Network Anomaly Detection: Methods, Systems and Tools

Monowar H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita

***Abstract*—Network anomaly detection is an important and dynamic research area. Many network intrusion detection methods and systems (NIDS) have been proposed in the literature. In this paper, we provide a structured and comprehensive overview of various facets of network anomaly detection so that a researcher can become quickly familiar with every aspect of network anomaly detection. We present attacks normally encountered by network intrusion detection systems. We categorize existing network anomaly detection methods and systems based on the underlying computational techniques used. Within this framework, we briefly describe and compare a large number of network anomaly detection methods and systems. In addition, we also discuss tools that can be used by network defenders and datasets that researchers in network anomaly detection can use. We also highlight research directions in network anomaly detection.**

***Index Terms*—Anomaly detection, NIDS, attack, dataset, intrusion detection, classifier, tools**

I. Introduction

DUE to advancements in Internet technologies and the concomitant rise in the number of network attacks, network intrusion detection has become a significant research issue. In spite of remarkable progress and a large body of work, there are still many opportunities to advance the state-of-the-art in detecting and thwarting network-based attacks [1].

According to Anderson [2], an intrusion attempt or a threat is a deliberate and unauthorized attempt to (i) access information, (ii) manipulate information, or (iii) render a system unreliable or unusable. For example, (a) *Denial of Service (DoS)* attack attempts to starve a host of its resources, which are needed to function correctly during processing; (b) *Worms and viruses* exploit other hosts through the network; and (c) *Compromises* obtain privileged access to a host by taking advantages of known vulnerabilities.

The term *anomaly-based intrusion detection in networks* refers to the problem of finding exceptional patterns in network traffic that do not conform to the expected normal behavior. These nonconforming patterns are often referred to as anomalies, outliers, exceptions, aberrations, surprises, peculiarities or discordant observations in various application domains [3], [4]. Out of these, anomalies and outliers are two of the most commonly used terms in the context of anomaly-based intrusion detection in networks.

Anomaly detection has extensive applications in areas such as fraud detection for credit cards, intrusion detection for cyber security, and military surveillance for enemy activities. For example, an anomalous traffic pattern in a computer network may mean that a hacked computer is sending out sensitive data to an unauthorized host.

The statistics community has been studying the problem of detection of anomalies or outliers from as early as the 19th century [5]. In recent decades, machine learning has started to play a significant role in anomaly detection. A good number of anomaly-based intrusion detection techniques in networks have been developed by researchers. Many techniques work in specific domains, although others are more generic.

Even though there are several surveys available in the literature on network anomaly detection [3], [6], [7], surveys such as [6], [7], discuss far fewer detection methods than we do. In [3], the authors discuss anomaly detection in general and cover the network intrusion detection domain only briefly. None of the surveys [3], [6], [7] include common tools used during execution of various steps in network anomaly detection. They also do not discuss approaches that combine several individual methods to achieve better performance. In this paper, we present a structured and comprehensive survey on anomaly-based network intrusion detection in terms of general overview, techniques, systems, tools and datasets with a discussion of challenges and recommendations. Our presentation is detailed with ample comparisons where necessary and is intended for readers who wish to begin research in this field.

A. Prior Surveys on Network Anomaly Detection

Network anomaly detection is a broad research area, which already boasts a number of surveys, review articles, as well as books. An extensive survey of anomaly detection techniques developed in machine learning and statistics has been provided by [8], [9]. Agyemang et al. [10] present a broad review of anomaly detection techniques for numeric as well as symbolic data. An extensive overview of neural networks and statistics-based novelty detection techniques is found in [11]. Patcha and Park [6] and Snyder [12] present surveys of anomaly detection techniques used specifically for cyber intrusion detection.

A good amount of research on outlier detection in statistics is found in several books [13]–[15] as well as survey articles [16]–[18]. Exhaustive surveys of anomaly detection in several

Manuscript received March 7, 2012; revised August 28, 2012 and February 27, 2013.

M. H. Bhuyan is with the Department of Computer Science and Engineering, Tezpur University, Napaam, Tezpur-784028, Assam, India (e-mail: mhb@tezu.ernet.in).

D. K. Bhattacharyya is with the Dept. of Computer Science and Engineering, Tezpur University, Napaam, Tezpur-784028, Assam, India (e-mail: dkb@tezu.ernet.in).

J. K. Kalita is with the Department of Computer Science, University of Colorado, Colorado Springs, CO 80918, USA (e-mail: jkalita@uccs.edu).

Digital Object Identifier 10.1109/SURV.2013.052213.00046

1553-877X/14/$31.00 © 2014 IEEE

domains have been presented in [3], [7]. Callado et al. [19] report major techniques and problems identified in IP traffic analysis, with an emphasis on application detection. Zhang et al. [20] present a survey on anomaly detection methods in networks. A review of flow-based intrusion detection is presented by Sperotto et al. [21], who explain the concepts of flow and classified attacks, and provide a detailed discussion of detection techniques for scans, worms, Botnets and DoS attacks.

Some work [22]–[25] has been reported in the context of wireless networks. Sun et al. [23] present a survey of intrusion detection techniques for mobile ad-hoc networks (MANET) and wireless sensor networks (WSN). They also present several important research issues and challenges in the context of building IDSs by integrating aspects of mobility.

Sun et al. [22] discuss two domain independent online anomaly detection schemes (Lempel-Ziv based and Markov-based) using the location history obtained from traversal of a mobile user. Sun et al. [25] also introduce two distinct approaches to build IDSs for MANET, viz., Markov-chain based and Hotelling's T2 test-based. They also propose an adaptive scheme for dynamic selection of normal profiles and corresponding thresholds. Sun et al. [24] construct a feature vector based on several parameters such as call duration, call inactivity period, and call destination to identify users' calling activities. They use classification techniques to detect anomalies.

An extensive survey of DoS and distributed DoS attack detection techniques is presented in [26]. Discussion of network coordinate systems, design and security is found in [27], [28]. Wu and Banzhaf [29] present an overview of applications of computational intelligence methods to the problem of intrusion detection. They include various methods such as artificial neural networks, fuzzy systems, evolutionary computation, artificial immune systems, swarm intelligence, and soft computing.

Dong et al. [30] introduce an Application Layer IDS based on sequence learning to detect anomalies. The authors demonstrate that their IDS is more effective compared to approaches using Markov models and k-means algorithms. A general comparison of various survey papers available in the literature with our work is shown in Table I. The survey contemplated in this paper covers most well-cited approaches and systems reported in the literature so far.

Our survey differs from the existing surveys in the following ways.

- Like [35], we discuss sources, causes and aspects of network anomalies, and also include a detailed discussion of sources of packet and flow level feature datasets. In addition, we include a large collection of up-to-date anomaly detection methods under the categories of statistical, classification-based, knowledge-based, soft computing, clustering-based and combination learners, rather than restricting ourselves to only statistical approaches. We also include several important research issues, open challenges and some recommendations.
- Like [36], we attempt to provide a classification of various anomaly detection methods, systems and tools introduced till date in addition to a classification of attacks and their characteristics. In addition, we perform detailed comparisons among these methods. Furthermore, like [36], we provide practical recommendations and a list of research issues and open challenges.
- Unlike [9], [19], our survey is not restricted to only IP traffic classification and analysis. It includes a large number of up-to-date methods, systems and tools and analysis. Like [19], we also include a detailed discussion on flow and packet level capturing and preprocessing. However, unlike [9], [19], we include ideas for developing better IDSs, in addition to providing a list of practical research issues and open challenges.
- Unlike [37], our survey is not restricted to those solutions introduced for a particular network technology, like CRN (Cognitive Radio Network). Also unlike [37], we include a discussion of a wide variety of attacks, instead of only CRN specific attacks.
- Unlike [27], our survey is focused on network anomalies, their sources and characteristics; and detection approaches, methods and systems, and comparisons among them. Like [27], we include performance metrics, in addition to a discussion of the datasets used for evaluation of any IDS.

B. The Problem of Anomaly Detection

To provide an appropriate solution in network anomaly detection, we need the concept of normality. The idea of *normal* is usually introduced by a formal model that expresses relations among the fundamental variables involved in system dynamics. Consequently, an event or an object is detected as anomalous if its degree of deviation with respect to the profile or behavior of the system, specified by the normality model, is high enough.

For example, let us take an anomaly detection system S that uses a supervised approach. It can be thought of as a pair $S = (M, D)$, where M is the model of normal behavior of the system and D is a proximity measure that allows one to compute, given an activity record, the degree of deviation that such activities have with regard to the model M. Thus, each system has mainly two modules: (i) a modeling module and (ii) a detection module. One trains the systems to get the normality model M. The obtained model is subsequently used by the detection module to evaluate new events or objects or traffic as anomalous or outliers. It is the measurement of deviation that allows classification of events or objects as anomalous or outliers. In particular, the modeling module needs to be adaptive to cope with dynamic scenarios.

C. Our Contributions

This paper provides a structured and broad overview of the extensive research on network anomaly detection methods and NIDSs. The major contributions of this survey are the following.

(a) Like the categorization of the network anomaly detection research suggested in ([8], [10]), we classify detection methods and NIDSs into a number of categories. In addition, we also provide an analysis of many methods in terms of their capability and performance, datasets

TABLE I
A COMPARISON OF OUR SURVEY WITH EXISTING SURVEY ARTICLES

Methods /NIDSs /Tools	*Topics covered*	[8]	[10]	[11]	[6]	[16]	[17]	[3]	[7]	[21]	[26]	[29]	[31]	[32]	[33]	[34]	*Our survey*
Methods	Statistical	√	√	√	√	√	√	√	√		√	√	√		√		√
	Classification-based	√	√		√		√	√	√	√		√	√	√			√
	Knowledge-based							√	√	√	√			√		√	√
	Soft computing												√		√		√
	Clustering-based	√	√	√	√			√	√				√				√
	Ensemble-based																√
	Fusion-based																√
	Hybrid																√
NIDSs	Statistical								√						√		√
	Classification-based																√
	Soft computing														√		√
	Knowledge-based								√							√	√
	Data Mining								√					√		√	√
	Ensemble-based																√
	Hybrid																√
Tools	Capturing, Preprocessing, Attack launching																√

used, matching mechanisms, number of parameters, and detection mechanisms.

(b) Most existing surveys do not cover ensemble approaches or data fusion for network anomaly detection, but we do.

(c) Most existing surveys avoid feature selection methods, which are crucial in the network anomaly detection task. We present several techniques to determine feature relevance in intrusion datasets and compare them.

(d) In addition to discussing detection methods, we present several NIDSs with architecture diagrams with components and functions, and also present a comparison among the NIDSs.

(e) We summarize tools used in various steps for network traffic anomaly detection.

(f) We also provide a description of the datasets used for evaluation.

(g) We discuss performance criteria used for evaluating methods and systems for network anomaly detection.

(h) We also provide recommendations or a wish list to the developers of ideal network anomaly detection methods and systems.

(i) Finally, we highlight several important research issues and challenges from both theoretical and practical viewpoints.

D. Organization

In this paper, we provide a comprehensive and exhaustive survey of anomaly-based network intrusion detection: fundamentals, detection methods, systems, tools and research issues as well as challenges. Section II discusses the basics of intrusion detection in networks while Section III presents network anomaly detection and its various aspects. Section IV discusses and compares various methods and systems for network anomaly detection. Section V reports criteria for performance evaluation of network anomaly detection methods and systems. Section VI presents recommendations to developers of network anomaly detection methods and systems. Section VII is devoted to research issues and challenges faced by anomaly-based network intrusion detection researchers. Opportunities for future research and concluding remarks are presented in Section VIII.

II. INTRUSION DETECTION

Intrusion is a set of actions aimed to compromise the security of computer and network components in terms of confidentiality, integrity and availability [38]. This can be done by an inside or outside agent to gain unauthorized entry and control of the security mechanism. To protect infrastructure of network systems, intrusion detection systems (IDSs) provide well-established mechanisms, which gather and analyze information from various areas within a host or a network to identify possible security breaches.

Intrusion detection functions include (i) monitoring and analyzing user, system, and network activities, (ii) configuring systems for generation of reports of possible vulnerabilities, (iii) assessing system and file integrity (iv) recognizing patterns of typical attacks (v) analyzing abnormal activity, and (vi) tracking user policy violations. An IDS uses vulnerability assessment to assess the security of a host or a network. Intrusion detection works on the assumption that intrusion activities are noticeably different from normal system activities and thus detectable.

A. Different Classes of Attacks

Anderson [2] classifies intruders into two types: external and internal. External intruders are unauthorized users of the machines they attack, whereas internal intruders have permission to access the system, but do not have privileges for the root or superuser mode. A masquerade internal intruder logs in as other users with legitimate access to sensitive data whereas a clandestine internal intruder, the most dangerous, has the power to turn off audit control for themselves.

There are various classes of intrusions or attacks [39], [40] in computer systems. A summary is reported in Table II.

TABLE II
CLASSES OF COMPUTER ATTACKS: CHARACTERISTICS AND EXAMPLE

Attack name	*Characteristics*	*Example*
Virus	(i) A self replicating program that infects the system without any knowledge or permission from the user. (ii) Increases the infection rate of a network file system if the system is accessed by another computer.	Trivial.88.D, Polyboot.B, Tuareg
Worm	(i) A self replicating program that propagates through network services on computer systems without user intervention. (ii) Can highly harm network by consuming network bandwidth.	SQL Slammer, Mydoom, CodeRed Nimda
Trojan	(i) A malicious program that cannot replicate itself but can cause serious security problems in the computer system. (ii) Appears as a useful program but in reality it has a secret code that can create a backdoor to the system, allowing it to do anything on the system easily, and can be called as the hacker gets control on the system without user permission.	Example-Mail Bomb, phishing attack
Denial of service (DoS)	(i) Attempts to block access to system or network resources. (ii) The loss of service is the inability of a particular network or a host service, such as e-mail to function. (iii) It is implemented by either forcing the targeted computer(s) to reset, or consuming resources. (iv) Intended users can no longer communicate adequately due to non-availability of service or because of obstructed communication media.	Buffer overflow, ping of death(PoD), TCP SYN, smurf, teardrop
Network Attack	(i) Any process used to maliciously attempt to compromise the security of the network ranging from the data link layer to the application layer by various means such as manipulation of network protocols. (ii) Illegally using user accounts and privileges, performing actions to delete network resources and bandwidth, performing actions that prevent legitimate authorized users from accessing network services and resources.	Packet injection, SYN flood
Physical Attack	An attempt to damage the physical components of networks or computers.	Cold boot, evil maid
Password Attack	Aims to gain a password within a short period of time, and is usually indicated by a series of login failures.	Dictionary attack, SQL injection attack
Information Gathering Attack	Gathers information or finds known vulnerabilities by scanning or probing computers or networks.	SYS scan, FIN scan, XMAS scan
User to Root (U2R) attack	(i) It is able to exploit vulnerabilities to gain privileges of superuser of the system while starting as a normal user on the system. (ii) Vulnerabilities include sniffing passwords, dictionary attack, or social engineering.	Rootkit, loadmodule, perl
Remote to Local (R2L) attack	(i) Ability to send packets to a remote system over a network without having any account on that system, gain access either as a user or as a root to the system and do harmful operations. (ii) Performs attack against public services (such as HTTP and FTP) or during the connection of protected services (such as POP and IMAP).	Warezclient, warezmaster, imap, ftp_write, multihop, phf, spy
Probe	(i) Scans the networks to identify valid IP addresses and to collect information about host (e.g., what services they offer, operating system used). (ii) Provides information to an attacker with the list of potential vulnerabilities that can later be used to launch an attack against selected systems and services.	IPsweep, portsweep

B. Classification of Intrusion Detection and Intrusion Detection Systems

Network intrusion detection has been studied for almost 20 years. Generally, an intruder's behavior is noticeably different from that of a legitimate user and hence can be detected [41]. IDSs can also be classified based on their deployment in real time.

1) Host-based IDS (HIDS): A HIDS monitors and analyzes the internals of a computing system rather than its external interfaces [42]. A HIDS might detect internal activity such as which program accesses what resources and attempts illegitimate access. An example is a word processor that suddenly and inexplicably starts modifying the system password database. Similarly, a HIDS might look at the state of a system and its stored information whether it is in RAM or in the file system or in log files or elsewhere. One can think of a HIDS as an agent that monitors whether anything or anyone internal or external has circumvented the security policy that the operating system tries to enforce.

2) Network-based IDS (NIDS): An NIDS deals with detecting intrusions in network data. Intrusions typically occur as anomalous patterns though certain techniques model the data in a sequential fashion and detect anomalous subsequences [42]. The primary reason for these anomalies is attacks launched by outside attackers who want to gain unauthorized access to the network to steal information or to disrupt the network.

In a typical setting, a network is connected to the rest of the world through the Internet. The NIDS reads all incoming packets or flows, trying to find suspicious patterns. For example, if a large number of TCP connection requests to a very large number of different ports are observed within a short time, one could assume that someone is committing a 'port scan' at some of the computer(s) in the network. Various kinds of port scans, and tools to launch them are discussed in detail in [43]. Port scans mostly try to detect incoming shell codes in the same manner that an ordinary intrusion detection system does. Apart from inspecting the incoming traffic, a NIDS also provides valuable information about intrusion from outgoing or local traffic. Some attacks might even be staged from the inside of a monitored network or network segment, and therefore, not regarded as incoming traffic at all. The data available for intrusion detection systems can be at different levels of granularity, e.g., packet level traces and IPFIX records. The data is high dimensional, typically, with a mix of categorical as well as continuous attributes.

Misuse-based intrusion detection normally searches for known intrusive patterns but anomaly-based intrusion detection tries to identify unusual patterns. Intrusion detection techniques can be classified into three types based on the detection mechanism [1], [3], [44]. This includes (i) misuse-based, (ii) anomaly-based, and (iii) hybrid, as described in Table III. Today, researchers mostly concentrate on anomaly-based network intrusion detection because it can detect known as well as unknown attacks.

There are several reasons that make intrusion detection a necessary part of the entire defense system. First, many traditional systems and applications were developed without security in mind. Such systems and applications were targeted to work in an environment, where security was never a major issue. However, the same systems and applications when deployed in the current network scenario, become major security headaches. For example, a system may be perfectly secure when it is isolated but becomes vulnerable when it is

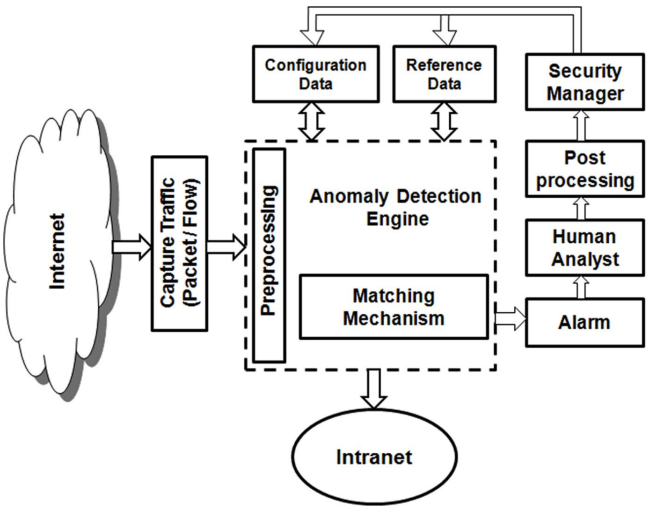

Fig. 1. A generic architecture of ANIDS

connected to the Internet. Intrusion detection provides a way to identify and thus allow response to attacks against these systems. Second, due to limitations of information security and software engineering practices, computer systems and applications may have design flaws or bugs that could be used by an intruder to attack systems or applications. As a result, certain preventive mechanisms (e.g., firewalls) may not be as effective as expected.

III. Overview of Network Anomaly Detection

Anomaly detection attempts to find patterns in data, which do not conform to expected normal behavior. The importance of anomaly detection is due to the fact that anomalies in data translate to significant (and often critical) actionable information in a wide variety of application domains [45]. For example, an anomalous traffic pattern in a computer network could mean that a hacked computer is sending out sensitive data to an unauthorized host. However, anomalies in a network may be caused by several different reasons.

As stated in [35], there are two broad categories of network anomalies: (a) performance related anomalies and (b) security related anomalies. Various examples of performance related anomalies are: broadcast storms, transient congestion, babbling node, paging across the network, and file server failure. Security related network anomalies may be due to malicious activity of intruder(s) who intentionally flood the network with unnecessary traffic to hijack the bandwidth so that legitimate users are unable to receive service(s). Security related anomalies are three types: (i) point, (ii) contextual and (iii) collective anomalies. This classification scheme is described in Table IV. However, this survey of ours is concerned with security related network anomalies only.

Currently, anomaly-based network intrusion detection is a principal focus of research and development in the field of intrusion detection. Various systems with anomaly-based network intrusion detection capabilities are becoming available, and many new schemes are being explored. However, the subject is far from mature and key issues remain to be solved before wide scale deployment of ANIDS platforms becomes practicable.

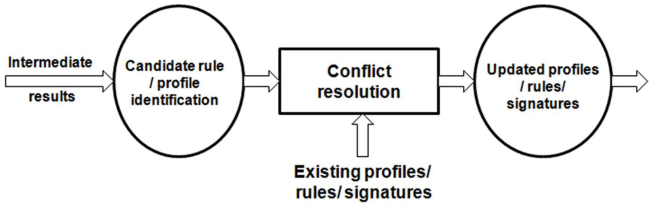

Fig. 2. Steps for updation of configuration data in ANIDS

A. Generic Architecture of ANIDS

Many NIDSs have been developed by researchers and practitioners. However, the development of an efficient ANIDS architecture is still being investigated. A generic architecture of an ANIDS is shown in Figure 1.

The main components of the generic model of the ANIDS are discussed below.

1) Anomaly detection engine: This is the heart of any network intrusion detection system. It attempts to detect occurrence of any intrusion either online or offline. However, before sending any network traffic to the detection engine, it needs preprocessing. If the attacks are known, they can be detected using the misuse detection approach. On the other hand, unknown attacks can be detected using the anomaly-based approach based on an appropriate matching mechanism.

Matching mechanism: It entails looking for a particular pattern or profile in network traffic that can be built by continuous monitoring of network behavior including known exploits or vulnerabilities. The following are some important requirements in the design of an efficient matching mechanism.

- Matching determines whether the new instance belongs to a known class defined by a high dimensional profile or not. Matching may be inexact.
- Matching must be fast.
- Effective organization of the profiles may facilitate faster search during matching.

2) Reference data: The reference data stores information about known intrusion signatures or profiles of normal behavior. Reference data needs to be stored in an efficient manner. Possible types of reference data used in the generic architecture of a NIDS are: *profile*, *signature* and *rule*. In case of an ANIDS, it is mostly profiles. The processing elements update the profiles as new knowledge about the observed behavior becomes available. These updates are performed in regular intervals in a batch oriented fashion.

3) Configuration data: This corresponds to intermediate results, e.g., partially created intrusion signatures. The space needed to store such information can be quite large. The steps for updation of the configuration data is given in Figure 2. Intermediate results need to be integrated with existing knowledge to produce consistent, up-to-date results.

4) Alarm: This component of the architecture is responsible for generation of alarm based on the indication received from the detection engine.

5) Human analyst: A human analyst is responsible for analysis, interpretation and for taking necessary action based on the alarm information provided by the detection engine. The analyst also takes necessary steps to diagnose the alarm

TABLE III
CHARACTERISTICS AND TYPES OF INTRUSION DETECTION TECHNIQUES

Technique	*Characteristics*
Misuse-based	(i) Detection is based on a set of rules or signatures for known attacks. (ii) Can detect all known attack patterns based on the reference data. (iii) How to write a signature that encompasses all possible variations of the pertinent attack is a challenging task.
Anomaly-based	(i) Principal assumption: All intrusive activities are necessarily anomalous. (ii) Such a method builds a *normal activity profile* and checks whether the system state varies from the established profile by a statistically significant amount to report intrusion attempts. (iii) Anomalous activities that are not intrusive may be flagged as intrusive. These are false positives. (iv) One should select threshold levels so that neither of the above two problems is unreasonably magnified nor the selection of features to monitor is optimized. (v) Computationally expensive because of overhead and possibly updating several system profile matrices.
Hybrid	(i) Exploits benefits of both misuse and anomaly-based detection techniques. (ii) Attempts to detect known as well as unknown attacks.

TABLE IV
ANOMALY: TYPES, CHARACTERISTICS AND EXAMPLES

Types	*Characteristics*	*Example*
Point anomaly	An instance of an individual data which has been found to be anomalous with respect to the rest of data.	Isolated network traffic instance from the normal instances at a particular time.
Contextual anomaly	(i) A data instance which has been found anomalous in a specific context. (ii) Context is induced by the structure in the dataset. (iii) Two sets of attributes are used for defining a context: (a) contextual and (b) behavioral attributes.	Time interval between purchases in credit card fraud
Collective anomaly	(i) A collection of related data instances found to be anomalous with respect to the entire dataset. (ii) Collection of events is an anomaly, but the individual events are not anomalies when they occur alone in the sequence.	A sequence such as the following: ... http-web, buffer-overflow, http-web, http-web, ftp, http-web, ssh, http-web, ssh, buffer-overflow ...

information as a post-processing activity to support reference or profile updation with the help of security manager.

6) Post-processing: This is an important module in a NIDS for post-processing of the generated alarms for diagnosis of actual attacks.

7) Capturing traffic: Traffic capturing is an important module in a NIDS. The raw traffic data is captured at both packet and flow levels. Packet level traffic can be captured using a common tool, e.g., Wireshark[1] and then preprocessed before sending to the detection engine. Flow level data in high speed networks, is comprised of information summarized from one or more packets. Some common tools to capture flow level network traffic include Nfdump[2], NfSen[3], and Cisco Netflow V.9[4].

8) Security manager: Stored intrusion signatures are updated by the Security Manager (SM) as and when new intrusions become known. The analysis of novel intrusions is a highly complex task.

B. Aspects of Network Anomaly Detection

In this section, we present some important aspects of anomaly-based network intrusion detection. The network intrusion detection problem is a classification or clustering problem formulated with the following components [3]: (i) types of input data, (ii) appropriateness of proximity measures, (iii) labelling of data, (iv) classification of methods based on the use of labelled data, (v) relevant feature identification and (vi) reporting anomalies. We discuss each of these topics in brief.

1) Types of input data: A key aspect of any anomaly-based network intrusion detection technique is the nature of the input data used for analysis. Input is generally a collection of data instances (also referred to as objects, records, points, vectors, patterns, events, cases, samples, observations, entities) [46]. Each data instance can be described using a set of attributes of binary, categorical or numeric type. Each data instance may consist of only one attribute (univariate) or multiple attributes (multivariate). In the case of multivariate data instances, all attributes may be of the same type or may be a mixture of data types. The nature of attributes determines the applicability of anomaly detection techniques.

2) Appropriateness of proximity measures: Proximity (similarity or dissimilarity) measures are necessary to solve many pattern recognition problems in classification and clustering. Distance is a quantitative degree of how far apart two objects are. Distance measures that satisfy metric properties [46] are simply called *metric* while other non-metric distance measures are occasionally called *divergence*. The choice of a proximity measure depends on the measurement type or representation of objects.

Generally, proximity measures are functions that take arguments as object pairs and return numerical values that become higher as the objects become more alike. A proximity measure is usually defined as follows.

Definition 3.1: A proximity measure S is a function $X \times X \rightarrow \mathbb{R}$ that has the following properties [47].

- Positivity: $\forall_{x,y} \in X, S(x,y) \geq 0$
- Symmetry: $\forall_{x,y} \in X, S(x,y) = S(y,x)$
- Maximality: $\forall_{x,y} \in X, S(x,x) \geq S(x,y)$

where X is the data space (also called the *universe*) and x, y are the pair of k-dimensional objects.

The most common proximity measures for numeric [48]–[50], categorical [51] and mixed type [52] data are listed in Table V. For numeric data, it is assumed that the data is represented as real vectors. The attributes take their values from a continuous domain. In Table V, we assume that there are two objects, $x = x_1, x_2, x_3 \cdots x_d$, $y = y_1, y_2, y_3 \cdots y_d$ and $\sum^{-1}$ represents the data covariance with d number of attributes, i.e., dimensions.

[1]http://www.wireshark.org/
[2]http://nfdump.sourceforge.net/
[3]http://nfsen.sourceforge.net/
[4]http://www.cisco.com

For categorical data, computing similarity or proximity measures is not straightforward owing to the fact that there is no explicit notion of ordering among categorical values. The simplest way to find similarity between two categorical attributes is to assign a similarity of 1 if the values are identical and a similarity of 0 if the values are not identical. In Table V, $S_k(x_k, y_k)$ represents per-attribute similarity. The attribute weight w_k for attribute k is computed as shown in the table. Consider a categorical dataset D containing n objects, defined over a set of d categorical attributes where A_k denotes the kth attribute. $S_k(x_k, y_k)$ is the per-attribute proximity between two values for the categorical attribute A_k. Note that $x_k, y_k \in A_k$. In Table V, IOF denotes *Inverse Occurrence Frequency* and OF denotes *Occurrence Frequency* [51].

Finally, mixed type data includes both categorical and numeric values. A common practice in clustering a mixed dataset is to transform categorical values into numeric values and then use a numeric clustering algorithm. Another approach is to compare the categorical values directly, in which two distinct values result in a distance of 1 while identical values result in a distance of 0. Of course, other measures for categorical data can be used as well. Two well-known proximity measures, general similarity coefficient and general distance coefficient [52] for mixed type data are shown in Table V. Such methods may not take into account the similarity information embedded in categorical values. Consequently, clustering may not faithfully reveal the similarity structure in the dataset [52], [53].

3) Labelling of data: The label associated with a data instance denotes if that instance is normal or anomalous. It should be noted that obtaining accurate labeled data of both normal or anomalous types is often prohibitively expensive. Labeling is often done manually by human experts and hence substantial effort is required to obtain the labeled training dataset [3]. Moreover, anomalous behavior is often dynamic in nature, e.g., new types of anomalies may arise, for which there is no labeled training data.

4) Classification of methods based on use of labelled data: Based on the extent to which labels are available, anomaly detection techniques can operate in three modes: *supervised*, *semi-supervised* and *unsupervised*.

In supervised mode, one assumes the availability of a training dataset which has labeled instances for the normal as well as the anomaly class. The typical approach in such cases is to build a predictive model for normal vs. anomaly classes. Any unseen data instance is compared against the model to determine which class it belongs to. There are two major issues that arise in supervised anomaly detection. First, anomalous instances are far fewer compared to normal instances in the training data. Issues that arise due to imbalanced class distributions have been addressed in data mining and machine learning literature [54]. Second, obtaining accurate and representative labels, especially for the anomaly class, is usually challenging. A number of techniques inject artificial anomalies in a normal dataset to obtain a labeled training dataset [55].

Semi-supervised techniques assume that the training data has labeled instances for only the normal class. Since they do not require labels for the anomaly class, they can be more readily used compared to supervised techniques. For example, in spacecraft fault detection [56], an anomaly scenario would signify an accident, which is not easy to model. The typical approach used in such techniques is to build a model for the class corresponding to normal behavior, and use the model to identify anomalies in the test data.

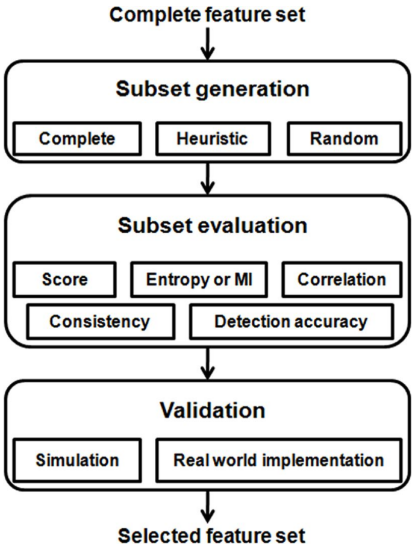

Fig. 3. Framework of feature selection process

Finally, unsupervised techniques do not require training data, and thus are potentially most widely applicable. The techniques in this category make the implicit assumption that normal instances are far more frequent than anomalies in the test data [57]. When this assumption is not true, such techniques suffer from high false alarm rates. Many semi-supervised techniques can be adapted to operate in an unsupervised mode by using a sample of the unlabeled dataset as training data [58]. Such adaptation assumes that the test data contains very few anomalies and the model learnt during training is robust to these few anomalies.

5) Relevant feature identification: Feature selection plays an important role in detecting network anomalies. Feature selection methods are used in the intrusion detection domain for eliminating unimportant or irrelevant features. Feature selection reduces computational complexity, removes information redundancy, increases the accuracy of the detection algorithm, facilitates data understanding and improves generalization. The feature selection process includes three major steps: (a) subset generation, (b) subset evaluation and (c) validation. Three different approaches for subset generation are: *complete*, *heuristic* and *random*. Evaluation functions are categorized into five [59] distinct categories: score-based, entropy or mutual information-based, correlation-based, consistency-based and detection accuracy-based. Simulation and real world implementation are the two ways to validate the evaluated subset. A conceptual framework of the feature selection process is shown in Figure 3.

Feature selection algorithms have been classified into three types: *wrapper*, *filter* and *hybrid* methods [60]. While wrapper methods try to optimize some predefined criteria with respect to the feature set as part of the selection process, filter methods

TABLE V
PROXIMITY MEASURES FOR NUMERIC, CATEGORICAL AND MIXED TYPE DATA

***Numeric* [48]**			
Name	*Measure*, $S_i(x_i, y_i)$	*Name*	*Measure*, $S_i(x_i, y_i)$
Euclidean	$\sqrt{\sum_{i=1}^{d} \lvert x_i - y_i \rvert^2}$	Weighted Euclidean	$\sqrt{\sum_{i=1}^{d} \alpha_i \lvert x_i - y_i \rvert^2}$
Squared Euclidean	$\sum_{i=1}^{d} \lvert x_i - y_i \rvert^2$	Squared-chord	$\sum_{i=1}^{d} (\sqrt{x_i} - \sqrt{y_i})^2$
Squared X^2	$\sum_{i=1}^{d} \frac{(x_i - y_i)^2}{x_i + y_i}$	City block	$\sum_{i=1}^{d} \lvert x_i - y_i \rvert$
Minkowski	$\sqrt[p]{\sum_{i=1}^{d} \lvert x_i - y_i \rvert^p}$	Chebyshev	$\max_i \lvert x_i - y_i \rvert$
Canberra	$\frac{\sum_{i=1}^{d} \lvert x_i - y_i \rvert}{x_i + y_i}$	Cosine	$\frac{\sum_{i=1}^{d} x_i y_i}{\sqrt{\sum_{i=1}^{d} x_i^2} \sqrt{\sum_{i=1}^{d} y_i^2}}$
Jaccard	$\frac{\sum_{i=1}^{d} x_i y_i}{\sum_{i=1}^{d} x_i^2 + \sum_{i=1}^{d} y_i^2 - \sum_{i=1}^{d} x_i y_i}$	Bhattacharyya	$-\ln \sum_{i=1}^{d} \sqrt{(x_i y_i)}$
Pearson	$\sum_{i=1}^{d} (x_i - y_i)^2$	Divergence	$2 \sum_{i=1}^{d} \frac{(x_i - y_i)^2}{(x_i + y_i)^2}$
Mahalanobis	$\sqrt{(x-y)^t \sum^{-1} (x-y)}$	-	-
***Categorical* [51]**			
w_k, k=1…d	*Measure*, $S_k(x_k, y_k)$	w_k, k=1…d	*Measure* $S_k(x_k, y_k)$
$\frac{1}{2}$	$Overlap = \begin{cases} 1 & \text{if } x_k = y_k \\ 0 & \text{otherwise} \end{cases}$	$\frac{1}{d}$	$Eskin = \begin{cases} 1 & \text{if } x_k = y_k \\ \frac{n_k^2}{n_k^2 + 2} & \text{otherwise} \end{cases}$
$\frac{1}{d}$	$IOF = \begin{cases} 1 & \text{if } x_k = y_k \\ \frac{1}{1 + \log f_k(x_k) x \log f_k(y_k)} & \text{otherwise} \end{cases}$	$\frac{1}{d}$	$OF = \begin{cases} 1 & \text{if } x_k = y_k \\ \frac{1}{1 + \log \frac{N}{f_k(x_k)} x \log \frac{N}{f_k(y_k)}} & \text{otherwise} \end{cases}$
***Mixed* [52]**			
Name	*Measure*	*Name*	*Measure*
General Similarity Coefficient	$s_{gsc}(x, y) = \frac{1}{\sum_{k=1}^{d} w(x_k, y_k)} \sum_{k=1}^{d} w(x_k, y_k) s(x_k, y_k)$, • For *numeric* attributes, $s(x_k, y_k) = 1 - \frac{\lvert x_k - y_k \rvert}{R_k}$, where R_k is the range of the kth attribute; $w(x_k, y_k) = 0$ if x or y has missing value for the kth attribute; otherwise $w(x_k, y_k) = 1$. • For *categorical* attributes, $s(x_k, y_k) = 1$ if $x_k = y_k$; otherwise $s(x_k, y_k) = 0$; $w(x_k, y_k) = 0$ if data point x or y has missing value at k^{th} attribute; otherwise $w(x_k, y_k) = 1$.	General Distance Coefficient	$d_{gdc}(x, y) = \left(\frac{1}{\sum_{k=1}^{d} w(x_k, y_k)} \sum_{k=1}^{d} w(s_k, y_k) d^2(x_k, y_k) \right)^{\frac{1}{2}}$, where $d^2(x_k, y_k)$ is the squared distance for the kth attribute; $w(x_k, y_k)$ is the same as in General Similarity Coefficient. • For *numeric* attributes, $d(x_k, y_k) = \frac{\lvert x_k - y_k \rvert}{R_k}$, where R_k is the range of kth attribute. • For *categorical* attributes, $d(x_k, y_k) = 0$ if $x_k = y_k$; otherwise $d(x_k, y_k) = 1$.

rely on the general characteristics of the training data to select features that are independent of each other and are highly dependent on the output. The hybrid feature selection method attempts to exploit the salient features of both wrapper and filter methods [60].

An example of wrapper-based feature selection method is [61], where the authors propose an algorithm to build a lightweight IDS by using modified Random Mutation Hill Climbing (RMHC) as a search strategy to specify a candidate subset for evaluation, and using a modified linear Support Vector Machines (SVMs) based iterative procedure as a wrapper approach to obtain an optimum feature subset. The authors establish the effectiveness of their method in terms of efficiency in intrusion detection without compromising the detection rate. An example filter model for feature selection is [62], where the authors fuse correlation-based and minimal redundancy-maximal-relevance measures. They evaluate their method on benchmark intrusion datasets for classification accuracy. Several other methods for feature selection are [39], [63]–[65].

6) Reporting anomalies: An important aspect of any anomaly detection technique is the manner in which anomalies are reported [3]. Typically, the outputs produced by anomaly detection techniques are of two types: (a) a *score*, which is a value that combine (i) distance or deviation with reference to a set of profiles or signatures, (ii) influence of the majority in its neighborhood, and (iii) distinct dominance of the relevant subspace (as discussed in Section III-B5). (b) a *label*, which is a value (normal or anomalous) given to each test instance. Usually the labelling of an instance depends on (i) the size of groups generated by an unsupervised technique, (ii) the compactness of the group(s), (iii) majority voting based on the outputs given by multiple indices (several example indices are given in Table VI), or (iv) distinct dominance of the subset of features.

IV. METHODS AND SYSTEMS FOR NETWORK ANOMALY DETECTION

The classification of network anomaly detection methods and systems that we adopt is shown in Figure 4. This scheme is based on the nature of algorithms used. It is not straightforward to come up with a classification scheme for network anomaly detection methods and systems, primarily because there is substantial overlap among the methods used

TABLE VI
CLUSTER VALIDITY MEASURES

Reference	*Name of Index*	*Formula*	*Remark(s)*
Dunn [66]	Dunn Index	$DI = \frac{d_{min}}{d_{max}}$, where d_{min} denotes the smallest distance between two objects from different clusters, d_{max} the largest distance within the same cluster.	(i) Can identify dense and well-separated clusters. (ii) High Dunn index is more desired for a clustering algorithm. (iii) May not perform well with noisy data.
Davies et al. [67]	Davies Bouldin's index	$DB = \frac{1}{n}\sum_{i=1,i\neq j}^{n} max(\frac{\sigma_j+\sigma_j}{d(c_i,c_j)})$, where n is the number of clusters; σ_i is the average distance of all patterns in cluster i to their cluster center, c_i; σ_j is the average distance of all patterns in cluster j to their cluster center, c_j; and $d(c_i, c_j)$ represents the proximity between the cluster centers c_i and c_j.	(i) Validation is performed using cluster quantities and features inherent to the dataset. (ii) For compact clustering, DB values should be as minimum as possible. (iii) It is not designed to accommodate overlapping clusters.
Hubert and Schultz [68]	C-index	$C = \frac{S-S_{min}}{S_{max}-S_{min}}$, where S is the sum of distances over all pairs of objects form the same cluster, n is the number of those pairs, S_{min} and S_{max} are the sum of n smallest distances and n largest distances, respectively.	It needs to be minimized for better clustering.
Baker and Hubert [69]	Gamma Index	$G = \frac{S+ - S-}{S+ + S-}$, where (S+) represents the number of times that a pair of samples not clustered together have a larger separation than a pair that were in the same clusters; $(S-)$ represents reverse outcome.	This measure is widely used for hierarchical clustering.
Rohlf [70]	G+ Index	$G+ = \frac{2(S-)}{n*(n-1)}$, where $(S-)$ is defined as for gamma index and n is the number of within cluster distances.	It uses minimum value to determine the number of clusters in the data.
Rousseeuw [71]	Silhouette Index	$SI = \frac{b_i-a_i}{max\{a_i,b_i\}}$, where a_i is the average dissimilarity of the ith object to all other objects in the same cluster; b_i is the minimum of average dissimilarity of the object from all objects in other clusters;	This index cannot be applied to datasets with subclusters.
Goodman and Kruskal [72]	Goodman-Kruskal index	$GK = \frac{N_c-N_d}{N_c+N_d}$, where N_c and N_d are the numbers of concordant and disconcordant quadruples, respectively.	(i) It is robust in outliers detection. (ii) It requires high computation complexity in comparison to C-index.
Jaccard [73]	Jaccard Index	$JI = \frac{a}{a+b+c}$, where a denotes the number of pairs of points with the same label in C and assigned to the same cluster in k, b denotes the number of pairs with the same label, but in different clusters and c denotes the number of pairs in the same cluster, but with different class labels.	It uses less information than Rand index measure.
Rand [74]	Rand Index	$RI = \frac{a+d}{a+b+c+d}$, where d denotes the number of pairs with a different label in C that were assigned to a different cluster in k, rest are same with JI.	It gives equal weights to false positives and false negatives during computation.
Bezdek [75]	Partition coefficient	$PC = \frac{1}{n}\sum_{i=1}^{N}\sum_{j=1}^{n_c} u_{ij}^2$, where n_c is the number of clusters, N is the number of objects in the dataset, u_{ij} is the degree of membership.	(i) It finds the number of overlaps between clusters, (ii) It lacks connection with dataset.
Bezdek [76]	Classification entropy	$CE = \frac{1}{N}\sum_{i=1}^{k}\sum_{j=1}^{n} u_{ij}log(u_{ij})$, same with partition coefficient.	It measures the fuzziness of the cluster partitions.
Xie and Beni [77]	Xie-Beni Index	$XB = \frac{\pi}{N.d_{min}}$, where $\pi = \frac{\sigma_i}{n_i}$, is called compactness of cluster i. Since n_i is the number of points in cluster i, σ, is the average variation in cluster i; $d_{min} = min\|\|k_i - k_j\|\|$.	(i) It combines the properties of membership degree and the geometric structure of dataset. (ii) Smaller XB means more compact and better separated clusters.

in the various classes in any particular scheme we may adopt. We have decided on six distinct classes of methods and systems. We call them *statistical*, *classification-based*, *clustering and outlier-based*, *soft computing*, *knowledge-based* and *combination learners*. Most methods have subclasses as given in Figure 4. Figure 5 shows the approximate statistics of papers published in each category.

We distinguish between network anomaly detection methods and systems in this paper, although such a distinction is difficult to make sometimes. A network intrusion detection system (NIDS) usually integrates a network intrusion detection method within an architecture that comprises other associated sub-systems to build stand-alone practical system that can perform the entire gamut of activities needed for intrusion detection. We present several NIDSs with their architectures and components as we discuss various anomaly detection categories.

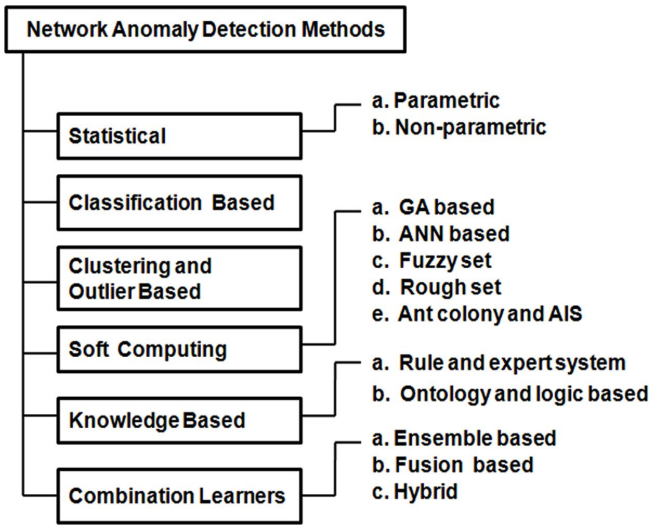

Fig. 4. Classification of network anomaly detection methods (GA-Genetic Algorithm, ANN-Artificial Neural Network, AIS-Artificial Immune System)

A. Statistical methods and systems

Statistically speaking, an anomaly is an observation which is suspected of being partially or wholly irrelevant because it is not generated by the stochastic model assumed [78]. Normally, statistical methods fit a statistical model (usually for normal behavior) to the given data and then apply a statistical inference test to determine if an unseen instance belongs to this model. Instances that have a low probability to be generated from the learnt model based on the applied test statistic are declared anomalies. Both parametric and non-parametric techniques have been applied to design statistical models for anomaly detection. While parametric techniques

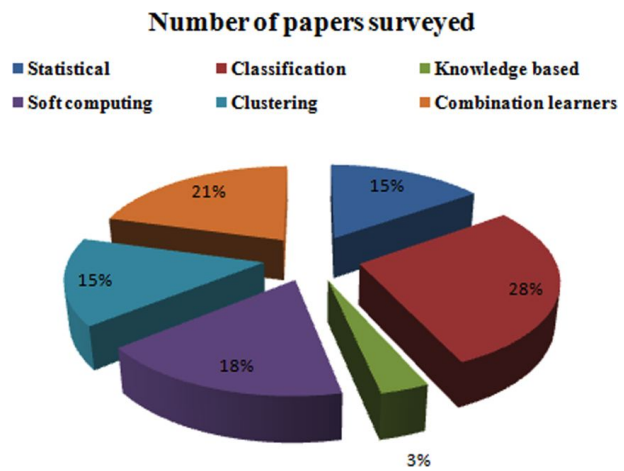

Fig. 5. Statistics of the surveyed papers during the years 2000 to 2012

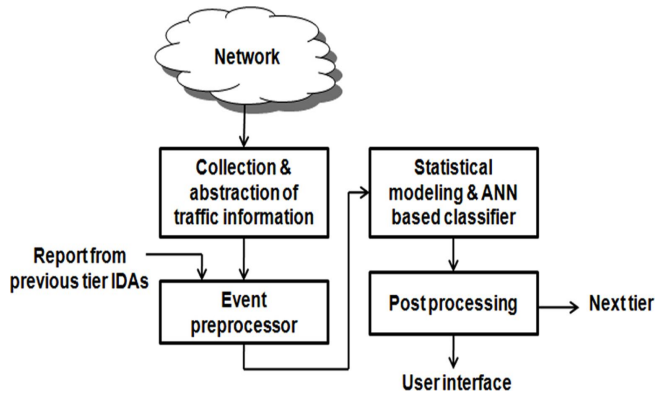

Fig. 6. Architecture of HIDE system

assume knowledge of the underlying distribution and estimate the parameters from the given data [79], non-parametric techniques do not generally assume knowledge of the underlying distribution [80].

An example of a statistical IDS is HIDE [33]. HIDE is an anomaly-based network intrusion detection system, that uses statistical models and neural network classifiers to detect intrusions. HIDE is a distributed system, which consists of several tiers with each tier containing several Intrusion Detection Agents (IDAs). IDAs are IDS components that monitor the activities of a host or a network. The probe layer (i.e., top layer as shown in Figure 6) collects network traffic at a host or in a network, abstracts the traffic into a set of statistical variables to reflect network status, and periodically generates reports to the event preprocessor. The event preprocessor layer receives reports from both the probe and IDAs of lower tiers, and converts the information into the format required by the statistical model. The statistical processor maintains a reference model of typical network activities, compares reports from the event preprocessor with the reference models, and forms a stimulus vector to feed into the neural network classifier. The neural network classifier analyzes the stimulus vector from the statistical model to decide whether the network traffic is normal. The post-processor generates reports for the agents at higher tiers. A major attraction of HIDE is its ability to detect UDP flooding attacks even with attack intensity as low as 10% of background traffic.

Of the many statistical methods and NIDSs [79], [81]–[89] only a few are described below in brief.

Bayesian networks [90] are capable of detecting anomalies in a multi-class setting. Several variants of the basic technique have been proposed for network intrusion detection and for anomaly detection in text data [3]. The basic technique assumes independence among different attributes. Several variations of the basic technique that capture the conditional dependencies among different attributes using more complex Bayesian networks have also been proposed. For example, the authors of [91] introduce an event classification-based intrusion detection scheme using Bayesian networks. The Bayesian decision process improves detection decision to significantly reduce false alarms.

Manikopoulos and Papavassiliou [81] introduce a hierarchical multi-tier multi-window statistical anomaly detection system to operate automatically, adaptively, and proactively. It applies to both wired and wireless ad-hoc networks. This system uses statistical modeling and neural network classification to detect network anomalies and faults. The system achieves high detection rate along with low misclassification rate when the anomaly traffic intensity is at 5% of the background traffic but the detection rate is lower at lower attack intensity levels such as 1% and 2%.

Association rule mining [92], conceptually a simple method based on counting of co-occurrences of items in transactions databases, has been used for one-class anomaly detection by generating rules from the data in an unsupervised fashion. The most difficult and dominating part of an association rules discovery algorithm is to find the itemsets that have strong support. Mahoney and Chan [83] present an algorithm known as LERAD that learns rules for finding rare events in time-series data with long range dependencies and finds anomalies in network packets over TCP sessions. LERAD uses an Apriori-like algorithm [92] that finds conditional rules over nominal attributes in a time series, e.g., a sequence of inbound client packets. The antecedent of a created rule is a conjunction of equalities, and the consequent is a set of allowed values, e.g., *if port=80 and word3=HTTP/1.0 then word1=GET or POST.* A value is allowed if it is observed in at least one training instance satisfying the antecedent. The idea is to identify rare anomalous events: those which have not occurred for a long time and which have high anomaly score. LERAD is a two-pass algorithm. In the first pass, a candidate rule set is generated from a random sample of training data comprised of attack-free network traffic. In the second pass, rules are trained by obtaining the set of allowed values for each antecedent.

A payload-based anomaly detector for intrusion detection known as PAYL is proposed in [84]. PAYL attempts to detect the first occurrence of a worm either at a network system gateway or within an internal network from a rogue device and to prevent its propagation. It employs a language-independent n-gram based statistical model of sampled data streams. In fact, PAYL uses only a 1-gram model (i.e., it looks at the distribution of values contained within a single byte) which requires a linear scan of the data stream and a small 256-

element histogram. In other words, for each ASCII character in the range 0-255, it computes its mean frequency as well as the variance and standard deviation. Since payloads (i.e., arriving or departing contents) at different ports differ in length, PAYL computes these statistics for each specific observed payload length for each port open in the system. It first observes many exemplar payloads during the training phase and computes the payload profiles for each port for each payload length. During detection, each incoming payload is scanned and statistics are computed. The new payload distribution is compared against the model created during training. If there is a significant difference, PAYL concludes that the packet is anomalous and generates an alert. The authors found that this simple approach works surprisingly well.

Song et al. [85] propose a conditional anomaly detection method for computing differences among attributes and present three different expectation-maximization algorithms for learning the model. They assume that the data attributes are partitioned into *indicator* attributes and *environmental* attributes based on the decision taken by the user regarding which attributes indicate an anomaly. The method learns the typical indicator attribute values and observes subsequent data points, and labels them as anomalous or not, based on the degree the indicator attribute values differ from the usual indicator attribute values. However, if the indicator attribute values are not conditioned on environmental attributes values, the indicator attributes are ignored effectively. The precision/recall of this method is greater than 90 percent.

Lu and Ghorbani [87] present a network signal modeling technique for anomaly detection by combining wavelet approximation and system identification theory. They define and generate fifteen relevant traffic features as input signals to the system and model daily traffic based on these features. The output of the system is the deviation of the current input signal from the normal or regular signal behavior. Residuals are passed to the IDS engine to take decisions and obtain 95% accuracy in the daily traffic.

Wattenberg et al. [88] propose a method to detect anomalies in network traffic, based on a nonrestricted α-stable first-order model and statistical hypothesis testing. The α-stable function is used to model the marginal distribution of real traffic and classify them using the Generalized Likelihood Ratio Test. They detect two types of anomaly including floods and flash-crowds with promising accuracy. In addition, a nonparametric adaptive CSUM (Cumulative Sum) method for detecting network intrusions is discussed in [89].

In addition to the detection methods, there are several statistical NIDSs. As mentioned earlier, a NIDS includes one or more intrusion detection methods that are integrated with other required sub-systems necessary to create a practically suitable system. We discuss a few below.

N@G (Network at Guard) [93] is a hybrid IDS that exploits both misuse and anomaly approaches. N@G has both network and host sensors. Anomaly-based intrusion detection is pursued using the chi-square technique on various network protocol parameters. It has four detection methodologies viz., data collection, signature-based detection, network access policy violation and protocol anomaly detection as a part of its network sensor. It includes audit trails, log analysis, statistical analysis and host access policies as components of the host sensor. The system has a separate IDS server, i.e., a management console to aggregate alerts from the various sensors with a user interface, a middle-tier and a data management component. It provides real time protection against malicious changes to network settings on client computers, which includes unsolicited changes to the Windows Hosts file and Windows Messenger service.

FSAS (Flow-based Statistical Aggregation Scheme) [94] is a flow-based statistical IDS. It comprises of two modules: *feature generator* and *flow-based detector*. In the feature generator, the event preprocessor module collects the network traffic of a host or a network. The event handlers generate reports to the flow management module. The flow management module efficiently determines if a packet is part of an existing flow or it should generate a new flow key. By inspecting flow keys, this module aggregates flows together, and dynamically updates per-flow accounting measurements. The event time module periodically calls the feature extraction module to convert the statistics regarding flows into the format required by the statistical model. The neural network classifier classifies the score vectors to prioritize flows with the amount of maliciousness. The higher the maliciousness of a flow, the higher is the possibility of the flow being an attacker.

In addition to their inherent ability to detect network anomalies, statistical approaches have a number of additional distinct advantages as well.

- They do not require prior knowledge of normal activities of the target system. Instead, they have the ability to learn the expected behavior of the system from observations.
- Statistical methods can provide accurate notification or alarm generation of malicious activities occurring over long periods of time, subject to setting of appropriate thresholding or parameter tuning.
- They analyze the traffic based on the theory of abrupt changes, i.e., they monitor the traffic for a long time and report an alarm if any abrupt change (i.e., significant deviation) occurs.

Drawbacks of the statistical model for network anomaly detection include the following.

- They are susceptible to being trained by an attacker in such a way that the network traffic generated during the attack is considered normal.
- Setting the values of the different parameters or metrics is a difficult task, especially because the balance between false positives and false negatives is an issue. Moreover, a statistical distribution per variable is assumed, but not all behaviors can be modeled using stochastic methods. Furthermore, most schemes rely on the assumption of a quasi-stationary process [6], which is not always realistic.
- It takes a long time to report an anomaly for the first time because the building of the models requires extended time.
- Several hypothesis testing statistics can be applied to detect anomalies. Choosing the best statistic is often not straightforward. In particular, as stated in [88] constructing hypothesis tests for complex distributions that are required to fit high dimensional datasets is nontrivial.

TABLE VII
COMPARISON OF STATISTICAL NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*No. of parameters*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Eskin [79]	2000	2	O	N	P	Numeric	DARPA99	C_4	Probability Model
Manikopoulos and Papavassiliou [81]	2002	3	D	N	P	Numeric	Real-life	C_2, C_5	Statistical model with neural network
Mahoney and Chan [83]	2003	2	C	N	P	-	DARPA99	C_1	LERAD algorithm
Chan et al. [82]	2003	2	C	N	P	Numeric	DARPA99	C_1	Learning Rules
Wang and Stolfo [84]	2004	3	C	N	P	Numeric	DARPA99	C_1	Payload-based algorithm
Song et al. [85]	2007	3	C	N	P	Numeric	KDDcup99	Intrusive pattern	Gaussian Mixture Model
Chhabra et al. [86]	2008	2	D	N	P	Numeric	Real time	C_6	FDR method
Lu and Ghorbani [87]	2009	3	C	N	P, F	Numeric	DARPA99	C_1	Wavelet Analysis
Wattenberg et al. [88]	2011	4	C	N	P	Numeric	Real-time	C_2	GLRT Model
Yu [89]	2012	1	C	N	P	Numeric	Real-time	C_2	Adaptive CUSUM

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, C_5-remote to local, and C_6-anomalous

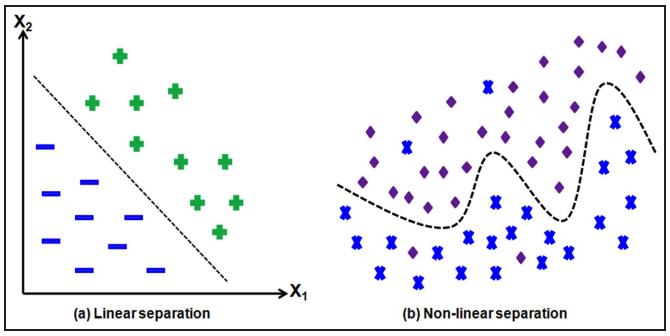

Fig. 7. Linear and non-linear classification in 2-D

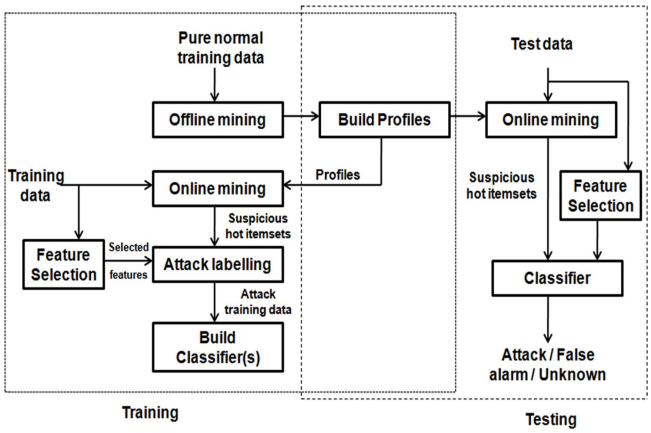

Fig. 8. Architecture of ADAM system

- Histogram-based techniques are relatively simple to implement, but a key shortcoming of such techniques for multivariate data is that they are not able to capture interactions among the attributes.

A comparison of a few statistical network anomaly detection methods is given in Table VII.

B. Classification-based methods and systems

Classification is the problem of identifying which of a set of categories a new observation belongs to, on the basis of a training set of data containing observations whose category membership is known. Assuming we have two classes whose instances are shown as $+$ and $-$, and each object can be defined in terms of two attributes or features x_1 and x_2, linear classification tries to find a line between the classes as shown in Figure 7(a). The classification boundary may be non-linear as in Figure 7(b). In intrusion detection, the data is high dimensional, not just two. The attributes are usually mixed, numeric and categorial as discussed earlier.

Thus, classification techniques are based on establishing an explicit or implicit model that enables categorization of network traffic patterns into several classes [95]–[100]. A singular characteristic of these techniques is that they need labeled data to train the behavioral model, a procedure that places high demands on resources [101]. In many cases, the applicability of machine learning principles such as classification coincides with that of statistical techniques, although the former technique is focused on building a model that improves its performance on the basis of previous results [7]. Several classification-based techniques (e.g., k-nearest neighbor, support vector machines, and decision trees) have been applied to anomaly detection in network traffic data.

An example of classification-based IDS is Automated Data Analysis and Mining (ADAM) [32] that provides a testbed for detecting anomalous instances. An architecture diagram of ADAM is shown in Figure 8. ADAM exploits a combination of classification techniques and association rule mining to discover attacks in a tcpdump audit trail. First, ADAM builds a repository of "normal" frequent itemsets from attack-free periods. Second, ADAM runs a sliding-window based online algorithm that finds frequent itemsets in the connections and compares them with those stored in the normal itemset repository, discarding those that are deemed normal. ADAM uses a classifier which has been trained to classify suspicious connections as either a known type of attack or an unknown type or a false alarm.

A few classification-based network anomaly detection methods and NIDSs are described below in brief.

Abbes et al. [102] introduce an approach that uses decision trees with protocol analysis for effective intrusion detection. They construct an adaptive decision tree for each application layer protocol. Detection of anomalies classifies data records

into two classes: benign and anomalies. The anomalies include a large variety of types such as DoS, scans, and botnets. Thus, multi-class classifiers are a natural choice, but like any classifier they require expensive hand-labeled datasets and are also not able to identify unknown attacks.

Wagner et al. [103] use *one-class classifiers* that can detect new anomalies, i.e., data points that do not belong to the learned class. In particular, they use a one-class SVM classifier proposed by Schölkopf et al. [104]. In such a classifier, the training data is presumed to belong to only one class, and the learning goal during training is to determine a function which is positive when applied to points on the circumscribed boundary around the training points and negative outside. This is also called *semi-supervised classification*. Such an SVM classifier can be used to identify outliers and anomalies. The authors develop a special kernel function that projects data points to a higher dimension before classification. Their kernel function takes into consideration properties of Netflow data and enables determination of similarity between two windows of IP flow records. They obtain 92% accuracy on average for all attacks classes.

Classification-based anomaly detection methods can usually give better results than unsupervised methods (e.g, clustering-based) because of the use of labeled training examples. In traditional classification, new information can be incorporated by re-training with the entire dataset. However, this is time-consuming. Incremental classification algorithms [105] make such training more efficiently. Although classification-based methods are popular, they cannot detect or predict unknown attack or event until relevant training information is fed for retraining.

For a comparison of several classification-based network anomaly detection methods, see Table VIII.

Several authors have used a combination of classifiers and clustering for network intrusion detection leveraging the advantages of the two methods. For example, Muda et al. [107] present a two stage model for network intrusion detection. Initially, k-means clustering is used to group the samples into three clusters: $C1$ to group attack data such as Probe, U2R and R2L; $C2$ to group DoS attack data, and $C3$ for normal non-attack data. The authors achieve this by initializing the cluster centers with the mean values obtained from known data points of appropriate groups. Since the initial centroids are obtained from known labeled data, the authors find that k-means clustering is very good at clustering the data into the three classes. Next, the authors use a Naive Bayes classifier to classify the data in the final stage into the five more accurate classes, Normal, DoS, Probe, R2L and U2R.

Gaddam et al. [96] present a method to detect anomalous activities based on a combined approach that uses the k-means clustering algorithm and the ID3 algorithm for decision tree learning [108]. In addition to descriptive features, each data instance includes a label saying whether the instance is normal or anomalous. The first stage of the algorithm partitions the training data into k clusters using Euclidean distance similarity. Obviously, the clustering algorithm does not consider the labels on instances. The second stage of the algorithm builds a decision tree on the instances in a cluster. It does so for each cluster so that k separate decision trees are built. The purpose of building decision trees is to overcome two problems that k-means faces: a) *forced assignment*: if the value of k is lower than the number of natural groups, dissimilar instances are forced into the same cluster, and b) *class dominance*, which arises when a cluster contains a large number of instances from one class, and fewer numbers of instances from other classes. The hypothesis is that a decision tree trained on each cluster learns the sub groupings (if any) present within each cluster by partitioning the instances over the feature space. To obtain a final decision on classification of a test instance, the decisions of the k-means and ID3 algorithms are combined using two rules: (a) the nearest-neighbor rule and (b) the nearest-consensus rule. The authors claim that the detection accuracy of the k-means+ID3 method is very high with an extremely low false positive rate on network anomaly data.

Support Vector Machines (SVMs) are very successful maximum margin linear classifiers [109]. However, SVMs take a long time for training when the dataset is very large. Khan et al. [106] reduce the training time for SVMs when classifying large intrusion datasets by using a hierarchical clustering method called Dynamically Growing Self-Organizing Tree (DGSOT) intertwined with the SVMs. DGSOT, which is based on artificial neural networks, is used to find the boundary points between two classes. The boundary points are the most qualified points to train SVMs. An SVM computes the maximal margins separating the two classes of data points. Only points closest to the margins, called support vectors, affect the computation of these margins. Other points can be discarded without affecting the final results. Khan et al. approximate support vectors by using DGSOT. They use clustering in parallel with the training of SVMs, without waiting till the end of the building of the tree to start training the SVM. The authors find that their approach significantly improves training time for the SVMs without sacrificing generalization accuracy, in the context of network anomaly detection.

In addition to the several detection methods viz., noted above, we also discuss a classification-based IDS known as DNIDS (Dependable Network Intrusion Detection System) [110]. This IDS is developed based on the Combined Strangeness and Isolation measure of the k-Nearest Neighbor (CSI-KNN) algorithm. DNIDS can effectively detect network intrusion while providing continued service under attack. The intrusion detection algorithm analyzes characteristics of network data by employing two measures: strangeness and isolation. These measures are used by a correlation unit to raise intrusion alert along with the confidence information. For faster information, DNIDS exploits multiple CSF-KNN classifiers in parallel. It also includes a intrusion tolerant mechanism to monitor the hosts and the classifiers running on them, so that failure of any component can be handled carefully. Sensors capture network packets from a network segment and transform them into connection-based vectors. The Detector is a collection of CSI-KNN classifiers that analyze the vectors supplied by the sensors. The Manager, Alert Agents, and Maintenance Agents are designed for intrusion tolerance and are installed on a secure administrative server called Station. The Manager executes the tasks of generating mobile agents and dispatching them for task execution.

TABLE VIII
COMPARISON OF CLASSIFICATION-BASED NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*No. of parameters*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Tong et al. [95]	2005	4	O	N	P	Numeric	DARPA99, TCPSTAT	C_1	KPCC model
Gaddam et al. [96]	2007	3	C	N	P	Numeric	NAD, DED, MSD	C_1	k-means+ID3
Khan et al. [106]	2007	3	C	N	P	Numeric	DARPA98	C_1	DGSOT + SVM
Das et al. [97]	2008	3	O	N	P	Categorical	KDDcup99	C_1	APD Algorithm
Lu and Tong [98]	2009	2	O	N	P	Numeric	DARPA99	C_1	CUSUM-EM
Qadeer et al. [99]	2010	-	C	R	P	-	Real time	C_2	Traffic statistics
Wagner et al.[103]	2011	2	C	R	F	Numeric	Flow Traces	C_2	Kernel OCSVM
Muda et al. [107]	2011	2	O	N	O	Numeric	KDDcup99	C_1	KMNB algorithm
Kang et al. [100]	2012	2	O	N	P	Numeric	DARPA98	C_1	Differentiated SVDD

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, and C_5-remote to local

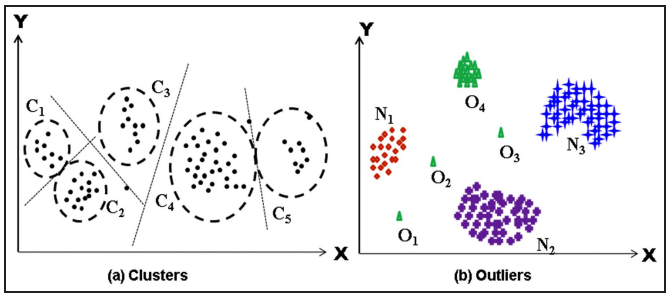

Fig. 9. Clustering and outliers in 2-D, where C_is are clusters in (*a*) and O_is are outliers in (*b*)

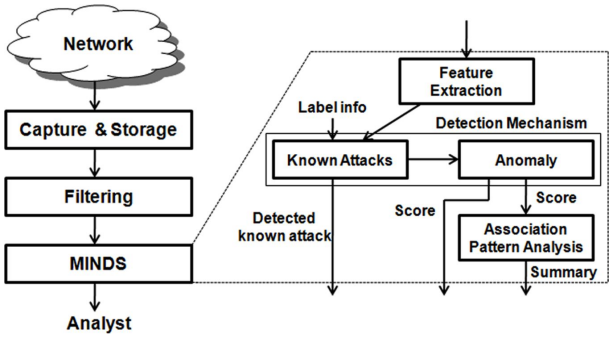

Fig. 10. Architecture of MINDS system

Classification-based anomaly detection approaches are popular for detecting network anomalies. The following are some advantages.

- These techniques are flexible for training and testing. They are capable of updating their execution strategies with the incorporation of new information. Hence, adaptability is possible.
- They have a high detection rate for known attacks subject to appropriate threshold setting.

Though such methods are popular they have the following disadvantages.

- The techniques are highly dependent on the assumptions made by the classifiers.
- They consume more resources than other techniques.
- They cannot detect or predict unknown attack or event until relevant training information is fed.

C. Clustering and Outlier-based methods and systems

Clustering is the task of assigning a set of objects into groups called *clusters* so that the objects in the same cluster are more similar in some sense to each other than to those in other clusters. Clustering is used in explorative data mining. For example, if we have a set of unlabeled objects in two dimensions, we may be able to cluster them into 5 clusters by drawing circles or ellipses around them, as in Figure 9(a). Outliers are those points in a dataset that are highly unlikely to occur given a model of the data, as in Figure 9(b). Examples of outliers in a simple dataset are seen in [111]. Clustering and outlier finding are examples of unsupervised machine learning.

Clustering can be performed in network anomaly detection in an offline environment. Such an approach adds additional depth to the administrators' defenses, and allows them to more accurately determine threats against their network through the use of multiple methods on data from multiple sources. Hence, the extensive amount of activities that may be needed to detect intrusion near real time in an online NIDS may be obviated, achieving efficiency [112].

For example, MINDS (Minnesota Intrusion Detection System) [34] is a data mining-based system for detecting network intrusions. The architecture of MINDS is given in Figure 10. It accepts NetFlow data collected through flow tools as input. Flow tools only capture packet header information and build one way sessions of flows. The analyst uses MINDS to analyze these data files in batch mode. The reason for running the system in batch mode is not due to the time it takes to analyze these files, but because it is convenient for the analyst to do so. Before data is fed into the anomaly detection module, a data filtering step is executed to remove network traffic in which the analyst is not interested.

The first step of MINDS is to extract important features that are used. Then, it summarizes the features based on time windows. After the feature construction step, the known attack detection module is used to detect network connections that correspond to attacks for which signatures are available, and to remove them from further analysis. Next, an outlier technique is activated to assign an anomaly score to each network connection. A human analyst then looks at only the most anomalous connections to determine if they are actual attacks or represent other interesting behavior. The association pattern

analysis module of this system is dedicated to summarize the network connections as per the assigned anomaly rank. The analyst provides feedback after analyzing the summaries created and decides whether these summaries are helpful in creating new rules that may be used in known attack detection.

Clustering techniques are frequently used in anomaly detection. These include single-link clustering algorithms, k-means (squared error clustering), and hierarchical clustering algorithms to mention a few [113]–[118].

Sequeira and Zaki [119] present an anomaly-based intrusion detection system known as ADMIT that detects intruders by creating user profiles. It keeps track of the sequence of commands a user uses as he/she uses a computer. A user profile is represented by clustering the sequences of the user's commands. The data collection and processing are thus host-based. The system clusters a user's command sequence using LCS (Longest Common Subsequence) as the similarity metric. It uses a dynamic clustering algorithm that creates an initial set of clusters and then refines them by splitting and merging as necessary. When a new user types a sequence of commands, it compares the sequence to profiles of users it already has. If it is a long sequence, it is broken up to a number of smaller sequences. A sequence that is not similar to a normal user's profile is considered anomalous. One anomalous sequence is tolerated as noise, but a sequence of anomalous sequences typed by one single user causes the user to be marked as masquerader or concept drift. The system can also use incremental clustering to detect masqueraders.

Zhang et al. [115] report a distributed intrusion detection algorithm that clusters the data twice. The first clustering chooses candidate anomalies at Agent IDSs, which are placed in a distributed manner in a network and a second clustering computation attempts to identify true attacks at the central IDS. The first clustering algorithm is essentially the same as the one proposed by [120]. At each agent IDS, small clusters are assumed to contain anomalies and all small clusters are merged to form a single candidate cluster containing all anomalies. The candidate anomalies from various Agent IDSs are sent to the central IDS, which clusters again using a simple single-link hierarchical clustering algorithm. It chooses the smallest k clusters as containing true anomalies. They obtain 90% attacks detection rate on test intrusion data.

Worms are often intelligent enough to hide their activities and evade detection by IDSs. Zhuang et al. [121] propose a method called PAIDS (Proximity-Assisted IDS) to identify the new worms as they begin to spread. PAIDS works differently from other IDSs and has been designed to work collaboratively with existing IDSs such as an anomaly-based IDS for enhanced performance. The goal of the designers of PAIDS is to identify new and intelligent fast-propagating worms and thwarting their spread, particularly as the worm is just beginning to spread. Neither signature-based nor anomaly-based techniques can achieve such capabilities. Zhuang et al.'s approach is based mainly on the observation that during the starting phase of a new worm, the infected hosts are clustered in terms of geography, IP address and maybe, even DNSes used.

Bhuyan et al. [122] present an unsupervised network anomaly detection method for large intrusion datasets. It exploits tree-based subspace clustering and an ensemble-based cluster labelling technique to achieve better detection rate over real life network traffic data for the detection of known as well as unknown attacks. They obtain 98% detection rate on average in detecting network anomalies.

Some advantages of using clustering are given below.

- For a partitioning approach, if k can be provided accurately then the task is easy.
- Incremental clustering (in supervised mode) techniques are effective for fast response generation.
- It is advantageous in case of large datasets to group into similar number of classes for detecting network anomalies, because it reduces the computational complexity during intrusion detection.
- It provides a stable performance in comparison to classifiers or statistical methods.

Drawbacks of clustering-based methods include the following.

- Most techniques have been proposed to handle continuous attributes only.
- In clustering-based intrusion detection techniques, an assumption is that the larger clusters are normal and smaller clusters are attack or intrusion [57]. Without this assumption, it is difficult to evaluate the technique.
- Use of an inappropriate proximity measure affects the detection rate negatively.
- Dynamic updation of profiles is time consuming.

Several outlier-based network anomaly identification techniques are available in [18]. When we use outlier-based algorithms, the assumption is that anomalies are uncommon events in a network. Intrusion datasets usually contain mixed, numeric and categorial attributes. Many early outlier detection algorithms worked with continuous attributes only; they ignored categorial attributes or modeled them in manners that caused considerable loss of information.

To overcome this problem, Otey et al. [123] develop a distance measure for data containing a mix of categorical and continuous attributes and use it for outlier-based anomaly detection. They define an anomaly score which can be used to identify outliers in mixed attribute space by considering dependencies among attributes of different types. Their anomaly score function is based on a global model of the data that can be easily constructed by combining local models built independently at each node. They develop an efficient one-pass approximation algorithm for anomaly detection that works efficiently in distributed detection environments with very little loss of detection accuracy. Each node computes its own outliers and the inter-node communication needed to compute global outliers is not significant. In addition, the authors show that their approach works well in dynamic network traffic situations where data, in addition to being streaming, also changes in nature as time progresses leading to concept drift.

Bhuyan et al. [124] introduce an outlier score function to rank each candidate object w.r.t. the reference points for network anomaly detection. The reference points are computed from the clusters obtained from variants of the k-means clustering technique. The method is effective on real life intrusion datasets.

Some of the advantages of outlier-based anomaly detection are the following.

- It is easy to detect outliers when the datasets are smaller in size.
- Bursty and isolated attacks can be identified efficiently using this method.

Drawbacks of outlier-based anomaly detection include the following.

- Most techniques use both clustering and outlier detection. In such cases the complexity may be high in comparison to other techniques.
- The techniques are highly parameter dependent.

A comparison of a few clustering and outlier-based network anomaly detection methods is given in Table IX.

D. Soft computing methods and systems

Soft computing techniques are suitable for network anomaly detection because often one cannot find exact solutions. Soft computing is usually thought of as encompassing methods such as Genetic Algorithms, Artificial Neural Networks, Fuzzy Sets, Rough Sets, Ant Colony Algorithms and Artificial Immune Systems. We describe several soft computing methods and systems for network anomaly detection below.

1) Genetic algorithm approaches: Genetic algorithms are population-based adaptive heuristic search techniques based on evolutionary ideas. The approach begins with conversion of a problem into a framework that uses a chromosome like data structure. Balajinath and Raghavan [127] present a genetic intrusion detector (GBID) based on learning of individual user behavior. User behavior is described as 3-tuple <matching index, entropy index, newness index> and is learnt using a genetic algorithm. This behavior profile is used to detect intrusion based on past behavior.

Khan [128] uses genetic algorithms to develop rules for network intrusion detection. A chromosome in an individual contains genes corresponding to attributes such as the service, flags, logged in or not, and super-user attempts. Khan concludes that attacks that are common can be detected more accurately compared to uncommon attributes.

2) Artificial Neural Network approaches: Artificial Neural Networks (ANN) are motivated by the recognition that the human brain computes in an entirely different way from the conventional digital computer [129]. The brain organizes its constituents, known as neurons, so as to perform certain computations (e.g., pattern recognition, perception, and motor control) many times faster than the fastest digital computer. To achieve good performance, real neural networks employ massive interconnections of neurons. Neural networks acquire knowledge of the environment through a process of learning, which systematically changes the interconnection strengths, or synaptic weights of the network to attain a desired design objective.

An example of ANN-based IDS is RT-UNNID [130]. This system is capable of intelligent real time intrusion detection using unsupervised neural networks (UNN). The architecture of RT-UNNID is given in Figure 11. The first module captures and preprocesses the real time network traffic data for the protocols: TCP, UDP and ICMP. It also extracts the numeric features and converts them into binary or normalized form. The converted data is sent to the UNN-based detection engine that uses Adaptive Resonance Theory (ART) and Self-Organizing Map (SOM) [131], [132] neural networks. Finally, the output of the detection engine is sent to the responder for recording in the user's system log file and to generate alarm when detecting attacks. RT-UNNID can work in real time to detect known and unknown attacks in network traffic with high detection rate.

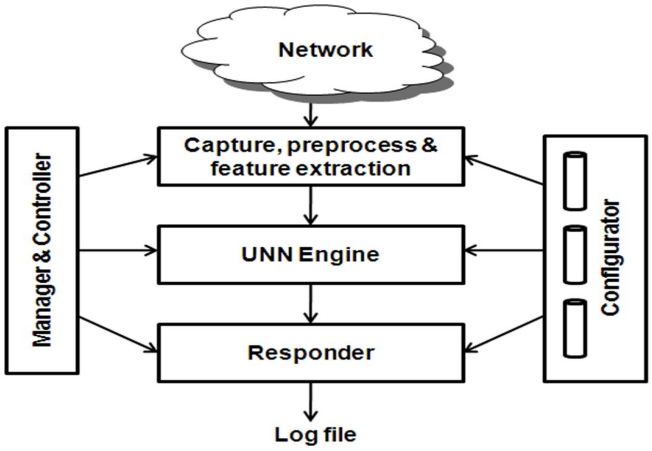

Fig. 11. Architecture of RT-UNNID system

Cannady's approach [133] autonomously learns new attacks rapidly using modified reinforcement learning. His approach uses feedback for signature update when a new attack is encountered and achieves satisfactory results. An improved approach to detect network anomalies using a hierarchy of neural networks is introduced in [134]. The neural networks are trained using data that spans the entire normal space and are able to recognize unknown attacks effectively.

Liu et al. [135] report a real time solution to detect known and new attacks in network traffic using unsupervised neural nets. It uses a hierarchical intrusion detection model using Principal Components Analysis (PCA) neural networks to overcome the shortcomings of single-level structures.

Sun et al. [136] present a wavelet neural network (WNN) based intrusion detection method. It reduces the number of the wavelet basic functions by analyzing the sparseness property of sample data to optimize the wavelet network to a large extent. The learning algorithm trains the network using gradient descent.

Yong and Feng [137] use recurrent multilayered perceptrons (RMLP) [138], a dynamic extension of well-known feed-forward layered networks to classify network data into anomalous and normal. An RMLP network has the ability to encode temporal information. They develop an incremental kernel principal components algorithm to pre-process the data that goes into the neural network and obtain effective results.

In addition to the detection methods, we discuss a few IDSs below.

NSOM (Network Self-Organizing Maps) [139] is a network IDS developed using Self-Organizing Maps (SOM). It detects anomalies by quantifying the usual or acceptable behavior and flags irregular behavior as potentially intrusive. To classify real

TABLE IX
COMPARISON OF CLUSTERING AND OUTLIER-BASED NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*No. of parameters*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Sequeira and Zaki [119]	2002	4	C	R	P	Numeric, Categorical	Real life	Synthetic intrusions	ADMIT
Zhang et al. [115]	2005	2	D	N	P	Numeric	KDDcup99	C_1	Cluster-based DIDS
Leung and Leckie [116]	2005	3	C	N	P	Numeric	KDDcup99	C_1	fpMAFIA algorithm
Otey et al. [123]	2006	5	C	N	P	Mixed	KDDcup99	C_1	FDOD algorithm
Jiang et al. [125]	2006	3	C	N	P	Mixed	KDDcup99	C_1	CBUID algorithm
Chen and Chen [126]	2008	-	O	N	-	-	-	C_3	AAWP model
Zhang et al. [117]	2009	2	O	N	P	Mixed	KDDcup99	C_1	KD algorithm
Zhuang et al. [121]	2010	2	R	C	P	-	Real time	C_6	PAIDS model
Bhuyan et al. [124]	2011	2	N	C	P,F	Numeric	KDDcup99	C_1	NADO algorithm
Casas et al. [118]	2012	2	N	C	F	Numeric	KDDcup99, Real time	C_1	UNIDS method

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, C_5-remote to local, and C_6-worms

time traffic, it uses a structured SOM. It continuously collects network data from a network port, preprocesses that data and selects the features necessary for classification. Then it starts the classification process - a chunk of packets at a time - and then sends the resulting classification to a graphical tool that portrays the activities that are taking place on the network port dynamically as it receives more packets. The hypothesis is that routine traffic that represents normal behavior would be clustered around one or more cluster centers and any irregular traffic representing abnormal and possibly suspicious behavior would be clustered in addition to the normal traffic clustering. The system is capable of classifying regular vs. irregular and possibly intrusive network traffic for a given host.

POSEIDON (Payl Over Som for Intrusion DetectiON) [140] is a two-tier network intrusion detection system. The first tier consists of a self-organizing map (SOM), and is used exclusively to classify payload data. The second tier consists of a light modification of the PAYL system [84]. Tests using the DARPA99 dataset show a higher detection rate and lower number of false positives than PAYL and PHAD [141].

3) Fuzzy set theoretic approaches: Fuzzy network intrusion detection systems exploit fuzzy rules to determine the likelihood of specific or general network attacks [142], [143]. A fuzzy input set can be defined for traffic in a specific network.

Tajbakhsh et al. [144] describe a novel method for building classifiers using fuzzy association rules and use it for network intrusion detection. The fuzzy association rule sets are used to describe different classes: normal and anomalous. Such fuzzy association rules are *class association rules* where the consequents are specified classes. Whether a training instance belongs to a specific class is determined by using matching metrics proposed by the authors. The fuzzy association rules are induced using normal training samples. A test sample is classified as normal if the compatibility of the rule set generated is above a certain threshold; those with lower compatibility are considered anomalous. The authors also propose a new method to speed up the rule induction algorithm by reducing items from extracted rules.

Mabu et al. report a novel fuzzy class-association-rule mining method based on genetic network programming (GNP) for detecting network intrusions [145]. GNP is an evolutionary optimization technique, which uses directed graph structures instead of strings in standard genetic algorithms leading to enhanced representation ability with compact descriptions derived from possible node reusability in a graph.

Xian et al. [146] present a novel unsupervised fuzzy clustering method based on clonal selection for anomaly detection. The method is able to obtain global optimal clusters more quickly than competing algorithms with greater accuracy.

In addition to the fuzzy set theoretic detection methods, we discuss two IDSs, viz., NFIDS and FIRE below.

NFIDS [147] is a neuro-fuzzy anomaly-based network intrusion detection system. It comprises three tiers. Tier-I contains several Intrusion Detection Agents (IDAs). IDAs are IDS components that monitor the activities of a host or a network and report the abnormal behavior to Tier-II. Tier-II agents detect the network status of a LAN based on the network traffic that they observe as well as the reports from the Tier-I agents within the LAN. Tier-III combines higher-level reports, correlates data, and sends alarms to the user interface. There are four main types of agents in this system: TCPAgent, which monitors TCP connections between hosts and on the network, UDPAgent, which looks for unusual traffic involving UDP data, ICMPAgent, which monitors ICMP traffic and PortAgent, which looks for unusual services in the network.

FIRE (Fuzzy Intrusion Recognition Engine) [142] is an anomaly-based intrusion detection system that uses fuzzy logic to assess whether malicious activity is taking place on a network. The system combines simple network traffic metrics with fuzzy rules to determine the likelihood of specific or general network attacks. Once the metrics are available, they are evaluated using a fuzzy set theoretic approach. The system takes on fuzzy network traffic profiles as inputs to its rule set and report maliciousness.

4) Rough Set approaches: A rough set is an approximation of a crisp set (i.e., a regular set) in terms of a pair of sets that are its lower and upper approximations. In the standard and original version of rough set theory [148], the two approximations are crisp sets, but in other variations the approximating sets may be fuzzy sets. The mathematical framework of rough set theory enables modeling of relationships with a minimum number of rules.

Rough sets have two useful features [149]: (i) enabling learning with small size training datasets (ii) and overall

simplicity. They can be applied to anomaly detection by modeling normal behavior in network traffic. For example, in [150], the authors present a Fuzzy Rough C-means clustering technique for network intrusion detection by integrating fuzzy set theory and rough set theory to achieve high detection rate.

Adetunmbi et al. [151] use rough sets and a k-NN classifier to detect network intrusions with high detection rate and low false alarm rate. Chen et al. present a two-step classifier for network intrusion detection [152]. Initially, it uses rough set theory for feature reduction and then a support vector machine classifier for final classification. They obtain 89% accuracy on network anomaly data.

5) Ant Colony and Artificial Immune System approaches: Ant colony optimization [153] and related algorithms are probabilistic techniques for solving computational problems which can be reformulated to find optimal paths through graphs. The algorithms are based on the behavior of ants seeking a path between their colony and a source of food.

Gao et al. [154] use ant colony optimization for feature selection for an SVM classifier for network intrusion detection. The features are represented as graph nodes with the edges between them denoting the addition of the next feature. Ants traverse the graph to add nodes until the stopping criterion is encountered.

Artificial Immune Systems (AIS) represent a computational method inspired by the principles of the human immune system. The human immune system is adept at performing anomaly detection. Visconti and Tahayori [155] present a performance-based AIS for detecting individual anomalous behavior. It monitors the system by analyzing the set of parameters to provide general information on its state. Interval type-2 fuzzy set paradigm is used to dynamically generate system status.

Advantages of soft computing-based anomaly detection methods include the following.

- Such learning systems detect or categorize persistent features without any feedback from the environment.
- Due to the adaptive nature of ANNs, it is possible to train and test instances incrementally using certain algorithms. Multi-level neural network-based techniques are more efficient than single level neural networks.
- Unsupervised learning using competitive neural networks is effective in data clustering, feature extraction and similarity detection.
- Rough sets are useful in resolving inconsistency in the dataset and to generate a minimal, non-redundant and consistent rule set.

Some of the disadvantages of soft computing methods are pointed out below.

- Over-fitting may happen during neural network training.
- If a credible amount of normal traffic data is not available, the training of the techniques becomes very difficult.
- Most methods have scalability problems.
- Rough set-based rule generation suffers from proof of completeness.
- In fuzzy association rule-based techniques, reduced, relevant rule subset identification and dynamic rule updation at runtime is a difficult task.

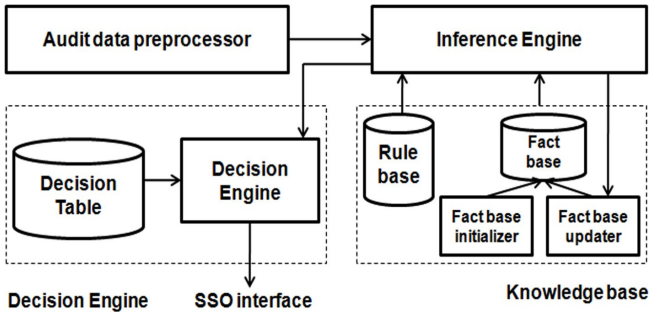

Fig. 12. Architecture of STAT system

Table X gives a comparison of several soft computing-based anomaly detection methods.

E. Knowledge-based methods and systems

In knowledge-based methods, network or host events are checked against predefined rules or patterns of attack. The goal is to represent the known attacks in a generalized fashion so that handling of actual occurrences becomes easier. Examples of knowledge-based methods are expert systems, rule-based, ontology-based, logic-based and state-transition analysis [156]–[159].

These techniques search for instances of known attacks, by attempting to match with pre-determined attack representations. The search begins like other intrusion detection techniques, with a complete lack of knowledge. Subsequent matching of activities against a known attack helps acquire knowledge and enter into a region with higher confidence. Finally, it can be shown that an event or activity has reached maximum anomaly score.

An example knowledge-based system is STAT (State Transition Analysis Tool) [160]. Its architecture is given in Figure 12. It models traffic data as a series of state changes that lead from secure state to a target compromised state. STAT is composed of three main components: knowledge base, inference engine and decision engine. The audit data preprocessor reformats the raw audit data to send as input to the inference engine. The inference engine monitors the state transitions extracted from the preprocessed audit data and then compares these states with the states available within the knowledge base. The decision engine monitors the improvement of the inference engine for matching accuracy of the state transitions. It also specifies the action(s) to be taken based on results of the inference engine and the decision table. Finally, the decision results are sent to the SSO (Site Security Officer) interface for action. STAT can detect cooperative attackers and attacks across user sessions well.

A few prominent knowledge-based network anomaly detection methods and NIDS are given below.

1) Rule-based and Expert system approaches: The expert system approach is one of the most widely used knowledge-based methods [161], [162]. An expert system, in the traditional sense, is a rule-based system, with or without an associated knowledge base. An expert system has a rule engine that matches rules against the current state of the system, and depending on the results of matching, fires one or more rules.

TABLE X
COMPARISON OF SOFT COMPUTING-BASED NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*No. of parameters*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Cannady [133]	2000	2	O	N	P	Numeric	Real-life	C_2	CMAC-based model
Balajinath and Raghavan [127]	2001	3	O	N	O	Categorical	User command	C_4	Behavior Model
Lee and Heinbuch [134]	2001	3	C	N	P	-	Simulated data	C_2	TNNID model
Xian et al. [146]	2005	3	C	N	P	Numeric	KDDcup99	C_1	Fuzzy k-means
Amini et al. [130]	2006	2	C	R	P	Categorical	KDDcup99, Real-life	C_1	RT-UNNID system
Chimphlee et al. [150]	2006	3	C	N	P	Numeric	KDDcup99	C_1	Fuzzy Rough C-means
Liu et al. [135]	2007	2	C	N	P	Numeric	KDDcup99	C_1	HPCANN Model
Adetunmbi et al. [151]	2008	2	C	N	P	Numeric	KDDcup99	C_1	LEM2 and K-NN
Chen et al. [152]	2009	3	C	N	P	Numeric	DARPA98	C_2	RST-SVM technique
Mabu et al. [145]	2011	3	C	N	P	Numeric	KDDcup99	C_1	Fuzzy ARM-based on GNP
Visconti and Tahayori [155]	2011	2	O	N	P	Numeric	Real-life	C_2	Interval type-2 fuzzy set
Geramiraz et al. [143]	2012	2	O	N	P	Numeric	KDDcup99	C_1	Fuzzy rule-based model

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, and C_5-remote to local

Snort [113] is a quintessentially popular rule-based IDS. This open-source IDS matches each packet it observes against a set of rules. The antecedent of a Snort rule is a boolean formula composed of predicates that look for specific values of fields present in IP headers, transport headers and in the payload. Thus, Snort rules identify attack packets based on IP addresses, TCP or UDP port numbers, ICMP codes or types, and contents of strings in the packet payload. Snort's rules are arranged into priority classes based on potential impact of alerts that match the rules. Snort's rules have evolved over its history of 15 years. Each Snort rule has associated documentation with the potential for false positives and negatives, together with corrective actions to be taken when the rule raises an alert. Snort rules are simple and easily understandable. Users can contribute rules when they observe new types of anomalous or malicious traffic. Currently, Snort has over $20,000$ rules, inclusive of those submitted by users.

An intrusion detection system like Snort can run on a general purpose computer and can try to inspect all packets that go through the network. However, monitoring packets comprehensively in a large network is obviously an expensive task since it requires fast inspection on a large number of network interfaces. Many hundreds of rules may have to be matched concurrently, making scaling almost impossible.

To scale to large networks that collect flow statistics ubiquitously, Duffield et al. [163] use the machine learning algorithm called Adaboost [164] to translate packet level signatures to work with flow level statistics. The algorithm is used to correlate the packet and flow information. In particular, the authors associate packet level network alarms with a feature vector they create from flow records on the same traffic. They create a set of rules using flow information with features similar to those used in Snort rules. They also add numerical features such as the number of packets of a specific kind flowing within a certain time period. Duffield et al. train Adaboost on concurrent flow and packet traces. They evaluate the system using real time network traffic data with more than a billion flows over 29 days, and show that their performance is comparable to Snort's with flow data.

Prayote and Compton [165] present an approach to anomaly detection that attempts to address the brittleness problem in which an expert system makes a decision that human common sense would recognize as impossible. They use a technique called *prudence* [166], in which for every rule, the upper and lower bounds of each numerical variable in the data seen by the rule are recorded, as well as a list of values seen for enumerated variables. The expert system raises a warning when a new value or a value outside the range is seen in a data instance. They improve the approach by using a simple probabilistic technique to decide if a value is an outlier. When working with network anomaly data, the authors partition the problem space into smaller subspaces of homogeneous traffic, each of which is represented with a separate model in terms of rules. The authors find that this approach works reasonably well for new subspaces when little data has been observed. They claim 0% false negative rate in addition to very low false positive rate.

Scheirer and Chuah [167] report a syntax-based scheme that uses variable-length partition with multiple break marks to detect many polymorphic worms. The prototype is the first NIDS that provides semantics-aware capability, and can capture polymorphic shell codes with additional stack sequences and mathematical operations.

2) Ontology and logic-based approaches: It is possible to model attack signatures using expressive logic structure in real time by incorporating constraints and statistical properties. Naldurg et al. [168] present a framework for intrusion detection based on temporal logic specification. Intrusion patterns are specified as formulae in an expressively rich and efficiently monitorable logic called EAGLE and evaluated using DARPA log files.

Estevez-Tapiador et al. [169] describe a finite state machine (FSM) methodology, where a sequence of states and transitions among them seems appropriate to model network protocols. If the specifications are complete enough, the model is able to detect illegitimate behavioral patterns effectively.

Shabtai et al. [170] describe an approach for detecting previously un-encountered malware targeting mobile devices. Time-stamped security data is continuously monitored within the target mobile devices like smart phones and PDAs. Then it is processed by the knowledge-based temporal abstraction (KBTA) methodology. The authors evaluate the KBTA model

by using a lightweight host-based intrusion detection system, combined with central management capabilities for Android-based mobile phones.

Hung and Liu [171] use ontologies as a way of describing the knowledge of a domain, expressing the intrusion detection system in terms of the end user domain. Ontologies are used as a conceptual modeling tool allowing a non-expert person to model intrusion detection applications using the concepts of intrusion detection more intuitively.

A comparison of knowledge-based anomaly detection methods is given in Table XI.

The main advantages of knowledge-based anomaly detection methods include the following.

- These techniques are robust and flexible.
- These techniques have high detection rate, if a substantial knowledge base can be acquired properly about attacks as well as normal instances.

Some disadvantages of knowledge-based methods are listed below.

- The development of high-quality knowledge is often difficult and time-consuming.
- Due to non-availability of biased normal and attack data, such a method may generate a large number of false alarms.
- Such a method may not be able to detect rare or unknown attacks.
- Dynamic updation of rule or knowledge base is a costly affair.

F. Combination learner methods and systems

In this section, we present a few methods and systems which use combinations of multiple techniques, usually classifiers.

1) Ensemble-based methods and systems: The idea behind the ensemble methodology is to weigh several individual classifiers, and combine them to obtain an overall classifier that outperforms every one of them [172]–[176]. These techniques weigh the individual opinions, and combine them to reach a final decision. The ensemble-based methods are categorized based on the approaches used. Three main approaches to develop ensembles are (i) bagging, (ii) boosting, and (iii) stack generalization. *Bagging* (Bootstrap Aggregating) increases classification accuracy by creating an improved composite classifier into a single prediction by combining the outputs of learnt classifiers. *Boosting* builds an ensemble incrementally by training mis-classified instances obtained from the previous model. *Stack generalization* achieves the high generalization accuracy by using output probabilities for every class label from the base-level classifiers.

Octopus-IIDS [177] is an example of ensemble IDS. The architecture of this system is shown in Figure 13. It is developed using two types of neural networks, Kohonen and Support Vector Machines. The system is composed of two layers: classifier and anomaly detection. The classifier is responsible for capturing and preprocessing of network traffic data. It classifies the data into four main categories: DoS, probe, U2R and R2L. A specific class of attack is identified in the anomaly detection layer. The authors claim that the IDS works effectively in small scale networks.

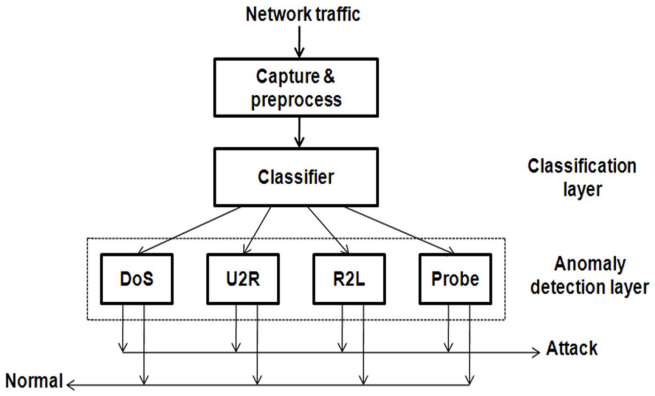

Fig. 13. Architecture of Octopus-IIDS system

Chebrolu et al. [178] present an ensemble approach by combining two classifiers, Bayesian networks (BN) and Classification and Regression Trees (CART) [90], [179]. A hybrid architecture for combining different feature selection algorithms for real world intrusion detection is also incorporated for getting better results. Perdisci et al. [180] construct a high speed payload anomaly IDS using an ensemble of one-class SVM classifiers intended to be accurate and hard to evade.

Folino et al. [181] introduce a distributed data mining algorithm to improve detection accuracy when classifying malicious or unauthorized network activity using genetic programming (GP) extended with the ensemble paradigm. Their data is distributed across multiple autonomous sites and the learner component acquires useful knowledge from data in a cooperative way and uses network profiles to predict abnormal behavior with better accuracy.

Nguyen et al. [58] build an individual classifier using both the input feature space and an additional subset of features given by k-means clustering. The ensemble combination is calculated based on the classification ability of classifiers on different local data segments given by k-means clustering.

Beyond the above methods, some ensemble-based IDSs are given below.

The paradigm of multiple classifier system (MCS) has also been used to build misuse detection IDSs. Classifiers trained on different feature subsets can be combined to achieve better classification accuracy than the individual classifiers. In such a NIDS, network traffic is serially processed by each classifier. At each stage, a classifier may either decide for one attack class or send the pattern to another stage, which is trained on more difficult cases. Reported results show that an MCS improves the performance of IDSs based on statistical pattern recognition techniques. For example, CAMNEP [182] is a fast prototype agent-based NIDS designed for high-speed networks. It integrates several anomaly detection techniques, and operates on a collective trust model within a group of collaborative detection agents. The anomalies are used as input for trust modeling. Aggregation is performed by extended trust models of generalized situated identities, represented by a set of observable features. The system is able to perform real time surveillance of gigabit networks.

McPAD (Multiple classifier Payload-based Anomaly Detection) [183] is an effective payload-based anomaly detection

TABLE XI
COMPARISON OF KNOWLEDGE-BASED NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*No. of parameters*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Noel et al. [156]	2002	-	O	N	O	-	-	-	Attack Guilt Model
Sekar et al. [157]	2002	3	O	N	P	Numeric	DARPA99	C_1	Specification-Based Model
Tapiador et al. [169]	2003	3	C	N	P	Numeric	Real-life	C_2	Markov Chain Model
Hung and Liu [171]	2008	-	O	N	P	Numeric	KDDcup99	C_1	Ontology-based
Shabtai et al. [170]	2010	2	O	N	O	-	Real-life	C_2	Incremental KBTA

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, and C_5-remote to local

TABLE XII
COMPARISON OF ENSEMBLE-BASED NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*Combination strategy*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Chebrolu et al. [178]	2005	Weighted voting	O	N	P	Numeric	KDDcup99	C_1	Class specific ensemble model
Perdisci et al. [180]	2006	Majority voting	O	N	Pay	-	Operational points	Synthetic intrusions	One-class classifier model
Borji [173]	2007	Majority voting	O	N	P	Numeric	DARPA98	C_1	Heterogeneous classifiers model
Perdisci et al. [183]	2009	Min and Max probability	O	R	Pay	-	DARPA98	C_1	McPAD model
Folino et al. [181]	2010	Weighted majority voting	O	N	P	Numeric	KDDcup99	C_1	GEdIDS model
Noto et al. [176]	2010	Information theoretic	O	N	-	Numeric	UCI	None	FRaC model
Nguyen et al. [58]	2011	Majority voting	O	N	P	Numeric	KDDcup99	C_1	Cluster ensemble
Khreich et al. [184]	2012	Learn and combine	O	N	pay	Numeric	UNM	C_4	EoHMMs model

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or payload-based (pay) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, and C_5-remote to local

system that consists of an ensemble of one-class classifiers. It is very accurate in detecting network attacks that bear some form of shell-code in the malicious payload. This detector performs well even in the case of polymorphic attacks. Furthermore, the authors tested their IDS with advanced polymorphic blending attacks and showed that even in the presence of such sophisticated attacks, it is able to obtain a low false positive rate.

An ensemble method is advantageous because it obtains higher accuracy than the individual techniques. The following are the major advantages.

- Even if the individual classifiers are weak, the ensemble methods perform well by combining multiple classifiers.
- Ensemble methods can scale for large datasets.
- Ensemble classifiers need a set of controlling parameters that are comprehensive and can be easily tuned.
- Among existing approaches, Adaboost and Stack generalization are more effective because they can exploit the diversity in predictions by multiple base level classifiers.

Here are some disadvantages of ensemble-based methods.

- Selecting a subset of consistent performing and unbiased classifiers from a pool of classifiers is difficult.
- The greedy approach for selecting sample datasets is slow for large datasets.
- It is difficult to obtain real time performance.

A comparison of ensemble-based network anomaly detection methods is given in Table XII.

2) Fusion-based methods and system: With an evolving need of automated decision making, it is important to improve classification accuracy compared to the stand-alone general decision-based techniques even though such a system may have several disparate data sources. So, a suitable combination of these is the focus of the fusion approach. Several fusion-based techniques have been applied to network anomaly detection [185]–[189]. A classification of such techniques is as follows: (i) data level, (ii) feature level, and (iii) decision level. Some methods only address the issue of operating in a space of high dimensionality with features divided into semantic groups. Others attempt to combine classifiers trained on different features divided based on hierarchical abstraction levels or the type of information contained.

Giacinto et al. [185] provide a pattern recognition approach to network intrusion detection employing a fusion of multiple classifiers. Five different decision fusion methods are assessed by experiments and their performances compared. Shifflet [186] discusses a platform that enables a multitude of techniques to work together towards creating a more realistic fusion model of the state of a network, able to detect malicious activity effectively. A heterogenous data level fusion for network anomaly detection is added by Chatzigiannakis et al. [190]. They use the Dempster-Shafer Theory of Evidence and Principal Components Analysis for developing the technique.

dLEARNIN [187] is an ensemble of classifiers that combines information from multiple sources. It is explicitly tuned to minimize the cost of errors. dLEARNIN is shown to achieve state-of-the-art performance, better than competing algorithms. The cost minimization strategy, dCMS, attempts to minimize the cost to a significant level. Gong et al. [191] contribute a neural network-based data fusion method for intrusion data

analysis and pruning to filter information from multi-sensors to get high detection accuracy.

HMMPayl [192] is an example of fusion-based IDS, where the payload is represented as a sequence of bytes, and the analysis is performed using Hidden Markov Models (HMM). The algorithm extracts features and uses HMM to guarantee the same expressive power as that of n-gram analysis, while overcoming its computational complexity. HMMPayl follows the Multiple Classifiers System paradigm to provide better classification accuracy, to increase the difficulty of evading the IDS, and to mitigate the weaknesses due to a non-optimal choice of HMM parameters.

Some advantages of fusion methods are given below.

- Data fusion is effective in increasing timeliness of attack identification and in reducing false alarm rates.
- Decision level fusion with appropriate training data usually yields high detection rate.

Some of the drawbacks are given below.

- The computational cost is high for rigorous training on the samples.
- Feature level fusion is a time consuming task. Also, the biases of the base classifiers affect the fusion process.
- Building hypotheses for different classifiers is a difficult task.

A comparison of fusion-based network anomaly detection methods is given in Table XIII.

3) Hybrid methods and system: Most current network intrusion detection systems employ either misuse detection or anomaly detection. However, misuse detection cannot detect unknown intrusions, and anomaly detection usually has high false positive rate [193]. To overcome the limitations of the techniques, hybrid methods are developed by exploiting features from several network anomaly detection approaches [194]–[196]. Hybridization of several methods increases performance of IDSs.

For example, RT-MOVICAB-IDS, a hybrid intelligent IDS is introduced in [197]. It combines ANN and CBR (case-based reasoning) within a Multi-Agent System (MAS) to detect intrusion in dynamic computer networks. The dynamic real time multi-agent architecture allows the addition of prediction agents (both reactive and deliberative). In particular, two of the deliberative agents deployed in the system incorporate temporal-bounded CBR. This upgraded CBR is based on an anytime approximation, which allows the adaptation of this paradigm to real time requirements.

A hybrid approach to host security that prevents binary code injection attacks known as the FLIPS (Feedback Learning IPS) model is proposed by [198]. It incorporates three major components: an anomaly-based classifier, a signature-based filtering scheme, and a supervision framework that employs Instruction Set Randomization (ISR). Capturing the injected code allows FLIPS to construct signatures for zero-day exploits. Peddabachigari et al. [199] present a hybrid approach that combines Decision trees (DT) and SVMs as a hierarchical hybrid intelligent system model (DTSVM) for intrusion detection. It maximizes detection accuracy and minimizes computational complexity.

Zhang et al. [200] propose a systematic framework that applies a data mining algorithm called random forests in building a misuse, anomaly, and hybrid network-based IDS. The hybrid detection system improves detection performance by combining the advantages of both misuse and anomaly detection. Tong et al. [201] discuss a hybrid RBF/Elman neural network model that can be employed for both anomaly detection and misuse detection. It can detect temporally dispersed and collaborative attacks effectively because of its memory of past events.

A intelligent hybrid IDS model based on neural networks is introduced by [202]. The model is flexible, extended to meet different network environments, improves detection performance and accuracy. Selim et al. [203] report a hybrid intelligent IDS to improve the detection rate for known and unknown attacks. It consists of multiple levels: hybrid neural networks and decision trees. The technique is evaluated using NSL-KDD dataset and results were promising.

Advantages of hybrid methods include the following.

- Such a method exploits major features from both signature and anomaly-based network anomaly detection.
- Such methods can handle both known and unknown attacks.

Drawbacks include the following.

- Lack of appropriate hybridization may lead to high computational cost.
- Dynamic updation of rule or profile or signature still remains difficult.

Table XIV presents a comparison of a few hybrid network anomaly detection methods.

G. Discussion

After a long and elaborate discussion of many intrusion detection methods and anomaly-based network intrusion detection systems under several categories, we make a few observations.

(i) Each class of anomaly-based network intrusion detection methods and systems has unique strengths and weaknesses. The suitability of an anomaly detection technique depends on the nature of the problem attempted to address. Hence, providing a single integrated solution to every anomaly detection problem may not be feasible.

(ii) Various methods face various challenges when complex datasets are used. Nearest neighbor and clustering techniques suffer when the number of dimensions is high because the distance measures in high dimensions are not able to differentiate well between normal and anomalous instances.

Spectral techniques explicitly address the high dimensionality problem by mapping data to a lower dimensional projection. But their performance is highly dependent on the assumption that normal instances and anomalies are distinguishable in the projected space. A classification technique often performs better in such a scenario. However, it requires labeled training data for both normal and attack classes. The improper distribution of these training data often makes the task of learning more challenging.

TABLE XIII
COMPARISON OF FUSION-BASED NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*Fusion level*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Giacinto et al. [185]	2003	Decision	O	N	P	Numeric	KDDcup99	C_1	MCS Model
Shifflet [186]	2005	Data	O	N	O	-	-	None	HSPT algorithm
Chatzigiannakis et al. [190]	2007	Data	C	N	P	-	NTUA, GRNET	C_2	D-S algorithm
Parikh and Chen [187]	2008	Data	C	N	P	Numeric	KDDcup99	C_1	dLEARNIN system
Gong et al. [191]	2010	Data	C	N	P	Numeric	KDDcup99	C_1	IDEA model
Ariu et al. [192]	2011	Decision	C	R	Pay	-	DARPA98, real-life	C_1	HMMPayl model
Yan and Shao [189]	2012	Decision	O	N	F	Numeric	Real time	C_2, C_3	EWMA model

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or payload-based (pay) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, and C_5-remote to local

TABLE XIV
COMPARISON OF HYBRID NETWORK ANOMALY DETECTION METHODS

Author (s)	*Year of publication*	*No. of parameters*	*w*	*x*	*y*	*Data types*	*Dataset used*	*z*	*Detection method*
Locasto et al. [198]	2005	2	C	R	P	-	Real-life	C_2	FLIPS model
Zhang and Zulkernine [194]	2006	2	C	N	P	Numeric	KDDcup99	C_1	Random forest-based hybrid algorithm
Peddabachigari et al. [199]	2007	2	C	N	P	Numeric	KDDcup99	C_1	DT-SVM hybrid model
Zhang et al. [200]	2008	2	C	N	P	Numeric	KDDcup99	C_1	RFIDS model
Aydin et al. [195]	2009	3	C	N	P	-	DARPA98, IDEVAL	C_1	Hybrid signature-based IDS
Tong et al. [201]	2009	1	C	N	P	Numeric	DARPA-BSM	C_1	Hybrid RBF/Elman NN
Yu [202]	2010	1	C	N	-	-	-	-	Hybrid NIDS
Arumugam et al. [193]	2010	-	C	N	P	Numeric	KDDcup99	C_1	Multi-stage hybrid IDS
Selim et al. [203]	2011	-	C	N	P	Numeric	KDDcup99	C_1	Hybrid multi-level IDS
Panda et al. [196]	2012	2	C	N	P	Numeric	NSL-KDD, KDDcup99	C_1	DTFF and FFNN

w-indicates centralized (C) or distributed (D) or others (O)
x-the nature of detection as real time (R) or non-real time (N)
y-characterizes packet-based (P) or flow-based (F) or hybrid (H) or others (O)
z-represents the list of attacks handled: C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, and C_5-remote to local

Semi-supervised nearest neighbor and clustering techniques that only use normal labels, can often be more effective than classification-based techniques. In situations where identifying a good distance measure is difficult, classification or statistical techniques may be a better choice. However, the success of the statistical techniques is largely influenced by the applicability of the statistical assumptions in the specific real life scenarios.

(iii) For real time intrusion detection, the complexity of the anomaly detection process plays a vital role. In case of classification, clustering, and statistical methods, although training is expensive, they are still acceptable because testing is fast and training is offline. In contrast, techniques such as nearest neighbor and spectral techniques which do not have a training phase, have an expensive testing phase which can be a limitation in a real setting.

(iv) Anomaly detection techniques typically assume that anomalies in data are rare when compared to normal instances. Generally, such assumptions are valid, but not always. Often unsupervised techniques suffer from large false alarm rates, when anomalies are in bulk amounts. Techniques operating in supervised or semi-supervised modes [204] can be applied to detect bulk anomalies.

We perform a comparison of the anomaly-based network intrusion detection systems that we have discussed throughout this paper based on parameters such as mode of detection (host-based, network-based or both), detection approach (misuse, anomaly or both), nature of detection (online or offline), nature of processing (centralized or distributed), data gathering mechanism (centralized or distributed) and approach of analysis. A comparison chart is given in Table XV.

V. EVALUATION CRITERIA

To evaluate performance, it is important that the system identifies the attack and normal data correctly. There are several datasets and evaluation measures available for evaluating network anomaly detection methods and systems. The most commonly used datasets and evaluation measures are given below.

A. Datasets

Capturing and preprocessing high speed network traffic is essential prior to detection of network anomalies. Different tools are used for capture and analysis of network traffic data. We list a few commonly used tools and their features in Table XVI. These are commonly used by both the network defender and the attacker at different time points.

The following are various datasets that have been used for evaluating network anomaly detection methods and systems. A taxonomy of different datasets is given in Figure 14.

1) Synthetic datasets: Synthetic datasets are generated to meet specific needs or conditions or tests that real data satisfy. This can be useful when designing any type of system for theoretical analysis so that the design can be refined. This allows for finding a basic solution or remedy, if the results

TABLE XV
COMPARISON OF EXISTING NIDSS

Name of IDS	*Year of publication*	*a*	*b*	*c*	*d*	*e*	*Approach*
STAT [160]	1995	H	M	R	C	C	Knowledge-based
FIRE [142]	2000	N	A	N	C	C	Fuzzy Logic
ADAM [32]	2001	N	A	R	C	C	Classification
HIDE [33]	2001	N	A	R	C	D	Statistical
NSOM [139]	2002	N	A	R	C	C	Neural network
MINDS [34]	2003	N	A	R	C	C	Clustering and Outlier-based
NFIDS [147]	2003	N	A	N	C	C	Neuro Fuzzy Logic
N@G [93]	2003	H	B	R	C	C	Statistical
FSAS [94]	2006	N	A	R	C	C	Statistical
POSEIDON [140]	2006	N	A	R	C	C	SOM & Modified PAYL
RT-UNNID [130]	2006	N	A	R	C	C	Neural Network
DNIDS [110]	2007	N	A	R	C	C	CSI-KNN based
CAMNEP [182]	2008	N	A	R	C	C	Agent-based Trust and Reputation
McPAD [183]	2009	N	A	N	C	C	Multiple classifier
Octopus-IIDS [177]	2010	N	A	N	C	C	Neural network & SVM
HMMPayl [192]	2011	N	A	R	C	C	HMM model
RT-MOVICAB-IDS [197]	2011	N	A	R	C	C	Hybrid IDS

a-represents the types of detection such as host-based (H) or network-based (N) or hybrid (H)
b-indicates the class of detection mechanism as misuse (M) or anomaly (A) or both (B)
c-denotes the nature of detection as real time (R) or non-real time (N)
d-characterizes the nature of processing as centralized (C) or distributed (D)
e-indicates the data gathering mechanism as centralized (C) or distributed (D)

TABLE XVI
TOOLS USED IN DIFFERENT STEPS IN NETWORK TRAFFIC ANOMALY DETECTION AND THEIR DESCRIPTION

Tool Name	*Purpose*	*Characteristics*	*Source*
Wireshark	Packet capture	(i) Free and open-source packet analyzer. (ii) Can be used for network troubleshooting, analysis, software and communications protocol development, and education. (iii) Uses cross-platform GTK+ widget toolkit to implement its user interface, and uses pcap to capture packets. (iv) Similar to tcpdump, but has a graphical front-end, plus some integrated sorting and filtering options. (v) Works in mirrored ports to capture network traffic to analyze for any tampering.	http://www.wireshark.org/
Gulp	Lossless gigabit remote packet capturing	(i) It allows much higher packet capture rate by dropping far fewer packets. (ii) It has ability to read directly from the network, but is able to even pipe output from legacy applications before writing to disk. (iii) If the data rate increases, Gulp realigns its writes to even block boundaries for optimum writing efficiency. (iv) When it receives an interrupt, it stops filling its ring buffer but does not exit until it has finished writing whatever remains in the ring buffer.	http://staff.washington.edu/ corey/gulp/
tcptrace	TCP-based feature extraction	(i) Can take input files produced by several popular packet-capture programs, including tcpdump, snoop, etherpeek, HP Net Metrix, Wireshark, and WinDump. (ii) Produces several types of output containing information on each connection seen, such as elapsed time, bytes and segments sent and received, retransmissions, round trip times, window advertisements, and throughput. (iii) Can also produce a number of graphs with packet statistics for further analysis.	http://jarok.cs.ohiou.edu/ software/tcptrace/
nfdump	netflow data collection	(i) Can collect and process netflow data on the command line. (ii) It is limited only by the disk space available for all the netflow data. (iii) Can be optimized in speed for efficient filtering. The filter rules look like the syntax of tcpdump.	http://nfdump.sourceforge. net/
nfsen	netflow data collection and visualization	(i) NfSen is a graphical Web-based front end for the nfdump netflow tool. (ii) It allows display of netflow data as flows, packets and bytes using RRD (Round Robin Database). (iii) Can process the netflow data within a specified time span. (iv) Can create history as well as continuous profiles. (v) Can set alerts, based on various conditions.	http://nfsen.sourceforge.net/
nmap	Scanning port	(i) Nmap (Network Mapper) is a free and open source utility for network exploration or security auditing. (ii) Uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts offer, what operating systems are running, type of firewall or packet filter used, and many other characteristics. (iii) It is easy, flexible, powerful, well documented tool for discovering hosts in large network.	http://nmap.org/
rnmap	Coordinated scanning	(i) Remote Nmap (Rnmap) contains both client and server programs. (ii) Various clients can connect to one centralized Rnmap server and do their port scanning. (iii) Server performs user authentication and uses excellent Nmap scanner to do actual scanning.	http://rnmap.sourceforge. net/
Targa	Attack simulation	(i) Targa is free and powerful attack generation tool. (ii) It integrates bonk, jolt, land, nestea, netear, syndrop, teardrop, and winnuke into one multi-platform DoS attack.	http://www10.org/cdrom/ papers/409/

prove to be satisfactory. Synthetic data is used in testing and creating many different types of test scenarios. It enables designers to build realistic behavior profiles for normal users and attackers based on the generated dataset to test a proposed system.

2) Benchmark datasets: In this subsection, we present six publicly available benchmark datasets generated using simulated environments that include a number of networks and by executing different attack scenarios.

(a) *KDDcup99 dataset:* Since 1999, the KDDcup99 dataset [205] has been the most widely used dataset for the evaluation of network-based anomaly detection methods and systems. This dataset was prepared by Stolfo et al. [206] and is built on the data captured in the DARPA98 IDS evaluation program. The KDD training dataset consists of approximately $4,900,000$ single connection vectors, each of which contains 41 features and is labeled as either normal or attack with a specific attack type. The test dataset contains about $300,000$ samples with 24 training attack types, with an additional 14 attack types in the test set only. The names and descriptions of the attack types are available in [205].

(b) *NSL-KDD dataset:* Analysis of the KDD dataset showed that there were two important issues in the dataset, which highly affect the performance of evaluated systems result-

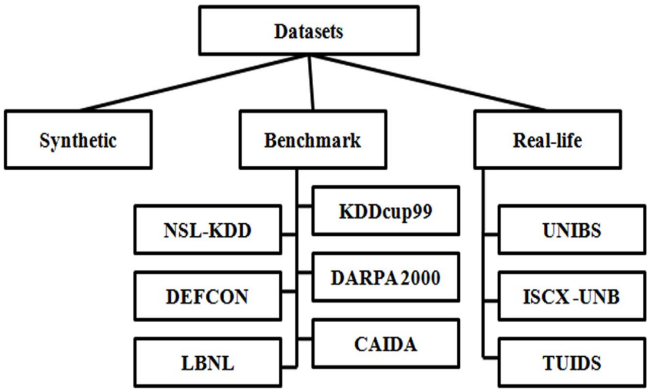

Fig. 14. Taxonomy of different datasets

ing in poor evaluation of anomaly detection methods [207]. To solve these issues, a new dataset known as NSL-KDD [208], consisting of selected records of the complete KDD dataset was introduced. This dataset is publicly available for researchers[5] and has the following advantages over the original KDD dataset.

- It does not include redundant records in the training set, so that the classifiers will not be biased towards more frequent records.
- There are no duplicate records in the test set. Therefore, the performance of the learners is not biased by the methods which have better detection rates on the frequent records.
- The number of selected records from each difficulty level is inversely proportional to the percentage of records in the original KDD dataset. As a result, the classification rates of various machine learning methods vary in a wide range, which makes it more efficient to have an accurate evaluation of various learning techniques.
- The number of records in the training and testing sets are reasonable, which makes it affordable to run experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research groups are consistent and comparable.

(c) *DARPA 2000 dataset:* A DARPA[6] evaluation project [209] targeted the detection of complex attacks that contain multiple steps. Two attack scenarios were simulated in the 2000 evaluation contest, namely, LLDOS (Lincoln Laboratory scenario DDoS) 1.0 and LLDOS 2.0. To achieve the necessary variations, these two attack scenarios were carried out over several network and audit scenarios. These sessions were grouped into four attack phases: (a) probing, (b) breaking into the system by exploiting vulnerability, (c) installing DDoS software for the compromised system and (d) launching DDoS attack against another target. LLDOS 2.0 is different from LLDOS 1.0 in the sense that attacks are more stealthy and thus harder to detect. Since this dataset contains multi-stage attack scenarios, it is also commonly used for evaluation of alert correlation methods.

(d) *DEFCON dataset:* The DEFCON[7] dataset is another commonly used dataset for evaluation of IDSs [210]. It contains network traffic captured during the hacker competition called Capture The Flag (CTF), in which competing college teams are divided into two groups: *attackers* and *defenders*. The traffic produced during CTF is very different from real world network traffic since it contains only intrusive traffic without any normal background traffic. Due to this reason, the DEFCON dataset has been found useful in evaluating alert correlation techniques.

(e) *CAIDA dataset:* CAIDA[8] collects many different types of data and makes it available to the research community. Most CAIDA datasets [211] are very specific to particular events or attacks (e.g., CAIDA DDoS attack 2007 dataset). All backbone traces are anonymized and do not have payload information.

(f) *LBNL dataset:* LBNL's (Lawrence Berkeley National Laboratory) internal enterprise traces are full header network traces [212], without payload. This dataset has undergone heavy anonymization to the extent that scanning traffic was extracted and separately anonymized to remove any information which could identify individual IPs. The packet traces were obtained at the two central routers of the LBNL network and they contain more than one hundred hours of traffic generated from several thousand internal hosts.

3) Real life datasets: In this subsection, we present three real life datasets created by collecting network traffic on several days, which include both normal as well as attack instances in appropriate proportions in the authors' respective campus networks.

(a) *UNIBS dataset:* The UNIBS packet traces [213] were collected on the edge router of the campus network of the University of Brescia, Italy, on three consecutive working days. This dataset includes traffic captured or collected and stored through 20 workstations running the GT client daemon. The authors collected the traffic by running tcpdump on the faculty router, which was a dual Xeon Linux box that connected their network to the Internet through a dedicated 100Mb/s uplink. The traces were captured and stored on a dedicated disk of a workstation connected to the router through a dedicated ATA controller.

(b) *ISCX-UNB dataset:* Real packet traces [214] were analyzed to create profiles for agents that generate real traffic for HTTP, SMTP, SSH, IMAP, POP3 and FTP protocols. Various multi-stage attack scenarios were explored for generating malicious traffic.

(c) *TUIDS dataset:* The TUIDS[9] dataset [215], [216] has been prepared at the Network Security Lab at Tezpur University, India based on several attack scenarios. Initially, the creators capture network traffic in both packet and flow level using gulp [217] and nfdump [218], then preprocess the raw traffic and label each traffic as attack or normal. The authors extract features such as basic, content, time, window and connectionless features from the preprocessed data, then correlate the features and generate the final datasets.

These datasets are valuable assets for the intrusion detection community. However, the benchmark datasets suffer from the

[5]http://www.iscx.ca/NSL-KDD/

[6]http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html

[7]http://cctf.shmoo.com/data/

[8]http://www.caida.org/home/

[9]http://agnigarh.tezu.ernet.in/∼dkb/resources/

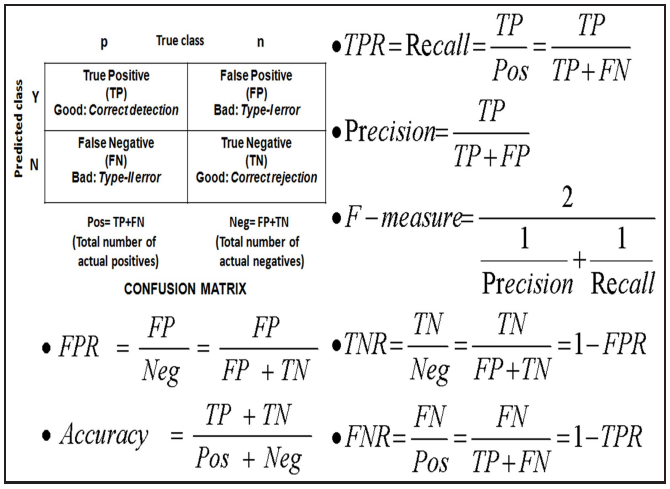

Fig. 15. Confusion matrix and related evaluation measures

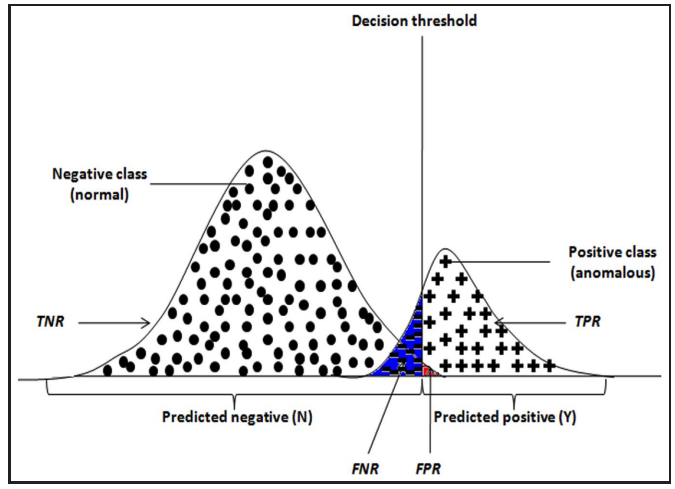

Fig. 16. Illustration of confusion matrix in terms of related evaluation measures

fact that they are not good representatives of real world traffic. For example, the DARPA dataset has been questioned about the realism of the background traffic [219], [220] because it is synthetically generated. In addition to the difficulty of simulating real time network traffic, there are some other challenges in IDS evaluation [221]. A comparison of datasets is shown in Table XVII.

B. Evaluation Measures

An evaluation of a method or a system in terms of accuracy or quality is a snapshot in time. As time passes, new vulnerabilities may evolve, and current evaluations may become irrelevant. In this section, we discuss various measures used to evaluate network intrusion detection methods and systems.

1) Accuracy: Accuracy is a metric that measures how correctly an IDS works, measuring the percentage of detection and failure as well as the number of false alarms that the system produces [223], [224]. If a system has 80% accuracy, it means that it correctly classifies 80 instances out of 100 to their actual classes. While there is a big diversity of attacks in intrusion detection, the main focus is that the system be able to detect an attack correctly. From real life experience, one can easily conclude that the actual percentage of abnormal data is much smaller than that of the normal [57], [225], [226]. Consequently, intrusions are harder to detect than normal traffic, resulting in excessive false alarms as the biggest problem facing IDSs. The following are the some accuracy measures.

(a) *Sensitivity and Specificity:* These two measures [227] attempt to measure the accuracy of classification for a 2-class problem. When an IDS classifies data, its decision can be either right or wrong. It assumes true for right and false for wrong, respectively.

If S is a detector and D_t is the set of test instances, there are four possible outcomes described using the confusion matrix given in Figure 15. When an anomalous test instance (p) is predicted as anomalous (Y) by the detector S, it is counted as true positive (TP); if it is predicted as normal (N), it is counted as false negative (FN). On the other hand, if a normal (n) test instance is predicted as normal (N) it is known as true negative (TN), while it is a false positive (FP) if it is predicted as anomalous (Y) [40], [227], [228].

The true positive rate (TPR) is the proportion of anomalous instances classified correctly over the total number of anomalous instances present in the test data. TPR is also known as *sensitivity*. The false positive rate (FPR) is the proportion of normal instances incorrectly classified as anomalous over the total number of normal instances contained in the test data. The true negative rate (TNR) is also called *specificity*. TPR, FPR, TNR, and the false negative rate (FNR) can be defined for the normal class. We illustrate all measures related to the confusion matrix in Figure 16.

Sensitivity is also known as the *hit rate*. Between sensitivity and specificity, sensitivity is set at high priority when the system is to be protected at all cost, and specificity gets more priority when efficiency is of major concern [227]. Consequently, the aim of an IDS is to produce as many TPs and TNs as possible while trying to reduce numbers of both FPs and FNs. The majority of evaluation criteria use these variables and the relations among them to model the accuracy of the IDSs.

(b) *ROC Curves:* The Receiver Operating Characteristics (ROC) analysis originates from signal processing theory. Its applicability is not limited only to intrusion detection, but extends to a large number of practical fields such as medical diagnosis, radiology, bioinformatics as well as in artificial intelligence and data mining. In intrusion detection, ROC curves are used on the one hand to visualize the relation between TP and FP rates of a classifier while tuning it and also to compare the accuracy with two or more classifiers. The ROC space [229], [230] uses the orthogonal coordinate system to visualize the classifier accuracy. Figure 17 illustrates the ROC approach normally used for network anomaly detection methods and systems evaluation.

(c) *Misclassification rate:* This measure attempts to estimate the probability of disagreement between the true and predicted cases by dividing the sum of FN and FP by the total number of pairs observed, i.e., (TP+FP+FN+TN). In other words, misclassification rate is defined as (FN+FP)/(TP+FP+FN+TN).

TABLE XVII
LIST OF DATASETS AVAILABLE AND THEIR DESCRIPTIONS

Dataset	*u*	*v*	*w*	*No. of instances*	*No. of attributes*	*x*	*y*	*z*	*Some references*
Synthetic	No	No	Yes	user dependent	user dependent	Not known	any	user dependent	[111], [124]
KDDcup99	Yes	No	Yes	805050	41	BCTW	P	C_1	[107], [115], [117], [123]
NSL-KDD	Yes	No	Yes	148517	41	BCTW	P	C_1	[207]
DARPA 2000	Yes	No	No	Huge	Not known	P	Raw	C_2	[214]
DEFCON	No	No	No	Huge	Not known	Raw	P	C_2	[214]
CAIDA	Yes	Yes	No	Huge	Not known	Raw	P	C_1	[214]
LBNL	Yes	Yes	No	Huge	Not known	Raw	P	C_2	[222]
ISCX-UNB	Yes	Yes	Yes	Huge	Not known	Raw	P	A	[214]
TUIDS	Yes	Yes	Yes	301760	50,24	BCTW	P, F	C_1	[124], [215]

u-realistic network configuration
v-indicates realistic traffic
w-describes the label information
x-types of features extracted as basic features (B), content-based features (C), time-based features (T) and window-based features (W)
y-explains the types of data as packet-based (P) or flow-based (F) or hybrid (H) or Others (O)
z-represents the attack category as C_1-all attacks, C_2-denial of service, C_3-probe, C_4-user to root, C_5-remote to local, and A-application layer attacks

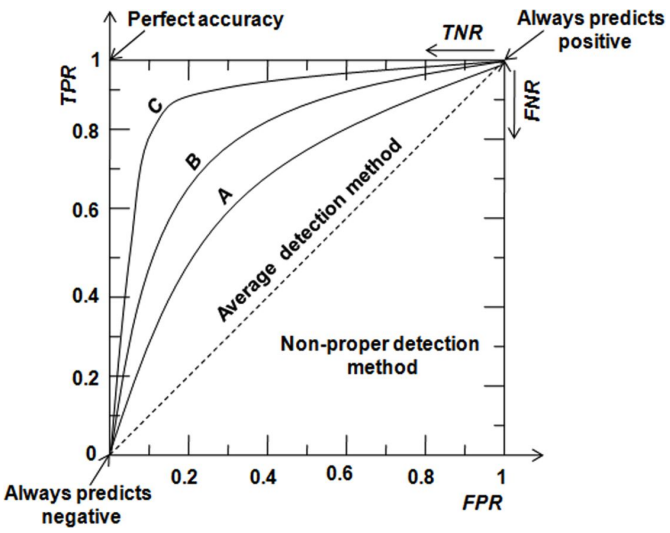

Fig. 17. Illustration of ROC measure where A, B, C represents the accuracy of a detection method or a system in ascending order.

(d) *Confusion Matrix:* The confusion matrix is a ranking method that can be applied to any kind of classification problem. The size of this matrix depends on the number of distinct classes to be detected. The aim is to compare the actual class labels against the predicted ones as shown in Figure 15. The diagonal represents correct classification. The confusion matrix for intrusion detection is defined as a 2-by-2 matrix, since there are only two classes known as intrusion and normal [40], [226], [228]. Thus, the TNs and TPs that represent the correctly predicted cases lie on the matrix diagonal while the FNs and FPs are on the right and left sides. As a side effect of creating the confusion matrix, all four values are displayed in a way that the relation between them can be easily understood.

(e) *Precision, Recall and F-measure:* Precision is a measure of how a system identifies attacks or normals. A flagging is accurate if the identified instance indeed comes from a malicious user, which is referred to as true positive. The final quantity of interest is recall, a measure of how many instances are identified correctly (see Figure 15). Precision and recall are often inversely proportional to each other and there is normally a trade-off between these two ratios. An algorithm that produces low precision and low recall is most likely defective with conceptual errors in the underlying theory. The types of attacks that are not identified can indicate which areas of the algorithm need more attention. Exposing these flaws and establishing the causes assist future improvement.

The F-measure mixes the properties of the previous two measures as the harmonic mean of precision and recall [40], [228]. If we want to use only one accuracy metric as an evaluation criterion, F-measure is the most preferable. Note that when precision and recall both reach 100%, the F-measure is the maximum, i.e., 1 meaning that the classifier has 0% false alarms and detects 100% of the attacks. Thus, a good classifier is expected to obtain F-measure as high as possible.

2) Performance: The evaluation of an IDS performance is an important task. It involves many issues that go beyond the IDS itself. Such issues include the hardware platform, the operating system or even the deployment of the IDS. For a NIDS, the most important evaluation criterion for its performance is the system's ability to process traffic on a high speed network with minimum packet loss when working real time. In real network traffic, the packets can be of various sizes, and the effectiveness of a NIDS depends on its ability to handle packets of any size. In addition to the processing speed, the CPU and memory usage can also serve as measurements of NIDS performance [231]. These are usually used as indirect measures that take into account the time and space complexities of intrusion detection algorithms. Finally, the performance of any NIDS is highly dependent upon (i) its individual configuration, (ii) the network it is monitoring, and (iii) its position in that network.

3) Completeness: The completeness criterion represents the space of the vulnerabilities and attacks that can be covered by an IDS. This criterion is very hard to assess because having omniscience of knowledge about attacks or abuses of privilege is impossible. The completeness of an IDS is judged against a complete set of known attacks. The ability of an IDS is considered complete, if it covers all the known vulnerabilities and attacks.

4) Timeliness: An IDS that performs its analysis as quickly as possible enables the human analyst or the response engine to promptly react before much damage is done within a specific time period. This prevents the attacker from subverting the audit source or the IDS itself. The response generated by the system while combating an attack is very important. Since the data must be processed to discover intrusions, there is

always a delay between the actual moment of the attack and the response of the system. This is called *total delay*. Thus, the total delay is the difference between t_{attack} and $t_{response}$. The smaller the total delay, the better an IDS is with respect to its response. No matter if an IDS is anomaly-based or signature-based, there is always a gap between the starting time of an attack and its detection.

5) Data Quality: Evaluating the quality of data is another important task in NIDS evaluation. Quality of data is influenced by several factors, such as (i) source of data (should be from reliable and appropriate sources), (ii) selection of sample (should be unbiased), (iii) sample size (neither over nor under-sampling), (iv) time of data (should be frequently updated real time data), (v) complexity of data (data should be simple enough to be handled easily by the detection mechanism), and so on.

6) Unknown attack detection: New vulnerabilities are evolving almost every day. An anomaly-based network intrusion detection system should be capable of identifying unknown attacks, in addition to known attacks. The IDS should show consistent abilities of detecting unknown or even modified intrusions.

7) Profile Update: Once new vulnerabilities or exploits are discovered, signatures or profiles must be updated for future detection. However, writing new or modified profiles or signatures without conflict is a challenge, considering the current high-speed network scenario.

8) Stability: Any anomaly detection system should perform consistently in different network scenarios and in different circumstances. It should consistently report identical events in a similar manner. Allowing the users to configure different alerts to provide different messages in different network environments may lead to an unstable system.

9) Information provided to Analyst: Alerts generated by an IDS should be meaningful enough to clearly identify the reasons behind the event to be raised, and the reasons this event is of interest. It should also assist the analyst in determining the relevance and appropriate reaction to a particular alert. The alert should also specify the source of the alert and the target system.

10) Interoperability: An effective intrusion detection mechanism is supposed to be capable of correlating information from multiple sources, such as system logs, other HIDSs, NIDSs, firewall logs and any other sources of information available. This helps in maintaining interoperability, while installing a range of HIDSs or NIDSs from various vendors.

VI. Recommendations

The following are some recommendations one needs to be mindful of when developing a network anomaly detection method or a system.

- Most existing IDSs for the wired environment work in three ways: flow level traffic or packet level feature data analysis, protocol analysis or payload inspection. Each of these categories has its own advantages and limitations. So, a hybridization of these (e.g., protocol level analysis followed by flow level traffic analysis) may give better performance in terms of known (with high detection rate) as well as unknown attack detection.
- Network anomalies may originate from various sources as discussed in Section III. So, a better IDS should be able to recognize origins of the anomalies before initiating the detection process.
- An IDS, to be capable of identifying both known as well as unknown attacks, should exploit both supervised (rule or signature-based learning) as well as unsupervised (clustering or outlier-based) at multiple levels for real time performance with low false alarm rates.
- The IDS developer should choose the basic components, method(s), techniques or rule/signature/profile base to overcome four important limitations: subjective effectiveness, limited scalability, scenario dependent efficiency and restricted security.
- The performance of a better IDS needs to be established both qualitatively and quantitatively.
- A better anomaly classification or identification method enables us to tune it (the corresponding normal profiles, thresholds, etc.) depending on the network scenario.

VII. Open Issues and challenges

Although, many methods and systems have been developed by the research community, there are still a number of open research issues and challenges. The suitability of performance metrics is a commonly identified drawback in intrusion detection. In evaluating IDSs, the three most important qualities that need to be measured are completeness, correctness, and performance. The current state-of-the-art in intrusion detection restricts evaluation of new systems to tests over incomplete datasets and micro-benchmarks that test narrowly defined components of the system. A number of anomaly-based systems have been tested using contrived datasets. Such evaluation is limited by the quality of the dataset that the system is evaluated against. Construction of a dataset which is unbiased, realistic and comprehensive is an extremely difficult task.

A formal proof of correctness [6] in the intrusion detection domain is exceptionally challenging and expensive. Therefore, "pretty good assurance" presents a way in which systems can be measured allowing fuzzy decisions, trade-offs, and priorities. Such a measure must take into consideration the amount of work required to discover a vulnerability or weakness to exploit for an attack and execute an attack on the system.

After a study of existing NIDSs, we find that it is still extremely difficult to design a new NIDS to ensure robustness, scalability and high performance. In particular, practitioners find it difficult to decide where to place the NIDS and how to best configure it for use within an environment with multiple stakeholders. We sort out some of the important issues as challenges and enumerate them below.

(i) Runtime limitation presents an important challenge for a NIDS. Without losing any packets, a real time IDS should be ideally able to capture and inspect each packet.
(ii) Most NIDSs and network intrusion detection methods depend on the environment. Ideally, a system or method should be independent of the environment.
(iii) The nature of anomalies keeps changing over time as intruders adapt their network attacks to evade existing

intrusion detection solutions. So, adaptability of a NIDS or detection method is necessary to update with the current anomalies encountered in the local network or the Internet.

(iv) Ideally, a NIDS or detection method should avoid a high rate of false alarms. However, it is not possible to escape totally from false alarms, even though it needs to aim for that in any environment and facilitate adaptability at runtime. This is another challenge for the NIDS development community.

(v) Dynamic updation of profiles in anomaly-based NIDSs without conflict and without compromising performance is an important task. The profile database needs to be updated whenever a new kind of attack is detected and addressed by the system.

(vi) Preparing an unbiased network intrusion dataset with all normal variations in profiles is another challenging task. The number of normal instances is usually large and their proportion with attack instances is very skewed in the existing publicly available intrusion datasets. Only a few intrusion datasets with sufficient amount of attack information are available publicly. Thus, there is an overarching need for benchmark intrusion datasets for evaluating NIDSs and detection methods.

(vii) Reducing computational complexity in preprocessing, training and deployment is another task that needs to be addressed.

(viii) Developing an appropriate and fast feature selection method for each attack class is yet another challenge.

(ix) Selection of an appropriate number of non-correlated, unbiased classifiers from a pool of classifiers by generating classifier hypothesis for building an effective ensemble approach for network anomaly detection is another challenge.

VIII. Concluding Remarks

In this paper, we have examined the state-of-the-art in the modern anomaly-based network intrusion detection. The discussion has emphasized two well-known criteria to classify and evaluate NIDSs: detection strategy and evaluation datasets. We have also presented many detection methods, systems and tools. In addition, we have discussed several evaluation criteria for testing the performance of a detection method or system. A brief description of the different existing datasets and its taxonomy is also provided. Finally, we outline several research issues and challenges for future researchers and practitioners who may attempt to develop new detection methods and systems for the latest network scenarios.

Acknowledgment

This work is supported by Department of Information Technology, MCIT and Council of Scientific & Industrial Research (CSIR), Government of India. It is also partially supported by NSF (US) grants CNS-0851783 and CNS-1154342. The authors are thankful to the funding agencies. The authors are also thankful to the esteemed reviewers for their extensive comments to improve the quality of the article.

References

[1] A. Sundaram, "An introduction to intrusion detection," *Crossroads*, vol. 2, no. 4, pp. 3–7, April 1996.

[2] J. P. Anderson, "Computer Security Threat Monitoring and Surveillance," James P Anderson Co, Fort Washington, Pennsylvania, Tech. Rep., April 1980.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection : A Survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15:1–15:58, September 2009.

[4] N. K. Ampah, C. M. Akujuobi, M. N. O. Sadiku, and S. Alam, "An intrusion detection technique based on continuous binary communication channels," *International J. Security and Networks*, vol. 6, no. 2/3, pp. 174–180, November 2011.

[5] F. Y. Edgeworth, "On discordant observations," *Philosophy Mag.*, vol. 23, no. 5, pp. 364–375, 1887.

[6] A. Patcha and J. M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer Networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[7] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-based network intrusion detection : Techniques, systems and challenges," *Computers & Security*, vol. 28, no. 1-2, pp. 18–28, 2009.

[8] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intellligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[9] T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning," *IEEE Commun. Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[10] M. Agyemang, K. Barker, and R. Alhajj, "A comprehensive survey of numeric and symbolic outlier mining techniques," *Intelligence Data Analysis*, vol. 10, no. 6, pp. 521–538, 2006.

[11] J. Ma and S. Perkings, "Online novelty detection on temporal sequences," in *Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2003, pp. 613–618.

[12] D. Snyder, "Online intrusion detection using sequences of system calls," Master's thesis, Department of Computer Science, Florida State University, 2001.

[13] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. John Wiley & Sons, 1987.

[14] V. Barnett and T. Lewis, *Outliers in Statistical Data*. John Wiley & Sons, 1994.

[15] D. Hawkins, *Identification of Outliers*. New York: Chapman and Hall, 1980.

[16] R. J. Beckman and R. D. Cook, "Outliers," *Technometrics*, vol. 25, no. 2, pp. 119–149, 1983.

[17] Z. Bakar, R. Mohemad, A. Ahmad, and M. Andderis, "A comparative study for outlier detection techniques in data mining," in *Proc. IEEE Conference on Cybernetics and Intelligent Systems*, 2006, pp. 1–6.

[18] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, "A Survey of Outlier Detection Methods in Network Anomaly Identification," *Computer Journal*, vol. 54, no. 4, pp. 570–588, April 2011.

[19] A. Callado, C. Kamienski, G. Szabo, B. Gero, J. Kelner, S. Fernandes, and D. Sadok, "A Survey on Internet Traffic Identification," *IEEE Commun. Surveys Tutorials*, vol. 11, no. 3, pp. 37–52, 2009.

[20] W. Zhang, Q. Yang, and Y. Geng, "A Survey of Anomaly Detection Methods in Networks," in *Proc.International Symposium on Computer Network and Multimedia Technology*, January 2009, pp. 1–3.

[21] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An Overview of IP Flow-Based Intrusion Detection," *IEEE Commun. Surveys Tutorials*, vol. 12, no. 3, pp. 343–356, quarter 2010.

[22] B. Sun, F. Yu, K. Wu, Y. Xiao, and V. C. M. Leung, "Enhancing security using mobility-based anomaly detection in cellular mobile networks," *IEEE Trans. Veh. Technol.*, vol. 55, no. 4, pp. 1385 –1396, July 2006.

[23] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," *IEEE Wireless Commun.*, vol. 14, no. 5, pp. 56–63, October 2007.

[24] B. Sun, Y. Xiao, and R. Wang, "Detection of Fraudulent Usage in Wireless Networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 6, pp. 3912–3923, November 2007.

[25] B. Sun, K. Wu, Y. Xiao, and R. Wang, "Integration of mobility and intrusion detection for wireless ad hoc networks," *International J. Communication Systems*, vol. 20, no. 6, pp. 695–721, June 2007.

[26] T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of network-based defense mechanisms countering the DoS and DDoS problems," *ACM Computing Surveys*, vol. 39, no. 1, pp. 1–42, April 2007.

[27] M. Al-Kuwaiti, N. Kyriakopoulos, and S. Hussein, "A comparative analysis of network dependability, fault-tolerance, reliability, security, and survivability," *IEEE Commun. Surveys Tutorials*, vol. 11, no. 2, pp. 106–124, April 2009.

[28] B. Donnet, B. Gueye, and M. A. Kaafar, "A Survey on Network Coordinates Systems, Design, and Security," *IEEE Commun. Surveys Tutorials*, vol. 12, no. 4, pp. 488–503, October 2010.

[29] S. X. Wu and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," *Applied Soft Computing*, vol. 10, no. 1, pp. 1–35, January 2010.

[30] Y. Dong, S. Hsu, S. Rajput, and B. Wu, "Experimental Analysis of Application Level Intrusion Detection Algorithms," *International J. Security and Networks*, vol. 5, no. 2/3, pp. 198–205, 2010.

[31] M. Tavallaee, N. Stakhanova, and A. A. Ghorbani, "Toward credible evaluation of anomaly-based intrusion-detection methods," *IEEE Trans. Syst. Man Cybern. C Appl. Rev.*, vol. 40, no. 5, pp. 516–524, September 2010.

[32] B. Daniel, C. Julia, J. Sushil, and W. Ningning, "ADAM: a testbed for exploring the use of data mining in intrusion detection," *ACM SIGMOD Record*, vol. 30, no. 4, pp. 15–24, 2001.

[33] Z. Zhang, J. Li, C. N. Manikopoulos, J. Jorgenson, and J. Ucles, "HIDE: a Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification," in *Proc. IEEE Man Systems and Cybernetics Information Assurance Workshop*, 2001.

[34] L. Ertoz, E. Eilertson, A. Lazarevic, P. Tan, V. Kumar, and J. Srivastava, *Data Mining - Next Generation Challenges and Future Directions*. MIT Press, 2004, ch. MINDS - Minnesota Intrusion Detection System.

[35] M. Thottan and C. Ji, "Anomaly detection in IP networks," *IEEE Trans. Signal Process.*, vol. 51, no. 8, pp. 2191–2204, 2003.

[36] J. M. Estevez-Tapiador, P. Garcia-Teodoro, and J. E. Diaz-Verdejo, "Anomaly detection methods in wired networks : a survey and taxonomy," *Computer Communication*, vol. 27, no. 16, pp. 1569–1584, October 2004.

[37] A. Fragkiadakis, E. Tragos, and I. Askoxylakis, "A Survey on Security Threats and Detection Techniques in Cognitive Radio Networks," *IEEE Commun. Surveys Tutorials*, vol. PP, no. 99, pp. 1–18, January 2012.

[38] R. Heady, G. Luger, A. Maccabe, and M. Servilla, "The Architecture of a Network Level Intrusion Detection System," Computer Science Department, University of New Mexico, Tech. Rep. TR-90, 1990.

[39] H. G. Kayacik, A. N. Zincir-Heywood, and M. I. Heywood, "Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets," in *Proc. 3rd Annual Conference on Privacy, Security and Trust*, October 2005.

[40] A. A. Ghorbani, W. Lu, and M. Tavallaee, *Network Intrusion Detection and Prevention : Concepts and Techniques*, ser. Advances in Information Security. Springer-verlag, October 28 2009.

[41] P. Ning and S. Jajodia, *Intrusion Detection Techniques*. H Bidgoli (Ed.), The Internet Encyclopedia, 2003.

[42] F. Wikimedia, "Intrusion detection system," http://en.wikipedia.org/wiki/Intrusion-detection_system, Feb 2009.

[43] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Surveying Port Scans and Their Detection Methodologies," *The Computer Journal*, vol. 54, no. 10, pp. 1565–1581, October 2011.

[44] B. C. Park, Y. J. Won, M. S. Kim, and J. W. Hong, "Towards automated application signature generation for traffic identification," in *Proc. IEEE/IFIP Network Operations and Management Symposium: Pervasive Management for Ubiquitous Networks and Services*, 2008, pp. 160–167.

[45] V. Kumar, "Parallel and distributed computing for cybersecurity," *IEEE Distributed Systems Online*, vol. 6, no. 10, 2005.

[46] P. N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Addison-Wesley, 2005.

[47] M. J. Lesot and M. Rifqi, "Anomaly-based network intrusion detection : Techniques, systems and challenges," *International J. Knowledge Engineering and Soft Data Paradigms*, vol. 1, no. 1, pp. 63–84, 2009.

[48] S. H. Cha, "Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions," *International J. Mathematical Models and Methods in Applied Science*, vol. 1, no. 4, pp. 300–307, November 2007.

[49] S. Choi, S. Cha, and C. C. Tappert, "A Survey of Binary Similarity and Distance Measures," *J. Systemics, Cybernetics and Informatics*, vol. 8, no. 1, pp. 43–48, 2010.

[50] M. J. Lesot, M. Rifqi, and H. Benhadda, "Similarity measures for binary and numerical data: a survey," *International J. Knowledge Engineering and Soft Data Paradigms*, vol. 1, no. 1, pp. 63–84, December 2009.

[51] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," in *Proc. 8th SIAM International Conference on Data Mining*, 2008, pp. 243–254.

[52] G. Gan, C. Ma, and J. Wu, *Data Clustering Theory, Algorithms and Applications*. SIAM, 2007.

[53] C. C. Hsu and S. H. Wang, "An integrated framework for visualized and exploratory pattern discovery in mixed data," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 2, pp. 161–173, 2005.

[54] M. V. Joshi, R. C. Agarwal, and V. Kumar, "Mining needle in a haystack: classifying rare classes via two-phase rule induction," in *Proc. 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2001, pp. 293–298.

[55] J. Theiler and D. M. Cai, "Resampling approach for anomaly detection in multispectral images," in *Proc. SPIE*, vol. 5093. SPIE, 2003, pp. 230–240.

[56] R. Fujimaki, T. Yairi, and K. Machida, "An approach to spacecraft anomaly detection problem using kernel feature space," in *Proc. 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*. USA: ACM, 2005, pp. 401–410.

[57] L. Portnoy, E. Eskin, and S. J. Stolfo, "Intrusion detection with unlabeled data using clustering," in *Proc. ACM Workshop on Data Mining Applied to Security*, 2001.

[58] H. H. Nguyen, N. Harbi, and J. Darmont, "An efficient local region and clustering-based ensemble system for intrusion detection," in *Proc. 15th Symposium on International Database Engineering & Applications*. USA: ACM, 2011, pp. 185–191.

[59] M. Dash and H. Liu, "Feature Selection for Classification," *Intelligent Data Analysis*, vol. 1, pp. 131–156, 1997.

[60] Y. Chen, Y. Li, X. Q. Cheng, and L. Guo, "Survey and taxonomy of feature selection algorithms in intrusion detection system," in *Proc. 2nd SKLOIS conference on Information Security and Cryptology*. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 153–167.

[61] Y. Li, J. L. Wang, Z. Tian, T. Lu, and C. Young, "Building lightweight intrusion detection system using wrapper-based feature selection mechanisms," *Computers & Security*, vol. 28, no. 6, pp. 466–475, 2009.

[62] H. T. Nguyen, K. Franke, and S. Petrovic, "Towards a Generic Feature-Selection Measure for Intrusion Detection," in *Proc. 20th International Conference on Pattern Recognition*, August 2010, pp. 1529–1532.

[63] A. H. Sung and S. Mukkamala, "Identifying Important Features for Intrusion Detection Using Support Vector Machines and Neural Networks," in *Proc. Symposium on Applications and the Internet*. USA: IEEE CS, 2003, pp. 209–217.

[64] H. Peng, F. Long, and C. Ding, "Feature Selection Based on Mutual Information : Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, August 2005.

[65] F. Amiri, M. M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *J. Network and Computer Applications*, vol. 34, no. 4, pp. 1184–1199, 2011.

[66] J. Dunn, "Well separated clusters and optimal fuzzy partitions," *J. Cybernetics*, vol. 4, pp. 95–104, 1974.

[67] D. L. Davies and D. W. Bouldin, "A Cluster Separation Measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 1, no. 2, pp. 224–227, 1979.

[68] L. Hubert and J. Schultz, "Quadratic assignment as a general data analysis strategy," *British J. Mathematical and Statistical Psychology*, vol. 29, no. 2, pp. 190–241, 1976.

[69] F. B. Baker and L. J. Hubert, "Measuring the power of hierarchical cluster analysis," *J. American Statistics Association*, vol. 70, no. 349, pp. 31–38, 1975.

[70] F. J. Rohlf, "Methods of Comparing Classifications," *Annual Review of Ecology and Systematics*, vol. 5, no. 1, pp. 101–113, 1974.

[71] P. J. Rousseeuw, "Silhouettes : a graphical aid to the interpretation and validation of cluster analysis," *J. Computational and Applied Mathematics*, vol. 20, no. 1, pp. 53–65, 1987.

[72] L. Goodman and W. Kruskal, "Measures of associations for cross-validations," *J. American Statistics Association*, vol. 49, pp. 732–764, 1954.

[73] P. Jaccard, "The distribution of flora in the alpine zone," *New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912.

[74] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

[75] J. C. Bezdek, "Numerical taxonomy with fuzzy sets," *J. Mathematical Biology*, vol. 1, no. 1, pp. 57–71, 1974.

[76] ——, "Cluster Validity with fuzzy sets," *J. Cybernetics*, vol. 3, no. 3, pp. 58–78, 1974.

[77] X. L. Xie and G. Beni, "A Validity measure for Fuzzy Clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 4, pp. 841–847, 1991.

[78] F. J. Anscombe and I. Guttman, "Rejection of outliers," *Technometrics*, vol. 2, no. 2, pp. 123–147, 1960.

[79] E. Eskin, "Anomaly detection over noisy data using learned probability distributions," in *Proc. 7th International Conference on Machine Learning*. Morgan Kaufmann, 2000, pp. 255–262.

[80] M. Desforges, P. Jacob, and J. Cooper, "Applications of probability density estimation to the detection of abnormal conditions in engineering," in *Proc. Institute of Mechanical Engineers*, vol. 212, 1998, pp. 687–703.

[81] C. Manikopoulos and S. Papavassiliou, "Network Intrusion and Fault Detection: A Statistical Anomaly Approach," *IEEE Commun. Mag.*, vol. 40, no. 10, pp. 76–82, October 2002.

[82] P. K. Chan, M. V. Mahoney, and M. H. Arshad, "A machine learning approach to anomaly detection," Department of Computer Science, Florida Institute of Technology, Tech. Rep. CS-2003-06, 2003.

[83] M. V. Mahoney and P. K. Chan, "Learning rules for anomaly detection of hostile network traffic," in *Proc. 3rd IEEE International Conference on Data Mining*. Washington: IEEE CS, 2003.

[84] K. Wang and S. J. Stolfo, "Anomalous Payload-Based Network Intrusion Detection," in *Proc. Recent Advances in Intrusion Detection*. springer, 2004, pp. 203–222.

[85] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Conditional Anomaly Detection," *IEEE Trans. Knowl. Data Eng.*, vol. 19, pp. 631–645, 2007.

[86] P. Chhabra, C. Scott, E. D. Kolaczyk, and M. Crovella, "Distributed Spatial Anomaly Detection," in *Proc. 27th IEEE International Conference on Computer Communications*, 2008, pp. 1705–1713.

[87] W. Lu and A. A. Ghorbani, "Network Anomaly Detection Based on Wavelet Analysis," *EURASIP J. Advances in Signal Processing*, vol. 2009, no. 837601, January 2009.

[88] F. S. Wattenberg, J. I. A. Perez, P. C. Higuera, M. M. Fernandez, and I. A. Dimitriadis, "Anomaly Detection in Network Traffic Based on Statistical Inference and α-Stable Modeling," *IEEE Trans. Dependable Secure Computing*, vol. 8, no. 4, pp. 494–509, July/August 2011.

[89] M. Yu, "A Nonparametric Adaptive CUSUM Method And Its Application In Network Anomaly Detection," *International J. Advancements in Computing Technology*, vol. 4, no. 1, pp. 280–288, 2012.

[90] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian Network Classifiers," *Machine Learning*, vol. 29, no. 2-3, pp. 131–163, November 1997.

[91] C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," in *Proc. 19th Annual Computer Security Applications Conference*, 2003.

[92] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," in *Proc. 20th International Conference on Very Large Data Bases*. San Francisco, CA, USA: Morgan Kaufmann, 1994, pp. 487–499.

[93] N. Subramoniam, P. S. Pawar, M. Bhatnagar, N. S. Khedekar, S. Guntupalli, N. Satyanarayana, V. A. Vijayakumar, P. K. Ampatt, R. Ranjan, and P. S. Pandit, "Development of a Comprehensive Intrusion Detection System - Challenges and Approaches," in *Proc. 1st International Conference on Information Systems Security*, Kolkata, India, 2005, pp. 332–335.

[94] S. Song, L. Ling, and C. N. Manikopoulo, "Flow-based Statistical Aggregation Schemes for Network Anomaly Detection," in *Proc. IEEE International Conference on Networking, Sensing*, 2006.

[95] H. Tong, C. Li, J. He, J. Chen, Q. A. Tran, H. X. Duan, and X. Li, "Anomaly Internet Network Traffic Detection by Kernel Principle Component Classifier," in *Proc. 2nd International Symposium on Neural Networks*, vol. LNCS. 3498, 2005, pp. 476–481.

[96] S. R. Gaddam, V. V. Phoha, and K. S. Balagani, "K-Means+ID3: A Novel Method for Supervised Anomaly Detection by Cascading K-Means Clustering and ID3 Decision Tree Learning Methods," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 345–354, Mar 2007.

[97] K. Das, J. Schneider, and D. B. Neill, "Anomaly pattern detection in categorical datasets," in *Proc. 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. USA: ACM, 2008, pp. 169–176.

[98] W. Lu and H. Tong, "Detecting Network Anomalies Using CUSUM and EM Clustering," in *Proc. 4th International Symposium on Advances in Computation and Intelligence*. Springer-verlag, 2009, pp. 297–308.

[99] M. A. Qadeer, A. Iqbal, M. Zahid, and M. R. Siddiqui, "Network Traffic Analysis and Intrusion Detection Using Packet Sniffer," in *Proc. 2nd International Conference on Communication Software and Networks*. Washington, DC, USA: IEEE Computer Society, 2010, pp. 313–317.

[100] I. Kang, M. K. Jeong, and D. Kong, "A differentiated one-class classification method with applications to intrusion detection," *Expert Systems with Applications*, vol. 39, no. 4, pp. 3899–3905, March 2012.

[101] C. F. Tsai, Y. F. Hsu, C. Y. Lin, and W. Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994–12 000, December 2009.

[102] T. Abbes, A. Bouhoula, and M. Rusinowitch, "Efficient decision tree for protocol analysis in intrusion detection," *International J. Security and Networks*, vol. 5, no. 4, pp. 220–235, December 2010.

[103] C. Wagner, J. François, R. State, and T. Engel, "Machine Learning Approach for IP-Flow Record Anomaly Detection," in *Proc. 10th International IFIP TC 6 conference on Networking - Volume Part I*, 2011, pp. 28–39.

[104] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, July 2001.

[105] M. Y. Su, G. J. Yu, and C. Y. Lin, "A real-time network intrusion detection system for large-scale attacks based on an incremental mining approach," *Computers & Security*, vol. 28, no. 5, pp. 301–309, 2009.

[106] L. Khan, M. Awad, and B. Thuraisingham, "A New Intrusion Detection System Using Support Vector Machines and Hierarchical Clustering," *The VLDB Journal*, vol. 16, no. 4, pp. 507–521, October 2007.

[107] Z. Muda, W. Yassin, M. N. Sulaiman, and N. I. Udzir, "A K-means and naive bayes learning approach for better intrusion detection," *Information Technology J.*, vol. 10, no. 3, pp. 648–655, 2011.

[108] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, March 1986.

[109] H. Yu and S. Kim, *Handbook of Natural Computing*. Springer, 2003, ch. SVM Tutorial - Classification, Regression and Ranking.

[110] L. V. Kuang, "DNIDS: A Dependable Network Intrusion Detection System Using the CSI-KNN Algorithm," Master's thesis, Queen's University Kingston, Ontario, Canada, Sep 2007.

[111] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "RODD: An Effective Reference-Based Outlier Detection Technique for Large Datasets," in *Advanced Computing*. Springer, 2011, vol. 133, pp. 76–84.

[112] W. Lee, S. J. Stolfo, and K. W. Mok, "Adaptive Intrusion Detection : A Data Mining Approach," *Artificial Intelligence Review*, vol. 14, no. 6, pp. 533–567, 2000.

[113] M. Roesch, "Snort - Lightweight Intrusion Detection for Networks," in *Proc. 13th USENIX Conference on System Administration*, Washington, 1999, pp. 229–238.

[114] B. Neumann, "Knowledge Management and Assistance Systems," http://kogs-www.informatik.uni-hamburg.de/ neumann/, 2007.

[115] Y. F. Zhang, Z. Y. Xiong, and X. Q. Wang, "Distributed intrusion detection based on clustering," in *Proc. International Conference on Machine Learning and Cybernetics*, vol. 4, August 2005, pp. 2379–2383.

[116] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proc. 28th Australasian conference on Computer Science - Volume 38*. Darlinghurst, Australia, Australia: Australian Computer Society, Inc., 2005, pp. 333–342.

[117] C. Zhang, G. Zhang, and S. Sun, "A Mixed Unsupervised Clustering-Based Intrusion Detection Model," in *Proc. 3rd International Conference on Genetic and Evolutionary Computing*. USA: IEEE CS, 2009, pp. 426–428.

[118] P. Casas, J. Mazel, and P. Owezarski, "Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge," *Computer Communications*, vol. 35, no. 7, pp. 772–783, April 2012.

[119] K. Sequeira and M. Zaki, "ADMIT: anomaly-based data mining for intrusions," in *Proc. eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 2002, pp. 386–395.

[120] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, *Applications of Data Mining in Computer Security*. Kluwer Academic, 2002, ch. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data.

[121] Z. Zhuang, Y. Li, and Z. Chen, "Enhancing Intrusion Detection System with proximity information," *International J. Security and Networks*, vol. 5, no. 4, pp. 207–219, December 2010.

[122] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "An effective unsupervised network anomaly detection method," in *Proc. International Conference on Advances in Computing, Communications and Informatics*. New York, NY, USA: ACM, 2012, pp. 533–539.

[123] M. E. Otey, A. Ghoting, and S. Parthasarathy, "Fast distributed outlier detection in mixed-attribute data sets," *Data Mining and Knowledge Discovery*, vol. 12, no. 2-3, pp. 203–228, 2006.

[124] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "NADO: network anomaly detection using outlier approach," in *Proc. ACM International Conference on Communication, Computing & Security*. USA: ACM, 2011, pp. 531–536.

[125] S. Jiang, X. Song, H. Wang, J.-J. Han, and Q.-H. Li, "A clustering-based method for unsupervised intrusion detections," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 802–810, May 2006.

[126] Z. Chen and C. Chen, "A Closed-Form Expression for Static Worm-Scanning Strategies," in *Proc. IEEE International Conference on Communications*. Beijing, China: IEEE CS, May 2008, pp. 1573–1577.

[127] B. Balajinath and S. V. Raghavan, "Intrusion detection through learning behavior model," *Computer Communications*, vol. 24, no. 12, pp. 1202–1212, July 2001.

[128] M. S. A. Khan, "Rule based Network Intrusion Detection using Genetic Algorithm," *International J. Computer Applications*, vol. 18, no. 8, pp. 26–29, March 2011.

[129] S. Haykin, *Neural Networks*. New Jersey: Prentice Hall, 1999.

[130] M. Amini, R. Jalili, and H. R. Shahriari, "RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks," *Computers & Security*, vol. 25, no. 6, pp. 459–468, 2006.

[131] G. Carpenter and S. Grossberg, "Adaptive resonance theory," *CAS/CNS Technical Report Series*, no. 008, 2010.

[132] T. Kohonen, "The self-organizing map," *Proc. IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[133] J. Cannady, "Applying CMAC-Based On-Line Learning to Intrusion Detection," in *Proc. IEEE-INNS-ENNS International Joint Conference on Neural Networks*, vol. 5, 2000, pp. 405–410.

[134] S. C. Lee and D. V. Heinbuch, "Training a neural-network based intrusion detector to recognize novel attacks," *IEEE Trans. Syst. Man Cybern. A*, vol. 31, no. 4, pp. 294–299, 2001.

[135] G. Liu, Z. Yi, and S. Yang, "A hierarchical intrusion detection model based on the PCA neural networks," *Neurocomputing*, vol. 70, no. 7-9, pp. 1561–1568, 2007.

[136] J. Sun, H. Yang, J. Tian, and F. Wu, "Intrusion Detection Method Based on Wavelet Neural Network," in *Proc. 2nd International Workshop on Knowledge Discovery and Data Mining*. USA: IEEE CS, 2009, pp. 851–854.

[137] H. Yong and Z. X. Feng, "Expert System Based Intrusion Detection System," in *Proc. International Conference on Information Management, Innovation Management and Industrial Engineering*, vol. 4, November 2010, pp. 404 –407.

[138] A. Parlos, K. Chong, and A. Atiya, "Application of the recurrent multilayer perceptron in modeling complex process dynamics," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 255–266, 1994.

[139] K. Labib and R. Vemuri, "NSOM: A Tool To Detect Denial Of Service Attacks Using Self-Organizing Maps," Department of Applied Science University of California, Davis Davis, California, U.S.A., Tech. Rep., 2002.

[140] D. Bolzoni, S. Etalle, P. H. Hartel, and E. Zambon, "POSEIDON: a 2-tier Anomaly-based Network Intrusion Detection System," in *Proc. 4th IEEE International Workshop on Information Assurance*, 2006, pp. 144–156.

[141] M. V. Mahoney and P. K. Chan, "PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic," Dept. of Computer Science, Florida Tech, Tech. Rep. cs-2001-04, 2001.

[142] J. E. Dickerson, "Fuzzy network profiling for intrusion detection," in *Proc. 19th International Conference of the North American Fuzzy Information Processing Society*, Atlanta, July 2000, pp. 301–306.

[143] F. Geramiraz, A. S. Memaripour, and M. Abbaspour, "Adaptive Anomaly-Based Intrusion Detection System Using Fuzzy Controller," *International Journal of Network Security*, vol. 14, no. 6, pp. 352–361, 2012.

[144] A. Tajbakhsh, M. Rahmati, and A. Mirzaei, "Intrusion detection using fuzzy association rules," *Applied Soft Computing*, vol. 9, no. 2, pp. 462–469, March 2009.

[145] S. Mabu, C. Chen, N. Lu, K. Shimada, and K. Hirasawa, "An Intrusion-Detection Model Based on Fuzzy Class-Association-Rule Mining Using Genetic Network Programming," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 41, no. 1, pp. 130–139, 2011.

[146] J. Q. Xian, F. H. Lang, and X. L. Tang, "A novel intrusion detection method based on clonal selection clustering algorithm," in *Proc. International Conference on Machine Learning and Cybernetics*. USA: IEEE Press, 2005, vol. 6.

[147] M. Mohajerani, A. Moeini, and M. Kianie, "NFIDS: A Neuro-Fuzzy Intrusion Detection System," in *Proc. 10th IEEE International Conference on Electronics, Circuits and Systems*, vol. 1, December 2003, pp. 348–351.

[148] Z. Pawlak, "Rough sets," *International J. Parallel Programming*, vol. 11, no. 5, pp. 341–356, 1982.

[149] Z. Cai, X. Guan, P. Shao, Q. Peng, and G. Sun, "A rough set theory based method for anomaly intrusion detection in computer network systems," *Expert Systems*, vol. 20, no. 5, pp. 251–259, November 2003.

[150] W. Chimphlee, A. H. Abdullah, M. S. M. Noor, S. Srinoy, and S. Chimphlee, "Anomaly-Based Intrusion Detection using Fuzzy Rough Clustering," in *Proc. International Conference on Hybrid Information Technology*, vol. 01. Washington, DC, USA: IEEE Computer Society, 2006, pp. 329–334.

[151] A. O. Adetunmbi, S. O. Falaki, O. S. Adewale, and B. K. Alese, "Network Intrusion Detection based on Rough Set and k-Nearest Neighbour," *International J. Computing and ICT Research*, vol. 2, no. 1, pp. 60–66, 2008.

[152] R. C. Chen, K. F. Cheng, Y. H. Chen, and C. F. Hsieh, "Using Rough Set and Support Vector Machine for Network Intrusion Detection System," in *Proc. First Asian Conference on Intelligent Information and Database Systems*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 465–470.

[153] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man Cybern. B, Cybern.*, vol. 26, no. 1, pp. 29–41, 1996.

[154] H. H. Gao, H. H. Yang, and X. Y. Wang, "Ant colony optimization based network intrusion feature selection and detection," in *Proc. International Conference on Machine Learning and Cybernetics*, vol. 6, aug. 2005, pp. 3871–3875.

[155] A. Visconti and H. Tahayori, "Artificial immune system based on interval type-2 fuzzy set paradigm," *Applied Soft Computing*, vol. 11, no. 6, pp. 4055–4063, September 2011.

[156] S. Noel, D. Wijesekera, and C. Youman, "Modern Intrusion Detection, Data Mining, and Degrees of Attack Guilt," in *Proc. International Conference on Applications of Data Mining in Computer Security*. Springer, 2002.

[157] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and et al., "Specification-based anomaly detection: a new approach for detecting network intrusions," in *Proc. 9th ACM Conference on Computer and Communications Security*, 2002, pp. 265–274.

[158] X. Xu, "Sequential anomaly detection based on temporal-difference learning: Principles, models and case studies," *Applied Soft Computing*, vol. 10, no. 3, pp. 859–867, 2010.

[159] A. Prayote, "Knowledge Based Anomaly Detection," Ph.D. dissertation, School of Computer Science and Egineering, The University of New South Wales, November 2007.

[160] K. Ilgun, R. A. Kemmerer, and P. A. Porras, "State transition analysis: A rule-based intrusion detection approach," *IEEE Trans. Software Eng.*, vol. 21, no. 3, pp. 181–199, 1995.

[161] D. E. Denning and P. G. Neumann, "Requirements and model for IDES a real-time intrusion detection system," Computer Science Laboratory, SRI International, USA, Tech. Rep. 83F83-01-00, 1985.

[162] D. Anderson, T. F. Lunt, H. Javitz, A. Tamaru, and A. Valdes, "Detecting unusual program behaviour using the statistical component of the next-generation intrusion detection expert system (NIDES)," Computer Science Laboratory, SRI International, USA, Tech. Rep. SRIO-CSL-95-06, 1995.

[163] N. G. Duffield, P. Haffner, B. Krishnamurthy, and H. Ringberg, "Rule-Based Anomaly Detection on IP Flows," in *Proc. 28th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies*. Rio de Janeiro, Brazil: IEEE press, 2009, pp. 424–432.

[164] R. E. Schapire, "A brief introduction to boosting," in *Proc. 16th International Joint Conference on Artificial Intelligence*, Morgan Kaufmann, 1999, pp. 1401–1406.

[165] A. Prayote and P. Compton, "Detecting anomalies and intruders," *AI 2006: Advances in Artificial Intelligence*, pp. 1084–1088, 2006.

[166] G. Edwards, B. Kang, P. Preston, and P. Compton, "Prudent expert systems with credentials: Managing the expertise of decision support systems," *International journal of biomedical computing*, vol. 40, no. 2, pp. 125–132, 1995.

[167] W. Scheirer and M. C. Chuah, "Syntax vs. semantics : competing approaches to dynamic network intrusion detection," *International Journal Securrity and Networks*, vol. 3, no. 1, pp. 24–35, December 2008.

[168] P. Naldurg, K. Sen, and P. Thati, "A Temporal Logic Based Framework for Intrusion Detection," in *Proc. 24th IFIP WG 6.1 International*

Conference on Formal Techniques for Networked and Distributed Systems, 2004, pp. 359–376.

[169] J. M. Estevez-Tapiador, P. Garcya-Teodoro, and J. E. Dyaz-Verdejo, "Stochastic protocol modeling for anomaly based network intrusion detection," in *Proc. 1st International Workshop on Information Assurance*. IEEE CS, 2003, pp. 3–12.

[170] A. Shabtai, U. Kanonov, and Y. Elovici, "Intrusion detection for mobile devices using the knowledge-based, temporal abstraction method," *J. System Software*, vol. 83, no. 8, pp. 1524–1537, August 2010.

[171] S. S. Hung and D. S. M. Liu, "A user-oriented ontology-based approach for network intrusion detection," *Computer Standards & Interfaces*, vol. 30, no. 1-2, pp. 78–88, January 2008.

[172] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, 2006.

[173] A. Borji, "Combining heterogeneous classifiers for network intrusion detection," in *Proc. 12th Asian Computing Science Conference on Advances in Computer Science: Computer and Network Security*. Springer, 2007, pp. 254–260.

[174] G. Giacinto, R. Perdisci, M. D. Rio, and F. Roli, "Intrusion detection in computer networks by a modular ensemble of one-class classifiers," *Information Fusion*, vol. 9, no. 1, pp. 69–82, January 2008.

[175] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, February 2010.

[176] K. Noto, C. Brodley, and D. Slonim, "Anomaly Detection Using an Ensemble of Feature Models," in *Proc. IEEE International Conference on Data Mining*. USA: IEEE CS, 2010, pp. 953–958.

[177] P. M. Mafra, V. Moll, J. D. S. Fraga, and A. O. Santin, "Octopus-IIDS: An Anomaly Based Intelligent Intrusion Detection System," in *Proc. IEEE Symposium on Computers and Communications*. USA: IEEE CS, 2010, pp. 405–410.

[178] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *Computers & Security*, vol. 24, no. 4, pp. 295–307, 2005.

[179] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984.

[180] R. Perdisci, G. Gu, and W. Lee, "Using an Ensemble of One-Class SVM Classifiers to Harden Payload-based Anomaly Detection Systems," in *Proc. 6th International Conference on Data Mining*. USA: IEEE CS, 2006, pp. 488–498.

[181] G. Folino, C. Pizzuti, and G. Spezzano, "An ensemble-based evolutionary framework for coping with distributed intrusion detection," *Genetic Programming and Evolvable Machines*, vol. 11, no. 2, pp. 131–146, June 2010.

[182] M. Rehak, M. Pechoucek, P. Celeda, J. Novotny, and P. Minarik, "CAMNEP: Agent-based Network Intrusion Detection System," in *Proc. 7th International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track*. Richland, SC: IFAAMS, 2008, pp. 133–136.

[183] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, "McPAD: A multiple classifier system for accurate payload-based anomaly detection," *Computer Networks*, vol. 53, no. 6, pp. 864–881, April 2009.

[184] W. Khreich, E. Granger, A. Miri, and R. Sabourin, "Adaptive ROC-based ensembles of HMMs applied to anomaly detection," *Pattern Recognition*, vol. 45, no. 1, pp. 208–230, January 2012.

[185] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks," *Pattern Recognition Letters*, vol. 24, no. 12, pp. 1795–1803, August 2003.

[186] J. Shifflet, "A Technique Independent Fusion Model For Network Intrusion Detection," in *Proc. Midstates Conference on Undergraduate Research in Computer Science and Mathematics*, vol. 3, 2005, pp. 13–19.

[187] D. Parikh and T. Chen, "Data Fusion and Cost Minimization for Intrusion Detection," *IEEE Trans. Inf. For. Security*, vol. 3, no. 3, pp. 381–389, 2008.

[188] L. Zhi-dong, Y. Wu, W. Wei, and M. Da-peng, "Decision-level fusion model of multi-source intrusion detection alerts," *J. Communications*, vol. 32, no. 5, pp. 121–128, 2011.

[189] R. Yan and C. Shao, "Hierarchical Method for Anomaly Detection and Attack Identification in High-speed Network," *Information Technology J.*, vol. 11, no. 9, pp. 1243–1250, 2012.

[190] V. Chatzigiannakis, G. Androulidakis, K. Pelechrinis, S. Papavassiliou, and V. Maglaris, "Data fusion algorithms for network anomaly detection: classification and evaluation," in *Proc. 3rd International Conference on Networking and Services*. Greece: IEEE CS, 2007, pp. 50–57.

[191] W. Gong, W. Fu, and L. Cai, "A Neural Network Based Intrusion Detection Data Fusion Model," in *Proc. 3rd International Joint Conference on Computational Science and Optimization - Volume 02*. USA: IEEE CS, 2010, pp. 410–414.

[192] D. Ariu, R. Tronci, and G. Giacinto, "HMMPayl: An intrusion detection system based on Hidden Markov Models," *Computers & Security*, vol. 30, no. 4, pp. 221–241, 2011.

[193] M. Arumugam, P. Thangaraj, P. Sivakumar, and P. Pradeepkumar, "Implementation of two class classifiers for hybrid intrusion detection," in *Proc. International Conference on Communication and Computational Intelligence*, December 2010, pp. 486–490.

[194] J. Zhang and M. Zulkernine, "A Hybrid Network Intrusion Detection Technique Using Random Forests," in *Proc. 1st International Conference on Availability, Reliability and Security*. USA: IEEE CS, 2006, pp. 262–269.

[195] M. A. Aydin, A. H. Zaim, and K. G. Ceylan, "A hybrid intrusion detection system design for computer network security," *Computers & Electrical Engineering*, vol. 35, no. 3, pp. 517–526, May 2009.

[196] M. Panda, A. Abraham, and M. R. Patra, "Hybrid intelligent systems for detecting network intrusions," *Computer Physics Communications*, vol. Early, 2012.

[197] A. Herrero, M. Navarro, E. Corchado, and V. Julian, "RT-MOVICAB-IDS: Addressing real-time intrusion detection," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 250–261, 2011.

[198] M. E. Locasto, K. Wang, A. D. Keromytis, and S. J. Stolfo, "FLIPS: Hybrid Adaptive Intrusion Prevention," in *Recent Advances in Intrusion Detection*, 2005, pp. 82–101.

[199] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems," *J. Network and Computer Applications*, vol. 30, no. 1, pp. 114–132, January 2007.

[200] J. Zhang, M. Zulkernine, and A. Haque, "Random-Forests-Based Network Intrusion Detection Systems," *IEEE Trans. Syst. Man Cybern. C*, vol. 38, no. 5, pp. 649–659, 2008.

[201] X. Tong, Z. Wang, and H. Yu, "A research using hybrid RBF/Elman neural networks for intrusion detection system secure model," *Computer Physics Communications*, vol. 180, no. 10, pp. 1795–1801, 2009.

[202] X. Yu, "A New Model of Intelligent Hybrid Network Intrusion Detection System," in *Proc. International Conference on Bioinformatics and Biomedical Technology*. IEEE CS, 2010, pp. 386–389.

[203] S. Selim, M. Hashem, and T. M. Nazmy, "Hybrid Multi-level Intrusion Detection System," *International J. Computer Science and Information Security*, vol. 9, no. 5, pp. 23–29, 2011.

[204] A. Soule, K. Salamatian, and N. Taft, "Combining filtering and statistical methods for anomaly detection," in *Proc. 5th ACM SIGCOMM conference on Internet Measurement*. USA: ACM, 2005, pp. 1–14.

[205] KDDcup99, "Knowledge discovery in databases DARPA archive," http://www.kdd.ics.uci.edu/databases/kddcup99/ task.html, 1999.

[206] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost-Based Modeling for Fraud and Intrusion Detection: Results from the JAM Project," in *Proc. DARPA Information Survivability Conference and Exposition*, vol. 2. USA: IEEE CS, 2000, pp. 130–144.

[207] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. 2nd IEEE International Conference on Computational Intelligence for Security and Defense Applications*. USA: IEEE Press, 2009, pp. 53–58.

[208] NSL-KDD, "NSL-KDD data set for network-based intrusion detection systems," http://iscx.cs.unb.ca/NSL-KDD/, March 2009.

[209] I. S. T. G. MIT Lincoln Lab, "DARPA Intrusion Detection Data Sets," http://www.ll.mit.edu/mission/communications/ ist/corpora/ideval/data/2000data.html, March 2000.

[210] Defcon, "The Shmoo Group," http://cctf.shmoo.com/, 2011.

[211] CAIDA, "The cooperative Analysis for Internet Data Analysis," http://www.caida.org, 2011.

[212] LBNL, "Lawrence Berkeley National Laboratory and ICSI, LBNL/ICSI Enterprise Tracing Project," http://www.icir.org/enterprise-tracing/, 2005.

[213] UNIBS, "University of Brescia dataset," http://www.ing.unibs.it/ntw/tools/traces/, 2009.

[214] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Towards developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, vol. 31, no. 3, pp. 357–374, 2012.

[215] P. Gogoi, M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Packet and Flow Based Network Intrusion Datasets," in *Proc. 5th International Conference on Contemporary Computing*, vol. LNCS-CCIS 306. Springer, August 6-8 2012, pp. 322–334.

[216] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "AOCD : An Adaptive Outlier Based Coordinated Scan Detection Approach," *International J. Network Security*, vol. 14, no. 6, pp. 339–351, 2012.

[217] C. Satten, "Lossless Gigabit Remote Packet Capture With Linux," http://staff.washington.edu/corey/gulp/, 2007.

[218] NFDUMP, "NFDUMP Tool," http://nfdump.sourceforge.net/, 2011.

[219] M. V. Mahoney and P. K. Chan, "An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection," in *Proc. 6th International Symposium on Recent Advances in Intrusion Detection*. Springer, 2003, pp. 220–237.

[220] J. McHugh, "Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory," *ACM Trans. Inf. System Security*, vol. 3, no. 4, pp. 262–294, November 2000.

[221] P. Mell, V. Hu, R. Lippmann, J. Haines, and M. Zissman, "An Overview of Issues in Testing Intrusion Detection Systems," http://citeseer.ist.psu.edu/621355.html, 2003.

[222] J. Xu and C. R. Shelton, "Intrusion Detection using Continuous Time Bayesian Networks," *J. Artificial Intelligence Research*, vol. 39, pp. 745–774, 2010.

[223] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Trans. Inf. System Security*, vol. 3, no. 3, pp. 186–205, August 2000.

[224] R. P. Lippmann, D. J. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. W. D. Wyschogord, R. K. Cunningham, and M. A. Zissman, "Evaluating Intrusion Detection Systems: The 1998 DARPA Offline Intrusion Detection Evaluation," in *Proc. DARPA Information Survivability Conference and Exposition*, January 2000, pp. 12–26.

[225] M. V. Joshi, R. C. Agarwal, and V. Kumar, "Predicting rare classes : can boosting make any weak learner strong?" in *Proc. 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. USA: ACM, 2002, pp. 297–306.

[226] P. Dokas, L. Ertoz, A. Lazarevic, J. Srivastava, and P. N. Tan, "Data Mining for Network Intrusion Detection," in *Proc. NSF Workshop on Next Generation Data Mining*, November 2002.

[227] Y. Wang, *Statistical Techniques for Network Security : Modern Statistically-Based Intrusion Detection and Protection*. Hershey, PA: Information Science Reference, IGI Publishing, 2008.

[228] S. M. Weiss and T. Zhang, *The handbook of data mining*. Lawrence Erlbaum Assoc Inc, 2003, ch. Performance Alanysis and Evaluation, pp. 426–439.

[229] F. J. Provost and T. Fawcett, "Robust Classification for Imprecise Environments," *Machine Learning*, vol. 42, no. 3, pp. 203–231, 2001.

[230] R. A. Maxion and R. R. Roberts, "Proper Use of ROC Curves in Intrusion/Anomaly Detection," School of Computing Science, University of Newcastle upon Tyne, Tech. Rep. CS-TR-871, November 2004.

[231] R. Sekar, Y. Guang, S. Verma, and T. Shanbhag, "A high-performance network intrusion detection system," in *Proc. 6th ACM Conference on Computer and Communications Security*. USA: ACM, 1999, pp. 8–17.

Monowar Hussain Bhuyan received his M.Tech. in Information Technology from the Department of Computer Science and Engineering, Tezpur University, Assam, India in 2009. Currently, he is pursuing his Ph.D. in Computer Science and Engineering from the same university. He is a life member of IETE, India. His research areas include machine learning, computer and network security, pattern recognition. He has published 15 papers in international journals and referred conference proceedings.

Dhruba K. Bhattacharyya received his Ph.D. in Computer Science from Tezpur University in 1999. He is a Professor in the Computer Science & Engineering Department at Tezpur University. His research areas include Data Mining, Network Security and Content-based Image Retrieval. Prof. Bhattacharyya has published 150+ research papers in the leading international journals and conference proceedings. In addition, Dr Bhattacharyya has written/edited 8 books. He is a Programme Committee/Advisory Body member of several international conferences/workshops.

Jugal K. Kalita is a professor of Computer Science at the University of Colorado at Colorado Springs. He received his Ph.D. from the University of Pennsylvania. His research interests are in natural language processing, machine learning, artificial intelligence and bioinformatics. He has published 120 papers in international journals and referred conference proceedings and has written two books.