Network Intrusion and Fault Detection: A Statistical Anomaly Approach

Constantine Manikopoulos and Symeon Papavassiliou, New Jersey Institute of Technology

ABSTRACT

With the advent and explosive growth of the global Internet and electronic commerce environments, adaptive/automatic network/service intrusion and anomaly detection in wide area data networks and e-commerce infrastructures is fast gaining critical research and practical importance. In this article we present and demonstrate the use of a general-purpose hierarchical multitier multiwindow statistical anomaly detection technology and system that operates automatically, adaptively, and proactively, and can be applied to various networking technologies, including both wired and wireless ad hoc networks. Our method uses statistical models and multivariate classifiers to detect anomalous network conditions. Some numerical results are also presented that demonstrate that our proposed methodology can reliably detect attacks with traffic anomaly intensity as low as 3–5 percent of the typical background traffic intensity, thus promising to generate an effective early warning.

INTRODUCTION

There currently exist multiple threats and vulnerabilities in the security of computer systems and networks against attacks. Along with the explosive growth of the Internet and the continued dramatic increase in all wireless services, the number and impact of attacks has been increasing. This is evidenced by recent well-publicized denial of service attacks against several prominent Web portals as well as many hushed over occurrences. The number of computer systems and their vulnerability have been rising, while the level of sophistication and knowledge required to carry out an attack has been decreasing, as much technical attack know-how is readily available on Web sites all over the world [1].

Recent advances in encryption, public key exchange, digital signatures, and the development of related standards have set a foundation for the flourishing of electronic commerce. However, security on a network goes way beyond encryption of data. It must include the security of computer systems and networks, at all levels, top to bottom. It is imperative to arm the network systems and elements with well designed, comprehensive, and integrated attack defeating policies and devices. However, foolproof prevention of attacks is challenging because at best the defensive system and application software may also contain unknown weaknesses and bugs. Thus, *early warning systems* (i.e., *intrusion detection systems* or IDSs), as components of a comprehensive security system, are required in order to prime the execution of countermeasures. The technique of statistical anomaly and intrusion detection is the focal point of this article.

METHODS OF COMPUTER NETWORK ATTACKS: THE INCREASED CHALLENGE

Attacks may be denial of service (DoS), unauthorized use of a system, probing of a system to gather information, a physical attack against computer hardware, worms or viruses, and so on. The possible types of computer attacks include:

- Attacks that deny someone else access to some services or resources a system provides
- Attacks that allow an intruder to operate on a system with unauthorized privileges
- Attempts to probe a system to find potential weaknesses

All these and other attacks have been gaining in sophistication and power to harm. Attacks are increasingly automated, so now the attack tools may initiate new attack cycles by themselves with no person involved. Distributed attack tools are capable of coordinating use of numerous attack platforms and scripts spread out through the Internet, thus launching truly devastating DoS attacks. These tools utilize highly functional coordinating modern communications protocols like Internet relay chat (IRC) and instant messaging (IM). Moreover, the attack methods seem

0163-6804/02/$17.00 © 2002 IEEE

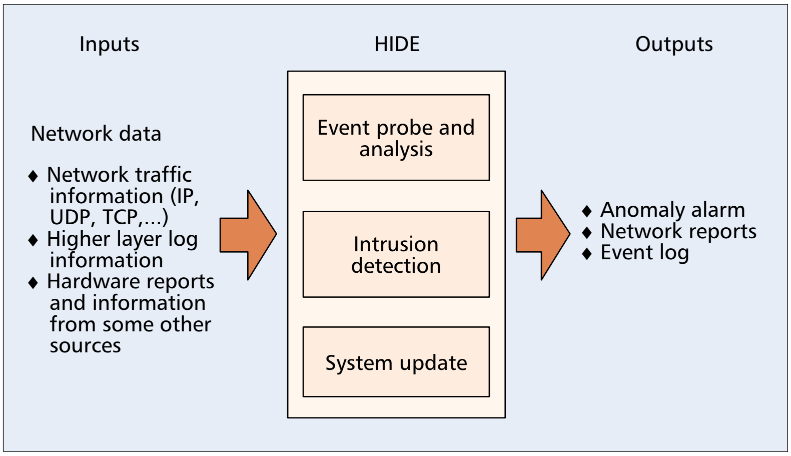
increasingly capable of considerable stealth that aims to evade recognition of their characteristic signature. As part of this masking strategy, they often use dynamic variation of methods and activities with predetermined or random patterns.

Types of Intrusion Detection Systems

An IDS may be of the network/node (NID) or host (HID) type. That is, the IDS may gather information from a computer (i.e., system audit logs and file states) or network of computers (i.e., packets passing on the wire between hosts or arriving at a particular host) in attempting to detect intruders or system abuse. Thus, network or node intrusion detection (NID) systems, typically referred to as "packet sniffers," intercept packets of various physical manifestations and protocols. In the past, NIDs have been challenged by switched, encrypted, and high-speed networks (> 100 Mb/s). However, recently these limitations have been largely overcome. For example, there are now NID systems that may be installed on switches as well as monitor at gigabit speeds. After capturing a packet, some NID devices will simply compare the packet to a signature database of known attacks ("fingerprints"), while others seek to detect "anomalous" packet activity. Thus, the NID techniques can be partitioned into two complementary trends: *misuse detection* and *anomaly detection*. Misuse detection systems [2] model known attacks and scan the system for occurrences of these patterns. *Anomaly detection systems* [3] flag intrusions by observing significant deviations from typical or expected behavior of systems or users. HIDs monitor and detect user and system activity and attacks on a given host, so these systems are best suited to counter internal (intranetwork) threats because they focus on monitoring user actions and file accesses on the host.

Intrusion detection research and development dates to the early 1980s starting with Anderson's paper [4] that introduced the concept of computer threats and detection of misuse, as applied to the host IDS, or HID. Dr. Denning's work [5] then introduced the first model for intrusion detection, followed with the Haystack project IDS and later the distributed IDS (DIDS). In the early 1990s, NID was introduced [6], along with early versions of commercial and government IDS systems. However, most IDS work is rather recent; many research systems and commercial products appeared in the late 1990s, following the growth of the Internet [7].

Moreover, the past few years have witnessed much progress in all aspects of fault detection, including path failure detection, anomaly detection in the Ethernet, anomaly and performance change detection in small networks, network alarm correlation, and real-time trouble ticketing information in a transaction-oriented communication network [8]. The main objective of fault detection is to identify network exceptional conditions, such as performance and utilization degradations. Under favorable conditions that can be accomplished by design, network fault detection can anticipate the occurrence of high-level service failures and compromises, and thereby increase the chances for proactive fault correction before service failures set in.


Inputs
HIDE
Outputs
Network data
♦ Network traffic information (IP, UDP, TCP,...)
♦ Higher layer log information
♦ Hardware reports and information from some other sources
Event probe and analysis
Intrusion detection
System update
♦ Anomaly alarm
♦ Network reports
♦ Event log

■ **Figure 1.** *Inputs and outputs of HIDE.*

Statistical Anomaly Detection

Statistical modeling and neural networks have been applied in building anomaly network intrusion and fault detection systems. A system that identifies intrusions using packet filtering and neural networks was introduced in [9]. The authors in [4] studied the employment of neural networks to detect anomalous and unknown intrusions against a software system. In [10], Kolmogorov-Smirnov statistics was used to model and detect DoS as well as probing attacks. In [11], we proposed the prototype of a Hierarchical Intrusion Detection system (HIDE), while in [12] we introduced the Generalized Anomaly and Fault Threshold system (GAFT); these systems use statistical preprocessing and neural network classification to detect network attacks and faults.

System Architecture

The input-output data types of HIDE are shown in Fig. 1. HIDE is a hierarchical multitier system that may run in a distributed as well as standalone fashion. It gathers data from network traffic, system logs, and hardware reports; statistically processes and analyzes the information; detects abnormal activity patterns based on the reference models, which correspond to the expected activities of typical users, for each parameter individually, or in combined groups using neural network classification; generates the system alarms and event reports; and finally, HIDE updates the system profiles based on the newly observed network patterns and system output. GAFT behaves in a similar fashion but monitors a different mix of network parameters.

HIDE is, in general, a distributed application, deployed hierarchically over several tiers, with each tier containing several intrusion/fault detection agents (IDAs). IDAs are security components that monitor the activities of a host

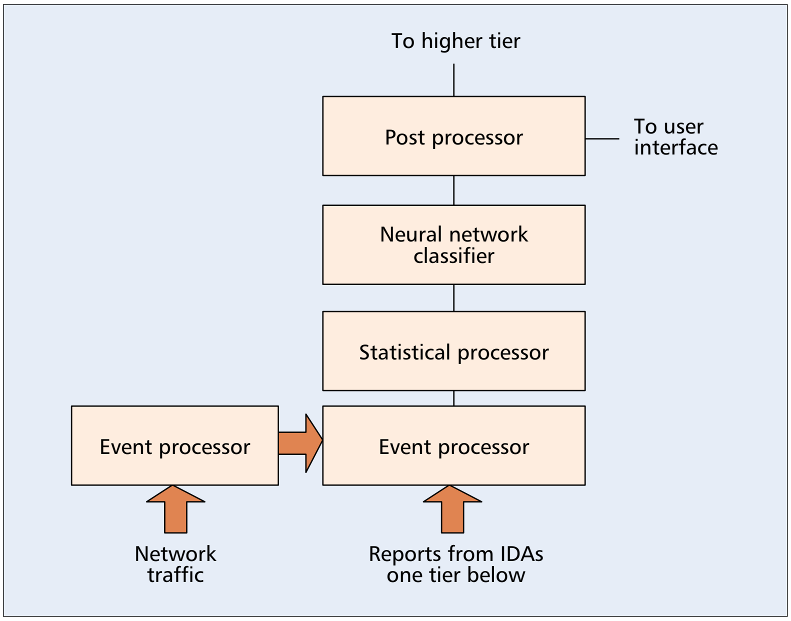


Figure 2. *An intrusion detection agent.*

	Scenario 1	Scenario 2
TCP background traffic (Mb/s)	1.05	1.05
UDP background traffic (Mb/s)	1.08	6.82
Attack traffic (Mb/s)	1.8	1.8

Table 1. *Traffic loads of two scenarios.*

or the network to which they are attached. Different tiers correspond to different network scopes that are monitored by the agents affiliated with them.

Each IDA consists of the following components: the probe, the event preprocessor, the statistical processor, the neural network classifier, and the post processor. A diagram of the IDA is given in Fig. 2. The *probe* collects the network traffic of a host or network, abstracts the traffic into a set of statistical variables to reflect the network status, and periodically generates reports to the event preprocessor. The event preprocessor 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 reference models of typical network activities, compares the reports from the event preprocessor to the reference models, and forms a stimulus vector to feed into the neural network classifiers. The *neural network classifier* analyzes the stimulus vector from the statistical model to decide whether the network traffic is normal or not. The *post processor* generates reports for the agents at higher tiers. At the same time, it may display the local results through a user interface.

Because network traffic is not stationary and network-based attacks may have different time duration, we need to monitor network traffic with different time windows. Thus, we adopted a nested layer-window model with each window corresponding to a geometrically increasing detection time slice (typically by a factor of 4). The newly arrived events will first be stored in the event buffer of observation window layer 1. The stored events will be compared to the reference model of that layer; the similarity results are then fed into a neural network classifier to detect the network status during that time window. The event buffer will be emptied once it becomes full, and the stored events will be averaged and forwarded to the event buffer of layer 2. The same process will be repeated recursively until data arrives at the top level where the events will simply be dropped after processing.

The Statistical Model

An *activity profile* characterizes the behavior of a given monitored parameter thereby serving as a description of normal activity for its respective subject. Observed behavior may be described using a statistical metric and model.

The period of observation may be a fixed interval of time (sec, minute, etc.), or the time between two audit-related events (i.e., between login and logout, file open and file close, etc.). Observations (sample points) x_i of x are used together with a statistical model to determine whether a new observation is abnormal. It is preferred that the statistical model make no assumptions about the underlying distribution of x; in such systems, all knowledge about x is obtained from observations. In our work, user profiles are represented by a number of probability density functions. Let S be the sample space of a random variable and events E_1, E_2, …, E_k a mutually exclusive partition of S. Assume that p_i is the expected probability of the occurrence of event E_i, p'_i represents the actual probability of the occurrence of E_i during a time interval, and N is the total number of occurrences.

Most earlier IDS systems simply measured the means and variances of the monitored variables and detected whether certain thresholds were exceeded; in nonstationary systems that often do not follow the normal distribution, such systems generate incorrect decisions. To overcome some of these problems, NIDES [3] used a χ^2-like statistical test to determine the similarity Q between the expected and actual distributions. However, in real applications empirically Q may not follow χ^2 distributions; therefore, NIDES solved this problem by building an empirical probability distribution for Q that is updated daily in real-time operation.

In our HIDE system, since we are using neural networks to classify patterns and identify possible intrusions, we are not as concerned with the actual distribution of Q; it is expected that the neural net will learn it from examples.

The similarity metrics we have investigated are based on the Kolmogrov-Smirnov (K-S) test. In using the K-S test, the reference models and observed system activities are represented by cumulative density functions (CDFs). What makes the K-S statistic powerful is the fact that it is distribution-independent and thus free of explicitly or implicitly assuming that the distribution is the normal one. This is a very important property for network monitoring, where very little may be known about the underlying distribution for either typical or anomalous traffic.

There are several variants on the K-S tests that we have studied and compared. Among them are the Anderson-Darling statistic that calculates a weighted K-S measure, and Kuiper's statistic. The similarity-measuring metric we are using is shown below:

$$Q = f(N)\cdot\left[\sum_{i=1}^{k}\left|p_i' - p_i\right| + \max_{i=1}^{k}\left(\left|p_i' - p_i\right|\right)\right]$$

where $f(N)$ is a function that takes into account the total number of occurrences during the corresponding time window.

Besides similarity measurements, we also designed an algorithm for real-time updating of the reference model. Let $\bar{p}_{old}$ be the reference model before updating, $\bar{p}_{new}$ the reference model after updating, and $\bar{p}_{obs}$ the observed user activity within a time window. The formula to update the reference model is

$$\bar{p}_{new} = s \times \alpha \times \bar{p}_{obs} + (1 - s \times \alpha) \times \bar{p}_{old},$$

where α is the predefined adaptation rate and s the value generated by the output of the neural network. Assume that the output of neural network t is a continuous variable between –1 and 1, where –1 means intrusion with absolute certainty and 1 means no intrusion, again with complete confidence. In between, the values of t indicate proportional levels of certainty. The function of calculating S is:

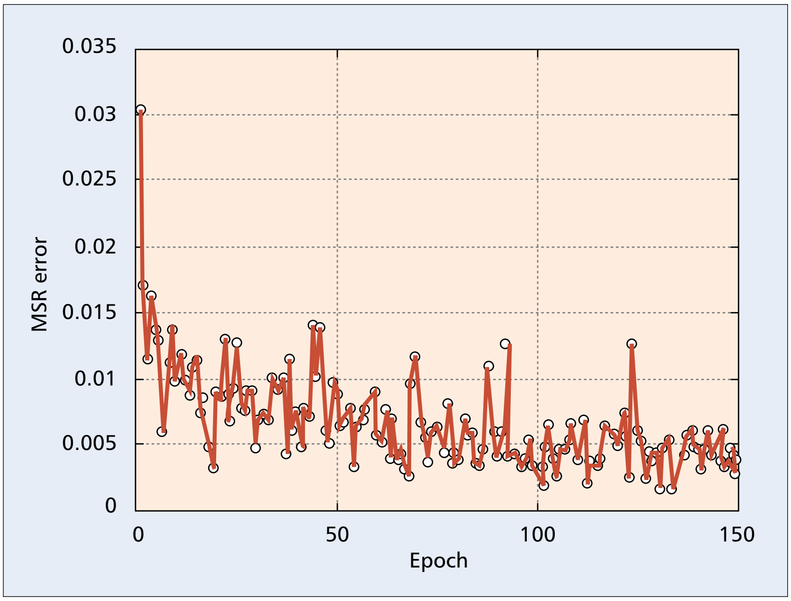
$$s = \begin{cases} t, \text{ if } t \geq 0 \\ 0, \text{ otherwise} \end{cases}$$

Through the above relations, we ensured that the reference model would be updated actively for normal traffic while kept unchanged when attacks occurred. The attack events will be diverted and stored for us as attack scripts in neural network learning.

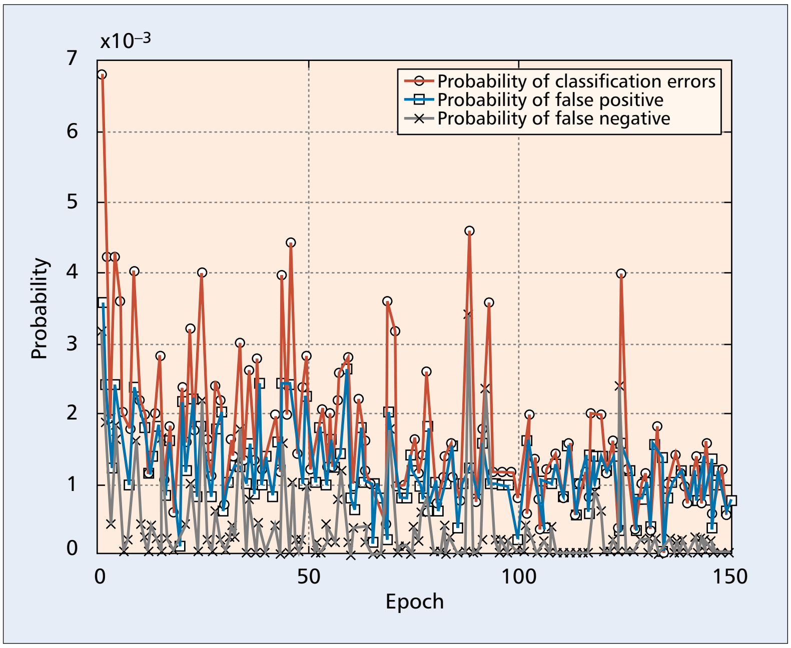
Neural Network Classification

As mentioned above, many types of distance metrics may be employed by the statistical analyzer to generate similarity measures. The output of the statistical processor is a k-dimensional ordered vector whose components are the similarity values of the measured PDFs to the reference PDFs of the k monitored network or system parameters collected during an observation window. This k-dimensional vector represents an evaluation of the status of the network or system during that observation time. Subsequently, this status vector gets presented to the multivariate classifier for a classification result. Clearly, the efficacy of this classifier is crucial to the success of the endeavor.

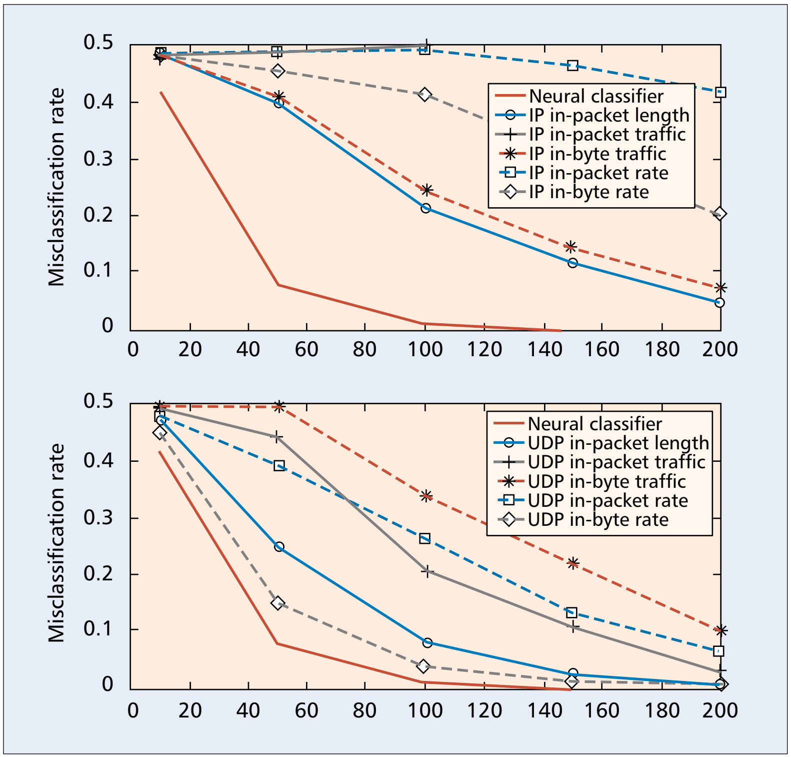
In this article we chose the neural network classifiers. The k-dimensional ordered vector produced by the statistical processor to the classifier may be thought of as an input pattern to the classifier. Neural networks are widely considered an effective approach to handling and classifying patterns. However, the sometimes high computation requirements as well as long training cycles have hindered their applications. This is due to the fact that such neural networks often intake a large number of inputs (several tens or hundreds) and utilize many (several dozen or more) computational neurons in one or several hidden layer(s). In other words, in such cases the neural networks are expected to carry out by themselves the majority of the computational work that leads to a decision. In contrast, our system uses powerful and comprehensive statistical preprocessing before presenting the vector of computed similarity values to the classifier for a decision and thus can perform effectively by deploying small and "lean" neural network classifiers, with only the minimum of inputs, a handful of computational neurons, and



■ **Figure 3.** *MSR as the training epochs increase.*

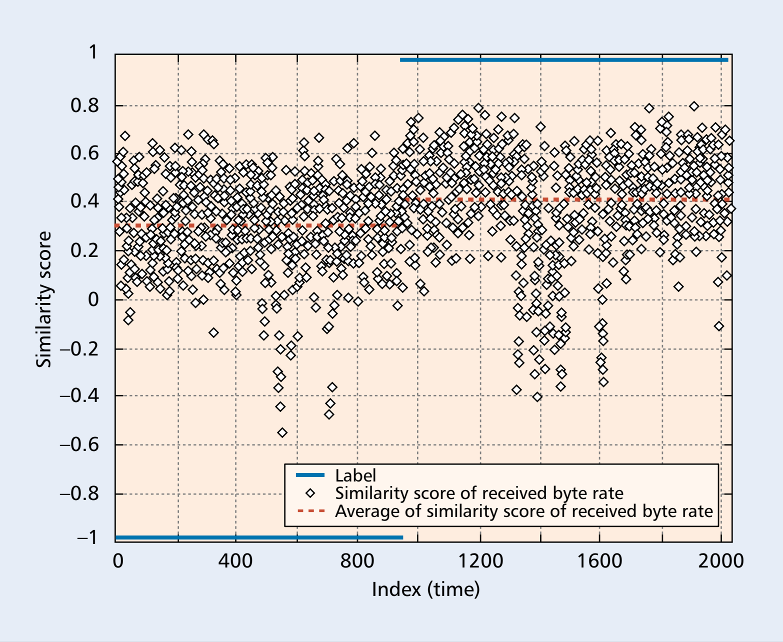


■ **Figure 4.** *Misclassification rates as epochs increase.*



■ **Figure 5.** *Results of single vs. multiple experiments: 2 Mb/s background.*

a correspondingly small total number of weights. For about a dozen simultaneously monitored parameters leading to an equal number of inputs, as is the case here, the resulting neural net can compute a classification decision using only a few hundred clock cycles; thus, the trained neural net classifier can definitely be part of real-time operation. Moreover, this neural net converges within less than 100 training epochs requiring at worst a few thousand clock cycles per training sample. Therefore, the computational demands for the training operation of our system are modest as well, and it is feasible to carry them out automatically, using either off-peak times or other schemes, including real-time operation.



■ **Figure 6.** *Similarity score of received byte rate at the IP layer.*

In [11], we studied the performances of five different types of neural networks: perceptron, back propagation (BP), perceptron-back propagation hybrid (PBH), radial-based function (RBF), and fuzzy ARTMAP. Our experimental results showed that the BP and PBH networks had stronger classification. Following this conclusion, we are using a BP network with two hidden neurons. This network is very small and efficient, but still has very good classification capability.

Performance Evaluation and Results

The experimental test results to date have emphasized DoS attacks and faults due to their prominent role in computer networks, in both wired and wireless ad hoc networks. These include UDP, ICMP, SYN, and remote-to-local (R2L) attacks as well as UDP small packet flooding, UDP broadcast flooding, IP small packets flood (soft), Ethernet runt flood, and Ethernet broadcast storm faults in simulation as well as in the DARPA 1998 and 1999 corpus of network data at Massachusetts Institute of Technology (MIT)/Lincoln Laboratories.

In order to demonstrate the efficiency and robustness of HIDE, along with its general-purpose applicability in networking environments with different characteristics and topologies, in this section we present some experiments and the corresponding numerical results we obtained applying our proposed system in two different networking environments:

- A wired conventional network with the presence of a UDP flood attack
- A wireless ad hoc network

UDP Flooding Attack Example

In this section we present our simulation approach and the results in applying the HIDE statistical models using PBH neural network classification to detect UDP flooding attacks. The experimental testbed we built using OPNET consists of three 10BaseX LANs interconnected by two routers. For each simulation scenario we collected 10,000 records of network traffic and trained the system for 150 epochs.

We ran two independent scenarios with different traffic loads and characteristics, shown in Table 1. The mean squared root errors (MSR) and misclassification rates (sum of both false positive and false negative rates) of the outputs of scenario 1 are shown in Figs. 3 and 4, respectively. It is important to note from Fig. 3 that the MSR decreases very fast after only the first few epochs, reaching satisfactory convergence levels (~ 0.05) within the first 10 epochs or so. The misclassification rates drop correspondingly fast as well to very low values. The results for scenario 2 are similar. Thus, the system is characterized by fast convergence and low misclassification rates. These features are especially desirable for network intrusion detection systems, which need real-time monitoring and online training.

Single vs. Combined Parameters

In this section the best possible detection rates using single parameters are calculated by optimum thresholding algorithms. Then these results are compared with those of neural classification of the combined parameters to illustrate the performance differences of the two approaches.

Ten different traffic parameters are monitored in the current experiments with UDP flooding attacks: *IP in-packet length*, *IP in-packet traffic*, *IP in-byte traffic*, *IP in-packet rate*, *IP in-byte rate*, *UDP in-packet length*, *UDP in-packet traffic*, *UDP in-byte traffic*, *UDP in-packet rate*, and *UDP in-byte rate*.

The misclassification rates associated with the 10 parameters, grouped by protocols, are plotted vs. attack intensities in Fig. 5 for 2 Mb/s background traffic. The misclassification rates using the neural network classifier are also plotted together with the parameters for performance comparison purposes. From this figure, we can see that the performance of combined parameter classification using a neural network classifier is consistently better than single-optimum-threshold parameter detection in all attack scenarios. The misclassification rates of the best single parameter, UDP in-byte rate, is close to those of combined parameter neural net classifier rates when attack levels are high — this happens to be a very effective parameter — but at low attack levels, the performance difference is noticeable. Thus, we may conclude that parameter combining is powerful in achieving effective and robust classification.

Ad Hoc Wireless Experiments

In order to demonstrate the applicability of our proposed approach as a general-purpose anomaly detection tool, we tested it in a dynamic mobile wireless ad hoc network. Due to the characteristics of open medium, dynamic changing topology, cooperative algorithms, and lack of a centralized monitoring and management point, wireless ad hoc networks are particularly vulnerable. Mobile nodes are autonomous units that are capable of roaming independently. This means that nodes with inadequate physical protection may be captured, compromised, or hijacked. Wireless ad hoc networks do not have a clear line of defense, and every node must be prepared for encounters with an adversary directly or indirectly; in other words, all other nodes must be treated as untrustworthy.

We carried out simulation experiments on a 20-node 4 km × 3 km size network, and investigated HIDE's detection effectiveness under various mobility scenarios, as well as three attack intensity levels of 1, 2, and 5 percent, with respect to the background traffic intensity. It was found that HIDE gives significantly high detection rates along with low misclassification rates when the attack intensity is at 5 percent of the background traffic, but deteriorates slightly, by the approximate factors of 3 and 4, respectively, at the lower attack levels of 1 and 2 percent.

In Figs. 6 and 7 we show the similarity distance for two of the monitored parameters, received packet rate at the IP layer and received byte rate at the IP layer, when the average speed of the mobile hosts is equal to 50 km/h and the attack intensity is equal to 5 percent of the total background traffic. Notice that the average similarity values on the left half in these two figures, where the attack data is plotted, are lower than those of the right half, where the typical or normal data is plotted, thus indicating an overall finding and detection of attack instances vs. normal instances.

It should be noted that for these two parameters there is only a slight shift in the similarity measurements between attack vs. normal data. However, as shown earlier, combining several

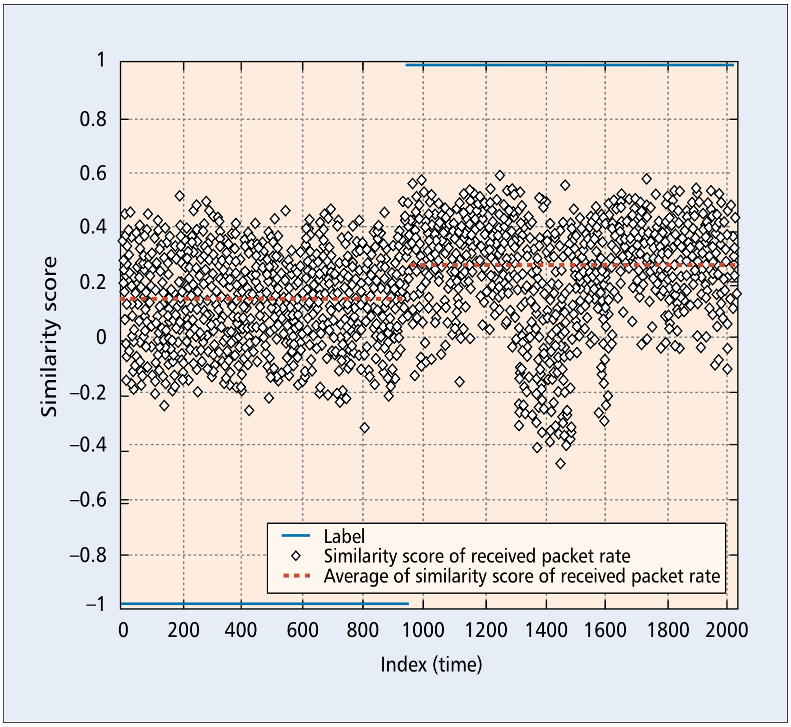


Figure 7. *Similarity score of received packet rate at the IP layer.*

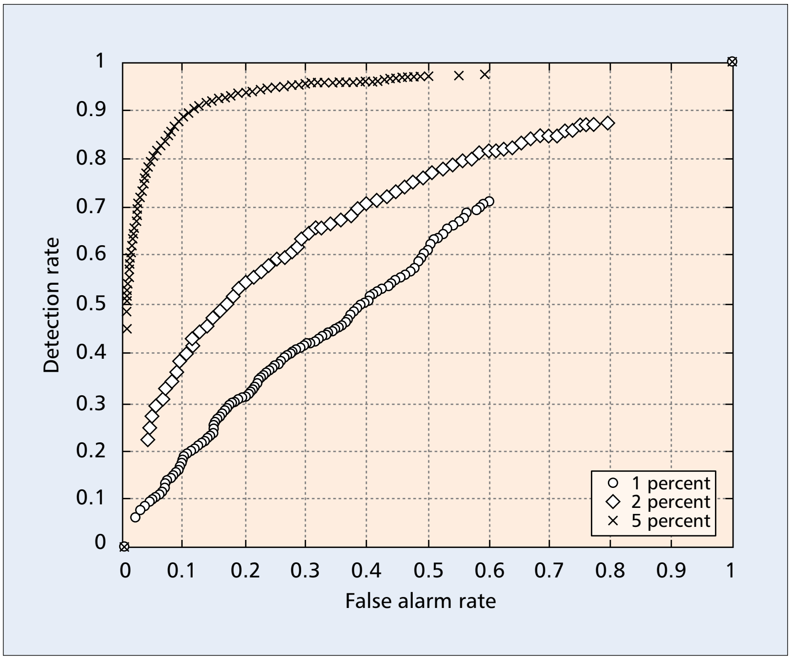


Figure 8. *ROC curve : the velocity of mobile hosts is equal to 50 km/h.*

parameters in one unified result enhances the classification ability substantially. This is shown in Fig. 8 that depicts the corresponding receiver operating characteristic (ROC) curves. It shows significantly high detection rates along with low misclassification rates when the attack intensity is as small as 5 percent of the background traffic. Similar results have been found for other mobility speeds (e.g., 20 and 80 km/h). In summary, these results indicate that the performance of our anomaly intrusion detection system performs well in the challenging setting of the mobile ad hoc network.

HIDE gives significantly high detection rates along with low misclassification rates when the attack intensity is at 5 percent of the background traffic, but deteriorates slightly at the lower attack levels of 1 percent and 2 percent.

CONCLUSION

The HIDE statistical anomaly detection technology is characterized by several innovative features. It is a hierarchical multitier multi-observation-window system that monitors several network traffic parameters simultaneously using a real-time PDF for each parameter, collected during the observation window. It calculates a similarity value of each measured PDF to the reference PDF, then intelligently combines the similarity measurements into an anomaly status vector that is classified by a neural network. This combining is powerful in that it achieves higher discrimination and decision robustness. All computation uses PDFs rather than individual or averaged sampled values. The numerical results demonstrate that this anomaly detection methodology can reliably detect attacks and soft faults with traffic anomaly intensity as low as 3–5 percent of typical background traffic intensity, thus promising to generate an effective early warning.

ACKNOWLEDGMENTS

This research acknowledges support by a Phase I and II SBIR contract with the U.S. Army and OPNET Technologies, Inc.™ for partially supporting the OPNET simulation software.

REFERENCES

[1] A. Housebolder, K. Houle, and C. Dougherty, "Computer Attack Trends Challenge Internet Security," *Security & Privacy —Supplement — IEEE Comp. Mag.*, Apr. 2002, pp. 5–7.

[2] G. Vigna and R. A. Kemmerer, "NetSTAT: A Network-based Intrusion Detection Approach," *Proc. 14th An. Comp. Sec. App. Conf.*, 1998, pp. 25–34.

[3] A. Valdes and D. Anderson, "Statistical Methods for Computer Usage Anomaly Detection Using NIDES," Tech. rep., SRI International, Jan. 1995.

[4] J. P. Anderson, "Computer Security Threat Monitoring and Surveillance," 1980.

[5] D. E. Denning, "An Intrusion-detection Model," *IEEE Trans. Soft. Eng.*, vol. SE-13, no. 2., Feb. 1987, pp. 222–32.

[6] L. Heberlein *et al.*, "A Network Security Monitor," *Proc. IEEE Comp. Soc. Symp., Research in Sec. and Privacy*, May 1990, pp. 296–303.

[7] R. P. Lippman *et al.*, "Results of the DARPA 1998 Offline Intrusion Detection Evaluation," *Proc. Recent Ad. Intrusion Det., RAID '99 Conf.*, West Lafayette, IN, Sept. 7–9, 1999.

[8] L. Ho *et al.*, "Adaptive/Automated Detection of Service Anomalies in Transaction WANs: Network Analysis, Algorithms, Implementation, and Deployment," *IEEE JSAC*, vol.18, no. 5, May 2000, pp. 744–57.

[9] J. M. Bonifacio *et al.*, "Neural Networks Applied in Intrusion Detection System," *IEEE*, 1998, pp. 205–10.

[10] B. D. Joao *et al.*, "Statistical Traffic Modeling for Network Intrusion Detection," *Proc. 8th Int'l. Symp. Modeling, Analysis Sim. Comp. Telecommun. Sys.*, Aug. 2000, pp. 466–73.

[11] Z. Zhang *et al.*, "HIDE: A Hierarchical Network Intrusion Detection System Using Statistical Preprocessing and Neural Network Classification," *Proc. 2nd An. IEEE Sys., Man Cyber. Info. Assurance Wksp.*, West Point, NY, June 2001.

[12] C. Manikopoulos *et al.*, "Generalized Anomaly Detection in Next Generation Internet: Architecture and Evaluation," submitted for publication, 2002.

BIOGRAPHIES

CONSTANTINE MANIKOPOULOS [M] (manikopoulos@njit.edu) received his B.S. degree from Hamline University, St. Paul. Minnesota, in 1967, and his Ph.D. degree in physics from Princeton University in 1973. He is a registered practicing engineer in the states of New Jersey and New York. During the time periods 1973–1978, 1978–1983, and 1986–1988, he served on the faculty of State University of New York at Buffalo, Rutgers University, and Fairleigh Dickinson University, respectively. From 1983 to 1986 he was vice president at Computron Technologies Corporation, where he was in charge of microcomputer development. He also served as adjunct assistant professor in the College of Medicine and Dentistry of New Jersey in Piscataway from 1981 to 1983. He is currently an associate professor in the Department of Electrical and Computer Engineering of the New Jersey Institute of Technology, and one of the founding members of the New Jersey Center for Wireless Networking and Security (NJWINS). He has published extensively in many journals and conferences, and has been awarded and administered many grants and contracts from various agencies. His current research interests are in the area of computer networks, neural networks, network security, and intrusion detection. He is a member of INNS, ACM, SPIE, and AAAUP.

SYMEON PAPAVASSILIOU (papavassiliou@adm.njit.edu) received his diploma in electrical engineering from the National Technical University of Athens, Greece, in 1990 and his M.Sc. and Ph.D. degrees in electrical engineering from Polytechnic University, Brooklyn, New York, in 1992 and 1995, respectively. From 1995 to 1996 he was a technical staff member at AT&T Bell Laboratories, Holmdel, New Jersey, and from 1996 to August 1999 he was a senior technical staff member at AT&T Laboratories, Middletown, New Jersey. From June 1996 till August 1999 he was also an adjunct professor at the Electrical Engineering Department of Polytechnic University. Since August 1999 he has been an assistant professor in the Electrical and Computer Engineering Department of New Jersey Institute of Technology, Newark. He was awarded the Best Paper Award in INFOCOM '94 and the AT&T Division Recognition and Achievement Award in 1997. He has an established record of publications in his field of expertise, is the director of the Broadband, Mobile and Wireless Networking Laboratory at NJIT, and one of the founding members of NJWINS. His main research interests lie in the areas of computer and communication networks with emphasis on wireless communications and high-speed networks, anomaly detection and fault management, network intrusion detection, computer network modeling and performance evaluation, and optimization of stochastic systems.