IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 13, Issue 1 (Jul. - Aug. 2013), PP 134-137
www.iosrjournals.org

Procuring the Anomaly Packets and Accountability Detection in the Network

V. Laxman[1], Ms. P. Subhadra[2]
[1](*Master of Technology, Computer Science and Engineering, Vardhaman College of Engineering, Hyderabad, India,*
[2](*Associate Professor Computer Science and Engineering, Vardhaman College of Engineering, Hyderabad, India)*

Abstract: *It is software that will be used to find the anomaly packets in Voice over Internet Protocol (VoIP) devices, such as soft phones and VoIP gateways to the Session Initiation Protocol specifications, and to test the compliance and interoperability of VoIP equipment produced by different manufacturers. Network traffic is often "different" from benign traffic in ways that can be distinguished without knowing the nature of the attack. We describe a two stage anomaly detection system for identifying suspicious traffic. First, we filter traffic to pass only the packets of most interest, e.g. the first few packets of incoming server requests. Second, we model the most common protocols (IP, TCP, telnet, FTP, SMTP, HTTP) at the packet byte level to flag events (byte values) that have not been observed for a long time.*

Different software's are available on the market to conduct a compliance and interoperability validation phase. However, they often have features limited to packet capturing and decoding, or they are simulation tools that often require a complex developing phase to define the behavior of each test. The proposed tool, instead, can be inserted into an Session Initiation Protocol (SIP) network and is capable of observing and finding, in an automatic way. It executes in three phases.

1. SIP messages flowing in the network are captured.
2. In charge of grouping SIP messages into transactions and dialog.
3. Operates by comparing the message flow with a set of predefined rules.

These Rules are classified into two groups.

1. Static Rules have been obtained by the direct analysis of SIP specifications.
2. Dynamic Rules have been obtained by experience with SIP compliance and interoperability testing.

If some rules failed during verification, an output is reported by indicating the rule that failed and a list of possible fault causes.

Index Terms- *Protocol, IDS, Anomaly..*

I. Introduction

At present, attacks on Internet infrastructure, in the form of denial of service (DoS) attacks and worms, have become one of the most serious threats to the network security. it could become possible to detect the attacks, anomalies and to appropriately take action to mitigate them before they have had much time to propagate across the network. Most network intrusion detection systems (IDS) that use anomaly detection look for anomalous or unusual port number and IP addresses, where "unusual" means any value not observed in training on normal (presumably non hostile) traffic. They use the firewall paradigm; a packet addressed to a nonexistent host or service must be hostile, so we reject it. The problem with the firewall model is that attacks addressed to legitimate services will still get through, even though the packets may differ from normal traffic in ways that could detect. In general, methodology to detecting anomalies envisions two kinds of detection mechanisms: postmortem and real-time modes.

Given the variety of anomalies, it makes sense to examine as many attributes as possible, not just the packet ports and addresses, as many systems now do. Also, because the rates of many types of network events vary unpredictably over a wide range of time scales, we use a time-dependent model, in which the probability of an event depends on the time since it last occurred, instead of on its average rate.

The idea is that if an attribute takes on a novel value, or at least one not seen recently, then the data is suspicious. In this paper, we briefly review previous work on packet buffer architectures and present scalable and efficient hierarchical packet buffer architecture. This is our first attempt to combine the merits of two previously published packet buffer architectures. Consequently, the SRAM occupancy has been significantly reduced. By fully exploring the advantage of parallel DRAMs, we first propose a memory management algorithm (MMA) called Random Round Robin (RRR). Thereafter, we devise a "traffic-aware" approach which aims to provide different services for different types of data streams. This approach further reduces the system overhead. Both mathematical analysis and simulation demonstrate that the proposed architecture together with

its algorithm reduce the overall SRAM requirement significantly while providing guaranteed performance in terms of low time complexity, upper bounded drop rate, and uniform allocation of resources.

II. Background And Literature Survey

Network intrusion detection systems like snort (2001) or Bro (Paxson, 1998) typically use signature detection, matching patterns in network traffic to the patterns of known attacks. This works well, but has the obvious disadvantage of being vulnerable to novel attacks. An alternative approach is anomaly detection, which models normal traffic and signals any deviation from this model as suspicious. The idea is based on work by Forrest et al. (1996), who found that most UNIX processes make (mostly) highly predictable sequences of system calls in normal use. When a server or suid root program is compromised (by a buffer overflow, for example), it executes code supplied by the attacker, and deviates from its normal calling sequence. It is not possible to observe every possible legitimate pattern in training, so an anomaly detector requires some type of machine learning algorithm in order to generalize from the training set. Models are usually based on the distribution of source and destination addresses and ports per transaction (TCP connections, and sometimes UDP and ICMP packets). For example, SPADE offers four probability models (estimated by average frequency) of incoming TCP connections:

· P(destination-address, destination-port)
· P(source-address, destination-address, destination-port)
· P(source-address, source-port, destination-address, destination-port)
· Bayes network approximation of the above.

Lower probabilities result in higher anomaly scores, since these are presumably more likely to be hostile. In general, five types of anomalies in hostile traffic are identified.

2.1 User Behavior: Hostile traffic may have a novel source address because it comes from an unauthorized user of a restricted (password protected) service. Also, probes such as ipsweep and portsweep may attempt to access nonexistent hosts and services, generating anomalies in the destination addresses and port numbers.

2.2 Bug Exploits: Attacks often exploit errors in the target software, for example, a buffer overflow vulnerability. Such errors are most likely to be found in the least-used features of the program, because otherwise the error is likely to have been discovered during normal use and fixed in a later version. Thus, any remaining errors are invoked only with unusual inputs (e.g. a very long argument to a seldom used command), which are not likely to occur during normal use.

2.3 Response Anomalies: Sometimes a target will generate anomalous outgoing traffic in response to a successful attack, for example, a victimized mail server passing root shell command responses back to an attacker. This is analogous to Forrest's host based detection method , where a server compromise can be detected because it makes unusual sequences of system calls.

2.4 Bugs in the attack: Attackers typically must implement client protocols themselves, and will fail to duplicate the target environment either out of carelessness or because it is not necessary. For example, many text based protocols such as FTP, SMTP and HTTP allow either uppercase or lowercase commands. An attacker may use lowercase for convenience, even though normal clients might always use uppercase.

2.5 Evasion: Attackers may deliberately manipulate network protocols to hide an attack from an improperly coded intrusion detection system (IDS) monitoring the application layer. Such methods include IP fragmentation, overlapping TCP segments that do not match, deliberate use

Of bad checksums, short TTL values, and so on. Such events must be rare in normal traffic, or else the IDS would have been written to handle them properly.

III. Model Description

Anomaly model detectors like PHAD, detects anomalies in network packets. However, it differs as follows:

1. The traffic is filtered, so only the start of incoming server requests are examined.
2. Starting with the IP header, we treat each of the first 48 bytes as an attribute for our models--we do not parse the packet into fields.
3. There are 9 separate models corresponding to the most common protocols (IP, TCP, HTTP, etc.).
4. The anomaly score tn/r is modified to (among other things) score rare, but not necessarily novel, events.

3.1. Traffic Filtering

The first stage of NETAD is to filter out uninteresting traffic. Most attacks are initiated against a target server or operating system, so it is usually sufficient to examine only the first few packets of incoming server requests. This not only filters out traffic likely to generate false alarms

NETAD separately models 9 subsets of the filtered traffic corresponding to 9 common packet types, as follows.

· All IP packets (including TCP, UDP, and ICMP).

· All TCP packets.
· All TCP SYN packets (with no other flags set, normally the first packet, usually containing TCP options and no payload).
· All TCP ACK packets (normally the second and subsequent packets, which contain a payload).
· TCP ACK packets to ports 0-255.
· TCP ACK to port 21 (FTP).
· TCP ACK to port 23 (telnet).
· TCP ACK to port 25 (SMTP mail).
· TCP ACK to port 80 (HTTP).

A packet may belong to more than one subset. For instance, an HTTP data packet is also TCP ports 0-255, TCP ACK, TCP, and IP. For each model, an anomaly score is computed, and the sum is assigned to the packet. The reason for modeling ports 0-255 is for ease of implementation. Each packet type can be distinguished from some other by examining only one attribute (byte). For instance, ports 0-255 can be distinguished from other TCP ACK packets by examining only the upper byte of the destination port number.

IV. Experimental Analysis

We employed a 2-hour DWT window in 1-minute sampling interval. It can be decomposed up to level 7. The results of our real-time analysis are shown in Fig. 1.

The DWT signal at each timescales is shown along with the horizontal detector (an anomaly detected over successive samples at the same level). The bottom-most picture shows the composite detector that employs two -dimensional mechanism using horizontal and vertical detection simultaneously . The results indicate that the real-time analysis detects all the attacks along with a few anomalies present in the base signal. It seems feasible to carry out a similar correlation and wavelet-based analysis of network packets based on their port numbers. Is it possible to combine several indicators to build a more robust anomaly detector that is less prone to false alarms? Fig. 2 shows the comprehensive anomaly detector based on a combination of addresses and port numbers.

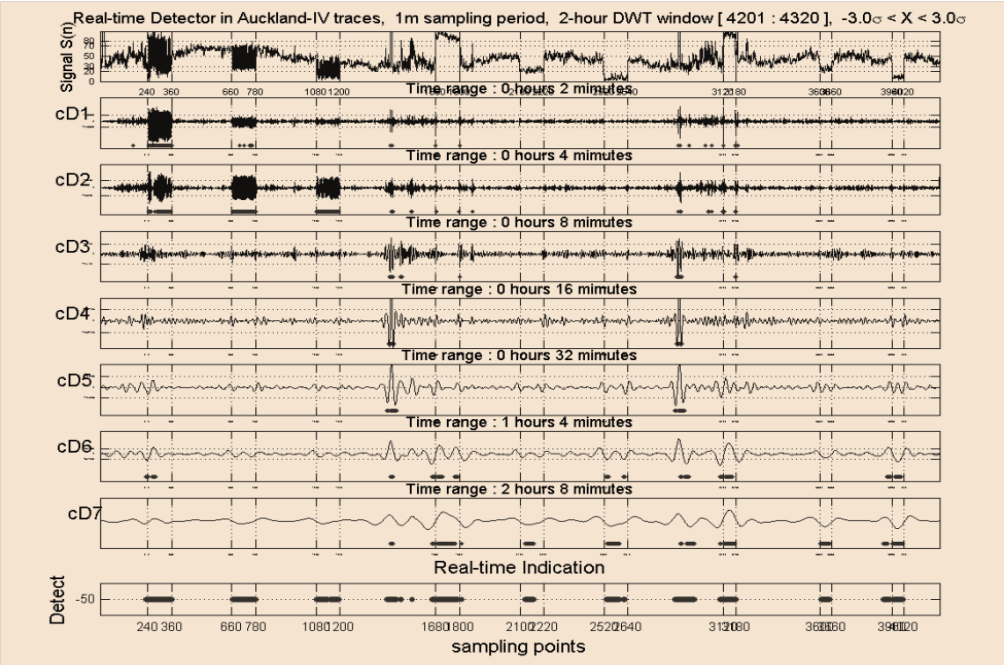

Figure 1. Address based detection results using simulated attack traces in real-time

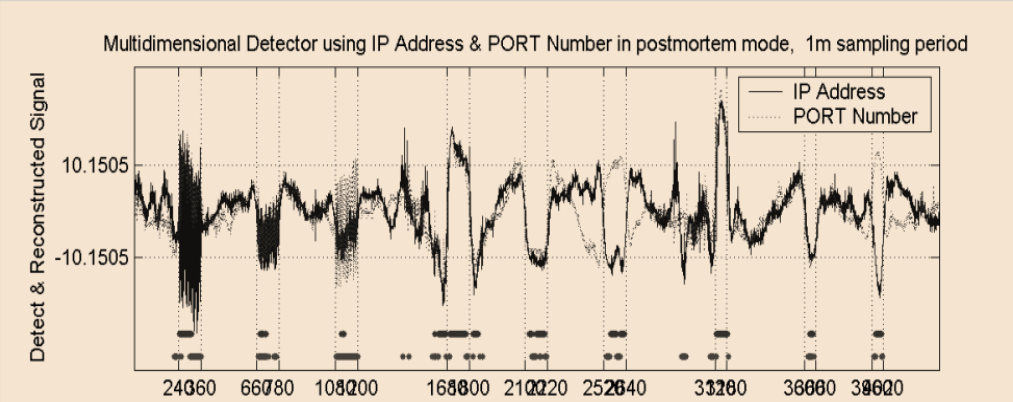

Figure 2. The multidimensinal detection results

The two kinds of dots at the bottom of the picture show detection results. The dots located on top are marked when both the address and port methods detect anomalies simultaneously.

V. Conclusions

Intrusions generate anomalies because of bugs in the victim program, the attacking program, or the IDS, or because the victim generates anomalous output after an attack. We have seen examples of all four types of anomalies:

1. Strange input to poorly tested software, e.g. the corrupted packets used by teardrop, pod, dosnuke.
2. Strange data from poorly written attacks, e.g. bad checksums in smurf and udpstorm.
3. Strange data used to hide an attack from layers above, e.g. FIN scanning by portsweep.
4. Strange responses from the victim, e.g. unusual TCP options generated by apache2.

The proposal like any network anomaly detector, does not describe the nature of an attack, or even indicate if an event is hostile or not. Instead, it just finds unusual or interesting events in a vast amount of data, and brings them to the attention of a network security expert for further analysis. It is meant to supplement, rather than replace, existing security tools and methods, such as code reviews, encryption, firewalls, virus detectors, and so on. Like all security tools, it requires some investment of time and effort to use it. The challenge is to use good filtering to minimize this effort.

References

[1] Bell, Timothy, Ian H. Witten, John G. Cleary, "Modeling for Text Compression", *ACM Computing Surveys (21)4*, Dec.1989 ,pp. 557-591.

[2] Barbará, D., N. Wu, S. Jajodia, "Detecting Novel Network Intrusions using Bayes Estimators", *First SIAM International Conference on Data Mining*, 2001,

[3] Floyd, S. and V. Paxson, "Difficulties in Simulating the Internet." *To appear in IEEE/ACM Transactions on Networking*, 2001. http://www.aciri.org/vern/papers.html

[4] Forrest, S., S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, "A Sense of Self for Unix Processes", *Proceedings of 1996 IEEE Symposium on Computer Security and Privacy.* ftp://ftp.cs.unm.edu/pub/forrest/ieee-sp-96-unix.pdf

[5] Lippmann, R., et al., "The 1999 DARPA Off-Line Intrusion Detection Evaluation", *Computer Networks 34(4)* , 2000,579-595.

[6] Mahoney, M., P. K. Chan, "PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic", *Florida Tech. technical report* 2001-04, http://cs.fit.edu/~tr/

[7] Mahoney, M., P. K. Chan, "Learning Models of Network Traffic for Detecting Novel Attacks", *Florida Tech. technical report* 2002-08, http://cs.fit.edu/~tr/

[8] Mahoney, M., P. K. Chan, "Learning Non stationary Models of Normal Network Traffic for Detecting Novel Attacks ", *Edmonton, Alberta: Proc. SIGKDD*, 2002, 376-385.

[9] Paxson, Vern, "Bro: A System for Detecting Network Intruders in Real-Time", *Lawrence Berkeley National Laboratory Proceedings, 7'th USENIX Security Symposium*, Jan. 26-29, 1998, San Antonio TX,

[10] F. Wang, M. Hamdi, and J. Muppala, "Using Parallel DRAM to Scale Router Buffers," *IEEE Trans. Parall and Distributed Systems*, vol. 20, May 2009, pp. 710-724.

[11] G. Appenzeler, I. Keslassy, and N. McKeown, "Sizing Router Buffers," *ACM SIGCOMM Computer Comm. Rev., vol. 34, no. 4*, Oct. 2004, pp. 281-292.

[12] G. Shrimali and N. McKeown, "Building Packet Buffers with Interleaved Memories," *Proc.Workshop High Performance Switching and Routing (HPSR '05),* May 2005, pp. 1-5.

[13] H. Wang and B. Lin, "Block-Based Buffer with Deterministic Packet Departure," *Proc. Int'l Conf. High Performance Switching and Routing (HPSR '10)*, June 2010, pp. 38-43.

[14] H. Wang, H. Zhao, B. Lin, and J. Xu, "Design and Analysis of a Robust Pipelined Memory System," *Proc. IEEE INFOCOM '10,* Mar. 2010 ,pp. 1-9,.

[15] J. Corbal, R. Espasa, and M. Valero, "Command Vector Memory Systems: High performance at Low Cost," *Proc. Int'l Conf. Parallel Architectures and Compilation Techniques,*Oct. 1998 ,pp. 68-77.

[16] J. Garcia, J. Corbal, L. Cerda, and M. Valero, "Design and Implementation of High-Performance Memory Systems for Future Packet Buffers," *Proc. IEEE/ACM 36th Ann. Int'l Symp. Micro architecture (Micro '03),* Dec. 2003, pp. 372-384.