Thesis for the degree of Doctor of Philosophy

Understanding Intrusion Detection Through Visualisation

STEFAN AXELSSON

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2005

Understanding Intrusion Detection Through Visualisation

STEFAN AXELSSON

ISBN 91-7291-557-9

Doktorsavhandlingar vid Chalmers Tekniska Högskola
Ny serie nr 2239
ISSN 0346-718X

Technical report 36D
ISSN 1651-4971
School of Computer Science and Engineering

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg, Sweden
Phone: +46 (0)31-772 1000

Contact information:
Stefan Axelsson
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 Göteborg, Sweden

Phone: +46 (0)31-772 5422
Email: 'sax@cs.chalmers.se'
URL: 'http://www.cs.chalmers.se/~sax'

Cover: From left to right clockwise: Screen dumps of *Bayesvis*, *3D access graph visualisation* and a *parallel coordinate plot*, all using malicious and benign web access requests as input data.

Printed in Sweden
Chalmers Reproservice
Göteborg, Sweden 2005

Understanding Intrusion Detection Through Visualisation

STEFAN AXELSSON
Department of Computer Science and Engineering,
Chalmers University of Technology

Thesis for the degree of Doctor of Philosophy

Abstract

With the ever increasing use of computers for critical systems, computer security, the protection of data and computer systems from intentional, malicious intervention, is attracting much attention. Among the methods for defence, *intrusion detection*, i.e. the application of a tool to help the operator identify ongoing or already perpetrated attacks has been the subject of considerable research in the past ten years. A key problem with current intrusion detection systems is the high number of false alarms they produce. This thesis presents research into why false alarms are and will remain a problem and proposes to apply results from the field of information visualisation to the problem of intrusion detection. This was thought to enable the operator to correctly identify false (and true) alarms, and also aid the operator in identifying other operational characteristics of intrusion detection systems. Four different visualisation approaches were tried, mainly on data from web server access logs. Two direct approaches were tried; where the system puts the onus of identifying the malicious access requests on the operator by way of the visualisation. Two indirect approaches were also tried; where the state of two self learning automated intrusion detection systems were visualised to enable the operator to examine their inner workings. This with the hope that in doing so, the operator would gain an understanding of how the intrusion detections systems operated and whether that level of operation, and the quality of the output, was satisfactory. Several experiments were performed and many different attacks in web access data from publicly available web servers were found. The visualisation helped the operator either detect the attacks herself and more importantly the false alarms. It also helped her determine whether other aspects of the operation of the self learning intrusion detection systems were satisfactory.

Keywords: Computer security, intrusion detection, visualisation, usability, human computer interaction.

This work was partially financed by SSF (Swedish foundation for Strategic Research) and Vinnova.

List of Appended Papers

This thesis is based on the work contained in the following papers:

[Axe00a] Stefan Axelsson. The Base-Rate Fallacy and the Difficulty of Intrusion Detection. In *ACM Transactions on Information and System Security (TISSEC), 3(3)*, pp. 186–205, ACM Press, ISSN: 1094–9224, 2000.

[Axe04b] Stefan Axelsson. Visualising Intrusions: Watching the Webserver. In *Proceedings of the 19th IFIP International Information Security Conference (SEC2004)*, Tolouse France, 22–27 Aug, 2004.

[Axe04a] Stefan Axelsson. Combining A Bayesian Classifier with Visualisation: Understanding the IDS. In *Proceedings of the ACM CCS Workshop on Visualization and Data Mining for Computer Security*, (Held in conjunction with the Eleventh ACM Conference on Computer and Communications Security), Oct 29, 2004.

[Axe04c] Stefan Axelsson. Visualising the Inner Workings of a Self Learning Classifier: Improving the Usability of Intrusion Detection Systems. *Technical report 2004:12*, Department of Computing Science, Chalmers University of Technology, Göteborg, Sweden, 2004. Submitted for publication.

[Axe03] Stefan Axelsson. Visualization for Intrusion Detection: Hooking the worm. In *Proceedings of the 8th European Symposium on Research in Computer Security (ESORICS 2003)*, Springer Verlag: LNCS 2808, 13–15 Oct, Gjøvik Norway, 2003.

Acknowledgements

Even though writing a thesis is at times a lonely task, the work on which it was based was not done in isolation. Far from it; I owe many more people my thanks than I can mention here. That said I would still like to take the opportunity to mention a few people who have been instrumental in my finishing my PhD.

I would like to start by relating the following anecdote: Quite a few years ago I met a theologian in Cambridge (who shall remain nameless) who was of the opinion that supervisors could be divided into two groups. The first were those that while nice enough people, really did not help your thesis work along, and the second were those that while they really did help your thesis work along, were less than pleasant to be around (I found the *actual* words used to describe the latter group surprisingly strong coming from a theologian). After having worked with my supervisor Professor David Sands for two years, I would dispute the above dichotomy. There are indeed supervisors of a third kind, who are both helpful when it comes to getting ones research moving in the right direction, while still being friendly and humorous. You are everything one could wish for in a supervisor Dave.

That is not to say that others at the department have not also lent support and brightened my day. While all my colleagues are too numerous to mention, I would especially like to thank (in no particular order) Daniel Hedin, Ulf Norell, Nils Anders Danielsson, Tobias Gedell, Claes Nyberg and Thorbjörn Axelsson. I would be less knowledgeable without having worked with you and I would certainly have had a much drearier time doing it. My erstwhile climbing partner, now turned colleague Dr. Rogardt Heldal deserves special mention, as he is put up with my comings and goings and still managed to provide valuable insights over the past few years. I would also like to thank previous and present colleagues at the department of Computer Engineering here at Chalmers and at Ericsson where I have been employed for the past few years.

Last but not least are the two people without whose support this work would not have got far. I am talking of course of my wife Hanna Tornevall who has had to bear the brunt of the work keeping the family going this autumn, and our son Oskar. In fact, Oskar's first proper two syllable word was "dat-oo" (clearly legible Swedish for *dator*, i.e. *computer*), as in: "Oskar, where's daddy?", "Dat-oo!" I know I have been an absent father at times when preparing this thesis, even when present in the flesh. Thank you Oskar for not holding that against me.

Contents

1 Introduction **1**

2 Computer Security **1**

3 Anti-Intrusion Techniques **2**

4 Intrusion Detection **4**

4.1 An Architectural Model of Intrusion Detection Systems 7

4.2 Explaining Intrusion Detection From the Perspective of Detection and Estimation Theory 9

5 Rationale and Problem Statement **15**

6 Introduction to Visualisation **15**

7 Overview of Appended Papers **17**

7.1 Paper A: The Base-Rate Fallacy and the Difficulty of Intrusion Detection 17

7.2 Paper B: Visualising Intrusions: Watching the Webserver 18

7.3 Paper C: Combining a Bayesian Classifier with Visualisation: Understanding the IDS 20

7.4 Paper D: Visualising the Inner Workings of a Self Learning Classifier: Improving the Usability of Intrusion Detection Systems 22

7.5 Paper E: Visualization for Intrusion Detection: Hooking the worm. . 24

8 Results in Perspective **25**

9 Related Work **26**

10 Conclusions and Future Work **28**

References **28**

Paper A **37**

Paper B **59**

Paper C **81**

Paper D **107**

Paper E **137**

Colour Plates **159**

All science is either physics or stamp collecting.
– Ernest Rutherford (1871–1937)

1 Introduction

Computer Security has been of interest since the beginning of electronic computing. This is perhaps not surprising considering the close ties the field has had with the military. However, with the emergence of the Internet as a household concept, the past ten years have seen a marked rise in the interest in computer security issues. Over the past decade the public has grown accustomed to hearing about the exploits of hackers, credit card fraudsters and the like on the evening news and today they have even been made targets themselves by *spammers*, *phishers*, *worms* etc. This interest will surely only continue to increase in the years to come, when (inter)networked computer systems will be relied on to handle increasing numbers of critical transactions. The computer crimes of yesterday, most of which were little more than pranks, have come of age with the realisation that there are huge sums up for grabs for the enterprising criminal with a technological knack.

This thesis presents research into one principle of protecting valuable computer resources: *Surveillance*, using information visualisation to aid the operator in understanding either the security state of the monitored system directly or indirectly by providing her insight into the operation of some intrusion detection system.

This thesis begins with a short introduction to computer security, and an introduction to intrusion detection to set the scene. A rationale for applying the principle of information visualisation to intrusion detection follows together with a short introduction to visualisation. Then comes the contributions in context, conclusions and the actual papers. A chapter of selected colour plates from the papers ends the thesis.

2 Computer Security

The computer security field is primarily concerned with protecting one particular resource: data.

The value of data can be compromised in three ways, commonly referred to as the *CIA* of computer security [CEC91]:

1. *Confidentiality* Prevention of the unauthorised disclosure of information.

   The value of much data relies on it being kept secret from prying eyes. Violating this secrecy thus entails a breach of confidentiality.

2. *Integrity* Prevention of the unauthorised modification of information.

   In some circumstances we may not be particular about the secrecy of our data, but it remains absolutely crucial that the data not be tampered with. We require a high level of trust in the accuracy of our data, i.e. for its integrity to remain unquestioned.

3. *Availability* Prevention of the unauthorised withholding of information or resources.

   Our data should be available to us when, where and in the form we need it. Data that is confidential and has the highest integrity will be of no use to us if we cannot process it when the need arises. Thus it is imperative that our data remains available to us at our convenience.

A fourth factor is sometimes added [Mea93, Jon98]: *No unauthorised use*, viz. that no unauthorised person should be allowed to use the computing resource, even though that in itself would not violate any of the *CIA* requirements. From a risk management perspective, it is easy see that such a person would probably end up in a position from which further violations were possible, and it is therefore appropriate to act to address that scenario.

Different owners of data make different decisions about the relative importance of these factors. Three hypothetical scenarios will suffice as examples. The first is that of a military entity, paranoid about confidentiality to the point that it would rather blow up its own computer installations then let them fall intact into the hands of the enemy. Integrity and availability play less of a role in such a decision. The second is that of a bank. Although it is anxious that information might leak into the wrong hands, it is more concerned with integrity. That someone can learn the balance of an account is less of a concern than the risk that someone could alter it, perhaps by adding a zero to the end. The third is a relatively new and is that of an internet merchant who is mostly concerned with the continued availability of her website as while she can tolerate the odd leaked credit card number of one of her customers, she cannot tolerate having her business shut down for any appreciable amount of time. The latter scenario has become increasingly important in the last few years.

Many security measures can be employed to defend against computer intrusions and other unauthorised tampering with protected resources, the establishment of a strong perimeter defence being only one possible measure. Another method well established in the traditional security field is that of an intrusion alarm coupled with a security response. A great deal of research has recently gone into the idea of automated intrusion alarm for computer systems, a so-called *intrusion detection system*, or *IDS* for short.

3 Anti-Intrusion Techniques

Several methods are available to protect a computer system or network from attack. A good introduction to such methods is [HB95], from which this section borrows heavily. The paper lists six general, non-exclusive approaches to anti-intrusion techniques: pre-emption, prevention, deterrence, detection deflection, and countermeasures (see Figure 1):

1. *Pre-emption* To strike against the threat before it has had a chance to mount its attack, in the spirit of: 'Do unto others, before they do unto you.' In a civilian setting, this is a dangerous and possibly unlawful approach, where innocent—and indeed not so innocent—bystanders may be harmed.

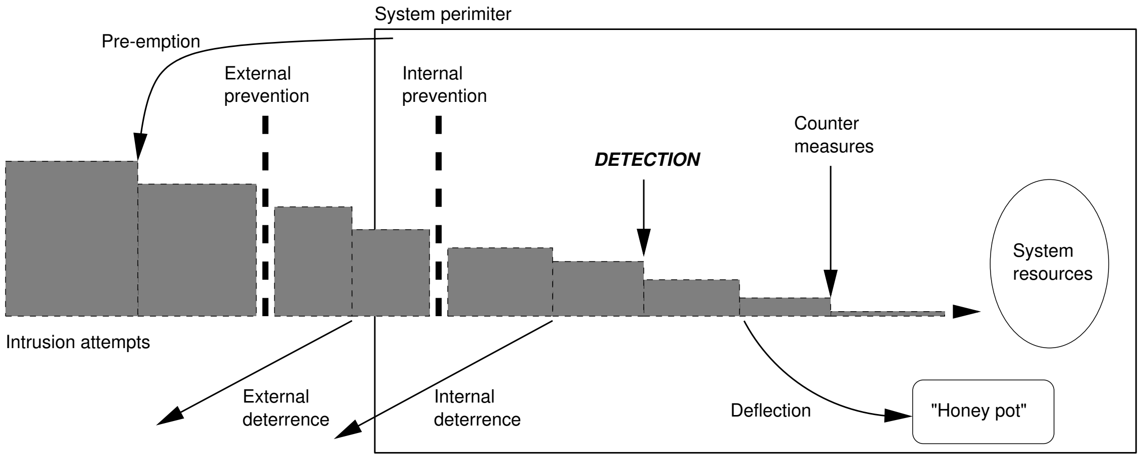

Figure 1: Anti-intrusion techniques (from [HB95])

2. *Prevention* To preclude or severely limit the likelihood of a particular intrusion succeeding. One can, for example, elect to not be connected to the Internet if one is afraid of being attacked by that route, or choose to be connected via some restriction mechanism such as a firewall. Proving your software free of security defects also falls under this heading. Unfortunately, this can be an expensive and awkward approach, since it is easy to throw the baby out with the bath water in the attempt to prevent attacks. Internal prevention comes under the control of the system owner, while external prevention takes place in the environment surrounding the system, such as a larger organisation, or society as a whole.

3. *Deterrence* To persuade an attacker to hold off his attack, or to break off an ongoing attack. Typically this is accomplished by increasing the perceived risk of negative consequences for the attacker. Of course, if the value of the protected resource is great, the determined attacker may not be scared off so easily. Internal deterrence can take the form of login banners warning potential internal and external attackers of dire consequences should they proceed. External deterrence could be effected by the legal system, with laws against computer crime and the strict enforcement of the same.

4. *Detection* To identify intrusion attempts, so that the proper response can be evoked. This most often takes the form of notifying the proper authority. The problems are obvious: the difficulty of defending against a hit-and-run attack, and the problem of false alarms, or failing to sound the alarm when someone surreptitiously gains, or attempts to gain, access.

5. *Deflection* To lure an intruder into thinking that he has succeeded when in fact he has been herded away from areas where he could do real damage. The main problem is that of managing to fool an experienced attacker, at least for a sufficient period of time.

6. *Countermeasures* To counter actively and autonomously an intrusion while it is in progress. This can be done without the need for detection, since the

countermeasure does not have to discriminate—although it is preferable if it can—between a legitimate user who makes a mistake and an intruder who sets off a predetermined response, or 'booby trap'.

The reasons for our desire to employ the principle of surveillance are much the same as in the physical security arena: We wish to deploy a defence in depth; we do not believe in the infallibility of the perimeter defence; when someone manages to slip through or even attempts to attack we do not want them to have undetected free reign of the system; for technical reasons we perhaps cannot strengthen our perimeter defences (lack of source code etc.); we wish to defend not only against outsiders, but also against insiders, those that already operate within the perimeter, etc.

4 Intrusion Detection

As the principle of surveillance stems from the application of intrusion detection systems to computer security it is fitting to start with a few definitions and introduction to that area of study. Research in intrusion detection is the study of systems that automatically detect intrusions into computer systems. They are designed to detect computer security violations made by the following important types of attackers:

- Attackers using prepacked 'exploit scripts.' Primarily outsiders.
- Automated attacks originating from other computer systems, so called *worms*.
- Attackers operating under the identity of a legitimate user, for example by having stolen that user's authentication information (password). Outsiders and insiders.
- Insiders abusing legitimate privileges, etc.

Defining these terms to our satisfaction turns out to be problematic. Although most computer users could easily describe what they do not want to happen with their computers, finding strict definitions of these actions is often surprisingly difficult. Furthermore, many security problems arise between the ordinary every day definitions that we use to communicate security, and the strict definitions that are necessary to research. For example the simple phrase 'Alice speaks to Bob on the freshly authenticated channel,' is very difficult to interpret in a packet-sending context, and indeed severe security problems have arisen from confusion arising from the application of such simple models such as 'speaking' in a computer communications context [Gol00]. That numerous, spectacular mistakes have been made by computer security researchers and professionals only serves to demonstrate the difficulty of the subject.

Definitions

That said, a definition of what we mean by *intrusion* and other related terms remains essential, at least in the context of *intrusion detection*:

Intrusion The *malicious* violation of a *security policy* (implied or otherwise) by an *unauthorised agent*.

Intrusion detection The automated detection and alarm of any situation where an intrusion has taken, or is about to take place. (The detection must be complemented with an alert to the proper authority if it is to act as a useful security measure.)

We will consider these definitions in greater detail in the following paragraphs:

Malicious The person who breaks into or otherwise unduly influences our computer system is deemed not have our best interests at heart. This is an interesting point, for in general it is impossible for the intrusion detection system to decide whether the agent of the security violation has malicious intent or not, even after the fact. Thus we may expect the intrusion detection system to raise the alarm whenever there is sufficient evidence of an activity that *could* be motivated by malice. By this definition this will result in a false alarm, but in most cases a benign one, since most people do not mind the alarm being raised about a potentially dangerous situation that has arisen from human error rather than malicious activity.

Security Policy This stresses that the violations we wish to protect against are to a large extent up to the owner of the resource being protected (in western law at least). Other legitimate demands on security may in future be made by the state legislature. Some branches of the armed services are already under such obligations, but in the civilian sector few (if any) such demands are currently made. In practice security policies are often weak, however, and in a civilian setting we often do not know what to classify as a violation until after the fact. Thus it is beneficial if our intrusion detection system can operate in circumstances where the security policy is weakly defined, or even non-existent. One way of circumventing this inherent problem is for the supplier of the intrusion detection system to define a *de facto* security policy that contains elements with which she hopes all users of her system will agree. This situation may be compared with the law of the land, only a true subset of which is agreed by most citizens to define *real* criminal acts. It goes without saying that a proper security policy is preferable. This ought to be defined as the set of actions (or rather principles) of operation that are allowed, instead of in the negative for best security.

Unauthorised Agent The definition is framed to address the threat that comes from an *unauthorised agent*, and should not be interpreted too narrowly. The term singles out all those who are not legitimate owners of the system, i.e. that are not allowed to make decisions that affect the security policy. This does not specifically exclude *insiders* i.e. people who are authorised to use the system to a greater or lesser extent, but not authorised to perform all possible actions. The point of this distinction is that we do not attempt to encompass those violations that would amount to protecting the owner from himself. To accomplish this is, of course, both simple and impossible: Simple in the sense that if the owner makes a simple legitimate mistake, a timely warning may make him see his error and take corrective action; impossible, in that if the person who legally commands the system wishes to destroy or otherwise influence the system, there is no way to prevent him, short of taking control of the system away from him, in which case he no longer 'legally commands the system.' When all is said and done, trust has to be placed in an entity, and our only defence against this trust being abused is to use risk management activities external to the intrusion detection system. Whether non-human entities such as other computers that are attacking us should be considered agents in themselves or merely tools acting on the behalf of some other agent is a difficult question that we will not delve more deeply into here.

Automated Detection and Alarm The research into intrusion detection has almost exclusively considered systems that operate largely without human supervision. An interesting class of systems that has not been studied to any significant degree before the advent of this thesis are those that operate with a larger degree of human supervision, placing so much responsibility on the human operator that *she* can be thought of as the detection element proper (or at least part of it). Such systems would support the human in observing and making decisions about the security state of the supervised system; a 'security camera' for computer systems. Continued reliance solely on fully automated systems may turn out to be less than optimal. More will be said about this in sections 5 and 8.

Delivered to the Proper Authority It cannot be overemphasised that the alarm must be *delivered* to the *proper authority*—henceforth referred to as the Site Security Officer or SSO—in such a manner that the SSO can take action. The ubiquitous car alarm today arouses little, if any, response from the public, and hence does not act as an effective deterrent to would-be car thieves. Thus the SSO's response, which may or may not be aided by automatic systems within the intrusion detection system itself, is a crucial component in the fielding of intrusion detection systems. There has been little research, even in the simpler field of automated alarms, into how to present information to the SSO so that she can make the correct decision and take the correct action. It is important that the authority that is expected to take corrective action in the face of computer security violations—keeping in mind that such violations often originate 'in house'—really *has* the authority to take the appropriate action. This is not always the case in a civilian setting.

Intrusion has Taken Place The phrase 'any situation where an *intrusion has taken place*' may seem self-evident. However, there are important questions over the exact moment when the intrusion detection system *can* detect the intrusion. It is clearly impossible in the general case to sound the alarm when mere intent is present. There is a better chance of raising the alarm when preparatory action is taking place, while the best chance comes when a bona fide violation has taken place, or is ongoing. The case of *is about to take place* is interesting enough to warrant special treatment. In military circles this falls under the heading of *indication and warning*; there are sufficient signs that something is imminent to ensure that our level of readiness is affected. In a computer security context, the study of such clues, many of which are of course not 'technological' in nature, is not far advanced. It is an important subject, however, since it actually gives us the opportunity to ward off or otherwise hinder an attack. Without such possibilities, an alarm can only help to reduce the damage after the fact, or can only function as a deterrent.

Intrusion Detection Systems

The study of intrusion detection is today some twentyfive years old. The possibility of automatic intrusion detection was first put forward in James Anderson's classic paper [And80], in which he states that a certain class of intruders—the so-called *masqueraders*, or intruders who operate with stolen identities—could probably be detected by their departures from the set norm for the original user. Later the idea of checking all activities against a set security policy was introduced.

We can group intrusion detection systems into two overall classes: those that detect anomalies, hereafter termed *anomaly detection systems*, and those that detect the signatures of known attacks, hereafter termed *signature based systems*. Often the former automatically forms an opinion on what is 'normal' for the system, for example by constructing a profile of the commands issued by each user and then sounding the alarm when the subject deviates sufficiently from the norm. Signature systems, on the other hand, are most often programmed beforehand to detect the signatures of intrusions known of in advance.

These two techniques are still with us today, and (ignoring hybrid approaches) nothing essentially new has been put forward in this area. Section 4.2 will explain these two approaches in terms of detection and estimation theory.

4.1 An Architectural Model of Intrusion Detection Systems

Since the publication of Anderson's seminal paper [And80], several intrusion detection systems have been invented. Today there exists a sufficient number of systems in the field for one to be able to form some sort of notion of a 'typical' intrusion detection system, and its constituent parts. Figure 2 depicts such a system. Please note that not all possible data/control flows have been included in the figure, but only the most important ones.

Any generalised architectural model of an intrusion detection system would contain at least the following elements:

Audit collection Audit data must be collected on which to base intrusion detection decisions. Many different parts of the monitored system can be used as sources of data: keyboard input, command based logs, application based logs, etc. In most cases network activity or host-based security logs, or both, are used.

Audit storage Typically, the audit data is stored somewhere, either indefinitely[1] for later reference, or temporarily awaiting processing. The volume of data

[1] Or at least for a long time—perhaps several months or years—compared to the processing turn around time.

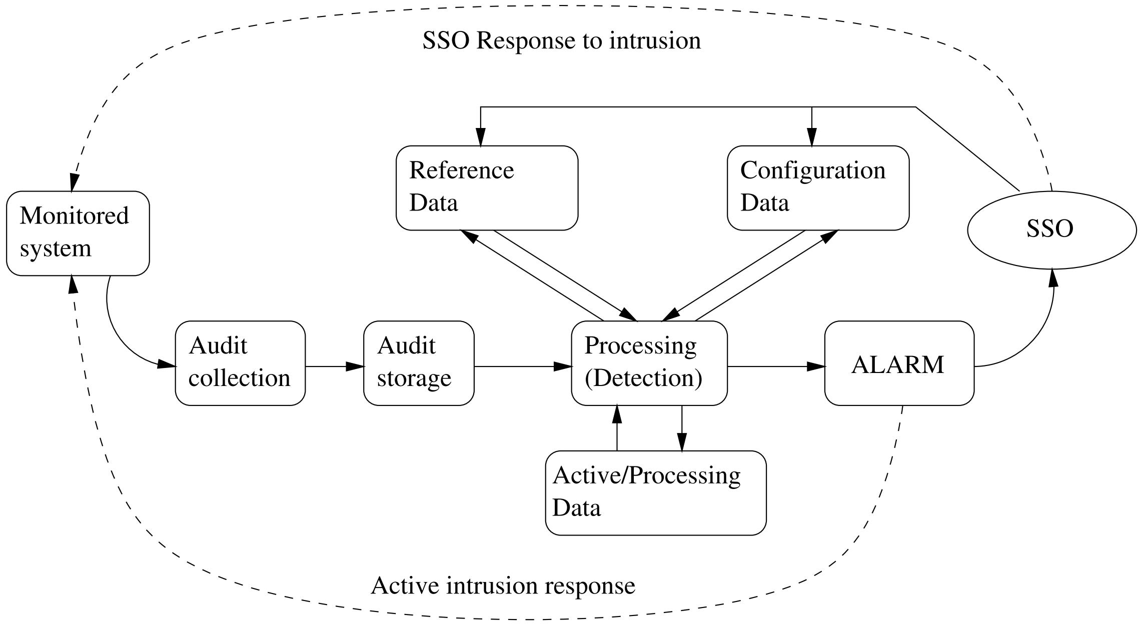

Figure 2: Organisation of a generalised intrusion detection system

is often exceedingly large[2], making this is a crucial element in any intrusion detection system, and leading some researchers to view intrusion detection as a problem in audit data reduction [Fra94, ALGJ98]

Processing The processing block is the heart of the intrusion detection system. It is here that one or many algorithms are executed to find evidence (with some degree of certainty) in the audit trail of suspicious behaviour. More will be said about the detector proper in section 4.2.

Configuration data This is the state that affects the operation of the intrusion detection system as such; how and where to collect audit data, how to respond to intrusions, etc. This is thus the SSO's main means of controlling the intrusion detection system. This data can grow surprisingly large and complex in a real world intrusion detection installation. Furthermore, it is relatively sensitive, since access to this data would give the competent intruder information on which avenues of attack are likely to go undetected.

Reference data The reference data storage stores information about known intrusion signatures—for misuse systems—or profiles of normal behaviour—for anomaly systems. In the latter case the processing element updates the profiles as new knowledge about the observed behaviour becomes available. This update is often performed at regular intervals in batches. Stored intrusion signatures are most often updated by the SSO, as and when new intrusion signatures become known. The analysis of novel intrusions is a highly skilled task. More often than not, the only realistic mode for operating the intrusion detection system is one where the SSO subscribes to some outside source of

[2]The problem of collecting sufficient but not excessive amounts of audit data has been described as "You either die of thirst, or you are allowed a drink from a fire hose."

intrusion signatures. At present these are proprietary. It nis difficult, if not impossible, to make intrusion detection systems operate with signatures from an alternate source, even though it is technically possible [LMPT98].

Active/processing data The processing element must frequently store intermediate results, for example information about partially fulfilled intrusion signatures. The space needed to store this active data can grow quite large.

Alarm This part of the system handles all output from the system, whether it be an automated response to suspicious activity, or more commonly the notification of a SSO.

4.2 Explaining Intrusion Detection From the Perspective of Detection and Estimation Theory[3]

Research into the automated detection of computer security violations is hardly in its infancy, yet little comparison has been made with the established field of detection and estimation theory (one exception being [LMS00]) the results of which have been found applicable to a wide range of problems in other disciplines. In order to explain the two major approaches behind intrusion detection principles we will attempt such a comparison, studying the problem of intrusion detection by the use of the introductory models of detection and estimation theory.

Classical Detection Theory

The problem of detecting a signal transmitted over a noisy channel is one of great technical importance, and has consequently been studied thoroughly for some time now. An introduction to detection and estimation theory is given in [Tre68], from which this section borrows heavily.

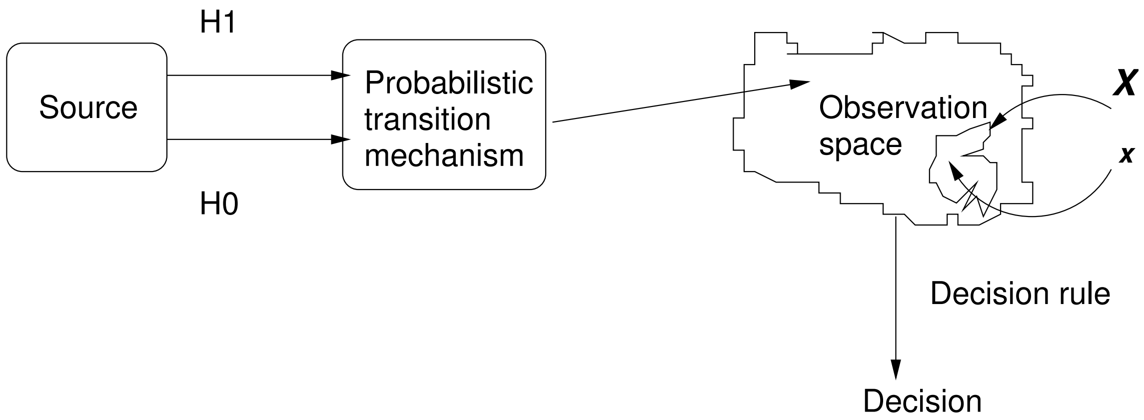

Figure 3: Classical detection theory model

In classical binary detection theory (see figure 3) we should envisage a system that consists of a source from which originates one of two signals, *H0* or *H1*. This signal is transmitted via some channel that invariably adds noise and distorts the signal

[3]This section is based on [Axe00b].

according to a probabilistic transition mechanism. The output—what we receive—can be described as a point in a finite (multidimensional) observation space, for example x in figure 3. Since this is a problem that has been studied by statisticians for some time, we have termed it the *classical detection model.* Based on an observation of the output of the source as transmitted through the probabilistic transition mechanism, we arrive at a decision. Our decision is based on a decision rule; for example: 'Is or is not x in X,' where X is the region in the observation space that defines the set of observations that we believe to be indicative of *H0* (or *H1*) (see figure 3). We then make a decision as to whether the source sent *H0* or *H1* based on the outcome of the comparison of x and X.

Note that the source and signal model *H0* and *H1* could represent any of a number of interesting problems, and not only the case of transmitting a one or a zero. For example, *H1* could represent the presence of a disease (and conversely *H0* its absence), and the observation space could be any number of measurable physiological parameters such as blood count. The decision would then be one of 'sick' or 'healthy.' In our case it would be natural to assign the symbol *H1* to some form of intrusive activity, and *H0* to its absence.

The problem is then one of deciding the nature of the probabilistic transition mechanism. We must choose what data should be part of our observation space, and on this basis derive a decision rule that maximises the detection rate and minimises the false alarm rate, or settle for some desirable combination of the two.

When deciding on the decision rule the *Bayes criterion* is a useful measurement of success [Tre68, pp. 24]. In order to conduct a Bayes test, we must first know the a priori probabilities of the source output (see [Axe05a] for further discussion). Let us call these P_0 and P_1 for the probability of the source sending a zero or a one respectively. Second, we assign a cost to each of the four possible courses of action. These costs are named C_{00}, C_{10}, C_{11}, and C_{01}, where the first subscript indicates the output from our decision rule—what we though had been sent—and the second what was actually sent. Each decision or experiment then incurs a cost, in as much as we can assign a cost or value to the different outcomes. For example, in the intrusion detection context, the detection of a particular intrusion could potentially save us an amount that can be deduced from the potential cost of the losses if the intrusion had gone undetected. We aim to design our decision rule so that the *average* cost will be minimised. The expected value—R for *risk*—of the cost is then [Tre68, p. 9]:

$$\begin{aligned} R = {} & C_{00}P_0P(\text{say } H0|H0 \text{ is true}) \\ & +C_{10}P_0P(\text{say } H1|H0 \text{ is true}) \\ & +C_{11}P_1P(\text{say } H1|H1 \text{ is true}) \\ & +C_{01}P_1P(\text{say } H0|H1 \text{ is true}) \end{aligned} \tag{1}$$

It is natural to assume that $C_{10} > C_{00}$ and $C_{01} > C_{11}$, in other words the cost associated with an incorrect decision or misjudgement is higher than that of a correct decision. Given knowledge of the a priori possibilities and a choice of C parameter values, we can then construct a Bayes optimal detector.

Though figure 3 may lead one to believe that this is a multidimensional problem, it can be shown [Tre68, p. 29] that a *sufficient statistic* can always be found whereby a coordinate transform from our original problem results in a new point that has the property that only one of its coordinates contains all the information necessary for making the detection decision. Figure 4 depicts such a case, where the only important parameter of the original multidimensional problem is named L.

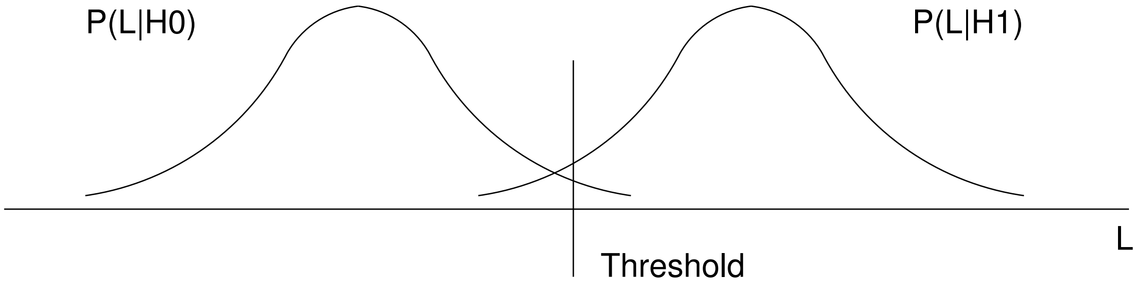

Figure 4: One dimensional detection model

It can furthermore be shown that the two main approaches to maximising the desirable properties of the detection—the Bayes or Neyman-Pearson criteria—amount to the same thing; the detector finds a likelihood ratio (which will be a function only of the sufficient statistic above) and then compares this ratio with a pre-set threshold. By varying the threshold in figure 4, it can be seen that the detection ratio (where we correctly say *H1*) and the false alarm rate (where we incorrectly say *H1*) will vary in a predictable manner. Hence, if we have complete knowledge of the probability densities of *H0* and *H1* we can construct an optimal detector, or at least calculate the properties of such a detector. We will later apply this theory to explain anomaly and signature detection.

Application to the Intrusion Detection Problem

This section is a discussion of the way in which the intrusion detection problem may be explained in light of the classical model described above.

Source Starting with the *source*, ours is different from that of the ordinary radio transmitter because it is human in origin. Our source is a human computer user who issues commands to the computer system using any of a number of input devices. In the vast majority of cases, the user is benevolent and non-malicious, and he is engaged solely in non-intrusive activity. The user sends only *H0*, that is, non-intrusive activity. Even when the user is malicious, his activity will still mostly consist of benevolent activity. Some of his activity will however be malicious, that is, he will send *H1*. Note that *malicious* has to be interpreted liberally, and can arise from a number of different types of activities such as those described by the taxonomies in for example [LBMC94, LJ97]. Thus, for example, the use of a pre-packed exploit script is one such source of intrusive activity. A *masquerading*[4] intruder can be another source of intrusive activity. In this case the activity that he initiates differs from the activity that the proper user would have originated.

[4] A *masquerader* is an intruder that operates under false identity. The term was first used by Anderson in [And80].

It should be noted that we have only treated the binary case here, differentiating between ‘normal’ behaviour and one type of intrusion. In reality there are many different types of intrusions, and different detectors are needed to detect them. Thus the problem is really a multi-valued problem, that is, in an operational context we must differentiate between *H0* and *H1, H2, H3*. . . ,where *H1–Hn* are different types of intrusions. To be able to discriminate between these different types of intrusions, some statistical difference between a parameter in the *H0* and *H1* situation must be observable. This is simple, almost trivial, in some cases, but difficult in others where the observed behaviour is similar to benevolent behaviour. Knowledge, even if incomplete, of the statistical properties of the ‘signals’ that are sent is crucial to make the correct detection decision.

It should be noted that earlier classifications of computer security violations that exist [LBMC94, NP89, LJ97] are not directed at intrusion detection, and on closer study appear to be formulated on too high a level of representation to be directly applicable to the problem in hand. There are now a handful of studies that links the classification of different computer security violations to the problem of detection, in this case the problem of what traces are necessary to detect intrusions after the fact [ALGJ98, Bar04a, KMT04, Max03].

Probabilistic Transition Mechanism In order to detect intrusive behaviour we have first to observe it. In a computer system context it is rare to have the luxury of observing user behaviour directly, looking over the user’s shoulder while he provides a running commentary on what he is doing and intends to do. Instead we have to observe the user by some other means, often by some sort of security logging mechanism, although it is also possible by observing the network traffic emanating from the user. Other more direct means have also been proposed, such as monitoring the user’s keystrokes.

In the usual application of detection theory, the probabilistic transition mechanism, or ’channel’, often adds noise of varying magnitude to the signal. This noise can be modelled and incorporated into the overall model of the transmission system. The same applies to the intrusion detection case, although our ‘noise’ is of a different nature and does not in general arise from nature, as described by physics. In our case we observe the subject by some (imperfect) means where several sources of noise can be identified. One such source is where other users’ behaviour is mixed with that of the user under study, and it is difficult to identify the signal we are interested in.

If, for example, our user proves to be malicious, and sends TCP-syn packets from a PC connected to a network of PCs to a target host, intended to execute a SYN-flooding denial-of-service attack on that host. Since the source host is on a network of PCs—the operating systems of which are known to suffer from flaws that make them prone to sending packet storms that look like SYN-flooding attacks to the uninitiated[5]—it may be difficult to detect the malicious user. This is because he operates from under the cover of the noise added by the poorly implemented TCP/IP stacks of the computers on the same source network as he is. It can thus

[5]Or at least *were* prone to ten years ago.

be much more difficult to build a model of our 'channel' than when the noise arises as a result of a purely physical process.

Observation Space Given that the action has taken place, and that it has been 'transmitted' through the logging system/channel, we can make observations. The set of possible observations, given a particular source and channel model, makes up our *observation space*. As said earlier, some results suggest that we can always make some sort of coordinate transformation that transforms all available information into one coordinate in the observation space. Thus in every detection situation we need to find this transformation.

In most cases the computer security audit data we are presented with will be discrete in nature, not continuous. This is different from the common case in detection theory where the signals are most often continuous in nature. In our case a record from a host-based security log will contain information such as commands or system calls that were executed, who initiated them, any arguments such as files read, written to, or executed, what permissions were utilised to execute the operation, and whether it succeeded or not. In the case of network data we will typically not have such high quality information since the data may not contain all security relevant information; for example, we will not know exactly how the attacked system will respond to the data that it is sent, or whether the requested operation succeeded or not [PN98]. The question of what data to log in order to detect intrusions of varying kinds is still open. We also know little of the way different intrusions manifest themselves when logged by different means.

Once again the literature is hardly extensive, although for example [ALGJ98, HL93, LB98] and more recently [Bar04b] have addressed the issues presented in this section, albeit from different angles.

Decision Rule Having made the coordinate transformation in the previous step we then need to decide on a threshold to distinguish between *H0* and *H1*.

Thus our hope when we apply anomaly detection is that all that is not normal behaviour for the source in question—that cannot be construed as *H0*—is some sort of intrusive behaviour. The question is thus to what degree abnormal equates to intrusive. This is perhaps most likely in the case of a *masquerader* who one may presume is not trained to emulate the user whose identity he has assumed. There are some studies that suggest that different users indeed display sufficiently different behaviour for them to be told apart [LB98].

Existing Approaches to Intrusion Detection

For a complete survey of existing approaches to intrusion detection see [BAJ03]. Here we will only outline the two major methods of intrusion detection: *anomaly detection* and *signature detection*. These have been with us since the inception of the field. In short, *anomaly detection* can be defined as looking for the unexpected—that which is unusual is suspect—at which point the alarm should be raised. *Signature detection*, on the other hand, relies on the explicit codifying of 'illegal' behaviour, and when traces of such behaviour is found the alarm is raised.

Anomaly Detection Taking the basic outline of detection and estimation theory laid out in the beginning of this section, we can elaborate upon it in describing these methods. In contrast to the model in figure 4, where we have knowledge of both *H0* and *H1*, here we operate without any knowledge of *H1*. Thus we choose a region in our observation space—X in figure 3. To do so, we must transform the observed, normal behaviour in such a manner that it makes sense in our observation space context. The region X will contain the transformed normal behaviour, and typically also behaviour that is 'close' to it, in such a way as to provide some leeway in the decision, trading off some of the detection rate to lower the false alarm rate. The detector proper then flags all occurrences of x in X as no alarm, and all occurrences of x *not* in X as an alarm. Note that X may be a disjoint region in the observation space.

Signature Detection The signature detector detects evidence of intrusive activity irrespective of the model of the background traffic; these detectors have to be able to operate no matter what the background traffic, looking instead for patterns or signals that are thought by the designers to stand out against any possible background traffic. Thus we choose a region in our observation space, but in this instance we are only interested in known intrusive behaviour. Thus X will here only encompass observations that we believe stem from intrusive behaviour plus the same leeway as before, in this case trading off some of the false alarm rate to gain a greater detection rate in the face of 'modified' attacks. During detector operation we flag all occurrences of x in X as an alarm, and all other cases as no alarm. X here may also consist of several disjoint regions, of course.

Comparison with Bayes Optimal Detectors It is an open question to what degree detectors in these classes can be made to, or are, approximate Bayes optimal detectors. In the case of non-parametric intrusion detectors— detectors where we cannot trade off detection rate for false alarm rate by varying some parameter of the detector—merely studying the receiver operating characteristics (ROC) curve cannot give us any clue as to the similarity to a Bayes optimal detector. This is because the ROC curve in this case only contains one point, and it is impossible to ascertain the degree to which the resulting curve follows the optimal Bayes optimal detector. (See [Axe05a], for a brief introduction to ROC curves, [Tre68] for a thorough one).

Summary

The dichotomy between *anomaly detection* and *signature detection* that is present in the intrusion detection field, vanishes (or is at least weakened) when we study the problem from the perspective of classical detection theory. If we wish to classify our source behaviour correctly as either *H0* or *H1*, knowledge of both distributions of behaviour will help us greatly when making the intrusion detection decision. Interestingly, early on only few research prototype took this view [Lee99, BAJ03]; all others were firmly entrenched in either the *H0* or *H1* camp. It may be that further study of this class of detectors will yield more accurate detectors, especially in the face of attackers who try to modify their behaviour to escape detection. A detector that operates with a strong source model, taking both *H0* and *H1* behaviour into account, will most probably be better able to qualify its decisions by stating strongly that this behaviour is not only known to occur in relation to certain intrusions, and further is not a known benign or common occurrence in the supervised system.

The detectors we have developed for this thesis (except for the one in [Axe05b]) all take both *H0* and *H1* into account.

5 Rationale and Problem Statement

As we shall see later, a significant problem with intrusion detection systems, is the high number of false alarms [Axe05a].[6] This is perhaps not surprising when making an analogy with the common burglar alarm. Burglar alarms operate under a very restricted security policy. Any activity whatsoever on the premises is suspicious. Intrusion detection systems on the other hand are active when the computer system is in full operation. There is much benign activity taking place. The analogy with a burglar alarm is apt then, and serve to explain the high number of false alarms. In the shop lifting scenario however, an ordinary burglar alarm would not be appropriate since there would be a multitude of normal, benign activity (the shopkeeper even encouraging this). The shoplifting problem is currently addressed among other things by *surveillance*, i.e. human supervision of the potential shoplifters. The human, using her senses unaided, is too expensive to employ directly, and therefore technology is brought to bear in the form of video cameras, video recorders, etc. Taking this analogy more literally leads to the idea of applying some form of *information visualisation* to the intrusion detection problem as computers do not have a generally interesting visual form. This thesis draws on techniques from this field. It should be noted that the operator of any intrusion detection system *must* have a rudimentary understanding of the assets that need protection and common ways of attacking said assets. To assume otherwise would be akin to staff the metal detectors at airports with personnel completely oblivious to the dangers of different types of firearms and sharp implements and at least the most common forms of evading the detector. A metal detector, however sophisticated, would not be of much use in such a situation, as the operator would not be able to evaluate the output. That is not to say that the staff necessarily would need to know how to *build* a metal detector. We aim for the same level of sophistication of the users of our tools.

Thus the *main* point of the work in this thesis is as follows: Given that false alarms are a problem with current approaches to intrusion detection: How do we apply information visualisation to aid the operator in identifying false alarms?

6 Introduction to Visualisation

Good introductions to this area are [Spe01] and [CMS99]. This section lends heavily from the latter.

[6]It has long been known in security circles that ordinary electronic alarm systems should be circumvented during the normal operation of the facility, when supervisory staff are more likely to be lax because they are accustomed to false alarms [Pie48].

The human mind's cognitive skills are limited. By cognition we mean "The acquisition or use of knowledge" [CMS99, p. 6]. To overcome the shortcomings of our limited cognitive skills, humans have invented external aids that help us in cognitive tasks. These aids are often in graphical form (c.f. doing longhand arithmetic using pencil and paper, where we aid limited short term memory by keeping intermediate results as glyphs on paper). The use of the external world in the aid in cognitive tasks is sometimes called "external cognition" [CMS99, p. 1]. The use of external aid is central to better utilise the limited human cognitive skills:

> ... visual artifacts aid though; in fact, they are completely entwined with cognitive action. The progress of civilization can be read in the invention of visual artifacts, from writing to mathematics, to maps to printing to diagrams, to visual computing. As Norman says, "The real powers come from devising external aids that enhance cognitive abilities." Information visualization is about just that—exploiting the dynamic, interactive, inexpensive medium of graphical computers to device new external aids enhancing cognitive abilities. It seems obvious that it can be done. It is clear that the visual artifacts ... have profound effects on peoples' abilities to assimilate information, to compute with it, to understand it, to create new knowledge. Visual artifacts and computers do for the minds what cars do for the feet or steam shovels do for the hands. But it remains to puzzle out through cycles of system building and analysis how to build net next generation of such artifacts. (Card et. al. [CMS99, pp. 5–6]).

Information visualisation then is the use of computers to give abstract data an interactive visual form. By *abstract* we mean that the data is non-physical in origin. One such origin of data that we deal with exclusively in this thesis is log data from computer systems, especially access log data from webservers.

The information visualisation process can be divided into three distinct steps:

> **Data transformations** map *Raw Data*, that is data in some idiosyncratic format into *Data Tables*, relational descriptions of data extended to include metadata.
>
> **Visual mappings** transform *Data Tables* into *Visual Structures*, structures that combine spatial substrates, marks, and graphical properties. Finally,
>
> **View transformations** create *Views* of the Visual Structure by specifying graphical parameters such as position, scaling, and clipping.
>
> User interaction controls parameters of these transformations, restricting the view to certain data ranges, for example, or changing the nature of the transformation. The visualizations and their controls are used in the service of some task. (Card et. al. [CMS99, p. 17]).

As a research area, information visualisation is now some twenty years old (even though the visual presentation of data of course is much older) with rapid development in the last ten years or so due to the advent of cheap personal computers with substantial processing and graphics capabilities. As such this thesis follows one trend in the area away from pure information visualisation studies with the goal of developing new generally applicable visualisation strategies towards application of the principles developed in the past to new problem domains.

7 Overview of Appended Papers

As this thesis is a collection thesis it is important to explain how the included papers relate to each other. The theme is one of *false alarm suppression*, i.e. how do we make the system as a whole (including the operator) able to handle false alarms? The first paper (Paper A) begins by motivating why false alarms are a problem, and always will be, by explaining the issue of false alarms by way of the base-rate fallacy. The following papers investigate the application of information visualisation to the intrusion detection problem and how this helps the operator more easily identify false alarms (and detect true alarms). First the visualisation of the output of an anomaly detection system—applied to unique web access request strings—is studied in Paper B. This study is successful but has drawbacks which are addressed in the following two papers (Paper C and Paper D) which develop successively more complex directed self learning detectors with integrated visualisation to enable the operator to detect false (and true) alarms but also to see a visual representation of the training process, and interactively alter it. The last paper (Paper E) then picks up where the previous left off by presenting a method for correlation of malicious web access request strings once they have been detected (by the previous methods for example) so that the operator may identify the entities making the requests.

7.1 Paper A: The Base-Rate Fallacy and the Difficulty of Intrusion Detection

Many different demands can be made of intrusion detection systems. An important requirement is that it be *effective*, in other words that it should detect a substantial percentage of intrusions into the supervised system while still keeping the *false alarm* rate at an acceptable level. This paper aims to demonstrate that, for a reasonable set of assumptions, the false alarm rate is the limiting factor for the performance of an intrusion detection system. This is due to the base-rate fallacy phenomenon, that to achieve substantial values of the Bayesian detection rate, $P(Intrusion|Alarm)$—which provides a measure of the extent to which an alarm is the result of an actual intrusion—we have to achieve a very low false alarm rate. A selection of reports on intrusion detection performance are reviewed, and the conclusion is reached that there are indications that at least some types of intrusion detection have far to go before they can attain such low false alarm rates.

This paper demonstrates that intrusion detection in a realistic setting is perhaps harder than previously thought. This is due to the base-rate fallacy problem, because of which the factor limiting the performance of an intrusion detection system is not the ability to identify intrusive behaviour correctly, but rather *its ability to suppress false alarms*. A very high standard, less than 1/100000 false alarms per 'event' given the stated set of circumstances, has to be reached for the intrusion detection system to live up to these expectations as far as *effectiveness* is concerned. The cited studies of intrusion detector performance that were plotted and compared indicate that anomaly-based methods may have a long way to go before they can reach these standards because their false alarm rates are several orders of magnitude larger than what is required. Turning to the case of signature based detection methods the picture is less clear. One detector performs well in one study—and meets expectations—but is much less convincing in another, where it performs on a par with the anomaly-based methods studied. Whether some of the more difficult demands, such as the detection of masqueraders or the detection of novel intrusions, can be met without the use of anomaly-based intrusion detection is still an open question.

It should be noted that the assumptions made above hinges on the *operator's* ability to deal with false alarms. Studies in psychology indicate that humans are typically ill equipped to effectively supervise complex systems in an environment where the monitoring systems produce alarms that turn out not to be real causes for concern [RDL87, WH99]. These result indicate that the more complex the system, and the less the human feels aware of *how* the system is operating (i.e. to what degree it seems 'automagical') the less effective the operator becomes in correctly identifying problematic situations and taking the necessary corrective action. The results seem remarkably stable regardless of the type of system under study, whether in the process industry (paper mill, steel mill, aluminium smelting facility etc.) [RDL87], or air craft cockpit or nuclear power plant control room [WH99].

Thus it is reasonable to assume that if we cannot reduce the false alarm rate of current intrusion detection systems, it would be beneficial to provide the operator with tools to help her address them, i.e. by identifying them, discarding them, and ultimately correcting the intrusion detection system that produced them. This will in effect provide the operator with more insight into how the intrusion detection system is operating. Thus in this thesis the application of information visualisation to the problem of making the operator more effective has been studied, especially when it comes to the question of handling false alarms. Paper A provides the rationale for addressing the false alarm problem.

7.2 Paper B: Visualising Intrusions: Watching the Webserver

Following the rationale in the previous section, applying visualisation to the output of a traditional anomaly based intrusion detection system could help the operator make sense of the output, helping her differentiating the false alarms from the true alarms and so make her more effective. This would combine advantages of both methods while mitigating their drawbacks:

Anomaly detection Being able to detect novel intrusions, i.e. new methods of intrusions we do not know about before, while as a consequence of detecting unusual behaviour instead of known violations, having a high false alarm rate.

Visualisation Increasing the operators insight into the data being presented, but not being able to display the amounts of data that intrusion detection systems typically deal with in a meaningful way.

To that end, a very simple anomaly detection based log reduction system with a 3D visualisation component was applied to the realistically sized log of a web server. The log was from the month of November of 2002 and came from the webserver of the Computer Science department at Chalmers. It contained on the order of 1.2 million accesses, comprised of about 220000 unique access requests. The anomaly based log reduction scheme worked by cutting up the unique requests into elements as per the HTTP specification, and then counting the frequencies of occurrences of the elements, assigning a score to the request as a whole by calculating the average of the element scores. A low score signifying that the request was comprised of *unusual* elements, and hence in some sense was anomalous. It should be noted that the element frequencies were maximised at a frequency of 1000, as a few very frequent elements otherwise would have completely dominated the score of the access requests they were a part of. The cut-off score was motivated visually. When applying an anomaly based intrusion detection system we would then have settled on a threshold score and marked all the requests with a lower score as anomalous. However, in this case instead we choose as many of the lowest scoring access requests as we thought we could handle with the visualisation component, irrespective of their score. So we did in fact not implement an anomaly based intrusion detection system, but instead an anomaly detection based log reduction scheme.

The visualisation component then performed the same separation into elements as the log reducer, but instead visualised the elements as a general graph, with directed edges connecting the elements. I.e. given an access request such as 'GET /index.html HTTP/1.0', it would first be cut up into the nodes: 'GET', 'index.html', 'HTTP' and '1.0', and then the edges between 'GET' and 'index.html' etc. would be added. Note that the resulting graph is a general graph (e.g. not necessarily acyclic etc.), where a node may be a part of several access requests at different places. The resulting (mostly treelike) structure was visualised as a 3D graph and even while the first feature that stood out turned out to be an attack, later investigation indicated that the visualisation was better suited to help identify benign requests than malicious requests. This was just as well, as the majority of the log was comprised of benign access requests. Even though a direct comparison between the false alarm rates defined in Paper A and the results in this paper was impossible, the false alarm rate was orders of magnitude worse than required in Paper A but the visualisation component was effective in helping the operator identifying the false alarms and hence by a process of elimination, the true alarms.

Many interesting attempted intrusions were found in the data and were divided into some seven classes. While the log reduction scheme did not have a perfect detection rate, it did not miss any one class completely, so evidence of all *types* of attacks was preserved. To ascertain the detection rate, all the 220000 access requests were classified by hand, an *extremely* tedious task.

7.3 Paper C: Combining a Bayesian Classifier with Visualisation: Understanding the IDS

While the method that is presented in Paper B is workable it does have some drawbacks. The main drawback pertains to the log reduction scheme. While it works as it stands, it does so without lending the user any real insight into its operation, the graphs motivating the cut off frequencies notwithstanding. Furthermore, it cannot be configured by the user, should e.g. the visualisation component have given any insight into how its performance could be improved. Also it is a pure anomaly based system and as we have previously mentioned in section 4.2 for better detection accuracy an intrusion detection system ought to have a model of both benign and malicious behaviour.

An anecdote from the paper serves to motivate the approach taken:

> When the author first started using the Bayesian spam filter recently added to the *Mozilla* ('http://www.mozilla.org') email client, the filter seemed to learn the difference between spam and non-spam email with surprisingly little training. It was not until some rather important email was misclassified as 'spam' that it was realised that what the filter had actually learnt, was not the difference between spam and non-spam, but between messages written in English and the author's native tongue. In fairness given a few more benign examples of English messages the system was successfully retrained and was again correctly classifying email, but some rudimentary insight into exactly what the system had learnt would have made us more sceptic of the quality of the classification, even though the classifier seemed to operate perfectly judging by the output.

To attempt to address this situation, a naive Bayesian classifier was developed. It was modelled after the now common spam filters first popularised by Paul Graham [Gra02]. The main reasons for this choice was that these classifiers have had some success in the similar field of spam detection and they also meet the requirement that they build a complete model given the available evidence, taking both benign and malicious clues into account. In fact the classifier cannot operate *without* both benign and malicious examples. In order to explain how the visualisation of the classifier works we will first have to go into a bit more detail explaining how the classifier actually operates. Naive Bayesian classification revolves around a scenario where that which we wish to classify can be divided into records (i.e. pieces of mail in the case of spam classification) that can be marked as benign or malicious as a whole. The records must furthermore be divisible into tokens (typically words in the case of spam classification, but also message headers etc). Bootstrapping the classifier consist of feeding it records the user has marked either benign or malicious. The principle behind the classifier is thus one of *directed self learning*. In more detail, the classifier operates by counting the frequencies of occurrence of the tokens that makes up the good and bad records. The frequency counts for each token can be interpreted (by the application of some conversion formula) as a probability indicating the relative maliciousness of the token, i.e. the probability that the token indicates a bad context. Let us call this probability P_l (for *local probability*). The probability that the same token is indicative of a good record is then of course simply $1 - P_l$. In order to classify a previously unseen record the classifier weighs together the evidence provided by the local probabilities of the tokens that makes up the record, using a neutral 0.5 probability if the token has not been seen previously. This result in a total probability for the record as a whole that can be interpreted analogously with the local probability. The weighing is performed by a naive version of the Bayesian chain rule. As the local probabilities do not actually take the dependant probabilities of the other tokens into account (as that would lead to a state explosion that would be prohibitively costly in terms of memory and processing resources) the classifier earns the moniker *naive*. It is also worth noting that in order for Bayes's theorem to hold the probabilities taken into account ought to be independent of each other. This restriction is often relaxed in practice.

Given this classifier one realises that the learning it does is condensed into the local probabilities. Therefore it was decided to try the heatmap visualisation principle. The heatmap visualisation works by mapping a continuous variable onto the colour wheel. From green via yellow, to red. In this case we map local probability from 0.0 being green to 1.0 being red, with 0.5 indicated by yellow onto the background of the textual representation of the tokens. This lends the operator visual insight into the evidence the classifier is basing its conclusion on. In the prototype developed the records are displayed one to a line with the total score also displayed heatmapped to the left of the record. As the resulting visualisation can also lend insight into the training process and not merely the output of the classifier once it is trained, a natural step is to make it interactive. The user can mark a record benign or malicious and immediately see the effect this update has on the classifier as a whole through the visualisation of the record and other records also visible. To help the user keep track of the training status of the record, a coloured marker is placed first on the line to indicate whether this record has been trained as 'good', 'bad' or not been part of training at all. In order to aid in training, the operator can sort the display according to training status e.g. to easily identify records that have been trained but still are misclassified. To effect actual detection the operator can import new records and sort on total score, which will single out the records most likely to be indicative of malicious activity.

In order to test the complete prototype, named *Bayesvis*, it was trained on the web server access request data described in Paper B. A training strategy of *train until no false positives* was adopted, i.e. the system was first trained on all the previously identified malicious requests and then enough of the benign requests were trained to make all the benign training request have an overall score lower than 0.5, signifying that they are benign. The resulting classifier was then tested on the available logs from the same web server for the months following November, i.e. December through February. While the December log contained on the order of the same number of access requests, many of these were identical to the November log and were removed from it. The same applied for the following logs, i.e. many of the requests in the January log were identical to requests seen in either the November or December logs. Thus the actual logs the classifier was tested on decreased in size as the experiment wore on. The results were promising, the number of false alarms was reasonable and because of the visualisation they were quite easily identifiable, as the operator could (the author would argue) see what tokens the classifier found objectionable. An access request consisting of predominantly green tokens with one or two red mixed in (perhaps as arguments to cgi scripts) would almost certainly indicate a false alarm. As the operator has knowledge of the meaning of the actual tokens in context (something the classifier itself is devoid of) she is poised to make a qualitative evaluation of the output of the classifier. The detection capabilities were also sufficient, the detector clearly managed to generalise its evidence from the training session to detect variations of previously known attacks.

7.4 Paper D: Visualising the Inner Workings of a Self Learning Classifier: Improving the Usability of Intrusion Detection Systems

A problem with the classifier described in Paper C is that it is simple (simplistic even) in that it neither takes the order nor the context of the tokens into account. While in fairness the naive Bayesian classifier shows sufficient performance on the data it was tested on, there is data that it cannot be tested on given the above mentioned limitations, and as it did not perform flawlessly there is furthermore some room for improvement.

In order to address these two points a more complex classifier based on two popular spam filters: CRM-114 [Yer04] and Spambayes [MW04] was developed. Our classifier works with the same notions of tokens, records, directed training etc. as the naive Bayesian classifier in Paper C. It works by sliding a window of length six over the input and considering as features all the possible combinations of the tokens in the window considering *skips*, i.e. the order of the tokens is preserved, but they may not be counted as present. E.g. the window “The quick brown fox jumps over”, gives rise to (among others) the features “The <skip> <skip> fox jumps over” and “<skip> quick brown fox jumps”, etc. until all possible combinations (i.e. the *powerset* excluding the empty set) has been generated. These features are first processed much as the tokens are in the naive Bayesian classifier, i.e. their presence in benign and malicious contexts are counted and the statistics allowed to influence a local probability. In this case the formula of the local probability is more sophisticated giving less weight to features for which low counts have been observed (i.e. for which there is less total evidence). However, as this would give equal weight to features that have many tokens present (i.e. few skips) as to features that have fewer tokens present, a superincreasing weight function is applied that modifies the local probabilities according to the formula: $W = 1/2^{2(n-1)}$. I.e. a feature with more tokens present can outweigh all of its ’children’—i.e. with skips in the positions that the feature have tokens—combined. This is believed to make the classifier non-linear i.e. a classifier that could e.g. learn that ’A’ and ’B’ in isolation were both indicative of a malicious context, but ’AB’ together was indicative of a good context, something the naive Bayesian classifier could not. Further study is required to confirm whether this scheme could indeed lead to a classifier that is non-linear. So far our classifier has been solely influenced by the CRM-114 classifier.

Given the local probabilities they have to be combined into an overall score indicating the probability the record is indicative of malicious activity much as in Paper C. To accomplish this a chi square test (or rather *two* tests) as in the

SpamBayes classifier was applied to the local probabilities. The local probabilities of the features are tested against the two hypotheses of them being indications of benign or malicious behaviour, thus resulting in two probabilities. These are then combined into one probability, taking the support for both hypotheses into account. For the situations where there is either strong evidence of malicious activity and none of benign (or vice versa) the situation is straightforward giving rise to the probability of either 1.0 or 0.0 respectively. Likewise, in the instance where we do not have much evidence of either; giving rise to the overall score of 0.5. The special case where we have equal evidence of *both* malicious and benign activity is interesting though, as that must also give rise to the overall score of 0.5, but of course still being a very different situation from the case where we do not have much evidence of either kind. As a result, all three probabilities of the classifier are returned to the application for visualisation.

Visualising this classifier was much more problematic than the naive Bayesian classifier as there was much more in the way of features and a more complex decision process to take into account. While we still deal with probabilities some form of heatmap could still be applied, no single token now have a score and the simple *line per record* display of Bayesvis could not be applied directly. Thus it was decided to apply the principle of *overview and detail*, whereby the data is displayed in progressively more detail as the user selects various regions of interest. So, at the most detailed level we have the actual features that makes up a record. They were visualised much as in Bayesvis, i.e. the local probability of feature was heatmapped in the lower third part of the screen. Note that these are the only features that actually matter in the classification. The next display (in the middle third) summarises the windows that the features are part of (and also serves to let the operator select a window for feature display). As each token in a window may be part of several features a straight heat map does not work here. Thus a (new) method of summarising contributions from several features had to be devised. The author chose to select all the features that had the token in question in the same position as in the window and subject the local probabilities of the features to the chi square test. The overall resulting score was then allowed to select the hue of the background in the same heatmap fashion as before. An added feature is that of a weight (to produce some form of quality indicator, i.e. to give an indication of how certain the classifier is of the classification) of the probabilities indicating the test for malice and lack thereof. This weight is allowed to affect the whiteness of the selected hue, i.e. the more uncertain the classification is, the more *washed out* (i.e. closer to the white point) the colour appears. At the top most level, at the top third of the screen, the entire record is visualised (much in the same way as for the Bayesvis prototype) using the same method for generating a summary display of the tokens, with the added complication that a token can now be part of several windows as well. In the last two summary displays an added feature over that of the Bayesvis prototype is that the tokens that we have not seen before are not mapped onto a yellow background (signifying that they are neutral as far as classification is concerned) but instead on a grey background to visually differentiate it from the case where the token is truly neutral, i.e. the classifier has seen it in an equal amount of benign and malicious contexts.

In order to evaluate the resulting prototype, called *Chi2vis*, it was trained on and applied to the November 2002 log as that had been fully evaluated for benign and malicious accesses. This because the more extensive (data-wise) evaluation Bayesvis was subjected to turned out to be unwieldy in practice. As is customary in classifier research, the system was trained on a randomly chosen ten percent subset from the seven classes of attacks (though at minimum one request) and the benign requests. The classifier was then evaluated on the remaining data for true and false positives and negatives. The resulting detector faired well, and the visualisation helped the operator identify false alarms, more so than Bayesvis, in that Chi2vis lets the operator see the (limited) context in which the training took place so that the operator gained extra insight into what the detector found objectionable and why that may not hold in the particular case. Chi2vis was also tested on traces of operating system calls. Unfortunately there was really not enough data available to train Chi2vis sufficiently but it still managed to correctly detect at least some (visually very uninteresting) bad traces, even though the performance of Chi2vis on this data set was not spectacular. To complete the evaluation, Bayesvis was then tested under the same circumstances to make a comparison possible. While Bayesvis required less benign training before the *train until no false positives* strategy was fulfilled, this was reflected in a higher false alarm rate and lower detection rate. Bayesvis faired almost universally worse on all aspects in comparison to Chi2vis.

7.5 Paper E: Visualization for Intrusion Detection: Hooking the worm.

This paper was the first paper the author wrote in the field of applying visualisation to intrusion detection. In it the access requests (in this case the complete records, not just the unique request strings) to a small personal web server was studied with a visualisation method called the parallel coordinate plot [Ins97]. The hypothesis was that the operator should be able to detect malicious accesses to the webserver—most notably from the various *worms* that crept around the internet at the time—and be able to correlate them to each other. It should be noted that the web server in this case was much smaller than the ones latter studied in the previously summarised papers, and did not have nearly the same number of accesses to it. It furthermore did not have much in the way of benign access requests. To further complicate the study of this web server, it used authentication for all accesses and hence all worms trying to access it got an error return. To accomplish the detection and classification of the worms (and other entities) that accessed the server a selection of variables that did not leak (directly or indirectly) information about the authentication process was visualised using the parallel coordinate plot. The parallel coordinate plot maps a point in multidimensional space onto the plane by placing all the axes vertically and equidistant and plotting the components of the point onto each respective axis, connecting the components with straight line segments. The detection and identification was done as a trellis plot, i.e. one of the variables (the unique access request string as in the previous papers) was held constant and a separate parallel coordinate plot generated for each unique access request. This meant that the patterns of access for the various unique request strings could be visually correlated to each other, i.e. entities making different requests but at similar times, from similar systems etc. could be identified and the access requests correlated.

Relatively little support for the hypothesis that malicious entities could be detected was found. While many of the worms showed markedly different access patterns from the benign patterns it is difficult to say how that would hold up given a larger web site with more benign traffic. The malicious access requests (and the benign) could be successfully correlated to each other though. In fact, one entity making access requests very similar to then popular worms was markedly different visually and turned out to be a then largely unknown instance of the application of a tool for breaking into web sites. Most security sources erroneously referred to this access request as coming from the worm. The visualisation made it easy to differentiate this access pattern from the others. Several other malicious access patterns were found. As the previous work that dealt with web access requests stopped when the *types* of malicious accesses were found the method investigated in this work nicely complements those methods in that with the approach presented here the investigation could continue and the actual entities making the request could be identified.

However, in the version of Paper E that was published as [Axe03], one malicious pattern slipped by the eyes of the author. This was because the pattern consisted of two separate unique access request strings and only a few accesses overall and was therefore similar to the benign traffic to the web server. This pattern turned out to be from the same tool as mentioned in the previous paragraph, but run with different options. The reason for this pattern escaping the author the first time around is illustrative as it makes the main drawback of all visualisation work clear: Any visualisation can only be as successful as the person viewing it. If that person falters thorough inattentiveness (perhaps brought on by tiredness, stress or boredom for example) then the visualisation cannot ameliorate the situation. Putting the human operator back into the driver's seat, so to speak, has both the benefit of putting the human in control of the events, but also the drawback of having to come to terms with human fallibility.

8 Results in Perspective

It is the author's opinion that computer security must rely mainly on different *perimeter* defences. These defences can and probably should be employed in depth. That is, just because one has been granted—or otherwise gained—access through the outer perimeter, one should not have free reign of the system. The term *perimeter* should not be interpreted too literally. It need not have a more literal interpretation as in case of firewalls etc. To our mind, other approaches that serves to separate the protected entity from the attacker's zones of observation and influence also falls under this heading, such as the approach of statically (or dynamically) verify that your source code is free of security defects etc. cf. the concept of *prevention* in Halme et. al. [HB95], described in section 3.

However, no matter how well protected a system is, there will always be chinks in its armour, and thus some sort of surveillance and response system must be in place to detect and deal with intruders as and when they appear. This system can sometimes possess a high degree of autonomy as is the case with virus scanners, spam filters (using signatures of known spam) and signature based intrusion detection systems. We would argue that in the general case, dealing with the more imaginative threats, a human operator needs to be in the loop and in order to be effective there should be tool support that enables her to quickly gain an understanding of the situation. We call this the principle of *surveillance* to set it apart from more traditional intrusion detection system principles. To believe that automated systems could deal with other than the most routine threats is overly optimistic, as the attacker in many cases could analyse the defences for weaknesses and attack us there, to wit: "there's no equipment that man's ingenuity can devise that man's ingenuity can't also defeat" [KCBH96, p. 51]. No perimeter defence, however strong, will not last if it is left unguarded, providing the attacker with ample time to analyse and ultimately defeat it.

9 Related Work

The first mention in the literature of the idea to apply visualisation to the field of computer security (specifically; intrusion detection) the author is aware of is by Vert et. al. in [VFM98]. The first time the author learnt of the idea of applying visualisation to intrusion detection predates this first publication though and is when Professor Erland Jonsson noted it in a meeting in the autumn of 1996. At the time of writing the area has seen more investigation, and as such we will limit the treatment here to selected applications of visualisation in an intrusion detection setting much as ours, where the intent has been to apply scientific visualisation to help the operator gain insight into the security state of the monitored systems.

Starting with the work by Vert. et. al: That work presents a preliminary visualisation of the security state of a computer system, by way of a *Spicule*—the characteristics of which is investigated—but provides no opinion on how that security state should be calculated. More recently Erbacher et. al. [EWF02] has presented work building on the previous work by Frincke et. al. [FTM98]. This work is based on encoding information about network traffic and alarms from a network of intrusion detection sensors, as glyphs onto a stylised map of the network.

A small subfield (e.g. [ROT03, JWK02, LZHM02]) of anomaly detection and visualisation has arisen through the application of self-organising maps (also called Kohonen maps) [Koh01] to intrusion detection. The question of visualisation arises because the Kohonen map itself is a visual representation of an underlying neural network model. The work cited above shares the characteristic that they all build some neural network model of network traffic or host data and then present the resulting two dimensional scatter plot to the user. The scatter plot typically illustrates various clusters within the data. A problem here is that the interpretation of the plot is known to be quite tricky [Koh01].

Girardin et. al. [GB98, Gir99] also uses self-organising maps, but stresses the link to the human operator. They also utilize other visualisation methods in addition to the self-organising map itself, using the self-organising map as an automatic clustering mechanism. They report on successful experiments on data with known intrusions. For input data they use connection statistics etc. from TCP/IP traffic as their input data. While they study parameters of TCP/IP connections, they do not study the data transferred.

Theo et. al. [TMWZ02] visualise communication between pairs of routers on the Internet using the BGP (Border Gateway Protocol) routing protocol. Their choice of problem and visualisation techniques are different from the one presented here, and they do not delve as deeply into the analysis of the security problems they detect (they are not as clearly *security problems*), but they do visualise a greater amount of data more compactly than done in this thesis and still manage to detect anomalies in the BGP traffic. This work has later been continued by adding a *NIDES* [AFV95] based anomaly based intrusion detection component and visualising the output of the classifier together with the BGP update messages. Another view then lets the user do *what if* calculations setting different classifier parameters with visual feedback [TZT+04].

On a similar note, visualising network flows (i.e. records that contain abstract information about communication sessions between computers such as source and destination IP addresses, how many bytes were transferred etc.) have also seen some work in the recent past by Yin et. al. [YYT+04]. Yin apply parallel coordinate visualisation (as we do) to selected parameters of these *netflow* records to detect anomalies in network traffic.

A quick survey of the available commercial intrusion detection systems was also made. Only two systems uses any degree of visualisation in our sense of the word. The first is *CA Network Forensics*:[7] uses N-gram clustering followed by a three dimensional visual display of the clusters. On the surface the visual representation of the data in the clusters is similar to the one presented in Paper B (i.e. a general 3D network) but while the graphs may look similar they express very different relations. There is no discussion as to the interpretation of these graphs and the underlying structure of the data is not allowed to influence the visualisation.

The second is *Lancope Therminator*[8] based on the Therminator project [ZME04]. Therminator is a network level anomaly detection tool inspired by methods from the field of statistical physics. The anomaly detector works by building a model of network traffic as a modified Ehrenfest urn model, the parameters of which are (in addition to other processing) visualised as three dimensional bar charts, to give the user an overview of the state space of the model. The authors report on experiments where anomalies have been injected into the traffic with the corresponding diagrams clearly showing a marked difference between the anomalous event and the steady state. Unfortunately the authors do not emphasise the visualisation portion of the work presented in [ZME04] and it is difficult to ascertain the degree to which the visualisation helps the operator gain insight into exactly what caused the deviation from the normal graph even though it seems promising.

The literature in the area has recently grown to become quite extensive, and we cannot do it justice here. The interested reader is referred to [BCLY04] as a starting point.

[7]'http://www3.ca.com/Solutions/Product.asp?ID=4856'. Verified 2004-12-20.

[8]'http://www.lancope.com'. Verified 2004-12-20.

10 Conclusions and Future Work

The marriage between visualisation and intrusion detection seems at the outset a happy one. The application of visualisation seems to bring benefits in the form of increased understanding of the security state of the monitored systems while not suffering from too crippling drawbacks.

Even though the usability of intrusion detection systems and the application of the principle of surveillance to the problem has seen some interest in the last year or so, much work remains to be done. The current research (including this thesis) really only scratches the surface of the possibilities in the field. Even though early results seem very promising there still remains much research to be done by including the actual operator: Notably absent from current research are *user studies*. These are more difficult to conduct than may first be thought though. The process of classifying behaviour into malicious and benign, using approaches such as ours, is a highly skilled task (where operator training would probably have a major influence on the results). It is also a highly cognitive task, and hence difficult to observe objectively. If such studies are to be of value they would almost certainly be costly, and the state of research into how to measure and interpret the results may not be as developed as one might first think.

If the author were to single out one area presented in this thesis as the most promising for further research it would be the application of visualisation to make machine learning systems more accessible to the user. We have not found much in the literature in the way of applying visualisation to this area, and based on the early results in this thesis the area looks promising.

References

[AFV95] D Anderson, T Frivold, and A Valdes. Next-generation intrusion-detection expert system (NIDES). Technical Report SRI-CSL-95-07, Computer Science Laboratory, SRI International, Menlo Park, CA 94025-3493, USA, May 1995.

[ALGJ98] Stefan Axelsson, Ulf Lindqvist, Ulf Gustafson, and Erland Jonsson. An approach to UNIX security logging. In *Proceedings of the 21st National Information Systems Security Conference*, pages 62–75, Crystal City, Arlington, VA, USA, 5–8 October 1998. NIST, National Institute of Standards and Technology/National Computer Security Center.

[And80] James P. Anderson. Computer security threat monitoring and surveillance. Technical Report Contract 79F26400, James P. Anderson Co., Box 42, Fort Washington, PA, 19034, USA, 26 February revised 15 April 1980.

[Axe00a] Stefan Axelsson. The base-rate fallacy and the difficulty of intrusion detection. *ACM Transactions on Information and System Security (TISSEC)*, 3(3):186–205, 2000.

[Axe00b] Stefan Axelsson. A preliminary attempt to apply detection and estimation theory to intrusion detection. Technical Report 00–4, Department of Computer Engineering, Chalmers University of Technology, SE–412 96, Göteborg, Sweden, March 2000.

[Axe03] Stefan Axelsson. Visualization for intrusion detection: Hooking the worm. In *The proceedings of the 8th European Symposium on Research in Computer Security (ESORICS 2003)*, volume 2808 of *LNCS*, Gjøvik, Norway, 13–15 October 2003. Springer Verlag.

[Axe04a] Stefan Axelsson. Combining a bayesian classifier with visualisation: Understanding the IDS. In Carla Brodley, Philip Chan, Richard Lippman, and Bill Yurcik, editors, *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 99–108, Washington DC, USA, 29 October 2004. ACM Press. Held in conjunction with the Eleventh ACM Conference on Computer and Communications Security.

[Axe04b] Stefan Axelsson. Visualising intrusions: Watching the webserver. In *Proceedings of the 19th IFIP International Information Security Conference (SEC2004)*, Tolouse, France, 22–27 August 2004. IFIP.

[Axe04c] Stefan Axelsson. Visualising the inner workings of a self learning classifier: Improving the usability of intrusion detection systems. Technical Report 2004:12, Department of Computing Science, Chalmers University of Technology, Göteborg, Sweden, 2004.

[Axe05a] Stefan Axelsson. Paper A: The base-rate fallacy and the difficulty of intrusion detection, 2005. In the PhD Thesis [Axe05f].

[Axe05b] Stefan Axelsson. Paper B: Visualising intrusions: Watching the webserver, 2005. In the PhD Thesis [Axe05f].

[Axe05c] Stefan Axelsson. Paper C: Combining a bayesian classifier with visualisation: Understanding the IDS, 2005. In the PhD Thesis [Axe05f].

[Axe05d] Stefan Axelsson. Paper D: Visualising the inner workings of a self learning classifier: Improving the usability of intrusion detection systems, 2005. In the PhD Thesis [Axe05f].

[Axe05e] Stefan Axelsson. Paper E: Visualization for intrusion detection: Hooking the worm, 2005. In the PhD Thesis [Axe05f].

[Axe05f] Stefan Axelsson. *Understanding Intrusion Detection Through Visualisation*. PhD thesis, School of Computer Science and Engineering, Chalmers University of Technology, Göteborg, Sweden, January 2005. ISBN 91-7291-557-9.

[BAJ03] Emilie Lundin Barse, Magnus Almgren, and Erland Jonsson. Consolidation and evaluation of ids taxonomies. In *Proceedings of the eighth*

Nordic Workshop on Secure IT systems (NordSec 2003), Gjøvik, Norway, October 2003.

[Bar04a] Emilie Lundin Barse. Extracting attack manifestations to determine log data requirements for intrusion detection. Technical Report 04-01, Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, June 2004.

[Bar04b] Emilie Lundin Barse. *Logging for intrusion and fraud detection*. PhD thesis, School of Computer Science and Engineering, Chalmers University of Technology, Göteborg, Sweden, 2004.

[BCLY04] Carla Brodley, Philip Chan, Richard Lippman, and Bill Yurcik, editors. *VizSEC/DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, Washington DC, USA, 2004. ACM Press.

[CEC91] Commission of the European Communities. *Information Technology Security Evaluation Criteria*, June 1991. Version 1.2.

[CMS99] Stuart K. Card, Jock D. MacKinlay, and Ben Shneiderman. *Readings in Information Visualization—Using Vision to Think*. Series in Interactive Technologies. Morgan Kaufmann, Morgan Kaufmann Publishers, 340 Pine Street, Sixth Floor, San Fransisco, CA 94104-3205, USA, first edition, 1999. ISBN 1-55860-533-9.

[EWF02] Robert F. Erbacher, Kenneth L. Walker, and Deborah A. Frincke. Intrusion and Misuse Detection in Large-Scale Systems. *Computer Graphics and Applications*, 22(1):38–48, January 2002.

[Fra94] Jeremy Frank. Artificial intelligence and intrusion detection: Current and future directions. Division of Computer Science, University of California at Davis, Davis, CA. 95619, 9 June 1994.

[FTM98] Deborah A. Frincke, Donald L. Tobin, and Jesse C. McConnell. Research Issues in Cooperative Intrusion Detection Between Multiple Domains. In *Proceedings of Recent advances in intrusion detetection RAID'98*, 1998.

[GB98] Luc Girardin and Dominique Brodbeck. A visual approach for monitoring logs. In *The Proceedings of the 12th Systems Administration Conference (LISA '98)*, pages 299–308, Boston, Massachusetts, USA, 6–11 December 1998. The USENIX Association.

[Gir99] Luc Girardin. An eye on network intruder-administrator shootouts. In *The Proceedings of the Workshop on Intrusion Detection and Network Monitoring*, Santa Clara, California, USA, 9–12 April 1999. The USENIX Association.

[Gol00] Dieter Gollmann. On the verification of cryptographic protocols. Presentation at Karlstad University, 11 February 2000.

[Gra02] Paul Graham. A plan for spam. http://www.paulgraham.com/spam.-html, August 2002.

[HB95] Lawrence R. Halme and Kenneth R. Bauer. AINT misbehaving—A taxonomy of anti-intrusion techniques. In *Proceedings of the 18th National Information Systems Security Conference*, pages 163–172, Baltimore, MD, USA, October 1995. NIST, National Institute of Standards and Technology/National Computer Security Center.

[HL93] Paul Helman and Gunar Liepins. Statistical foundations of audit trail analysis for the detection of computer misuse. *IEEE Transactions on Software Engineering*, 19(9):886–901, September 1993.

[Ins97] Alfred Inselberg. Multidimensional Detective. In *Proceedings of InfoVis'97, IEEE Symposium on Information Visualization*, pages 100–107. IEEE Information visualisation, IEEE, 1997.

[Jon98] Erland Jonsson. An integrated framework for security and dependability. In *Proceedings of the New Security Paradigms Workshop 1998*, Charlottesville, VA, USA, 22–25 September 1998.

[JWK02] Chaivat Jirapummin, Naruemon Wattanapongsakorn, and Prasert Kanthamanon. Hybrid neural networks for intrusion detection system. In *Proceedings of The 2002 International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC 2002)*, pages 928–931, Phuket, Thailand, 16–19 July 2002.

[KCBH96] Andrew Kain, Ken Connor, Paul Brown, and Neil Hanson. *SAS Security Handbook*. William Heinemann, Reed Intl. Books Ltd., Michelin House, 81 Fulhamn Rd., London SW3 6RB, first edition, 1996. ISBN 0-434-00306-9.

[KMT04] Kevin S. Killourhy, Roy A. Maxion, and Kymie M. C. Tan. A defence-centric taxonomy based on attack manifestations. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN 2004)*, Florence, Italy, June 2004.

[Koh01] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer Verlag, Third edition, 2001. ISBN 3–540–67921–9, ISSN 0720–678X.

[LB98] Terran Lane and Carla E. Brodie. Temporal sequence learning and data reduction for anomaly detection. In *5th ACM Conference on Computer & Communications Security*, pages 150–158, San Francisco, California, USA, 3–5 November 1998.

[LBMC94] Carl E Landwehr, Alan R Bull, John P McDermott, and William S Choi. A taxonomy of computer program security flaws. *ACM Computing Surveys*, 26(3):211–254, September 1994.

[Lee99] Wenke Lee. A data mining framework for building intrusion detection models. In *IEEE Symposium on Security and Privacy*, pages 120–132, Berkeley, California, May 1999.

[LJ97] Ulf Lindqvist and Erland Jonsson. How to systematically classify computer security intrusions. In *Proceedings of the 1997 IEEE Symposium on Security & Privacy*, pages 154–163, Oakland, CA, USA, 4–7 May 1997. IEEE, IEEE Computer Society Press, Los Alamitos, CA, USA.

[LMPT98] Ulf Lindqvist, Douglas Moran, Phillip A Porras, and Mabry Tyson. Designing IDLE: The intrusion data library enterprise. Abstract presented at RAID '98 (First International Workshop on the Recent Advances in Intrusion Detection), Louvain-la-Neuve, Belgium, 14–16 September 1998.

[LMS00] W. Lee, M. Miller, and S. Stolfo. Toward cost-sensitive modeling for intrusion detection, 2000.

[LZHM02] P. Lichodzijewski, A.N. Zincir-Heywood, and Heywood M.I. Host-based intrusion detection using self-organizing maps. In *The proceedings of the IEEE International Joint Conference on Neural Networks*. IEEE, May 2002.

[Max03] Roy A. Maxion. Masquerade detection using enriched command lines. In *International Conference on Dependable Systems & Networks (DSN-03)*, pages 5–14, San Fransisco, California, USA, 22–25 June 2003. IEEE.

[Mea93] Catherine A Meadows. An outline of a taxonomy of computer security research and development. In *Proceedings of the 1992–1993 ACM SIGSAC New Security Paradigms Workshop*, pages 33–35, Little Compton, Rhode Island, 22–24 September 1992 and 3–5 August 1993. IEEE Computer Society Press.

[MW04] T.A. Meyer and B. Whateley. SpamBayes: Effective open-source, Bayesian based, email classification system. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, USA, 30–31 July 2004.

[NP89] Peter G Neumann and Donn B Parker. A summary of computer misuse techniques. In *Proceedings of the 12th National Computer Security Conference*, pages 396–407, Baltimore, Maryland, 10–13 October 1989.

[Pie48] G. McGuire Pierce. Destruction by demolition, incendiaries and sabotage. Field training manual, Fleet Marine Force, US Marine Corps, 1943–1948. Reprinted: Paladin Press, PO 1307, Boulder CO, USA.

[PN98] Thomas H. Ptacek and Timothy N. Newsham. Insertion, evasion, and denial of service: Eluding network intrusion detection. Technical report, Secure Networks Inc., January 1998.

[RDL87] Jens Rasmussen, Keith Duncan, and Jacques Leplat, editors. *New Technology and Human Error (New Technologies and Work)*. John Wiley & Sons, March 1987.

[ROT03] Manikantan Ramadas, Shawn Ostermann, and Brett Tjaden. Detecting anomalous network traffic with self-organizing maps. In *Proceedings of the Sixth International Symposium on Recent Advances in Intrusion Detection*, LNCS, Pittsburgh, PA, USA, 8–10 September 2003. Springer Verlag.

[Spe01] Robert Spence. *Information Visualization*. ACM Press Books, Pearson education ltd., Edinburgh Gate, Harlow, Essex CM20 2JE, England, first edition, 2001. ISBN 0-201-59626-1.

[TMWZ02] Soon Tee Teoh, Kwan-Liu Ma, S. Felix Wu, and Xiaoliang Zhao. Case Study: Interactive Visualization for Internet Security. In *Proceedings of IEEE Visualization 2002*, The Boston Park Plaza hotel, Boston, Massachusetts, USA, 27 October to 1 November 2002. IEEE Computer society.

[Tre68] Harry L. Van Trees. *Detection, Estimation, and Modulation Theory, Part I, Detection, Estimation, and Linear Modulation Theory*. John Wiley and Sons, Inc., 1968.

[TZT+04] Soon Tee Teoh, Ke Zhang, Shih-Ming Tseng, Kwan-Liu Ma, and S. Felix Wu. Combining visual and automated data mining for near-real-time anomaly detection and analysis in bgp. In *VizSEC/DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 35–44, Washington DC, USA, 2004. ACM Press.

[VFM98] Greg Vert, Deborah A. Frincke, and Jesse C. McConnell. A Visual Mathematical Model for Intrusion Detection. In *Proceedings of the 21st National Information Systems Security Conference*, Crystal City, Arlington, VA, USA, 5–8 October 1998. NIST, National Institute of Standards and Technology/National Computer Security Center.

[WH99] Christopher D. Wickens and Justin G. Hollands. *Engineering Psychology and Human Performance*. Prentice Hall, third edition, September 1999. ISBN 0–32–104711–7.

[Yer04] William S. Yerazunis. The spam-filtering accuracy plateau at 99.9% accuracy and how to get past it. In *Proceedings of the 2004 MIT Spam Conference*, MIT Cambridge Massachusetts, USA, 16 January 2004. Revised 6 February.

[YYT+04] Xiaoxin Yin, William Yurcik, Michael Treaster, Yifan Li, and Kiran Lakkaraju. Visflowconnect: netflow visualizations of link relationships for security situational awareness. In *VizSEC/DMSEC '04: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 26–34, Washington DC, USA, 2004. ACM Press.

[ZME04] John Zachary, John McEachen, and Dan Ettlich. Conversation exchange dynamics for real-time network monitoring and anomaly detection. In *IWIA '04: Proceedings of the Second IEEE International Information Assurance Workshop (IWIA'04)*, page 59. IEEE Computer Society, 2004.