Survey of Intrusion Detection Research

Survey of Intrusion Detection Research

Emilie Lundin and Erland Jonsson
Department of Computer Engineering
Chalmers University of Technology
412 96 Göteborg, Sweden
Tel: +46 31 772 10 00
Fax: +46 31 772 36 63
{*emilie, Erland.Jonsson*}*@ce.chalmers.se*

Technical Report nr. 02-04

Abstract

The literature holds a great deal of research in the intrusion detection area. Much of this describes the design and implementation of specific intrusion detection systems. While the main focus has been the study of different detection algorithms and methods, there are a number of other issues that are of equal importance to make these systems function well in practice. I believe that the reason that the commercial market does not use many of the ideas described is that there are still too many unresolved issues.

This survey focuses on presenting the different issues that must be addressed to build fully functional and practically usable intrusion detection systems (IDSs). It points out the state of the art in each area and suggests important open research issues.

1 Introduction

An intrusion detection system (IDS) is an ad hoc security solution to protect flawed computer systems. It works like a burglar alarm that goes off if someone tampers with or manages to get past other security mechanisms such as authentication mechanisms and firewalls.

The major tasks of an IDS are to collect data from a computer system, analyse these data to find security relevant events, and present the results to the administrator. More or less automatic response mechanisms may also be built into the system.

Intrusion detection has been discussed in public research since the beginning of the 1980s. In the 1990s, intrusion detection became a "hot topic" and commercial IDSs started to emerge. A number of research prototypes exist, some of which have evolved into commercial products. The algorithms and methods used in the research prototypes are generally much more advanced than what is used in the commercial systems. For different reasons, the research ideas are often not considered efficient enough for the commercial market. By concentrating research on the critical issues, our research results may reach a wider audience.

This survey focuses on research in the area as a whole. It does not survey the characteristics of IDSs but instead focuses on research and tries to identify how far we have come in the different areas connected to intrusion detection (state of the art). It especially examines work done in the areas indicated to be open research questions by other authors, e.g. Axelsson [Axe98]. The aim is also to point out where research efforts are needed to carry the area forward. Papers that focus on a special issue and treat it thoroughly will be given the greatest attention in this survey. It is not intended to be exhaustive, and there are inevitably research papers that are not mentioned although they would be well worth bringing up.

The report is organised as follows. Previous surveys, i.e. other surveys and taxonomies in the intrusion detection area, are described in section 4. Each of the research areas in figure 2 are presented in depth in section 5. There is an introduction to each area, a list of research issues within that area, a presentation of the most interesting papers in the area and a discussion of the status of the area and what are still the pertinent open research issues. Section 6 discusses and points out some trends in the research surveyed. Some conclusions are given in section 7.

2 Outline of generic IDS

Figure 1 shows the components of a general IDS.

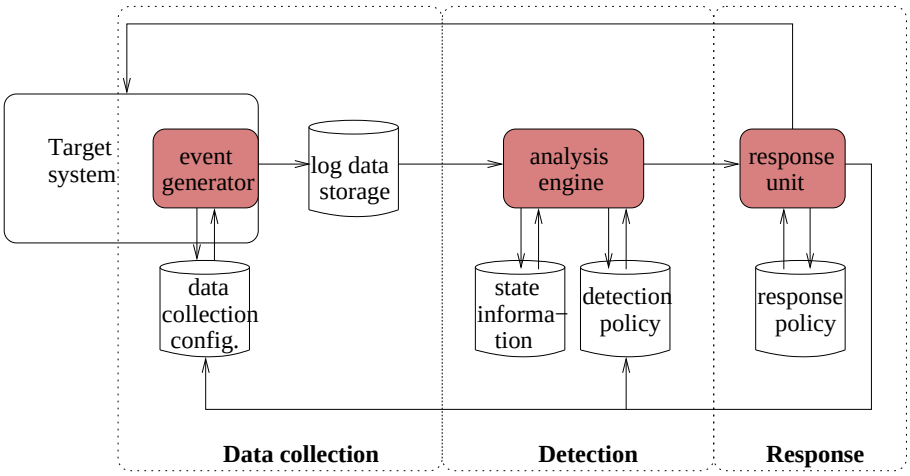

Figure 1: IDS components

The target system has mechanisms to collect various types of data, such as net-

work traffic, operating system logs and application logs. The *event generator* takes care of the logged information and may also collect data itself. Some preprocessing can be done in this component, for example to transform data to a common format and to make a rough filtering of the data. Often, a *log data storage* component is used to deposit data before it is sent to the *analysis engine*. This storage can also be used for the investigation of alarms, and perhaps also for forensics in a trial.

The *analysis engine* implements the detection algorithm. A simple detection method is to use scripts to match text strings that are unique to different intrusions. Other types of pattern matching techniques are also possible. This is similar to how most anti-virus systems work today and requires a database of "signatures" of all known malicious events we wish to detect. Thresholds for certain types of events may also be used. Signatures can consist either of a single event or of a sequence of events. Expert systems can be used to implement advanced forms of signature detection. What these methods have in common is that they are preprogrammed to detect events that are considered intrusive.

Another way to perform detection is to make the detection system distinguish between "normal" behaviour and "anomalous" behaviour in the target system. Here we create behaviour profiles for programs or users in the system and classify everything that deviate from the profiles as possibly intrusive. This can be done using simple statistics or using "intelligent" methods, such as neural networks, data mining techniques, genetic programming, and visualisation. Often, these techniques are self-learning and will update the profiles automatically. Some of these intelligent techniques can also be trained to detect known intrusive behaviour. The analysis engine may combine several detection methods to achieve a more effective detection.

The *detection policy* component contains preprogrammed information about how to detect intrusions. Here is where the intrusion signatures and thresholds are stored. Configuration information for anomaly detection and rules for what information to send to the response unit are also stored here. The *state information* database contains dynamic information used for detection. This may be state information about partially fulfilled intrusion signatures and about current behaviour in the system.

Information about events that are classified as intrusive or anomalous by the analysis engine are sent to the *response unit*. With the guidance of the preprogrammed rules in the *response policy* database, it is decided how to respond to different events. The decision may be affected by parameters such as the certainty connected to the event and the potential impact of the event. In a distributed system, the response unit may get input from several analysis engines and must thus also correlate alarms. The possible response actions are to notify the administrator, to automatically reconfigure the target system to shut the intruder out, or to provide response mechanisms to support manual response. Another response option would be to let the IDS change the data collection configuration or the detection policy to collect more information about an event in progress.

3 Intrusion detection research areas

I have identified a number of research issues in the intrusion detection area. These are shown in figure 2. As we can see here, developing intrusion detection systems is not only about finding a suitable detection algorithm but also about deciding what data to collect, how to adapt the IDS to the resources of the target system, how to test the IDS, and so on. Research results in each of these areas are necessary to build usable and well functioning intrusion detection systems.

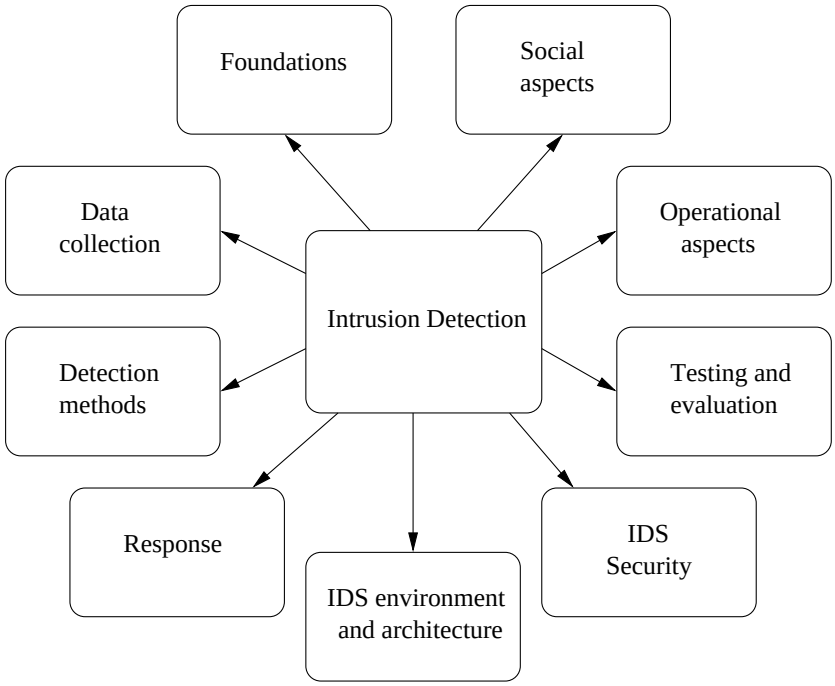

Figure 2: Intrusion detection research areas

The areas shown in the figure are issues addressed in the existing research. Some of the areas are mentioned in particular as open research questions. The components mentioned in standardisation efforts in the intrusion detection area are included as separate issues.

The following is a short description of each of these areas.

- **Foundations:** This is the fundamental work in the area. We need to find out what types of intrusions and what types of intruders we want to defend ourselves against in order to be able to build an effective defence. Here we study intrusions, intruders, and vulnerabilities.

- **Data collection:** This area is about defining what data to collect as input to the detection system. The main questions here are how to collect the data, what logging mechanisms to use, where to store the data, and the points in the system at which the data should be collected.

- **Detection methods:** The detection method is the kernel of the detection system. Here we want to find the best algorithms to distinguish between

normal and fraudulent behaviour for different situations and different input data. It may also include research on how to combine or improve methods to achieve more effective and efficient detection.

- **Reporting and response:** When the detection algorithm has indicated that a certain event or series of events are intrusive, we must bring this to the attention of the system administrator or the person responsible for security. This area deals with finding the best way to present alarms and to respond to alarms. Research issues here may include designing advanced graphical user interfaces to support manual response, creating an active response toolkit, or defining limitations for automatic response.

- **IDS environment and architecture:** Most computer systems today form more or less complex networks. It is not obvious how to distribute the IDS components in the system to achieve effective detection. Furthermore, computer systems today are often heterogeneous, which adds extra complexity when the IDS must collect data from computers running different operating systems etc. Another research issue here is the question of how to detect intrusions when other security mechanisms form obstacles, such as when network links are encrypted.

- **IDS security:** IDS security deals with how to protect the different components of the IDS against a direct attack. Data collection and storage must be protected and it must be possible to verify them. The detection and response units must also be functional under an attack and it must not possible to deceive them.

- **Testing and evaluation:** This area has to do with finding methods to compare and evaluate intrusion detection systems. This has proved to be a difficult task. Some of the issues here are how to create test data with the right properties, finding out what properties to measure, and defining what properties are important for different types of target systems.

- **Operational aspects:** Operational aspects are the technical issues important to customers using IDSs. They include for example maintenance, portability and upgradability. Issues such as interoperability between IDSs or IDS components are also important.

- **Social aspects:** The last area discussed in this report is the social aspects of deploying IDSs. Some ethical and legal issues may hinder the spreading of IDSs. The main issues that have been discussed so far in public research are privacy issues and questions about how to make the output of IDSs usable as forensics.

Each of these areas is described in depth in section 5. Some interesting research issues in each area are suggested, and the existing research in these issues is summarised and briefly discussed.

4 Previous surveys

During the period of time since intrusion detection became a familiar concept, a number of surveys and taxonomies of intrusion detection systems and research have emerged. Most of them survey existing IDSs and IDS characteristics, especially the detection techniques used. Many also mention open research issues and trends in the area. However, none has a broader focus on all the research issues in the area or presents them in a structured way.

4.1 Surveys

The first intrusion detection survey to my knowledge, was published by Lunt [Lun88]. It gives an overview of automated audit trail analysis techniques and intrusion detection systems at that time. It describes work done at Sytek and SRI, two of the first places to focus on intrusion detection research. Sytek used mainly pattern recognition techniques. They identified features that performed well (low false-alarm rate) for user statistics, file statistics, and process statistics. SRI used statistical techniques to create behaviour profiles for users and detected deviations from these profiles. The IDSs surveyed in this paper are e.g. IDES, MIDAS, NAURS, Keystroke Dynamics, and Discovery. These are probably the major part of the IDSs that existed at that time.

McAuliffe et al. [MWS+90] published a survey of the same type as Lunt's. It came out in 1990 and thus added a number of new IDSs to the survey. It also includes a discussion of the importance of the interaction with the SSO (Site Security Officer) and describes the user interface of the surveyed systems in some detail. The IDSs described in this paper are Haystack, ComputerWatch, ISOA, NSM, NADIR, and W&S.

Mukherjee, Heberlein and Lewitt [MHL94] surveyed host-based and network-based intrusion detection systems. The first class covers ComputerWatch, Discovery, Haystack, IDES, ISOA, MIDAS, and W&S. The second class includes NADIR, NSM, and DIDS. An overview is presented for each system and system organisation and operation are described. It mentions, but does not discuss, some open research issues. These are benchmarking, representation of attacks and misuse, more effective detection strategies, detection in arbitrarily large networks, and how to protect the IDS itself from attacks.

Another paper that attempts to survey current (1995) approaches to intrusion detection is [ESNP95]. It discusses network vs stand-alone and batch vs real-time analysis. The main part of the paper surveys detection methods and mentions different IDSs in this context. A great deal has happened in the intrusion detection area since this paper was written. It mentions, but does not discuss, the following future research issues.

- Testing IDS and measuring its effectiveness
- Detecting intruders in distributed systems
- Determining the most appropriate technologies to use in specific situations

- Examining computer user acceptance of intrusion detection (privacy issues)

Kumar's PhD thesis [Kum95] includes a detailed overview of detection methods and gives a great deal of references to papers reporting the different methods. A number of shortcomings in current (1995) intrusion detection research are presented. These are:

- No generic building method (substantial effort necessary to build IDS from scratch)
- Efficiency
- Portability
- Upgradability
- Maintenance
- Performance and coverage benchmarks
- No good testing method

4.2 Taxonomies

Later surveys all include taxonomies, which are used to classify the systems surveyed.

The paper by Debar et al. [DDW99] describes a taxonomy of IDS characteristics. The taxonomy focuses on four areas: detection method, behaviour on detection, audit source location, and usage frequency. They categorise contemporary IDSs using their taxonomy. Open research issues mentioned in this paper are:

- Efficiency of network and host audit sources
- Content and format of the audit trail
- New approaches to analysis for more effective detection
- Detection of insiders (abuse-of-privilege attacks)

One of the newer and more up-to-date surveys is by Axelsson [Axe98], which focuses on surveying intrusion detection systems. It includes a taxonomy of interesting IDS properties and classifies a large number of both commercial systems and research prototypes according to this taxonomy. It further discusses trends and yet to be resolved issues. The following open research questions are pointed out and thoroughly discussed.

- Active response
- How to collect, store, and prune the vast amounts of audit data
- Coverage. Lack of study of the nature of the intrusions the systems should be able to classify

- How, and how much, to report to the SSO (Site Security Officer)
- How to deal with attacks against the IDS itself

The most recent intrusion detection taxonomy I have found is [ACD+01]. The aim here is to create a more fine-grained classification scheme for IDSs. They describe IDSs by formalising their characteristics, but they limit their model of an intrusion detection system to consist of only the *intrusion detection engine* and the *sensor/information source*. The taxonomy describes the capabilities of an IDS and provides a framework for evaluating IDSs and determining whether a given IDS has the potential of detecting a given attack or whether it generates false alarms. As far as I know, they have not validated this taxonomy yet by evaluating IDSs.

4.3 Open research issues and trends

Below is a collection of the open research issues mentioned and/or discussed in the other surveys and taxonomies mentioned above.

Foundations

- A generic building method (substantial effort is required to build IDS from scratch)
- Representation of attacks and misuse
- Coverage. Study of the nature of the intrusions the systems should be able to classify

Response

- How, and how much, to report to the SSO (Site Security Officer)

IDS architecture and environment

- Detecting intruders in distributed systems

Testing and evaluation

- Testing IDS and measuring its effectiveness

Security

- How to deal with attacks against the IDS itself

Social aspects

- Examining the computer user acceptance of intrusion detection (privacy issues)

Operational aspects

- Portability
- Upgradability
- Maintenance

Some trends pointed out are that, in the middle of the 1990s, many of the IDSs went from being host-based to network-based while, at the end of the 1990s, the interest had turned back to host-based IDSs. Further, many of the first IDSs were not intended to run in a real-time environment. In recent years, most of them do.

Another trend is that the interest in anomaly-based detection fell in the second half of the 1990s, but seems to be returning in recent research.

4.4 Commercial surveys

A number of surveys of commercial IDSs has been made. In general, they are not of great interest in this paper, since they do not give much information on the technical details and do not really reach state of the art within the research. Some reports are more thorough, however, and may be interesting from a research perspective.

One of these surveys of commercial systems is [Kva99]. This paper gives a number of comparison criteria. These include granularity of data processing, source of audit data, detection method, response to detected intrusions, system organisation, security, degree of interoperability, manageability, adaptivity, and system and network infrastructure requirements. These criteria have been derived from the classification by Axelsson [Axe98]. 17 IDS products were reviewed and evaluated according to each of the comparison criteria.

Another rather interesting survey of commercial systems is [Jac99]. She identifies a number of important IDS properties. Among the characteristics are deployment, information source, detection method, execution, and response. Other attributes discussed are suitability, flexibility, protection, interoperability, comprehensiveness, event management, active response, acquisition, and support. A number of commercial products are described according to these characteristics and attributes. She also surveys 17 IDS products, whereof 12 are the same as in [Kva99].

These two surveys give a good overview of available IDSs and their properties. The information in these two surveys is comparable, and the trends seem to point in the same direction. Some trends that can be spotted in these surveys are that more systems are network-based than host-based, and that most systems only use network packets as the information source. Other trends are that all systems use knowledge-based detection, i.e. looking for patterns of known misuse, while only a few systems use some form of behaviour-based detection, and all systems except one have real-time processing. Also, more than half of the systems have some form of active response. Some systems have protection mechanisms against attacks directed towards themselves, but the security of the IDSs is generally rather low.

5 Research areas

Research in each of the areas named in section 3 is reviewed here. The area that has received the most attention so far is the *foundations* and *detection method*. Areas currently considered hot include *testing and evaluation* and *response*. Future research will probably be more in *IDS environment and architecture* and hopefully also in *data collection*.

5.1 Foundations

This is research on the foundations of the area, which has to do both with definitions and terminology as well as studies and taxonomies of attackers, vulnerabilities and intrusions. We need to know the threat, i.e. study attackers, their objectives and methods, to know what kind of attacks we can expect and whom we need to defend ourselves against. The areas of "data collection" and "detection method" are heavily dependent on this research. A suitable taxonomy of intrusions may be very helpful in deciding what data to collect to efficiently trace intrusions within a specific class. A description of the intrusion technique or vulnerability exploited for a collection of attacks makes it possible to design intrusion signatures to detect each of them. Another taxonomy may help in deciding what detection method would be most effective for a specific class of intrusions. A third type of taxonomy may help in deciding what classes of intrusions are most likely to appear or are most important to set up a defence against in a specific type of computer system. Risk and cost analysis are also interesting since they motivate the use of IDSs.

5.1.1 Study of attackers

Jonsson and Olovsson [JO97] studied the intrusion process by letting students (unexperienced in the field of cracking) attack a computer system. They measured the time to the first security breach for each attacker and had a mean value of four hours in their case. They see the attack session as a process in which the attacker first goes through learning phase, then starts making many standard (widely known) attacks, and finally comes into an innovative attack phase where the intrusions are fewer but typically novel.

There are also studies of the means and motives of attackers. Denning has written several papers about hackers, their motives and their own view of what they are doing, e.g. [Den90].

5.1.2 Vulnerability analysis and categorisation

Landwehr [LBMC94] presents a taxonomy of computer program security flaws, and a number of flaws in different computing environments have been classified according to his taxonomy. Three dimensions are used to classify the flaws. The first one is genesis (how did the flaw enter the system?), the second is time of introduction (when did the flaw enter the system?) and the last is location (where in the system is it manifested?).

Krsul [Krs98] made a thorough analysis of software vulnerabilities in his thesis. He collected a database of vulnerabilities and presents a framework for developing taxonomies for software vulnerabilities. He suggests a large number of features that can be used to classify vulnerabilities that he divides into classes, e.g. threat features, environmental assumption features, objects affected, and effect on objects. He also discusses classification theory and points out the weaknesses of other classification schemes and taxonomies. However, the main focus of this work is to make it possible to study and understand the nature of software vulnerabilities. This would especially help to improve the design and development of software. The aim is not to make the taxonomy directly useful for intrusion detection, even though it may be possible to create detection patterns from the information collected about the vulnerabilities.

5.1.3 Intrusion categorisation

Lindqvist and Jonsson [LJ97] present a survey of work in intrusion classification and a new scheme for intrusion classification. The aim of this classification scheme is to be useful in incident reporting and to present intrusions to a system owner in a helpful way. The focus is on external observations about the attacks, unlike Landwehr's approach which requires information about the software development process. The two chief dimensions used in this classification are *intrusion technique* and *intrusion result*. The *intrusion technique* category with subcategories is an extension of the taxonomy in [NP89].

Howard [How97] investigated several thousand incidents reported to the CERT Coordination Center and presented a great deal of statistics about these incidents. He also introduces a taxonomy for classification of the incidents, which views an attack as a process, as below.

Attackers => tools => access => results => Objectives

This taxonomy has the same focus as that of Lindqvist and Jonsson [LJ97] but includes some additional steps in the process. The *intrusion technique* in [LJ97] approximately corresponds to *tools* and *access*, and *intrusion result* corresponds directly to *results*.

The database of incidents used here is not publicly available. This information would be of great help to people developing or doing research on intrusion detection tools.

Kendall [Ken99] collected a large database of intrusions that have been used in the DARPA intrusion detection evaluation [DAR01]. They were collected from publicly available sources, such as Rootshell, the Bugtraq mailing list, CERT, ISS X-force, and vendor-initiated bulletins. Some new exploits were created especially for this evaluation. He also developed a taxonomy for these intrusions. The main dimension of this classification scheme is the transitions between *privilege levels*. Other dimensions are the *method of transition or exploitation* and *action*. The four main groups of attacks used in the DARPA evaluation are *Denial of Service*, *Remote to User*, *User to Superuser*, and *Surveillance/Probing*. It is not actually explained why these four categories are used to present the attacks or why they are particulary useful for intrusion injection.

5.1.4 Intrusion categorisation based on complexity of matching

Kumar [Kum95] presents a scheme to classify intrusion patterns on the basis of the complexity of matching. This scheme has four categories. The first is *existence patterns*, i.e. the fact that something exists is sufficient for detecting the intrusion attempt. Examples may be the existence of specific permissions on special files, presence of certain files, and format and content of files. This type of pattern looks for evidence left by an intruder. It can especially be used to detect configuration mistakes. The time required to match a pattern of this type is constant and not dependent on the history of events.

The second category is *sequence patterns*. This type of pattern can be used to specify intrusions that are characterised by a strict sequence of events. There are two special cases of this category. The first is *interval*, where the matching requires that two events happen within a special interval apart. The second is *duration*, where events must not exist or happen for more than or less than a certain interval of time. For example, race condition attacks can be represented by sequence patterns. The time to process an event for sequence patterns depends on the events in the event stream that occurred before the event.

The third category is *RE patterns*, i.e. extended regular expressions including the primitive AND to construct patterns. This type of pattern can represent intrusions that include a number of activities done jointly but in an arbitrary order. This category is a superset of sequence patterns.

The fourth category is *other patterns*. This category contains all intrusion signatures that can not be represented by one of the earlier categories. Patterns that fall into this category include those that require matching negations of the type "not followed by" and patterns using selection of the type "x-3 out of x conditions must be satisfied to match".

These categories form a hierarchy, where category three is a subset of category four, category two is a subset of category three, and category one is a subset of category two. In another project they studied intrusions and created intrusion signatures with which they populated the hierarchy. They claim that most intrusions they have studied fall into the first three categories and state that different approaches to detection can be used for the different categories.

This may be the most useful approach to a categorisation of intrusions for purposes of intrusion detection, and perhaps the only published material that truly focuses on creating a classification scheme with intrusion detection in mind.

5.1.5 Intrusion categorisation based on criteria relevant to the intrusion detection process

Alessandri et al. [ACD+01] presents an "activity" taxonomy, which is validated by classifying 358 attacks from IBM's vulnerability database, VulDa. An activity can be described as an event or sequence of events that are relevant to IDSs. The goal of this taxonomy is to allow a description of activities based on criteria relevant to the intrusion detection process conducted by IDSs. It should also be possible to describe activities that are similar to other activities that threaten or violate the security policy, i.e. activities that may cause false alarms. The criteria

used in this taxonomy are directly relevant to how IDSs analyse input data to detect attacks. The taxonomy is ment to be used when evaluating IDSs in order to support the choice of suitable "activities" from different classes.

The two main classes of this taxonomy are static and dynamic activity characteristics. Subclasses to static characteristics are *affected object*, e.g. memory, operating system core, or network stack, and *interface object*, e.g. system call, socket, or network layer. Subclasses to dynamic characteristics are *communication*, *method invocation*, and *additional attributes*.

5.1.6 Discussion and conclusions

The following issues have been studied or would be interesting to study to form a complete picture of the foundations of the intrusion detection area.

- Study of attackers
- Vulnerability categorisation
- Intrusion collection
- Intrusion categorisation
- Normal behaviour categorisation

Study of attackers has been done but not very extensively. However, this area may not have a very large impact on the development of intrusion detection.

A great many *vulnerability categorisations* exist. These may be of interest for intrusion detection researchers, even though they are not of direct help in the development of better IDSs.

Intrusion collection has been done in several projects. The intrusion databases are often not publicly available, however, which of course makes them less useful. Public databases with extensive information about intrusions would carry the intrusion detection research forward.

Intrusion categorisation is more interesting in the development of the intrusion detection research at this time. It would be of great help to find categorisations of intrusions that have similar detection properties. An *intrusion categorisation* can have many different goals and therefore can be of many types.

Some possible types are:

- Vulnerability exploit
- Intrusion technique
- Result of attack
- Traceability
- Detectability
- Cost

- Type of system attacked

Although vulnerability exploit, intrusion technique, and result of attack are very common dimensions of intrusion taxonomies, they may not be the most interesting characteristics for research in intrusion detection.

Some initial efforts have been made on categorisation based on traceability and detectability. These are very interesting for the development of the areas of evaluation and testing and of improving efficiency. They may also be useful in studying how to combine detection methods. I believe that these areas deserve more attention.

A cost has been associated with each attack in the area of fraud, e.g. in [SFL+00]. The cost parameter is used in the training of the fraud detection system to make it prioritise costly frauds. I have not yet seen any similar work in the area of intrusion detection.

I have not seen any studies of the types of attacks (or attackers) that hit different types of systems. This may be an interesting issue, since it may help us define a more accurate and adapted picture of the threat.

A structured approach to *normal behaviour categorisation* seems not to exist in intrusion detection research as yet.

5.2 Data collection

The desire is for suitable, high quality input to the IDS. If the quality of the input data is very good, many other issues will resolve themselves. For example, if the amount of input data is kept low, the efficiency of the IDS is not as critical. Also, if we use input data containing exactly the information we need for effective detection, we will miss fewer intrusions. Data collection is one of the most important issues in intrusion detection. Even if only manual intrusion detection is used, the importance of high quality input data is still valid to the same degree.

Data collection has to do with deciding what information to collect, how to collect it, and when to collect it. The choice of data collection procedures may be affected by the type of detection algorithm used and the type of system in which the detection will be done. It is important to study the threat and decide what type of incidents we wish to detect so that we can properly specify the data collection procedure. At the same time the resources used for data collection must be limited.

Key issues are to identify what data sources are most suitable for intrusion detection and to define a common log format for the data collected. Another issue is how to reduce the amount of log data from these data sources while still ensuring enough information for correct detection.

5.2.1 Data sources

Every project on developing or testing intrusion detection systems uses some kind of data. In most cases, the data used seem to be what is easily available with the standard logging mechanisms. This may be e.g. network traffic, logged by the *tcpdump* tool, system calls logged by the *SUN BSM* mechanism, or *syslog*

logs generated by the OS kernel and various applications, such as ftp and smtp. If data from different sources are used by the same detection mechanism, they are in most cases transformed to a common log format.

Many papers do not really focus on data collection but reveal a great deal of information about the data sources they use.

One is about the network Bro IDS [Pax98]. This author uses a low-level packet-capture library (libpcap) to collect network traffic. The traffic is used to detect network attacks, i.e. attacks against the IP and TCP/UDP protocols. This information is easy to extract from the network traffic. He also describes how information from the finger, ftp, portmapper, and telnet application protocols is extracted. This is a more demanding task, and he describes a number of problems in parsing the application-specific information. Some data reduction is also done. Low level filters decide whether the full packet, only the header, or nothing at all should be recorded. Packets that are not processed are simply thrown away.

Forrest et al. [FHSL96] used the Sun BSM (Basic Security Module) to collect system calls. They then used the system calls to establish normal behaviour for processes in a Unix system. This information was then used for anomaly detection.

Sun BSM system calls are also used in the EMERALD environment developed at SRI. Some examples of BSM audit trail analysis are described in [LP99]. They give examples of expert system rules for detecting failed authentications and for detecting generic buffer overflow attacks.

In the DARPA intrusion detection evaluation [LFG+00], both network traffic collected by *tcpdump* and system calls collected by Sun BSM were used.

The log watching tool Swatch [HA93] used *syslog* files as a source of information. They modified some system utilities, such as fingerd, ftpd, and login, to achieve more extensive reporting. The syslog files are sent to a central logserver, since it is not secure to store them locally on the machines being monitored.

In [LB98], Unix shell commands are used to create user profiles that are then used for anomaly detection experiments. They claim that they use this type of data since tools exist that make it convenient to collect it. However, a problem is that almost no shell traces of attacks are available, which makes it more difficult to test the detection mechanism.

5.2.2 Data format

A paper that discusses the importance of a standard audit trail format is [Bis95]. A standard audit trail format is necessary to be able to correlate logs from heterogeneous systems and to allow interoperability of audit systems on a large scale. Bishop suggests an extensible and portable log format, but this format does not specify at all what information should be logged.

Price [Pri97] covers the subject of operating system audit data collection and the needs of intrusion detection systems. It includes a survey of systems with extended auditing capabilities, but she claims that all of the audit trails suggested are too closely linked to the operating system and do not easily extend to support other operating systems. It also includes a thorough survey of the information

that five different misuse detection systems use and a review of the information are offered by the common operating system audit trails. The requirements and the information offered are compared to find out whether the information is sufficient. The conclusion is that the offered information is not sufficient to cover the needs of the misuse detection systems surveyed. This may be the work that thus far best covers the issue of suitable audit format and content of data for intrusion detection.

The intrusion detection working group (IDWG) [IDW01] from IETF has the task of providing a common way of communicating between IDSs. They recently produced some Internet drafts on a standardised intrusion detection exchange format. The high-level requirements for sharing information of interest to intrusion detection and response systems and the connected management systems are described in [WE01]. Draft [CD01] defines data formats and exchange procedures and gives an implementation of the data model in XML. Draft [FMW01] specifies an application-level protocol for exchanging the IDMEF messages described in [CD01]. This protocol supports mutual authentication, confidentiality and integrity. It remains to be seen whether this format will have impact in the IDS society.

The CERIAS group at Purdue University started up projects on *Audit Trails Format* [Cer98b] and *Audit Trail Reduction* [Cer98a]. I have not however seen very much of the results of this work as yet.

5.2.3 Effective/efficient logging

A form of light-weight logging is suggested in [ALGJ98]. Only the exec() system call, together with its arguments is logged. Data were collected during realistic intrusion experiments and categorised into ten classes. It was shown that this form of logging is more effective in detecting the intrusions generated in these experiments than "traditional" logging. It also consumes less storage space. By traditional logging we mean the built-in UNIX logging mechanisms that log connect time (who is logged in on the system and when), process accounting by pacct, and syslog logging from various daemons, user programs, and the kernel. Seven of the ten classes of intrusions were traceable using the exec() logging, while only three classes were traceable using the other logging mechanisms. They discuss the traditional logging mechanisms, point out their disadvantages, and explain why they did not catch the intrusions in these experiments. This is the only paper, to my knowledge, that actually focuses on creating a more effective and a more efficient logging suitable for intrusion detection.

Another way to achieve a more efficient logging is to use some form of dynamic or adaptive logging.

In, for example, Haystack [Sma88], there is a possibility to flag single user accounts to collect more information about them. This can be viewed as a form of adaptive logging.

Paxon [Pax98] suggests dynamic logging as a way of making graceful degradation. His solution is to drop all traffic from certain pre-defined protocols when the load becomes too high.

5.2.4 New collection mechanisms

One recent report that may be a result of the CERIAS projects is [SZ00a]. They draw conclusions about data collection from their experience in work with the AAFID distributed intrusion detection system. They claim that it is preferable to make direct data collection and give a number of reasons for this. Direct data collection means that functions in the operating system and applications are equipped with collection mechanisms that are adapted to intrusion detection needs, and this information is not recorded in a file before it is captured by the IDS. This gives a number of advantages. For example, it is more difficult for the intruder to modify the information used by the IDS and that only the information needed is collected, which makes the data collection more efficient. They also claim that direct data collection is best implemented with internal sensors, i.e. the operating system and application code must be modified. They have also started work on implementing internal sensors in a Unix system.

Other papers from CERIAS are [DS99] and [DS00]. The first of these studies the information needed to detect low-level network attacks with host-based detection. They state that existing host-based audit trails do not contain the required information. The second paper describes an implementation of a collection mechanism for creating a network audit on a host by modifying the operating system.

5.2.5 Log file security

Schneier and Kelsey [SK99] discuss how to secure logs, since it is important that the input to the IDS is correct. More detail on this work is given in section 5.6, as I consider this to be a security issue.

However, the authors also wrote a paper on how to make their log encryption scheme more efficient in order to make it useful in low-bandwidth environments [KS99].

5.2.6 Commercial status

Jackson's survey [Jac99] gives information about information sources used in commercial IDSs. Most systems use only network packets. A few use operating system logs, some use specific application logs, e.g. from web servers or firewalls, and some use file system information.

5.2.7 Discussion and conclusions

The following issues have been studied or would be interesting to study to form a complete picture of the foundations of the data collection area.

- Data sources - An evaluation of existing data sources and suggestion for new types of data sources.
- Data format
- Dynamic logging

- Efficient logging - fast and storage efficient
- Effective logging
- Secure logging

Different *data sources* have been used, studied and evaluated, and some new sources have been suggested, but there is room for more work in this area. Most intrusion detection systems just use the available audit data. *Data format* has also been studied, and some standardisation efforts have been made but it is not yet clear what the content of the logs should be or what they should look like.

Dynamic logging, i.e. adapting the amount of log data and the contents of log data to the current threat, has been suggested, but, as far as I know, no work focuses on this issue.

Much more work can be done on both *efficient logging* and *effective logging*. Effective logging is about collecting all data necessary to conduct correct and reliable detection. Efficient logging has to do with minimising the amount of data collected and the storage needs. However, people designing new collection mechanims are aware of these issues and do address the problems to some extent.

While *secure logging* is an important issue and has been addressed, most research papers on intrusion detection ignore this issue.

A conclusion to be made from this survey of work on data collection is that this has been neglected in most research presented on intrusion detection but that interest in the issue seems to be on the rise. Much more research must be done in the area to achieve high quality data collection suitable for intrusion detection.

A significant problem in the area of data collection is that we often do not know exactly what intrusions we wish to detect. This means the data collection may need to be more generic and will not be able to be as limited and efficient as we would like. Furthermore, if we choose to limit data collection, we always risk missing intrusions when the attacker finds out how to act in a way that does not leave "fingerprints". We are thus dependent on work in the *foundations* area to establish better data collection.

5.3 Detection method

Detection method in this context refers to automatic detection. Purely manual detection and manual output processing goes under the reporting and response area, even though there may only be a small difference between visualisation and "simple" reporting. Many different detection methods have been implemented and tested in the area of detection. Some were mentioned in section 2.

An interesting issue is how to make detection more effective. Many methods produce a great many false alarms, and this is one of the greatest problems in intrusion detection today. One approach may be to combine different detection methods. Certain methods may be more suitable for certain classes of attacks. Another way to combine detection components may be to use them sequentially, i.e. to use one method for initial data filtering or reduction and another for the final filtering before reporting.

Another issue is how to make detection more efficient. We want to minimise the computing power and storage space needed. Efficiency may also be about timely detection, i.e. making the time from the start of the malicious activity until it is detected as short as possible.

The detection methods can be categorised in many different ways, and several taxonomies and classifications on detection method have been proposed, e.g. [DDW99] and [Axe98]. I will use only a rough division into anomaly and misuse detection, since these are the most commonly used categories. Visualisation may be considered an anomaly detection approach but, since it is less explored and may be a promising approach, I treat it as a separate area. Also, since the work on different anomaly and misuse detection methods has been described thoroughly in other surveys, I will only make a quick summary of the different approaches. More effort is put into surveying work on visualisation, how to combine detection components, and on effective and efficient detection.

5.3.1 Data reduction

Data reduction can be used as preprocessing to detection or to support manual analysis of the output data. This may be done by filtering log data to capture the security-relevant events. It may also be useful to reduce data by grouping it into sessions. ComputerWatch [DR90] is one example of an audit reduction tool.

5.3.2 Misuse detection

Misuse detection is primarily done using some form of pattern matching.

The most simple form of pattern matching is string matching. In Swatch [HA93], this is done using the well-known *regex* tool. It is easy for the administrator to add his own patterns for events he wishes to be notified about.

Expert systems have been used in several IDSs, for example EMERALD [LP99] and ASAX [Mou97]. The underlying idea is to create a knowledge base by talking to experts in the area. The rules in the expert system are often in the form of if-then questions, which are used to derive relevant facts from the log data. It is possible to create more complex "scenarios" using an expert system than using simple string matching, and it is probably easier to encode knowlegde about misuse. Both these papers include descriptions of special languages that are designed to make it easier to write pattern matching rules.

Another approach is *state transition analysis*. This is done in the USTAT tool [Ilg93]. Intrusions are encoded as a series of transitions between system states, where the system is first in a secure state and then when the intrusion has been made, putting the system in an insecure state.

Kumar and Spafford model intrusions using *coloured Petri nets* in their prototype called IDIOT [KS94].

Both state transition analysis and coloured Petri nets require a great deal of work in constructing the rules. However, they both have positive features.

Data mining is another popular method in intrusion detection. [LSM99] use data mining to extract useful patterns from network traffic and create intrusion models. This method is interesting since it automates the creation of detection

rules somewhat. The authors claim that their rules become more general and produce fewer false alarms than "hand-coded" rules. Data mining can also be used for anomaly detection.

5.3.3 Anomaly detection

Anomaly detection can be done in many ways. Statistics, neural networks, and genetic programming are some of the methods that have been implemented.

The best described statistical anomaly detection component is the one used in NIDES [JV94]. This is also one of the most advanced anomaly detection systems. These authors calculate statistics for a large number of parameters in the system. The parameters are updated daily to adapt to changes in behaviour. Each audit record that is compared to the profiles is given a score value that represents its similarity to the profile parameters. A threshold value can be set to report only those events that have a higher score value.

Neural networks have been used for anomaly detection, for example by [DBS92]. Neural networks are good at classifying input data automatically. A problem is that suitable data are needed to train the neural network, and it may be difficult to understand why it classifies an event as anomalous.

Another "artificial intelligent method" used for anomaly detection is genetic algorithms. As far as I know, it has only been used in the Gassata analysis tool [Mé98].

Lane and Brodey use *instance based learning* to create profiles of normal behaviour and discuss different ways to make the method more efficient. They also describe an experiment in which they use their method to create profiles of user behaviour from Unix shell command data.

Forrest et al. [FHSL96] used system calls to establish normal behaviour for processes in a Unix system. This is done by creating a database of short sequences of system calls for each process. It is then possible to carry out anomaly detection, i.e. to detect deviations from normal behaviour by comparing sequences of system calls generated by the running process to the database of normal behaviour.

Most anomaly detection methods are more or less self-learning and can be trained to recognise normal behaviour. Ko [KRL97] used another approach that he calls specification-based detection. The intended security-relevant behaviour of programs is manually specified using a formal method. Deviations from this specification are then detected. This may be very useful for monitoring security-critical programs.

5.3.4 Bayesian techniques

Another detection method used in EMERALD is Bayes' net technology [VS00]. The method is model-based and uses probabilistic reasoning and interference. They claim that the *eBayes* detection component using this method combines the best features of misuse and anomaly detection.

5.3.5 Visualisation

Girardin [Gir99] presents a new approach to detection. He used an unsupervised neural network to reduce the dimensions of the input data. A self-organising map was used to project network events on a space appropriate for visualisation. The output was visualised in a grid where the units of the map were portrayed as squares. Squares of different sizes and colours represent the number of events mapped in the unit and the weight of the selected attribute. This is a recent paper and the first I am aware of that focuses on visualisation for use in intrusion detection.

Work by Erbacher and Frincke [EF00] aims at visualising intrusion detection data in a large-scale network of computers. They use an IDS to collect data from the network and analyse this data to present the communication between the active nodes (hosts) in the network. Circles represent nodes in the network. Different colours, directions of arrows, line thicknesses etc. represent different protocols, properties and events.

5.3.6 Combining detection methods and multi-tiered detection

Combining methods may give better coverage, and may make the detection more effective. The question is what methods to combine and how.

A paper that combines three different methods in one system is [FP96]. This is a fraud detection system, but the techniques should be applicable to intrusion detection as well. Data mining is used to find indicators of fraud, constructive induction is used to create profiling detectors, and an evidence-combining component correlates output from the profiling detectors and generates alarms.

Many intrusion detection systems use both anomaly and misuse detection components, but most reports of them do not give a discussion of how to integrate them.

A way to combine methods is to use several layers of filtering. For example, it may be very useful to analyse alarms to collect them into scenarios, to remove redundant alarms, or to get more information about the incident. NIDES and EMERALD include a component called the *resolver* that filters the alarms so that redundant alarms are removed.

Valdes [VS01] suggests an approach to sensor correlation, which means that alarms from different components in the detection system are analysed and correlated at different levels.

Another method that can be used to correlate and draw conclusions from data from many distributed sources is *multisensor data fusion*. This method is described for example in Bass [Bas00].

5.3.7 Effective detection

Axelsson [Axe99] studies the effectiveness of intrusion detection and states that it may be very difficult to make detection (especially anomaly detection) sufficiently effective with respect to the false alarm rate.

On the other hand, many papers using anomaly detection criticise misuse detection for not being effective in terms of coverage . Misuse detection can never achieve complete coverage because it does not detect "new" intrusions.

A suggestion for making misuse detection more effective is to write generic detection signatures. This can be done by writing signatures that detect the effect of the intrusion rather than detecting the use of a specific vulnerability. As shown in [LP99], all types of buffer overflow attacks can potentially be caught with a simple set of rules if they are made generic.

5.3.8 Efficient detection

Some papers indicate that their methods are efficient in their use of computing or storage resources, and some even compare the efficiency of different realisations of the method, e.g. [LB98]. This paper mentions short time to detection which may be considered an efficiency aspect. However, there does not seem to be more thorough studies on this subject.

5.3.9 Discussion and conclusions

The following is a summary and discussion of various possible research issues in the area of *detection method*.

- Anomaly detection
- Misuse detection
- Visualisation
- Data reduction and manual detection
- Integration of methods
- New methods/improved methods
- Filtering analysis output
- Effective detection
- Efficient detection
- Detection with limited input

Many *anomaly detection* and *misuse detection* methods have been implemented and tested. Of course there is always room for improvement and suggestions to achieve more *effective* and *efficient* detection. The false alarm rate in particular is still an unsolved problem.

Visualisation can be done in many ways, and only a few have been tried out for intrusion detection. I believe that there is much more work to be done in this area.

Integration of methods should be an area of interest in today's research. Many papers suggest that combining anomaly and misuse detection can provide better coverage and even say that the false alarm rate will be better. Very few papers mention any details on how to do it, however, or what specific detection methods to use. Detection in several layers is also an interesting method for achieving more effective detection. *Filtering analysis output* and correlation of alarms may also be an area to explore.

The discussion of how to carry out detection when the amount of information that can be gathered in the system is limited. This may be the case when different components in the computer system belong to different organisations or when we wish to coordinate detection between completely separate computer systems. I have not seen any research on this in the intrusion detection area.

5.4 Reporting and response

Every intrusion detection system needs to communicate with the outside world. This may be done passively by notifying the SSO or actively by trying to hinder the intruder or striking back. It is often not possible to use active response since the IDSs generate too many false alarms and the response would hit innocent users or hosts. The only possible action is then to report what has happened to the SSO and perhaps provide him with the means to investigate the event further.

If anomaly detection is used, the SSO must be informed of the cause of an alarm. This type of system often requires that the SSO makes an investigation as it is not obvious what type of attack is being made when it is only known, for example, that the number of false logins in the system have increased to an abnormal level. It is not known whether it is an attack or simply a malfunction or change of behaviour in the system.

Misuse detection makes it easier to describe the cause of the alarm to the SSO, although this is still not obvious. One problem is that many different names exist for the same attack and, even when there a name for the attack, the SSO may not know what it is about or what to do about it. Another problem is that alarms must be listed in some way, for example by urgency or according to the probabilty that it is a true alarm.

Active response can be done in many ways, the most suitable way probably depending on the situation. If availability is not very important, an easy solution is just to shut services down or disconnect users. However, this is not always an option for a business that is dependent on e-commerce. There are also legal and ethical issues to consider.

A home user may find it easier to shut down a service if it is tampered with. He has no possibility to study the output and make decisions about the response. He is probably mostly interested in detecting a security incident, than responding quickly to it. In a business system, availability can be important. However, it is also important to preserve business secrets, and, here, a tradeoff must be made. A military system probably has people who are dedicated to the security of the computer system. They may study detection output in real-time and have a list of advanced response options at hand.

A classification of response functions in other IDSs is given in [CHSP00]. The response function in a detection system can be categorised as a notification system, manual response system, or automatic response system. According to these authors, most systems today are notification systems.

5.4.1 Reporting

A PhD thesis on intrusion damage control and assessment is [Fis96] where it is stated that the existing IDSs at that time all had the ability to generate reports of suspicious activity but that the damage control and assessment capabilities were minimal. He surveyed the reporting and response capabilites of about ten different IDSs. The thesis was written in 1996, and it is obvious that much has happened since then. Most of the systems only generate daily or weekly reports of suspicious events or users. The only systems that had graphical user interfaces that could alert the administrator in real-time (if he is present) were Haystack, IDES, and ISOA. The user interfaces of these systems are not described in detail here.

The thesis also described a new damage control and assessment system. The focus of this system is not on the user interface for reporting and notifying the SSO but instead relies on automatic response. It probably has some notification capabilites, although these are not described in detail.

I have not found more recent work that focuses on the issue of how to report to and alert the system administrator when suspicious activities occur. However, most IDS papers describe the user interface in some detail.

Some examples of what is described in the research follow.

Swatch [Sma88] filters out "security relevant" events from the syslog log files and can echo them to the Swatch controlling terminal. Particularly important events can be mailed to the system administrator or cause Swatch to call his pager. This interface is flexible but not very advanced.

NIDES [ALJ+94] has a better thought out interface that is graphical and has menus and point-and-click selections. It presents information on monitored systems and summaries of system throughput and alert generation. It is also possible to configure many parameters and monitoring options through the interface.

Most commercial systems have graphical interfaces. An alarm is often presented with a description of the presumed attack. However, they are generally not very advanced.

5.4.2 Active response

Fisch's PhD thesis [Fis96] surveys response mechanims, and only one of the IDSs surveyed supports active response (ISOA). However, the response mechanisms are not automatic and require the presence of an administration. The new response system described in this thesis has an extensive set of active response mechanisms. These include the following actions: suspend user jobs, terminate user session, warn user, lock user account, create backup files, require additional authentication, and employ shadow files.

A taxonomy of intrusion response is proposed in [CP00]. The main elements of this taxonomy are response timing, type of attack, type of attacker, strength of suspicion, implications of the attack, and environmental constraints. The AAIRS intrusion response system is described briefly here.

The components of AAIRS, which are based on intelligent agents, are described in more detail in [CHSP00]. A classification of response functions in other IDSs can also be found here. The response function in a detection system can be categorised as a notification system, manual response system, or automatic response system. They state that most systems are notification systems. The systems with automatic response are not very advanced, with two exceptions. The Cooperating Security Managers (CSM) and Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD) have some real-time response adaption. Both use expert systems to determine an appropriate response based on the current suspicion level. EMERALD also uses a severity metric to determine the appropriate response.

An adaptive intrusion detection system is described in [RCHP00]. This system is used together with AAIRS to provide both adaptive detection and response. The response system is relatively advanced. It keeps track of previous alarms and classifies attacks on the basis of whether they are a continuation of an existing incident or whether it is a new attack. Alarms from different IDS agents in the system have different confidence metrics according to previous detection results. The confidence in a suspected incident and the nature of the incident affects the course of action taken. The response is also affected by a "policy specification agent" to adapt the response to different constraints.

Active response capabilities of commercial IDSs are surveyed in [Jac99]. The following types of active response are used in one or more of the surveyed systems.

- Session termination
- Router or switch reconfiguration
- Firewall reconfiguration
- Vulnerability correction

Session termination, router or switch reconfiguration, and firewall reconfiguration are used in several systems. Vulnerability correction is used in only one system. Other active responses suggested here are session hijacking and deception techniques, but these are not used in any of the systems surveyed.

5.4.3 Discussion and conclusions

The following is a summary and discussion of various possible research issues within the *Reporting and response* area.

Passive response and human interaction issues:

- Presentation of detection data

- What to expect from the SSO

Active response/countermeasure issues:

- Automatic response
- Support for manual response

Presentation of detection data is a very important part of an IDS. Many papers on intrusion detection prototypes barely mention the user interface. Some things that may be included in an alarm report are a summary and information about the intrusion. Many different alarms can probably be connected to the same intrusion scenario. For example, it should be possible to trace an intruder from the first probing of the system until he is inside and has root access. Graphical presentations of information can also help the SSO. Other issues are how to grade alarms with respect to severity or urgency and how to present these different levels. It should furthermore be possible to set probabilities on alarms, which may make it easier for the SSO to manually filter out false alarms. I have not seen any thorough work on the best way to present the output data of an IDS, and I think that we need this.

Studies of *what to expect from the SSO* seem to be lacking. I believe that more human interaction research in connection to intrusion detection may be needed. One question is how to adapt the presentation of alarms to different administrators with different amounts of time and knowledge.

It seems that the research community has begun to realise the importance of *active response* in the last few years. Some work has been done in this area. However, until we can trust the output of the detection system, manual analysis of the alarms is needed. This means that automatic response can be used only for special incidents and must be defensive in nature. *Support for manual response* has been implemented and may be useful when automatic response can not be used.

5.5 IDS environment and architecture

The idea of a distributed IDS has existed for a long time and today more or less all IDSs are distributed. This definition of distributed means only that the IDS has components that can be placed at different points in the computer network and that these components can communicate in some way.

One type of distributed IDS only has data collection components distributed in the network and makes all analyses at a central point. In another type, independent data collection and analysis components are distributed in the network and only a minimal management function is located at a central place. Everything falling between these two types is of course also possible. Even when analysis is centralised, the data collection components can use pre-filtering and formatting of data before sending them to analysis. It is common for the central management unit to present the alarms from all the distributed components. It may also correlate alarms and carry out system-wide detection of distributed attacks. A general architecture of a distributed IDS is shown in figure 3.

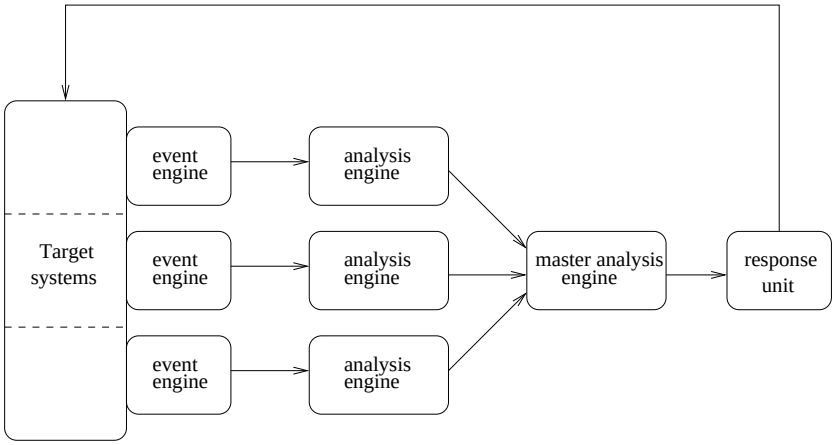

Figure 3: Distributed IDS

5.5.1 Distributed IDSs

DIDS [SSTG92] was one of the first distributed IDSs and has host and LAN monitors that collect data and perform some pre-processing and analysis. The resulting information is sent to the DIDS director for further analysis and presentation to the SSO. Several detection components have been used in the implementation of the architecture, such as Haystack, a signature analysis component, and a subset of NSM, which is a network IDS.

Mounji includes a chapter about the implementation of the distributed version of ASAX in his thesis [Mou97]. He compares the performance of centralised and distributed analysis and concludes that distributed analysis is more efficient. However he did not discuss the best way to distribute analysis between the distributed and the central components. He also discussed some other issues in the use of a distributed IDS. Some problems mentioned are clock synchronisation, heterogeneous environment, scalability, and distributed intrusion patterns.

EMERALD [NP99] is one of the more modern distributed IDS architectures. It is intended to be very general and to support interoperability and scalability. Each EMERALD monitor is independent and may use input data from both raw data sources and event reports from other detection components. The monitor consists of a pluggable configuration library with connected profiler engines, signature engines and a resolver that correlates alarms from the other components. It is possible to plug in third-party modules for both detection and correlation. The monitors can be used in a layered hierarchy, which gives it good scalability.

5.5.2 Agent-based IDSs

Another rather advanced IDS architecture is described by Spafford and Zamboni [SZ00b]. They use what they call autonomous agents to do detection in a distributed computer network. In this architecture they use filters to deal with heterogeneous systems. One filter is used per data source, which provides the agents with system independent input. More than one agent can "subscribe" to a filter. The idea is that the agents should be reasonably small and dedicated to one task.

Each host also has a transceiver that collects data from the agents, does the appropriate processing and sends it on to other agents or to a monitor that takes care of system-wide analysis. They claim that the advantage of autonomous agents is that it is possible to dynamically enable and disable them, which makes it possible to provide continuous monitoring. They can also be used in a hierarchical structure, which makes the system scalable.

A system with lightweight, mobile agents is described by [HWHM00]. The idea here is that the agents move around in a networked system between the target hosts and collect information from local data cleaning agents.

A review of work on agents in intrusion detection and a thorough discussion of its features and problems is given by Jansen et al. at NIST [JMKM99].

Another interesting use of mobile agents is described by Asaka et al. [ASAS99]. They treat the use of agents to trace an intruder along the intrusion route.

5.5.3 Detection in large-scale networks

An approach especially designed for detection in large networks is suggested by Staniford-Chen et al. [SCCC+96]. Their detection prototype is called GrIDS (Graph based Intrusion Detection System), since it aggregates information on activity in the computer network into graphs. The approach is scalable and is especially good at detecting distributed attacks.

5.5.4 Discussion and conclusions

Here is given a summary and discussion of various possible research issues within the *IDS environment and architecture* area.

- Distributed data collection and detection
- Correlation of alarms
- System-wide detection and correlation of alarms between networks
- Heterogenous systems
- How to deal with high-bandwidth and encrypted networks
- Scalability
- Mobile agents
- Security problems

Distributed data collection and detection have been treated in several papers. What is missing is a thorough investigation of how much analysis should be done in the distributed components and how much done centrally. Another interesting issue is how to correlate alarms to detect distributed attacks.

Dealing with *heterogeneous systems* has been discussed and some solutions suggested. *Scalability* is discussed in many papers and seems to be a solvable problem. There may however be greater problems in making anomaly detection scale well.

Concern has been expressed about *how to deal with high-bandwidth and encrypted networks*, but I have not seen very much work in this area.

Agents, and especially *mobile agents*, have recently attracted the attention of intrusion detection researchers. This is an interesting technology and may be the solution used in the future. However, the area probably needs to become more mature before we can tell whether it solves more problems than it creates.

Distributed IDSs add some *security problems*. These are discussed in section 5.6.

Distributed and modular IDSs are probably here to stay. A specially designed IDS for an important application or service in a system may be needed in some situations, but it is generally better when they are "pluggable" and part of a distributed IDS architecture. Most of the commercial systems today are distributed and will remain so in the future.

5.6 IDS security

If the attacker is aware that an intrusion detection system exists, he will probably start by studying the IDS to be able to shut it down, cripple it, or circumvent it. The IDS will be the first point of attack, since the attacker can work undisturbed when it is out of operation. He can be compared with a burglar who disables the alarm system in a building before he starts his actual business.

Thus an IDS that can not resist attacks against itself is not of much use as a security mechanism. It is stated in [HLJ01] that the IDS is not secure if it depends upon other components that are insecure. For example, the input to the detection system must be trusted, the operating system and hardware it is running on must be trusted, and, if it is a distributed system, the communication links must also be trusted.

A detection system is useless if its input is manipulated. Both the logging mechanism and the log files must be protected to ensure the authenticity of the input data. While deletion of log files makes it difficult to find out what happened, it is at least an evident sign of an attack. Modification or deletion of selected entries in the log file are more tricky, since an attack can be covered up. In this case, it is not even possible to know that something has happened.

It is also a security problem if an unauthorised person can read the log files. These may contain information of help in mapping the behaviour of users to prepare for a social engineering attack or to study the organisation of the system. Log files also contain personal data that may be considered sensitive and should be protected. Privacy problems associated with log data are discussed in [HLJ01] and [SFHR97].

While a weak IDS may give a false sense of security, that is not the only problem. The IDS is also a valuable source of information to an attacker. This problem is discussed in [HLJ01], where it is stated that the detection policy used by the

IDS is extremely sensitive. If the exact detection rules are known by the attacker, it is much easier to avoid detection. It is also a privacy problem if stored user profiles are leaked, since they reveal a great deal of sensitive information about people's habits.

5.6.1 Protection of input data

One paper is prominent in the area of protecting log data. This is the work of Schneier and Kelsey [SK99]. They suggest different solutions to the problem of storing log files on an untrusted machine. They write that no security measures can protect the log entries written after an attacker has gained control over the computer. The solutions mentioned in this paper are:

- Write logs to "write once media": printer, writable CDROM, WORM disk.
- Send log entries to a trusted log server. There must be a reliable, high-bandwidth channel available.
- Encrypt the log files.

Schneier and Kelsey propose an interesting way of encrypting log files. The log entries are encrypted in a "hash chain", which makes it impossible to modify or delete entries undetected. Furthermore, it is not possible to read the entries without the correct encryption key. This seems to be a well-functioning technique that would be possible to use in practice.

5.6.2 IDS resilience

Few papers discuss IDS resilience, i.e. the ability of the IDS to resist attacks against itself. One of the few papers that does more than simply mention the problem is [Pax98], which describes the Bro network IDS. A network IDS is a detection system that uses only the traffic on the network to which it is attached as input for analysis. A design goal in Bro is that the monitor should be able to resist attacks. It is assumed that the attackers have full knowledge of the techniques used by the monitor. However, the detection policy is assumed to be secret and well protected.

Three types of attacks are considered in this paper:

- *Overload attacks*. These are attacks in which the monitor is overburdened with traffic or suspicious events. The monitor can not keep up with the data stream, starts dropping network packets, and may miss attacks. Bro is designed to be lightweight and can keep up with a high traffic volume. It is also suggested that the monitor be able to shed load when it is overwhelmed, for example ceasing to capture packets from a specific protocol according to a predefined policy. This would give some kind of graceful degradation.

- *Crash attacks*. A crash attack aims at taking the monitor completely out of action. It may fail or run out of resources. Even single packets can have this effect on a computer. Suggestions for defence are to test the monitor carefully, to keep the system light weight so that it does not consume a great deal of memory etc., to use a "watch dog" timer to see that a single packet does not take to much time to process, and to have backup logging units.
- *Subterfuge attacks*. A subterfuge attack misleads the monitor as to the meaning of the traffic it analyses. There are several ways to do this, such as by sending fragmented IP packets. In Bro, defence is implemented against a number of subterfuge attacks, but there is no general defence.

To our knowledge, this is the only paper that describes a IDS prototype with an explicit design goal of being resilient.

A practical study of the resilience of four popular network IDSs is described in [PN98]. The main problem identified and discussed in this paper is that a network IDS has no way of knowing whether it interprets the captured network packets in the same way as the target system (the same problem as in the subterfuge attacks mentioned above). This makes it possible to evade detection in various ways. It is possible to make the IDS accept packets that the target system rejects. The opposite is also true - that it is possible to make the IDS reject packets that the target system accepts. The reasons for these problems are that different operating systems implement the packet processing differently and that there exist network ambiguities. It is possible, for example, to set a short TTL (time to live) on a packet that will make a router drop it before it reaches the target system (or the IDS). Some network links will also use fragmentation to handle large packets, which makes it possible to craft packets such that it reaches the target system in one piece and reaches the IDS in fragment form.

The other problem discussed is that it is possible to make a "denial of service" (DoS) attack against the IDS, which will leave the target system unprotected. This can be done by exhausting the IDS's resources, for example, CPU processing capabilities, low-level packet capture capabilities, memory used for TCP connection state, or disk space used for log files.

The tests done in this paper show that all the systems examined had flaws of the types described above. Some of the specific problems found would be easy to fix while others had no obvious solution.

The commercial surveys show that the protection in commercial IDSs is either non-existent or rather poor.

5.6.3 Security problems in distributed IDSs

The security implications in a distributed intrusion detection architecture is discussed in [HKJ99]. They state that a distributed IDS is more vulnerable than an IDS where all functions are placed physically together, and and that it is important both to protect the confidentiality of the data collected and of the detection policy used in the analysis engines. The confidentiality of the data collected can be protected by using a fully distributed IDS. In [HKJ01], it is described how to protect distributed detection policies using one-way functions.

5.6.4 Discussion and conclusions

Various possible research issues in the *IDS security* area are summarised and discussed below.

- Protection of log files
- Self-testing IDS
- Protection of distributed IDSs

Protection of log files and different aspects of *protection of distributed IDSs* have been discussed and some solutions have been suggested. There is room for more work within this area, however.

An interesting area is *self-testing IDSs*, but I have not yet seen very much work on how to do it.

Most commercial systems are probably not concerned enough with the security of the log files. However, if the IDS is distributed, they often encrypt the traffic (including log data and alarms) sent between the components.

There are of course numerous other problems that should be addressed by a resilient IDS. An example is dynamic learning systems that can be taught to accept abnormal behaviour. Furthermore, if the attacker knows the threshold values used in the system, he can decrease the rate of suspicious events to keep it below the threshold. Another known way to immobilise the IDS is to make it flood the system administrator with seemingly meaningless alarms in order to make him switch it off. Finally, the attacker may also choose to do his business quickly and run for it before the IDS or security personnel reacts. Part of the IDS functionality is thus also the reaction time.

5.7 Testing and evaluation

This is the area of research on IDS testing and comparison. Some of the issues here are how to create test data with the right properties, to find out what properties should be measured, and to define what properties are important for different types of target systems.

Intrusion detection researchers have recently become aware that testing and evaluating IDSs is an areas that needs more work. It is difficult to create better IDSs when we have no notion of what a good characteristic of an IDS is.

5.7.1 General testbeds and test data generation

Perhaps the greatest effort in testing and comparing intrusion detection systems was made by DARPA [DAR01]. A paper that describes the experiments is [LFG+00]. A great deal of work has been done on generating large amounts of test and training data for this project. The generation of data is described, but details and analysis of its properties are lacking. The test data contain network traffic and system call log files (SUN BSM log files) from a simulated large computer network. Both

attacks and background data have been generated synthetically, but the background data are said to be similar to the sampling data from a number of Air Force bases. The attacks are generated by running scripts taken from a database of attacks and are described in [Ken99]. More detail about the generation of sessions and the technique used to scale up data is given in [DCW+99].

McHugh [McH00] criticises the lack of validation of test data in the DARPA evaluation. The process of generating data is vaguely described, and it is difficult to determine the quality of the data. There are no guarantees that the attacks used are in fact representative and realistic. He also questions whether the background data are actually representative of the user behaviour in the computer systems used and whether the behaviour in these systems represents user behaviour in other computer systems. Another question he raises is whether attacks are realistically distributed in the background data.

Debar et al. [DDWL98] developed a generic intrusion detection testbed. They showed how to simulate user behaviour with a finite state automata to obtain normal traffic, but stated that this is only a practical approach if the set of user commands is limited. Instead, they carried out experiments with recorded live data that were replayed inside the workbench. They also describe how they create attack scripts to be used in the testing process.

Chung et al. [CPOM95] claim that concurrent and distributed intrusions are likely to occur in a computer system and therefore should be included in the test data for intrusion detection systems. They developed a tool that parallelises attack scripts to simulate concurrent intrusions.

Puketza et al. [PZC+96] described a software platform for testing intrusion detection systems. They used the *expect* UNIX package to generate synthetic user sessions. However, they did not analyse the quality of these data to any great extent.

An effort to produce high quality test data is made by [LKJ01]. The data are synthetic and were generated to test a fraud detection system. However, the authors believe that the method may also be useful for intrusion detection. This paper also thoroughly discusses important properties of test data.

5.7.2 Testing special properties

Maxion and Tan [MT00] discuss the effects of more or less irregular background data. They generate "random" background data with different degrees of regularity and show that they drastically affect the false alarm rate. The paper also measures regularity in real-world log data.

5.7.3 Discussion and conclusions

The following is a summary and discussion of various possible research issues in the area of *testing and evaluation*.

- IDS quality measures
- Measuring detection rate

- Measuring false alarm rate
- Efficiency testing
- Test data generation
- Intrusion injection

Some work has been done on testing IDSs, but a great deal more must be done to make it easier and more accurate. In particular, *IDS quality measures* are missing. Since it is difficult to retrieve authentic data with the right properties for intrusion detection testing, we need to develop better methods for *test data generation* and realistic *intrusion injection*. A conclusion is that all aspects of testing need more work.

5.8 Operational aspects

Operational aspects are about the technical issues important to customers using IDSs. They include maintenance, portability, and upgradability. Such issues as interoperability between IDSs or IDS components are also important, as is the knowledge required to configure and operate the IDS. These are very important issues in terms of making IDSs useful on the commercial market.

The work on EMERALD [NP99] brings up these questions, where the design goal is to make EMERALD generic and easy to upgrade. The work on using agents for intrusion detection (see section 5.5.2) mentions that this method makes the IDS more portable and easier to maintain during runtime. Studies on requirements of the person operating the IDS are still lacking but would give valuable information.

In [ACF+00], which describes the state of the practice of intrusion detection technologies, recommendations are given to ID sponsors, users, vendors, and researchers. They describe what they call organisational issues, which are recommendations about, for example, deployment, operation and use, and maintenance. A deployment issue is to install and configure the IDS to reflect the security policy. An example of an issue of operation and use is the allocation of roles and responsibilities for analysing the results that an IDS produces and acting on those results. Maintenance issues are updating the signature database and replacing the old version of the IDS with a new version.

An issue of interest is finding efficient ways of creating and adding new patterns to look for in misuse detection systems. This has of course been discussed in papers on misuse detection, but I have not seen any general solutions to the problem.

Automatic system tuning would also be interesting. Both anomaly and misuse detection systems need a great deal of parameter tuning when installed in a new system. Ways to do this more automatically would be useful.

There is not a great deal of work that focuses on these issues. These issues may often not be considered research issues, but perhaps there will be more interest in operational aspects in the future.

5.9 Social aspects

Some ethical and legal issues may be obstacles to the spreading of IDSs. The primary issues discussed so far in public research are privacy issues and questions about how to make the output of IDSs usable as forensic evidence.

5.9.1 Privacy

A thorough discussion is given in [SFHR97] on laws concerning privacy and on motivation for why privacy enhancing techniques should be used in IDSs. They claim that use of IDSs may not be permitted in some countries if privacy is not protected.

Techniques for anonymisation or pseudonymisation of log files are found e.g. in [SFHR97] and [LJ99]. These papers do not address the problem of a subverted log server. We can assume that a trusted log server is used here. However, even if log data are collected and stored in a secure fashion, they may still end up in the wrong hands. The log data may be used in other contexts and handled by people with no need to know everything about the users and computers in the system. People may be interested in the information in the log files for security analysis etc., but have no need to know the real names of the users and the IP addresses of the computers in the log files.

5.9.2 Forensics

The use of detection output as evidence of computer crime has been discussed in some work.

Stephenson [Ste00] describes the framework of a project that aims at exploring the applicability of intrusion detection systems to evidence collection and management. He states that a requirement in a forensic system is "the maintenance of a chain of custody whereby all evidence can be accounted for and its integrity attested from the time of its collection to the time of its use in a legal proceeding". A concern is whether the IDS can perform its primary mission effectively at the same time that it handles the capturing and management of forensic evidence.

Schneier and Kelsey [SK99] claim that the primary benefit of their tamper-proof encrypted audit logs is that they aid forensic analysis. The encryption scheme makes it possible to identify which part of the log is valid and which part that may be tampered with, which is an important feature if it is to be used as forensic evidence.

5.9.3 Discussion and conclusions

Some work has been done on both privacy and legal aspects. It seems that interest in privacy enhancing techniques is not at a high right now but it may be necessary to address these aspects in the future if privacy legislation becomes more stringent. Privacy enhancing techniques may also prove useful in some special applications, such as third-party analysis of log files and detection output, which is relatively common today.

Not much has been done on legal aspects. I believe that forensic management capabilities in IDSs will be more common when the intrusion detection area matures.

More social aspects of intrusion detection may surface, but these are the only areas in which I have found work at this time.

6 Research trends

As we can see in this survey, a huge amount of research has been done in the area of intrusion detection. Efforts are not always well organised, however, and, while some areas receive a great deal of attention, other are almost completely neglected.

Hot areas right now are agent-based IDSs and the development of testbeds. Other areas that have recently received some attention are active response and the security of IDSs. Many people also mention other important areas with unresolved issues, but many of these have been known for a long time without any effort being made to work on them and find solutions.

In my opinion, the most important remaining research issues are the following.

Classification of intrusions. A reason why we do not know what data to collect and use as input for IDSs is that we do not have a systematic approach toward identifying what data are needed to detect specific intrusions or classes of intrusions. We must have a classification of IDSs that is based on the traceability or characteristics important in the detection process. Some initial work in this area has been done, but more is needed.

Effective data collection. So far, most projects have used easily available data sources. Many comment that these sources are not really suitable for intrusion detection. We need to find out in detail what data are necessary for detection and develop new mechanisms to collect them in an efficient way.

Reporting of alarms. We need an intuitive and informative user interface that allows the SSO to explore the computer system. Such an interface does not exist today. I believe that this may be one of the most neglected issues, since it is very important for the understanding of security issues in a computer system.

Methodology for test data generation. Testing and comparing IDSs is important to the progress of the area. The greatest obstacle here is the difficulty of generating suitable test data. The process of generating test data must be easier and more automated, to make it feasible to test and evaluate different aspects of IDSs.

Our view of intrusion detection is that it is not just a fad that will disappear. However, it may experience some change in its shape and functions. A fear is that future intrusion detection systems will just work as anti-virus software. There is

a potential to do much more with an IDS than merely have it detect pre-defined patterns of well known attacks. An intrusion detection system can create a picture of the computer system behaviour and give the system administrator an entirely new opportunity to learn his system and become familiar with its behaviour. This may pave the way for a better understanding of security and intrusion management.

7 Conclusions and future work

There are still many aspects of intrusion detection that require more research to make IDSs effective and usable. The area is still immature, despite the extensive research efforts made and the enormous amount of money that has been spent.

A hope is that this collection of research efforts and presentation of areas and open research issues may inspire people doing research in intrusion detection. Efforts can be concentrated to the right areas and research ideas can be brought to the commercial market to a greater extent.

We plan to extend this work to include research in fraud detection. It is suggested in [KLJ00] that the areas of intrusion detection and fraud detection have quite a lot to learn from each other. Some fraud papers have been included in this survey, and methods used there are in many cases applicable to intrusion detection. A similar research survey of the fraud detection area may offer many ideas on common interests and facilitate an exchange of ideas.

References

[ACD+01] Dominique Alessandri, Christian Cachin, Marc Dacier, Oliver Deak, Klaus Julisch, Brian Randell, James Riordan, Andreas Tscharner, Andreas Wespi, and Candid Wüest. Towards a taxonomy of intrusion detection systems and attacks. Technical Report Research Report RZ-3366, IBM Research, Zurich Research Laboratory, Zurich, CH, 2001.

[ACF+00] J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, and E. Stoner. State of the practice of intrusion detection technologies. Technical Report Technical Report CMU/SEI-99TR -028, Software Engineering Institute, Carnegie Mellon, 2000.

[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, Arlington, Virginia, October 5-8 1998. National Institute of Standards and Technology/National Computer Security Center.

[ALJ+94] Anderson, Lunt, Javits, Tamaru, and Valdes. *NIDES: Software Users Manual — Beta-Update Release*, December 1994. Available at: http://www.sdl.sri.com/papers/7sri/.

[ASAS99] M. Asaka, S.Okazawa, A.Taguchi, and S.Goto. A method of tracing intruders by use of mobile agents. In *Proceedings of INET'99*, June 1999.

[Axe98] Stefan Axelsson. Research in intrusion detection systems: A survey. Technical Report 98-17, Dept. of Computer Engineering, Chalmers University of Technology, Dept. of Computer Engineering, Chalmers University of Technology, Sweden, Dec 15 1998. revised Aug 19, 1999.

[Axe99] Stefan Axelsson. The base-rate fallacy and its implications for intrusion detection. In *Proceeding of the 6th ACM Conference on Computer and Communications Security*, Kent Ridge Digital Labs, Singapore, November 1-4 1999.

[Bas00] Tim Bass. Intrusion detection systems and multisensor data fusion. *Communications of the ACM*, 43(4):99–105, April 2000.

[Bis95] Matt Bishop. A standard audit trail format. In *Proceedings of the Eighteenth National Information Systems Security Conference*, pages 136–145, October 1995.

[CD01] D. Curry and H. Debar. Intrusion detection message exchange format data model and extensible markup language (XML) document type definition. Internet draft - http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-05.txt, November 16 2001.

[Cer98a] Cerias. Coast - audit trail reduction group. Project description available at http://www.cerias.purdue.edu/coast/projects/audit-trails-reduce.html, 1998.

[Cer98b] Cerias. Coast - audit trails format. Project description available at http://www.cerias.purdue.edu/coast/projects/audit-trails-format.html, 1998.

[CHSP00] C. A. Carver, J. M. D. Hill, J. R. Surdu, and U. W. Pooch. A methodology for using intelligent agents to provide automated intrusion response. In *IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop*, pages 110–116, West Point, NY, June 6-7 2000.

[CP00] C. A. Carver and U. W. Pooch. An intrusion response taxonomy and its role in automatic intrusion response. In *IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop*, pages 129–135, West Point, NY, June 6-7 2000.

[CPOM95] Mandy Chung, Nicholas J. Puketza, Ronald A. Olsson, and Biswanath Mukherjee. Simulating concurrent intrusions for testing intrusion detection systems: Parallelizing intrusions. In *Proceedings of the 1995 National Information Systems Security Conference*, pages 173–183, Baltimore, Maryland, October 10-13 1995.

[DAR01] DARPA. DARPA intrusion detection evaluation. http://www.ll.mit.edu/IST/ideval/, July 2001. The main web page for the DARPA evaluation experiments.

[DBS92] H. Debar, M. Becker, and D. Siboni. A neural network component for an intrusion detection system. In *Proceedings of the IEEE Symposium on Research in Computer Security and Privacy*, pages 240 – 250, Oakland, CA, 1992.

[DCW+99] Robert Durst, Terrence Champion, Brian Witten, Eric Miller, and Luigi Spagnuolo. Testing and evaluating computer intrusion detection systems. *Communications of the ACM*, 42(7):53–61, July 1999.

[DDW99] Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805–822, April 1999.

[DDWL98] H. Debar, M. Dacier, A. Wespi, and S. Lampart. An experimentation workbench for intrusion detection systems. Technical Report RZ2998, IBM Research Division, Zurich Research Laboratory, Zurich, Switzerland, March 1998.

[Den90] Dorothy E. Denning. Concerning hackers who break into computer systems. Presented at the 13th National Computer Security Conference, Washington, D.C., Oct. 1-4 1990. Available at: http://www.cpsr.org/cpsr/privacy/crime/denning.hackers.html.

[DR90] C. Dowell and P. Ramstedt. The COMPUTERWATCH data reduction tool. In *Proceedings of the 13th National Computer Security Conference*, pages 99–108, Washington, D.C., October 1990.

[DS99] Thomas Daniels and Eugene Spaffor. Identification of host audit data to detect attacks on low-level ip vulnerabilities. *Journal of Computer Security*, 7(1):3–35, 1999.

[DS00] Thomas Daniels and Eugene Spafford. A network audit system for host-based intrusion detection (NASHID) in linux. In *Proceedings of the Annual Computer Security Applications Conference*, New Orleand, LA, Dec 2000.

[EF00] R.F. Erbacher and D. Frincke. Visualization in detection of intrusions and misuse in large scale networks. In *Proceedings of the International Conference on Information Visualisation (IV2000)*, London, England, July 2000. Available at http://www.computer.org/Proceedings/iv/0743/0743toc.htm.

[ESNP95] M Esmaili, R Safavi, Naini, and J Pieprzyk. Intrusion detection: A survey. In *Proceedings of ICCC'95. (12th International Conference on Computer Communication)*, volume xxxxii+862, pages 409–414. IOS Press, Amsterdam, Netherlands, 1995.

[FHSL96] Stephanie Forrest, Steven A. Hofmeyr, Anil Somayaji, and Thomas A. Longstaff. A sense of self for Unix processes. In *Proceedinges of the 1996 IEEE Symposium on Research in Security and Privacy*, pages 120–128. IEEE Computer Society Press, 1996.

[Fis96] E. A. Fisch. *Intrusion Damage Control and Assessment: A Taxonomy and Implementation of Automated Responses to Intrusive Behavior*. PhD thesis, A&M University, Texas, A&M University, College Station, Texas, 1996.

[FMW01] B. Feinstein, G. Matthews, and J. White. The intrusion detection exchange protocol (idxp). Internet draft - http://www.ietf.org/internet-drafts/draft-ietf-idwg-beep-idxp-03.txt, September 11 2001.

[FP96] Tom Fawcett and Foster Provost. Combining data mining and machine learning for effective user profiling. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 8–13, 1996.

[Gir99] Luc Girardin. An eye on network intruder - administrator shootouts. In *Proceedings of the workshop on intrusion detection and network monitoring*, Santa Clara, California, April 1999.

[HA93] Stephen E. Hansen and E. Todd Atkins. Automated system monitoring and notification with swatch. In *Proceedings of the seventh Systems Administration Conference (LISA '93)*, Monterey, CA, November 1993.

[HKJ99] H. Hedbom, H. Kvarnström, and E. Jonsson. Security implications of distributed intrusion detection architectures. In *Proceedings of the Fourth Nordic Workshop on Secure IT Systems (NordSec99)*, Stockholm, Sweden, November 1999.

[HKJ01] H. Hedbom, H. Kvarnström, and E. Jonsson. Protecting stateless security policies using one-way functions. Technical Report Technical report 01-3, Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, 2001.

[HLJ01] Hans Hedbom, Stefan Lindskog, and Erland Jonsson. Risks and dangers of security extensions. In *Proceedings of Security and Control of IT in Society-II (IFIP SCITS-II)*, pages 231–248, Bratislava, Slovakia, June 15-16 2001.

[How97] John D Howard. *An analysis of security incidents on the Internet 1989-1995*. PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania, April 1997.

[HWHM00] G. Helmer, J. Wong, V. Honavar, and L. Miller. Lightweight agents for intrusion detection, 2000.

[IDW01] IDWG. Intrusion detection working group - internet engineering task force (IETF). Home page: http://www.ietf.org/html.charters/idwg-charter.html, October 16 2001.

[Ilg93] Koral Ilgun. USTAT: A real-time intrusion detection system for UNIX. In *Proceedings of the 1993 IEEE Symposium on Research in Security and Privacy*, pages 16–28, Oakland, CA, 1993.

[Jac99] Kathleen A. Jackson. Intrusion detection system (IDS) product survey. Technical Report Report LA-UR-99-3883, Los Alamos National Lab, 1999.

[JMKM99] W. Jansen, P. Mell, T. Karygiannis, and D. Marks. Applying mobile agents to intrusion detection and response. Technical Report Technical report, National Institute of Standards and Technology, October 1999.

[JO97] Erland Jonsson and Tomas Olovsson. A quantitative model of the security intrusion process based on attacker behavior. *IEEE Transactions on Software Engineering*, 23(4), April 1997.

[JV94] Harold S. Javitz and Alfonso Valdes. The NIDES statistical component: Description and justification. Technical report, SRI Computer Science Laboratory, Menlo Park, CA, March 1994. Available for download at: http://www.sdl.sri.com/nides/index5.html.

[Ken99] Kristopher Kendall. A database of computer attacks for the evaluation of intrusion detection systems. Master's thesis, MIT, 1999.

[KLJ00] Håkan Kvarnström, Emilie Lundin, and Erland Jonsson. Combining fraud and intrusion detection - meeting new requirements. In *Proceedings of the fifth Nordic Workshop on Secure IT systems (NordSec2000)*, Reykjavik, Iceland, October 12-13 2000.

[KRL97] C. Ko, M. Ruschitzka, and K. Levitt. Execution monitoring of security-critical programs in distributed systems: A specification-based approach. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, Oakland, California, 1997.

[Krs98] Ivan V Krsul. *Software vulnerability analysis*. PhD thesis, Purdue University, West Lafayette, Indiana, May 1998.

[KS94] Sandeep Kumar and Eugene H. Spafford. A pattern matching model for misuse intrusion detection. In *Proceedings of the 17th National Computer Security Conference*, pages 11–21, Baltimore, MD, October 1994.

[KS99] J. Kelsey and B. Schneier. Minimizing bandwidth for remote access to cryptographically protected audit logs. In *Proceedings of Second International Workshop on the Recent Advances in Intrusion Detection (RAID '99)*, September 1999.

[Kum95] Sandeep Kumar. *Classification and detection of computer intrusions*. PhD thesis, Purdue University, West Lafayette, Indiana, August 1995.

[Kva99] Håkan Kvarnström. A survey of commercial tools for intrusion detection. Technical Report Technical Report TR99-8, Chalmers University of Technology, 1999.

[LB98] Terran Lane and Carla E. Brodley. Temporal sequence learning and data reduction for anomaly detection. In *Proceedings of the 5th Conference on Computer & Communications Security*, pages 150–158, San Francisco, CA, USA, November 2-5 1998. ACM, ACM Special Interest Group.

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

[LFG+00] Richard P. Lippman, David J. Fried, Isaac Graaf, Joshua W. Haines, Kristopher R. Kendall, David McClung, Dan Weber, Seth E. Webster, Dan Wyschogrod, Robert K. Cunningham, and Marc A. Zissman. Evaluating intrusion detection systems: The 1998 DARPA offline intrusion detection evaluation. In *DISCEX 2000*. IEEE Computer Society Press, January 2000.

[LJ97] Ulf Lindqvist and Erland Jonsson. How to systematically classify computer security intrusions. In *Proceedings of the 1997 Symposium on Security and Privacy*, pages 154–163, Oakland, California, May 4-7 1997.

[LJ99] Emilie Lundin and Erland Jonsson. Privacy vs intrusion detection analysis. In *Proceedings of the Second International Workshop on the Recent Advances in Intrusion Detection - RAID'99*, West Lafayette, Indiana, USA, September 7-9 1999.

[LKJ01] Emilie Lundin, Håkan Kvarnström, and Erland Jonsson. Generation of high quality test data for evaluation of fraud detection systems. In *Proceedings of the sixth Nordic Workshop on Secure IT systems (NordSec2001)*, Copenhagen, Denmark, November 1-2 2001.

[LP99] Ulf Lindqvist and P.A. Porras. Detecting computer and network misuse through the Production-Based Expert system Toolset (P-BEST). In *Proceeding of the 1999 Symposium of Security and Privacy*, Oakland, California, May 1999. IEEE Computer Society.

[LSM99] Wenke Lee, Salvatore J. Stolfo, and Kui W. Mok. A data mining framework for building intrusion detection models. In *Proceedings of the 1999 IEEE Symposium on Security and Privacy, 1999*, pages 120–132, 1999.

[Lun88] Teresa F. Lunt. Automated audit trail analysis and intrusion detection: A survey. In *Proceedings of the 11th National Computer Security Conference*, Baltimore, MD, 1988.

[McH00] John McHugh. The 1998 Lincoln Laboratory IDS evaluation: A critique. In *Recent Advances in Intrusion Detection, Third International Workshop, RAID 2000, Toulouse, France, 02-04 October 2000, pages*, pages 145–161, Toulouse, France, October 2-4 2000. Lecture Notes in Computer Science #1907, Springer-Verlag, Berlin.

[Mé98] Ludovic Mé. GASSATA, a genetic algorithm as an alternative tool for security audit trails analysis. In *Proceedings of the First International Workshop on the Recent Advances in Intrusion Detection*, Louvain-la-Neuve, Belgium, September 1998. Available at: http://www.zurich.ibm.com/dac/Prog_RAID98/Full_Papers/gassata_paper.ps.

[MHL94] B. Mukherjee, L. T. Heberlein, and K. N. Levitt. Network intrusion detection. *IEEE Network*, pages 26–41, May/June 1994.

[Mou97] A. Mounji. *Languages and Tools for Rule-Based Distributed Intrusion Detection*. PhD thesis, Facult'e Universitaire Notre de la Paix de Namur, Belgium, September 1997.

[MT00] Roy A. Maxion and Kymie M.C. Tan. Benchmarking anomaly-based detection systems. In *International Conference on Dependable Systems and Networks*, pages 623–630, New York, New York, June 2000. IEEE Computer Society Press.

[MWS+90] N. McAuliffe, D. Wolcott, L. Schaefer, N. Kelem, and T. Haley B. Hubbard. Is your computer being misused? a survey of current intrusion detection system technology. In *Proceedings of the 6th Annual Computer Security Applications Conference*, Tucson, AZ, December 1990.

[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, October 10-13 1989. National Institute of Standards and Technology/National Computer Security Center.

[NP99] Peter G. Neumann and Phillip A. Porras. Experience with EMERALD to DATE. In *Proceedings of 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, pages 73–80, Santa Clara, California, 11-12 April 1999.

[Pax98] Vern Paxon. Bro: A system for detecting network intruders in real-time. In *Proceedings of the Seventh USENIX Security Symposium*, pages 31–51, San Antonio, Texas, January 1998. USENIX.

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

[Pri97] Katherine E. Price. Host-based misuse detection and conventional operating systems' audit data collection. Master's thesis, Purdue University, 1997.

[PZC+96] Nicholas J. Puketza, Kui Zhang, Mandy Chung, Biswanath Mukherjee, and Ronald A. Olsson. A methodology for testing intrusion detection systems. *Software Engineering*, 22(10):719–729, 1996.

[RCHP00] D.J. Ragsdale, Jr. Carver, C.A., J.W. Humphries, and U.W. Pooch. Adaptation techniques for intrusion detection and intrusion response systems. In *2000 IEEE International Conference on Systems, Man, and Cybernetics*, volume 4, pages 2344–2349, 2000.

[SCCC+96] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle. GrIDS – A graph-based intrusion detection system for large networks. In *Proceedings of the 19th National Information Systems Security Conference*, 1996.

[SFHR97] Michael Sobirey, Simone Fischer-Hübner, and Kai Rannenberg. Pseudonymous audit for privacy enhanced intrusion detection. In *Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC '97)*, pages 151–163, Copenhagen, Denmark, May 1997. IFIP, Chapman & Hall.

[SFL+00] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan. Cost-based modeling for fraud and instrusion detection: Results from the jam project. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, pages p. II 130–144. IEEE Computer Press, 2000.

[SK99] Bruce Schneier and John Kelsey. Secure audit logs to support computer forensics. *ACM Transactions on Information and System Security (TISSEC)*, 2(2), May 1999.

[Sma88] Steven E. Smaha. Haystack: An intrusion detection system. In *Fourth Aerospace Computer Security Applications Conference (IEEE Cat.No.CH2619-5)*, volume xii+440, pages 37–44, Washington, DC, USA, 1988. IEEE Comput. Soc. Press.

[SSTG92] Steven R Snapp, Stephen E Smaha, Daniel M Teal, and Tim Grance. The DIDS (distributed intrusion detection system) prototype. In *Proceedings of the Summer USENIX Conference*, pages 227–233, San Antonio, Texas, June 8-12 1992. USENIX Association.

[Ste00] Peter Stephenson. The application of intrusion detection systems in a forensic environment. Presented at Third International Workshop on the Recent Advances in Intrusion Detection (RAID 2000), Toulouse, France. Available at: http://www.raid-symposium.org/raid2000/program.html, October 2-4 2000.

[SZ00a] Eugene Spafford and Diego Zamboni. Data collection mechanisms for intrusion detection systems. Technical Report CERIAS technical report 2000-08, Center for Education and Research in Information Assurance and Security (CERIAS), Purdue University, West Lafayette, Indiana, June 2000.

[SZ00b] Eugene H. Spafford and Diego Zamboni. Intrusion detection using autonomous agents. *Computer Networks*, 34(4):547–570, October 2000.

[VS00] Alfonso Valdes and Keith Skinner. Adaptive, model-based monitoring for cyber attack detection. In H. Debar, L. Me, and F. Wu., editors, *Recent Advances in Intrusion Detection (RAID 2000)*, number 1907 in Lecture Notes in Computer Science, Toulouse, France, October 2000. Springer-Verlag.

[VS01] Alfonso Valdes and Keith Skinner. Probabilistic alert correlation. In *Recent Advances in Intrusion Detection (RAID 2001)*, number 2212 in Lecture Notes in Computer Science, Davis, California, October 2001. Springer-Verlag.

[WE01] M. Wood and M. Erlinger. Intrusion detection message exchange requirements. Internet draft - http://www.ietf.org/proceedings/01mar/I-D/idwg-requirements-05.txt, February 20 2001. work in progress.