Network Intrusion Detection Systems Evasion Techniques – an Investigation Using Snort

J.A.Ytreberg and M.Papadaki

Network Research Group, University of Plymouth, Plymouth, United Kingdom
e-mail: info@cscan.org

Abstract

Intrusion Detection Systems (IDS) provide an extra security precaution by detecting attacks that have bypassed the firewall. Snort IDS is one of the most widely used IDS (Siddhart, 2005). When a network is monitored by an IDS, attackers can send evading attack packets that will try avoiding detection. This research conducted experiments testing Snorts alerting capabilities when mutated attack packets where sent to a web server, using an IDS evasion tool called Nikto. It was found that Snort alerted for about half of the attack packets. Weaknesses in Snorts capabilities in detecting certain evasion attacks where found, which can be solved by creating customized rules. The research also proposes a new detection method for Snort, dividing large request strings into smaller ones, analyzing each of them against the rules. The total danger level of these combined strings could decide if Snort would alert for the request.

Keywords

NIDS, IDS, Evasion, Snort, Nikto, Rules, Detection, Networks, Monitoring

1 Introduction

As the world depend increasingly more on computer technology, so does the need for ways of securing these technologies. A network intrusion detection system (NIDS) usually lies behind the firewall of a security implementation, monitoring the network for attacking packets bypassing the security devices. Where malicious packets are found the NIDS will trigger an alert and log an event, it will not stop the packet in any way. Its purpose is to act like a smart network scanner, combining network capture packet techniques with a rule and alerting procedure. Intrusion detection systems is like a second line of defense (Anderson, 1980), and they all have their own strengths and weaknesses. Depending on configuration, placement, upgrade, etc. they all behave differently (Del Carlo, 2003).

NIDS are not 100% reliable and we do not expect it alerting all attacking packets, simply because this is a very difficult task. By configuring NID sensors to be too sensitive it would alert for too many packets which actually are legitimate network traffic (false positive). On the other hand, if the NID system is configured to be less sensitive we would most likely see attacking packets bypassing the NIDS unnoticed (false negative). What we do want is a balanced relationship between the number of false positives and false negatives, also called Crossover Error Rate (CER) (Chapple, 2003). According to Sodiya et al. (2004) IDS systems still produce too many false positives and false negatives and lack the efficiency sorely needed.

Evasion techniques are ways of mutating packets, forcing the NIDS not to trigger off an alert, simply because it thinks the packets are legitimate traffic. There are many different evasion techniques all designed to evade the IDS in the best manner possible, still producing the right end results at the targeted node (i.e. web server).

The research has conducted a series of experiments using Snort IDS, one of the most popular IDS on the market (Siddhart, 2005). The purpose of the research experiments were divided into three parts:

- Evaluation on how well Snort IDS responded to evasion attacks from Nikto
- How well does Snort function when its processor is fully engaged
- Improvement needed areas of Snort rules and detection engine

This paper will present some of the prior work in the area of IDS, followed by the experiments methodology and results. Finally the findings will be analyzed and discussed, ending off with a concluding part of this paper.

2 Background

A research conducted by Richard Barber in 2001 found that IDS applications analyzing protocols and packet behavior were more efficient than applications utilizing pattern matching techniques. The research also discovered that if network load exceeds 35%, NIDS can suffer in its performance and start dropping packets. If the load at the Network Interface Card (NIC) on the NIDS get to high, packet will start evading it, simply because it does not have the capacity to analyze them all (Barber, 2001).

There has been a previous research including Snort, done by Vlasti Broucek and Paul Turner conducted in 2004. This research used Snort to observe the hit rate, and false-positives generated when monitoring a web server over a two month period. It was found that Snort using fine-tuned rules, still was a subject to a number of false-positives. When discovering attacks it was found hard tracing these attackers back using the Snort logs. Only IP addresses are available for inspection. Snort also had severe difficulties in analyzing encrypted communication, especially tracing packets back to the attackers (Broucek, Turner, 2004).

One way of increasing the performance on Snort was discovered by Ian Graham and his research on how to use multi-processors in combination with Snort. The performance on Snort was greatly improved by balancing the load over several processors. Traffic will vary from different networks, but there will always be a performance gain if using multi-processors (Graham, 2006).

Another method of improving an IDS is the use of a keyword selection method, with the intention of making the IDS smarter by counting keywords and calculating the risk of the attack probability. This technique improved the hit rate of an IDS, without increasing the false alarms (Lippmann, Cunningham, 2000).

3 Methodology

The experiments will have three essential components in use: a web server, an attacking computer running Nikto, and the NIDS installed on a monitoring computer. All these devices are connected to each other via a hub, enabling the NIDS to monitor all traffic between the attacking computer and the web server. The NIDS will also be configured to have a NULL IP. This because of two important factors:

The NIDS should stay hidden to prevent attackers noticing it

The NIDS should not be able to send packets through the monitoring NIC, only receive.

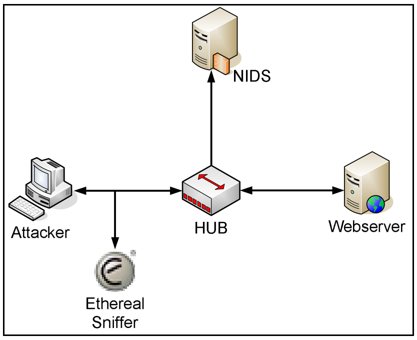

Figure 1: Experiment network topology and traffic flow

Figure 1 illustrates the network packet flow and the placement of the devices. The Ethereal sniffer is installed on the attacker computer for analyzing reasons. Notice that the NIDS will only receive packets, never allowed to send any. The NIDS will not have an IP address, but will still be able to sniff all traffic passing through the HUB (Kistler, 2003). The attacking PC will use the network tool Nikto, with its built-in evasion techniques. The NIDS will consist of a laptop installed with Snort IDS version 2.4.5. The research accomplished four different experiments, each one with dissimilar goals and test setup and configuration:

Test Description	Test Relevance
Test1 is an experiment where focus is upon Snort's detection abilities when submitted with mutated evasion packets.	Snort's performance can be compared to other IDS
Test2 is dedicated to finding out how Snort reacts when the CPU runs at maximum capacity (Snort recommends at least 300 MHz and 128 RAM for a low level traffic network) (Bull, 2002).	How would Snort perform if installed on a busy network or overloaded client?
Test3 is designed to find how new modified signatures affect the results. This experiment is much like Test1, with the exception of the adding of the new signatures.	Research upon writing own signatures and configuration options in Snort.
Test4 combines more evasion techniques to each attack packets using a method in Nikto which allows several evasion techniques at once. The goal of this experiment is to see if the research modified signatures still alert for complex packets involving several evasion techniques combined.	To what degree does several evasion techniques combined together affect the attacks, and the newly created signatures.

These experiments will be conducted in the same manner to produce the most accurate results. Where there are unexpected results, the tests will be run several times to produce more stabile results. This way we eliminate incidents with extreme results only occurring once.

4 Results

4.1 Results for Test1 – Snort's detection engine efficiency

Snort alerted differently based on the evasion attacks sent to the web server. The number of total attacks sent from Nikto varied by just a few packets per experiment, while the Snort detection varied much more frequently. Snort normally alerted for about 50% of the total evasion attack packets that where on the wire. The exceptions occured when Nikto used its evasion technique four "prepend long random string to request". Snort actually alerted for more than the total evasion packets sent on this occasion (see figure 2).

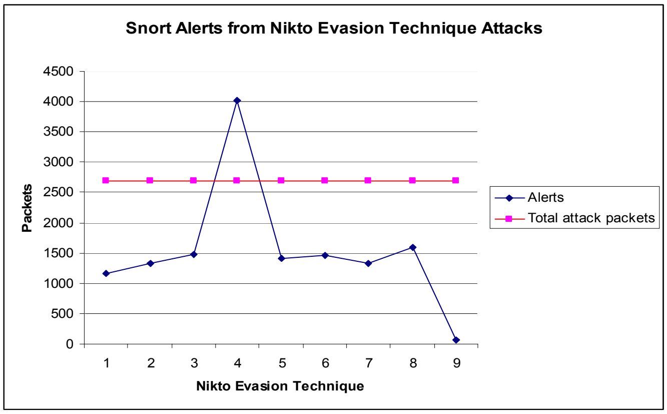

Figure 2: Snort's alerts compared to the total evasion attack packets sent

Figure 2 also show a relative small alert figure when Nikto used its evasion technique nine (9) "session splicing". This number is to be taken lightly, because of an incomplete experiment when testing this method (see chapter 5.1.9, NIDS evasion techniques - Thesis, Ytreberg, 2007).

4.2 Results for Test2 – Snort Detection under extreme conditions

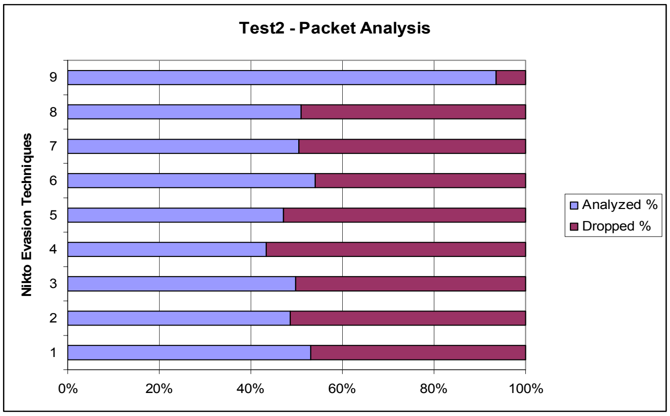

Figure 3: Snort's alerts compared to the total evasion attack packets sent

By using an option called verbose mode, making the NIDS output all packets received on its interface, the research stressed the CPU and memory of the NIDS. The goal was to see how Snort reacted when it had less processor capacity and memory than needed. As seen on figure 3, Snort started dropping packets after a few minutes of the experiment. Snort dropped around 50% of the packets when face upon the evasion attacks, with the exception of evasion technique nine.

4.3 Results for Test3 – Enhanced Signature Testing

The research created five (5) new signatures with the intention of improving Snort's hit rate. The signatures were created to improve areas where Snort usually alerts, but fails to do so because of some of Nikto's evasion techniques. The five new signatures looked like this (truncated):

```
content:"etc"; nocase; content:"/./"; content:"passwd"; nocase;
content:"/./"; content:"usr"; nocase; content:"bin"; nocase;
content:"etc"; nocase; content:"/./"; content:"hosts"; nocase;
content:"boot.ini"; nocase;
content:".passwd"; nocase;
```

After similar testing with the new signatures in place the results showed improved hit rate on the Snort detection (see figure 4).

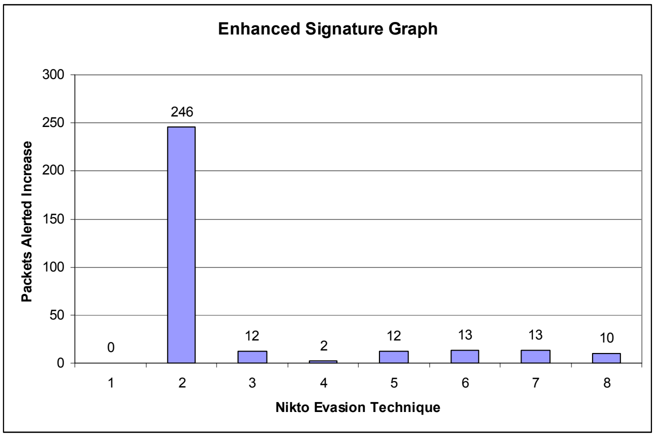

Figure 4: Alerts increase with new signatures

4.4 Results for Test4 – Combining Nikto Attacks

This test proved that the research's newly created signatures even work when Nikto mutates its attack packets by using several techniques per packet. Figure 5 illustrate a steep increase when Nikto combines method two and three, and a smaller increase when Nikto uses evasion method five (5) and six (6) combined.

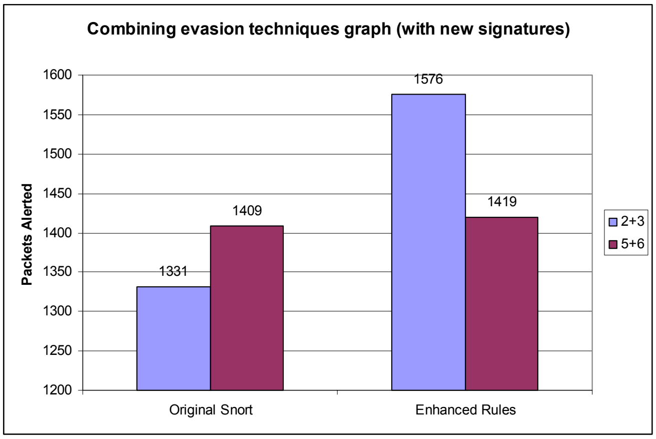

Figure 5: Packet increase alerts with new signatures

5 Discussion

The reason for Test1 - evasion technique four (4) triggering the high number of alerts, is because of two factors. Firstly, all packets look almost identical sending around 700 bytes of random strings before the actual request string in the end. This leads to Snort triggering off an alert per packet. Secondly, Snort will also alert on packets with malicious requests in the end, meaning each attack packet can get several alerts. This leads to the high number of alerts. When using evasion technique nine (9) the low number of alerts is simply because the experiments never ended. Session splicing requires many hours to complete, and the research had to abort after about five hours. Parallels can be drawn however, towards a NIDS on a busy network encountering session splicing attacks. If the NIDS is quite busy, it will not have the ability waiting for all session packets for reassembly, thus the packets will evade the NIDS.

If the NIDS is installed on a computer that is occupied with other duties as well as the intrusion detection system, it can be subject to dropping packets. As Test2 show the NIDS will drop packets if its resources are fully engaged. The most ideal placement of a NIDS would be on a separate node in a network, with no other applications and a processor of at least 1 GHz and memory of preferably 512 kb or more. With these resources the NIDS have the ability to withstand a high load on the network.

The area of which the research had focus on creating new signatures was towards Nikto's evasion technique two "add directory self-reference /./". These new signatures improved Snort's hit rate greatly when Nikto used this technique, and also when this technique was combined with most of the other evasion techniques. The command "nocase" were added to all signatures to prevent evasion techniques evading the signatures by using random casing. The problem occurred when Nikto used its evasion technique one (1) “random URI-encoding”. The new signatures did not alert when this technique was used, or any other in combination with this one. This is why figure 5 shows no increase in alert using evasion technique one. All the other evasion techniques had an improved hit rate when using the new signatures (exception session splicing, not enough testing).

The work done by Lippmann and Cunningham in 2000 was a new way of thinking, trying to make the IDS smarter. The research find that there should be created some sort of system where Snort instead of scanning the requests for incidents, analyzes the whole packet in a different way. By dividing the request content into smaller strings, and then to analyze each string a danger level classification (see table 1) could be the answer. If the total level of danger for the entire request exceeds a limit, an alert would be raised. This method would eliminate any incidents where one attack packet gets several alerts, but most importantly it would make the Snort IDS more dynamic. More dynamic in the way it divides and conquers any mutated packet trying to bypass its systems, looking for known dangerous format patterns in its rules.

Content	Danger level (1-10)
/./	4
cgi-bin	2
/./	4
passwd	8
Total:	18

Table 1: Content separation and danger classification

6 Conclusion

The research experiments provided evidence of Snort’s capabilities for detecting mutated packets with evading features. It also answered these following questions:

To what degree is a Snort NIDS capable of detecting evading packets from an attacking computer?

Snort without any special customized configuration and new rules will detect around 50% of evasion-only packets sent from Nikto. This number is to be taken lightly as it only concludes how Snort reacts to these kinds of evasion attacks. There are many other techniques that can be used to evade Snort as well.

How well does the NIDS detect alerts when its processing power is fully engaged?

When Snort NIDS CPU was overloaded it started dropping packets, but it was consistent in all experiments around this. It processed all packets it could and alerted them, until it suddenly dropped all further packets. The results were exactly the same as on Test1 up till the point where the NIDS had to drop packets. This is better than dropping a packet or two in between, because it would then be harder for administrators to notice that the NIDS CPU or memory was overloaded.

Is there a need for new and improved signatures?

Yes. There will always be a need for new signatures in Snort as attack tools and new techniques continue to develop and arise. Depending on the area of use, attack tools can be used to test a business IDS, and then to analyze the results. These results can then lead to new and customized rules that will protect the business optimally. Snort is very easy to configure and writing rules can be done in a couple of minutes. New signatures can improve a needed area greatly compared to the released Snort rules. Each NIDS has its own different network to protect, with all kinds of devices with lots of incoming requests. It is not an easy task to write common rules that is applicable for all these devices on Snort. The fact that Snort is an open source project means that it is the users that write the signature or the application would slowly die out.

7 References

Anderson, J.P. (1980), "Computer Threat Monitoring and Surveillance", (In Anderson, J.P. Technical report, Fort Washington, Pennsylvania)

Barber, R. (2001), "The Evolution of Intrusion Detection Systems – The Next Step", Computers & Security, Vol. 20, pp 132-145

Broucek, V., Turner, P. (2004), "Intrusion Detection: Issues and Challenges in Evidence Acquisition", Internal Review of Law Computers & Technology, Vol. 18 No.2, pp 149-164

Bull, J. (2002), "Snort's Place in a Windows 2000 Environment", www.snort.org/docs/snort-win2k.htm (Accessed 19 December 2006)

Chapple, M. (2003), "Evaluation and tuning an intrusion-detection system", searchsecurity.techtarget.com/tip/1,289483,sid14_gci918619,00.html?track=IDSLG, (Accessed 4 January 2007)

Del Carlo, C. (2003), "Intrusion Detection Evasion: How an attacker get past the burglar alarm", www.securitytechnet.com/resource/security/ids/1284.pdf#search=%22past%20and%20current%20work%20in%20evasion%20ids%20techniques%22, (Accessed 3 January 2007)

Graham, I. (2006), "Achieving Zero-loss Mutli-gigabit IDS – Results from Testing Snort® on Endace Accelerate Multi-CPU Platforms", www.touchbriefings.com/pdf/2259/graham.pdf (Accessed 3 January 2007)

Kistler, U. (2003), "Snort IDScenter 1.1 manual", www.engagesecurity.com/docs/idscenter/ (Accessed 30 December 2006)

Lippmann, R.P., Cunningham, R.K. (2000), "Improving intrusion detection performance using keyword selection and naural networks", Computer Networks, Vol. 34, pp 597-603

Rain Forest Puppy - rfp (1999), "A look at whisker's anti-IDS tactics", www.ussrback.com/docs/papers/IDS/whiskerids.html, (Accessed 30 December 2006)

Siddharth, S. (2005), "Evadion NIDS, revisited", www.securityfocus.com/infocus/1852, (Accessed 19 December 2006)

Sodiya, A.S, Longe, H.O.D and Akinwale, A.T. (2004), "A new two-tiered strategy to intrusion detection", Information Management & Computer Security, Vol. 12, No. 1, pp 27-44.