

Chapter 2

A Survey and Taxonomy of Classifiers of Intrusion Detection Systems

Tarfa Hamed, Jason B. Ernst, and Stefan C. Kremer

2.1 Introduction

Nowadays, the Internet is experiencing many attacks of various kinds that put its information under risk. Therefore, information security is currently under real threat as a result of network attacks [40]. Therefore, to overcome the network attacks, intrusion detection systems (IDS) have been developed to detect attacks and notify network administrators [16]. The IDSs are now being studied widely to provide the defense-in-depth to network security framework. The IDSs are usually categorized into two types: *anomaly detection* and *signature-based detection* [40]. Anomaly detection utilizes a classifier that classifies the given data into normal and abnormal data [34]. Signature-based detection depends on an up-to-date database of known attacks' signatures to detect the incoming attacks [40]. **Network Intrusion Detection Systems (NIDS)** are considered as classification problems and are also characterized by large amount of data and numbers of features [44].

In recent years, Internet users have suffered from many types of attacks. These cyber attacks are sometimes so damaging and cost billions of dollars every year [28]. Some of these attacks were able to access sensitive information and reveal credit cards numbers, delete entire domains, or even prevent legitimate users from being served by servers such as in the case of denial-of-service (DoS) attacks. The most common type of Internet attack is intrusion. These days, the most popular Internet services are being attacked by many intrusion attempts every day. Therefore,

T. Hamed (✉) • S.C. Kremer
School of Computer Science, University of Guelph, Guelph, ON, Canada
e-mail: tarafayaseen@gmail.com; skremer@uoguelph.ca

J.B. Ernst
Left Inc., Vancouver, BC, Canada
e-mail: jason@left.io

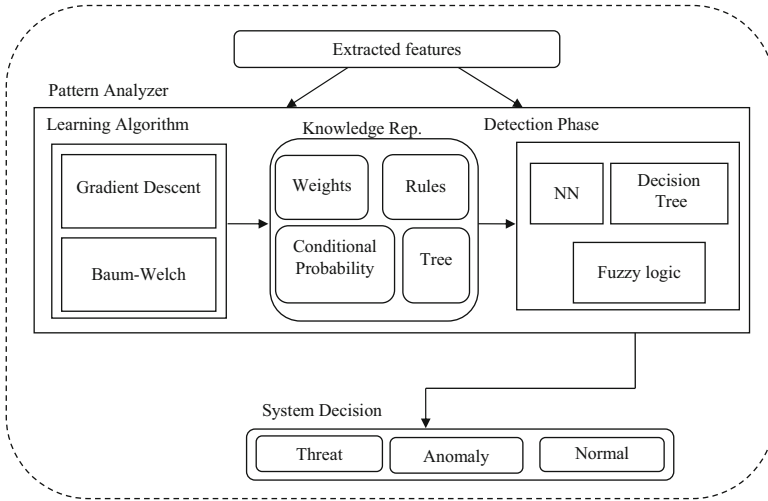


Fig. 2.1 The IDS components covered in this chapter

designing a robust system to detect against cyber attacks has become a necessity that needs the collaborations from all individuals.

The chapter is organized as follows: in Sect. 2.2 we explain the extracted features that result from the pre-processing phase. Next, the different IDS pattern analyzers are presented in detail in Sect. 2.3 with the knowledge representation used by those learning algorithms and the classifier systems. In Sect. 2.4 we present the decision making component of the IDS. The algorithms used in the detection phase produce different system decisions and are explained in this section. The system decision component with some details is presented in Sect. 2.5. The remaining parts of the IDS framework are beyond the scope of this chapter. Section 2.6 presents the conclusions of the chapter in addition to the open issues. We also provided a useful comparison and some critiques at the end of each component. The IDS components covered in this chapter are shown in Fig. 2.1.

2.2 Extracted Features

The pre-processing phase produces some patterns (sets of features) that would be suitable for the pattern analyzer, and the classification phase. These patterns are of different types (integer, float, symbols, etc.) according to the learning algorithm used. In [32], the resulting features are the statistical properties of packet sequences after converting them into statistical properties related to the transitions of the state machine.

In [27], the produced patterns represent the signature generated from the pre-processing phase. The algorithm presented in [24] calculates the empirical probability of a token appearing in a sample (whether it is *malicious* or *normal*). In [6], the extracted features included normal behaviors, audit data from ten users which have been collected for users who performed several types of actions such as programming, navigating Web pages, and transferring FTP data over the course of 1 month.

Now, having explained the extracted features resulting from the pre-processing phase and their types, we will explain the pattern analyzer of the system in the next section.

2.3 Pattern Analyzer

The next step is to use a suitable classifier to categorize the resulting extracted features from the pre-processing phase into one of threat, anomaly or normal data. In this section, the aim is to explain how the pattern analyzers work. In later sections, when discussing the classifiers, comparisons are given between the performances of various approaches after defining the typical metrics used to evaluate them. Some classifiers of intrusion detection systems deal with a user profile and behavior, therefore they use machine learning to learn the user profile in order to compare this profile later with the observed behavior to detect anomalies [22].

However, some other intrusion detection classifiers do not use any learning algorithm in making the final decision [43].

In general, a classification system can be viewed as consisting of three major components:

1. A decision making component, which ultimately classifies the data coming from the preceding phase,
2. A knowledge representation component, which incorporates information gathered from example data and informs the decision making component, and
3. An optional learning algorithm which is used to generate the knowledge representation of the previous component.

However, the chronological order of the above components is just the opposite, but we want here to start with the decision making component since the main objective of this phase is the detection process which is done by the decision making component. In addition, the decision making component needs a knowledge representation to make its decision, and to generate the knowledge representation, a learning algorithm is required to perform this task. The next sections will explain each part in details.

2.3.1 Learning Algorithms

In order to utilize the resulting features from the pre-processing phase for detecting intrusions, it is desirable to use a learning algorithm to learn from this data and later to use it to detect the intrusions. Learning algorithms are different in terms of the used input data whether they are labeled, un-labeled, and the type of the features. Some datasets like KDD Cup 99 contain labeled data either normal or attack (with only one specific attack type) for training and testing purposes, while some other datasets do not label their data. Researchers have been using several kinds of learning algorithms for intrusion detection purposes. In this context, several learning algorithms are discussed: *gradient descent*, *Baum–Welch algorithm*, *learning statistical properties*, *Genetic Network Programming*, and *some other machine learning algorithms*.

2.3.1.1 Gradient Descent Algorithm

Neural networks are one of the active approaches in building a learning system for detecting intrusions. In [22], the researcher has used back-propagation as a learning algorithm to train the network on the input data and use it to classify the test data. **Back-Propagation (BP)** is an algorithm used to train multi-layer, feed-forward, and supervised neural network. In this approach, the network is trained on different types of attacks and normal data to make it able to detect different attacks. Finding the optimal weights of the networks is accomplished by applying conjugate gradient descent algorithm. The host-based intrusion detection system is another type of intrusion detection system which collects input data from the host being monitored. The model proposed in [17] was used to detect both anomaly and misuse intrusions by incorporating two approaches: log file analysis and (BP) neural network. The researcher proposed a host-based intrusion detection system using a (BP) neural network to detect anomaly and misuse intrusions. The BP network was trained on the mentioned values to construct a user profile using a multi-layer neural network in anomaly detection [17].

2.3.1.2 Baum–Welch Algorithm

The **Hidden Markov Model (HMM)** is another technique used in intrusion detection. In [6], an HMM is used to model sequence information regarding system tasks, in order to minimize the false-positive rate and maximize the detection rate for anomaly detection. Usually, to estimate the parameters for an HMM, a standard Baum–Welch algorithm with the maximum-likelihood estimation (ML) criterion is used. The researcher used the Baum–Welch algorithm for HMMs since it is simple, well-defined, and stable [6].

2.3.1.3 Learning Statistical Properties

This approach focuses on unusual behavior to detect anomalies, so the approach needs to learn the frequency of making a transition from a state representing normal behavior to a state representing abnormal behavior. In this approach, the researchers used frequency distributions to represent network phenomena. Frequency distributions are used for type 1 properties (when there is a specific transition on the state machine) while for type 2 properties (the value of a specific state variable or a packet field when a trace traverses a transition), distribution of values for the state variable of interest are applied [32].

2.3.1.4 Genetic Network Programming (GNP)

Genetic Network Programming (GNP) is another approach for detecting intrusions of both types: anomaly and misuse. In [9], a learning algorithm starts with rule mining, which uses GNP to check the attribute values and compute the measurements of association rules using processing nodes.

In order to obtain the distribution of the average matching, the average matching degree between normal connection data and the rules in the normal rule pool is calculated. The matching degrees will be used later in the classification phase (detection phase) to make the system's decision.

2.3.1.5 Some Other Machine Learning Algorithms

In [15], where the researcher uses machine learning for detecting anomalies, the detection phase consisted of two steps: computing sequence similarity and classifying user behavior. In step one: the system calculates a numerical similarity measure which results from the number of adjacent matches between two sequences. Higher score of this measure means higher similarity [15].

The second step of the detection phase is classifying user behavior. This step processes the stream, token by token, and indicates at each point whether the user is a normal or an anomalous user. This determination is called classification of users. The classification is achieved according to a threshold value. If the mean value of the current window is greater than the threshold, then the current window is classified as normal, otherwise the window is classified as abnormal [15].

In [35], which employs a machine learning algorithm for anomaly detection, the empirical detection phase consists of three sub-steps: packet filtering, field selection, and packet profiling. Each sub-step is explained as follows [35]:

- a. Packet filtering: The goal of packet filtering is to eliminate malformed packets from raw traffic.

- b. The field selection scheme is performed using a Genetic Algorithm (GA). Preliminary tests are done using the typical genetic parameter values to find acceptable genetic parameters.
- c. For packet profiling, a Self-Organized Feature Map (SOFM) neural network is used to create different packet clusters. The prepared raw packets are 60,000 raw packets from two different sources with 30,000 each. One source was for normal data and the other was for different types of packets aggregated from the internet.
- d. Comparisons among the three SVMs and cross-validation tests: This step involves testing the three SVMs: soft margin SVM as a supervised method, one-class SVM as an unsupervised method, and the proposed enhanced SVM. The test for all of them was concluded using four different kinds of SVM kernel functions

In [45], the learning phase is divided into two steps: rule growing and rule pruning. In the rule growing step (GrowRule), the rule growing algorithm is used to handle each feature attribute in a growing set and decide the best split condition. During the learning process, the network is trained on normal and attacking data. The rule learning algorithm (FILMID) is utilized to perform inductive learning and construct a double-profile detection model from labeled network connection records. Besides FILMID, another two algorithms (RIPPER and C4.5) have been used in the training for four attack classes.

From the above learning algorithms used in pattern analysis phase, several comparisons can be drawn. Using neural networks helps in constructing a user profile or to train on a training data and test on testing data to detect both anomaly and misuse intrusions [17, 22], while the HMM is used to model normal behavior only from normal audit data [4]. Learning statistical properties was used in detecting anomalies only by learning the frequency distribution of the network to detect unusual behavior [32]. GNP was used by rule mining in checking the attribute values and computing the measurements of association rules using processing nodes to detect both anomaly and misuse intrusions [9]. Anomalies only were detected using machine learning in [15] by comparing the sequence similarities of the observed behavior and the stored behavior and then classifying user behavior to know whether the user is normal or anomalous. The POMDP learning algorithm was used in [14] in both anomaly and misuse detection. The learning involved the model parameters using an EM algorithm. Machine learning was used also in [35] for anomaly detection only. The detection phase of the approach involved packet filtering, field selection, and packet profiling to achieve detecting intrusions. The model comprised of building a double profile based on inductive learning to take the advantages of both anomaly and misuse detection techniques.

Some learning algorithms produce intermediate data which can be used later for classifier decision making during the detection phase. Some common forms of the generated knowledge representations are explained in the next section.

2.3.2 Knowledge Representation

In the intrusion detection problem, the knowledge representation can be one of the following types: weights resulting from training a neural network, rules resulting from fuzzy logic, conditional probabilities resulting from applying Hidden Markov Models, a cost function from POMDP, events from a log monitor, decision trees, or signature rules. Each of the aforementioned knowledge representation types is explained in the next sections.

a. Weights

The result of the gradient descent learning algorithm represents the values of connection weights between the neurons which are normally organized as matrix and called a weight matrix. As an example of using the neural networks in IDS is the model presented in [22], where the conjugate gradient descent algorithm has been used to train a feed-forward neural network on both normal and attack data. In [10], the same concept was used but on two neural networks: Multi-Layer Perceptron (MLP) and Self-Organizing Maps (SOM). The used approach utilized the SOM network first to cluster the traffic intensity into clusters and then trained the MLP network to make the decision.

b. Rules

Fuzzy rules are another form of knowledge representation that is used to provide effective learning. In [33], fuzzy rules consisted of numerical variables which represent the IF part and a class label which is represented by THEN part. Fuzzy rules are obtained automatically by “fuzzifying” the numerical variable of the definite rules (IF part) while the THEN part is the same as the resultant part of the definite rules [33].

c. Conditional probabilities

The Baum–Welch learning algorithm produces a conditional probability which can be used later in the detection phase to check the status of the system if it is under attack or not. In [6], after providing an input sequence, the HMM performs the modelling for this sequence with its own probability parameters using the Markov process. After finishing building the model, then evaluating the probability with which a given sequence is generated from the model is performed [6].

d. Cost Function in POMDP

The model presented in [14] is based on representing both the attacker and the legitimate user as unobservable, homogeneous Markov random variables by $A_t \in \{a_1, \dots, a_n\}$ and $U_t \in \{u_1, \dots, u_n\}$, respectively. At time t the computer state is called X_t which is generated by either an intruder (attacker) or a user and is controlled by a decision variable $D_t \in \{USER, ATTACKER\}$. The system is considered under intrusion when the captured data is produced by intruder, i.e.,

when $D_t = ATTACKER$. The next step is the action selection when an additional variable $C_t \in \{ALARM, NOALARM\}$ is used to model intrusion detection system actions [14].

e. Events from log monitor

In [17], the log file is monitored by the log monitor and events are sent to the log analyzer in case of a log change. In addition, system resources are also monitored by the systems resource monitor and their status is sent to the system resources analyzer during each time unit [17]. Finally, the active response unit, which receives the events from the log analyzer and system resources analyzer, is responsible for choosing the appropriate response to that situation which can be: notifying users, auditing, disconnecting from the network, etc. [17].

f. Decision trees

Pattern analyzers can also utilize decision trees in building an intrusion detection model. Decision trees have a learning process that results into the knowledge representation (the tree itself) that can be used in the detection phase. The main goal of decision tree classifier is to repeatedly separate the given dataset into subsets so that all elements in each final subset belong to the same class [12]. Three models of decision trees were used in [12] in the classification process: ID3, C4.5, and C5.0 algorithms. Another type of decision trees is called NBTree which is a hybrid between decision trees and NaïveBayes. The knowledge representation that results from NBTree is a tree whose leaves are NaïveBayes classifiers [8]. In intrusion detection problem, the decision tree classifier can be used to identify network data as malicious, benign, scanning, or any other category utilizing information like source/destination ports, IP addresses, and the number of bytes sent during a connection [12].

g. Signature Rules

One of the effective techniques in detecting intrusions is to use signature rules. In [19], several firewall rules were generated from network information such as packet source address, packet destination address, port from where packet is received, and packet type (protocol). The generated rules (knowledge representation) are dynamically modified based on the network requirement [19]. Behavior rule is another kind of rules that can be used to detect intrusions such as the model proposed in [20]. The knowledge representation of the model was based on behavior rules for defining acceptable behaviors of medical devices [20].

2.4 Decision Making Component (Detection Phase)

The second phase of the intrusion detection systems is the actual process of detecting the intrusions. Different detection algorithms need different steps to achieve this goal. Some of them need training and some do not, while others need rule generation as shown in some of the following examples.

2.4.1 Neural Networks

In [22], after a network was trained on two classes of data: normal and attack, the network now is ready for the testing phase. The three networks have shown detection accuracy of about 99%. The limitation of this approach is that it did not take into account a specific kind of attack and it dealt with only two classes of data: normal and attack. Some of the new datasets now differentiate between the attacks as the reaction of the IDS would be different against each type of attack.

2.4.2 Decision Tree

Decision trees have been successfully used in many applications due to its effectiveness. In [8], the researcher used an approach for network intrusion detection based on classifiers, decision trees, and decision rules. The detection phase in this work consisted of multiple steps and used multiple classifier algorithms and decision trees. For the classification algorithms, J48 (C4.5 Decision Tree Revision 8) was used. Next, the NnaïveBayes Tree classification algorithm was applied, and then decision table was used to evaluate feature subsets using a best-first search algorithm. The last classification algorithm was OneR, which was used for generating a one-level decision tree with a set of rules representation [8]. However, the approach did not involve calculating the False Alarm Rate (FAR), which is an important metric in evaluating an IDS.

2.4.3 Fuzzy Logic

The model presented in [33] was used to detect anomaly intrusions on the network. The researcher applied the model on the KDD cup99 dataset. Since the KDD cup99 dataset is very large to deal with, only 10% of the whole dataset is selected for training and testing and the data is selected from normal and attack data. The detection phase which uses fuzzy logic to detect intrusions consists of two sub-steps: a fuzzy decision module and finding an appropriate classification for a test input. The first step is used to select the most suitable attribute for a record's classification (normal or attack). This selection is performed by applying the deviation method [33] which uses the mined 1-length frequent items from each attribute and stores them in a vector.

The rule base is a knowledge base consisting of a set of rules acquired from the definite rules. The result of the inference engine would be selected from the set{Low, High}. Then, the “defuzzifier” transforms that output into useful values.

These useful values vary between 0 and 1, where 0 indicates normal data and 1 indicates pure attack data [33].

2.4.4 Genetic Network Programming

After calculating the matching degree in the learning phase, the class of a new connection data d needs to be recognized. The detection phase involves entering into a set of IF-THEN-ELSE statements to predict from the mentioned calculations the class of the current connection data whether it is normal, a known intrusion or an unknown intrusion or [9]. However, the limitation of this approach is that it did not give better accuracy than 90% which is not considered that high compared to the recent approaches.

2.4.5 Support Vector Machine

The model proposed in [42] depends on using a support vector machine (SVM) approach in detecting network intrusions. The proposed model was tested against four intrusion types: DoS, R2L, U2R, and probing attack. The intrusion detection system consists of three parts: an acquisition module of data packets, an intrusion detection agent, and a management agent. The intrusion detection agent is responsible for detecting illegal network activity (i.e., an attack). This agent uses a support vector machine to identify intrusions. The management agent—the third part—is responsible for organizing the performance of the intrusion detection agents and maintaining the whole system.

A possible drawback of this could be its lack in applying cross-validation in evaluating the results of the SVM classifiers to obtain more reliable results.

2.4.6 Some Other Decision Making Approaches

For space restriction, we are providing here some of other decision making approaches that we encourage readers to explore such as specification-based approach [32], mobile agent approach [7], situation awareness [13], malware detection [27], fast inductive learning [45], and negative selection [29]. Table 2.1 gives a brief summary on the detection approaches and their benefits discussed above.

Table 2.1 Detection approaches of IDS and their benefits

Reference	Detection approach	Benefits
[22]	Neural networks	The ability of neural networks to learn and generalize to detect attacks in a testing dataset
[8]	Decision tree	Accurate classification results for the input patterns
[33]	Fuzzy logic	The Fuzzifier was used to convert input attributes to linguistic variables and the Defuzzifier was used to transform the output of the inference engine to useful values (0 for normal and 1 for attack)
[32]	Specification-based method	Detecting anomalies when the observed statistics were so different from what was learnt
[27]	Malware detection	Efficient malware detection which can discover if there is any malware from the tokens of the signature
[7, 43]	Mobile agent	Efficient intrusion detector which was based on comparing the information collected by mobile agents with intrusion patterns
[9]	Genetic network programming (GNP)	Predicts the current connection's class whether it is normal or, known, or an unknown intrusion
[45]	Fast inductive learning	Used double profile to decrease the false positive and false negative in the classification results
[13]	Situation awareness	Distinguished attacks by maintaining a network security situation awareness graph and updating it periodically to detect attacks
[42]	Support vector machine (SVM)	Four SVMs were used as the kernel of an IDS to detect normal data and DoS, R2L, and U2R attacks
[29]	Negative selection	The detectors were able to reduce the detection speed by 50% in anomaly detection

2.5 Classifier's Decision

Generally, the detection phase should give a decision about what was discovered from the detection algorithm used. In some works like [17], the decision is made as a report to the administrator and called an auditing report. This report may involve notifying users, auditing or disconnecting from the network. The process of intrusion detection and the attack type are recorded by the audit database to be

used in the future [17]. Generally, a system decision can be one of the following three forms: threat, anomaly, or normal.

Different papers have been surveyed in this chapter with different types of decisions. Some of them just give a decision whether the data was an anomaly or normal such as [2, 15, 32, 38]. Some other papers limited their decisions to one of three options: anomaly, misuse, or normal. The coming sections explain the IDS decisions in more details.

2.5.1 Threat

Computer networks are the targets of many kinds of attacks and they are exposed to many new kinds of threats through the internet every day. In this section, four fundamental classes of attacks [18] are explained and illustrated with their subclasses in Fig. 2.2. The four fundamental classes are explained in detail as follows:

a. Access

When an attacker tries to obtain information that the attacker is not allowed to access. Information may be exposed to this kind of attack while residing or during transmission. This type of attack puts the confidentiality of the information at risk. In general, access attacks are divided into three subclasses [18]:

- 1. Snooping: Snooping examines information files in order to find useful information.
- 2. Eavesdropping: Is listening on a conversation by a person who is not part of it. This typically occurs when an unauthorized person occupies a location where the information is expected to pass by as is shown in Fig. 2.3.

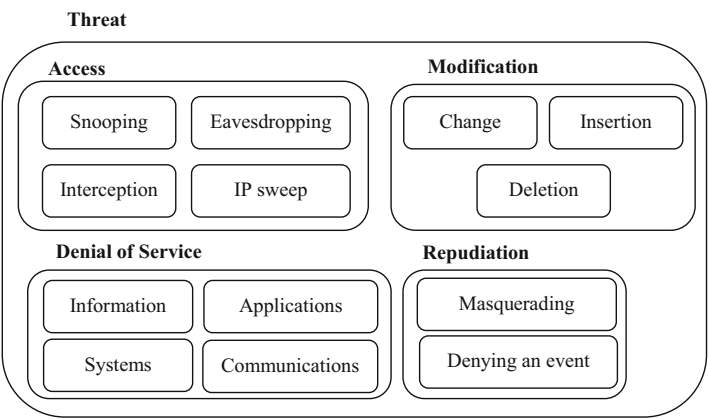


Fig. 2.2 Threat types with their subclasses

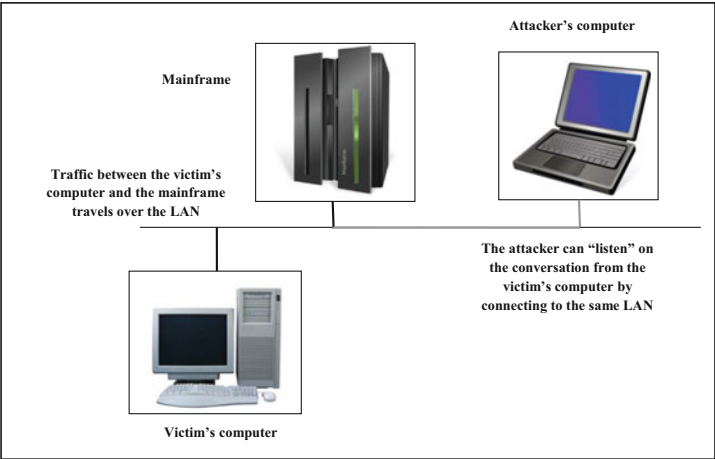


Fig. 2.3 Eavesdropping

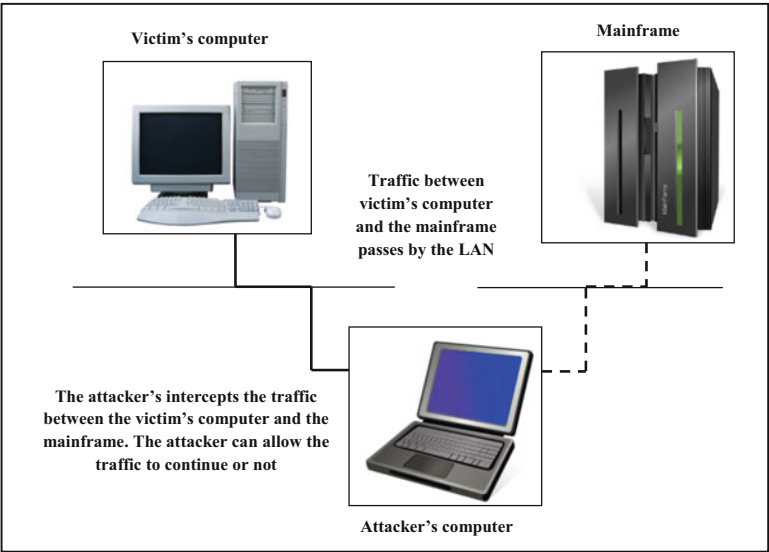


Fig. 2.4 Interception

3. Interception: Interception is considered more serious than eavesdropping to the system. That is because the attacker intercepts information, inserts himself/herself in the path of the information and captures it before it reaches its destination. After analyzing the information, the attacker has the choice to let the information continue to its destination or not as shown in Fig. 2.4.

4. IP sweep (Network scanning): This kind of attack is designed to identify the range of IP addresses that are mapped to live hosts on a target network by sending many ping requests to the full IP range and waiting for the reply. The purpose behind this technique is that it helps the attacker to know legitimate IP addresses in the target domain at the time of attack [32].

A probing attack is another kind of network scanning attack [36]. This attack occurs when an attacker performs a comprehensive scan on a network to collect information or find known vulnerabilities. Port scanning is a technique used to determine what ports are open, and that can inform the attacker what potential services running on a system are available to the attacker. There are two sides to port scanning. The first one is that the result can be utilized by network and system administrators as a part of network security audits for network maintenance. The second face is that it can be utilized by attackers who aim to compromise the system by exploiting a known vulnerability for one of the detected running services on its open port. Port scanning has some additional applications such as [36]:

- Identifying which hosts are active on a network.
- Identifying other network topological information, such as IP addressing, MAC addressing, router and gateway filtering, firewall rules, and IP-based trust relationships.

b. *Modification*

A modification attack is when the attacker tries to alter information that the attacker is not authorized to. Generally, modification attacks are divided into three subclasses [18]:

1. Changes: This kind of attack involves changing existing sensitive information by the attacker such as an employee's salary.
2. Insertion: This kind of attack involves inserting information that did not exist before. For example, in a banking system an attacker might insert a transaction that moves money from a customer's account to his own account.
3. Deletion: Basically, a deletion attack is the removal of existing information such as the removal of a historical record that represents a transaction record in a banking system.

c. *Denial of Service*

Denial-of-service (DoS) attacks are attacks that forbid the use of resources to authentic users of the system [18]. DoS attacks usually target the availability goal of the security and sometimes they are called availability attacks [26]. DoS attacks have been considered as one of the most significant attacks to networks in the last few years since they can cause a huge disorder to network functions. In addition, DoS attacks have been proven to be difficult to protect from [41].

d. *Repudiation*

A repudiation attack is an attempt to give false information, or to deny that a real event or transaction has been performed by a particular entity [18]. Preventing

an entity from denying that it has taken part in a message exchange is called non-repudiation [23].

Usually, repudiation attacks are divided into two subclasses [18].

1. Masquerading

Masquerading means the attacker attempts to imitate or impersonate someone else or some other system. This attack can threaten personal communication, transactions, or system-to-system communications.

2. Denying an Event

Denying an event is the rejection of an event such as denying a bill of some purchase or denying cash withdrawal from a bank account.

2.5.2 *Anomaly*

If the intrusion detection system was designed to detect anomalies in the network, then it should be able to distinguish these events from those that it has seen previously in a training phase. Usually, this kind of IDS considers any deviation from the normal behavior of that network as an anomaly. However, this kind of IDS tends to suffer from false-positive classification.

Anomaly intrusions are of different types and a remote to local (R2L) attack is a class of anomaly attacks where an attacker tries to exploit the machine's vulnerability to illegally gain local access (becomes a local user) to a machine by sending packets to a machine over a network. It can happen when an attacker sends packets over a network to a machine which the attacker does not have an account on and exploits some vulnerability on that machine to acquire local access as a legitimate user of that machine. The Dictionary, FTP-Write, Guest, and Xsnoop attacks are types of R2L attack which all attempt to take advantage of any weaknesses or shortcoming in the configuration of system security policies [36].

User to root (U2R) attacks form another class of anomaly attacks when an attacker starts with access to a normal user account on the system then tries to acquire root access to the system by exploiting any vulnerability on the system. Usually, the attacker starts by accessing the system using a normal user account—which might have been obtained by some techniques like: sniffing passwords, a dictionary attack, or social engineering—and moves to exploiting some vulnerability to achieve the goal (gaining root access to the system). Buffer overflow attacks are the best known type of U2R attacks which come in many forms. Buffer overflows happen when a program copies a huge amount of data into a static buffer without checking the space availability in that buffer [36].

In [2], an approach for reusing information from a different layer for intrusion detection was adopted. WSN was divided into several layers and each layer ran a different protocol. The proposed technique used different information from different layers for ID as shown in Table 2.2.

Table 2.2 Summary of the layer information taken for ID

Layer	Protocols/techniques for anomaly detection	Use
Physical	RSSI value	Detects masquerade
Routing	Maintain neighbor lists, MAC layer transmission schedules are also used	Guarantees information authentication
MAC	TDMA: Check if adversary follows TDMA schedule	Keeps track of TDMA schedules of other nodes
	S-MAC: Check if sender is supposed to be sleeping	Keep track of sleep-wake up schedules of other nodes
Application	Use triangulation to detect intrusions	Detects masquerade
	Round trip time	Detects masquerade

Table 2.3 Types of decisions for each kind of attack and the references used

Type of attack	References
Access	[1, 8, 11, 14, 18, 21, 22, 30, 32, 33, 37, 39, 42]
DoS	[8, 9, 11, 18, 22, 30, 32, 33, 37, 39, 42, 43, 45]
Repudiation	[31, 45]
Anomaly	[2, 9, 14, 15, 17, 25, 27, 32, 35, 37, 38]

2.5.3 Normal

When the data is neither a threat nor an anomaly then it is considered normal data. This normal data represents the regular network traffic for that network or user. From the above, we can summarize the types of decisions used in this chapter for each class of data, as shown above in Table 2.3.

2.6 Conclusion and Open Issues

In this chapter and taxonomy of the IDSs, we have explored a wide range of pattern analyzers (classifiers) used in the IDSs and presented the taxonomy of the knowledge base that is produced as intermediate step. We also presented different techniques that have been utilized in the actual detection phase of the IDSs. We also explored the taxonomy of the classifiers’ decision and explained each subcategory of these decisions.

As a matter of fact, the intrusion detection will keep developing as long as there are new attacks on the computer networks every day. In the last years, the Internet witnessed severe attacks that led to catastrophic consequences on multiple levels of computer users (i.e., end-users, governments, companies, etc.). Having said that, the world will still need to find new techniques to defend the computer networks and provide the ultimate security to the users from these attacks.

One of the major open issues is that since pattern classifiers have to work in adversarial environments (where the classifier needs to discriminate between normal and hostile patterns such as spam filtering, intrusion detection, and biometric identity verification), these classifiers need to deal with the attacks that try to avoid detection or force a classifier to generate many false alarms [4]. These days the attacks are being more sophisticated such that the input data can be intentionally tampered by skilful adversary to overcome the classifiers. According to [5], now this is considered as an arm race between adversary and classifier designers. The procedure of classifier designer could be either “reactive” or “proactive” arm race between the adversary and the classifier designer. The “reactive” procedure starts after an adversary analyzes the classifier defenses and formulates an attack strategy to defeat them. The designer reacts to the attack by analyzing the attack’s effects and devising countermeasures. The “proactive” arm race involves the designer’s attempt to anticipate the adversary by mimicking possible attacks, evaluating their effects, and developing countermeasures if necessary [5]. To improve the robustness of a classifier, different techniques have been used in the literature. One of the early efforts was proposing multiple classifiers systems (bagging and random subspace method) to improve the robustness of linear classifiers to adversarial data manipulation [3].

References

1. Bergadano, F., Gunetti, D., & Picardi, C. (2003). Identity verification through dynamic keystroke analysis. *Intelligence Data Analysis*, 7(5), 469–496.
2. Bhuse, V., & Gupta, A. (2006). Anomaly intrusion detection in wireless sensor networks. *Journal of High Speed Networks*, 15(1), 33–51.
3. Biggio, B., Fumera, G., & Roli, F. (2010). Multiple classifier systems for robust classifier design in adversarial environments. *International Journal of Machine Learning and Cybernetics*, 1(1), 27–41. doi:[10.1007/s13042-010-0007-7](https://doi.org/10.1007/s13042-010-0007-7)
4. Biggio, B., Fumera, G., & Roli, F. (2011). Design of robust classifiers for adversarial environments. In *IEEE international conference on systems, man, and cybernetics (SMC)* (pp. 977–982). IEEE.
5. Biggio, B., Fumera, G., & Roli, F. (2014). Security evaluation of pattern classifiers under attack. *IEEE Transactions on Knowledge and Data Engineering*, 26(4), 984–996. doi:[10.1109/TKDE.2013.57](https://doi.org/10.1109/TKDE.2013.57)
6. Cho, S. B., & Park, H. J. (2003). Efficient anomaly detection by modeling privilege flows using hidden markov model. *Computers & Security*, 22(1), 45–55. doi:[10.1016/S0167-4048\(03\)00112-3](https://doi.org/10.1016/S0167-4048(03)00112-3)
7. Dastjerdi, A. V., & Bakar, K. A. (2008). A novel hybrid mobile agent based distributed intrusion detection system. *Proceedings of World Academy of Science, Engineering and Technology*, 35, 116–119.
8. Gandhi, G. M., Appavoo, K., & Srivatsa, S. (2010). Effective network intrusion detection using classifiers decision trees and decision rules. *International Journal of Advanced Networking and Applications*, 2(3), 686–692.
9. Gong, Y., Mabu, S., Chen, C., Wang, Y., & Hirasawa, K. (2009). Intrusion detection system combining misuse detection and anomaly detection using genetic network programming. In *ICCAS-SICE, 2009*, (pp. 3463–3467).

10. Haidar, G. A., & Boustany, C. (2015). High perception intrusion detection system using neural networks. In *2015 ninth international conference on complex, intelligent, and software intensive systems* (pp. 497–501). doi:[10.1109/CISIS.2015.73](https://doi.org/10.1109/CISIS.2015.73)
11. Jalil, K. A., Kamarudin, M. H., & Masrek, M. N. (2010). Comparison of machine learning algorithms performance in detecting network intrusion. In *2010 international conference on networking and information technology* (pp. 221–226). doi:[10.1109/ICNIT.2010.5508526](https://doi.org/10.1109/ICNIT.2010.5508526)
12. Kumar, M., Hanumanthappa, M., & Kumar, T. V. S. (2012). Intrusion detection system using decision tree algorithm. In *2012 IEEE 14th international conference on communication technology* (pp. 629–634). doi:[10.1109/ICCT.2012.6511281](https://doi.org/10.1109/ICCT.2012.6511281)
13. Lan, F., Chunlei, W., & Guoqing, M. (2010). A framework for network security situation awareness based on knowledge discovery. In *2010 2nd international conference on computer engineering and technology* (Vol. 1, pp. V1–226–V1–231). doi:[10.1109/ICCET.2010.5486194](https://doi.org/10.1109/ICCET.2010.5486194).
14. Lane, T. (2006). A decision-theoretic, semi-supervised model for intrusion detection. In *Machine learning and data mining for computer security* (pp. 157–177). London: Springer.
15. Lane, T., & Brodley, C. E. (1997). An application of machine learning to anomaly detection. In *Proceedings of the 20th national information systems security conference* (Vol. 377, pp. 366–380).
16. Lin, W. C., Ke, S. W., & Tsai, C. F. (2015). Cann: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-Based Systems*, 78, 13–21. doi:[10.1016/j.knosys.2015.01.009](https://doi.org/10.1016/j.knosys.2015.01.009)
17. Lin, Y., Zhang, Y., & Ou, Y-J (2010). The design and implementation of host-based intrusion detection system. In *2010 third international symposium on intelligent information technology and security informatics* (pp. 595–598). doi:[10.1109/IITSI.2010.127](https://doi.org/10.1109/IITSI.2010.127)
18. Maiwald, E. (2001). *Network security: A beginner's guide*. New York, NY: New York Osborne/McGraw-Hill. <http://openlibrary.org/books/OL3967503M>
19. Mantur, B., Desai, A., & Nagegowda, K. S. (2015). *Centralized control signature-based firewall and statistical-based network intrusion detection system (NIDS) in software defined networks (SDN)* (pp. 497–506). New Delhi: Springer. doi:[10.1007/978-81-322-2550-8_48](https://doi.org/10.1007/978-81-322-2550-8_48)
20. Mitchell, R., & Chen, I. R. (2015). Behavior rule specification-based intrusion detection for safety critical medical cyber physical systems. *IEEE Transactions on Dependable and Secure Computing*, 12(1), 16–30. doi:[10.1109/TDSC.2014.2312327](https://doi.org/10.1109/TDSC.2014.2312327)
21. Mo, Y., Ma, Y., & Xu, L. (2008). Design and implementation of intrusion detection based on mobile agents. In *2008 IEEE international symposium on IT in medicine and education* (pp. 278–281). doi:[10.1109/ITME.2008.4743870](https://doi.org/10.1109/ITME.2008.4743870)
22. Mukkamala, S., Janoski, G., & Sung, A. (2002). Intrusion detection: Support vector machines and neural networks. *IEEE International Joint Conference on Neural Networks (ANNIE)*, 2, 1702–1707.
23. Muntean, C., Dojen, R., & Coffey, T. (2009). Establishing and preventing a new replay attack on a non-repudiation protocol. In *IEEE 5th international conference on intelligent computer communication and processing, ICCP 2009* (pp. 283–290). IEEE.
24. Newsome, J., Karp, B., & Song D. (2005). Polygraph: Automatically generating signatures for polymorphic worms. In *2005 IEEE symposium on security and privacy (S&P'05)* (pp. 226–241). IEEE.
25. Pannell, G., & Ashman, H. (2010). Anomaly detection over user profiles for intrusion detection. In *Proceedings of the 8th Australian information security management conference* (pp. 81–94). Perth, Western Australia: School of Computer and Information Science, Edith Cowan University.
26. Pfleeger, C. P., & Pfleeger, S. L. (2006). *Security in computing* (4th ed.). Upper Saddle River, NJ: Prentice Hall PTR.
27. Rieck, K., Schwenk, G., Limmer, T., Holz, T., & Laskov, P. (2010). Botzilla: Detecting the phoning home of malicious software. In *Proceedings of the 2010 ACM symposium on applied computing* (pp. 1978–1984). ACM.
28. Di Pietro, R., & Mancini, L. V. (2008). *Intrusion detection systems* (Vol. 38). New York, NY: Springer Science & Business Media.

29. Sadeghi, Z., & Bahrami, A. S. (2013). Improving the speed of the network intrusion detection. In *The 5th conference on information and knowledge technology* (pp. 88–91). doi:[10.1109/IKT.2013.6620044](https://doi.org/10.1109/IKT.2013.6620044)
30. Sarvari, H., & Keikha, M. M. (2010). Improving the accuracy of intrusion detection systems by using the combination of machine learning approaches. In *2010 international conference of soft computing and pattern recognition* (pp. 334–337). doi:[10.1109/SOCPAR.2010.5686163](https://doi.org/10.1109/SOCPAR.2010.5686163)
31. Schonlau, M., DuMouchel, W., Ju, W. H., Karr, A. F., Theus, M., & Vardi, Y. (2001). Computer intrusion: Detecting masquerades. *Statistical Science*, 16(1), 58–74.
32. Sekar, R., Gupta, A., Frullo, J., Shanbhag, T., Tiwari, A., Yang, H., & Zhou, S. (2002). Specification-based anomaly detection: A new approach for detecting network intrusions. In *Proceedings of the 9th ACM conference on computer and communications security, CCS '02* (pp. 265–274). New York, NY: ACM. doi:[10.1145/586110.586146](https://doi.org/10.1145/586110.586146)
33. Shanmugavadivu, R., & Nagarajan, N. (2011). Network intrusion detection system using fuzzy logic. *Indian Journal of Computer Science and Engineering (IJCSSE)*, 2(1), 101–111.
34. Sheng Gan, X., Shun Duanmu, J., Fu Wang, J., & Cong, W. (2013). Anomaly intrusion detection based on {PLS} feature extraction and core vector machine. *Knowledge-Based Systems*, 40, 1–6. doi:[10.1016/j.knosys.2012.09.004](https://doi.org/10.1016/j.knosys.2012.09.004)
35. Shon, T., & Moon, J. (2007). A hybrid machine learning approach to network anomaly detection. *Information Sciences*, 177(18), 3799–3821. doi:[10.1016/j.ins.2007.03.025](https://doi.org/10.1016/j.ins.2007.03.025)
36. Singh, S., & Silakari, S. (2009). A survey of cyber attack detection systems. *IJCSNS International Journal of Computer Science and Network Security*, 9(5), 1–10.
37. Terry, S., & Chow, B. J. (2005). *An assessment of the DARPA IDS evaluation dataset using snort* (Technical report, UC Davis Technical Report).
38. Trinius, P., Willems, C., Rieck, K., & Holz, T. (2009). *A malware instruction set for behavior-based analysis* (Technical Report TR-2009-07). University of Mannheim.
39. Vasudevan, A., Harshini, E., & Selvakumar, S. (2011). Ssenet-2011: a network intrusion detection system dataset and its comparison with kdd cup 99 dataset. In *2011 second asian himalayas international conference on internet (AH-ICI)* (pp. 1–5). IEEE.
40. Wang, W., Guyet, T., Quiniou, R., Cordier, M. O., Masegaglia, F., & Zhang, X. (2014). Autonomic intrusion detection: Adaptively detecting anomalies over unlabeled audit data streams in computer networks. *Knowledge-Based Systems*, 70, 103–117. doi:[10.1016/j.knosys.2014.06.018](https://doi.org/10.1016/j.knosys.2014.06.018)
41. Wang, Y., Lin, C., Li, Q. L., & Fang, Y. (2007). A queueing analysis for the denial of service (dos) attacks in computer networks. *Computer Networks*, 51(12), 3564–3573.
42. Xiaoqing, G., Hebin, G., & Luyi, C. (2010). Network intrusion detection method based on agent and svm. In *2010 2nd IEEE international conference on information management and engineering* (pp. 399–402). doi:[10.1109/ICIME.2010.5477694](https://doi.org/10.1109/ICIME.2010.5477694)
43. Xu, J., & Wu, S. (2010). Intrusion detection model of mobile agent based on aglets. In *2010 international conference on computer application and system modeling (ICCASM 2010)* (Vol. 4, pp. V4-347–V4-350). doi:[10.1109/ICCASM.2010.5620189](https://doi.org/10.1109/ICCASM.2010.5620189)
44. Xue-qin, Z., Chun-hua, G., & Jia-jun, L. (2006). Intrusion detection system based on feature selection and support vector machine. In *2006 first international conference on communications and networking in China* (pp. 1–5). doi:[10.1109/CHINACOM.2006.344739](https://doi.org/10.1109/CHINACOM.2006.344739)
45. Yang, W., Wan, W., Guo, L., & Zhang, L. J. (2007). An efficient intrusion detection model based on fast inductive learning. In *2007 international conference on machine learning and cybernetics*, (Vol. 6, pp. 3249–3254). doi:[10.1109/ICMLC.2007.4370708](https://doi.org/10.1109/ICMLC.2007.4370708)

Computer and Network Security Essentials

Daimi, K. (Ed.)

2018, XV, 618 p. 80 illus., 50 illus. in color., Hardcover

ISBN: 978-3-319-58423-2