An Integrated Network Security Approach

Pairing Detecting Malicious Patterns with Anomaly Detection

Ulrich Ultes-Nitsche and InSeon Yoo
Department of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, United Kingdom, e-mail: {uun,isy01r}@ecs.soton.ac.uk

Key words: Integration of different network security mechanisms, combining pattern matching and anomaly detection for network traffic monitoring

Abstract: We report in this paper on research in progress concerning the integration of different security techniques. A main purpose of the project is to integrate as many security functionality as possible into the firewall. We will report in this paper on the concept of an intelligent firewall that contains a smart detection engine for potentially malicious data packets.

1. INTRODUCTION

Internet security breaches are growing. Viruses are one of the major causes of the rising number of security breaches. Nowadays some companies, which recognize the importance of security, adopt security systems such as firewalls, intrusion detection systems, and virus scanning proxy servers. It is not easy to use all security systems owing to high cost. However, it is also not enough to protect a company's system with only a single security system, because each security system has different features in present days, in which particularly DoS (Denial of Service) and virus attacks are becoming serious.

We report in this paper on an ongoing project on the integration of different techniques in network security. We investigate the mutually beneficial effects that integrating firewalls, intrusion detections systems, and virus monitors may have to network security. In particular, we aim at integrating as many security measures as possible into the firewall, creating

what we will call an *intelligent firewall*. One of the main features of the intelligent firewall will be an anomaly detection capability, which will be achieved by applying AI techniques to detect unusual network traffic (an intrusion detection capability) and to detect unusual content of data packets (a capability to identify unknown potential viruses).

2. RELATED WORKS

Related to our concept of an intelligent firewall, there are two existing approaches we would like to discuss here. The first one in [JM99] proposes an ATM firewall using a proxy cache, which uses a QoF (Quality of Firewalling) scheme. Its main components are *call screening*, *proxy*, *traffic monitoring service*, *packet filtering service*, and *firewall management*. These combined components determine a packet's safeness. The packet-filtering service inspects the headers of IP packets to block unsafe packets, while allowing safe packets to pass. The traffic-monitoring service checks the packet headers against the traffic-monitoring rules, which are similar to the packet-filtering rules.

The second approach in [Hughes96], which was referred to in [JM99], proposes a *policy cache architecture*. To determine whether or not a packet is safe, only the first cell will be checked, which contains the IP header, protocol, TCP/UDP ports and TCP flags. However, there are limitations: IP packets with IP option fields are not accepted, because IP options can be as large as 40 bytes and may push the TCP headers to the second cell. Using CAM (Content Addressable Memory) to cache a safe header is not a scalable solution. CAM cannot scale to a large size due to technological constraints and is extremely expensive.

Both of these papers use only TCP/IP headers – no payload information is used – to detect whether data packets are safe or not, even though they aim to develop a new firewall architecture. It does not seem that inspecting only header information is sufficient to overcome weaknesses of firewalls.

3. VIRUS AND INTRUSION DETECTION

3.1 Traditional Virus Monitors

Virus monitors examine network traffic, aiming to prevent malicious code from entering network nodes by detecting *known* malicious-code patterns, for instance in an e-mail attachment. Apparently, they can detect only known viruses.

All of the major anti-virus vendors have produced networked products and systems that scan incoming e-mail. However, because Trojan horses, worms and viruses can spread through local networks, shared hard drives and individual document files, as well as through the Internet, it is always necessary to have virus checking on each client machine as well as on Internet gateways. Too often, patterns that catch new malware are not ready until days or even weeks after serious damage has been done.

New viruses will only become detectable after their pattern characteristics have been analysed and are made available. Looking at techniques applied by other security systems, in our case intrusion detection systems, seems to benefit virus detection [S2000].

3.2 Applying Ideas from Intrusion Detection Systems

3.2.1 Anomaly Detection in Intrusion Detection Systems

Intrusion detection systems (IDSs) very frequently run a process known as anomaly detection. An IDS based on this paradigm will constantly monitor network traffic and compare the stream of network packets with what it perceives as *normal* network traffic. As soon as it observes an anomalous pattern in the traffic it will throw a warning for the network administrator to deal with it.

Anomaly detection appears to be applicable not only to intrusion detection but also to virus monitoring [S2000], now not being applied on the level of the full network traffic, but to single data packets: The improved virus monitor will examine data packets as usual. But besides checking against known malicious-code patterns, it will check whether it sees a pattern that it perceives as *potentially malicious* and will react accordingly, e.g. by creating a warning of some sort.

3.2.2 Using Neural Networks for Detection

Nowadays almost all practical IDSs are signature-based systems. These systems work based on predefined descriptions of attack signatures. Various data source and type-of-pattern recognition techniques are still used. However, many known attacks can be easily modified to present many different signatures. If not all variations are in the database, a known attack may be missed. Moreover, early systems were constructed around concepts of statistical anomaly detection. These systems faced practical and theoretical difficulties, such as performance and creation of false positives.

Some more modern approaches try to exploit neural-network-based techniques. To explore attack spaces, [SD2001] uses a hierarchy of back

propagation (BP) neural networks (NN) for the protocol and the self-organizing map (SOM) technique for the anomaly classification. In an experiment, however, only artificially generated data was used.

[GS99] uses a classical feed-forward multi-layer perceptron network: a back propagation neural network and time delay neural network to program-based anomaly detection.

Lastly, to identify and classify network activity based on limited, incomplete, and nonlinear data sources, [CM98] presents an analysis of the applicability of neural networks. In the neural network architecture, multi-layer perceptron (MLP) and hybrid forms (MLP + SOM) are adapted.

3.3 Integrating Firewalls

Firewalls are used to guard and isolate connected segments of inter-networks. “Inside” network domains are protected against “outside” un-trusted networks, or parts of a network are protected against other parts [Schuba97]. There are three types of firewalls [JKJ2002].

3.3.1 Types of Firewalls

Firstly, there are packet filter firewalls. Packet filtering focuses mainly on accepting or denying packets. It’s not suitable for defence means against intruders and therefore just appropriate as another security measure. Main strengths of packet filter firewalls are their speed and flexibility. These systems can be used to secure nearly any type of network communication or protocol. They can be deployed easily into nearly any enterprise network infrastructure. However, they cannot prevent the network from elaborate attacks, because they do not examine upper-layer data. For instance, they do not support advanced user authentication schemes and cannot detect network packets in which the OSI layer 3 addressing information has been altered.

Secondly, stateful inspection firewalls add layer 4 awareness to the standard packet filter architecture. These systems share the strengths and weaknesses of packet filter firewalls. The actual stateful inspection technology is relevant only to TCP/IP. Moreover their use is very costly as the state of connection is monitored at all times.

Thirdly, application-proxy-gateway firewalls have more extensive logging capabilities, are capable of authenticating users directly, and can be made less vulnerable to address spoofing attacks. These systems are, however, not generally well suited for high-bandwidth or real-time applications.

3.4 Necessity of Data Examination Ability

Current IDSs do not prevent an intrusion from happening; they only detect and report it. When a virus associated with a DoS (Denial of Service) attack spreads through the Internet (e.g. the CodeRed virus), virus monitors and IDSs should co-operate to prevent such an attack. However, although virus monitors and IDSs are installed, new virus information needs to be updated constantly. To prevent malicious virus data from entering a LAN, firewalls should be equipped with a virus-detection ability to inspect not only the header but also the data part (payload) of packets.

4. THE INTELLIGENT FIREWALL

The intelligent firewall is our proposal for increasing the strength of firewalls. It will have the ability to examine entire data packets and to apply standard as well as intelligent detection techniques to identify misuse.

4.1 Data Packet Detection

There are several types of computer virus classes: viruses, Trojan horses, worms, hoaxes, jokes etc. Among these types, a virus is a piece of code that adds itself to other programs and cannot run independently. However, worms are programs that can run by themselves and propagate a fully working version of themselves to other machines.

As Microsoft Windows became popular, windows viruses and windows-application-derived viruses using VBA (Visual Basic for Applications) spread widely. Moreover, a common way of windows virus prevalence is through emails. The recent important one was Code Red. The Code Red worm is a malicious self-propagating code [CERT2002] that spreads surreptitiously through a hole in certain Microsoft software, such as Internet Information Server (IIS) Web software and the Windows NT and Windows 2000 operating systems.

To identify viruses/worms in e-mail attachments, one exploits that data packets have a unique character, the *virus signature*. In addition to the usual network control ability of a firewall, data packet detection is necessary in the intelligent firewall to identify packets containing malicious data: The virus signatures are also appearing in data packets. For instance, the beginning of the Code Red's attack packet looks like the following [L117]:

```
GET/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3
```

Network-based IDSs also monitor network traffic on the wire for specific activities or signatures that represent an attack [Richards99]. Strengths of IDSs are to monitor a large network and to have little impact on an existing network [RP2001]. Moreover, they detect malicious and suspicious behaviour in true real-time and provide faster response and notification to the attack at hand [Brian2000]. They examine all packet headers for signs of malicious and suspicious activity and can also investigate the content of the payload. They use live network traffic for attack detection in real-time and a hacker cannot remove this evidence once captured. However, a weakness of IDSs is their inability to process all packets, which can lead to possibly failing to recognise an attack during high traffic and the need to analyse packets quickly. Performance problems, especially with increasing network speeds, and resource exhaustion problems can lead to difficulties [JAJ2000]. Resource exhaustion problems can occur when an IDS must maintain attack-state information for many attacked hosts over a long period of time. It is also possible to misunderstand normal traffic as malicious traffic. Many approaches can trigger numerous false positives, because of normal traffic looking very close to malicious traffic. In addition, a network-based IDS does not control the network or maintain its connectivity [TT98]. Hence these systems are vulnerable to DoS (Denial of Service) attacks.

4.2 Need for a Dynamic Packet Handling Ability

Although a firewall is able to control a network and maintain its connectivity, it handles packets only statically. Through open ports, a firewall would not inspect/control packet willingly.

DoS (Denial of Service) attacks can interrupt services by flooding networks or systems with unwanted traffic. A service is denied either because the network/system is overwhelmed or because the network/system turns offline. The service will be denied until the source of the attack can be identified and calls from that source are blocked. DoS attacks are easy to perpetrate and almost impossible to defend against, even though firewalls and IDS are installed. Even if the Intrusion Detection System generates alerts, logs packets, and sends emails to and calls pagers of system administrators, the attacker could still get in, and by the time somebody

could respond the damage would be done. An attacker could spoof attacks from many sources and effectively deny everybody access to the server. A firewall would be of no help either: It has no way of determining whether a request being sent to a web server is benign or malicious. While the firewall could stop traffic to ports that do not need to be publicly accessible, it is useless in the discussed situation.

On the other hand, distributed systems based on the client/server model have become increasingly popular. Using this scheme, DDoS (Distributed Denial of Service) attacks are also getting escalated. In DDoS form, an attacker controls a number of handlers. A handler is a compromised host with a special program running on it. Each handler is capable of controlling multiple agents. An agent is a compromised host, which is responsible for generating a stream of packets that is directed toward the intended victim.

According to the analysis of distributed denial of service attack tools, TFN [Dittrich99a], TFN2K, Trinoo [Dittrich99b] and Stacheldraht [Dittrich99c] are well known how to use. These programs not only use TCP and UDP but also ICMP packets. Moreover, because the programs use ICMP_ECHOREPLY packets for communication, it will be very difficult to block attacks without breaking most Internet programs that rely on ICMP. Since TFN, TFN2K and Stacheldraht use ICMP packets, it is much more difficult to detect them in action, and packets will go right through most firewalls. The current only sure way to destroy this channel is to deny *all* ICMP_ECHO traffic into the network. Furthermore, the tools mentioned above use any port randomly; it is hard to prevent the port from malicious attack in advance using the fixed port close scheme in current firewalls.

Code Red, which leaves computers open to hijacking, has caused a lot of traffic being sent, clogging the bandwidth on the Internet. An infected system will show an increased processor and network load. The worm could easily permit hackers to take control of hundreds of thousands of infected machines. The worm has this magnifying effect on network traffic during attacks on internal networks. Even though Code Red attacks are not real DDoS attacks, their result is similar to DDoS. Therefore, to prevent degradation of service on the network and to deny this kind of malicious packet, dynamic packet handling on the level of firewalls is crucial.

4.3 Intelligent Detection Engine

To prevent malicious self-propagating virus attacks from entering into intranets, dynamic filtering of data packets is compulsory. Having the ability of anti-virus systems and IDSs, a malicious-data-packet-detection engine needs to be developed. Like a firewall, this system would have the ability to

accept or deny packets. Like an intrusion detection system, it would examine a packet's content, flags, and headers to make a decision.

4.3.1 Anomaly Packet Formation

As we examine virus packets filtered by MailScanner, which is running in the Southampton ECS intranet, the front parts of these packets display similar patterns. And indeed, if we capture "good packets," which have already been filtered by the ECS firewall and MailScanner, these good packets are different. Most front parts of the good packets display a variety of patterns. To capture the investigated packets we have used the *libcap* [PCAP] packet capturing ability of *snort* [SNORT].

However, in some of the good packets, we could see patterns very similar to the ones possible in packets containing malicious code. These were packets sent by Microsoft Servers to NetBios and DNS lookup services. For example, port 137 is reserved for the NetBIOS name service and port 138 is reserved for the NetBIOS datagram service. The subsequent packet was assumed to contain the signature of the "BAT 911 /Chode" worm even though it was a benign packet:

```
05/24-13:10:13.082716 152.78.70.46:137 -> 152.78.70.127:137
UDP TTL:128 TOS:0x0 ID:47635 IpLen:20 DgmLen:78
Len: 58
……..
0x0030: 00 00 00 00 00 00 20 45 45 46 44 46 44 45 46 43  ..... EEFDFDEFC
0x0040: 41 43 41 43 41 43 41 43 41 43 41 43 41 43 41 43  ACACACACACACACAC
0x0050: 41 43 41 43 41 42 4C 00 00 20 00 01              ACACABL.. ..
```

Except these Microsoft Server packets, all other packets we have captured had a considerably different front part from data packets containing virus code. This observation suggests that malicious and benign packets look sufficiently different to consider an anomaly-based virus detection approach possible, which could identify new, *unknown* viruses.

4.3.2 Considerable Approaches for the Detection Engine

To accomplish the detection engine sketched above, we consider several approaches. Each approach obviously has its own advantages and disadvantages; we will therefore decide on which approaches to deploy in the detection engine only after their thorough examination.

The first approach to consider is the Bayesian Belief Network (BBN). A BBN is a special type of diagram, or graph, together with an associated set

of probability tables. Nodes of the BBN represent entities and their attributes; the arcs describe the relationships between these entities. BBNs are beneficial to model uncertain events and arguments about them. Moreover, if Bayesian probability is applied to propagate, it produces useful probability estimates for uncertain outcomes.

The second approach is Bayesian estimation. As an aspect of incremental learning, learning the parameters of a model can be addressed by Bayesian estimation. This has the following advantages: incorporation of prior knowledge and constraints, incorporation of models of time evolution, choice of optimality criteria, and, for linear Gaussian models, the Bayesian estimator is optimal under almost any rational optimality criterion. However, it has also disadvantages: a prior distribution must be known, and closed-form solutions are often hard to find.

Thirdly, the statistical approach requires that behaviour profiles are generated and updated over time. Statistical systems were confronted with practical and theoretical difficulties. A practical difficulty is that nominal usage has high variability and changes over time. To meet this challenge, systems had a fairly loose threshold for tolerance of anomalous behaviour, and were designed to learn new nominal statistics as they worked. This solution to the practical limitations of statistical anomaly detectors led to the theoretical difficulty [SD2001] that intruders could work below the threshold of tolerance and teach the systems to recognize increasingly abnormal patterns as normal.

Next, the data mining approach describes the discovery of useful summaries of data based on relations, patterns, and rules that exist in the data. Pattern extraction and discovery as well as feature extraction capabilities of data mining processes seem to offer interesting possibilities for our detection engine.

Finally, there is the Neural Network approach. As the anomaly of data packets cannot be reduced explicitly to matching exact patterns, we need a mechanism that is capable of distinguishing between “good” and “bad” patterns in data packets based on (incomplete) knowledge about the general structure of these packets. To detect “bad” patterns, we take much account of five aspects: pattern classification, competitive learning, unsupervised learning, good performance in noise and error, and flexible time delay. Possession of such features can be found in neural-network-based detection. We will closely investigate 5 neural network models to deal with the 5 aspects of pattern classification as mentioned above: Self-Organizing Maps (SOM), Hopfield Networks, Hamming Networks, Probabilistic Neural Networks (PNN), and Time-Delay Neural Networks (TDNN). An experiment-led detailed analysis of these different models in regard to their suitability for the detection engine is part of future work in this project.

Currently we have produced a firewall layout using libpcap [PCAP] and snort [SNORT] modules as already mentioned in sub-section 4.3.1. To capture packets from the datalink layer we have used libpcap, and to decode data packets we have modified parts of snort. Our goal is to build a smart detection engine and integrate it into the prototype firewall we have built. This engine will be useful if its detection rate is higher than that of traditional IDSs' anomaly detection rate with an acceptably low rate of false positive. Please recall that the smart detection engine will not only aim at detecting anomalous network traffic as in classical IDSs, but also to detect unusual structures in data packets that suggest the presence of virus data. To accomplish this goal we are currently investigating the above-mentioned different classification techniques in detail to identify suitable candidates to implement and experiment with in our firewall prototype.

5. CONCLUSIONS AND FUTURE RESEARCH

Integrating techniques of different security systems appears to offer interesting possibilities. We have reported in this paper on an ongoing project that aims at fulfilling such a task. The basic concept is to integrate a smart detection engine into a firewall, resulting in what we call an *intelligent firewall*. To date, our experimentation platform only has the capability to capture layer 4 data packets and analyse them exhaustively. For packet comparison, we use a database containing a substantial number of data packets related to various known viruses.

We are currently exploring the potential of using several AI methods in the smart detection engine, adding anomaly-detection-based IDS and virus monitor capability to the engine. A neural network, for instance, trained with a set of malicious-code patterns, will create warnings when *similar* patterns are detected. A very brief outline of such an idea can be found in [TKS96]. Problems to tackle include the performance of the decision process (to deal with real-time network traffic) and optimisations to avoid the creation of too many false positives. The selection of suitable techniques will require sufficient experimental evidence of their applicability. Such experiments form the next step in the discussed project.

Additional information about the project we reported on can be found at http://www.ecs.soton.ac.uk/~isy01r/.

6. REFERENCES

[Brian2000] Brian Laing. *How To Guide-Implementing a Network Based Intrusion Detection System*. Sovereign House 57/59 Vaster Road Reading RG1 8BT, UK, 2000.

[CERT2002] CERT Advisory CA-2001-23. *Continued Threat of the "Code Red" Worm*. 2002. http://www.cert.org/advisories/CA-2001-23.html

[CM98] James Cannady and James Mahaffey. The application of artificial neural networks to misuse detection: initial results. In: *Proceedings of the 1st International Workshop on Recent Advances in Intrusion Detection* (RAID 1998), 1998.

[Dittrich99a] D. Dittrich. *The "Tribe Flood Network" distributed denial of service attack tool*. October 1999. http://staff.washington.edu/dittrich/misc/tfn.analysis.

[Dittrich99b] D. Dittrich. *The DoS Project's "trinoo" distributed denial of service attack tool*. October 1999. http://staff.washington.edu/dittrich/misc/trinoo.analysis.

[Dittrich99c] D. Dittrich. *The "stacheldraht" distributed denial of service attack tool*. December 1999. http://staff.washington.edu/dittrich/misc/stacheldraht.analysis.

[Hughes96] J. Hughes. *A high speed firewall architecture for ATM/OC-3C*. StorageTek Corp, 1996. http://www.network.com

[GS99] Anup K. Ghosh and Aaron Schwartzbard. *A study in using neural networks for anomaly and misuse detection*. 1999. http://www.docshow.net/ids/usenix_sec99.zip

[JAJ2000] John Mchugh, Alan Christie, and Julia Allen. Defending Yourself: The Role of Intrusion Detection Systems. In: IEEE Software, September/October 2000, pp 42-51.

[JKJ2002] John Wack, Ken Cutler and Jamie Pole. *Guidelines on Firewalls and Firewall Policy*. NIST (National Institute of Standards and Technology), Special Publication 800-41, January 2002. http://csrc.nist.org/publications/nistpubs/800-41/sp800-41.pdf

[JM99] Jun Xu and Mukesh Singhal. Design of a High-performance ATM firewall. In: *ACM Transactions on Information and System Security*, Vol.2, No.3, August 1999, pp 269-294.

[L117] L-117: *The Code Red Worm, U.S. Department of Energy*. CIAC (Computer Incident Advisory Capability), July 19, 2001. http://www.ciac.org/ciac/bulletins/l-117.shtml

[PCAP] *LIBPCAP*. The Tcpdump Group.
http://www.tcpdump.org

[Richards99] Kevin Richards. Network Based Intrusion Detection: A Review of Technology. In: *Computers & Security*, 18(1999) 671-682.
http://zen.ece.ohiou.edu/~inbounds/DOCS/reldocs/NBIDRT.ps

[RP2001] Rebecca Bace and Peter Mell. *NIST special publication on Intrusion Detection System*. NIST (National Institute of Standards and Technology) Special Publication 800-31, August 2001.
http://csrc.nist.gov/publications/nistpubs/800-31/sp800-31.pdf

[S2000] M. Swimmer. Review and Outlook of the Detection of Viruses using Intrusion Detection Systems. In: *Proceedings of the 3rd International Workshop on Recent Advances in Intrusion Detection* (RAID 2000).

[Schuba97] Christoph Ludwig Schuba. *On the Modeling, Design and Implementation of Firewall Technology*. Purdue University, PhD thesis, 1997.
ftp://ftp.cerias.purdue.edu/pub/papers/christoph-schuba/schuba_phddis.pdf

[SD2001] Susan C. Lee and David V. Heinbuch. Training a neural network-based intrusion detector to recognize novel attacks. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, July, 2001, 31(4) pp.294-299.

[SNORT] SNORT. *The Open Source Network Intrusion Detection System*.
http://www.snort.org

[TKS96] G. Tesauro, J.O. Kephart, and G.B. Sorkin. Neural Networks for Computer Virus Recognition. In: *IEEE Expert*, 11(4): 5-6, August 1996.

[TT98] Thomas H. Ptacek and Timothy N. Newsham. *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*, Secure Networks, Inc. January, 1998.
http://www.snort.org/docs/idspaper/