572.1
Off the Disk and Onto the Wire

https://t.me/learningnets

Copyright © 2019, Lewes Technology Consulting, LLC and Mat Oldham. All rights reserved to Lewes Technology Consulting, LLC and Mat Oldham and/or SANS Institute.

PLEASE READ THE TERMS AND CONDITIONS OF THIS COURSEWARE LICENSE AGREEMENT ("CLA") CAREFULLY BEFORE USING ANY OF THE COURSEWARE ASSOCIATED WITH THE SANS COURSE. THIS IS A LEGAL AND ENFORCEABLE CONTRACT BETWEEN YOU (THE "USER") AND THE SANS INSTITUTE FOR THE COURSEWARE. YOU AGREE THAT THIS AGREEMENT IS ENFORCEABLE LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY YOU.

With the CLA, the SANS Institute hereby grants User a personal, non-exclusive license to use the Courseware subject to the terms of this agreement. Courseware includes all printed materials, including course books and lab workbooks, as well as any digital or other media, virtual machines, and/or data sets distributed by the SANS Institute to the User for use in the SANS class associated with the 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.

BY ACCEPTING THIS COURSEWARE, YOU AGREE TO BE BOUND BY THE TERMS OF THIS CLA. BY ACCEPTING THIS SOFTWARE, YOU AGREE THAT ANY BREACH OF THE TERMS OF THIS CLA MAY CAUSE IRREPARABLE HARM AND SIGNIFICANT INJURY TO THE SANS INSTITUTE, AND THAT THE SANS INSTITUTE MAY ENFORCE THESE PROVISIONS BY INJUNCTION (WITHOUT THE NECESSITY OF POSTING BOND), SPECIFIC PERFORMANCE, OR OTHER EQUITABLE RELIEF.

If you do not agree, you may return the Courseware to the SANS Institute for a full refund, if applicable.

User may not copy, reproduce, re-publish, distribute, display, modify or create derivative works based upon all or any portion of the Courseware, in any medium whether printed, electronic or otherwise, for any purpose, without the express prior 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.

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.

SANS acknowledges that any and all software and/or tools, graphics, images, tables, charts or graphs presented in this courseware are the sole property of their respective trademark/registered/copyright owners, including:

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.

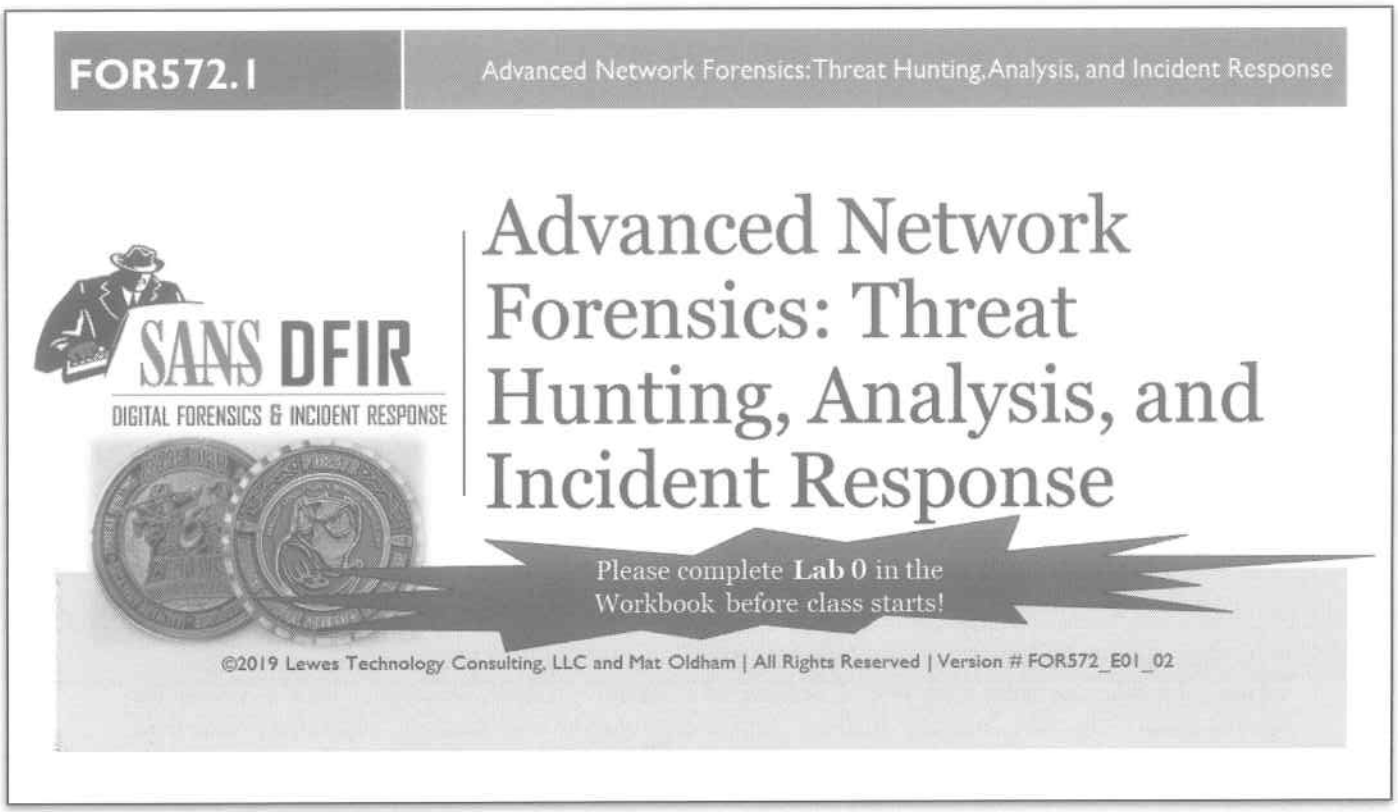
PMP and PMBOK are registered marks of PMI.

SOF-ELK® is a registered trademark of Lewes Technology Consulting, LLC. Used with permission.

SIFT® is a registered trademark of Harbingers, LLC. Used with permission.

Governing Law: This Agreement shall be governed by the laws of the State of Maryland, USA.

https://t.me/learningnets



Authors:
Phil Hagen, Lewes Technology Consulting, LLC
phil@lewestech.com | @philhagen
Mat Oldham
mat.oldham@gmail.com | @roujisecurity

Special thanks:
Rob Lee, George Bakos, Judy Novak, Mike Clark, Ben Knowles, Zoher Anis, Randy Marchany, Christian Prickaerts, Mike Pilkington, Ryan Johnson, Matt Bromiley, and David Szili

http://twitter.com/sansforensics

Welcome to SANS FOR572, Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response! After you get settled on Day 1, please complete Lab 0 in your separate workbook, which contains all exercise content. This lab will get your SIFT Workstation VM installed and operational, and load evidence to your laptop for later use. Doing this now will streamline the exercises during the rest of class.

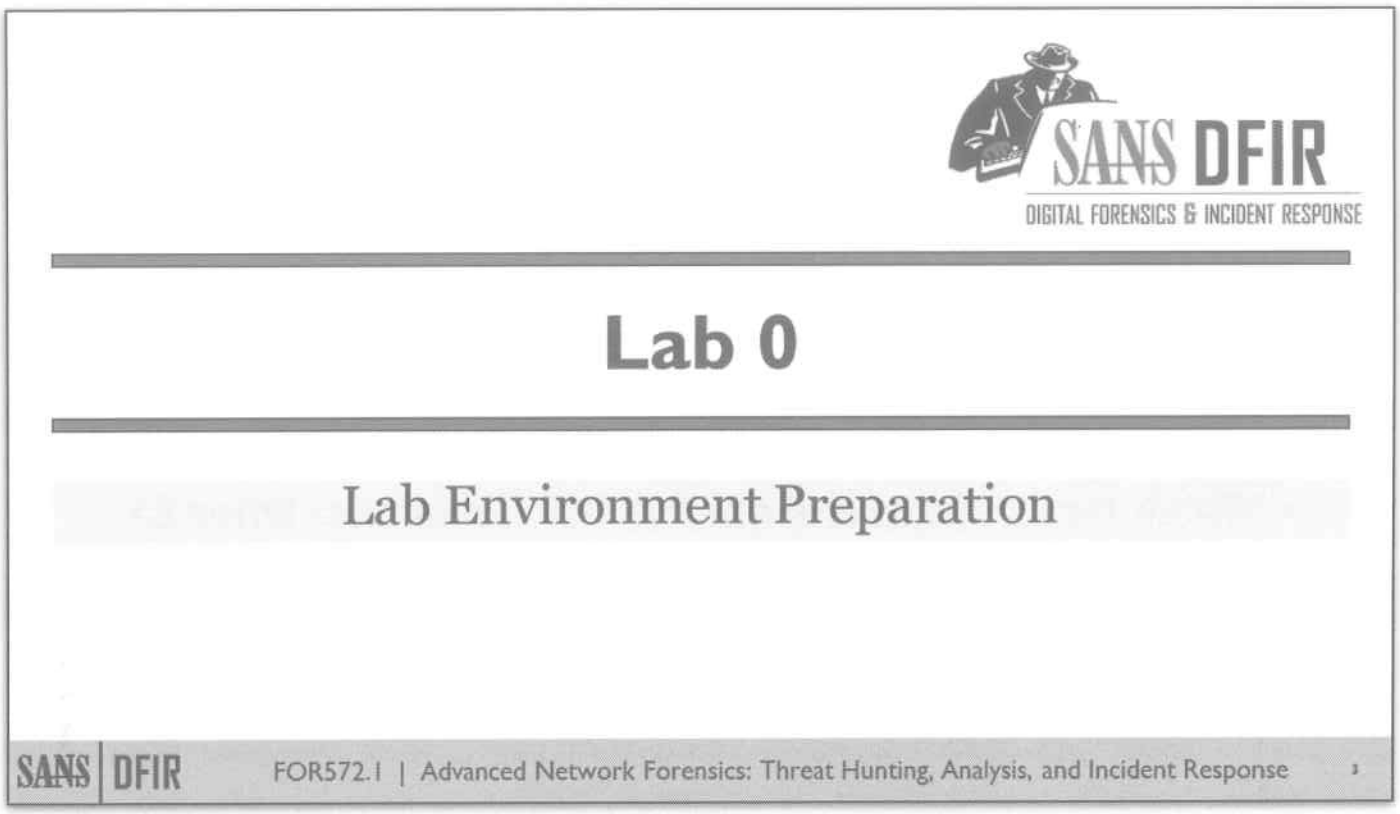
https://t.me/learningnets

This course is dedicated to my amazing wife, Marie, and my awesome kids, Genevieve and Burton. Thanks for believing in this crazy project, pushing me when it was needed, allowing me the time to keep the material ahead of the curve, and most of all for your ceaseless support while teaching. I could have never done it without you, and this is as much yours as anyone's.
To those I've had the privilege of working with in this industry, thank you for the opportunity to have learned and to continue to learn from your expertise. We're all in this game together and it's an honor to work by your side.
–Phil

First and foremost, to my loving wife, thank you for your patience and understanding. In this career field, protecting good guys from bad can be a long and thankless job at times, but knowing you're always there for me and supporting me makes this worth it. To my kids, know that your dad will have a lifetime of love for you and stories to tell when you are older. To all my colleagues in the cyber-security industry, commonly fighting behind the scenes and being asked to do more with less, keep fighting the good fight. The work you are doing is critical as the world becomes more and more dependent on technology, both large and small.
–Mat

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Lab 0 Takeaways: Lab Environment Preparation

- SIFT Workstation booted and operational
  - Additional VMware images if time allows
  - Command lines available in `~/for572-commands/`
- Syntax and environmental familiarity
  - **`Bold Courier New font designates user input`**
  - `Regular Courier New font designates output`
  - Watch for "`1`" (numeral one) vs. "`l`" (lowercase letter L)
  - Command-line continuation indicated with "⏎" character
  - Tips for high-res screens, international keyboards, Shared Folders/USB bridging, file decompression, SSH/SCP use

This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

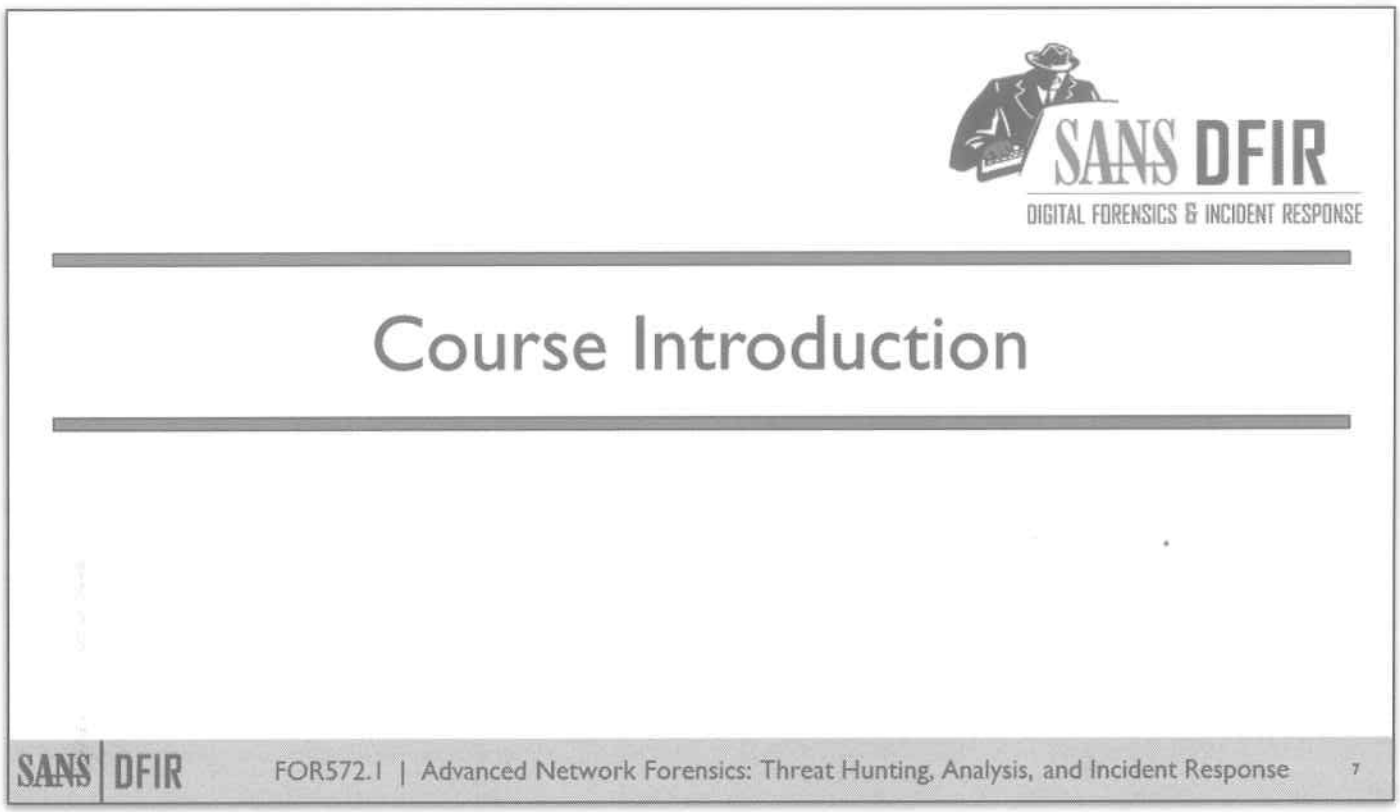
This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets

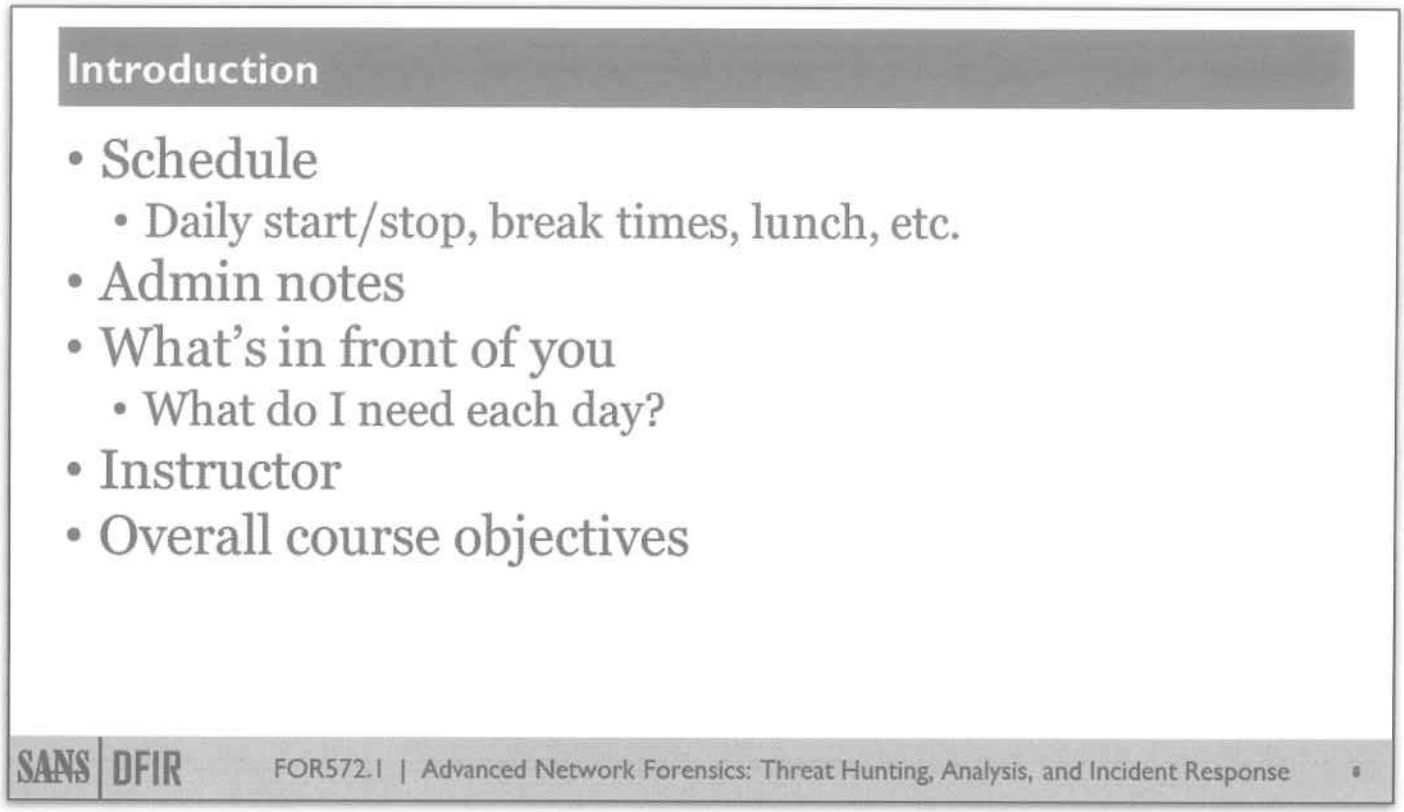
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Network activity can be represented in innumerable different ways. It can be heavily abstracted, such as in statistical aggregations, NetFlow, graphical, or host-based logs. At the other end of the spectrum is the bit-for-bit copy containing a perfect record of every packet that crossed the wire (or air). In between are dozens of other sources of network evidence, each of which can provide a unique and valuable perspective on activity that occurred on the network at a given point in time. Together, these perspectives and the evidence the "witnesses" provide can inform incident handling, host forensic activity, damage reporting, and improvements in security. Although network forensics by itself might still be unlikely to meet the burden of proof for most criminal investigations, it can be compelling in the justification of resource allocation or other administrative actions, and it can be a significant lead generator in identifying additional sources of traditional forensic evidence.

We'll step through a number of investigations in this class—twisting, turning, and massaging network data to arrive at ground truth. Along the way, you'll be learning and developing tools and techniques that you can take back to the workplace and put to use immediately, giving you a better understanding of what's happening on your home and employer's networks (assuming you're authorized to look).

Your instructor will review the course materials, including books, handouts, and USB drives with the VMware images and supporting evidence you will need for the course. Our recommended platform is VMware Workstation on Windows or Linux hosts and VMware Fusion on macOS hosts. While other virtualization software might work, our testing is performed against VMware software, so other options are not supported.

Bring your laptop to each class and the appropriate book. (Bring Section 2 on Day 2, etc.) If you find that you need some extra references or support with certain topics, you might want to consider also bringing that section's book with you until you're more comfortable with the material.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

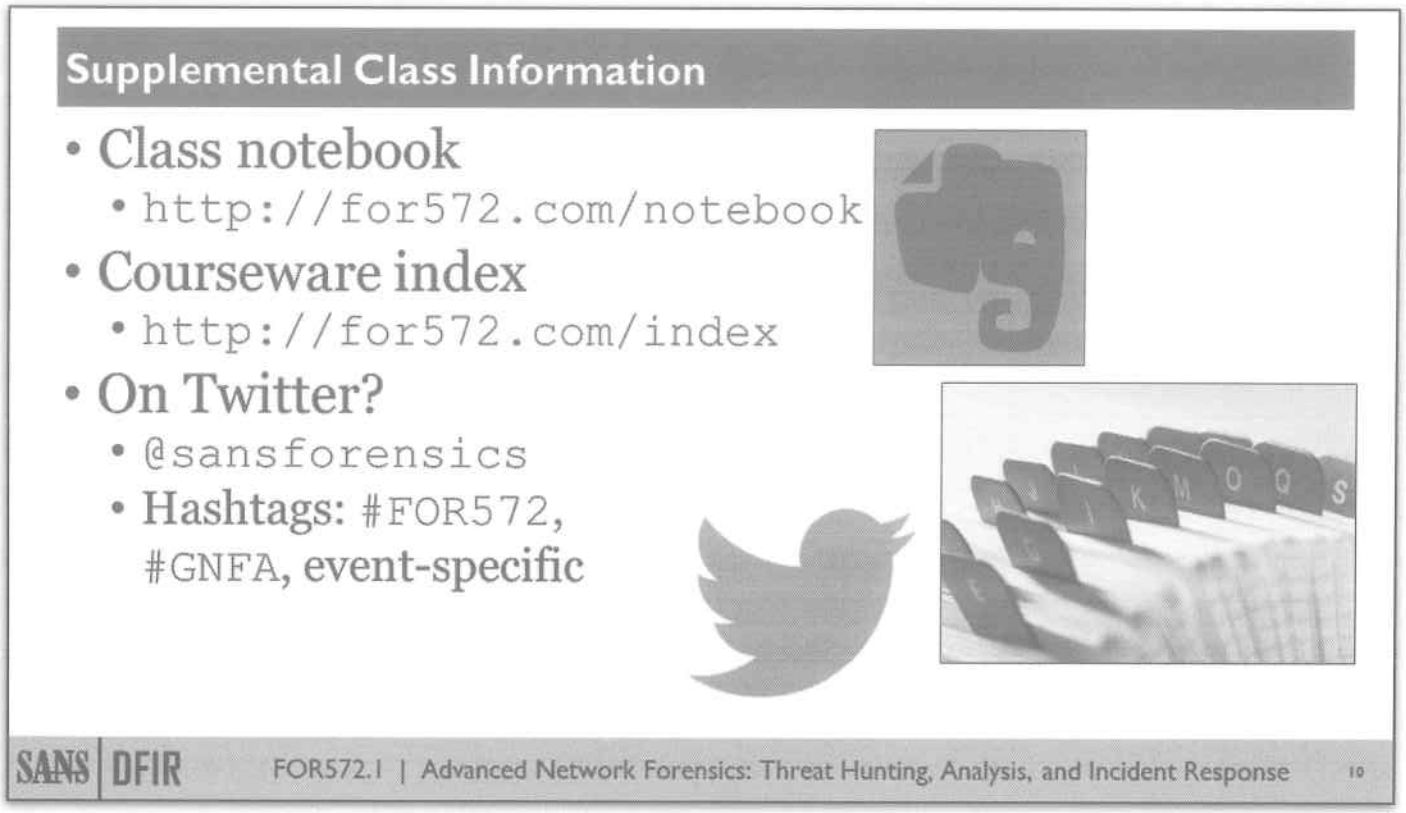
Your Instructor

- At this time, your instructor will ...
  - Introduce himself or herself
  - Briefly describe his or her background and interest in network forensics

This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



We've collected a large (and growing) number of resources that are either too small or too new for inclusion in the course. These are available at the URL `http://for572.com/notebook`, which is an open read-only Evernote notebook. You can access and search this via a web browser (no Evernote account required), or add it to your own Evernote account for continuously-synchronized updates.

Additionally, we have generated an electronic index of material from the courseware books, which you can download and use as a baseline for your GNFA preparation or just for your reference to the material. We do not provide this in the books because we use a community-driven word list (formally known as a concordance) that undergoes frequent revision. You can download the latest copy for your version of the material at `http://for572.com/index`, as well as learn about how you can contribute to the concordance for this and other SANS DFIR courses.

If you use Twitter, it's a great way to stay in touch with students at conferences, or even those who have taken a given class at some point in the past. You'll likely find some good discussions about the course overall with the "`#FOR572`" hashtag, and more specifically about the GIAC Network Forensic Analyst (GNFA) certification with "`#GNFA`". At SANS conference events there are also event-specific hashtags that make it easy to interact with other attendees. Of course, the `@sansforensics` account is a good one to follow as well.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

COURSE AGENDA	SECTION
Off the Disk and Onto the Wire	1
Core Protocols and Log Aggregation/Analysis	2
NetFlow and File Access Protocols	3
Commercial Tools, Wireless, and Full-Packet Hunting	4
Encryption, Protocol Reversing, OPSEC, and Intel	5
Network Forensics Capstone Challenge	6

This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Why Network Forensics?

- Hard drive may not be available
- Evidence may already be collected
- Can't practically investigate 1k, 10k, or 100k+ machines
- Hunt for the attacker you don't yet know exists

SANS DFIR FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response 12

Those familiar with computer forensics know that artifacts are left by even the most careful suspects. Deleting data doesn't often truly delete anything and wiping evidence leaves more behind. Even cryptographic protections can be defeated if the system performing the encryption isn't diligently sanitized, including RAM and pagefiles. Unfortunately, the system used or touched by a malicious actor isn't always available, or perhaps for OPSEC reasons, we cannot access the system for fear of tipping the attacker off to the existence of an investigation.

When a machine is attacked, or otherwise accessed via a network—whether or not the attempt was successful—there are traces left behind that can help us reconstruct the attacker's intent and actions. Even if their tactics, techniques, and procedures (TTPs) were ineffective, we might still be able to generate valuable intelligence to help understand their motives, capabilities, and likely next moves.

What if the attack was successful? What happened next? Often, in the case of advanced attacks such as industrial espionage or foreign threats such as Advanced Persistent Threat (APT) actors, the initial intrusion is just the beginning of a long period of activity that includes a deep, pervasive presence on the network. If you identify that the attacker used domain credentials in an attempt to move laterally within a network environment with tens of thousands of hosts, what do you do next? What if you know the systems in that environment are not configured to generate sufficient logging data to learn the attacker's actions? Dust off your resume?

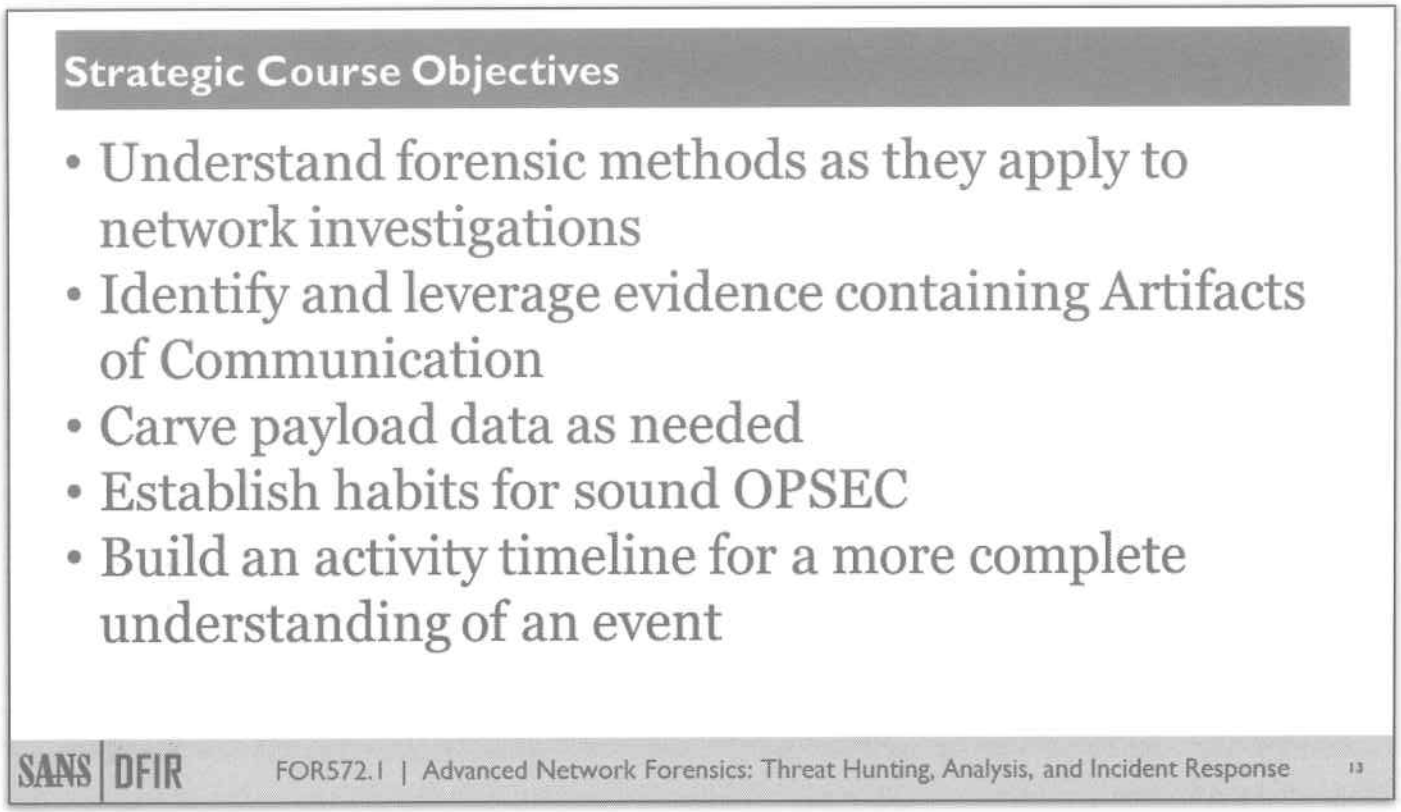
No! Each network transaction potentially touches dozens of devices and observation points—NetFlow, packet capture, IDSes, or proxies all may be in use, and log data from switches, routers, firewalls, and other infrastructure components could have been generating valuable log data all along. With the proper training, tools, and techniques, that impossible containment challenge may be more manageable than you think.

Often one of the biggest challenges facing a network investigator is in emplacing and managing the systems to continually collect data before you realize you need it. Unlike a hard drive, the wire alone remembers nothing. Once the data is collected and available, though, it's a treasure trove! Even without a long-term packet capture or NetFlow data collection, a skilled investigator will consider the entire scope of available evidence to fill as many voids in their hypotheses as possible.

Image Credit: (CC) pyroclastichawk on Flickr

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



When examining an advanced attacker's activities, the investigator can be easily misled by the deliberate injection of false evidence or through the utilization of covert communication mechanisms. Similarly, when investigating an unsophisticated suspect's actions, their inexperience may cause unexpected network behavior. Any of these scenarios could easily confuse the investigator who wrongly assumes all evidence must fit in a neat, tidy box. Occam's Razor does not always apply.

- Forensicators should be disinterested, impartial, and objective when gathering data.
- When an experiment or an observation gives a result contrary to the prediction of a certain theory, all ethical forensicators must abandon that theory.
- Forensicators must never dogmatically believe in or exclude an idea nor use rhetorical exaggeration in promoting an idea or ruling it out.

– Paraphrased from "Conduct, Misconduct and the Structure of Science",
American Scientist. *Sep/Oct 1996*

As you will learn in this course, there are many different customers that a network forensic investigation can serve. Each potential customer has their own mission and priorities, but they all share one thing in common—the need to understand what happened. As forensicators, we must stay disciplined in our processes, and diligently pull threads even when existing tools and techniques are insufficient to analyze them. Through this, we can be successful in our job, which is to provide the customer with the understanding they need, based on data that will withstand challenges and that does not rely on conjecture or unfounded opinions.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response

Off the Disk and Onto the Wire

©2019 Lewes Technology Consulting, LLC and Mat Oldham | All Rights Reserved | Version # FOR572_E01_02

Authors:
Phil Hagen, Lewes Technology Consulting, LLC
phil@lewestech.com | @philhagen
Mat Oldham
mat.oldham@gmail.com | @roujisecurity

https://t.me/learningnets

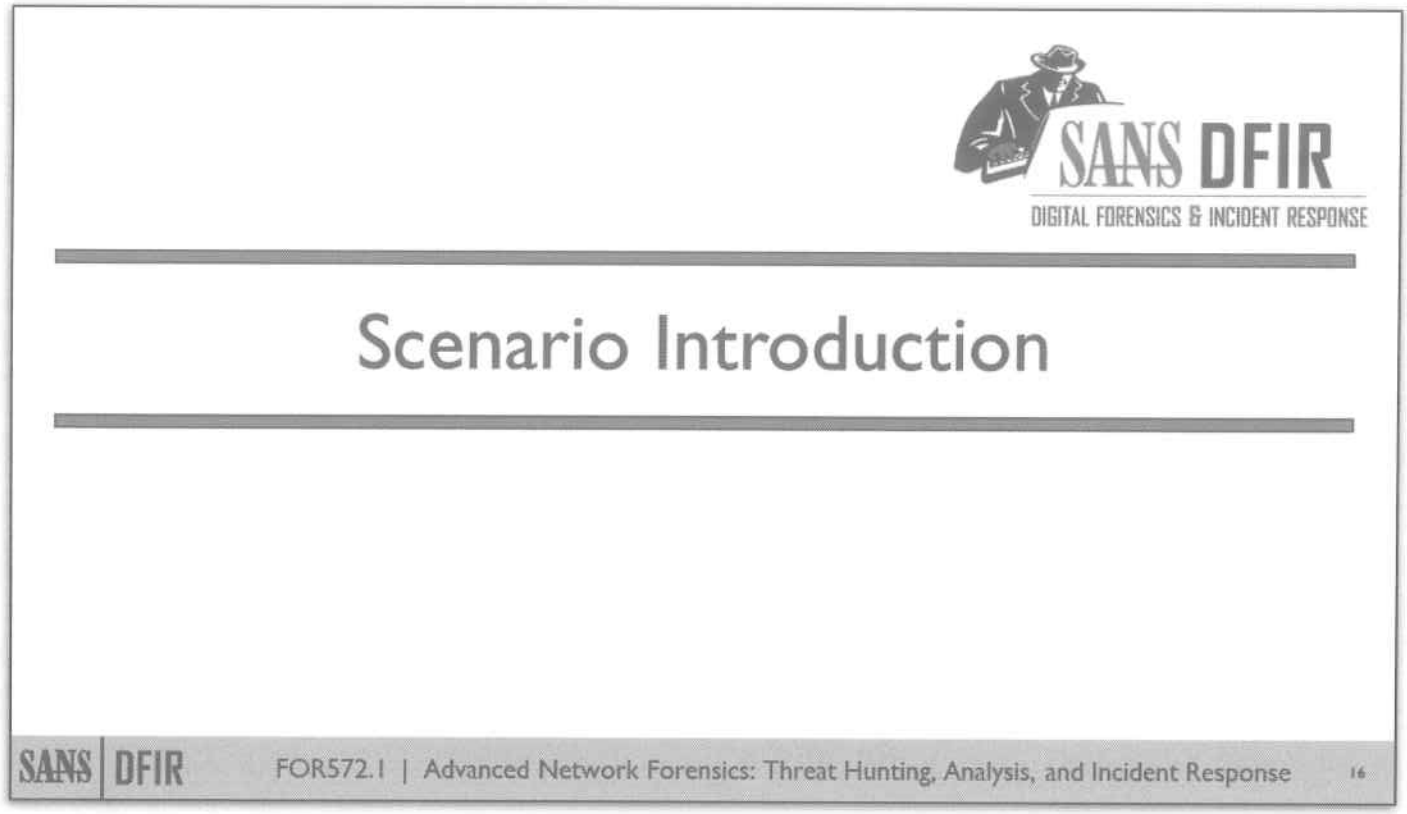
© 2019 Lewes Technology Consulting, LLC and Mat Oldham

TABLE OF CONTENTS	PAGE

This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

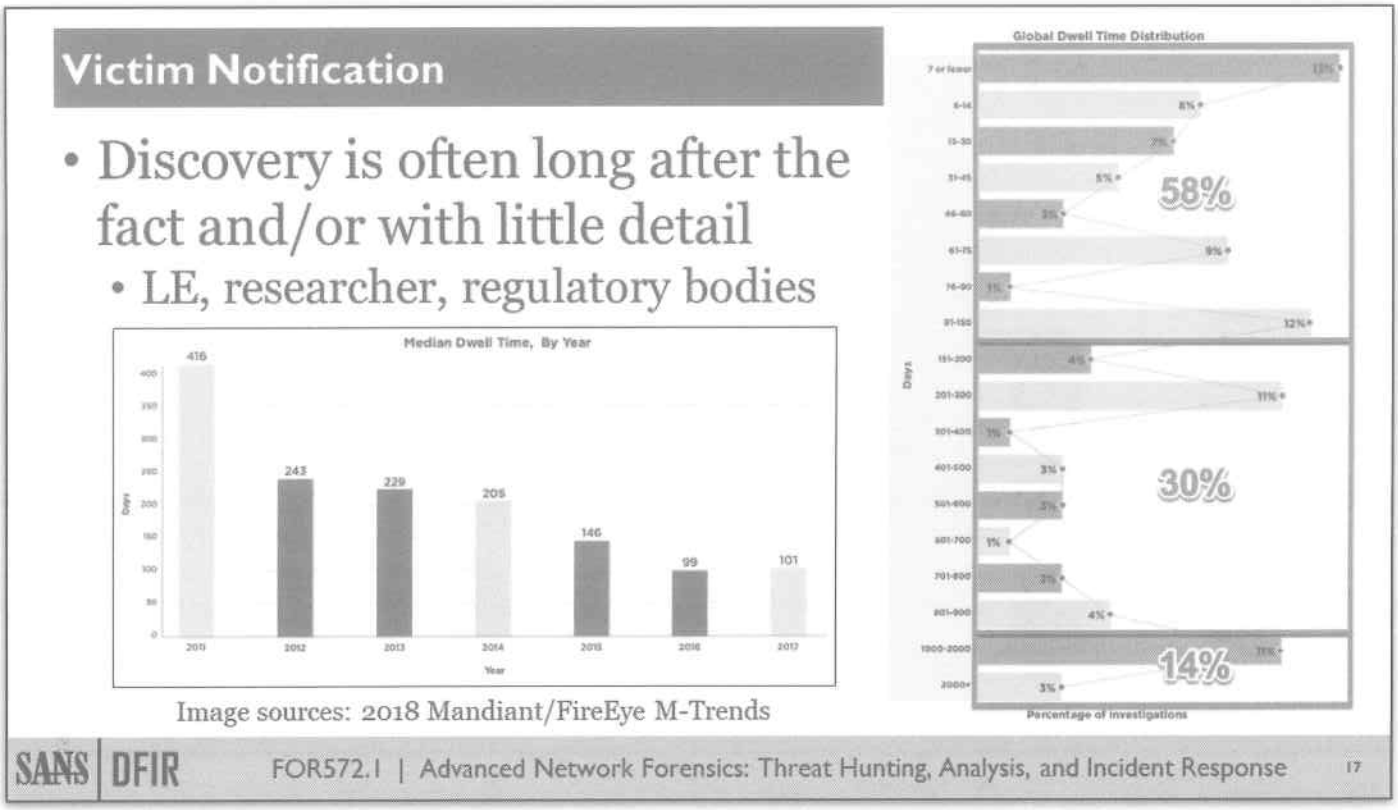


Image sources: 2018 Mandiant/FireEye M-Trends

To set the stage for the course scenarios, let's start off with a realistic notification that will drive our investigations.

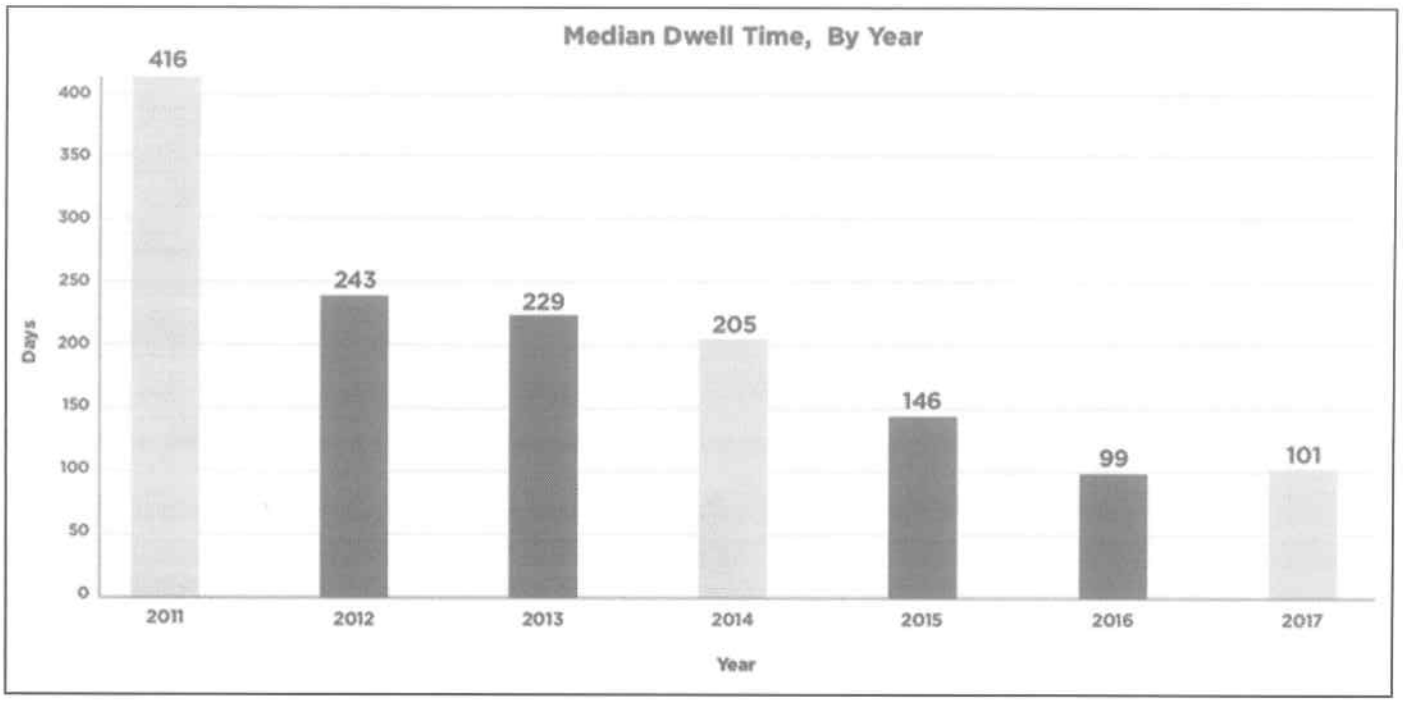
The cold, hard reality is that 38% of victims first learn that their network has been compromised after a call from an external party. This has been a marked improvement over the years. You may not have a great amount of detail about an incident, but the potential damage could be significant enough to justify a full investigation. Further complicating matters, the 2018 Mandiant/FireEye M-Trends report indicates that the median time between a breach and the detection of that breach is still 101 days[1]—long after the most critical evidence has been overwritten or otherwise unavailable.

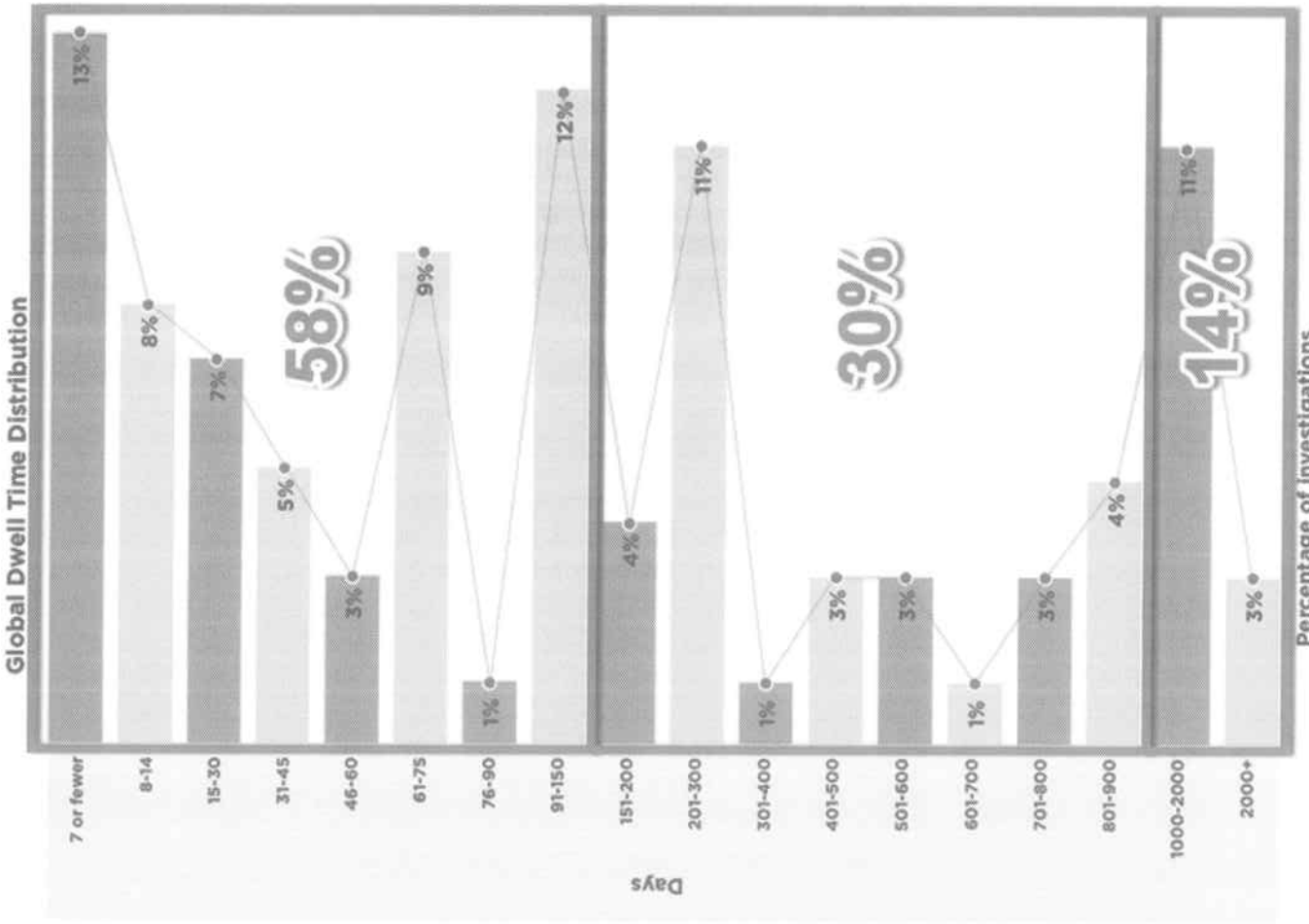
References:

[1] `http://for572.com/5fopa`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham





https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Your Network for the Week: SRL

- Government-sponsored R&D laboratory
  - Metal alloy, bioengineering specialties
  - Currently seeking formula for "Carbonadium" alloy
- Mail server anomalies caused initial tipoff
  - Enterprise-wide host forensics underway
- Russian APT HAMMER is active threat
  - Target metallurgy organizations
  - Operate aggressively and freely within
  - Use both commodity and custom tooling
- This scenario also used in FOR508!

SANS | DFIR FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response 19

SRL is a government-sponsored laboratory that researches specialized metal alloys and bioengineering capabilities. Lately, SRL has been tasked to find the secret formula for an alloy called Carbonadium.

The Metallurgy Intelligence Sharing and Analysis Center (ISAC) recently noted a significant uptick in Russian APT activity targeting their sector. In particular, threat intelligence specifically suggests that a Russian group dubbed "APT HAMMER" has been probing SRL and its competitors in the industry.

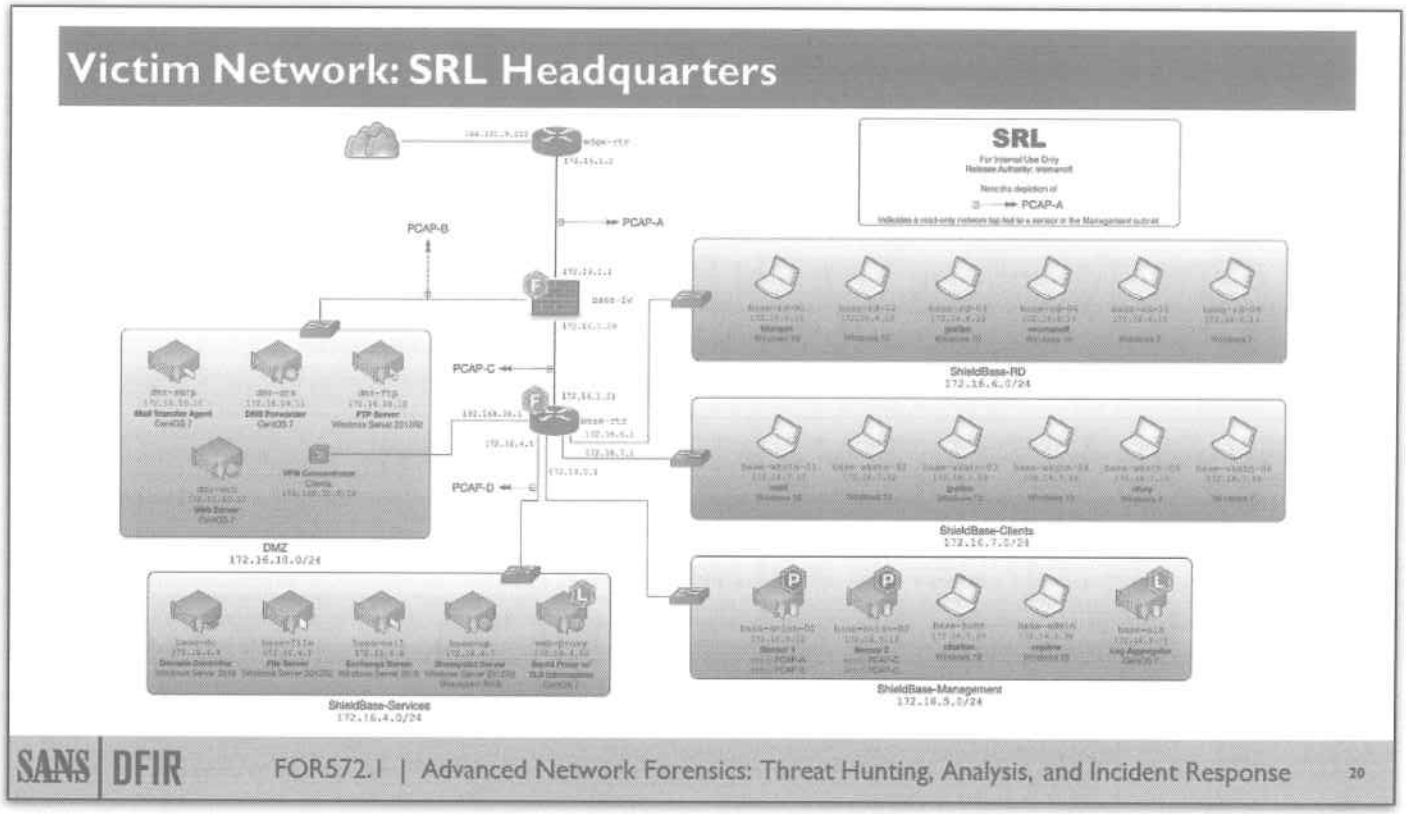
On 5 September 2018, some intermittent connectivity issues with the mail server, which followed similar odd behavior on the Internet-facing web server, caused IT and security staff at SRL to research the root causes of the activity. After cursory research, malicious activity was suspected and more comprehensive incident response actions were initiated. Much of this activity focused on the host systems, but network data has been collected to support the team's investigation.

A more extensive introduction to the Stark environment is in your Lab Workbook. A brief overview of both the main SRL Headquarters and SRL Outpost contingency operating location are in the following pages.

This scenario is not only woven through the FOR572 course material, but is also shared with FOR508, Advanced Digital Forensics, Incident Response, and Threat Hunting. While we will be relying solely on the network-based source data, FOR508 uses the disk and memory images from the environment.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



The SRL Headquarters network is representative of many medium-to-large enterprises. Note that there are several dozen systems in operation that are not depicted on the diagram. The network has several internal segments interconnected with a core router, which has an upstream connection to the firewall. The firewall brokers traffic between the internal segments (via the core router), the DMZ, and the Internet router.

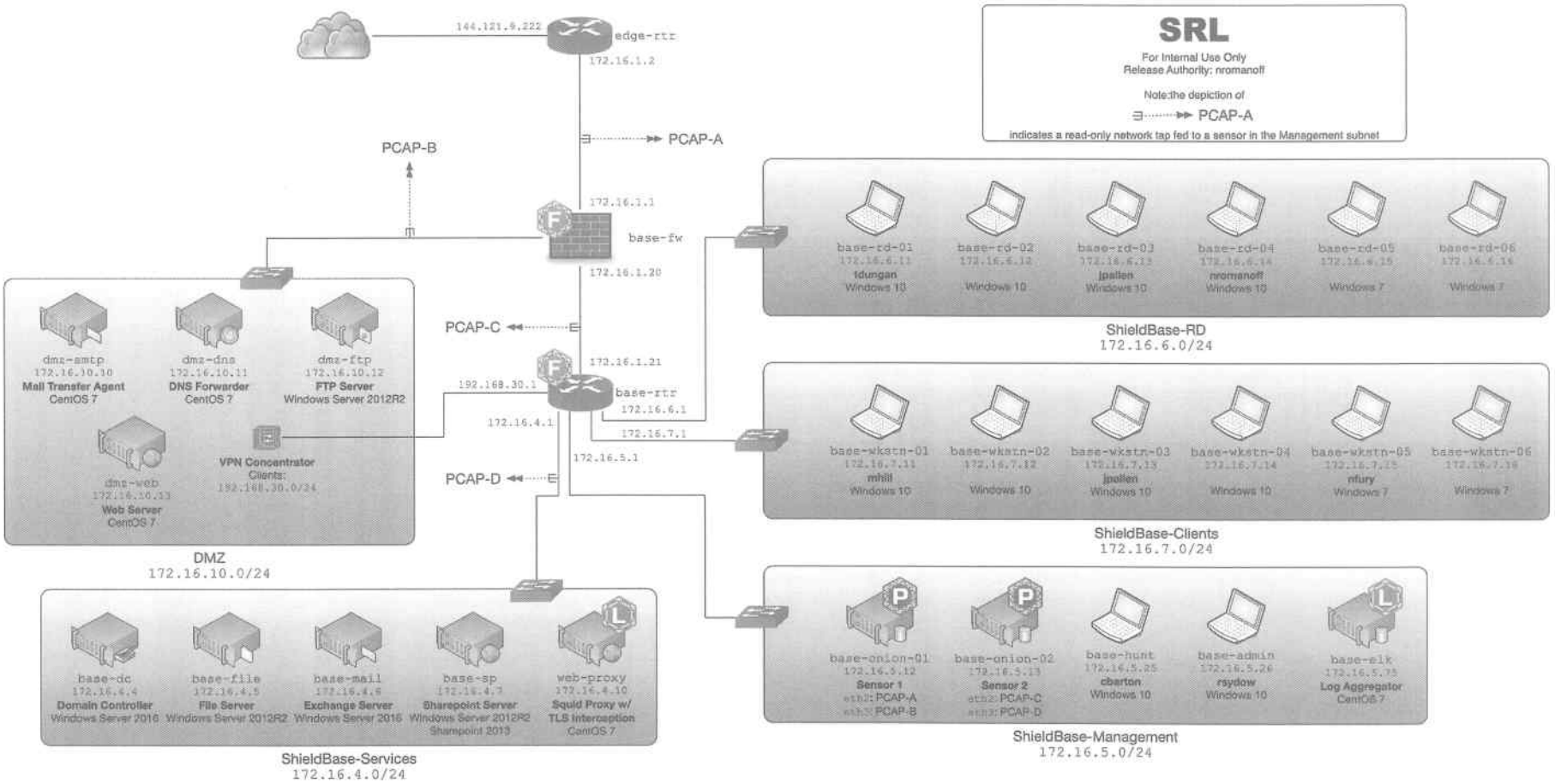
There are four taps in place, as indicated in the diagram. The full-content data from these taps is sent to two different sensors/collectors, which reside in the Management subnet. NetFlow is exported from the internal router and the firewall, each of which provide visibility from all of their interfaces. Logs are aggregated from across the environment, from a variety of source systems.

Key infrastructure is annotated on the network diagram. Key client systems include:

Username	Role at SRL	Hostname	Host IP	Host OS
mhill	CEO	base-wkstn-01	172.16.7.11	Win10
nfury	Sr. Manager	base-wkstn-05	172.16.7.15	Win7
tdungan	Sr. Researcher	base-rd-01	172.16.6.11	Win10
nromanoff	Program Manager	base-rd-04	172.16.6.14	Win10
rsydow	IT Admin User	base-admin	172.16.5.26	Win10
rsydow-a	Domain Admin Acct			
cbarton	IT Security User	base-hunt	172.16.5.25	Win10
cbarton-a	Security Admin Acct			

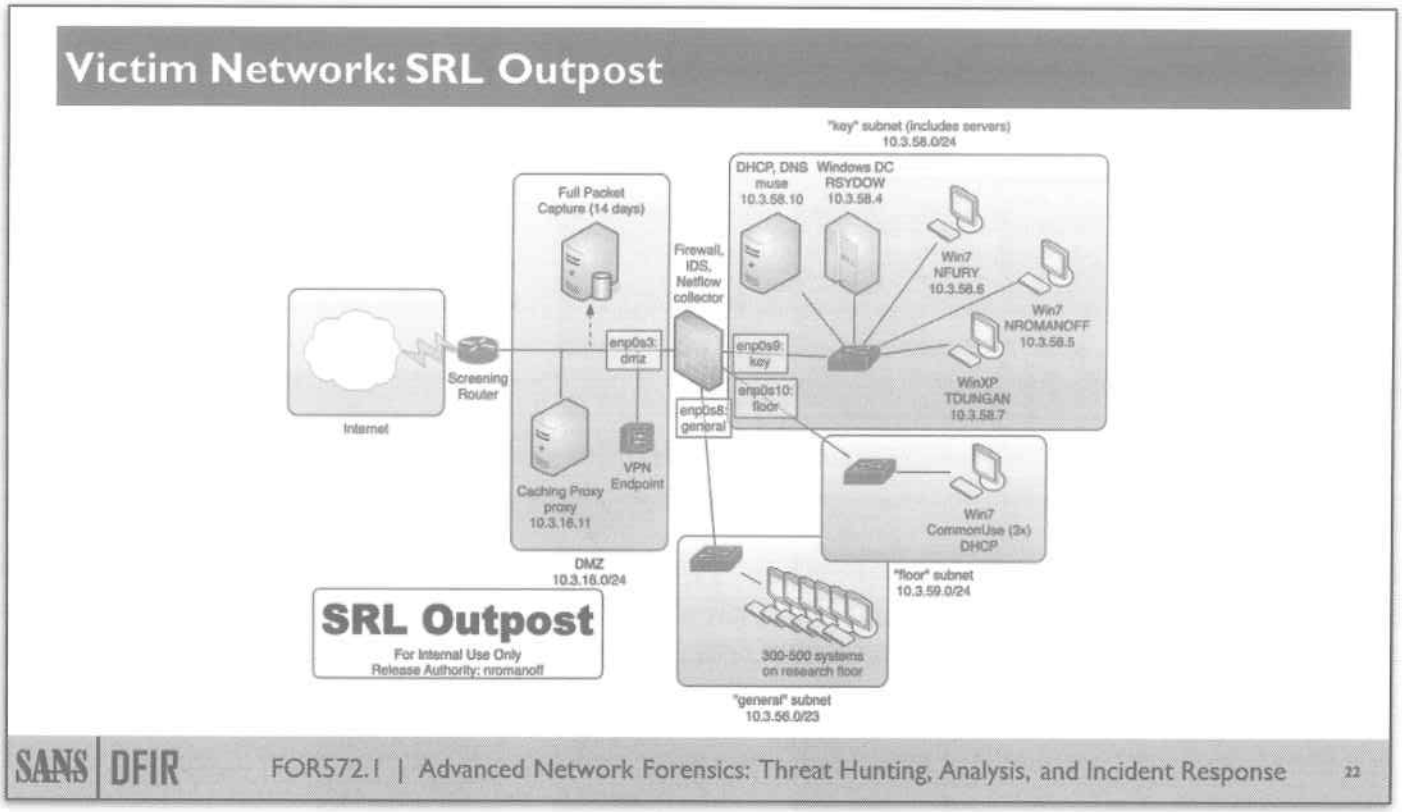
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



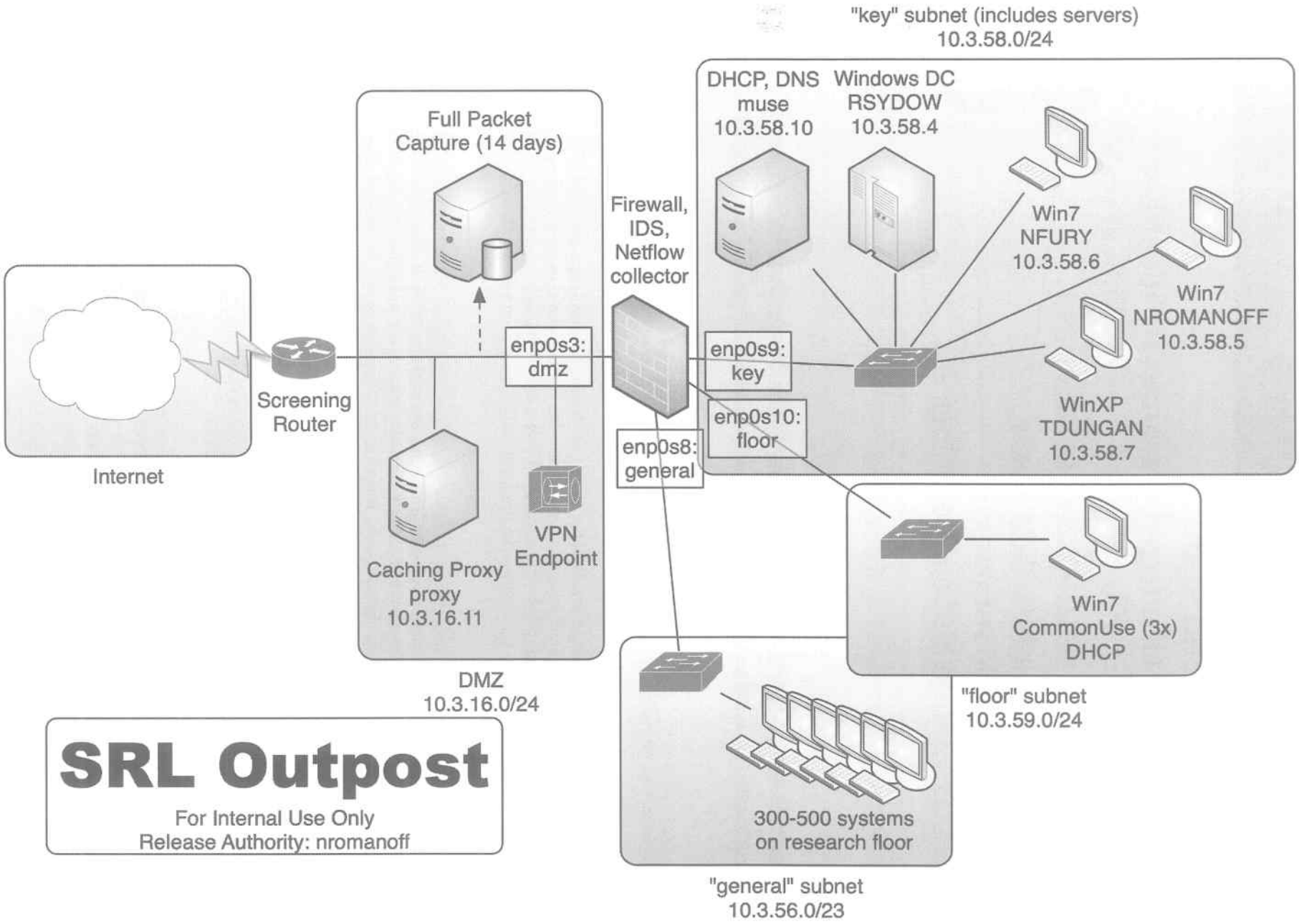
The SRL Outpost network is representative of many hastily-built, small-site networks, albeit with at least some of the critical security controls in place. This location is considered a contingency operating location to be used in case of emergency at the main headquarters facility. There are several internal segments separated by a single firewall, but very little network protection between hosts once on the inside of that perimeter.

Key systems we are aware of include:

Host	IP Address	Role
Router	10.3.16.1	Inside interface of Internet screening router
Firewall	10.3.56.1/23 10.3.58.1/24 10.3.59.1/24 10.3.16.99/24	General administration and users Key users and SHIELDBASE servers subnet Production floor systems External (DMZ) interface
VPN	10.3.16.5/24	VPN Concentrator external interface
Proxy	10.3.16.11/24	Caching proxy for outgoing HTTP
Sniffer	N/A (via tap)	`tcpdump` rotating buffer. Approx. 14 days of storage
Web Server	10.3.16.3/24	External presence of SRL Outpost
rsydow	10.3.58.4	SHIELDBASE Domain Controller—Windows 2008r2
tdungan	10.3.58.7	R&D workstation
nromanoff	10.3.58.5	Workstation
nfury	10.3.58.6	Workstation

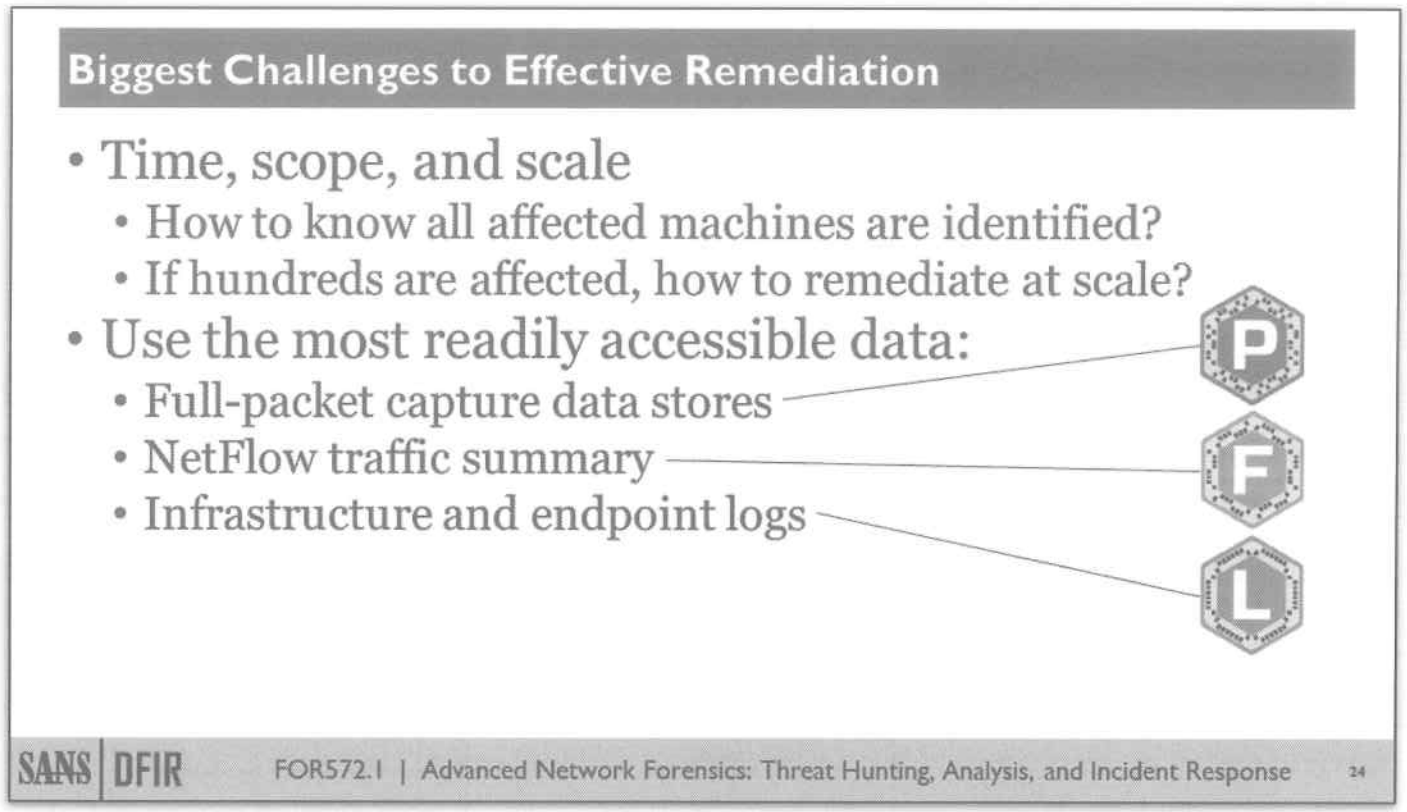
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



The incident response team may be ready to spring into action, but there are some challenges to their effectiveness. The vague impetus for the investigation – that there were outages on some servers – is not a very specific lead to follow by itself.

In many organizations, IR teams experience that the biggest challenges are time, scope, and scale. While we are almost always under a time constraint to accomplish our tasks, this is compounded by the fundamental necessity of accuracy – sometimes to a pedantic level. The practice of DFIR is not one that allows guesses or unfounded claims, after all. Creating a reliable work product requires discipline and knowledge on how to balance expectations against deadline requirements.

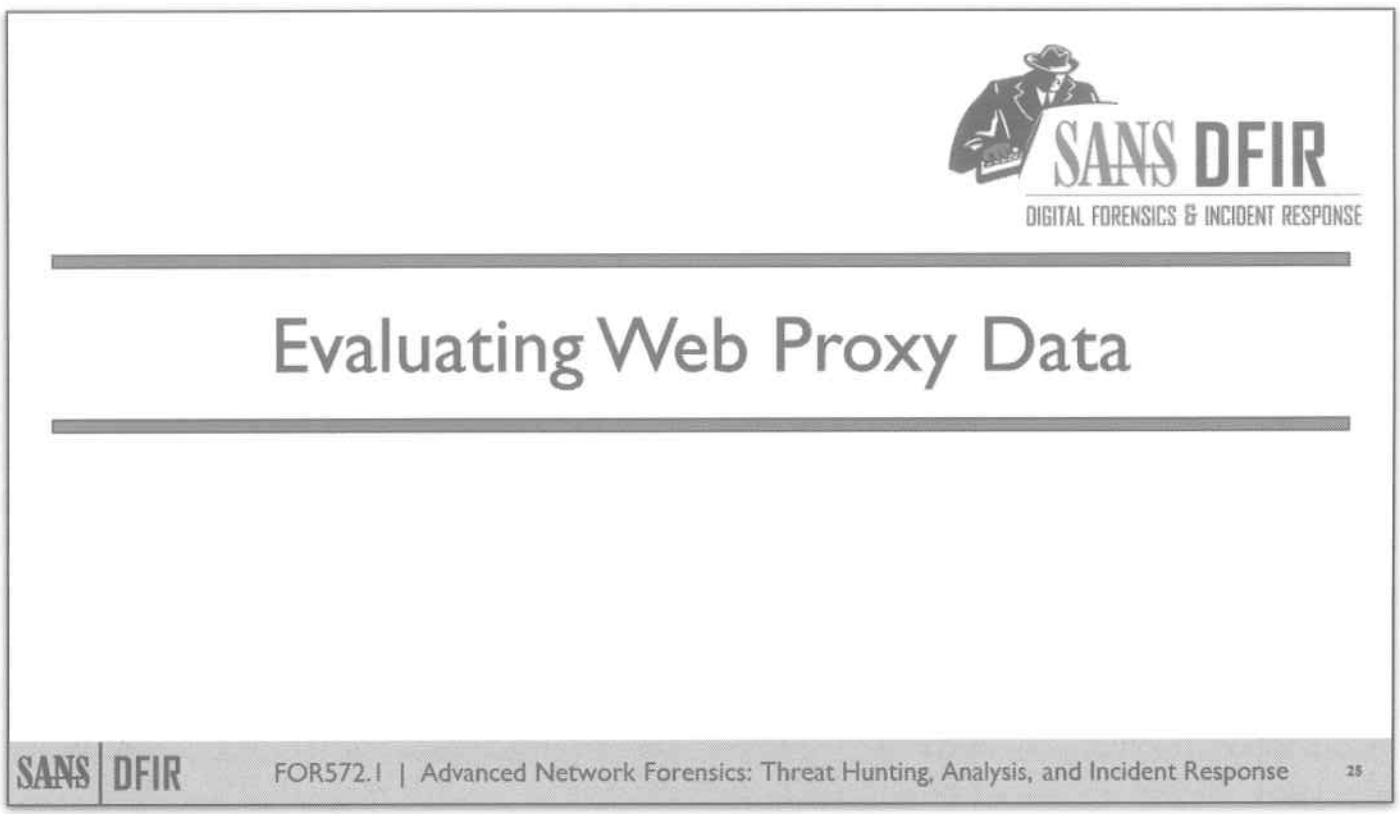
However, the sheer size of many environments presents a significant challenge as well. Scoping a compromise or other incident to just the 10-15 systems involved is critical, but unmistakably difficult when the environment ranges into the hundreds of thousands of systems and users. Even with perfect environmental knowledge and reliable tooling, complete scoping could take days or weeks, if not longer. Complicating this factor is that many tools simply cannot operate at that large a scale, requiring the IR team to segment their operations to a manageable size, which linearly increases the complexity of IR operations.

Fortunately for us, the data commonly provided by the network can be a significant benefit to IR tasks. In this class, we will use full packet capture, NetFlow and other flow-based traffic summaries, and log evidence from infrastructure devices like proxy servers as well as from endpoints to help alleviate the pressures that decreased time and increasing scope and scale introduce into our workflow.

As you will see, starting with what evidence we have immediately available can contribute to a more efficient, effective, and ultimate successful IR team. This will depend on how well the source data can be integrated into the investigative process. As with many facets of the forensicator's work, the best skill is knowing how to adapt while maintaining proper adherence to established process and policy.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



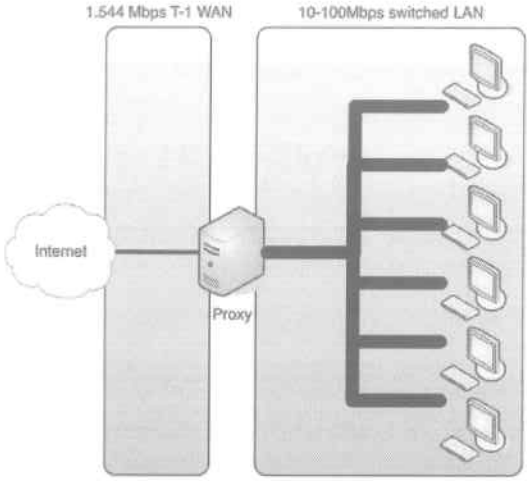
This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Web Proxy Servers

- Traditionally used for performance reasons
- Now serve proactive and post-incident purposes
  - Prevent known "bad" things
  - Log data: Access list for all HTTP(S) traffic
  - Cache data: Copies of objects sent via HTTP(S)
- SSL/TLS can hinder but interception is becoming more common



SANS DFIR FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response 26

A proxy server, as the name implies, is a server that is configured to broker network traffic between a client system and a server system. Although proxies can be used with nearly any protocol or network service, today we most frequently identify them in association with web traffic that uses the HTTP and HTTPS protocols.

Originally, proxies were developed to conserve precious bandwidth on lower-speed network links. For example: A proxy server might have been deployed in front of several hundred clients sharing a T-1. In this case, the proxy would receive the clients' requests for web content from systems residing outside the network, then request that content from the server responsible for that URL and provide it to the requesting client system. However, the proxy would keep a copy of that content as well. Then, if another client were to request the same page soon thereafter, the proxy would simply provide the cached content without re-requesting the same web page over the T-1.

Proxy servers can also function as "reverse proxies." In this model, proxy servers generally broker requests from a large number of client systems to a small number of servers. Often, a reverse proxy will provide load-balancing, compression, and other performance-enhancing functionality.

The reason we spend time this early in the course covering proxy servers is that they provide a very useful view of users' and systems' network activities, with regard to certain protocols—namely HTTP. By demonstrating the use of a proxy in a typical investigative scenario, we cover a number of network forensic and analysis topics and disciplines.

As with many infrastructure devices, the security movement also identified value in the data a proxy server often holds. Network administrators can configure proxy servers to block undesirable content, preventing their client systems from accessing prohibited subject matter. In addition, the "gatekeeper" nature of proxy servers provides two vital resources for information security professionals: Content transaction logs and the cached data itself.

The logs created by a web proxy server are invaluable in determining which URLs were requested by clients. This can (quickly!) answer the question, "Which inside systems attempted to access a known malicious site or download?" Whether investigating a phishing incident or a botnet using HTTP for command and control, proxy

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

logs can establish the scope of the incident much faster than system-by-system analysis could ever do. Typical proxy logs not only include elements like the time, requestor's IP address, and URL, but also the result status of the request, and sometimes the username that made it.

In addition, a caching proxy server's very purpose is to keep copies of resources retrieved by client systems, meaning that security teams can retrieve those cached objects for further analysis without ever laying a finger on the keyboard of an infected client system. If you've ever had to examine a compromised system that no longer contains a piece of downloaded malware, you're sure to appreciate the beauty of this option!

The proactive nature of a proxy server can also help during the incident response process. An administrator can reconfigure a properly implemented network and proxy server to block malicious HTTP traffic. This can be accomplished nearly instantaneously from one central point for an entire enterprise. In a situation where speed and decisiveness are critical, a proxy server brings immeasurable value to the incident response process.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

There are a variety of proxy solutions available today, in both FOSS and commercial forms.

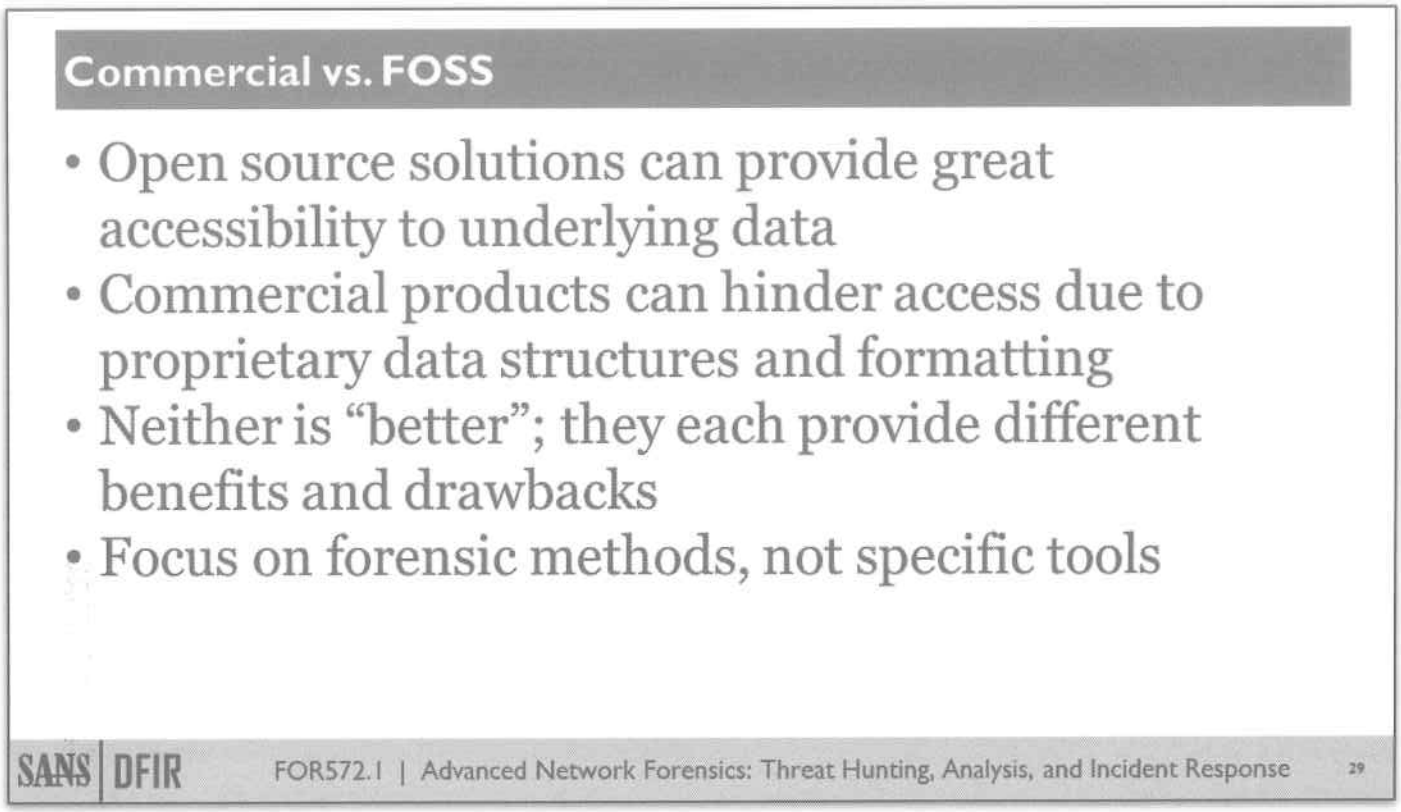
Perhaps the most commonly recognized is the Squid proxy, which was originally developed in part under a grant from the National Science Foundation. Today, it runs on a wide variety of operating system platforms and is an attractive option due to its cost (free), and versatility.

NGINX (pronounced "Engine-X") traditionally operates as web server software but can be configured as a web proxy server. (Most frequently, web server software is deployed in a reverse proxy configuration.) Apache Traffic Server is also a viable solution, donated to the Apache Foundation by Yahoo! in 2009.

Commercial web proxy solutions are available in both hardware and software forms. The Symantec/Blue Coat proxy is an appliance used widely in corporate enterprise networks. It includes the built-in ability to perform SSL/TLS proxying, as discussed in the previous slide. Forcepoint Web Security is also used in large-scale environments, and Barracuda Networks also offers web caching and filtering capabilities (among other security-focused features) in an appliance form factor.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



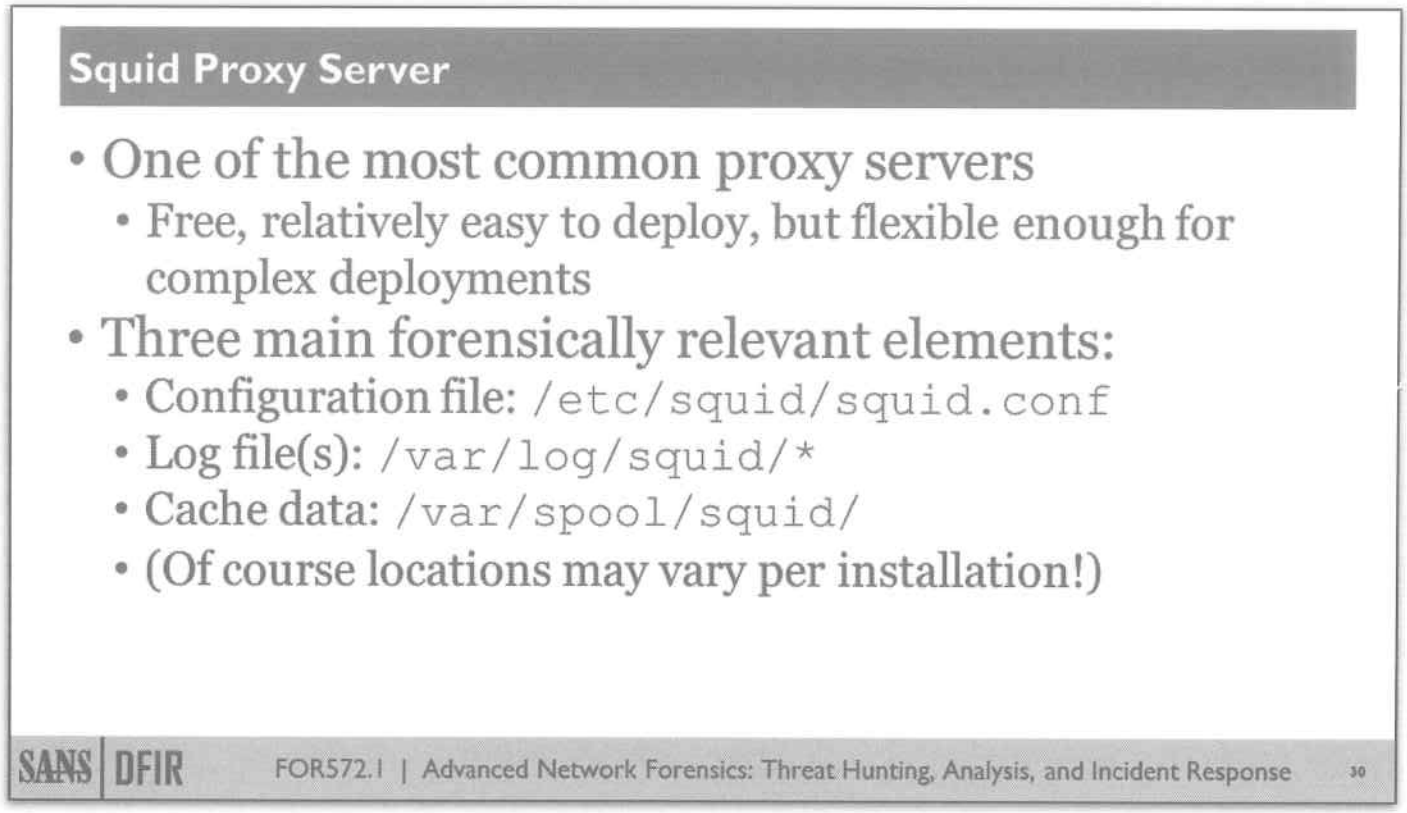
There is no one clear winner in the eternal "commercial vs. open source" battle, and proxy solutions are no exception. In courses like this, we tend toward open source solutions because they are better suited to allow in-depth, under-the-hood review. If we don't know why or how the code performs a certain function, the code is available to complete our understanding.

However, commercial solutions are quite commonly seen in large enterprises, and can often perform their designated tasks quite well. In terms of commercial proxy servers, we tend to see that their data storage formats are proprietary, and require some tinkering before relevant data items can be extracted for analysis. Another common shortcoming for commercial products is that the interface they provide to administrators and/or users tends to be lacking in power and flexibility. They are designed to allow access to a small handful of functions, but straying from that narrow lane can be quite difficult.

Predictably, in this course we will focus most heavily on the Squid proxy server. Squid is open source, widely used, and provides a very good foundation on which we can build some fundamental analytic skills. Although we address Squid-specific options and behaviors, any major commercial proxy server provides the same or similar functionality. As always, this course does not teach a tool, but the underlying methodologies you would need to analyze proxy servers in general.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



As previously mentioned, the Squid proxy server is prevalent. It is frequently used in large and small networks alike due to its cost (free) and relative ease of deployment (most Linux distributions include out-of-the-box functional installations). However, the initial simplicity hides the incredible power of a complex network service.

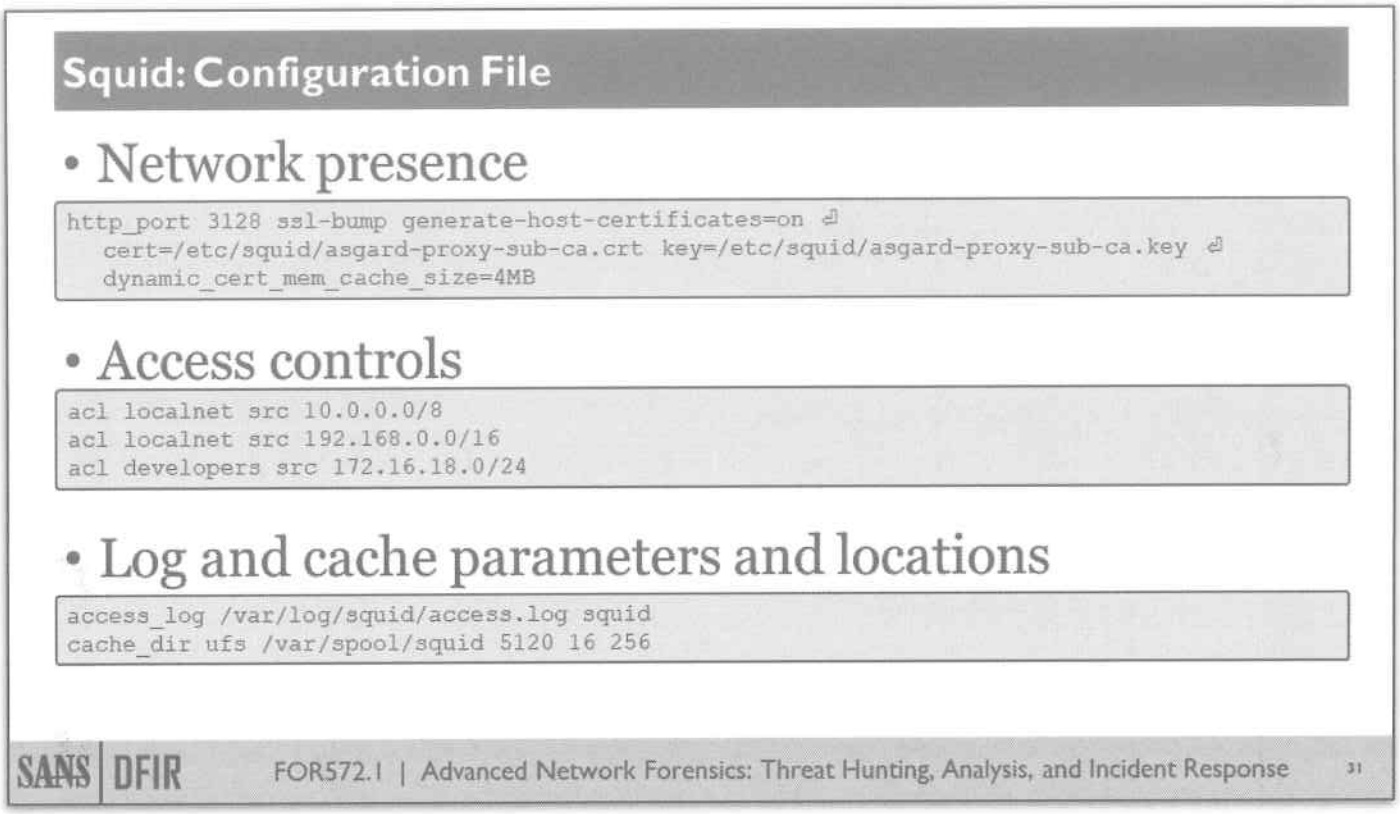
A skilled Squid administrator can configure fleets of proxy servers to work in unison, leverage varying cache retention policies for different types of files (keep .exe files for longer than .html, for example), or at least a hundred other interesting deployment options. For this reason, the proxy server configuration file is a key piece of evidence to collect and review during an investigation. This will be a common theme throughout the course—the configuration files will prove invaluable in determining the location and meaning of other sources of evidence. They should be collected soon after a device or system is determined to be relevant to the investigation. Doing so will ensure that useful evidence is not overlooked or lost due to spoliation. The default `/etc/squid/squid.conf` file is generally kept in `/etc/squid/squid.conf.default`. This is a *very* well-documented file and should be consulted to fully understand the various configuration settings available to the Squid server.

Other critical evidence provided by a Squid proxy server are the log files and cache data. (The values shown are typical defaults—be sure to verify them in the configuration file, of course.)

Log files provide what is essentially an access roster for all client requests that the proxy handled, while the responses that web servers provided to those requests can often be recovered from the proxy's cache.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



As with any network-facing service, Squid listens for incoming connections on a specified port. By default, this is TCP port 3128. For various reasons, an administrator may wish to change this value, however. Knowing Squid's network profile will be helpful as you incorporate other devices, such as firewalls, into the investigation.

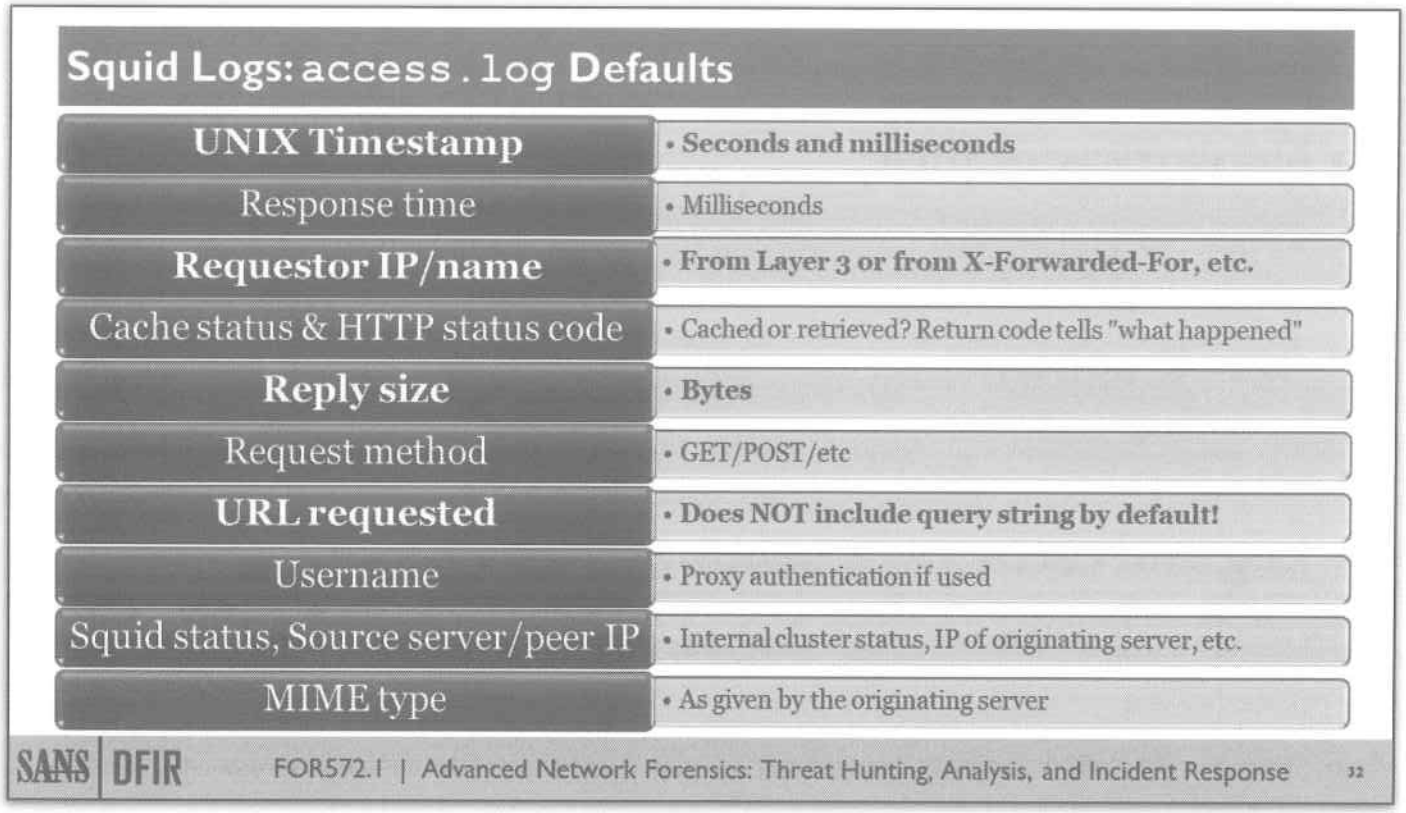
The administrator can also enable Squid's "bump server first" features, as shown here. This gives Squid the functionality needed to provide SSL/TLS interception features. While this incurs important privacy and legal concerns that must be addressed, the increasing use of SSL/TLS has led to many organizations enabling such features after a careful review between technical, legal, and management teams. This feature requires control of each endpoint being intercepted, however. Each client must be preconfigured to trust the signing certificate authority used by the proxy, or each SSL/TLS request would trigger a warning. Even still, some software may not work in this scenario, should the developer have configured "certificate pinning", a technology that hard-codes the expected SSL/TLS certificate directly into the binary.

Squid administrators can create access control lists to aid in creating fine-grained behaviors based on numerous characteristics of proxy traffic. Besides the IP-based ACLs shown here, these characteristics may include user authentication, "User-Agent" strings (a browser software identification string), day and time constraints, regular expression matching on the requested URLs, and more. Although such detailed tuning is outside the scope of this course, you should browse the contents of `/etc/squid/squid.conf.default` to become familiar with the kind of configuration options available through these access control lists.

Predictably, the Squid configuration file also specifies the parameters and locations for our other two sources of evidence: The log file and cache data. We will review the contents of a squid log file, which documents the requests clients have made for web-based content, in greater detail during this module. The cache directory provides access to the reply traffic that web servers provided to clients behind the proxy. During an investigation, we can extract those replies—which often contain all kinds of useful data, including the contents of web pages and downloads such as malware and other binaries. This cache data is very useful to an investigator, as it permits us to substantiate or refute theories about HTTP activity.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



By default, Squid provides a wealth of data for each request it receives from a client system. During the course of an investigation, these log records can quickly and decisively establish a complete picture of what HTTP traffic has traversed the proxy server. Because attackers often use HTTP for malware installation (for example, during a spear phishing operation) as well as command and control (operations, updates, exfiltration, etc.), the proxy server is an ideal place to gather comprehensive information.

Typical Squid proxy log entries look like the following (the definition for each of the ten items in the boxes below are reflected in the slide):

```
1509038269.433    531 192.168.75.19 TCP_MISS/200 17746  GET
  http://www.nu.nl/ - DIRECT/62.69.184.21 text/html
1509038295.705    246 192.168.75.19 TCP_REFRESH_HIT/304 303 GET
  http://www.nu.nl/ - DIRECT/62.69.184.21 -
```

Of particular note is that query strings are not logged by default. To enable this feature, add the following to `/etc/squid/squid.conf`:

```
strip_query_terms off
```

WARNING: Enabling this option has privacy implications for users behind the proxy! Be sure to consult with the proper authorizing parties before using it.

The cache status messages indicate whether the requested resource was served from the cache itself or if the proxy server had to retrieve a copy to satisfy the request. There are a number of these, detailed in the Squid documentation[1].

References:

[1] `http://for572.com/13ftw`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Squid Logs: Custom Formats

- Default format leaves some things out
  - User-agent strings, Referer URLs, human-readable date/timestamp
- `logformat` directive in `squid.conf`
  - Permits custom log formats

```
logformat combined   %>a %[ui %[un [%tl] "%rm %ru HTTP/%rv" %>Hs %<st ⏎
  "%{Referer}>h" "%{User-Agent}>h" %Ss:%Sh
access_log /var/log/squid/access.log combined
```

```
10.8.0.6 - - [26/Nov/2017:14:25:05 +0000] "GET
  http://assets.adobedtm.com/ad708e6ebc3df7970b1e4295ad1877b18b06f04f/satelliteLib-
  2ce6978cdc0bd4a63a115a542a6fd97a908d0648.js HTTP/1.1" 200 38986
  "http://www.getwestlondon.co.uk/news/west-london-news/heathrow-airport-security-worker-
  charged-13955421" "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:47.0) Gecko/20100101
  Firefox/47.0" TCP_MISS:HIER_DIRECT
```


The default `access_log` format is good, but sometimes just doesn't address all of our requirements. In particular, neither HTTP User-Agent strings nor Referer URLs are logged. And most of all, Squid defaults to the ever-popular UNIX timestamp, consisting of the number of seconds elapsed since midnight on January 1, 1970.

The "User-Agent" string most frequently identifies the web browser that humans use to browse the Internet. We will discuss User-Agent strings in greater detail later in the course.

The HTTP "referer" header contains the URL of the page that sent a client to the current page. For example, when clicking on a result from within Google, the user is directed to "`http://foo.com/page.php`". The request to "`foo.com`" includes a referer header with the full URL of the Google search result page. If a user clicks a link within a mail client, enters the URL to the "Location" bar directly, or clicks a bookmark in their browser, the "Referer" tag is generally empty.

The HTTP referer value is helpful to us because it can help establish a user's path through a web session. Malware may use a long and complicated sequence of redirects and other "hops" during the infection and operation processes, which could be difficult to sequence without the referer values. Similarly, users generally forget the details of their web browsing when asked, especially if a lot of time has passed since the time period in question. Maintaining a comprehensive log can help provide vital clarity in those and similar situations.

To overcome these limitations, Squid allows an administrator to define a custom log format, containing the required data in the best format. Although we will be discussing the default log format in greater detail, the ability to change the log format can be a very powerful tool. Squid even allows the creation of multiple simultaneous log destinations, created based on its own internal access control lists. This is a great benefit in Network Forensics because we can work with the proxy administrator to create another log file containing ONLY entries related to a particular set of known-malicious traffic patterns, or a secondary log file with a more comprehensive data set.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Here is one example of the syntax for this configuration directive:

```
logformat combined   %>a %[ui %[un [%tl] "%rm %ru HTTP/%rv" %>Hs %<st ⏎
   "%{Referer}>h" "%{User-Agent}>h" %Ss:%Sh
access_log /var/log/squid/access.log combined
```

These field format codes used above are:

- `%>a` Client source IP address
- `%ui` User name from ident
- `%un` User name (any available)
- `%tl` Local time. Optional strftime format argument
- `%rm` Request method (GET/POST etc.)
- `%ru` Request URL from client (historic, filtered for logging)
- `%rv` Request protocol version
- `%>Hs` HTTP status code sent to the client
- `%<st` Total size of reply sent to client (after adaptation)
- `%{Referer}` HTTP Referer header
- `%{User-Agent}` HTTP User-agent header
- `%Ss` Squid request status (TCP_MISS etc)
- `%Sh` Squid hierarchy status (DEFAULT_PARENT etc)

A sample log entry using the format string above is shown here:

```
10.8.0.6 - - [26/Nov/2017:14:25:05 +0000] "GET
   http://assets.adobedtm.com/ad708e6ebc3df7970b1e4295ad1877b18b06f04f/satell
   iteLib-2ce6978cdc0bd4a63a115a542a6fd97a908d0648.js HTTP/1.1" 200 38986
   "http://www.getwestlondon.co.uk/news/west-london-news/heathrow-airport-
   security-worker-charged-13955421" "Mozilla/5.0 (Windows NT 10.0; WOW64;
   rv:47.0) Gecko/20100101 Firefox/47.0" TCP_MISS:HIER_DIRECT
```

You can see that the log entry contains a human-readable UTC date, as well as the requested URL (`http://assets.adobedtm.com/.....js`), the referer URL (`http://www.getwestlondon.co.uk/.....`), and "User-Agent" string (`Mozilla/5.0.....Firefox/47.0`).

The full range of options for the `logformat` directive is quite extensive.

References:

`http://for572.com/fc60a`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Squid Logs: Analytic Tools

- calamaris
  - `calamaris.cord.de`
- sarg
  - `sarg.sourceforge.net`
- squidview
  - `rillion.net/squidview`
- And ... 100 others
  - `squid-cache.org/Misc/log-analysis.html`

Owing to Squid's popularity, there are countless ways to analyze, visualize, and otherwise report on the status of Squid's cache, log files, and other behaviors. Each has its own benefits and drawbacks, but most were designed for proxy administrators rather than forensic professionals. That said, we can often benefit from the open-source nature of these tools to improve our insight into the data we have available.

As with any automated tools or parsers, consider the value each can provide, then use (and/or modify) those that best suit your circumstances and style.

The URLs for the listed tools at the time of this writing are:

- `calamaris` `http://calamaris.cord.de`
- `sarg` `http://sarg.sourceforge.net`
- `squidview` `http://rillion.net/squidview`

The Squid project itself maintains an updated list of analysis tools.[1]

References:

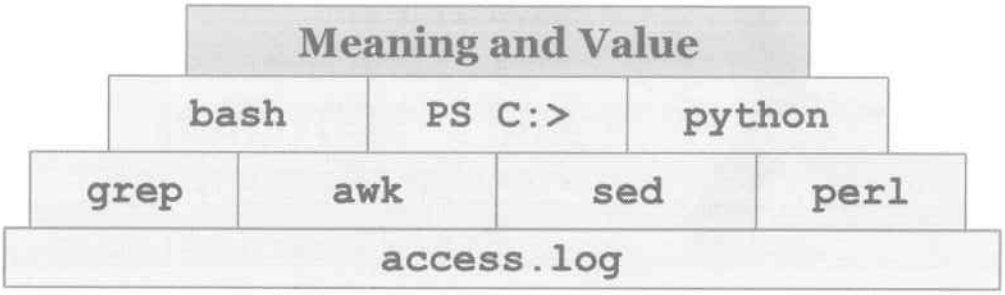
[1] `http://for572.com/qawcf`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Squid Logs: Raw Analysis

- Raw text logs mean all you need is a shell!
- Automated tools can identify outliers, but digging into logs can divine meaning



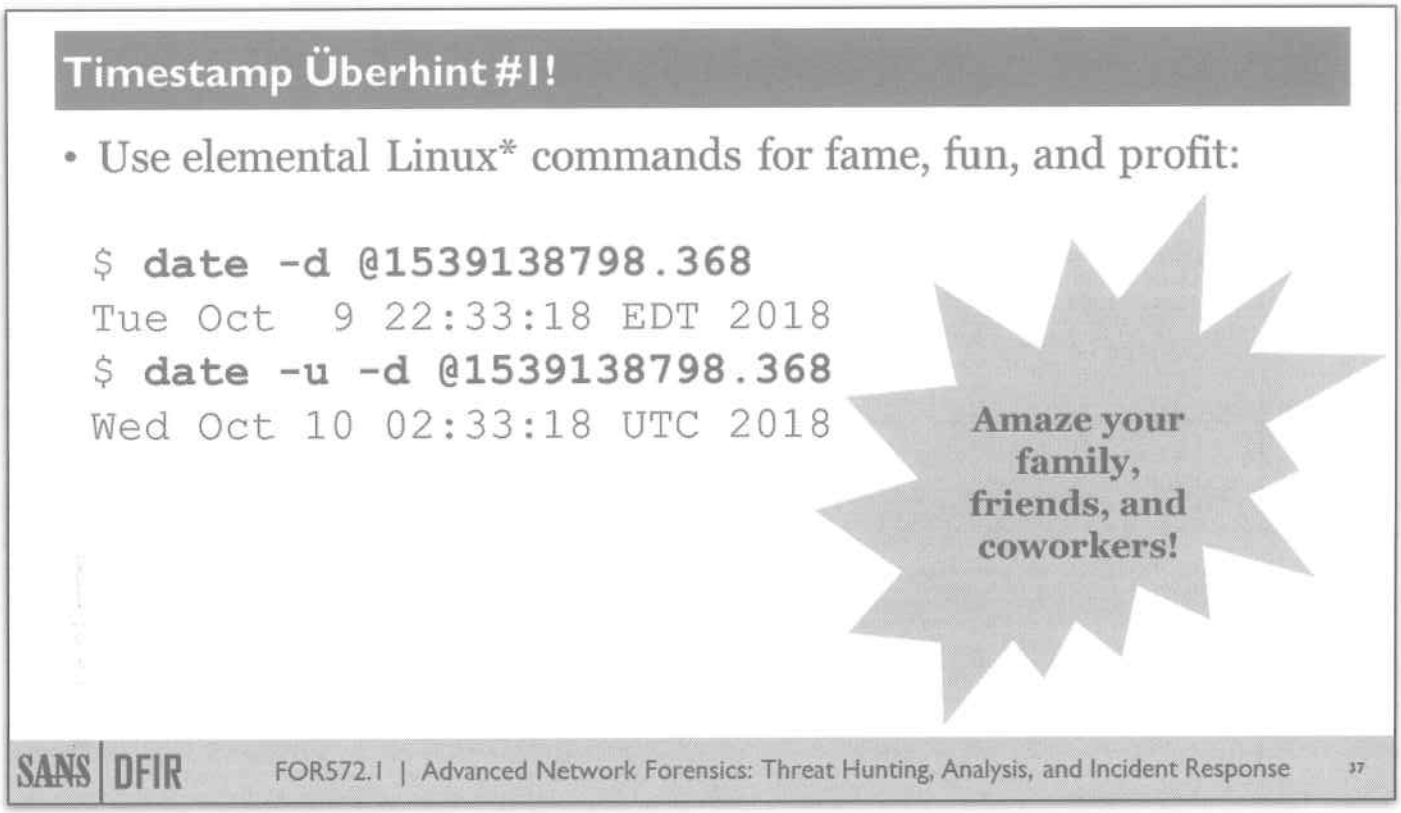
The beauty of Squid's log files is that they are simply plain text, ready for you to slice, dice, and forge into meaningful information. Although it is possible to do this with proprietary formats, or binary/database log records, it is often difficult. At a minimum, it requires additional steps to derive the native formats into a normalized form—before analysis can start.

The text-based nature of these log files also means that we can use the tools we know best. This, in turn, means that we will more expeditiously arrive at our goal: Meaningful findings from the evidence.

Automated tools are excellent to narrow down vast amounts of source data to a form that makes identifying outliers easy. We use that machine guidance to lead human resources to the data that will help answer the questions at hand.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



As previously mentioned, one of the oddities of the default Squid log format is that timestamps are in the less-than-human-readable UNIX epoch. This counts the time elapsed since January 1, 1970, at 00:00:00 UTC.

Because we'll already be working at the shell prompt for much of our work with Squid log files, this trick will prove useful. Simply run the LINUX "date" command, substituting any UNIX timestamp into the field after the @ sign. By default, the command will display the human-readable value in the local time zone. If you add the "`-u`" switch, the command will display the time in UTC.

```
$ date -d @1539138798.368
Tue Oct  9 22:33:18 EDT 2018
$ date -u -d @1539138798.368
Wed Oct 10 02:33:18 UTC 2018
```

Even some UNIX veterans may be impressed with your use of this trick!

* Note that some versions of the "`date`" utility do not behave in this way. However, the version distributed with your SIFT Workstation (as well as that included with most modern Linux distributions) supports this syntax and use case. To do this in Apple macOS, for example, the syntax is similar (but note that the seconds value needs to be truncated to an integer):

```
$ date -j -r 1539138798
Tue Oct  9 22:33:18 EDT 2018
$ date -j -u -r 1539138798
Wed Oct 10 02:33:18 UTC 2018
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Timestamp Überhint #2!

- Convert a lot of timestamps at once

```
$ sudo cat /var/log/squid/access.log | ⏎
  awk '{$1=strftime("%F %T", $1, 1); ⏎
  print $0}'
2018-06-09 12:30:48
   403 192.168.75.119 TCP_MISS/301 522
   GET http://cnn.com/ -
   DIRECT/157.166.255.18 text/html

$ sudo cat /var/log/squid/access.log | ⏎
  squidtime
2018-06-09 12:30:48
   403 192.168.75.119 TCP_MISS/301 522...
```

(Thanks and apologies: hyperboleandahalf.com)

Converting timestamps one at a time is a neat trick, but with a quick `awk` command, we can convert all the timestamps at once! Way cooler…

Without going into great detail on the nuances of the command, the following awk statement will quickly and efficiently convert UNIX epoch timestamps to human readable UTC, leaving the rest of each line intact. Feel free to explore the awk syntax to change the date formatting to your liking. The format here will permit chronological sorting due to the "big endian"-like structure.

```
$ sudo cat /var/log/squid/access.log | ⏎
  awk '{$1=strftime("%F %T", $1, 1); ⏎
  print $0}' > /tmp/squid_access_humanreadable.log
```

What's nice about this method is that it will also work inline with any `grep` or other text manipulation commands you use to cull Squid's log data.

```
$ sudo grep google.com /var/log/squid/access.log | ⏎
  awk '{$1=strftime("%F %T", $1, 1); print $0}'
2018-06-09 12:30:37 243 192.168.75.119 TCP_MISS/200 17962 GET
http://www.google.com/ - DIRECT/173.194.73.105 text/html
2018-06-09 12:30:38 231 192.168.75.119 TCP_MISS/204 279 GET
http://clients1.google.com/generate_204 - DIRECT/173.194.43.38 text/html
```

As a convenience, we've provided this command to you in the custom FOR572 SIFT VM as an alias named "`squidtime`". You can review the `sansforensics` user's "`~/.bash_aliases`" file or a corresponding note in the Evernote notebook to see how to do this for your own needs.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Log Walkthrough: Intro

- Illustrate value of proxy logs during network forensic process:
  - Squid handles all perimeter-crossing HTTP and HTTPS traffic
  - Query string, user agent, referer logged
  - SSL/TLS interception approved and implemented
  - Believed "dirty" employee leaking IP
  - Employee apprehended at gate of international flight with no IP or technology on person
  - Forensics showed only job-appropriate IP access

Now we'll walk through an example where just reviewing the proxy logs will provide critical insight to an incident. This scenario was based on a real-world event, with some details altered to protect the guilty. The very, very guilty.

The files for this walkthrough are available on your course USB drive in the "`/lab_data/demo-01/`" directory, which has already been extracted to the "`/cases/for572/demo-01/`" directory on your class SIFT Workstation VM. Since you have the data, don't just watch, follow along—call out if you think you find something that looks interesting or out of place.

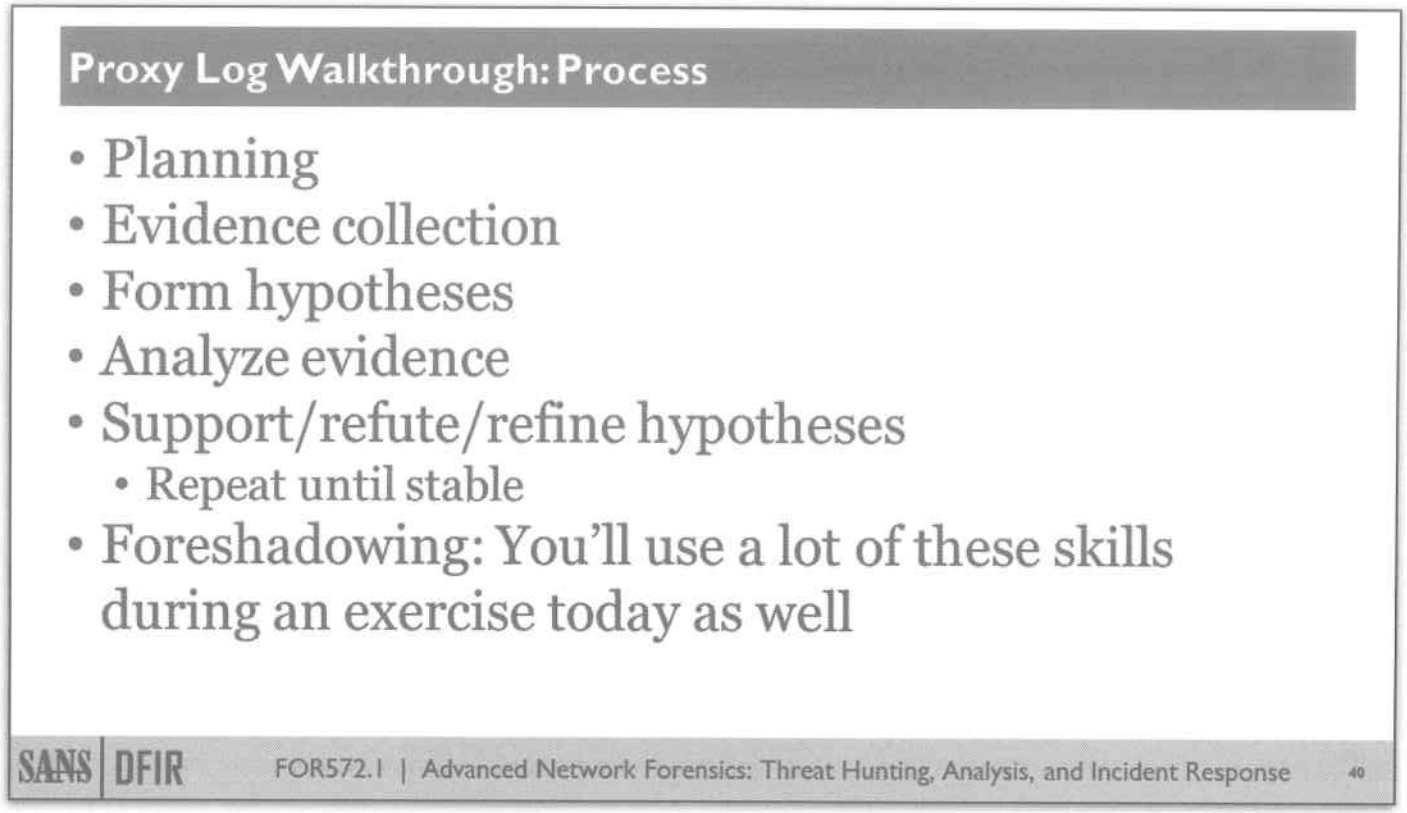
For the purposes of this walkthrough, our enterprise network is properly configured and all web traffic from client systems is funneled through a Squid proxy server. Firewalls and other access control mechanisms prevent any leakage. Our proxy server is configured to log all query terms, and users have been duly informed of this through the acceptable use policy that each must sign upon hiring. The "`access.log`" file uses a custom format that includes human-readable timestamps, referer strings, and user-agent strings. SSL/TLS proxying using dynamic certificate generation has been authorized and enabled.

In this scenario, there was an existing suspicion that an employee was leaking intellectual property. The employee abruptly left work one afternoon and did not report to work the next day. The employee was apprehended at the gate of an international flight to London (Virgin flight 22, IAD->LHR) due to an exceedingly large sum of cash in his luggage that had not been properly declared. He also had a ticket from London to Singapore (Singapore Airlines 305 LHR->SIN). The employee did not have corporate intellectual property of any kind, nor any technology items (USB thumb drives, laptops, phones, etc.) on his person when he was apprehended. However, given the suspicious confluence of these events and the existing concern about the employee's loyalties, management has directed us to investigate.

A triage analysis of his workstation did not identify any artifacts of IP access outside the employee's normal duties, nor any other obvious artifacts of wrongdoing. A more comprehensive analysis of that workstation is underway, but management wants us to perform a network forensic investigation in parallel to that work.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



First, we'll plan our investigation. Obviously, we take into account the resources and evidence we'd like to access, as well as the time allotted for the task. We'll also consider what existing analysis has been completed—the triage work on the employee's workstation (`192.168.75.119`). In this case, we can use evidence from the proxy server.

Next, we'll collect evidence. The following items have been deemed available and relevant:

- `/etc/squid/squid.conf`: Configuration file, on next slide. Identifies locations of log files, and configuration directives that describe the behavior of the proxy server.
- `/var/log/squid/access.log`, `/var/log/squid/referer.log`, `/var/log/squid/useragent.log`, `/var/log/squid/cache.log`, `/var/log/squid/store.log`: Squid proxy log files. Contents have been minimized to include only records from the employee's workstation.

Consider some initial hypotheses or focus areas that you'd like to pursue. Remember that we're on a short timeline, and aren't after ultimate ground truth at this point—we just need to determine enough to make an informed recommendation regarding whether or not we feel the employee leaked any intellectual property.

As we continue the walkthrough, we will discuss whether each new finding supports or refutes our developing hypotheses and whether those should be refined in light of new observations within the evidence.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Log Walkthrough: `squid.conf` file

- Contents of `/etc/squid/squid.conf`
  - See Notes page
- Generally default CentOS 7 version except:
  - Enabled 500GB disk cache
  - Enabled logging HTTP query strings, user agents, and referer strings via custom log format
  - Enabled SSL/TLS interception with "bump-server-first"
  - Custom `access.log` format
  - Locally-based IP ranges


```
#
# Recommended minimum configuration:
#

# Example rule allowing access from your local networks.
# Adapt to list your (internal) IP networks from where browsing
# should be allowed
#acl localnet src 10.0.0.0/8     # RFC1918 possible internal network
#acl localnet src 172.16.0.0/12  # RFC1918 possible internal network
#acl localnet src 192.168.0.0/16 # RFC1918 possible internal network
acl localnet src 10.3.56.0/23  # STARK General
acl localnet src 10.3.58.0/24  # STARK Key
acl localnet src 10.3.59.0/24  # STARK Floor
acl localnet src 10.3.16.0/24  # STARK DMZ
acl localnet src fc00::/7       # RFC 4193 local private network range
acl localnet src fe80::/10      # RFC 4291 link-local (directly plugged)
   machines

acl SSL_ports port 443
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 443         # https
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http
acl CONNECT method CONNECT
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

```
#
# Recommended minimum Access Permission configuration:
#
# Deny requests to certain unsafe ports
http_access deny !Safe_ports

# Deny CONNECT to other than secure SSL ports
http_access deny CONNECT !SSL_ports

# Only allow cachemgr access from localhost
http_access allow localhost manager
http_access deny manager

# We strongly recommend the following be uncommented to protect innocent
# web applications running on the proxy server who think the only
# one who can access services on "localhost" is a local user
#http_access deny to_localhost

#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#

# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
http_access allow localnet
http_access allow localhost

# And finally deny all other access to this proxy
http_access deny all

# Squid normally listens to port 3128
#
#  Added ssl bump feature to handle SSL interception
sslpassword_program /usr/local/bin/pass.sh
http_port 3128 ssl-bump generate-host-certificates=on cert=/etc/squid/asgard-
   proxy-sub-ca.crt key=/etc/squid/asgard-proxy-sub-ca.key
   dynamic_cert_mem_cache_size=4MB

always_direct allow all
ssl_bump server-first all
sslproxy_cert_error deny all
sslproxy_flags DONT_VERIFY_PEER
sslcrtd_program /usr/lib64/squid/ssl_crtd -s /var/lib/ssl_db -M 4MB
   sslcrtd_children 8 startup=1 idle=1

icap_enable             on
icap_preview_enable     on
icap_preview_size       128
icap_send_client_ip     on
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

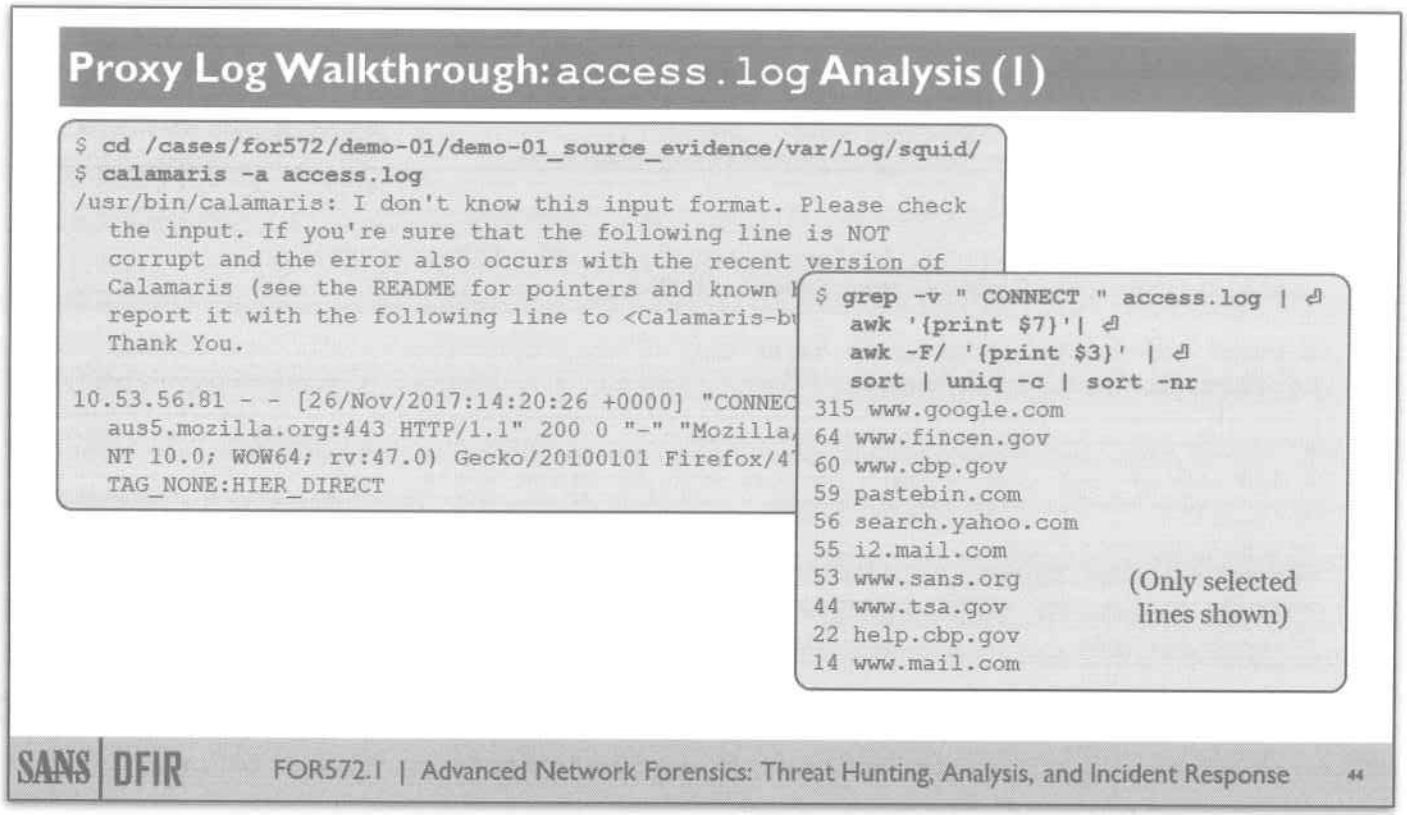
```
##  Adjust logging:
logformat combined   %>a %[ui %[un [%tl] "%rm %ru HTTP/%rv" %>Hs %<st
   "%{Referer}>h" "%{User-Agent}>h" %Ss:%Sh
access_log stdio:/var/log/squid/access.log combined
strip_query_terms off

# Uncomment and adjust the following to add a disk cache directory.
cache_dir ufs /var/spool/squid 500000 16 256

# Leave coredumps in the first cache dir
coredump_dir /var/spool/squid

#
# Add any of your own refresh_pattern entries above these.
#
refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern -i (/cgi-bin/|\?) 0     0%      0
refresh_pattern .               0       20%     4320
```

https://t.me/learningnets



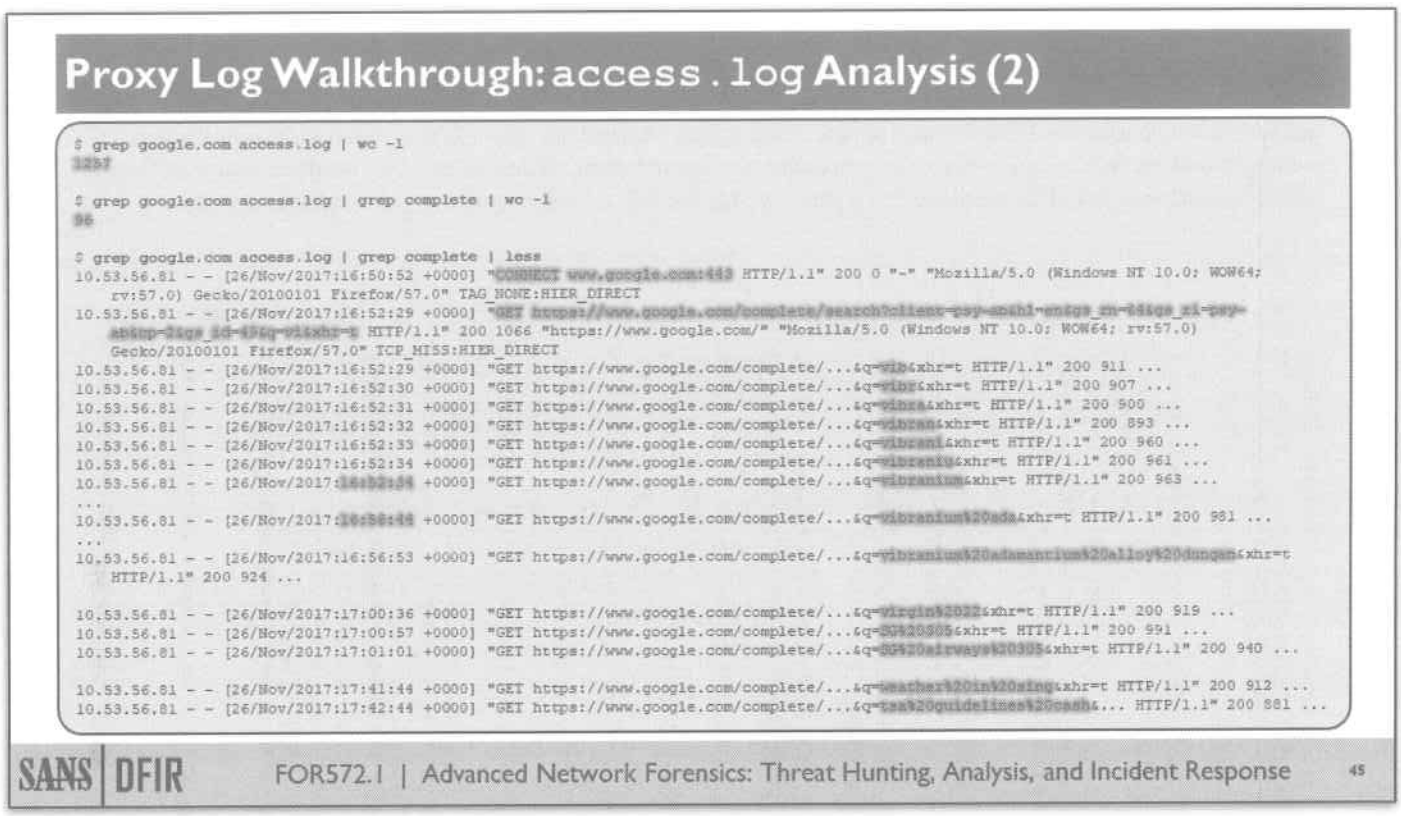
Let's start with an automated proxy log analysis tool to see if there are any standout items. We'll use `calamaris` here, but any summarization tool would provide roughly the same data. Remember that our scenario dictated the logs were already reduced to the employee's workstation's IP address. Logs from a real proxy server would contain *much* more data!

Sadly, the automated tool is of no use for this data set. While `calamaris` is a great tool for the overall toolbox, the custom log format on this proxy server isn't easily handled. So in this case, we'll need to walk away from the "easy option" and see what else we can do. (Though it does bear mentioning that the open-source nature of the `calamaris` tool means an enterprising developer could add more features to the codebase, allowing additional log formats!)

Instead, consider the simple yet effective sequence of shell primitives on the right. While it may take a few minutes to get your head around the syntax used, it's an easy way to get a quick look at the most common hostnames in the HTTP and HTTPS requests contained in the log files. (Note that we've omitted lines that use the "CONNECT" request method, as these entries document the initial stage of an SSL/TLS communication, and have a different format. We'll see more of these in the rest of this section.)

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Let's start by looking at the Google URLs. It's reasonable to assume these may detail the employee's search activity, and the log records show that to be the case. However, nearly 2,700 requests including the "`google.com`" hostname somewhere in the record are probably more than we want to wade through manually. Fortunately, the "suggested answer" feature for most search engines works because with each new keypress, the browser makes an API call (usually with an AJAX feature) via HTTP. As you can see here, the practical result of this feature in the proxy logs is not far from a keylogger-like record of the employee's activity! (Good thing we have those signed Acceptable Use Policy documents, isn't it?)

Through these records, we can get a good idea of the types of things the subject employee was looking for and at what time they did so. Notice there are both the "CONNECT" records and logged https:// URLs. This is a byproduct of the SSL/TLS interception feature. The "CONNECT" records indicate the beginning on an SSL/TLS session between the proxy and the client system, and the intercepted records are listed later.

The subject user searched for several terms of concern here at SRL. The "vibranium" project is particularly sensitive, as is the work being done between that matter and another known as "adamantium". The presence of a researcher's name (Dungan) also suggests this employee may be up to something fishy.

Later in the search activity, though, the subject employee searched for "`virgin 22`", which is the flight he was removed from, leaving IAD and arriving at LHR. The next search is for "`SG airways 305`", a likely search for a Singapore Airlines flight—the same one he was ticketed for from LHR to SIN. (However, you may have recognized that "SG" is not the proper airline code for this carrier—someone more familiar with this kind of travel would have looked for "`SQ 305`" perhaps.) The planned air travel to Singapore aligns with the apparent search for "`weather in sing`"[apore], which appears to have been truncated for some reason. Lastly, given the circumstances of the employee's apprehension, the "`tsa guidelines cash`" search is particularly interesting.

https://t.me/learningnets

But wait—if the employee used his web browser to search for these, how did a triage analysis of his system miss the typical artifacts associated with web browsing and search? Perhaps the employee used Incognito mode, or manually cleared the contents of his browser's cache when he was done. Those artifacts may or may not be recoverable with more in-depth forensic review of his system, but they're quite plain as day from the proxy server's vantage point. In fact, the proxy server logs provide even greater depth of knowledge than would be expected from a full forensic analysis of the employee's system—we see the URLs built as granularly as one letter at a time.

```
$ grep google.com access.log | wc -l
1257

$ grep google.com access.log | grep complete | wc -l
96

$ grep google.com access.log | grep complete | less
10.53.56.81 - - [26/Nov/2017:16:50:52 +0000] "CONNECT www.google.com:443 HTTP/1.1" 200 0 "-"
   "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101 Firefox/57.0" TAG_NONE:HIER_DIRECT
10.53.56.81 - - [26/Nov/2017:16:52:29 +0000] "GET https://www.google.com/complete/search?client=psy-
   ab&hl=en&gs_rn=64&gs_ri=psy-ab&cp=2&gs_id=49&q=vi&xhr=t HTTP/1.1" 200 1066
   "https://www.google.com/" "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101
   Firefox/57.0" TCP_MISS:HIER_DIRECT
10.53.56.81 - - [26/Nov/2017:16:52:29 +0000] "GET https://www.google.com/complete/...&q=vib&xhr=t
   HTTP/1.1" 200 911 ...
10.53.56.81 - - [26/Nov/2017:16:52:30 +0000] "GET https://www.google.com/complete/...&q=vibr&xhr=t
   HTTP/1.1" 200 907 ...
10.53.56.81 - - [26/Nov/2017:16:52:31 +0000] "GET https://www.google.com/complete/...&q=vibra&xhr=t
   HTTP/1.1" 200 900 ...
10.53.56.81 - - [26/Nov/2017:16:52:32 +0000] "GET https://www.google.com/complete/...&q=vibran&xhr=t
   HTTP/1.1" 200 893 ...
10.53.56.81 - - [26/Nov/2017:16:52:33 +0000] "GET https://www.google.com/complete/...&q=vibrani&xhr=t
   HTTP/1.1" 200 960 ...
10.53.56.81 - - [26/Nov/2017:16:52:34 +0000] "GET https://www.google.com/complete/...&q=vibraniu&xhr=t
   HTTP/1.1" 200 961 ...
10.53.56.81 - - [26/Nov/2017:16:52:34 +0000] "GET
   https://www.google.com/complete/...&q=vibranium&xhr=t HTTP/1.1" 200 963 ...
...
10.53.56.81 - - [26/Nov/2017:16:56:44 +0000] "GET
   https://www.google.com/complete/...&q=vibranium%20ade&xhr=t HTTP/1.1" 200 981 ...
...
10.53.56.81 - - [26/Nov/2017:16:56:53 +0000] "GET
   https://www.google.com/complete/...&q=vibranium%20adamantium%20alloy%20dungan&xhr=t HTTP/1.1" 200
   924 ...

10.53.56.81 - - [26/Nov/2017:17:00:36 +0000] "GET
   https://www.google.com/complete/...&q=virgin%2022&xhr=t HTTP/1.1" 200 919 ...
10.53.56.81 - - [26/Nov/2017:17:00:57 +0000] "GET https://www.google.com/complete/...&q=SG%20305&xhr=t
   HTTP/1.1" 200 991 ...
10.53.56.81 - - [26/Nov/2017:17:01:01 +0000] "GET
   https://www.google.com/complete/...&q=SG%20airways%20305&xhr=t HTTP/1.1" 200 940 ...

10.53.56.81 - - [26/Nov/2017:17:41:44 +0000] "GET
   https://www.google.com/complete/...&q=weather%20in%20sing&xhr=t HTTP/1.1" 200 912 ...
10.53.56.81 - - [26/Nov/2017:17:42:44 +0000] "GET
   https://www.google.com/complete/...&q=tsa%20guidelines%20cash&... HTTP/1.1" 200 881 ...
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Log Walkthrough: access.log Analysis (3)

```
$ grep mail.com access.log
10.53.56.81 - - [26/Nov/2017:17:08:29 +0000] "GET https://www.mail.com/ HTTP/1.1" 200
  16756 ...

10.53.56.81 - - [26/Nov/2017:17:09:26 +0000] "POST https://login.mail.com/login HTTP/1.1"
  302 507 ...

10.53.56.81 - - [26/Nov/2017:17:38:12 +0000] "POST https://login.mail.com/login HTTP/1.1"
  302 507 ...

10.53.56.81 - - [26/Nov/2017:17:38:28 +0000] "POST https://3c-
  lxa.mail.com/mail/client/mail/compose/html;...&editorAction=NEW HTTP/1.1" 200 967 ...

10.53.56.81 - - [26/Nov/2017:17:41:14 +0000] "GET https://3c-
  lxa.mail.com/mail/client/mail/sentconfirm;...&uc=SHOW_CONFIRMPAGE HTTP/1.1" 200 7161
  ...

10.53.56.81 - - [26/Nov/2017:17:41:23 +0000] "GET https://www.mail.com/logout/ HTTP/1.1"
  200 15169 ...
```


The previous list of hostnames in the log included several on the "`mail.com`" domain. This is one of hundreds of different webmail providers, perhaps indicating that our subject used the service to send a message. An initial examination of the logs seems to confirm this.

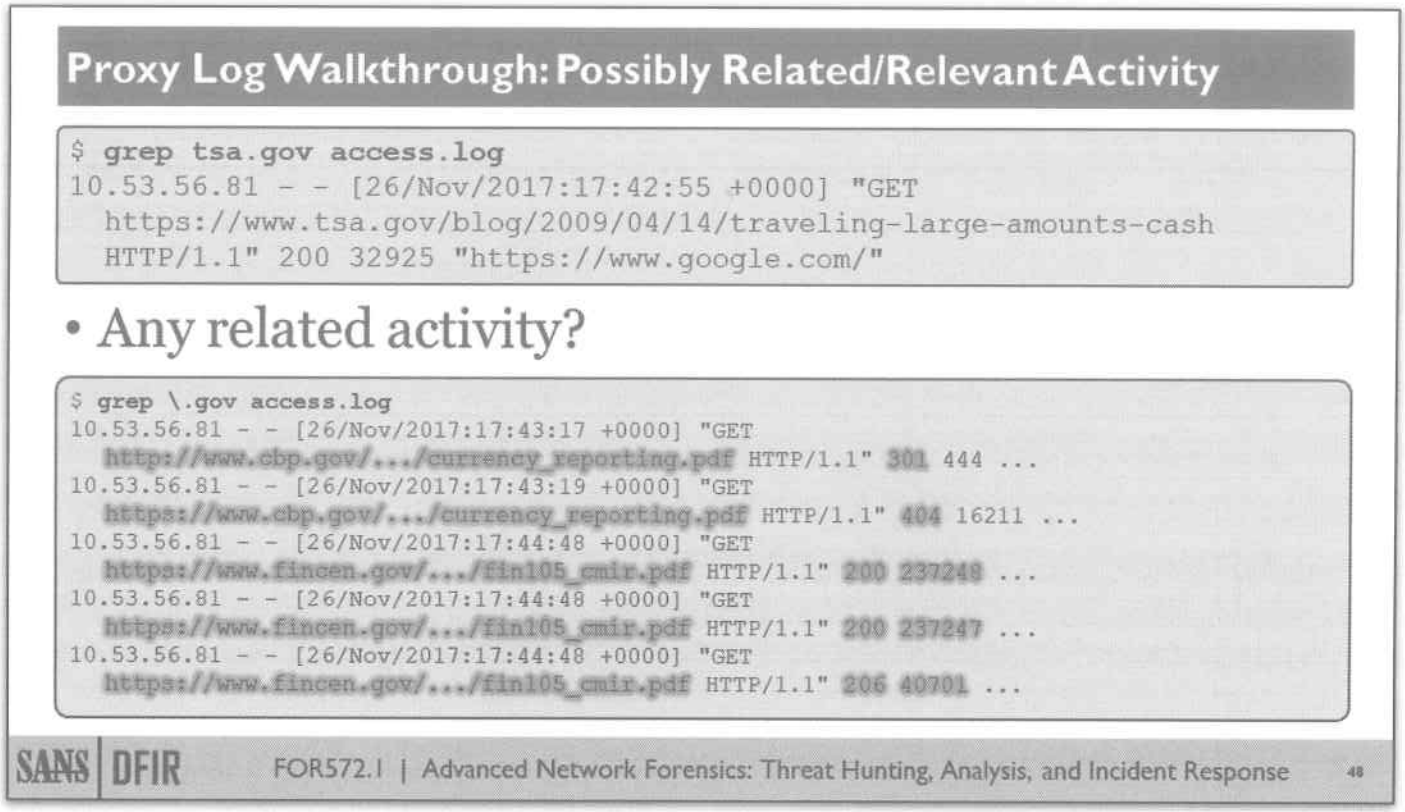
We can see what appears to be two login events, but they are nearly a half hour apart. During the second session, we can see what appears to be the subject composing a new email. There are a number of these events in the log (though just one is shown above), which might be the result of AJAX-like features. AJAX allows the web application to exchange HTTP data in both directions without reloading the page—perhaps providing an "autosave" functionality. However, since these use the POST HTTP request method, the useful data seems to have been included in the HTTP body. This is not logged or cached, as it consists of data that's not part of the URL and is transmitted out of the environment—not requiring any cache optimization.

Eventually, there appears to be a message sent, as the "`sentconfirm`" and "`SHOW_CONFIRMPAGE`" components of the logged URL. While it is a reasonable hypothesis that these are the result of sending a message, we can't tell for sure without testing. A good method for this would be to use `mail.com` to send an email, then observe the evidence the proxy creates for the test case. By comparing the two sets of evidence, we can generally get a reasonably good idea of what occurred during the subject's activity on the site.

Lastly, we see the subject appears to have explicitly logged out of `mail.com`—but again, this could be confirmed by similarly observing a test case.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



The TSA search string and corresponding domain result are interesting—by `grep`'ing the `access.log` file and walking through the results, we can see what URLs the employee may have visited around those searches. There are several, but one, in particular, stands out:

```
10.53.56.81 - - [26/Nov/2017:17:42:55 +0000] "GET
   https://www.tsa.gov/blog/2009/04/14/traveling-large-amounts-cash HTTP/1.1"
   200 32925 "https://www.google.com/"
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

We could visit this page to see what is there, but let's also consider any related activity—perhaps using the ".gov" top-level domain. (Recall that we also saw "cbp.gov" in the domain search.)

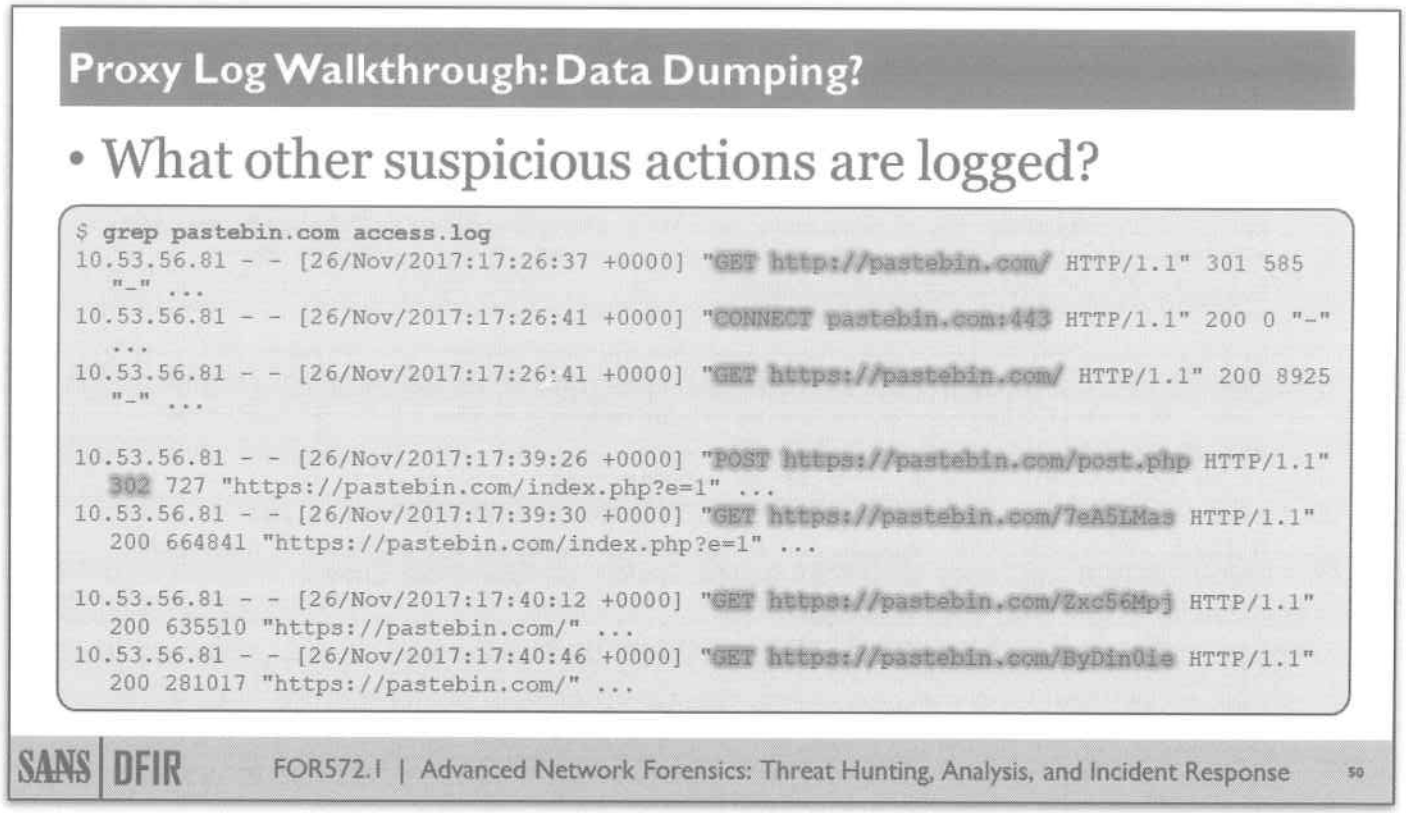
```
$ grep \.gov access.log
10.53.56.81 - - [26/Nov/2017:17:43:17 +0000] "GET
   http://www.cbp.gov/linkhandler/cgov/newsroom/publications/travel/currency_
   rpt_flyer/currency_reporting.ctt/currency_reporting.pdf HTTP/1.1" 301 444
   "-" "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101
   Firefox/57.0" TCP_MISS:HIER_DIRECT
10.53.56.81 - - [26/Nov/2017:17:43:19 +0000] "GET
   https://www.cbp.gov/linkhandler/cgov/newsroom/publications/travel/currency
   _rpt_flyer/currency_reporting.ctt/currency_reporting.pdf HTTP/1.1" 404
   16211 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101
   Firefox/57.0" TCP_MISS:HIER_DIRECT
10.53.56.81 - - [26/Nov/2017:17:44:48 +0000] "GET
   https://www.fincen.gov/sites/default/files/shared/fin105_cmir.pdf
   HTTP/1.1" 200 237248 "https://www.fincen.gov/forms/files/fin105_cmir.pdf"
   "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101
   Firefox/57.0" TCP_MISS:HIER_DIRECT
10.53.56.81 - - [26/Nov/2017:17:44:48 +0000] "GET
   https://www.fincen.gov/sites/default/files/shared/fin105_cmir.pdf
   HTTP/1.1" 200 237247 "https://www.fincen.gov/forms/files/fin105_cmir.pdf"
   "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101
   Firefox/57.0" TCP_MISS:HIER_NONE
10.53.56.81 - - [26/Nov/2017:17:44:48 +0000] "GET
   https://www.fincen.gov/sites/default/files/shared/fin105_cmir.pdf
   HTTP/1.1" 206 40701 "https://www.fincen.gov/forms/files/fin105_cmir.pdf"
   "Mozilla/5.0 (Windows NT 10.0; WOW64; rv:57.0) Gecko/20100101
   Firefox/57.0" TCP_MEM_HIT:HIER_NONE
```

These requests for PDF files could be related—the first results in a redirect (return code 301), the second was not found (return code 404), and the third series of requests reflects three within the same one-second increment. One of these three showed partial content (return code 206), the other two were successful results (return code 200) but reflected a one-byte difference in requested size.

At this point, we could request the PDF directly from the site to see what it contains, but this might not always be possible. Operational security concerns could prohibit it, the object could have been removed from the active website, or the content could have been dynamically generated based on the requestor's IP address, cookies, or other settings.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Based on the domain review conducted previously, we saw there was activity involving the popular data sharing/dumping site, `pastebin.com`. The proxy logs may provide a detailed view of what may have been our subject walking out the virtual door with company property.

One useful "feature" of `pastebin.com`, in particular, is that after creating a new paste, the site sends you to the paste's "short URL" via an HTTP redirect. In this case, the URL is `http://pastebin.com/7eA5LMas`. There are additional pastes that were created a short time after this—with IDs `Zxc56Mpj` and `ByDin0ie`. These unique paste IDs could be very helpful in identifying exactly what was leaked because we should be able to view the paste ourselves. However, another "feature" of `pastebin.com` is that pastes can optionally be set to expire after a specified time period. In this case, the site indicates these paste pages are no longer available, so it appears we're out of luck, as least as far as the proxy logs are concerned.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

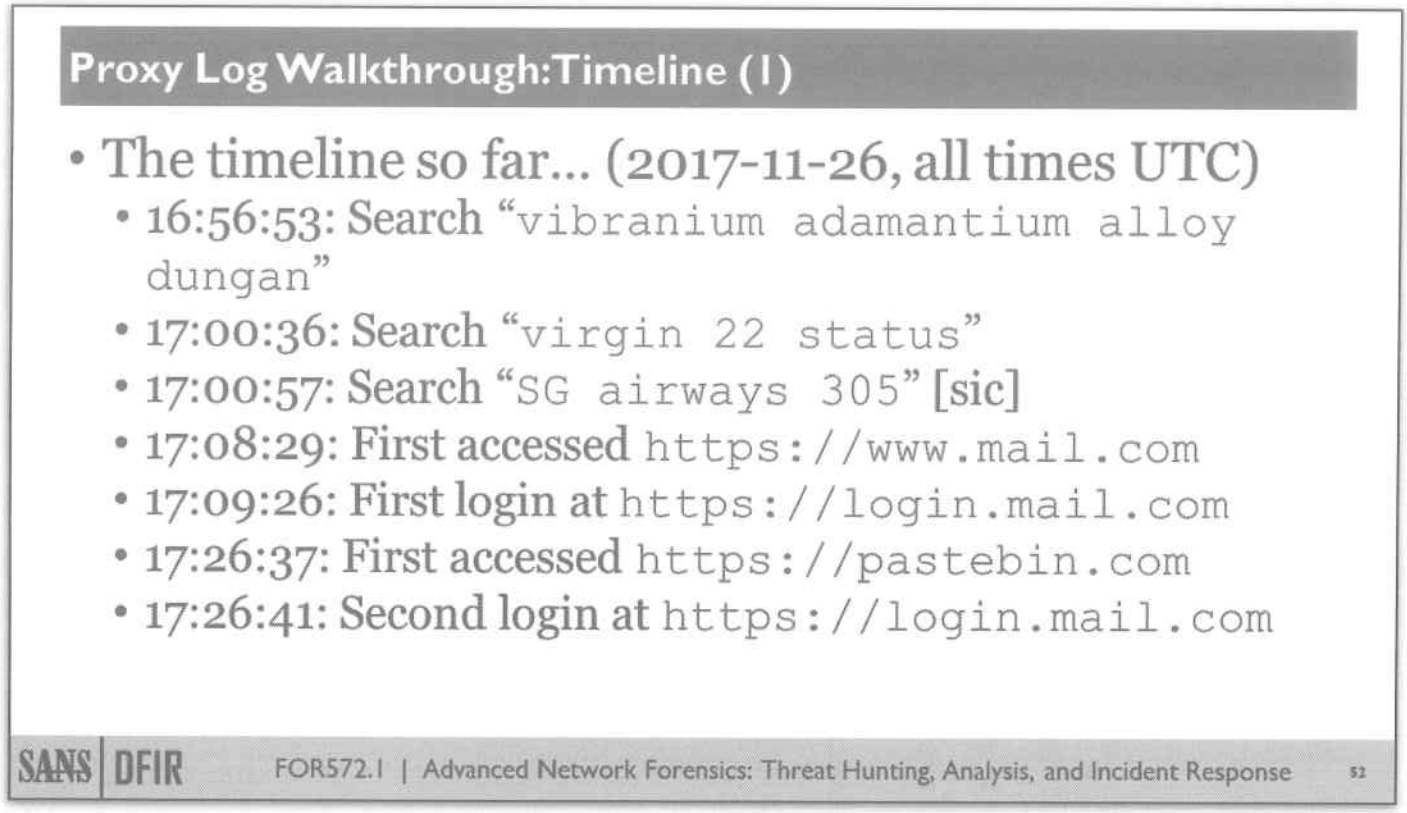
```
$ grep pastebin.com access.log
10.53.56.81 - - [26/Nov/2017:17:26:37 +0000] "GET http://pastebin.com/ HTTP/1.1" 301 585 "-" ...
10.53.56.81 - - [26/Nov/2017:17:26:41 +0000] "CONNECT pastebin.com:443 HTTP/1.1" 200 0 "-" ...
10.53.56.81 - - [26/Nov/2017:17:26:41 +0000] "GET https://pastebin.com/ HTTP/1.1" 200 8925 "-" ...

10.53.56.81 - - [26/Nov/2017:17:39:26 +0000] "POST https://pastebin.com/post.php HTTP/1.1" 302 727
"https://pastebin.com/index.php?e=1" ...
10.53.56.81 - - [26/Nov/2017:17:39:30 +0000] "GET https://pastebin.com/7eA5LMas HTTP/1.1" 200 664841
"https://pastebin.com/index.php?e=1" ...

10.53.56.81 - - [26/Nov/2017:17:40:12 +0000] "GET https://pastebin.com/Zxc56Mpj HTTP/1.1" 200 635510
"https://pastebin.com/" ...
10.53.56.81 - - [26/Nov/2017:17:40:46 +0000] "GET https://pastebin.com/ByDin0ie HTTP/1.1" 200 281017
"https://pastebin.com/" ...
```

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



Now that we've completed our proxy log analysis and established a good but non-sequential picture of the employee's activities the day before he left work and tried to leave the country, let's put everything in order.

All events occurred November 26, 2017, and all timestamps are (of course) in UTC format.

First, we saw some potentially interesting Google searches. These were for very sensitive materials, and the research behind them cost an exceedingly large amount of R&D funds. Of particular interest is that the employee included the name of a staff researcher who was involved with the breaking discovery of the process to create the material.

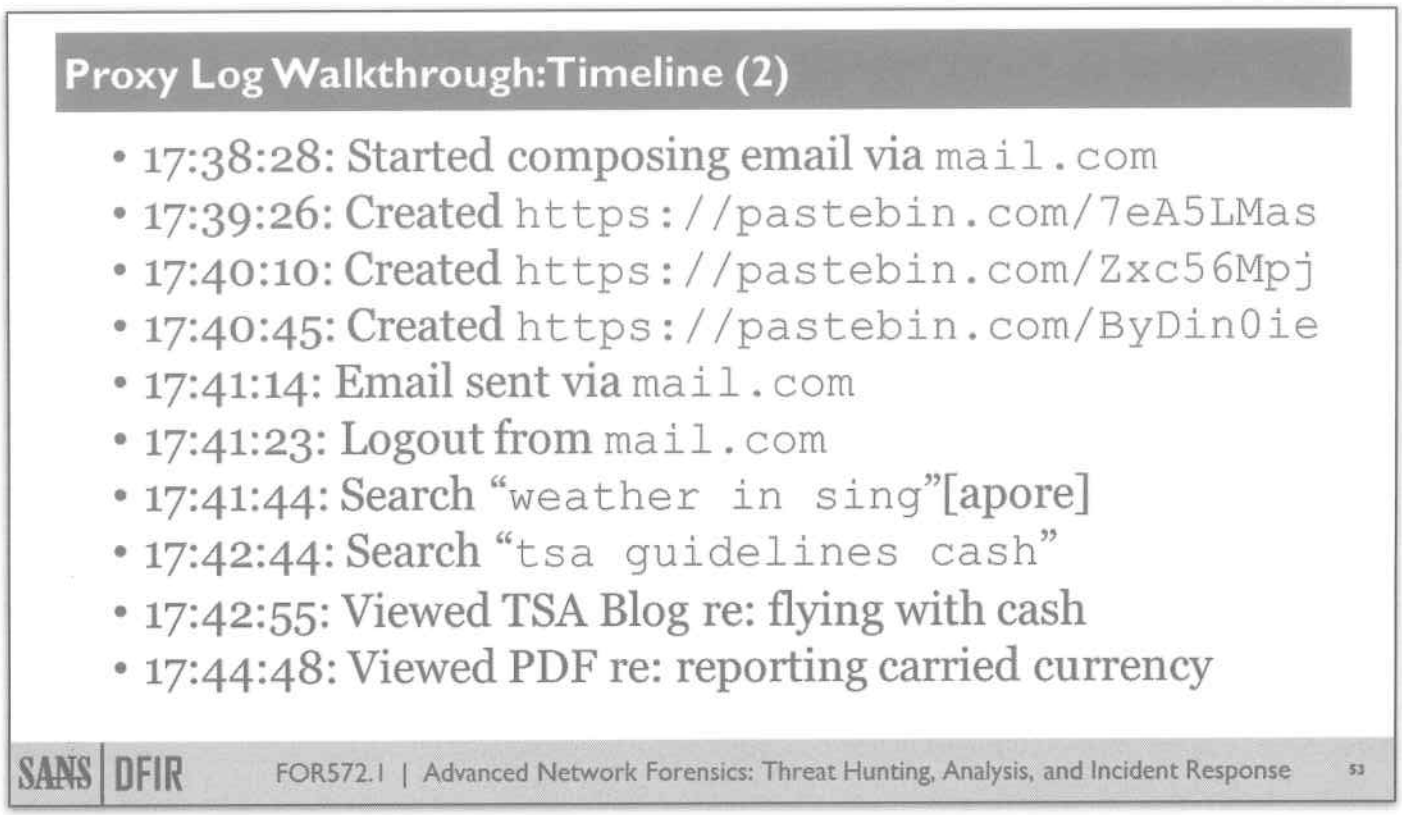
The suspected leaker then searched for the status of two flights—Washington/Dulles airport to London Heathrow, then Heathrow to Singapore. (Interestingly, Singapore is the nearest international airport to Madripoor, the location for the corporate base of operations of the victim company's chief competition.)

Approximately eight minutes later, the employee first accessed `mail.com`, a free webmail provider. About one minute after that, a login occurred.

Some time then passed before the next significant event. The employee first accessed `pastebin.com` and almost immediately logged back in at `mail.com`.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



After 12 minutes, the employee started writing an email via the `mail.com` interface.

The employee created three `pastebin.com` items, but the contents of these are unknown at this time. Their paste ID values might be useful if we could work with `pastebin.com` to recover the content that has been aged out of active use. This may be a fruitless venture, however—likely a nonstarter without law enforcement support or some other mechanism to garner the cooperation from the site's management.

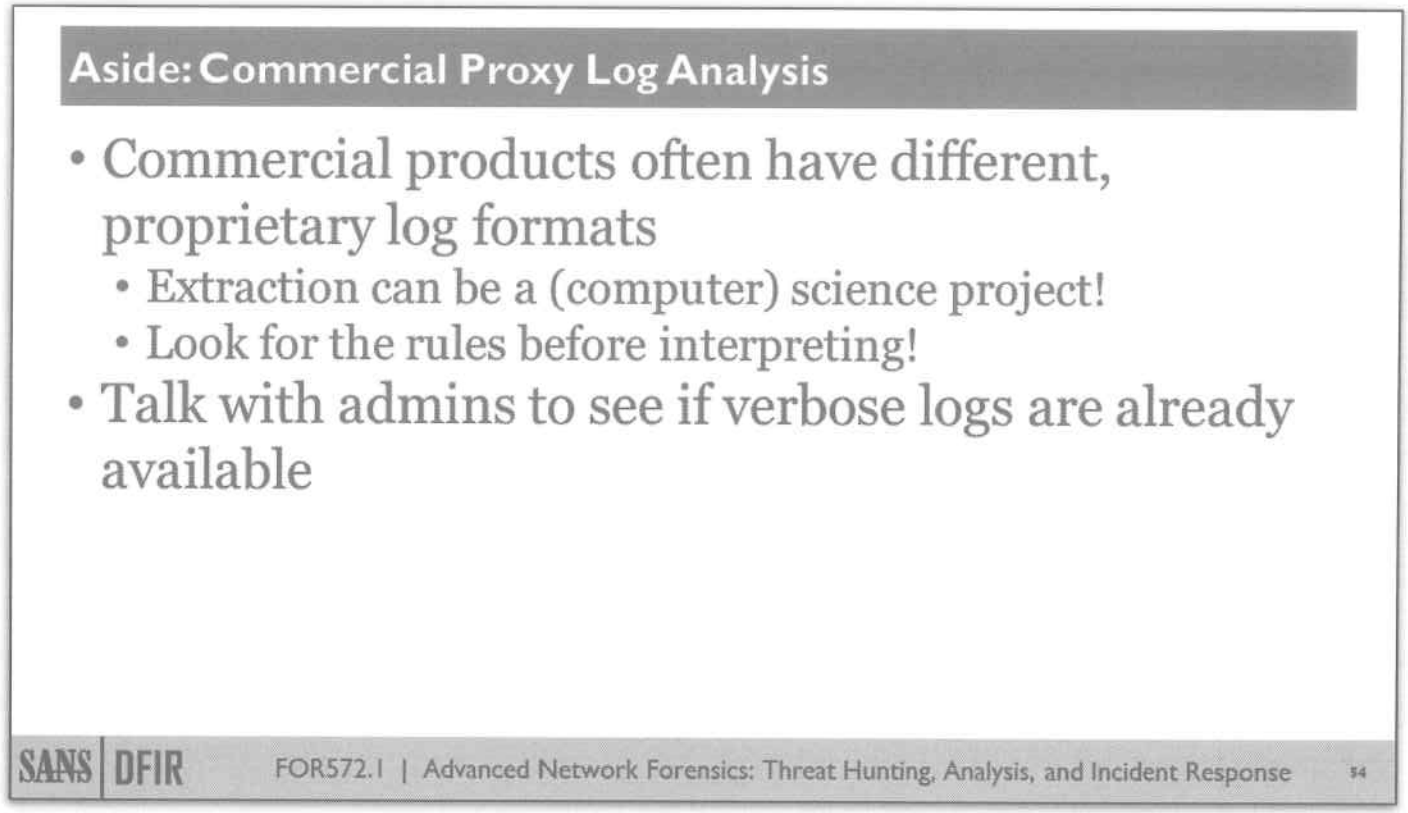
After the paste creation events, an email was sent via `mail.com` and the employee promptly logged out of the webmail service afterward.

The last few minutes of the examined activity involve searching for information regarding the weather in Singapore and how to transport large amounts of cash through airport security and across borders. The employee viewed a blog post from the Transportation Security Administration regarding the cash movement topic, and eventually ended up downloading the form required to declare such financial instruments in PDF form.

Given the circumstances under which the employee was apprehended (aboard aircraft to London with a large amount of undeclared cash on his person), our findings certainly point to a premeditated act. The most significant missing piece is that we have not yet been able to confirm the content of the pastes the employee created.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



We mentioned that commercial proxy servers are not always as easy to "get into" as an open source proxy like Squid. That doesn't mean that the data isn't available in a useful format, just that it takes some work to get there. For example, enterprises that use the Blue Coat proxy server can batch upload logs via FTP or export through additional third-party connectors like ArcSight and Forcepoint. At that point, a forensic analyst can *start* to try making heads or tails of the log entries, which look something like the following:

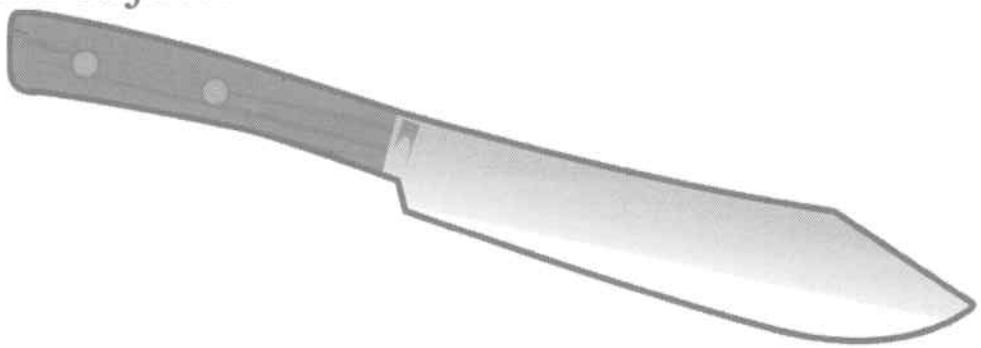
```
#Fields: date time time-taken c-ip sc-status s-action sc-bytes cs-bytes cs-
method cs-uri-scheme cs-host cs-uri-port cs-uri-path cs-uri-query cs-username
cs-auth-group s-hierarchy s-supplier-name rs(Content-Type) cs(Referer)
cs(User-Agent) sc-filter-result cs-categories x-virus-id s-ip
2012-04-20 11:05:04 145 10.10.8.33 304 TCP_HIT 291 2214 GET http
www.washingtonpost.com 80 /rw/WashingtonPost/Content/Staff-Bio/Images/cindy-
boren_80x72.jpg - leero MYDOMAIN DIRECT 63.233.110.25 image/jpeg
http://www.washingtonpost.com/sports "Mozilla/4.0 (compatible; MSIE 7.0;
Windows NT 5.1; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729; MS-RTC LM 8)" OBSERVED "News and Media" - 10.10.10.1
2012-04-20 11:05:04 50 172.29.221.56 407 TCP_DENIED 1126 1518 POST http
notification.yoono.com 80 /notification - - - NONE - - - "Mozilla/5.0
(Windows; U; Windows NT 5.1; en-US; rv:1.9.2.16) Gecko/20110319
Firefox/3.6.16" DENIED "Social Networking;Society and Lifestyles" -
10.10.10.17
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Cache Extraction

- Proxy servers usually cache data
  - Speeds up subsequent accesses
- Cached items can be deconstructed—or "carved"—into original objects

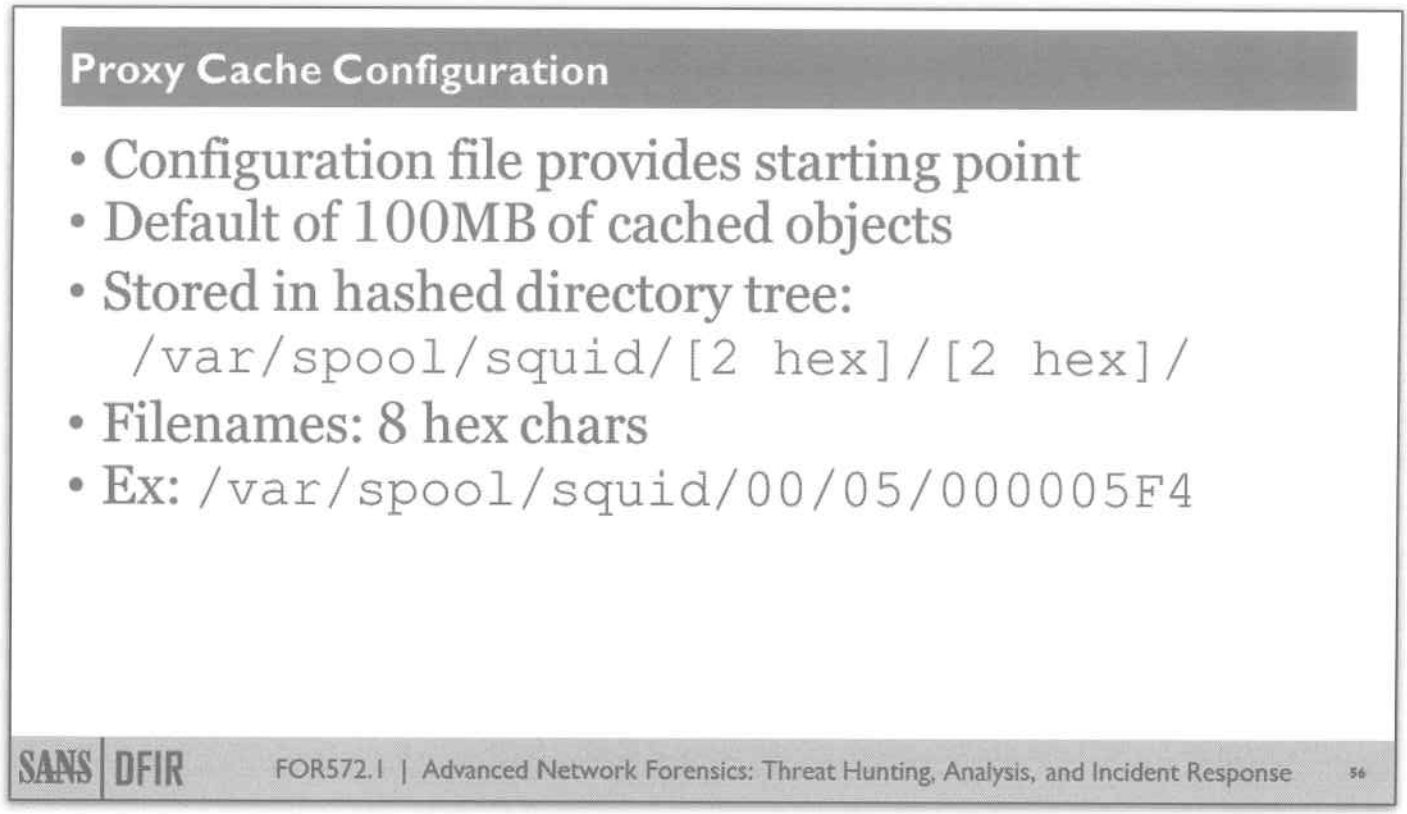


Recall the original use case for web proxies—efficiently use narrow links to the Internet by keeping local copies of frequently requested, unchanged content. In network forensics, we can use this to our advantage as well. Cached data can provide vital clues to the activities conducted on the network, including malware delivered to victims, data exfiltrated to outside parties, and more.

Performing this step requires knowledge of how a proxy server will identify which data to store, as well as how that data is packaged for storage on the server.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



The primary Squid configuration options we would be interested in while conducting a cache extraction would be the directory or directories in which cached files are stored, the ACLs that may affect what is or is not permitted to enter the cache, and many other settings.

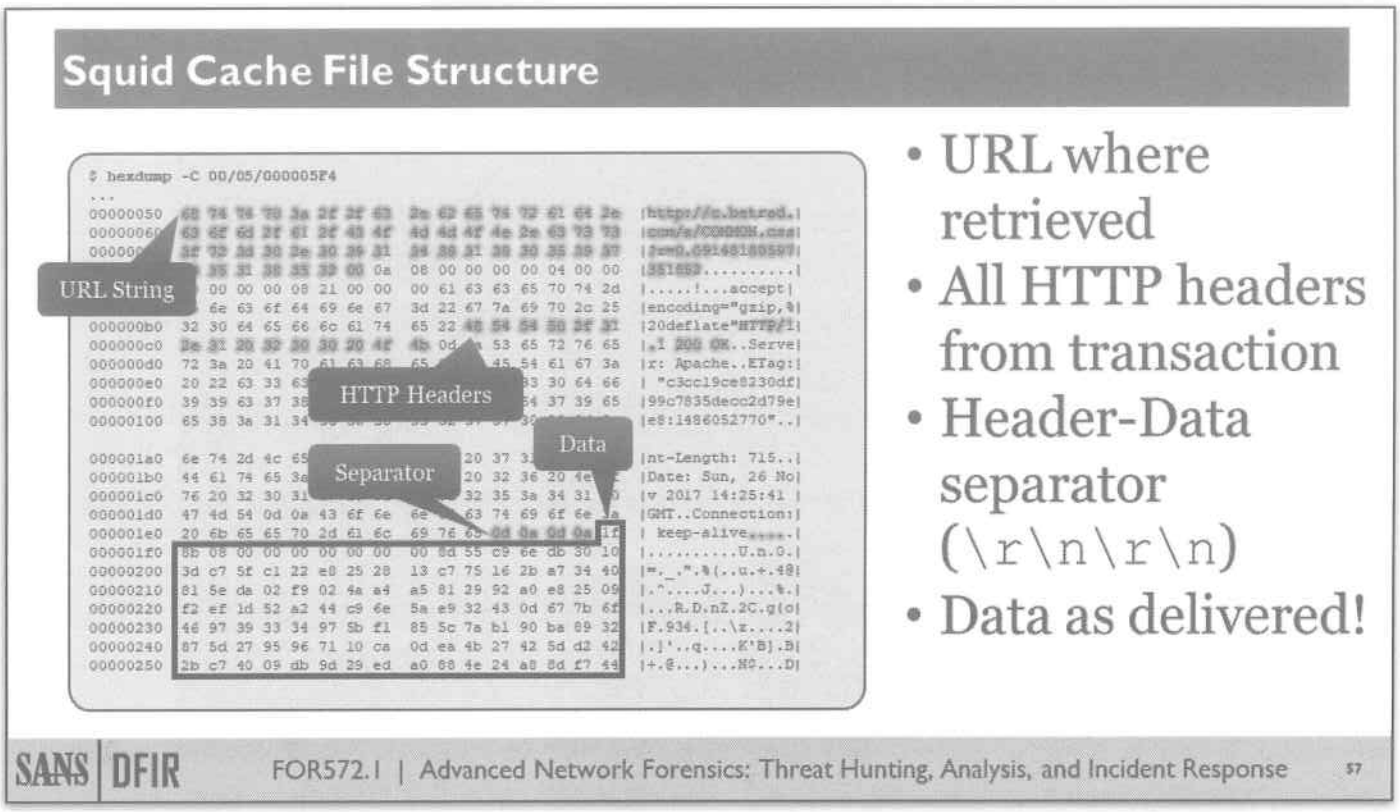
The cache directory seems straightforward, and in most cases, it is left as the default. Note, however, that there can be more than one cache directory specified. In such a case, the server "load balances" cached content among the multiple cache directories. If recovering the cache from a proxy server, always ensure you're acquiring all the cache data! There are also multiple tuning options for the several different cache file systems available. Again, because these are almost always left to their defaults (which are shown on the screen), we won't dive into the syntax here, but if you see a nonstandard configuration, be sure you are aware of the meaning.

The default configuration is very commonly used in Squid deployments. This includes the default cache location of "/var/spool/squid/". However, most administrators boost the disk space allocated for the cache from the baseline of just 100MB.

The cached files themselves are stored in a form such as "`/var/spool/squid/00/05/000005F4`". Unfortunately, such filenames do not give any insight to the origin, file type, size, time, or anything else we might like to see.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Although each cache file is generally recognized as a binary file, there are enough printable ASCII components that it is easy enough to visually parse. On a small enough scale, manual carving is a very reasonable approach.

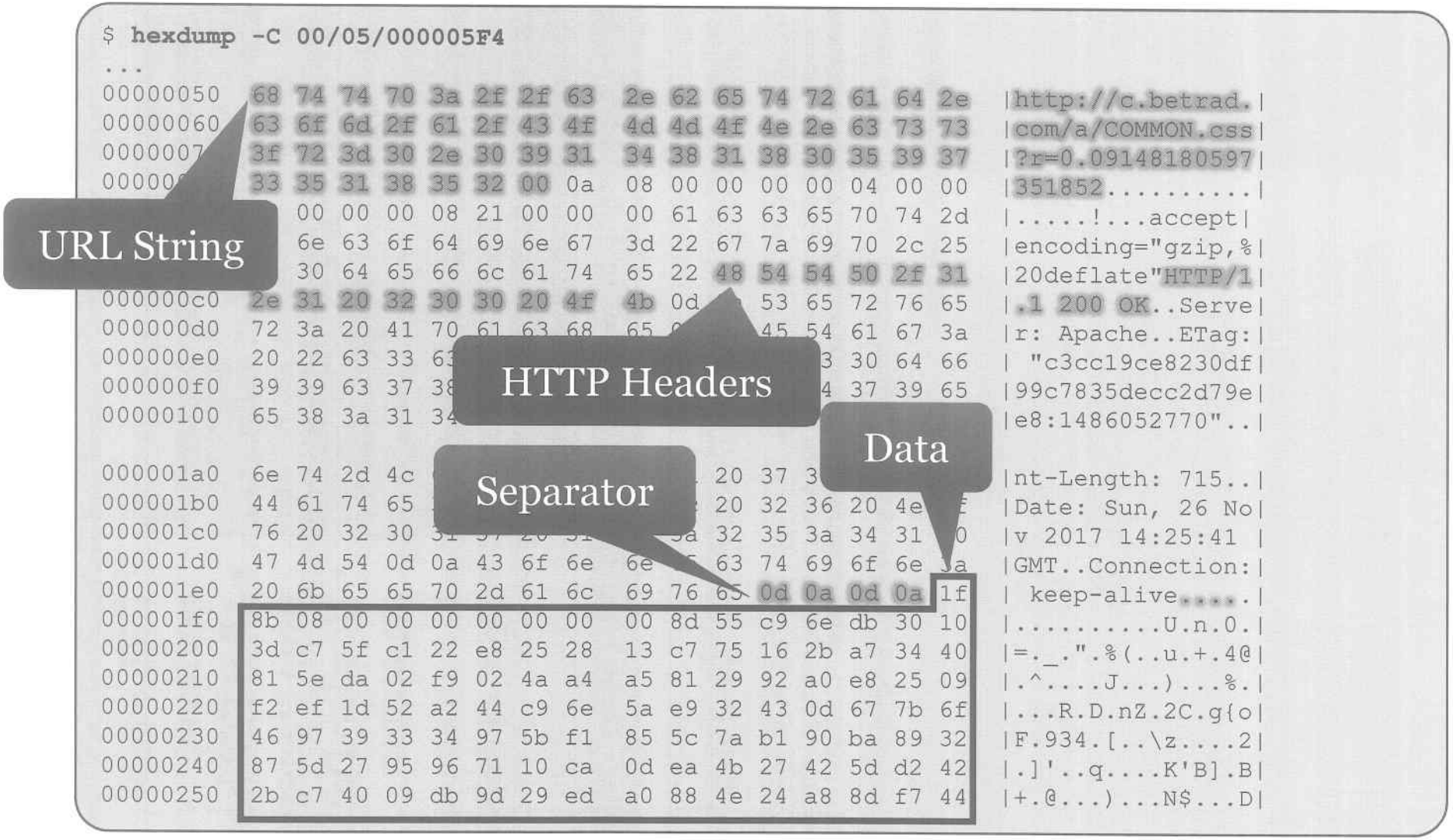
The first recognizable string is the URL, prefixed with the protocol designator. The headers that the server originally delivered are stored, as is the original data. In fact, from the start of the headers to the end of the cache object, the contents are exactly the response the server delivered for the original request.

The data section is easily identified by the double carriage-return/newline sequence, or hex bytes `0x0D0A0D0A`. This separator is dictated by the HTTP protocol specification itself. Immediately following that byte sequence is the first byte of the data portion of the cache object.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Squid Cache Objects: Quick Look (1)

- Run a "railgrep" against cache directory
- Cast a wide net against arbitrary data with embedded ASCII

- Many options to `grep`—use as needed
  - `-r`: Recurse
  - `-a`: Treat as ASCII
  - `-i`: Case insensitive
  - `-l`: List matching filenames
  - `-F`: Prevent regex engine

```
$ cd /cases/for572/demo-01/demo-01_source_evidence/var/spool/squid/
$ grep -rail www.google-analytics.com *
00/04/00000468
00/04/00000469
00/0C/00000C2E
00/0C/00000C5A
...
```

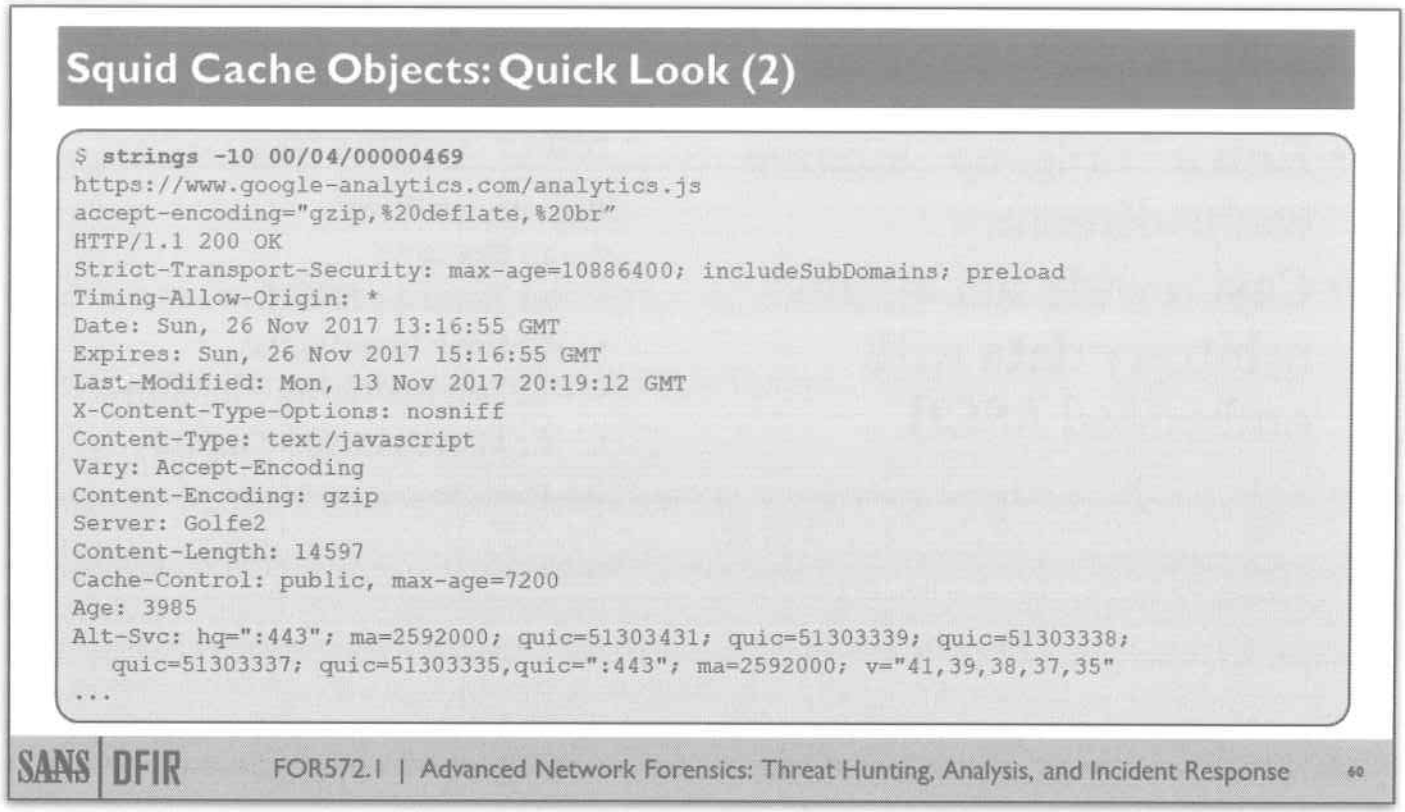

Because the cache objects contain useful ASCII text strings, we can use common shell utilities to examine the contents. To cast a wide net first, we use `grep` to identify cache objects of interest.

The switches to `grep` shown above are:

- `-r` Recursion through entire directory tree
- `-a` Treat all files as ASCII
- `-i` Case-insensitive search
- `-l` List matching filenames, not matching lines
- `-F` Disable the regular expression parsing engine, significantly increasing efficiency

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



After identifying items of interest with `grep`, as described above, we can use `strings` to quickly look at the header contents of each file.

In this example, the URL, HTTP headers, the MIME type, and many other metadata items describing the content contained in the cache object are shown.

The URL will be easily recognizable in a cache object file, so you could combine "`strings -10`" (to limit to strings that are 10 or more consecutive printable characters) with "`grep`" and "`head -1`" to retrieve just a list of the URLs that have been cached:

```
$ for file in $( grep -rail www.google-analytics.com ⏎
    /var/spool/squid/ ); do
      strings -10 $file | grep  ^http | head -1;
  done
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Walkthrough, Revisited

- One major remaining question: What was uploaded to `pastebin.com`?
- Pastebin performed 302 redirect to the paste itself—what if it was cached?

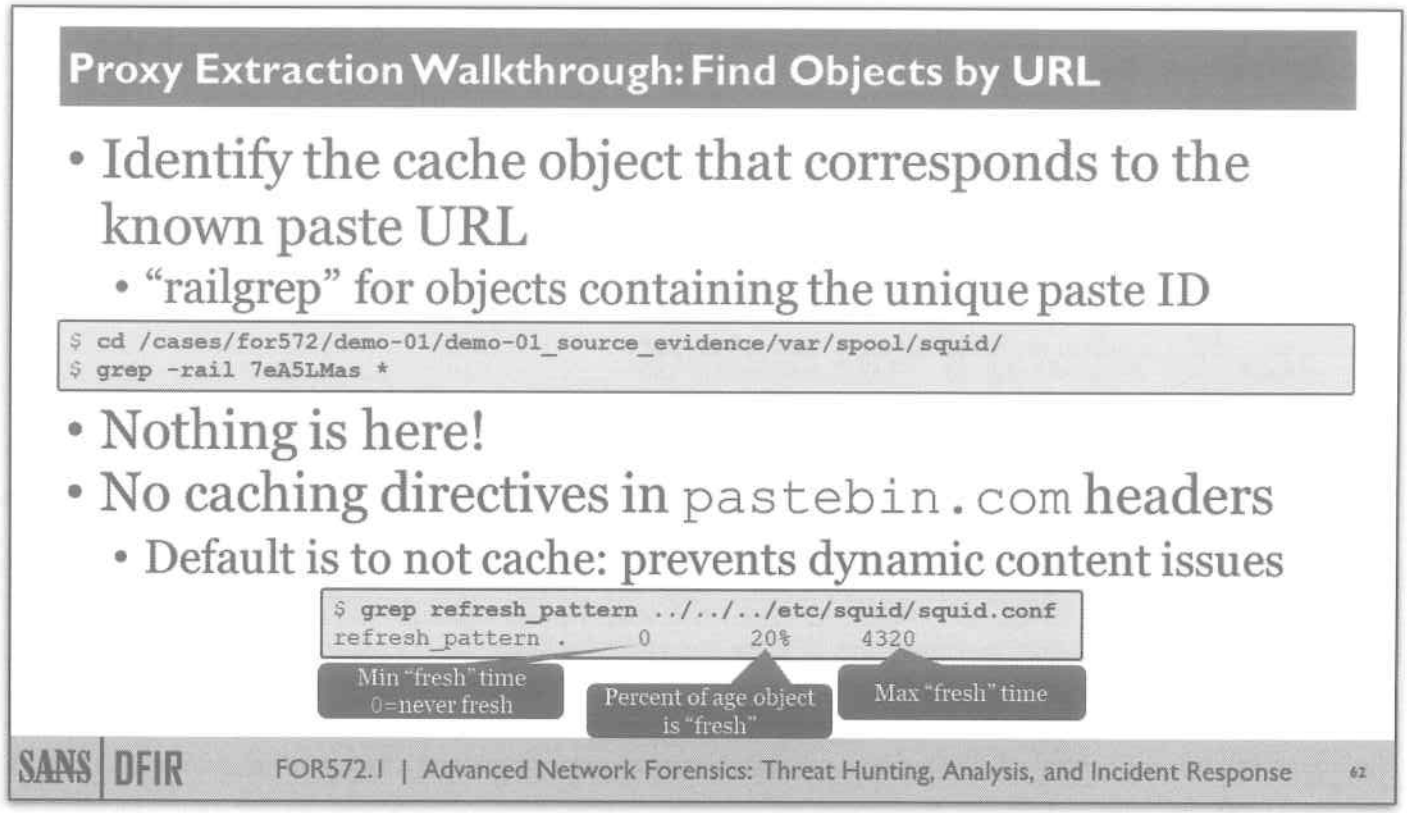
```
$ grep pastebin.com access.log

10.53.56.81 - - [26/Nov/2017:17:39:26 +0000] "POST https://pastebin.com/post.php HTTP/1.1"
  302 727 "https://pastebin.com/index.php?e=1" "Mozilla/5.0 (Windows NT 10.0; WOW64;
  rv:57.0) Gecko/20100101 Firefox/57.0" TCP_MISS:HIER_DIRECT
10.53.56.81 - - [26/Nov/2017:17:39:30 +0000] "GET https://pastebin.com/7eA5LMas HTTP/1.1"
  200 664841 "https://pastebin.com/index.php?e=1" "Mozilla/5.0 (Windows NT 10.0; WOW64;
  rv:57.0) Gecko/20100101 Firefox/57.0" TCP_MISS:HIER_DIRECT
```


Given our newfound ability to extract useful information from the Squid cache, let's think back to our likely source of leaking intellectual property. If you recall, the paste was no longer accessible, so we were unable to determine what data was on the paste. However, because pastebin.com performed an HTTP/302 redirect back to the newly created paste, our proxy may have caught the data. Let's investigate the data further.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



To start, let's look for any cache object that contains the URL for one of the pastes the employee created. Using `grep`, this results in… well, nothing at all. It seems this proxy's spool doesn't have anything about the paste. Further examination will show there is no content for the other pastes either.

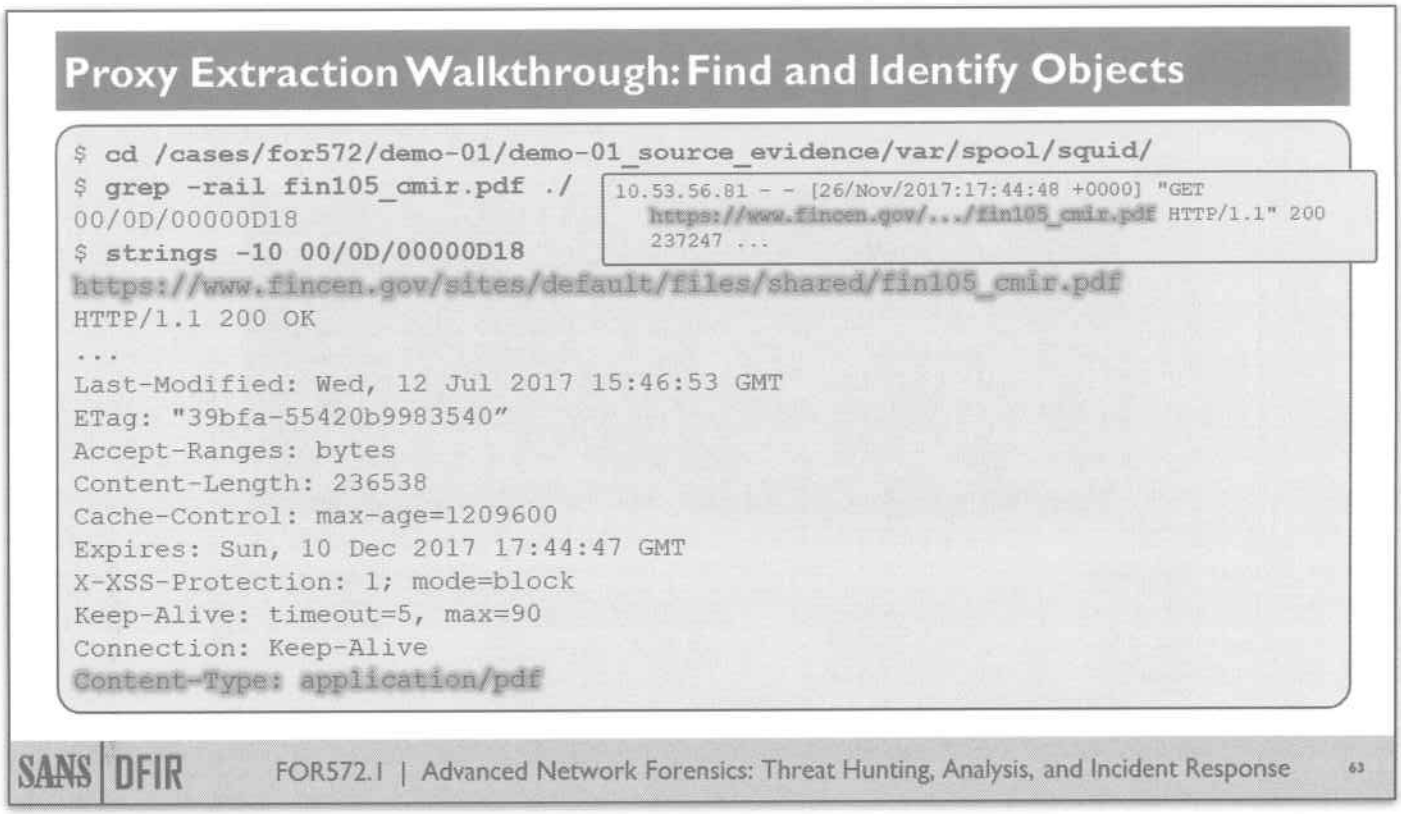
It turns out this happens more than you may expect, because the default proxy behavior is often to NOT cache returned objects without specific caching instructions that the server may opt to include. This may seem counterintuitive at first. However, it makes sense when you consider the increasingly dynamic nature of web-based content. If proxy servers were to aggressively cache dynamic content, subsequent requests for a previously-requested URL would return the cached content rather than the "live" version.

This behavior is, unsurprisingly, governed by the `squid.conf` file. The "`refresh_pattern`" directives include a number of defaults that can be used to calculate an amount of time that the proxy should retain each object as fresh. Since the minimum "fresh" time in the default file is zero minutes, nothing will cache unless the server specifically wants that behavior for the object. Since `pastebin.com` does not include the necessary headers to direct cache behavior, the default of zero is used, meaning we're out of luck for this object.

References:
`http://for572.com/45qt2`

https://t.me/learningnets

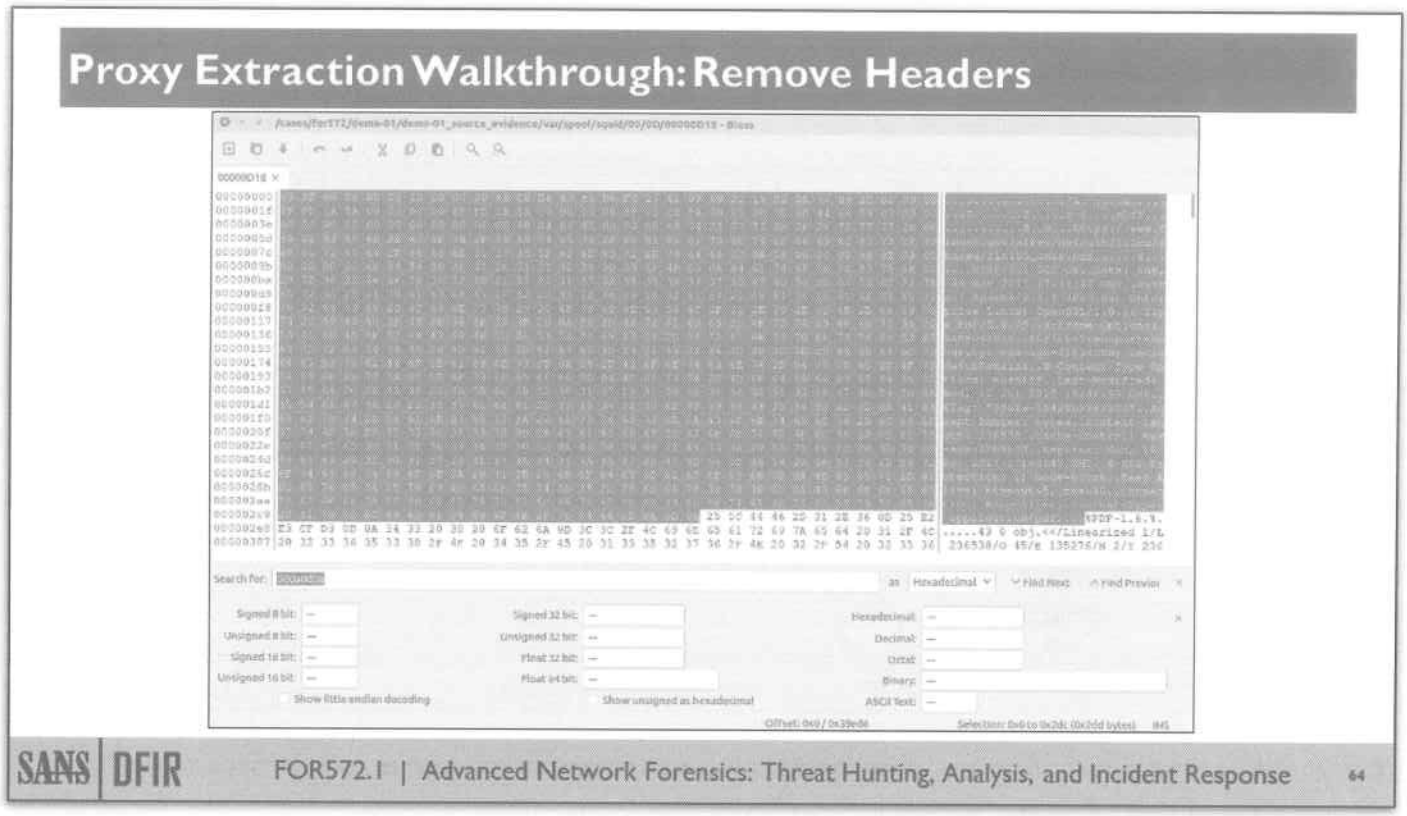
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



However, maybe we'd like to take a look at something that was retained in the cache. Let's consider that PDF file that was retrieved toward the end of our subject employee's web surfing session. Using the blunt but effective "railgrep" technique, we quickly identify that the proxy file `00000D18` appears to contain the filename of interest. Examining the headers seems to support this theory based on the URL and the "`Content-Type:`" header. As an aside, you can also see the "`Cache-Control:`" and "`Expires:`" headers, which confirm why the proxy retained this object.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



In this screenshot, we are using the "bless" hex editor that is available on the Linux SIFT Workstation. Just look for the `0x0D0A0D0A` separator sequence, and remove it with all bytes before it. Save the results to a new file named `/tmp/proxy_cache_extraction`.

```
20 4A 75 6C 20 32 30 31 37 20 31 35 3A 34 36 3A 35 33 20 47 4D 54 0D 0A  Wed, 12 Jul 2017 15:46:38 GMT..
33 39 62 66 61 2D 35 35 34 32 30 62 39 39 38 33 35 34 30 22 0D 0A 41 63  ETag: "39bfa-55420b9983540"..Ac
6B 67 65 73 3A 20 62 79 74 65 73 0D 0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E  cept-Ranges: bytes..Content-Len
36 35 33 38 0D 0A 43 61 63 68 65 2D 43 6F 6E 74 72 6F 6C 3A 20 6D 61 78  gth: 236538..Cache-Control: max
30 39 36 30 30 0D 0A 45 78 70 69 72 65 73 3A 20 53 75 6E 2C 20 31 30 20  -age=1209600..Expires: Sun, 10
37 20 31 37 3A 34 34 3A 34 37 20 47 4D 54 0D 0A 58 2D 58 53 53 2D 50 72  Dec 2017 17:44:47 GMT..X-XSS-Pr
6E 3A 20 31 3B 20 6D 6F 64 65 3D 62 6C 6F 63 6B 0D 0A 4B 65 65 70 2D 41  otection: 1; mode=block..Keep-A
69 6D 65 6F 75 74 3D 35 2C 20 6D 61 78 3D 39 30 0D 0A 43 6F 6E 6E 65 63  live: timeout=5, max=90..Connec
65 65 70 2D 41 6C 69 76 65 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A  tion: Keep-Alive..Content-Type:
61 74 69 6F 6E 2F 70 64 66 0D 0A 0D 0A 25 50 44 46 2D 31 2E 36 0D 25 E2  application/pdf....%PDF-1.6.%.
20 30 20 6F 62 6A 0D 3C 3C 2F 4C 69 6E 65 61 72 69 7A 65 64 20 31 2F 4C  .....43 0 obj.<</Linearized 1/L
2F 4F 20 34 35 2F 45 20 31 33 35 32 37 36 2F 4E 20 32 2F 54 20 32 33 36  236538/O 45/E 135276/N 2/T 236
```

```
$ ls -l /tmp/proxy_extraction
-rw-rw-r-- 1 sansforensics sansforensics 236538 Nov 26 22:45 /tmp/proxy_extraction

$ md5sum /tmp/proxy_extraction
c5f59208d73648fbc9ae9503c2d3c46c  /tmp/proxy_extraction
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Extraction Walkthrough: Content Type

- Use the `file` command to test the file contents

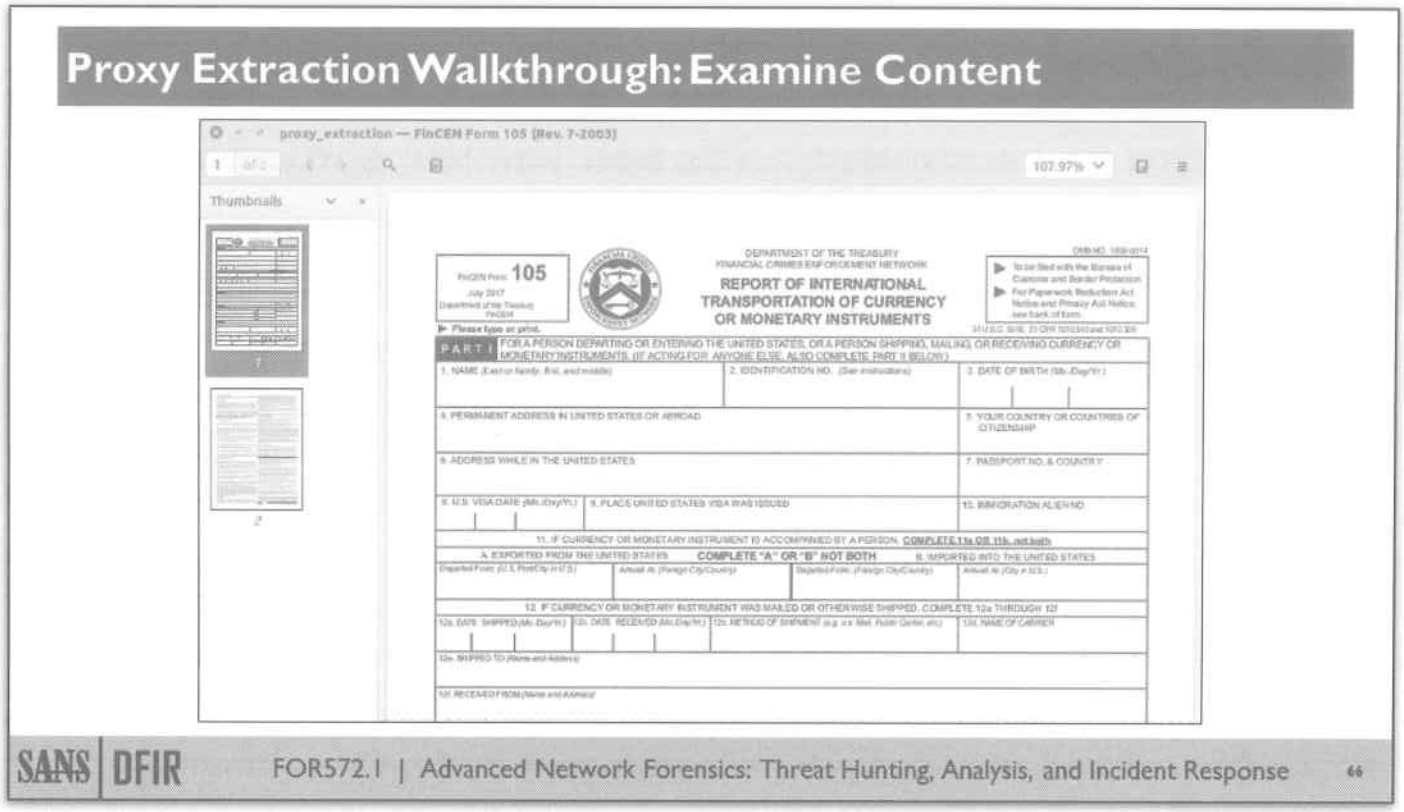
```
$ file /tmp/proxy_extraction
/tmp/proxy_extraction: PDF document, version 1.6
```

  - Checks first few bytes of a file against known file types
  - A quick and reasonable cross-check against HTTP headers

The built-in `file` command is extremely helpful in confirming what we have carved from the cache object. In this case, it's apparently a native PDF file, meaning no further decapsulation or decoding should be required. However, if it were gzip-compressed or otherwise compressed/encoded, we'd need to decompress or otherwise decode as required. This may even need several iterations of the decode/decompress process.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



To view the content, just open in your preferred PDF software—in this case the "`evince`" utility installed on the SIFT Workstation. In a real-world scenario, you'd want to approach this part of the task quite cautiously, though. It's not safe practice to open untrusted, un-validated file content, as it may contain a malicious payload. This is part of why it's a good idea to have a "throwaway" system where such research can be conducted without potentially damaging the forensic analysis platform.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Proxy Extraction Optional Homework

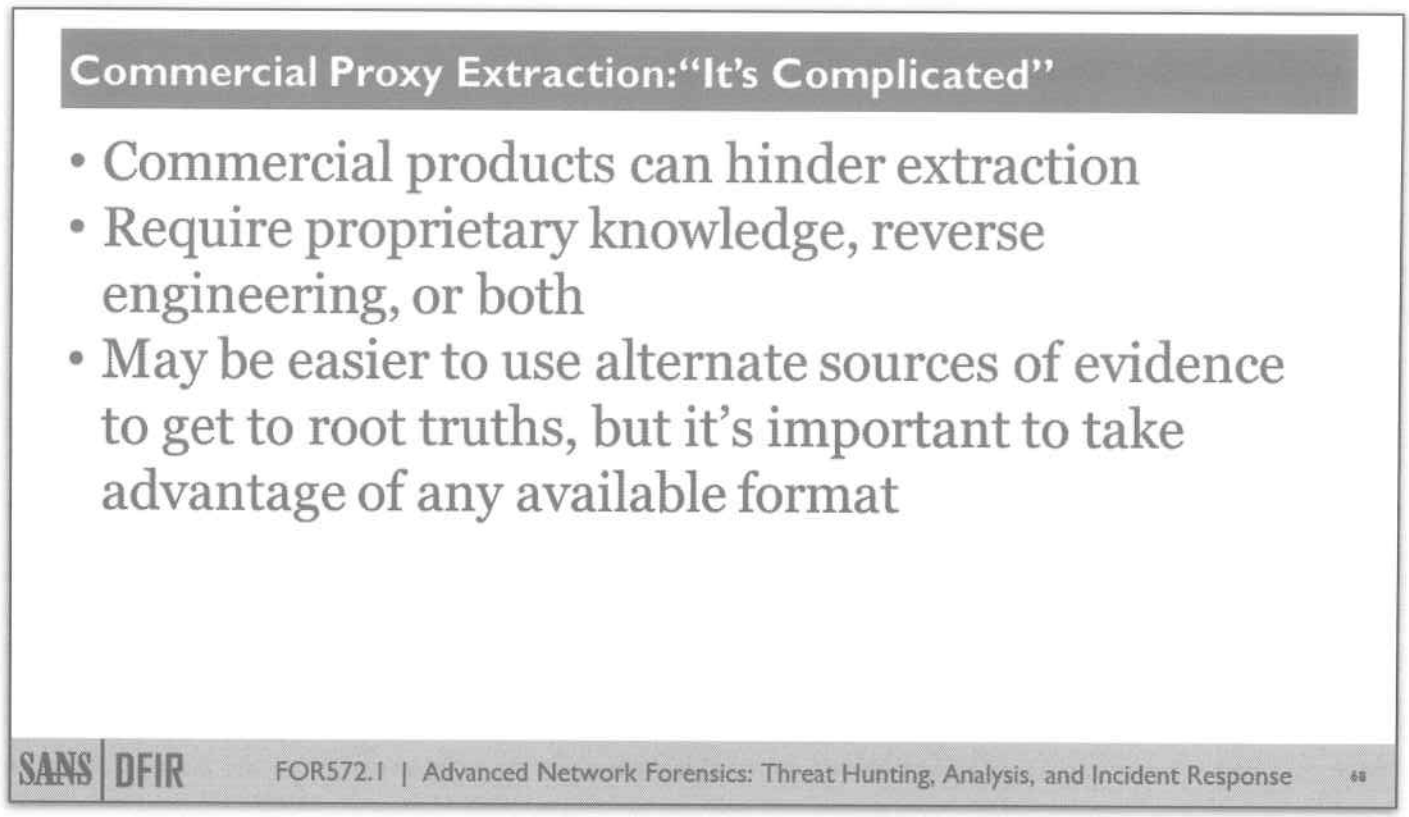
- Were there any other potential exfiltration events?
- Was there any other suspicious activity leading up to the events we explored here?
- Any other useful Artifacts of Communication?
- What other evidence would you want to better understand and explain what happened?

We've come quite a long way in this walkthrough. However, we must acknowledge that we still don't know what was uploaded to `pastebin.com` and most likely won't identify that from the proxy evidence alone.

However, we haven't fully exhausted the evidence in this walkthrough. Consider reviewing the full set of available log and cache data to gain a more comprehensive understanding of the situation. The questions above are a good guide, but as you likely know already, one of the forensicator's biggest challenges is to know how to effectively manage time against the priorities set forth for the case or incident. Aimlessly reviewing data is often referred to as the "rabbit hole" and can be an exhausting process without practical results.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

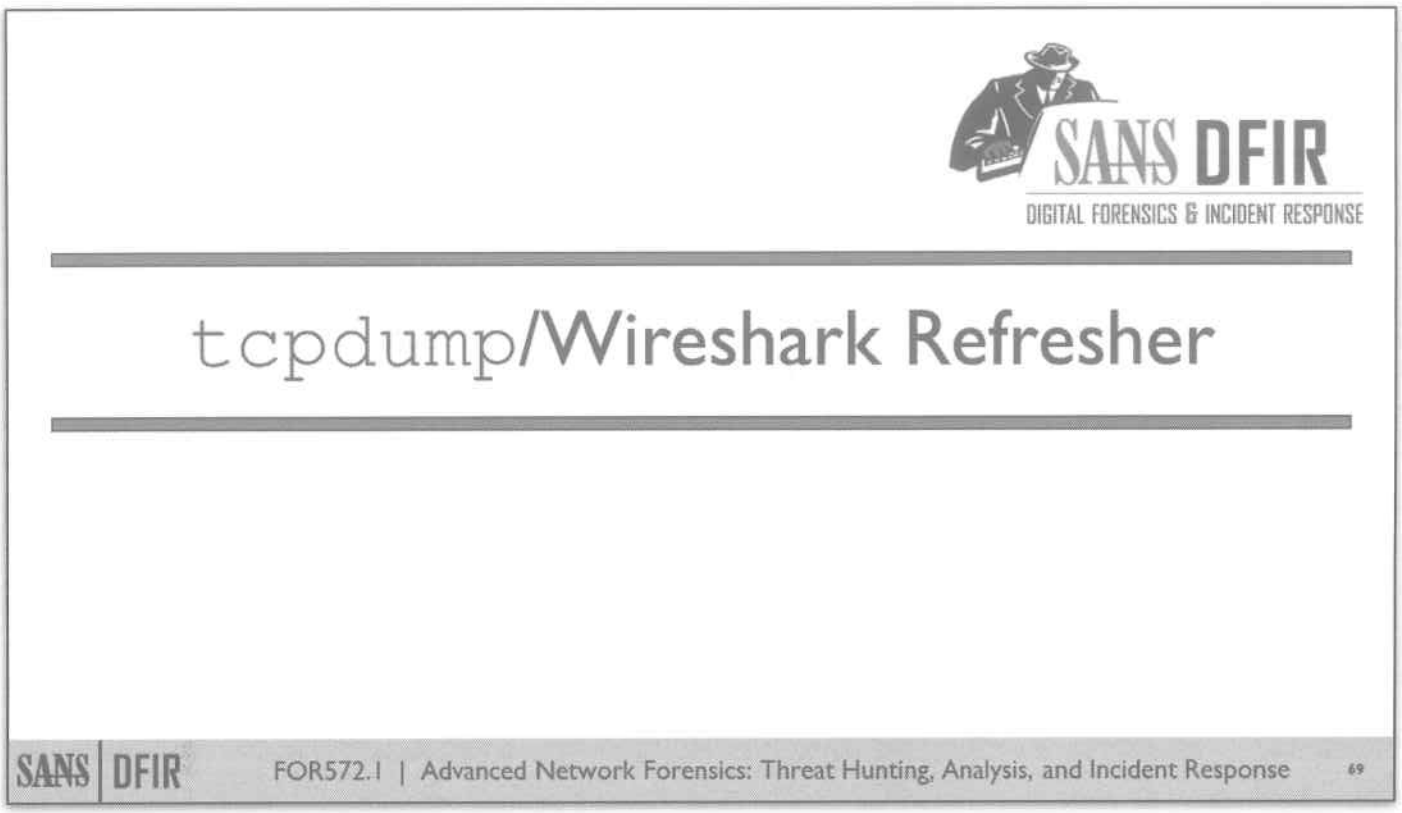


Although extracting content from the cache objects in Squid can be accomplished—or automated—pretty easily, commercial products can make this difficult. Where commercial log analysis is complex, commercial cache extraction can be all but impossible.

In those cases, we may find it easier to just skip the complex and undocumented cache format and go straight to the root data that crossed the network. To do this, we need to understand how network data is stored, analyzed, and replayed.

https://t.me/learningnets

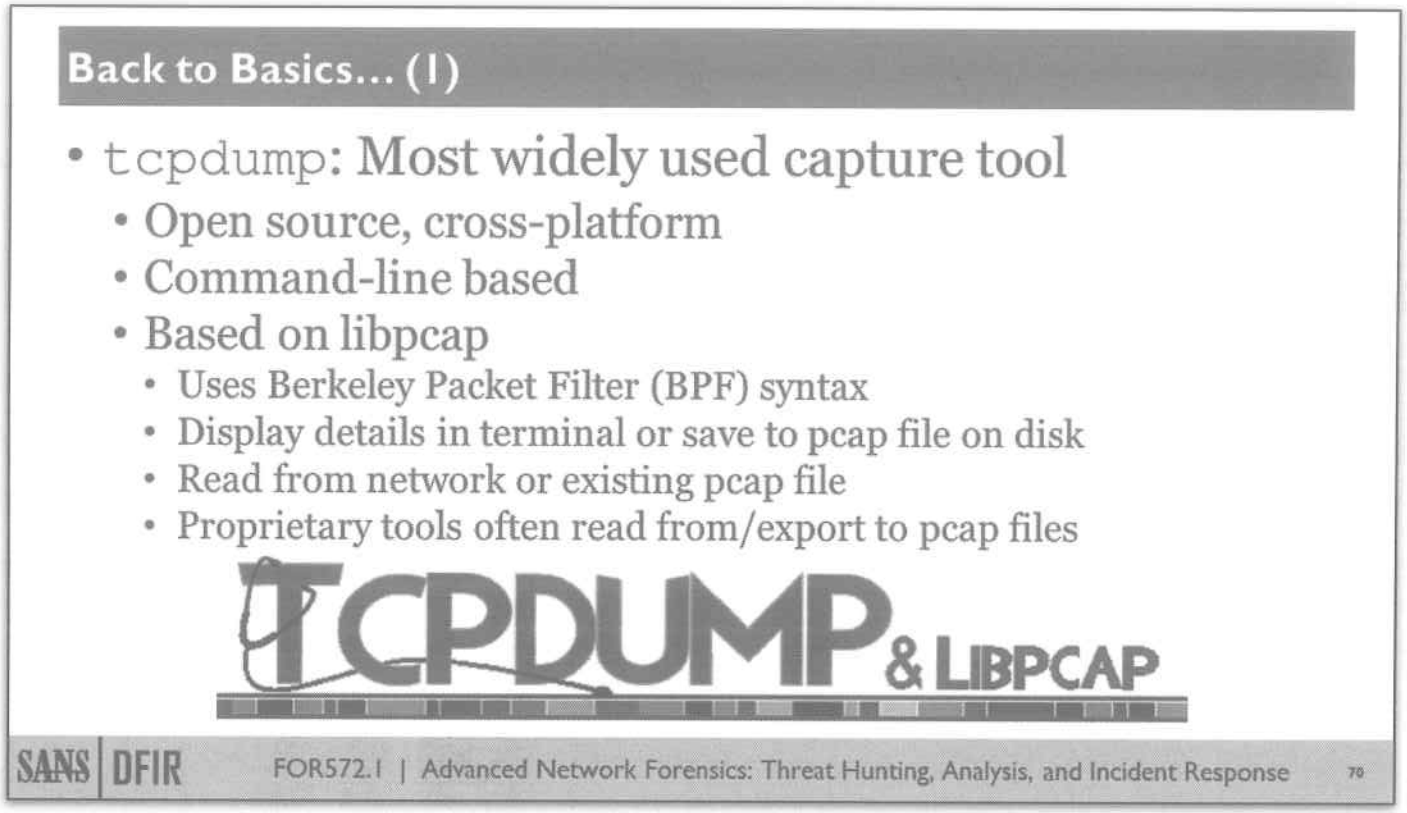
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Probably the most fundamental tool that a network forensic practitioner needs to know is tcpdump. tcpdump is primarily used to capture packets from the network medium, after which it will either display a definable amount of information about each packet or write the entire packet (or a portion of it!) to disk. It is a command-line tool, making it suitable for a wide variety of applications—local and remote, human-driven, and automated. Although tcpdump was originally built for the *NIX world, it has been ported to other operating systems and is a staple for Windows and *NIX network administrators and investigators alike.

The core features tcpdump provides are thanks to its underlying use of the libpcap library. This powerful library exposes a number of key capabilities. Some of the more important of these include:

- The Berkeley Packet Filter (BPF) language, a basic filtering language that is used to limit the data captured from the network. BPF uses a set of defined "primitives" to determine the data to be acquired, and these primitives can be logically combined to form elaborate filters.
- Display packet details on screen or save packets to a "pcap" file on disk. The pcap format is a standardized format to store network packet data on disk. Almost all of the data we'll be examining in this class has been previously created and distributed to you in pcap files.
- Read network data from a live network or from a pcap file. In a reverse approach to the save-to-pcap feature, libpcap transparently provides access to saved data just as if it were acquired from a live network, as well. From a forensic perspective, these two awesome features mean that we can acquire data in one step, then use a wide variety of tools to later analyze perfect working copies of the source data, even years later.
- Runtime specification of how much data to capture. Although the idea of capturing every byte of traffic is gaining traction in many network environments, any number of legal or practical limitations may prevent us from doing so. In these cases, libpcap allows the operator to specify a "snaplen", or number of bytes to capture from each packet. For example, this functionality would allow us to capture only the packet headers.

Because these features are available from the libpcap *library*, they can also be incorporated into other tools. In fact, the overwhelming majority of network analysis tools are based on libpcap, which means they have the full complement of these features available.

References:
`http://for572.com/7itb1`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Back to Basics... (2)

- Wireshark: GUI that decodes protocols
  - Powerful parsing for hundreds of protocols
  - Can add new protocols as required
  - Open source, cross-platform
  - Comes with `tshark`, a console-mode equivalent

Although tcpdump is a great console packet capture application, it doesn't provide many features in the area of deep packet analysis. Wireshark takes the reins at that point in many of our processes.

Wireshark is a graphical application that allows us to visually examine the contents of network traffic and perform some very useful high-level analysis. Although Wireshark can capture packets, the analyst must consider all the extra functions going on in the background before looking to it as a large-scale capture solution. Generally, Wireshark is fine for captures in a lab environment, but when things really get cranking, you'll certainly want to migrate to a more native tcpdump-type solution.

Wireshark comes preloaded with several hundred protocol parsers, which is a HUGE help during the analysis process. These parsers (or decoders) allow us to explore the contents of a given packet, and see the meaning of each field and its associated data. So rather than immersing oneself in the ever fun-to-read RFC for a given protocol, we can use the power of Wireshark to get us productive much, much faster. Because some protocols may not have an RFC due to their proprietary nature or restrictions, the Wireshark community often contributes parsers to the code base so all can benefit. Similarly, you can also create your own protocol decoders, and consider releasing them through the Wireshark community.

As with tcpdump, Wireshark is based on libpcap, so we gain all the great benefits discussed previously. It's also cross-platform (Linux, macOS, Windows, and many less-common operating systems).

Wireshark is also distributed with "`tshark`", which is a console application that performs all the same features as the GUI variant. This proves extremely helpful because we can build an analytic plan in the GUI, then transfer that to tshark at the console to scale our analysis across a wider corpus of network data.

References:
`http://for572.com/mhczk`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

PCAP File Format (1)

- Magic: `0xa1b2c3d4`
  - Or: `0xd4c3b2a1`
- Version: 2.4
  - For libpcap >=1.1.1
- TZ always UTC = 0
- Accuracy always = 0
- Snapshot length (max packet size)
- Many link types

pcap file header

	0	1	2	3
0x00	Magic Number			
0x04	Major version		Minor version	
0x08	Time zone offset			
0x0C	Time stamp accuracy			
0x10	Snapshot length			
0x14	Link-layer header type			

We briefly mentioned the pcap file format before, but let's take a brief look at what the file itself actually contains. Although the obvious answer is "network packets!", knowing what metadata is stored in each pcap file is also critical when using network-based data as evidence.

- First, as you should already be familiar with, there is a sequence of magic bytes to start the file off. This is a unique series of bytes that indicates the file contains pcap data. Of particular note, though, is that endianness of a system can mangle this value. In a nutshell, some operating systems read values from right to left, and others read from left to right. We won't dig too deeply into endianness in this class, but it's important to recognize that the actual byte sequence will differ if a capture file that was created on a Linux system is examined on a Windows system.
- The version number will occasionally change, but may be useful in researching unusual behavior that can be associated with a particular version of libpcap.
- As in all forensic processes, time zone is a critical value to consider. Fortunately, the libpcap developers have shared in the maddening task that is reconciling timestamps from systems in varying time zones. To rectify this, the standard is simply to store everything in UTC! (Just as it should be!!) So, this four-byte offset will always be zero. The accuracy field is also set to zero.
- You recall that one helpful libpcap feature is that we can specify the number of bytes in each packet that should be acquired. Let's say an analyst was reviewing an IPV4 pcap file in which all packets were no greater than 40 bytes. She would want to know if the packets were truncated by some network-based error or if tcpdump was run with a 40-byte snaplen. This field would provide a definitive answer to that question. It is important to note that the default snaplen varies widely on different platforms and distributions. This demonstrates why it is important to fully test and validate tools before using them!
- The link type is a field that indicates the type of media from which the packets were acquired. There are many, many values, and the man page for `pcap-linktype` contains a full and annotated list of acceptable values and their meanings.

References:
`http://for572.com/jlvcz`
pcap-savefile(5) ; pcap-linktype(7)

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

PCAP File Format (2)

pcap packet/frame header

	0	1	2	3
0x00	Time stamp, seconds value			
0x04	Time stamp, microseconds value			
0x08	Length of captured packet			
0x0C	Un-truncated length of packet data			

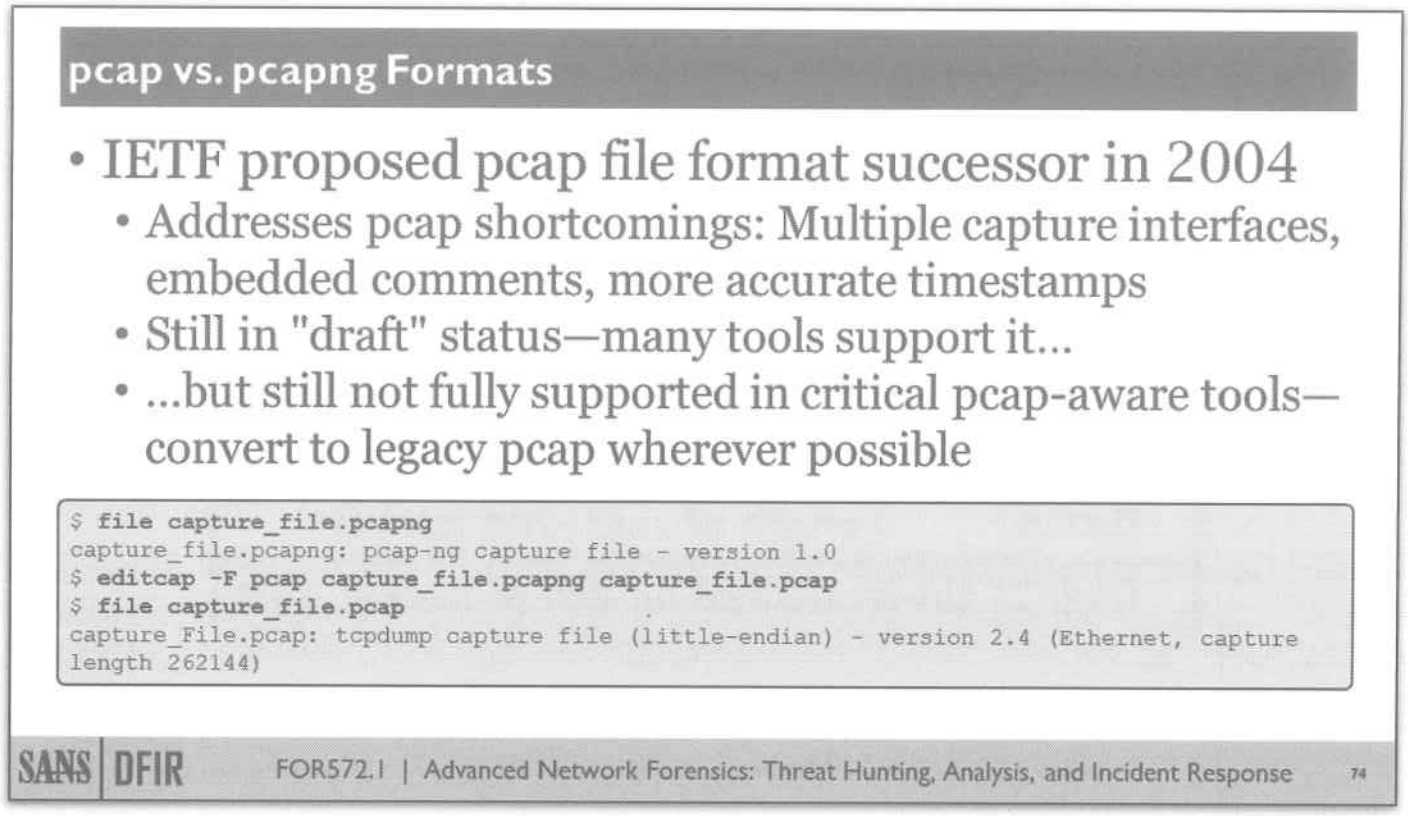
After the pcap file header are zero or more packets. The pcap format prepends a set of metadata to each stored packet as well. These four, four-byte fields contain useful data for an analyst:

- The time each packet was captured is written to the pcap file in two segments—the number of seconds since the UNIX epoch (Jan 1, 1970, 00:00:00 UTC), and the number of microseconds after that second-resolution timestamp.
- The number of bytes captured from the network.
- The number of bytes that were present on the network before any snaplen-based truncation.

References:
pcap-savefile(5)

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Because the pcap file format has its shortcomings, the Internet Engineering Task Force devised a successor format named "pcapng", short for "pcap next generation". The format is completely different than the legacy format[1], built around an extensible, tag-based structure. The primary benefits generally touted[2] include:

- Ability to store captures from multiple interfaces (and interface types) in the same file
- Better timestamp resolution options
- Expanded metadata fields including comments, statistics, DNS activity, and more

The new pcapng file format is usable in numerous tools, but the support is not universal. Even many implementations of `tcpdump` cannot fully use the new format, and the error messages given are seldom useful in tracking down the problem[3].

Instead, it is strongly recommended to convert any pcapng files to legacy pcap format before handling them with an untested tool or process. A number of resources can accomplish this, but the easiest is built into the `editcap` utility, which is a command-line component of the Wireshark suite.

```
$ file capture_file.pcapng
capture_file.pcapng: pcap-ng capture file - version 1.0
$ editcap -F pcap capture_file.pcapng capture_file.pcap
$ file capture_file.pcap
capture_File.pcap: tcpdump capture file (little-endian) - version 2.4
  (Ethernet, capture length 262144)
```

References:

[1] `http://for572.com/8-qoy`
[2] `http://for572.com/gie9l`
[3] `http://for572.com/fxa32`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Fundamental tcpdump Usage

- Can still lose packets (CPU, storage, etc. limitations)
- BPF allows capture minimization

```
$ sudo tcpdump -n -s 0 -i ens33 -w out.pcap 'host 1.2.3.4 and port 80'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 65535 bytes
```

- Reference tcpdump(1) and pcap-filter(7) man pages and FOR572 Linux Shell Survival Guide handout
  - `-n`: Don't do DNS lookup for each IP
  - `-s 0`: Snapshot length of zero (capture all bytes)
  - `-i ens33`: Capture from "`ens33`" network interface device
  - `-w out.pcap`: Write to output file
  - `'host 1.2.3.4 and port 80'`: BPF to use

Now that we've reacquainted ourselves with the venerable pcap file format, we'll take a brief look at how to use tcpdump to capture packets from a network. A basic command is listed on the slide, and the meaning for each of the options is below. As with any *NIX command, the man pages are your friends. Consult them often!

```
$ sudo tcpdump -n -s 0 -i ens33 -w out.pcap ⏎
  'host 1.2.3.4 and port 80'
```

`-n`	Don't look up DNS or protocol names. Important to minimize network traffic generation and prevent OPSEC problems.
`-s 0`	Capture all bytes from packets. Note that default snaplen differs between tcpdump versions and operating system platforms.
`-i ens33`	Capture from device "ens33". Use `-D` for a list of valid devices.
`-w out.pcap`	Write data to a local file named "`out.pcap`", rather than displaying details in terminal.
`'host 1.2.3.4 and port 80'`	BPF syntax for Layer 3 host address and Layer 4 port specification.

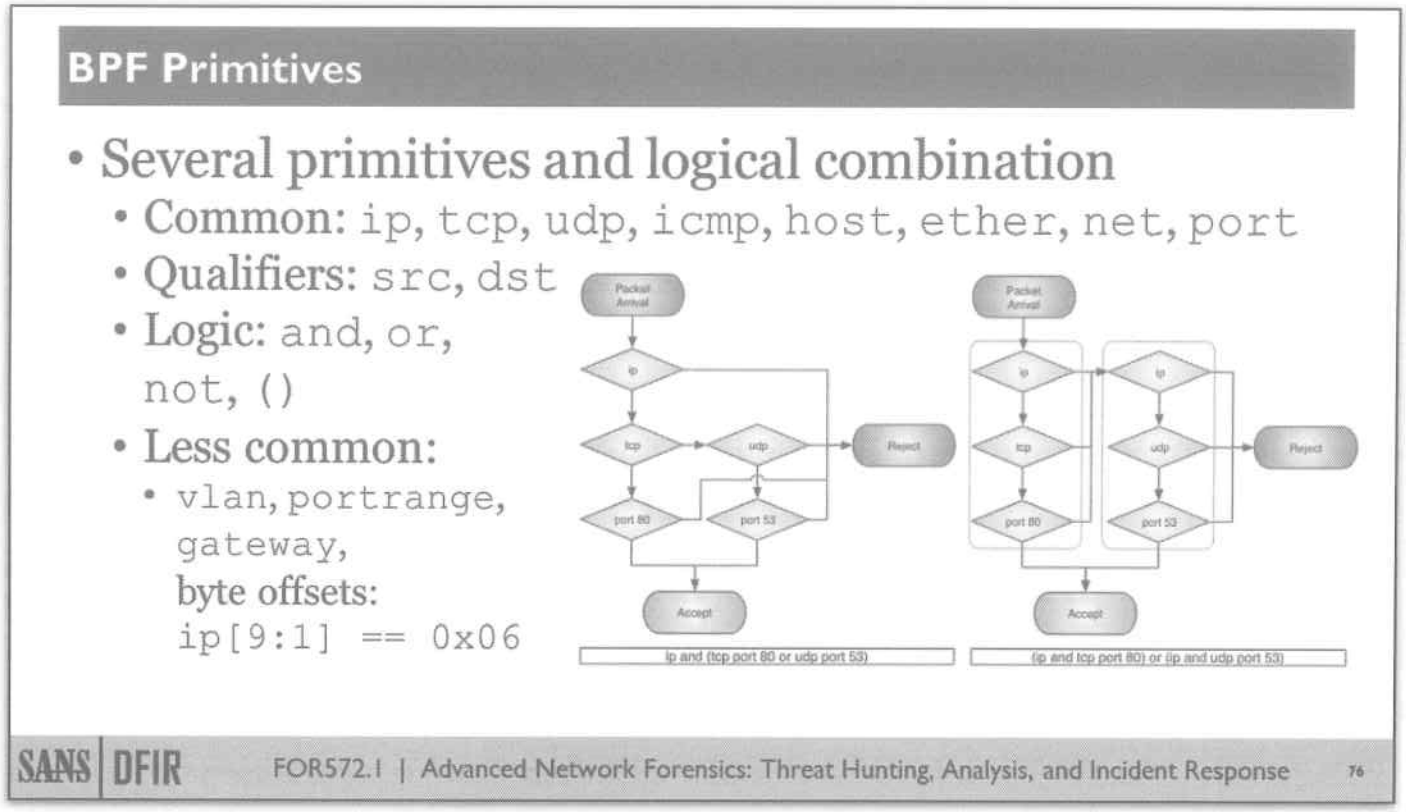
Note that although this command specified a snapshot length of zero, there are a number of situations that could lead to partial or complete loss of some packets. The system's CPU and storage are the largest contributors here. A CPU-bound capture system will simply drop traffic if it can't keep up. Similarly, if writing the packets to slower storage media, the network traffic could conceivably exceed the rate at which the network data can be written out. Be sure to test any capture solution at speed and scale before relying on it for operational acquisitions.

We'll examine the BPF in more detail next, but it is always best practice to enclose the BPF in single or double quotes to prevent the shell from interpreting spaces, parentheses and other reserved characters.

Hint: Remember that running tcpdump in "promiscuous" mode requires root or administrative privileges. Promiscuous mode allows the network interface device to acquire packets that are not destined for one of its own hardware addresses.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



At its core, the BPF syntax provides a way to designate which traffic is or is not "interesting", and therefore worthy of the CPU cycles required to handle it. Because they operate at such a low layer within the operating system, they offer the most efficient way to limit which packets are "allowed through" to the next steps in the process—whether displayed to the console, written to disk, or passed off to other utilities for processing.

The BPF structure itself is rather basic but still extremely powerful. Remember that nearly all libpcap-based software exposes the BPF functionality, so knowing how to create proper filters is a very valuable skill. Although there are many different primitives that we can use, we'll focus on some of the more common ones here and leave the rest as a research topic for you to take on.

Perhaps the easiest to understand are the primitives that match against a given protocol. Simply, these are "`ip`", "`tcp`", "`udp`", "`icmp`", or one of a handful of others. If the protocol matches, it is considered "interesting" and processing continues. If it does not, the packet still goes on its way, but the libpcap process stops processing the packet, waiting for the next one in the queue.

The "`host`" primitive limits on the Layer 3 IP address. By specifying "`host 192.168.75.104`", only packets to or from the specified IP will match to be passed along. Similarly, the "`ether`" primitive allows us to limit by address, but at Layer 2—the MAC address. A BPF of "`ether 00:10:FA:3b:45:c3`" will only catch traffic involving that particular device. We can implement a "`net`" filter to expand the idea of an IP address to the network level—simply specify a network in CIDR notation, such as "`net 10.54.115.0/24`" and the filter will match accordingly. The last of the most common BPF primitives is "`port`", which, as you should have guessed, will match based on the Layer 4 port for either end of the connection. Remember that when using "`port`" alone, the BPF engine will not take protocol into account—so "`port 53`" will catch both TCP and UDP traffic on port 53.

As mentioned above with the "`host`", "`ether`", and "`port`", the directionality of the packet is not taken into consideration—so we must look at the directionality qualifiers "`src`" and "`dst`". For unknown reasons, the directionality qualifier precedes each of the object primitives *except* for "`ether`". So, you'll use "`src host`", "`src port`", and "`src net`", but when examining MAC addresses, the filter is "`ether src`". Additionally,

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

keep in mind that "`src host 172.30.33.120`" will only catch one side of the connection—those packets generated by a system with the specified IP address. The easiest way to remember this is that the BPF is applied to each and every packet that libpcap inspects. The filter process examines each field as we see in the protocol header and makes the interesting decision based on that one single packet. As soon as the interesting decision is made, the engine forgets anything it's seen so far and re-evaluates the next packet from the start of the BPF.

Because the BPF primitives are rather basic, the developers added the ability to combine them with standard and/or/not syntax. This gives us the ability to craft elaborate logical evaluations all within the BPF itself. Order of operations can be enforced with parentheses—but be careful! By default, the parentheses are special characters in UNIX shells, and space characters can also trip up a well-intentioned filter statement. This is why we consider it best practice to enclose all BPF statements in single or double quotes. An example of a more elaborate BPF would be:

```
'tcp and (port 80 or port 443 or port 8080) and (not dst host 192.168.65.1)'
```

This would catch any TCP packets sent to or from ports 80, 443, or 8080, but not those sent to the IP address 192.168.65.1. This is also a good point to remind you that testing BPFs in the lab and under network load is critical! The more elaborate you get with a BPF, the greater the chance of a syntax error. Similarly, the more complex the BPF statement, the harder it is for a human to understand and explain what is going on. Although tcpdump and other BPF-capable tools will optimize filters before using them, consider that a human's grasp of the process is just as important—if not more so! (For instance, the two flowcharts on the slide detail computationally identical BPFs, the logic path does not.)

Again, these are just a few of the more common and helpful BPF primitives—there are many others that you may find helpful in this class's exercises and your daily jobs. A few of the cleverer of the lot are below, but you should absolutely take a look at the man pages to become better acquainted with the full menu you have available.

- `vlan`: When inspecting 802.1Q "trunk" traffic, limit based on the numerical VLAN identifier. This can get tricky, as encapsulation can change offsets for the overall BPF expression.
- `portrange`: Instead of just a single port, catch traffic that matches based on a contiguous range of port numbers. Can be used with "src" and "dst", just like the bare "port" primitive.
- `gateway`: Match on traffic that contains the specified destination MAC (Layer 2) address, but a Layer 3 address that does not belong to that MAC address. Could be useful to find packets sent to rogue MITM gateways.
- Byte offsets: Perhaps the most cryptic yet powerful filter gets down to the byte—or even bit—level. Using this notation, you can match based on the decimal or hex value for a particular byte or series of bytes within the packet. The offset is zero-based and calculated from the first byte of the IP header. Beyond simply matching at the byte level, though, you can also use typical bitmasking syntax to isolate the "zero" or "one" value for a single bit within the packet. Although this may seem rather elaborate for a BPF's interesting decision, remember that these are applied very close to the kernel, and are therefore very fast and efficient. If this is the best, most unique way to isolate interesting traffic before handing it off to a much more CPU-intensive process, then we should absolutely implement the filter in a BPF!

References:

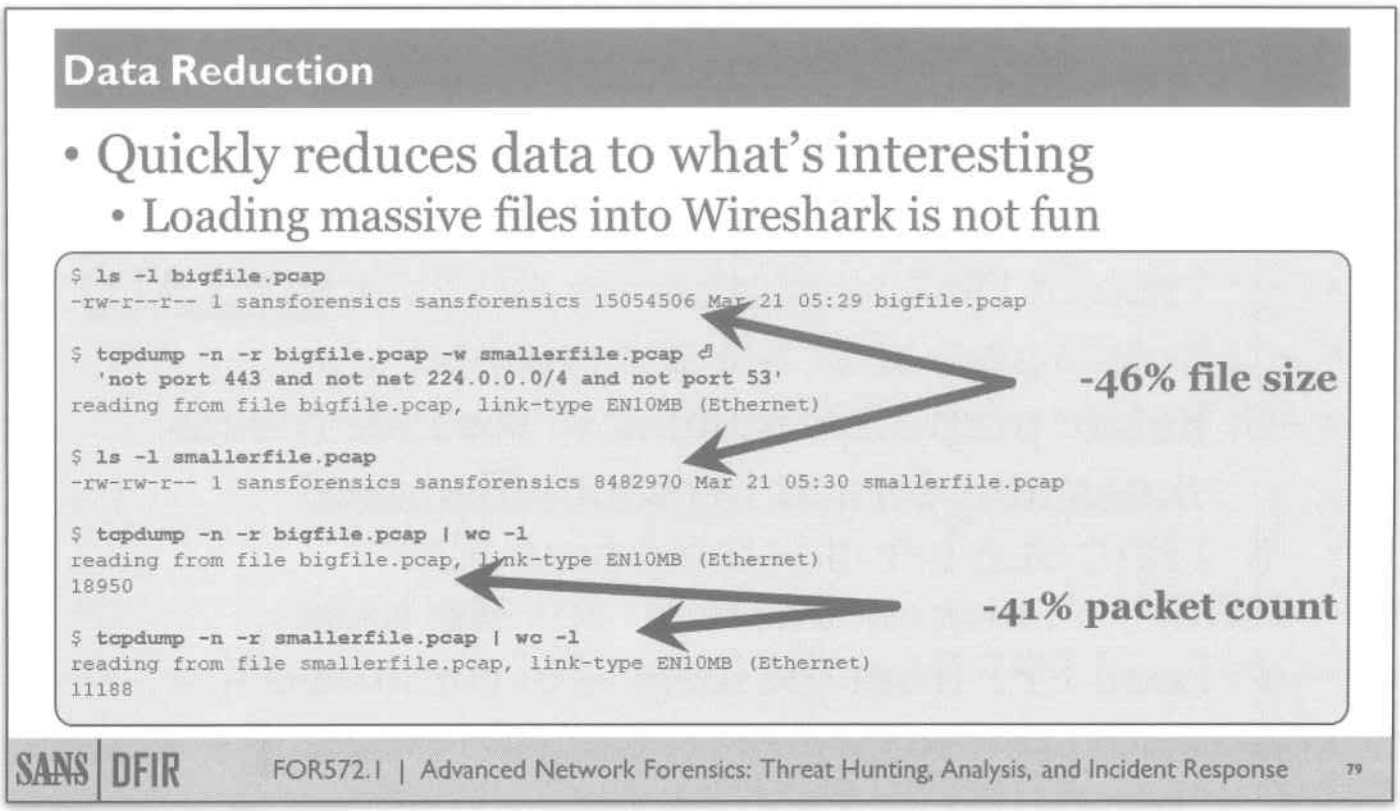
`http://for572.com/3f0zg`

pcap-filter(7)

https://t.me/learningnets

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



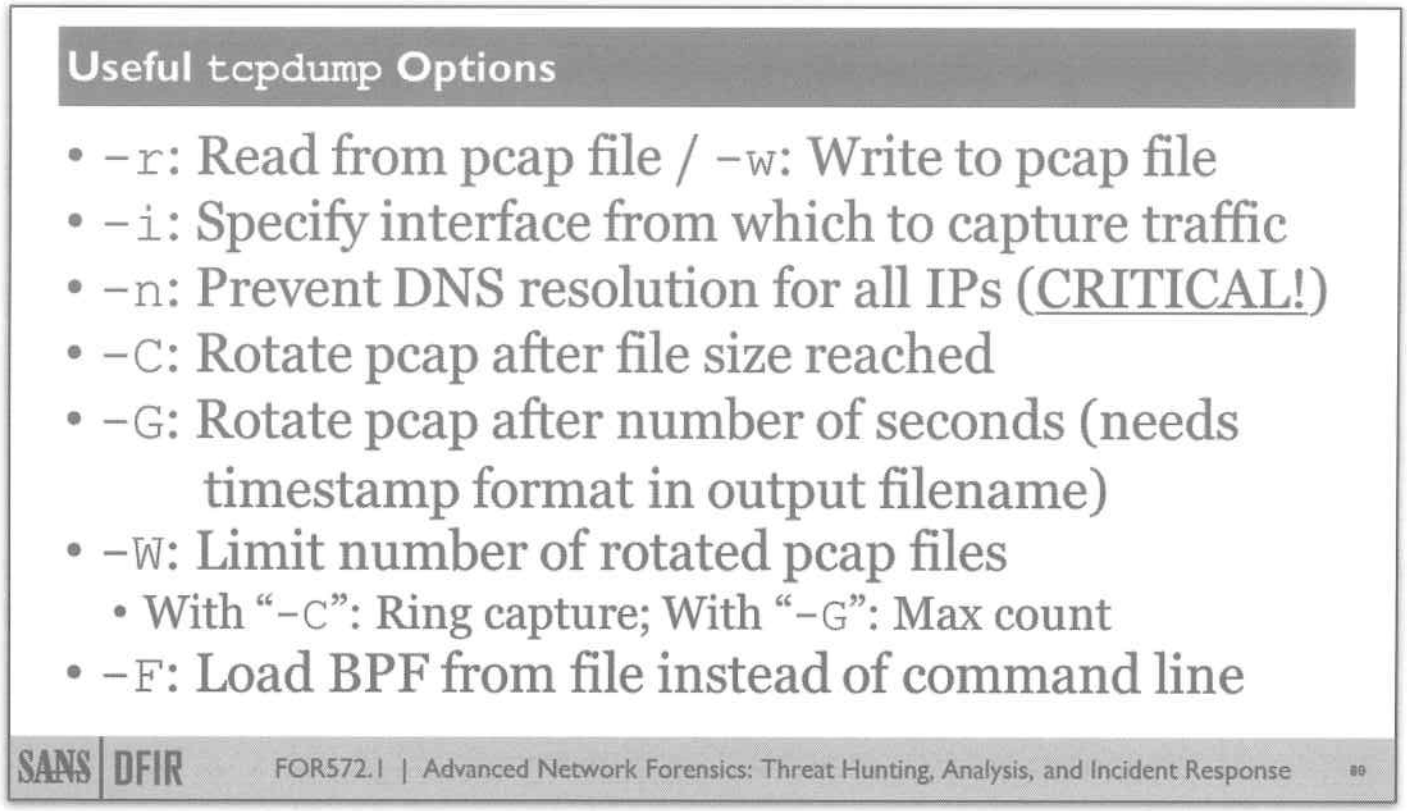
Another key use of tcpdump is to reduce the data in an existing pcap file through the use of a BPF filter. This will allow us to keep the original source data intact while pulling just what is relevant, or eliminating what is not relevant.

If an analyst plans to load data to Wireshark, minimization is even more important. As you'll see in this class, loading large pcap files to Wireshark is very slow—any portion of data we can eliminate from the input file will help to cut down on the load time. Because we are not modifying the source file, we can always go back to review additional data if required.

In this case, we are removing some traffic that might be reflected in other log files within the environment, generally not useful in Wireshark. (Port 443 (likely HTTPS), CIDR block 224.0.0.0/4 (multicast traffic), and port 53 (likely DNS).) This step reduces our pcap file to just traffic that's not being reflected elsewhere, reducing the size of the pcap by 46% and the number of packets by 41%.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

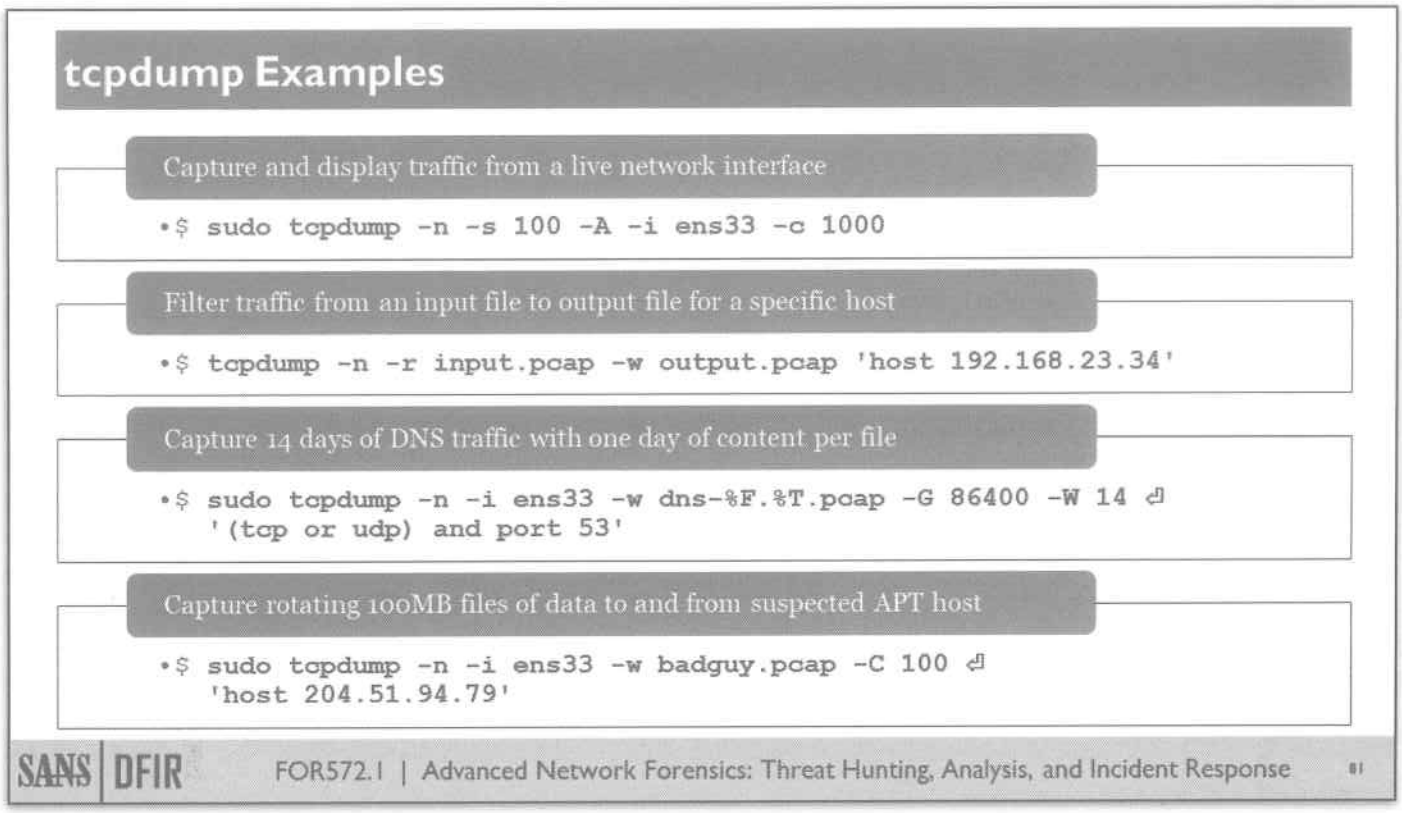


Although the number of `tcpdump` tricks could take up a blog entry per day for a year, here are a few that can be particularly helpful in supporting forensic and investigative processes.

`-r`	Read packet data from an existing pcap file rather than a live network interface.
`-w`	Write packets matching the supplied BPF to a pcap file rather than the screen.
`-i`	Specify interface from which to capture traffic. May be a specific interface name such as "`venet0`", "`enp0s3`", or "`en1`", or the special "`any`" pseudo-interface to simultaneously capture from all available interfaces on platforms that support it.
`-n`	Prevent any DNS resolution. Without this, `tcpdump` will perform a DNS lookup for each IP address it observes. This is undesirable for several reasons. It adds a significant amount of network traffic into the environment, causing delays in processing and storing the packets. It also runs the risk of alerting an attacker that you're looking at (or at least capturing) their traffic—a troubling occurrence! The "`-n`" option should become automatic whenever you use `tcpdump`.
`-C`	When used with "`-w`" to write packet data to a file instead of displaying data in the console, this option will create a new pcap file after the output reaches the specified size (in ~megabytes).
`-G`	Similar to "`-C`", but will rotate file after the specified number of seconds, i.e. "`-G 86400`" will rotate daily. However, this option requires a specially formatted filename, using format string placeholders compatible with `strftime(3)`. For example, an output filename of "`outfile_%F.%T.pcap`" will result in filenames such as "`outfile_2016-07-22.12:25:13.pcap`", with the time representing the start time of each file created. Caution! Using a filename without the format string will result in a continually-overwritten output file!
`-W`	When used with "`-w`" and either "`-C`" or "`-G`", only the specified number of files will be retained. With "`-C`", this will result in a rotating sequence of the requested number of files. With "`-G`", `tcpdump` will create the requested number of files and then exit.
`-F`	Instead of specifying the BPF on the command line, load it from a file. Particularly helpful when working with complex BPFs or sharing BPFs among a team. For example, you may want to keep your team's preferred BPFs in a git repository to ensure everyone is using the correct ones.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Four separate examples of common `tcpdump` usage are provided above.

The first command monitors the network interface identified as `ens33` and outputs the contents of the first 1,000 packets to the console. The command uses the following options:

`-n`	Do not resolve IP addresses to hostnames or ports to protocols. This option significantly speeds up `tcpdump`.
`-s 100`	Capture the first 100 bytes of each packet.
`-A`	Output the packet contents (application data) to the console in addition to the standard packet metadata.
`-i ens33`	Use `ens33` as the network interface for capture.
`-c 1000`	Capture the first 1,000 packets from the network interface.

The second example takes traffic from an input file and filters it by a specified IP address to an output file called "`output.pcap`". This command uses the following new options:

`-n`	Do not resolve IP addresses to hostnames or ports to protocols.
`-r input.pcap`	Use the `input.pcap` capture file for input.
`-w output.pcap`	Output the filtered traffic to the file "`output.pcap`".
`'host 192.168.23.34'`	This is the Berkeley Packet Filter (BPF) that is used by tcpdump to reduce packets to those to or from the specified IP address.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

The third command captures 14 days of DNS traffic and then exits. Each 24-hour period has a separate file (24 hours * 60 minutes * 60 seconds = 86400 seconds). The BPF limits output to just TCP/UDP traffic on port 53.

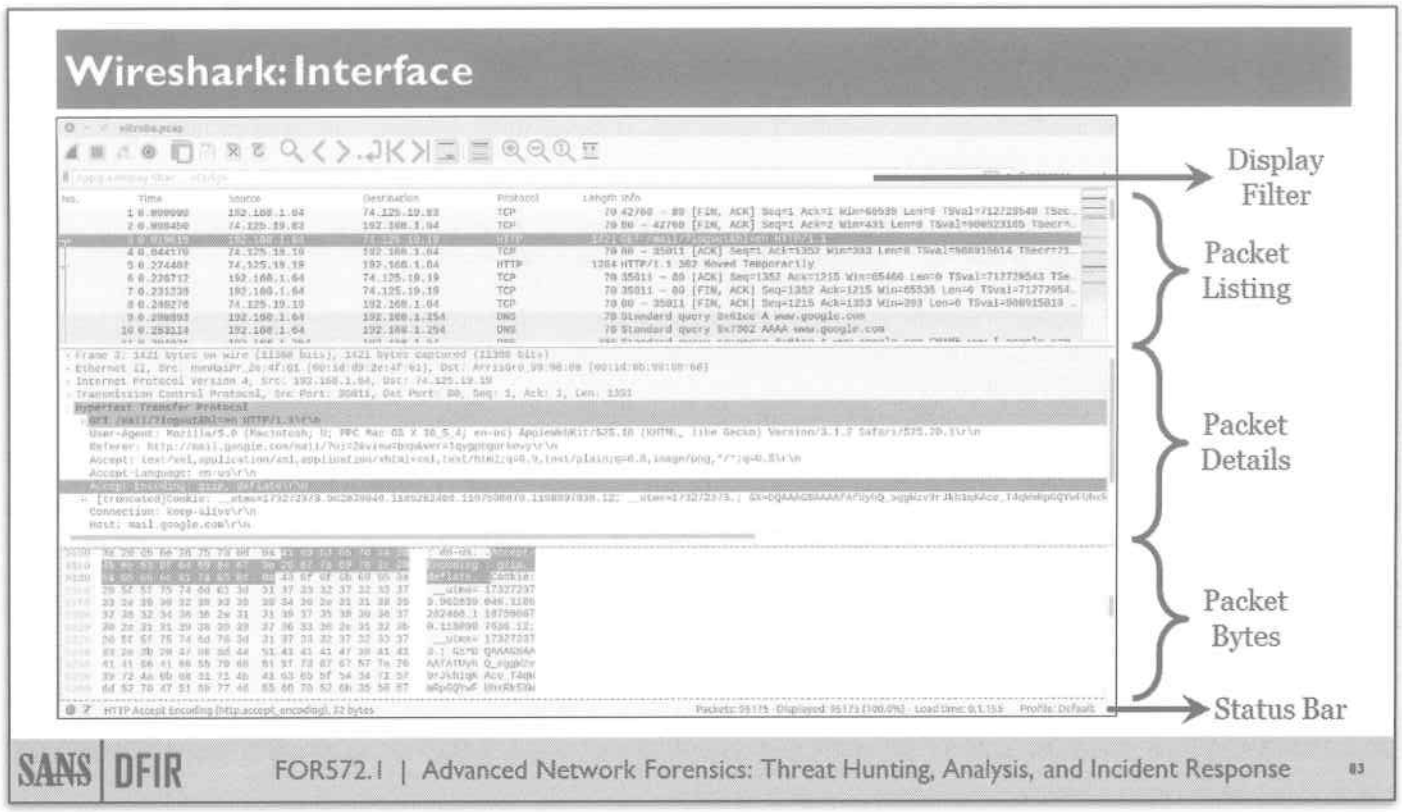
`-n`	Do not resolve IP addresses to hostnames or ports to protocols.
`-i ens33`	Use ens33 as the network interface for capture.
`-w dns-%F.%T.pcap`	Output the filtered traffic to files named with the date and time each one starts—e.g. `dns-2016-03-21.16:22:49.pcap`.
`-G 86400`	Rotate the output files after 86,400 seconds.
`-W 14`	Create 14 output files and then exit.
`'(tcp or udp) and port 53'`	BPF to limit collection to port 53 on either the TCP or UDP protocol.

Finally, the last example creates an infinite number of files of 100MB each. These files will contain traffic to or from a particular IP address of interest.

`-n`	Do not resolve IP addresses to hostnames or ports to protocols.
`-i ens33`	Use ens33 as the network interface for capture.
`-w badguy.pcap`	Output the filtered traffic to the file "`badguy.pcap`".
`-C 100`	Rotate the output files after 100MB (where 1MB == 1,000,000 bytes).
`'host 204.51.94.79'`	BPF to limit collection to a particularly shady character.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Now we'll take a look at Wireshark. This is the application where you'll be spending A LOT of time this week.

The GUI has three main sections annotated as the Packet Listing, Packet Details, and Packet Bytes panes.

The first pane, the Packet Listing, shows one row per packet. The user can configure which columns are displayed, as well as a number of display parameters. Of particular note is that the default "Time" value is the number of seconds and microseconds since the capture started. This format can be helpful in determining the intervals between packets, for example. However, recall the time-related metadata included in the pcap file format—one of the file header fields includes the UTC timestamp. With that basis, Wireshark gives the user an option to display each packet's timestamp in more human-readable formats. The "Info" column can also be helpful in that many of Wireshark's protocol decoders will provide a high-level summary of the decoded data in this view, making it easier for an analyst to visually survey a large list of packets for relevant or interesting entries.

The Packet Details pane is where we can see the real power of the protocol decoders at work. This pane contains a hierarchical list of every field within the packet—from the IP and TCP or UDP headers, all the way to the Layer 7 application data. By simply clicking through each field and value, we can inspect even binary protocols without getting elbow deep into the RFCs or reverse engineering the protocol. Often, this is the place to start exploring an unfamiliar protocol because Wireshark generally includes human-readable field names and values.

Finally, the Packet Bytes pane contains a hex-dump-style listing of all bytes contained for that packet as stored in the pcap file or as they transited the network. At first, this may seem unnecessarily complicated when compared to the Packet Details pane, but having the raw data often helps an analyst to find interesting segments of the packet. To aid in this method of research, Wireshark conveniently lets the user click a byte in the raw hex or ASCII portions of the Packet Bytes pane, which triggers it to jump to and highlight the corresponding decoded elements in the Packet Details pane. The opposite also works—selecting a decoded field in the Packet Details pane will highlight the corresponding raw bytes in the lowest pane. This relationship is critical for an analyst to see and understand!

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

There are two additional items in the GUI that are extremely important to note. First, recall that Wireshark provides a robust set of display filters for us to use in narrowing down the number of packets we're looking at. We will take a closer look at these in a few pages, but the Filter Toolbar, seen at the top of the window, is the primary interface for this function.

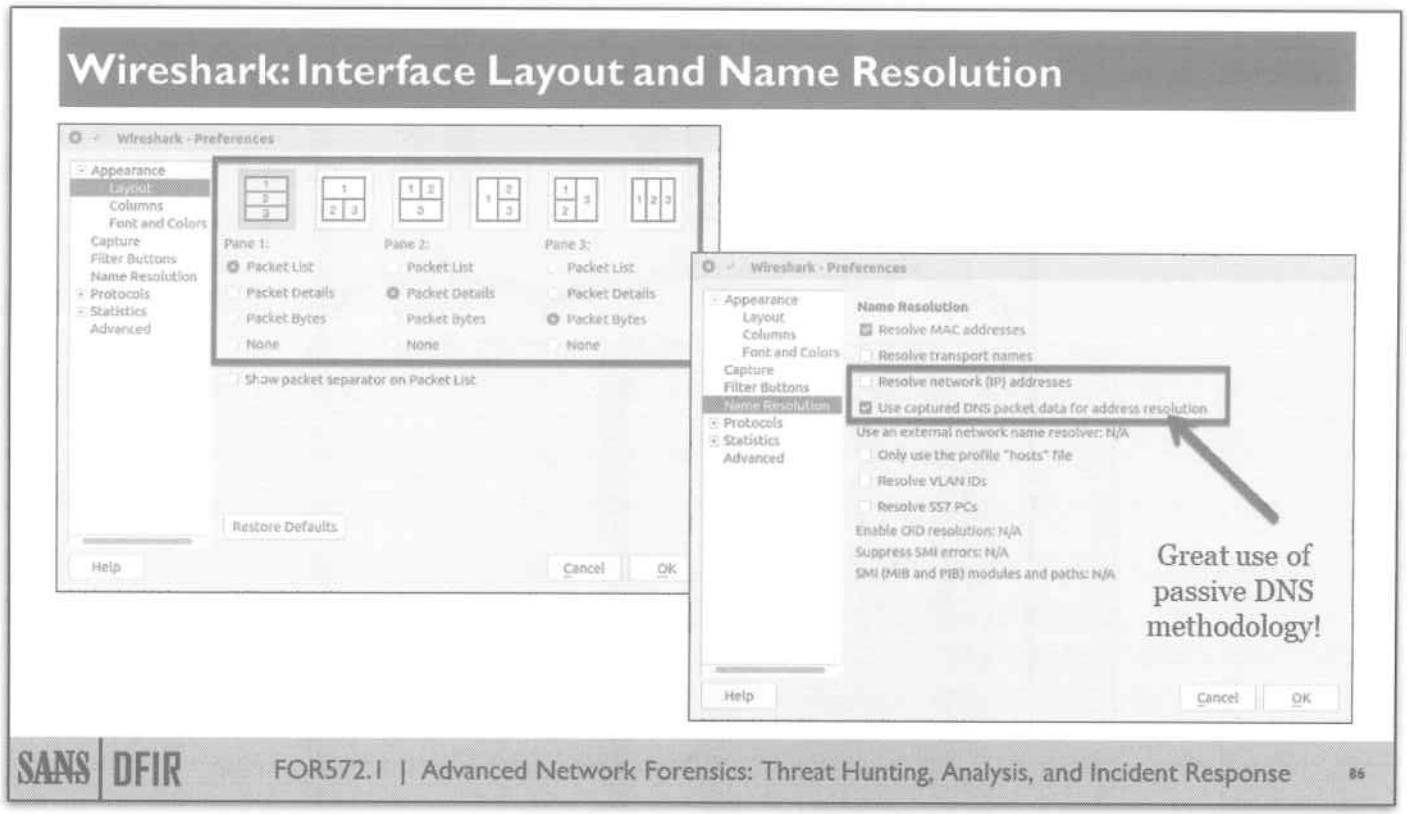
Finally, the data presented in the Status Bar at the bottom can help provide a quick frame of reference within the larger packet capture data. Wireshark first displays the decoder's unique field name and byte length. This is invaluable information for effectively creating display filters, which we will discuss in a moment. The center data set includes the total number of packets, as well as how many packets match the current display filter. If using Wireshark to capture packets, some additional values are displayed, such as the number of packets that libpcap had to drop due to resource limitations.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



There are a huge set of options that an analyst can use to tune Wireshark for very specific purposes. We'll focus on just a few of those options most relevant to our investigative goals.

One overlooked but useful feature allows the analyst to change the display panes to best suit their needs. This is often helpful if the system's screen is an odd size, or if one is displaying something on a large screen rather than a single-user workstation. The analyst can choose any of the six pane layouts, as well as which information will be displayed in each pane. While a purely cosmetic adjustment, this easy tweak can often provide a massive boost in usability.

The next feature deals primarily with OPSEC (operational security). By default, Wireshark will not perform DNS lookups for each IP address it sees. Their reasons are likely to minimize network traffic and load on the system, but ours includes the security posture of the analysis activity itself. For this reason, it is worth verifying this setting before you start looking at what could be hostile network traffic. It is increasingly common for an adversary to run their own DNS servers and monitor the queries made against them. In doing so, a well-funded, disciplined adversary could build a feature into malware that would cause DNS lookups of a certain pattern when traffic is observed in Wireshark. If their DNS servers saw this lookup pattern, the adversary would know two critical aspects of your analysis activity. First, they'd know that someone was looking at their traffic. Second, they'd know the IP addresses from where the DNS queries came, meaning they'd have a pretty good idea of where YOU are!

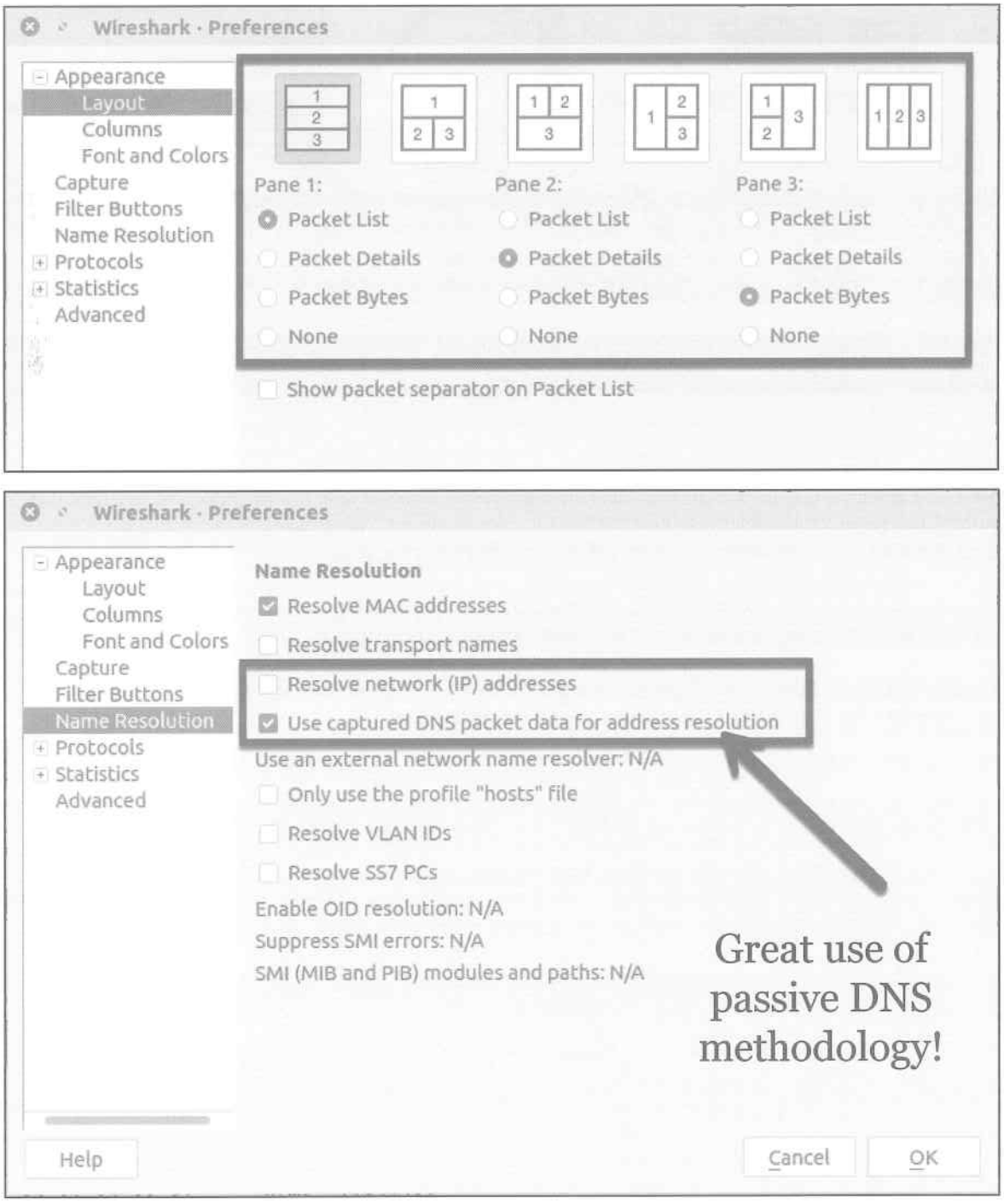
Although this may seem far-fetched or even a little bit paranoid, it has occurred in the wild, and adversaries are known to advance their capabilities quite quickly. Because best practice is to minimize unnecessary network traffic while reasonably maximizing OPSEC posture, this option should almost never be enabled.

Recent Wireshark versions have added a useful feature to address this risk (performance or OPSEC) by incorporating passive DNS features into the tool. If Wireshark's packet capture includes DNS traffic for the IP addresses, it will use that information to populate the user interface. This provides the analyst with the DNS data while avoiding the OPSEC risk, but also allows for point-in-time DNS resolution, enabling us to see what the DNS results were when the evidence was collected—not what it may (or may not) be at analysis time.

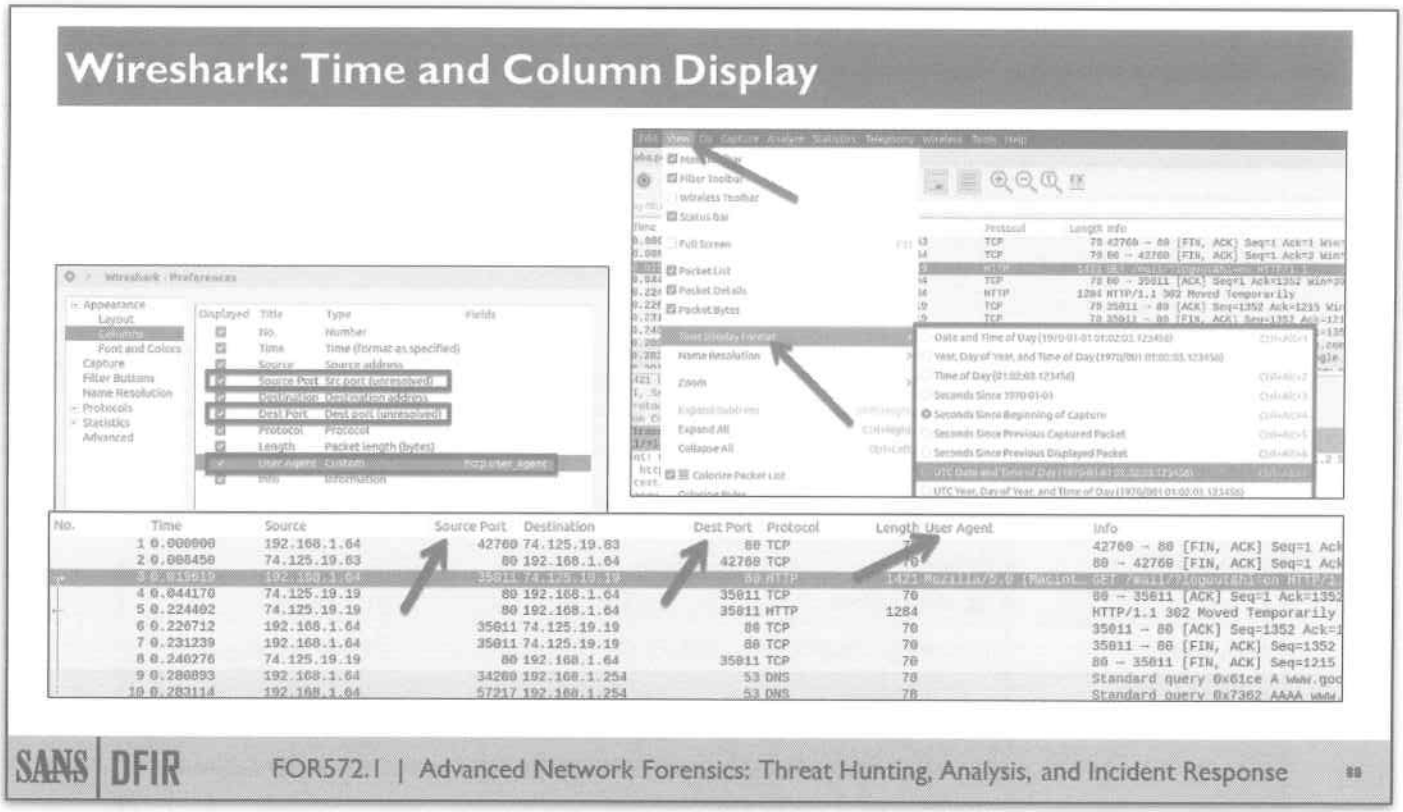
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

It's usually a personal preference whether an analyst enables MAC and transport name resolution, and neither causes Wireshark to generate its own network traffic or any OPSEC concerns. These both look up their data from files on the local system. MAC name resolution means that Wireshark will look up the first three bytes of a MAC address against a vendor table (e.g., "`00:03:47`" (and others) indicates an Intel device). This data is retrieved from Wireshark's "`manuf`" file (`/usr/share/wireshark/manuf`). Similarly, transport name resolution will use Wireshark's `services` file (`/usr/share/wireshark/services`) to look up human-readable network service names instead of ports (e.g. "80" indicates "http"). This is a convenience feature in the GUI and is looked up independently of whether the protocol in use actually matches the port.



https://t.me/learningnets



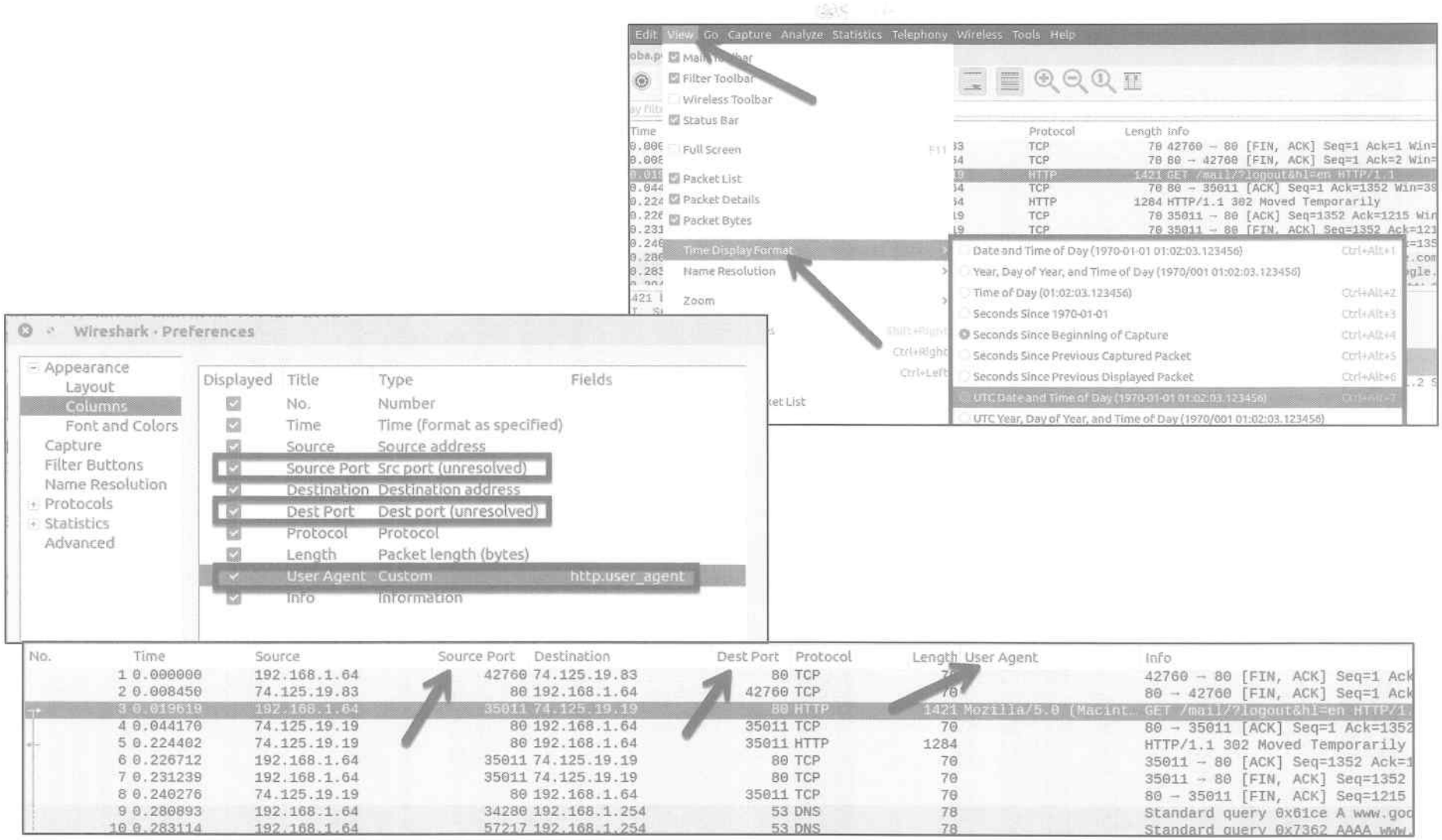
Although the default columns that Wireshark includes in the Packet Listing pane are often sufficient, you can add, remove, and reorder those that are shown. There are a number of standard columns available, but any field that Wireshark can parse can be added—as with the "`http.user_agent`" field shown in the screenshot. Packets without that field will simply show an empty value in the Packet Listing pane.

Previously, we discussed that there are different timestamp formats available in the GUI. You can see here the options you have in displaying these values, including formatting and precision.

Remember that the timestamps for all packets in a pcap file are UTC! (Unfortunately, this does not apply to timestamp values for data WITHIN the packet… but more on that later.)

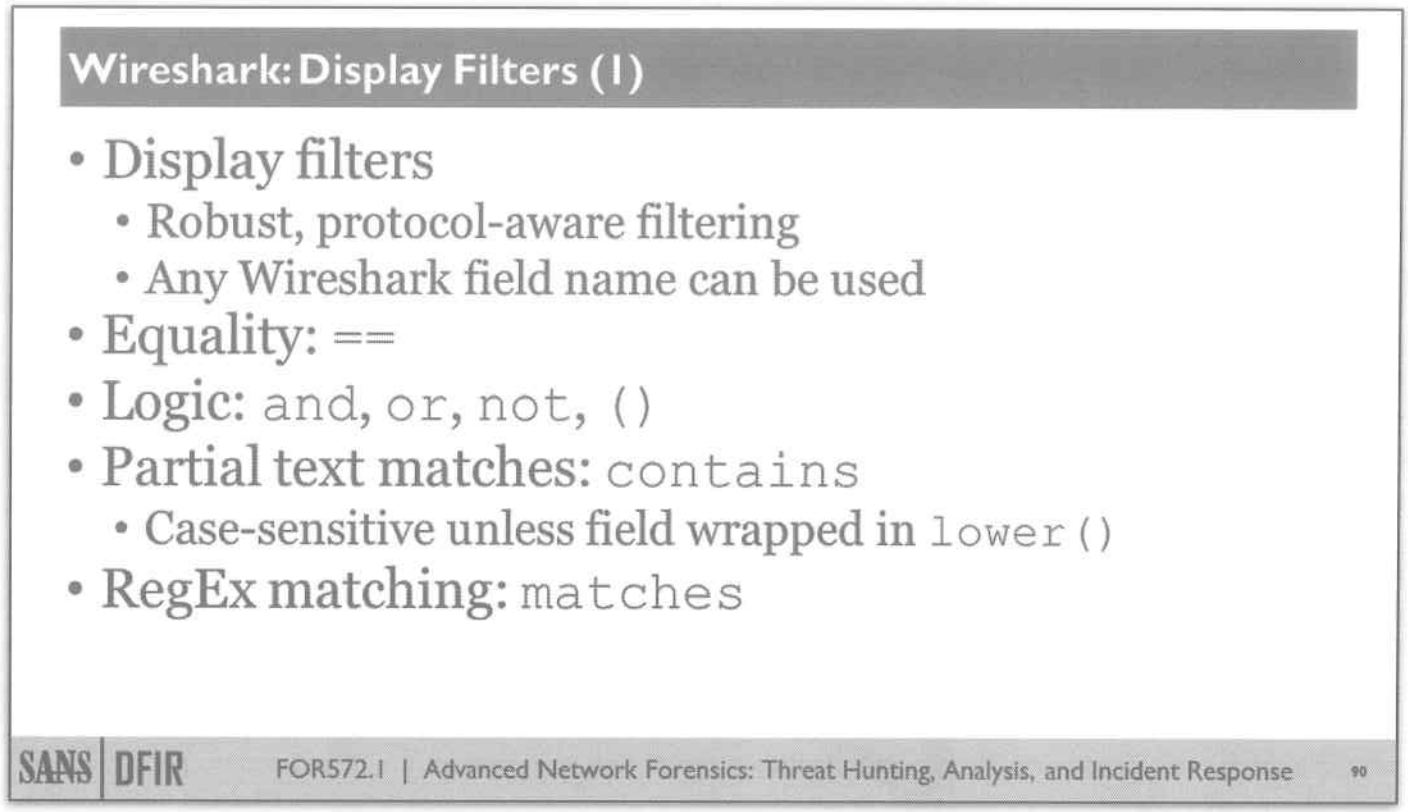
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



We've characterized Wireshark's Packet Details pane and its litany of decoders as one of the more powerful features. Display filters are probably the other contender for first place in that race. Essentially, the display filters provide the ability to sift through packets using all of the decoder-based features. Any field that Wireshark can decode can also be used as a display filter.

We already know that the BPF operates very close to the kernel, and is, therefore, an efficient and proper choice to limit what's pulled off the wire. However, there are quite literally hundreds or thousands of higher-layer protocols in use on the typical network, and using offsets and bit-masking to isolate interesting traffic is simply not feasible, even for the most brilliant of network investigators. The Wireshark developers saw this shortcoming and built the ability to use their decoders as filters directly into the software. Although not the fastest or most efficient way to filter packets, we most often use Wireshark on traffic that has already been recorded, not on real-time captures. In that vein, the speed-for-robustness trade-off is generally easy to justify.

Because Wireshark does parse IP, TCP, and UDP, there are display filters that provide the same functionality as a BPF would: `"tcp.port == 80"` and `"ip.src == 204.51.94.202"` do exactly what you would expect of them. However, BPFs will be significantly faster than their equivalent Wireshark display filters, so you should seek to use BPF wherever possible.

But, that doesn't start to scratch the surface for what we can do with display filters. The possibilities here are virtually endless. For example, if we wanted to identify HTTP traffic on a nonstandard port that contained the word "stolen", we could use the following display filter:

```
'(not tcp.port == 80 and not tcp.port == 8080) and ⏎
    http contains "stolen"'
```

The power of a well-crafted display filter becomes even more evident when looking for DNS replies containing more than five responses for a query against a known hostile domain:

```
'dns.flags.response == 1 and dns.count.answers > 5 and ⏎
 dns.qry.name contains "cz.cc"'
```

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Note that the "contains" parameter is case sensitive. Fortunately, Wireshark also provides a Regular Expression matching parameter, which will allow case-insensitive matches, as well as all the other power that RegEx brings to the table. The following example will match HTTP traffic that contains the string "username" in the cookie value, but in a case-insensitive manner:

```
'http and http.cookie matches "(?i)username"'
```

Another clever means of doing case-insensitive searching is to convert a field value to lowercase before evaluation, such as the following:

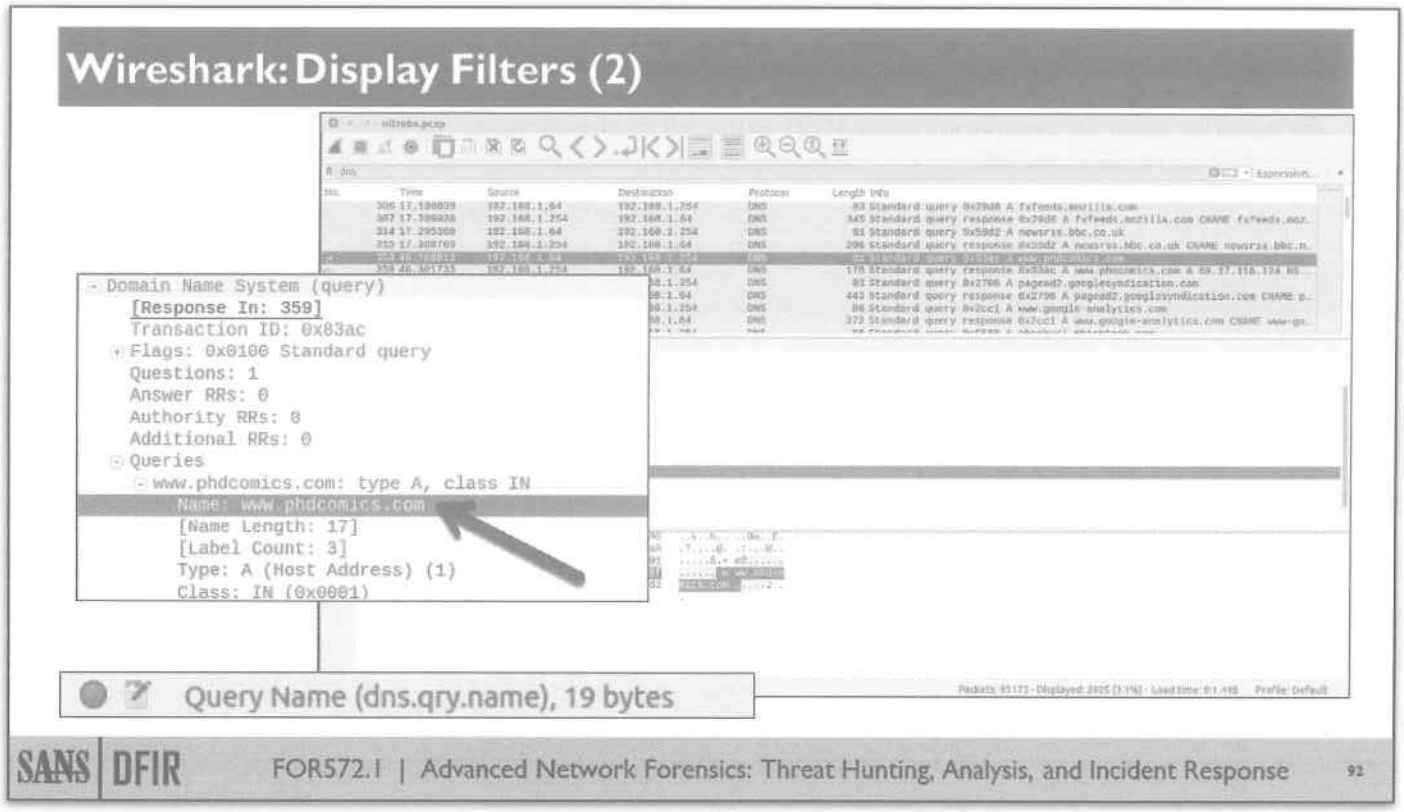
```
'http and lower(http.cookie) contains "username"'
```

But, with so many protocols, how can we easily identify the field names to use in the filters? Back to the GUI…

References:

http://for572.com/-52to
http://for572.com/g2ayh
wireshark-filter(4)

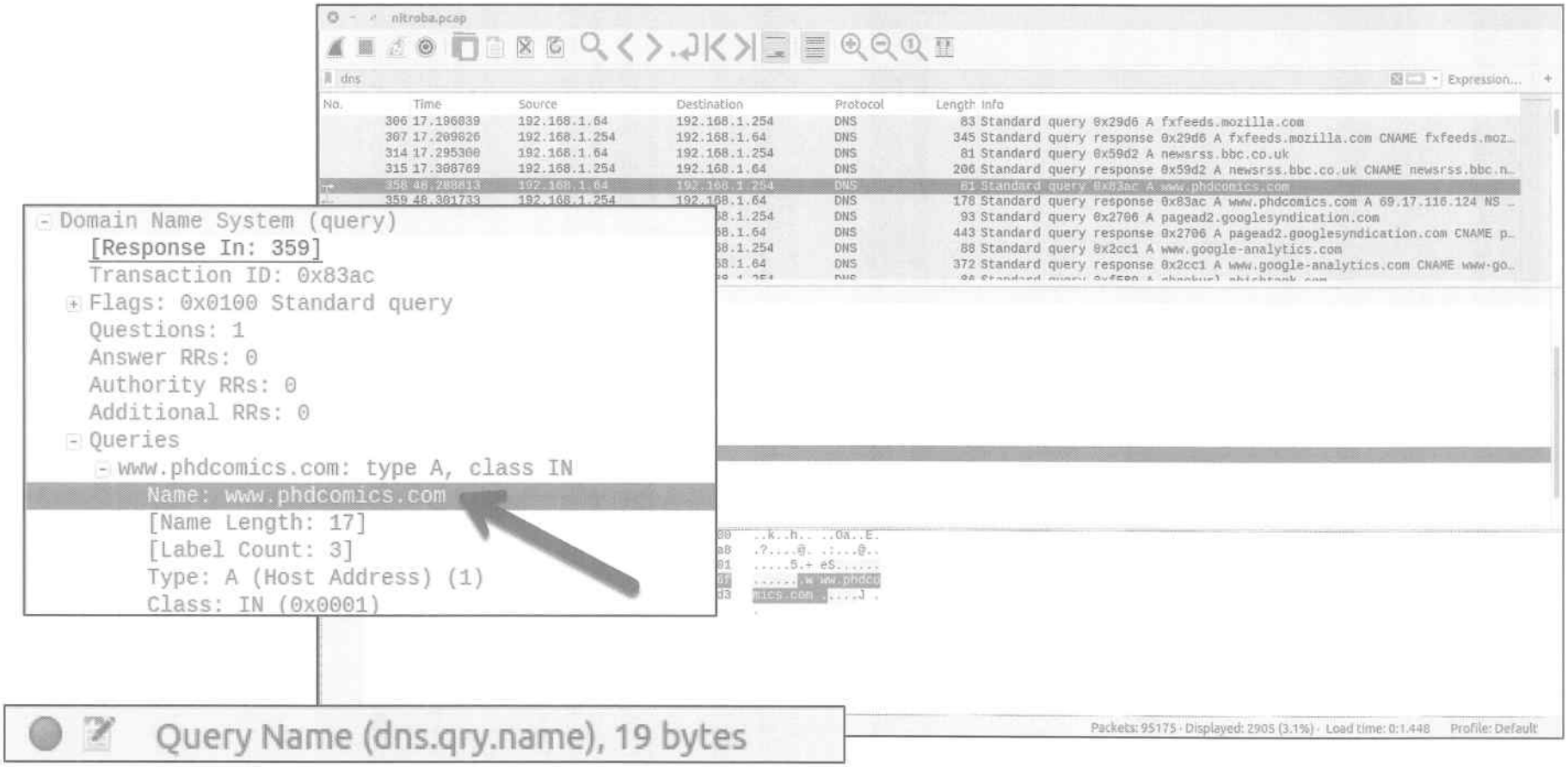
https://t.me/learningnets



Take the example above. The Packet Details pane shows a basic DNS query for the A record associated with the hostname "`www.phdcomics.com`". By looking at the left side of the status bar, we can see that the DNS decoder uses the name "`dns.qry.name`" to denote this field, and that in this case, the field is 19 bytes long (17 bytes of text in three labels plus two bytes for the interstitial dot characters). We can then use this field name to craft a display filter that will quickly and nondestructively narrow down the packet listing to just those we are interested in.

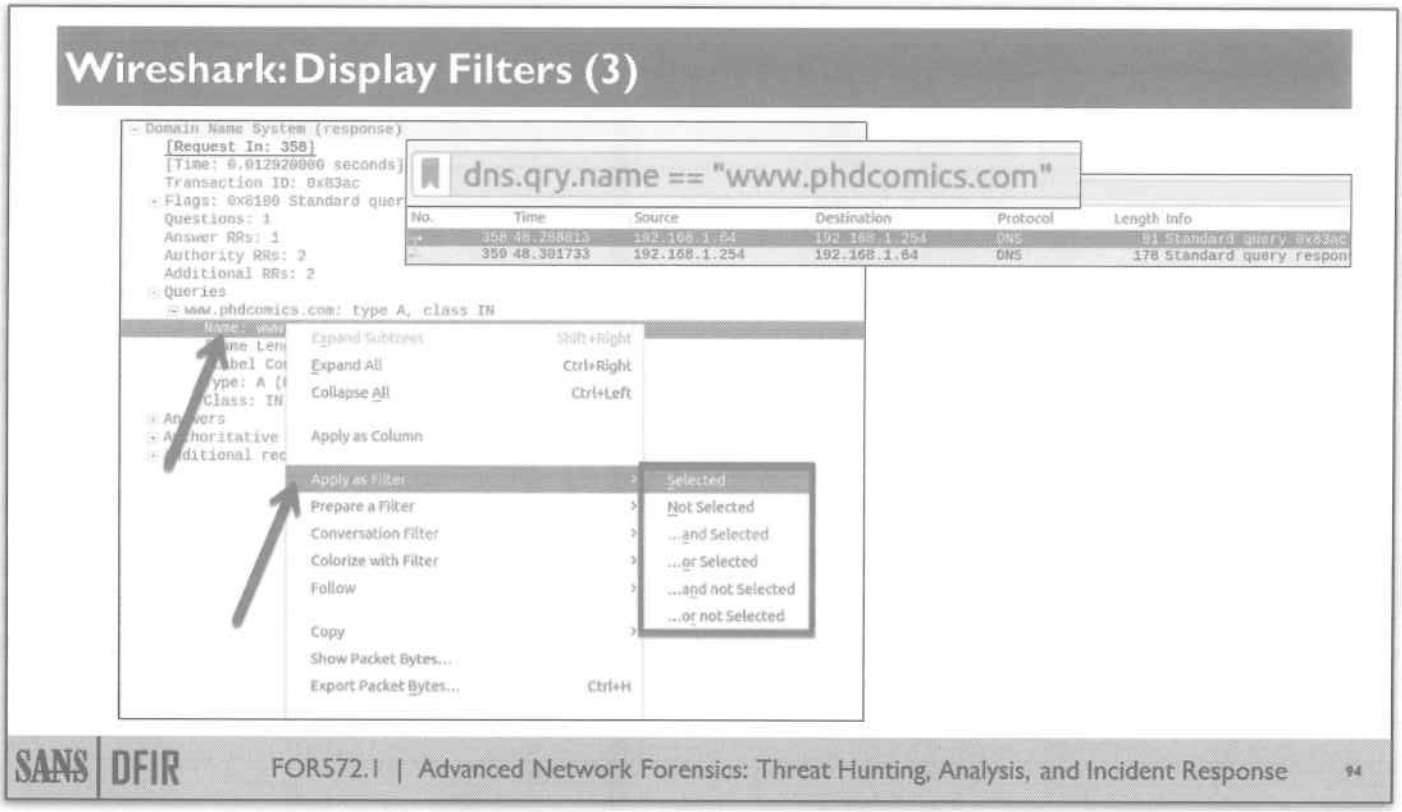
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



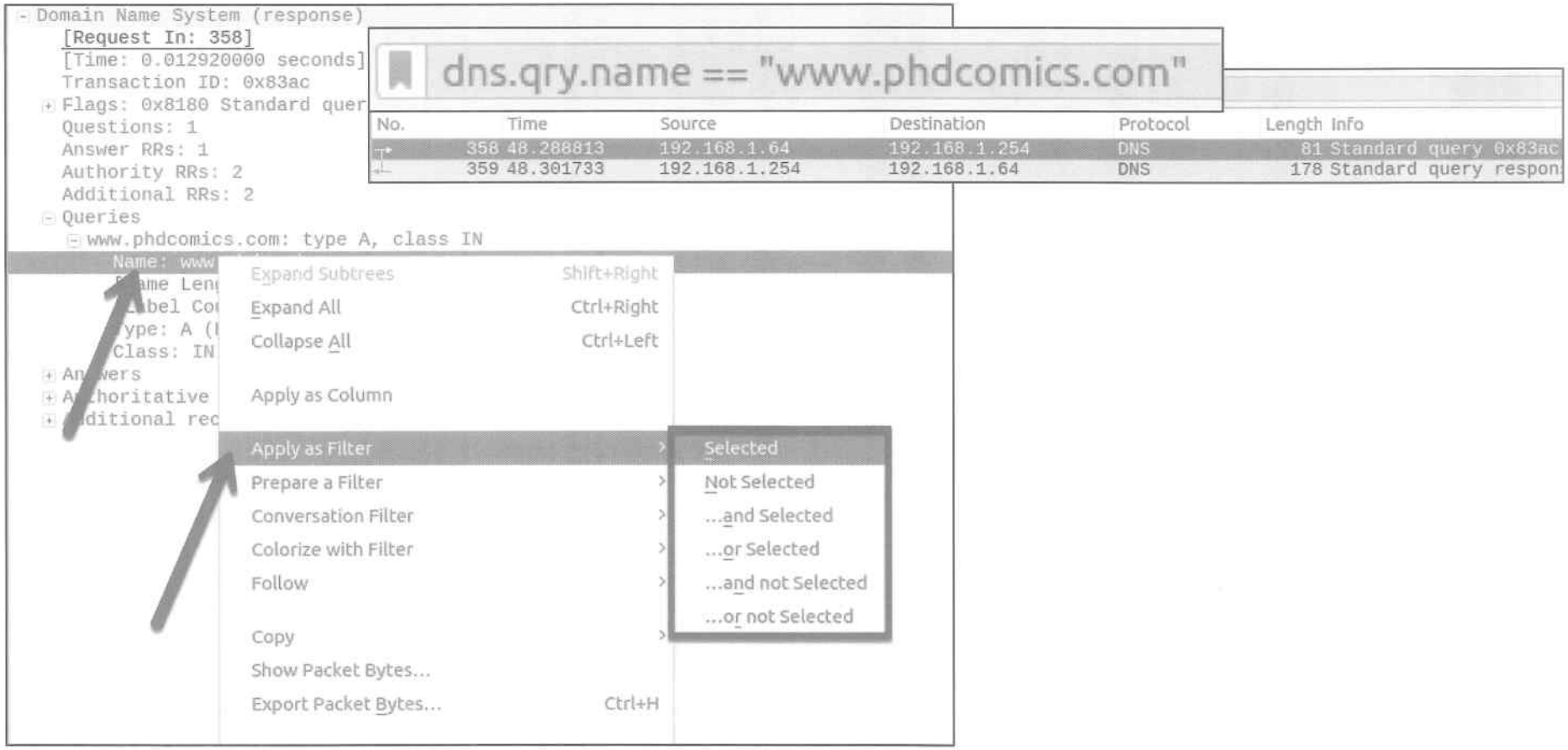
Here, we've taken that field name (`dns.qry.name`) and crafted a simple display filter: Packets with an exact match of "`www.phdcomics.com`" in the field. Wireshark allows the analyst to create display filters manually, or, as shown here, use the GUI to build them interactively. Simply right-click a field name, then select the "Apply as Filter" or "Prepare a Filter" submenus and one of the resulting pop-up selections.

The difference between "Apply" and "Prepare" may be subtle at first: **Apply** will immediately apply the newly created filter to the traffic, whereas **Prepare** will enter the filter only into the display filter bar. As you'll learn during this class, if you haven't already experienced it, applying display filters to a large packet capture can be a VERY long process. If you wanted to refine a filter a bit before applying it or build a display filter with multiple conditions, the **Prepare** option can save you a lot of time watching the progress bar.

As an aside, those familiar with the inner workings of the DNS protocol will be pleased to realize that display filters apply to the logical contents of the packet, not necessarily the bytes contained within. DNS has a particularly hairy "compression" algorithm that improves transmission efficiency by de-duplicating byte sequences within a packet. Though an elegant solution, it makes walking through the raw bytes of a DNS packet a rather unpleasant experience. Wireshark decodes first, and filters second—saving the analyst's precious brain cells in the process.

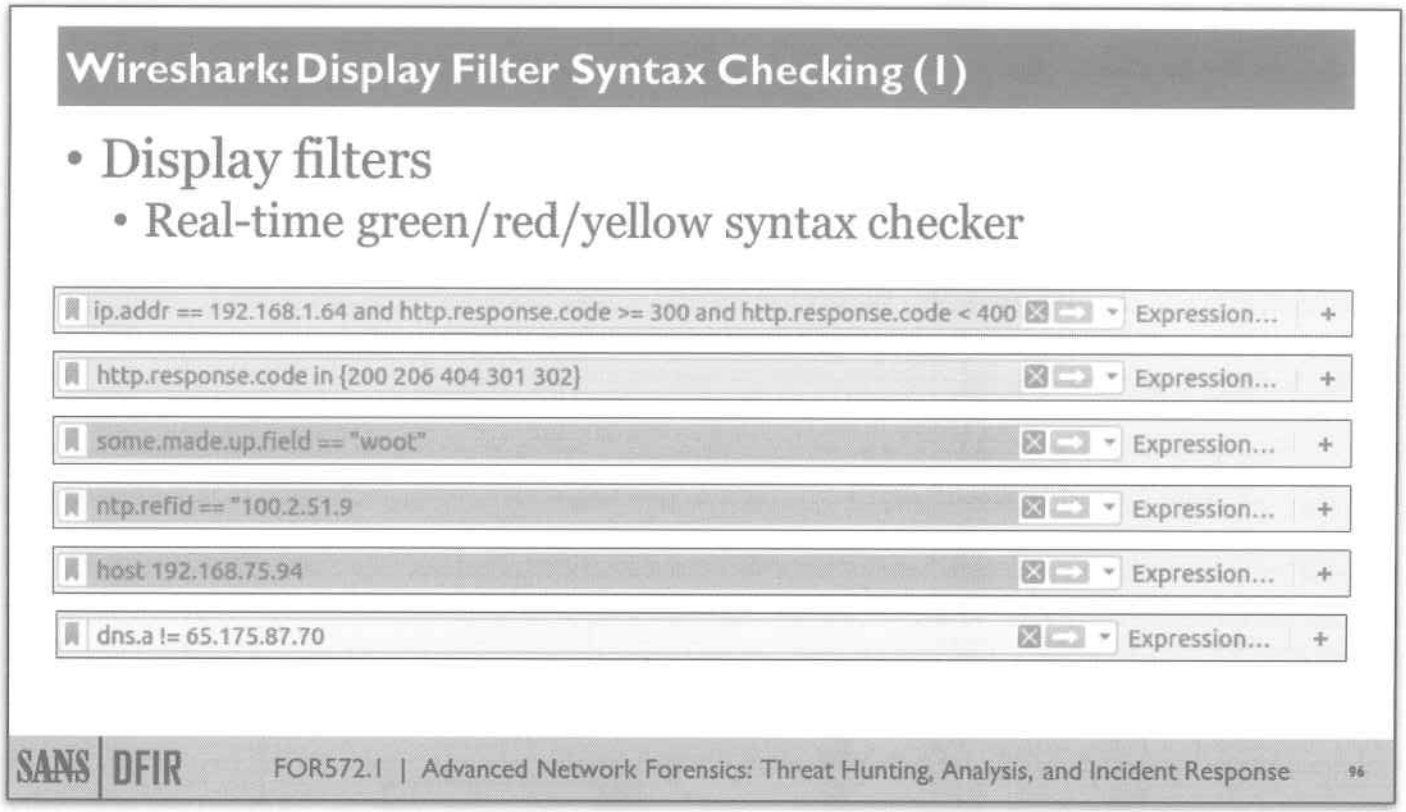
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



Because the display filters can get pretty complex, the Wireshark developers have employed a convenient "stoplight" visual aid that checks the syntax of a display filter as it is typed. A green background predictably indicates that the syntax is valid. Red, of course, means that you've either specified an unknown field or that there is some kind of syntax error in the field. In the second **red** line above, the double-quote has not been closed in an otherwise valid filter. In the third **red** example, a `tcpdump` capture filter was used, which is incompatible with the Wireshark display filter syntax.

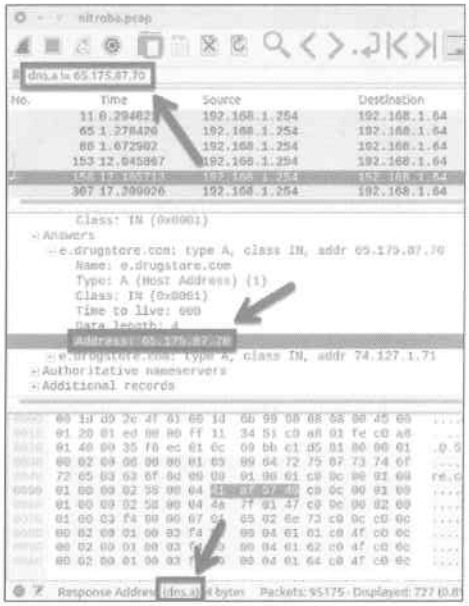
However, the **yellow** indicator is a little less intuitive and warrants further examination. This will trigger any time the "!=" operator is used, which may not behave as common computer science would suggest. It specifically behaves in a somewhat unexpected way when building a filter with a field name that can occur multiple times in a single frame.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Wireshark: Display Filter Syntax Checking (2)

- Protocols with multiple values per field name require special attention
  - DNS responses (`dns.a`, `dns.ns`), IP addresses (`ip.addr`), etc.
  - How to filter for responses that do not contain a specific address?



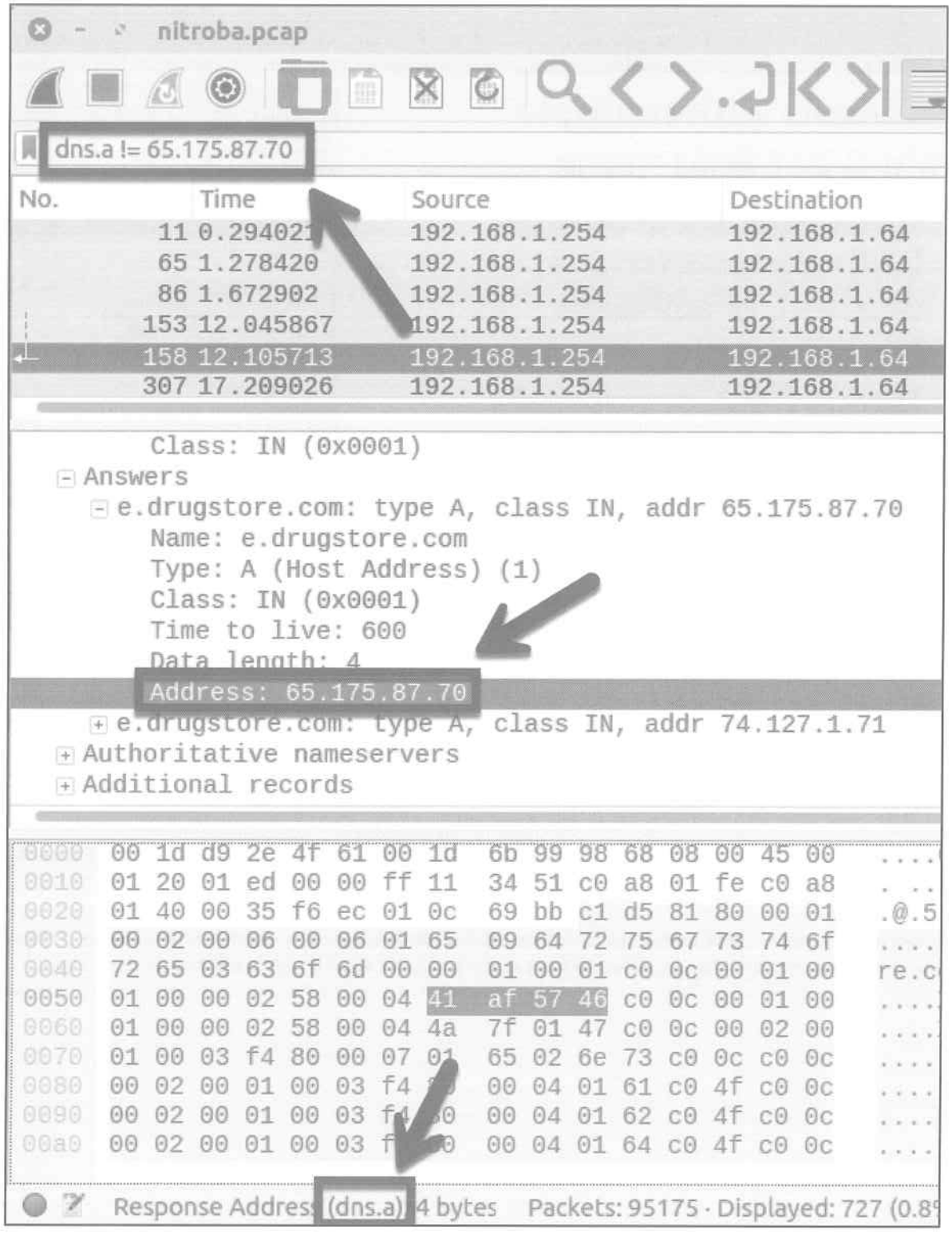

```
dns.a && !(dns.a == 65.175.87.70)
```

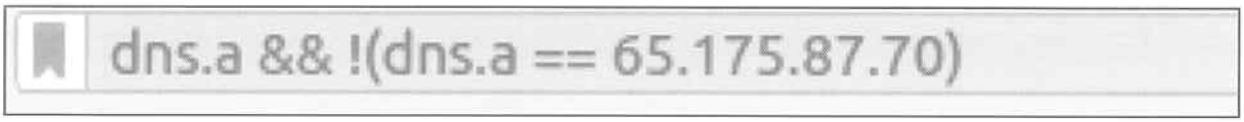

The screenshots above demonstrate the nuance of the "!=" operator and the resulting **yellow** syntax checker status. The larger screenshot on the right side shows a display filter of "`dns.a != 65.175.87.70`", but the contents of the DNS response data clearly show that IP address is present. This is because the response also contained an additional "`dns.a`" field with a value other than "`65.175.87.70`"—resulting in a matched frame. However, the display filter "`dns.a && !(dns.a == 65.175.87.70)`" will match any frame that contains the "`dns.a`" field itself, but will exclude those with ANY such field's value set to the specified IP address. It's critical to understand these nuances before scaling a display filter to a large data set, or you will generate grossly inaccurate results.

One key thing to keep in mind when using the syntax checker, though, is that it's merely a syntax checker—not a logic checker. Just because a display filter validates as **green**, doesn't mean it will match the packets you want.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham





https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Wireshark: Follow TCP Stream

- Follow TCP Stream
  - View ASCII/hex content of a connection
  - Right-click TCP packet, select "Follow | TCP Stream"
  - Choose client-to-server, server-to-client, or both
  - Save selected side(s) of stream to file

The "Follow TCP Stream" feature is very powerful and one that you will use extensively during this course and your real-world cases. The feature is activated by selecting a packet, then right-clicking the packet (or selecting the "Analyze" menu item) and selecting "Follow | TCP Stream". Wireshark will pop up a window that displays one or both sides of a TCP conversation in ASCII or hex-dump-style output. Wireshark colorizes these conversations with red text representing the client-to-server side, and blue for server-to-client. It's particularly helpful for observing ASCII protocols, as you can easily see an entire exchange in one view.

As you can see in the screenshot here (and larger on the next page), the client requests to the server are in red, and the resulting responses are in blue.

Note the dropdown labeled "Entire Conversation" in this screenshot can be used to select just one side or the other of the conversation. This is helpful when examining a stream that is less distinct than the HTTP example here in terms of a block of client-to-server and server-to-client transmission.

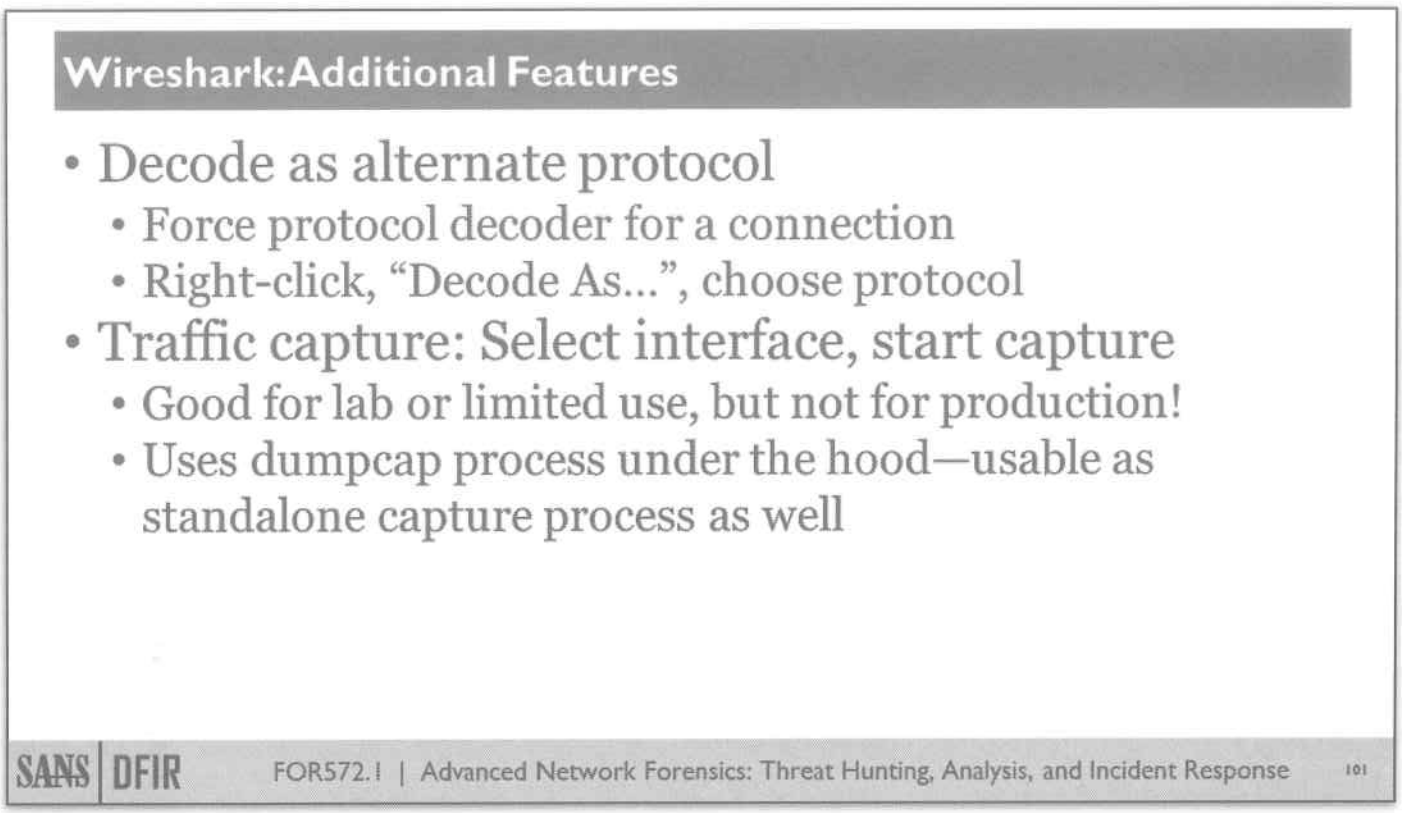
The "Save As" button also provides a helpful function. This will save to disk the raw content of just the data portion of the side(s) selected in the dialog box.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

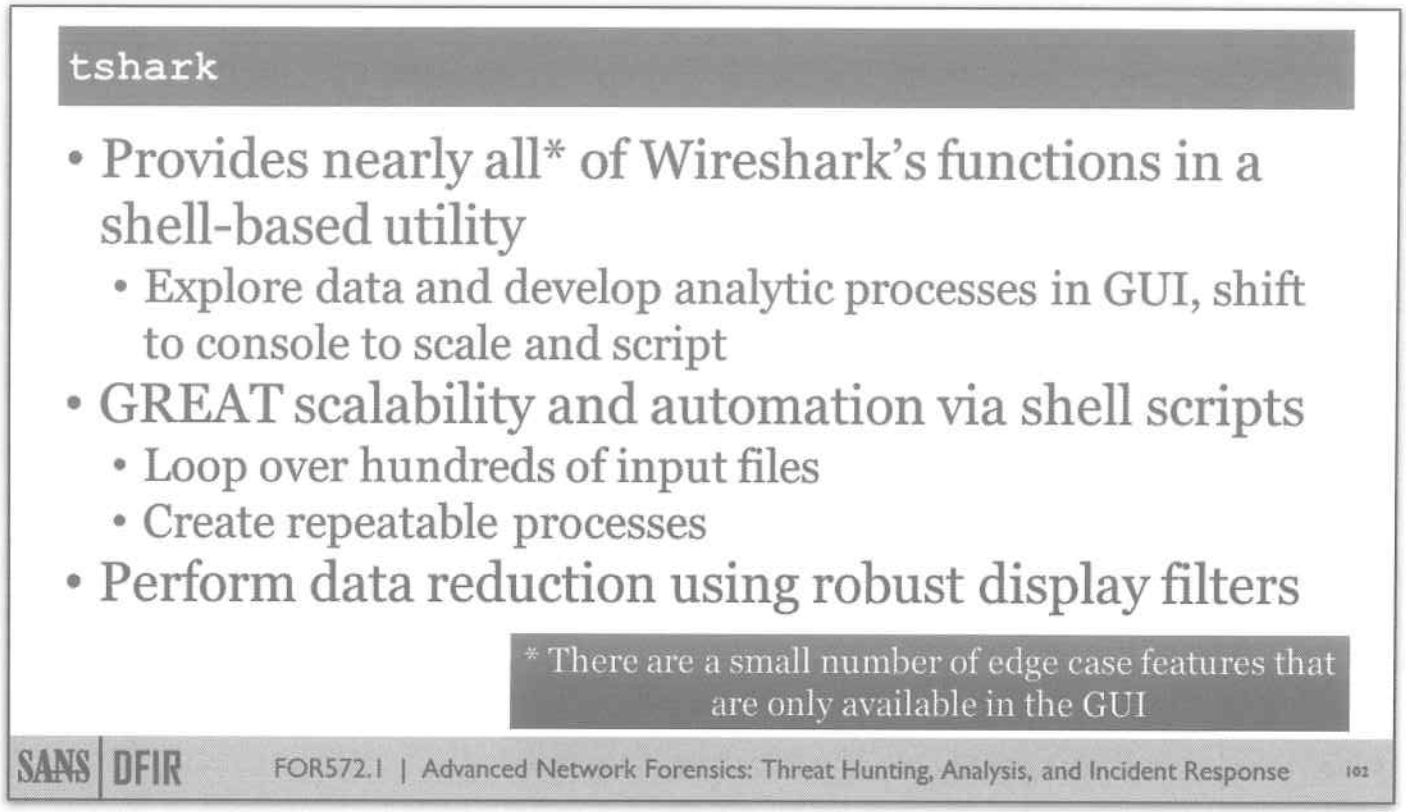


Wireshark is a very powerful and exceedingly useful tool. There are countless features that could benefit you during your investigations, and there is no way for us to fully cover them all here. The wiki-based documentation is quite extensive and can be a great resource for even Wireshark veterans. As you're able, here are a few of its features that you might want to research, as they'll come in quite handy during this class:

- Decode as alternate protocol: This is handy when you have a sneaky user that's trying to push one protocol over an alternate port in an attempt to evade detection. Instead of relying on Wireshark's own protocol detection routines, you can force a connection to be parsed with a decoder of your choosing.
- Traffic capture: It is often convenient to capture and analyze traffic in one step, and Wireshark allows this functionality. Although doing so in the GUI exposes all the same functionality as tcpdump does natively, don't overlook the significant amount of processing overhead involved with the GUI and protocol decoders. Wireshark should never be used to capture mission-critical data because the dropped packets will add up quickly. But, in the right environment, skipping the dedicated capture step can be a time saver. However, the latest versions of Wireshark do use a separate child process (dumpcap) to perform the capture itself. Because this utility can also be used directly to capture traffic without the GUI, it may provide a suitable option in some cases. As always, be sure to test your tools before using in production!

References:
`http://for572.com/w16uf`

https://t.me/learningnets



A handy tool that we will use extensively in this course is `tshark`, Wireshark's "fraternal console twin". In some form, nearly all of Wireshark's features are available in the console. This is most often useful because we can use the Wireshark GUI to explore a small subset of network traffic, then establish a plan on how to process a larger data set. Those methods transfer from the Wireshark GUI to `tshark`, where we can run against extremely large data sets without the overhead from the GUI.

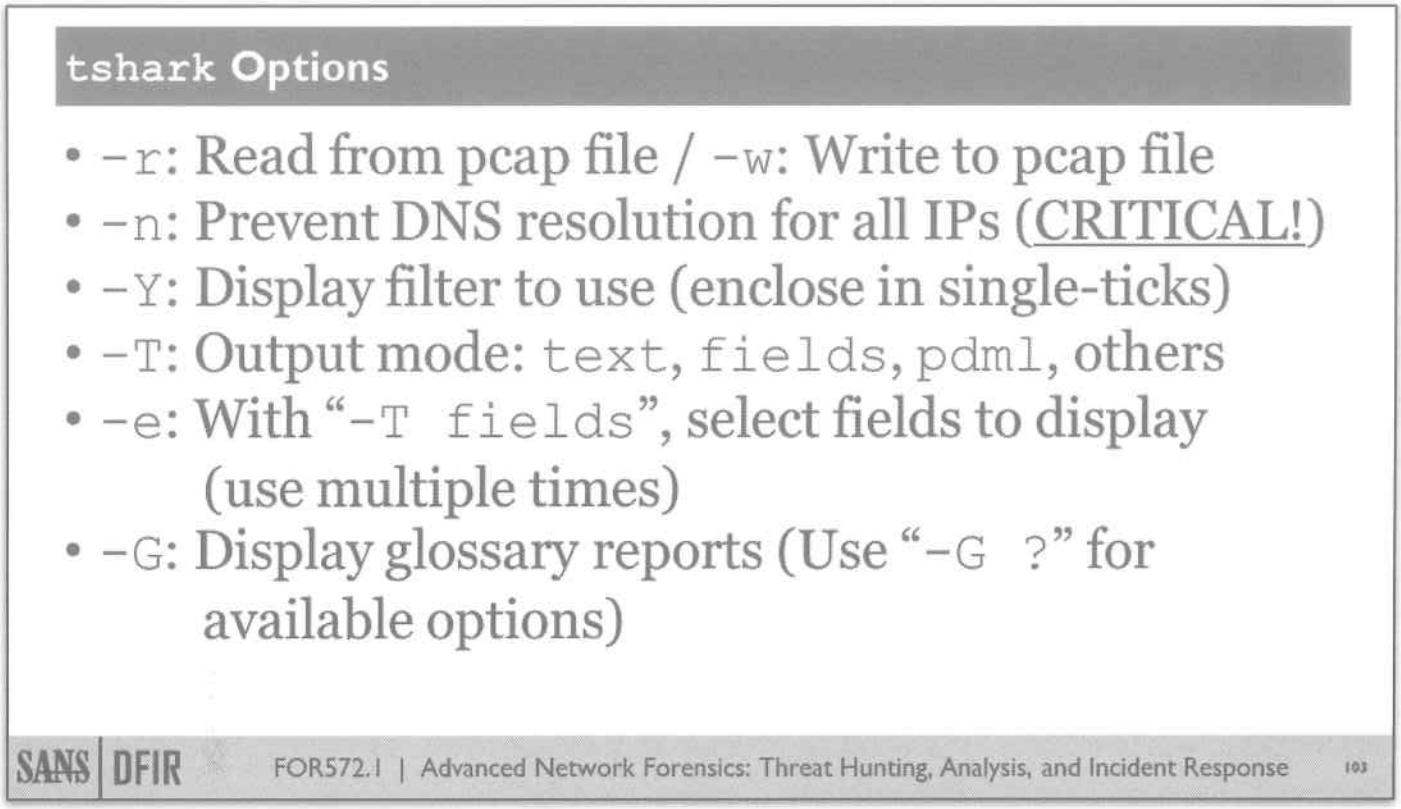
For example, you might explore a small set of DNS traffic, then identify a hostname you believe is associated with malicious activity. With this knowledge, you can apply the resulting display filter to a much, much larger data set, identifying all client IP addresses that made a DNS request for the suspect hostname.

This mode of using the power of Wireshark/`tshark` is especially seen when using shell scripts to build scalable and repeatable processes. You might create a loop that iterates over dozens or hundreds of smaller input pcap files, applying the same process to each. You might also build up a series of command lines that perfectly addresses your requirements, enabling a repeatable process.

Because `tshark` can also write matching packets to a pcap file, as most libpcap-based utilities do, we can also perform data reduction based on the powerful display filter functions it provides.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



The `tshark` utility provides a multitude of powerful options. Some of those that we will frequently use in this course are listed above. As with most tools, the `tshark` man page, as well as the "`-h`" option will help you to understand the full complement of options it provides.

`-r` Read packet data from an existing pcap file rather than a live network interface.

`-w` Write packets matching the supplied BPF and/or display filter to a pcap file.

`-n` Prevent any DNS resolution. This is the same reasoning we discussed with `tcpdump` and should also become an automatic flag!

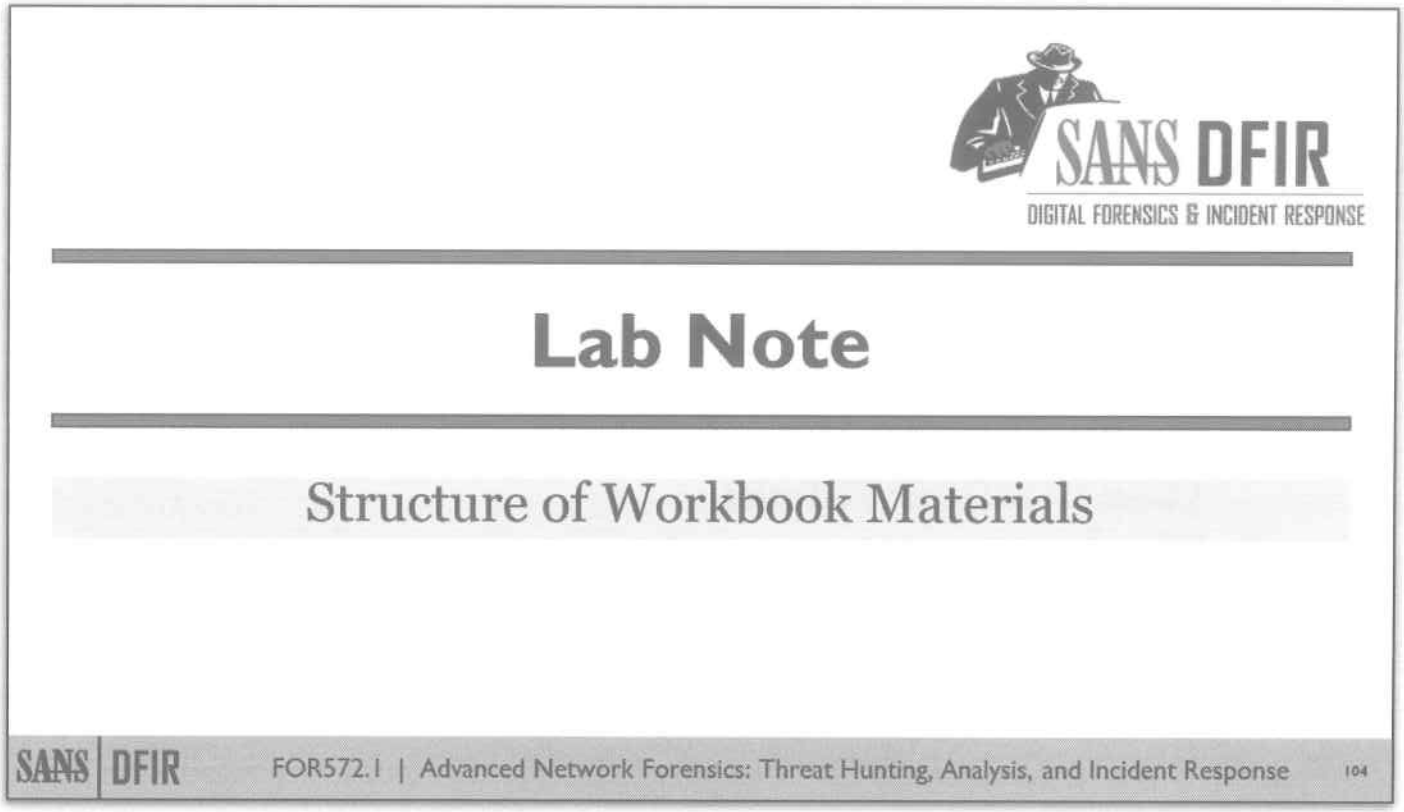
`-Y` Apply a display filter using the same syntax as the GUI version of Wireshark. Because these may include characters that would cause special handling in the shell (parenthesis, spaces, etc.), it's best to enclose the filter string in single quotes.

`-T` Specify the output format for the results. The default of "text" displays a human-readable, one-line summary per packet similar to that of `tcpdump`. The "pdml" format is an XML-based format suitable for further machine parsing. However, perhaps the most novel and useful output format is the "fields" mode. This mode allows the user to display only the specific Wireshark/`tshark` fields of interest, in tab-separated or a similar machine-parseable format. We will use this output mode extensively throughout the class, and you'll soon see its clear benefit to the analyst. The "fields" output mode allows shell-based analytics to provide quick and decisive insights, and the analyst can also use this mode to prepare intermediate results for subsequent visualization or similar processes.

`-e` To specify which values will be displayed in the "fields" output mode, add one or more "`-e`" options to the `tshark` command line. Specify field names in the "machine-usable" format, as shown in the Wireshark status bar.

`-G` If you are unable to use the Wireshark interface to determine field names, or simply want to perform a broader search, the built-in glossary can be helpful. For example, the command "`tshark -G fields`" will display a list of both the machine- and human-readable formats for each field that `tshark` can decode. This is most useful in conjunction with "`grep`" to isolate the field(s) of interest because the entire list of over 174,000 fields from over 2,200 protocols is a bit unwieldy!

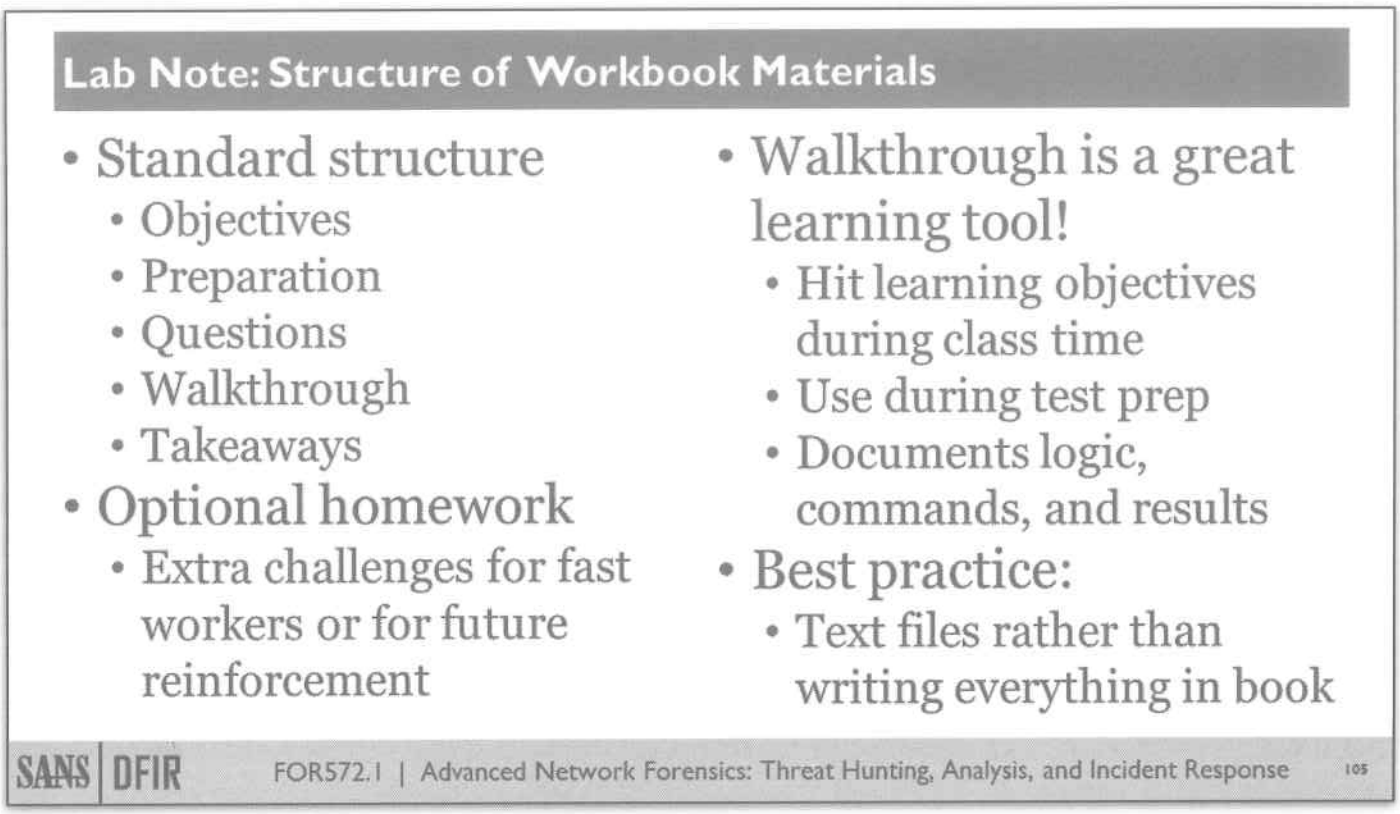
https://t.me/learningnets



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



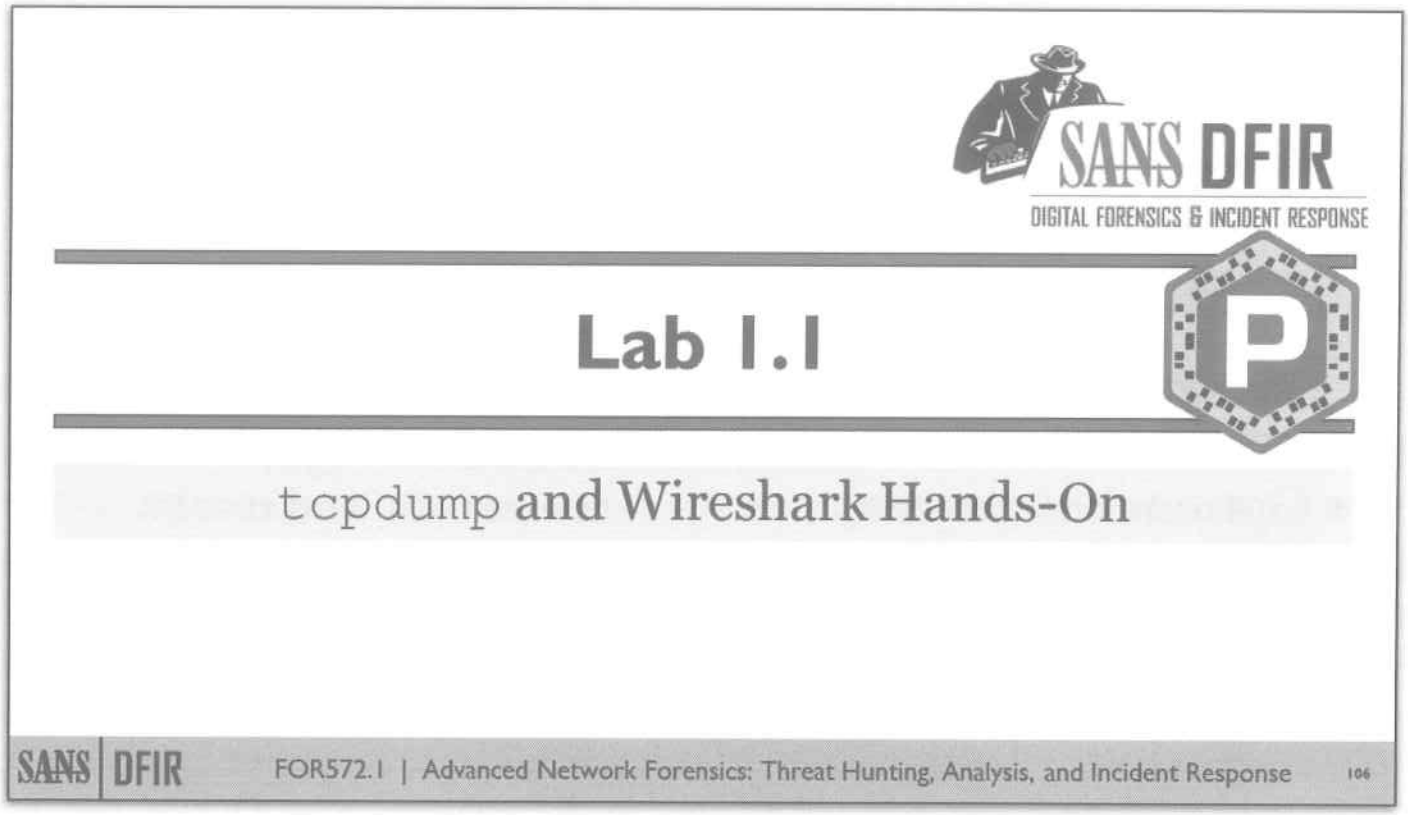
Because this is the first hands-on lab in the class, let's take a moment to explain the overall structure for each of the exercises you'll use. These labs reinforce classroom material as well as introduce some new tools and concepts. However, they all follow a common structure:

- The "Objectives" section details what concepts and learning objectives each lab will address.
- The "Preparation" section indicates what source evidence files will be used (if any), as well as what fundamental items of interest are provided to "kick off" the lab.
- The "Questions" and "Questions with Walkthrough" sections reflect identical content, with the exception being that the walkthrough includes the logic, commands, and expected results for one possible path through the evidence. That's not to say this is the only solution—there could be countless ways to arrive at the same findings. This section is also not "the answers"! You're not "cheating" by looking at the walkthroughs. Some students prefer to take the challenge head-on and avoid peeking at any of the commands—which is great! Most will use the walkthroughs when they hit a roadblock, then shift back to the "Questions" section and keep plugging away. Those who are not as comfortable with the command line prefer to use the walkthrough as a guide during class, then shift back to the raw questions after the class when they're reinforcing the learning.
- Finally, the "Takeaways" section reiterates how the lab addresses the learning objectives and the key findings from each hands-on exercise.

During the labs, don't feel compelled to write every answer down in the book. Nobody expects you to write out an MD5 checksum, or a 120+-character HTTP User-Agent string! Instead, consider creating text files to house your results in the `/cases/for572/<exercise number>/` directories, or on your host via a `/mnt/hgfs/` shared folder. (This will help to avoid data loss if you revert the snapshot on your SIFT VM.)

https://t.me/learningnets

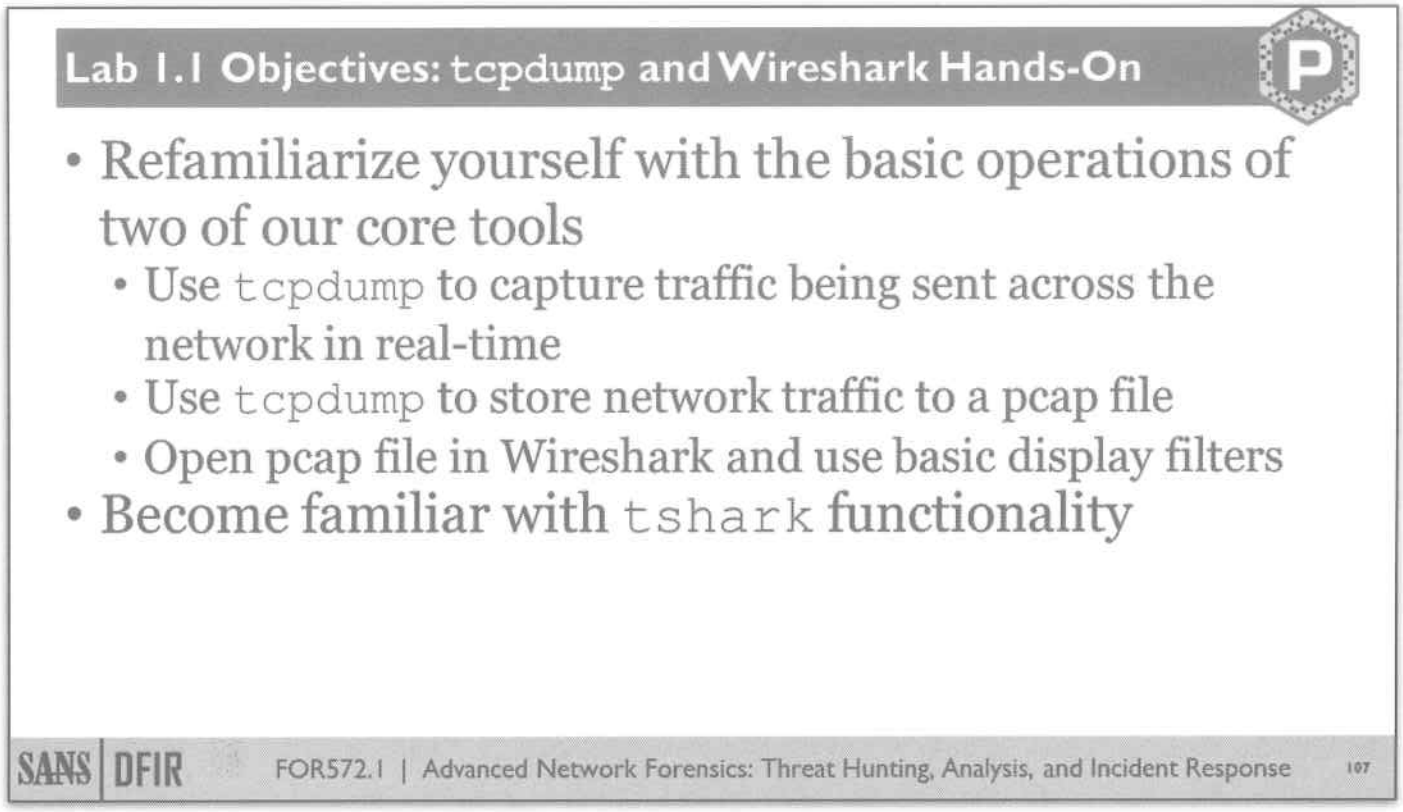
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

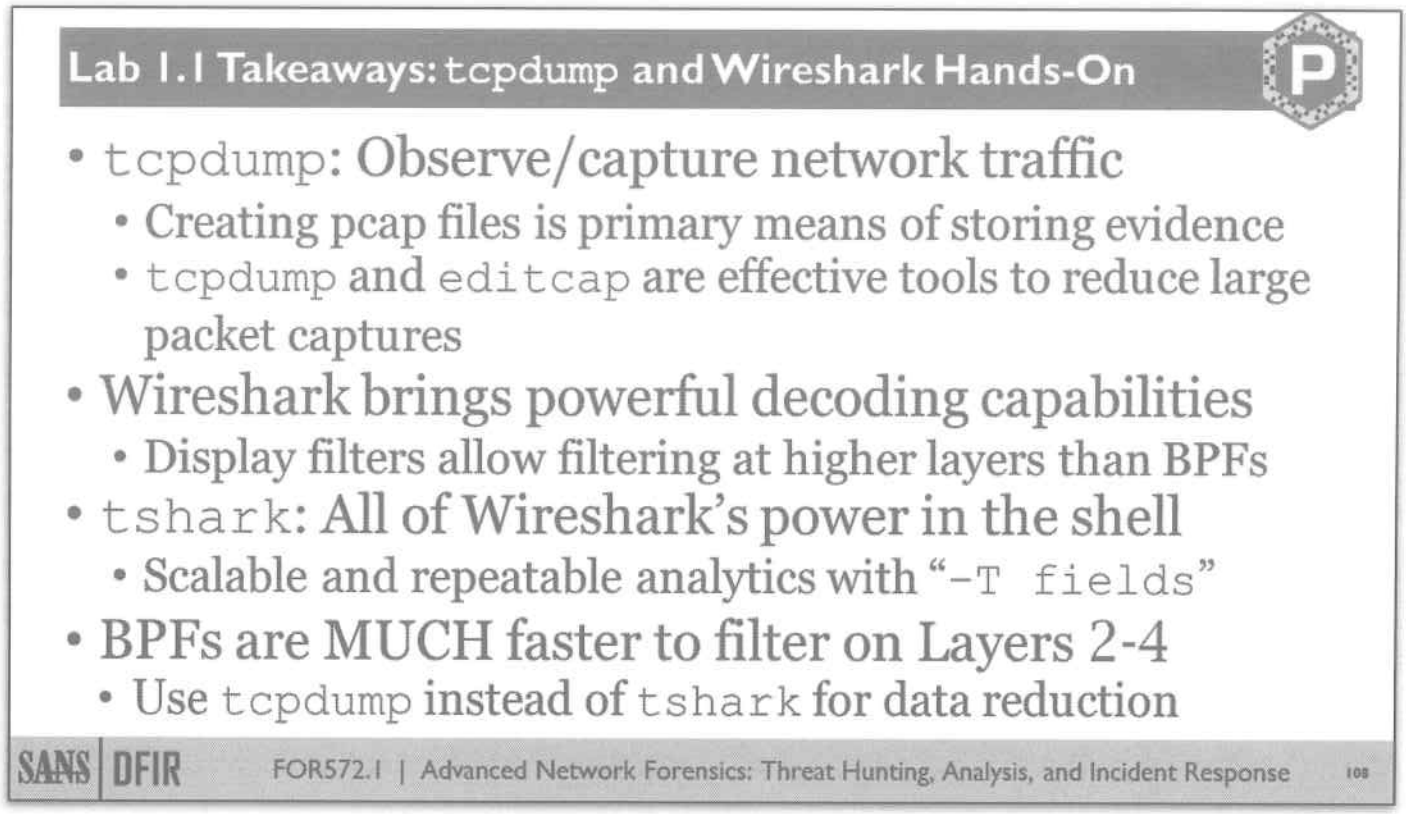
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

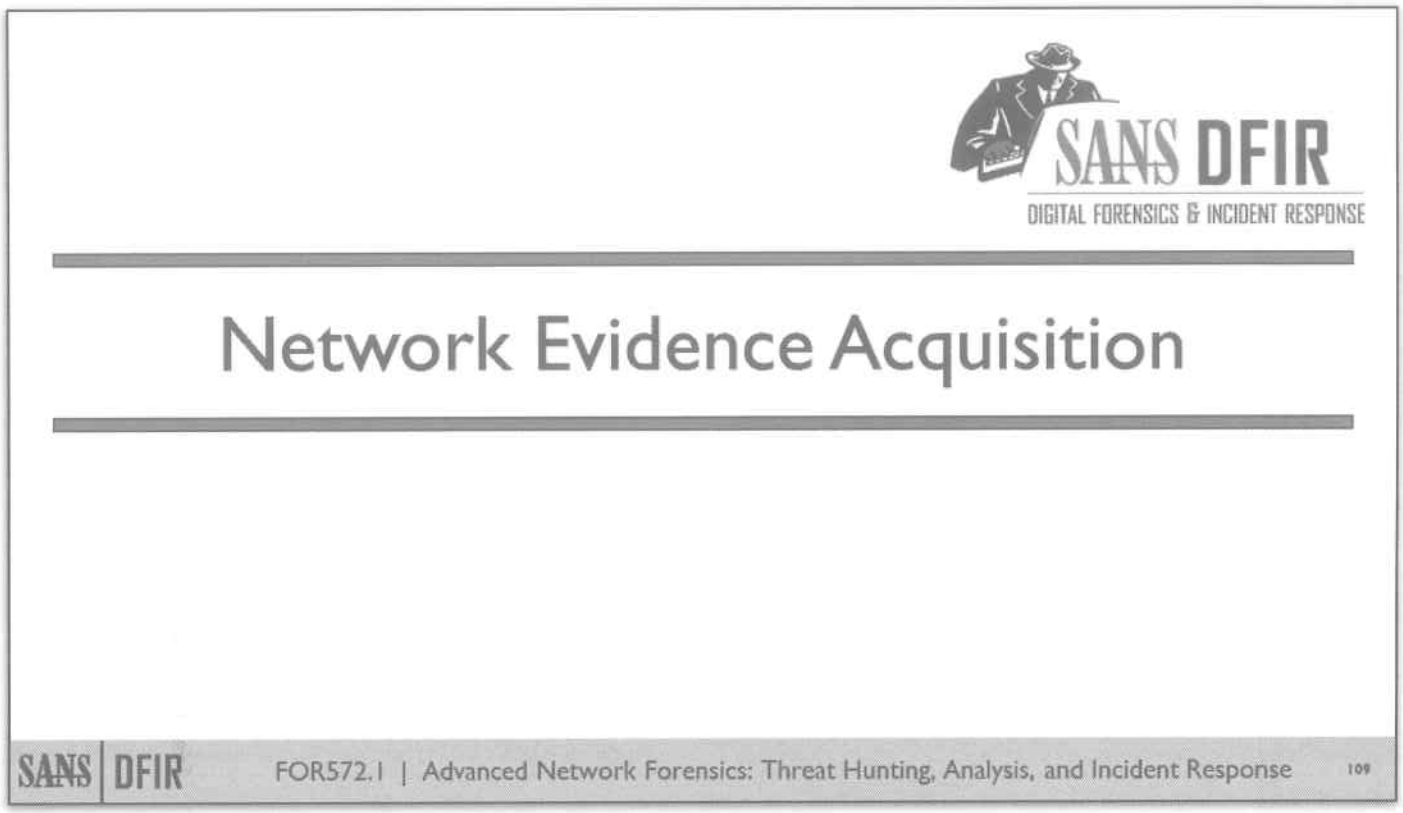
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

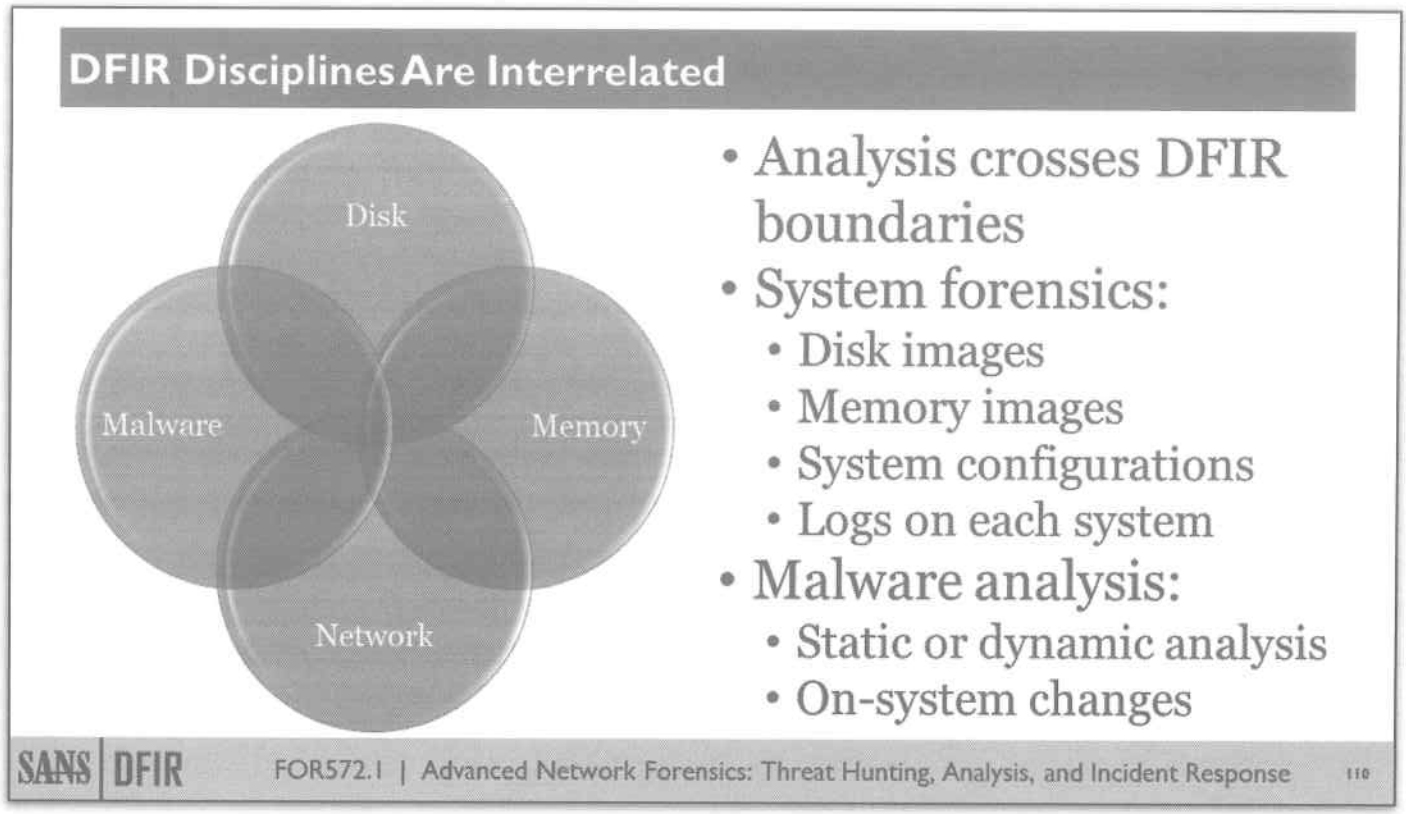
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Of course, no single DFIR discipline is an island. In fact, today it would be more surprising to work on a DFIR case involving evidence from a single source than it would be to cross the boundaries of disk, memory, malware, and network forensic analysis in a single day. Despite this inevitable crossover, we will primarily focus on the network side of your analysis in this class. However, with time, experience, and training, you'll certainly identify aspects of network forensics that will improve or be improved by other disciplines as well. Whether considering the vantage point of a single system's image (most directly addressed in SANS FOR500, FOR585, and FOR518), multiple systems' images (FOR508), memory analysis (FOR526), or reverse engineering malware (FOR610), or any of the other evolving aspects of a forensic investigation, you'll be well-served by building a deep bench of knowledge among a team or on your own.

When considering the network forensic subdiscipline we will focus on during this class, it's important to consider the evidence we'll use and how it's acquired. As you will see, working in the "live" and dynamic world of network forensics brings some unique challenges to the table.

References:

`http://for572.com/for500`
`http://for572.com/for585`
`http://for572.com/for518`
`http://for572.com/for508`
`http://for572.com/for526`
`http://for572.com/for610`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

The Blank Slate: Where to Start

- Common questions for the network forensicator
  - What data was taken (damage assessment)?
  - What was the system doing on the network before, during, and after an event of interest?

- How did malware get onto the system?
- What were other machines doing at that time?

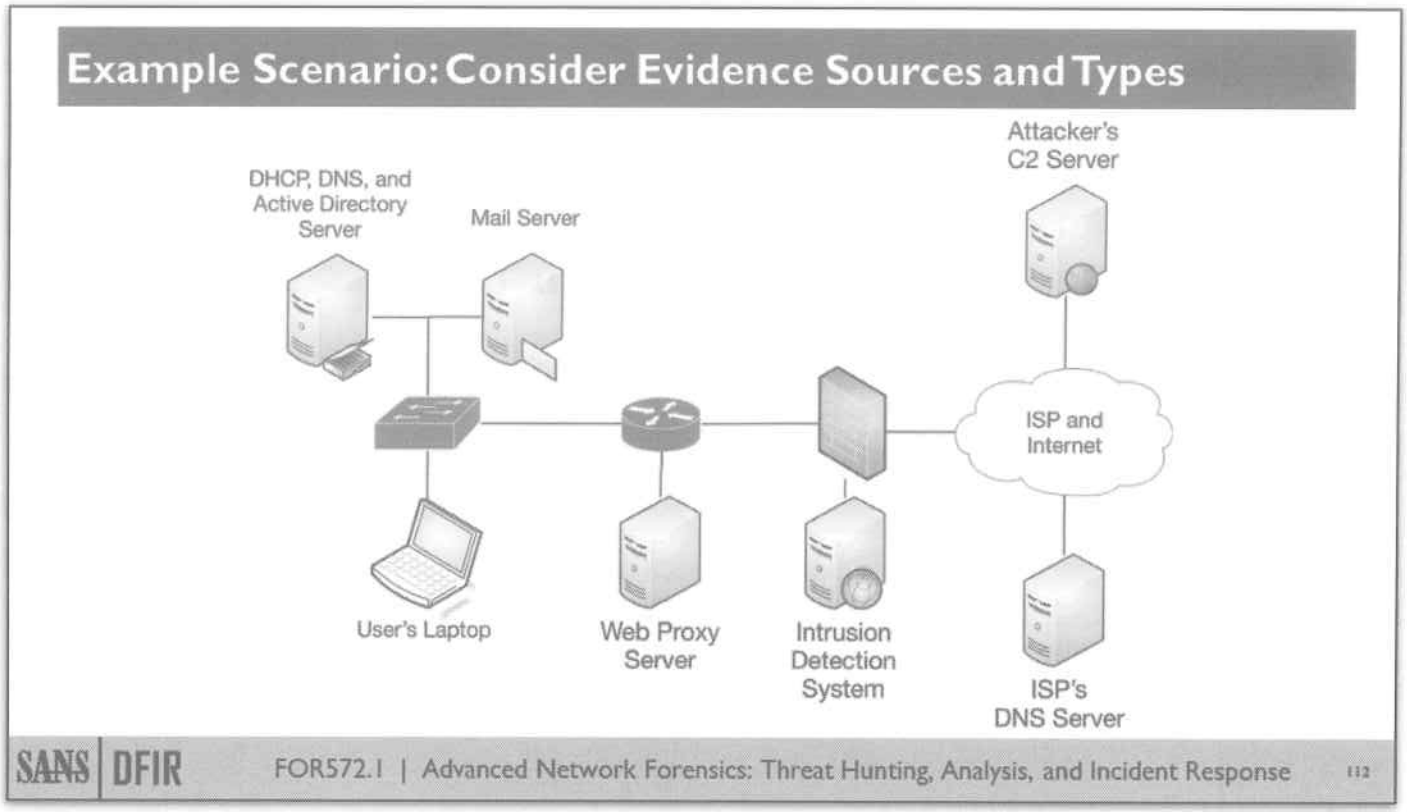
SANS | DFIR FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response 111

Perhaps the most important aspect of any forensic investigation or hunting expedition is where to start. Sometimes, you'll be fortunate enough to have a lead on which to follow up—other times, you'll be faced with the unenviable position of addressing a task such as "Something may be wrong here and we don't know what. We need you to run that down." This is not ideal, but it is still manageable if you frame your approach around some of the more typical investigative questions.

- Damage assessment: What was taken, when was it stolen, where did it go, how was it taken, and who did it? This is probably the most common family of questions any DFIR professional is asked to address. Victims tend to be most interested in (or are required to track down) details surrounding what data was stolen. Although attribution is not typically the most important question, knowing who your likely attack(s) may be will help you to understand their intent and capabilities, and hence what data they would be most interested in acquiring.
- Although we're clearly interested in learning about the compromise or theft events themselves, it is often helpful to learn more about the events that occurred immediately before or after an event as well. This can give the analyst helpful context about why other observations may or may not be important, as well as if there are any other systems in the environment that may also be compromised.
- One of the most common questions we're faced with is how a malware event came to be—did the malware land via a web-based drive-by download from a compromised advertising network? Was the user enticed to open an attachment or click on a link in a phishing or spear phishing campaign? The network perspective can often give clear answers about how the event was initiated, which can again characterize an attacker's intent and capabilities.
- Lastly, it may be exceedingly helpful to establish an understanding of what else was going on throughout the victim's environment at the time of a suspicious or known malicious event. This is an excellent method to hunt for adversarial activity in a process known as scoping. It's rare that a single observation is the entire extent of malicious activity—so looking at which other systems exhibit known or believed behaviors associated with an event of interest can provide an excellent window into how extensive a compromise may be.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Consider the systems involved in a common but potentially devastating event—the phishing email.

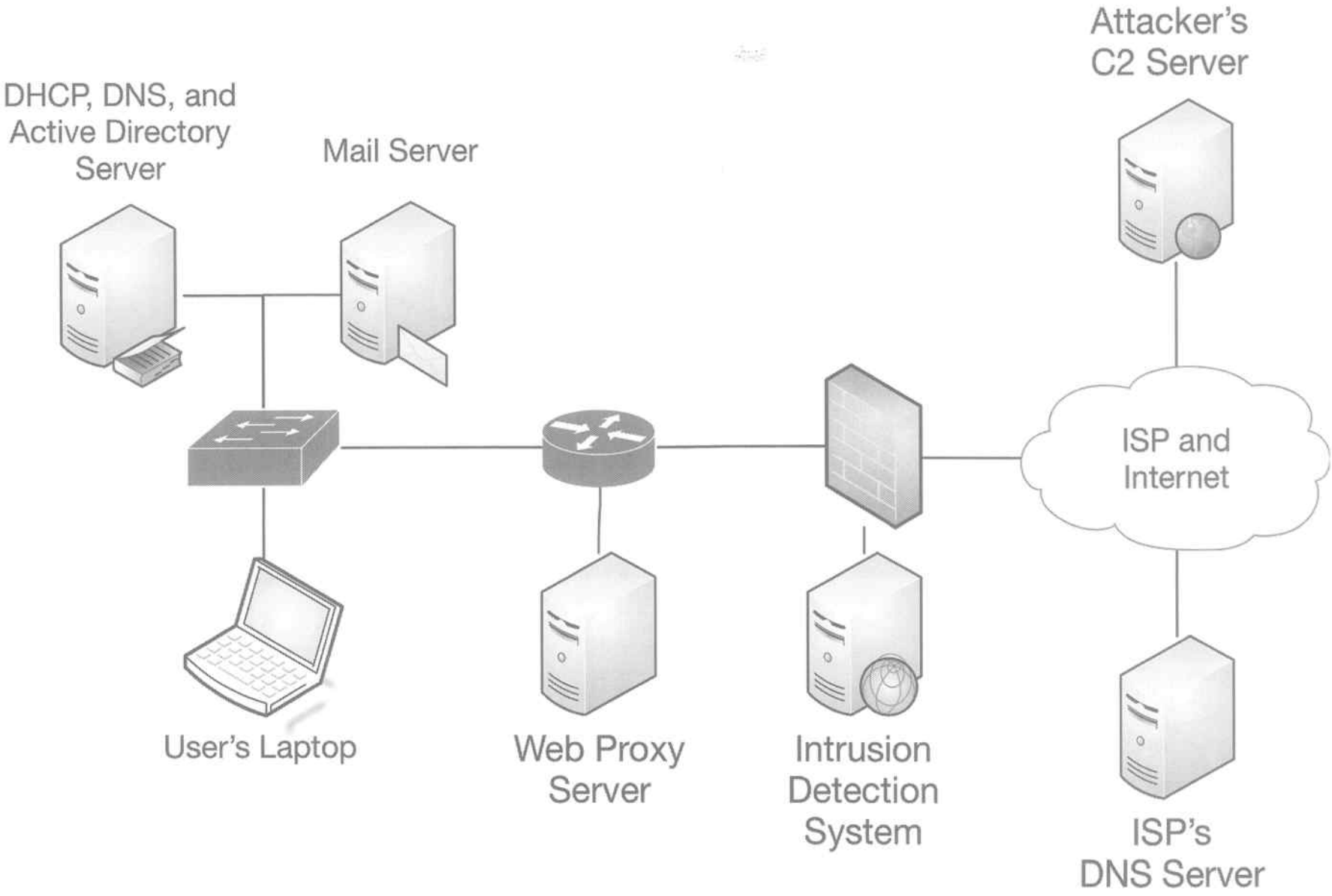
In this example, the user has just connected their laptop to the office network and has clicked on a link in an email that was delivered to his or her mail client software. This resulted in the browser downloading a file, a bespoke piece of malware from a targeted and well-funded attack; the malware was not quarantined by inline host-based barriers (a typical APT attack).

Breaking the communication down into steps, this simple scenario may result in the sequence of events below:

1. Laptop acquires IPv4 settings via DHCP
2. User logs onto laptop via domain authentication
3. User launches mail client
4. User clicks link in phishing email
5. DNS request and response for the link's URL
6. Phishing page serves drive-by download
7. User runs drive-by download binary
8. Drive-by binary retrieves second stage
9. Second stage automatically executed
10. Second stage looks up primary C2 hostname
11. Primary C2 IP blocked at perimeter firewall
12. Second stage looks up backup C2 hostname
13. Backup C2 communication established

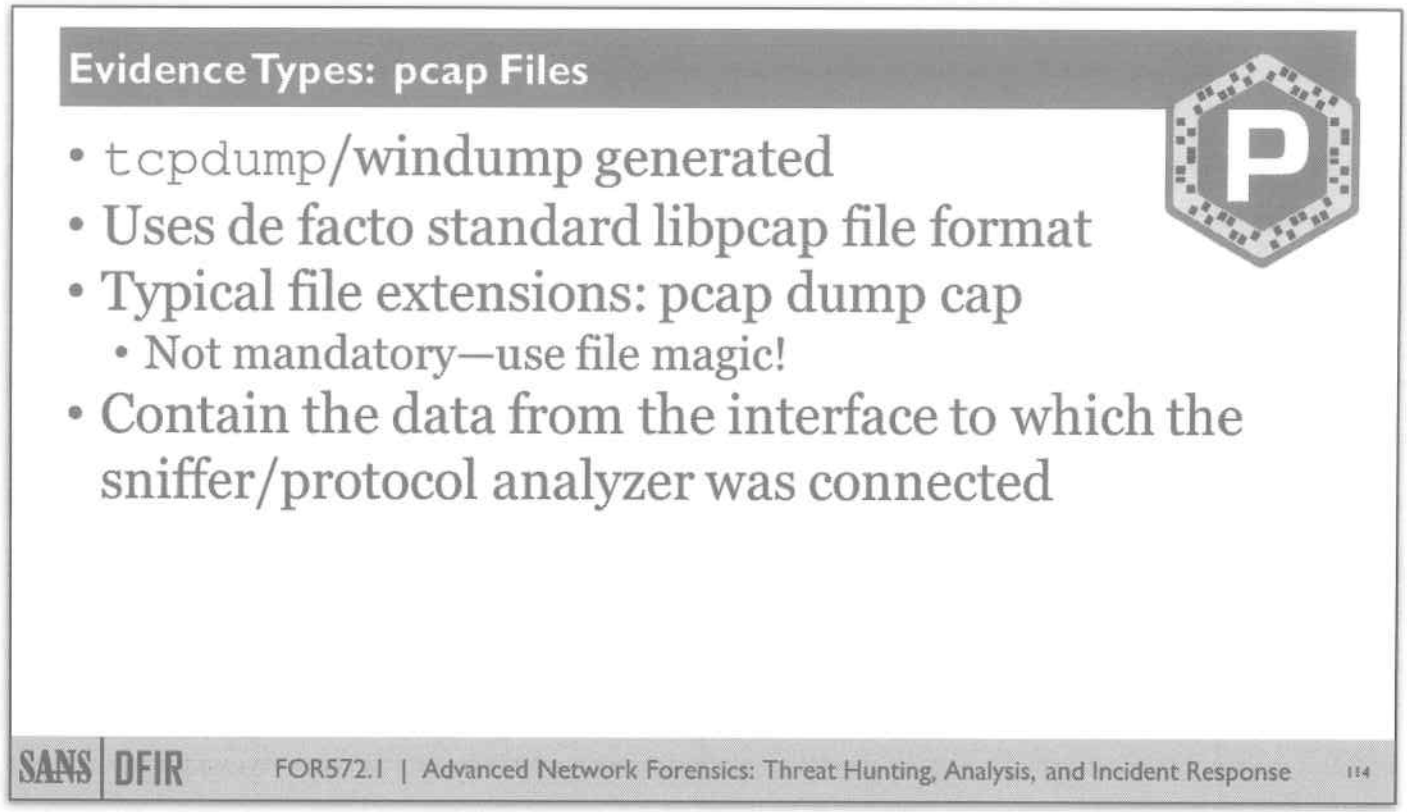
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



© 2019 Lewes Technology Consulting, LLC and Mat Oldham

https://t.me/learningnets



The term PCAP comes from **P**acket **CAP**ture and relates to the file format that the libpcap tool generates as it saves captured network data. libpcap, a Unix/Linux tool, has been ported to Microsoft's Windows operating systems in the form of the winpcap library[1].

PCAP files in this class use the .pcap extension but .dump and .cap are also common extensions. (Note that other capture utilities may use the same extensions, so verify file types with the "magic" values before attempting to load or parse them.)

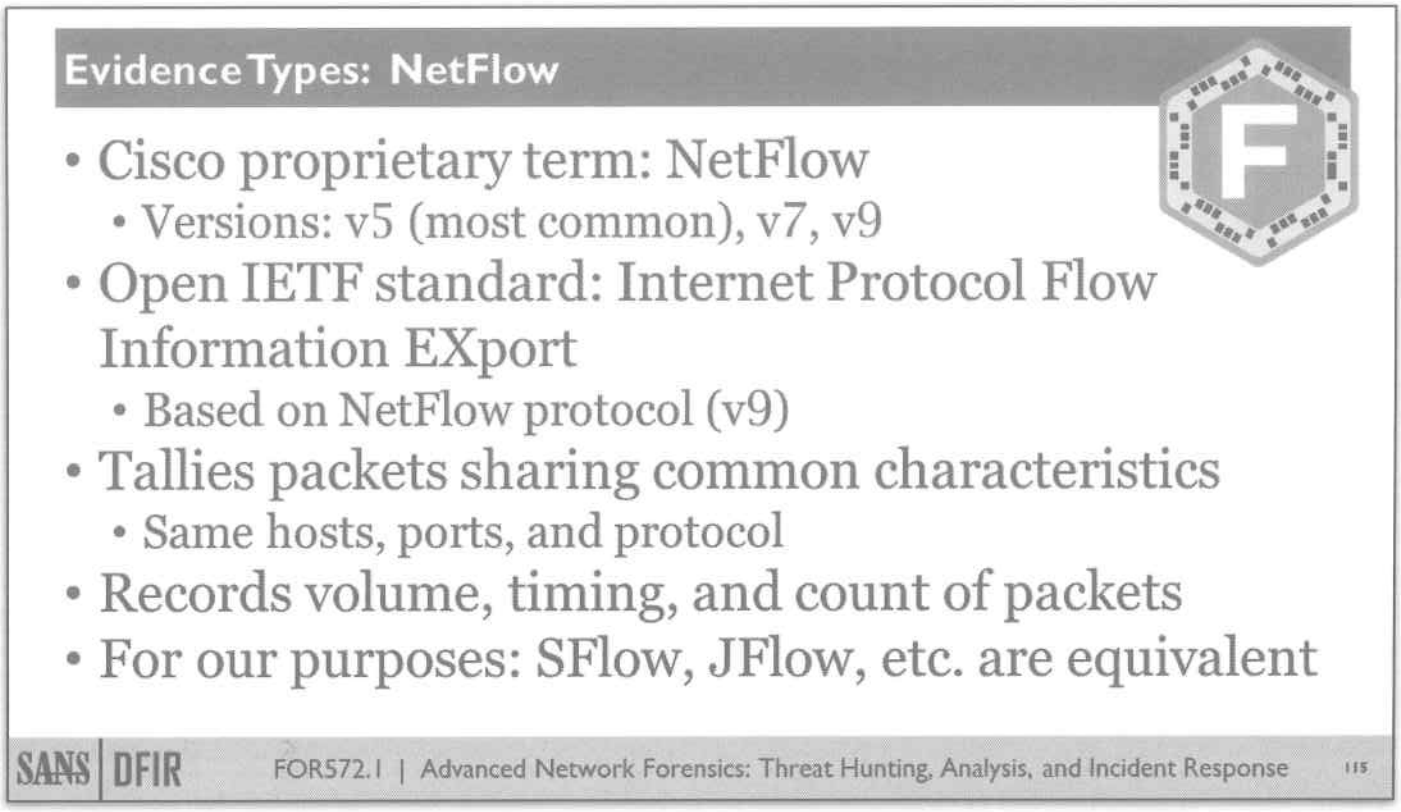
The sniffer will generate a file containing the data taken from whichever interface it was connected to; it is important that the analyst understands the type of interface and thus data the pcap contains. For example, if connected to a Linux WLAN interface in managed mode, the pcap will contain only data frames. If the WLAN interface was in monitor mode, the pcap would contain the 802.11 management frames as well as the data frames, but only for the WLAN channel that the interface was tuned to. The moral of the story is to understand how a pcap was generated and to what it was connected.

References:

[1] `http://for572.com/i7hke`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Cisco pioneered NetFlow technology, though the name has grown to be used as a common term for any flow collection architecture or technology. However, the IETF has created a vendor-neutral extension called IPFIX, which is covered by RFCs including RFC 5470, RFC 5101, RFC 5102, and RFC 5103. (Note that this course uses the colloquial term "NetFlow" to reflect actual Cisco-based NetFlow and IETF-based IPFIX for any protocol used to define IP flow data.)

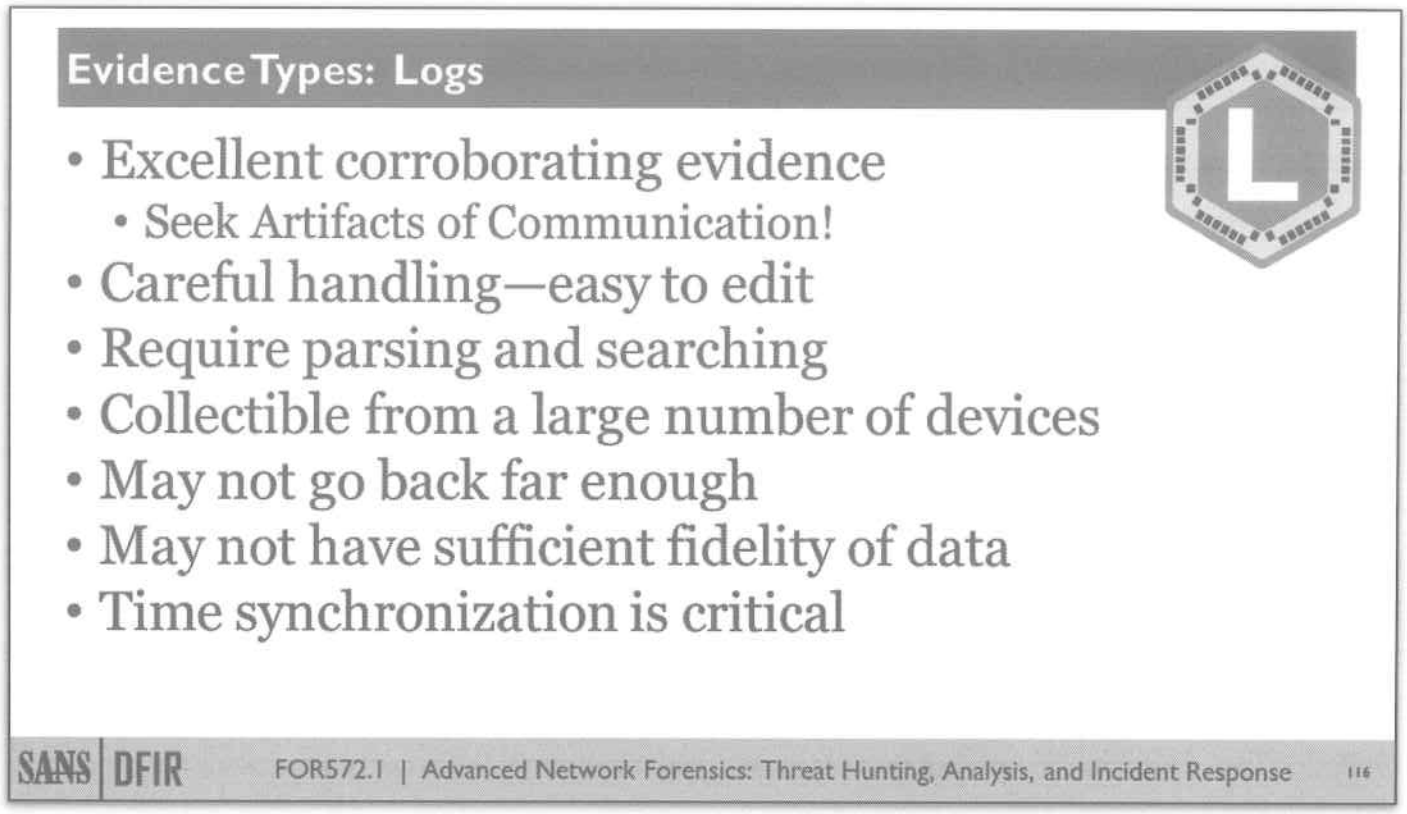
NetFlow data is generated from specified interfaces on a network device (called an **exporter**) that are configured to send NetFlow updates to a data **collector**. Analysts then query the collector (often with its own database and storage) with analysis software to generate reports for the user to review and interpret.

A flow is considered to be a communication between the same hosts using the same source and destination ports and the same protocol. Some protocols have additional identifiers. In the case of TCP, the initial sequence number is also used to identify a distinct flow. The flow record includes the number of packets transmitted and the total volume of data transmitted in the payloads of all packets.

NetFlow exporters and collectors need to be configured and implemented well in advance of any incident as they take considerable work when compared to enabling port mirroring and capturing pcaps. However, on large or high-bandwidth networks, IPFIX or NetFlow data can be of greater use than pcaps, as it contains connection metadata that is much smaller than the data itself. The biggest problem with high-bandwidth networks is that pcaps rapidly become unwieldy, difficult to move, and slow to analyze.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



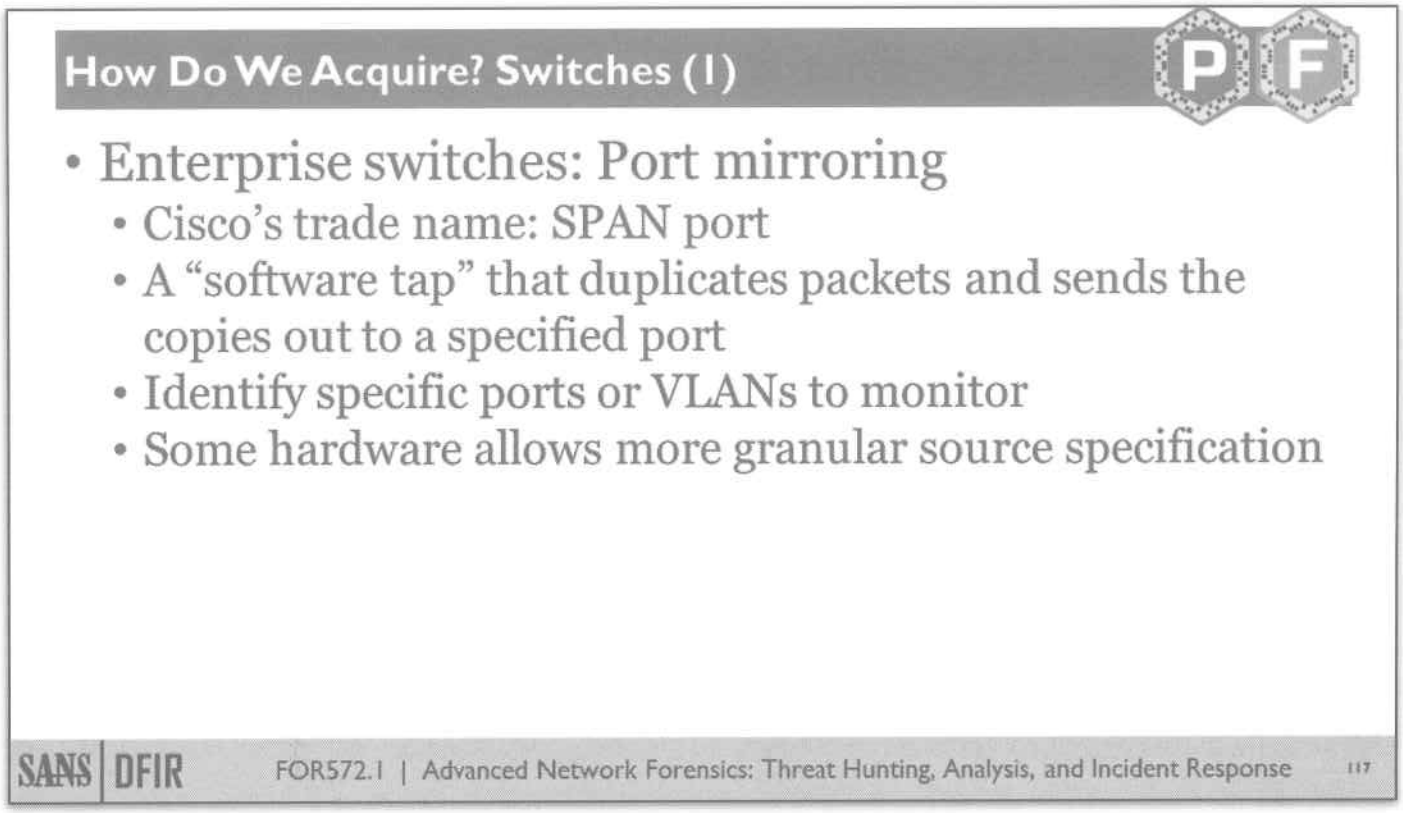
Logs from other devices can be excellent at corroborating activity observed on the network. However, they are easy to modify, so the analyst must ensure that hashes of log files are taken as soon as possible when they are received. Additionally, logs should be kept in a read-only repository where only authorized staff can access them.

It is common during an investigation for files to be trimmed. It is important that these activities are not conducted on the original files and that all subsequent saved edits clearly identify what was changed and the name of the original file.

Text logs compress really well, and a good logging solution will ensure a sufficient duration of logs is available to the analyst. For critical servers, it is recommended that you retain log evidence for as long as your data retention policies allow. Consider that breach-detection time frames often grow to one year or longer—and without evidence, it's all but impossible to run a credible investigation. Log quality is an important aspect, and in-house incident responders and forensic analysis should review the types and content of logs during the "Preparation Stage" (from the SEC504 "6 Stages of Incident Response").

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Let's take just a few moments to refamiliarize ourselves with the hardware methods and technologies we use to capture network traffic in the first place. We've already discussed the software side of things, where libpcap rules the roost. However, we need to establish a hardware-based point of presence on the network before running any capture software.

This is not a comprehensive review of all methods to capture traffic. We will not cover the more exotic technologies worthy of Hollywood movies (nuclear-powered undersea cable taps), or more recent ugly-yet-functional solutions (such as using a "dumb" hub to interrupt a link of interest). Instead, we'll focus on the more prevalent solutions in use today—those you would expect to see in a typical enterprise or consider for deployment in new monitoring/capture solutions.

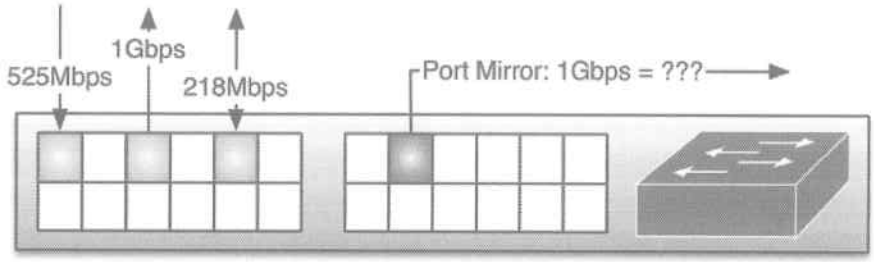
First, we'll address the venerable and ubiquitous switch. Because a typical switched network segments traffic to improve efficiency, we are limited to capturing a small subset of traffic that the switch has determined is needed on our specific switch port. However, enterprise-switching hardware provides "port mirroring" capabilities. The Cisco proprietary name is "SPAN ports" (Switch Port ANalyzer), so many network engineers will use this colloquial term. Port mirroring is basic, yet effective—this solution simply duplicates traffic from one or more ports or VLANs, directing the copied streams to a designated destination port. Some larger-scale switching devices also provide the ability to mirror traffic based on specific protocol characteristics.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

How Do We Acquire? Switches (2)

- Pro: Hardware already in place!
  - Minimize downtime
  - Simplify/avoid accreditation hurdles
- Con: Speed can suffer
  - Limited to capture at speed of half-duplex destination port



Perhaps the greatest benefit to using port mirroring is that the platform already exists in most enterprises—it's merely a software configuration tweak to activate the feature. The traffic of interest is automatically pushed to the designated port in addition to its existing flow. In environments where downtime is unacceptable, or network environments are particularly strict (e.g., due to accreditation issues), it's nice to have an option that doesn't ruffle too many feathers.

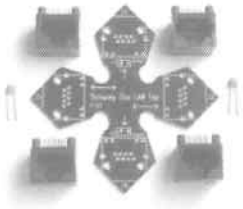
The major downside to using port mirroring is that while the switch's backplane may be engineered to simultaneously handle the traffic for all 24 ports at a full 1Gbps, the port mirroring port can still just handle its own measly 1Gbps. This means that if network throughput on the ports or VLANs we request be mirrored exceeds the available bandwidth of the mirror port, bad things(tm) can happen. This may lead to excess traffic being dropped from the mirror or to the switch entering a failure mode that disables the port mirror entirely. Each device may handle this differently, so it's critical to research the specific hardware you plan to use with the feature and be very cautious if you plan to monitor more than one port worth of traffic.

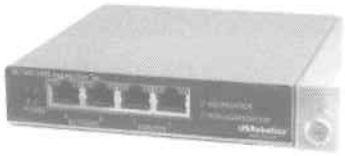
https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

How Do We Acquire? Taps (1)

- Hardware taps are a purpose-built solution
  - By design, all they do is duplicate traffic for monitoring and/or capture
  - May use one monitor port for each direction of link
  - Some provide multiple ports of aggregated traffic
  - Decrypting taps becoming more widely available





Because of the shortfalls associated with port mirroring on switches, the security community developed something better. A network tap, as the name implies, is a dedicated, single-purpose device that does an excellent job of duplicating network traffic so we can capture it.

Many tap devices use two ports to provide the duplicated network traffic—one for each half of the link. This means that the monitoring hardware must reaggregate the traffic to a comprehensive, full-duplex stream. Other taps perform aggregation, presenting a single full-duplex monitoring port so monitoring hardware can use a single link. A similar complication may be that the link itself consists of two unidirectional connections, meaning two taps are needed to fully monitor the link, and any collected data must be recombined after the capture is complete.

"Regenerating" taps provide multiple monitoring ports. These make multiple copies of each packet, so more than one monitor or capture device can be used at once. This is especially helpful when an intrusion detection system is in use at all times, but a pre-positioned tap point for as-needed traffic capture solution is also useful.

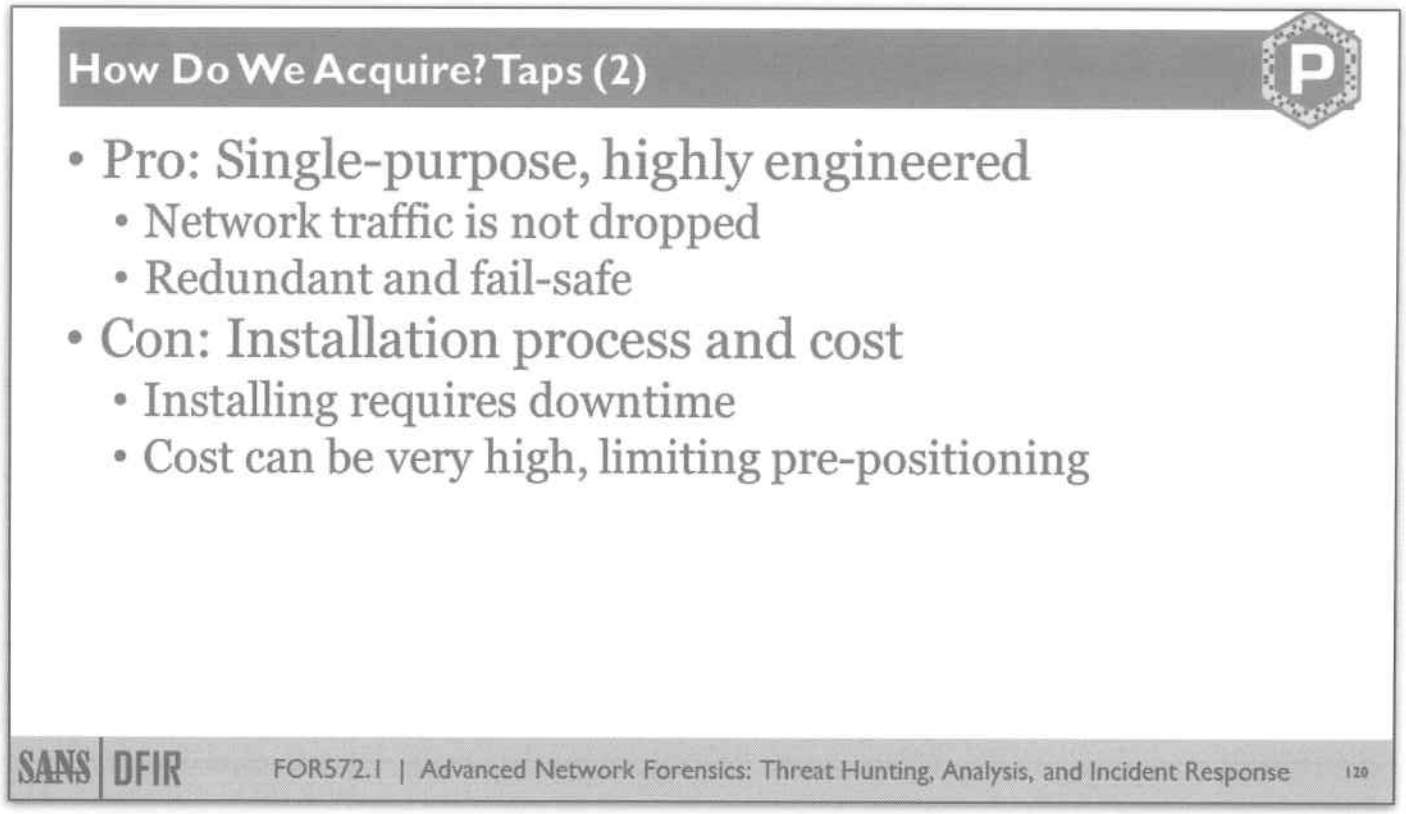
The Ninja Throwing Star Network Tap[1] is an interesting hardware variation on the 100Mbps split-direction network tap. Dualcomm offers several affordable tap platforms including 1Gbps[2] models. Depending on your budget and operational requirements, these may be good options for a tactical "Incident Response Kit".

In some enterprises, taps that also perform decryption capability, such as some from the GigaMon company[3], are being placed into operation. These provide useful insight and inspection capabilities for environments that require it, but also introduce a notable level of privacy risk and complexity.

References:

[1] `http://for572.com/sby16`
[2] `http://for572.com/0s6n7`
[3] `http://for572.com/rdsf8`

https://t.me/learningnets



The main benefit of a purpose-built device such as a tap is that it's engineered to do its job exceedingly well. When using the right tap for the task, you can be assured the hardware will not lose traffic. Higher quality devices may include features such as multiple monitoring ports to support additional activities, redundant power supplies to minimize downtime, and more. Most enterprise-grade hardware in this sector will allow traffic to pass across the monitored link even without power to the device. The monitoring ports would not function in that case, but risk-aware network engineers and CIOs will appreciate that a failure point has not been introduced to the environment.

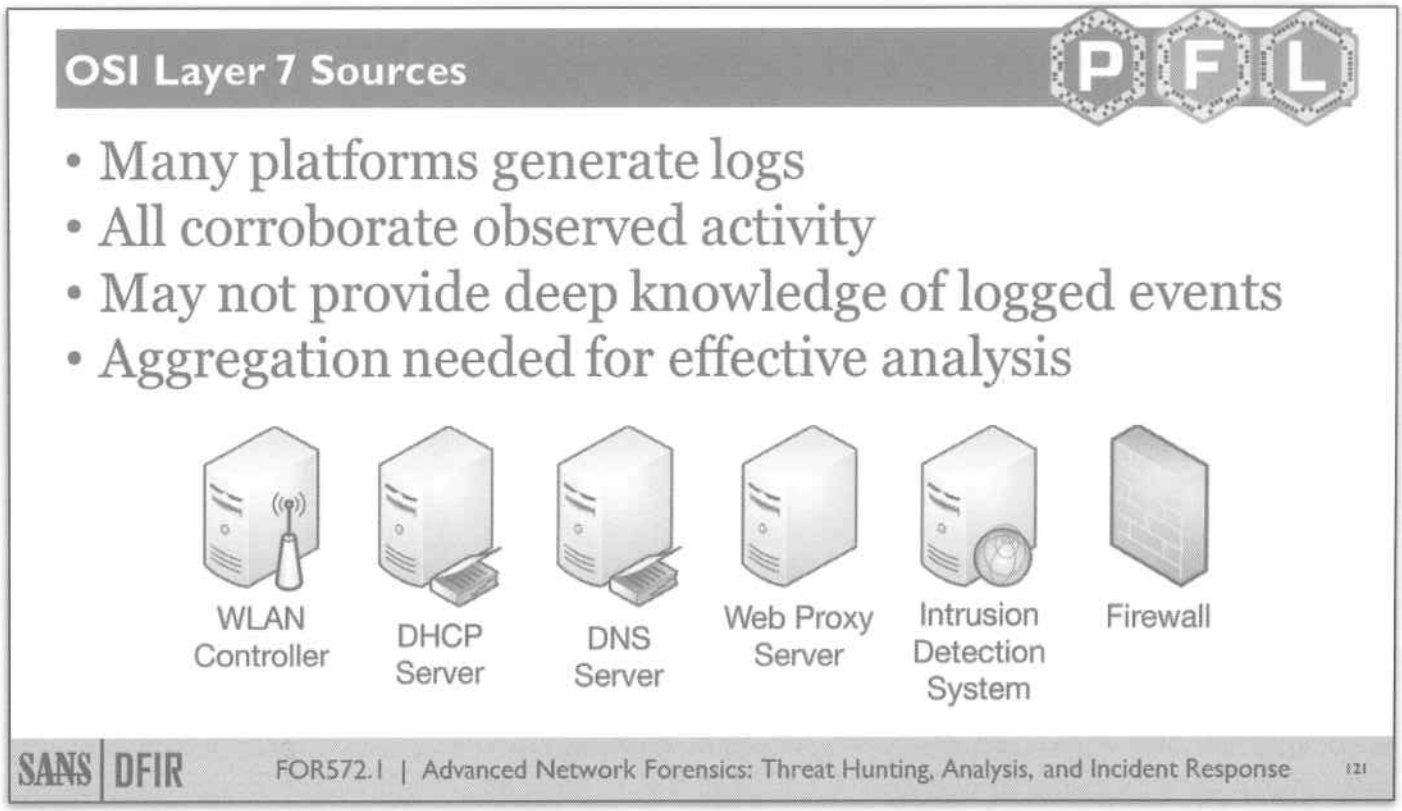
Some newer taps also provide higher-layer protocol filtering capabilities, which can help to narrow the scope of what is collected. However, this is a capture filter—and as we learned earlier, capture filters must be treated carefully. If a capture filter omits traffic, there is no going back to get it!

Of course, with great power comes large price tags. The major drawback on taps is the cost—it's not uncommon for a high-grade device to cost several thousand dollars. At that price point, scattering a few dozen taps around a network environment is not an option for most environments. This leads us to another drawback—installing a tap requires downtime. It may only be a few seconds while cables or fibers are rerouted through the tap device, but in some situations, even a minimal service interruption is forbidden.

Together, the cost and downtime constraints mean that an enterprise may want to pre-position a small number of taps at strategic points on the network, then plan to install additional devices where necessary, with the understanding that some downtime coordination will occur.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



In our example, the following devices can all provide evidence of network activity; however, getting access to useful logs is not always easy.

Each of the devices in the slide usually generates logs that detail the operations and data within its purview or scope.

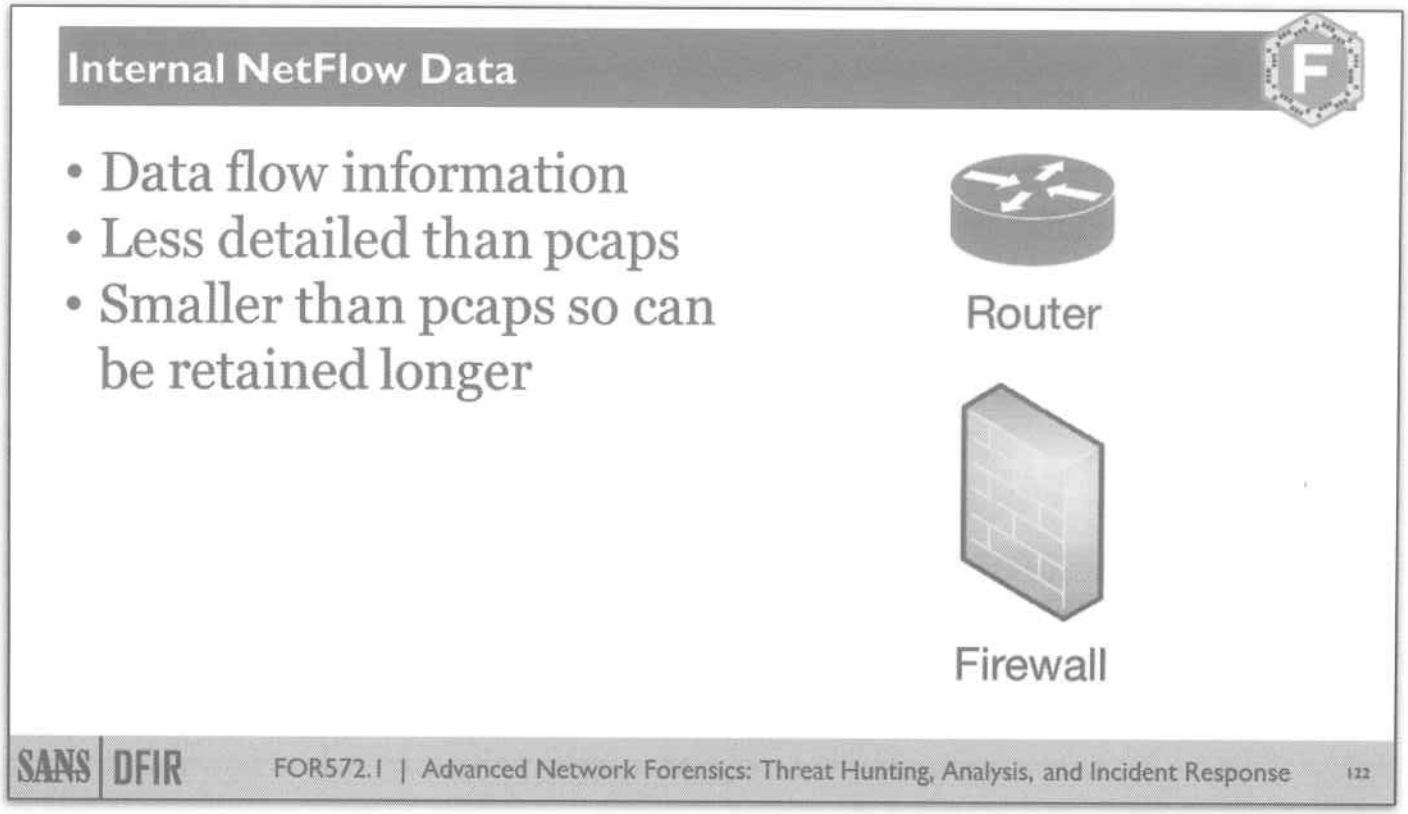
Many security devices include web interfaces that can make log export difficult. (MSSP-provided devices are particularly notorious for this.) Ensure you know what logs you can acquire, in terms of both quality and quantity, and have tested the extraction method in advance of any incident.

If you are not able to access the devices directly (e.g., for technical, organizational, or outsourcing reasons), ensure the managers/administrators are able to do so quickly and efficiently. Finally, if outsourced, develop a method to acquire logs that meet some basic standards, such as:

- Log data is secured, both at rest and in transit
- The transfer mechanism(s) accommodate the file sizes and types that will be sent
- The sender and receiver are comfortable and trained to handle the technology and processes that will be used to transfer the data

Getting staff members to do this on a regular basis is an excellent live training lesson that will highlight issues before they are critical to an investigation.

https://t.me/learningnets



Internal network flow records can be collected from various network devices, depending upon their functionality.

Most routers and firewalls have this capability, though it may be disabled by default. Some "enterprise-grade" switches with Layer-3 and Layer-4 visibility may offer export as well. Endpoint devices (workstations, servers, etc.) can also be configured to perform NetFlow collection using utilities such as `fprobe`[1], `pmacct`[2], or `nprobe`[3]. (Of particular DFIR interest, the endpoint approach can also be used in conjunction with a port mirror, to allow focused/tactical collection during an IR or hunting engagement.)

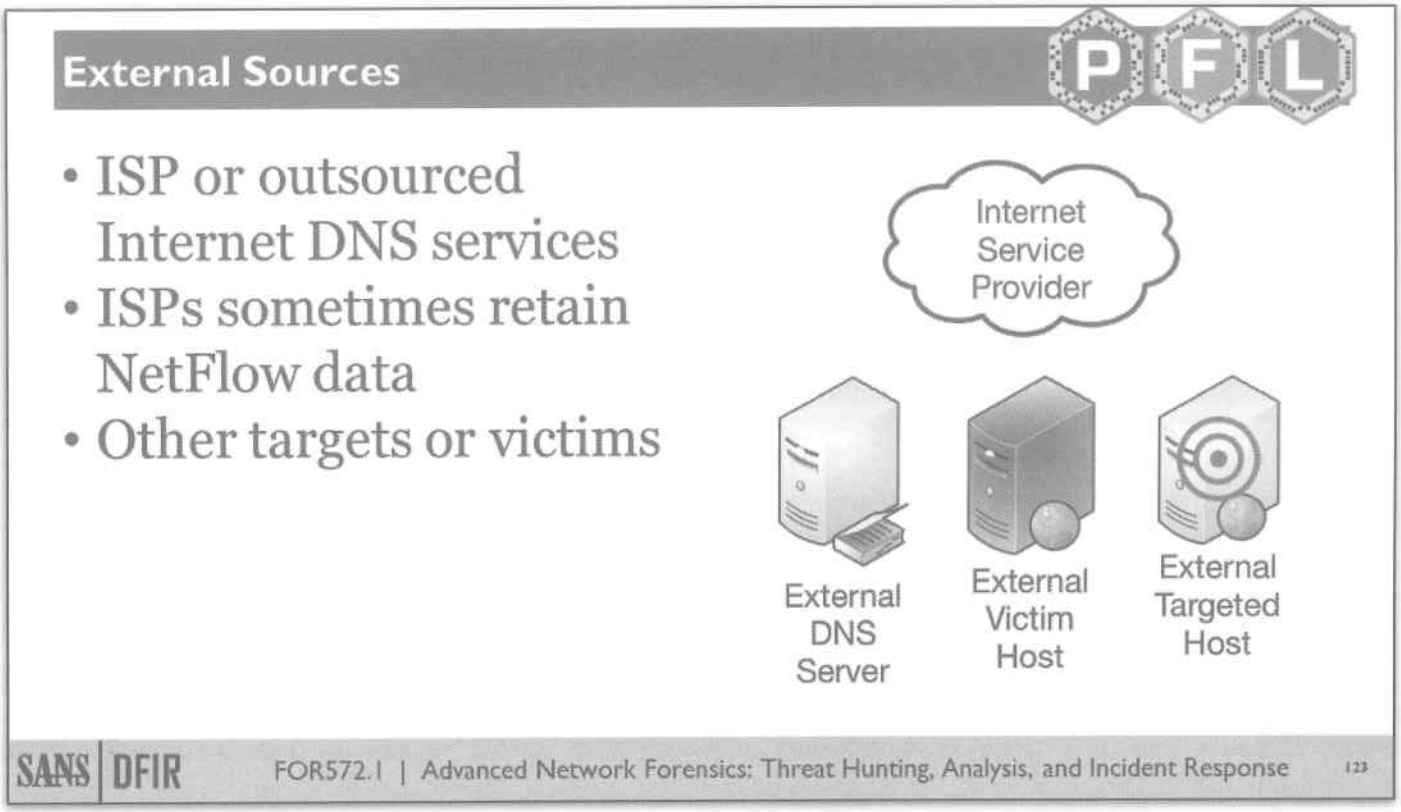
As we will explore in a future module, NetFlow collection can be focused where the most sensitive data is but is often also deployed on the perimeter. Because querying NetFlow is extremely fast, and no data is retained (decreasing the sensitivity of the collection itself), it offers an easy way to scope a compromise and evaluate whether newly identified indicators identify potential attack traffic in past collections. It can also quickly identify events of interest—such as large flows that suggest data exfiltration, large counts of frequent/small traffic bursts that suggest command and control, or known communication with identified APT IP addresses.

Whether used alone or in conjunction with other sources of evidence, NetFlow is one of the most valuable sources of network forensic evidence in a modern investigation. DFIR teams should seek NetFlow data and ensure its data retention policies adequately cover expected requirements for typical investigative tasks.

References:
[1] `http://for572.com/rzvq7`
[2] `http://for572.com/mu4t5`
[3] `http://for572.com/105zd`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Other sources of evidence can be external to the organization. These are more common than you would expect.

Your ISP may collect NetFlow data from the routers and boundary devices within their operational control. By developing a good relationship with the ISP, you may be able to negotiate access to the data from your connections. If you use just one ISP, this can provide details of all data that went into and out of your network! That would be invaluable data when attackers have destroyed other logs, your devices did not record the activity, or the logs have been truncated due to a retention policy. However, you shouldn't exclusively rely on external evidence—collecting your own NetFlow is a critical component of a solid IR program.

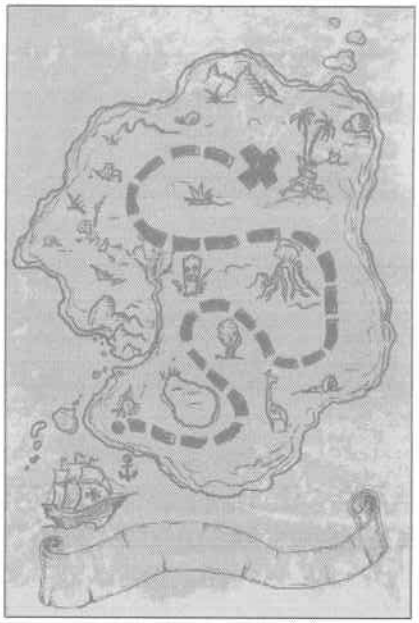
Externally-compromised systems that are being used to compromise your network can provide some intelligence as to malicious activity; however, legal advisors usually prevent the disclosure of detailed logs between separate organizations. Similarly, external organizations that are attacked ***from*** your infrastructure can sometimes provide details of source ports, external IP addresses, times, and packet contents. Although it is unlikely that you'll want to start a relationship with these external sources during an investigation, the use of the various industry-centric Information Sharing and Analysis Councils (ISACs)[1] may provide a useful framework for these important conversations and intelligence-sharing arrangements.

Finally, because DNS traffic has become so critical in both scoping and detailed investigations alike, having a means of examining all of your DNS requests and corresponding responses can be a critical part of an investigation. Although forcing the use of one or more internal DNS resolvers allows easy visibility to this critical data, some external providers may also give their customers useful visibility to their DNS activity.

References:
[1] `http://for572.com/tenwr`

https://t.me/learningnets

Network Data Collection Planning: Ideal Case (1)



- Design capture and visibility into the network
  - Strategic infrastructure planning
  - Pre-position capture/collection devices at points of interest

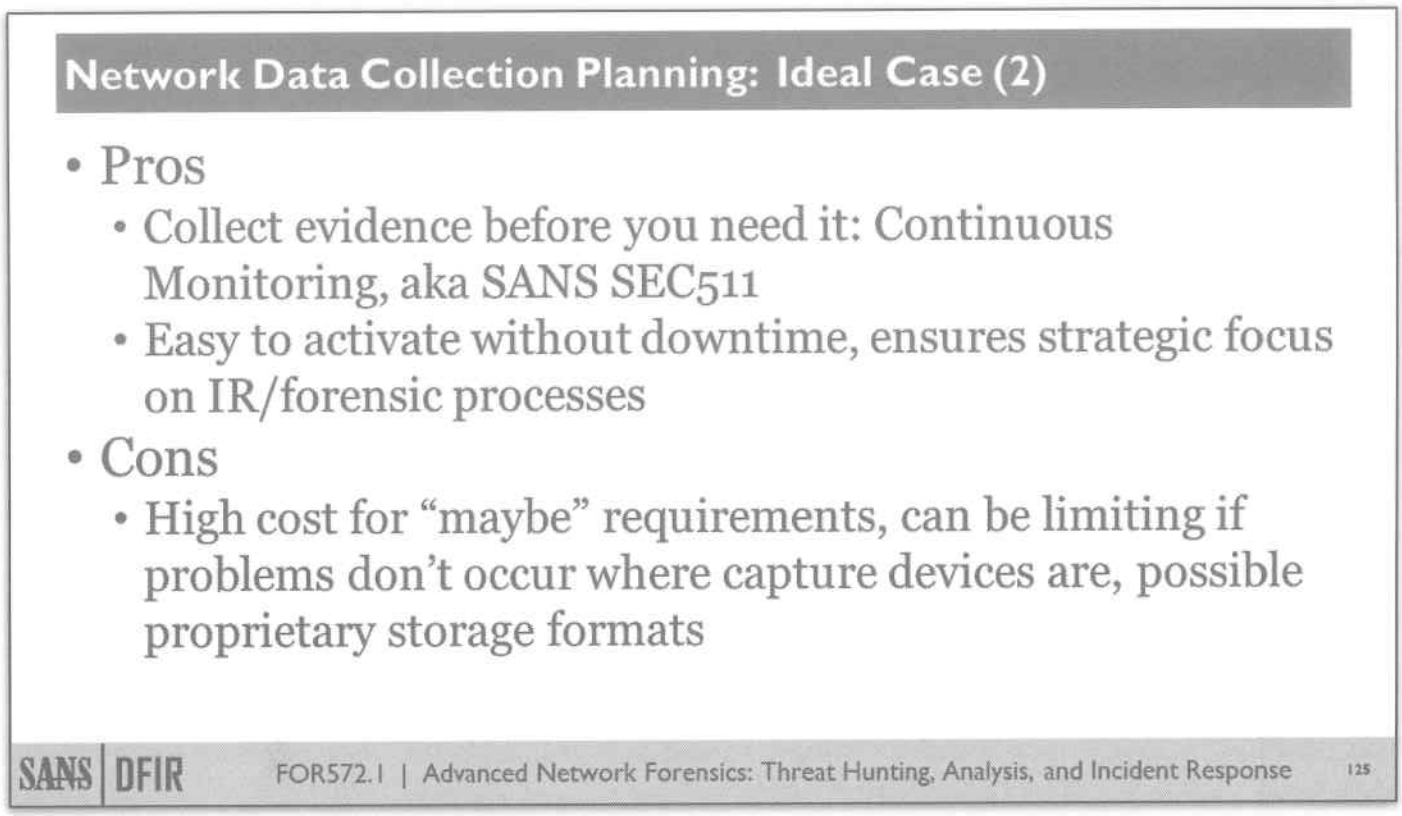
SANS DFIR FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response 124

Let's first look at the strategic, baked-into-the-network solution. Under this scenario, the security teams would assist during the design phase, helping to identify what network segments would provide a relevant or useful vantage point during anticipated security incidents. The easy example here is in determining where to place a web proxy server to handle internal users' requests—and of course, mitigating the opportunities for users to bypass the proxy function. However, as we'll discuss in a moment, there are a variety of network-based devices and appliances that we could use to improve our awareness of network activity during an investigation.

Obviously, this scenario is best suited to new deployments or fundamental network overhauls, but retrofitting such a function into an existing network is also possible.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



One key benefit to this approach is that network monitoring is either ongoing or can be started without downtime. It's unfortunate but not uncommon to see a victimized organization delay critical incident response activities because inserting a network monitor onto a particular link would incur several minutes of downtime, which may be an unacceptable business or mission impact. This approach also has a nice side benefit: It ensures that network engineers have some level of focus on secure design from the start of the project. Such a focus should result in more defensible networks overall.

A primary drawback of this approach is the cost. Such enterprise-grade solutions typically come with a large price tag that can be difficult for project managers and budget owners to stomach. Complicating the process is that these are largely "insurance purchases." Their value can be hard to state until it's too late, so deploying a proper solution can be a difficult case to make to budget owners. Fortunately, we currently see there is increasing support for proper security design principles, and hope to see that trend continue. Another possible drawback is that we can't possibly predict the extent or nature of a security incident ahead of time. A secure network design can address a wide variety of situations, but we should never expect that strategic planning alone can address all possible requirements.

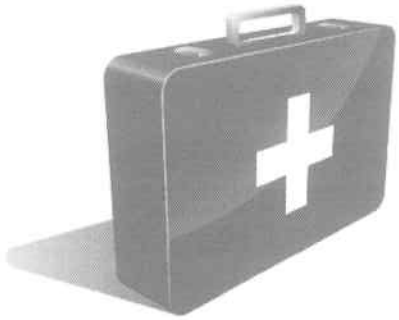
Finally, it's possible that enterprise-grade solutions use proprietary data storage formats that could be constraining in the future. Although many/most of the solutions we'll discuss in this section include capabilities to export data into common formats, the possibility of "vendor lock-in" tends to be higher with enterprise-grade solutions.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Network Data Collection Planning: Acceptable Case

- Standby hardware and good plans
  - Prebuild capture platforms to address dynamic requirements when needed
  - Pros
    - Flexible deployments, training on the actual gear to be used, cost typically lower
  - Cons
    - May need coordination and downtime prior to activation



A second planning case we'll consider includes the development of one or more "standby" capture systems that can be deployed when needed. Of course, any "ad-hoc" solution like this also needs documentation to address the who/why/how parameters for its use.

The primary benefit for this kind of solution is in its flexibility. Because they can be placed during an incident at the point of interest on the victim's network, we're better able to ensure collection of data relevant to the investigation. Also, security personnel can more easily train on the actual hardware they would use during an investigation, and refine processes and even the platform itself to address emerging threats. This is not always easy to accomplish when expensive platforms sit on a production network. Because these standby platforms can be more tactical in their design, they typically don't have such a hefty price tag, either.

However, deployment of these tactical capture platforms can sometimes be more complicated. If downtime to a production system is required before it can start collecting packets, then business and mission impacts may outweigh the immediacy of placing it into operation. (I have personally seen victims of serious breaches delay sensor placement for as much as five days.)

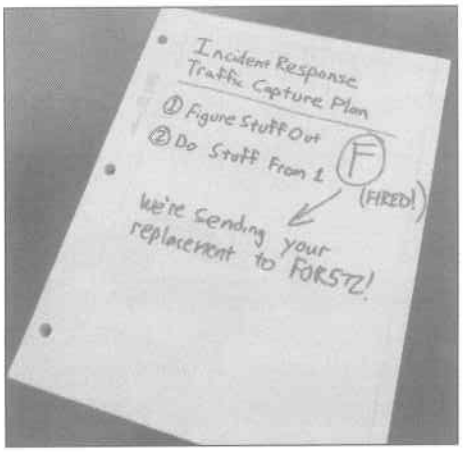
Incorporating this kind of solution into the process helps ensure security teams can flexibly respond to unforeseen requirements without requiring huge up-front and ongoing investments.

https://t.me/learningnets

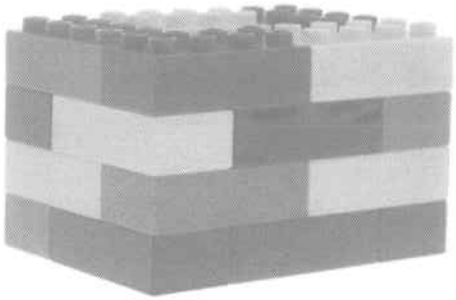
© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Network Data Collection Planning: Worst and Realistic Cases

- Case never:



- Realistic case:
  - Hybrid solution



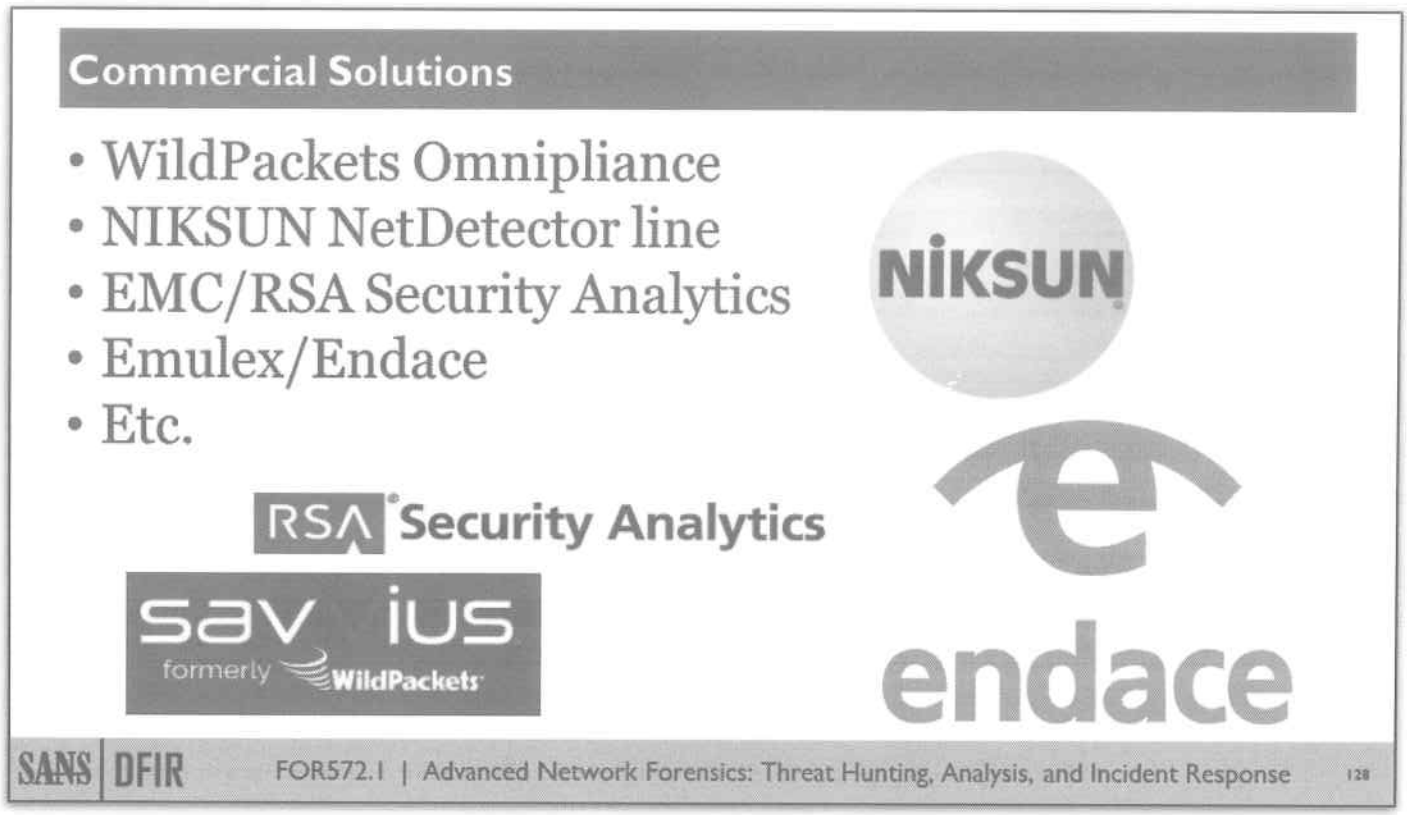
SANS | DFIR FOR572.1 | Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response 127

One design case that we should **never** pursue is to just "figure it out if we ever need to". All that you've heard about building airplanes while in the air, etc., is true. It's a bad idea. A really bad idea.

More realistically, though, we can leverage a blend of these two design cases to best meet our known and unknown requirements, while also managing cost to a reasonable level. For example, one may want to pre-position capture and visibility platforms on the perimeter as well as in network enclaves where the organization's most sensitive data is processed. Then, by maintaining a few ad-hoc platforms that can be deployed quickly across the environment, the DFIR team can quickly mount a meaningful IR action, or deploy a hunt team to various positions within the environment as needed.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Network security (and, by extension, network forensics) is big business, so there is no shortage of commercially available solutions that claim to (and often do) provide a great deal of functionality to their customers. This is just an abbreviated list of solutions available on the market today, and URLs with more information on each are included below.

The assortment of tools listed here may meet your organizational requirements, and they can also provide network capture functionality, including full packet content. Many such solutions are marketed as "network flight data recorders", which is a reasonable analogy. At the core of each type of device is the ability to acquire and store packets. A variety of additional analytic functions is also available, depending on the specific platform.

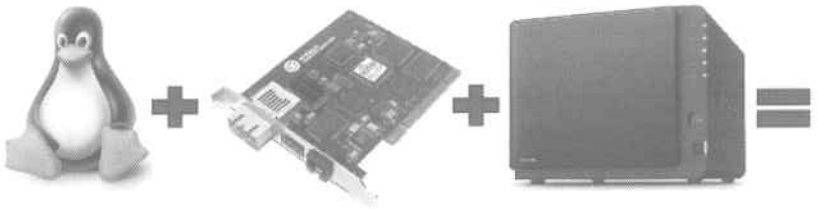
- Savvius (formerly WildPackets) Omnipliance TL Network Analysis and Recorder Appliance
  - `http://for572.com/y9306`
- NIKSUN NetDetector line
  - `http://for572.com/xkas-`
- EMC/RSA Security Analytics (including the former NetWitness product family)
  - `http://for572.com/wch5k`
- Endace Systems product line
  - `http://for572.com/crbox`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Homegrown Capture Platforms

- Have time and expertise?
- Can provide excellent performance at a reasonable price
- Tailor platform to specific requirements
  - Data retention policy, hardening standards, budget, bandwidth, etc.



Although the enterprise-driven capture appliances are excellent choices, sometimes we need a more tactical, customized, or cheaper solution. In this case, building your own capture platform is actually a simple task if you consider the specific requirements the task demands.

The trade-off, of course, is that what we gain in cost savings comes at the "expense" of the time and expertise required to build and maintain the platform. (The saying "Linux is only free if your time has no value" [1] applies here!)

That said, a homegrown platform can be tailored perfectly to your specific requirements. Before building such a platform, we would want to consider our requirements—both technical and non-technical.

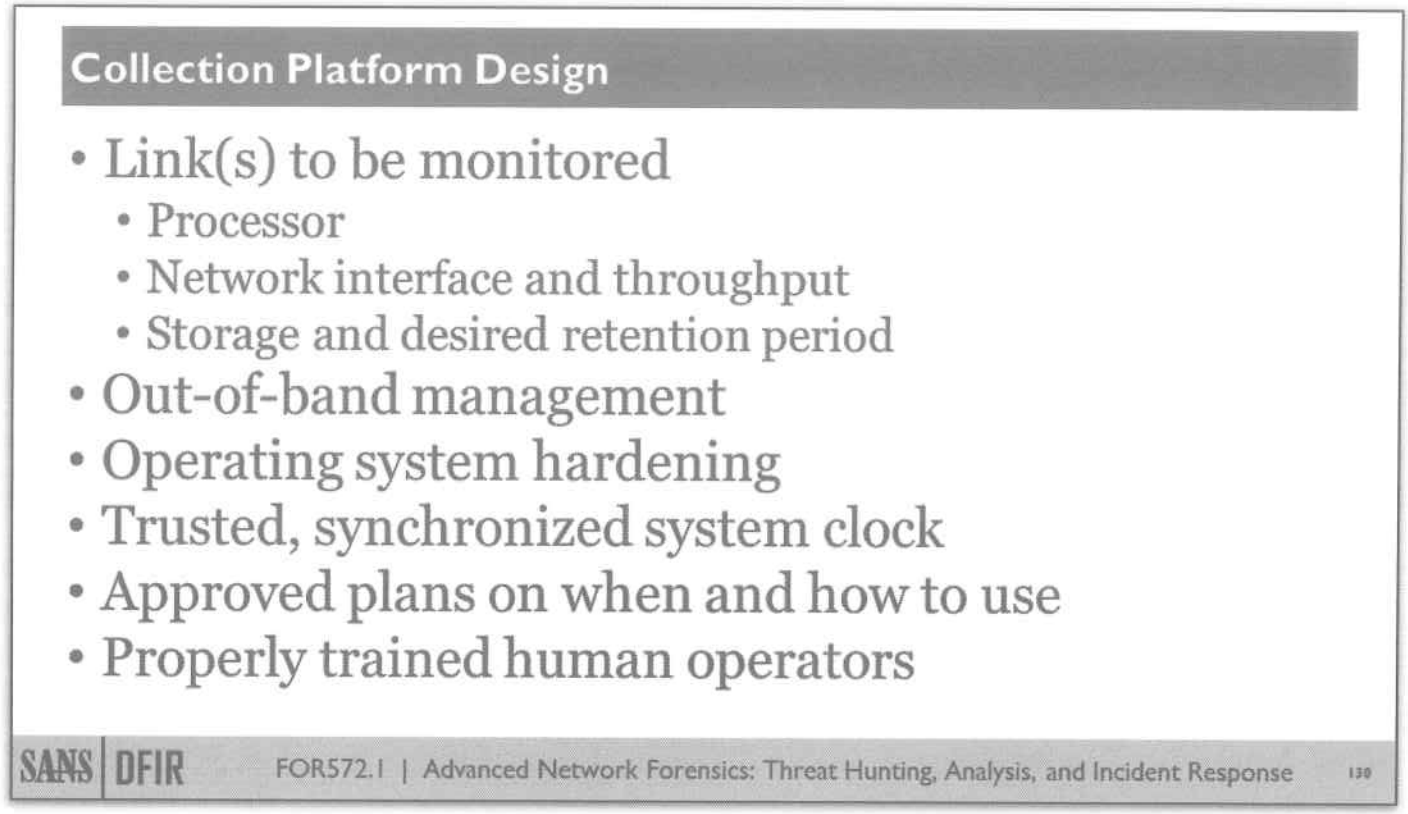
Several SANS Gold Certification papers have proven to be excellent resources for building these platforms in both lab and operational environments. FOR572 alumni have contributed great work to the community. Derek Banks wrote, "Custom Full Packet Capture System"[2] and Gordon Fraser wrote, "Building a Home Network Configured to Collect Artifacts for Supporting Network Forensic Incident Response"[3]. More such papers and related useful resources are added to the FOR572 course notebook online as they are identified.

References:

[1] `http://for572.com/-xman`
[2] `http://for572.com/i59j1`
[3] `http://for572.com/zko73`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



A key requirement will be the links from which our platform will be capturing network traffic. We want to ensure that our capture card—whether a common network interface card or specialized capture card—matches the tap or other device we'll be using. Recently, the use of SFP transceivers has enabled flexible configurations that can accommodate multiple Layer 1 form factors without the need to replace hardware. The system libraries used to perform the acquisition are also a factor. For example, without extensive tuning, `libpcap`'s `AF_INET` kernel interface may not be suitable for 10Gbps+ networks (or even heavily loaded networks at slower speeds). However, raw socket interface via the `AF_PACKET`[1] method has become a preferred solution that is better suited for this level of bandwidth, and `PF_RING`™[2] may be another solution to consider. As with any system engineering effort, testing is required to determine the best solution for the problem at hand.

Typical system-level considerations apply as well—the processor and RAM should be scoped to match expected load. Storage becomes a major concern, however. We need to calculate the necessary storage based on link speed, typical and peak link throughput, and required data retention period. The storage bus speed can also be a constraining factor—remember that USB 2.0 maxes out at just 480Mbps and USB 3.0 can theoretically handle 5Gbps. Remember (and test) storage speeds before planning to capture from a saturated gigabit Ethernet link! (Also: "max" seldom means "actual"!)

We also need to consider how we'll be managing the capture platform. In some cases, direct access may be the only permissible option for security, regulatory, or other reasons. In others, perhaps an appropriately-secured remote access method such as SSH or HTTPS may be permissible or even required due to geographical separation. Regardless of access methods, the operating system should be hardened to meet any local policy requirements and good common sense.

A requirement that's less visible, but absolutely critical, is that the capture platform's clock is accurate and synchronized to a trusted source. It may not warrant a dedicated GSM or GPS clock device, but a proper NTP configuration will prevent many headaches during analysis.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

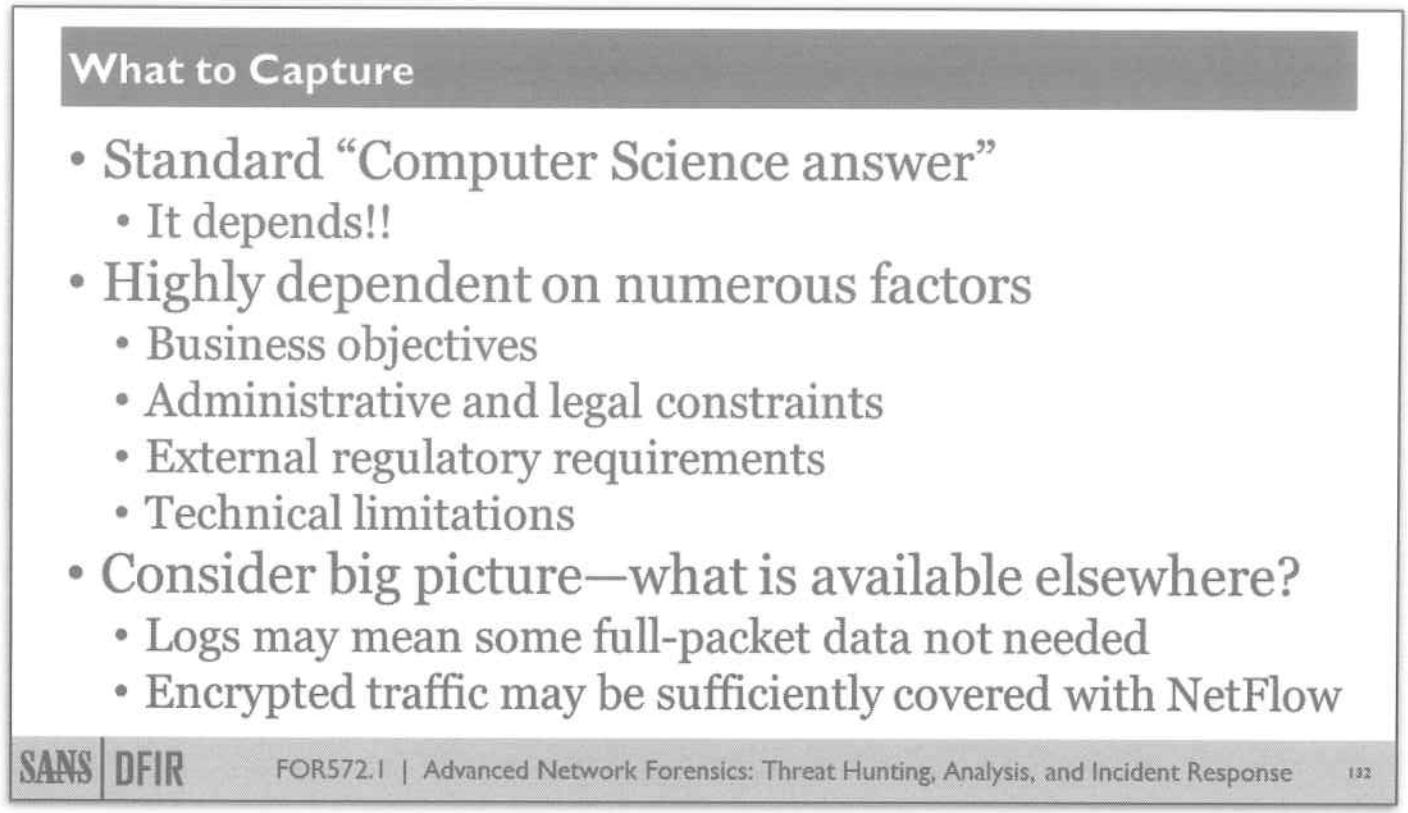
On the non-technical side, a documented, approved plan on when and how to use the capture platform is critical. Because capturing network traffic has specific legal requirements and caveats, it's always important to create a standard plan that has been approved by the appropriate management personnel. Of course, after that plan is in place, we'll also need to follow it! Finally, it goes without saying that employees who will be tasked to carry out the capture activity should be trained on both the equipment and the plan.

References:

[1] `http://for572.com/d2noi`
[2] `http://for572.com/abzg9`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Regardless of what platforms, software, or hardware will be used to acquire network data, there is still a fundamental question about what to capture with all these toys. We quickly realize that there is no single good answer to this question—a familiar outcome to many "What's best?" questions.

First, consider the various influences on what should be captured. Whether you're a part of a multi-billion-dollar international corporation, a "Mom-and-Pop" small business, or a governmental agency, our work supports the business (or mission) objectives set by the powers that be. Unless our actions directly contribute to the organization's core reason for being, these activities quickly become a pure cost, and will surely receive heavy scrutiny, or be subject to cancellation.

For that reason, we ask the core question: What does the organization need from a network capture solution that will support their core objectives? This may not be a very technical aspect of such a solution, but it's at the root of every business decision. Ignoring this factor would be a mistake.

Aside from the business impact, we must also consider administrative and legal constraints. If, for example, the organization's employee privacy policy guarantees each user on the network an expectation of privacy with regard to their email communications, it would be inconsistent with this policy to capture some or all email-based data—at least without a policy change. Similarly, if a private organization capturing and storing full-packet data from its employees is flatly illegal in a certain jurisdiction, it would be a bad move to attach a network tap on the main Internet link and start storing pcap data to disk. Another aspect of these constraints is that some countries, industries, or organizations are subject to certain record-retention policies that limit the amount of time different types of data can be held. Although a security-minded individual might prefer to retain all data indefinitely, they may have to accept a policy that limits that retention to a few days or weeks. (This applies to almost every type of evidence we will discuss—not just pcap data.)

Aside from the non-technical concerns, we must also include factors such as available disk space, processor and memory resources, typical network-utilization rates, link speeds, and related details in determining what to capture and how much resulting evidence to keep on hand.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

In selecting what network data we may want to acquire with a capture solution as discussed in this section, it's quite likely that "tcpdump everything to disk" is overkill. Take a step back and survey the broader environment—consider what else is already available that might minimize the amount a packet capture solution would need to pull from the wire.

For example, consider the vast amount of evidence available and sound findings made during the proxy walkthrough earlier in this section. Without touching a single pcap file, we clearly identified a subject's web activity, potentially recovering the data he or she uploaded to a suspicious site. In such an environment, perhaps packet captures for all proxy-based web traffic would be redundant. Depending on the volume of such traffic, eliminating it from the packet capture might result in significantly lower storage requirements.

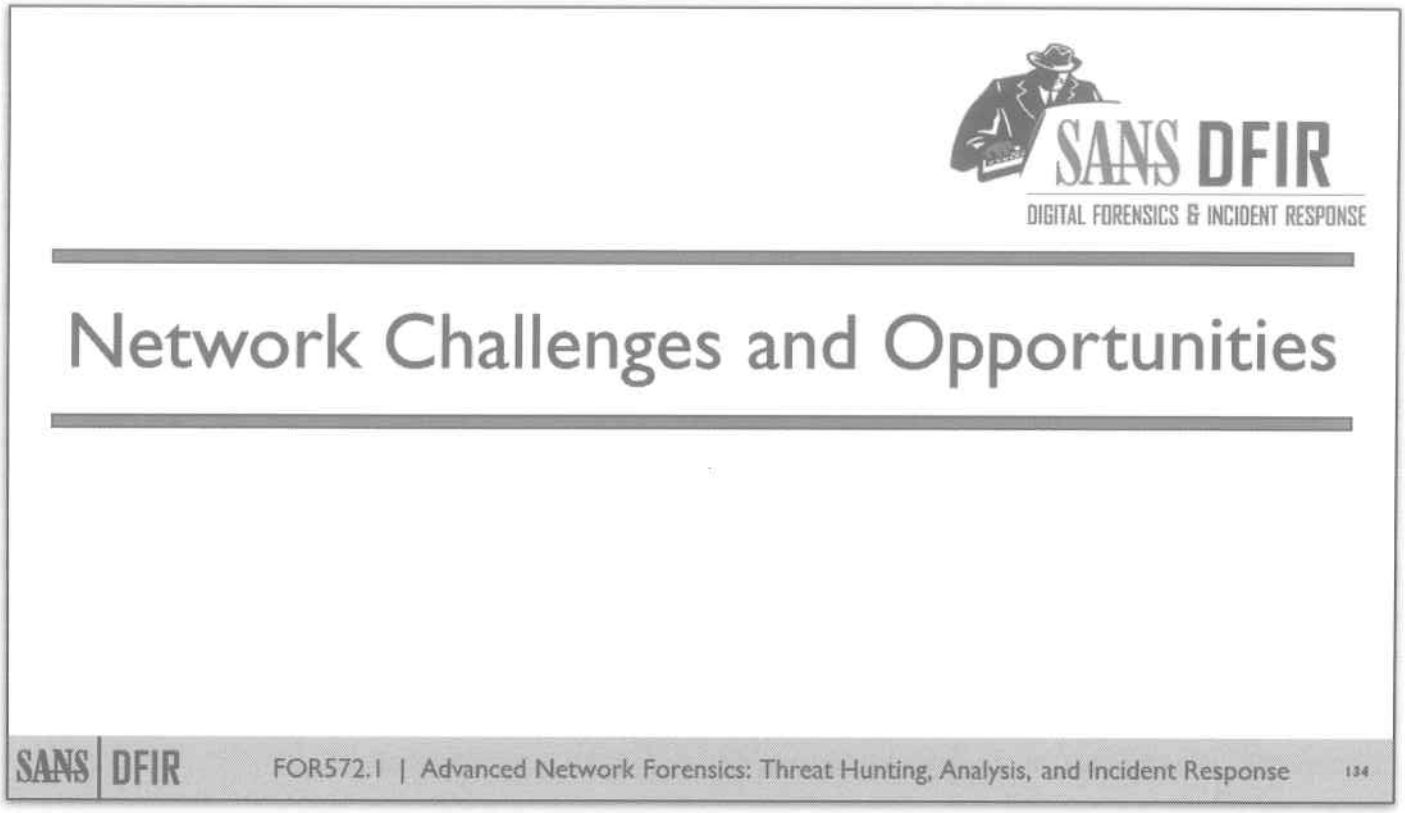
Imagine a log file that contains every DNS query and response observed within an environment. In many cases, this is a sufficiently detailed body of evidence to provide context to other network activity that occurs after the hostname resolution process. Logging the raw queries and responses to pcap files is generally unnecessary with such logs.

We could discuss hundreds of different types of log files and their potential value in offsetting pcap-based capture and storage requirements. Without diving to that level, though, it's sufficient to state that log files often contain pure evidentiary gold. Because they compress well and are usually covered by existing "business records" laws or policies, the amount of available log data should be a primary factor in identifying the type and amount of data a full packet capture system acquires.

NetFlow collections can also provide excellent visibility, especially for encrypted communications.

https://t.me/learningnets

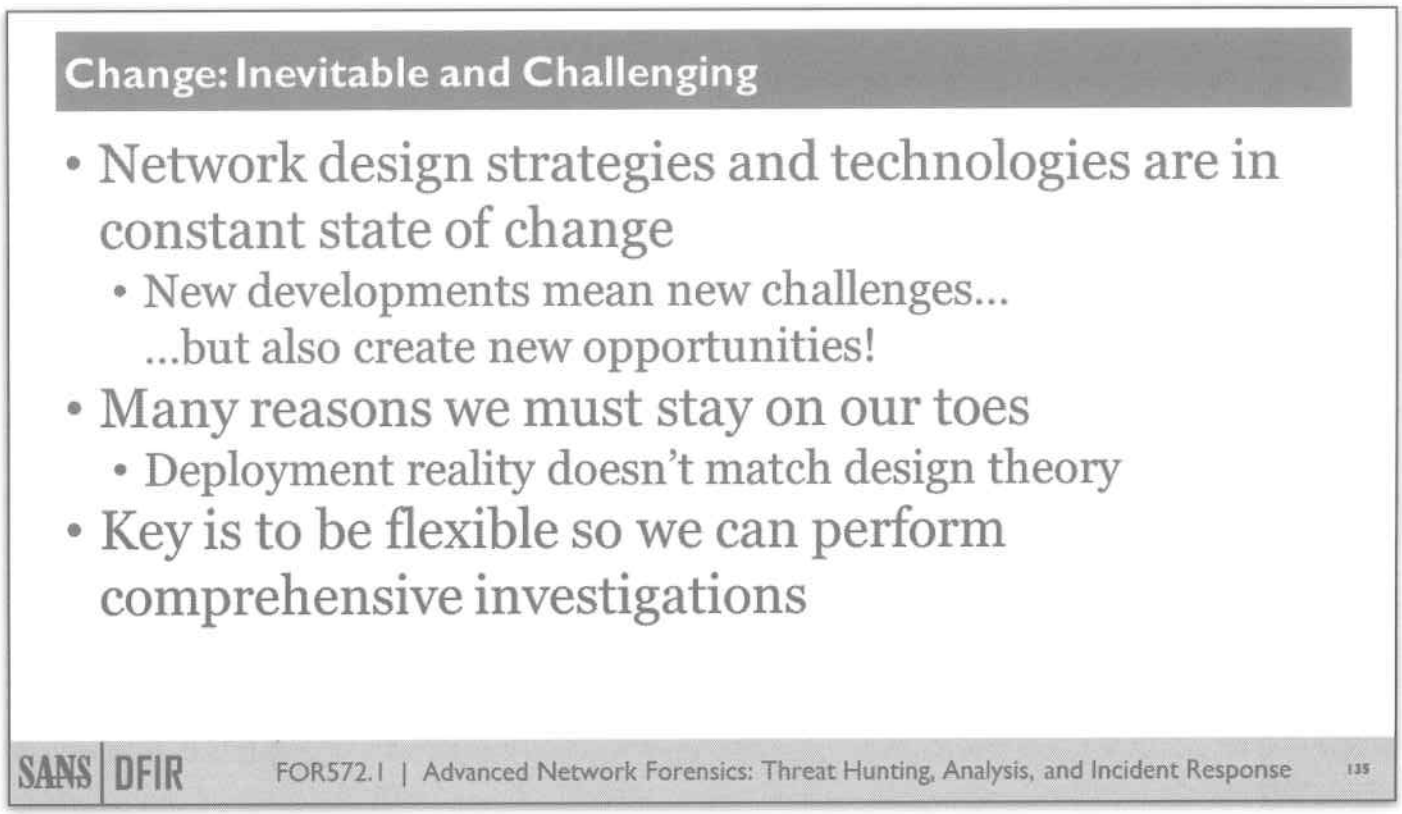
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



They say, "May you live in interesting times", though the origin of the phrase is debated....

We work in a field that has and will continue to undergo constant change. As the network becomes central to so much more of modern business processes and our daily lives, the constant change is perhaps, even more, a part of our work as network investigators. With each new technology that hits the market, or a new strategy (or even just a buzzword) that takes the community by storm, we will have new developments that need to be incorporated into analytic toolboxes and incident response plans. Nobody wants to be the one to tell the C-suite that you can't provide an answer because a device that was put into operation a few months ago rendered the entire IR process useless!

However, by engaging the security team during the development and deployment of new solutions, we can capitalize on the opportunities these technologies bring to the environment. Our success will always be a direct product of how well we can adapt to new and unexpected situations more than our ability to run a checklist, or click a predetermined sequence of buttons and menu items in a preselected piece of software.

This mindset is not limited to new technology, either. Sure, most organizations have a detailed network map, operating plan, and related documentation. However, the reality of operations is that these documents are never 100% accurate—they're often outdated as soon as they are created.

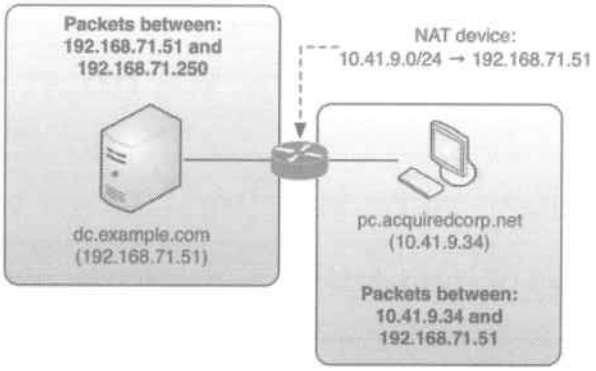
The key to victory lies in our ability to handle the unexpected. This game is often won or lost on an investigator's ability to seize upon unexpected opportunities.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Architecture: NAT

- Maps "inside" IP(s) to "outside" IP(s)
  - Modifies IP addresses—position of observation matters
  - May be a workaround for proper network integration
- Some NetFlow includes pre- and post-NAT IPs
- Often must correlate
  - Timestamp + source port
  - Logs describing transactions



The original scope of the Internet's addressing structure seemed limitless. Who could imagine more than four billion endpoint devices online back in 1980, when RFC 760 (superseded by RFC 791[1]) defined that 32 bits would be used for an IP address? Today, of course, the IP exhaustion is a serious problem facing network engineers around the world. One means of addressing this problem has been the employment of Network Address Translation, or NAT. At its core, NAT is a technology that allows a device to track one or more "inside" IP addresses, allowing them to map to one or more "outside" IP addresses. The "inside/outside" nature of this architecture lends itself to routing devices, which pass packets between multiple segments on a network, or segments on different networks. The NAT device will keep track of these connections, rewriting the source and destination IP addresses as needed, ensuring that each endpoint sees traffic that is consistent with its own place on the network.

It's important to note that while NAT is frequently performed between "private" IP netblocks (RFC 1918[2]) and public IP space, routers handling two publicly-addressable network segments or two privately-addressable segments can also perform NAT between the IPs under their respective purviews. This may be seen after network merges, which often occur after one business acquires another.

Some enterprise-grade network devices account for tracking NAT actions. For example, most NetFlow deployments will store both the original and NAT'ed IP addresses for each connection. This makes the analyst's job much easier when, say, an outside entity tells them that the company's public-facing IP made a connection to a known malware C2 server. The analyst simply reviews the NetFlow data to identify the connection in question, which includes the internal IP address of the offending system. Tracking down the source of that activity becomes a much easier task.

However, if this kind of data is not available, all hope is not lost. Correlating activity between multiple sources of evidence is possible, if occasionally tedious. Reconsider the above example, but imagine that only the inside IP addresses are available. If the external entity's notification included the timestamp and source port, an analyst could use these additional pieces of data to isolate the connection of interest, even if there were a large number of

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

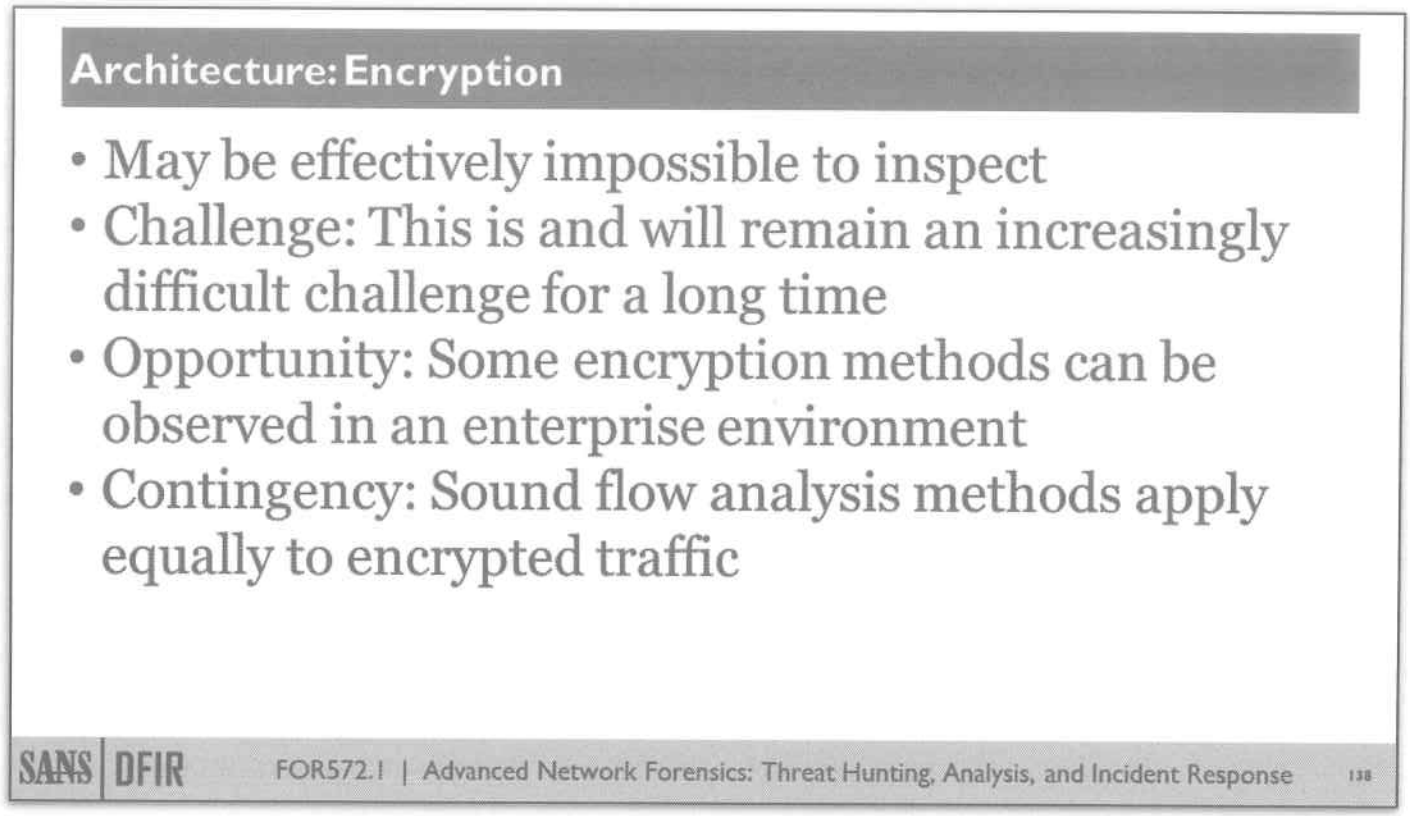
connections to the malware C2 system's IP address. Another common option would be to use web proxy logs, if available. These logs can contain valuable information, such as the IP address, hostname, URL, and other data regarding each request their respective proxy systems handle. If the malware used HTTP for its C2 activities, the proxy logs would be invaluable for identifying internal systems that are likely to be infected.

References;

[1] `http://for572.com/zsuwd`
[2] `http://for572.com/j2skt`

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Perhaps the biggest boogeyman in network forensics and investigations is encryption. Put simply, there is no easy way to "crack" into high-grade, properly-deployed encryption. The increasing emphasis on security in the IT marketplace all but ensures that the days of widespread plaintext services are numbered. If our analytic methodologies are built upon the ability to inspect the traffic contents, effectiveness is immediately minimized!

That said, encryption is often based on trust, which is… complicated. There are sound, robust methods that can be used to help reclaim some ground in this arena. In a typical enterprise environment, commercial and free solutions can be used to enable plaintext inspection and capture of most TLS-encrypted traffic.

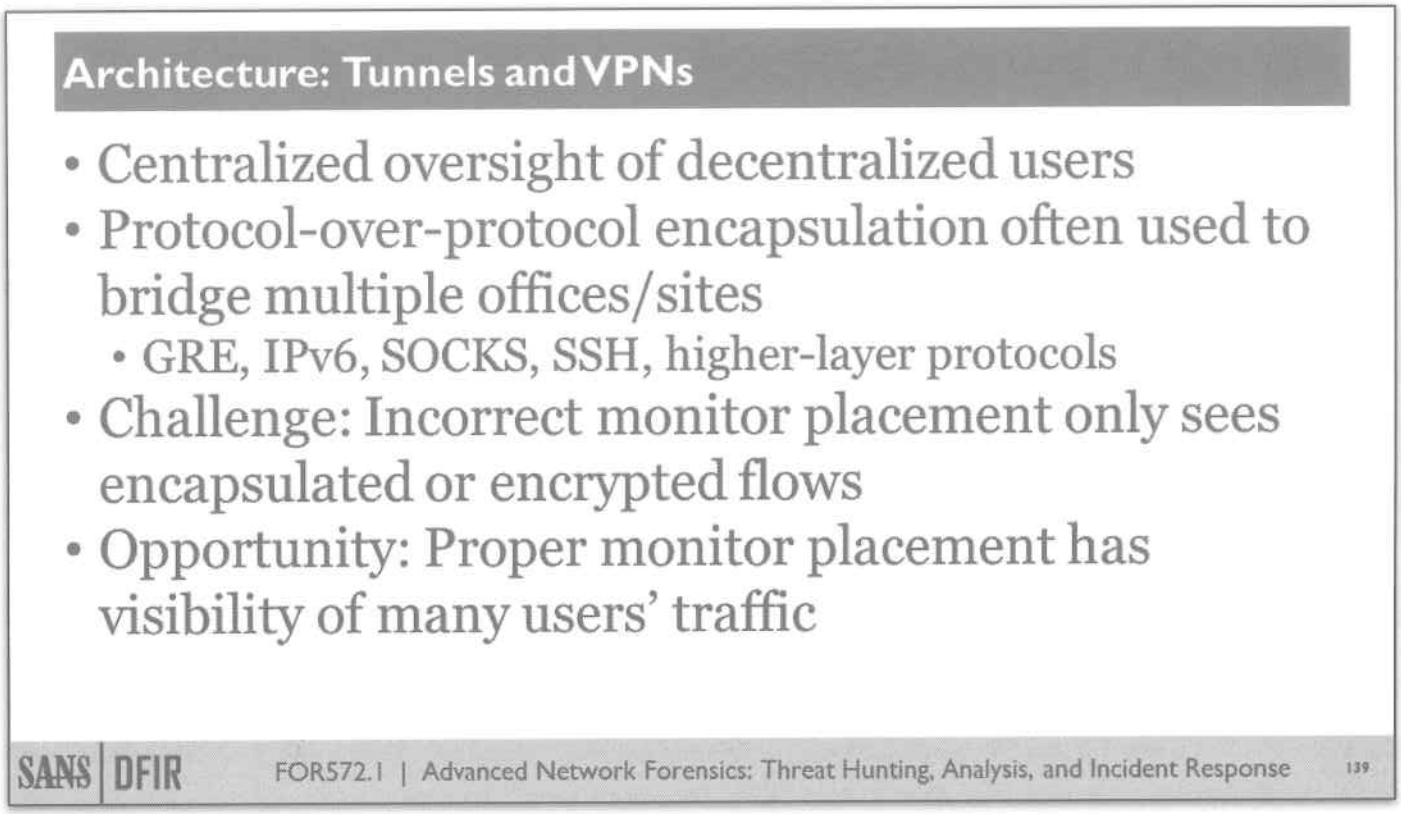
From a law enforcement perspective, capturing TLS-encrypted traffic for later reference may be feasible. For example, if you might acquire the private key material in the future, capturing the encrypted contents now could be a very useful step. With key material in hand (after a search warrant is executed), standard tools such as Wireshark can decrypt the traffic for analysis.

With some protocols—most notably SSHv2—accessing decrypted content is not feasible at this time.

Currently, when dealing with properly encrypted or otherwise "securely" encrypted traffic, we are limited to analytic processes that don't require access to the data portion of the network traffic. These fall into the "flow analysis" category, and can still prove quite useful. Flow analysis focuses on the metadata of the network traffic—packet headers in this case. Fields such as source and destination addresses and ports are useful, but later in the class, we will also learn how to leverage packet size and timing data to generate leads and build hypotheses.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



In a pure tunneling setup, one stream of network traffic becomes the payload of another stream of network traffic. This can occur over purpose-designed tunneling protocols such as GRE, or through repurposed higher-layer protocols such as HTTP, DNS, or ICMP. There are also specialized-use cases such as moving IPv6 traffic between IPv6 enclaves connected by an IPv4 network (aka "6in4"), and proxy/tunnel hybrid protocols such as SOCKS.

This encapsulation can present challenges for network administrators because many traffic management and control utilities are not tunnel-aware. Additionally, the sheer number of possible tunneling protocols make detection, let alone management, difficult. These challenges translate to the investigative domain, as we would need to first identify that tunneling is present, then determine how to decapsulate it before starting to analyze the contents. Although it's not impossible to handle such situations, the added steps can create a lot of extra work.

In the case of engineered tunneling, such as GRE or IPv6, proper monitor/capture placement is key. As long as we can confirm the points of encapsulation and decapsulation, we can ensure that the capture platform is acquiring the decapsulated traffic, avoiding a messy preprocessing step in the workflow.

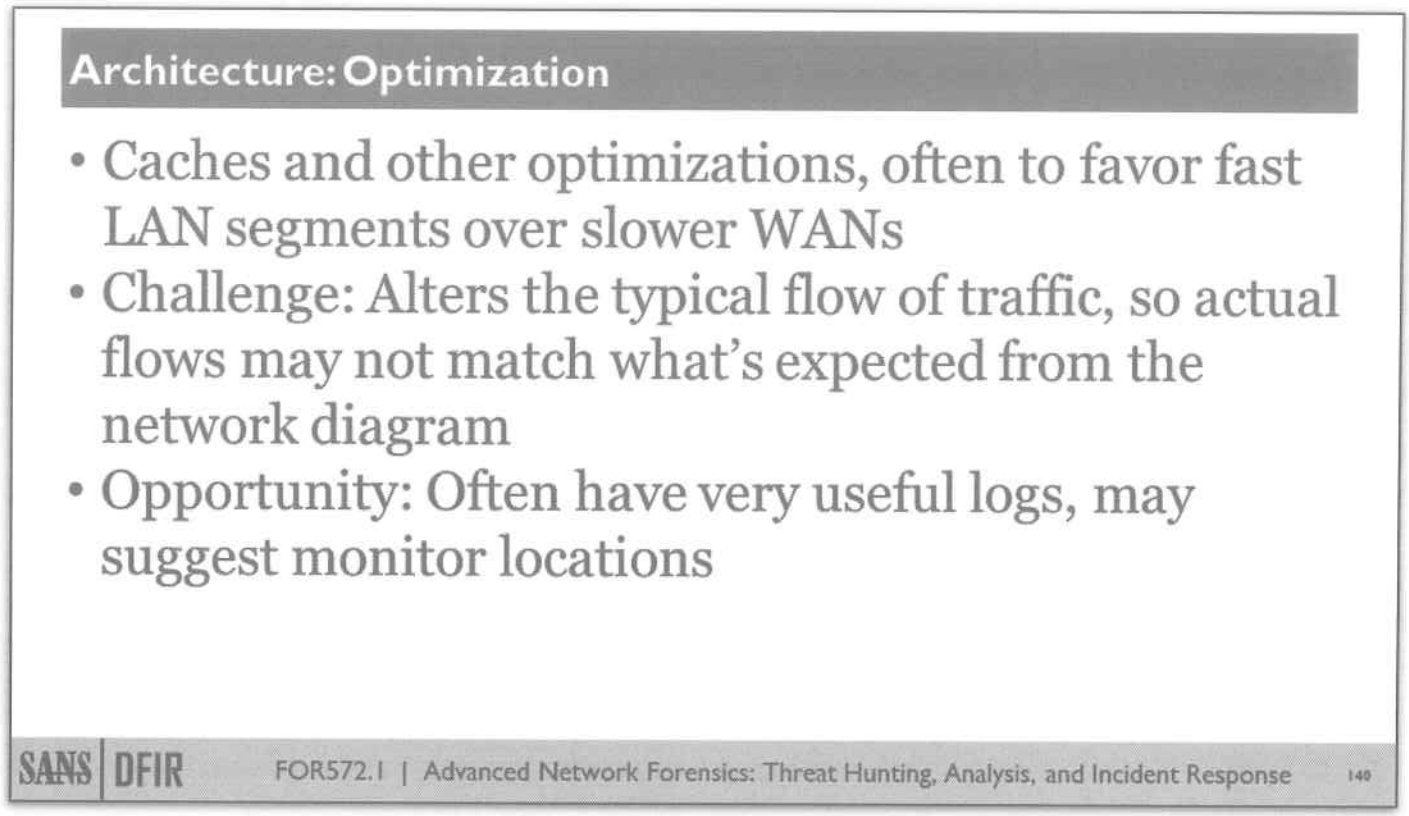
Virtual Private Networks (VPNs) are a special case of tunneling and have become highly used in most networks. A VPN provides an encrypted tunnel through which remote users can establish a network presence as if they were sitting within the "home office" network perimeter.

Although a huge boon to network and system administrators, an investigator must fully understand the VPN architecture before placing a network capture platform. If placed on the wrong side of the VPN endpoint, all we would see is the encrypted traffic—this might be of some utility, depending on the investigative priorities—but in most cases, we would want to see the traffic after it's been decrypted and is "native" within the environment.

Given that relatively minor hurdle to clear, a VPN means that most or all of the remote users' traffic can be centrally monitored or captured. When cross-checked against the VPN authentication logs, it becomes trivial to associate traffic of interest to a source system, user account, and possibly a location.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



A common way to improve speed and network utilization is to deploy network optimization devices. Perhaps the most common of these is the web proxy server. Of course, this device keeps local copies of HTTP-sourced objects, ensuring identical content is only pulled from a remote server once, regardless of how many times the pool of local clients request it. This methodology helps to efficiently use smaller WAN pipes while taking advantage of the fast internal LAN for client servicing. This concept has been extended beyond HTTP browsing traffic, and a wide array of devices offer the same result for other protocols and use cases.

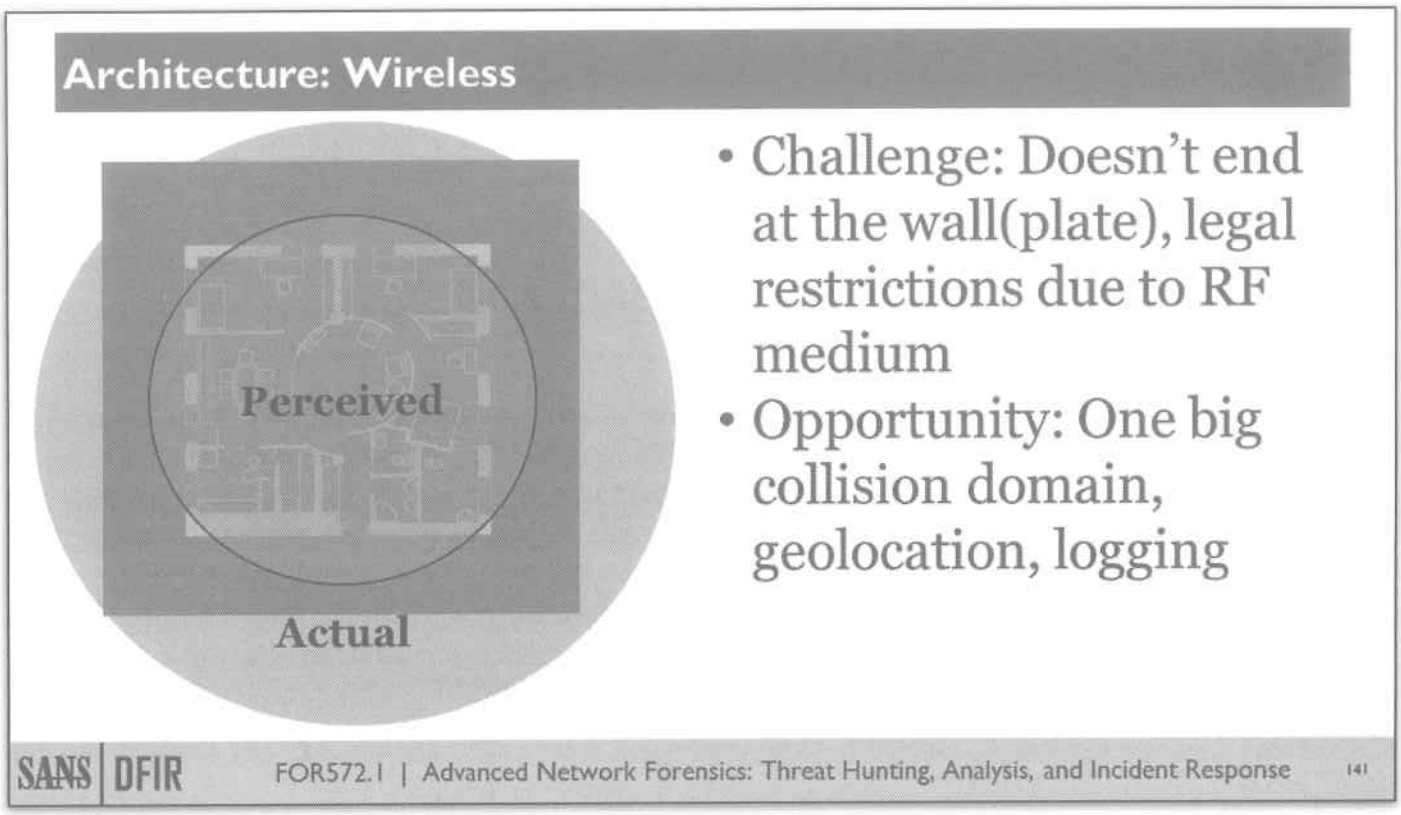
WAN optimization has become a niche industry, with commercial offerings such as the Riverbed SteelHead, F5 BIG-IP, SonicWall WXA, Blue Coat MACH 5, and many others vying for market share. Several open-source solutions such as Traffic Squeezer and WANProxy are also solid solutions. Although specific features vary, many of these transparently perform proxy-like functions for email, web applications, file system access, video streaming, and more. Some manufacturers even claim 100x speed increases in certain scenarios!

Clearly, the speed increase will make users very happy, but consider how such a solution alters the typical flow of network data. From the client's perspective, the connection is established with a remote server, which provides the requested data. However, the WAN optimizer is in the middle of that connection and may provide the requested data from a local cache. Although the optimizer generally performs some verification that the cached object is still current and valid, the remote server (and therefore any network segment on the front side of the optimizer) may not even be aware that a client request was made and likely won't have any knowledge about the client's identity. From the remote server's perspective, the connection request was from the optimizer.

Despite these complications, most devices provide extremely good logging features. Such logs are invaluable during an investigation and can correlate activity observed on either side of the optimizer device. Additionally, because these devices tend to be deployed at the perimeter of large networks, they may sit on segments that provide ideal visibility for network traffic capture or monitoring.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Obviously, wireless networks have become a major component of network architectures, and the rate of adoption has skyrocketed. Just five or ten years ago, many enterprises would never have considered deploying a wireless network component. However, user demand, mobile device adoption, availability of security-conscious deployment options, convenience, and other factors have all contributed to enterprise adoption over that period.

The primary challenge to investigating wireless incidents is that by design, the network is everywhere. It's quite difficult to restrict where a mobile device can access the network. In fact, the availability of a network is a product of both the network infrastructure itself as well as the capabilities of a client device. Although in the United States the FCC governs the power of radio transmitters, those intent on doing harm are not inclined to follow such guidelines. In addition, using a large or directed receiver is not regulated and could enable a wireless client to access the network traffic from much farther away than intended. This means that even physically "secured" wireless deployments certainly extend past their designed service areas.

Another consideration, mentioned briefly before, is that legal requirements often change significantly when entering the wireless realm. These legal stipulations and restrictions vary greatly between jurisdictions. Because this is not a law class, we'll just say that you must get appropriate legal guidance before setting out down this road. (Getting adventurous in this department didn't go so well for Google's Street View project!)

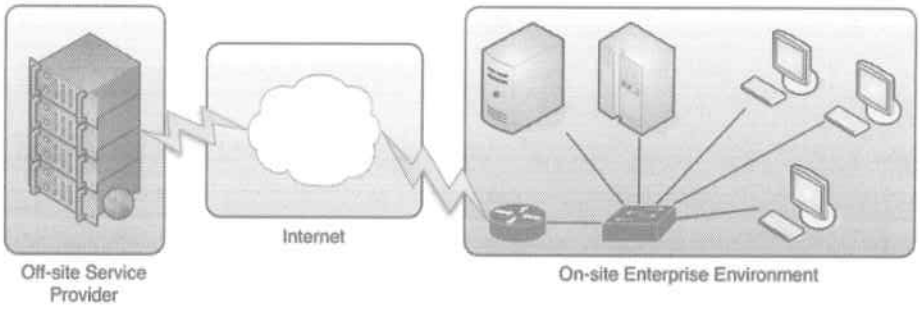
On the bright side, the nature of wireless networks dictates that there is one big collision domain—so we don't have to worry about the same collection limitations as when working with wired, switched networks. Also, while the large footprint of a wireless network is a challenge, the RF medium also presents an opportunity for us to geolocate actors on the network. Both commercial and free tools are available to use the relative signal strength of network stations and known location data—such as a GPS feed or clicking on a map—to identify the approximate physical location of a device of interest. Finally, most large-scale wireless deployments use hardware that keeps great logs, which can be extremely useful during an investigation, as we'll see later.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Architecture: THE CLOUD

- Challenge: Off-sited functions lack visibility, new capabilities in uncharted territory
- Opportunity: Often expose data and functionality through HTTP APIs



Although the "cloud" term rapidly achieved buzzword status, the trend back to centralized, remote computing is very real. Whether considering hosted applications, remote storage, or elastic computing capabilities, the typical enterprise landscape no longer ends at its traditional perimeter. Because the core functionality of such services relies on network connectivity, we must consider how to address them within the scope of an investigation.

Perhaps the biggest concern most CIOs have with cloud-based services is that they are off-site, and therefore seen as less "under control". The extent to which that changes the security posture is difficult to calculate, but there is a definite and justifiable hesitation to move major functions "outside the house". This lack of visibility and control can be detrimental to an investigation—for example, how many failed logins occurred before the successful one that subsequently exported an entire client database? From what IP address(es) did those login attempts come? Depending on the service, that data may not exist. If it does, the service provider may not provide it without legal intervention.

Another hurdle to overcome when investigating an incident involving a hosted service is that the model is new and undergoing rapid change. Although the forensic community is starting to address the cloud trend, each service has the potential to be completely different than anything previously seen or examined. We must be prepared to "start from zero" when a new service falls into scope during an investigation.

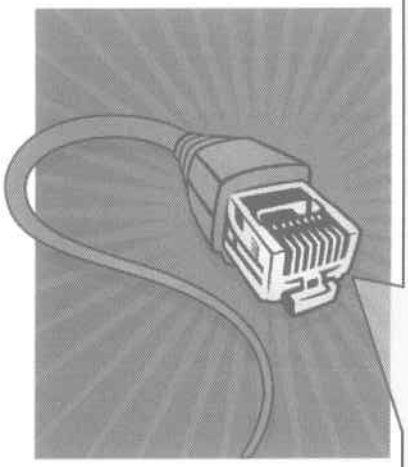
One interesting aspect of most cloud-based services is that they provide access to data and functions through structured HTTP requests called APIs. Because HTTP(S) can be proxied, we can intercept and log the requests and corresponding responses. Because service providers generally publish the APIs, we can use that documentation to characterize or even create a transcript of activity on the service.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

Trend: Everything-over-the-Network

- Voice over IP, Storage Area Networks, etc. use the network
- Challenge: New data means new analytic challenges, traffic volume can be HUGE
- Opportunity: Job security!



Data
Video
Voice
Security Cameras
Storage
...
???

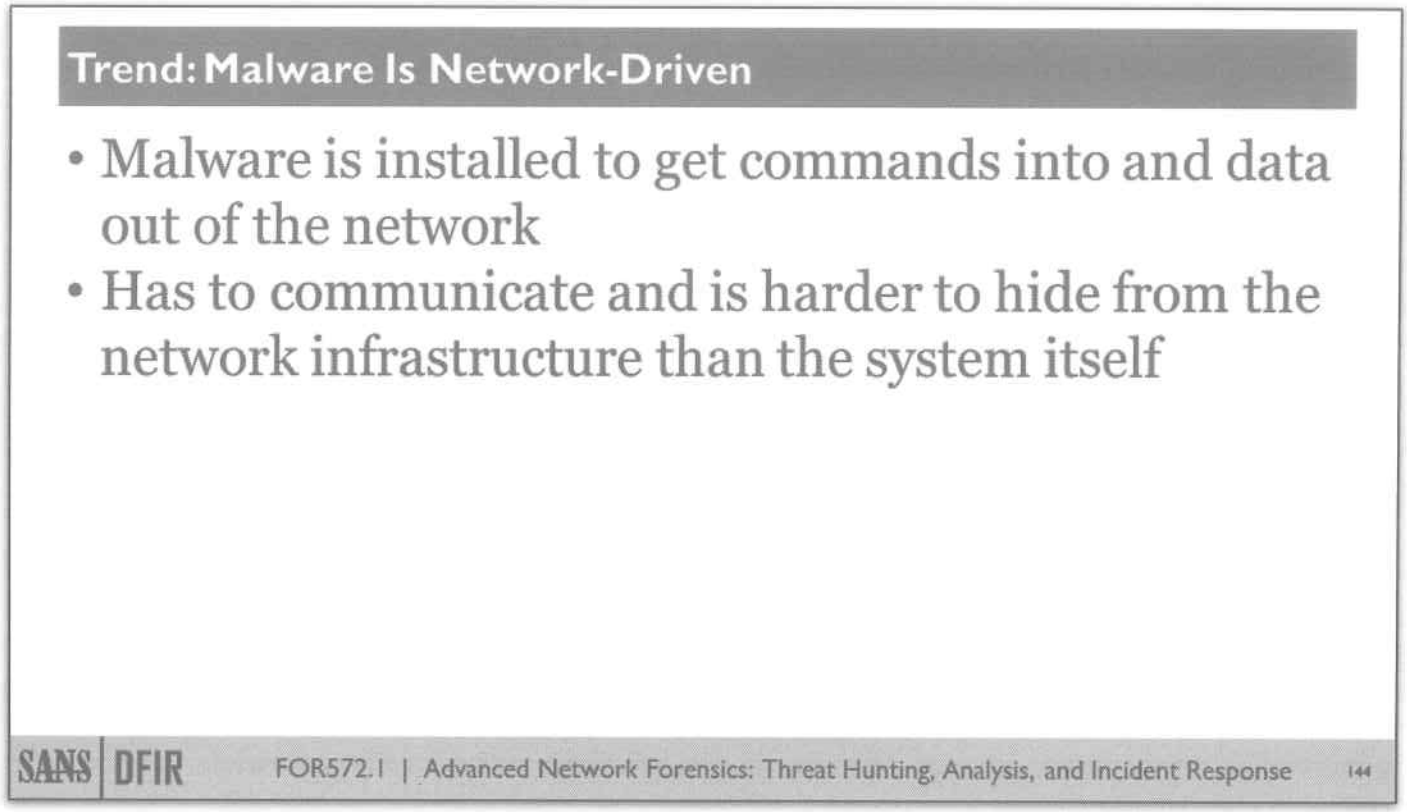
The fact that we are all here in this class today is a testament to the HUGE trend toward network-centric business operations! What used to be an underused 10base2 "thinnet" LAN is now a high-speed, VLANned gigabit Ethernet switched fabric that never seems to be fast enough to keep the users happy.

We're seeing a huge shift to using the network for more and more business communications. Voice over IP (VoIP) technology and devices are converging the data and voice networks. Storage Area Networks (SANs) and iSCSI are using Ethernet as a foundation to perform storage-related tasks. Backups, remote synchronization services, and more are all using network features to perform their core functionality.

Although it's always nice to be needed, think of the volume of data that is being generated. It's common to see link rates of tens of gigabits per second, with sustained data rates amounting to several gigabytes per second, which means a LOT of work for the investigator to separate out the relevant data! This is why we spend so much time discussing high-level flow analysis and data reduction. Flow analysis is a great skill that we can use to quickly characterize large volumes of traffic and identify time slices of interest that warrant in-depth investigation. Then, we can use data-reduction techniques to extract traffic from the period of interest and dig more deeply into the traffic to seek answers.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



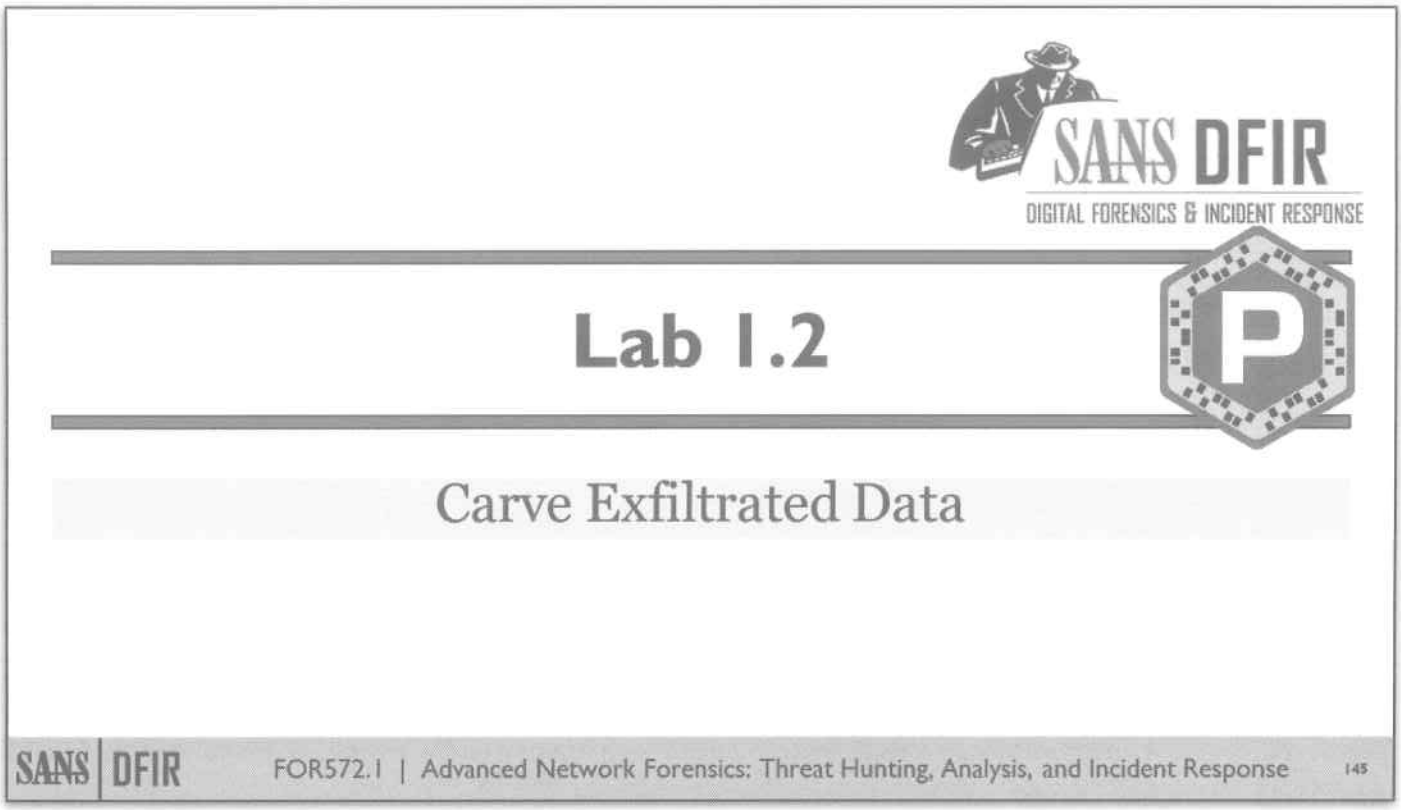
Lastly, we have to consider that the fundamental nature of malware is to accept commands from its master and provide data of some kind in return. Hollywood spy movies notwithstanding, malware generally communicates using the network. Although there are some clever methods for malware to "hide" from host-based discovery tools, the network path involves numerous pieces of hardware and a variety of strict(ish) protocols. Hiding in that environment is much more difficult. Therefore, network-based malware discovery and analysis are sometimes faster and more fruitful than starting from the host and working up.

Although attackers can use encryption, obfuscation, and other methods to make discovery or analysis more difficult, the data is there on the wire and can be discovered in a properly instrumented environment. We should never forget that this business is always an arms race, and with each new capability we acquire, attackers will find a way to thwart it. Lather, rinse, repeat.

At this time, the network represents a weakness to malware: The malware relies on the network for its core functions but cannot control or manipulate the network extensively. We should seize upon this weakness while it still exists!

https://t.me/learningnets

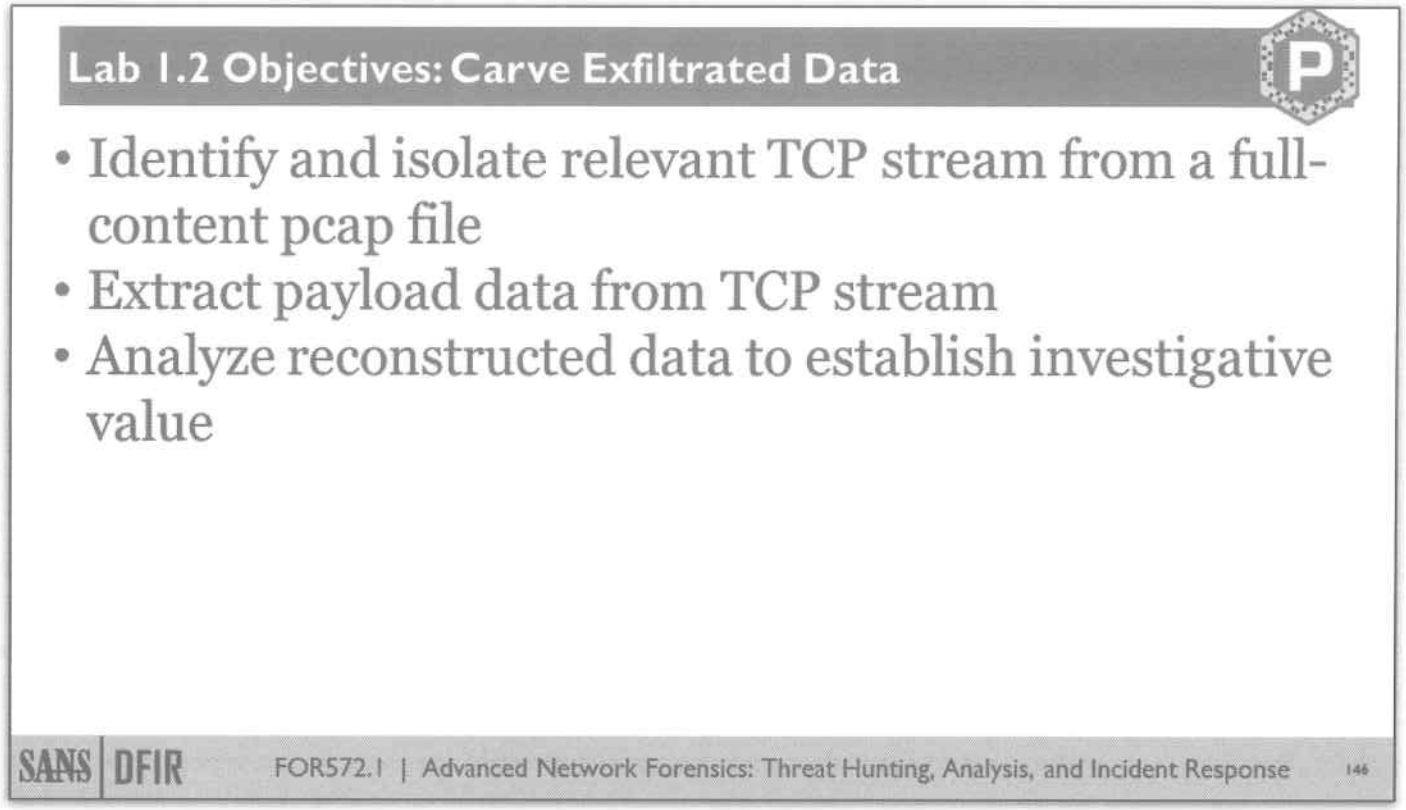
© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

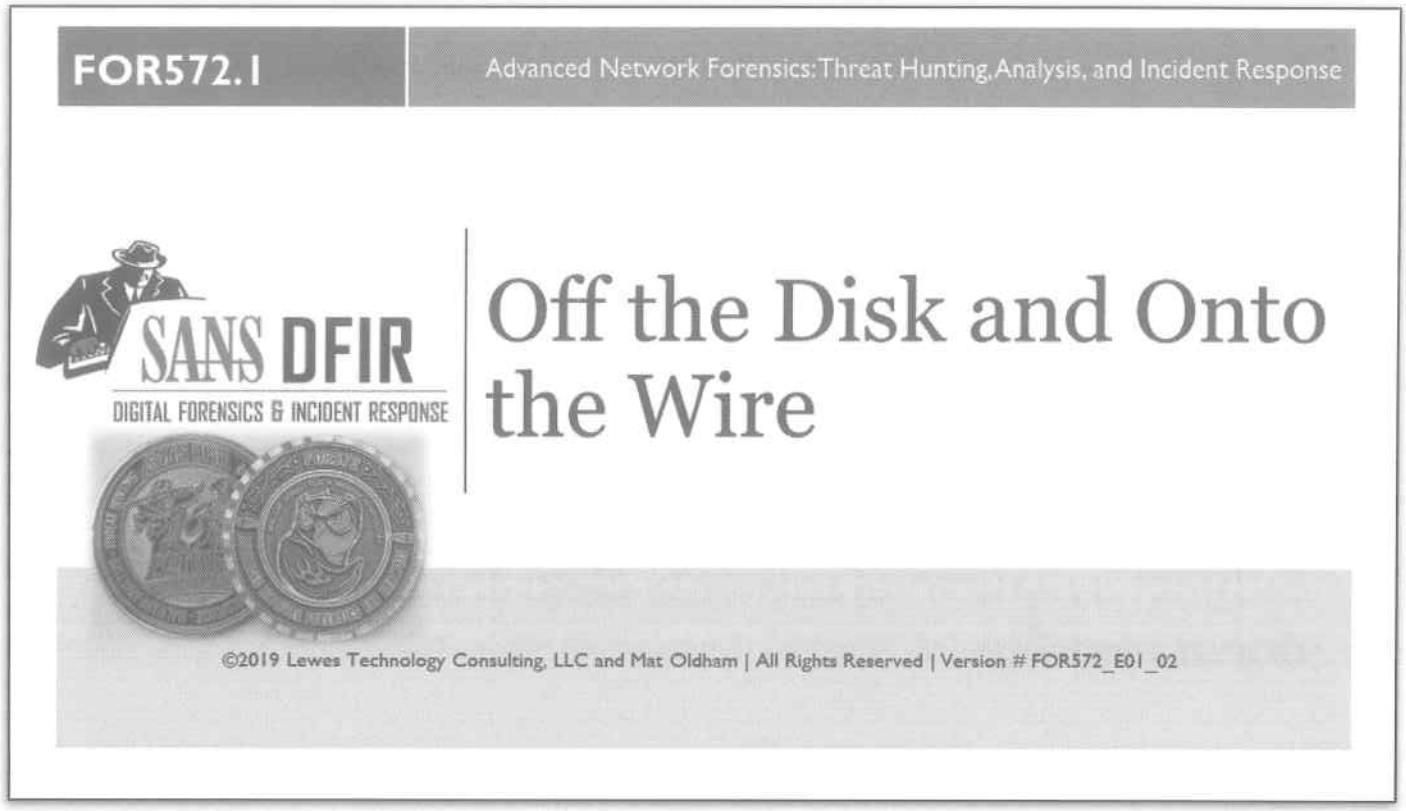
Lab 1.2 Takeaways: Carve Exfiltrated Data

- Subject conducted suspicious web searches
  - Searches generated different types of traffic, depending on how they were performed
- Subject performed two uploads to a paste site
  - One was a "test run" and the second contained the exfiltrated data
- Reconstructing files can be multi-step process
- Subject's malicious intent is clear from the document file

This page intentionally left blank.

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham



Authors:
Phil Hagen, Lewes Technology Consulting, LLC
phil@lewestech.com | @philhagen
Mat Oldham
mat.oldham@gmail.com | @roujisecurity

https://t.me/learningnets

© 2019 Lewes Technology Consulting, LLC and Mat Oldham

This page intentionally left blank.

https://t.me/learningnets

https://t.me/learningnets