

Using Automatic Signature Generation as a Sensor Backend

Daniel Wyschogrod

BAE Systems

6 New England Executive Park

Burlington, MA 01803

1-781-505-4594

dan.wyschogrod@baesystems.
com

Jeffrey Dezso

BAE Systems

6 New England Executive Park

Burlington, MA 01803

1-781-262-4523

jeffrey.dezso@baesystems.com

ABSTRACT

The techniques and supporting tools for signature based intrusion detection have reached a high level of maturity. They are well understood by the community and have hardware implementations capable of matching rules at high speed. Their major shortcomings involve handling “zero-day” attacks. Anomaly or protocol-adherence based sensors are capable of detecting zero-day attacks, but with high false alarm rates and at more limited speeds. The design proposed here combines the zero-day detection capabilities already supplied by anomaly detection front ends with the speed, hardware compatibility and mature infrastructure of signature based systems. A unique capability of this proposed technology is that false alarm rates of matched rules can be reduced to arbitrarily low levels by increasing the amount of training on benign traffic. A goal of future work would be to produce an efficient and secure mechanism to distribute automatically generated signatures with the goal of broadening the perimeter of protection and blocking attacks farther away from sensitive servers and hosts.

Categories and Subject Descriptors

C.2.0 [General]: Security and protection.

General Terms

Algorithms, Security

Keywords

Security, signature matching, automatic signature generation, pattern matching.

1. INTRODUCTION

Various commercial and open source systems currently exist for signature-based intrusion detection. Many of these systems are at a high level of maturity and are able to match large numbers of signatures at high data rates. Typically, signatures are generated by highly trained analysts who evaluate attacks by hand and perform manual extraction of byte sequences of interest. Once generated, signatures can be installed on existing “bump-on-the-wire” hardware appliances or coprocessors that provide line rate deep packet inspection capabilities. However, the principal shortcoming of existing IDS technology is the inability to handle attacks for which no known signature applies. These attacks, known as “zero-day” exploits, have the highest effective penetration during the vulnerability window between the time the exploit is unleashed and signatures are created and uploaded via

traditional means. At present, this vulnerability window can persist over a significant time scale.

On the other hand, various sensors exist, many in the experimental stage, for the detection and analysis of zero-day attacks. Typically, these sensors are either anomaly-based or protocol-adherence based. The primary difficulties with these technologies include high false alarm rates, sensitivity to the statistics of background traffic and system response times. Also, many of these sensors detect attacks only after a system has been compromised.

In this paper, we discuss a system which receives as input sets of packets identified as malicious and outputs signatures that can automatically be used as inputs to classical signature based matching systems. Of particular importance is our innovative mechanism for driving down the false alarm rate associated with the produced signatures via using readily available samples of benign, non-attack traffic.

2. APPROACH

In this section, we describe the approach and goals motivating our design.

Firstly, the ASG subsystem is not a sensor. It is not designed to perform the initial detection of an attack. It is designed to operate as a second stage to high quality front-end sensors. The desired result of our backend module is a set of signatures with a very low false alarm rate. High false alarm rates serve as a continuous denial of service and are therefore considered a major system risk. These signatures are designed to be compatible with simple signature matchers so that they can be easily disseminated to distant pattern matchers. The advantage of this is that pattern matchers are typically cheaper and have been implemented in ASIC and FPGA and can therefore keep up with network speeds. Ultimately, the purpose of signature dissemination is to automatically widen the perimeter of protection.

Finally, in the presence of polymorphic attacks (attacks that change from instance to instance), in order to find novel instances of an attack, one must produce signatures for attack invariants (parts of the attack that are constrained to remain constant). It is therefore important to produce signatures of minimum length that still also have low false alarm rates.

3. BENIGN TRAFFIC FILTERING

At the core of our innovation is the capability to drive down false alarm rates for signatures produced from instances of attacks.

Existing signature extraction systems[4] use only multiple examples of exploits to generate signatures. These systems do not directly address questions of false alarm rates. In addition, instances of exploits are hard to come by while benign traffic examples can be easily acquired.

3.1 Outline of Technique

The technique used by our system involves observing large numbers of network packets containing benign, non-attack traffic and extracting all substrings within a given range of lengths and storing them in efficient data structures. When presented with a set of attack packets, the system applies a technique involving benign traffic filtering where only substrings never seen in benign traffic are used to produce signatures.

3.2 Trie Based Filtering

The filtering operation can be performed efficiently using a *trie* data structure. Tries efficiently store large numbers of strings and allow for straightforward implementation of intersection, subtraction and other operations[1]. During a benign traffic training phase, we create a *trie* for each TCP or UDP service of interest which contains all of the extracted substrings from all packets. A simple example of such a structure is shown in Figure 1 and is labeled “Benign Traffic Trie”. A double circled node indicates the terminal node of a string. Note that all substrings of all lengths can be included, but usually they are limited to substrings within a length range.

A smaller *trie* is produced for a set of packets corresponding to an attack. Subtracting the benign traffic *trie* from the *trie* of the attack packets leads to a signature *trie* which contains substrings that can be used as signatures for that attack. A simple example is worked out in Figure 1. The Benign Traffic Trie in the example is extremely small for illustrative purposes.

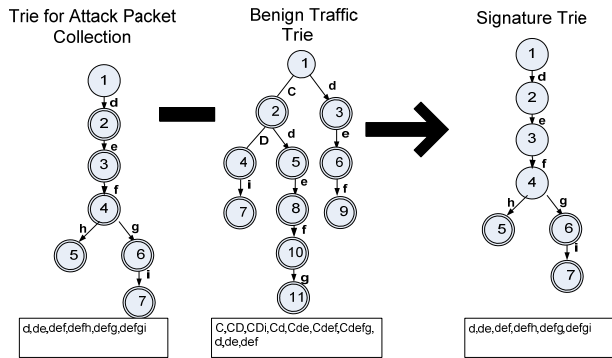


Figure 1. Signature trie created by subtracting benign traffic trie from attack packet trie.

A number of points should be noted with regards to this methodology. They include:

- 1) All substrings found in benign traffic which would lead

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CSIIRW '09, April 13-15, Oak Ridge, Tennessee, USA
 Copyright © 2009 ACM 978-1-60558-518-5 ... \$5.00.

to false alarm signatures are removed.

- 2) As more benign traffic is used for the production of the benign traffic *trie*, the system does a better job of filtering out less frequent occurrences of signatures that may occur in benign traffic (this is discussed at greater length below).
- 3) Since short signatures will be produced when possible, there is a better chance that attack invariants for polymorphic attacks will be captured.
- 4) If a service is highly structured, as time goes by strings will repeat themselves and the benign traffic *trie* will grow more slowly. If the service has random content, the *trie* will grow quickly.

4. PREDICTIVE BENIGN FILTERING MODELS

Predictive models are useful for comparative analysis of the performance of signature filtering schemes with expected results. We present here two simple analytical models to characterize benign traffic. The purpose of this section is to present a basic analytical model for relating the amount of benign traffic used for training to the expected false alarm rate for attack signatures from benign test traffic.

4.1 Models for Types of Service

In this section, we build two simple mathematical models for different types of services. Real world services can adhere reasonably well to one of these models but can also be a combination.

4.1.1 Syntactic Data Services

In this model, we assume that text is linguistic in nature in that it is composed of words that repeat themselves with varying frequency. We approximate word frequency using Zipf's law[2] with the s parameter set to 1 which is characteristic of English text. Under these assumptions, the probability of false alarm is given by:

$$P_{FA}(S_c) = \frac{P_{B/A}}{WK_Z} \frac{1}{1 + S_c}$$

S_c is the amount of traffic that has been collected for benign traffic training. P_{FA} is the false alarm rate per MB of test benign traffic seen. $P_{B/A}$ is the average percentage of each attack packet that contains benign traffic. \bar{W} is the average word size. K_z is the Zipf normalization factor.

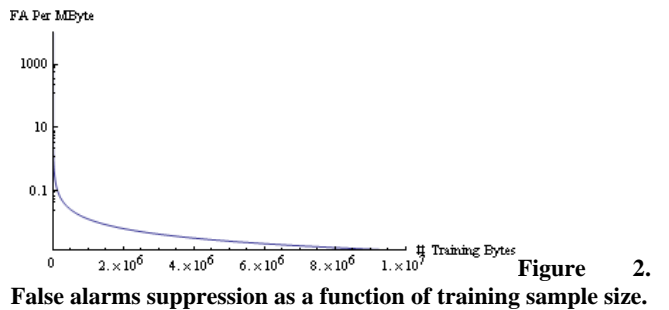


Figure 2. False alarms suppression as a function of training sample size.

This model predicts that the number of false alarms per MB of test data is inversely proportional to the number of MB of training data as illustrated in Figure 2.

Clearly, false alarm rates drop rapidly at first with reasonable amounts of training, but the tail requires more and more training to further lower the false alarm rate. This is reasonable, since we must wait longer and longer to capture the few uncommon strings that we haven't seen yet.

4.1.2 Random Data Services

In this second model, services can be modeled as streams of uncorrelated, random data. Encrypted services such as SSH or SSL fall in this category. Services which transport compressed data also have this behavior. We have derived a model of the form:

$$P_{FA}(S_c) \approx \frac{P_{B/A}}{2^{16|s|}}$$

P_{FA} is the false alarm rate per MB of test benign traffic seen. $P_{B/A}$ is the average percentage of each attack packet that contains benign traffic. $|s|$ is the length of a false alarm string in characters and S_c is the amount of traffic that has been collected for benign traffic training.

The above proportionality expresses the fact that as a random string becomes longer, the probability of seeing it again becomes exponentially smaller.

This result is not necessarily useful on its own since it is unlikely that a front-end sensor would be capable of detecting an attack in pure encrypted or compressed traffic. However, some services consist of mainly syntactical traffic with the occasional transfer of random data such as images which may cause false alarms.

As a rule, services that are predominantly high-entropy, such as SSH and SSL are not candidates for signature extraction. The ultimate outcome is that random traffic as part of attacks may create large numbers of signatures which will not be filtered out since they are unlikely to occur again in benign traffic, but the fact that they won't occur in benign traffic makes them unlikely to be a source of false alarms. Generally, short string sequences reoccur more frequently and are thus filtered out, while long random signatures survive since they occur most infrequently.

5. EXPERIMENTAL RESULTS

In this section, we examine a number of experimental results. We first examine some statistics describing the rate at which the number of novel N-grams in the benign traffic *trie* grows. We then examine the number of false alarms per MB of test data as a function of the number of MB of training data for a number of attacks launched using the Metasploit framework[3].

5.1 Benign Traffic Training Data Analysis

Figure 3 illustrates benign traffic database size in terms of number of unique inserted N-grams as a function of number of bytes of training data. Here, the N-gram size is fixed at 5. The data was taken from a representative sample of actual network traffic. The illustration shows a number of different types of services. The growth of the size of the *ssh* database, which is an encrypted service and therefore random, is explosive. On the other extreme, port 631 which is the Internet Printing Protocol, is syntactical in

nature with a small vocabulary and therefore flattens out very quickly. Another interesting finding involves http, port 80. Two plots are displayed. The first presents results for all packets to port 80. At one particular point, the plot rises dramatically. On examination of the data, it turned out that an image was being transmitted. The second plot removes the packets containing the image data to illustrate that pure protocol packets demonstrate a less dramatic increase in the number of unique N-grams.

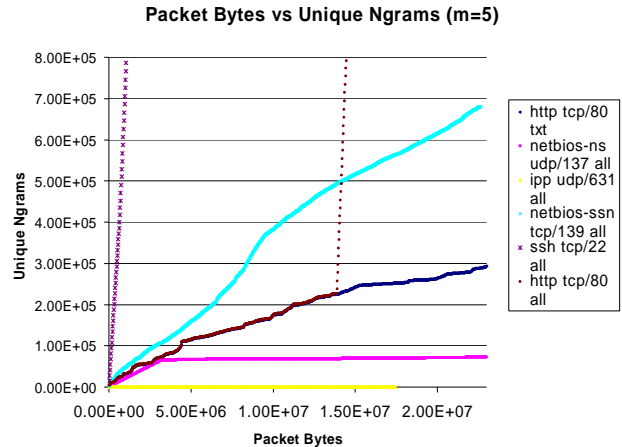


Figure 3. Number of unique N-grams as a function of number of bytes of training data. N-gram size is fixed at 5.

5.2 Pattern Matching False Alarm Rates

Figure 4 illustrates the false alarm probability on test traffic of a set of signatures for a Port 80 Apache attack[5] for various amounts of training data. We compare this to a best fit syntactic model curve. The fit is quite close with the exception that the actual data's false alarm rate decreases in discrete steps as various signatures are removed with more training. It should be noted that for this data set, with sufficient training, the false alarm rate actually reaches zero.

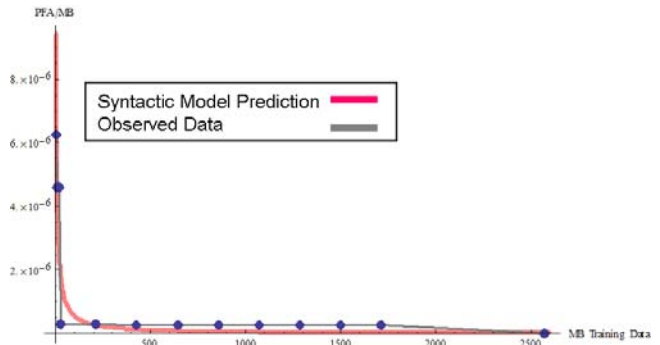


Figure 4. Apache attack signatures and FA rates as a function of benign traffic training. Test set contains 856 MB of packet payload.

In Figure 5, we show a port 139 Samba attack[6]. The Samba attack requires less training to achieve a zero false alarm rate. For the port 80 case, we require a training set about three times the size of the testing set to achieve zero false alarm, while with 139 the factor is only two.

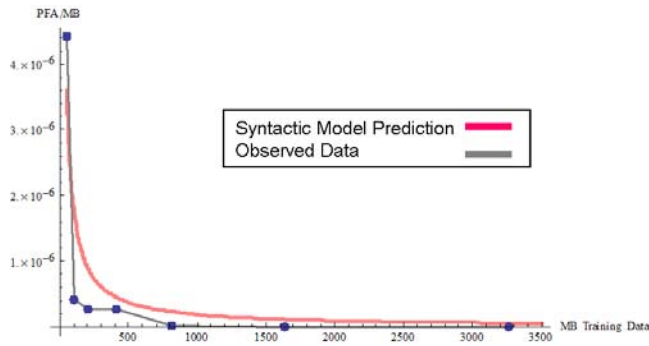


Figure 5. Samba attack signatures and FA rates as a function of benign traffic training. Test set contains 3,265 MB of packet payload.

6. ENTERPRISE-WIDE APPLICATION

The current ASG system has been combined with both anomaly based and honeypot type front-ends and has run in prototype environments. The produced signatures were tested for false alarm rate performance and behaved as expected.

An important future direction is the integration of Automatic Signature Generation technology with advanced sensors and high speed pattern matchers throughout the enterprise and out into the cloud. This is motivated by the simple observation that signatures produced locally can be distributed globally in order to extend the perimeter of protection. High quality but expensive and slow sensors can be used to capture instances of attacks from which signatures can be extracted. These signatures can then be distributed to enterprise gateways and to other installations (see Figure 6). In many cases, systems that are elaborately instrumented with anomaly detectors may end up being compromised as part of the attack process, with the net benefit that the attack is detected and its corresponding network traffic captured. The network pattern matchers, however, would be at some distance from sensitive hosts and servers and would block instances of re-infection far from their targets.

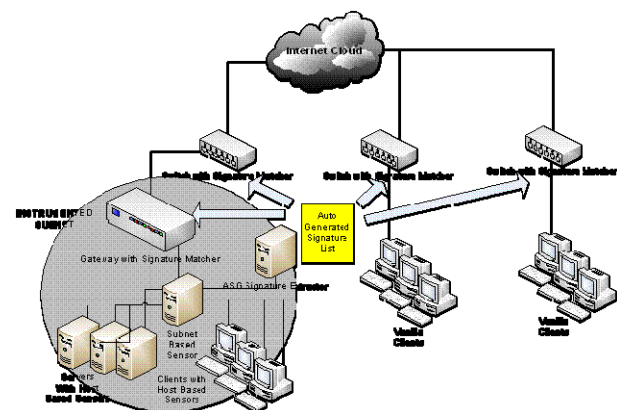


Figure 6. Distribution of signatures to other subnets.

While the signatures produced by ASG would be more numerous and less tailored than those produced by a human analyst, they would serve as initial zero-day protection. In addition, within reasonable limits, string matchers don't degrade in performance with large numbers of patterns, particularly those implemented in

hardware so that the large number produced would not be too great a concern. As time went on, the automatically generated signatures could be used as aids to human analysts in producing refined, hand-tuned versions which would presumably also decrease their number.

Secure, efficient automated signature distribution would thus constitute a significant step towards the goal of protection perimeter expansion.

A more distant goal would be to propagate signatures at a rate sufficient to contain a propagating threat, such as a worm. This would require high-speed, realtime responses on the part of various system components

7. CONCLUSIONS

We feel that using Automatic Signature Generation as an add-on component to various zero-day exploit detection systems has a number of important advantages:

- 1) As a separate component, it can integrate with a variety of sensors. As sensors improve their sensitivity, signatures will benefit.
- 2) Improvements in false alarm rates are achieved through training on benign traffic and do not need additional attack instances. Benign traffic is typically readily available.
- 3) Existing high-speed signature matching capabilities are leveragable for use with produced signatures.
- 4) Random data services may produce more signatures but do not increase false alarm rates.
- 5) Perimeter of protection capability allows the user to move defenses farther away from vulnerable assets.
- 6) Future possibility of protection outstripping propagating threats.
- 7) Ability to assist human analysts in drawing attention to attack portions of packets involved in an exploit.

8. REFERENCES

- [1] Rieck, K., Laskov, P. 2006. Detecting Unknown Network Attacks using Language Models. Third International Conference on Detection of intrusions and malware & vulnerability assessment, Berlin, Germany
- [2] Manning, C.D., Schütze H. 1999. Foundations of Statistical Natural Language Processing. MIT Press
- [3] <http://www.metasploit.com>.
- [4] Newsome, J., Karp, B., Song, D. 2005 Polygraph: Automatically Generating Signatures for Polymorphic Worms. 2005 IEEE Symposium on Security and Privacy
- [5] <http://www.osvdb.org/838>
- [6] <http://www.osvdb.org/4469>