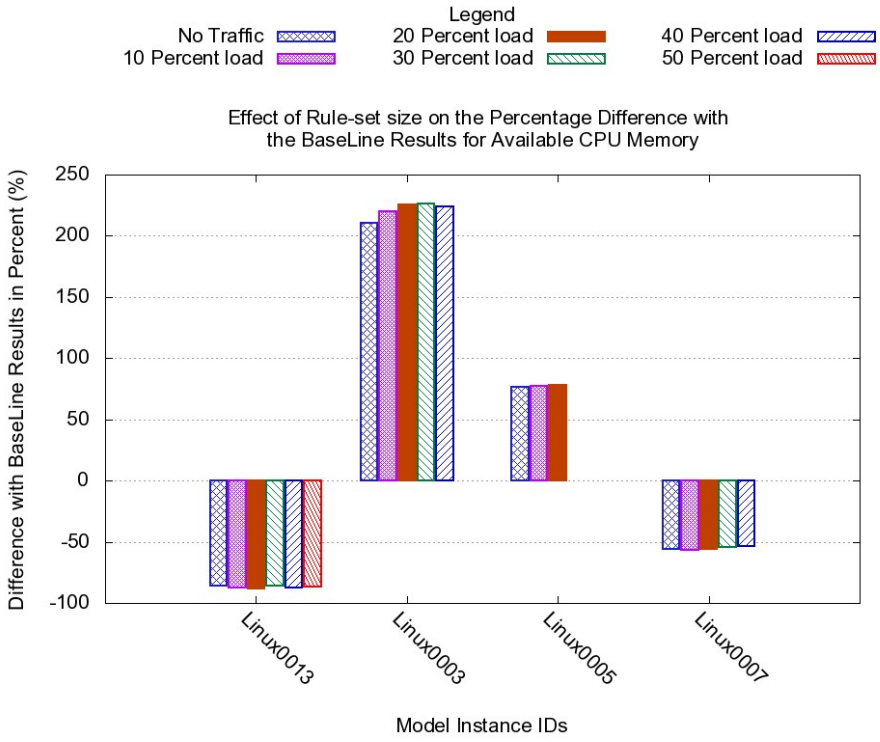

Figure A.14 – Effect of increasingly large rule-set on the available memory to the processor.

A.3 Scenario 2: Performance Evaluation of Critical Rule Positioning

This section contains the tables and figures related to the second evaluation scenario (Section 5.2.3, and Section 6.3). This section is further divided amongst the parts that compose the scenario (Section 5.3.3).

A.3.1 Data related to the first part of the Scenario 2

This section contains all the figures and tables on which the analysis, for the Cisco DUT, of the impact on dynamic performance of critical rule repositioning is based.

Table A.34 – Percentage difference between the model instances CPU usage readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Cisco0036	126.92	-5.32	-1.39	-1.87	-10.3	-14.14
Cisco0033	-88.46	8.52	13.72	5.12	-8	-
Cisco0125	-50	-8.39	2.47	2.78	-2.36	0
Cisco0128	246.15	-14.23	-5.59	-1.4	-3.13	-1.68

Table A.35 – CPU usage error-rate for each individual model instance.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0.27	0.67
Cisco0036	0	0	0	0	0.6	0.8
Cisco0033	0	0	0	0.4	0.6	1
Cisco0125	0	0	0	0	0.6	0.9
Cisco0128	0	0	0	0	0.4	0.7

Table A.36 – Percentage difference between the model instances Processor Memory readings and the baseline results.

Model Instance ID	Network load 0%	10%	20%	30%	40%	50%
Cisco0036	-3.57	-3.62	-3.55	-3.55	-3.48	-3.43
Cisco0033	-3.12	-3.35	-3.6	-3.58	-3.5	-
Cisco0125	-0.5	-0.63	-0.89	-0.92	-0.77	-0.71
Cisco0128	-0.85	-0.67	-0.95	-0.93	-0.76	-0.7

Table A.37 – Available memory measurements error-rate for each individual model instance.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	0	0	0	0	0.27	0.67
Cisco0036	0	0	0	0	0.5	0.7
Cisco0033	0	0	0	0.3	0.9	1
Cisco0125	0	0	0	0	0.4	0.9
Cisco0128	0	0	0	0	0.6	0.9

Table A.38 – Percentage difference between the model instances latency readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Cisco0036	7.16	6.197	40.069	56.409	101.277	34.431
Cisco0033	10.939	81.753	266.085	419.629	183.119	44.504
Cisco0125	-6.829	4.971	84.989	147.386	143.72	26.164
Cisco0128	-10.001	-4.202	43.095	75.56	88.206	16.291

Table A.39 – Latency across the DUT measurement error-rate.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	0.056	0	0.002	0.006	0.035	0.281
Cisco0036	0	0	0	0	0.17	0.35
Cisco0033	0	0	0	0.02	0.28	0.53
Cisco0125	0	0	0	0	0.23	0.34
Cisco0128	0	0	0	0.01	0.12	0.36

Table A.40 – Actual available network throughput readings, in megabits per second (Mbps), of the selected model instances.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	54.75	41.71	28.29	15.02	4.61	0.84
Cisco0036	46.71	33.83	20.89	9.33	1.49	0.16
Cisco0033	46.43	31.07	16.03	4.69	0.37	0.06
Cisco0125	49.92	34.23	18.8	6.17	0.77	0.06
Cisco0128	46.16	29.6	20.71	9.07	1.21	0.07

Table A.41 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Cisco0036	-14.68	-18.89	-26.15	-37.9	-67.79	-80.48
Cisco0033	-15.2	-25.5	-43.35	-68.8	-91.89	-93.25
Cisco0125	-8.81	-17.95	-33.53	-58.94	-83.41	-93.1
Cisco0128	-15.7	-29.04	-26.8	-39.63	-73.73	-91.8

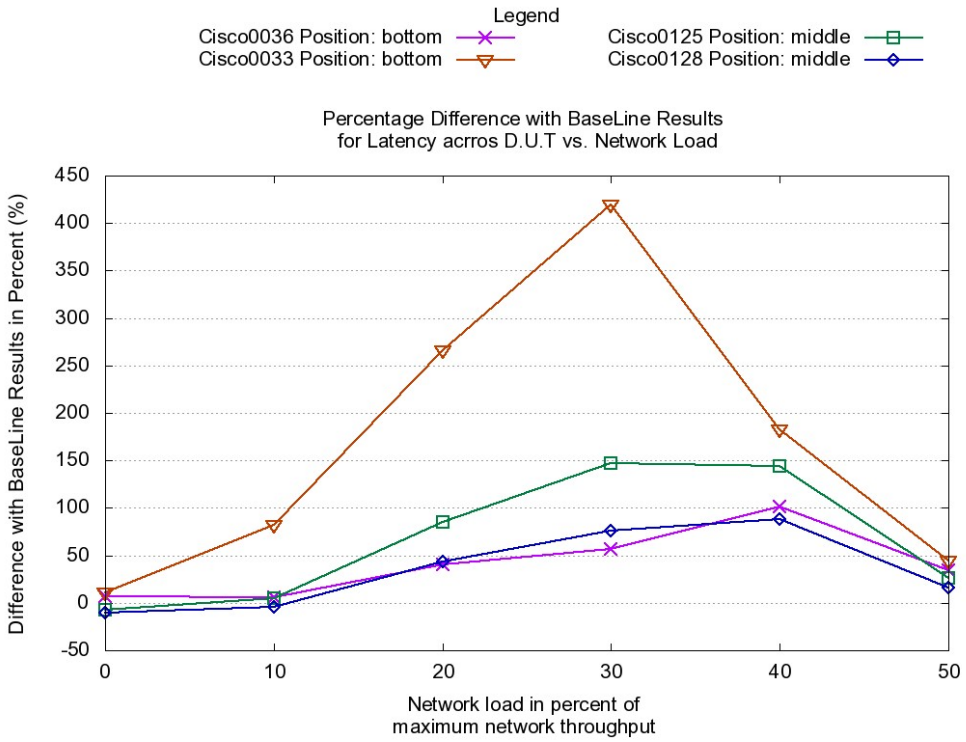

Figure A.15 – Effect of the critical rule positioning on percentage difference between the measurements of the latency across the DUT for the selected model instances and the baseline results.

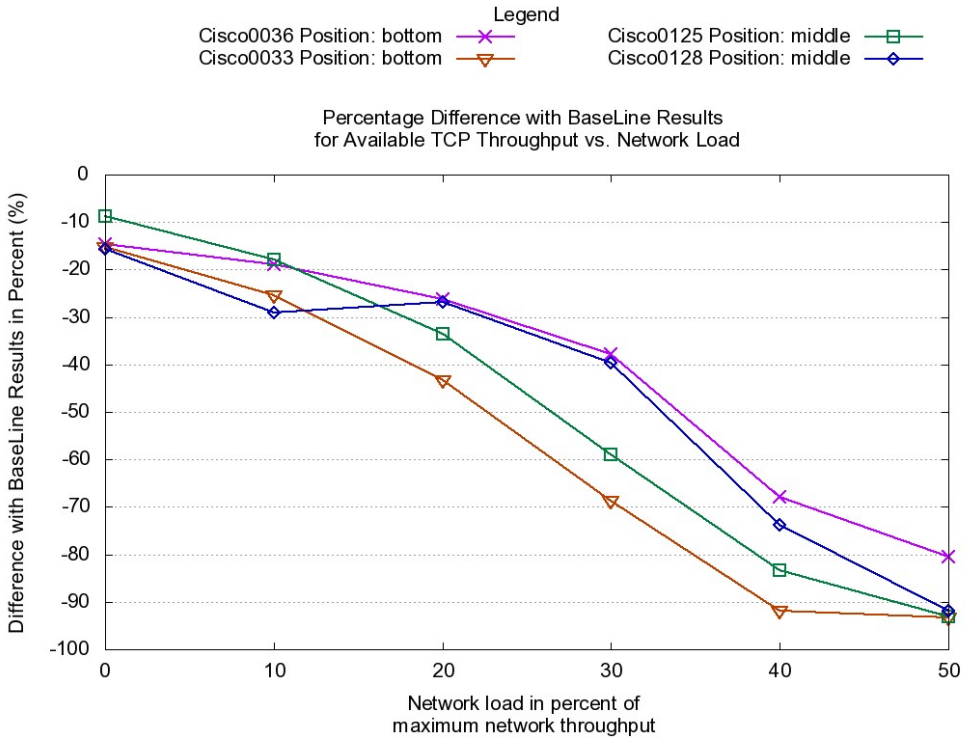

Figure A.16 – Effect of the critical rule position on the available network throughput percentage difference between the measurements of the selected model instances and the baseline results.

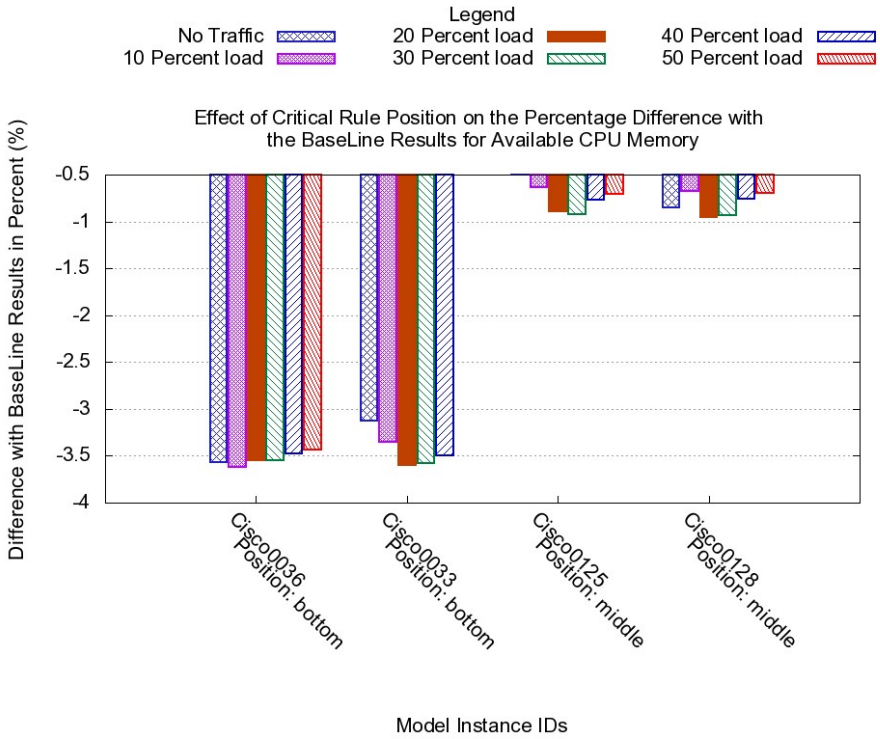

Figure A.17 – Effect of the critical rule position on the amount of memory available to the processor.

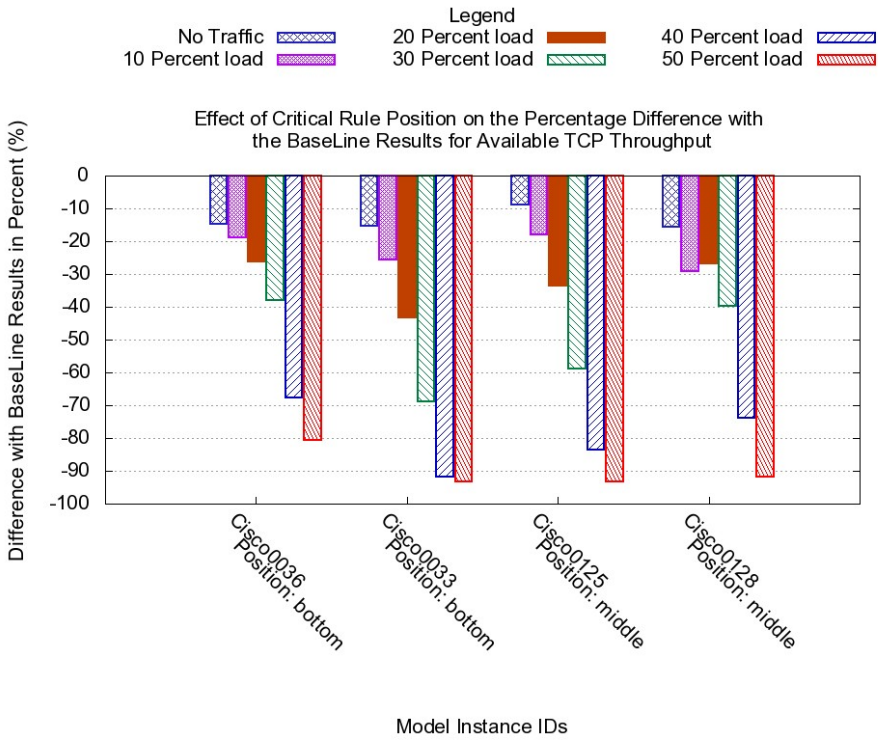

Figure A.18 – Transposed representation of the effect of the critical rule position on the available network throughput percentage difference between the measurements of the selected model instances and the baseline results.

A.3.2 Data related to the second part of Scenario 2

This section contains all the figures and tables on which the analysis, for the Linux DUT, of the impact on dynamic performance of critical rule repositioning is based.

Table A.42 – Percentage difference between the model instances Processor Memory readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Linux0003	210.9	219.77	225.82	226.63	224.08	-
Linux0211	-86.05	-87.82	-86.54	-87.97	-88.37	-

Table A.43 – Actual latency readings, in milliseconds (ms), of the selected model instances.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	0.297	0.32	0.473	0.695	0.913	1.65
Linux0003	0.627	3.2	72.485	226.775	247.187	277.512
Linux0211	0.781	3.652	76.154	227.989	252.364	270.43

Table A.44 – Percentage difference between the model instances latency readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Linux0003	111.081	899.85	15224.528	32529.473	26974.175	16718.922
Linux0211	162.81	1041.146	16000.141	32704.154	27541.146	16289.693

Table A.45 – Actual available network throughput readings, in megabits per second (Mbps), of the selected model instances.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	87.99	75.7	57.95	65.55	57.67	49.34
Linux0003	38.06	22.5	6.39	1.88	0.63	0
Linux0211	36.88	22.14	6.19	1.81	0.6	0.01

Table A.46 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Linux0003	-56.74	-70.27	-88.97	-97.14	-98.9	-100
Linux0211	-58.09	-70.75	-89.31	-97.23	-98.95	-99.99

A.4 Scenario 3: Performance Evaluation of Fine-grained Filtering on Stateless Firewalls

This section contains the tables and figures related to the third evaluation scenario (Section 5.2.4, and Section 6.4). This section is further divided amongst the parts that compose the scenario (Section 5.3.3).

A.4.1 Data related to the first part of the Scenario 3

This section contains all the figures and tables on which the dynamic performance analysis of firewalling statements complexity on the Cisco DUT is based.

Table A.47 – CPU usage error-rate for model instances based on fast network settings.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Cisco1001	0	0.9	1	1	1	1
Cisco00004	0	0	0	0	0.6	0.6
Cisco00001	0	0	0	0	0.6	0.8
Cisco1002	0	0	0	0	0.6	0.9

Table A.48 – Actual latency readings, in milliseconds (ms), of the model instances based on fast network settings.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Cisco1001	2.088	-	-	-	-	-
Cisco00004	0.672	1.07	3.802	21.866	114.464	171.425
Cisco00001	0.704	1.195	5.973	30.566	162.46	183.352
Cisco1002	2.08	3.017	6.033	23.67	112.637	195.783

Table A.49 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Cisco1001	-88.07	-	-	-	-	-
Cisco00004	0.64	-7.43	-14.98	-29.86	-59.5	-66.67
Cisco00001	-4.78	-11.89	-21.53	-41.82	-76.05	-89.14
Cisco1002	-88.05	-89.45	-90.66	-91.61	-94.75	-99.4

Table A.50 – Actual CPU usage reading for model instances based on slow network settings.

Model Instance ID	Network load							
	0%	10%	20%	30%	40%	50%	60%	70%
Baseline	2.6	5.8	8.67	11.6	14.53	16.76	20.4	22.93
Cisco00005	5.7	8.1	11.8	15.5	18	19.9	24.7	26.5
Cisco01003	6.8	31.3	57.4	75.17	80	76	60	-
Cisco00008	6.8	7.8	11.1	14.5	17.3	18.9	23.5	25.1
Cisco01004	4.7	7.6	9.8	11.6	16	18.5	21.7	24.6

Table A.51 – Cisco1003 error-rate for the CPU usage measurements. The other model instances do not have any missing measurements

Model Instance ID	Network load							
	0%	10%	20%	30%	40%	50%	60%	70%
Cisco1003	0	0	0	0.4	0.8	0.9	0.9	1

Table A.52 – Percentage difference between the model instances latency readings and the baseline results.

Model Instance ID	Network load							
	0%	10%	20%	30%	40%	50%	60%	70%
Cisco00005	2.524	2.778	6.273	0.103	7.653	4.12	6.287	4.636
Cisco01003	123.1	1135.9	15575	40276.3	80435.9	97879.5	91837.8	-
Cisco00008	1.559	1.173	5.703	0.804	3.194	0.912	5.482	1.297
Cisco01004	121.9	124.7	118.6	109.2	111.2	101.7	99.4	91.3

Table A.53 – Percentage difference between the model instances available network throughput readings and the baseline results.

Model Instance ID	Network load							
	0%	10%	20%	30%	40%	50%	60%	70%
Cisco00005	0	-0.08	0.37	-1.38	0.47	-1.16	2.32	0.44
Cisco01003	-31.84	-49.96	-70.28	-89.88	-98.84	-99.93	0	-
Cisco00008	0	-0.09	0.51	-1.21	0.67	-0.82	1.59	0.18
Cisco01004	-31.74	-30.57	-34.29	-29.61	-29.84	-31.51	-32.31	-33.61

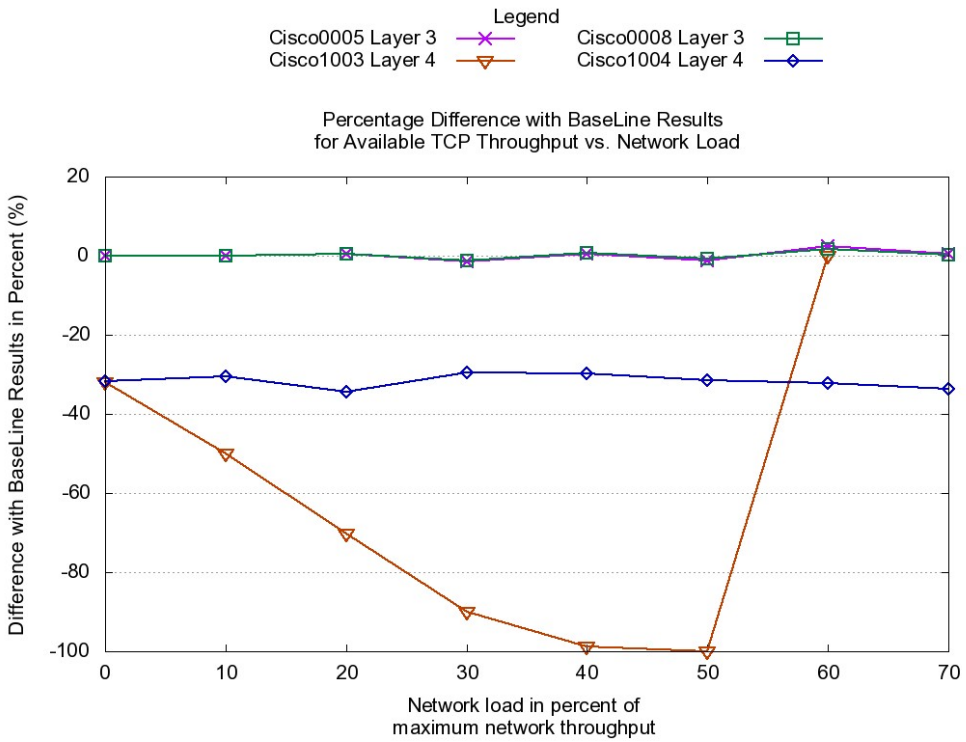

Figure A.19 – Effect of the firewalling statement complexity on the available network throughput percentage difference between the measurements of the selected model instances and the baseline results.

A.4.2 Data related to the second part of Sceanio 3

This section contains all the figures and tables on which the dynamic performance analysis of firewalling statements complexity on the Linux DUT is based.

Table A.54 – Actual latency readings, in milliseconds (ms), of the selected model instances.

Model Instance ID	Network load					
	0%	10%	20%	30%	40%	50%
Baseline	0.297	0.32	0.473	0.695	0.913	1.65
Linux1011	0.541	0.931	3.827	26.829	102.037	143.696
Linux0013	0.574	0.979	4.405	29.221	109.99	148.108
Linux1001	0.751	3.279	69.33	224.785	249.714	269.97
Linux0003	0.627	3.2	72.485	226.775	247.187	277.512

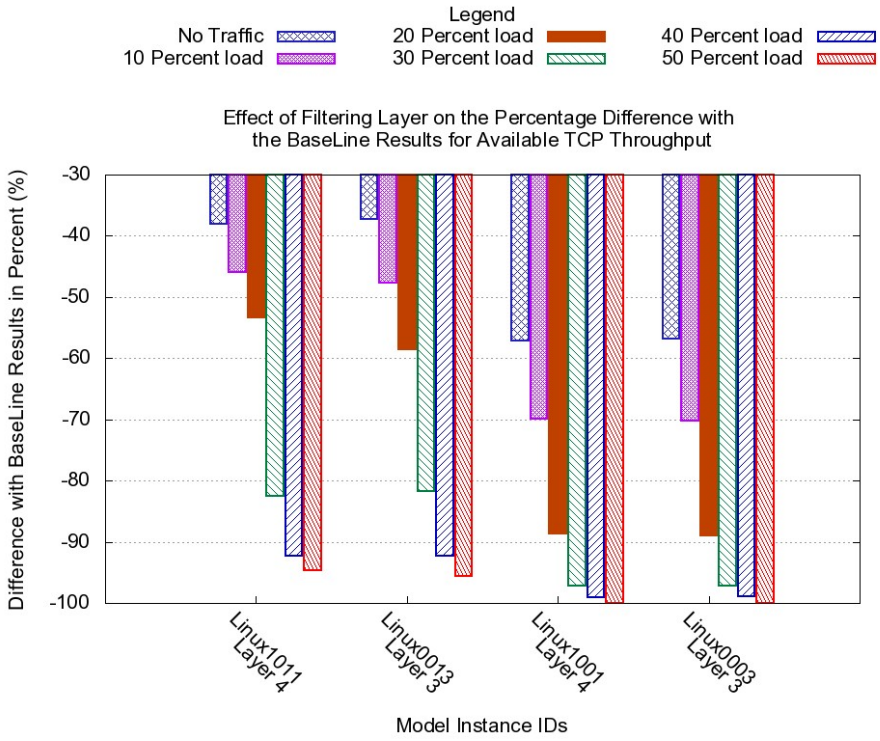

Figure A.20 – Effect of the firewalling statement complexity on the percentage difference between the readings for each selected model instance and the baseline results in terms of available network throughput.

Appendix B

Baseline Readings

B.1 Synopsis

This appendix contains the totality of the readings collected for the baseline evaluations for both Cisco 2600 XM, and Ubuntu Linux Netfilter. This section is thus organised as follows; First of all, the readings for fast network settings are listed for the Cisco firewall, followed by the associated error-rate. This structure is then repeated for the slow network settings. The readings for the Ubuntu Linux firewall are listed in the same fashion.

B.2 Cisco readings for Fast network settings

This section lists the readings for the Cisco 2600 XM device gathered when the network settings were set to 100 Mbps throughput. The readings are divided between the device oriented metrics typically obtained via the SNMP, and the network oriented metrics. For each metric type, the table of the associated error-rate is provided.

B.2.1 Device centric metrics

Table B.1 – Error-rate associated to device centric metrics for fast network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%
Free Processor Memory	0	0	0	0	0.27	0.67
CPU Usage	0	0	0	0	0.27	0.67

Table B.2 – Readings for the device centric metrics on fast network throughput settings.

Metric's Name	Unit	Network load					
		0%	10%	20%	30%	40%	50%
Free Processor Memory	byte	38555387.47	38573772.8	38566448.27	38561005.07	38518619.64	38495520
CPU Usage	%	2.6	32.53	59.53	83.87	97.55	99

B.2.2 Network oriented metrics

Table B.3 – Readings for the network oriented metrics on fast network throughput settings.

Metric's Name	Unit	Network load 0%	10%	20%	30%	40%	50%
Latency	ms	0.659	1.112	3.081	13.011	60.344	146.83
TCP Stream	Mbps	54.75	41.71	28.29	15.02	4.61	0.84

Table B.4 – Error-rate associated to network oriented metrics on fast network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%
Latency	0.056	0	0.002	0.006	0.035	0.281
TCP Stream	0	0	0	0	0	0.13

B.3 Cisco readings for Slow network settings

This section contains the readings gathered for the Cisco 2600 XM for slow network throughput, and is organised in a similar fashion as Section B.2.

B.3.1 Device centric metrics

Table B.5 – Error-rate associated to device centric metrics for slow network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%	60%	70%
Free Processor	0	0	0	0	0	0	0	0
Memory CPU Usage	0	0	0	0	0	0.06	0	0

Table B.6 – Readings for the device centric metrics on slow network throughput settings.

Metric's Name	Unit	Network load							
		0%	10%	20%	30%	40%	50%	60%	70%
Free Processor Memory	byte	38493805.87	38494253.6	38492788.27	38648217.07	38631263.47	38624468.53	38618136.27	38612921.07
CPU Usage	%	2.6	5.8	8.67	11.6	14.53	16.76	20.4	22.93

B.3.2 Network oriented metrics

Table B.7 – Error-rate associated to network oriented metrics on slow network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%	60%	70%
Latency	0.039	0	0	0	0	0	0	0.004
TCP Stream	0	0	0	0	0	0	0	0

Table B.8 – Readings for the network oriented metrics on slow network throughput settings.

Metric's Name	Unit	Network load 0%	10%	20%	30%	40%	50%	60%	70%
Latency	ms.	1.149	1.155	1.213	1.323	1.366	1.485	1.582	1.71
TCP Stream	Mbps	9.39	9.02	8.68	8.42	8.03	7.76	7.41	7.09

B.4 Linux readings for Fast network settings

This section contains the readings for the Linux Ubuntu `iptable` system gathered when the network settings were set to 100 Mbps throughput. The readings are divided between the device oriented metrics typically obtained via SNMP, and the network oriented metrics. For each metric type, the table of the associated error-rate is provided.

B.4.1 Device centric metrics

Table B.9 – Error-rate associated to device centric metrics for fast network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%
Free Memory	0	0	0	0	0	0
Processor CPU Usage	0	0	0	0	0	0

Table B.10 – Readings for the device centric metrics on fast network throughput settings.

Metric's Name	Unit	Network load 0%	10%	20%	30%	40%	50%
Free Processor Memory	kilobyte	35826	34236	33058.4	32034	31104	30162
CPU Usage	%	99	99.09	99.18	99.26	99.36	99.45

B.4.2 Network oriented metrics

Table B.11 – Readings for the network oriented metrics on fast network throughput settings.

Metric's Name	Unit	Network load 0%	10%	20%	30%	40%	50%
Latency	ms	0.297	0.32	0.473	0.695	0.913	1.65
TCP Stream	Mbps	87.99	75.7	57.95	65.55	57.67	49.34

Table B.12 – Error-rate associated to network oriented metrics on fast network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%
Latency	0	0	0.002	0.005	0.013	0.013
TCP Stream	0	0	0	0	0	0

B.5 Linux readings for Slow network settings

B.5.1 Network oriented metrics

Table B.13 – Error-rate associated to network oriented metrics on slow network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%	60%	70%
Latency	0	0	0	0	0	0	0	0
TCP Stream	0	0	0	0	0	0	0	0

Table B.14 – Readings for the network oriented metrics on slow network throughput settings.

Metric's Name	Unit	Network load 0%	10%	20%	30%	40%	50%	60%	70%
Latency	ms	0.74	0.753	0.845	0.874	0.964	1.056	1.115	1.229
TCP Stream	Mbps	9.39	9.03	8.79	8.39	8.02	7.67	7.37	7.04

B.5.2 Device centric metrics

Table B.15 – Error-rate associated to device centric metrics for slow network throughput settings.

Metric's Name	Network load 0%	10%	20%	30%	40%	50%	60%	70%
Free Processor Memory	0	0	0	0	0	0	0	0
CPU Usage	0	0	0	0	0	0	0	0

Table B.16 – Readings for the device centric metrics on slow network throughput settings.

Metric's Name	Unit	Network load							
		0%	10%	20%	30%	40%	50%	60%	70%
Free Processor Memory	kilobyte	128846	127652	126926	126074	125324	124658	123764	122996
CPU Usage	%	99	99.01	99.02	99.03	99.04	99.05	99.06	99.07

Appendix C

List of equipment used during the experiments

C.1 Synopsis

This appendix details of the hardware equipment, and software applications that were used to create the dynamic evaluation environment, and thus perform the various measurements.

C.2 Network Devices

This section focuses on the intermediate network devices employed during this research work. These are thus the devices evaluated and network switches.

C.2.1 D.U.T 1: Cisco Device

- **Hardware**: Cisco c2600MX series .
- **Operating System**: Cisco IOS 12.3 (27), release software fc3.

C.2.2 D.U.T 2: Linux netfilter

- **Hardware**:
  - Processor: Intel Pentium III 700MHz.
  - RAM: 256MB.
  - Network Cards: 3 Com 10/100Mbps compatible Ethernet cards.
- **Operating System**: Debian Ubuntu Linux 6.06 Dapper Drake.
- **Routing software**: quagga 0.99.2-1ubuntu3.
- **Firewall engine**: netfilter.

C.2.3 Network Switches

- **Hardware**: Cisco Catalyst 2950 series.
- **Operating System**: Cisco IOS 12.1(13) EA1a, release software fc1.

C.3 Network Hosts

This section focuses on the computer hosts employed to either produce the network traffic, or measure the different metrics.

C.3.1 Traffic Generation Node

- **Hardware**:
  - Processor: Intel Pentium IV 2.4Ghz.
  - RAM: 512MB.
  - Network Cards: 3 Com 10/100/1000Mbps compatible Ethernet cards
- **Operating System**: Debian Ubuntu Linux 6.06 Dapper Drake.
- **Replay software**: TCPReplay version 2.2.0.
- **Network traces**: DARPA trace from the 1^{st} Monday of the 1^{st} week of the year 1998.

C.3.2 Measurement Nodes

All the computer hosts used to perform the different measurements had the same hardware characteristic which are as follows:

- **Hardware**:
  - Processor: Intel Pentium III 700MHz.
  - RAM: 256MB.
  - Network Card: 3Com 10/100Mbps compatible Ethernet.
- **Operating System**: Debian Ubuntu Linux 6.06 Dapper Drake.
- **Latency measurement software**: Hping version 2.0.0r3.
- **Available throughput measurement software**: Netperf version 4a.
- **SNMP software**: net-snmp 5.2.1.2-4ubuntu2.

Orchestration Software Details

A bespoke software package to orchestrate the evaluation and gather the metric was created, and its details are as follows:

- Microsoft .NET Framework 1.1
- C# programming language for .NET

C.4 Sink Nodes

Most of the measurements require a node to be present on the other side of the DUT to operate properly (Figure 4.2, and Figure 4.1). Due to some limitations with the setups listed in the above, the following was equipment was employed instead:

- **Hardware**:
  - Processor: Intel Pentium III 700MHz.
  - RAM: 256MB.
  - Network Card: 3Com 10/100Mbps compatible Ethernet.
- **Operating System**: FreeBSD 5.6.
- **Available throughput measurement software**: Netperf version 4a.

Appendix D

Network Boundaries

D.1 Synopsis

Computer networks boundaries are physical or logical links where nodes controlled by a given organisation meet systems administered by other parties. The technologies that allow this separation include NAT, and PAT which are discussed in this section.

D.2 Scaling Addresses

The IP addressing system has several weaknesses. A major weakness is that there are only a predefined number of address which is not enough to cope with current demands. It also suffers from privacy issues, as the source and destination addresses appear in the data packets. Furthermore, organisations might possess a great number of nodes connected to its networks, and potentially exhaust their pool of addresses. Hence, NAT and its overloaded version PAT [37] have been introduced to isolate networks, and thus, overcome some of the problems of the IP addressing, and the Internet. In essence, using NAT allows organisation to register for only a few Internet IP addresses with relevant authorities, and still have connectivity to the Internet for all their network nodes. Figure D.1 illustrates that many organisations can have a great number of nodes connected to their internal networks, without hindering their capabilities in establishing a connection to the Internet.

However, NAT is a one-for-one translation method, thus the number of internal hosts that can communicate with remote nodes is limited to the size of the organisation's public address pool. When the number of nodes allowed to access Internet resources exceeds the pool's size, PAT can be implemented. PAT maps both the address, and TCP communication port, as illustrated in Figure D.2 and Table D.1.

The key advantage of NAT is that it facilitates organisation's growth. When an organisation chooses to remotely access resources via NAT, it can simply move from one addressing scheme to another, without changing its public address(es). NAT, however, does not offer audit facilities, thus does not permit trace-back when problems occur. In terms of access control, the challenge is to ensure that security requirements do not prevent legitimate entities to operate properly.

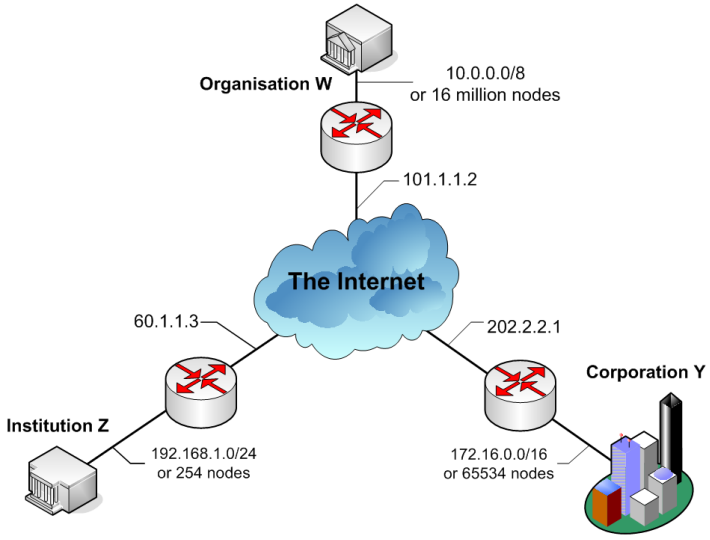

Figure D.1 – Illustration of network address translation usage

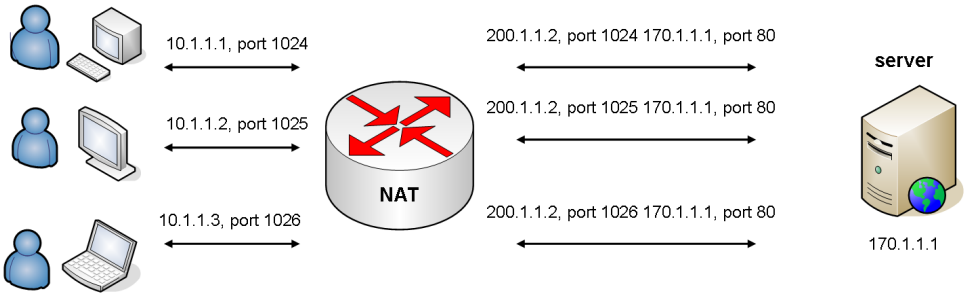

Figure D.2 – NAT overload using PAT by Odom [37]

Table D.1 – Dynamic NAT table with overloading by Odom [37]

Local Address	Public Address
10.1.1.1:1024	200.1.1.2:1024
10.1.1.2:1025	200.1.1.2:1025
10.1.1.3:1026	200.1.1.2:1026

D.3 Security Techniques

It is not possible to achieve adequate security standards in computer networks by solely using one methodology or device. Organisations need to employ an array of approaches to that end, thus this section presents the most utilised techniques and devices for this purpose, and it also highlights where these fit within the core security principles.

D.3.1 Network Design

Computer network systems have been progressively deployed within organisations. This deployment is often linked to an objective, or set of objectives, that these systems will support [13, 147]. Hence, computer network design or architecture defines the logical arrangements and connection of nodes, such as routers, servers, hosts and so on. For instance, a **W**ide **A**rea **N**etwork (WAN) design enables reliable communication between the computer systems housed at the main campus of an organisation, and systems at remote offices [13], however this results in limited bandwidth availability. On the other hand, a **L**ocal **A**rea **N**etwork (LAN) environment is better suited for applications with high bandwidth requirements, such as rapidly transferring large amount of data between any two closely-located computer systems. Hence, proper designs typically reflect a hierarchy that allows for performance, ease of management, and facilitates growth. It thus is possible to summarise that network design is often derived from the following parameters:

- Type of activity.
- Traffic volume.
- User's requirements.
- Administration, both in terms, of human resources, and budget overhead.
- Security.

Network Segmentation

The hierarchy of the organisation also translates into the design in particular in terms of segmentation. This relates to parts of the overall network, in other words subnetworks, that will house applications, services, and protocols, dedicated to a specific work unit, or department, such as IT support. Intermediate devices, such as Ethernet switches, routers, or network firewalls, typically create these segments either physically, or with the usage of vLAN technology that allows a single physical medium to be separated into isolated logical units. The benefit of strong segmentation includes the possibility of containing malware outbreaks as communications between any two segments can be filtered with network firewalls for instance [113, 131, 112, 71]. Overall, choices in terms of boundaries, number of redundant paths and logical arrangements influence the security of an organisation.

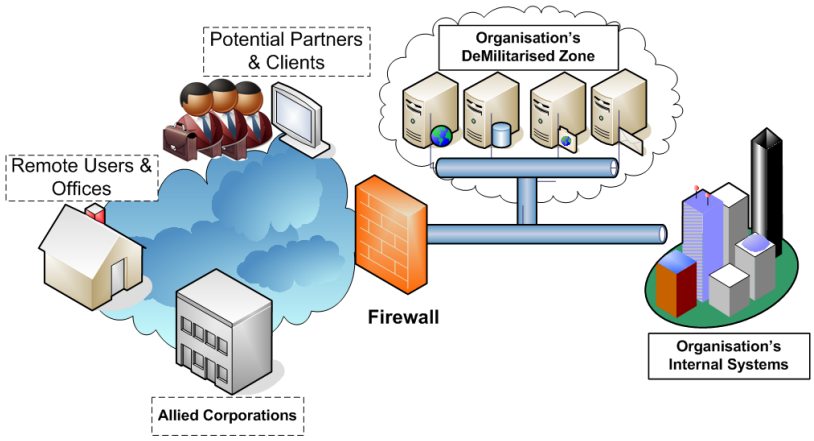

Figure D.3 – Typical position of the DMZ in modern networks

Demilitarised Zone

Arguably, one of the noted evolution in network segmentation is the increase utilisation of the **De**-**M**ilitarised **Z**one (DMZ). The DMZ represents an area of a corporate network where services to the public are offered, such as a corporate web site or file transfer. In other words, the DMZ offers services that represent the public face of the organisation [74], and Figure D.3 illustrates a DMZ location within the corporate network architecture. In order to fully benefit from the presence of a DMZ, it is important to tighten the flow of communication between the DMZ and internal networks [131]. DMZs are promoted in works such as of Hanson et al. [131], and Rosamond [125], however do not appear in earlier publications where themes of network segmentations and interaction are studied, such as in Al-Tawil and Al-Kaltham [74], Avolio [72], or Guttman [52], for instance. Arguably, these publications appeared at a time when only a few institutions were connected to the Internet, and Internet connectivity was not viewed as a fundamental part of an organisation's success. This demonstrates that there are dangers in offering services to outsiders from within the corporate network, especially in situations where compromises are performed using widely used protocols or services, such as the HTTP. Some advanced implementations use two distinct DMZs. This allows easier identification of compromises, faster recovery, and the possibility of alerting relevant personnel if necessary (Figure D.4).

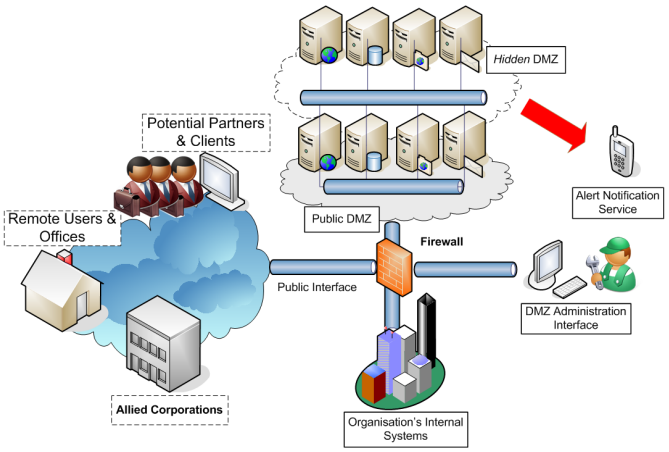

Figure D.4 – Implementation of two DMZs proposed by Rosamond [125]

Novel Framework for Automated Security Abstraction, Modelling, Implementation and Verification

Lionel Saliou, William J Buchanan, Jamie Graves and Jose Munoz
School of Computing, Napier University, Edinburgh, UK
l.saliou@napier.ac.uk

Abstract: This paper presents a novel framework for network security, and provides a complete solution to integrated security policies, which meets the objectives of an organisation, and also an automated verification process. The framework uses a security compiler, which converts high-level abstract definitions of the objectives of an organisation, and its security requirements. The output of this is then converted into an XML abstraction of security requirements, which can then be modelled, and converted into an implementable form, such as using firewall and IDS rules. Once it has been implemented, network agents are then used to generate and gather data allowing the security policy to be verified against the requirements.

The main areas of the framework are:

- Formal definition and abstraction. This involves the application of formal abstract security languages, such as an ontology mark-up language, and the novel implementation of integrated social rules, along with some form of definition of the aims and objectives of the organisation.
- Implementation. This involves converting the abstraction of the security policy into code and configurations, which can be implemented in the network devices, such as in the implementation of firewall and IDS rules, along with rules for data gathering agents. The paper shows practical implementations of these.
- Test and verification. This involves using data gathering and test generation agents to test and verify that the security system meets its initial objectives. This is obviously a key element in the system, as it provides automated feedback, and refinement.

The paper also provides novel results, which show how network agents can detect threats, and how the network can reconfigure itself, and limit its damage. It also shows typical delays for well-known worm threats and concludes with a novel method of detection and proposes methods on how the network could automate its configuration to overcome typical network threats, such as worms and viruses.

Keywords: Network agents, security abstraction, formal definition, reconfiguration, automated verification

1. Introduction

A key objective of enhancing computer network security is to promote it as an integrated process, rather than as an addition to the overall operation of the organisational network. It is thus necessary to create a framework that fully integrates the security requirements of an organisation, such as integrity, accessibility, and availability, in order to facilitate the creation of a consistent and configurable set of parameters for the assets. This will enable automatic deployment of a security policy that can be implemented on a live system, and evaluated in a real-time manner, to show the compliance of the system with the organisational objectives. This paper outlines a novel framework, which shows a logical flow of a security policy from its definition to its verification.

Organisations have a range of techniques available to keep their assets secured, such as smart cards, firewalls, intrusion detection systems (IDS), software patches, usernames, and passwords. These may be efficient from a technical point-of-view, but mostly rely on human intervention for configuration, programming, installation, and maintenance. They also often suffer from the fact that many security implementations do not actually reflect the operational and/or hierarchical structure of the organisation (Danchev 2003, p4). Thus security is often viewed as a limitation to functionality (Smetters, et al. 2002, p83), or an obstacle to usability (Viega, et al. 2004, p62), and it becomes challenging to make choices or implement changes. A critical factor in network security, though, is time (Zou, et al. 2003, p199) and the weakness of most of the security solutions is that they require substantial efforts in upkeep (Rosamond 2004, p25). Hence, computer networks do not adapt very well to new threats or new organisational requirements (Corbitt 2002, p21). Organisations should thus have policies (Rosamond 2004, Corbitt 2002, Timms, et al. 2004, Fraser 1997) that dictate how mechanisms and procedures must be coordinated. This paper highlights essential features of a framework and outlines a prototype of the system to prove the principle with automated rules generation, and a mitigation process to thwart a particular threat. The paper shows an example of a

Analysis of Firewall Performance Variation to Identify the Limits of Automated Network Reconfigurations

Lionel Saliou, William Buchanan, Jamie Graves and Jose Munoz
Centre of Mobile Computing and Security, Napier University, Edinburgh, UK
l.saliou@napier.ac.uk
w.buchanan@napier.ac.uk
j.graves@napier.ac.uk
j.munoz@napier.ac.uk

Abstract: Security in computer networks is typically passive, static, and reactive. This is typically due to most networking devices being rule-based, and when updates are necessary, they are normally done manually. Ultimately, the social and hierarchical structure of an organisation should be visible within the configuration of networks. Hence, it is desirable for a distributed system to be capable of reconfiguring itself in a timely-manner to reflect changes in policy, in practices, and in the social hierarchy, such as the promotion of a member of staff, or in the face of a security threat, such as in malware propagation.

This paper builds on the concept of an automated mitigation and reconfiguration system for networked devices, and evaluates key firewall system performance tests. These could be important in defining the criteria for the success of this type of security implementation. It thus defines a range of experiments, which evaluate firewall parameters, such as number of rules, and their position in relation to performance metrics, such as CPU utilisation, bandwidth consumption, and network latency. The paper also includes tests with up to 65,000 rules, and presents results on the positions of the rules, such as on the incoming and outgoing ports, and the effect of different network throughputs.

It concludes that networks can be made more resilient, under heavy network loads and large rule sets, if rule sets are applied on the outgoing ports. It also shows evidence that configuration interfaces are the performance bottleneck for multi-agent systems that may use these to reconfigure network equipments dynamically.

Keywords: firewall management, computer network defence, dynamic reconfiguration, mitigation, firewall performance metrics.

1. Introduction

This paper defines, and presents results for, a range of experiments designed to identify the limitations, in terms of performance, of network firewalls. It also critically evaluates the performance of software agents that could be interfaced with the equipment to address the lack of dynamic enforcement of such equipment, and their limited abilities to evolve. In conclusion, it advocates a scientific, and repeatable, approach to network security evaluation.

Most networked systems are rule-based, static, and, therefore, are often difficult to evolve. In addition, such systems are often vulnerable from internal intruders, or anyone capable of exploring the network infrastructure, because flaws, or lapses, remain unfixed for long periods. Some researchers, such as Glenn (2003, p12), argue that modification tasks which are typically required in these situations, are much more costly in terms of human resources, planning and so on, than the fresh deployment of new systems. Alternatively, flexible and dynamically configurable equipment, such as Active Network (AN), exists. However, they do not offer the same level of performance as traditional equipment, and hence are seldom deployed in corporate networks (Campbell *et al.* 1999, p7). These issues are particularly relevant when organisations use an integrated security framework that aims to ensure the implementation of security requirements, as well as enabling networked system with the ability to thwart threats in real-time, without hindering the organisation's objectives (Saliou *et al* 2005, p306).

2. Related research work

This paper defines that a hybrid architecture, based on multi-agents system (Santana Torrellas *et al.* 2003, p369), could be used to meet: security requirements; performance; and, in addition, provide dynamic reconfiguration capabilities. A critical factor in the success of such architecture is

Scenario Analysis using Out-of-Line Firewall Evaluation Framework

Lionel Saliou, William Buchanan, Jamie Graves, and Jose Munoz
Napier University, Edinburgh, UK
l.saliou@napier.ac.uk
w.buchanan@napier.ac.uk
j.graves@napier.ac.uk
j.munoz@napier.ac.uk

Abstract: Distributed Denial-of-Service (DDoS) attacks against corporate networks and assets are increasing, and their potential risk for future attacks is also a major concern. These attacks typically aim at disabling computer network infrastructure, and, since there is no one method to mitigate this type of threat, organisations must deploy adequate solutions, and assess the adequacy of their choices against their network requirements, through analysis, such as a simulation, or through network device modelling. A key factor is that DDoS is a dynamic type of attack, and thus device performance is a key parameter, especially for intermediate devices, such as network firewalls. Most of the modelling, though, for firewalls is focusing on static and logical performance attributes, such as whether traffic is denied or permitted. Thus existing models typically cannot deal with dynamic issues when related to intermediate devices. Simulation tools might be possible, but it is often difficult to cover a whole range of devices, thus this paper outlines a novel method of modelling the dynamic performance of network firewalls, and in measuring if they can cope with varying network loads.

Keywords: Dynamic evaluation, network firewall, analysis, security, out-of-line evaluation

1. Introduction

Increased reliance on networked systems poses several challenges to organisations, especially in terms of: protecting data from unauthorised access or corruption; guarantying accessibility; and; resilience of network services among others. Safeguarding assets, such as Web servers, customers' databases, and so on, is often achieved by means of security policies. However, this is a non-trivial task (Eloff *et al.* 2003), and these policies often do not reflect organisational aims, objectives, or work practices (Danchev 2003a). There could be also an issue with policy interpretation by technical staff (Saliou *et al.* 2005 2006) as they might not fully understand legal requirements (Barton *et al.* 2003). In addition, the discrepancy between the expression of policies and the actual deployment on live systems, often plays an important role in security breaches (Danchev 2003b, and Ioannidis *et al.* 2000). Furthermore, security solutions, such as network firewalls, are often deployed with little testing (Danchev 2003b), or succinct investigation, regarding their performance in a production environment. This is a serious issue, as, of all the security solutions that an organisation could use to defend its assets (Saliou *et al.* 2006) with, the network firewall is the element whose configuration is the most closely linked to the organisation's security policy.

Many threats exist in networked systems and most systems can cope with known threats, such as worms and viruses, as they have well-known traffic and activity signatures, and also have reliable mitigation procedures. Unfortunately, DDoS is one of the most difficult attacks to cope with, as it is almost impossible to differentiate between legitimate and non-legitimate traffic when they are requesting resources, such as from a Web server, or networked device. It is thus a growing threat, as Yegneswaran *et al.* 2003 highlight, and comes in many forms (Glenn 2003, and Paxson 2001). Along with this, servers can be hardened against DDoS, however there is an effect on intermediate devices, such as network firewalls, and is a major threat. Hence, a strong understanding of how intermediate device cope with traffic flows is just as important as how an end-device, such as a server, might cope with dynamic attacks.

One of the methods used to enhance the network infrastructure resilience is to deploy rules against a complete network domain, such as for an Internet Service Provider (ISP) domain, however this would also ban hosts within that domain which were non-malicious. Thus, an alternative is to apply firewall rules that deny known offending nodes access to the corporate network (Glenn 2003). This