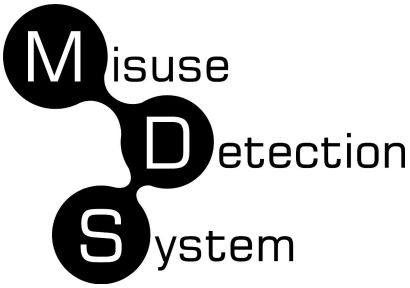

MDS – Misuse Detection System

Contract No. 26459

D2.2.1 AI Solutions for MDS: Artificial Intelligence Techniques for Misuse Detection and Localisation in Telecommunication Environments

Deliverable type:	**PU – public**
Contractual date:	31 May 2006
Actual date:	15 July 2006
Activity:	WP2
Lead partner:	Bournemouth University
Authors:	BU: Dr. Jonathan Vincent, Dr. Robert Mintram, Dr. Keith Phalp, Chukwudi Anyakoha
	UNI HB: Hanna Bauerdick, Björn Gottfried
	BRU: Dr. Sethuraman Muthuraman

Abstract

This report considers the application of Artificial Intelligence (AI) techniques to the problem of misuse detection and misuse localisation within telecommunications environments. A broad survey of techniques is provided, that covers *inter alia* rule based systems, model-based systems, case based reasoning, pattern matching, clustering and feature extraction, artificial neural networks, genetic algorithms, artificial immune systems, agent based systems, data mining and a variety of hybrid approaches. The report then considers the central issue of event correlation, that is at the heart of many misuse detection and localisation systems. The notion of being able to infer misuse by the correlation of individual temporally distributed events within a multiple data stream environment is explored, and a range of techniques, covering model based approaches, 'programmed' AI and machine learning paradigms. It is found that, in general, correlation is best achieved via rule based approaches, but that these suffer from a number of drawbacks, such as the difficulty of developing and maintaining an appropriate knowledge base, and the lack of ability to generalise from known misuses to new unseen misuses. Two distinct approaches are evident. One attempts to encode knowledge of known misuses, typically within rules, and use this to screen events. This approach cannot generally detect misuses for which it has not been programmed, *i.e.* it is prone to issuing false negatives. The other attempts to 'learn' the features of event patterns that constitute normal behaviour, and, by observing patterns that do not match expected behaviour, detect when a misuse has occurred. This approach is prone to issuing false positives, *i.e.* inferring misuse from innocent patterns of behaviour that the system was not trained to recognise. Contemporary approaches are seen to favour hybridisation, often combining detection or localisation mechanisms for both abnormal and normal behaviour, the former to capture known cases of misuse, the latter to capture unknown cases. In some systems, these mechanisms even work together to update each other to increase detection rates and lower false positive rates. It is concluded that hybridisation offers the most promising future direction, but that a rule or state based component is likely to remain, being the most natural approach to the correlation of complex events. The challenge, then, is to mitigate the weaknesses of canonical programmed systems such that learning, generalisation and adaptation are more readily facilitated.

Contents

Chapter 1

Introduction

1.1 Context

This report reviews the application of techniques drawn from the domain of Artificial Intelligence (AI) for the detection and localisation of misuse events within telecommunications environments. The underlying premise is that systems currently deployed in real networks are fundamentally rule based systems. It is given that these suffer from drawbacks such as poor generalisation and adaptability. The purpose of this work is to confirm these premises and identify techniques from the domain of AI that could be applied to misuse detection and, in particular, the correlation of high level alarms from network resources, with a view to addressing the perceived weaknesses of rule based systems. The report considers a broad range of techniques and application scenarios from low-level intrusion detection (*e.g.*, based on analysis of TCP/IP packets) to correlation of high level alarms from network resources. Event correlation is treated in detail, with reference to both the academic literature and existing systems. Further, the authors distinguish between misuse detection approaches and misuse localisation techniques, which are also treated in this report.

It is worth noting that the MDS project focusses on four broad classifications of misuse – Fraud Detection(FM), Fault Management (FM), Performance Management (PM), and Business Processes (BP) – within the domain of telecommunications networks. However, the primary purpose of this review is to identify application techniques for event correlation, and hence a broader range of literature is considered. It is found that the majority of literature considers misuse (often specifically intrusion detection) within computer networks. There is relatively little literature concerning the misuse areas considered by MDS, however, the techniques examined here are considered equally applicable to the MDS project.

1.2 Misuse Detection and Localisation

The term 'misuse' is herein defined in a broad sense as the use or behaviour of a networked environment in any way that is not consistent with the system's expected functionality, as perceived by the provider of the network service. This definition covers a range of possible misuses related to *inter alia* fraud, business processes, performance, and system faults. This work focusses on the detection of such misuse events. The misuse is often that of unauthorised access of the system or using the system in an unauthorised way. In this case, the detection of such events is usually referred to as 'intrusion detection' and the protection mechanism is called an Intrusion Detection System (IDS).

This report considers the specific detection mechanism – typically an event correlation engine – that examines events occurring within the networked environment and attempts to classify these events, or patterns of events, as representing either normal or abnormal behaviours. It also goes one step further discusses some localisation techniques which are used to localise a misuse in the network and to name its root cause. A broad range of misuse scenarios from the literature are examined. In addition, this report considers the correlation of alarms from network resources that may be indicative of a misuse of some kind.

It is assumed that the detection mechanism reside within a larger system – the 'Misuse Detection System' (MDS) – that interfaces with the networked environment and has overall responsibility for the gathering of events, presentation to the detection mechanism, and processing of the results. On occasion, alarms are raised by various network resources. These may indicate misuse of the system in some fashion. In many cases, many alarms will be created relating to the same act of misuse. It is possible that these many not be synchronised within a short time period and may even be interleaved with other alarms indicating entirely different misuse activity. The particular protection mechanism considered within this report is that which would form a smaller part of a larger mechanism and assumes that:

- Data in the form of alarms is available as input.
- There could be many such streams of alarm data arriving in parallel.
- Alarms could be indicated within different streams in different ways.
- Alarms may be related to the same misuse act but could arrive in different streams at different times.

Following the identification of a misuse, the task is to locate the precise source in a process generally referred to as misuse localisation. This is particular prevalent in

the context of fault or performance problems. Most localisation approaches assume that further domain specific knowledge is available, typically relying on network topology or the specification of failure classes. However, there are some approaches that try to deduce this knowledge from the network itself. For localisation this knowledge is needed because the root cause of a misuse has to be named and has to be inferred from the alarms available. Sometimes the faulty network element does not even generate an alarm due to its unavailability. Then it is important to have something like failure dependencies between the network elements to deduce the original root cause.

At this point it is worthwhile clarifying the difference between an IDS or MDS and a firewall. A firewall will forbid access to a system via a network according to a set of predefined rules and conditions. Any network content that satisfies a subset of these rules will be refused entrance to the system. A firewall must function at a portal to a system and acts as a 'doorman' to the system by vetting potential entrants. Anything suspicious is refused entrance. On the other hand an IDS or MDS detects an intrusion or misuse after (or possibly during) its occurrence. Notice that an IDS or MDS can also detect intrusions or misuses that originate from within the system and as such it does not necessarily need to function at a system portal.

Figure 1.1 (adapted from Steinder and Sethi [199]) shows a coarse taxonomy of detection mechanisms and the techniques that have been employed.

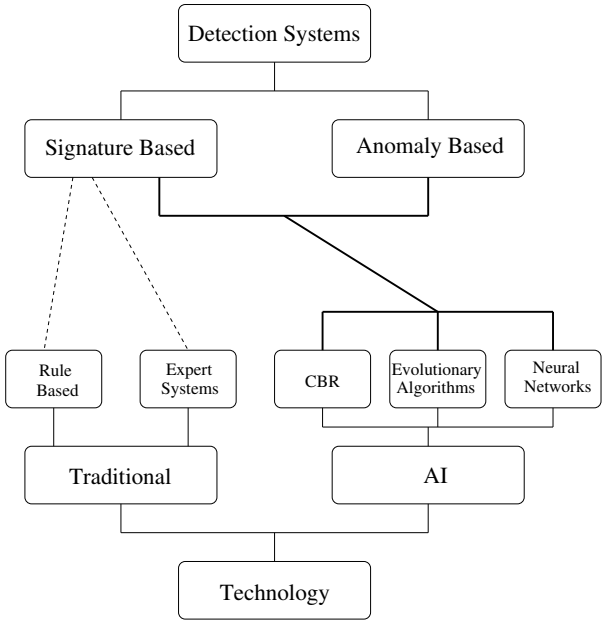

Figure 1.1: Overview of detection mechanisms

Detection mechanisms can therefore be classified as:

- **Signature based** where the intrusion can be recognized by comparison with known patterns. Somewhat confusingly, this is also often referred to as **misuse detection**. A database of known cases of misuse (signatures) is maintained and incoming events are compared with these signatures to determine if they are representative of a misuse. The detection system would maintain a large database of signatures, which could be updated as new misuses are identified. Such systems are inherently limited by the size of the database and as such are prone to issuing false negatives – *i.e.* indicating normal behaviour for unknown misuses.

- **Anomaly detection** where a baseline of normal network activity is compiled and deviations from this baseline then imply that an anomalous event has occurred, and this is taken to be a misuse. Such a system can detect new misuses provided that they are sufficiently different from normal behaviour. Given the range of normal activity and the quantity of data involved, this type of detection mechanism is prone to issuing false positives – *i.e.* indicating abnormal behaviour for innocent events that deviate from known patterns of activity.

Cuppens [43] describes these two approaches as *behavioural* referring to anomaly detection and *signature analysis* referring to pattern recognition. Cuppens also observes the problems stated above, but further notes that attacks could be orchestrated to occur very slowly, which can cause problems with anomaly detection systems, and signature based systems can suffer simply by becoming out of date. Further, signature based systems are typically pre-configured by experts based on domain knowledge, whereas anomaly detection systems are expect to 'learn' what constitutes normal traffic.

For the purpose of this report, we define the following terms:

- **Misuse** Use or behaviour of a networked environment in any way that is not consistent with the system's expected functionality.

- **Misuse detection** Following the convention of the literature, this is taken to mean detection based specifically on identification of known misuses, *i.e.* signature based as opposed to anomaly detection.

- **Misuse Detection System (MDS)** A system designed to detect instances of misuse within a networked environment. When referred to as an MDS, it is meant in the general sense, and not restricted to systems operating on the misuse detection principle defined above.

- **Misuse localisation** Also referred to as fault localisation, this refers to the process of identifying the root cause of a misuse, *e.g.* naming a faulty network element.
- **Intrusion** Subset of misuse concerned primarily with unauthorised access to a system, or use of a system in an unauthorised way.
- **Intrusion Detection System (IDS)** A system designed to detect instances of intrusion within a networked environment.
- **Anomaly** Behaviour of a system that departs from known normal behaviour and may or may not constitute one or more instances of misuse.
- **Anomaly detection** Detection of misuses by identifying patterns of behaviour that deviate from learnt patterns of normal behaviour.

1.3 Artificial Intelligence

In this report, Artificial Intelligence (AI) is viewed as comprising a breadth of systems with the ability to learn, classify, adapt, and/or predict, to incorporate classical AI (*e.g.* symbolic AI and expert systems), related statistical techniques (*e.g.* clustering and regression), and natural computing (*e.g.* artificial neural networks, evolutionary computation, and swarm intelligence). In general, an understanding of AI techniques is assumed. A brief introduction to some of the basic principles of selected techniques of particular relevance to this report is included in chapter 2. For further information, readers are referred to the following texts: Russell and Norvig [186] and Luger [152] for a general treatment of artificial intelligence, Haykin [95] on artificial neural networks, Goldberg [79] and Mitchell [163] on genetic algorithms, Dorigo and Stutzle [57] on ant colony systems, Riesback and Schank [183] on case based reasoning, Mirkin [162], Everitt [65], Kaufman and Rousseeuw [120], and Jajuga [106] on clustering and classification, Tan [202] on data mining, Turban [208] on decision support systems, Giarratano and Riely [78] and Nikolopoulos [169] on expert systems, Cockayne and Zyda [41] and Pierre and Glitho [178] on mobile agents.

1.4 Structure

The remainder of this report is structured as follows:

Chapter 2 introduces the general notions behind a number of prominent Artificial Intelligence (AI) techniques, including 'programmed' paradigms, as exemplified by rule based systems, and 'learning' paradigms, as exemplified by artificial neural networks. This is not intended to be a comprehensive review of AI, but a brief overview of some of the concepts that impinge particularly on later discussion.

Chapter 3 provides a broad survey of the field of intrusion detection. The bias is towards the application of AI techniques, but also touches on some more traditional approaches, discussing, for example, clustering, feature extraction, and pattern matching. Much of the literature focusses on the detection of misuse (in the general sense) by inspection of low level traffic. Considerably less literature considers the correlation of high level alarms, although there is some recent work in this area. However, it is observed that many of the techniques could be equally applicable regardless of the level of the input data; that is, all of the techniques are concerned with identifying spatial patterns within event streams. Hence, it is necessary to conduct a particularly broad survey to ensure that promising techniques are not overlooked.

Chapter 4 treats the subject from the specific perspective of event correlation. It explores and formalises the general notions of event correlation and describes previous research solutions. There is, of necessity, some overlap with the preceding chapter, with each in essence representing a different 'view' of the same topic. A taxonomy of techniques is developed, which then provides a framework for considering specific approaches.

Chapter 5 provides an overview of current misuse localisation approaches and describes the general advantages and disadvantages of the each. This chapter focusses more on knowledge based techniques. However, it is advantageous to combine those approaches with the learning approaches, which are mostly covered in the precedeing chapters.

Chapter 6 summarises the key findings of the survey and analysis of event correlation. In particular, it reviews the weaknesses of rule based systems and makes recommendations for the exploration of specific directions for hybridisation.

Chapter 2

Artificial Intelligence Techniques

2.1 Introduction

This section describes the fundamental principles of several of the prominent Artificial Intelligence (AI) paradigms that have been applied to intrusion detection. The reader is referred to section 1.3 for references to appropriate literature for further reading. The intention is not to provide an exhaustive treatment, but simply to introduce some of the basic notions to readers who may not be familiar with AI, but who wish to appreciate the nature of its application to the domain of intrusion detection.

2.2 Rule Based System

The terms rule based systems and expert systems are often used synonymously within the literature. Strictly, an expert system is a rule based system, where the rules are derived from expert knowledge, through a process referred to as knowledge engineering. Expert systems form an approach to artificial intelligence that proposes to leverage the experience and knowledge of experts in particular fields. In general, a rule based system will contain a corpus of known facts about the field in which it is intended to possess expertise. In addition it will also contain a set of rules which represent relationships between these facts.

The main driver behind expert systems is the construction of systems with ability to infer new knowledge from existing knowledge. The set of facts and rules alone is really nothing more than a database. Although access to a database of information can be sophisticated and subtle, it is not possible to obtain new knowledge. It is only possible to present existing knowledge in knew ways. Expert systems have the ability to exploit the knowledge within the database to infer new knowledge. In all

cases this new knowledge is derivable from the existing corpus.

The clue to the strength of an expert system is in the term 'derivable'. This means that, within the context of a formal system, often predicate logic, the rules and facts can be manipulated to adduce new rules that were not previously part of the corpus. As mentioned, the formal system is often predicate calculus but other possibilities exist that allow greater flexibility but perhaps less determinism. In particular it is quite possible to use a probabilistic inferential mechanism to infer new knowledge that in this case may be true with some attendant probability. This type of inferential process is often called forward chaining because it moves forward from known relationships to new relationships in an incremental way.

Another procedure is also feasible where a relationship can be hypothesized and the inferential formal mechanism is then employed to find a path from this to a set of known facts and relationships that support this contention. This can be thought of as backward chaining. That is, the system is given new knowledge and is then asked to support this by showing an chain of inference from known facts and rules to this hypothesis; that is, in essence, proving the hypothesis.

The use of expert systems as an approach to artificial intelligence was initially very popular because it was assumed that true understanding in a field of expertise, a difficult concept bound up with the syntax versus semantics dichotomy, would eventually somehow appear as the facts and rules and new knowledge became richer and more comprehensive. However, an aspect of intelligence might conceivably be agreed to be the ability to generalise to new knowledge from known knowledge. It seems that humans can do this quite well but the canonical expert system approach can not. In this case we do not mean generalisation as provable knowledge but rather as similar knowledge, *e.g.* extrapolation.

An expert system has an advantage over many other approaches to AI because it is possible to see exactly how the new knowledge is derived. That is, we have absolute certainty about the existence and derivation of the relationships in the knowledge corpus. This determinism is important in some applications. Aside from generalisation, a key weakness of expert systems is the time and effort required to develop, and maintain, the knowledge containers, *e.g.* the rules that encapsulate the expertise.

2.3 Case Based Reasoning

AI is essentially concerned with effecting a (non-trivial) mapping from some input space to some output space. Case Based Reasoning (CBR) exploits an assumption that examples from similar points in the input space (however similarity might be

measured), map to similar points in the output space [136].

A representative example, where the utility of CBR has been well explored, is the estimation of effort required to complete a software engineering project. It is natural, when faced with such a question, to consider previous projects and their relationship to the project in question. One might then extrapolate or interpolate from one or more previous projects of a similar nature to make an informed estimate of likely effort on the new project. CBR is the foremost Instance Based Learning (IBL) paradigm, which essentially formalises this notion. Yaner [222] defined CBR as the reasoning about new problems by reusing past cases.

CBR differs from rule based models such as expert systems. A rule based model generalises by formally inferring new knowledge without necessarily having a target in view, known as eager generalisation, whereas CBR delays generalisation until a new case is available, known as lazy generalisation. This makes CBR an attractive approach for solving problems in complex domains where there exist many ways to generalise from a specific case. Riesbeck [183] outlined two differences between CBR and rule based reasoning. First, the knowledge base of CBR contains cases rather than rules, and, second, partial or fuzzy matching is inherent within the CBR paradigm.

The process of arriving at an inference using a canonical CBR approach involves four stages, often referred to as the 4'R's [165].

- The system starts by **R**etrieving the case(s) that are most similar to the specified problem description.
- Next it **R**euses the knowledge in the case(s) that were retrieved to solve the problem.
- Then it **R**evises the solution.
- Finally it **R**etains the parts of the experience that is most likely to be useful for future problem solving.

Mitra [165] points out that the reuse and revise stages in CBR are collectively referred to as the *Case Adaptation* process and that this procedure is absolutely necessary because some problem definitions that occurred previously often do not match with new problem definitions. Retrieval can be facilitated in a variety of ways, but is based on a distance of similarity metric. Any number of cases can be drawn from the case base to determine the solution, although this tends to be a fairly small number, which are then combined to make the best estimate for the new case. For

example, a simple approach where cases contain numeric data, might use the k-Nearest Neighbour notion to select the k cases that are most similar to the new case, and take an arithmetic mean to derive the (predicted) solution.

2.4 Artificial Neural Networks

Artificial Neural Networks (ANN) are a biologically inspired computational paradigm modelled on the functionality of neurons. Haykin [95] provides a standard reference to the field. The specific aspects of biological neurons that pass over to most ANNs are the weighted aggregation of input signals which are transformed to a single output signal (although more biological models are also being explored, based on networks of spiking neurons). Neurons perform a many to one transformation, and the functionality of an individual neuron is limited. In particular it is incapable of performing a classification of non-linearly separable data [161]. Hence, to increase the computational power networks of neurons are created by attaching the output of one to the input of one or many others. When we consider networks we again have an input to output transformation but now the output can be of dimension larger than one. Indeed, it is provable that this configuration (specifically a three layer fully connected feedforward network) possesses universal computational power.

Connecting artificial neurons together in this fashion can be achieved in many ways. There are in fact several standard topologies with understood characteristics. Perhaps the most common is the Multi Layer Perceptron (MLP) which is a topology formed from layers of neurons with no interlayer connections and only forward pointing connections between adjacent layers. Often within this restriction MLPs are fully connected. When connections are permitted between non adjacent layers and indeed some connections are permitted to be missing then the topology is just called feedforward. The computational power of either network is equal.

Operation is a compound application of a single neuron function. By this we mean that each single neuron performs its input to output transformation passing its output to any receiving neurons. They in turn perform a similar transformation and the entire process is repeated. For a feedforward network this concludes after a finite number of these individual neuron transformations have been executed and the entire network has settled to a stable state. However, if there are feedback loops in the topology, typically referred to as recurrent networks, then the network may have much more complex dynamics and stabilisation may not always be achieved.

Neural networks learn through a training process that adjusts, primarily, the weights

on input connections. There are many solutions to this training process, but they often are dependent on the topology of the network. Typically a training process will take a set of data where it is known what the transformation should be for each datum, and through a supervised process the ANN will be instructed on how to reproduce this transformation. For an MLP, a common supervision mechanism is called the backpropagation algorithm popularised by Rumelhart and McClelland [185]. Other supervised training algorithms exist for non feedforward or recurrent networks. These can be very complex and in many cases present extreme mathematical difficulties proving that they actually train the network. Optimisation heuristics such as Genetic Algorithms (GA) and Particle Swarm Optimisation (PSO) can also be used to evolve the topology and/or weights of an ANN provided a measure of the fitness for purpose of a given network can be determined, *e.g.* based on degree of success in learning the training transformations. ANNs can also be trained in unsupervised mode, where they essentially perform some form of data clustering via a number of different algorithms. The Kohonen network [127] is a popular example of this class of network.

The advantages of neural approaches include the ability to learn by example and automatically extract pertinent features of data sets, and to generalise to unseen data. The nature of neural networks is such that training does not just learn to replicate the transformations defined by the training data, but learns instead a more general function to facilitate this mapping. Hence, provided that the training regime is appropriate and the training data encapsulates sufficient knowledge of the 'essence' of this mapping, the neural network will be able to extract this knowledge, represent it in the form of connection weights, and exploit it to process samples that are similar in essence but different in value from those used in training. Disadvantages arise when considering the reliability and non-deterministic nature of generalisation and, of import in some domains, the inability to analyse the network for explanatory purposes. That is, it is often difficult to know what exactly an ANN has inferred from the training data and consequently how it will perform when presented with unseen data.

2.5 Genetic Algorithms

Genetic Algorithms (GA) are, perhaps, the most common form of a general class of related approaches referred to as Evolutionary Algorithms (EA). They are a biologically inspired search or optimisation heuristic based on notions of evolution and genetics. They offer an approach to the indirect solution of a problem, and as such can be applied to complex domains about which there is insufficient knowledge

of the problem or theoretical framework with which to reach an analytical answer. They are often employed to relatively quickly find good quality solutions in search spaces far too large to be considered for an exhaustive search.

Genetic Algorithms assume that a potential solution to a problem can be encoded in some form of representation reminiscent of a chromosome. Representations based on strings of numeric values (*e.g.* integer or floating-point), sequences and graphs (*e.g.* path, edge and ordinal), and trees (*e.g.* as applied to Genetic Programming) are common, but custom representations are possible. A function is defined (in the general sense of providing an input–output mapping; this could be a mathematical function, a simulation, or measurement of a physical process) that provides an assessment of the fitness for purpose of a given solution. A population of diverse randomised solutions is generated, which will typically be of low fitness. A selection process, inspired by the Darwinian principle of survival of the fittest, selects a number of individuals from the population to reproduce and thereby develop new solutions (offspring).

New solutions are created by genetic operators such as crossover, mutation and inversion, inspired by reproduction processes in natural populations. The assumption is that two or more solutions can be combined by a fusion of various traits encoded within the representation. This is not usually done in any prearranged way, but the underlying premise is that the offspring of two parents of above average fitness will tend to be also of above average fitness. This implies that the reproductive operators must combine candidate solutions in a valid and meaningful way. In addition to recombining existing traits from the parents, exploration of the solution space is facilitated by variation operators, such as mutation, which contribute random changes to the solutions to prevent a loss of diversity within the population. Offspring which are fit for purpose tend to replace those that are not so fit. The basic notion is that repeated application of the selection–reproduction–replacement cycle results in a continual increase in the average fitness of the population, as it searches for and exploits traits leading to good fitness scores. Genetic Algorithms are a particularly general problem solving technique requiring three elements: a representation, reproductive operators that can produce meaningful offspring from selected parents, and a fitness function to assess the utility of offspring. These do not have to be especially effective for a genetic search to function, for example, the fitness function could be approximate or noisy and the population will still learn to adapt to the environment. Similarly, the reproductive operators do not have to create fit offspring from parents very often, so long as they do so occasionally. It is generally considered a relatively easy task to get a GA to work, but a very difficult task to get a GA to work well, due to a range of factors, including the typically large configuration space of the algorithms, complexities of the search space, complex non-linear

dynamics of the algorithm that make theoretical analysis virtually impossible. The performance is ultimately dependent on the success of the reproductive operators and the balance achieved between the conflicting goals of exploration of the search space to find fit regions, and exploitation of discovered fit regions to improve accuracy. This balance is effected, to varying degrees, by all of the operators (*e.g.* selection, crossover, mutation, and replacement) and configuration parameters (*e.g.* population size, mutation rate, and selection pressure), hence fine tuning is challenging. As with neural approaches, it is difficult to say *a priori* what the result of a given GA will be on an unseen problem.

GAs are one of an increasingly large heuristic optimisation toolkit. Related techniques include Ant Colony Optimisation (ACO) and Particle Swarm Optimisation (PSO). Both are biologically inspired and population based. ACO is inspired by the emergent behaviour resulting from low level cooperation, facilitated by pheromone deposition, between ants in their search for efficient pathways from a nest site to a food source. PSO is inspired by the social flocking and schooling behaviour of various animal species, but particularly with reference to birds and fish. Both are effective optimisation paradigms, the former particularly for combinatorial optimisation, the latter for numerical optimisation. As with GAs, these techniques suffer from high computation cost of exploring a (potentially very large) search space through essentially non-uniform random transformations. These population based approaches compare favourably with 'trajectory' based algorithms, such as Simulated Annealing (SA) and Tabu Search (TS), on large and complex problems due to their true global-local search abilities.

2.6 Mobile Agents

Mobile Agents (MA) are autonomous software entities. They can execute in a quite general computational architecture, and exist in environments that facilitate movement between such architectures. Agents contain program code, internal state and data attributes [38]. Agents can communicate with one another, they are goal oriented, and can create duplicate instances of themselves. Therefore with this functionality, agents are mobile computing environments that can migrate between execution architectures. Advantages of mobile agents include their ability to support asynchronous task execution as client systems are able to continue performing other tasks while the agents act on behalf of the client on a remote site. Communication bandwidth is also reduced with this technology as computation (that is the agents) is transferred to the data and not vice versa. It also offers improved real time abilities as agents acting locally on a remoter site suggest a faster reaction to remote events

than if the events have to be communicated between the remoter machine and a central host. Agents have the ability to react dynamically to any undesirable condition and also provide improved support for intermittently connected devices. Chess [38] also discusses some security issues to be considered in implementing mobile based computing. The first is the authentication of both the user that sent the agent by the server and the server or agent execution environment. This is aimed at preventing attacks such as spoofing. Another issue is the confirmation of the rights of the user to execute applications on the server. Finally, it is necessary to determine the possibility of the agent infecting the server and denying network users and resources of any service.

In addition to network intrusion detection, this technology is also being implemented for embedded applications such as missile guidance and satellite monitoring Eswari [54]. Other application domains for mobile agents include data collection from multiple sources, searching and filtering, and parallel processing.

Chapter 3

Misuse Detection Systems

3.1 Introduction

This section considers the general problem of misuse detection from a variety of perspectives. Due to the extensive literature in the area of intrusion detection, this forms the core of the discussion. However, the techniques are considered to be equally applicable to a range of misuse detection systems. For the most part, it avoids specific discussion of issues related to event correlation, which is considered in section 4. Instead, techniques applicable to low level data, *e.g.* TCP/IP data, dominate both the literature and this part of the survey. A broad range of techniques has been surveyed, with a particular bias towards Artificial Intelligence (AI) approaches, and some overlap between this chapter and chapter 4 is unavoidable. However, the perspective is different. In addition to AI techniques, a number of related issues are also discussed, such as clustering, pattern matching and profiling, as they crop up frequently in the literature. In general, however, traditional approaches have not been surveyed to any great degree.

This chapter organises the literature according to the technique involved. However, some techniques do not easily fit within these categories, whilst others compound two or more approaches. In such cases, the techniques have either been placed under the category of the dominant approach, or where they make most sense to the discussion. On occasion this has placed seemingly unrelated techniques together due to some shared underlying notion. The section on data mining is noteworthy principally because it does not reflect a specific technique, but rather, the general concept of (typically) extracting information from large data sets using unsupervised learning methods. Some of the approaches within this section could, therefore, be relocated to specific categories based on the techniques used, whilst, occasionally, approaches that could reasonably be referred to as data mining, appear elsewhere.

This is deliberate, and is again based on the nature of the discussion. The section on hybrid approaches is similar, in that one could treat them under the categories of their constituent technologies, however, in some cases, the arguments are about specific approaches, whilst in others they are about combining approaches to leverage advantages of multiple paradigms. Rather than replicate the treatment of the techniques under each of the applicable categories, they are generally considered only once, wherever deemed most appropriate.

It is evident from the literature, and hence reflected in this survey, that a number of key concepts are common across a range of work. For example, the opposing notions of anomaly detection and misuse detection, with the former being able to generalise to some degree to unknown attacks but suffering from a high rate of false alarms, and the latter generating few false alarms but not being able to generalise. Other common factors include the idea of profiling, *i.e.* constructing representations of behaviour, whether at user, application, or system level, and an assumption of consistency of profiles over time; the notion of feature extraction, which is of import with a wide range of techniques; and the dominance of rule based approaches for misuse detection and non-rule based approaches for anomaly detection.

3.2 Overview of Intrusion Detection

Intrusion detection is the process of monitoring and analysing events that occur in a computer or networked computer system. Although the term 'intrusion' has quite specific connotations, intrusion detection is often used in the literature in a general sense, to refer to analysis performed to obtain indications of fraud, misuse or unauthorised access, *e.g.* Bace and Mell [13]. Intrusion detection systems employ techniques for modelling and recognising intrusive behaviours in a system, which come in different variants. Bolton and Hand [23], for example, identify the following fraud variants:

- Frauds aimed at the service provider. For example the resale of stolen call time.
- Frauds enabled by the service provider. For example, interfering with telephone banking instructions.

Typically, Intrusion Detection Systems (IDS) first operated at host level. In this context the IDS monitor was concerned with single host machines and made extensive use of the host operating system. However the trend has shifted to networked systems with the IDS monitoring any number of systems on a network by analysing the

audit trails and the network traffic [121, 122]. Kim and Bently [121, 122] consider requirements for network based IDSs to include: robustness, extendibility, scalability, adaptability, configurability and efficiency. Kachirski and Guba [116] observe that traditional IDSs for fixed networks can be considered as either network based or host based. A network based IDS functions by probing the network and examining packets, listening either actively or passively to the network traffic. Conversely, a host based IDS functions by detecting actions in each host in a network, and normally operates by accessing log files or monitoring real time system usage, such as failed access attempts or attempted modifications to system files.

Two main categories of intrusion detection (ID) are misuse detection and anomaly detection [138]:

- **Anomaly detection** (AD) is used to identify patterns of behaviour that differ from learnt patterns of normal behaviour, and hence might be representative of malicious activity.
- **Misuse detection** (MD) is used to identify the occurrence of specific patterns of behaviour that are known to represent malicious activity.

Some intrusion detection systems offer both capabilities, typically through hybridisation. The limitation of anomaly detection is that it could generate a high rate of false positives Dokas *et al.* [56], *i.e.* when the system is not sufficiently aware of all of the possible legitimate patterns of behaviour and assumes that a previous unseen pattern constitutes an attack. Another possibility is the presence of an unknown intrusion in the sample data on which the system learnt the expected normal behaviour. Conversely, misuse detection has the limitation of being unable to generalise from known attacks to unknown attacks, and is therefore prone to generating false negatives.

Verwoerd and Hunt [211] recognise some common building blocks for an intrusion or misuse detection system:

- **Sensor** probe that gathers data from the system under inspection
- **Monitor** that receives events from a number of sensors and forwards suspicious content to a 'resolver'
- **Resolver** that determines a suitable response to suspicious content
- **Controller** that provides administrative functions

Verwoerd and Hunt also describe a range of modelling techniques for both anomaly detection and misuse detection. For the former, they consider techniques such as artificial immune systems, artificial neural networks, protocol verification, statistical modelling methods and clustering analysis. For the latter, they consider techniques such as expression matching, genetic algorithms, state transition analysis, and the use of dedicated languages (*e.g.* such as proposed by Paxson [176] and Ranum *et al.* [182]). It is clear from the literature that a diversity of both traditional and AI based techniques can be applied to both anomaly and misuse detection. However, when it comes to event correlation, it is equally clear that rule based approaches are not only well established[1] but dominate the field. The limitations of rule based approaches centre around two key points: the difficulty of developing and maintaining the rule base, and a lack of generalisation ability. Thus, a range of AI techniques, such as artificial neural networks, have been combined with rule based systems in order to alleviate these difficulties. Such hybridisation remains an active area of research.

The following sections consider some of the diverse approaches from the literature, starting with canonical rule based systems, before moving on to consider approaches based on more contemporary AI.

3.3 Rule Based Systems

Much of the literature concerning rule based and expert systems focusses, specifically, on event correlation, and is therefore treated in section 4. Further, many rule based systems have been hybridised with a range of other techniques. Examples are mentioned below, but are treated in more detail in section 3.12. In this section, several examples of conventional rule based systems are provided.

Lindqvist and Porras [146] describe an expert system toolset referred to as the Production-Based Expert System Toolset (P-BEST). This was applied to the development of a generic signature analysis engine for computer and network misuse detection. Rule sets where described for detecting subversion methods (*e.g.* SYN flooding and buffer overruns), to which systems are often vulnerable. The approach consists of a rule translator, a runtime library, and a set of garbage collection routines. The results from the evaluations performed, together with some examples, indicate that expert systems of this type are a good choice for handling real time misuse detection. It was observed that the ease of implementation of this system

[1] For example, systems such as Multics Intrusion Detection and Alerting System (MIDAS) [188] and Next Generation Intrusion Detection Expert System (NIDES) [8], both of which are based on the Production Based Expert System Toolset (P-BEST) [146].

and its close integration with the C programming language makes it easy to use and flexible. P-BEST was found to be adequately fast for real time detection of most widely used attack methods.

Lindqvist and Porras [147] later describe a real time forward reasoning expert system, eXpert-BSM, that is able to analyse Sun Solaris audit trails. The system has a large knowledge base, comprised of years of intrusion detection research, and is thus able to detect a wide range of specific and general types of misuse. It is host based and operates as a security daemon within the host, using few CPU cycles and little memory. Different sets of eXpert-BSM rules may be deployed throughout a network, and their alarms managed, correlated, and used by local security services. The system is purported to be able to provide detailed reports and recommendations to system administrators, whilst offering a low rate of false alarms.

Vaarandi [209] presents a platform independent open source tool for rule based event correlation called the Simple Event Correlator (SEC). This tool is able to deal with the complexity, size and cost issues typical of event correlators. SEC configuration is stored in text files and each configuration contains one or more rules. Sets of rules from different files are applied logically and in parallel. SEC receives input events from pipes, files, and standard input, and can therefore be utilised by any application that is able to write its output events to a file stream. It produces output events by executing shell commands specified by the user. Of particular interest, is the range of correlation rule types that the system supports. As specified in [209], these include *inter alia*

- Rules that involve a simple match with an input event and the execution of some action.
- Matching an input event, executing a list of actions and applying some external script or program if certain exit values are returned.
- Matching input events, with execution of an action list, whilst ignoring subsequent matching events for a certain time period.
- Matching input events, executing an action list, then ignoring subsequent matching events until some other input event arrives, at which time another action list is executed.
- Matching an input event and waiting for a certain time period for another input event to arrive, and then executing an action if it arrives on time.
- Counting matching input events over a certain number of seconds and, if some threshold is exceeded, executing an action list and ignoring subsequent matching events during the remaining time span.

Other rule types exist, and this highlights the need for a rich language for rule specification, to enable the varied relationships between events to be satisfactorily correlated.

Rule based systems have been hybridised with a variety of techniques. Hanemann *et al.* [86] describe a combination of a rule based reasoning engine with a case based reasoning engine. Yang *et al.* [223] propose the use of cluster analysis and expert systems. Marin *et al.* [155] present a hybrid technique that applies expert rules for the reduction of the dimensionality of the data, followed by initial clustering, and subsequent refinement of the cluster locations using a competitive network, Learning Vector Quantisation (LVQ). Bridges and Vaughn [24] present a hybrid of fuzzy data mining and expert system techniques. Guerer *et al.* [84] present an approach that utilises both neural networks and case based reasoning techniques for handling fault management in distributed and heterogeneous information networks, with an expert system to filter network alarms. See section 3.12 for further discussion of hybrid systems.

3.4 Case Based Reasoning

Case based reasoning (CBR) refers to reasoning about new problems based on past known cases [222]. Somewhat surprisingly, CBR has not been found to be a common approach despite its attraction in prediction, based predominantly on pattern matching. This may be due to the non-trivialities of the representation of complex event patterns in a form amenable to CBR. There are, however, some examples, such as Koford-Peterson and Aamodt [126], who describe an approach for identifying and specifying user situations in a context aware mobile environment. This approach uses CBR as the main reasoning engine of the system, and incorporates it within a multi agent system for providing the user with personalised and context sensitive information. The reasoning mechanism of the approach is adaptive and thus it is relatively easy to introduce new services.

Canonical CBR systems have been extended to handle sequences of concurrent events from multiple sources. Ceaseless CBR [157] is essentially a modification of the retrieval and reuse stages of traditional CBR. Cases that are retrieved are sequences of sequences of alerts, which the system is able to sub-divide into sections as the alerts are received. Reuse involves the exploration of the grouping of the cases that best describes the most recent occurring events.

Case based reasoning approaches have also been hybridised and applied to intrusion detection in various studies. Some of these include studies by Hanemann [86], who

proposed a hybrid approach that combined a rule based reasoning engine and a case base reasoning engine. See section 3.12 for further discussion of hybrid systems.

3.5 Pattern Matching

Pattern matching algorithms are a critical element in many signature based approaches. The term refers to the process of searching for certain patterns, expressed as sequences or tree structures within a body of data. It is most often employed to identify attack signatures in network packets [105]. Weaknesses of the approach include the computational load, as the number of comparisons required to ascertain the presence of a range of signatures within a large sample of traffic can be large, and the use of fixed signatures, which limits detection to known cases [105]. Techniques range from general purpose string matching functions to more specialised matching engines, customised for instruction detection.

Kumar and Spafford [132] describe a generic model of matching based on Coloured Petri Nets (CPN), a form of Petri net where the arcs contain data and can be used to describe a variety of different systems. In their approach, pattern matching is used to detect system attacks, with the knowledge concerning the nature of attacks represented as specialised graphs. The graphs are an adaptation of CPNs, with vertices representing the system states and signature contexts represented by the guards. The graphs thus represent those state transitions that lead to a detectable intrusion state. Their system separates the various concerns of a generic misuse detector into components

- The information layer, encapsulating the audit trails.
- The signature layer, providing the internal representation of signatures.
- The matching engine.

Wespi *et al.* [217] present an approach to build tables of variable length patterns based on Teiresias, which was originally developed for the discovery of patterns in biological sequences. This study applied the approach to the detection of attacks against UNIX processes, achieved by observing behavioural deviations in the processes. The system uses a table of patterns to model the normal behaviour of such processes, based on audit events, and offers both off-line and real time detection.

Kuri *et al.* [133] present a pattern matching approach based on insertion distance. This model handles an attack as a sequence of letters, and the algorithm detects the

text portions everywhere the events of the attack appear, in order, within a window of k other events. This enables the algorithm to quickly filter out large portions of the text and leave the remaining parts to be examined by another algorithm. The approach was evaluated using audit trails and an attack database, and was successful in addressing some of the problems of speed and complexity of intrusion detection algorithms.

Markatos *et al.* [156] describe a string matching algorithm designed specifically according to the features of network intrusion detection string matching. This approach, called ExB, provides exclusion based multiple string matching. The central notion is the examination of the contents of each packet received, to confirm if it contains all fixed sized bit strings of the signature string while not taking into consideration whether the bit strings appear in sequence. It was evaluated against the set of algorithms being implemented in SNORT using trace driven execution with real packet traces. Reported advantages include efficient and easy implementation, and considerable improvement in system performance, particularly with increasing packet size, having good scalability to increasing rule sets.

Clark and Schimmel [40] describe a module generator making use of non-deterministic finite automata to create efficient circuits for matching patterns, as specified in a standard rule language. The system, consisting of a rule parse and circuit generator[2] is able to translate a SNORT rule file into a circuit description, which is then used to match the content strings. The system also offers approximate matching.

Anagnostakis *et al.* [7] describe a string matching algorithm, E2xB, for increasing the efficiency and capacity of network intrusion detection systems. The approach is based on confirming the presence of every character of the input string in the space searched. E2xB was implemented in SNORT and performance compared against the alternative solutions of Fisk and Varghese [67]. Fisk and Varghese based their work on a set-wise Boyer Moore-Hospool algorithm, an adaptation of the Boyer-Moore algorithm, which is shown to be faster for matching less than 100 patterns. For typical traffic patterns, E2xB gave increase performance and improvement detection capability.

Norton [170] presents an algorithm for intrusion detection based on an optimised version of the Aho-Corasick algorithm [5]. The latter was designed to be able to locate all occurrences of any of a finite number of keywords in a string based on a finite state pattern matching machine. The version described by Norton is an enhancement to the implementation presented in SNORT [31], employing an optimised vector implementation of the Aho-Corasick state table, with improvements

[2]Specifying FPGA circuit components and their connections.

in algorithm performance and on large pattern groups. A SNORT based intrusion detection performance test, and several dictionary tests, indicated significant improvements in pattern matching capacity, whilst being less memory intensive.

Abbes *et al.* [2] introduced an 'on-the-fly' multi-pattern IDS for real time analysis of network traffic. The central notion is to process packets as they are received and strip out and retain only those aspects that are required for the pattern matching function. The approach, employed as an extension of SNORT, was said to be robust to denial of service and port scanning attacks, and to address the complicated techniques that are employed to evade an IDS.

Song and Lockwood [193, 194] present a novel packet classification architecture called BV-TCAM for a FPGA based network intrusion detection system. This architecture combines the Ternary Content Addressable Memory (TCAM) (used for packet classification) and the Bit Vector (BV) algorithm to compress the data representations and enhance throughput. The BV algorithm performs the function of lookup of source and destination ports, whilst the TCAM performs the role of lookup of the other header fields. It is able to report multiple matches at fast network link rates and sustain traffic throughput. Song and Lockwood [194] also introduced a data structure known as an Extended Bloom Filter (EBF) [22] and a hardware based algorithm for multi-pattern signature matching. A further technique, to support long signature matching, was also introduced, enabling the maintenance of a limited number of supported signature lengths for the EBFs. It was demonstrated that fast exact matching for thousands of signatures was achievable at a reasonable hardware cost. The algorithm was integrated into a hardware based network intrusion detection system and demonstrated to be highlight efficient in terms of throughput and memory utilisation, achieving fast exact matching for thousands of signatures.

Antonatos *et al.* [10] describe an algorithm for pattern matching, referred to as Piranha, that is based on the notion that if the rarest substring of a pattern is not present, then the whole pattern cannot match. The algorithm is implemented within SNORT customised for intrusion detection and is said to offer significant improvements in both processing and memory utilisation, when compared with existing approaches, such as the Wu-Manber [220] multi-pattern matching algorithm (MWM) and domain specific exclusion based string matching algorithm (E2xB) [7].

3.6 Clustering and Feature Extraction

Clustering is primarily used in the detection of patterns in network traffic. Portnoy *et al.* [181] use single linkage hierarchical clustering to separate anomalous and normal traffic patterns from the KDD-Cup99 data set. This method is able to automatically detect both new and known intrusions. Their algorithm follows the typical agglomerative approach; *i.e.* each new instance retrieved from the the normalised data set is compared with the existing clusters and, following the single linkage rule (*i.e.*, based on the distance between the instance and the nearest of the current members of the cluster) is added to the nearest cluster, provided that it is within a predefined cluster width. If it falls outside of that width, it becomes the first member of a new cluster. Whilst the need to specify *a priori* the number of clusters, it does require an appropriate value for the cluster width to be determined. The authors found a detection rate of approximately 50% and false alarm rate of below 1%.

Leung and Leckie [141] present a grid based clustering approach for unsupervised anomaly detection. Taking unlabelled data as input, the technique attempts to find intrusions that are embedded in the data, based on two assumptions from Portnoy *et al.* [181] and Javitz and Vadles [111]. The former observe that the majority of network connections represent normal behaviour, whilst the latter contend that traffic due to an attack is statistically different from normal traffic. It is this latter property that is of particular import to traditional clustering and classification techniques. The benefit of such a system is its ability to detect previously unknown attacks. The authors also claim that unsupervised learning with unlabelled data improved the efficiency of the detection system as it removes the requirement for a database of signatures. Evaluation on actual network data suggests that results were consistent with similar studies by Oldmeadow *et al.* [174] and Eskin *et al.* [64]. However the underlying assumptions do lead to limitations. The authors provide the example of detecting a bandwidth denial of service (DoS) attack, where there are as many instances of intrusion as of non-intrusive behaviour. The same underlying principles hold for many techniques of this nature, and they can suffer the general drawback of anomaly detection, that of being prone to issuing false positives.

Leon *et al.* [140] tackled intrusion detection with an approach based on Unsupervised Niche Clustering (UNC). They described UNC as a genetic niching technique for clustering that is robust to the effects of noise and can automatically determine the number of clusters in a system. Niching is a diversity mechanism that is capable of maintaining multiple optima in genetic and other population based search techniques [187]. Leon *et al.* combined UNC with fuzzy set theory and a membership function associated with each of the evolved clusters.

Hu and Zhan [103] also consider a hierarchical approach, but go on to develop a technique for optimising the cluster allocation by identifying anomaly clusters with degrees of membership, and optimising the system to maximise the detection rate and minimise the false alarm rate. The authors validate their work on the DARPA data set, and find detection rates of over 70% for known attacks and 50% for unknown attacks.

Clustering also forms a part of a number of hybrid approaches to intrusion detection. For example, Yang *et al.* [223] proposed a hybrid of cluster analysis and expert systems, Marin *et al.* [155] present a hybrid technique that applies expert rules for the reduction of the dimensionality of the data, followed by initial clustering, and subsequent refinement of the cluster locations using a competitive network, Learning Vector Quantisation (LVQ). Refer to section 3.12 for more detailed consideration of hybrid approaches.

3.7 Artificial Neural Networks

Neural networks have been applied in a variety of intrusion detection contexts; on their own merits for the detection of attacks, to perform some preprocessing of input data, and as part of hybrid approaches. Their strength lies in their ability to learn by example, automatically extract pertinent features from data sets, and, critically, to generalise from known cases to the unknown.

Ghosh and Schwartzbard [76] employed a neural network to learn the profile of normal system behaviour. This was then applied to both anomaly and misuse detection. Their approach is process based and capable of generalising from previously observed attacks to novel attacks. An adaptive neural approach to anomaly detection was presented by Cannady [29] that was able to learn new attacks without the need for complete retraining. It also offered the ability to autonomously improve on previous analysis.

Jing Xin *et al.* [113], motivated by the need to reduce the rate of false positives, applied a neural network based classifier, within a modular system. They suggest a need for different modules that a neural network based intrusion detection systems should include:

- packet monitor for capturing network packets for input to the detection system
- feature extraction to transform the network stream into a feature vector for classification

- neural network based classifier to infer from the feature vector the absence or presence of an intrusion
- mechanism for providing feedback to the user
- database of intrusion samples for training the neural network
- event database for recording the correlated information on occurrence of an intrusion

Their approach adopted a Back Propagation (BP) classifier and, although it reduced the rate of false positives against traditional intrusion detection systems, they identified a limitation in the generalisation ability of the classifier.

Lei and Ghorbani [139] describe a technique for detecting network intrusions based on the Standard Competitive Learning Network (SCLN). This is essentially a single layer network, with output neurons fully connected to the input neurons. Output neurons compete to activate, with winning neurons being reinforced. There is, therefore, some similarity with the notions of the Self Organising Map SOM. The performance of the approach, applied to the KDD-Cup99 data set showed similar detection rates for both the SCLN and a SOM, however the SCLN is said to require only one fourth of the computational effort of the SOM.

Chen *et al.* [37] note that it is computationally expensive to examine all of the features in a data set to detect misuse patterns and further observe that some of the data is redundant. This led them to propose a mechanism based on an enhanced Flexible Neural Tree (FNT), to identify the important input features for building detection systems that were effective and computationally efficient. An evolutionary algorithmwas used in developing the FNT structure, whilst particle swarm optimisation was used to optimise the parameters. Exploration of the approach suggested that a reduction in the feature space is possible, and that this does lead to performance improvements.

Depren *et al.* [55] introduced an architecture combining an anomaly detection module and a misuse detection module with a decision support system that combines the results from the individual detection mechanisms. Anomaly detection uses a SOM to model normal behaviour. The SOM essentially transforms high dimensional input data to a low dimensional (often two dimensional) output space, with different output neurons representing the classifications of the input data [39]. Classification is based on similarity, *i.e.*, grouping together entities that are geometrically close. This can also be used as a means to visualise complex data sets. Hoglund and Hatonen [101], for example, present a system that provides automated anomaly

detection as well as visualisation of user behaviour. This study focussed in particular on the visualisation aspect, which it achieved using a a two dimensional Self Organising Map (SOM). Returning to Depren *et al.*, their misuse detection module employed the J.48 decision tree algorithm to classify various types of attacks. A rule based decision support system then interprets the results from both modules. Performance evaluation based on application to the KDD-Cup99 data set suggests that the combination is more effective than either approach used in isolation.

Eskin *et al.* [64] presented an algorithm for anomaly detection using feature mapping. Two feature maps were created, one for network connections and one for call traces. A number of algorithms were applied to the feature maps with a view to identifying sparse regions of the map, the contention being that these are representative of anomalies. Evaluation on the KDD-Cup99 data set showed that the approach was successful in detecting intrusions from unlabelled data.

Neural networks have been widely used as components in hybrid systems. Pan *et al.* [175] combine neural networks with the C4.5 decision tree algorithm. Zhang *et al.* [230] combine statistical preprocessing and neural network classification to construct a hierarchical intrusion detection system. Taniguchi *et al.* [204] combine feedforward neural networks based on supervised learning, with Gaussian mixture models and Bayesian networks. Endler [60] examined statistical likelihood analysis and neural networks, where the former was used for anomaly detection and the latter for misuse detection. See section 3.12 for further information on hybrid approaches.

3.8 Genetic Algorithms

A Genetic Algorithm (GA), a class of evolutionary algorithm, is a population based search algorithm, which can be applied to a wide range of numeric and combinatorial search and optimisation scenarios. It is often used to evolve optimal rules for use within other detection mechanisms, to automate the process of knowledge generation. Sinclair *et al.* [191] suggested that GAs can be used to evolve simple rules for network traffic, which can then be sorted in a rule base and used to filter network traffic. Li [144] also use GAs to evolve *if...then* rules to differentiate between normal and anomalous connections, based on both temporal and spatial information related to network connections. Pillai *et al.* [179] also adopt a GA to generate rules for specific connections based on TCP/IP fields.

Balajinath and Raghavan [14] present a genetic algorithm based intrusion detection (GBID) system, which uses genetic algorithms to learn a network user's behaviour, with regard to commands issued. Within a given sample of commands, three mea-

sures are applied: the ratio of commands predicted correctly to the length of the sample, the distribution of commands, and the number of commands occurring that are not in the user's past command history. These measures of current behaviour are compared with known non-intrusive behaviours to identify anomalous behaviour patterns. Features of this approach are said to include a low false alarm rate and the ability to detect new attacks in real time. In their study, a detection accuracy of 96.8% was reported.

Gonzalez *et al.* [80] use an evolutionary approach to generate fuzzy signatures for detecting intrusions, referred to as Evolving Fuzzy Rules (EFR). This was tested on a number of different data sets and is said to offer advantages such as a clear separation between anomalies and normal network activities, attributed to the fuzzy representation of the rules. Abadeh *et al.* [1] also propose an evolutionary fuzzy rule learning algorithm, in this case employing particle swarm optimisation (PSO), another population based search heuristic, to generate a set of rules which then form the basis of an IDS. The proposed evolutionary algorithm is based on the Michigan approach [12].

3.9 Artificial Immune Systems

In the context of intrusion detection, artificial immune systems, applying mechanisms inspired by natural immune systems, are typically applied to identify anomalous patterns ('non-self'), based on an understanding of normal behaviour ('self'). Forrest *et al.* [68] present an approach, known as negative selection, based on the mechanism used by the immune system to train T-cells to recognise antigens and prevent them from recognising the body's own cells (the self or normal cells). This essentially discards detectors that match any self element. Hofmeyr and Forrest [100] successfully applied a variation of this algorithm for anomaly detection.

Dasgupta and Attoch-Okine [48] present a study on immune system approaches to intrusion detection. They draw comparisons between the idiotypic approach, based on the reaction of antibodies to each other as well as the antigen, and the negative selection based approach, or self/non-self model. They surveyed the application of both techniques to anomaly detection, pattern recognition and fault diagnosis. Dasgupta [45, 46] later considered the application of immunological principles to the design of multi-agent intrusion detection systems, where immunity based agents roam around the system (*e.g.* nodes or routers), monitor the situation in the network and search for changes such as malfunctions, abnormalities, misuse, intrusions, etc. Agent activities are coordinated in a hierarchical manner, with agents being able to mutually detect the abilities of other agents. The agents are able to learn and

adapt to their environment and detect both known and unknown intrusions and the techniques work simultaneously on different levels (*e.g.*, packet, process, system and user), and can detect misuse from within the system as well as outside attacks.

Dasgupta and Gonzalez [47] applied the negative selection mechanism for the detection of patterns that are alien to a network. They presented approaches based on both positive characterisation (PC) and negative characterisation (NC). For the former, positive samples from the training data were used to build a categorisation of self space. The measure of abnormality of an element was measured by the distance from the element to the nearest neighbour in the self set. It was observed that memory requirements can be a drawback when it is necessary to store all the samples that constitute a normal profile, which calls into question the feasibility when applied to real volumes of network traffic, hence the motivation for the NC approach, which uses a real valued representation to characterise the self/non-self space and a genetic algorithm to evolve the set of detectors.

Hofmeyr and Forrest [99] also describe a distributed intrusion detection system, operating at the TCP/IP level. They adopted the self/non-self approach in their architecture, and defined self as a set of normal pair-wise connections and non-self as connections not normally observed on the network. The system contains immature, mature and memory detectors. The former are randomly generated and undergo a negative selection process to remove those detectors that bind to self. The mature and memory detectors are responsible for detecting anomalous activity. A successful matching of a non-self by a mature detector, causes it to become a memory detector, which has a longer lifespan. Various other immune system concepts were also integrated within their architecture. The system was reported to give favourable results when evaluated on a 50 machine subnet.

Hou *et al.* [102] adapted the work of Hofmeyr and Forrest [99] for monitoring TCP/IP traffic on a simulated broadcast local area network. They employed constraint based detectors, which they argue are easier to understand compared with those based on binary strings, and facilitate analysis of the traffic, during an intrusion, by the system security officer. By varying parameters of the technique Hou *et al.* managed to improve the detection rate.

Kim and Bentley [121] investigated the role of negative selection in the performance of artificial immune systems for intrusion detection. By applying this mechanism to real network data, they determine that negative selection is infeasible as an approach to generating detectors, as the computation time prevents sufficient detectors from being found. They conclude that negative selection is best applied to filter invalid detectors rather than as a mechanism for generating competent detectors.

Kim and Bentley [122] later describe a dynamic selection algorithm referred to

as Dynamic Clonal Selection, dynamicCS. Adopting the technique described by Hofmeyr and Forrest [99], a particular feature of this technique is its ability to adapt to continuously changing environments, and thereby learn the patterns of self and predict new patterns of non-self. Their study shows that dynamicCS is effective at incremental learning on converged data and can adapt to new attack patterns as they arise.

Balthrop *et al.* [15] described an artificial immune system framework known as LISYS, specialised for network intrusion detection. It learns to detect abnormal packets by observing normal traffic. By using only a partial sample of network traffic, it is able to generalise from its observations to correctly classify abnormal behaviour patterns. This again uses negative selection to retain only those detectors that fail to match normal traffic patterns.

Hall and Frincke [85] proposed an architecture known as Immune System Network Intrusion Detection System (ISNIDS), for intrusion detection in a broad based multi-enterprise misuse management system. This comprised two related systems – a primary and a secondary IDS, able to communicate across the network. The primary IDS is centralised and creates the detectors, whilst the secondary IDS is distributed and responsible for data gathering, reduction, detection and response, and forwarding successful detections to the primary IDS. A rule based detection scheme was built to facilitate the evaluation process, and results indicated that the immune system approach was better at detecting anomalous patterns.

The development of systems inspired by immune system principles remains an active area of research. Other work includes Gu *et al.* [83], who employed an immune system for the detection of Internet viruses and hackers attempting to penetrate the system, and De Castro and Timmis [50], who describe an immune system approach for pattern recognition and detection, where the knowledge is distributed among the components of a population based system.

3.10 Agent Based Approaches

Agent based approaches, and Mobile Agents (MA) in particular, are of interest for the detection of attacks within a distributed environment. They have been applied by numerous authors, for example, Zhicai *et al.* [229], Helmer *et al.* [96] and Kachirski and Guba [116]. However, some authors point to disadvantages or challenges with the approach. For example, Jansen *et al.* [110] consider problems related to speed, volume of the code required to implement an MA, deployment, limited methodologies and tools. They can also present threats to a network when they are often

required to execute with administrative privileges to perform autonomous diagnoses and responses. Jansen *et al.* have also highlighted a number of advantages, including mobility, overcoming network latency, robustness and fault tolerance. Agent approaches have been applied for both anomaly and misuse detection in a range of studies.

In the model developed by Helmer *et al.* [96] the mobile agents travel between the monitored systems in a network and obtain information from data clearing agents. This information is then organised, correlated, then reported to a user interface and database through mediators. Kim *et al.* [123] observe that different security rules may be needed in different security systems within distributed networks, and the propagation of these rules is important. They made use of mobile agents to facilitate this propagation in large-scale networks, and report that it is able to spread the rules rapidly. Kachirski and Guba [116] also proposed a distributed IDS based on mobile agent technology and applied this to ad hoc wireless networks. Their approach merged audit data from multiple network sensors and analysed the entire network for actions that suggest the occurrence of intrusions.

Zhicai *et al.* [229] proposed a hierarchical model of multi-level intrusion detection. They observe that intrusive behaviours are becoming more complex and hence detection is becoming increasingly more difficult. They explored the use of intelligent mobile agents to develop a distribution IDS. The model adopted a multi-level approach, employing node, subnet and network level detectors. The detectors are autonomous yet can share information to work together to detect complex intrusions. The authors found this model to be capable of effectively identifying complicated attacks, whilst having the additional advantages of reducing the the communication load and adaptation to changing environments. Deeter *et al.* [53] also describe a system for efficient and flexible distribution of analysis and monitoring tasks. Their approach, referred to as APHIDS (A Mobile Agent Based Programmable Hybrid Intrusion Detection System), defines a high level scripting language which is application specific, to identify the interaction between monitoring and analysis agents. The motivations in this case refer to bandwidth and processing scalability.

Ahmed *et al.* [4] used mobile agents to monitor the usage of various system resources to detect deviations from normal usage. Their approach, known as Intrusion Detection System Using Distributed Agents (IDSUDA), was claimed to extend the capabilities of conventional intrusion detection systems. The agents monitor network traffic behaviour at multiple levels (*e.g.* packet, process, system and user) to identify anomalous patterns.

Somewhat related to agent based systems are models based on ant colony systems. Tsang and Kwong [207] presented a multi-agent IDS framework for decentralised

intrusion prevention and detection. The learning is based on an ant colony clustering model, ACCM, and four unsupervised feature extraction algorithms are applied to address the problem of dimensionality. The feature extraction algorithms were evaluated on their effectiveness to enhance the clustering solution, by application to the KDD-Cup99 data set. Results suggest that ACCM combined with one of the feature extraction algorithms enables effective detection of both known and unknown attacks with a high detection rate, whilst screening normal traffic with a low false positive rate. Banerjee *et al.* [16] describe an adaptive ant colony based method for intrusion detection that is also able to monitor intruder trails. It was designed for securing sensor networks, and is referred to as Intrusion Detection based on Emotional Ants for Sensors (IDEAS). Soroush *et al.* [195] proposed an ant colony based data miner for intrusion detection, that is able to extract a set of classification rules from network data to distinguish between normal and anomalous behaviours.

3.11 Data Mining

Data mining is not a specific paradigm, but refers instead to the processing of a (typically large) data set with a view to summarising the data into a more usable form and/or gaining insights concerning patterns in the data that are statistically reliable [58]. Data mining comprises a variety of tools, including those drawn from statistics, machine learning and natural computing. Such approaches, could, therefore, be be distributed amongst those sections concerned with their component technologies, however, they approach the problem of intrusion detection in the same sort of way – by mining the data (typically using unsupervised learning) to extract pertinent relationships. A number of examples from the literature are briefly presented.

Lavrac and Dzeroski [135] and Ko [125] employ Inductive Logic Programming (ILP) to develop rules for detecting attacks from analysing specifications. Lee and Stolfo [137] used data mining techniques to discover models that describe system and user behaviours based on network logs. Pertinent system features were also used for creating classifiers that can recognise anomalies and known intrusions. Lane and Brodley [134] also considered the problem of characterising the behaviours of an individual system and its users, to facilitate anomaly detection. However, their approach was based on a form of Instance Based Learning (IBL). In order to apply IBL, they defined a transformation from a sequence of observations to a metric space, then the features of valid user behaviour and a domain heuristic were used to select classification boundaries. Their empirical evaluation established that the system is able to differentiate between the profiled users and other users, provided that sufficient information has been encoded by the system features. War-

render *et al.* [215] explored the ability of different modelling methods to accurately represent normal behaviour, and hence identify intrusions, by examining data sets comprising calls to the operating system kernel. Their work favoured a Hidden Markov Model (HMM).

Neri [168] examined distributed genetic algorithms and a heuristics based algorithm for modelling network traffic. The impact of a compressed representation for network packet data was considered. Neri suggests that this compression of features might result in an abstract representation that could allow better recognition performance or simplify modelling. Yairi *et al.* [221] employ clustering to automatically construct a system behaviour model in the form of a set of rules. Ye [224] used a data mining algorithm (Clustering and Classification Algorithm – Supervised (CCA-S) for learning the signature patterns of normal and intrusive activities, and then for classification of unseen data.

Barbara *et al.* [17, 18] applied the Audit Data Analysis and Mining (ADAM) as a testbed for studying the application of data mining approaches to intrusion detection. ADAM is an anomaly detection system composed of three modules: a pre-processing engine, which scans TCP/IP traffic and extracts information from the header of each connection; a mining engine that, in training mode, is capable of creating profiles for normal user and system behaviour and generating association rules, and, in detection mode, mining actual traffic for unanticipated associate rules; and a classification engine that determines whether unanticipated rules are different from the profile and hence to be considered as abnormal behaviour. Their later work [18] applies psuedo-Bayes estimators to enhance the system's ability to detect unknown attacks whilst reducing false positives.

Dokas *et al.* [56] present an approach based on building rare class prediction models for identifying known intrusions and their variations, together with anomaly detection schemes for detecting new attacks. Ertôz *et al.* [62] introduce the Minnesota Intrusion Detection System (MINDS), which is able to automatically detect attacks using a set of data mining techniques. Brugger [27] observed that intrusion detection systems have difficulties in detecting novel attacks without an unacceptable level of false positives. To address this, a group of data mining techniques (*e.g.*, statistical and clustering) were applied to data from off-line network connections, to supplement the real-time sensors. Different techniques where applied to the network connection data and a meta-classifier used to combine the results. This is said to enable activities such as low and slow scans, slowly proliferating worm attacks, and unusual activities of a user based on some new pattern of activity, to be handled, but which are otherwise prone to generating false positives.

Many of the applications of data mining found in the literature were used in com-

bination with other techniques. See, for example, Kumar and Spafford [132] on Coloured Petri Nets, Bridges and Vaughn [24] on fuzzy data mining and expert systems, and Soroush *et al.* [195] on ant colony based data miners for extracting classification rules.

3.12 Hybrid Approaches

Guerer *et al.* [84] describe a hybrid approach combining an expert system, a neural network and case based reasoning techniques for fault management. The expert system is employed to filter network alarms, which are then correlated with a neural network. The CBR system analyses the correlated alarms in order to identify the faults that have given rise to the alarms. A further CBR engine is then used to suggest a plan for fault correction. The results of executing this plan are subsequently added to the case base for later use. Although intended for fault management, the basic principles of event correlation could equally apply to intrusion detection. The authors considered the combination of techniques to provide superior flexibility and capability for tackling fault diagnosis in large networks.

Taniguchi *et al.* [204] combined a feedforward neural network, trained through supervised learning, with a Gaussian mixture model and Bayesian networks. The specific application was fraud detection based on billing records. The neural network was used to classify subscribers based on summary statics. The Gaussian mixture model essentially considered a probabilistic model of past and current subscriber behaviour to detect anomalies. The Bayesian networks approach was then employed to describe the statistics of a particular subscriber and that of various fraud scenarios. Sterritt and Bustard [200] use Bayesian belief networks to supplement a rule based system. Jiang and Cybenko [112] observe that attacks against networks and network resources often involve multiple steps and therefore most single event IDSs register high false alarm rates. They address this using Kalman filters and Bayesian estimation methods. They further proposed a Process Query System (PQS), that is able to scan and correlate distributed events according to the users' high-level process description. This approach was also used in the detection of Internet worms by Berk *et al.* [21]. Endler [60] also explored a combination of neural networks and statistical approaches, in this case likelihood analysis, for intrusion detection. The neural network implemented misuse detection, whilst the statistical approach facilitated anomaly detection. Zhang *et al.* [230] describe the Hierarchical Intrusion DEtection (HIDE) system for anomaly detection. The system is organised into layers, with each layer consisting of Intrusion Detection Agents monitoring the activities of host or network. A statistical component maintains a reference model of

network activities with which it compares reports received, and produces a 'stimulus vector' that is then fed into a neural network based classifier to determine whether it is anomalous.

Bridges and Vaughn [24] used a hybrid of fuzzy data mining and expert systems, where fuzzy data mining addresses anomaly based intrusion detection and misuse detection is achieved through conventional rule based techniques. The anomaly component searches for deviations from recorded patters of normal behaviour. Genetic algorithms are used to tune the fuzzy membership functions and select an appropriate feature set.

Marin *et al.* [155] present a hybrid technique that applies expert rules for the reduction of the dimensionality of the data, followed by initial clustering using a k-means algorithm, and subsequent refinement of the cluster locations using a competitive network, Learning Vector Quantisation (LVQ). They based their approach on the notion that legitimate users can be grouped according to the percentage of commands that they use within a given period. They use a nearest neighbour technique for classification, making it possible to train without anomalous data. Initial results showed an approximately 80% detection rate. Yang *et al.* [223] also consider the combination of cluster analysis and expert systems. Data is collected from audit trails or network traffic. The expert system is used to identify intrusion and generate an alarm if found or remove the data if not. After some preprocessing, cluster analysis is used for anomaly detection. Intrusion rules are then extracted, and the system is able to adapt prior unknown intrusions to known intrusions.

Pan *et al.* [175] use a combination of back propagation (BP) neural networks and the C4.5 decision tree algorithm to analyse TCPdump data for misuse detection. The general notion is to exploit the different classification capabilities of each technique for different attacks. Their experiments did confirm that each approach is better able to distinguish certain attack types than the other. Yu *et al.* [227] combined neural models, a back propagation network and a Cellular Neural Network (CNN), referring to the hybrid as BP/CNN. The approach is said to offer high detection rates whilst reducing false positives.

Ping *et al.* [180] present a distributed security framework for *ad hoc* networks, that combines immune systems and multi-agent methods, with an effort to be scalable, distributed and adaptable. The immune system element allows it to detect unknown attacks and provides adaptive learning abilities.

Chebrolu *et al.* [32] observe that some features of the data that systems use to detect intrusion patterns may be redundant. They identify input features that are considered important using Bayesian networks (BN) and Classification and Regression Trees (CART), as well as a hybrid. Empirical results suggest that selection of sig-

nificant input features is indeed important in the design of efficient and effective detectors.

Peddabachigari *et al.* [177] consider two hybrid approaches for modelling intrusion detection systems. The first combines decision trees (DT) and Support Vector Machines (SVM) to form a hierarchical model (DT-SVM). The second combines individual base classifiers with other machine learning approaches. The performances of these hybrid approaches were evaluated on the KDD-Cup99 data set, and showed that the hybrids offered improved accuracy.

Hanemann [86] describes a hybrid approach comprising two reasoning engines – one rule based, the other case based. The case based reasoning module (CBR) is employed as a backup in case of incorrect modelling. This work is considered in more detail in 4.3.

3.13 Other Approaches

This section considers a number of approaches that are best treated outside of the previous classifications. Some of the notions have already been mentioned in the preceding sections, although from a different perspective.

3.13.1 User Profiling

The technique of user profiling assumes that access to a system can be monitored and that use of system functionality is both available and recordable. This in turn means that a user's typical usage of a system can be deduced, represented in some way, and stored for later consideration. Subsequent use of the system by the same user can be compared against this 'profile' and atypical system use can be recognised. This notion is employed in many commercial systems to detect anomalous system use that might be representative of, say fraudulent activities. Examples are payroll systems that flag unusual payments and bank clearing houses that recognize so called normal financial behaviour against which abnormal transactions can be recognised. In the context of IDS applications, Hilas and Sahalos [97] describe a statistical learning approach that creates user profiles for intrusion detection in telecommunication networks. Their approach employs the usual assumption that user profiles are constant for the same user, and as such, comparisons could be made between the normal profile and test profiles to isolate attacks. This technique is dependent critically on access to the system being user based. In a sense this means that an aspect of event aggregation has already been performed by relating

different transactions to the same user. Systems that are not user centric would need to perform this aggregation stage and would not, therefore, be amenable to user profiling approaches.

Erbacher and Frincke [61, 69, 71] describe a method for analysing visually network and computer log information. This approach was based on the premise that each user's behaviour was fundamental to determining their intent and activity. It was also stated that visual analysis of the user's behaviour is adaptive and difficult to counteract. The logs are examined to identify and particular attack and, with visualisation, it is possible to examine an individual's activity as it is occurring and thereby determine immediately the attack type. It was stated that visualisation tools can aid in the reduction of false alarms and also detect previously unnoticed patterns of intrusions.

Coull *et al.* [42] employ a bioinformatics inspired paradigm where a modified sequence alignment algorithm is used to detect masquerade attacks in a network. Masquerade attacks occur when a user password and other login information is obtained by an intruder. The authors attempt to draw an analogy with the way sequence alignment is used to determine the similarity between two DNA or protein sequences in the area of bioinformatics. Their approach applies a semi-global alignment and a scoring system to measure similarity between a sequence of commands produced by a potential intruder and the signature of the user (*i.e.* the sequence of commands made by a legitimate user). Results suggest that this approach offers an effective combination of high detection rate with low false positive rate.

3.13.2 Application Profiling

Another profiling approach was presented by Wagner and Dean [213] based on an analysis of an application's normal behaviour in contrast to a user's normal profile. They were able to automatically derive a model of a system application's behaviour using static analysis of logs of the application behaviour. They proposed a layered model where a very high level or coarse view of application behaviour was determined down to a fine grained model of particular activity. The advantages and disadvantages of this approach are identical to those of the user profile technique.

Liao and Vemuri [145] use a *k*-Nearest Neighbour (*k*NN) algorithm to classify program behaviour as either normal or intrusive. Application behaviour is determined by the frequency and types of system calls. This technique was evaluated using a subset of the 1998 DARPPA audit data. The results found the technique to be effective in detecting intrusions, and achieving a low false positive count. However, it was highlighted that if an intrusion did not reveal any abnormalities in the frequency

of system calls, it will not be detectable by this approach.

Sekar *et al.* [189] also base their approach on application system calls and capture both short term and long term relationships among system calls, therefore performing more accurate detection. Finite State Automata (FSA) were used to represent these sequences. Primitive data is composed of the system call name and the program point from which the system call was made, given by the value of the program counter. Each distinct value of the program counter indicates a different state of the FSA, whilst the system calls correspond to the transitions. The number or length of the system call sequences was not limited. This approach was evaluated using file transfer protocol, web access, and network file system protocol data. The results suggest that the FSA learning algorithm converges quickly, has low memory and runtime overhead, and delivers a high detection rate and a low false positive rate.

Yin *et al.* [226] presented an approach for anomaly detection based on the Markov chain model and the linear prediction approach. Linear prediction is used to extract features from the system calls of privileged processes, and the Markov chain model is created based on these features. The key step in implementation was the selection of features that best described the user or system usage patterns so that non-intrusive activities would not be classified as anomalous. This method was evaluated with data obtained from the University of New Mexico and MIT Lincoln Labs and the results show that making comparisons between the probabilities of the intrusion and that of the normal application is useful for detecting intrusions when normal behaviours have been modelled.

Other work related to application profiling includes that of Ghosh *et al.* [77] using Elman neural networks and Kosoresow and Hofmeyr [128] using FSAs to model the temporal domain of application behaviour. In both cases the consequent IDS is claimed to be effective.

3.13.3 Information Retrieval

This technique is related to the idea of data mining but rather than using machine learning techniques to index data these are created by other techniques; see, for example, Bloom filters [22], where data is efficiently compressed and encoded for data transmission. Anderson and Khattak [9] presented an approach to intrusion detection, where audit trails are indexed by file inversion. This in effect completely replaces all datum items in a file with an index to the datum, thereby creating a file of index with a supplementary file of data rather than a data file with a supplementary index. Anderson and Khattack argue that this enables even the most obscure and, importantly, small attacks to be recognised. A clear disadvantage must be com-

putational, because of the need to invert the audit data files, which implies that the approach may be appropriate if used in an off-line environment, but has drawbacks if applied in real time.

3.13.4 Event Aggregation

Julisch [115] describes a system that turns alarm cascading into an asset by clustering alarm events and thereby finding the primitive event that initiated the problem. They argue that it is then more feasible to address the problem once the root cause is identified, and their approach was aimed at solving the problem of intrusion detection systems overloading their human operators by triggering many alarms each day. It was further contended that the alarms that occur persistently account for 90% of the alarms triggered by IDS applications. They further observe that these persistent root causes are problematic because they cause event cascades and can therefore distract the intrusion detection analyst from spotting more subtle attacks. Consequently they argue, alarms should be filtered by identifying, isolating and removing the most predominant and persistent root causes.

Tedesco and Aickelin [205] describe a method for detecting alert flooding attacks. Their approach was able to reduce alert throughput during the attacks by detecting when a flood is occurring and then adjusting to the threat and making the IDS more precise. They also demonstrated how compression techniques such as run length encoding (RLE), can be used to ensure that the important information about the attack is still logged. They were able to group the many related alerts that make up the attack, thus simplifying the attack analysis process.

Chapter 4

Event Correlation

4.1 Introduction

In this report, the misuse that is considered is that which can be effected over a set of data channels, *i.e.* it is network based. More precisely, it is understood that there will be in some sense *normal* traffic and that occasionally there could be *abnormal* traffic, and that the abnormal traffic is assumed to be misuse. This abnormal traffic could be composed of many different data types and could be constructed in many different ways. It is further possible that there could be many data streams within the network and that normal and abnormal traffic could occur over these streams. Another sophistication could occur within the definition of abnormal traffic. It is possible that traffic may only become abnormal when certain data appears in more than one data stream. That is to say, that normal traffic may occur in one stream but with particular data appearing in another stream this traffic could then become abnormal. This leads to the idea of abnormal traffic occurring within multi stream data transmissions.

To understand this situation one must recognise the data streams within a system and the type of data that they could carry. It is then necessary to characterise what constitutes abnormal or normal traffic. For misuse detection, normal traffic is defined as the absence of abnormal traffic, whilst for anomaly detection, abnormal traffic is defined as the absence of normal traffic. Hence, it is not necessary to define abnormal and normal traffic separately. From these basic notions it is feasible to proceed to an understanding of the idea of events within a data stream. These are dependent in their turn on the idea of tokens and how these can be represented. Abnormal traffic then becomes the existence of a set of events over a set of data streams within a specified time frame. Finally, the process of event correlation be-

comes one of determining how a machine[1] can be fed information that will allow it to make a determination regarding the normality or otherwise of the data of a set of streams within a specified time frame. Passing information to the machine will require discussion of data representations. It is also possible that this data could be structured and it may then be necessary to prescribe a representation that includes the essential features of the data content together with its structural context. An intelligent machine that can perform an abnormality determination will need to be trained and attendant considerations must be given to the construction and content of a training set. The performance of the machine obtained from such a set will be dependent on the distribution of *normal* and *abnormal* data within the training set.

Broadly, then, event correlation is the process of identifying significant system events by recognising combinations of fundamental or more primitive events. Furthermore, event correlation should enable the essential characteristics of such combinations to be aggregated into a single compound representative event. Eventually these aggregations should be recognisable as system misuse or as some form of attack upon the system.

Event correlation takes place in the context of a computer system, which may be as simple as a single application running on a single computer or could be as complex as a multi-functional distributed application running across a network of connected computers. In the case of a distributed system the atomic events could originate from many different functional modules executing on any of the host computers, or from any part of the network infrastructure. Typically, these fundamental events will be the result of monitor programs that deliver output streams consisting of telemetric data. This data can be interpreted in terms of atomic events and consequently fed in some way to the event correlation system. Alternatively, it is feasible that sub-functions of the event correlation system execute on each node within the distributed system. This configuration implies the consequent problem of coordinating the separate event correlators, which in turn has an implication for distributing the output of these correlators across all the nodes of the system. This, incidentally, is the motivation behind approaches based on mobile agents.

There are certain features and characteristics of the event correlation process that are in a sense generic and not related to particular system hardware topologies or application implementations. This chapter begins with a description of the intrinsic nature of event correlation. This involves a distillation and summary of much of the terminology that is prevalent in the literature. Further it involves erecting an abstract description of a the event correlation process. The importance of this becomes clear when proceeding to discuss various solutions to the event correlation problem.

[1] Which, for our purposes, is taken to mean some machine based on AI techniques.

A generic description allows an implementation to be considered in the context of this generic model. This in turn allows seemingly quite different systems to be indirectly compared and contrasted with each other by comparing each with a different abstract or generic model.

It is also important in the context of this research to understand the techniques that were employed to construct the correlators. This is so because the techniques will imbue the correlators with characteristics that are inherent within the technique. For example, using artificial neural networks demands that an effective representation is created for a compound event. Rule based systems do not necessarily need such a representation because the compound event is understood atomically, *i.e.* as a collection of parts. In the case of the neural network it would usually be necessary to have a holistic view of a compound event, perhaps to compare against known malicious events with some form of fuzzy comparison.

Finally, a taxonomy of event correlation solutions obtained from the research literature is derived, where the particular generic characteristics and the techniques employed for the implementation of each are known. A brief examination of a number of correlators, either commercial or well known open source implementations, is included in Appendix A.

4.2 General Considerations

This section explores some general issues relating to event correlation. Consider a networked system where a fault has occurred. Further, assume that this system is monitored and that events within the system are alerted in some fashion. In many cases event alerts will represent so called normal behaviour. In other cases they may clearly indicate a bad failure of the system. In yet other situations the events may only take significance when they occur in the context of further system events, *i.e.* they are significant when correlated with other events. If system alerts are thought of as either a single event or, as described, possibly a set of events, then it is understood that there is a clear issue of fault or alert recognition.

It is also feasible that some faults can lead to alarm cascades or event showers. Clearly a sensible approach to the analysis of such a collection of alerts is to correlate those that are related to the same underlying cause. Then, if possible it would be desirable to replace the multitude of events with a single alert that contains the essential characteristics of the underlying problem. This of course comes with the attendant issue of finding a suitable representation that signifies the fundamental characteristics of the problem. This is an issue of event aggregation.

Finally, the seriousness of the problem must be considered, and what actions must be taken for its rectification. Thus, tackling the general event correlation problem is really about answering the following questions:

- What constitutes an alert?
- How is it recognized?
- Does it consist of many correlated events relating to the same problem?
- How can these correlated alerts be aggregated?
- What actions must be taken as a consequence?

So, there are clearly separate issues of recognition, correlation, representation, aggregation and action.

At this stage some initial definitions are made that are significant within the subsequent discussion. The necessity to find a primitive event (if possible) is referred to as finding a *root cause*. In this case there is an implicit assumption that the events form a causally connected tree structure and the root of that tree is sought. In reality, of course, there could be more than one root cause, and then the problem is one of dealing with a mathematical lattice structure. These ideas can be important when formal methods approaches are used to create event correlators. Briefly, a root cause exists when a fault or problem occurs in a distributed networked system that can give rise to many consequential alarms that also apparently indicate a system fault. The root cause refers to the original problem that is the cause of the consequent dependent problems. For example, a network channel may become inoperative. As a consequence an FTP session may time out which is flagged as a fault in a system journal. There may be many other faults raised as a consequence of the failed channel. However, the root cause is the failed channel and not any of the subsequent dependent problems. Brown *et al.* [26] observe

> One of the most significant challenges in managing modern enterprise systems lies in the area of problem determination – detecting system problems, isolating their *root causes*, and identifying proper repair procedures.

There is also the consequent idea of *root cause analysis*, which refers to the task of determining the root cause from a set of apparently related problems that occur within a distributed system. Note that there may be more than one root cause, which will happen when the set of problems contains some that are independent.

This distinction is clarified at this stage because there seems to be a tendency in the research literature to refer sometimes to root cause analysis and sometimes to event correlation. Here, the clear distinction is made that root cause analysis is a part of event correlation. For example, event correlation also includes the task of event aggregation which forms no part of root cause analysis.

4.3 Previous Research and Solutions

In the following sections, important work from the research literature is highlighted. Sources have been organised chronologically then thematically, charting the progression of various pertinent lines of research.

1995

Hasan *et al.* [94] argue that the power and robustness of event correlation systems varies considerably and that a formal specification of the system is necessary to enable effective comparisons to be performed. They present a formal method based on a combination of causal and temporal information within the context of the topology of the monitored system. They contend that the importance of Event Correlation Systems (ECS) have triggered the development of a number of systems and refer to Sheers [190], Kliger *et al.* [124], Jakobson and Weissman [109], and Hasan [93]. They claim that their formal system is compelling because:

- It is sufficiently general to encompass many of the possible architectures and approaches to ECS design.
- It is scalable. For example, temporal relations are incorporated by a natural extension of the language.
- It is associated with a proof system that enables very strong characterisations of system behaviour.
- The event relationships described within the language are effectively computable.

2000

Lo *et al.* [149] proposed two event correlation approaches that are based on the

coding scheme presented by Yemini *et al.* [225], and these were applied for handling the identification of faults in communication networks. They described the causality graph model which is a technique used to describe the cause and effect relationships between network events. In their approach, the notation scheme involved transformation of the information in the correlation graph into a set of codes; one code per problem. Next is the event correlation stage which involves seeking the problems that cause the observed symptoms. In the first of the two schemes they proposed, the event correlation is achieved in the selection process and followed by the identification process, while in the second scheme it is done in two phases, the codebook selection phase and the decoding phase. Simulation results show that these schemes are able to identify multiple problems at a single cycle but that they are also susceptible to noise.

Asaf Adi [3] describes a system called Amit that performs event correlation and aggregation. They position this within applications they refer to as reactive, which they define as applications that respond to events by triggering alerts. They extend the domain of applicability of alert systems to include any application that reacts to events. For example e-commerce, system management and command and control applications. They argue that there is a requirement for middleware systems that implement so called 'push technology'. This is technology that has the ability to transfer information as a consequence of an event instance.

2001

Krugel *et al.* [129] presented a specification language for defining intrusions as distributed patterns. The system models intrusions as patterns of events that occur at different hosts. It consists of collaborating sensors arranged at differing positions in the network. A peer-to-peer algorithm, known as Quicksand, for detecting these patterns of events was also presented. The different hosts in the system run a Quicksand sensor, which is made up of several local intrusion detection units (LIDU), which are responsible for gathering events, and an event correlation component that receives the streams of events from the LIDU. In addition, there is also a control unit that the system administrator makes use of for system configuration. The system was evaluated using fault tolerance and scalability as metrics and was found to be satisfactory. Fault tolerance indicates the percentage of distributed patterns that can still be detected, whilst scalability is described by both the total network traffic between all nodes and total network traffic at a single node.

Al-Shaer [6] presented a group management framework for event monitoring based on the IP multicast standard for distributed correlation. The multicast is able to dynamically reconfigure group structures and membership assignments at run-time

depending on the requirements of the event correlation that allows for optimal delivery of multicast messages between entities. It was added that this framework provides techniques for solving agent state synchronisation and the bootstrap problems in distributed event monitoring. Some advantages of this approach include *inter alia* the support for dynamic and scalable monitoring of information dissemination to agents, the provision of a fine-grain group communication that enabled the agents to disseminate monitoring information based on the event correlation tasks, and the support for an agent synchronisation protocol that ensures state consistency of the agents during group communication. However, a limitation of this approach is the possibility of creating too many groups that may consume much of the resources.

Albaghdadi [153] uses object oriented concepts to encapsulate event functionality within a class structure, and also leverages object orientation inheritance capability to provide scalability. Graph theory and coding techniques are utilized. To address the area of event correlation, Albaghdadi points out that AI techniques have been used by Appleby *et al.* [11], Jakobson *et al.* [108], Sheers [190], Wietgrefe *et al.* [218], and Weiss *et al.* [216]. Causality graphs are used by Kliger *et al.* [124] and Yemini *et al.* [225]. In particular this research is an extension and development of the research reported by Yemini. Albaghdadi reports that Hasan [94] utilised graph theory and propositional relations and that Finite State Machines (FSM) and probabilistic FSMs were used by Wang *et al.* [214]. Albaghdadi assumes that an event has the following structure:

- Reason for the event. This could be a type or an error code and may have additional detailed information.
- The source of the event. In a distributed system this is the network node that logs this event.
- The date and time of the event.

Events are considered to be primitive in the sense that they can be thought of as an alphabet. This leads to the idea of event words, which is defined as a concatenation of events without regard to arrival sequence. For example, if $\{e_1e_2e_3\cdots\}$ is a set of possible events then $e_1e_2e_3$ and $e_2e_3e_1$ are equivalent words. Further, multiplicities are ignored and only single event instances occur in words. Let Σ represent the entire spectrum of possible events and therefore the complete alphabet. Also Σ^k represents the set of words of length $|w| = k$. Then

$$\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \Sigma^3 \cdots = \bigcup_{k=1}^{\infty} \Sigma^k$$

In this case Σ^+ is the set of words that might be constructed from one or more events of Σ and is the maximal set of words from the system under examination. An event pattern can be composed of many events which may be contributed by many different objects throughout the system. In this context an object is something autonomous that can give rise to events. It is made clear that there is a many–to–many relationship between patterns and events.

The correlation procedure is composed of two processes.

1. **Local correlation** where an object generates events that are caused by changes in the state of the object and possibly as a result of external stimuli. This maps very neatly to an object oriented paradigm where the objects are class instantiations which exchange messages with other objects.

2. **Global Correlation** which combines the outcomes of local correlation to obtain a higher level view of system behaviour. This looks a little like a form of aggregation.

The correlation process is directly related to finding patterns that exist given the occurrence of a set of events. Since the relationship is many–many it is quite possible that many patterns may be identified given a certain event set occurrence. It is reported that neural networks have been used to perform this procedure. Reference is made to the SNNS – Stutgart Neural Network Simulator with an approach along the lines described by Yemini *et al.* [225]. A correlation matrix can be created with the columns representing possible patterns and the rows representing events. A '1' in the element c_{ij} means that the i^{th} event is contained in the j^{th} pattern. This gives a set of binary valued vectors that can be learnt, presumably by an MLP). In order to compensate for a noisy environment, a match between a recognised vector and the rows of the correlation matrix is effected by a scalar product metric used between normalised vectors. This technique is then discussed in the contexts of local and global correlation and also within the context of a temporally evolving system.

For local correlation, the topology field within the event is used to identify the event source. This event is pushed onto the event queue associated with the particular source. In object oriented terms each source becomes an object and the event queue is associated with this object. Each queue has a system dependent length and if necessary old events are dropped from the event queue. Initially this new event is marked as uncorrelated. For global correlation the authors introduce the concept of a correlation zone. This is an aggregation of objects or event sources. Zones cannot raise events and are not the source of events. However they are recursive and allow for hierarchies of zones to be constructed. Rules can be associated with zones

enabling a comprehensive rule based system to be erected. In conclusion, the object oriented class concept is used to encapsulated network node event processing and, further, to naturally model correlation zones as recursive object instances. Simple vector normalised scalar products are used as similarity measures.

Appleby *et al.* [11] report on collaborative research between Appleby, at IBM's T. J. Watson Research Center, and Steinder at the University of Delaware. In this paper, they describe the 'Yemanja' event correlation engine. This is based on a model of the computer system and employs a rule based correlation kernel. The authors argue that commercial correlation engines (circa 2001) suffer from the following disadvantages:

- They possess hard coded network connectivity information.
- Entity dependencies are also hard coded.
- The system model is traversed from all event sources. By this they mean there is not a central organised information base.
- Inability to integrate device attributes with events, which makes root cause analysis difficult.
- Failure to effectively correlate events over different time scales.

The distributed networked application is composed of entities, which are devices or possibly logical/conceptual entities. Each entity has a problem behaviour model which contains problem scenarios. In many respects these equate in other systems to rules. Scenarios consume events in the sense that events are recognised as forming part of a scenario. The scenarios publish further events, which can be consumed (the authors' terminology) by other scenarios. Entities can be grouped hierarchically which admits the concept of an event level. Entities correlate events at a specific level. In general, producers are unaware of which entities consume the generated events. This model is intended to allow greater flexibility and to facilitate the addition of new entities to the overall model. In conclusion, the Yemanja product was, at the time of publication, in use in a commercial application. The authors conclude that the consumer/producer design of the entity classes, the hierarchic design of the inter-entity relationships and the independence of the rule based analysis from the system topological information, makes the system flexible and fast.

Steinder and Sethi [199] are a source of many recent publications. This paper describes fault localisation and gives a selected taxonomy of techniques that are used in the design and implementation of algorithms. Fault localisation is a process of isolating and recognising faults that cause observable malfunctions of managed

systems. The word 'managed' in this context means systems where normal and abnormal service functionality is monitored. Associated with fault localisation is a particular vocabulary. They define

- An event as an exceptional condition within a managed system.
- Faults or Root Problems as basic events that are immediately recognisable and can be directly managed.
- An Error is a consequence of a fault and is defined as a difference between an actual condition and an expected theoretical condition. A condition is a value of some measurable quantity.
- Symptoms or Alarms are external consequences of failures.

This vocabulary, which is consistent with those terms adopted by the software engineering community in respect of testing and reliability, is then employed to describe how failures can cause fault cascades and collections of related and connected faults. Consequently the need for fault or event correlation and aggregation.

2002

Vaarandi [209] presented a lightweight and open source tool for rule based event correlation called the Simple Event Correlator (SEC). This is a platform independent tool said to be able to deal with the complexity and size issues of other event correlators. SEC is configured via a number of files, each of which contains one or more rules. It produces output events by executing shell commands specified by the user. A wide range of correlation rule types are supported, as previously described in section 3.3. Morin [167] argues that event correlation can only be performed effectively when there is a comprehensive model of the system within which the correlation is to be performed. In particular the authors contend that the contextual model should contain information relating to:

- System characteristics, which combine the topology of the system with the functionality of the nodes within the network. Further information is encoded that determines allowable message paths and allowable message types exchangeable between nodes.
- System vulnerabilities, specifying the types of attack of which the system must be particularly aware.

- Monitoring Tools, which are used and are available to collect the information upon which event correlation can be performed.
- The Events, which are the lowest level event instances.

Further, they propose a formal system that can be used to model the rules that determine when events are malicious or anomalous. Rather like a formal logical system the context provides the semantic framework within which semantic interpretation can be performed whereas the formal rules provide the syntactic rules for the formal system. The formal language is a derivative of Vigna's model [212]. The system is tested against three types of event correlation.

- Aggregation of alerts that refer to the same host or system node.
- Identification of nodes or hosts that are vulnerable to an attack.
- Reduction of false positives by analysing the reaction of the security tools that could possibly react to a particular attack.

The authors emphasise that this system (referred to as an M2D2 model) is an attempt to provide responses to the three issues of

- The large number of alerts that are provided in any meaningfully sized system.
- The quality of these alerts.
- The consequent diagnosis provided to the system managers.

Their approach integrates details of the environment with the description or rules determining the anomalous event descriptions and uses a formal language to specify those rules. Three examples are given that show how the formal language can be used to perform effective event correlation, or, perhaps more precisely, event aggregation.

2003

Benferhat *et al.* [20] contend that IDS systems are expected to recognise simple single event attacks as well as complex multiple event attacks. As an example of the former they describe the so called *ping of death* attack that tries to effect a denial of service by sending an excessively long IP packet. Alternatively, the *Mitnick* attack is a multiple event attack that involves the following stages.

- First, an intruder floods the host computer H.
- Next, the intruder sends spoofed SYN messages to a server S which claim to have originated from H. When S send a SYN–ACK message to H then H would normally send a RESET message to close the connection. But this does not happen because H is flooded. In this case the intruder sends an ACK message to open a TCP connection with S. In effect H is kept busy while the intruder borrows H's IP address for a set of spoofed messages to S.
- Now that a connection is opened with S the intruder can attempt further ingress by, for example, trying an *rlogin*.

The authors argue that there are so many possible multiple event scenarios that it is very difficult for an event correlation engine to make a judgement regarding which, if any, is actually in progress. In order to simplify this congestion the authors propose a ranking system whereby the scenarios are given a weighting that is intended to reflect the probability that a particular scenario will occur. In conclusion, the authors argue that event correlation is fundamentally a rule based process.

Morin and Debar [166] propose a multi-alarm misuse correlation component that is based on the Chronicles formalism. Chronicles are temporal models that represent potential explanations of an observed system; they offer a high level declarative language as well as a recognition system for monitoring dynamic systems, whilst formalism enhances the quality of the diagnosis provided by the system as well as allowing for reduction of the number of alarms sent to the operator. Chronicles comprise of a set of time points, a set of temporal constraints between time points, a set of events models, a set of assertion models, and a set of external actions that will be performed when a chronicle is recognised. The models proposed in this paper were not tested on live data but only on alarm logs. However it was noted that chronicles were primarily designed for diagnosing failures in telecommunication networks by analysing alarms issued by equipments.

2004

Jiang and Cybenko [112] analysed the application of control methods such as Kalman Filters and estimation methods such as Bayesian estimation in the correlation of distributed events for network security. It was remarked that attacks against networks and network resources often involve multiple steps and therefore most single event-based traditional Intrusion Detection Systems register high false alarm rates. They further proposed a Process Query System (PQS), that is able to scan

and correlate distributed events according to the user's high level process description. This approach was also used in the detection of Internet worms by Berk *et al.* [21].

Totel *et al.* [206] presented a language driven signature based event correlation system for intrusion detection. The language known as ADeLe is used to defining correlation properties, and aims at combining all of the knowledge available for an attack in one high level description. The ADeLe language correlation process makes use of filters that describes the events or alerts that must be used as elementary elements in the correlation process. Correlation operators are used to define the signature of the attacks. These are: Sequence, which facilitates the description of a sequence of events; OneAmong, which defines a separation between the elements during each detection; NonOrdered, which describes a signature in which all elements are recognised without any order constraint; Without, which facilitates the description of the way a negative signature is able to remove positive signature detection when a negative signature is detected; and Repeat, which permits the description of a repetition of an event or an expression. As with other sets of correlation rules, such as those of SEC [209], a need for flexibility and expressiveness is apparent.

Jakobson *et al.* [107] presented a survey of event based situation analysis. They argue for the integration of an event correlation system and a situation awareness system, and proposed an architecture combining rule based spatio-temporal event correlation and case based reasoning for understanding and managing events.

Hanemann and Schmitz [91, 92] present a service oriented event correlation technique for the correlation of reports from the system and/or users. It was identified that different reports could have the same course and therefore reports could be linked together and the process of analysing the reports performed once. The Munich Network Management (MNM) Service Model was used. This is a generic service management model proposed by the MNM team, used for retrieving appropriate modelling of the necessary correlation information. The model distinguished between client side and provider side roles, and consists of two main views, the service view and the realisation view. This approach was evaluated using the e-Mail service offered by a supercomputing centre.

2005

Hanemann [91, 92, 88] discusses the context of service level provision and the consequences of failure to provide service levels in accordance with Service Level Agreements (SLA) and Quality–of–Service (QoS) expectations. This paper de-

scribes a method for automatically determining the impact of service failures. Service failures are defined to be of short, medium or long term duration. In each case, different actions must be taken as a consequence of the occurrence of such a failure. This paper presents a framework and workflow within which and against which appropriate actions corresponding to service events and failures can be determined. This is in contrast to the more traditional human intervention technique. In summary, this paper deals with the question of what to do once a root cause has been identified, which itself is the result of event correlation analysis. Hanemann [87] again emphasises the difference between what is defined as *resource* events and *service* events. The former mainly occur as a result of hardware or vendor firmware code. For example, a link is up or down, a router has failed or not, or perhaps some form of authentication failure related to the lower levels of the network hierarchy. Service level events include, for example, customer reported problems. These occur at a much higher level than the network hierarchy. Even in some ways above the application layer of the ISO model. Such events need to be correlated with the underlying resource events. The root cause event will almost certainly come from one of the resource events. This paper therefore describes a framework that enables service and resource events to be aggregated and correlated.

Hanemann [89, 90, 86] later proposes a rule based reasoning engine to manage event correlation. In addition, a case based reasoning engine is invoked to recognise alarms and events that are not covered within the rule based system. The case based system is then used to update the rule based system. Reference is made to the HP Openview system, see section A.3. A distinction is drawn between network and systems management faults or events, and service based events. The former are connected intrinsically with the hardware and firmware of the distributed applications and in these cases can even be specified as failure conditions attached to the hardware or software specifications. Service based events on the other hand originate from the needs of the customer and are usually formalised within a SLA. Often the customer may cause a service event by pointing out that the supplied service fails a QoS guarantee. This paper uses an example scenario to demonstrate the current position regarding service fault diagnosis. It further examines how event correlation underpins this fault diagnosis. The hybrid architecture proposed in this paper is described and using the example it is shown how this facilitates fault diagnosis through event correlation. The service scenario is based around the Leibniz Supercomputing Center (LRZ) which acts as an Internet Service Provider (ISP). The particular service used as an example is the e-Mail service provided to the staff and students of Munich universities. Of course, this service is dependent on a hierarchy of underlying services such as domain name servers, routers, mail transport agents *etc*. The fault management is entirely user centric, where events are related to help desk calls raised by dissatisfied users. HP Openview (section A.3) is available to

perform event correlation and to identify root causes. No automated system exists to connect related events and correlate customer reports to these events. This paper relates the qualities of different event correlation approaches to some erected criteria that are considered important for service based systems. The criteria are:

- **Maintainability.** Service provision is an extremely dynamic and volatile application. Any event correlation system must be easily updatable or amendable to new conditions and circumstances.
- **Modelling.** The relationships inherent in service provision applications can be extremely complex because there is a deep hierarchy from inter service dependencies to network level dependencies. Any model must be sufficiently flexible to enable this complexity to be encompassed.
- **Robustness.** The complexity of service related applications also suggests that the event correlation model may be incomplete. In this case the event correlation system must be able to cope with unknown or previously unseen event circumstances.
- **Performance.** Although service type events do not occur with the potential frequency of, for example, network based events the performance must still be considered since network type events may become part of the event model and consequently will need to be accommodated.

Notice that these criteria may not be appropriate for all event correlation scenarios, being specifically targeted at service based systems. However, there may be some overlap of characteristics. Hanemann offers the following table as a guide to the usability of event correlation techniques against the above criteria.

	Maintainability	Modeling	Robustness	Performance
MBR	-2	+2	0	0
RBR	-1	+1	-1	+1
CBook	-1	-1	0	+1
CBR	+1	+1	+1	-1

MBR refers to model based reasoning, RBR is rule based reasoning, CBR is case based reasoning and CBook refers to CodeBook systems. Of course, as is argued in this paper, advantage may be gained by using combined or hybrid approaches. For example RBR plus CBR has been proposed by Jakobson *et al.* [107] to manage highly dynamic systems, where in this instance the RBR and CBR run in parallel. As a consequence of the previous analysis, a hybrid architecture is proposed in this paper. The components that comprise the service event correlator are:

- The ServiceMIB which is a manually created database of service and resource dependencies.
- The RBR which takes rules automatically generated from the ServiceMIB. Events input to the RBR are matched against known rules.
- The CBR, a case based reasoner that attempts continuously to match the events to a known case. If this is successful and the result is in agreement with the RBR then the root cause is found with some justification. Previous cases and their consequent root causes are configured manually.

Observe that the events that are input to the service event correlator are already aggregated and correlated externally in the sense that some degree of service event correlation is performed by the external service management software. The interesting situation is when the RBR fails to match and the event is forwarded to the CBR. In some situations an adaptation to an existing case will match the event. This adaptation together with the current ServiceMIB is then used to update the ServiceMIB which in turn becomes the source for an update of the RBR.

4.4 Techniques

A wide variety of techniques have been employed for the solution of the event correlation problem [52, 51, 142, 66]. A frequently employed approach to an integrated, intelligent fault diagnosis is by means of rule based systems [146, 147, 153]. These systems are often limited to simplified rules for event correlation and are inclined to be difficult to scale to large systems [66]. Connectionist and other AI techniques [190] have been employed in an attempt to obtain self learning of the complex correlation rules, but again these seem to suffer from problems of scalability. Model based paradigms have been proposed [124, 225, 73, 216, 66] that it is argued can overcome many of the problems inherent in scaling to large systems. Steinder and Sethi [199] observe that event correlation techniques have been derived from a selection of computer science paradigms. They list, among others, AI, graph theory, information theory and automata theory. They offer a classification hierarchy based on Katker and Paterok's [118] analysis of the most often employed techniques:

- Model based reasoning tools.
- Fault propagation models.

- Model traversing techniques.
- Case based reasoning tools.

Appleby *et al.* [11] argue that the following six research areas are significant within the field of event correlation. In particular they make a distinction between model based systems and model traversing systems. In many cases this distinction is not made apparent in other reported research.

- Rule Based.
- Model Based Reasoning Systems.
- Model Traversing Systems.
- Case Based.
- Fault Propagation Models.
- Code Book Approach.

Rule based systems [148, 219] are synonymous with expert systems. Model based reasoning [108, 172] and model traversing systems [114] are slightly different aspects of a similar idea. Both rely on a model of the underlying system. Model based techniques relate the events to the topology of the system which is encoded within the model whereas model traversing create a model of the relationships between the topological entities of the system. Event correlation is performed via correspondence with the model's relationships. Case based reasoning is heavily promoted by Lewis [143]. Fault propagation models, Hasan *et al.* [94], are similar to model traversing but in this case the model is a highly abstract view of a combination of the topology and the causal relationships of the system entities. Codebooks, Yemini *et al.* [225], are essentially look up tables.

From these observations techniques are classified into the taxonometric tree depicted in Figure 4.1.

4.4.1 AI Techniques

There are at least two significant models that come under the heading of AI techniques. The first are those where intelligent behaviour is obtained by actual configuration or programming of the machine that is to perform the behaviour. This

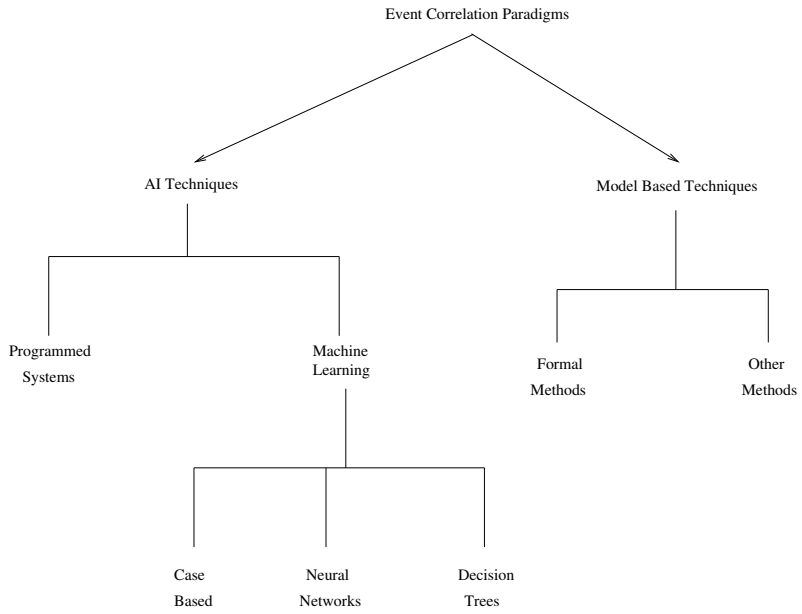

Figure 4.1: Taxonomy of detection mechanisms

approach assumes that the behaviour of the machine is predetermined by the subtlety and extent of the programming from which it was constructed. In other words, it depends on how good the expert was that built the AI. The second covers those models where the AI obtains its so called intelligence through a process of learning. Such paradigms cover all of the procedures where a machine is trained to perform some function on some form of exemplar set of data, usually referred to as the training set, and is then expected to perform the same function on previously unseen or novel data. The function performed is often some type of pattern recognition but is not restricted to this action. Expert systems most definitely adhere to preprogrammed paradigm, whilst neural networks employ the machine learning model.

1. **Programmed systems.** This is a broad and now somewhat ill defined area but effectively the knowledge about the system and the valid inferences that can be adduced from this are preprogrammed into an abstract model. There are many specific techniques that are derived from this broad approach.

   (a) **Expert Systems** The knowledge about the system and the logical relationships that hold intrinsically with this knowledge are usually obtained from experts in the field of enquiry. The knowledge inherent in an expert system is deemed to be either surface knowledge which is derived from experience, or deep knowledge which implies an understanding of the underlying principles of the system.

(b) **Rule Based Systems** Approaches that rely on surface experiential knowledge are referred to as rule based systems. These systems employ a forward chaining inferential system, *i.e.* they proceed from accepted knowledge by applying selected rules to obtain inferred or deduced knowledge. Rule based systems do not require complex understanding of a system and work well for small systems. Challenges encountered with rule based systems include their difficulty in coping with unseen problems. This is because they are often hard coded or preconfigured and therefore incapable of adding to their experiential knowledge base. More importantly they do not have the ability to generalise from known situations.

(c) **Model Based Approaches** This paradigm is predicated on the existence of a model of the underlying system [108]. The model makes available information on the topology, both physical and theoretical, of the system and allows failure conditions in different parts of the system to be correlated. Due to the deep knowledge of the system these approaches have the ability to solve novel problems and with appropriate organisation the model is adaptable and expandable.

(d) **Code Book Techniques** These techniques use information theory [154] and were first employed by Yemini *et al.* [225] in the SMARTS InCharge system. Fault propagation patterns are represented by a codebook [124, 26] where, for each fault a distinguishable code is created. In a deterministic model a code can be a sequence with each element taken from the set $\{0, 1\}$. Codes are generated dependent upon a system model and a fault model. A causality graph is used to generate the propagation model from which an intermediate code is generated. The graph is reduced by pruning cycles, removing unobservable events and indirect symptoms that cause other events that are already in the graph and do not therefore add any additional information. After reduction the graph contains only direct cause effect relationships between fault events and symptoms. The graph is then converted to a matrix representation where the rows are indexed by symptoms and the columns are indexed by fault events. The element p_{ij} contains the probability that fault event e_j causes symptom s_i. In a deterministic model $p_{ij} \in \{0, 1\}$. This matrix is referred to as the correlation matrix.

2. **Machine Learning**

(a) **Case Based Reasoning** These systems make their determinations based on previous examples of similar situations [143], referred to as cases.

The problem is to find a similar case to the one currently under consideration and the central issue is to define in this context what is meant by similarity. Often this will be related to how the situations or cases are represented and will sometimes be a simple Euclidean distance metric. Case based systems are resilient to changes in the topology of the system and by their nature are easily adapted. The challenge in deploying them in event correlation systems is the representation of temporal relationships between an arbitrary number of events that form a case. Some work has been done on adapting CBR to deal with sequences of concurrent events from multiple sources [157], and CBR has also been employed as part of hybrid correlation engines [89, 90, 86] .

(b) **Neural Networks** Neural networks are a machine learning paradigm modelled on the biological behaviour of the brain and the neurons within the brain [95]. The capability of a neural network is not computationally limited and its functionality and behavioural characteristics are determined by the interconnections of the neurons, its topology, and the weights associated with these connections. Usually the topology is fixed and the weights are modifiable. The weights in a sense form the knowledge container. The attraction of neural architectures is that they do not function on the basis of a formal representation of the problem, but instead derive their own internal representation by distilling the essence of the problem from the training set. This therefore enables neural networks to generalise to unseen data, assuming that the training set is sufficiently representative of the domain. Cannady [28] and Wietgrefe *et al.* [218] discuss the use of artificial neural networks and, in particular, feedforward networks for misuse detection. This approach does not facilitate event correlation for events widely distributed in time. Rather, a feedforward neural network with backpropagation training is used as a pattern recogniser for signature recognition. Neural networks are classed as an adaptive system and do not suffer the problems of expert systems where rules are precoded and difficult to change. A neural network can at least learn from a training data set what it is supposed to know. One drawback, however, is that this knowledge is not evident in the configuration of the network. Their functioning is taken on trust once they have been trained and tested.

(c) **Decision Trees** Decision trees are generally thought of as a form of machine learning. Essentially, a tree is created, the use of which enables decisions about a particular data set to be performed. Traditionally these decisions might be an allocation of the data into a particular class. Another possibility is to predict the value of some variable depending on

the predicted values within a given data set. Learning for decision trees involves creating the tree itself. Use of the tree is simply a matter of traversing the tree, making a decision at each node point depending on the value of the attribute of a specified data item. Classically the algorithm for training the tree is called the ID.3 algorithm and is based on the principle of entropy maximization. This algorithm is well described by Mitchell [164] and Luger [152]. This paradigm was used by Pan *et al.* [175] as a hybrid approach combining neural networks and decision trees within the intrusion detection process.

4.4.2 Model Traversing Techniques

The problem of event correlation can be considered through the use of abstract models [74]. The idea is that a thorough understanding of the topological and functional properties of the computer system facilitates an understanding of the types of intrusion that could be attempted against this system. Typically, this approach expects there to be significant topological and/or functional dependencies. In this solution an abstract simulation model is built, and used to predict the profile of alarm propagation and dissemination throughout the system. By comparing actual profiles with the model it is possible to detect the root cause of the alarm. In many cases these models are constructed from formalized mathematical systems [73] and indeed this must largely be the case in the model based approach. Logical connections between distributed entities within the system are represented by relationships in the formal model. When a fault occurs, these relations are inspected to establish impacted entities, which may themselves raise further alarms. By this technique a form of event or alarm correlation is performed. These techniques are described by Gruschke [81, 82] and Katker and Paterok [118]. The algorithms necessary to support and implement model based systems must often be distributed throughout the networked system [158]. One approach by Fabre *et al.* [66] employs a domain oriented design where a supervisor controls each domain and is responsible for creating a coherent local view, in the sense of agreed interface protocols, of its own domain. Flagged events are then distributed throughout the system.

1. **Formal Methods** These techniques include formal grammar systems that use the fundamentally recursive nature of grammar expressions to represent complex hierarchic event dependencies. Other techniques use formal logic systems to specify the rules in a complex model based system. This has been attempted in the context of cellular phone networks using extended logic programming by Frohlich *et al.* [73]. See also Eskin *et al.* [64], where a geometric framework for unsupervised anomaly detection is described. Vert

et al. [210] begin the process of describing intrusion detection mathematically. This centres around a mathematical description of the event correlation mechanism. Jakobson *et al.* [107] describe an architecture for reasoning about event based dynamic situations. Cuppens [44] has actually created a formal language he calls LAMBDA used to create system models for event correlation. The name is deliberately reminiscent of the rather more abstract invention of Lambda Calculus [19].

2. **Other Methods** Zerkle *et al.* [228] consider a data mining approach. Eskin [63, 64] considers the probability distributions of anomalous events occurring over noisy communication channels. Gruschke [81, 82] uses dependency graphs as a tool for event correlation specification and analysis. Frincke *et al.* [70, 69, 71] realised that event correlation systems could be more effectively constructed if the event dependencies and underlying system models could be efficiently visualized. This developed into something of a side issue but nevertheless stands as a significant tool during the development and subsequent verification of IDS applications.

4.4.3 Hybrid Systems

It is difficult to find an AI technique that is applicable to all of the issues raised by event correlation engine design. Where, for example, a rule based system would function well once the supplied events have been sequenced and their essential features have been extracted it is not a good choice of technique for actually performing the extraction or sequencing. Further, as noted by Hanemann *et al.* [91, 92, 88, 87, 89, 90, 86], rule based engines, once configured, are not adaptable to novel anomalous event combinations. Hanemann *et al.* observed that event correlation is a complex problem and could, perhaps, best be approach by combining the capabilities of several AI techniques. They describe the combination of rule based signature recognisers with case based reasoning systems that provide a defence against new and unseen attack profiles. The CBR system is also used to update and inform the rule based recognisers. Section 3.12 presents a range of hybrid systems approaches. Very often, these attempt to find a combination of technologies to implement both misuse and anomaly detection paradigms, *e.g.* using a rule based system for the detection of known misuses, and neural networks for learning normal behavioural patterns for anomaly detection.

4.5 Discussion

A comprehensive review of the literature shows that event correlation is a difficult problem with the most common approach to a solution oriented around rule based systems. Rule based systems certainly suffer from the disadvantage of becoming out of date and pure machine learning approaches suffer by being unable to grasp a large enough temporal picture. The approach of Hanemann *et al.* was to use a hybrid system where rule based techniques were expected to provide a front line defence against intrusion and a case based reasoning system attempted to understand anomalous behaviour as a fuzzy match with previously known behaviour. The hybrid approach suffers predominantly in the complexity of the management of the event correlation system.

Chapter 5

Misuse Localisation

The previous chapters have covered approaches that are primarily used to detect misuse in the network, *i.e.* to specify whether there is a misuse in the network or not. This chapter goes one step further in the network management process and focusses on the localisation of misuses. Typically the misuses of interest are related to faults or performance management.

To realise the localisation of a fault normally requires some kind of knowledge. Thus, most approaches which can be found in the literature do include knowledge in their process. Mostly, the network topology is taken as a basis for the approach, but there are other kinds of knowledge that may help to offer good root cause hypotheses. The quality of knowledge based approaches normally depends on the accuracy of the given knowledge. Consequently, it would be advantageous if the knowledge could be automatically derived from the network or if techniques can be applied to adapt the knowledge when there are changes in the network configuration.

A number of surveys can be found in the literature covering localisation in network management, and sometimes more specifically in fault management and performance management, such as [198], [159], [98], [173] and [131]. These overviews normally make a distinction between different kinds of localisation approaches. Figure 5.1 (taken from [198]) gives a classification of existing approaches. However, some approaches fall into two or more categories. As shown, Steinder *et al.* distinguish between AI techniques (such as rule based systems, model based systems, case based reasoning and machine learning techniques), model traversing techniques (which were mentioned in Section 4.4.2) and fault propagation models (which can also be used in model based techniques).

As previously mentioned, rule based approaches are widely used in the area of network management, despite the fact that they still have a number of disadvantages. In rule based systems, network knowledge is represented by rules. These rules

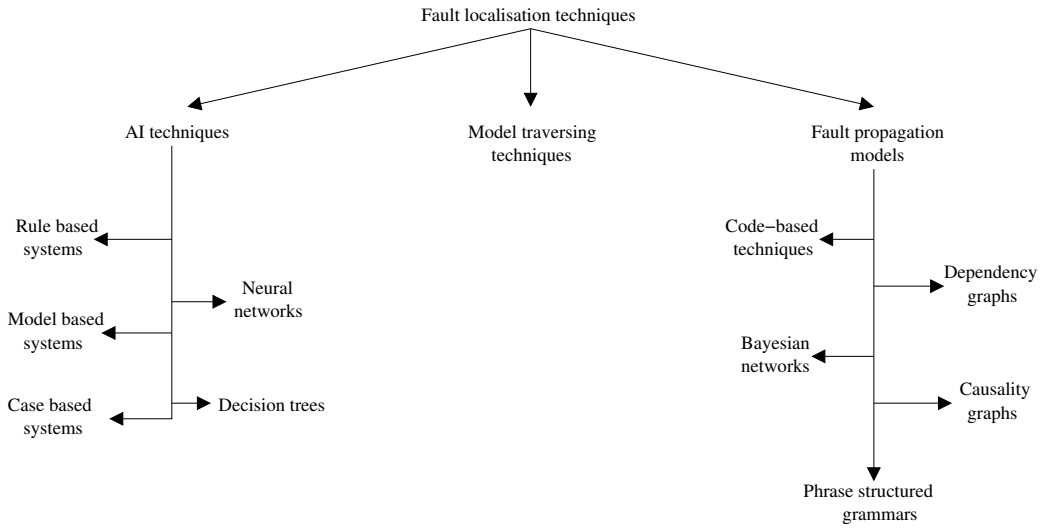

Figure 5.1: Overview of localisation techniques

describe patterns of faulty situations. There are two general approaches for rule based systems referred to as deductive and abductive methods. The former infers new knowledge from the existing knowledge base, and in so doing, tries to infer the observations made. The latter approach takes the observations as assumptions and generalises from given rules (*i.e.* reverses the rule implications). The knowledge bases of these two methods are different.

Normally, forward chaining algorithms are used in the inference engine. Within each step the applicable rules (*i.e.* rules whose conditions are fulfilled by the current state) are looked up and executed. Through this, possible candidates for the root cause are generated. In [151] the distinction between core (*i.e.* general) and custom knowledge is made for the purpose of reducing the effort of rule generation, due to the possibility of reusing core rules.

In general, rule based approaches are easy for human beings to understand and, consequently, an explanation of the diagnosis can be easily extracted. However, they are not able to learn from experience, *i.e.* they normally repeat faulty diagnoses over and over again. Further, unknown problems (which are not covered by the rules) cannot be handled. This means that all problems have to be known and specified in advance, which in the case of networked environments is not possible due to the dynamic environment. Another difficulty is that network configurations are hard coded in the rules. If a configuration changes, all affected rules have to be detected and updated. This is a non-trivial problem.

However, the problem of knowledge adaptation can be partly overcome by model based approaches. Model based localisation is a knowledge based approach that

uses some kind of model as the knowledge base. As mentioned in previous chapters, there are three ways to model a network: to model the normalities, to model the abnormalities, and to model both the normalities and abnormalities of the network. The advantage of modelling the normalities is that unknown misuses are covered with such a model. If only the abnormalities are modelled, all possible misuses have to be considered in the model. However, modelling the normalities is much more expensive than modelling the abnormalities.

When modelling the knowledge in a model, the adaptation of the specified network configurations is much easier, because it is only stated in one place. Due to the deep knowledge of the domain, more demanding patterns can be detected. Another advantage is that the decisions of the system are explainable. In addition, model based systems have the potential to solve novel problems, because they normally provide a better abstraction of the problems or they have a model of the normality of the system. However, model based systems are founded on the assumption that the knowledge that is saved in the knowledge base is correct. Otherwise the generated root cause hypotheses would not be accurate. The main disadvantage of model based systems is the knowledge acquisition, which is time consuming and typically requires expert involvement. In any event, it is normally impossible to provide a complete model of the network and it is a demanding task to keep any model up-to-date. Thus, some approaches try to automatically generate and update the model from the network itself.

Case based reasoning combines normal knowledge based approaches (such as rule based and model based) with the capability of learning from past experience. Within these approaches, past situations that were solved successfully are stored in the case base. If there is a new fault situation it is compared to previous situations on the basis of a similarity measure. The most similar experienced situation is taken and its solution is adapted to the current problem situation. The main disadvantage of case based approaches is that it is hard to define a similarity measure. In particular, the expressiveness of the similarity measure is limited, since it is the same for all cases. The time complexity of these approaches is also normally high and they can therefore be problematic if real time performance is desired.

It may be that a good performance and an adaptation to changing network structures can be achieved by keeping the case base size to some constant level. This requires a drop out strategy for the cases, *e.g.* cases that lead to many false classifications, the oldest cases, or those that are in a crowded area according to the similarity measure can be dropped to constrain the case base size.

During the last few years researches in the area of network management have focussed on distributed approaches. When applying a distributed technique the di-

agnosis problem is broken down into smaller sub-problems, which often can more easily be solved from local diagnosis components. In [159] Meira distinguishes between three common distributed architectures:

- **Centralised localisation** refers to a hierarchical architecture, where all local problems are handled by local managers (or agents) and domain overlapping faults are covered by one central manager.
- **Decentralised localisation** where local managers are responsible for faults in their domain as well as problems that go beyond their domain. In that latter case, the central manager coordinates the fault localisation process.
- **Distributed localisation** refers to an approach where the localisation process is realised only by local managers, *i.e.* no central manager exists. Thus, the degree of autonomy is most significant in this architecture.

The distributed approaches have several advantages compared with centralised systems. Principally, the flow of management data in the network is greatly reduced. In addition, the complexity of the fault diagnosis should be reduced due to the decomposition of the problem. A further potential advantage is that the agents may be capable of initiating corrective actions even if the communication to others is lost. However, one potential disadvantage is that the global view on the problem can be lost if there is no central component, and this has to be made up through communication between the local components.

5.1 Rule Based Localisation

Elmzabi *et al.* [59] propose a fuzzy rule based approach for the realisation of performance management. They improved a former fuzzy rule based approach with the output being linearly related to the rule's input. The new method suggested by the authors has a fuzzy output, meaning that the output also expresses the quality of the generated hypothesis even if the latter is not linearly dependent on the input. The approach consists of two steps. The first includes the initialisation of clusters (each cluster corresponds to a fuzzy rule). The second step employs a genetic algorithm to optimise the parameters of the fuzzy model. The approach was tested in a local area network to predict the state and evolution of traffic in the network.

5.2 Model Based Localisation

This section introduces some existing model based localisation approaches, which use graphs, logic or probabilistic models as knowledge representations.

Fabre *et al.* [66] suggest an underlying model based on a Petri net to realise fault management. Within this Petri net the network elements are described as nodes and the failure dependencies are the edges between nodes. Root cause candidates are identified using failure propagation based on the network elements' dependencies. The authors also propose a distributed version of their approach, which is realised by using local supervisors that complement the missing failure propagation steps of one another.

Kätker *et al.* [117] propose a generic model for fault management to integrate all relevant aspects of the different network layers (the network, the system, and the service management layers). By using a dependency graph they model services and their dependencies. Fault isolation is realised by performing fault propagation on the dependency graph. The authors define some restrictions on the network, which must be met. The graph should be directed, acyclic and finite, *i.e.* it has to be a tree. It is assumed that only one fault exists in the network at a time. If a service S_1 depends on another service S_2 it fails if the service S_2 fails, *i.e.* a dependency expresses a failure dependency.

Katzela *et al.* [119] introduce a model for fault localisation, which is also based on the dependencies between the network elements. These dependencies are expressed in a dependency graph, and they are provided with the probability with which one network element is faulty due to another network element with which it is directly connected. The fault localisation is performed on the basis of these probabilities, *i.e.* the elements with the highest probability are possible candidates for the root cause. Additionally, an alarm cluster is defined. Each alarm cluster contains those network elements that can invoke this specific alarm.

In [119] the network model is tested with two different fault localisation approaches (one for independent failures and one for dependent failures). The approach is based on some assumptions concerning the network. Thus, only one fault at a time is assumed to be in the network. In addition, the alarm has at least to provide the identity of the object that has sensed the malfunction. The authors suggest that the dependency model is general, flexible (*i.e.* it can easily be adapted, although it is not described how the adaptation should be performed), simple (*i.e.* sufficiently rich, but still manageable) and a similarity measure can easily be found.

Steimann *et al.* [196] propose a logic based approach that tries to reduce the prob-

lem of knowledge acquisition by taking most of the information from the OSI Management Information Base (MIB). Certainly, the information in the MIB is not expressed in logic; in addition, information about the behaviour of the network has to be provided by the user. To ease the specification of the behaviour Steimann *et al.* introduce a specification language, which they claim is easy to use. Both the behaviour specification of the user and the information from the MIB are then transformed into order sorted logic. Then the authors define a transformation from this first order logic representation to propositional logic. This reduction can be realised, because all network elements have to be known in advance and thus the network consists of a finite set of managed objects.

After the process of knowledge acquisition, fault detection and localisation processes can be performed on the basis of observations in the network. For every observation the consistency of the model is tested and possible failure models are generated using their DRUM-II algorithm. The DRUM-II algorithm is a model repair algorithm which tests if there is an inconsistency in the model and generates a set of consistent models representing all minimal diagnoses. Further information can be found in [72]. One advantage of this approach is the adaptation of the knowledge base in case of network configuration changes which are inconsistent with the knowledge base. The main problem of this adaptation is that the dynamic structure of the network cannot be adequately handled, *i.e.* all instances have to be known in advance. In addition, temporal patterns are not taken into considerations and thus cannot be detected. The approach also works on some assumptions made concerning the network. First, a closed world assumption is made regarding the information from the MIB. Second, the network structure has to be acyclic, and third all network elements have to be known in the initialisation phase. These assumptions are restrictive and narrow the applicability of the approach.

Kumar *et al.* [130] introduce an abductive reasoning approach for performance management, which employs knowledge of the network in the form of a hyper-bipartite network; a graph where symptoms, faults and hypotheses are represented as nodes and related through edges. The reasoning approach applied to this model is a combination of abductive and deductive reasoning, which has the advantage that it can handle noise in the data (due to the abductive reasoning) and does not generate too many hypotheses (due to the deductive approach).

Barros *et al.* [49] propose an model based approach that employs different kinds of knowledge to allow a deeper analysis of the problem. For example, the network knowledge consists of eight different knowledge levels (e.g. one level is the spatial distribution level, which describes how the network elements are distributed over the area, buildings, rooms, *etc.*). Besides the network knowledge, which is represented using ontologies, there should be knowledge of the network management

and its functionality, such as failure classes, MIBs and diagnostic knowledge (*e.g.* which diagnosis technique should be applied in which situation). Problem solving methods are applied to reason about the knowledge to isolate the possible fault hypotheses. A disadvantage of this approach is that all this knowledge has to be acquired, which could be excessively time consuming. The approach also assumes that there is only one single fault in the network at a time.

Bronstein *et al.* (HP) [25] employ local diagnosis to consider the health of a network element. This local diagnosis, referred to as health awareness, is realised using Bayesian networks. The authors propose a generic model based on the data of sensors of the specific network element. The sensors are mostly network element independent, however, some sensors might be designed just for one specific type of network element. The model of the health engine is divided into four different layers. The first layer refers to the sensors of the network element. The second layer ('measures') collects the information provided by the sensors. The architecture allows information to be stored of any type in these measures. The amount of past data stored here is measure specific. A given sensor can contribute data to several measures. The third layer is the evaluation layer. Its task is to evaluate given measures and create an opinion on the network element's health. The top layer employs reasoning based on Bayesian networks to the different opinions of the evaluation layer and combines them. The accuracy of every evaluator is included in this reasoning process. This accuracy was initially defined by an expert in the conditional probability tables (CPT). The paper proposes an extension by employing Voting EM to this approach. Voting EM is a learning approach that can be used to realise online learning to initialise and improve the CPT, *i.e.* the definition by experts is then no longer needed.

Rish *et al.* (IBM) [184] propose an alternative to event correlation, namely active probing. A probe is a test that collects information about specific network elements. With this information one can diagnose whether any of these network elements is faulty or not. An example of a probe is the *ping* command, but there are other frameworks that offer better possibilities for probing, *i.e.* probes that are tailored for this specific purpose. Rish *et al.* also describe a knowledge based approach. The knowledge about the network is restricted to the given probes and their relationships to network elements, *i.e.* which network elements are covered by this probe. The beliefs about the system state are held in a two layer Bayesian network and can be updated using probabilistic inference.

According to the authors, probing can already be found in the literature. The new approach differs in that it offers 'active' probing, *i.e.* that the probes that will obtain the best information are selected at run time. This offers the possibility that previously obtained knowledge can be included in the probe selection process. If a fault

is detected, the active probing selects the next probe that has the highest information gain. After receiving the result and updating the beliefs concerning the system state, this step is repeated until no more information can be obtained. The conditional entropy specifies the amount of uncertainty. The goal of active probing is to reduce this entropy within every step until the entropy equals zero. The authors also offer some extensions for active probing to reduce some its drawbacks. One drawback, for example, is that the system is assumed to be static during diagnosis. In dynamical systems this is normally not the case. However, this can be partly overcome by using Dynamic Bayesian Networks (DBN) or the sequential multi-fault approach, which is an efficient approximation to DBN's. The authors show that active probing reduces the number of needed probes compared to existing probing approaches. However, the quality of active probing is not compared to non-probing approaches, such as event correlation. Different from some previous approaches, this approach only assumes that there are no more than s (a constant to be specific by an expert) simultaneous faults in the network. The main advantage of this approach is that it can handle noise in data, which can occur in networks due, for example, to packet loss. It is also suggested that a combination of event correlation techniques and active probing is possible, and that some disadvantages of active probing can be overcome by this combination.

Tang *et al.* [203] propose such an approach (Active Integrated Fault Localisation - AIR), which combines active probing with passive fault localisation (such as model based reasoning). They extend the often used symptom-fault map where the faults and their symptoms are saved along with the probability of a fault causing a specific symptom. This map is extended with actions (probes), which can be used to test the existence of specific symptoms. Instead of specifying the relationship between faults and probes, this model relates symptoms and probes. The approach consists of three different components that fulfil different functions. The first is the fault reasoning module. This is a passive fault localisation approach that collects incoming events from the network elements and correlates them for the generation of root cause hypotheses. The fidelity evaluation component measures these hypotheses regarding their plausibility (the hypothesis that assumes the fewest faults and whose faults most probably cause the observed symptoms is assumed to be the most likely). If the fidelity value of the most probable hypothesis is satisfactory (beyond a given threshold) the fault localisation process terminates. Otherwise, the action selection component tries to prove this hypothesis. For this purpose, probes are send to obtain the missing information or to underline the existence of a symptom. Consequently, noise in the data (such as the loss of events and consequently symptoms) has less effect on the accuracy of the approach, because unobserved symptoms can be detected when employing the active probing.

By combining passive fault localisation and active probing, some of their individual weaknesses can be addressed. The passive fault localisation is normally susceptible to noise in the data. This can be overcome by using active probing, which obtains additional information concerning the existence of symptoms. Active probing lacks the ability to track intermittent network faults and performance related faults, and normally does not scale well. These disadvantages can be overcome, because the passive fault localisation approach is applied first. Tang *et al.* contend that AIR increases the detection rate and minimises the false positive rate, as well as reducing the fault detection time compared to passive fault localisation techniques. According to the authors, the AIR approach also scales well, even in large networks, and is noise tolerant. The approach can be applied to probabilistic as well as deterministic models. Active probing may offer the possibility to gain automatic feedback from the system and thus reduce expert involvement.

Huard (AT&T) [104] introduces a basic probabilistic approach for fault management. The knowledge base is realised by a Bayesian network containing an acyclic model of the possible faults and their dependencies. Based on this knowledge the approach generates the possible root cause candidates, which should be confirmed during further testing.

Steinder *et al.* [197] suggest an approach employing a symptom-fault map as a probabilistic model. Similar to [203] the causality relationships between faults and symptoms are saved along with their probability of causal implication. In addition, the probability of independent failure is associated with every fault. This approach assumes that a symptom may be caused by only one fault at a time, which naturally reduced the applicability of the approach. The main advantage is that it is able to provide a set of most likely hypotheses at any time. Since the approach works incrementally, these hypotheses are updated according to new obtained knowledge (*i.e.* of newly received events) and refines them step-by-step. All hypotheses are ranked according to a measure of 'goodness'. To be able to handle spurious symptoms and to incorporate positive symptoms (*i.e.* if a failure symptom is disproved or was not observed at all) the authors extended the above approach further. A flag that indicates if a symptom is observable or not and the probability with which a symptom might get lost is required. Further, the probability with which a symptom may be spuriously generated needs to be defined. These probabilities are included in the ranking of the hypotheses. One clear disadvantage is that the different probabilities may be difficult to define and the whole approach relies on their correctness. Two heuristics are applied to reduce the complexity. For the first, it is assumed that the hypothesis with less faults are more probable than others. The second heuristic limits the number of hypotheses that are generated. Consequently, the less probable hypotheses are left out of further proceedings. As an effect, the optimal solution

may not be found, but the complexity of the approach is reduced.

5.3 Case based Localisation

Melchiors *et al.* [160] propose an approach that uses case based reasoning for fault diagnosis. This is based on Trouble Ticket Systems (TTS), which are widely used in this domain and provide information on the occurred alarm (like events in OSI networks). Based on previous failure cases, and further domain knowledge (such as failure classes and their properties), similar cases are detected and adapted to the present situation. This approach also employs an additional learning component that improves the similarity measure. Additionally, the similarity measure is computed depending on the diagnosed failure class, *i.e.* the similarity measure does not have to be completely general but can be tailored for a specific failure class.

5.4 Decision Trees

5 Chen *et al.* [36] propose an approach that makes use of request logs which have to be labelled, *i.e.* every request has to be labelled if it was faulty, and if so it has to be provided with its root cause. On the basis of these request logs, they build a decision tree that classifies the failed and successful requests. The failure paths of the decision tree are then post-processed and root cause candidates are extracted. For learning, the C4.5 algorithm and an information gain algorithm, MinEntropy, was applied. Four heuristics are applied afterwards to filter out false positives. The first is that leaves with successful requests are left out. Second, noise in the data is filtered out according to a noise threshold: every leaf that contains less failures than specified in the threshold is left out of further proceedings. Third, nodes that can be logically subsumed by their successor nodes are merged together. The forth heuristic ranks the candidates according to their degree of correlation with failure. Within this paper the probability of independent failures is assumed to be low. This approach was applied to localise failure in Internet pages and was tested at EBay. The main advantage of the approach is that the time consuming knowledge acquisition can be skipped and that an explanation for the decision can easily be extracted. However, the false positive rate is still high (approximately 25%).

5.5 Hybrid Approaches

Hanemann [86] proposes an approach for service fault management, *i.e.* a fault management component that offers a user interface for customers and automatically processes customer complaints. For this task the author proposes a hybrid approach, namely a combination of a rule based and a case based reasoning. The rule based component serves as the standard fault localisation approach. If the rule based component is unable to offer a solution (due to incomplete knowledge), the case based reasoning module is then used to overcome this problem. The rules for the knowledge base should be automatically derived from the Service MIB. Only in some cases should a manual intervention of the administrators be necessary. The derived knowledge should normally include dependencies between the services themselves and their dependencies on the network resources.

Nuansri *et al.* [171] propose a hybrid approach consisting of a rule based and a neural network component. The neural network component (namely BRAINNE) was used for knowledge acquisition. Its inputs were former log files (error messages and their root cause) and it was used to generate corresponding rules that represent this knowledge. The rule based component (namely NEXPERT) works on the basis of the rule base generated by the BRAINNE module. It is used as the actual fault management component, *i.e.* it receives the errors that have occurred, and generates the appropriate root cause. To be able to react on network changes, the BRAINNE module tests from time to time if there are new rules that can be obtained. If so, these rules are added to the knowledge base, thus adapting the acquired knowledge where necessary.

5.6 Distributed Localisation Approaches

Cheikhrouhou *et al.*[34, 33, 35] describe an approach where a multi-agent system is employed to realise fault management. The authors explain the functionality of a framework where intelligent agents can be adapted to a specific scenario by defining skills for the agents. These skills can be loaded onto the agent at runtime, and correspond to a specific ability of the agent (such as monitoring, fault detection, *etc.*). In addition, each agent is provided with general functionalities, such as communication and skill management. A multi-agent system acts as a container. Thus, the concrete realisation of fault detection can be exchanged by others. However, the authors mention the BDI-architecture (Beliefs–Desires–Intentions) used for knowledge representation within this approach. With this logical representation one can express the network state on the one hand and the goals of agents on the other. Due

to the distributed solution proposed here, the underlying data is filtered earlier and the amount of data that has to be sent over the network can be reduced. Further, because of the skills, the framework provides a generic solution, which can be widely used and easily adapted. It also offers the possibility to employ fault detection and fault localisation techniques other than the BDI-architecture. The agents have a hierarchical order, which means that they can be controlled and monitored by a central position.

Garijo *et al.* [75] describes a framework for fault management which makes use of a multi-agent system. The different tasks in a fault management system, such as monitoring, diagnosis and user interaction, are realised by different agents. The only interface to the network is realised by the Access Agent. This agent receives the events from the network and provides operations to obtain further information from the network. It is the only component that depends on the specific type of the network. There are two interfaces to the outside: the User Agent and the External Communication Coordinator. The User Agent represents the interface to the network operator, *i.e.* the user of the system. Through this agent, the operator can supervise the management process and can interfere using operations provided by the User Agent. The External Communication Coordinator offers the possibility to communicate with other Management Frameworks. The heart of the fault management framework is the Recognition Agent. This agent receives the events and clusters them according to their presumed root cause. Then, a Diagnostic Agent is created, which realises the event correlation for one cluster of events and which generates root cause hypotheses. These hypotheses are verified by the Operation Agent, which performs tests on the network elements involved (using their Access Agents). The result (the probable root cause) is presented to the user thereafter. This paper only describes the framework for a fault management system. Nothing is said about the concrete realisation (for example of the event correlation). In addition, the required knowledge is only specified vaguely.

Chapter 6

Conclusions

6.1 Summary

This report has considered the application of Artificial Intelligence (AI) to the domain of misuse detection and localisation in telecommunications networks. Much of the literature focusses on intrusion detection at a low level (*e.g.* based on TCP/IP traffic), for which a diversity of AI techniques has been applied, from classical approaches (*e.g.* clustering and classification, and expert systems) to natural computing (*e.g.* artificial neural networks, evolutionary computation, and artificial immune systems). A broad survey of such techniques and applications has been reported and, although the literature is biased towards computer networks, the techniques are considered to be equally applicable to telecommunications networks. This survey was followed by a more detailed examination of event correlation, as the accepted approach to misuse detection and localisation (in the general sense) based on higher level sources of data, *e.g.* event logs or alarms from network resources. There is less concentration in the literature on event correlation, and approaches at this level are dominated by rule based systems, often pre-configured based on expert knowledge. These suffer from a number of drawbacks, including lack of generalisation and adaptability. Thus, the initial premises of the work, as laid out in the introduction, were confirmed. Current approaches for the localisation of misuse sources were also considered. The majority of approaches were applied in the area of fault and performance management and focussed on knowledge based techniques. However, in recent years researchers have tried to combine learning and knowledge based approaches or to realise distributed approaches. These seem to be very promising directions for misuse localisation but still have a number of drawbacks, such as time consuming knowledge acquisition and the assumptions made to reduce their complexity.

6.2 Findings

This report has covered a diversity of literature on the subject of intrusion detection, misuse detection and event correlation. The applicable techniques from the domain of AI that can be identified from the literature include *inter alia* expert systems, case based reasoning, artificial neural networks, evolutionary computation, swarm intelligence, artificial immune systems, and agent based systems, as well as a wide variety of statistical approaches to clustering, classification and regression. The main focus of this work for misuse detection is on event correlation, which, unlike low level intrusion detection, does not enjoy such a variety of currently applicable tools and techniques; most systems being rule or state based. Drawbacks with canonical rule based systems, of import in this domain, include:

- The need for a large knowledge base of rules for detecting known misuses, often requiring considerable investment in expertise and knowledge engineering.
- The lack of generalisation ability to unseen misuses, with such systems being built on the principle of misuse detection rather than anomaly detection.
- The lack of automated processes for adapting to new misuses and/or a changing environment.
- The interaction between rules in the knowledge base, which can make updating the system to cover a new misuse a non-trivial task.
- The need for house-keeping to purge the system of out-of-date or conflicting rules.

As pointed out in this report misuse localisation needs some kind of topological knowledge of the network. Consequently, most localisation techniques include some form of additional in their process. However, it is still unclear how to shorten the knowledge acquisition made by experts or how to automatically generate the knowledge from the network itself. In recent years research has focussed on hybrid approaches that combine knowledge based approaches and learning, and on distributed approaches.

It is evident from the literature, that these remain active research topics, and that there are no canonical solutions, especially in the telecommunications domain where the network topology can change frequently.

6.3 Recommendations

The principal difficulty in developing an event correlation engine is the reliance on rule or state based approaches. This is virtually dictated by the nature of the problem to be solved. AI techniques are promising as they tend to offer, to varying degrees, the ability to learn, generalise and adapt. Hence, one could seek to develop entirely new architectures for event correlation based purely on contemporary AI techniques. However, the dominance of rule based systems is not without reason - they are, in essence, the only architecture currently capable of processing, with any measure of success, the potentially complex temporal and causal relationships required contemporary event correlation engine. A logical first step seems, therefore, to explore further opportunities for hybridisation of rule based systems with more flexible AI techniques. Specifically, it is desirable to address the following key issues:

- Automating the task of populating the knowledge containers prior to system deployment.
- Enabling the generalisation of detection capability to unknown misuses.
- Facilitating the (semi-)on-line adaptation and optimisation of knowledge containers to:
  - add or modify rules to counter false negatives
  - remove or modify rules to counter false positives
  - remove redundant rules to maintain a sensible knowledge base size (this may involve a trade-off between factors such as detection capability, knowledge base size and run-time performance)

It should be noted that these activities are not without user involvement. For example, the initial population of knowledge containers would be based on training data prepared by the user, and subsequent adaptation would require the system to be instructed when false negatives or false positives occur.

From the field of AI, a variety of techniques could be explored to achieve these goals. Based on the comprehensive review presented herein, the most promising approaches for initial exploration would seem to be neural and fuzzy systems to realise some measure of generalisation, and evolutionary techniques for both the initial population of the knowledge containers and subsequent adaptation and optimisation. The ability of evolutionary techniques to cope with multi-criteria problems may be of particular use when exploring the trade-off between different, potentially

conflicting, objectives, such as minimising knowledge base size whilst maximising detection capability.

These recommendations are not without precedent. Neural approaches have been widely applied for the detection of known and unknown misuses (*e.g.* [29] [76][139][37]), based typically on low level data. Evolutionary approaches have been used to evolve rules for expert systems (*e.g.* [146] [144]), and also combined with fuzzy systems (*e.g.* [80]). Other related techniques have also been used for developing rules, such as particle swarm optimisation (*e.g.* [1]), although the general notions are similar. Rule based systems have also been hybridised with other techniques, such as case based reasoning [86], case based reasoning and neural networks [84], learning vector quantisation [155], and fuzzy data mining [24].

When moving away from largely deterministic systems, it is worth bearing in mind the fundamental premises of the AI techniques involved; principally:

- That it must be possible to infer the output from the input, *i.e.* the data set being processed must have the appropriate information content to arrive consistently at the desired conclusions. This property is essential for learning.
- That similarity in the input space, however that might be measured, must map to similarity in the output space. Note that the converse is not necessary; different inputs could map to the same or similar outputs. This property is essential for generalisation.

Whilst these may seem obvious preconditions, the nature of the alarm data to be correlated for the identification of misuses, from frauds to faults, needs to be examined to confirm that these are satisfied before AI techniques can be applied with any degree of confidence. In particular, the preparation of test data from which to derive the initial population of the knowledge containers, should consider the 'coverage' of misuse types. Specifically, a misuse (as opposed to anomaly) detection mechanism, is not likely to detect unknown attacks that are fundamentally different in nature from the known attacks on which it was trained. It should also be noted that the more 'flexible' architectures, such as neural and evolutionary systems, reduce the explanatory power of any detection system in which they are employed. A major advantage of canonical rule based systems is that they can present the user with the chain of reasoning by which they arrived at their conclusion. Other techniques, such as case based reasoning, can offer some explanation based on similarity measures, whilst others are often unable to present any indication of why they reached a particular conclusion. This is a natural dichotomy, rather than a weakness in the techniques - if one requires the flexibility to learn, generalise and adapt, then the determinism of rule based systems must be sacrificed.

Appendix A

Applications

The following sections briefly describe a number of 'in use' Intrusion Detection Systems; that is, systems that have moved from theoretical investigation to implementation. This is not intended to be a comprehensive list, but rather to provide a selection that exemplifies the evolution of IDS systems and the techniques used for their development. The extensive research in the area of intrusion detection in computer networks has led to a large number of practical systems, more so than in the telecommunications domain, hence we have drawn some examples from conventional intrusion detection systems. The systems outlined in this appendix are:

- **DIDS** Distributed Intrusion Detection System.
- **HP Openview**.
- **LIDS** Linux Intrusion Detection System.
- **FLIPS** Feedback Learning Intrusion Prevention System.
- **SPECTRUM**.
- **SNORT**.

A.1 DIDS

Distributed Intrusion Detection System. Refer to Snapp *et al.* [192] for details. This development is concerned with the so called 'network user identification problem' which addresses the issue of tracking a user who moves across a network from host to host with potentially a new user id on each machine. An instance of this type of misuse is the *doorknob* attack. Here an intruder's objective is gain access to a system

through an insufficiently protected host on the network, *i.e.* the attacker knocks on as many doors as necessary until a weakness is found that enables ingress.

The DIDS architecture is a combination of distributed monitoring, subsequent data reduction and central analysis. Hosts that do not possess a monitor function can be protected by monitoring the network between hosts. This architecture was proposed and implemented in 1991 and can therefore be considered an early example of effective intrusion detection systems.

Inevitably, DIDS employs an a rule based system as its central event correlation mechanism. This system was written in Prolog and the expression of the rules is closely related to the intrinsic Prolog formalism. The rule based system requires a fundamental model of the system which, in Prolog terms, means the existence of a set of primitives or facts and a set of relations connecting them. Essentially this is a semantic net forming a model of the system. Rules become superposed Prolog relations derived from this net. The semantic model is layered, much in the style of the ISO layered model that was gaining strong credibility at the time. The layers are:

Level	Name	Explanation
6	Security State	Overall network security level
5	Threat	Definition of categories of abuse
4	Context	Event placed in context
3	Subject	Definition and disambiguation of network user
2	Event	Representation of user action
1	Data	Low level log data

It can be seen from this table how the process of event correlation involves a sequence of aggregation and correlation steps as moving up the table to higher levels of abstraction.

The DIDS system was implemented on a network of Sun SPARC workstations. It performed adequately according to Snapp *et al.* [192]. Improvements were possible by using a different rule based generation system, as Prolog suffers performance problems. The most significant defect was in the absence of a signature recognition system.

A.2 FLIPS

Feedback Learning Intrusion Prevention System. According to Locasto *et al.* [150] FLIPS is a host based system intended to augment perimeter type prevention mech-

anisms like firewalls and IDS functions thereby implementing a defence in depth policy. FLIPS is implemented as an HTTP server protection mechanism. Its principal function is to prevent code injection attacks. Locasto *et al.* argue that an IDS is a passive system, because it essentially performs a pattern recognition function. They propose an Intrusion Prevention System (IPS) which combines this function with the objective of preventing the intrusion to the system. To facilitate this, they contend that the system must incorporate feedback learning. This allows passive pattern recognition functions to be updated with known or active attack profiles.

FLIPS consists of an anomaly based classifier, a signature based filtering scheme and a supervision framework that enables instruction set randomisation (ISR). It should be noted that the principal advantage gained from employing an IPS, and FLIPS in particular, is to detect and prevent code injection attacks.

A.3 HP OpenView

HP Openview is a management software suite that provides a comprehensive support for IT infrastructure management. HP Openview consists of many modules and one in particular, Event Correlation Services, is responsible for intrusion detection within the IT framework. It employs a GUI based rule generation module that assists with the generation of rules for the event correlation engine. Sheers [190] provides a description of HP OpenView. The system is distributed, since each separate network unit is expected to perform local correlation and aggregation. The underlying architecture employs networks of rules called nodes, through which events flow. The rules could combine or aggregate events and contain a temporal dimension that permits long separated events. An interesting feature is that the temporal sequence is not one way. The correlation engine permits events to arrive in different temporal orders. That is, it seems possible for a correlated event to occur backwards in time but the engine understands that this is only a function of network delay causing unusual sequencing of correlated events. The nodes form a circuit diagram and a functional programming language is employed as a specification framework. The correlation engine effectively executes the specification directly. Hanemann [86] refers to this product and supplies the following active Web Site.

`http://www.managementsoftware.hp.com/products/ecs/`

A.4 LIDS, Linux Intrusion Detection System

This section considers, for completeness, what appears to be an example of a freely available IDS based around the Linux operating system. Unix type operating systems are constructed around a superuser model where there are 'normal' users with local privileges and a user, often called root, with unrestricted access to all parts of the system. Root is not subject to any of the restrictions of access that may apply within the file system nor is root subject to restrictions regarding execution access and system modification. In effect the user logged into root owns the system. The Linux Intrusion Detection System (LIDS) is a modification to the Linux kernel that allows many of the root privileges to be removed. Further, LIDS in fact implements an access mechanism that enables privileges to be allocated on a needed basis. A LIDS kernel also implements a TCP/IP port scanner. Although there are many separate port scanners, the one provided with LIDS is strongly coupled with the entire LIDS functionality. LIDS also prevents the indiscriminate installation of software. Attempts to install port sniffers, for example, are rejected. It seems clear that LIDS does not implement an event correlation engine and is actually an access allocation mechanism combined with log scanners and comprehensive system monitoring functions. In this respect LIDS is similar to many other freely available so called IDS applications based around the Linux operating system. See for example SNORT, AIDE, LOGCHECK, *etc*. The current Web site for LIDS is

`http://www.lids.org`

A.5 SPECTRUM

SPECTRUM is a commercial system from Concord Communications Inc. Concord provides Business Service Management software with the objective of increasing system availability be improving system reliability and implementing tight Service Level Agreements.

`http://www.concord.com/aboutus/`

The SPECTRUM software manages the availability and performance of IT infrastructures and the consequent dependent services. It combines model based, rule based, and something referred to as Policy Based approaches, in order to obtain the best from each. Product documentation can be found on the product Web Site.

```
http://www.aprisma.com/literature/brochures/
   br0518.pdf
```

SPECTRUM is composed of many connected and collaborating modules. Of particular interest is the approach taken to event correlation. This is managed by event rules, examples of which are:

- **Event Pair (Event Coincidence)** This rule covers the situation when events are expected to occur in pairs. These events do not need to be contiguous. Of course, inductively, it is possible to extend pairs to multiple event occurrences.
- **Event Rate Counter (Event Frequency)** For example, if an event occurs a certain number of times in a particular time interval it is abnormal. Denial of service attacks certainly fall into this category.

A.6 SNORT

The following summary is take from the SNORT Web site.

> Snort is an open source network intrusion prevention and detection system utilizing a rule-driven language, which combines the benefits of signature, protocol and anomaly based inspection methods. With millions of downloads to date, Snort is the most widely deployed intrusion detection and prevention technology worldwide and has become the de facto standard for the industry.

```
http://www.snort.org
```

The SNORT package is a rule based system and derives its basic information from the network layer. The rules are flexible, and are often developed and offered by the software community in general. The Web site maintains a large set of validated and verified rules for general use. It is possible to create bespoke rules and also possible, and encouraged, to submit these to the SNORT management group as additions to the publicly available rule set. The Web site maintains a large set of documents including a FAQ. There are also many books available on this subject, for example, Caswell *et al.* [30].

Appendix B

Test Data

Many of the 'in use' IDS applications described in this report have been extensively tested using common or standard test data sets. Since these are referred to frequently in connection with the review of artificial intelligence techniques, this appendix describes the DARPA data set, and a derivation of this, the KDD-Cup99 data set. These data sets consist of low level network events in various forms.

B.1 DARPA

The Information Systems Technology Group (IST) of MIT Lincoln Laboratory, within the remit of a DARPA sponsored project, collected a standard set of test data for the purposes of testing and evaluating Intrusion Detection Systems. In 1999 the evaluations were performed on a number of IDS applications. Measurements of the probability of detection and probability of false alarms were obtained. MIT provide an updated Web site containing the test data, documentation, descriptions and publications. The URL for this is:

```
http://www.ll.mit.edu/IST/ideval/data/1999/
    1999_data_index.html
```

As observed on this site

> These evaluations contributed significantly to the intrusion detection research field by providing direction for research efforts and an objective calibration of the state-of-the-art. They are of interest to researchers working on the general problem of workstation and network intrusion detection. The evaluation was designed to be simple, to focus

on core technology issues, to encourage the widest possible participation by eliminating security and privacy concerns, and by providing data types that were used commonly by the majority of Intrusion Detection Systems.

IDS applications could be tested within an offline or real time environment. Offline tests used, for example, audit logs and TCP/IP dumps collected from a simulation network. Real time tests were expected to operate within actual networks under operational timescales. The IDS applications were expected to identify anomalous and attack behaviour within a background of normal network activity. Documentation and related publications are also available from this site. The agency supplied three weeks of training data. Only the second week of data contained attacks. The remaining two weeks were provided to enable 'normal' activity to be sampled.

B.2 KDD-Cup99

This data set was used for the *Third International Knowledge Discovery and Data Mining Tools Competition* and is a version of the DARPA data set described above. This was conducted in parallel with KDD-99 the *Fifth International Conference on Knowledge Discovery and Data Mining*. The objective of the competition was to build an IDS application that was capable of distinguishing between normal and anomalous network behaviour. The data was a simulation of a military network environment. The Web site provides a detailed description of the required IDS application, the full simulation data set, and a description of the attack data types embedded within the otherwise normal activity of the network. The URL for this is:

`http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`

Comprehensive details concerning the data set and the results of the competition are provided by Stolfo *et al.* [201]. The data set is essentially a set of TCP/IP dump data. The types of attack embedded within this data are:

- Denial of service, *e.g.* SYN floods attacks.
- Unauthorized access from a remote machine, *e.g.* guessing password.
- Unauthorized access to local superuser (root) privileges, *e.g.* various 'buffer overflow' attacks.

- Surveillance and other probing, *e.g.* port scanning.

In nearly all of these attack types the composite attack events are distributed, asynchronous and atemporal, and as a consequence place considerable demands on the underlying event correlation engines.

References

[1] M. S. Abadeh, J. Habibi, and S. Aliari. Using a Particle Swarm Optimization Approach for Evolutionary Fuzzy Rule Learning: A Case Study of Intrusion Detection. In *Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU)*, Paris, France, July 2–7 2006.

[2] T. Abbes, A. Bouhoula, and M. Rusinowitch. On the Fly Pattern Matching for Intrusion Detection with SNORT. *Annals of Telecommunications*, 59(9–10), Sep–Oct 2004.

[3] A. Adi, D. Botzer, O. Etzion, and T. Yatzkar-Haham. Push Technology Personalization Through Event Correlation. In *Proceedings of the 26th Conference on Very Large Databases*, pages 643–645, 2000.

[4] S. H. Ahmed, I. A. S. Ismail, and A. S. A. Alim. IDSUDA: An Intrusion Detection System Using Distributed Agents. *International Journal of Computer Networks and Internet Research*, 5, 2005.

[5] A. V. Aho and M. J. Corasik. Efficient String Matching: An Aid to Bibliographic Search. *Communications of the ACM*, 18(6), 1975.

[6] E. Al-Shaer. A Dynamic Group Management Framework for Distributed Event Large Scale Monitoring. In *Proceedings of the 7th IFP/IEEE International Symposium on Integrated Network Management (IM'01)*, pages 565–578, Seattle, 2001.

[7] K. G. Anagnostakis, E. P. Markatos, S. Antonatos, and M. Polychronakis. $E^2 \times B$: A Domain Specific String Matching Algorithm for Intrusion Detection. In *Proceedings of the 18th IFIP International Information Security Conference (SEC2003)*, Athens, Greece, 2003.

[8] D. anderson, T. Frivold, and A. Valdes. Next Generation Intrusion Detection Expert System (NIDES). Technical report, International Computer Science Laboratory Technical Report SRI-CSL-95-07, May 1995.

[9] R. anderson and A. Kattak. The Use of Information Retrieval Techniques for Intrusion Detection. In *Proceedings of the 1st International Workshop on Recent Advances in Intrusion Detection (RAID98)*, Louvain-la-Neuve, Belgium, 1998.

[10] S. Antonatos, M. Polychronakis, P. Akritidis, D. Kostas, K. G. Anagnostakis, and E. P. Markatos. Piranha: Fast and Memory Efficient Pattern Matching for Intrusion Detection. In *Proceedings of the 20th International Information Security Conference (IFIP/SEC 2005)*, May 2005.

[11] K. Appleby, G. Goldszmidt, and M. Steinder. Yemanja - A Layered Event Correlation Engine for Multi Domain Server Farms. In *Integrated Network Management VII*, pages 329–344, 2001.

[12] W. H. Au, K. C. C. Chan, and X. Yao. A Novel Evolutionary Data Mining Algorithm with Application to Churn Prediction. *IEEE Transactions on Evolutionary Computation*, 7(6), 2003.

[13] R. Bace and P. Mell. Intrusion Detection Systems. *NIST Special Publication on Intrusion Detection Systems*, November 2001.

[14] B. Balajinath and S. V. Raghavan. Intrusion Detection Through Learning Behaviour Model. *International Journal of Computer Communications*, 24(12):1202–1212, July 2001.

[15] J. Balthrop, F. Esponda, S. Forrest, and M. Glickman. Coverage and Generalization in an Artificial Immune System. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002)*, pages 3–10, New York, 2002. Morgan Kaufmann.

[16] S. Banerjee, C. Crina Grosan, and A. Abraham. IDEAS: Intrusion Detection Based on Emotional Ants for Sensors. In *5th International Conference on Intelligent Systems, Design and Applications (ISDA-05)*, Wroclaw, Poland, 2005.

[17] D. Barbara, J. Couto, S. Jajodia, and N. Wu. ADAM: A Testbed for Exploring the Use of Data Mining in Intrusion Detection. *Special Interest Group on Management of Data (SIGMOD)*, 30(4), 2001.

[18] D. Barbara, N. Wu, and S. Jajodia. Detecting Novel Network Intrusions Using Bayes Estimators. In *Proceedings of the SIAM International Conference on Data Mining*, Chicago, USA, April 2001.

[19] H. Barendregt. *The Lambda Calculus its Syntax and Semantics*. springer Verlag, 1984.

[20] S. Benferhat, F. Autrel, and F. Cuppens. Enhanced Correlation in an Intrusion Detection Process. In *Lecture Notes in Computer Science*, volume 2776, pages 157–170. Springer, 2003.

[21] V. Berk, W. Chung, V. Crespi, G. Cybenko, R. Gray, D. Hernando, G. Jiang, H. Li, and Y. Sheng. Process Query System for Surveillance and Awareness. In *7th World Multi Conference on Systemics, Cybernetics and Informatics (SCI 2003)*, Orlando, Florida, July 2003.

[22] B. H. Bloom. Space/time Trade-offs in Hash Coding with Allowable Errors. *Communications of the ACM*, 13(7):422–426, 1970.

[23] R. J. Bolton and D. J. Hand. Statistical Fraud Detection: A Review. *Statistical Science*, 17(3):235–255, 2002.

[24] S. M. Bridges and R. B. Vaughan. Fuzzy Data Mining and Genetic Algorithms Applied to Intrusion Detection. In *Proceedings of the National Information Systems Security Conference (NISSC)*, Baltimore, MD, October 2000.

[25] A. Bronstein, J. Das, M. Duro, R. Friedrich, G. Kleyner, M. Mueller, S. Singhal, and I. Cohen. Self-aware services: Using bayesian networks for detecting anomalies in internet-based services. Technical Report HPL-2001-23 (R.1), Internet Storage and Systems Laboratory, HP Laboratories Palo Alto, October 2001.

[26] A. Brown, G. Kar, and A. Keller. An Active Approach to Characterizing Dynamic Dependencies for Problem Determination in a Distributed Application Environment. In *7th IEEE/IFIP International Symposium on Integrated Network Management*, Seattle, 2001. Chapman and Hall Ltd.

[27] S. Brugger. Data Mining Methods for Network Intrusion Detection. *ISCR Computer Science Series, University of California*, June 2004.

[28] J. Cannady. Artificial Neural Networks for Misuse Detection. In *Proceedings of the National Information Systems Security Conference*. NIST, Washington, DC, 1998.

[29] J. Cannady. Next Generation Intrusion Detection: Autonomous Reinforcement Learning of Network Attacks. In *Proceedings of the 23rd National Information Systems Secuity Conference*, 2000.

[30] B. Caswell, J. C. Foster, R. Russell, J. Beale, and J. Posluns. *Snort 2.0 Intrusion Detection*. Syngress Publishing, 2003.

[31] B. Caswell and M. Roesch. Snort: The open source network intrusion detection system. In *Internet: http://www.snort.org/*, 2004.

[32] S. Chebrolu, A. Abraham, and J. P. Thomas. Feature Deduction and Ensemble Design of Intrusion Detection Systems. *Computers and Security*, 24(4):295–307, June 2005.

[33] M. M. Cheikhrouhou. Using bdi-oriented agents for network management, 1998.

[34] M. M. Cheikhrouhou, P. Conti, and J. Labetoulle. Intelligent agents in network management, a state-of-the-art. *Networking and Information Systems*, 1(1):9–38, 1998.

[35] M. M. Cheikhrouhou, P. Conti, K. Marcus, and J. Labetoulle. A software agent architecture for network management: Case-studies and experience gained. *Journal of Network and Systems Management*, 8(3), 2000.

[36] M. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. A. Brewer. Failure diagnosis using decision trees. In *1st International Conference on Autonomic Computing (ICAC 2004)*, pages 36–43, 2004.

[37] Y. Chen, A. Abraham, and J. Yang. Feature Selection and Intrusion Detection Using Hybrid Flexible Neural Tree. In *International Syumposium on Neural Networks (ISNN 03)*, pages 439–444, Chongqing, China, 2005.

[38] D. Chess, C. Harrison, and A. Kershenbaum. Mobile Agents: Are They a Good Idea? Technical Report RC 19887 (December 21, 1994 - Declassified March 16, 1995), IBM Research Report, Yorktown Heights, New York, 1994.

[39] K. Chokshi, C. Panchev, S. Wermter, and K. Burn. Self Organising Neural Place Codes for Vision Based Robot Navigation. In *Proceedings of the International Joint Conference on neural Networks*, pages 2501–2506, Budapest, Hungary, July 2004.

[40] R. C. Clark and D. E. Schimmel. A Pattern Matching Co-Processor for Network Intrusion Detection Systems. In *IEEE International Conference on Field Programmable Technology*, Tokyo, Japan, December 2003.

[41] W. Cockayne and M. Zyda, editors. *Mobile Agents*. Prentice-Hall, 1998.

[42] S. Coull, J. Branch, B. Szymanski, and E. Breimer. Intrusion Detection: A Bioinformatics Approach. In *Proceedings of the 19th Annual Computer Security Applications Conference*, Las Vegas, Nevada, December 2003.

[43] F. Cuppens and A. Miege. Alert Correlation in a Cooperative Intrusion Detection Framework. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 202–215, 2002.

[44] F. Cuppens and R. Ortalo. LAMBDA: A Language to Model a Database for Detection of Attacks. In *RAID '00: Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection*, pages 197–216, London, UK, 2000. Springer-Verlag.

[45] D. Dasgupta. Immunity Based Intrusion Detection System: A General Framework. In *Proceedings of the 22nd National Information Systems Security Conference*, pages 147–160, 1999.

[46] D. Dasgupta. An Immune Agent Architecture for Intrusion Detection. In *GECCO'00 - Workshop Proceedings*, pages 42–44, 2000.

[47] D. Dasgupta and F. González. An Immunity Based Technique to Characterize Intrusions in Computer Networks. *IEEE Transactions on Evolutionary Computation*, 6:1081–1088, 2002.

[48] D. Dasgupta and N. Attoh Okine. Immunity Based Systems: A Survey. In *IEEE International Conference on Systems, Man and Cybernetics*, Hyatt Orlando, Florida, 1997.

[49] L. N. de Barros, M. Lemos, V. Bernal, and J. Wainer. Model based diagnosis for network communication faults. In *AIDIN '99: International Workshop on Artificial Intelligence for Distributed Information Networking*, pages 57–62. AAAI Press, 1999.

[50] L. N. de Castro and J. I. Timmis. Artificial Immune Systems: A Novel Paradigm to Pattern Recognition. In J. M. Corchado, L. Alonso, and C. Fyfe, editors, *Artificial Neural networks in pattern Recognition*, pages 67–84. Springer Verlag, University of Paisley, UK, 2002.

[51] H. Debar. An Introduction to Intrusion Detection Systems. In *Connect 2000*, 2000.

[52] H. Debar, M. Dacier, and A. Wespi. Towards a Taxonomy of Intrusion Detection Systems. *Computer Networks*, 31:805–822, 1999.

[53] K. Deeter, K. Singh, S.Wilson, L. Filipozzi, and S.T. Vuong. APHIDS: A Mobile Agent Based Programmable Hybrid Intrusion Detection System. In *Mobility Aware Technologies and Applications/Mobile Agents for Telecommunication Applications*, pages 244–253, Floianópolis, Brazil, 2004.

[54] S.S. Eswari Dei and V. Ramachandran. Agent based control for embedded applications. In *9th International Conference on High Performance Computing (HiPC 2002)*, Bangalore, India, December 2002.

[55] O. Depren, M. Topallar, E. Anarim, and M.K. Ciliz. An Intelligent Intrusion Detection System for Anomaly and Misuse Detection in Computer Networks. *Expert systems with Applications*, 29:713–722, 2005.

[56] P. Dokas, L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, and P. Tan. Data Mining for Network Intrusion Detection. In *Proceedings of the NSF Workshop on Next Generation Data Mining*, Baltimore, MD, November 2002.

[57] M. Dorigo and T. Stutzle. *Ant Colony Optimization*. The MIT Press, July 2004.

[58] C. Elkan. Magical Thinking in data Mining: Lessons from CoIL Challenge 2000. In *Proceedings of SIGKDD01*, pages 426–431, 2001.

[59] A. Elmzabi, M. Bellafkih, and M. Ramdani. An adaptive fuzzy clustering approach for the network management. *International Journal of Information Technology*, 3(1):1305–2403, 2006.

[60] D. Endler. Intrusion Detection: Applying Machine Learning to Solaris Audit Data. In *Proceedings of the Annual Computer Security Applications conference*, pages 267–279, 1998.

[61] R. F. Erbacher and D. Frincke. Visual Behavior Characterization for Intrusion and Misuse Detection. In *Proceedings of the SPIE Conference on Visual Data Exploration and Analysis*, pages 210–218, 2001.

[62] L. Ertoz, A. Lazarevic, E. Eilertson, A. Lazarevic, P. Tan, P. Dokas, V. Kumar, and J. Srivastava. Protecting Against Cyber Threats in Networked Information Systems. In *SPIE Annual Symposium on AeroSense, Battlespace Digitization and Network Centric Systems III*, Orlando, FL, April 2003.

[63] E. Eskin. Anomaly Detection over Noisy Data using Learned Probability Distributions. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 255–262, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[64] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. In *Data Mining for Security Applications*. Kluwer, 2002.

[65] B. S. Everitt. *Cluster Analysis*. Edward Arnold: A division of Hodder & Stoughton, New York, USA, 3 edition, 1993.

[66] E. Fabre, A. Benveniste, S. Haar, C. Jard, and A. Aghasaryan. Algorithms for Distributed Fault Management in Telecommunications Networks. In *Proceedings of the International Conference on Telecommunications*, pages 820–825. Springer, 2004.

[67] M. Fisk and G. Varghese. An Analysis of Fast String Matching Applied to Content Based Forwarding and Intrusion Detection. Technical report, University of California - San Diego, 2002.

[68] S. Forrest, A. Perelson, L. Allen, and R. Cherukuri. Self Nonself Discrimination in a Computer. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1994.

[69] D. Frincke. Balancing Cooperation and Risk in Intrusion Detection. *ACM Trans. Inf. Syst. Secur.*, 3(1):1–29, 2000.

[70] D. Frincke, J. McConnell, D. Tobin, J. Marconi, and D. Polla. A Framework for Cooperative Intrusion Detection. In *21st National Information Systems Security Conference*, pages 361–373, 1998.

[71] D. Frincke and E. Wilhite. Distributed Network Defense. In *IEEE Workshop on Information Assurance and Security*, United states Military Academy, West Point, NY, 2001.

[72] P. Fröhlich. *DRUM-II: Efficient Model-Based Diagnosis of Technical Systems*. PhD thesis, Universität Hannover, 1998.

[73] P. Fröhlich, W. Nejdl, M. Schroeder, C. Damsio, and L. M. Pereira. Using Extended Logic Programming for Alarm Correlation in Cellular Phone Networks. In *Proceedings of the International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 343–352. Springer, 1999.

[74] P. Frohlich, W. Nejdland, K. Jobmann, and H.Wietgrefe. Model Based Alarm Correlation in Cellular Phone Networks. In *Fifth Int. Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 1997.

[75] M. Garijo, A. Cancer, and J. J. Sanchez. A multiagent system for cooperative network-fault management. In *PAAM '96: Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-agent Technology*, pages 279–294, 1996.

[76] A. Ghosh and A. Schwartzbard. A Study in Using Neural Networks for Anomaly and Misuse Detection. In *Proceedings of the 8th USENIX Security Symposium*, 1999.

[77] A. K. Ghosh, C. Michael, and M. Schatz. A Real Time Intrusion Detection System Based on Learning Program Behaviour. In *Proceedings of the Third International Workshop on the Recent Advances in Intrusion Detection*, pages 93–109, 2000.

[78] J. C. Giarratano and G. D. Riley. *Expert Systems: Principles and Programming*. Course Technology, Boston, MA USA, 4 edition, 1998.

[79] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Co., Reading, Massachusetts, 1989.

[80] F.A. Gonzlez, J. Gmez, M. Kaniganti, and D. Dasgupta. An Evolutionary Approach to Generate Fuzzy Anomaly Signatures. In *IEEE Information Assurance workshop*, pages 251–259, West Point, NY, 2003.

[81] B. Gruschke. A New Approach for Event Correlation based on Dependency Graphs. In *Proceedings of the 5th Workshop of the OpenView University Association*, 1998.

[82] B. Gruschke. Integrated Event Management: Event Correlation using Dependency Graphs. In *Proceedings of the 9th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management*, 1998.

[83] J. Gu, D. Lee, S. Park, and K. Sim. An Immunity Based Security Layer Model. In *GECCO'00 Workshop on Artificial Immune Systems and Their Applications*, pages 47–48, 2000.

[84] D.W. Guerer, I. Khan, R. Ogler, and R. Keffer. An Artificial Intelligence Approach to Network Fault Management. In *SRI International*, Menlo Park, CA, 1996.

[85] J. M. Hall and D. A. Frincke. An Architecture for Intrusion Detection Modeled After the Human Immune System. In *Proceedings of the International Conference on Computer Communication and Control Technologies*, volume 3, pages 75–78, 2003.

[86] A. Hanemann. A Hybrid RuleBased/CaseBased Reasoning Approach for Service Fault Diagnosis. In *Proceedings of the 2006 International Symposium on Frontiers in Networking with Applications (FINA 2006)*, Vienna, Austria, 2006.

[87] A. Hanemann and M. Sailer. A Framework for Service Quality Assurance using Event Correlation Techniques. In *Proceedings of the International Conference on Service Assurance with Partial and Intermittent Resources (SAPIR 2005)*, Lisbon, Portugal, 2005.

[88] A. Hanemann, M. Sailer, and D. Schmitz. A Framework for Failure Impact Analysis and Recovery with Respect to Service Level Agreements. In *Proceedings of the IEEE International Conference on Services Computing (SCC 2005)*, Orlando, Florida, 2005.

[89] A. Hanemann, M. Sailer, and D. Schmitz. Towards a Framework for Failure Impact Analysis and Recovery with Respect to Service Level Agreements. In *Proceedings of the 9th IFIP/IEEE International Conference on Integrated Network Management (IM 2005)*, Nice, France, 2005.

[90] A. Hanemann, M. Sailer, and D. Schmitz. Towards a Framework for IT Service Fault Management. In *Proceedings of the European University Information Systems Conference (EUNIS 2005)*, Manchester, England, 2005.

[91] A. Hanemann and D. Schmitz. Service Oriented Event Correlation the MNM Service Model Applied to EMail Services. In *11th International Workshop of the HP OpenView University Association (HPOVUA 2004)*, Paris, France, 2004.

[92] A. Hanemann and D. Schmitz. Service Oriented Event Correlation Workflow and Information Modeling Approached. In *Third International Workshop on Distributed Event Based Systems (DEBS2004)*, Edinburgh, Scotland, 2004.

[93] M. Hasan. An Active Temporal Model for Network Management Databases. In A. S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors, *4th IEEE/IFIP International Symposium on Integrated Network Management*, pages 524–535. Chapman and Hall, 1995.

[94] M. Hasan, B. Sugla, and R. Viswanathan. A Conceptual Framework for Network Management Event Correlation and Filtering Systems. In A. S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors, *Integrated Network Management IV*, pages 233–246, 1999.

[95] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 2 edition, 1998.

[96] G. Helmer, J.S.K. Wong, V.Honavar, L. Miller, and Y. Wang. Lightweight Agents for Intrusion Detection. *The Journal of Systems and Software*, pages 109–122, 2003.

[97] C. Hilas and J. Sahalos. User Profiling for Fraud Detection in Telecommunication Networks. In *Proceedings of the 5th International Conference on Technology and Automation*, pages 382–387, Thessalonika, Greece, 2005.

[98] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[99] S. A. Hofmeyr and S. Forrest. Immunity by Design: An Artificial Immune System. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 1289–1296, 1999.

[100] S. A. Hofmeyr and S. Forrest. Architecture for an Artificial Immune System. *Evolutionary Computation*, 8(4):443–473, 2000.

[101] A. J. Hoglund and K. Hatonen. Computer Network User Behaviour Visualisation Using Self Organising Maps. In *Proceedings of the 8th International Conference on Artificial Neural Networks*, pages 899–904, 1998.

[102] H. Hou, J. Zhu, and G. Dozier. Artificial Immunity Using Constraint Based Detectors. In *Proceedings of the 5th Biannual World Automation Congress*, pages 239–244, 2002.

[103] W. S. Hu and H. Zhan. A Study of a Clustering Based Intrusion Detection Algorithm. *Postgraduate Journal, Wuhan University*, 21(3), 2004.

[104] J.-F. Huard. Probabilistic reasoning for fault management on xunet. Technical report, AT&T Bell Laboratories, Murray Hill, NJ, September 1994.

[105] Network ICE. Protocol Analysis vs Pattern Matching in Network and Host Intrusion Detection Systems, 2000.

[106] K. Jaguja, A. Sokolowski, and H. Bock, editors. *Classification, Clustering and Data Analysis* . Springer, 1 edition, August 15 2002.

[107] G. Jakobson, J. Buford, and L. Lewis. Towards and Architecture for Reasoning About Complex Event Based Dynamic Systems. In *Proceedings of the Third International Workshop on Distributed Event Based Systems (DEBS 2004)*, 2004.

[108] G. Jakobson and M. Weissman. Alarm correlation. *IEEE Network*, pages 52–59, 1993.

[109] G. Jakobson and M. Weissman. Real Time Telecommunications Network Management: Extending Event Correlation with Temporal Constraints. In

A.S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors, *4th IEEE/IFIP International Symposium on Integrated Network Management*, pages 290–301. Chapman and Hall Ltd, 1995.

[110] W. Jansen and T. Karygiannis. Applying Mobile Agents to Intrusion Detection and Response. Technical report, NIST Interim Report, October 1999.

[111] H. S. Javitz and A. Vadles. The NIDES Statistical Component: Description and Justification. *SRI International Computer Science Laboratory Technical Report CSL-93-12*, 1993.

[112] G. Jiang and G. Cybenko. Temporal and Spatial Distributed Event Correlation for Network Security. In *Proceedings of the American Control Conference*, volume 2, pages 996–1001, 2004.

[113] W. Jing-Xin, W. Zhi-Ying, and D. Kui. A Network Intrusion Detection System Based on Artificial Neural Networks. In *Proceedings of the 3rd ACM International Conference on Information Security*, volume 85, pages 166–170, Shanghai, China, 2004.

[114] J. F. Jordaan and M. E. Paterok. Event Correlation in Heterogeneous Networks Using the OSI Management Framework. In *Integrated Network Management III*, pages 683–695, San Francisco, CA, April 1993.

[115] K. Julisch. Clustering Intrusion Detection Alarms to Support Root Cause Analysis. *ACM Transactions on Information and System Security*, 6(4):443–471, 2003.

[116] O. Kachirski and R. Guba. Effective Intrusion Detection Using Multiple Sensors in Wireless Ad Hoc Networks. In *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, 2003.

[117] S. Kätker and K. Geihs. A generic model for fault isolation in integrated management system. *J. Network Syst. Manage.*, 5(2), 1997.

[118] S. Katker and M. Paterok. Fault Isolation and Event Correlation for Integrated Fault Management. In *Proceedings of the Fifth IFIP/IEEE International Symposium on Integrated Network Management (IM 97)*, pages 583–596, San Diego, California, May 1997.

[119] I. Katzela and M. Schwartz. Schemes for fault identification in communication networks. *IEEE/ ACM Transactions on Networking*, 3(6):753–764, 1995.

[120] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. A Wiley Interscience Publication, New York, USA, 1990.

[121] G. H. Kim and P. J. Bentley. An Evaluation of Negative Selection in an Artificial Immune System for Network Intrusion Detection. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, 2001.

[122] G. H. Kim and P. J. Bentley. Towards Artificial Immune Systems for Network Intrusion Detection: An Investigation of Dynamic Clonal Selection. In *Proceedings of the Congress on Evolutionary Computation*, 2002.

[123] T-K. Kim, D-Y Lee, O.-H. Byeon, and T-M. Chung. A Policy Propagation Model Using Mobile Agents in large Scale Distributed Network Environments. In *First International Conference on Service Oriented Computing (ISOC 2003)*, Trento, Italy, 2003.

[124] S. Klinger, S. Yemini, Y. Yemini, D. Ohsie, and S. Stolfo. A coding Approach to Event Correlation. In *Proceedings of the fourth international symposium on Integrated network management IV*, pages 266–277, London, UK, UK, 1995. Chapman & Hall, Ltd.

[125] C. Ko. Logic Induction of Valid Behavior Specifications for Intrusion Detection. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 14–17, Oakland, CA, May 2000.

[126] A. Kofod-Petersen and A. Aamodt. Case Based Situation Assessment in a Mobile Context Aware System. In *Proceedings of the Workshop on Artificial Intelligence in Mobile Systems*, Seattle, USA, 2003.

[127] T. Kohonen. *Self Organisation and Associative Memory*. Springer, 1989.

[128] A. P. Kosoresow and S. A. Homfmeyr. Intrusion Detection via System Call Traces. *IEEE Software*, 14:24–42, 1997.

[129] C. Krugel, T. Toth, and C. Kerer. Decentralized Event Correlation for Intrusion Detection. In *International Conference on Information Security and Cryptology*, 2001.

[130] G. Prem Kumar and P. Venkataram. Network performance management using realistic abductive reasoning model. In *Integrated Network Management*, pages 187–198, 1995.

[131] G. Prem Kumar and P. Venkataram. Ai approaches to network management: Recent advances and a survey. *Computer Communications*, 20:1313–1322, 1997.

[132] S. Kumar and E. H. Spafford. A Pattern Matching Model for Misuse Intrusion Detection. In *Proceedings of the 17th National Computer Security Conference*, 1994.

[133] K. Kuri, G. Navarro, L. M, and L. Haye. A Pattern Matching based Filter for Audit Reduction and Fast Detection of Potential Intrusions. In *Proceedings of the International Workshop on the Recent Advances in Intrusion Detection*, pages 17–27, Tolouse, France, 2000.

[134] T. Lane and C. E. Brodley. Temporal Sequence Learning and Data Reduction for Anomaly Detection. *ACM Transactions on Information and System Security*, 2(3):295–331, 1999.

[135] N. Lavrac and S. Dzeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York, 1994.

[136] D. Leake. CBR in Context: The Present and Future. In D. Leake, editor, *Case-Based Reasoning: Experiences, Lessons, and Future Directions*, Menlo Park, USA, 1996. AAAI Press/MIT Press.

[137] W. Lee and S. J. Stolfo. Data Mining Approaches for Intrusion Detection. In *Proceedings of the 7th USENIX Security Symposium*, San Antonio, Texas, January 1998.

[138] W. Lee and D. Xiang. Information Theoretic Measures for Anomaly Detection. In *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2001.

[139] J. Z. Lei and A. A. Ghorbani. Network Intrusion Detection Using an Improved Competitive learning neural Network. In *Conference on Communication networks and Services Research*, pages 190–197, Fredericton, Canada, 2004.

[140] E. Leon, O. Nasaoui, and J. Gomez. Anomaly Detection Based on Unsupervised Niche Clustering with Application to Network Intrusion Detection. In *Proceedings of the congress of evolutionary Computation*, 2004.

[141] K. Leung and C. Leckie. Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters. In *Proceedings of the 28th Australasian Computer Science Conference*, pages 333–342, Newcastle, Australia, 2004.

[142] L. Lewis. *Managing Business and Service Networks*. Kluwer, 2001.

[143] L. M. Lewis. A Case-Based Reasoning Approach to the Resolution of Faults in Communication Networks. In *Proceedings of the IFIP TC6/WG6.6 Third International Symposium on Integrated Network Management with participation of the IEEE Communications Society CNOM and with support from the Institute for Educational Services*, pages 671–682. North-Holland, 1993.

[144] W. Li. Using Genetic Algorithms for Network Intrusion Detection. In *Proceedings of the United States Department of Energy Cyber Security Group 2004 Training Conference*, Kansas City, USA, May 2004.

[145] Y. Liao and V. R. Vemuri. Using Text Categorization Techniques for Intrusion Dectection. In *Proceedings of the 11th USENIX Security Symposium*, pages 51–59, 2002.

[146] U. Lindqvist and P. A. Porras. Detecting Computer and Network Misuse Through the Production-Based Expert System Toolset (P-BEST). In *IEEE Symposium on Security and Privacy*, 1999.

[147] U. Lindqvist and P. A. Porras. eXpert-BSM: A Host Based Intrusion Detection Solution for Sun Solaris. In *Proceedings of the 17th Annual computer Security Applications Conference*, pages 240–251, New Orleans, USA, Dec 2001.

[148] G. Liu, A. K. Mok, and E. J. Yang. Composite Events for Network Event Correlation. In *Integrated Network Management III*, pages 247–260, Boston, MA, May 1999.

[149] C. C. Lo, S. H. Chen, and B. Y. Lin. Coding Based Schemes for Fault Identification in Communication networks. *International Journal of Network Management*, pages 157–164, 2000.

[150] M.E. Locasto, J.J. Parekh, A.D. Keromytis, and S.J. Stolfo. Towards Collaborative Security and P2P Intrusion Detection. In *IEEE Workshop on Information Assurance and Security*, West Point, NY, 2005.

[151] K.-W. E. Lor. A network diagnostic expert system for acculink multiplexers based on a general network diagnostic scheme. In *Integrated Network Management*, pages 659–669, 1993.

[152] G. F. Luger. *Artificial Intelligence*. Addison–Wesley, 5 edition, 2002.

[153] Albaghdadi M, B. Briley, M.W. Evens, R. Sukkar, M. Petiwala, and M. Hamlen. A Framework for Event Correlation in Communication Systems. In *Lecture Notes in Computer Science*, page 271, 2001.

[154] D. J. C. Mackay. *Information Theory, Inference and Learning Algorithms*. Springer, 2000.

[155] J. Marin, D. Ragsdale, and J. sirdu. A Hybrid Approach to Profile Creation and Intrusion Detection. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, pages 69–76, Los Alamitos, USA, 2001.

[156] E.P. Markatos, S. Antonatos, M. Polychronakis, and K.G. Anagnostakis. ExB: Exclusion Based signature Matching for Intrusion Detection. In *Proceedings of Communications and Computer Networks*, MIT, USA, 2002.

[157] F. J. Martín and E. Plaza. Ceaseless Case-Based Reasoning. In *Advances in Case-Base Reasoning, 7th European Conference on Case-Based Reasoning*, volume 3155 of *Lecture Notes in Computer Science*, pages 287–301. Springer, 2004.

[158] J.-P. Martin-Flatin. Distributed Event Correlation and Self–Managed Systems. In *Proceedings of the International Conference*, pages xxx–yyy. IEEE Press, 2000.

[159] D. M. Meira. *A Model For Alarm Correlation in Telecommunications Networks*. PhD thesis, Federal University of Minas Ferais, November 1997.

[160] C. Melchiors and L. M. Rockenbach Tarouco. Fault management in computer networks using case-based reasoning: Dumbo system. In *ICCBR*, pages 510–524, 1999.

[161] M. L. Minsky and S. A. Papert. *Perceptrons, An Introduction to Computational Geometry*. MIT press, expanded edition edition, 1969.

[162] B. Mirkin. *Mathematical Classification and Clustering*, volume 11 of *Nonconvex Optimization and its Applications*. Kluwer Academic Publishers, 3300 AA Dordrecht, The Netherlands, 1996.

[163] M. Mitchell. *An Introduction to Genetic Algorithms* . The MIT Press, Cambridge, Massachussets, reprint edition edition, February 6 1998.

[164] T. Mitchell. *Machine Learning*. McGraw–Hill, 1997.

[165] R. Mitra and J. Basak. Methods of Case Adaptation: A Survey. *International Journal of Intelligent Systems*, 20(5):627–645, 2005.

[166] B. Morin and H. Debar. Correlation of Intrusion Symptoms: An Application of Chronicles. In *Proceedings of the 6th symposium on Recent Advances in Intrusion Detection*, Carnegie Meoon, Pittsburg, 2003.

[167] B. Morin, L. Mé, H. Debar, and M. Ducassé. M2D2: A Formal Data Model for IDS Alert Correlation. In *RAID*, pages 115–127, 2002.

[168] F. Neri. Mining TCP/IP Traffic for Network Intrusion Detection by Using a Distributed Genetic Algorithm. In *Proceedings of the European Conference on Machine Learning*, pages 313–322, Barcelona, Spain, 2000.

[169] C. Nikolopoulos. *Expert Systems: Introduction to First and Second Generation and Hybrid Knowledge Based Systems*. 0824799275. Marcel Dekker Inc, New York, January 10 1997.

[170] M. Norton. Optimizing pattern matching for intrusion detection. Technical report, Columbia, Maryland, September 2004.

[171] N. Nuansri, T. S. Dillon, and S. Singh. An application of neural network and rule-based system for network management: Application level problems. In *HICSS (5)*, pages 474–483, 1997.

[172] Y. A. Nygate. Event Correlation Using Rule and Object Based Techniques. In A. S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors, *Integrated Network Management IV*, pages 278–289, Santa Barbara, CA, May 1995.

[173] T. Oates. Fault identification in computer network: A review and a new approach. Technical Report UM-CS-1995-113, Computer Science Department Lederle Graduate Research Center University of Massachusetts Amherst, MA 01003–4601, 1995.

[174] J. Oldmeadow, S. Ravinutala, and C. Leckie. Adaptive Clustering for Network Intrusion Detection. In *Proceedings of the 3rd International Pacific Asia Conference on Knowledge Discovery and Data Mining*, 2004.

[175] Z.S. Pan, S.C. Chen, G. Bao Hu, and D.Q. Zhang. Hybrid Neural Network and C4.5 for Misuse Detection. In *Machine Learning and Cybernetics*, pages 2463–2467. Xi'an, 2003.

[176] V. Paxson. Bro: A System for Detecting Network Intruders in Real Time. In *USENIX symposium*, 1998.

[177] S. Peddabachigari, , A.Abraham, C. Grosan, and J. Thomas. Modeling Intrusion Detection Systems Using Hybrid Intelligent Systems. *Journal of Network and Computer Applications*, 2005.

[178] S. Pierre and R. Glitho, editors. *Mobile Agents for Telecommunication Applications*. 3540424601. Springer, Third International Workshop, MATA 2001, 1 edition, August 14-16 2001.

[179] M.M. Pillai, J.H.P. Eloff, and H.S.Venter. An Approach to Implementing a network Intrusion Detection System using Genetic Algorithms. In *Proceedings of the Annual south African Institute of Computer Scientists and Information Technologists*, Stellenbosch, SA, 2004.

[180] Y. Ping, Y. Yan, H. Yafei, Z. Yiping, and Z. Shiyong. Securing ad hoc Networks Through Mobile Agents. In *Proceedings of the 3rd ACM International Conference on Information Security*, pages 125–129, Shanghai, China, 2004.

[181] L. Portnoy, E. Eskin, and S. Stolfo. Intrusion Detection With Unlabeled Data Using Clustering. In *Proceedings of the ACM Workshop on Data Mining Allied to Security*, 2001.

[182] M.J Ranum, L. Landfield, M.Stolarchuk, M.Sienkiewicz, A. Lambeth, and E. Wall. Implementing a Generalized Tool for Network Monitoring. In *Proceedings of the 11th Systems Administration Conference*, San Diego, USA, 1997.

[183] C. K. Riesbeck and R. C. Schank. Inside Case-based Reasoning. In *Technical Report*. Lawrence Erlbaum Association, Cambridge, 1989.

[184] I. Rish, M. Brodie, S. Ma, N. Odintsova, A. Beygelzimer, G. Grabarnik, and K. Hernandez. Adaptive diagnosis in distributed systems. *IEEE Transactions on Neural Networks*, 16(5):1088–1109, 2005.

[185] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing, Volume 1*. MIT Press, 1986.

[186] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, 1995.

[187] K. Sastry, H. A. Abbass, and D. E. Goldberg. Sub-Structural Niching in Non–Stationary Environments. In *Australian Artificial Intelligence Conference*, pages 873–885, 2005.

[188] M.M. Sebring, E. Shellhouse, M. Hanna, and R. Whitehurst. Expert Systems in Intrusion Detection: A Case Study. In *Proceedings of the 11th National Computer Security Conference*, October 1988.

[189] R. Sekar, M. Bendre, D. Dhurjati, and P. Bollinen. A Fast Automation Based Method for Detecting Anomalous Behaviours. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 144–155, 2001.

[190] K. Sheers. HP OpenView Event Correlation Services. *Hewlett–Packard Journal*, 11(1):31–42, 1996.

[191] C. Sinclair, L. Pierce, and S. Matzner. An Application of Machine Learning to Network Intrusion Detection. In *Proceedings of the Annual Computer Security Applications Conference*, pages 371–377, Phoenix, 1999.

[192] S. Snapp, J. Brentano, G. Dias, T. Goan, L. Heberlein, C. Ho, K. Levitt, B. Mukherjee, S. Smaha, T. Grance, D. Teal, and D. Mansur. DIDS (Distributed Intrusion Detection System) Motivation, Architecture and An Early Prototype. In *Proceedings of the National Information Systems Security Conference*, pages 167–176. NIST, Washington, DC, 1991.

[193] H. Song and J. W. Lockwood. Efficient Packet Classification for Network Intrusion Detection using FPGA. In *Proceedings of the 2005 ACM/SIGDA 13th International Symposium on Field Programmable Gate Arrays*, pages 238–245, Monterey, USA, 2005.

[194] H. Song and J. W. Lockwood. Multi Pattern Signature Matching for Hardware network Intrusion Detection Systems. In *Workshop 2nd IEEE International Workshop on Adaptive Wireless Networks*, St Louis, USA, November 2005.

[195] E. Soroush, M. S. Abadeh, and J. Habibi. Intrusion Detection Using a Boosting Ant Colony Based Data Miner. In *Proceedings of the 11th International CSI Computer Conference*, pages 563–566, 2006.

[196] F. Steimann, P. Fröhlich, and W. Nejdl. Model-based diagnosis for open systems fault management. *AI Communications*, 12(1-2):5–17, 1999.

[197] M. Steinder and A. S. Sethi. *Non-deterministic Event-driven Fault Diagnosis through Incremental Hypothesis Updating*, pages 635–648. Colorado Springs, Co, 2003.

[198] M. Steinder and A. S. Sethi. A survey of fault localization techniques in computer networks. *Sci. Comput. Program.*, 53(2):165–194, 2004.

[199] Malgorzata Steinder and Adarshpal S. Sethi. The Present and Future of Event Correlation: A Need for End to End Service Fault Localization. In *Proceedings of the 2001 World Multi–Conference on Systematics, Cybernetics and Informatics*, 2001.

[200] R. Sterritt and D. W. Bustard. Fusing Hard and soft Computing for Fault Management in Telecommunication Systems. *IEEE Transactions On Systems, Man and Cybernetics*, 32(2), May 2002.

[201] S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan. Cost-based Modeling for Fraud and Intrusion Detection: Results from the JAM Project. In

Proceedings of the DARPA Information Survivability Conference and Exposition, pages 130–144. IEEE Computer Press, 2000.

[202] P. N. Tan, M. Stenbach, and V. Kumar, editors. *Introduction to Data Mining*. Addison Wesley, 1 edition, May 2005.

[203] Y. Tang, E. S. Al-Shaer, and R. Boutaba. Active integrated fault localization in communication networks. In *IM '05: 9th IFIP/IEEE International Symposium on Integrated Network Management*, pages 543– 556, May 2005.

[204] M. Taniguchi, M. Haft, J. Hollmn, and V.Tresp. Fraud Detection in Communications Networks Using Neural and Probabilistic Methods. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1241–1244, Seattle, USA, May 1998.

[205] G. Tedesco and U. Aickelin. Adaptive Alert Throttling for Intrusion Detection Systems. *Journal of Intelligent Information Systems*, 2003.

[206] E. Totel, B. Vivinis, and L. Me. A Language Driven Intrusion Detection System for Event and Alert Correlation. In *Proceedings of the IFIP International Information Security Conference*, August 2004.

[207] C. H. Tsang and S. Kwong. Multi Agent Intrusion Detection System in Industrial Networks Using Ant Colony Clustering Approach and Unsupervised Feature Extraction. In *Proceedings of the IEEE International Conference on Industrial Technology*, pages 51–56, 2005.

[208] E. Turban, J. E. Aronson, and T. P. Liang. *Decision Support Systems and Intelligent Systems* . Prentice Hall, 7 edition, April 8 2004.

[209] R. Vaarandi. Sec - A Lightweight Event Correlation Tool. In *Proceedings of the IEEE Workshop on IP Operations & Management*, Dallas, USA, 2002.

[210] G. Vert, J. McConnell, and D. Frincke. Towards a Mathematical Model for Intrusion. In *Proceedings of the National Information Systems Security Conference*, pages 329–337. NIST, Washington, DC, 1998.

[211] R. Verwoerd and R. Hunt. Intrusion Detection Techniques and Approaches. *Computer Communications*, 25(15):1356–1365, September 2002.

[212] G. Vigna and R. A. Kemmerer. NetSTAT: A Network Based Intrusion Detection System. *Journal of Computer Security*, 7(1):37–71, 1999.

[213] D. Wagner and D. Dean. Intrusion Detection via Static Analysis. In *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, USA, 2001.

[214] C. Wang and M. Schwartz. Fault Detection with Multiple Observers. *IEEE/ACM Trans. Netw.*, 1(1):48–55, 1993.

[215] C. Warrender, S. Forrest, and B. Pearlmutter. Detecting Intrusions Using System Calls: Alternative Data Models. In *Proceedings of the IEEE Symposium on Security and Privacy*, 1999.

[216] G. Weiss, J. Eddy, and S. Weiss. Intelligent Telecommunication Technologies. In L. Jain, R. Johnson, Y. Takefuji, and L. Zadeh, editors, *Knowledge–Based Intelligent Techniques in Industry*, pages 251–275. CRC Press, 1999.

[217] A. Wespi, M. Dacier, and H. Debar. An Intrusion Detection System Based on the Teiresias Pattern Discovery Algorithm. In *Proceedings of the EICAR Conference*, Copenhagen, Denmark, May 1999.

[218] H. Wietgrefe, K-D. Tuchs, K. Jobmann, G. Carls, P. Frhlich, W. Nejdl, and S. Steinfeld. Using Neural Networks for Alarm Correlation in Cellular Phone Networks. In *International Workshop on Applications of Neural Networks to Telecommunications*, 1997.

[219] P. Wu, R. Bhatnagar, L. Epshtein, M. Bhandaru, and S. Zhongwen. Alarm Correlation Engine (ACE). In *IEEE/IFIP Network Operation and Management Symposium*, volume 3, pages 733–742, 1998.

[220] S. Wu and U. Manber. A Fast Algorithm for Multipattern Searching. Technical report, University of Arizona, USA, 1994.

[221] T. Yairi, Y. Kato, and K. Hori. Fault Detection by Mining Association Rules from Housekeeping Data. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space*, Quebec, Canada, 2001.

[222] P. W. Yaner and A. K. Goel. Visual Case Based Reasoning: Memory Retrieval. In *Proceedings of the Joint International Conference on Artificial Intellignece*, Hyderbad, India, 2003.

[223] D. G. Yang, C. Y. Hu, and Y. H. Chen. A Framework of Cooperating Intrusion Detection Based on Clustering Analysis and Expert System. In *Proceedings of the International Conference on Information Security*, Shanghai, China, November 2004.

[224] N. Ye and X. Li. A scalable clustering technique for intrusion signature recognition. In *The 2001 IEEE Workshop on Information Assurance and Security*, United States Military Academy, West Point, NY, June 2001. IEEE Publications.

[225] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. High Speed and Robust Event Correlation. *IEEE Communications Magazine*, 34(5):82–90, 1996.

[226] Q. Yin, R. Zhan, and X. li. A New Intrusion Detection Method Based on Linear Prediction. In *Proceedings of the International Conference on Information Security*, Shanghai, China, 2004.

[227] Y. Yu, G. Fu-xiang, and Y. Ge. Hybrid BP/CNN Neural Network for Intrusion Detection. In *Proceedings of the International Conference on Information Security*, Shanghai, China, 2004.

[228] D. Zerkle, S. Manganaris, M. Christensen, and K. Hermiz. A Data Mining Analysis of RTID Alarms. In *Recent Advances in Intrusion Detection*, Netherlands, 1999. Elsevier.

[229] S. Zhicai, J. Zhenzhou, and H. Mingzeng. A Novel Distributed Intrusion Detection Model Based on Mobile Agent. In *Proceedings of the International Conference on Information Security*, Shanghai, China, 2004.

[230] X. Zhuang, T. Zhang, and S. Pande. HIDE: an infrastructure for efficiently protecting information leakage on the address bus. In *ASPLOS-XI: Proceedings of the 11th international conference on Architectural support for programming languages and operating systems*, pages 72–84, New York, NY, USA, 2004. ACM Press.

Index