1-2012

Network Intrusion Detection Types and Computation

Purvag Patel
Southern Illinois University Carbondale

Chet Langin
Southern Illinois University Carbondale

Feng Yu
Southern Illinois University Carbondale

Shahram Rahimi
Southern Illinois University Carbondale, rahimi@cs.siu.edu

Published in *International Journal of Computer Science and Information Security* , Vol. 10 No. 1 (January 2012) at https://sites.google.com/site/ijcsis/vol-10-no-1-jan-2012

Recommended Citation

Patel, Purvag, Langin, Chet, Yu, Feng and Rahimi, Shahram. "Network Intrusion Detection Types and Computation." (Jan 2012).

Network Intrusion Detection Types and Computation

Purvag Patel, Chet Langin, Feng Yu, and Shahram Rahimi
Southern Illinois University Carbondale, Carbondale, IL, USA

Abstract—**Our research created a network Intrusion Detection Math (ID Math) consisting of two components: (1) a way of specifying intrusion detection types in a manner which is more suitable for an analytical environment; and (2) a computational model which describes methodology for preparing intrusion detection data stepwise from network packets to data structures in a way which is appropriate for sophisticated analytical methods such as statistics, data mining, and computational intelligence. We used ID Math in a production Self-Organizing Map (SOM) intrusion detection system named ANNaBell as well as in the SOM+ Diagnostic System which we developed.**

Index Terms—**Computational intelligence, Data Mining, ID Math, Intrusion Detection Types, Log Analysis**

I. Introduction

Every hacker in the world is one's neighbor on the Internet, which results in attack defense and detection being pervasive both at home and work. Although hundreds of papers have been written on a large variety of methods of intrusion detection—from log analysis, to packet analysis, statistics, data mining, and sophisticated computational intelligence methods—and even though similar data structures are used by the various types of intrusion analysis, apparently little has been published on a methodical mathematical description of how data is manipulated and perceived in network intrusion detection from binary network packets to more manageable data structures such as vectors and matrices.

We developed a comprehensive methodology of information security Intrusion Detection Math (ID Math) which overhauls concepts of intrusion detection including a new model of intrusion detection types and a computational model created in order to lay a foundation for data analysis. Our intrusion detection types are necessary, complete, and mutually exclusive. They facilitate *apples-to-apples* and *oranges-to-oranges* comparisons of intrusion detection methods and provide the ability to focus on different kinds of intrusion detection research. Our computational model converts intrusion detection data from packet analysis step-by-step to sophisticated computational intelligent methods. These concepts of ID Math were implemented in a production Self-Organizing Map (SOM) intrusion detection system named ANNaBell and were introduced in publication as part of the SOM+ Diagnostic System in [1].

Section II describes background and literature. We describe the new types of local network intrusion detection in section III, and we propose the network intrusion detection computation model in section IV. The conclusion is in section V.

II. Background and Literature

Intrusion detection is the process of identifying and responding to malicious activity targeted at computing and networking sources [2]. Over the years, types of intrusion detection have been labeled in various linguistic terms, with often vague or overlapping meanings. Not all researchers have used the same labels with the same meanings. To demonstrate the need for consistent labeling of intrusion types, previous types of intrusion detection are listed below in order to show the variety of types of labeling that have been used in the past.

Denning [3] in 1986 referred to intrusion detection methods which included profiles, anomalies, and rules. Her profiling included metrics and statistical models. She referred to misuse in terms of insiders who misused privileges.

Young in 1987 [4] defined two types of monitors: appearance monitors and behavior monitors, the first performing static analysis of systems to detect anomalies and the second examining behavior.

Lunt [5] in 1988 referred to the misuse of insiders; the finding of abnormal behavior by determining departures from historically established norms of behavior; a priori rules; and using expert system technology to codify rules obtained from system security officers. A year later, in 1989, Lunt mentioned knowledge-based, statistical, and rule-based intrusion detection. In 1993, she referred to model-based reasoning [6].

Vaccaro and Liepins [7] in 1989 stated that misuse manifests itself as anomalous behavior. Hellman, Liepins, and Richards [8] in 1992 stated that computer use is either normal or misuse. Denault, et al, [9] in 1994 referred to detection-by-appearance and detection-by-behavior. Forrest, et al, [10] in 1994 said there were three types: activity monitors, signature scanners, and file authentication programs.

Intrusion detection types began converging on two main types in 1994: misuse and anomaly. Crosbie and Spafford [11] defined misuse detection as watching for certain actions being performed on certain objects. They defined anomaly detection as deviations from normal system usage patterns. Kumar and Spafford [12] also referred to anomaly and misuse detection in 1994. Many other researchers, too numerous to mention them all, have also referred to misuse and anomaly as the two main types of intrusion detection, from 1994 up to the present time.

However, other types of intrusion detection continue to be mentioned. Ilgun, Kemmerer, and Porras [13] in 1995 referred to four types: Threshold, anomaly, rule-based, and model-based. Esmaili, Safavi-Naini, and Pieprzyk [14] in 1996 said the two main methods are statistical and rule-based expert systems.

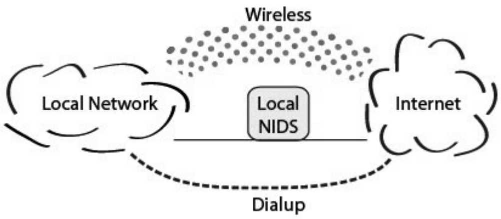


Fig. 1. A Local Landline NIDS

Debar, Dacier, and Wespi, [15] in 1999 referred to two complementary trends: (1) The search for evidence based on knowledge; and, (2) the search for deviations from a model of unusual behavior based on observations of a system during a known normal state. The first they referred to as misuse detection, detection by appearance, or knowledge-based. The second they referred to as anomaly detection or detection by behavior. Bace [16] in 2000 described misuse detection as looking for something bad and anomaly detection as looking for something rare or unusual. Marin-Blazquez and Perez [17] in 2008 said that there are three main approaches: signature, anomaly, and misuse detection.

While descriptive, these various labels over time are inconsistent and do not favor an analytical discussion of network intrusion detection. Not all of them are necessary, they are not mutually exclusive, and as individual groups they have not been demonstrated as being complete. Rather than arbitrate which of these labels should be used and how they should be defined, new labels have been created to describe types of local network intrusion detection in a manner which favors an analytical environment.

III. LLNIDS Types of Intrusion Detection

The new types are explained below, but first some terminology needs to be stated in order to later describe the types. An Intrusion Detection System (IDS) is software or an appliance that detects intrusions. A Network Intrusion Detection System (NIDS) is an appliance that detects an intrusion on a network. In this research, network means a landline network. Local network intrusion detection refers to the instant case of network intrusion detection.

Figure 1 illustrates the location of a Local Landline Network Intrusion Detection System (LLNIDS) as used in this research. The LLNDS in Figure 1 is represented by the rounded box in the center labelled “Local NIDS”. It is an IDS on a landline between a local network and the Internet. The point of view of this research is from inside the LLNIDS. Users on the local network may have other ways of accessing the Internet that bypass the LLNIDS, such as wireless and dialup. This research is restricted to the LLNIDS as described here.

Examples of detection which are not Local Landline Network Intrusion Detection (LLNID) include detection on the host computer, detection by someone else out on the Internet, or detection by someone out in the world, such as someone witnessing a perpetrator bragging in a bar. This research concerns LLNID and the new types described in this paper refer to LLNID. A network intrusion in this context means one or more transmissions across the network that involves an intrusion. A single Internet transmission is often called a packet. Therefore, using this terminology, the physical manifestation of an intrusion on a network is one or more packets, and intrusion detection is the detection of these packets that constitute intrusions. In this context, intrusion detection is similar to data mining. Intrusion detection research needs a model of types of intrusions and types of intrusion detection that benefits analysis of methods. This research focuses only on LLNID. These are the proposed types of intrusions for the special case of local landline network intrusion detection that facilitate intrusion detection research analysis in the LLNID context:

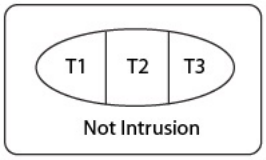


Fig. 2. Types of Intrusions for LLNIDS

- *Type 1 Intrusion*: An intrusion which can be positively detected in one or more packets in transit on the local network in a given time period.
- *Type 2 Intrusion*: An intrusion for which one or more symptoms (only) can be detected in one or more packets in transit on the local network in a given time period.
- *Type 3 Intrusion*: An intrusion which cannot be detected in packets in transit on the network in a given time period.

These three types of intrusions are necessary for analytical research in order to indicate and compare kinds of intrusions. A positive intrusion is different than only a symptom of an intrusion because immediate action can be taken on the first whereas further analysis should be taken on the second. Both of these are different than intrusions which have been missed by an LLNIDS. To show that these three types are mutually exclusive and are complete for a given time period, consider all of the intrusions for a given time period, such as a 24-hour day. The intrusions which were positively identified by the LLNIDS are Type1 intrusions. Of the remaining intrusions, the ones for which the LLNIDS found symptoms are Type 2. Here the hypothesis is that the LLNIDS can only find an intrusion positively or only one or more symptoms are found. No other results can be returned by the LLNIDS. Therefore, the remaining intrusions are Type 3, which are intrusions not detected by the LLNIDS. No other types of intrusions in this context are possible.

Figure 2 is a diagram that illustrates the types of intrusions as described above. An intrusion is either Type 1, Type 2, Type 3, or it is not an intrusion.

Those were the types of intrusions. Next are the types of intrusion detection. There are three types of network intrusion detection that correspond to the three types of intrusions in the LLNID context:

- *Type 1 Network Intrusion Detection*: A Type 1 Intrusion is detected in a given time period.

- *Type 2 Network Intrusion Detection*: One or more symptoms (only) of a Type 2 Intrusion are detected in a given time period.
- *Type 3 Network Intrusion Detection*: No intrusion is detected in a given time period.

Admittedly, Type 3 is not a detection but the lack of detection. It is included because these three types of detection correspond to the three types of intrusions and Type 3 Intrusion Detection facilitates analysis of intrusion detection methods. Examples of Type 3 Intrusion Detection are nothing was detected; no attempt was made at detection; an intrusion occurred but was not detected by the LLNIDS; and, no intrusion occurred. All of these have the same result: there was no detection of an intrusion by the LLNIDS.

Each of the three network intrusion detection types is necessary to describe all of the types of intrusion detection. A positive detection of an intrusion is different than just a symptom of an intrusion because a positive detection can be immediately acted upon while a symptom indicates that further analysis is needed. Both of these are different than intrusions that are missed by network intrusion detection. To show that these types are mutually exclusive and complete for a given time period, consider an LLNIDS looking at network packets for a given time period, say a 24-hour day. For all packets that the LLNIDS determines positively indicates an intrusion the LLNIDS has accomplished Type 1 intrusion detection. Of the remaining packets, for each packet that the LLNIDS determines is a symptom of an intrusion the LLNIDS has accomplished Type 2 intrusion detection. The remaining packets represent Type 3 intrusion detection. These three types of network intrusion detection are complete in this context because they cover all possibilities of intrusion detection. In common language, Type 1 is a certainty, Type 2 is a symptom, and Type 3 is an unknown.

Those were types of intrusion detection. Next are types of methods and alerts. LLNID methods can be defined in terms of the three intrusion types:

- *Type 1 NID Method/Alert*: A method that detects a Type 1 Intrusion and an alert that indicates a Type 1 Intrusion.
- *Type 2 NID Method/Alert*: A method that detects a symptom of a Type 2 Intrusion and an alert that indicates a symptom (only) of a Type 2 Intrusion.
- *Type 3 NID Method/Alert*: A method that does not exist, thus there is no alert.

These types of methods and alerts are necessary to differentiate that some methods are positively correct, other methods only indicate symptoms of intrusions, and some methods do not exist. They are mutually exclusive because a local method either positively indicates an intrusion (Type 1), it only detects a symptom of an intrusion (Type 2), or it does not exist (Type 3). They are complete because there are no other types of methods in this context.

Those were types of methods and alerts. Next are types of false positives. The term false positive generally has meant that an intrusion detection system has sent a false alarm. False positives are generally undesirable because the false positive rate of intrusion detection systems can be high and can use up a lot of seemingly unnecessary, and limited, resources. However, with these new types, the concept of a false positive is different for different intrusion detection types in the LLNIDS context.

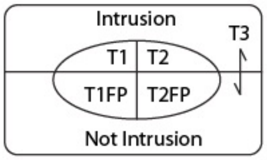


Fig. 3. Types of Intrusion Detection for LLNID

- *Type 1 False Positive*: A Type 1 Method produces an alarm in the absence of an intrusion.
- *Type 2 False Positive*: A Type 2 method produces an alarm in the absence of an intrusion.
- *Type 3 False Positive*: Does not exist because no alarm is produced.

A Type 1 False Positive indicates a problem with the Type 1 method which should be corrected. Type 2 False Positives are expected because Type 2 Methods do not positively detect intrusions, they only detect symptoms of intrusions. There is no Type 3 False Positive because no detections and alerts are produced for Type 3 Intrusion Detections. These types of false positive are necessary because they each indicate separate network intrusion detection issues. Type 1 is a network intrusion detection problem which needs to be corrected and Type 2 is expected. The two types of false positive are mutually exclusive and complete because only Type 1 Network Intrusion Detection can produce a Type 1 False Positive and only Type 2 Network Intrusion Detection can produce a Type 2 False Positive. No other types of false positives in this context are possible. Since Type 1 and Type 2 of local network intrusion detection methods are mutually exclusive, these are also mutually exclusive.

Figure 3 is a Venn diagram which illustrates types of intrusion detection in the LLNIDS context. The horizontal line separates intrusions at the top from non-intrusions at the bottom. A Type 1 detection is in the upper left of the circle if it is actually an intrusion or it is in the lower left of the circle if it is a false positive. A Type 2 detection is in the upper right of the circle if it is actually an intrusion or it is in the lower right of the circle if it is a false positive. Everything outside of the circle is Type 3 detection whether it is an intrusion or not.

This typing system allows illustration that empirically most intrusion detection is not Type 1 (positive detections), but Type 2 (symptoms of detections), and Type 3 (missed detections). This differentiation is essential in proceeding in a scientific way for improved intrusion detection.

Previously labeled types of intrusion detection do not fit neatly into these three new types. Misuse detection, for example, in some cases could indicate a definite intrusion and would then be Type 1, or it could indicate only symptoms of intrusions in other cases and would then be Type 2. The comparison of false positives of different methods of Misuse Detection is an invalid technique unless Type 1 methods are compared only with Type 1 methods and Type 2 methods are compared only with Type 2 methods. Anomaly detection, for example, would tend to be Type 2, but some anomalies could clearly indicate intrusions and would be Type 1. Type 1 and Type 2 methods of Anomaly Detection should be separated before making any comparisons. Likewise with intrusion detection labels based on activity, appearance, authentication analysis, behavior, knowledge, models, profiles, rules, signature, static analysis, statistics, and thresholds. These are still useful as descriptive terms, but they are not as useful in analyzing methods of determining whether or not an intrusion has occurred because they allow the comparisons of *apples* and *oranges* in numerous ways. The labels Type 1 and Type 2 give us more analytical information: either an intrusion has occurred or else only a symptom of an intrusion has occurred. Type 3 intrusions tell us that we should find out why an intrusion was not detected in the network traffic so that we can create new rules to find more intrusions in the future. Previously labeled types of intrusion detection do not give us as much analytical information as do types 1, 2, and 3.

Using this system, one can clearly state objectives of LLNID research in a new way which was previously only implied. The significance of given time period is apparent in the descriptive of these objectives because the objectives are stated in terms of progress from one time period to another time period. Here are specifics for LLNID research:

- *Type 3 NID Research*: Find ways of detecting intrusions that are currently not being detected, moving them up to type 2 or 1 intrusion detection.
- *Type 2 NID Research*: Improve Type 2 Intrusion Detection with the goal of moving it up to Type 1 Intrusion Detection.
- *Type 1 NID Research*: Improve Type 1 Intrusion Detection so that it is faster, uses fewer resources, and has fewer false positives.

Each of these types of research are necessary because finding new methods of intrusion detection is different than improving symptom detection which is different than making Type 1 Intrusion Detection more efficient. They are also complete because there are no other types of intrusion detection research in this context.

Table 1 summarizes the types discussed in this section. These are some ways of how researchers can use these types: research that compares false positive rates of Type 1 methods with false positive rates of Type 2 methods is not valid because Type 1 methods are not supposed to have false positives whereas Type 2 methods are expected to have false positives. Discounting Type 3 intrusion detection because of the amount of time taken may be irrelevant if otherwise the intrusion would not be found, at all. Proposing that intrusion prevention will replace intrusion detection is a false claim so long as types 2 and 3 intrusions continue to exist. Rather than disregarding Type 2 methods, research should attempt to fuse the results of Type 2 methods in order to move them up to Type 1.

IV. The LLNIDS Computational Model

A few number of researchers have described intrusion detection in limited mathematical ways, with [18][19], in the context of attack trees, and [20], in the context of game theory, being representative. Network Monitoring was formulated as a language recognition problem in [21].

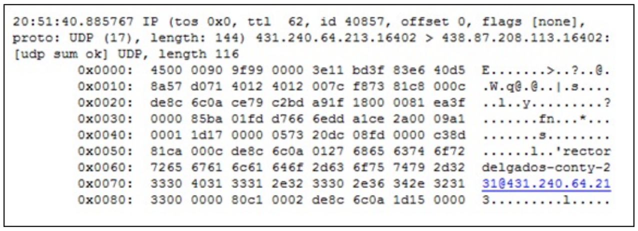
TABLE I
Summary of LLNID Types

	Type 1	Type 2	Type 3
Intrusion	This can be positively detected by LLNIDS	A symptom of this can be detected by LLNIDS	This is not detected by LLNIDS
Intrusion Detection	This positively detects an intrusion	This detects one or more symptoms (only) of an intrusion	An intrusion is not detected
Method	How to positively detect an intrusion	How to positively detect a symptom of an intrusion	An intrusion is not detected
Alert	This positively signifies an intrusion	This signifies a symptom of an intrusion	This does not occur
False Positive	An alert positively signifies an intrusion, but there is no intrusion	An alert signifies a symptom of an intrusion, but there is no intrusion	An alert does not occur
Research	Improve Type 1 Intrusion Detection, such as by increasing the speed of detection, using less resources, and having fewer false positives	Improve Type 2 Intrusion Detection so that it becomes Type 1 Intrusion Detection	Detect Type 3 intrusions so that they become Type 2 or Type 1

We propose Local Landline Network Intrusion Detection System (LLNIDS) Computational Model that covers intrusion detection data from packet analysis to sophisticated computational intelligent methods. This ID Math computational model begins with a transmission of digital network traffic and proceeds stepwise to higher concepts. The terminology for the input data changes depending upon the level of the concept. The lowest level concept in this research is the network transmission, which is a series of bits called a frame or a packet. Frame refers to a type of protocol, such as Media Access Control (MAC), which is used between two neighboring devices, where the series of bits are framed by a header at the start and a particular sequence of bits at the end. Packet refers to many types of protocols, such as Internet Message Control Protocol (ICMP), User Datagram Protocol (UDP), and Transmission Control Protocol (TCP). A packet is used for hops between numerous devices, such as Internet traffic. The length of the series of bits in a packet is often indicated at certain locations in the headers of the packets. A frame passes a packet between two neighboring devices, where another frame passes the same packet between the next two devices, and subsequent frames keep passing the packet forward until the journey of the packet is concluded. Since frames and packets are variable lengths, they are represented by a set of objects which represent the various elements of information inside the frame or packet.

A Transmission (T) consists of a set of objects (o) representing elements of information in that transmission.

$$T = \{o_1, o_2, o_3, \ldots, o_{n_T}\} \tag{1}$$



```
20:51:40.885767 IP (tos 0x0, ttl  62, id 40857, offset 0, flags [none],
proto: UDP (17), length: 144) 431.240.64.213.16402 > 438.87.208.113.16402:
[udp sum ok] UDP, length 116
        0x0000:  4500 0090 9f99 0000 3e11 bd3f 83e6 40d5  E.......>..?..@.
        0x0010:  8a57 d071 4012 4012 007c f873 81c8 000c  .W.q@.@..|.s....
        0x0020:  de8c 6c0a ce79 c2bd a91f 1800 0081 ea3f  ..l..y.........?
        0x0030:  0000 85ba 01fd d766 6edd a1ce 2a00 09a1  .......fn...*...
        0x0040:  0001 1d17 0000 0573 20dc 08fd 0000 c38d  .......s........
        0x0050:  81ca 000c de8c 6c0a 0127 6865 6374 6f72  ......l..'rector
        0x0060:  7265 6761 6c61 646f 2d63 6f75 7479 2d32  delgados-conty-2
        0x0070:  3330 4031 3331 2e32 3330 2e36 342e 3231  31@431.240.64.21
        0x0080:  3300 0000 80c1 0002 de8c 6c0a 1d15 0000  3.........l.....
```

Fig. 4. A Sample Packet

where $n_T \in \mathcal{N}$. Examples of objects in a transmission are the source MAC address, source IP address, source port, destination MAC address, destination IP address, destination port, the apparent direction of the traffic, protocols used, flags set, sequence numbers, checksums, type of service, time to live, fragmentation information, and the content being sent.

Figure 4 is a sample packet as displayed by tcpdump [22]. Header information extracted from the packet is displayed across the top. The leftmost column is the byte count in hexadecimal. The packet itself is displayed in hexadecimal in columns in the middle. Character representations of the hexadecimal code, when possible, are shown on the right. The packet is a transmission set, T, with variable length objects as elements. Example object elements for this set are the protocol, UDP, and the destination port, 16402, both of which have been extracted from the packet code.

If an intrusion occurs on a local landline, it occurs in one or more T, so LLNID means inspecting T's for intrusions. Not all of the available data in T has equal relevance to intrusion detection and the reduction of the amount of data is desirable in order to reduce the resources needed for analysis. This process has been called feature deduction [23], feature reduction [23], feature ranking [24], or feature selection [23]. The first feature selection must be done manually by a knowledge engineer, after that the features can be ranked and/or reduced computationally. Soft Computing methods often use data structures of n-tuple formats, such as one-dimensional arrays, sets, vectors, and/or points in space. Since sets can be used as a basis to describe these data structures, the next step in the computational model is to convert features of T into higher levels of sets which can be further manipulated for data analysis. The next set to be considered is an Event (E) which consists of a set of elements (e) obtained from the objects of T, and which changes the concept level from a transmission of objects to a set of elements:

$$E = \{e_1, e_2, e_3, \ldots, e_{n_E}\} \tag{2}$$

where $n_E \in \mathcal{N}$ and the following condition is also met:

$$\forall e_i \in E, 1 \leq i \leq n_E, e_i \in T \tag{3}$$

How to construct e_i from the objects of T is feature selection–elements should be selected which can detect intrusions. An example of possible elements for an event is the source IP address, the destination IP address, the source and destination ports, the protocol, and the size of a packet crossing the network.

TABLE II
A SAMPLE EVENT

UDP	231.240.64.213	238.87.208.113	16402

TABLE III
SAMPLE META-DATA

20100916	00:14:54	FW

Table 2 shows a sample event with the following elements: The protocol is UDP, the source IP address is 231.240.64.213, the destination IP address is 238.87.208.113, and the destination port is 16402. These elements were object elements in the sample transmission set shown above. The process of pulling data objects from a packet and saving them as Event elements is called parsing the data.

The next step is to add Meta-data (M), if appropriate, about the event consisting of meta-data elements (m):

$$M = \{m_1, m_2, m_3, \ldots, m_{n_M}\} \tag{4}$$

where $n_M \in \mathcal{N}$. Meta-data is data about data. In this context, it means data about the transmission that is not inside the transmission, itself. Examples of meta-data are the time when a packet crossed the network, the device which detected the packet, the alert level from the device, the direction the packet was travelling, and the reason the packet was detected. The concept level has changed from a set of elements to a set of meta-data about the set of elements.

Table 3 shows sample meta-data for an event. The meta-data in this table is the date, 20100916, and the time, 00:14:54, at which an appliance detected the transmission, and a label for the appliance that detected the packet, FW.

A Record (R) of the event includes both the event, itself, plus the meta-data:

$$R = M \cup E \tag{5}$$

An example of a record is an entry in a normalized firewall log. The concept level has changed from a set of meta-data to a set that includes both the elements and meta-data about those elements. In practice, the meta-data typically occurs in R before the elements to which the meta-data refers.

Table 4 is a sample record, which consists of meta-data and elements from the previous examples for M and E. Before proceeding to the next step, the attributes of R for a given analysis should be in a fixed order because they can later become coordinates in a location vector. Processing the data into fixed orders of attributes is called normalizing the data.

A Log (L) of records is a partially ordered set:

$$L = \{R_i\}_{i \in \mathcal{N}} \tag{6}$$

An example of a log is a file containing normalized firewall log entries. An infinite-like log could be live streaming data.

TABLE IV
A SAMPLE RECORD

20100916	00:14:54	FW	UDP	231.240.64.213	238.87.208.113	16402

TABLE V
A SAMPLE LOG

20100916	00:14:54	FW	UDP	231.240.64.213	238.87.208.113	16402
20100916	00:14:56	FW	TCP	216.162.156.85	198.18.147.222	40833
20100916	11:14:57	FW	ICMP	90.29.214.20	198.18.147.221	41170

Table 5 shows a sample log. It is like the sample record, above, except there are three entries instead of just one entry. The concept level has changed from a set of meta-data and elements to a collection of sets of meta-data and elements. L can be considered to be a set of vectors; L can also be considered to be a matrix. If L is a text file, each line of the file is one location vector and the entire file is a matrix, changing the concept level to a matrix.

If the features have been selected successfully, an intrusion, or one or more symptoms of it, should be able to be detectable in L. Therefore, LLNIDS intrusions and intrusion detection can be defined in terms of R and L. Let $\mathscr{R}$ be the universal set of R and let I_1 represent a set of $\mathscr{R}$ that describe a Type 1 Intrusion. Then I_1 is the set:

$$I_1 = \{R | R \in \mathscr{R}, R \ involves \ a \ Type \ 1 \ Intrusion \ \} \tag{7}$$

Formula 7 formulates a Type 1 Intrusion. Examples of Type 1 intrusions are a Ping of Death and a get request to a known malicious web site. These intrusions can potentially be prevented. I_1 has the same attributes as L in that it can be considered to be a set of location vectors or it can be considered to be a matrix. As matrices, the number of columns in I_1 and L for an analysis must be the same, but the number of rows in I_1 and L can be different. For reference below, let $\mathscr{I}_1$ be the universal set of all Type 1 intrusions. The concept level for $\mathscr{I}_1$ has changed from a matrix to a set of matrices. That was about intrusions. Now here is the function for Type 1 Intrusion Detection, I_1^D:

$$I_1^D(L) = \begin{cases} \text{TRUE}, & \exists I_1 \in \mathscr{I}_1 : I_1 \subseteq L \\ \text{FALSE}, & otherwise \end{cases} \tag{8}$$

Formula 8 is the function for Type 1 Intrusion Detection, which returns True if an intrusion has been detected, otherwise it returns False. Next is Type 2 intrusions and intrusion detection. In most cases, one or more events occur which makes the security technician suspicious that an intrusion has occurred, but more investigation is necessary in order to reach a conclusion. This scenario, which is Type 2 Intrusion Detection, is similar to a patient going to a physician, who looks for symptoms and then makes a decision about whether or not the patient has a medical problem. The security technician also looks for symptoms and then makes a decision about whether or not an intrusion has occurred. Let $\mathscr{R}$ be the universal set of R and let I_2 represent a set of R that describes one or more symptoms of a Type 2 Intrusion. Then I_2 is the set:

$$I_2 = \{R | R \in \mathscr{R}, R \ involves \ a \ Type \ 2 \ Intrusion \ \} \tag{9}$$

Formula 9 formulates a Type 2 Intrusion. Let $\mathscr{R}_2$ be the universal set of all Type 2 intrusions. Now here is a formula for Type 2 Intrusion Detection, I_2^D:

$$I_2^D(L) = \begin{cases} \text{TRUE}, & \exists I_2 \in \mathscr{I}_2 : I_2 \subseteq L \\ \text{FALSE}, & otherwise \end{cases} \tag{10}$$

The $I_2^D(L)$ function returns True if a symptom of an intrusion has been detected; otherwise it returns False. Possible examples of Type 2 intrusions are the following: The set of records consisting of a single local source IP address and numerous unique destination addresses all with a destination port of 445; the set of records consisting of a local IP address sending numerous e-mails during non-working hours; and, the set of records consisting of high volumes of UDP traffic on high destination ports to a single local IP address matching criteria set by a Self-Organizing Map. Like a cough does not necessarily indicate a cold, the detection of an intrusion symptom does not always indicate an intrusion.

That was Type 2 intrusions and intrusion detection. Next is Type 3 intrusions, which are not detected in a given time period. Let $\mathscr{R}$ be the universal set of R and let I_3 represent a set of R that describes a Type 3 Intrusion. Then I_3 is the set:

$$I_3 = \{R | R \in \mathscr{R}, R \ involves \ a \ Type \ 3 \ Intrusion \ \} \tag{11}$$

As a summary, compare these three types of intrusion detection in a medical context to typhoid fever, which is spread by infected feces. Type 1 intrusion detection (prevention) is to wash one's hands after using the toilet; Type 2 intrusion detection is to recognize the symptoms, such as fever, stomach ache, and diarrhea; Type 3 detection is represented by Typhoid Mary, who had no readily recognizable symptoms.

The next step involves changing the data formats from R and L into forms which can be directly manipulated by analysis software. (Packet analysis can already occur directly on T.) This involves converting records into vectors and logs into matrices. This conversion is straightforward with a Detailed Input Data Vector, V_D, which starts as a set and is then used later as a location vector:

$$V_D \subseteq R \tag{12}$$

More feature reduction can occur at this step. If the order of each element in the set is fixed, i.e., if the order of the attributes of the set are fixed, then the set can become a location vector. An example of V_D as a set is $\{$1280093999, 10.3.4.10, 10.3.4.12, 445, TCP$\}$ which could indicate a time stamp in seconds, a source IP address, a destination IP address, a destination port, and a protocol. Converting IP addresses to numerical formats, and assigning a numerical label to TCP, the same example of V_D as a location vector could be (1280093999, 167969802, 167969804, 445, 6).

Aggregate elements are also possible for a given time period, such as aggregate data for each local IP address for a day. Examples of such aggregate elements are the total number of R for the local IP address, the count of unique source IP addresses communicating with the local IP address, and the percentage of TCP network traffic for the local IP address. Many other types of aggregate elements are possible. These aggregate elements can be converted to an Aggregate Input Data Vector, V_A, with f being an aggregation function:

$$V_A = \{f_1(L), f_2(L), f_3(L), \ldots, f_{n_V}(L)\} \tag{13}$$

where $n_V \in \mathcal{N}$. Again, the order of the attributes of the set are fixed so that the set can become a location vector. An example of V_A as a set is $\{20100725, 428, 10.3.4.10, 48, 0.89\}$ which could indicate that on 7/25/2010 428 unique source IP addresses attempted to contact destination IP address 10.3.4.10 on 48 unique destination ports with the TCP protocol being used 89 percent of the time. The date and IP address become a label for the location vector when the location vector is created. From the same example above, the location vector for IP address 10.3.4.10 on 7/25/2010 is (428, 48, 0.89).

Both of these types of sets/vectors can be generalized as a General Input Data Vector, V:

$$V = V_D \ or \ V = V_A \tag{14}$$

The next concept level is to generalize V so that it can be used as input to a wide variety of Soft Computer and other methods. The generalized elements of V are be represented by e. V is an n-tuple of real numbers which can be perceived, depending upon how it is intended as being used, as being a set, a location vector, or a matrix:

$$Set : V = \{e_1, e_2, e_3, \ldots, e_{n_V}\} \tag{15}$$

$$Vector : V = (e_1, e_2, e_3, \ldots, e_{n_V}) \tag{16}$$

$$Matrix : V = [e_1 \ e_2 \ e_3 \ \ldots \ e_{n_V}] \tag{17}$$

where $n_V \in \mathcal{N}$. For example, if the elements of V are an n-tuple of the real numbers 0.6, 0.5, 0.4, 0.3, 0.2, and 0.1, then V can be perceived as being a set, a vector or a matrix:

$$Set : V = \{0.6, 0.5, 0.4, 0.3, 0.2, 0.1\} \tag{18}$$

$$Vector : V = (0.6, 0.5, 0.4, 0.3, 0.2, 0.1) \tag{19}$$

$$Matrix : V = [0.6 \ 0.5 \ 0.4 \ 0.3 \ 0.2 \ 0.1] \tag{20}$$

An Input Data Matrix, D, is a collection of similar types of V. Here D is represented as a set of V:

$$D = \{V_1, V_2, V_3, \ldots, V_{n_D}\} \tag{21}$$

where $n_D \in \mathcal{N}$. D is on the same concept level as L. Both D and L can be considered to be sets of location vectors or a matrix. Here is how D can be represented as a matrix:

$$D = \begin{bmatrix} V_{1,1} & \cdots & V_{1,n_V} \\ \vdots & \ddots & \vdots \\ V_{n_D,1} & \cdots & V_{n_D,n_V} \end{bmatrix} \tag{22}$$

where $n_D \in \mathcal{N}$ and $n_V \in \mathcal{N}$.

For example, given these three location vectors, each represented as a matrix,

$$V_1 = [0.6 \ 0.5 \ 0.4 \ 0.3 \ 0.2 \ 0.1] \tag{23}$$

$$V_2 = [0.1 \ 0.2 \ 0.3 \ 0.4 \ 0.5 \ 0.6] \tag{24}$$

$$V_3 = [0.9 \ 0.8 \ 0.7 \ 0.6 \ 0.5 \ 0.4] \tag{25}$$

D would be represented this way as a matrix:

$$D = \begin{bmatrix} 0.6 & 0.5 & 0.4 & 0.3 & 0.2 & 0.1 \\ 0.1 & 0.2 & 0.3 & 0.4 & 0.5 & 0.6 \\ 0.9 & 0.8 & 0.7 & 0.6 & 0.5 & 0.4 \end{bmatrix} \tag{26}$$

D_D can refer to an Input Data Matrix consisting of V_D and D_A can refer to an Input Data Matrix consisting of V_A. D can also be one of these three types:

1) D_{Train} refers to a data set which is used to train the software intelligence
2) D_{Test} refers to a data set which is used to test the software intelligence
3) D_{Real} refers to feral data.

D can be used in virtually an infinite variety of analysis methods, from spreadsheet methods to statistics and data mining, to machine learning methods. For example, D_{Train} can be used by clustering software which, after testing, would then classify D_{Real} for intrusion detection.

The ID Math Method more accurately defines information security concepts and scientifically ties components of information security together with structured and uniform data structures. The LLNIDS can be extended to describe existing and potential methodologies of analysis methods including statistics, data mining, AIS, NeuroFuzzy, Swarm Intelligence, and SOM, as well as Bayes Theory, Decision Trees, Dempster-Shafer Theory, Evolutionary Computing, Hidden Markov Models, and many other types of analysis.

V. Conclusion

This paper provided a new way of looking at network intrusion detection research including intrusion detection types that are necessary, complete, and mutually exclusive to aid in the fair comparison of intrusion detection methods and to aid in focusing research in this area. This paper also provided a methodical description of intrusion detection data and how this data is manipulated and perceived from packet analysis to sophisticated computational intelligence methods. This new ID Math provides a methodological archetype from which to move forth. Future work in intrusion detection research should leverage these intrusion detection types and this computational model for better descriptions of the problem sets and for presenting solutions to intrusion detection.

References

[1] Langin, C. L. A SOM+ Diagnostic System for Network Intrusion Detection. Ph.D. Dissertation, Southern Illinois University Carbondale (2011)

[2] Amoroso, E.: Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps, and Response. Intrusion.Net Books (1999)

[3] Denning, D.: An Intrusion-Detection Model. IEEE Transactions on Software Engineering 13(2), 118-131 (1986)

[4] Young, C.: Taxonomy of Computer Virus Defense Mechanisms. In : The 10th National Computer Security Conference Proceedings (1987)

[5] Lunt, T.: Automated Audit Trail Analysis and Intrusion Detection: A Survey. In : Proceedings of the 11th National Computer Security Conference, Baltimore, pp.65-73 (1988)

[6] Lunt, T.: A Survey of Intrusion Detection Techniques. Computers and Security 12, 405-418 (1993)

[7] Vaccaro, H., Liepins, G.: Detection of Anomalous Computer Session Activity. In : Proceedings of the 1989 IEEE Symposium on Security and Privacy (1989)

[8] Helman, P., Liepins, G., Richards, W.: Foundations of Intrusion Detection. In : Proceedings of the IEEE Computer Security Foundations Workshop V (1992)

[9] Denault, M., Gritzalis, D., Karagiannis, D., Spirakis, P.: Intrusion Detection: Approach and Performance Issues of the SECURENET System. Computers and Security 13(6), 495-507 (1994)

[10] Forrest, S., Allen, L., Perelson, A., Cherukuri, R.: Self-Nonself Discrimination in a Computer. In : Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy, Los Alamos, CA (1994)

[11] Crosbie, M., Spafford, G.: Defending A Computer System Using Autonomous Agents., COAST Laboratory, Department of Computer Science, Purdue University, West Lafayette, Indiana, USA (1994)

[12] Kumar, S., Spafford, E.: An Application of Pattern Matching in Intrusion Detection., Purdue University (1994)

[13] Ilgun, K., Kemmerer, R., Porras, P.: State Transition Analysis: A Rule-Based Intrusion Detection Approach. IEEE Transactions on Software Engineering 21(3), 181-199 (March 1995)

[14] Esmaili, M., Safavi-Naini, R., Pieprzyk, J.: Evidential Reasoning in Network Intrusion Detection Systems. In : Proceedings of the First Australasian Conference on Information Security and Privacy, pp.253-265 (1996)

[15] Debar, H., Dacier, M., Wespi, A.: Towards a Taxonomy of Intrusion-Detection Systems. Computer Networks 31, 805-822 (1999)

[16] Bace, R.: Intrusion Detection. MacMillan Technical Publishing (2000)

[17] Marin-Blazquez, J., Perez, G.: Intrusion Detection Using a Linguistic Hedged Fuzzy-XCS Classifier System. Soft Computing – A Fusion of Foundations, Methodologies, and Applications 13(3), 273-290 (2008)

[18] Wang, L., Noel, S., et al. Minimum-Cost Network Hardening Using Attack Graphs. Computer Communications 29(18), 3812-3824 (2006)

[19] Dewri, R., Poolsappasit, N., et al. Optimal Security Hardening Using Multi-objective Optimization on Attack Tree Models of Networks. 14th ACM Conference on Computer and Communications Security (2007)

[20] Chen, L. and Leneutre,J. A Game Theoretical Framework on Intrusion Detection in Heterogeneous Networks." IEEE Transactions on Information Forensics and Security 4(2), 165-178 (2009)

[21] Bhargavan, K., Chandra, S., McCann, Peter J. and Gunter, C. A.What packets may come: automata for network monitoring. Proceedings of the 28th ACM SIGPLAN-SIGACT symposium on Principles of programming languages (2001)

[22] Tcpdump/Libpcap: Tcpdump/Libpcap Public Repository. In: Tcpdump.org. Available at: http://www.tcpdump.org/

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

[24] Mukkamala, S., Sung, A.: Identifying Significant Features for Network Forensics Analysis Using Artificial Intelligent Techniques. International Journal on Digital Evidence (IJDE) 1(4) (2003)