Evading IDS Detection

M.Batta and M.Papadaki

Centre for Security, Communications and Network Research,
Plymouth University, Plymouth, UK
e-mail: info@cscan.org

Abstract

Detecting intrusions is an arduous task, and although IDS technologies are getting better with time, it is still not possible to always detect intrusions accurately. IDS evasion techniques are becoming more complex and advanced, allowing them to operate under the radar of an IDS, and thereby bypass detection. The aim of this project is to investigate how easy it is to evade an IDS, and how different IDS configurations can influence its resilience to evasion techniques.

Keywords

Snort, Network Intrusion Detection System, preprocessors, datasets/pcap files, rulesets, fragmentation

1 Introduction

In an ever evolving world of computer systems and networks, complex security threats continue to surface rapidly. Latest firewalls and updated antivirus might be adequate solutions to the most common threats. But, the drawback of these protection mechanisms is that their main focus is on application behaviour and looks, not on examining the content. The attacker can bypass both the firewalls and antivirus simply by transferring data over firewall-accepted protocols or applications. This shows that both these mechanisms provide some level of security, but have their own limitations. Here arises the need for a security device which has the capability to perform all these functions plus to scan the contents of the traffic. Such a device is an Intrusion detection system (IDS) and it acts as a second line of defence (Anderson, 1980).

The purpose of this paper is to investigate how easy it is to evade an IDS, and how different IDS configurations can influence its resilience to evasion techniques. It performs an in-depth analytical study of the various tools and techniques employed to evade the IDS, intending to fortify the defence mechanism with a view to detecting of threats and attacks. It focuses on the pertinence and serviceability to ward-off untoward situations, and recommend pre-emptive measures to address the challenges posed by IDS. This research chooses to use Snort as a network intrusion detection system (NIDS) and enhance it with the latest rule set to detect any incoming attacks or threats.

2 Existing Research

In 2007, Jarle A. Ytreberg tested the resilience of Snort against certain IDS evasion tools such as Nikto. In his research, experiments were performed which tested Snort's alerting capabilities on sending mutated attack packets to a web server. Some weaknesses were discovered in Snort's capabilities to detect certain kinds of evasion attacks. But these could be dealt by creating customized rulesets. It was found that Snort was able to detect around 50% of the attacking packets sent from Nikto. All of the packets used a range of evasion techniques, which ideally should have been detected by Snort and alerted accordingly. When the computer was at its maximum processing speed, 50% of the packets were being dropped. The research also wrote five new rules, which on being implemented; Snort alerted about the dangerous evading packets and most evasion attacks. The research also proposes a new detection method for Snort which stated that the large request strings should be segmented into smaller strings, which would then be analysed individually against the rulesets (Ytreberg, 2007).

But, Snort has certainly improved with time and lived up to its reputation, which has led it to be one of the most popular and successful intrusion detection tools. This was evident by the research done by Ibrahim ALRobia, in 2010, who conducted research to test the durability of Snort against evasion attacks from Nikto. Unlike Ytreberg's research results in 2007, the results of this research revealed that Snort successfully detected all evasion techniques that were employed by Nikto and 104 alerts were flagged whether the test was conducted by single evasion technique or by combining multiple evasion techniques to strengthen the attack. Hence, the research concluded that Snort remained unaffected by the presence of any other application sharing the same processor. However, it also stated that such an improvement to Snort's detection ability was attributed to the preprocessors and Barnyard (ALRobia, 2010).

3 Test Setup and Configurations

In order to test Snort's efficiency, a number of tests, with different Snort configurations, would be performed. Results of these tests would be compared, deliberated upon and analysed; concluding with recommendations for achieving enhanced Snort's performance. In the course of detection process, the research will try and examine if Snort was successfully able to detect the evasion attempts for each dataset by looking into the results generated. Hence, the results of these experimental tests will enlighten users on how secure Snort really is, and what may or may not be its loopholes.

3.1 Download and relevance of pcap files

For a comprehensive investigation of Snort, there was a requirement for 'pcap' or packet capture files which could determine how reliable and efficient, in fact, the latest version of Snort IDS is, when exposed to attacks or threats. For this, it was essentially required that the pcap files consisted of built-in evasion techniques, especially devised to bypass the protection mechanisms of Snort. This would boost

the research to enlighten the users on how secure Snort really is, and what may or may not be its loopholes.

After immense research and many thanks to my supervisor Dr. Maria Papadaki, 23 pcap files were found with Advanced Evasion Techniques (AETs). With these pcap files, this 'Antievasion' website claims to have "discovered a new, dangerous set of evasion techniques that threaten to penetrate even the most sophisticated networks." All these packet capture files and their details are enlisted on the 'Antievasion' website, from where these can be downloaded and used (Antievasion, 2011).

3.2 Modifications made to Snort's default configuration

The modifications that are done to Snort's default configuration would be based on the configuration changes suggested in a blog entry of Joel Ester (Ester, 2011).

Modification 1: Enabling all rulesets

Following are the rulesets which were enabled in this research. In the default configuration of Snort, these rulesets were commented-out (#) or were not in use.

```
# Policy related rules:
 include $RULE_PATH/policy.rules
 include $RULE_PATH/community-policy.rules
 include $RULE_PATH/porn.rules
 include $RULE_PATH/community-inappropriate.rules
 include $RULE_PATH/chat.rules
 include $RULE_PATH/multimedia.rules
 include $RULE_PATH/p2p.rules
 include $RULE_PATH/community-game.rules
 include $RULE_PATH/community-misc.rules
# Extremely chatty rules:
 include $RULE_PATH/info.rules
 include $RULE_PATH/icmp-info.rules
 include $RULE_PATH/community-icmp.rules
```

Modification 2: DCE/RPC2 preprocessor

Following changes are made to the default DCE/RPC2 preprocessor configuration:

```
preprocessor dcerpc2: memcap 102400, events [co ]
preprocessor dcerpc2_server: default, policy WinXP, \
detect [smb [139,445], tcp 135, udp 135, rpc-over-http-server 593], \
autodetect [tcp 1025:, udp 1025:, rpc-over-http-server 1025:], \
smb_max_chain 3, smb_invalid_shares ["C$", "D$", "ADMIN$"]
```

Modification 3: RPC_DECODE preprocessor

Modifications made to default RPC_DECODE preprocessor configuration:

```
preprocessor rpc_decode: 111 32770 32771 32772 32773 32774 32775 32776 32777
32778 32779 no_alert_multiple_requests no_alert_large_fragments
no_alert_incomplete
```

Modification 4: SSL Preprocessor

Following are the modifications made to default configuration of SSL preprocessor:

```
preprocessor ssl: ports { 443 465 563 636 989 992 993 994 995 7801 7702 7900
7901 7902 7903 7904 7905 7906 6907 7908 7909 7910 7911 7912 7913 7914 7915
7916 7917 7918 7919 7920 }, trustservers, noinspect_encrypted
```

Modification 5: HTTP_PORTS

The updated configuration of HTTP_PORTS reads:

```
portvar HTTP_PORTS
80,311,591,593,901,1220,1414,1830,2301,2381,2809,3128,3702,5250,7001,7777,
7779,8000,8008,8028,8080,8088,8118,8123,8180,8181,8243,8280,8888,9090,
9091,9443,9999,11371]
```

Modification 6: ORACLE_PORTS

The modified and updated Oracle configuration line now reads like this:

```
portvar ORACLE_PORTS 1024:
```

Modification 7: stream5 preprocessor

Default configuration of stream5 preprocessor is modified to:

```
preprocessor stream5_global: max_tcp 8192, track_tcp yes, track_udp no
# preprocessor stream5_tcp: policy first
preprocessor stream5_tcp: policy first, use_static_footprint_sizes, detect_anomalies,
overlap_limit 1
# preprocessor stream5_udp: ignore_any_rules
```

This modifications done to the stream5 preprocessor's default configuration is based on the document written by Richard Bejtlich on "Snort's Stream5 and TCP overlapping fragments". These changes will cause Stream5 to alert when it sees at least one overlapping TCP segment (Bejtlich, 2007).

4 Results and analysis

Beyond doubt, the test results proved that modifications made to Snort's default configuration have indeed enhanced Snort's capability of detecting and alerting evasion attempts. The 22 datasets acted as a polestar to keep a vigilant eye on Snort's performance after each and every modification. The following graph 1 recapitulates the results obtained by performing the various modifications in Snort's default configuration:

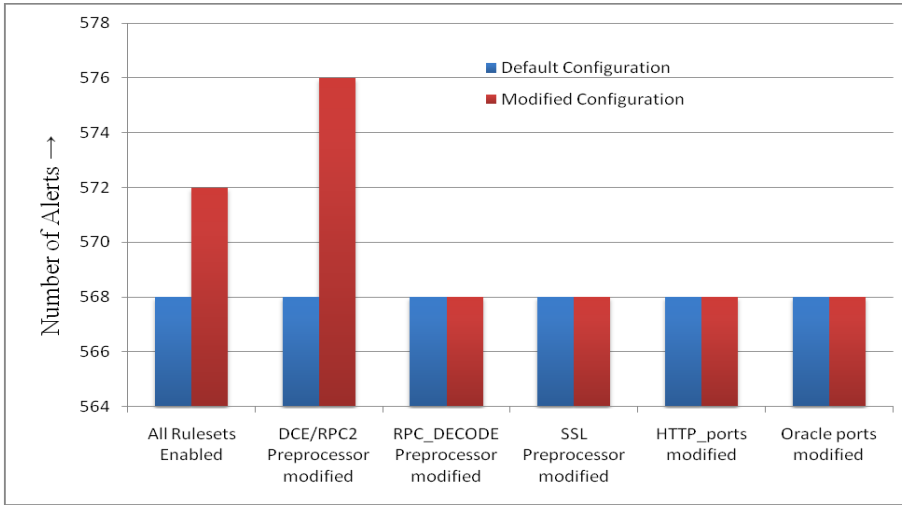

Figure 1: Comparison of alerts flagged with Snort's default configurations v/s all modifications done

As seen in the above figure, Snort's default configuration flagged only 568 alerts. However, when all the rulesets present in Snort's configuration were enabled (modification 1), it led Snort to flag 572 alerts; indicating an improvement in Snort's performance. Furthermore, modifications made to DCE/RPC2 preprocessor (modification 2) bettered Snort's performance all the more; total number of alerts triggered being 576. But, by far, outstanding and most encouraging results have been accrued by modifications done to stream5 preprocessor (modification 7); success of which can be accredited to Richard Bejtlich. Here, the total number of alerts showed a phenomenal increase to a staggering 2756, because of which results of stream5 preprocessor modification are not displayed in the above graph as it is beyond its scale. Therefore, the two most effective modifications established by means of this research are stream5 preprocessor (modification 7) and DCE/RPC2 preprocessor (modification 2).

Besides, the above graph also shows that there have been some modifications which have not affected Snort's performance at all; keeping the number of alerts same as in

Snort's default configuration. The modifications which have flagged 568 alerts, same as the default configuration are:

- RPC_decode preprocessor (modification 3)
- SSL preprocessor (modification 4)
- HTTP_ports (modification 5)
- Oracle_ports (modification 6)

Nonetheless, as also mentioned in the previous chapter, it should be carefully noted that in spite of the fact that these modifications have not enhanced Snort's capabilities of detecting evasion attacks specific to these datasets, they could prove to be of utmost significance in live commercial environments; especially due to ever evolving techniques of evading Intrusion Detection Systems.

The core reason behind the improved detection in Snort can be accredited to the highly significant modifications made to the default configuration of Snort. For instance, stream5 preprocessor modification flagged a total of 2756 alerts compared to 568 alerts in the default configuration. To be more precise, the dataset 'CVE-2003-0533-EvasionTCPSegment3-SMBDecoyWrites5 SMBResourceSegment33.pcap' flagged a mere 5 alerts in the default configuration whereas, after the modifications the same dataset flagged a staggering 1950 alerts. This is because, as a consequence of the modification, Snort was able to detect the most characteristic evasion technique 'Fragmentation Overlap'. Snort's default configuration was unable to detect this evasion technique because the default configuration uses the keyword 'policy_first' meaning that Snort will favour the first overlapped segment and the 'overlap_limit' is set to zero by default meaning that there is no limit to the number of overlapping packets per session. Hence, the default configuration does not inspect the contents of overlapping packets and simply considers the first of the overlapping packets. However, the modified configuration performs with flying colours and flags tremendous number of alerts because the keyword 'policy_first' is disabled (commented-out) plus the 'overlap_limit' is set to 1 which limits the number of overlapping packets per session to one. Therefore, nothing goes undetected and Snort scrutinises each and every packet overlap.

In spite of all the achievements of the experiments conducted, there were a few limitations of the research. Due to time constraints, the research was unable to deeply analyse the aspect of false positives. Also, an in-depth analysis of the contents and characteristics of the packets in the datasets could not be performed. Thus could have shed more light on the behaviour of Snort under different configurations and the legitimacy of the alerts generated.

5 Conclusion, Recommendations & Future

The aim of the research to improve detection capabilities and performance of Snort can said to be accomplished because the research has demonstrated awareness of intrusion detection technologies as well as IDS evasion techniques; designed and implemented tests that investigated the evasion resilience of Snort. Most importantly,

the research has meticulously tested and validated that the modifications made to Snort's default configuration file indeed proved to be beneficial by increasing the total number of alerts triggered.

According to the findings of this research, Snort would exhibit its maximum performance and would be most effective as well as effective in tackling evasion attempts provided all the rulesets are enabled along with all the preprocessors, suggested in the test results in previous chapter, are modified. The evidence of this is that Snort recorded the maximum number of alerts (2768) when it was run with this combined modification (Rulesets + Preprocessor).

Specific recommendations of this paper would be to preferably set 'Overlap_limit' to 1. This way it would become almost impossible to use 'fragmentation overlap' as an evasion technique to bypass Snort's detection, since Snort would monitor even a single overlapping TCP segment. Hence, nothing goes undetected. Moreover, all the mentioned changes in configuration should be adopted in order to stay at par with advancing evasion techniques; enable greater detection functionality and improve Snort's performance.

However, there are still a few stones left to be turned. One of the key areas for examination by the new researcher would be to delve into the phenomena of false positives. There could be an outside chance of the additional alerts triggered due to changes in Snort's default configuration, being false positives. Harmless enough, false positives could be a real nuisance as they bring down Snort's performance considerably.

6 References

ALRobia, I. (2010). "Evading IDS Detection: An extensive research about Snort IDS ability to detect Nikto evasion techniques", Masters Dissertation, University of Plymouth, UK.

Anderson, J.P. (1980), "Computer Threat Monitoring and Surveillance", http://seclab.cs.ucdavis.edu/projects/history/CD/ande80.pdf, (Accessed on 30 January 2011)

Antievasion Web Site (2010), "Technical details of the first 23 AETs and pcap files", http://www.antievasion.com/principles/principles/part-3, (Accessed 01 April 2011)

Bejtlich, R. (2007), "Snort's Stream5 and TCP overlapping fragments", http://searchsecuritychannel.techtarget.com/tip/Snorts-Stream5-and-TCP-overlapping-fragments?ShortReg=1&mboxConv=searchSecurityChannel_RegActivate_Submit&, (Accessed 16 July 2011)

Ester, J. (2011), "New Rule Pack and check your Snort.conf", http://blog.snort.org/2011/01/new-rule-pack-and-check-your-snortconf_04.html, (Accessed 22 July 2011)

Holland, T. (2004), "Understanding IPS and IDS", http://www.sans.org/reading_room/whitepapers/detection/understanding-ips-ids-ips-ids-defense-in-depth_1381, (Accessed 23 November 2010)

Kelley, B. (2006), “Databases, Infrastructure and Security”, http://www.sqlservercentral.com/blogs/brian_kelley/archive/2006/05.aspx, (Accessed 25 November 2010)

Magalhaes, R. (2006), “Host-Based IDS vs Network-Based IDS (Part 1)”, http://www.windowsecurity.com/articles/Hids_vs_Nids_Part1.html, (Accessed 25 November 2010)

Roesch, M. (1999), “Snort – Lightweight Intrusion Detection for Networks”, http://www.usenix.org/event/lisa99/full_papers/roesch/roesch.pdf, (Accessed 29 January 2011)

Rowland, C. “Intrusion Detection System”. United States Patent (Patent No. US 6,405,318 B1, 11 Jun 2002), http://www.google.com/patents?hl=en&lr=&vid=USPAT6405318&id=9-sLAAAAEBAJ&oi=fnd&dq=18.%09Rowland,+C.+%E2%80%9CIntrusion+Detection+System%E2%80%9D&printsec=abstract#v=onepage&q&f=false, (Accessed 24 November 2010)

Sourcefire Web Site (2009), “Snort® Threat Prevention Components”, http://www.imerja.com/files/file/White_Papers/Sourcefire/Snort%20Threat%20Prevention.pdf, (Accessed 19 July 2011)

Ytreberg, J. (2007). “Network Intrusion Detection Systems Evasion Techniques: an Investigation using Snort”, Masters Dissertation, University of Plymouth, UK.