An Achilles' Heel in Signature-Based IDS: Squealing False Positives in SNORT

Samuel Patton[1] William Yurcik David Doss

Department of Applied Computer Science
Illinois State University
Normal IL USA
[1]work done jointly with Cisco Systems Inc. and Millenium Computing
{stpatto, wjyurci, dldoss}@ilstu.edu

Abstract. We report a vulnerability to network signature-based IDS which we have tested using Snort and we call "Squealing". This vulnerability has significant implications since it can easily be generalized to any IDS. The vulnerability of signature-based IDS to high false positive rates has been well-documented but we go further to show (at a high level) how packets can be crafted to match attack signatures such that a alarms on a target IDS can be conditioned or disabled and then exploited. This is the first academic treatment of this vulnerability that has already been reported to the CERT Coordination Center and the National Infrastructure Protection Center. Independently, other tools based on "squealing" are poised to appear that, while validating our ideas, also gives cause for concern.

keywords: squealing, false positive, intrusion detection, IDS, signature-based, misuse behavior, network intrusion detection, snort

1 Introduction

This paper documents (at a high level) the ability to generate specific false positive alarms in the intrusion detection logs of the Snort IDS.[12] This vulnerability has been empirically verified on Snort version 1.6.3. While this vulnerability has not been tested against other IDSs, it is theoretically possible and indeed highly likely that many IDS products (both misuse and anomaly IDSs) are vulnerable to this type of attack.[1]

[1] While we have only empirically tested "squealing" for a network signature-based (misuse) IDS, we feel this same attack has similar implications for anomaly IDSs but only allude to those implications in this paper as appropriate. A separate development with empirical results is needed for anomaly IDS. A good start is "Benchmarking Anomaly-Based

We have labeled this type of attack “squealing” after the noise made by pigs during periods of distemperment (Snort has a pig logo). Squealing attacks exploit the vulnerability of IDSs to high false positive rates. The basic concept is similar to the children’s story where a boy falsely cries wolf so many times that when the wolf actually appears no one believes him. We add to this story additional bylines that the boy may not even recognize the wolf when the wolf actually appears or may spend all available time crying wolf.

The vulnerability of network signature-based IDS to high false positive rates has been well-documented.[7,9] We go further to show how packets can be crafted to match attack signatures such that alarms on a target IDS can be conditioned or disabled and then exploited. ***We agree with the analysis contained in [1] that the limitation of IDS is not the ability to accurately detect misuse behavior but rather the ability to suppress false alarms.***

At present, a solution to suppressing false positives is not imminent and the problem is getting worse as the number of reported *new* attacks increases each month - since a signature IDS has to guard against more types of attacks, the number of false positives is likely to increase.[2] The problem of suppressing false positives due to detection uncertainty is a scenario which becomes orders of magnitude more difficult when an attacker has the ability to generate decoy false positives at will.

The remainder of this paper is organized as follows: Section 2 briefly describes Snort signature pattern matching rules. Section 3 details experimentation and the implications of squealing. Section 4 surveys related work that has converged upon squealing attacks. Section 5 proposes potential solutions and we end with conclusions in Section 6.

2 Snort Pattern Matching

Snort utilizes a pattern matching model for detection of network attack signatures using identifiers such as TCP fields, IP addresses, TCP/UDP port numbers, ICMP type/code, and strings contained in the packet payload. For example, Snort may have a rule such as the following:

```
Alert tcp $HOME_NET 12345 -> $EXTERNAL_NET any
(msg:"IDS80–BACKDOOR ACTIVITY- Possible
Netbus/GabanBus";flags:SA)
```

This is the pattern-matching rule for the Netbus Trojan. Let us break this rule down to understand how the Snort packet engine recognizes signatures.

```
alert : this is an alert message
```

Detection Systems” by Roy Maxion and Kymie Tan presented at the Intl. Conference on Dependable Systems Networks, June 2000.

2 For instance, Internet Security Systems (ISS) reported 5 new attacks in June 1998, 15 new attacks in June 1999, and 77 new attacks in June 2000.[2]

```
tcp               : snort will be focused on
                    the IP protocol
$HOME_NET         : HOME_NET is a variable set to an
                    organization's IP address range (for
                    example 10.0.0.0/16)
12345             : destination TCP port number of origi-
                    nal SYN packet from $EXTERNAL_NET.
                    This represents the SYN/ACK portion
                    of the TCP handshake.
->                : Indicates that traffic will be matched
                    for source IP of HOME_NET and desti-
                    nation IP of EXTERNAL_NET.
$EXTERNAL_NET     : EXTERNAL_NET is a variable set to an
                    IP address range to be matched.  For
                    instance, this might be set to
                    0.0.0.0  if the IDS is placed at an
                    Internet connection.
any               : the "any" keyword refers
                    to TCP source port numbers for
                    the originator of the connection (in
                    this case $EXTERNAL_NET)
msg: "…"          : this message is printed to the
                    snort.alert log file.
Flags             : SYN and ACK flags are set.  This is
                    represented by S and A respectively.
                    Other flags such as PSH, FIN, RST,
                    and URG could also be specified as
                    part of a signature
```

3 Experimentation with Generating False Positives

It is possible to spoof packets crafted to match any snort signature. As a proof-of-concept, we developed a tool to provide scriptable access to several IP-based protocols. The tool is intended for internal use (it will not be released) and was given a name with a negative connotation, PCP, to recognize the danger it presents (analogy includes units are packets, can be snorted, gives user perception of extraordinary abilities).

The architecture of PCP is based on reusable C routines that can be separated into two classes: packet writing functions and argument parsing functions. For example, the packet writing functions are used to write TCP, UDP, or ICMP packets. The argument parsing functions are used to provide a command line packet writing capability.

PCP was initially used to create packets that cause problems for network routers and to test the correctness of firewall rule configurations. Other features include sample scripts which can be edited and handling ARPs for non-existent source IPs.

As a proof-of-concept, PCP emulates almost 100% of the signatures in the snort rules files which translates into several hundred attacks alarms.[3] These attacks can be executed sequentially or simultaneously.

3.1 Squeal Attack Types

Misuse pattern matching is weak because, as we have proved empirically, signatures found in packets can easily be recreated. The significant exploitations of this vulnerability include:

1) Noise-Masked Attacks. By producing a large number of false positives with varying source addresses and signature patterns, "squealing" could be used to conceal a real attack. This is especially true if "squealing" is used to divert attention from another covert attack.

2) Attack Misdirection. Falsifying the source of an attack using spoofed packets. For example, machine H (Hacker) could make it appear like entity X is attacking entity Y when it is not the case.

3) Evidence Reputability Attack. "Squealing" has the effect of repudiating all evidence provided by signature-based IDS due to both high false alarms rates and strategically implanted false alarms.

4) Target Conditioning Attack. If found non-malicious, previous alarm patterns are often used for filtering false positives.[4] Machine H could condition entity X's security administrator to become accustomed to activity A. The security administrator investigates and flags activity A as a false positive. Eventually, after machine H has conditioned entity X to ignore activity A, machine H uses activity A to exploit a vulnerability.

5) Statistical Poisoning Attack. Intentionally feeding incorrect data to anomaly IDSs to shape statistical benchmarks especially when training a system.

3.2 Attack Feasibility

It is difficult to say if this vulnerability is already being exploited but the plausibility of squealing appearing "in the wild" is highly likely now. This section describes some of the currently available software (SOCK_RAW, LIBNET, NEMESIS) with functionality to exploit this vulnerability. Tools to generate this attack have gone from requiring a moderate knowledge of the Unix network API to script driven. With the emergence of Unicode, the ultimate problem is the ability to disguise character strings

[3] snort rule sets tested include: x11, web-misc, web-iis, web-frontpage, web-coldfusion, web-cgi, virus, telnet, sql, smtp, scan, rservices, rcp, policy, netbios, misc, info, icmp, ftp, finger, exploit, dos, dns, ddos, backdoor

in various ways. Since signature-based IDS systems look for character strings within packets indicating certain network attacks, Unicode threatens to make avoiding "squeal" attacks insurmountable.[11]

SOCK_RAW: SOCK_RAW is used to create a raw socket which can be used to send packet data with arbitrary values. Use of SOCK_RAW is well documented in [15].

LIBNET is a set of functions which providing a set of wrapper functions to make use of existing raw IP networking through an easier interface. LIBNET provides a C library, libnet.h, for writing raw packets for most versions of unix and windows (with a .dll interface). Enough example LIBNET code exists for an attacker to easily adapt this code to an attack tool.

NEMESIS is a tool written utilizing LIBNET.[10] It implements functionality for packet types for many IP-based protocols to be injected at the network layer using a command line interface. NEMESIS is distributed in BSD (ports/net/nemesis) and has been ported to Windows.

The tools we have described will eventually mature and increase the difficulty of detecting a squeal attack. One possible indicator would be a large number of denies at a border router with egress filtering indicating a squeal attack may have originated from within the network. NEMESIS has statically encoded default values which, if not changed, provide a signature. Attack detection will become more difficult if static defaults are replaced with pseudo-random defaults.

4 Related Work

Our work with squealing began in the Fall 2000 with preliminary results in January 2001. The authors know of at least three other projects that have independently developed tools similar to PCP to create squealing: Snot[13], Stick[3], and Trichinosis[17]. Like PCP, these tools generate specific strings within individual packets that the signature-based IDS "wants to see" in order to match a pattern matching rule. Unlike PCP, these tools may be released "into the wild" which has caused recent warnings by the NIPC and the FBI.[16]

Of the three other projects, we have the most information about Stick. Stick generates 450 signature attacks from random IP addresses within two seconds to flood a target IDS system. Thus Stick is more a denial-of-service tool to disable an IDS than an attack tool for feeding false information to a target IDS but the technique of creating false positives is the same. Competent systems administrators will recognize a Stick attack and will soon create counter-measures. In contrast, squealing is the more general case of a covert attack that creates insidious IDS false positives.

5 Proposed Solutions

We propose two techniques in combination to address squealing: (1) adaption and (2) state awareness. The first technique is adaption – changing the signature-matching

algorithm randomly during IDS operation.[5,8] This will prevent squealing attacks based on IDS rule knowledge from having guaranteed success. Problems with this technique include (A) not all attack signatures may have multiple ways to parse for different pattern matching algorithms and (B) in response, attacks will escalate by learning to adapt making it even harder to detect squealing.

The second and more powerful technique to be used with adaption is state awareness. For speed and processing power, IDS inspect individual packets independently focusing on matching signatures in headers and data payload – no precursor events or post events, in order, are considered.[3] If IDSs had a context for monitored protocols they might be able to distinguish, for instance, a single TCP packet designed to trigger an alarm from an actual TCP connection with a precursor event (3-way handshake). An IDS with initial state awareness may seek to verify a TCP handshake with SYN/ACK, SYN, or PS/ACK flags set with specific port indicators within packets as indicated in Figure 1. An IDS with more complex state awareness may perform sequence number and acknowledgement checks. Stateless protocols, like UDP, are more difficult to monitor but multiple indicators could be correlated to trigger a single alarm.

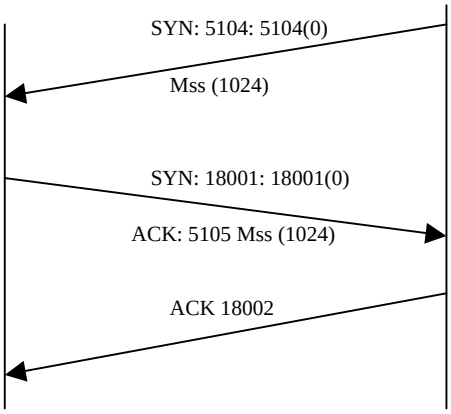

Fig. 1. Illustration of TCP handshake and initial sequence numbers derived from [14].

State awareness of data direction makes a squealing attack more difficult (see Figure 2). If the attack signature were a TCP packet with SYN/ACK flags set and a specified source port, the IDS would expect to see this packet on the inside sensor followed by same packet on the outside sensor (using timestamps for relative timing). Even in a well-crafted false positive attack, the entire session would originate from the outside thus allowing state awareness to identify it as a squealing attack.

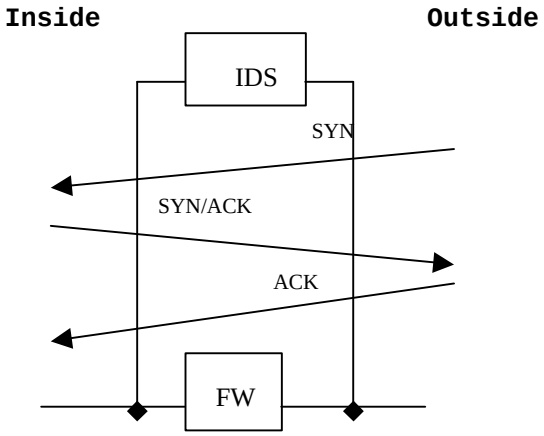

Fig. 2. An IDS implementation designed to distinguish data direction.

6 Summary

We have documented (at a high level) the ability to generate specific false alarms on network signature-based IDS – naming this ability "squealing". We successfully tested squealing by targeting the Snort IDS and we warn this attack can exploit pattern matching vulnerabilities on most signature-based IDSs. We feel this is significant evidence to support the position that the major limitation of a signature-based IDS is not the ability to accurately detect misuse behavior but rather the ability to suppress false alarms. While this is the first academic treatment of this type of attack, we have concern about other work that, while independently verifying our conclusions, may lead to the release of squealing exploitation software "into the wild". For this reason we are continuing to develop a solution to address squealing as we have outlined in this paper.

Acknowledgements

We would like to thanks John McHugh and Jeff Havrilla of the CERT/CC who encouraged us to expedite academic peer review of our work.

References

1. Axelsson, S. The Base-Rate Fallacy and the Difficulty of Intrusion Detection, ACM Transactions on Information and System Security, Vol. 3 No. 3, (August 2000) 186-205.
2. Berinato, S. The Software That Cried Wolf. ZDNetNews, (25 July 2000)
3. Giovanni, C. Fun With Packets: Designing a Stick. Draft White Paper on Stick <http://www.eurocompton.net/stick/>
4. Julisch, K. Dealing with False Positives in Intrusion Detection. 3rd Intl. Workshop on the Recent Advances in Intrusion Detection (RAID 2000), Toulouse FR (Oct. 2000) <http://www.raid-symposium.org/raid2000/program.html>
5. Kenyon, H. Adaptive Response Tool Foils Hacker Intrusion, Signal (Aug. 1999) 33-35.
6. LIBNET [written by Mike Shiffman] <http://www.packetfactory.net>
7. Lippman, R. et. al. Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation. Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'00), Hilton Head SC USA, (Jan. 2000) 12-26.
8. Loyall, J. et. al., Building Adaptive and Agile Applications Using Intrusion Detection and Response. Network and Distributed System Security Symposium (NDSS) (Jan. 2000) San Diego CA USA
9. McHugh, J. Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. ACM Transactions on Information and System Security, Vol. 3 No. 4. (Nov. 2000)
10. NEMESIS <http://www.packetninja.net/nemesis/>
11. Schneier, B. Security Risks of Unicode. Crypto-Gram, Counterpane Internet Security (15 July 2000)
12. SNORT [written by Martin Roesch] <http://www.snort.org/>
13. SNOT Discussion on Security Focus IDS Mailing List <http://www.securityfocus.com/ids>
14. Stevens, W.R. TCP Illustrated, Volume 1: The Protocols. Addison-Wesley (1994)
15. Stevens, W.R. Unix Network Programming, Volume 1: Networking APIs – Sockets and XTI. Prentice Hall (1997)
16. Sullivan, B. 'Stick' Causes an Anti-Hacking Panic. MSNBC (18 March 2001 3:11PM PT)
17. Vision, M. personal communication (Email) with the first author (25 Jan 2001)