EVOLUTIONS OF EVASION TECHNIQUES AGAINST NETWORK INTRUSION DETECTION SYSTEMS

Jorge Maestre Vidal, Jaime Daniel Mejía Castro, Ana Lucila Sandoval Orozco, Luis Javier García Villalba

Group of Analysis, Security and Systems (GASS)
Department of Software Engineering and Artificial Intelligence (DISIA)
School of Computer Science, Office 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases s/n, Ciudad Universitaria, 28040 Madrid, Spain
{jmaestre, j.mejia, asandoval, javiergv}@fdi.ucm.es

Abstract

Abstract –With the rise of the information society, new types of attacks and strategies to make detection more difficult are discovered every day. Currently, Network Intrusion Detection Systems can be avoided in different ways, so it is essential to know the different existing evasion techniques when planning the defense perimeter of a network. This paper explains the most historically representative evasion strategies. It also analyzes the current impact of these threats, emphasizing those that currently still pose a risk, thus allowing researchers to develop more effective defense mechanisms.

Keywords - Network Intrusion Detection System, NIDS, Evasion Techniques, Attacks.

1 INTRODUCTION

The NIDS (*Network Intrusion Detection System*) is a kind of IDS (*Intrusion Detection System*) [1]. It analyzes traffic on a network looking for any malicious activity. As initial consideration, at any time,it must be taken into account that the development in the field of NIDS has been fueled by the emergence of new forms of attacks capable of avoiding those systems, and that at any time some variation of an old technique may appear,which is capable of jeopardizing the current systems.

In 1998, Ptacek et al. [2] first proposed the existence of these NIDS evasion techniques. Such evasion techniques involved the exploitation of ambiguities between the NIDS and the TCP/IP protocols. They also consisted in the application of various Denial of Service strategies to block the NIDS´s functionality.Initially,intrusion detection was based in the search for attack signatures. But the rapid proliferation of malware, led to the need to address unknown threats: the Zero Day Attacks. Consequently a new detection strategy appeared which was based in the anomalies search.

But with the appearance of anomalies based NID [3], emerged new NIDS evasion techniques. Such new strategies were inspired by malicious network traffic injected by the attacker trying to look like legitimate network traffic and consist mainly in the malware obfuscation mechanisms: cipher, oligomorphism, polymorphism and metamorphism.

The intruder may have physical access to the medium where is placed the victim system: In recent times Link Layer vulnerabilities have also been exploited for placing the malicious entity in positions that do not require access through the NIDS. Although such vulnerabilities seem obsolete, IPv4 and IPv6 still contain dangerous variants. The attacker can also exploit other vulnerabilities in certain application layer protocols to disguise his nature.

The network protected by the NIDS must be accompanied by other security systems. Security tools as HIDS (*Host Intrusion Detection System*), firewalls or honeypots may complement those cases where NIDS is not as effective. Furthermore, to reach the highest protection, the NIDS must be connected to an alert correlation system and must be consistent with the rest of the network.

This paper is structured in three sections, and the first is this introduction. The second section describes the different NIDS evasion techniques, detailing the most representative examples. The last section contains the conclusions.

2 NIDS EVASION TECHNIQUES

Figure 1 shows a classification of the main NIDS evasion techniques. Is described below each of them:

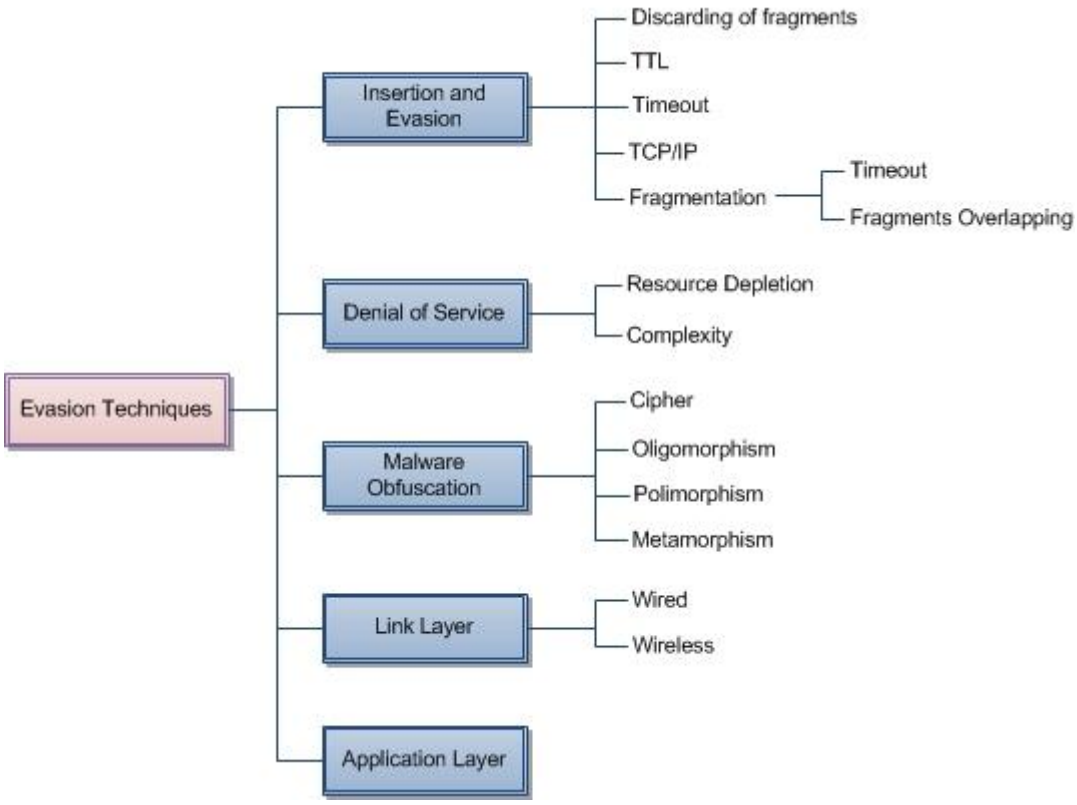

Figure1: NIDS Evasion Techniques.

Insertion and Evasion: The Insertion and Evasion techniques, along with Denial of Service attacks were the first IDS evasion techniques. Insertion and evasion attacks are based on the lack of coherence and information among the NIDS and the protected system. In particular, the Insertion attack takes advantage of the supposition that specific fragments will be processed by the target system but in reality will be discarded. The sequence of fragments which NIDS gets is not equal to the original attack. When the system gets rid of such fragments, the final sequence corresponds to the threat. Then the full attack is rebuilt to reach the victim and cannot be detected by the NIDS.

The evasion attack occurs exactly in the opposite case: when the NIDS discards certain useless padding packets. It lets the full attack to the target system pass. Such attacks can be carried out by exploiting a simple difference between the restriction levels on NIDS discard policies and the target system, or by using more advanced techniques. For example, if the attacker knows the network topology may change the TTL (*Time to Live*) of the fragments, to make some of them disappear before reaching the victim system. It is also possible to exploit vulnerabilities in network protocols such as TCP/IP, by using malformed headers or integrity incorrect codes, allowing rid of fragments [4].

One of the functions of the IP protocol is to take care of the fragmentation of packets in order to allow its flow on subnets with MTU (*Maximum Transfer Unit*)smaller than the packet length. The routing of the fragments also means that fragments rarely reach the destination in their original order. A very basic technique of evasion could be mixing the fragments. It causes the NIDS to ignore them but luckily this technique is not very effective. The packet fragmentation can also lead to denial of service attacks, flooding the target system with non-joinable fragments. But this method is no longer effective because of policies established to discarding unconnected fragments after a certain period of time. However, the attacker may exploit the fragments Timeout field, and the ambiguity on the fragments overlapping policies between the NIDS and the victim system [2] to complete successful attacks.

Denial of Service: The denial of service attacks [2] tries to exploit the availability of a resource through their disproportionate consumption. If a denial of service attack hits a NIDS, several things can happen: as first, and in the best case, the compromised system will stop servicing the network. It prevents malicious traffic from reaching the protected systems. On the other hand, some NIDS devices may act on Fail-Open mode. It allows the malicious traffic flow through it. The denial of service attacks may threaten several objectives: basically computational resources such as bandwidth, disk space or processor time. It can also alter control information such as routing paths or TCP session information. Ultimately, it could threaten the physical layer and the link layer.

There are many techniques for denial of service [5], but possibly the most specialized against NIDS is the alert flood attack. In 2001 the first tool for NIDS evasion appeared based on the Alert Flood, called *Stick*. It works by saturating the control channel that connects the IDS with the operators, through an injection of false alerts, thus denying service. In the work of G. Tedesco et al. [6] Alert Flood variants were explained not just for the denial of service, but also to obfuscate the true nature of the attack in a tangle of false attacks. Such actions are quite common today.

The denial of service attack can also be done by exploiting the complexity of the own NIDS's algorithms and other design features. These attacks consist in ensuring that certain parts of NIDS make code computationally more expensive. Backtracking algorithms are a common target, because its medium computational cost is polynomial, but in the worst cases, it could consume resources NP-order complex (NP-Hard) [7]. These kinds of attacks are very effective against rule-based NIDS. In them the resources for assessments could become saturated, preventing proper evaluation of some predicates. When this happens the NIDS can be pierced by the malicious network traffic. The solution to this type of attacks usually has to be in the development phase of the NIDS. It is difficult to give specific guidelines to avoid them. There are specific vulnerabilities for each system, so it is advisable to study in detail each algorithm implementation. You must work with the secure code and work with the latest stable versions of the software/firmware.

Malware Obfuscation: Malware obfuscation is to make a program difficult to understand. Although initially the origin of the software obfuscation was to protect intellectual property of developers, currently is also used for malicious purposes. Generating an obfuscated malware is the creation of a new malware. This new malware preserves its original functionality but the code is different. The NIDS can hardly detect it. The malware obfuscation started with simple programs with encrypted or compressed malicious code to evade detection. Since then, progress has been made in three methodologies: oligomorphism, polymorphism and metamorphism [8].

Besides encrypting the body of the attack, it also became necessary to disguise the decryption module to obtain a completely undetected malware. These modules started to be easily recognized by the NIDS based in signatures. The attackers had to start applying oligomorphism techniques for hiding attacks. These techniques are based in generating different encryption modules, instead of always the same code. The first oligomorphic malware known was called Whale, in 1990. Whale randomly selects one decryption module of a decryption module list. However, the progress by NIDS based in anomalies turned oligomorphism into an obsolete strategy

The first known polymorphic virus also appeared in 1990. It was called 1260 and was developed by Mark Washburn. 1260 was based on the realization of random changes on the encryption module, which turned it almost unique. This type of mutation is precisely where the difference lies with his predecessors. One of the best known polymorphic attacks is Mimicry. Mimicry converts malicious code in system calls by padding [9] techniques. In this action, Mimicry is trying to be invisible to the NIDS. Also uses Sled techniques [10] to reach certain regions in memory to be executed. At the time, Mimicry was particularly resistant against the system call monitor, but with the arrival of the anomaly based NIDS[11], it became easier to detect

In contrast to previous strategies, metamorphic attacks has not decryption module. A metamorphic malware is known for being able to mutate its own body. This will generate a different anatomy after each mutation. In 1998, the first metamorphic virus known as ACG (Amazing Code Generator) appeared. ACG was originally developed for DOS and until 2000 it was not adapted to 32-bit architectures (W32.Appartition) [12]. Today these are present on any platform. Metamorphic malware detection can become too complicated. Some versions end up having a totally different code from the original code. The anomaly-based NIDS is effective in some advanced mutations. But totally different

mutations can be really hard to detect. At this point the NIDS may not be sufficiently effectives. Therefore they need to complement its functionality with other defense systems as HIDS

Link Layer Vulnerabilities: In the link layer, the system could be evaded by any previously mentioned way. Also in this layer there is the risk that the attacker may physically access the LAN. The attacker could know the topology of the network. The enumeration of a LAN is not difficult because the large number of services which allow it. Therefore, the attacker may know the MAC address and the address of NIDS victim system. With this information simple phishing techniques could be used which could jeopardize the system. Usually in this kind of evasion methods the attacker is allowed to send padding fragments to the NIDS by creating false routing tables (*IPv4 ARP Spoofing* or *IPv6 Neighbor Discovery Spoofing*). In this way they are placed as a *Man-in-the-Middle*[13]among the NIDS and the victim system, and could inject malicious traffic is placed without the NIDS protection.

Another way to attack a NIDS under the link layer is by denial of service. One of the most common ways of doing this is by ARP flood attack. This strategy involves sending ARP reply packets falsely associated to the victim system. The victim may come to consider that it has more routes than those actually have. This will lead to redirecting traffic to nonexistent places. There are also Denial of Service methods exclusives for this layer. For example the *jamming* attacks, which operate against wireless protocols by generating noise by sending indiscriminate unauthenticated packages to the various stations of the network. Another example is the creation of network sinks, known as *Sinkhole* attacks. It consists in the attraction of all the network traffic to the NIDS region, causing saturation.

***Application Layer Vulnerabilities*:** Each application layer protocol has vulnerabilities that could make an attack difficult to be detected by the NIDS. The malware could avoid the security systems because specific languages syntax. Also the ambiguity when representing arithmetic operations can be used by the attacker. Additionally, multimedia data such as images or video employ specific compression techniques. These compression techniques can hide the attack. For example it works well with buffer overflow attacks or malicious code such as obfuscated Viruses or Trojans. Therefore it is an obfuscation strategy that instead of involve low-level malware characteristics as explained above, exploits application-level characteristics.

A popular example of application layer vulnerabilities is to send certain Web attacks by exploiting the HTTP [2] protocol vulnerabilities. The NIDS can be avoided by using simple techniques to convert URLs to hexadecimal. It is also possible to exploit CGI vulnerabilities by exchanging the used methods (GET or HEAD). The attacker can also transform Unicode content to hexadecimal content, generating a large amount of different signatures. As a last example, the NIDS can be avoided by placing instructions parameters in a different place than the html request. These examples illustrate that there are many ways to exploit these vulnerabilities. Most attacks on this protocol are well known and are based on different ways to obfuscate malicious web content, exploiting vulnerabilities that do not involve encryption. The attacker usually attempts to perform a combination of them.

Another instance is the SQL injection. SQL injection occurs when the attacker inserts malicious SQL statements in a text field that interacts with a database. It produces an anomalous result rather than the results of the original query. It usually targets the privilege escalation, data collection through consultation enforced or enumerating the database. The difficulty of detecting SQL injection attacks is that the attack syntax is similar to SQL queries. This makes malicious traffic looks like legitimate traffic.

3 CONCLUSIONS

A NIDS is truly effective if it detects malicious activities and it is able to detect each of the evasion techniques. These threats can take advantage of the exploitation of vulnerabilities in any service or network layer, and they are constantly adapting to new technologies evolutions. Therefore a NIDS must be consistent and remain synchronized with the protected system. The protected system must use security protocols whenever possible. The existence of obfuscated malware must be taken into account when designing NIDS. Finally it must also be able to defend evasion techniques in application layer. As we have seen, the use of an alert correlation system implements a rapid response and defeats attacks by flooding alerts.

It is also interesting to know that such attacks aim to understand the mind of the attacker and to practice a better defense. As already mentioned, the NIDS is a good way to protect a system, but not

the only one. Against certain types of attacks it will be necessary to supplement the security perimeter with other security systems as firewalls or HIDS.

ACKNOWLEDGMENTS

This work was supported by the Agencia Española de CooperaciónInternacionalpara el Desarrollo (AECID, Spain) through AcciónIntegrada MAEC-AECID MEDITERRÁNEO A1/037528/11.

REFERENCES

[1] C. A. Catania, C. G. Garino.Automatic Network Intrusion Detection: Current Techniques and Open Issues. *IEEE Computers & Electrical Engineering*, Vol. 38, No. 5, pp.1062–1072, September2012.

[2] T. H. Ptacek, T. N. Newsham.Insertion, Evasion and Denial of Service: Eluding Network Intrusion Detection. *Technical Report*, Secure Networks Inc Calgary Alberta, USA,January 1998.

[3] V. Jyothsna, V. V. R. Prasad, K. M. Prasad. A Review of Anomaly based Intrusion Detection Systems. *International Journal of Computer Applications (IJCA)*, Vol. 28, No. 7, pp. 26-35,August2011.

[4] U. Shankar, V. Paxson. Active Mapping. Resisting NIDS Evasion without Alerting Traffic.*In Proceedings of the Symposium on Security and Privacy*, The Claremont Resort, Oakland, California, USA,pp. 44-61,May 2003.

[5] J. Mirkovic, P. Reiher. A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. *In Proceedings of the annual conference of the Special Interest Group on Data Communication (SIGCOMM)*, Portland, USA.Computer Communication Review, Vol. 34, No. 2, pp. 39-53, April 2004.

[6] G. Tedesco, U. Aickelin. Strategic Alert Throttling for Intrusion. *In Proceedings of the 4th International Conference on Information Security (WSEAS)*,Tenerife,Spain, pp. 246-251,May 2005.

[7] S. Arora, D. Karger, M. Karpinski. Polynomial Time Approximation Schemes for Dense Instances of NP-hard Problems.*In Proceedings of the 27th annual ACM symposium on Theory of computing (STOC)*, New York,USA,pp. 284-293, 1995.

[8] P. O'Kane, S.Sezer, K. McLaughlin. Obfuscation: The Hidden Malware.*IEEE Security & Privacy*,Vol. 9, No. 5, pp. 41-47, October 2011.

[9] Q. Li, D. M. Chiu, J. C. Lui. On the Practical and Security Issues of Batch Content Distribution via Network Coding. *In Proceedings of the 14th IEEE International Conference In Network Protocols (ICNP)*, Santa Barbara, California, USA, pp. 158-167,November2006.

[10] X. Zhang, H. Liu. Design and Implementation of Remote BufferOverflow and Implanted Backdoor. *In Proceedings of the National Conference on Information Technology and Computer Science (CITCS)*,lanzhou, China, pp. 158-167,October2012.

[11] K. Wang, J. J. Parekh, S. J. Stolfo. Anagram: A Content Anomaly Detector Resistant to Mimicry Attack. *In Proceedings of the 9th International Symposium on Recent Advances in Intrusion Detection (RAID)*, Hamburg, Germany. Lecture Notes in Computer Science, Vol. 4219, pp. 226-248, September 2006

[12] M. Polychronakis, K. G. Anagnostakis, E. P. Markatos. Network-level Polymorphic Shellcode Detection using Emulation. *In Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA)*, Berlin, Germany.Lecture Notes in Computer Science, Vol. 4064, pp. 54-73,July2006.

[13] J. Wells, D. Hutchinson, J. Pierce. Enhanced Security for Preventing Man-in-the-Middle Attacks in Authentication, Data Entry and Transaction. *In Proceedings of the 6th Australian*

Information Security Management Conference (AISM), Perth, Australia, pp. 1-14,December2008.