Evading Deep Inspection for Fun and Shell

Olli-Pekka Niemi
Stonesoft Corporation
Helsinki, Finland
olli-pekka.niemi@stonesoft.com

Antti Levomäki
Stonesoft Corporation
Helsinki, Finland
antti.levomaki@stonesoft.com

ABSTRACT

This paper describes the state of contemporary network deep inspection devices from the viewpoint of an attacker attempting to evade detection. Despite claims to the contrary, even basic transport protocol layer evasions can still fool network security devices. We will present a set of working evasion techniques along with a tool that can be used to test the reactions of security devices to these techniques.

Keywords

Network, intrusion prevention, evasion, IDS, IPS.

1. INTRODUCTION

Network intrusion detection systems (IDS) and intrusion prevention systems (IPS) are middleboxes used to protect hosts and services on the Internet. These systems do real-time analysis on network traffic and attempt to alert and possibly terminate connections that are deemed harmful.

Successful traffic analysis requires that the IPS device interprets traffic in the same way as the host it is protecting. As many protocols are built according to the robustness principle stated by Jon Postel in RFC 793 [1] "be conservative in what you do, be liberal in what you accept from others", there is a large gap between how protocols should be used and how they can be used. We call deliberately sending traffic in a way that is difficult to analyze by a middlebox an evasion technique.

Evasion techniques have been researched actively in the past, and there are open source tools that can be used to do evasions. However, most of the tools are at least ten years old, limited in scope to a few protocols or otherwise unsuitable for automated black box testing of a network device.

In this paper we highlight some evasion techniques that work against current commercial and open source IPS devices. We also describe our testing tool Evader, which can be used to test the inspection capabilities of security devices.

2. BACKGROUND

IPS evasion techniques have been actively researched both by the scientific and the security communities.

- Ptacek and Newsham [2], and Horizon [3] published a set of TCP/IP layer evasion techniques in 1998. The use of TCP urgent pointer as an evasion was published in Phrack [4]. Fragroute [5] can be used to perform some of these evasions.
- HTTP evasions used by Whisker were documented by Rain Forest Puppy [6] and URI related evasions by Daniel Roelker [7]. Metasploit [8] implements most of these evasions.
- Caswell and Moore [9] summarized the current state of IPS evasions in 2006 and introduced new application layer techniques. Bidou [10] surveyed IPS issues and demonstrated bypassing multiple consecutive IPS systems. Gorton and Champion [11] systematically combined evasions to find working combinations against tested systems.
- Evasion blocking performance of various IPS devices has been tested in [12] [13] [14].

IPS devices are typically implemented as middleboxes which inspect traffic flowing through them. Unlike firewalls, IPS devices let through all traffic that is not deemed malicious. Evading IPS therefore requires that traffic is obfuscated so that the IPS cannot classify it as evil.

Reliable inspection also requires that the traffic is interpreted on the device similarly to the mediated hosts. This includes performing IP reassembly, TCP reassembly and application specific tasks like HTTP chunked encoding handling.

3. EVADING DEEP INSPECTION

In this paper we have concentrated on examples of evading TCP reassembly because TCP is complex and is used as the base for most application layer protocols. If an IPS cannot perform correct TCP reassembly, it cannot parse any of the protocols transported over TCP either. IPS devices can additionally run signature matching on separate TCP segments as an optimization. Evading all inspection

therefore may require both breaking TCP reassembly capabilities and making sure that individual segments do not contain anything a signature could match to.

We will highlight a few evasion techniques that currently have a high success rate in thwarting the TCP reassembly of the tested systems. All methods can be combined with non-standard TCP segmentation to avoid packet-based signatures.

Additionally, custom shellcode encoders should be used to avoid signatures targeting commonly used encoders, e.g., stock Metasploit encoders.

3.1 TCP PAWS

The PAWS (Protection Against Wrapped Sequence numbers) algorithm is defined in RFC1323 [12]. The algorithm uses TCP timestamps to drop segments that contain timestamps older than the last successfully received segment. Most modern hosts implement PAWS. Its use as an evasion was described already by Ptacek and Newsham in 1998 [2].

PAWS causes problems for TCP reassembly when the IPS does not know which segments the end hosts accept or discard. Once the IPS decides to accept a segment that the end host does not, or vice versa, the reassembled TCP streams differ in the IPS and the end host.

TCP segments designated for PAWS elimination can be created by duplicating a valid TCP segment and moving its timestamp value backwards. The actual contents of the duplicate segment can be arbitrary, e.g., a non-malicious version of a protocol message.

3.2 SYN retransmit

TCP assigns a sequence number for each byte of payload and the control flags SYN and FIN [1]. Hosts receiving data with an already acknowledged sequence number will discard the parts already handled and process new data if present.

This behavior allows for retransmitting the initial SYN flag along with the first TCP segment containing payload. IPS devices may have problems with the unexpected combination of a retransmitted SYN flag and new payload in an established connection.

3.3 IPv4 options

IPv4 packet headers can contain options [13]. If any of the options are invalid, the whole IPv4 packet should be discarded by the receiving host. This can cause problems if the inspecting device and the end host discard different packets.

3.4 TCP urgent data

TCP payload can be marked as urgent with the urgent pointer. The receiving TCP socket can handle urgent data as either inline or out-of-band. Out-of-band data is not returned via normal recv() calls and gets discarded by applications that do not use urgent data. Most operating systems default to out-of-band urgent data [13].

The use of TCP urgent data as an evasion was documented in 2001 [4]. It is problematic for IPS devices because the choice between handling data as inline or out-of-bound is application specific.

3.5 TCP receive window

TCP receive window is the amount of new data that the sending side is willing to receive. An attacker can advertise a small window size to force the other end of the TCP connection into sending small segments.

This complements sending small TCP segments by allowing the attacker to control TCP segment sizes in both directions.

4. EVADER

Evader [14] is a tool for testing the deep inspection capabilities of an IPS device. It has a few exploits using different application layer protocols and a set of evasions that can be applied to them. By using real exploits and real vulnerable victim hosts we can easily verify that an attack was successful and that the applied evasions did not make the traffic unintelligible.

Evader is built on a proprietary user-space TCP/IP stack. It has application clients/servers for higher layer protocols. This allows Evader to have complete control over every packet sent during execution, as opposed to using a combination of separate tools which each handle their own layer.

Exploits are further divided into stages over time. Each stage corresponds to a step in the application protocol, for example “SMB Session Setup” or “MSRPC Bind”. Evasions can be targeted to all or a set of stages. Targeting evasions to specific stages critical to IPS detection makes anomaly-based blocking more difficult as the protocol anomaly may be present only once during a connection instead of occurring frequently.

All exploits containing shellcode can be run ‘obfuscated’. The obfuscated version generates a different shellcode encoder and possible NOP sled for each execution to avoid exploit based detection. Normal versions of these exploits attempt to look like commonly found public exploits.

4.1 Exploits

Because we are testing IPS inspection capabilities and not doing penetration testing, we have chosen exploits that are old and well known. All IPS devices should detect them with no evasions applied:

- CVE-2008-4250, MSRPC Server Service Vulnerability [14].
  - A buffer overflow vulnerability in Microsoft Windows allowing arbitrary code execution. Widely exploited by the Conficker worm.
  - Evader targets a Windows XP SP2 host.
  - Protocols used: IP, TCP, NetBIOS, SMB, MSRPC.
- CVE-2004-1315, HTTP phpBB highlight
  - Input sanitation vulnerability in phpBB allowing arbitrary PHP execution. Exploited by the Santy.A worm in 2004.
  - Protocols used: IP, TCP, HTTP
- CVE-2012-0002, Windows RDP Denial of Service [15].
  - Vulnerability in the Remote Desktop Protocol implementation in Microsoft Windows.
  - Exploit in Evader crashes unpatched Windows 7 hosts.
  - Protocols used: IP, TCP, RDP

4.2 Evasions

We have implemented atomic evasions for a number of widely used protocols.

Table 1: IPv4 evasions

ipv4_frag	Set IPv4 maximum fragment size
ipv4_order	IPv4 fragment reordering
ipv4_opt	Duplicate IPv4 packets with broken options.

Table 2: TCP evasions

tcp_chaff	Duplicate TCP packets with broken headers
tcp_initialseq	Initial TCP sequence number modification
tcp_inittsopt	Initial TCP timestamp modification
tcp_nocwnd	Disable TCP congestion control
tcp_nofastretrans	Disable TCP fast retransmit
tcp_order	TCP segment reordering
tcp_osspoof	TCP SYN OS fingerprint spoofing
tcp_overlap	Send overlapping TCP segments
tcp_paws	PAWS elimination, duplicate TCP segments with old timestamps
tcp_recv_window	Modify TCP receive window
tcp_seg	TCP segment maximum size
tcp_segvar	TCP segmentation with variable segment sizes
tcp_synretranswithpayload	Retransmit SYN with first payload segment
tcp_synwithpayload	Send payload in initial SYN packet
tcp_timewait	Open decoy TCP connections from same IP-port pair before attack
tcp_tsoptreply	TCP timestamp echo reply modifications
tcp_urgent	Add urgent data to TCP segments

Table 3: HTTP evasions

http_header_lws	Add linear white spaces to HTTP headers
http_known_user_agent	Use a common HTTP user agent
http_request_line_separator	Modify HTTP request line separator
http_request_method	Modify HTTP request method
http_request_pipelined	Send extra pipelined HTTP requests before exploit
http_url_absolute	Use absolute URLs
http_url_dummypath	Add dummy paths to URL
http_url_encoding	Encode URL
http_version	Set used HTTP version

Table 4: NetBIOS evasions

netbios_chaff	Extra NetBIOS messages
netbios_init_chaff	Extra NetBIOS messages before first normal message

Table 5: SMB evasions

smb_chaff	Extra invalid SMB messages
smb_decoytrees	Opens extra SMB trees with decoy writes
smb_fnameobf	SMB filename obfuscation
smb_seg	SMB write segmentation
smb_writeandxpad	SMB WriteAndX message extra padding

Table 6: MSRPC evasions

msrpc_bigendian	Force big endianness
msrpc_ndrflag	MSRPC NDR field modifications
msrpc_seg	MSRPC request segmentation

5. MONGBAT

Mongbat is an automated test tool that runs multiple instances of Evader in parallel with random evasion combinations. Successful exploits are reported along with a command line for easy repeatability.

A single execution of Evader through Mongbat usually includes the following steps when using an exploit payload that opens a shell:

- Run a clean traffic test with no exploit and no evasions (*optional)*. This is used to test that the victim service is up and reachable. It also verifies that the security device is not blocking the whole service or protocol, and allows normal traffic through. If the clean test is used and it fails, no exploit run is attempted as the service is considered to be down. The clean check cannot be used if the tested security device is configured to block access to the tested vulnerable service.
- Check that the shell port is not open.
- Run exploit with evasions.
- Check if the shell port opened.

If the shell port opened, the victim host was successfully exploited and the security device did not block the attack. Worker threads running Evader use unique source address and shell port combinations to determine which parallel attack succeeded. Running exploits with payloads that do not open a shell are more difficult for automatic testing, and usually require human verification.

Mongbat randomly selects a number of evasions and their parameters for each Evader execution. No special care is taken to produce only evasion combinations that produce legitimate traffic. The victim computer is used to validate working combinations as it will discard broken traffic.

6. RESULTS

Here we present the results of running Mongbat against 9 vendors' commercial IPS devices. The vendors include most Gartner 2012 IPS and 2013 NGFW Magic Quadrant leaders and challengers. The devices have up-to-date software and updates installed. We have attempted to configure the devices for maximum detection and blocking while still allowing the Evader clean check without an exploit to succeed.

We have defined 12 evasion test cases and a baseline test case without evasions. The evasion tests are run with Mongbat so that only the listed evasions are used. If Mongbat does not find a working evasion combination in one minute the test case is marked as failed, otherwise as success. In test cases 6-11 we additionally require that all given evasions be used with the exploit.

Table 7: Evasion test cases

ID	Name	ID	Name
0	No evasions	7	TCP PAWS + TSR
1	TCP PAWS	8	SYN retransmit + TSR
2	SYN retransmit	9	IPv4 options + TSR
3	IPv4 options	10	TCP urgent data + TSR
4	TCP urgent data	11	TCP receive window + TSR
5	TCP receive window	12	All listed evasions
6	TCP segmentation and reordering, referred to as TSR later		

All test cases were executed three times per vendor:

- With MSRPC exploit (CVE-2008-4250), no stages used. All evasions are applied for the whole TCP connection. Results in Table 8.
- With MSRPC exploit (CVE-2008-4250) using stages. Mongbat can specify that the evasions are applied only during given protocol steps. Results in Table 9.
- With HTTP exploit (CVE-2004-1315). As the whole exploit is delivered in a single HTTP GET request no stages are available. All evasions are applied to the whole connection. Results in Table 10.

The RDP exploit against CVE-2012-0002 was not used in this test because it is a denial of service attack. Automatically determining which evasion combination was successful is more difficult when the victim host crashes. The exploits used open a specified unique shell port when successful, allowing automatic verification.

Table 8: Results for MSRPC exploit against CVE-2008-4250 with no stages used.

Vendor	Test case index												
	0	1	2	3	4	5	6	7	8	9	10	11	12
Vendor1													
Vendor2			x		x		x	x		x	x	x	x
Vendor3			x								x		x
Vendor4		x		x			x	x		x	x	x	x
Vendor5		x	x	x				x		x			x
Vendor6						x	x	x		x		x	x
Vendor7		x	x	x	x	x	x	x		x	x	x	x
Vendor8		x	x		x	x	x	x		x	x	x	x
Vendor9								x			x		x

x indicates a successful exploit through IPS device.

empty space indicates IPS blocked all exploit attempts.

Table 9: Results for MSRPC exploit against CVE-2008-4250 using stages.

Vendor	Test case index												
	0	1	2	3	4	5	6	7	8	9	10	11	12
Vendor1													
Vendor2			x		x		x	x	x	x	x	x	x
Vendor3			x					x	x		x		x
Vendor4		x		x			x	x	x	x	x	x	x
Vendor5		x	x	x				x	x	x			x
Vendor6						x	x	x	x	x	x	x	x
Vendor7		x	x	x	x	x	x	x	x	x	x		x
Vendor8		x	x		x	x	x	x	x	x	x	x	x
Vendor9								x	x		x		x

x indicates a successful exploit through IPS device.

empty space indicates IPS blocked all exploit attempts.

All vendors are able to stop the MSRPC exploit with no evasion applied. TCP segmentation and reordering by itself seem to go through most vendors' IPS devices. When this is combined with segments destined for PAWS elimination and applied to stages almost all vendors' inspection can be bypassed.

The MSRPC exploit requires multiple SMB requests and responses before the vulnerability can be exploited. This allows IPS devices to perform protocol validation and possibly terminate evasion attempts during the session setup phase. Evasions using just a small TCP receive window probably cause some devices to lose protocol state due to a failure in parsing server responses.

Table 10: Results for HTTP exploit against CVE-2004-1315.

Vendor	Test case index												
	0	1	2	3	4	5	6	7	8	9	10	11	12
Vendor1													
Vendor2		x	x	x	x	x	x	x		x	x	x	x
Vendor3	x	x		x	x	x	x	x		x	x	x	x
Vendor4		x	x	x	x	x	x	x		x	x	x	x
Vendor5	x	x	x	x	x	x	x	x		x	x	x	x
Vendor6	x	x	x	x	x	x	x	x		x	x	x	x
Vendor7			x		x						x		x
Vendor8	x	x	x	x	x	x	x	x		x	x	x	x
Vendor9		x	x				x	x			x	x	x

x indicates a successful exploit through IPS device.

empty space indicates IPS blocking all exploit attempts.

All vendors were not able to block the obfuscated HTTP exploit without evasions. The complete set of test cases was still run to see if the devices block some evasions as anomalies. TCP segmentation and reordering was again successful also over HTTP, especially when combined with TCP segments containing urgent data.

In cases when no exploit succeeded Mongbat executed around 500-2000 attempts in the 60 second test period. Most successful evasion combinations were found in 1-10 attempts.

7. CONCLUSIONS

We have highlighted several evasions that work against modern, up-to-date IPS devices. The evasions attack TCP reassembly, which makes them usable with different application protocols, e.g. HTTP, SMB and SIP. We have also described our freely available Evader tool that can be used to reproduce our findings and test the inspection capabilities of security devices.

8. REFERENCES

[1] IETF, "Transmission Control Protocol," *IETF, RFC793,* 1981.

[2] T. Ptacek and T. Newsham, "Insertion, Evasion and Denial of Service; Eluding Network Intrusion Detection," Secure Networks Inc., 1998.

[3] Horizon, "Defeating Sniffers and Intrusion Detection Systems," *Phrack Magazine,* vol. 8, no. 54, 1998.

[4] "NIDS Evasion Method named "SeolMa"," *Phrack,* vol. 0x0b, no. 0x39, 2001.

[5] D. Song, "fragroute," [Online]. Available: http://www.monkey.org/~dugsong/fragroute/. [Accessed 27 June 2013].

[6] R. F. Puppy, "A look at whisker's anti-IDS tactics," 1999. [Online]. Available: http://www.ussrback.com/docs/papers/IDS/whiskerids.html. [Accessed 27 June 2013].

[7] D. J. Roelker, "HTTP IDS Evasions Revisited," 2004. [Online]. Available: http://www.snort.org/assets/164/sf_HTTP_IDS_evasions.pdf. [Accessed 27 June 2013].

[8] "Metasploit," [Online]. Available: http://www.metasploit.com/. [Accessed 27 June 2013].

[9] B. Caswell and H. D. Moore, "Thermoptic Camouflage: Total IDS Evasion," in *Blackhat*, 2006.

[10] R. Bidou, "IPS Shortcomings," in *Blackhat*, 2006.

[11] A. Gorton and T. Champion, "Combining Evasion Techniques to Avoid Network Intrusion Detection Systems," Skaion, 2004.

[12] E. Korhonen, "Advanced Evasion Techniques: Measuring the threat detection capabilities of up-to-date network security devices," Master's Thesis, Aalto University, 2012.

[13] M. Dyrmose, "Beating the IPS," SANS Institute, 2013.

[14] C. Xynos, I. Sutherland and A. Blyth, "Effectiviness of blocking evasions in Intrusion Prevention Systems," University of South Wales, 2013.

[15] IETF, "TCP extensions for high performance," *RFC1323,* 1992.

[16] IETF, "Internet Protocol," *RFC791,* 1981.

[17] IETF, "On the Implementation of the TCP Urgent Mechanism," *RFC6093,* 2011.

[18] Stonesoft, "Evader," [Online]. Available: http://evader.stonesoft.com/. [Accessed 27 June 2013].

[19] Microsoft, "Vulnerability In Server Service Could Allow Remote Code Execution. MS08-067, CVE-2008-4250," 2008.

[20] Microsoft, "Vulnerabilities in Remote Desktop Could Allow Remote Code Execution. MS12-020, CVE-2012-0002," 2012.