Aalto University
School of Science
Degree Programme of Computer Science and Engineering

Eetu Korhonen

Advanced Evasion Techniques:

Measuring the threat detection capabilities of up-to-date network security devices

Master's Thesis
Espoo, August 31, 2012

Supervisors:	Professor Aurélien Francillon, EURECOM
	Professor Antti Ylä-Jääski, Aalto University School of Science
Instructors:	MSc. Antti Levomäki
	Olli-Pekka Niemi

Aalto University
School of Science
Degree Programme of Computer Science and Engineering

ABSTRACT OF
MASTER'S THESIS

Author: Eetu Korhonen

Title: Advanced Evasion Techniques: Measuring the threat detection capabilities of up-to-date network security devices

Date: August 31, 2012 **Pages:** 63

Professorship: Data Communication Software **Code:** T-110

Supervisors: Professor Aurélien Francillon
Professor Antti Ylä-Jääski

Instructors: MSc. Antti Levomäki
Olli-Pekka Niemi

The robustness principle of software design as stated in RFC 761 in 1980 is "Be conservative in what you do, be liberal in what you accept from others." Although useful for building interoperable software systems, this principle opens room for an attacker to deliberately cause traffic ambiguities to fool network based threat detectors.

This study examines a set of commercial network security solutions against a complex set of evasive techniques and works to construe an overview on the performance of such tactics and what is the current state of network based defences when countering these threats.

We established a set of comparable device configurations, metrics for evaluation and three controlled experiments for our study. We present the results of our experiments and analyze them for observable patterns and deviations.

Our findings suggest that most commercial solutions still have trouble countering techniques discovered more than a decade ago and nearly every tested device is eludible in seconds of persistent search. However, improved evasion resistance appears feasible as a number of devices performed better than others.

Keywords: Network Security, Network Intrusion Prevention Systems, Evasions, Multi-protocol Evasions, Hybrid Evasions, Advanced Evasion Techniques, IDS, IPS, AET

Language: English

Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan tutkinto-ohjelma

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Eetu Korhonen

Työn nimi: Kehittyneet evaasiotekniikat: Nykyaikaisten verkkoturvallisuuslaitteiden uhkien havainnointikyvyn mittaaminen

Päiväys: 31. elokuuta 2012 **Sivumäärä:** 63

Professuuri: Tietoliikenneohjelmistot **Koodi:** T-110

Valvojat: Professori Aurélien Francillon
Professori Antti Ylä-Jääski

Ohjaajat: FM Antti Levomäki
Olli-Pekka Niemi

Verkkoliikenteen luotettavuusperiaate, “Ole varovainen lähettäessä, liberaali vastaanottaessa.” on auttanut rakentamaan Internetin yhteensopivuutta vuodesta 1980. Eduistaan huolimatta periaatteen toteutus myös avaa hyökkääjälle mahdollisuuden hyväksikäyttää monitulkintaista verkkoliikennettä tietoturvalaitteiden hämäämiseksi.

Tutkimuksemme tarkastelee joukkoa kaupallisia tietoturvaratkaisuja useita “evaasiotekniikoita” vastaan. Tällä pyrimme selvittämään kyseisten hyökkäystekniikoiden suorituskykyä ja arvioimaan nykyisten tietoturvalaitteiden kykyä torjua näitä uhkia.

Diplomityössä rakennamme vertailukelpoiset kokoonpanot laitteista, mittaustekniikat saatujen tulosten arviointiin ja toteutamme kolme koetta. Esittelemme kokeiden tulokset ja analysoimme niistä havaittavia säännönmukaisuuksia ja poikkeavuuksia.

Tuloksemme viittaavat siihen, että useimmat kaupalliset ratkaisut ovat edelleen riittämättömiä yksinkertaisia, yli kymmenen vuotta sitten havaittuja tekniikoita vastaan. Lähes jokainen testattu laite on väistettävissä sekunneissa satunnaisilla evaasioyhdistelmillä.

Asiasanat: Tietoturva, tunkeutumisen havainnointijärjestelmät, tunkeutumisen estojärjestelmät, evaasiot, moniprotokollaevaasiot, kehittyneet evaasiotekniikat, IDS, IPS, AET

Kieli: Englanti

Acknowledgements

This thesis would not have been possible without the patience and outstanding advisory of my supervisors, Olli-Pekka Niemi and Antti Levomäki and the Stonesoft VAG team. Not to forget my EURECOM supervisor, Aurélien Francillon and the rest of my academic acquaintances for guiding and inspiring me. I keep being amazed, how knowledge is daily cultivated – pushing people forward.

My family has been incredibly supportive and has set a great example to strive for. Particular accolades for my aunt Paula, who has a wonderful habit for helping out poor students in need.

I am grateful for all of my friends who share the air I breathe. Especially for the ones who have responded with feedback and sympathy to my enthusiasm over working with this book. I never really got to understand, why some people view thesis work as frustrating.

in Espoo, August 31, 2012

Eetu Korhonen

Abbreviations and Acronyms

AET	Advanced Evasion Techniques
CERT	Computer Emergency Response Team
CERT-FI	Finnish National Computer Security Incident Response Team
CIO	Chief Information Officer
CVE	Common Vulnerabilities and Exposures
DARPA	Defense Advanced Research Projects Agency of United States Department of Defense
DGA	Domain Generation Algorithm
DUT	Device Under Test
HTTP	Hypertext Transfer Protocol
MSRPC	Microsoft Remote Procedure Call
(N)IPS	(Network) Intrusion Prevention System
(N)IDS	(Network) Intrusion Detection System
NOP	No Operation instruction
QoS	Quality of Service
SMB	Server Message Block
TCP	Transmission Control Protocol
VPN	Virtual Private Network

Contents

Chapter 1

Introduction

Protecting information networks from unwanted intrusions is a problem traditionally assessed with a multidirectional approach. Due to the unpredictable configurations of the systems under threat, it is not sufficient to identify any singular points of defense to cover critical system components from misuse. New strains of malware, emergence of novel security threats and the monetization development of computer crime are pushing security organizations and end-users to respond with an expanding stack of countermeasures.

One method of providing a layer of protection is to use a Network Intrusion Detection and Prevention System (IDS and IPS). NIDS is transparently inserted to monitor a network link and contains the necessary software to detect security threats from the traffic as illustrated in figure 1.1. Upon detecting a suspicious pattern in the traffic flow, IDS devices report and log the incidents to the network administration, whereas Intrusion Preventing (IPS) devices can also apply reactive countermeasures to block invasions as described in figure 1.2. For example, one can reset a connection with malicious payload or reconfigure a firewall to block further connection attempts emerging from a malicious source.

Providing real time security by inspecting the link layer raises several issues. Doing comprehensive inspection for large streams of traffic can be

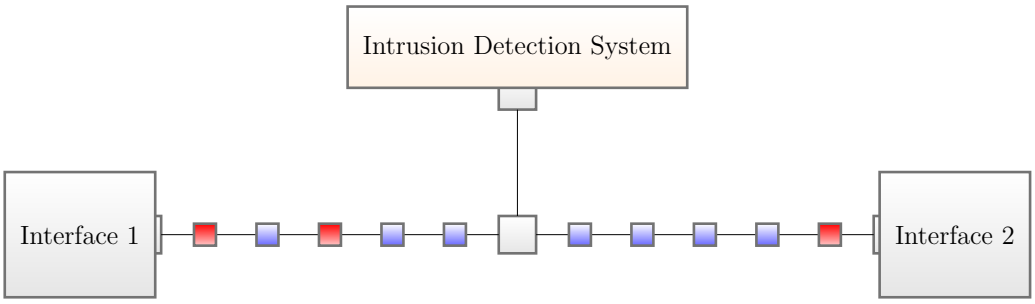

Figure 1.1: Intrusion Detection System works as a passive monitor of the network traffic

Figure 1.2: Intrusion Prevention System can intervene with inline network traffic

resource intensive and create bottlenecks for network performance. Devices capable of doing this are priced out from consumer markets and ones deployed can compromise with inspection quality and adaptability [1] to achieve satisfactory throughput with low delay.

Another challenge is with the limitations of protocol analysis. Whether the detection is done by observing statistical anomalies or defined signatures within the packet flow, the content of the traffic will ultimately be evaluated by the communicating endpoints. Jon Postel's robustness principle [2] of Internet architecture[2–4] allows us near infinite amount of reconfigurations considerable as potentially valid network traffic by tolerant endpoints. Developing signatures capable of inspecting the content regardless of the form is considered an unfeasible problem for the IDS model to solve reliably.

1. For example, some hardware accelerations may be difficult to upgrade against evolving threats.

2. "Be conservative in what you send, be liberal in what you accept from others"

We call the methods of exploiting the limitations of network security devices to bypass their inspection as *evasions*. This study examines a set of evasion techniques, their reconfigurations and effectiveness against leading IPS devices on the market.

1.1 Problem statement

Within this study, we deal with three issues. First we describe the *current state* of IDS evasion research. What has been done before and how our study relates to it? We try to illustrate the evasion research performed and what are the "Advanced Evasion Techniques" (AET's) Stonesoft has researched, publicized and claims to counter.

Secondly, we attempt to provide methods and metrics to evaluate the *effectiveness* of these evasion techniques. How well do they evade the applied defenses and how their performance changes when they get modified?

Our last goal is to estimate the *defensive performance* of current state-of-the-art network security devices against evasions of varying complexity. With this evaluation, we seek to answer whether evasions pose a threat against networks protected in link layer, whether increasing the complexity of evasions increases the threat and if the presented solutions offer credible protection against this class of attacks.

1.2 Structure of the dissertation

The dissertation is split into eight chapters. After introducing the nature of the problem under study, we provide a brief background on the state of network security evasion techniques, existing research and defensive technologies available. The next chapters contain the details of our experiment implementations. We describe the used network and software environments; define the methodologies used and the motivation for their use. In the core of the study we describe three experiments we have conducted and the initial

results extracted. We continue with a deeper evaluation and cross-analysis of these results and try to answer the questions defined in section 1.1. We conclude by reviewing possible implications of the observed nature of this threat and examine possible directions for further research.

Chapter 2

AET Background

In this chapter, we provide background information on the state of Intrusion Prevention Systems and the principles behind most used detection techniques. We also present a definition for Advanced Evasion Techniques and how we consider the experiments presented in this study to be beyond trivial. For last, we introduce the most relevant research preceding and influencing this study.

2.1 Intrusion Prevention System

According to the National Institute of Standards and Technology, A Network Intrusion Detection System (NIDS or IDS) is a software system to automate "the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices." A Network Intrusion Prevention System (NIPS or IPS) is defined to be an IDS system with an added capability to attempt stopping possible incidents[33].

Manipulating passing traffic in an IPS can reduce the functional requirements for the detection. Since the data streams passing through an IPS eventually differ from a flow going through a purely observing IDS, the detection

mechanism of IPS is left with a reduced set of possible traffic combinations to inspect. In addition to removing malicious data, an IPS system can further limit their search spaces by normalizing[18] the data flow by removing ambiguities with low likelihood of affecting the end-to-end communication from the traffic.

Intrusion Prevention Systems are not the only class of devices implementing packet inspecting security measures in the network. For what is regarded as "Next Generation Firewall" (NGFW) since 2009[25], much of the deep inspection functionality is being integrated to existing firewall devices. Unified Threat Management (UTM) solutions are considered to be even broader products with features such as anti-virus, anti-spam, VPN tunneling and content filtering[27].

2.1.1 Intrusion Detection Techiques

Intrusion Detection Systems can base their detection logic to several principles. The most common distinction of detection techniques is between statistical anomaly detection and predesigned signature-based detection.

An anomaly-based intrusion detector relies on information it knows on what kind of traffic can be considered "normal" to the network and any significant deviations from this traffic will trigger an alert. This kind of detection is well suited for dedicated, well defined systems where the requirements for network traffic do not change in the long term. As unable to identify threats, anomaly-based approach is less ideal for environments where distinct rules for intended traffic are harder to craft, such as corporate wide intranets where requirements shift quickly and the basic policies needs to be permissive enough to support changing and updating applications under use. As a benefit, anomaly-based detection can block unknown 0-day threats unlike approaches relying on attack databases.

The techniques we use to evade the detectors can often be classified as network anomalies. As anomaly-based detection is much prone for raising false positive alerts, we are interested in evaluating when the devices under

test make their decisions due to detecting anomalies instead of parsing for genuine threats.

Signature-based detector contains crafted knowledge of vulnerabilities, exploits and implementations of possible attacks. The flow of information going through the IDS is compared to this database and any traffic matching to a set of signatures raises an alert. This model is dependent on the quality and availability of the signatures themselves, which makes the model more vulnerable against 0-day exploits, obfuscation techniques and possible endpoint encryption used in individual attacks.

A simple way of examining the traffic is to perform signature matching against the raw traffic stream in the network visible to the IDS inline interfaces. However, in order to provide versatility and fault tolerance, the protocols used offer near limitless permutation possibilities to transfer analogous information while remaining interpreted correctly by the end-hosts. Creating signatures capable of matching all or even most possible permutations can be difficult or computationally unfeasible in use.

An intelligent traffic analyzer can inspect the application stack contained within the data stream and reinterpret the traffic to be examined independently against multiple layers of the networking stack. A reactive device such as IPS can also perform an amount of traffic normalization as mentioned in section 2.1. These allow the device to use simpler and more accurate signatures, since there are less possible data permutations.

2.2 IDS Evasion Techniques

In our study, methods of bypassing the detection capabilities of network security systems are called evasions. Evasions are often well known and documented features of the used protocols with real use cases. Protocol features, such as flexible TCP window size and retransmission policies[12] are designed to provide improved fault tolerance, but they can also be used to make the traffic look unidentifiable by existing signatures. Security policies

treating protocol variations with suspicion are often impractical as they can lead to unnecessary termination of legitimate communication.

In addition to protocol evasions, our study involves payload obfuscation techniques, where a well known payload sample is replaced with non-identifiable data sequences with orthogonally evaluating content. Examples of common obfuscation techniques include encryption, character re-encoding and content padding. Although obfuscation mutates the appearance of an exploit to the inspecting device, our obfuscated attacks target a shared vulnerability detectable by appropriate signatures.

2.3 Advanced Evasion Techniques

Advanced Evasion Techniques (AET's) is a lately established term in network security industry referring to a set of non-trivial and expensive means to fool network based threat detection systems.

Originally coined in 2010, AET's were defined as "any techniques that allow an intruder to evade or bypass security detection during a network-based attack[1]." While the broad definition is inclusive to any techniques considered trivial or non-technical, increasing the evasion complexity with additional techniques [1] can be used to increase the effectiveness and computational demands for detecting the attack.

While it is unfeasible to define a level of sophistication where an individual security evading attack vector becomes an advanced evasion, we can assess the tools, resource requirements and engineering effort behind an effective attack to be beyond trivial, cheap or easy to replicate.

In this AET research, we consider two non-trivial advantages are used to increase the sophistication of attacks:

– The testing tools used to attack and modify the evasions are a product of significant research and development. By extending well known eva-

1. Such as mutating the exploit during execution, applying evasive measures within multiple protocol layers or designing target specific custom evasions.

sion techniques and implementing a custom Internet stack, the Evader test kit supports advanced features like dynamic attack modifications, parallel evasions on multiple layers, false positive probing and evasion permutations with high number of supported evasion techniques. The effectiveness and complexity of the evasions generated with the tools have surpassed all analogous tools previously used in internal testing.

– We have access to a number of costly, state-of-the-art network defense systems. Hands-on knowledge of the systems and capability to directly measure their performance makes it possible to engineer individual attacks to be most effective against specific known security measures – or to be statistically more effective against unspecific security measures when the target environment is unknown. In addition, this allows us to develop attacks in secret without alerting a live target production environment, which could lead to manually elevated security measures in blind probing and brute force scenarios.

In 2010, the Evader research led to a CERT-FI disclosure for 23 evasion techniques discovered as a byproduct of internal product testing[13]. This was followed by another batch of 124 evasions[14] and the company has continued providing samples to cert and IPS vendors thereafter.

The experiments in our study and other subsequent tests with upgraded inspection policies have revealed many of the first 23 evasions are still unpatched by the vendors. Some of the patches claiming to counter the problems raise issues of network flow fault tolerance and performance. After the initial public disclosures, the network security evading modifications were applied on common HTTP service vulnerabilities in order to demonstrate the threat to be extendable beyond internal and isolated networks and protocols.

In February 2012, Damballa Labs reported about an analysis on six crimeware families they consider to be utilizing techniques known as Domain Generation Algorithms which were considered as Advanced Evasion Techniques[6, 17]. According to Damballa report, the DGA use helped the malware to evade blacklist, signature filter and static reputation based

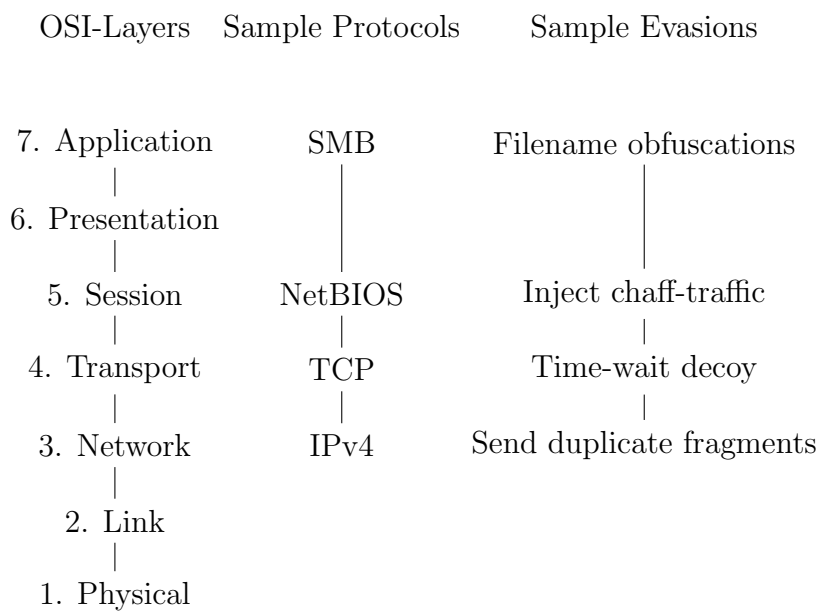

Figure 2.1: A sample hybrid evasion illustrated.

network- and host based security measures as a byproduct to providing redundant communication channels to the botnet control infrastructure.

2.4 Related Research

Evasion techniques – including the hybrid-evasion approach presented in this study – are not a new finding. Issues with interpreting and routing ambiguous traffic have been known since the defining standards of the internet were developed. The IDS evasion techniques discovered by Ptacek and Newsham in 1998[28] are considered to form the ground for current evasion research and resistance evaluation. Despite being aged and well known, simple techniques like the ones described by Ptacek and Newsham are still found effective against network defenses. While Ptacek and Newsham focus mainly on the issues regarding network- and transport layer interpretation, their work has been extended by examining evasion and spoofing techniques in other protocols and applications, such as HTTP[29–31], NetBIOS/SMB,

MSRPC[10, 11] and with more adaptable solutions – such as evasions with changing character encodings and address obfuscations[24].

Combining known evasion techniques to hybrid evasions was a well researched topic in the 2000's. Released tools capable of performing evading modifications to the network traffic include Fragrouter[36], which focused on replicating the evasions presented by Ptacek and Newsham; Whisker[29] simulating HTTP-based attacks with evasions and Mucus[23] crafting TCP & UDP packets matching to an IDS signature and attempts to mutate them in order to avoid intrusion detection. THOR[21] introduced a proxy-based tool for creating traffic "variations" between data link- and application layer. AGENT[32] implements a set of transport and application level evasions and uses a transformation engine to compute improved attack signatures for mutated attacks as a proof of concept for defending against similar attacks. Sploit[5] and our Evader tool share much with their multi-stack evasion approach, though the test conditions used in presenting articles[9, 38] focus more on IDS signature quality.

To protect against mutated evasions, other solutions besides deep protocol analysis have been presented. Aside from common wisdom of keeping a holistic approach of multi-layered network defense, most approaches involve normalization of inspected traffic[18, 39] as described in section 2.1.1. Other novel approaches include automated scans of the network and building context awareness for feasible whitelisting [34] – and by improving the signature-structure used by an IDS[35, 37].

Most of the research done on evasions has focused on individual techniques and their implementations. The actual effectiveness of evasion techniques against available commercial and non-commercial network defenses has been less explored. A number of broad evaluations have been made, most notably the ones performed by DARPA in 1998[19] and 1999[20]. The design and execution of DARPA evaluations were extensively criticized by John McHugh[22] in 2000.

As intrusion detection has grown as a commercial service, a number of

independent security agencies have provided evaluation services for network security devices. The most renowned evaluations are done by NSS Labs [2] and ICSA [3]. The methodologies used by the agencies are designed to differentiate the available security devices in order to aid network administrators and CIO's building organization security.

Our study builds on the existing research as we are able to combine the scale of multi-device evaluation done by independent reviewers with the established evasion research theory and practice.

2. http://www.nsslabs.com/
3. http://www.icsalabs.com/

Chapter 3

Environment and Tools

3.1 Hardware Setting

The testing environment consists of a set of virtualized "target" computers reachable throgh a physical switch. The switch receives traffic from 11 interfaces – each connected to a different retail IPS configuration. The Devices Under Test (DUT's) are connected into a switch computer running a custom routing software. The software is used for routing the traffic though the interfaces with the corresponding IPS, giving us flexibility to test our devices with little need for reconfiguration between tests.

When the experiments were done between July 7th and July 11th, our switch allowed us to route our traffic through the following IPS devices:

Device	Used threat signature database
A clean route	No IPS in between. This is used as a calibration technique to confirm whether the presented evasions open shells in the receiving end.
Cisco IPS-4240	Signature-set 654.0-2012.06.29
Checkpoint UTM-1 270	IPS fingerprints 634123998
Fortinet Fortigate 800	IPS definitions 3.00205
IBM Proventia GX4004	Intrusion prevention DB 32.060

Juniper SSG 320M & IDP-75	Attack DB 2158
McAfee M-1450	Signature 7.5.16.21
PaloAlto PA-500	Threat DB 317-1438
SourceFire 3D Sensor 2100	VDB 108
Stonesoft IPS 5.2	Update package 462
HP TippingPoint 110 T200E	Digital Vaccine 2.5.2.8348

The chosen devices represent most of the industry leading solutions according to the Gartner Magic Quadrant for Network Intrusion Prevention Systems[26].

3.1.1 System Configuration

The tested network security devices are rarely usable "out of the box" without requiring custom configurations depending on the desired functionality and the network involved. As the devices operate differently, have differing opinions on the nature of threats and provide different services for the end-user, we can not create completely comparable configurations and traffic termination policies between the devices. For a workaround, we established a set of configuration principles to represent well hardened retail systems as well as possible. These rules are in descending order of importance:

1. A device must pass through standard traffic.
2. A device must block the designated exploits when detected.
3. A device must pass through non-malicious traffic using the protocol used by the exploit.
4. A device must block the designated exploits when evasive techniques are used.
5. A device must pass through non-malicious traffic using the protocol when evasive techniques are used.

Configuration is a live procedure and is regularly altered as devices are updated with improved threat databases and firmware. Vendor provided updates are applied in weekly cycles to our testing environment. Traffic termination policies are therefore constantly reviewed and improved towards our desired functionality.

According to our configuration principles, an optimal device would be one which could perfectly inspect the traffic and predict how the endpoint will evaluate it regardless of the abnormalities and routing involved. An ideal device would only drop traffic at the point of identifying a certain attack.

In practice, the detection signature quality between devices varies and effective evasion techniques can fool the inspection – rendering the detection mechanism effectively useless for worst cases. Sometimes systems offer a possibility to terminate traffic when uninterpretable network anomalies are encountered. Since we prioritize blocking possible attacks over allowing similarly malformed clean traffic through, we activate these kind of anomaly filters if the alternative would allow more attacks to pass through.

The configuration principle is followed *by our best human effort* and is likely prone to some errors. Either due to configuration errors or inadequate detection quality of the devices, we were not always able to follow the configuration principles stated above. Most notable cases would include DUT 5 where we were not able to block the attacks without blacklisting all traffic using the same protocol – and DUT 2, where we were unable to find a configuration capable of countering our attack shellcode obfuscations.

Despite these limitations, we consider our configurations to be suitable for research. Hardening against our limited set of threats and their deviations is not a realistic practice in most live deployments and *there is no policy intentionally set more lax* than any hardened default configuration would suggest for use. Even still, we avoid taking a stance for estimating the capabilities of any single DUT, but treat our systems as anonymous "configurations" resembling good but imperfect real-life deployments of the devices.

3.2 Software Setting

3.2.1 Network Configuration

The network contains two virtual switches connected to the stack of physical security devices. The DUT's work between them as inline devices and pass their traffic to another virtual switch routing the attacker to a network of victim computers. The software switches are programmed to route the reply traffic back through the interfaces and DUT's they were received from, simulating a symmetric routing configuration keeping nodes unaware of the existence of multiple routing options.

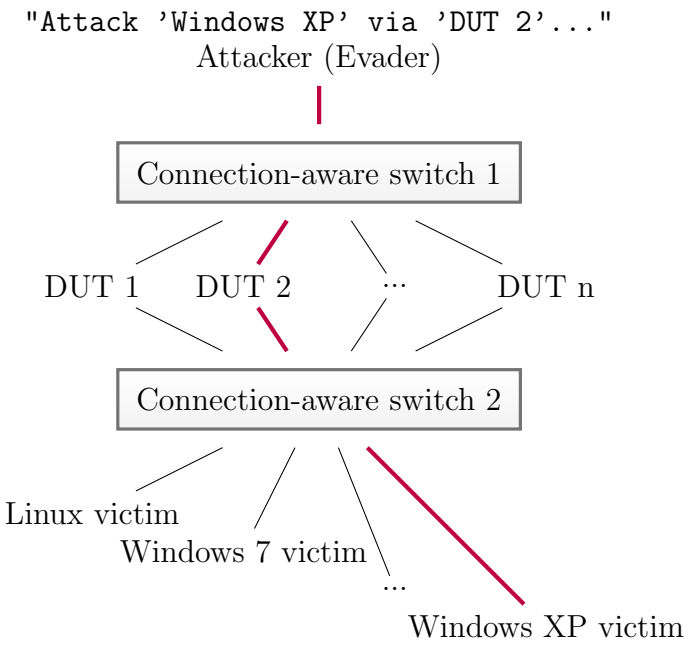

Figure 3.1: Network routing configuration in our experiments.

3.2.2 Stonesoft Evader

Our primary tool for testing is the commercial version of Stonesoft Evasion Readiness Test tool[1]. Slightly limited version of the tool was publicly released in July 2012 by the name of "Stonesoft Evader"[7]. The product is based on an internal testing tool implementing a custom TCP stack capable of generating abnormal network traffic to evade the connected network security devices. Evader contains set of public exploits (such as an implementation of CVE-2008-4250 and CVE-2004-1315 [15, 16]) and a database of known deviations (evasions) applicable to the traffic generated by exploits. Depending on the used protocols in the exploit, Evader libraries offers a set of application- and OS-specific evasions possible within the stack. For example, CVE-2008-4250 exploits vulnerability in Microsoft Server Message Block, which allows us to perform evasions for MSRPC, SMB, NetBIOS protocols and within the lower layers of transport, Internet and link connectivity. As vulnerabilities like CVE-2004-1315 exploits a web application, Evader can adapt to it by offering HTTP evasions for the application layer.

As of August 2012, the publicly available Evader supports 35 different families of evasion techniques. These are sometimes called "atomic evasions". 15 of them are generic TCP and IP layer evasions available for both exploits presented, 11 of which are only usable within the SMB stack of CVE-2008-4250 exploit and 9 techniques to mask the HTTP query of CVE-2004-1315 exploit implementation. The commercial edition we are using in this study supports 3 additional techniques for CVE-2008-4250 attack and 9 more techniques below the transport layer. These techniques are listed in appendix A.

Each evasion family contains a set of possible options and parameters to define the exact modifications they do for the traffic flow. Possible parameters often involve randomness and 32-bit offset options, making the possible space for deviations too big for meaningful, complete traversal for any DUT.

1. Working name "Predator". This study was performed with Predator version 4.2.0

In addition to the set of possible evasion deviations, Evader contains support for pinpointing individual techniques to individual stages – if the traffic generated can be divided to interactive stages of execution. For example, one can inject random TCP chaff traffic to affect only the SMB handshake phase of CVE-2008-4250 exploit.

```
$ ./evader --attack=conficker
             --src_ip=10.1.10.8 --if=eth24 --src_mask=16
             --dst_ip=10.1.0.164 --evasion=ipv4_frag,8
             --evasion=tcp_seg,2 --evasion=smb_seg,1

Info: Using random seed 44lus5dyCiI
  - IPv4 fragments with at most 8 bytes per fragment
The following evasions are applied from stage netbios_connect
to end:
  - TCP packets are segmented to contain at most 2 bytes of
    payload.
The following evasions are applied from stage msrpc_bind
to end:
  - SMB writes are segmented to contain at most 1 bytes of
    payload.

Error: MSRPCServerExploit::MSRPCBind() - SMB session setup
failed.
Error: Exploit running failed
302: SMB session setup failed.
$
```

Evaluation 3.1: A sample execution of Evader using CVE-2008-4250 vulnerability

Depending on the exploit used, a successful attack can be verified by

several means. An attack might return a usable shell window, return a non-authorized file or crash the target computer. In addition to testing for vulnerabilities, exploit implementations contain corresponding method of testing for false positives [2] using the protocols used in attacks. These clean requests can be manipulated with the same set of evasions than what are available for malicious exploits to estimate whether a connection termination happens due to network anomalies or genuine attacks. Another way of evaluating traffic terminations could be by monitoring the IPS event logs corresponding to the generated attacks. In addition, Evader has support to analyze the traffic responses it has received for each attack, providing real-time insight of possible reasons for individual traffic terminations.

Evader can also be used by external applications. Although calling the program via command line via automatically generated scripts is sufficient for most testing purposes, Evader has been extended with tools like web interfaces for better usability and the Mongbat fuzz testing tool presented in section 3.2.3.

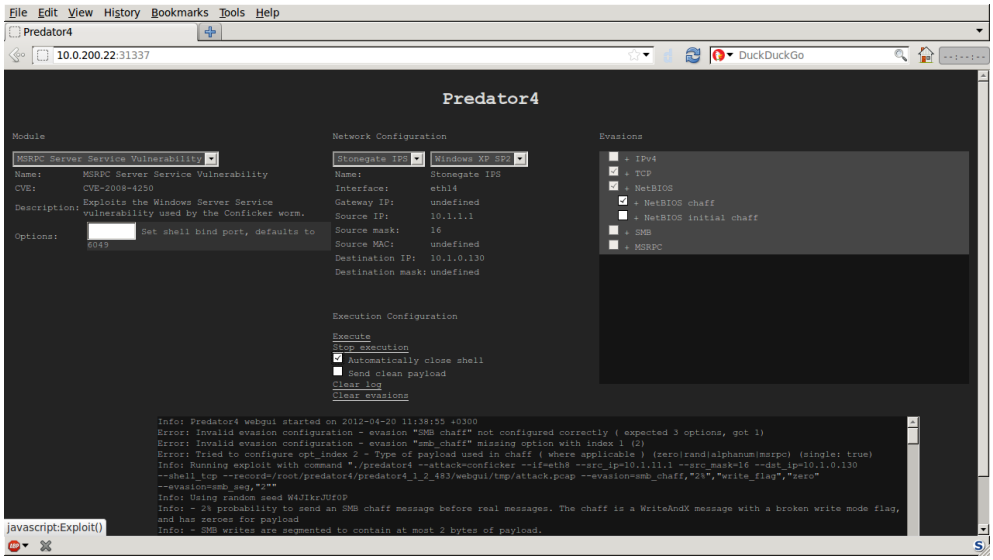

Figure 3.2: Evader Web-GUI

2. For example, a non-malicious GET request to a HTTP server

3.2.3 Mongbat

Mongbat is an application for Evader designed to automate execution of multiple, parallel randomized attacks supported by the framework. The primary use for Mongbat is to fuzz search working exploitation tactics in order to improve product signatures and traffic normalization. To sniff for possible network errors, Mongbat normally probes the target computer with a clean payload without any evasive techniques prior to each attack. Once initialized, Mongbat performs the attacks as instructed and prints out the evasion combinations it discovered to work against the target network configuration.

As of current, Mongbat has some extension support for guiding and limiting the set of generated evasions we experiment with. This is mainly used to make the search more efficient, as we can disregard irrelevant data flow configurations without testing [3]. As Evader reports the connection stages it is able to execute, we can do deductions for the reasons why a DUT might terminate individual connections.

3. For example, forbidding setting an encapsulating TCP segment size greater than needed for contained SMB segment

Chapter 4

Methodology

The core challenge for our work was to provide a reliable, repeatable and descriptive way of measuring the performance of devices under test.

As configurable multi purpose devices, differing and unexpected behavior in tested devices can be anything from networks errors and badly designed tests to humane error in configuration. The configurational challenges were addressed in section 3.1.1. In this chapter, we introduce the testing practices implemented in our experiments and how they improve the reliability of this study.

4.1 Test-set Structuring

To establish comparable data from the performance of security devices, we needed methods for generating series of tests with comparable and descriptive outcomes.

4.1.1 Pre-created Test-suites

The first experiments we executed consisted of pre-known series of evasion techniques represented as a list of parameters to be given as arguments for Evader. The main motivation of engineering a specific set of tests are with

comparability of devices receiving similar sets of traffic and for testing certain families of evasions specifically against a security device.

Another benefit of pre-created tests are with measuring the response for the CERT-FI evasion disclosures mentioned earlier, since the tested suites utilized were based on the disclosures of 2010 and 2011.

4.1.2 Fuzz Testing

Prime reason for developing Evader was to use it in a flexible test generator for improving the inspection quality of intrusion prevention solutions. Generating sets of malicious traffic flows randomly has proven to be an effective way of revealing errors and design flaws in company's own products and can be used to discover similar vulnerabilities in the other devices under examination. For this, we use the Mongbat application described in section 3.2.3.

Random executions give us an insight to the inspection quality of devices when the exact form of an attack is unknown. Performance can be measured with percentages of successful evasions, the amount of required attempts before a successful breach or through false positives caused by hardening. An important benefit in randomized test generation is with the enormous space of potential evasion permutations we can test with. With thousands of test cases, the importance of individual tests and their methodological correctness mitigates and we can give focus for the bigger picture.

4.2 Test Isolation Techniques

As our experiments generate much of unconventional and often malicious traffic, we apply techniques to isolate individual test cases from each other. Randomizing source IP addresses and shell binding destination ports, testing for connectivity errors and test repeating aim to increase the reliability of our experiments.

4.2.1 Source IP and Destination Port Number Fluctuation

Depending on the device, it is possible that continuous flow of suspicious traffic and detected exploit successes may trigger a change in defensive policies. Issues like Denial of Service defenses or automatic blocking of traffic towards opened shells can prevent us from repeated testing of evasion detection capabilities directly.

Since the toolkit we are using is fully in control of the socket behind the attacking end of the security devices, one way of reducing non-interpretable results is to change the source IP address and desired exploit binding ports between tests. Depending on the experiments, we either used incremental or randomized variation of fluctuated sources and target ports to fool DUT's to evaluate each attack in their own context instead of as instances from a series of attacks.

Changing the targeting parameters also adds the additional benefit of ensuring that the possible replies from the target computer are linkable to the current evasion being tested, instead of an old or parallel connection attempt without standing connection closure.

4.2.2 Connection Probing

By connection probing, we are trying to verify that the attempted connections should be possible and that the target system responds to non-malicious queries under shared circumstances. Our tests include a possibility to inject a clean payload instead of the exploits used in real attacks. In practice, we send a connection probe before each exploit payload to verify whether the possible connection error happens due to a disabled target system or because of an intervention by DUT.

Additional testing by combining connection probing with evasion techniques can also reveal us information of the nature of blocking rules in use. Should we see a blocking happen to a clean payload with no evasions, we can

make assumptions such as that the DUT works by blocking entire protocols. But when evasions are applied, we can reason that the blocking decisions are made due to the anomalies in traffic instead of the encapsulated payload. This reduces the need for monitoring DUT event logs for individual test cases and helps us upscale our tests as individual devices can be treated as "black boxes" during evaluation. Deducing device performance by their responses to different traffic can be used for configuration guidance as described in section 3.1.1.

4.2.3 Incremental Repetition

A number of reasons can make an individual test fail. Our exploit can leak memory within the target systems when successful, security devices can adapt their behavior as the test progresses, our randomizations within traffic may not get interpreted at all and the false negative results may have been caused by reasons other than misbehavior of the DUT. For example, running parallel attacks through multiple defenses can cause a race condition in a single target computer where a tester might not understand an opening to have been caused by a parallel tester. The testing suite itself may contain bugs making some individual tests produce unusable results.

To mitigate this issue, we repeat the test results interpretable as inadequate behavior from the DUT. Individual tests in a set measured as false positives from clean traffic probing and tests indicating false negatives are added to a new subset of tests – which are to be repeated after the initial execution is performed. The new results from running the subset are then appended to the original set of results, replacing the conflicting test outputs with fresh ones.

This procedure is recursively repeatable. Each round of repetition gives us increased assurance whether our observed shortcomings are solid. Selective repetition for confirming inadequate behavior also biases the evaluation into being more forgiving towards the tested devices. We are more likely to miss holes in the defenses than to report a nonexistent flaw.

4.2.4 Payload Obfuscation

The quality of traffic inspection depends on more than indications of blocking known types of payload. In practice, attackers employ methods of dynamically mutating the attack shellcode to fool the defenses. Encryption, NOP injection and code order randomization are among the simpler methods of fooling insufficient signature analyzers. A decent security system can detect attempted uses of vulnerabilities instead of individual exploit implementations.

Our experiments support obfuscating the malicious payload we are trying to use. Passing an obfuscated exploit through a security device when a non-obfuscated payload is dropped can tell us about the quality of the inspection performed by the device. Being capable of blocking both attacks with and without obfuscation while letting the corresponding clean traffic through signals that the device has a through understanding of the nature of the traffic involved and versatile signatures against threats. It is preferable for vulnerability matching techniques to be versatile enough to work against generic threat types in addition to known and disclosed exploit implementations.

Chapter 5

Experiment Implementation

In this chapter, we introduce three different and repeatable experiments to evaluate the threat our evasion techniques portrays, present details from our implementation and provide samples from results discovered when they are applied against the configurations described in chapter 3.

The results are presented for every DUT configuration available at the time of the experiment. The experiments were performed in series, based on the hardware described in section 3.1 and configured by the principles presented in section 3.1.1. Although being the same across all three experiments, the configuration order in results has been changed. This was done as the results are not intended to provide a benchmark of the devices.

All test cases used in this study are based on the MSRPC[16] exploit available in Evader. This was used as the underlying networking stack is more complex for broad multi-layer evasion study, the signatures in DUT's for the vulnerability appeared to work well and as the earlier AET disclosures of 2010 and 2011 used in section 5.1 were made by using this vulnerability.

5.1 Experiment 1: Pre-generated Test-set Comparison

First executed tests were to evaluate how the examined vendors have reacted to the Evader exposures disclosed in 2010 and 2011. By the time of the experiments, the first evasions with corresponding packet capture recordings had been known and addressed by vendors for nearly two years. The two sets examined here contain 23+121 evasions masking the MSRPC-exploit available in Evader.

In addition of replaying the disclosures, we improved the metrics by adding a round with shellcode obfuscations to the exploit as presented in 4.2.4 and rounds of false positive testing by sending the end system a clean payload with and without the evasions as described in section 4.2.2.

In essence, we created six connections with each test case:

1. Initial connection probe to test whether a shell binding destination port replies to queries from an IP address.
2. A clean payload with evasions to test for false positives [1].
3. Second connection probe to confirm another port being open in the target system.
4. A known malicious payload with evasions.
5. Third connection probe targeted to a third port within the target system.
6. An unknown, randomly obfuscated malicious payload using the same vulnerability as the prior attack.

To improve the reliability of measurements, we also performed incremental repetition as described in section 4.2.3. All DUT's went through at least 4 incremental repetition rounds over inconsistent and false negative results

Our results are presented in appendix B. The first column of the tables indicates an unprotected target device. As the clean route is fully exposed to

1. A drop action done by DUT due to a false alarm.

attacks, it results close to 100% of successful attacks. Rest of the 10 columns correspond each to an individual DUT as shown in section 3.1.

As we generated six tests for each evasion combination, we can observe four distinguished result types from each test set. Optimal test results are ones where all four connection probes went through the DUT regardless of evasions used and the two breach attempts were stopped. These are marked as blue cells in the figures of appendix B. Slightly worse results are marked as yellow cells, indicating a situation where non-evading connection attempts pass through, but any connection with evasions gets dropped regardless of their payload. We assume these are likely false positives due to anomaly-based termination. Inconclusive results are marked in black, indicating a test set where no clean connection probe has passed through. Successful penetrations with evasions are marked in red and are considered to be the worst case scenario. There is no distinction in color coding whether the breach was possible with or without the shellcode obfuscation techniques, though they are separated in the summary tables.

It was discovered, that the last evasion techniques in the first set were able to render a target system unresponsive upon breach, which prohibited reliable batch testing after a successful attack. The pattern is observable from the ordered results of devices C (unprotected system), 6 and 9 in figure B.1.

5.2 Experiment 2: Random Evasion Search and Analysis

The second experiment was designed to measure the effectiveness of increasing evasion sophistication against the target configurations. For this, we executed a series of fuzz tests against a DUT while increasing the amount of applied atomic evasions. For the first round, we applied a random evasion with randomized parameters for each attack. The second round contains two random evasions per attack and the third round was performed with three

randomized evasion techniques.

We configured the Mongbat evasion search tool to perform attacks for each operational device for a given time and given evasion depth. Running Mongbat 15 minutes against each DUT available during the test resulted to a total of 26138 performed attacks with three levels of evasion complexity. From these, we counted 2318 successful penetrations compared to 22749 halted attacks. A number of results were discarded as inconclusive, as we were able to detect a number of system malfunctions and networking errors with connection probing prior to each attack.

As the test was performed as a random search, we re-evaluated the results to remove likely duplicate evasion techniques from the data set. To remove entries considered as "minor changes", we normalized any numerical parameters away from the discovered attack vectors of a DUT and removed the resulting duplicates from the evaluated data.

We also considered the possibility of higher level evasions being explainable by a set of working lower level evasions being present as composites of more complex evasions. To reduce the likelihood of this, we removed any results which contained a full subset of any evasion capable of evading the IPS.

After the reductions illustrated in figure 5.1, we were left with a total of 814 successful attacks with differing evasions. It is notable, that DUT 9 was offline during the testing and DUT 3 was found to be significantly more vulnerable against evasions, accounting for 602 of the unique breaches. Two of the devices did not pass any attack with our configurations. The rest showed a comparable level of vulnerability.

From the results, we see that executing the tests with time constraints resulted in ample variance in the amount of samples we were able to generate. Some devices may terminate different traffic in different states, resulting with significant waiting periods when exploiting issues with extended connection handling [2]. Other reasons for variance could be simple software failures

2. For example, TCP TIME-WAIT evasions.

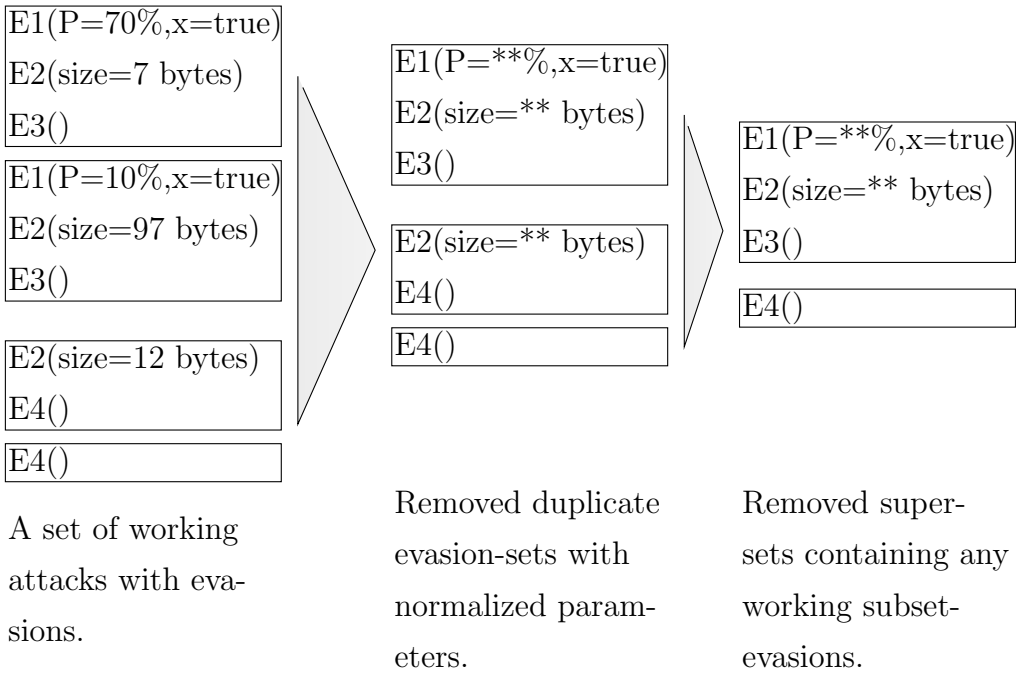

Figure 5.1: An illustration of reductions used to analyze experiment 2 results.

	Attacks	Successful exploits	Unique evasions	Coverage %
DUT 1	2271	65	46	97,1
DUT 2	2810	0	0	100
DUT 3	2944	1835	602	37,7
DUT 4	2602	129	121	95
DUT 5	975	44	15	95,5
DUT 6	1588	47	41	97
DUT 7	2114	0	0	100
DUT 8	4975	130	70	97,4
DUT 9	-	-	-	-
DUT 10	5859	68	39	98,8

Table 5.1: Refined results from experiment 2

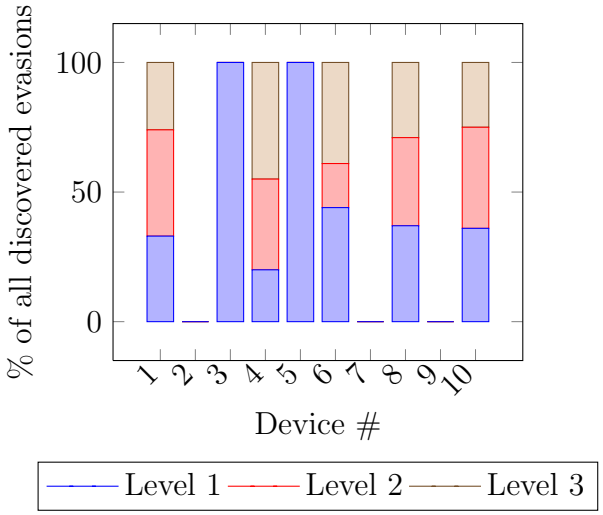

Figure 5.2: Distribution of successful attacks by amount of evasion-layers in experiment 2

within testing tools or disproportionate amount of time intensive tests within the randomized set.

Although stopping more than 95% of generated attacks show proof that the devices are improving the level of protection in a network, we do not draw comparable conclusions on the DUT performance from this set due to incomparable data sets. However, when one discovers a working evasion technique for an attack, it can be used to fool most or all similar devices. As an average for our DUT's, we were able to discover 93 different, working evasion techniques in 15 minutes of benchmarking.

5.3 Experiment 3: Random Evasion Search with Heuristics and Analysis

In Experiment 2, we calculated a set of evasions randomly and retroactively analyzed how many techniques in the set were distinctive. As the possible combination space available within the tests is too large for com-

prehensive traversal [3] and individual executions with identical or resembling configurations do not necessarily render similar results – we needed an alternative method for identifying complex evasion techniques and their effectiveness.

Our new approach takes the random search from section 5.2 and attempts to estimate the effectiveness of individual evasion combinations as they are discovered. For this purpose, we implemented a custom extension for Mongbat to alter the evasion search criteria whenever a working combination is discovered. Each discovered evasion is then used to calculate possible evasion sub-combinations. Sub-combinations are then queued for repeated testing to estimate the lowest complexity required for the technique to be effective. This can be considered as a type of incremental repetition described in section 4.2.3, as possible breaches go through a refinement process. An illustration of this process is presented in figure 5.3.

As a result, we get log entries of successful attacks with pointer data referring to the original, randomly discovered evasion used to generate the sub-evasions. These log files are then analyzed to evaluate the properties of each discovered technique. From the data, we calculate the following indicators for evaluation:

- How many individual tests were executed within the time constraint?
- How many of those tests are interpreted as reliable [4]?
- How many randomly searched evasions can be interpreted from the log files?
- How many successful attacks in total were performed with the searched evasions and their derivatives?
- How many discovered evasions were not repeatable without using *at least* two of the composite sub-evasions simultaneously?

3. Discounting possible parameter permutations and possibilities to target individual evasions to 2-8 different stages of communication, 48 evasion families offered by the test tool alone offer more than 281 trillion possible evasion combinations.

4. Test reliability is assumed when the end-point answers to a clean network connection test as described in section 4.2.2

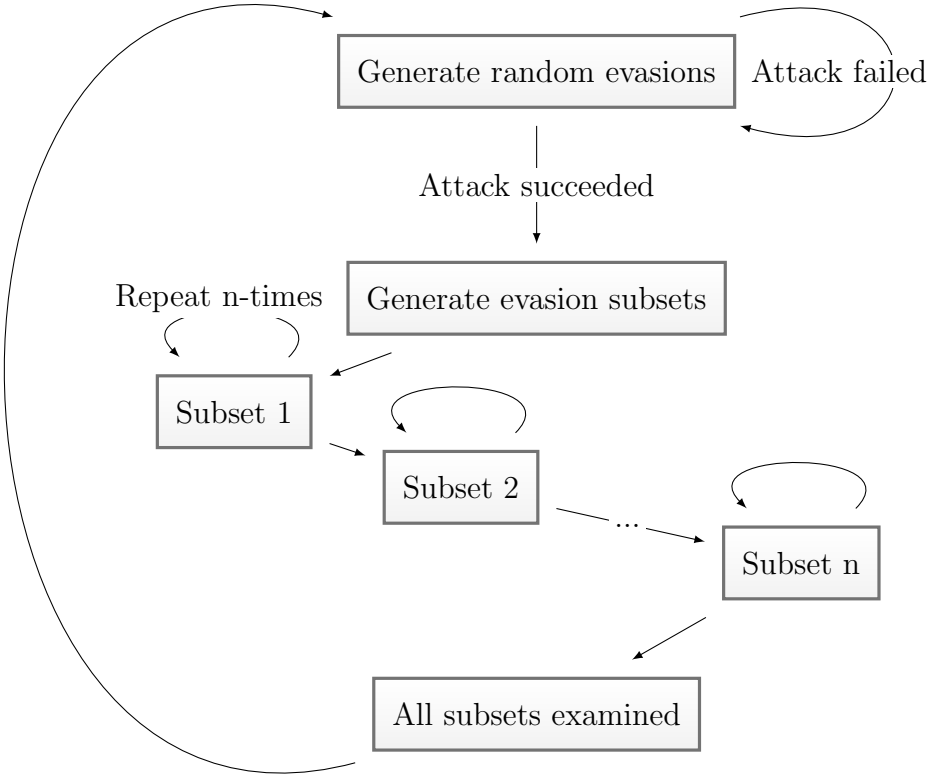

Figure 5.3: Experiment 3 evasion generation and evaluation process

We applied this method for each device under test. We set the search function to look for 3-layered random evasion permutations and to attempt repeating each discovered sub-permutation 10 times. As three-layered evasions can be divided into

$$\binom{3}{1} + \binom{3}{2} + \binom{3}{3} = 7$$

unique subset combinations, each randomly discovered evasion are subjected to be re-evaluated with 70 rounds of refinement. The results from this are presented in table 5.2 [5].

	Total Attack Samples	Completed Tests	Failed Connection Tests	Discovered Evasions	Successful Evasion Repeats	Non-reducible Evasions
DUT 1	2599	2351	248	73	555	8
DUT 2	1476	1473	3	0	0	0
DUT 3	2485	1434	1051	55	1233	2
DUT 4	3639	3240	399	65	415	6
DUT 5	2534	2182	354	53	749	0
DUT 6	2046	149	1897	3	3	3
DUT 7	4260	4260	0	75	1233	1
DUT 8	2322	731	1591	24	237	0
DUT 9	2882	2796	86	90	692	4
DUT 10	3849	2819	1030	73	1233	0

Table 5.2: Summary from experiment 3

5. No connection probing was done for DUT 7

Chapter 6

Evaluation

In this chapter, we explore the results obtained in chapter 5 further and provide possible interpretations touching the questions presented in section 1.1. In addition, we examine the limits and reliability of the experiments we have performed.

6.1 Defending against AET's

Despite initial disclosure of the tactics in late 2010, repeating the sets two years later in section 5.1 show most of the configurations to remain vulnerable to same techniques. With most cases of successful attack terminations, one can suspect too tight policies for network anomalies, as non-malicious payloads get dropped when analogous evasive tactics are applied.

In appendix B, we see DUT 6 and 7 having overly strict policies warding off many or nearly all legitimate connection attempts to the protocol being exploited. These could be disqualified from our evaluation due to a possible misconfiguration. Interestingly, we still discovered a number of attacks passing through these defenses. Not only were the evasions able to mask the attack from inspection, they circumvented the apparent protocol block used by the defense.

In section 5.1, every test executed in device 3 resulted to a successful

attack. However, from the tables in appendix B we can observe that most of the successful attacks were discovered when the payload obfuscation was active. Performance of DUT 3 is significantly better without obfuscation, indicating a deficiency in the vulnerability signature of the device.

The defensive performances vary much across the configurations. 3rd, 5th, 6th and 9th DUT fail to counter nearly half of the techniques disclosed in 2010 and 2011. Devices 2, 4, 8 and 10 show complete or near complete defense within the same tests. Although with devices like DUT 8 and 10, much of the protection can be attributed to overtly strict policy regarding network anomalies as the rate of false positives appears significant.

Fuzz testing in section 5.2 and 5.3 seem to confirm variance in AET defense capabilities, though experiment execution does not permit similar comparison between the devices. Based on this, we argue that:

- We see great variance between the evasion defense capabilities between our tested configurations with likely links to the inspection quality of the devices under test.
- Protecting against the AET's examined is possible and there are devices providing adequate defense.
- Advanced evasion resistance may degrade fault tolerance in network configurations.
- The relative age and enduring performance of the techniques indicate that defending against the type of evasive tactics examined is an overlooked or overwhelming challenge for many network-based security devices.

6.2 AET Effectiveness

As we discovered in section 5.2 and 5.3, majority of techniques discovered with random search were explainable as added complexity to other atomic evasions capable of breaching the defenses on their own. The prevalence of weaknesses against attacks considered "simple" within this study does

little to prove this kind of threat to be much more than theoretical for now. Although we find some evidence of non-reducible hybrid-evasions, the amount of sampling, repetition and external review for the findings do not give strong justification for treating most of them as uniquely effective combinations.

However, increasing the amount of techniques used also increases the likelihood of an attack passing through the defenses. Therefore, stacking effective anomalies together can be assumed to increase the likelihood of successful attack and penetrating multiple configurations with the same attack.

Time required for finding effective evasion combination against a single device is within or less than seconds with current hardware and tools. Collecting enough knowledge of multiple weaknesses and their rate of success with variations should not be an issue if an attacker has access to the devices and tools involved. Dropping the biggest deviations from the average evasion coverage illustrated in figure 5.1, approximate 4% of generated attacks flow through most devices.

6.3 Reliability of evaluation

The prime issue regarding the reliability of our study is whether the configuration of the devices involved has been kept comparable. For this reason, we avoid making statements on the defense capabilities of individual devices and their comparability to each other. Given the number of devices used and humane difficulty of hardening them in equal effort, we can not assume them to be fully representable of up-to-date maintained deployments.

Nevertheless, as the configuration policy described in section 3.1.1 was built to counter the threats examined as well as we could, we expect the examined devices *to be more hardened* against the examined threats than most real life deployments. Experiment results from most DUT's are similar, indicating equal and balanced approach towards configuration. Given the number of devices involved, we estimate broad misconfigurations unlikely and see the aggregate results as representable sample of the current state of

AET defense.

Using random search as basis for experiments section 5.2 and 5.3 is problematic. The atomic evasions involved are not equally effective, significant or configurable. The line between what is considered an atomic evasion or only a configuration of an evasion is often artificial. Our random search is primarily focused in high-level atomic evasions. Therefore, our search is biased heavily towards individual "families" of evasion techniques rather than seeking for good implementations. This is problematic with the heuristics used in section 5.3 as the findings tend to centralize towards few working, less configurable evasion families and bypass interesting variations requiring more fine tuned configuration. This limits our ability to examine and distinguish the individual evasive tactics and their effectiveness. These heuristic limitations and small sample size involved can be seen to weaken our evidence.

Chapter 7

Discussion

7.1 Implications of the state of defense

At the time of testing, nearly all configurations showed repeatable vulnerabilities against numerous low cost evasion tactics. Most of the tested techniques have been known for over a decade, making most claims of network based evasion resistance in the near future to be unconvincing at best. Several network connected devices offer limited freedom for host-based hardening [1], making network based defenses essential against emerging vulnerabilities. As of now, a well crafted AET could easily bypass most such defenses.

Wide possibilities of deliberate malformations to network traffic and other security concerns have raised critique towards the network robustness principles we have internalized for more than 30 years[8]. Though redesigning fundamental standards of the Internet is time consuming and possibly inappropriate to address security concerns, restricting traffic within controlled networks could provide a level of transparent defense against evasive techniques. In addition to active traffic monitoring security devices, we could design our networks to normalize their traffic as strictly as possible. Usable tools could include reverse proxies, VPN technologies, QoS technologies and whitelists to limit the space of evasive techniques possible within an admin-

1. For example, most mobile handsets and industrial control systems.

istered network.

7.2 Implications of AET characteristics

From what we have demonstrated, AET's are successful in discovering repeatable attack vectors to the networks we are attempting to protect. Although much of AET's can be countered with anomaly based detection, our configurations saw an increase in attack success when multiple techniques were applied simultaneously. In addition to the increased success for breaking known networks, low cost or penalty for deliberate abnormalities opens new possibilities for potential AET use cases.

An obvious use for composite evasions would be to increase the breadth of vulnerable configurations, reducing needs for preceding survey of the target network. Frequency analysis for working techniques, improving the search to find optimal parameters and evaluating the alert logs created by the security devices could be used to dynamically develop sets of working evasive attacks uncomprehendable by multiple network security devices.

The discrepancy between the capability of network security device packet inspection and the actual implementation of networking stack at host endpoints could affect malware proliferation and communication design as well. Tracking and detecting contaminations by relying to network-based signatures can be interfered with similar techniques, which was the case with the DGA-utilizing crimewares presented in the Damballa report[17].

7.3 Future Research

In July 2012, the core tool used in this study was released as a free downloadable testing environment[7]. Although free Evader is slightly more limited than the commercial one used in this test, nearly all evasions used in section 5.1 are recreatable with the set available in the public version. Randomized tests in 5.2 and 5.3 utilize an extended set of evasions. Though

hit-frequency would likely be lower with free version, we expect otherwise similar results when applied with the reduced evasion set.

Our study takes a broad view towards evasion techniques and what can be considered as "advanced" evading. Whilst we observe a set of patterns and evidence emerging from the experiments performed, we take a minimal look upon the generated evasions themselves and what makes them effective. There would be much gain from analyzing individual techniques and their structure. Optimizing used probability parameters, byte-offsets, examining evasion classes and shared traits could signal emerging vulnerabilities and bottlenecks in devices and protocols involved.

Another overlook is with the effects of AET hardening to the performance of individual devices. As original motivation for developing Evader was to harden intrusion prevention technologies against evading threats, evaluating individual devices can be used to improve other network security devices and understanding the costs of effective protection.

AET's offer an improvement for the delivery vector of malicious data. As of current, they are hardly needed to perform successful host infiltrations. It is often good enough to use easier, well known tactics and penetration techniques. Therefore, it is arguable whether evasive techniques are widely used in the public Internet. Apart from individual samples of highly evasive malware[17], we do not know if they contribute to a trend. Acquiring and analyzing live streams of Internet traffic for possible traces of intentional ambiguities and their quality appears as an unexplored topic. As evasive techniques appear to cause minimal trouble for the traffic interpretation at endpoints, there should be little reason to exclude them from possible malware proliferation mechanisms.

Limiting our study to a normalized lab environment leaves us with distant grasp of real networking realities. Much of the false positive traffic terminated as anomalies could be ordinary in other real-life networks. Combining knowledge of evasion resistance requirements with known network characteristics could be useful for evaluating individual device suitability in different

environments.

Our study has also delimited much of the existing capabilities of Evader toolkit out from our scope. Evading attacks targeting other vulnerabilities, evasions in IPv6 environment and testing against other potentially evasion resistant techniques [2] could expand our understanding on these threats.

2. For example, traffic proxies and web application firewalls.

Chapter 8

Conclusions

In this thesis, we presented an insight on how network traffic is inspected and how most common network security devices protect the network. We defined Advanced Evasion Techniques as "a set of expensive and non-trivial means to fool network based threat detection devices" and explained why we believe the experiments in this study fills this criteria. We examined the related research on evasions and countering techniques. Then we describe how the scale of this study contributes upon the field of network security evasion research.

Using our custom testing environment and error mitigating methods, we performed three experiments to evaluate the threat of AET's and how ten up-to-date network security devices from industry leading vendors can defend against our mutated AET's. We discovered that several devices have chronic difficulty defending against attacks using evasive techniques and that increasing complexity of used techniques seems to increase the odds of penetrating the devices. We also observed systems with much better evasion resistance. Although, this often appears with an added cost of more false positives and reduced tolerance for network anomalies.

In addition, we examined the costs of increasing evasion complexity by measuring the effectiveness of multi-layered evasive tactics. We found some evidence of added layers making previously countered evasions undetectable;

though we avoid making strong claims due to small size of analyzed test sets. More importantly, we found little or no cost in adding layers of counterable evasions to an attack. These indicate that effective evasions can mask otherwise detectable anomalies and improve the likelihood for a technique to penetrate one or multiple defenses.

This is expanded by defining scenarios where this type of attack could be useful and an estimate how current security devices can protect networks now and in the future. Due to minor readiness to an old threat, we do not see prevalent solutions to AET attacks in the near future.

We conclude by defining directions for further research. Our study is set to be a broad overview and has little consideration towards the individual techniques, their parameters and underlying reasons for their effectiveness. Neither do we focus on individual devices under test nor how their performance could be improved. Prevalence of traffic transformations classifiable as AET's and whether we could engineer individual attacks penetrating device protections in great numbers are seen to offer intuitive directions for new research as well.

Bibliography

[1] Stonesoft antievasion website. http://aet.stonesoft.com/.

[2] Department of defense standard internet protocol. RFC 760, January 1980. http://tools.ietf.org/html/rfc760.

[3] DARPA internet program protocol specification. RFC 793, September 1981. http://tools.ietf.org/html/rfc793.

[4] Architectural principles of the internet. RFC 1958, June 1996. http://tools.ietf.org/html/rfc1958.

[5] Sploit, a mutant exploit generator, 2006. http://www.cs.ucsb.edu/~seclab/projects/sploit/.

[6] Damballa discovers advanced evasion techniques being used by six crimeware families to carry out global cyber attacks, February 2012. http://www.damballa.com/press/2012_02_28PR.php.

[7] Stonesoft evader, July 2012. http://evader.stonesoft.com/.

[8] Allman, E. The robustness principle reconsidered. *Queue 9*, 6 (June 2011), 40:40–40:47.

[9] Balzarotti, D. *Testing Network Intrusion Detection Systems*. PhD thesis, Politecnico di Milano, 2006.

[10] Bidou, R. IPS shortcomings. In *Black Hat US Proceedings* (2006).

[11] Caswell, B., and Moore, H. D. Thermoptic camouflage, total IDS evasion. In *Black Hat US Proceedings* (2006).

[12] Cerf, V., Dalal, Y., and Sunshine, C. Specification of internet transmission control program. RFC 675, December 1974. `http://tools.ietf.org/html/rfc675`.

[13] Advisory on IDS/IPS device vulnerabilities that may circumvent protections. CERT-FI, October 2010. `http://www.cert.fi/en/reports/2010/vulnerability385726.html`.

[14] Advisory on further IDS/IPS device vulnerabilities that may circumvent protections. CERT-FI, June 2011. `http://www.cert.fi/en/reports/2011/vulnerability487536.html`.

[15] Vulnerability summary for CVE-2004-1315. National Vulnerability Database, November 2004. `http://nvd.nist.gov/nvd.cfm?cvename=CVE-2004-1315`.

[16] Vulnerability summary for CVE-2008-4250. National Vulnerability Database, October 2008. `http://nvd.nist.gov/nvd.cfm?cvename=CVE-2008-4250`.

[17] Damballa Labs. DGAs in the hands of cyber-criminals - examining the state of the art in malware evasion techniques, February 2012.

[18] Handley, M., Paxson, V., and Kreibich, C. Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics. In *USENIX Security Symposium* (2001), D. S. Wallach, Ed., USENIX.

[19] Lippmann, R., Fried, D., Graf, I., Haines, J., Kendall, K., McClung, D., Weber, D., Webster, S., Wyschogrod, D., Cunningham, R., and Zissman, M. Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation.

In *DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings* (2000), vol. 2, pp. 12 –26 vol.2.

[20] Lippmann, R., Haines, J. W., Fried, D. J., Korba, J., and Das, K. The 1999 DARPA off-line intrusion detection evaluation. *Computer Networks 34*, 4 (2000), 579 – 595. Recent Advances in Intrusion Detection Systems.

[21] Marty, R. Thor: A tool to test intrusion detection systems by variations of attacks. Diploma thesis, Swiss Federal Institute of Technology Zurich, March 2002.

[22] McHugh, J. Testing intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by lincoln laboratory. *ACM Trans. Inf. Syst. Secur. 3*, 4 (2000), 262–294.

[23] Mutz, D., Vigna, G., and Kemmerer, R. A. An experience developing an IDS stimulator for the black-box testing of network intrusion detection systems. In *ACSAC* (2003), IEEE Computer Society, pp. 374–383.

[24] Ollman, G. URL embedded attacks, attacks using the common web browser, March 2005. `http://www.technicalinfo.net/papers/URLEmbeddedAttacks.html`.

[25] Pescatore, J., and Young, G. Defining the next-generation firewall. *Gartner RAS Core Research Note* (October 2009).

[26] Pescatore, J., and Young, G. Magic quadrant for network intrusion prevention systems. *Gartner RAS Core Research Note* (December 2010).

[27] Pescatore, J., and Young, G. Magic quadrant for unified threat management. *Gartner RAS Core Research Note* (March 2012).

[28] Ptacek, T., and Newsham, T. Insertion, evasion, and denial of service: Eluding network intrusion detection. Tech. rep., Secure Networks, Inc., January 1998.

[29] Rain Forest Puppy. A look at whisker's anti-IDS tactics, 1999. `http://www.ussrback.com/docs/papers/IDS/whiskerids.html`.

[30] Ristic, I. Protocol-level evasion of web application firewalls. In *Black Hat US Proceedings* (July 2012).

[31] Roelker, D. HTTP IDS evasions revisited. *Sourcefire Inc* (2004).

[32] Rubin, S., Jha, S., and Miller, B. P. Automatic generation and analysis of NIDS attacks. In *ACSAC* (2004), IEEE Computer Society, pp. 28–38.

[33] Scarfone, K., and Mell, P. Guide to intrusion detection and prevention systems (IDPS), nist special publication 800-94. In *Recommendations of the National Institute of Standards and Technology* (February 2007), National Institute of Standards and Technology.

[34] Shankar, U., and Paxson, V. Active mapping: resisting NIDS evasion without altering traffic. In *Security and Privacy, 2003. Proceedings. 2003 Symposium on* (may 2003), pp. 44 – 61.

[35] Sommer, R., and Paxson, V. Enhancing byte-level network intrusion detection signatures with context. In *Proceedings of the 10th ACM conference on Computer and communications security* (New York, NY, USA, 2003), CCS '03, ACM, pp. 262–271.

[36] Song, D. fragrouter(8). `http://www.monkey.org/~dugsong/fragroute/`.

[37] Taleck, G. Ambiguity resolution via passive os fingerprinting. In *Recent Advances in Intrusion Detection*, G. Vigna, C. Kruegel, and E. Jonsson, Eds., vol. 2820 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2003, pp. 192–206. 10.1007/978-3-540-45248-5_11.

[38] Vigna, G., Robertson, W. K., and Balzarotti, D. Testing network-based intrusion detection signatures using mutant exploits. In *ACM Conference on Computer and Communications Security* (2004), V. Atluri, B. Pfitzmann, and P. D. McDaniel, Eds., ACM, pp. 21–30.

[39] Vutukuru, M., Balakrishnan, H., and Paxson, V. Efficient and robust tcp stream normalization. In *IEEE Symposium on Security and Privacy* (2008), IEEE Computer Society, pp. 96–110.

Appendix A

Used evasions

Protocol	Description
MSRPC	*MSRPC big endian*
MSRPC	*Group MSRPC fragments to a single lower layer send*
MSRPC	*MSRPC NDR modifications* - Set NDR types not related to endianness
MSRPC	*MSRPC request segmentation* - Set the maximum number of bytes written in a single MSRPC fragment
NetBIOS	*NetBIOS chaff* - Send extra NetBIOS packets to break the packet flow
NetBIOS	*NetBIOS initial chaff* - Send chaff NetBIOS packets when establishing the NetBIOS connection
SMB	*SMB chaff* - Send SMB messages to baffle inspection
SMB	*SMB decoy trees* - Perform extra SMB writes before each normal SMB write
SMB	*SMB filename obfuscation* - Obfuscate the tree name used in the SMB NT Create AndX method
SMB	*SMB random flags* - Randomize SMB flags
SMB	*SMB reserved bytes* - Randomize reserved bytes in SMB messages

SMB	*SMB response flag*
SMB	*SMB write segmentation* - Set the maximum number of bytes written in a single SMB write
SMB	*SMB WriteAndX padding* - Insert extra padding between the WriteAndX header and payload
TCP	*TCP bogus FINs sent after SYN* - Sends an out-of-window FIN after a SYN in TCP Connect()
TCP	*TCP bogus ACKs in syn-sent-state* - Sends a number of broken ACKs as a reply to incoming SYNACKs when the TCP socket is in the SYN-SENT state.
TCP	*TCP Chaff* - Send chaff TCP segments to baffle inspection
TCP	*TCP initial sequence number* - Set initial sequence number used by a TCP socket to 0xffffffff - n
TCP	*TCP timestamp option settings* - Set initial TCP timestamp option settings
TCP	*TCP multisegment* - Add nonstandard valid TCP segmentations to each normal TCP segment
TCP	*Disable TCP congestion avoidance*
TCP	*Disable TCP fast retransmit*
TCP	*Set the number of TCP retransmits and the TCP user timeout* - Set the number of retransmits and the TCP user timeout in seconds. When either is full, the TCP connection is reset
TCP	*TCP segment order* - Change the order of TCP segments
TCP	*TCP segment overlap* - Make segments sent in a single send() overlap
TCP	*TCP PAWS elimination* - Send extra TCP segments that are eliminated by PAWS in the destination stack
TCP	*TCP receive window* - Set the receive window of the TCP socket, used to force the other host to send small TCP segments

TCP	*TCP segmentation* - Set the MTU of a TCP socket, used to create non-standard TCP segmentation
TCP	*TCP TIME-WAIT decoys* - Open decoy connections from the same TCP source port before the actual attack
TCP	*TCP timestamp echo reply modifications* - Modify TCP timestamp options echo reply field
TCP	*TCP urgent data* - Set urgent data into TCP segments
IPv4	*IPv4 chaff* - Send chaff IPv4 packets interleaved with normal packets
IPv4	*IPv4 fragmentation* - Fragment IPv4 packets to given size
IPv4	*IPv4 options* - Send duplicate IPv4 packets with changed payload and some IPv4 options (Broken RecordRoute, incrementing packet number)
IPv4	*IPv4 fragment order* - Change the order of IPv4 fragments
	Supported, but unused evasions in this study
IPv6	*IPv6 chaff* - Send chaff IPv6 packets interleaved with normal packets
IPv6	*IPv6 dummy option headers* - Insert dummy IPv6 option extension headers
IPv6	*IPv6 fragmentation* - Fragment IPv6 packets to given size
IPv6	*IPv6 fragment order* - Change the order of IPv6 fragments
HTTP	*HTTP header linear whitespace* - Converts whitespaces in HTTP headers to LWS with a given probability
HTTP	*HTTP known user agent* - Sets the user agent to a widely known user (Firefox, Internet Explorer, Opera...)
HTTP	*HTTP request line separator* - Sets the separator used on the request line
HTTP	*HTTP request method* - Sets the request method (E.g. GET, POST, HELLO, SMTP HELO...)

HTTP	*HTTP request pipelined* - Send an ok request pipelined before the exploit
HTTP	*HTTP URL absolute* - Changes relative URIs into absolute URLs
HTTP	*HTTP dummy paths* - Add dummy paths into URL
HTTP	*HTTP URL encoding* - Encode characters in URL
HTTP	*HTTP request version* - Sets the request version number

Table A.1: "Atomic" evasions available in commercial Stonesoft Evader (Predator 4.2.0)

Appendix B

Experiment 1 results

DUT#	Clean	1	2	3	4	5	6	7	8	9	10	Ideal
Succesful connection-probes	54	69	69	69	69	69	54	0	69	56	69	69
Succesful evading connection-probes	14	10	9	12	16	5	7	1	11	13	4	23
Blocked attacks	5	23	23	19	23	23	18	22	21	16	22	23
Blocked obfuscated attacks	5	23	23	10	23	23	20	22	21	15	22	23
Blocked connection-probes	15	0	0	0	0	0	15	69	0	13	0	0
Blocked evading connection-probes	9	13	14	11	7	18	16	22	12	10	19	0
Succesful attacks	18	0	0	4	0	0	5	1	2	7	1	0
Succesful obfuscated attacks	18	0	0	13	0	0	3	1	2	8	1	0
Probe-block & succesful attack	0	0	0	0	0	0	0	1	0	0	0	0
Probe-block & succesful obfuscated attack	0	0	0	0	0	0	0	1	0	1	0	0
Probe-block & blocked attack	5	0	0	0	0	0	5	22	0	4	0	0
Probe-block & blocked obfuscated attack	5	0	0	0	0	0	5	22	0	4	0	0
Standard attack blocked but success with obfuscations	0	0	0	9	0	0	0	0	0	2	0	0
Fully passed test-sets	0	10	9	0	16	5	3	0	10	6	3	23

Table B.1: Results from Cert 2010_1 tests, taken 5th of July 2012

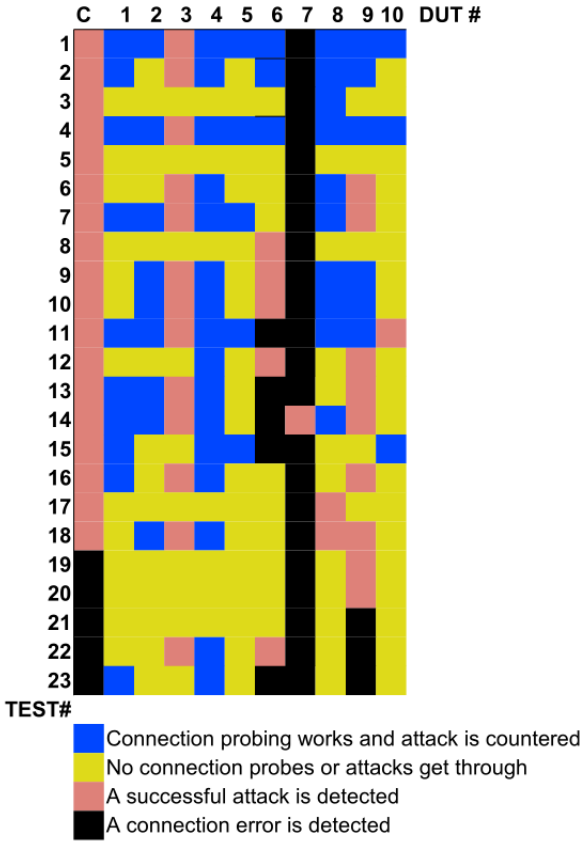

Figure B.1: Summary of test-results from cert2010_1 set

DUT#	Clean	1	2	3	4	5	6	7	8	9	10	Ideal
Succesful connection-probes	363	363	363	363	363	363	253	0	363	363	363	363
Succesful evading connection-probes	114	48	89	95	105	64	52	20	36	87	3	121
Blocked attacks	0	112	121	43	119	69	70	101	121	48	121	121
Blocked obfuscated attacks	0	112	121	27	120	68	74	101	121	47	121	121
Blocked connection-probes	0	0	0	0	0	0	110	363	0	0	0	0
Blocked evading connection-probes	7	73	32	26	16	57	69	101	85	34	118	0
Succesful attacks	121	9	0	78	2	52	51	20	0	73	0	0
Succesful obfuscated attacks	121	9	0	94	1	53	47	20	0	74	0	0
Probe-block & succesful attack	0	0	0	0	0	0	6	20	0	0	0	0
Probe-block & succesful obfuscated attack	0	0	0	0	0	0	5	20	0	0	0	0
Probe-block & blocked attack	0	0	0	0	0	0	34	101	0	0	0	0
Probe-block & blocked obfuscated attack	0	0	0	0	0	0	28	101	0	0	0	0
Standard attack blocked but success with obfuscations	0	0	0	17	1	2	8	1	0	1	0	0
Fully passed test-sets	0	39	89	0	103	15	3	0	36	20	3	121

Table B.2: Results from Cert 2010_1 tests, taken 5th of July 2012

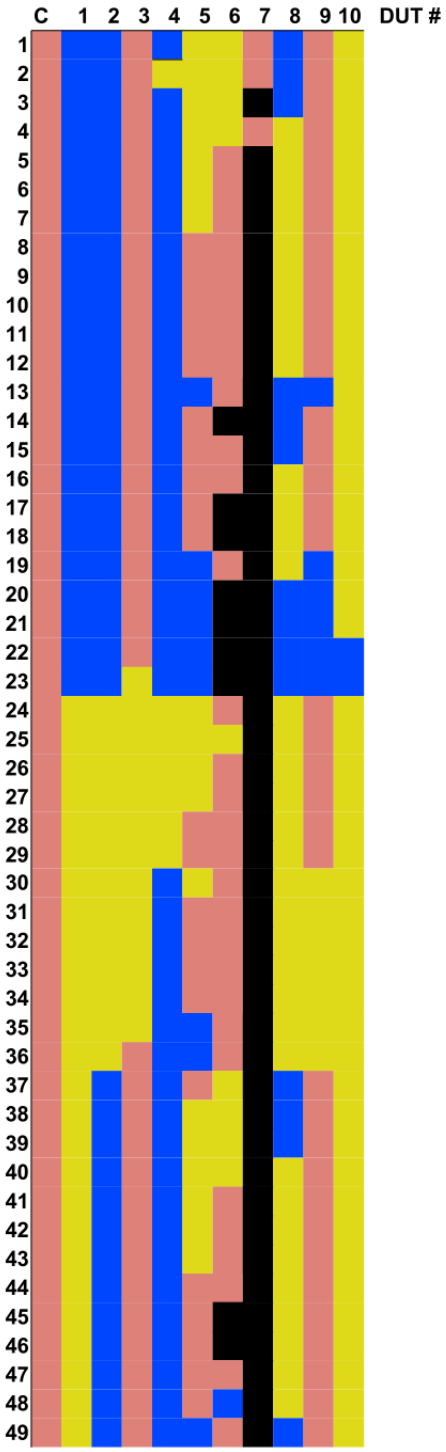

Figure B.2: Summary of test-results from cert2011_1 set, 1/3

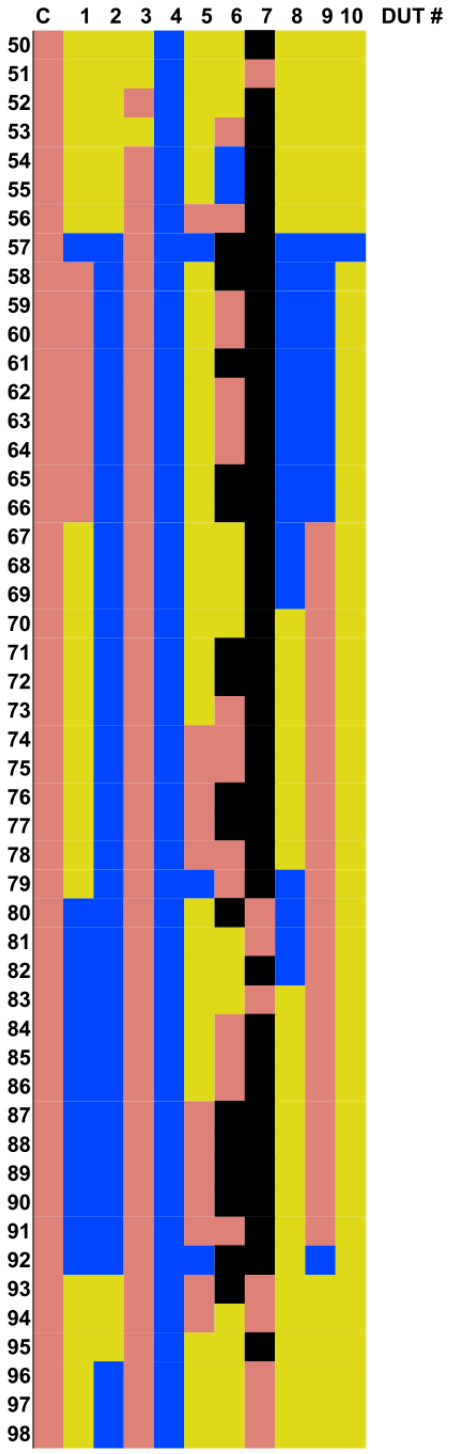

Figure B.3: Summary of test-results from cert2011_1 set, 2/3

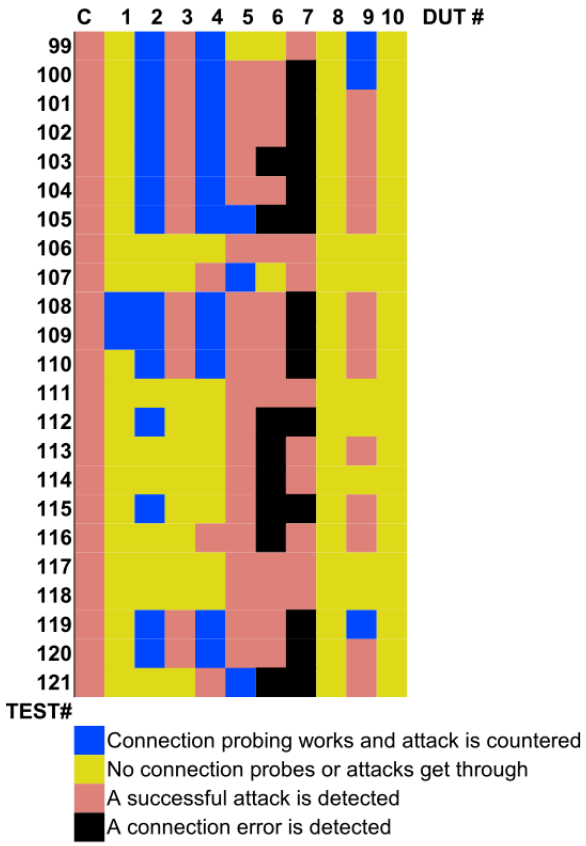

Figure B.4: Summary of test-results from cert2011_1 set, 3/3