CHAPTER 2

REVIEW OF LITERATURE

2.1 Introduction

The modern information era depends on network enhancement process for all its applications. The network use and its user are based on the secure services provided in the network, as well as the related services. A secure network facilitates the right access, and right information to the right users. It must provide data confidentiality, integrity and data availability. Intrusion is an illegal act of entering, seizing, or taking control of another's property (computer system). It refers to a code that affects the proper flow of traffic on the network, or steals the information from the traffic. Intrusion detection provided the auditing of system configurations and vulnerabilities, assessing the integrity of the critical system and data files, and the statistical analysis of activity patterns based on the matching to known attacks [103]. This chapter provides the review of the NIDS process and its developments.

Network Intrusion detection systems gather information from a computer or network of computers, and attempt to detect intruders or system abuse, and newer systems take active steps to stop an intruder at the time of detection, approach carried out by different researchers[2,3,9,10,11,12,88,26,32,59,60,67]. The IDS is implemented in observing traffic for all the hosts on the network connected to an Ethernet, the auditing of operating system events, and the evaluation by the participants of the information about the file system state [49].

According to the resource attack, intruders can be divided into two groups, external and internal [48]. Intrusion techniques may include exploiting software bugs and improper system configurations, password cracking, sniffing unsecured traffic, or exploiting the design flaw of specific protocols [44, 13,65].
IDS can also be divided into two groups, depending on where they look for intrusive behavior: Network-based IDS (NIDS) and Host-based IDS [90]. Emerging techniques known as blocking IDS, combine host-based IDS with the ability to modify the firewall rules [66].

The IDS is classified into three main categories [47] as follows: 1) signature or misuse based IDS, 2) anomaly based IDS, and 3) specification based IDS, which is a hybrid of both signature and the anomaly based IDS. The classification of the IDS is shown in Figure 2.1, given below.

![Classification of the IDS System](image-url)

*Figure 2.1. Classification of the IDS System*
The signature-based IDS uses pre-known attack scenarios (or signatures), and compares them with incoming packet traffic. There are several approaches in signature detection, which differ in the representation and matching algorithm employed, to detect the intrusion patterns. The detection approaches, such as an expert system [62][93], pattern recognition [34], colored petri nets [51], and state transition analysis [76] are grouped on the misuse. This detection uses several techniques, i.e.: statistics [75], neural networks [27], and other techniques such as data mining [16, 17, 54,38 ], clustering[85],Chi-square test utilization [104] and taxonomy of wired IDS, as presented by Debar [28].The following section provides the components of the Intrusion Detection system and the different approaches to detect the same.

2.2 Components and types of IDS

Intrusion Detection could be achieved at three levels on the networks. The Network Intrusion Detection system (NIDS) – performs an analysis for the passing traffic on the entire subnet, and matches the traffic that is passed on the subnets to the library of knows attacks. Once the attack is identified, or any abnormal behavior is sensed, an alert can be sent to the administrator [89]. The Network Node Intrusion Detection System (NNIDS) performs the analysis of the traffic that is passed from the network to a specific host. The Host Intrusion Detection System (HIDS) – takes a snap shot of the existing system files and matches it with the previous snap shot [105].

Intrusion detection approaches are carried out in anomaly and misuse / signature detection. In anomaly detection, patterns are detected in a given data set, that is not conventional to an established normal behavior [94]. Anomalies are often referred to as outliers, unusual, divergence etc. It refers to storing the features of the user’s usual behavior into a database, then comparing it with the user’s current behavior. If a huge deviation occurs, it is said that the data tested is abnormal It lies in
the complete insignificance of the system, its strong flexibility and the likelihood to
detect the attack that was never detected before.

A network intrusion detection system (NIDS) monitors the packets that
traverse a given network link. Such a system operates by placing the network
interface in the promiscuous mode, according to the advantage of being able to
monitor an entire network, while not divulging its existence to potential attackers.
Because the packets of a NIDS is monitoring are not actually addressed to the host,
the NIDS resides on; the system is also impervious to an entire class of attacks such as
the “ping-of-death” attack that can disable a host without ever triggering a HIDS. A
NIDS is obviously of little value in detecting attacks that are launched on a host
through an interface other than the network.

Network data has a variety of characteristics that are available for a NIDS to
monitor: most of them operate by examining the IP and transport layer headers of
individual packets, the content of these packets, or some combination thereof.
Regardless of which characteristics a system chooses to monitor, however, the
positioning of a NIDS fundamentally presents a number of challenges to its correct
operation [98].

On a heterogeneous network, a NIDS generally does not possess intimate
knowledge of all the hosts on the network, and is incapable of determining how a host
may interpret packets with ambiguous characteristics. Without explicit knowledge of
a host system's protocol implementation, The NIDS is impotent in determining how a
sequence of packets will affect that host, if different implementations interpret the
same sequence of packets in different ways.

The Misuse/Signature detection technique deals with identifying and storing
signatures of known intrusions, and then comparing the activities occurring on an
information system to these signatures, so as to detect whether the system is undergoing an attack.

Terminologies in Intrusion detection system (IDS) Alert/Alarm a signal show, which system has been or is being attacked. True Positive is a genuine attack, which triggers the IDS to produce an alarm and show that an attack has taken place. False Positive is an event generated, to signal IDS to produce an alarm when no attack has taken place. Intrusion Detection Systems (IDS) have become a standard component in security infrastructures, as they allow network administrators to detect policy violations, ranging from external attackers trying to gain unauthorized access, to insiders abusing their access. Current IDS are usually tuned to detect known service level network attacks. This leaves them vulnerable to original and novel malicious attacks.

Data mining can help improve intrusion detection by addressing each and every one of the above mentioned problems. The selection of features [19] from the available data is essential for the effectiveness of the methods employed. Researchers apply various analytic procedures to the accumulated data, in order to select the set of features that they think maximizes the effectiveness of their data mining techniques. The following section of this chapter provides a view of data mining, and describes the technical approach to detect intrusions.

Sinclair [88] introduced the concept of the GA to detect IDS. As per the researcher’s concept, Genetic algorithms [4][61] can be used to evolve simple rules for network traffic [86]. These rules are used to differentiate normal network connections from anomalous connections. These anomalous connections refer to events with the probability of intrusions.

Network data has a variety of characteristics that are available for a NIDS to monitor: most of them operate by examining the IP and transport layer headers of
individual packets, the content of these packets, or some combination thereof. Protocol ambiguities can also present a problem to a NIDS in the form of crud. Crud appears in a network stream from a variety of sources, including erroneous network implementations, faulty network links, and network pathologies that have no connection to intrusion attempts [64].

2.3. IDS Techniques

This research examines the two basic types of IDS (HIDS and NIDS). For each of the two types, there are four basic techniques used to detect intruders: anomaly detection, misuse detection (signature detection), target monitoring and stealth probes.[97]. The anomaly and misuse detections have been described in the previous chapter. The target monitoring is presented below.

Target Monitoring: These systems do not actively search for anomalies or misuse, but instead look for the modification of specified files. This is more of a corrective control, designed to uncover an unauthorized action after it occurs, in order to reverse it. One way to check the covert editing of files is by computing a cryptographic hash beforehand and comparing this to the new hashes of the file at regular intervals. This type of system is the easiest to implement, because it does not require constant monitoring by the administrator. Integrity checksum hashes can be computed at whatever interval, and on either all the files or just the mission/system critical files.

Stealth Probes: This technique attempts to detect any attackers that choose to carry out their mission over prolonged periods of time. Attackers, for example, will check for system vulnerabilities and open ports over a two-month period, and wait another two months to actually launch the attacks. Stealth probes collect a wide-variety of data throughout the system, checking for any methodical attacks over a long period of time. They take a wide-area sampling and attempt to discover any
correlating attacks. In effect, this method combines anomaly detection and misuse detection, in an attempt to uncover suspicious activity [50].

2.4 Intrusion Detection Issues

Individual systems take differing approaches to the problem of intrusion detection. There exist, however, a number of common issues that plague the range of detection strategies. This section examines a number of these issues and some of the ways in which researchers have attempted to ameliorate them. As previously mentioned, one challenge of misuse detection systems is their reliance on an expert rule set, that traditionally must be constructed by a human domain expert. This rule set is expensive to produce and susceptible to human error [52].

Snort, a real-time NIDS, addresses this conundrum by minimizing the effort required to develop new attack rules. In Snort, each attack rule is a single line of text that specifies exactly which characteristics of a packet are to be examined, and what values these characteristics must equal, in order to trigger the rule [44]. This approach to the problem is helpful, but is limited in its ability to detect variations of codified attacks, and does not resolve the issue of requiring a human expert to devise a knowledge base for the IDS.

A technique that allows for the automated construction of attack rules based on a model of the human immune system, is ARTIS [96]. It achieves one of the benefits of anomaly systems, in the ability to detect new attacks without relying on a human-encoded expert knowledge base. One of the critical challenges of such a system would be the proper encoding of an attack into an appropriate attack rule with the proper characteristics. To date, there has been little exploration of this problem space.
2.4.1 Detecting Attack Variations

Detecting subtle variations of known attacks presents a sizeable challenge to many systems that rely upon misuse detection. Because of the way intrusion signatures must be codified into an expert knowledge base, it can be very difficult for misuse systems to identify attacks that may originate from more than one source, and vary in the means by which they are conducted, or are protracted over long periods of time. The State Transition Analysis (STAT) [76] is one technique that addresses this issue. In such a system, attacks are represented by a state transition diagram: the start state represents a pristine system, intermediate states represent changes to the system that occur during an attack, and the final state represents a system compromise. Through this means, countless variations of a single attack can still be detected, because the system monitors system state changes that are symptomatic of an intrusion attempt, rather than monitoring the actions that cause those state changes. The efficiency of such a system would be rapidly eroded in a production environment, in which an attacker could render the IDS useless by overwhelming the analysis engine.

2.4.2 Training Behavioral Models

Anomaly systems universally suffer from the problem of how to correctly construct a baseline model of behavior, which is sufficient for the complete and correct operation of the system. Any successful means of training the system must expose it to the full range of normal behavior, in order to minimize the false positives as well as avoid exposing the system to properties of anomalous activity that may desensitize the IDS to attacks. Depending on the design of the anomaly system, systems can be trained with just normal data, or with two sets of data that are correctly identified as normal and intrusive. In presenting intrusive data that is used to train the system, it is important that this data represents a range of anomalies, so that the trained system is not incapable of identifying certain classes of attack.
Ghosh et al. [41] have experimented, using randomly generated events to represent anomalous behavior, in order to train the neural networks that provide the analysis for their IDS. In empirical tests, those networks that were trained with randomly generated anomalous data consistently out-performed those that did not receive this training; by reducing the number of false negatives in the system (none of the networks produced any false positives). While these approaches are very promising, they suggest that the neural networks trained with random data performed well. The normal data set with which they were trained defined a narrow range of behavior that very closely resembled the normal data that the system was tested against. In such a case, the system could reasonably have been expected to generate at least a small number of false positives as some of the randomly generated events trained as anomalies would fall into the range of normal activity. It is further unclear how well a system trained over random anomalous data would perform in correctly identifying actual attacks, as the attack data against which this system was tested also consisted of randomly generated events [56].

A more ideal solution to the training problem would be one that allows for an anomaly system to be correctly trained over noisy data, i.e., data that contains an assortment of both normal and anomalous behavior. This would allow the system to be effectively trained in a production environment without relying on hypothetical data sets representing normal and anomalous behavior in the network [46].

Eleazar Eskin [32, 33] has developed a process that uses learned probability distributions to train an anomaly system over noisy data. This technique uses machine learning to create a probability distribution of the training data and then applies a statistical test to identify anomalies [69]. Interestingly, Eskin's technique [33] requires no domain specific knowledge. It does, however, operate on three assumptions about the training data: normal data can be effectively modeled using a probability
distribution; anomalous events differ significantly enough from normal events that they can be identified; and the number of anomalous events is compared to the number of normal events. Furthermore, before the system is trained, one must define a value indicating the percentage of the training data that is expected to be anomalous. Because this data is noisy and not artificially constructed, choosing the best value ensures correct operation of the system is very difficult. This model is additionally limited by its assumption that normal data is distributed normally across noisy data: in actuality, it is likely that intrusion attempts are temporally clustered. Regardless of the means by which one is trained, there is also the issue of evolving normal behavior with which anomaly systems must contend. One option is that a system be retrained periodically with new training data that represents current normal behavior. This, however, increases the dependency of the system on training and underscores the inherent difficulties in developing sufficient data or an appropriate technique for this task. Adaptive anomaly systems have been suggested as a solution, which would allow for systems to evolve their normal behavior models gradually, as normal behavior evolves [55].

Lane and Bradley’s [53] machine learning system is an example of a system that makes use of this technique. This system maintains a finite-sized dictionary of normal event sequences, and uses a Least-Recently Used (LRU) policy, to replace seldom-occurring sequences with new ones that are determined to be normal. It is important to note that systems which use adaptive training techniques face the problem of preventing an attacker from gradually training the system over time, to accept a range of anomalous behavior as normal. Resolving this difficulty remains an open challenge.
2.5 Attacks against the IDS

While the purpose of an intrusion detection system is to detect attacks against a host or set of hosts, an ironic consequence of its existence is that, the IDS itself may draw an attack from an attacker seeking to disable the IDS. It is critical that the design of a system be performed within the framework, that the IDS itself should be resistant to and tolerant of attack attempts, designed to obstruct its ability to correctly detect intrusions. One class of such attacks referred to as “crash attacks”, attempts to disable an IDS by causing it to fault or to run out of some critical resource. Assuming that it is infeasible to totally prevent these attacks, the goal of the IDS in the face of such an attack, is to minimize the extent to which the attacker is successful in disabling the IDS.

Bro [73], a real-time system for detecting network intrusions, provides two mechanisms for maximizing operation in the face of crash attacks. First, Bro maintains a “watchdog” timer that expires in a configurable interval, and checks to see if the system is still analyzing the same event that it was, when the previous timer expired. If this is the case, the system assumes that it is in a processing jam and terminates the monitoring process, so that the system can continue with the next event. Second, Bro [72] is launched from a script that can recognize if the system ever crashes, in which case it launches TCP dump in order to gather the data. This data can then be analyzed at a later time or by another system.

Vern Paxson [73] suggests that such systems perform “triage” against incoming flows: if the system detects that it is nearing exhaustion, it can shed the load by discarding the state for monitored flows that do not appear to be making progress. This suggestion operates under the assumption that an attacker is less likely to have complicity from hosts on both sides of the monitor, therefore making it difficult for the attacker to fake a large number of active connections.
The idea of triage is a precarious one. On the one hand, the load that is shed during triage can allow for the IDS to operate continuously, helping to maximize the coverage of the system and prevent an attacker from denying service to the system by overwhelming it with illegitimate data. On the other hand, when a system enters a mode of triage, it is in essence performing a denial of service on itself. The efficiency of a triage mechanism, therefore, hinges on the system's ability to properly determine which data it can safely ignore and which data it cannot: if the system were able to make this distinction perfectly, however, there would never be any need to examine the irrelevant data. As with the survival facilities that Bro [72] employs, the adoption of triage involves a tradeoff between correctness and performance, but is certainly preferable to the absence of such mechanisms, which may sacrifice both.

The current state of the art in intrusion detection restricts the measurement of new systems to tests over incomplete data sets, and micro benchmarks that can test a narrowly defined component of the system. Presently, a number of anomaly-based systems are tested over contrived data sets in order to determine how well the system classifies anomalies. This evaluation is limited by the quality of the data set that the system is measured against: constructing data sets that are both realistic and comprehensive is an extremely hard and open problem.

The intrusion detection domain is exceptionally challenging and expensive, assurance presents a way in which systems can be measured, that allows fuzzy decisions [84], trade off, and priorities as long as these properties are accompanied by appropriate assurance arguments. There are a number of unresolved issues regarding the scope of analysis that an IDS performs, and the interoperability of intrusion detection systems. Most intrusion detection efforts today focus on providing an analysis for a relatively localized target: either a single host or a collection of hosts joined by a network. A system that operates with a more global scope may be capable
of detecting distributed attacks or those that affect an entire enclave [39]. The development of such a system would be a valuable contribution to the study of intrusion detection. Table 2.1 given below illustrates the characteristics of NIDS and HIDS.

<table>
<thead>
<tr>
<th>S.No</th>
<th>Characteristics</th>
<th>NIDS</th>
<th>HIDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>Network Monitoring</em>[72]</td>
<td>Monitor the whole network or any subsets of the network from one location.</td>
<td>Monitor Individual Node.</td>
</tr>
<tr>
<td>2</td>
<td><em>Network Attack</em>[46]</td>
<td>It can monitor and detect Network attacks (e.g., probes, scans, malicious and anomalous activity across the whole network.)</td>
<td>It can detect elevated privileges attacks.</td>
</tr>
<tr>
<td>4</td>
<td><em>Accessibility</em>[75]</td>
<td>It becomes “invisible” for access.</td>
<td>Can detect critical data access and modification.</td>
</tr>
<tr>
<td>5</td>
<td><em>Detection</em>[85]</td>
<td>Detect Network Activity</td>
<td>Detect the Host Activity.</td>
</tr>
<tr>
<td>6</td>
<td><em>Protocol analysis</em>[18][25]</td>
<td>Cannot scan protocols or content if network traffic is encrypted.</td>
<td>It scans protocols or content if network traffic is encrypted.</td>
</tr>
<tr>
<td>7</td>
<td><em>Hi Speed Network</em>[63][90]</td>
<td>It monitors and detects on modern Switched network.</td>
<td>Can interfere with implemented service activities running in the host.</td>
</tr>
<tr>
<td>8</td>
<td><em>Packet Loss</em>[75]</td>
<td>It can lose some packet when working in high speed network.</td>
<td>Detect all the packets.</td>
</tr>
<tr>
<td>9</td>
<td><em>Data Confidence</em>[90]</td>
<td>Trust with the server report</td>
<td>Can not totally trust the host Information, once the machine is compromised.</td>
</tr>
</tbody>
</table>

*Table 2.1 Characteristics of NIDS and HIDS*
From the above table, it is understood that the unsolved issues are more in NIDS than HIDS. Hence the research work carried out in NIDS.

2.6 Summary

There have recently been a number of efforts including the Common Intrusion Detection Format (CIDF) and the Internet Engineering Task Force (IETF) standardization effort, motivated towards providing interoperability among intrusion detection systems. Such frameworks can provide the means by which differing analyses. The data collection techniques can be aggregated within a system, improving both the coverage and redundancy of the system. An increasing number of intrusion detection systems, such as EMERALD are beginning to make use of this idea, although it will likely be some time before a standard framework finds its way into widespread use. As part of the identified issue the protocol standardization approach IDS is initiated, using the genetic approach.