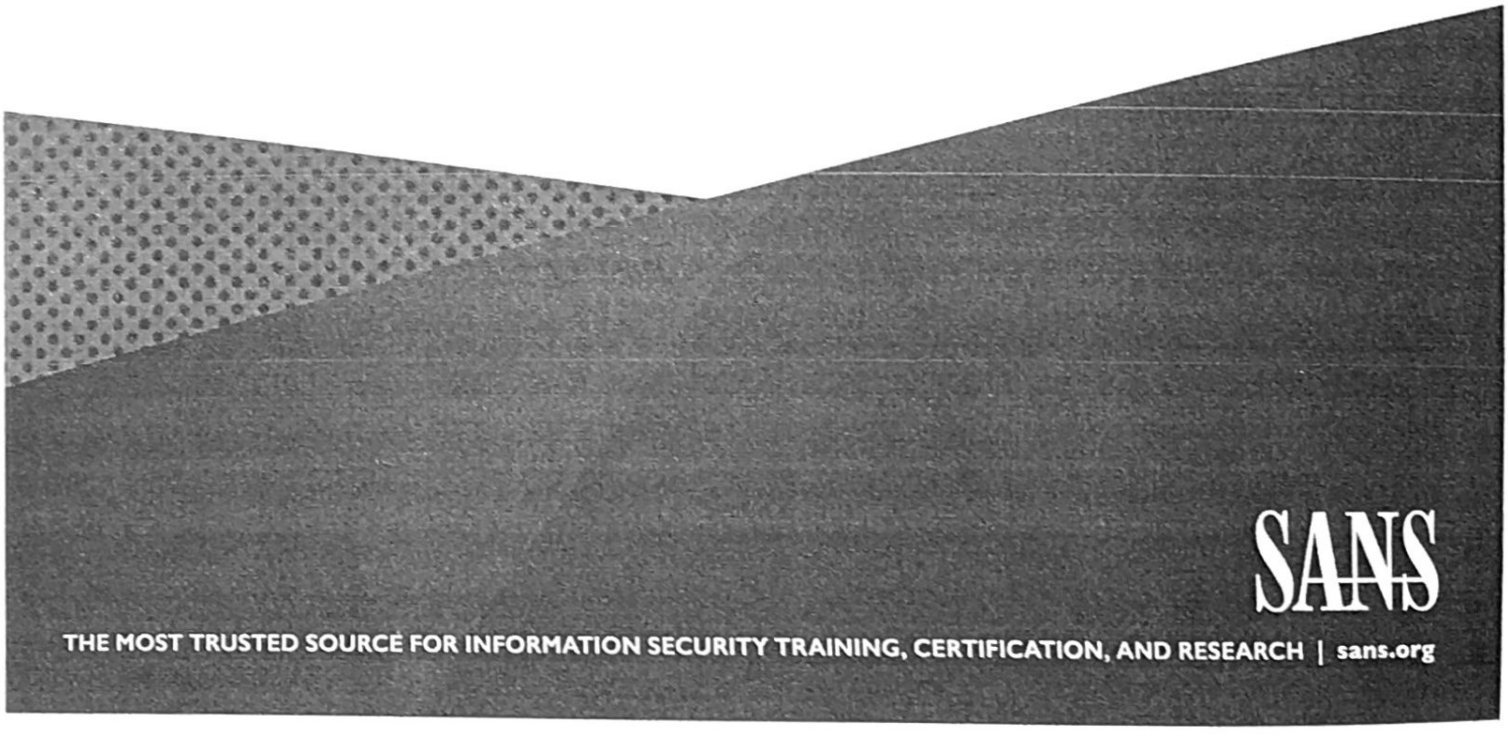
572.1
Off the Disk and Onto the Wire

SANS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

572.1
Off the Disk and Onto the Wire



SANS
THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

Copyright © 2017, The SANS Institute. All rights reserved. The entire contents of this publication are the property of the 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.

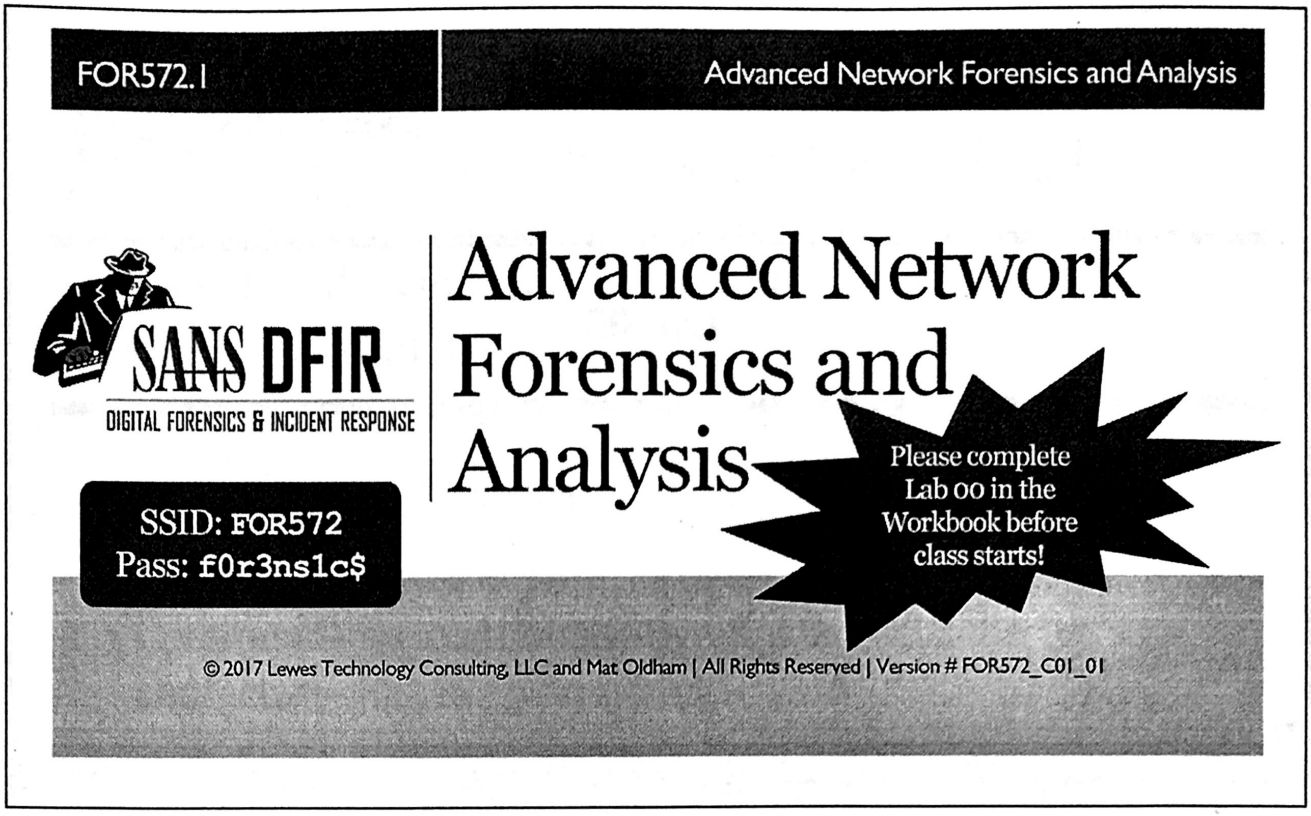
User may not copy, reproduce, re-publish, distribute, display, modify or create derivative works based upon all or any portion of the Courseware, in any medium whether printed, electronic or otherwise, for any purpose, without the express prior written consent of the SANS Institute. Additionally, User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

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

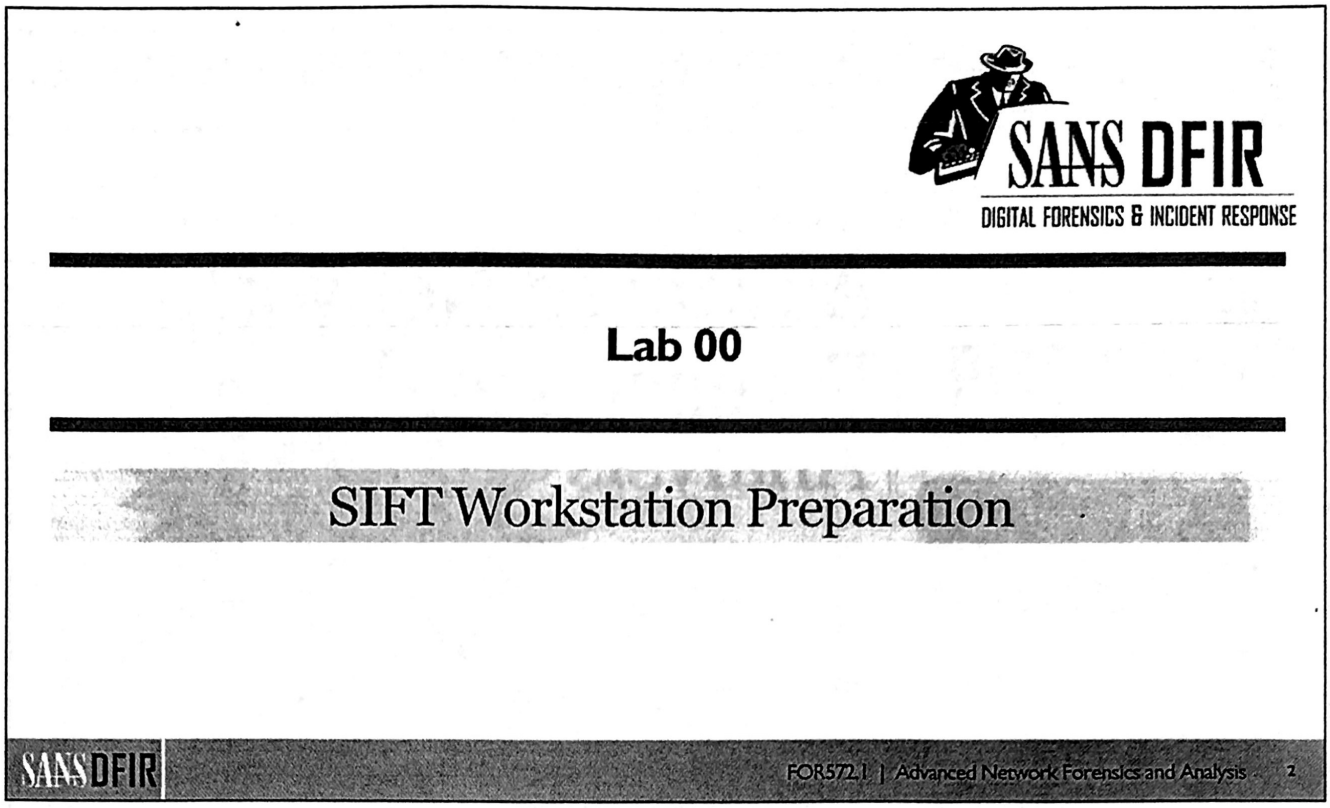


Authors:
Phil Hagen, Lewes Technology Consulting, LLC
phil@lewestech.com | @philhagen
Mat Oldham
mat.oldham@gmail.com | @roujisecurity

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

http://twitter.com/sansforensics

Welcome to SANS FOR572: Advanced Network Forensics and Analysis! After you get settled on Day 1, please complete Lab 00 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.



This page intentionally left blank.

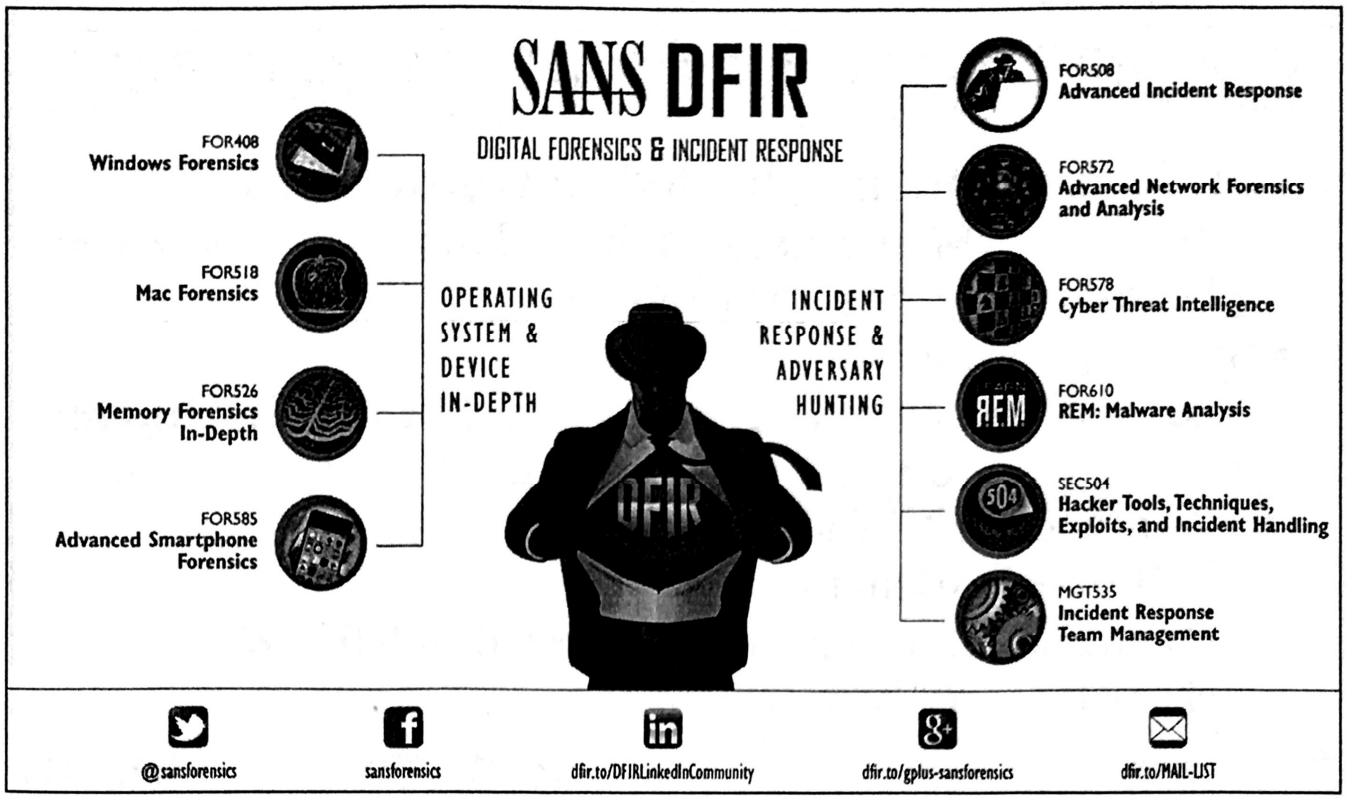
© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Lab 00 Takeaways: SIFT Workstation Preparation

- SIFT Workstation v3 booted and operational
  - Additional VMware images if time allows
  - Shared folders for evidence access
- Syntax familiarity
  - **`Bold Courier New font designates user input`**
  - `Regular Courier New font designates output`
  - Watch out for "`1`" (numeral one) vs. "`l`" (lowercase letter L)
  - Command-line continuation indicated with the "⏎" character – just keep on typing!

This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

SANS DFIR

DIGITAL FORENSICS & INCIDENT RESPONSE

OPERATING SYSTEM & DEVICE IN-DEPTH

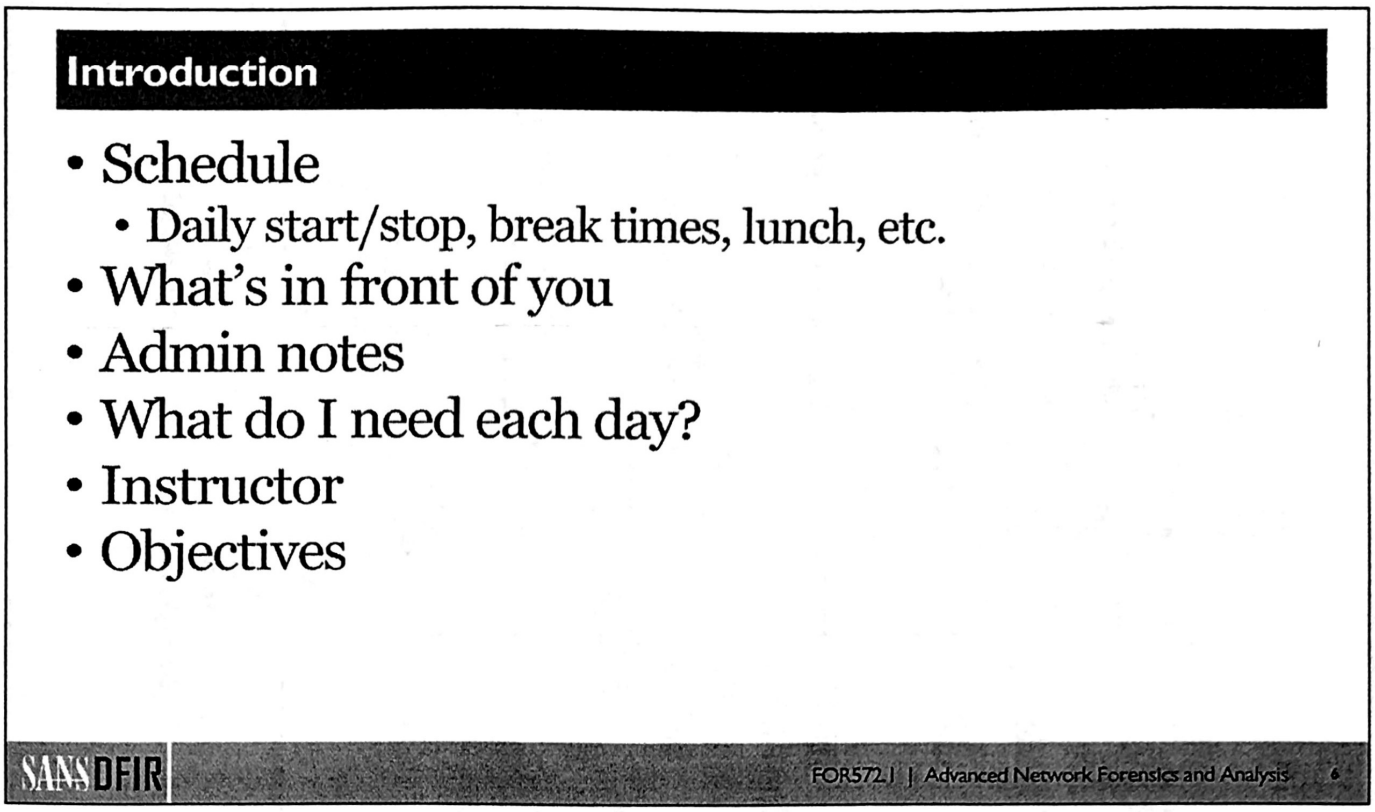
- FOR408 **Windows Forensics**
- FOR518 **Mac Forensics**
- FOR526 **Memory Forensics In-Depth**
- FOR585 **Advanced Smartphone Forensics**

INCIDENT RESPONSE & ADVERSARY HUNTING

- FOR508 **Advanced Incident Response**
- FOR572 **Advanced Network Forensics and Analysis**
- FOR578 **Cyber Threat Intelligence**
- FOR610 **REM: Malware Analysis**
- SEC504 **Hacker Tools, Techniques, Exploits, and Incident Handling**
- MGT535 **Incident Response Team Management**

@sansforensics | sansforensics | dfir.to/DFIRLinkedInCommunity | dfir.to/gplus-sansforensics | dfir.to/MAIL-LIST

© 2017 Lewes Technology Consulting, LLC, 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.

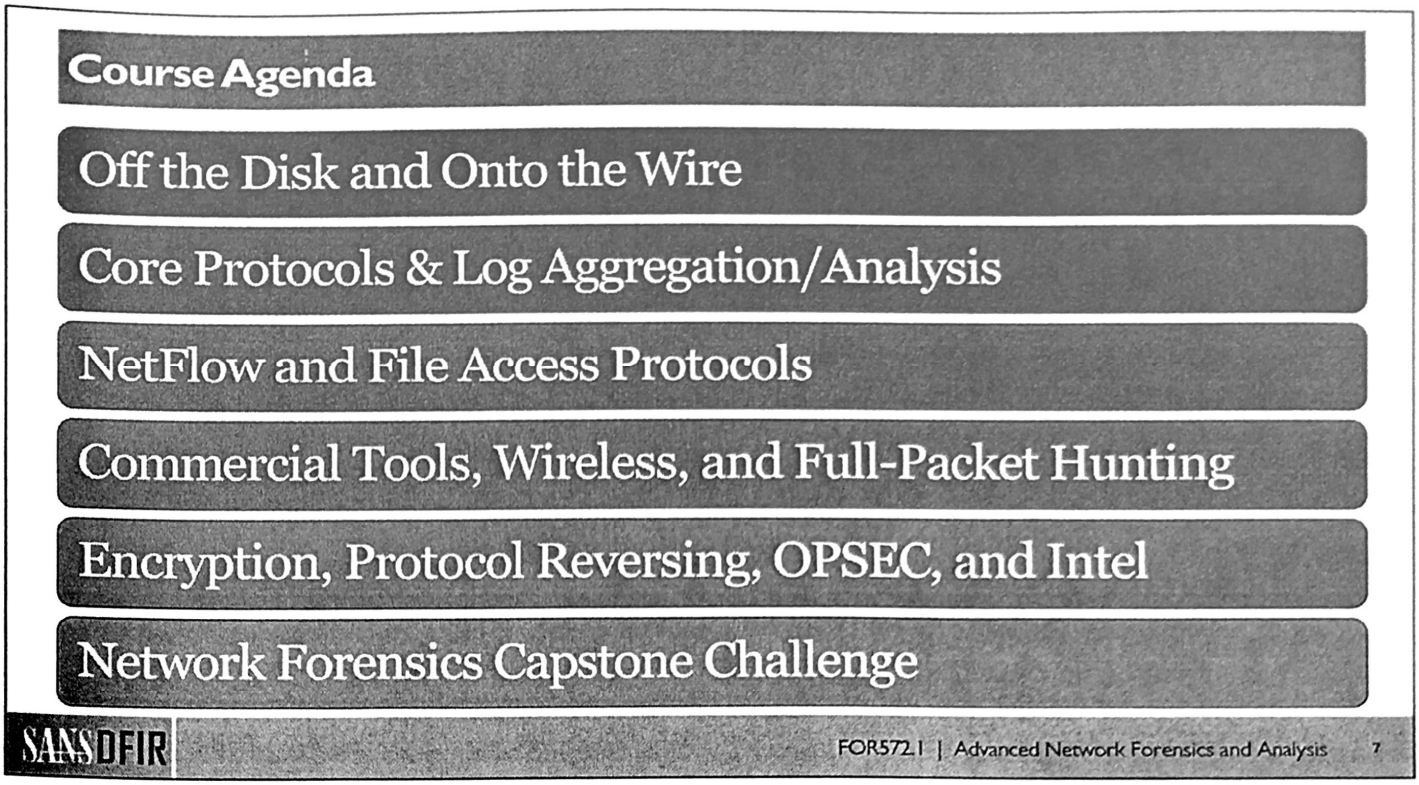
We'll step through a number of investigations throughout this class—twisting, turning, and massaging network data to arrive at ground truth. Along the way, you'll be learning and developing tools and techniques that you can take back to the workplace and put to use immediately, giving you a better understanding 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, DVDs, and/or USB drives with the VMware images you will need for the course. Our recommended platform is VMware Workstation on Windows or Linux hosts and VMware Fusion on macOS hosts. You are free to try other virtualization software, but there is no guarantee the VMs or exercises will work, and you'll be on your own for support.

Bring your laptop to each class and the appropriate book. (e.g. 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 bringing that section's book with you until you're more comfortable with the material.

The course material and exercises do not fundamentally rely on Internet connectivity.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

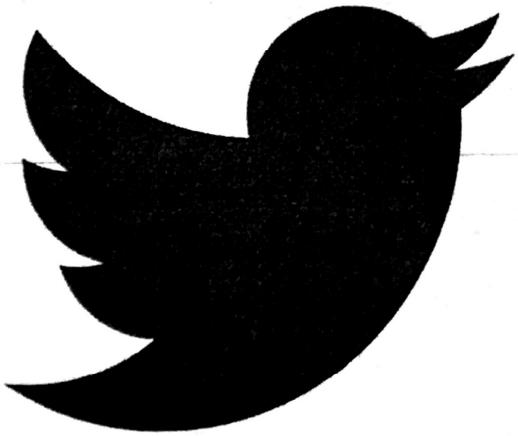


This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Tweet During Class

- On Twitter?
- Meet classmates
- Connect online
- `@sansforensics`
  - Hashtags: `#FOR572`, `#GNFA`, event-specific



- Class notebook
  - `http://for572.com/notebook`

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 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.

We've also collected a large (and growing) number of resources that are either too small or 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.

© 2017 Lewes Technology Consulting, LLC, 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Why Network Forensics?

- Hard drive may not be available
- Evidence may already be collected
- Can't practically investigate 6,000 machines
- Hunt for the attacker you don't know exists

SANS DFIR | FOR572.1 | Advanced Network Forensics and Analysis 10

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 them 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 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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Strategic Course Objectives

- Understand forensic methods as they apply to network investigations
- Identify and leverage evidence containing Artifacts of Communication
- Carve payload data as needed
- Establish habits for sound OPSEC
- Build an activity timeline for a more complete understanding of an event

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 that 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 it 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 does not rely on conjecture or unfounded opinions.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

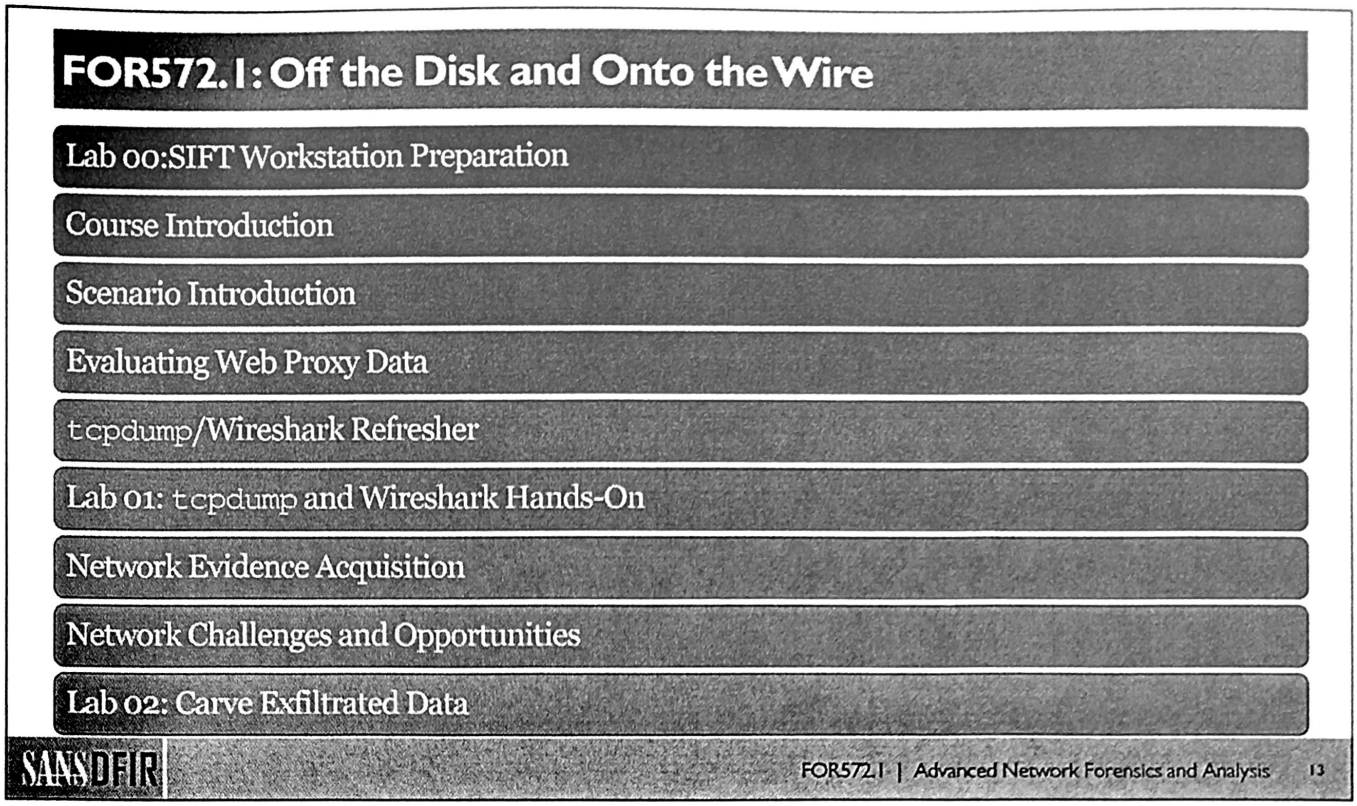
FOR572.1 | Advanced Network Forensics and Analysis

Off the Disk and Onto the Wire

© 2017 Lewes Technology Consulting, LLC and Mat Oldham | All Rights Reserved | Version # FOR572_C01_01

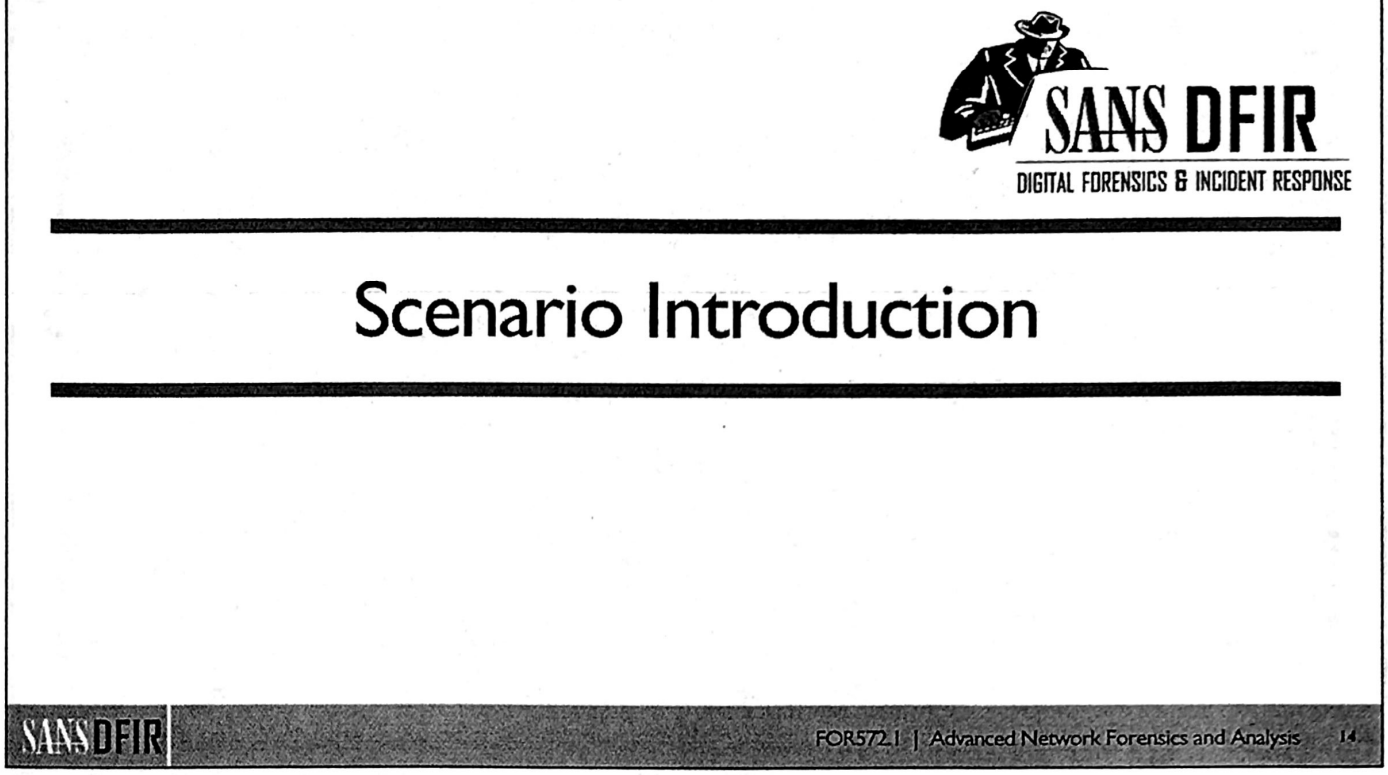
Authors:
Phil Hagen, Lewes Technology Consulting, LLC
phil@lewestech.com | @philhagen
Mat Oldham
mat.oldham@gmail.com | @roujisecurity

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



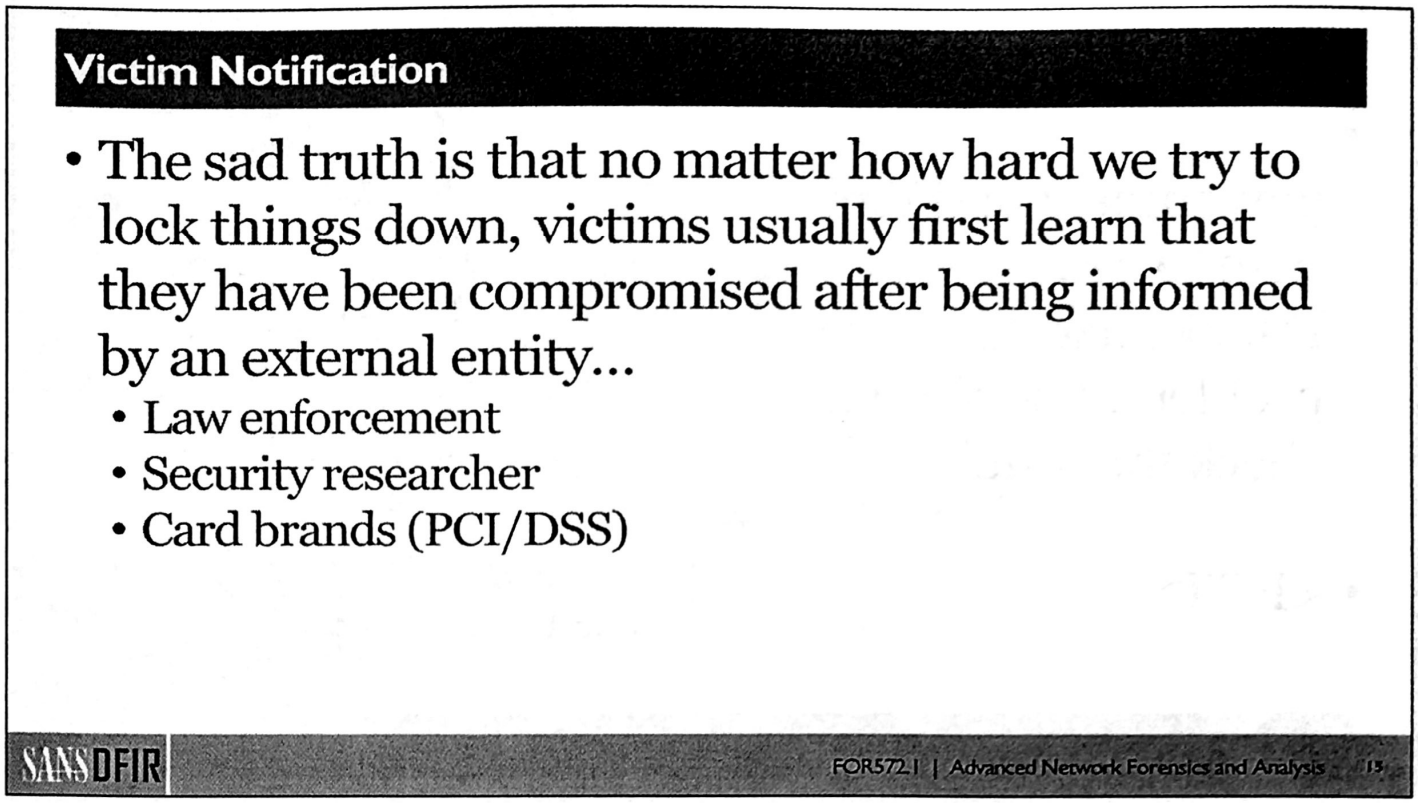
This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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 nearly 65% of victims first learn that their network has been compromised after a call from an external party. 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, Mandiant/FireEye reports that victims learn of a compromise only after an average of 146 days[1]—long after the most critical evidence has been overwritten or otherwise unavailable.

The following slide sets the stage for a typical victim notification.

References:

[1] http://for572.com/u0srz

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Stark Research Labs

- "I'm with the government and I'm here to help. It looks like you have a problem. You should check that out."
- <EOT>

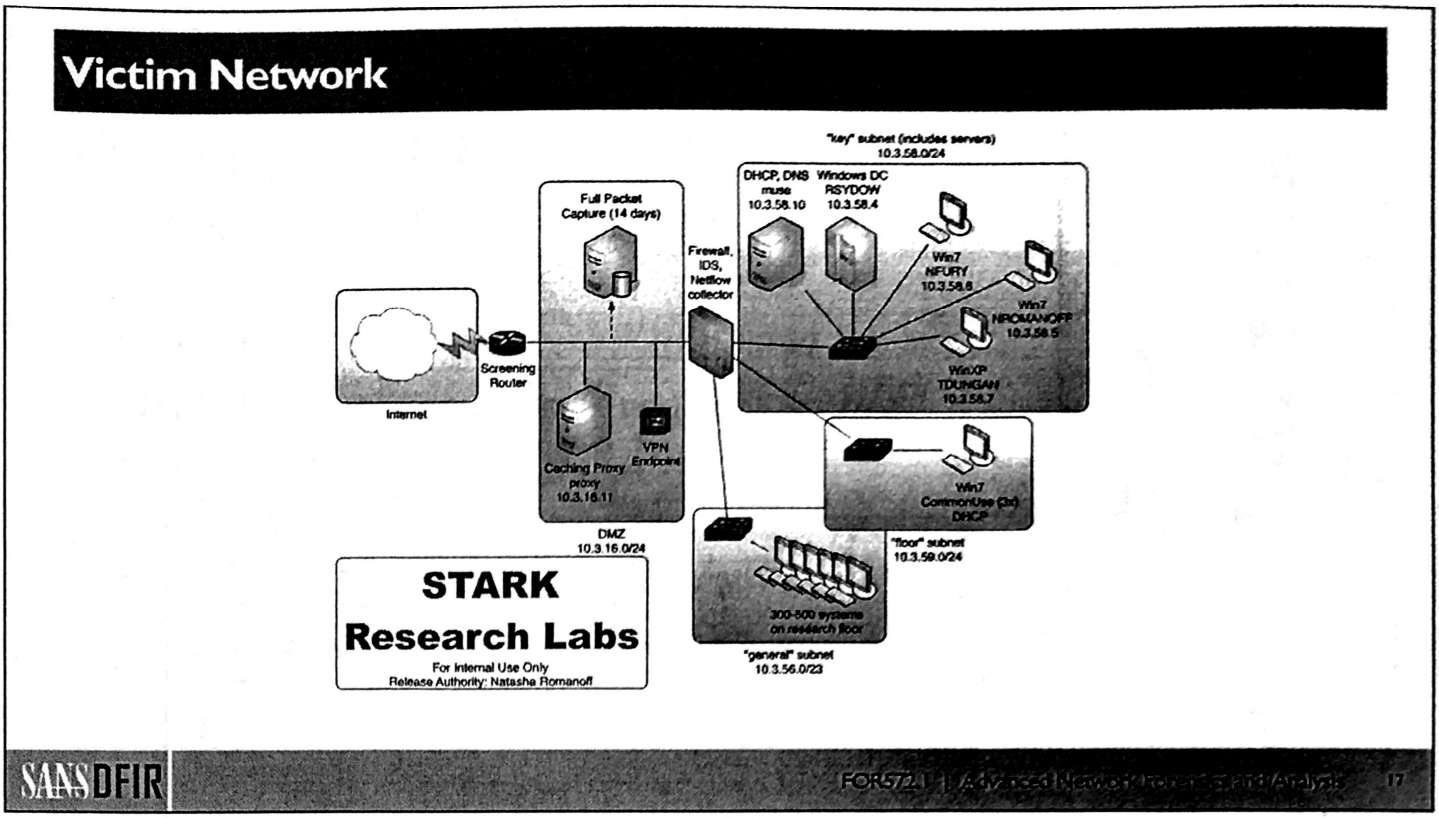
SANS DFIR | FOR572 | Advanced Network Forensics and Analysis 16

Stark Research Labs (SRL) is a government-sponsored R&D facility. Stark is renowned for its advanced metallurgy and bioengineering research.

Unknown to the general populace, SRL is also a front company for the United Nations Strategic Hazard Intervention Espionage Logistics Directorate—S.H.I.E.L.D. The current high-priority project is cloaked in secrecy. It aims to re-create the long-lost formula for the Vibranium alloy used in Captain America's shield and use it to build advanced vehicles. S.H.I.E.L.D.'s primary adversary is HYDRA, which has been moving away from kinetic and HUMINT collection operations wherever possible, instead leveraging electronic compromises to simply steal targeted technologies.

The greatest fear of the government, and that of SRL's lead researcher, Timothy Dungan, is that HYDRA may have agents inside SRL who are after these metallurgical formulas and other technologies. If HYDRA succeeds in stealing them—locally or across the Internet—they could move to manufacture the vehicles before we are ready to counter the threat.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

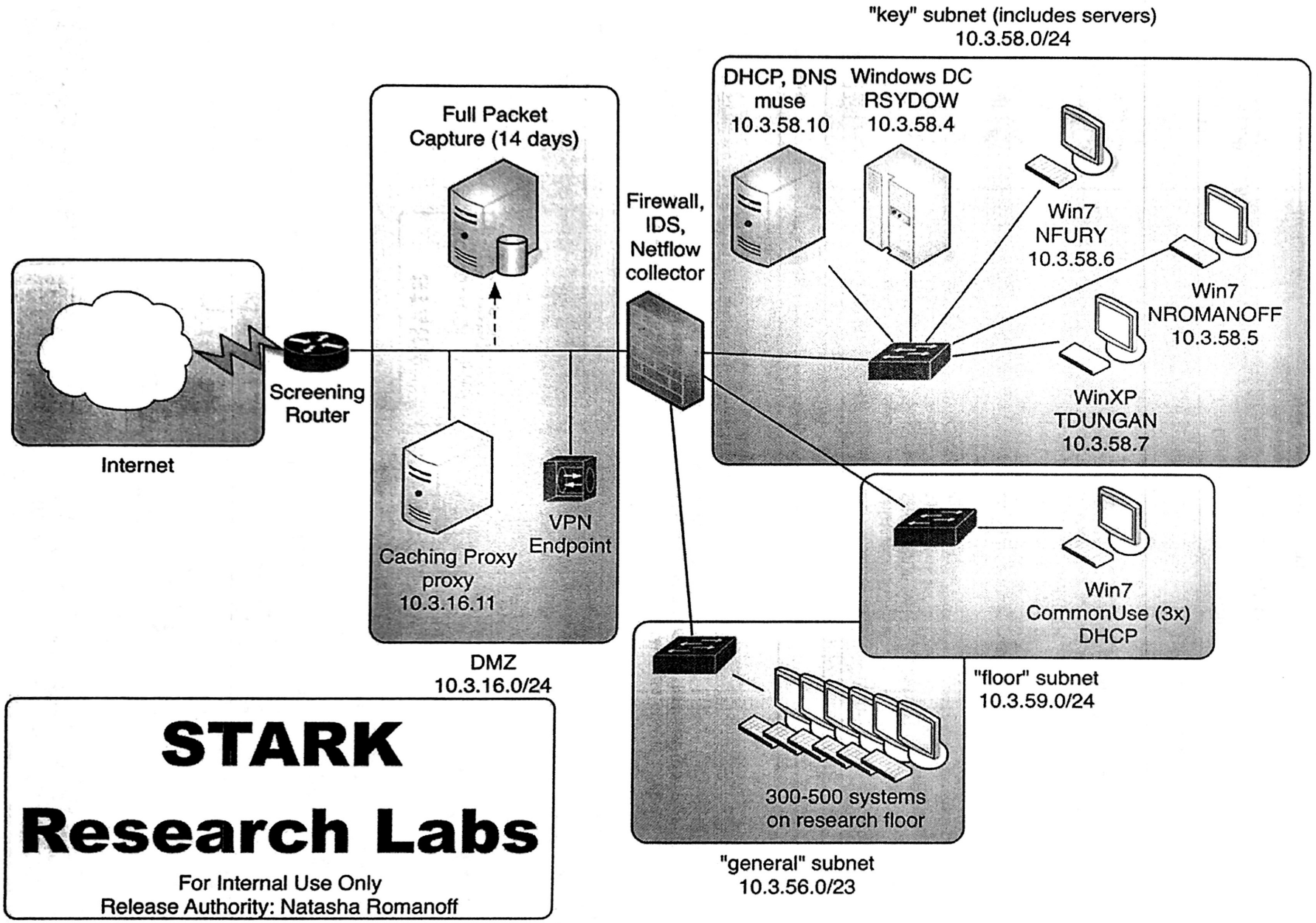


The SRL network is representative of many medium-sized enterprises, with several major internal segments separated by a single firewall, but very little network protection between hosts once on the inside of that perimeter.

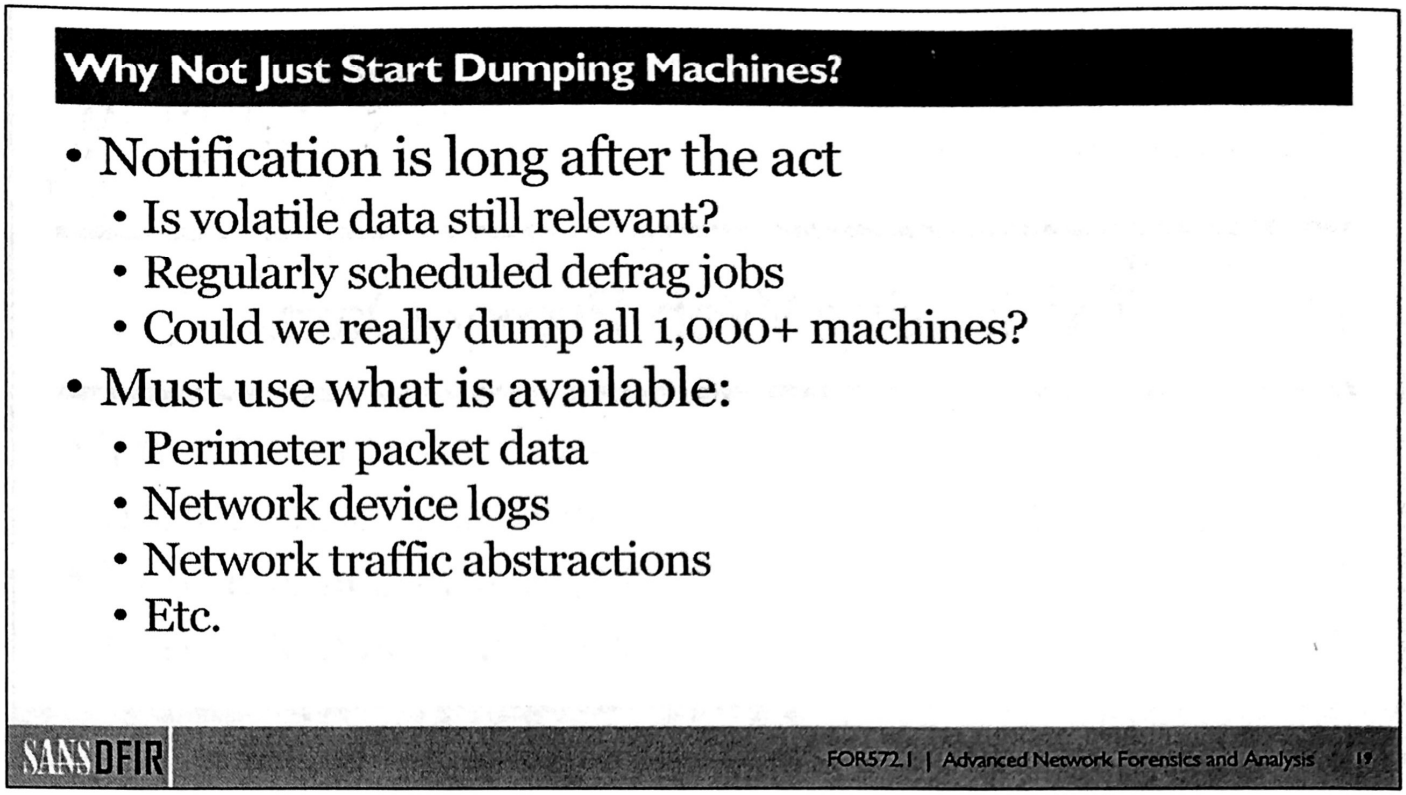
Key systems we are aware of (not an exhaustive list) include:

Host	IP Address	Role
Router	10.3.16.1	Inside interface of Internet screening router
Firewall	10.3.56.1/23	General administration and users
	10.3.58.1/24	Key users & SHIELDBASE servers subnet
	10.3.59.1/24	Production floor systems
	10.3.16.99/24	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	10.3.56.254/24	Tcpdump rotating buffer. Approx 14 days of storage
Webserver	10.3.16.3/24	External presence of SRL
rsydow	10.3.58.4	SHIELDBASE Domain Controller—Windows 2008r2
tdungan	10.3.58.7	Tim Dungan's R&D workstation
nromanoff	10.3.58.5	Natasha Romanoff's workstation
nfury	10.3.58.6	Nick Fury's workstation

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



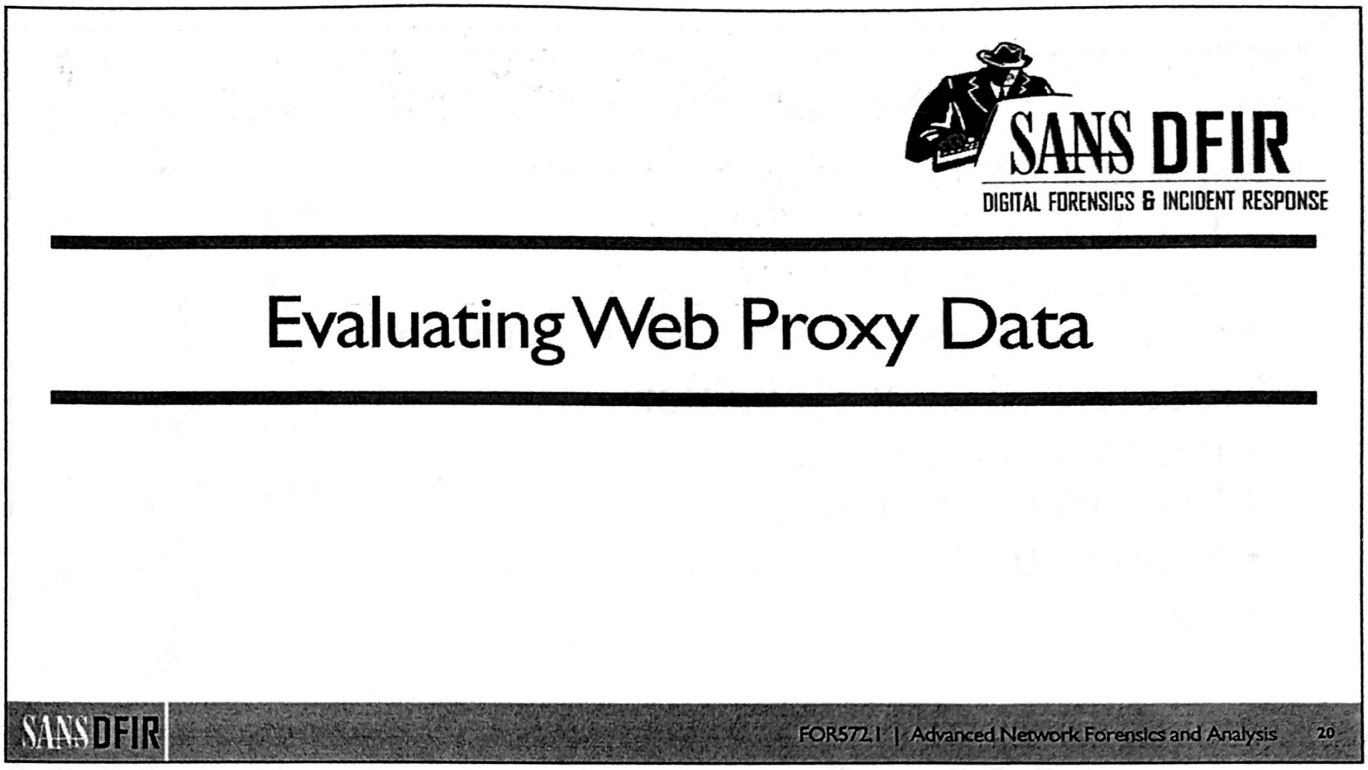
© 2017 Lewes Technology Consulting, LLC, Mat Oldham



The incident response team may be ready to spring into action, but there are some challenges to their effectiveness, given such a vague basis for the investigation. Automatic patching and rebooting are necessary to maintain a proper defensive network posture, but such actions flush volatile data and often overwrite valuable filesystem evidence. An alarmingly small number of breaches are detected with Antivirus or intrusion detection systems, so simply looking for quarantines or poring over IDS alerts probably won't yield much value. The sheer scope of a victim's network environment also creates a significant hurdle—effectively and efficiently searching thousands of machines is seldom feasible. Although new technologies and services aim to improve an organization's ability to quickly scope an incident, many organizations have not yet deployed them.

In this case, we must start with what evidence we have immediately available. This could come in many, varied forms, and our efficiency, effectiveness, and ultimate success will depend on how well we can integrate these sources 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

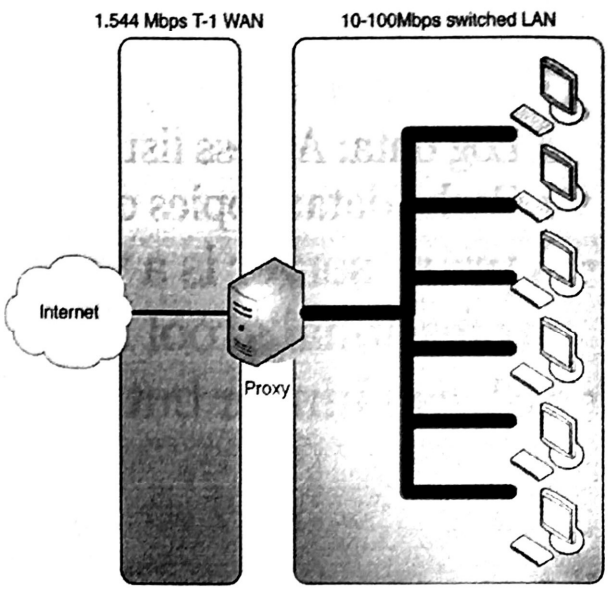


This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Web Proxy Servers (1)

- A server acting on behalf of a client
  - Most common for HTTP
  - Concepts apply to other caches/proxies as well
- Traditionally used for performance reasons
- Valuable accounting of network-based activity!



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 (say, http://sans.org), 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, or cache, 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.

It should also be noted that while we specifically discuss web proxy servers in this section, the same investigative concepts and processes generally apply to platforms that optimize other protocols' delivery as well.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Web Proxy Servers (2)

- Serve both proactive and post-incident purposes
  - Prevent "bad" things
  - Log data: Access list for all HTTP(S) traffic
  - Cache data: Copies of objects sent via HTTP(S)
- A proxy server is a versatile ally in the security professional's tool kit
- SSL can hinder but is a solvable problem

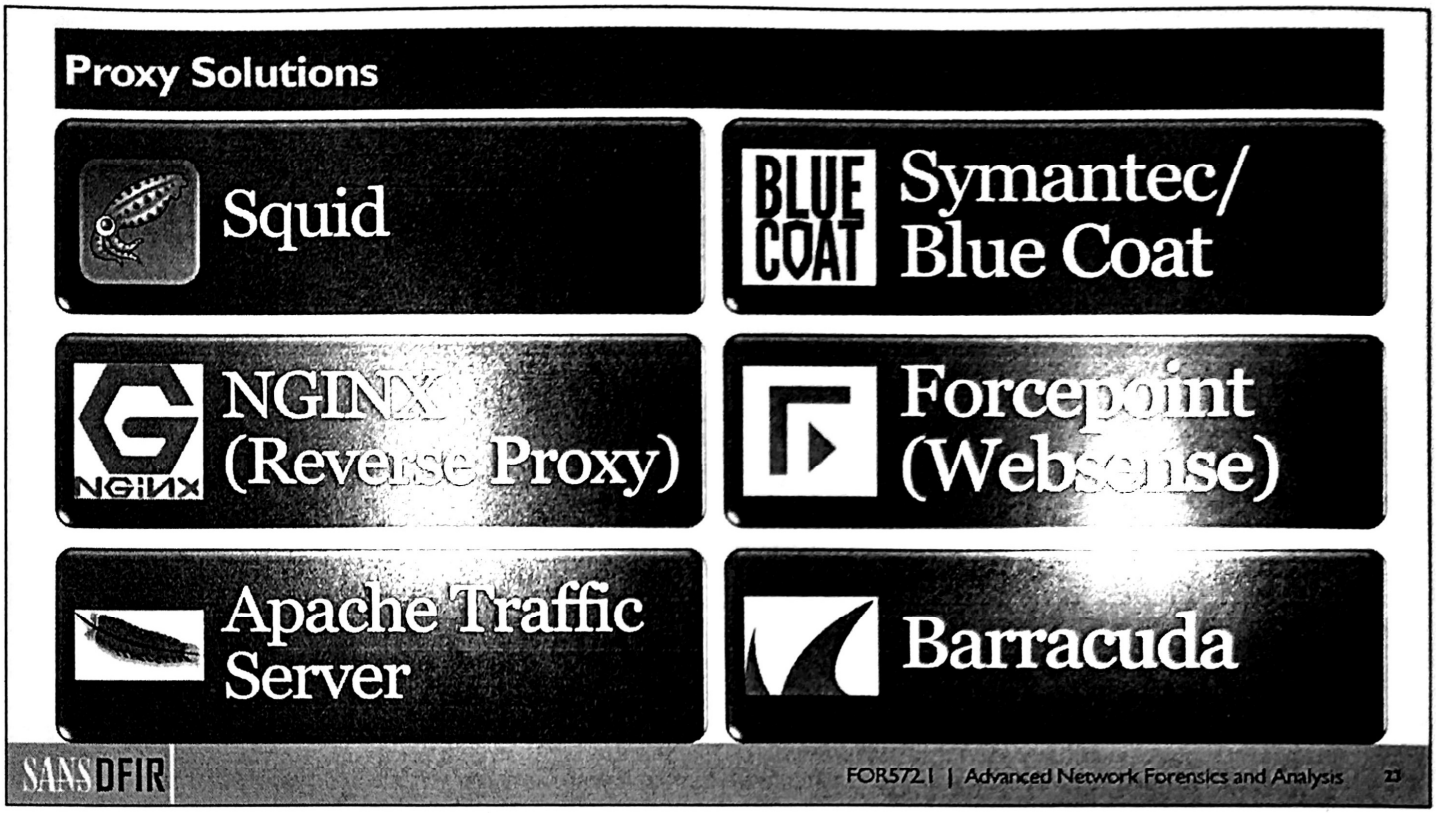
As with many infrastructure devices, the security movement also identified value in the data a proxy server often holds. Network administrators can configure proxy servers to block undesirable content, preventing their client systems from accessing prohibited subject matter. In addition, the "gatekeeper" 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 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 which 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 re-configure 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



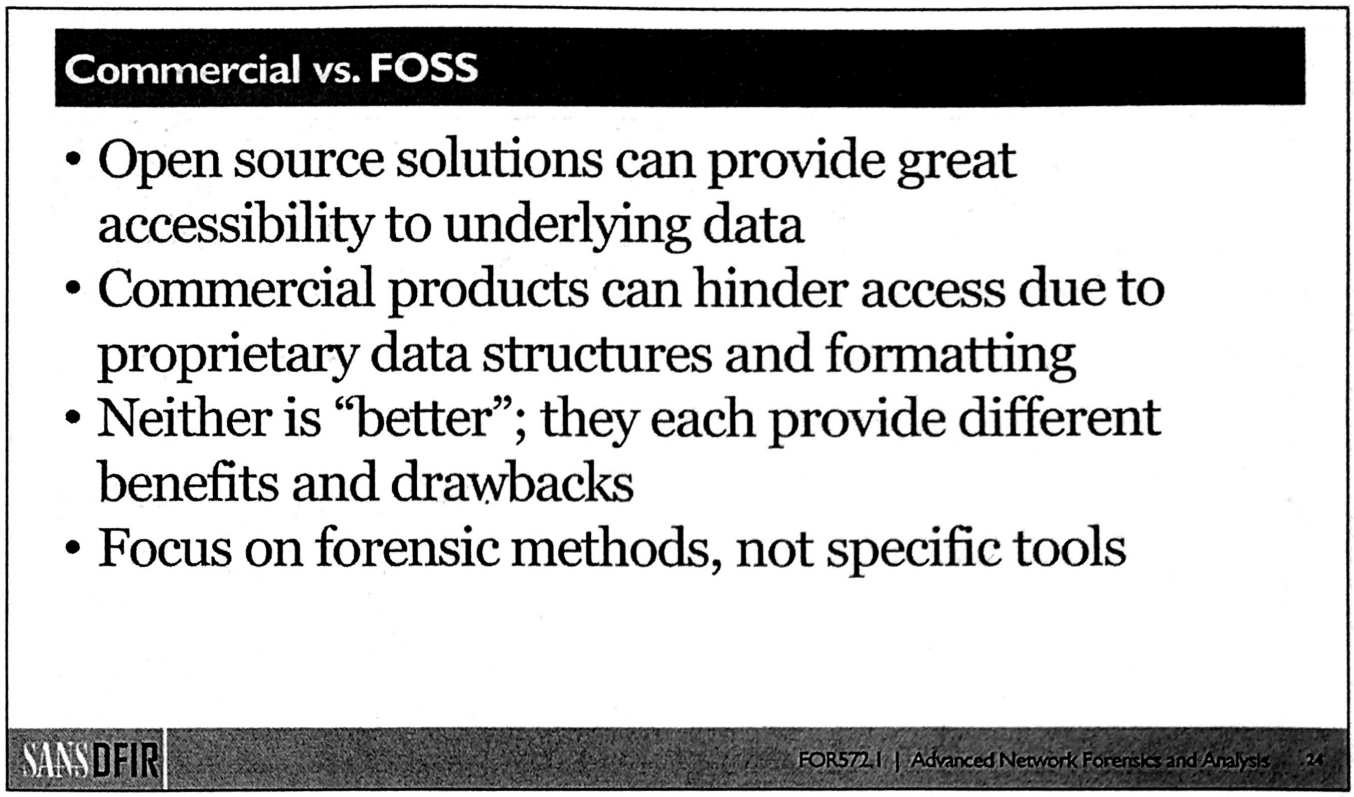
There are a variety of proxy solutions available today, in both FOSS and commercial forms.

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 proxying, as discussed in the previous slide. Websense (now part of Forcepoint) 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.

© 2017 Lewes Technology Consulting, LLC, 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid Proxy Server

- One of the most common proxy servers
  - Free, relatively easy to deploy but flexible enough for complex deployments
- Three main forensically relevant elements:
  - Configuration file: `/etc/squid/squid.conf`
  - Log file(s): `/var/log/squid/*`
  - Cache data: `/var/spool/squid/`

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!)

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid: Configuration File

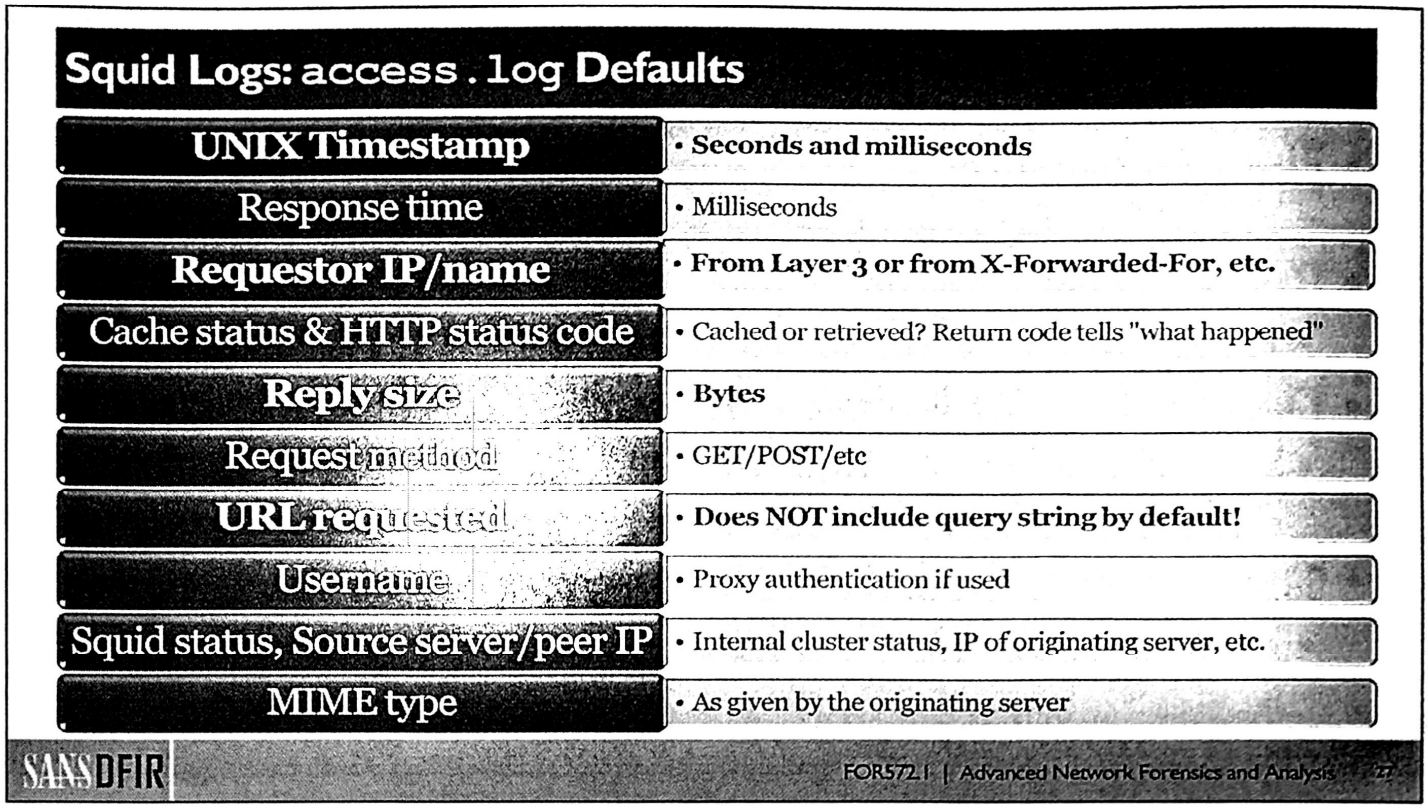
- Network presence
  - `http_port 3128`
- Access controls
  - `acl localnet src 192.168.0.0/255.255.0.0`
  - `acl developers src 172.16.18.0/24`
- Log and cache locations
  - `access_log /var/log/squid/access.log squid`
  - `cache_dir ufs /var/spool/squid 100 16 256`

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.

Squid administrators can create access control lists to aid in creating fine-grained behaviors based on numerous characteristics of proxy traffic. Beside 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 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.

© 2017 Lewes Technology Consulting, LLC, 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 spearphishing 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):

```
1339038269.433     531 192.168.75.19 TCP_MISS/200 17746  GET
http://www.nu.nl/ - DIRECT/62.69.184.21 text/html
1339038295.705     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
```

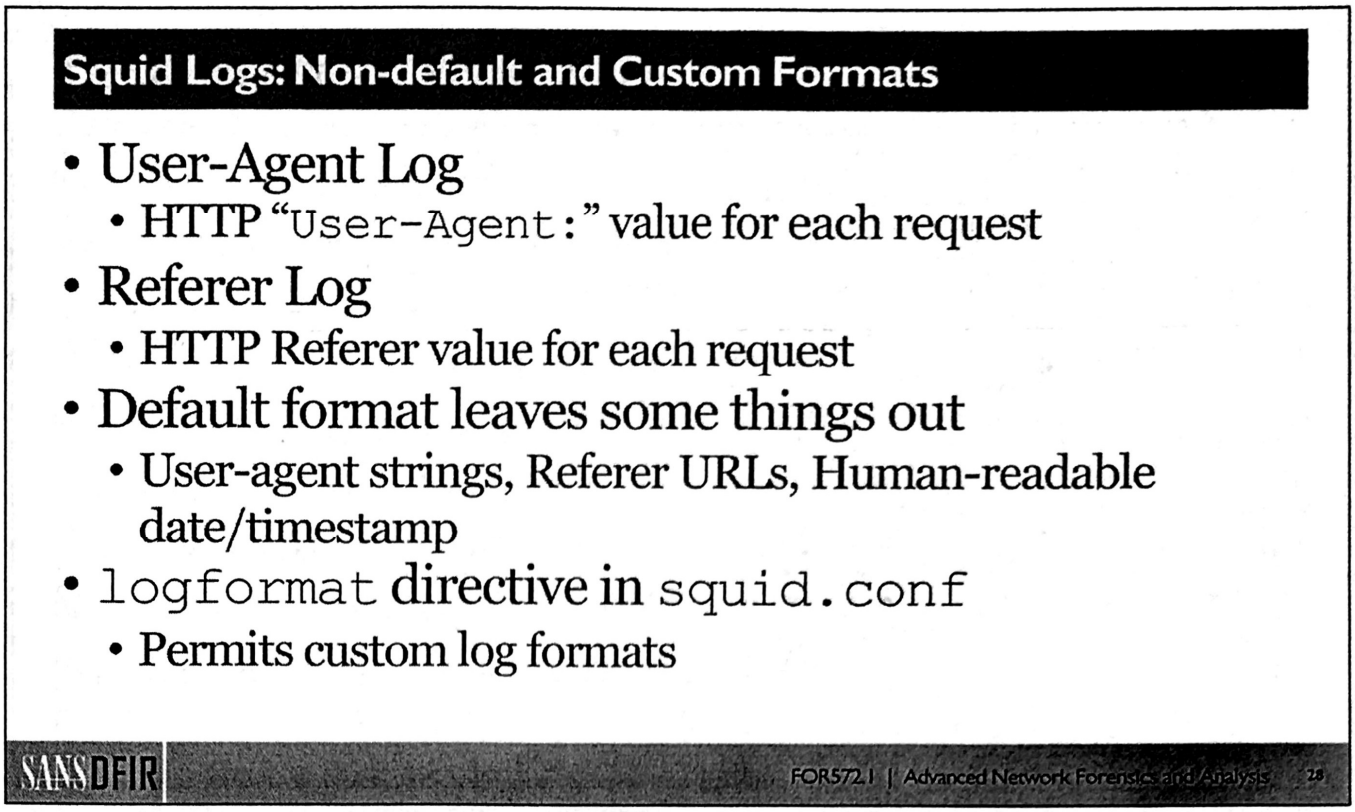
WARNING: Enabling this option has privacy implications for users behind the proxy! Be sure to consult with the proper authorizing parties before using it.

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`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



Although not enabled by default, the optional User-agent log can be quite useful in Network Forensics. This file contains one entry per cache request, containing the client's IP address, date and time, and full "User-Agent" string presented by the client software. An example is below:

```
192.168.75.19 [07/Jun/2012:00:09:40 -0400] "Mozilla/5.0 (Macintosh; Intel Mac
OS X 10_7_4) AppleWebKit/5d.5 (KHTML, like Gecko) Chrome/19.0.1084.54
Safari/536.5"
```

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.

Another non-default log file that can provide value during the forensic process is the referer log. (Note that this is basically an accepted misspelling. The HTTP developers used one "r" and it stuck.) 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.

The default `access_log` format is good, but sometimes just doesn't address all of our requirements. As described previously, 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

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. In fact, Squid version 2.6 introduced the ability to create multiple simultaneous log destinations, and even to shape the contents of each log file using standard Squid 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.

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

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

A sample log entry using this format is below:

```
02/Jun/2012:13:13:52 +0000 192.168.224.101 "GET
http://ia.media-
imdb.com/images/M/MV5BMTU5OTMwNjU4OF5BMl5BanBnXkFtZTcwOTM5OTg5NA@@._V1._CR332
,0,1384,1384_SS105_.jpg
HTTP/1.1" 200 4350 "http://www.imdb.com/title/tt0285351/"
"Mozilla/5.0 (X11; Linux i686; rv:7.0.1) Gecko/20100101
Firefox/7.0.1" TCP_MISS:DIRECT
```

You can see that the log entry contains a human-readable UTC date, as well as the requested URL (http://ia.media-imdb.com/.....jpg), referer URL (http://www.imdb.com/title/tt0285351/), and "User-Agent" string (Mozilla/5.0.....Firefox/7.0.1). The full range of options for this directive is quite extensive.

References:

http://for572.com/fc60a

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 to the data we have available.

We'll only look at `calamaris` to illustrate the kind of analysis that is possible. However, please take a look at the value each can provide, then use (and 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`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid Analysis: calamaris (1)

- Console utility for common aggregation
- Many different kinds of report types
  - Including HTML or e-mail
- Parses data from STDIN
  - Easy to use with grep and shell scripts
  - Expects default log format–adjust as needed

```
$ sudo grep \.gif /var/log/squid/access.log | calamaris -a | less
$ sudo grep evil.cc /var/log/squid/access.log | calamaris -a | less
```


The `calamaris` tool is a straightforward yet powerful tool that parses Squid's `access.log` file, and provides a variety of aggregated reporting data. It is a console utility, so it's ideal to use over an SSH connection right on the proxy server if needed. This also makes it ideal to run as a part of an automated script. These scripts can even be run from the system's `cron` scheduler, delivering periodic reports via e-mail.

There are a multitude of different report types available. Using the "`-a`" option will run them all, but you can pick and choose, tailoring the results to your needs. In your SIFT Workstation VM, you can run "`calamaris -h`" to see a list of the available options.

By default, calamaris parses data from STDIN, making it ideal for typical shell output redirection.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid Analysis: calamaris (2)

```
# Request-destinations by 2nd-level-domain
destination                          request     %   hit-%    Byte      %   hit-%
-------------------------------- --------- ------ ------ -------- ------ ------
*.turner.com                           713   4.11  25.81  9999192   3.36  19.08
*.doubleclick.net                      680   3.92   1.76  1122687   0.38   2.45
*.redditmedia.com                      665   3.83   0.00  1358663   0.46   0.00
*.yimg.com                             489   2.82   5.73  5587726   1.88   7.72
*.cnn.com                              484   2.79   0.83  2151485   0.72   0.57
*.scorecardresearch.com                469   2.70   2.35   273824   0.09  10.85
*.google.com                           313   1.80   5.43 10299131   3.46   1.27
*.yahoo.com                            300   1.73   1.33  1001825   0.34   0.38
*.foxnews.com                          266   1.53   0.00  1191855   0.40   0.00
...
*.livefyre.com                         204   1.17  92.16   245848   0.08  65.90
69.31.72.*                             201   1.16   0.00   135855   0.05   0.00
*.reuters.com                          201   1.16  39.30  1464292   0.49  45.48
*.adzerk.net                           198   1.14   0.00  1675468   0.56   0.00
other: 471 2nd-level-domains         10534  60.67   9.54   245163K 84.39   4.67
-------------------------------- --------- ------ ------ -------- ------ ------
Sum                                  17364 100.00   9.42   290521K 100.00   5.31
```


One of the more relevant reports that `calamaris` provides is a histogram by second-level domain. The request count and byte count breakdown can help to point out large data transfers, such as a tool download, or frequent small-volume transfers that could indicate a command and control channel.

The important thing to consider when reviewing these reports is that we are looking for indicators, not smoking guns. Any reputable criminal, nation state actor, or other attacker isn't going to call out to "`*.badguy.com`", but they will need to communicate across the network perimeter somehow. Because that is often using HTTP, these reports can be helpful in identifying the scope of and later remediating a compromise incident.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid Analysis: calamaris (3)

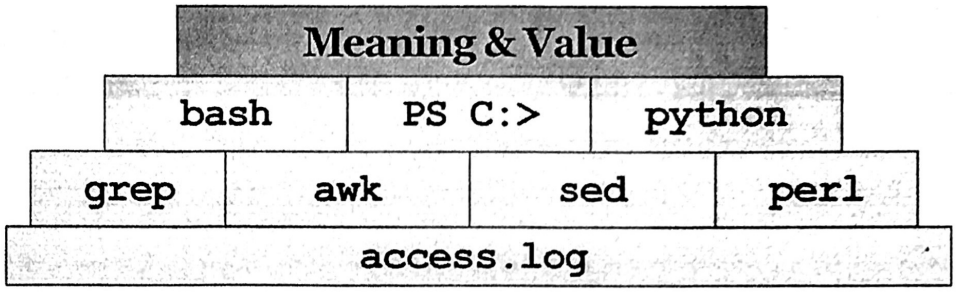
```
# Incoming TCP-requests by host
host                          request   hit-% sec/req    Byte    hit-% kB/sec
------------------------------ -------- ------ ------- -------- ------ -------
10.3.59.158                       7005   8.12    0.34  140904K   4.83   59.93
10.3.59.159                       6695  12.26    0.28 65724268   8.79   34.68
10.3.59.157                       3664   6.69    0.30 87483325   3.48   76.54
------------------------------ -------- ------ ------- -------- ------ -------
Sum                              17364   9.42    0.31  290521K   5.31   54.63
```


This section of the `calamaris` report details the access count and related metrics broken down by client. This would be very useful to identify hosts making a large number of request counts with a small byte-per-request ratio (possibly C2 activity) or very large transfer volumes.

© 2017 Lewes Technology Consulting, LLC, 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.

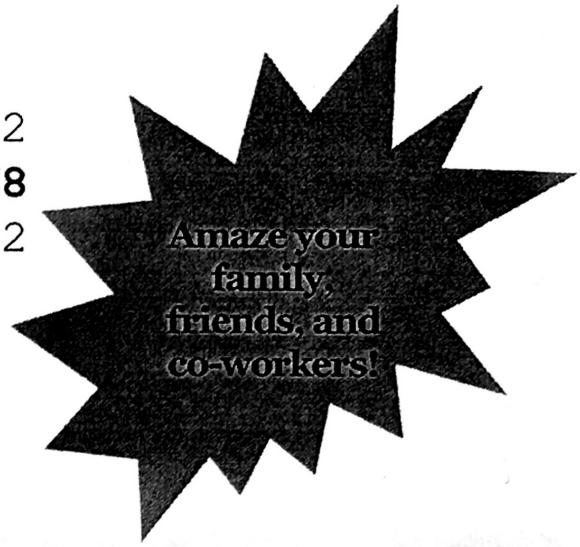
Automated tools are excellent to narrow down vast amounts of source data to a form that makes identifying outliers easy. We use that machine guidance to lead human resources to the data that will help answer the questions at hand.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Timestamp Überhint #1!

- Use elemental Linux* commands for fame, fun, and profit:

```
$ date -d @1339038798.368
Wed Jun  6 23:13:18 EDT 2012
$ date -u -d @1339038798.368
Thu Jun  7 03:13:18 UTC 2012
```



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 @1339038798.368
Wed Jun  6 23:13:18 EDT 2012
$ date -u -d @1339038798.368
Thu Jun  7 03:13:18 UTC 2012
```

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 1339038798
Wed Jun  6 23:13:18 EDT 2012
$ date -j -u -r 1339038798
Thu Jun  7 03:13:18 UTC 2012
```

© 2017 Lewes Technology Consulting, LLC, 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}'
2012-06-09 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
2012-06-09 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}'
2012-06-09 12:30:37 243 192.168.75.119 TCP_MISS/200 17962 GET
http://www.google.com/ - DIRECT/173.194.73.105 text/html
2012-06-09 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: Intro

- Illustrate value of proxy logs during network forensic process:
  - Squid handles all perimeter-crossing HTTP traffic
  - Query terms and user agents ARE logged
  - SSL proxy IS NOT enabled
  - 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, very guilty.

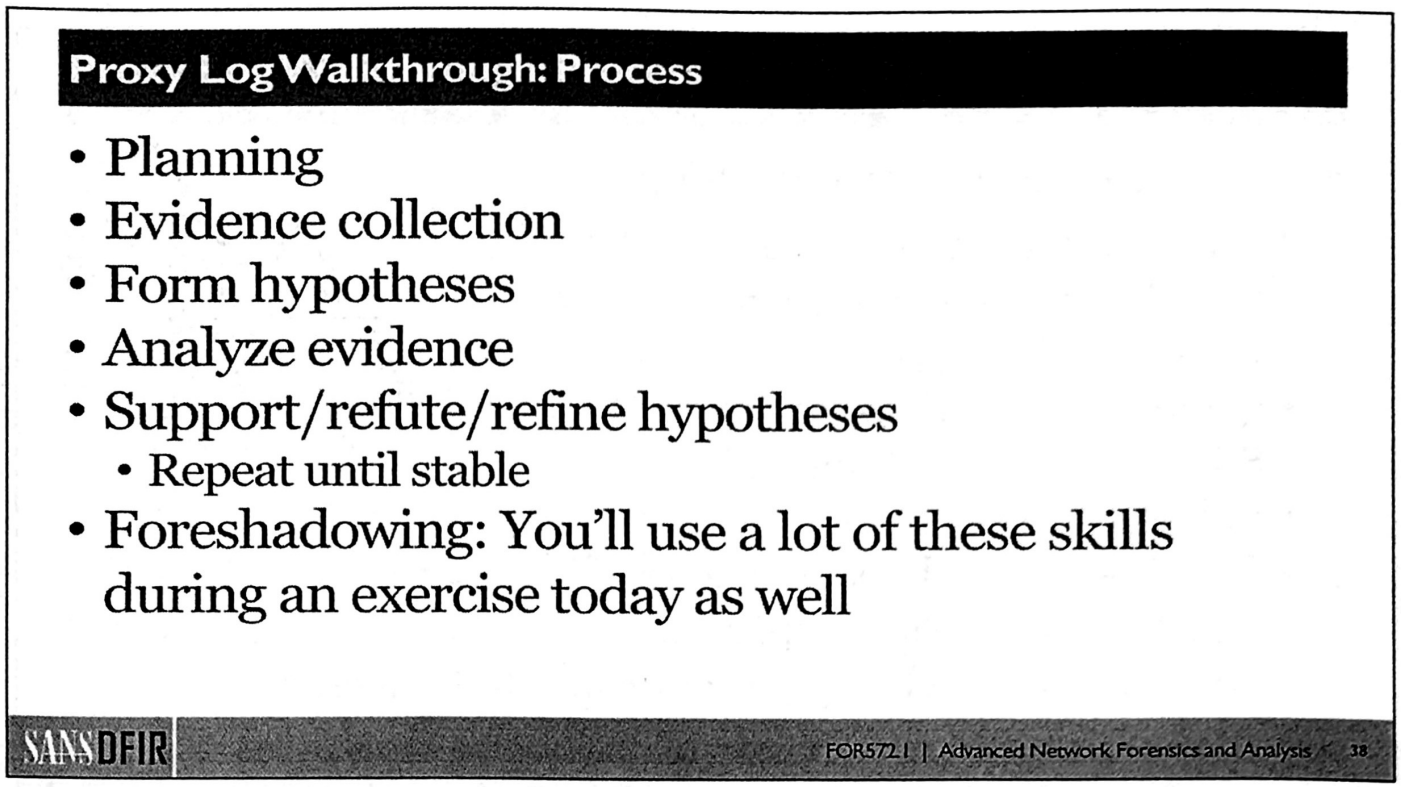
The files for this walkthrough are available on your course USB drive in the `/exercise_data/Demo-01/` directory, so don't just watch, follow along—call out if you think you find something that looks interesting or out of place.

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. User-Agent and referer strings are logged into separate files. SSL proxying has NOT been deployed.

In this scenario, there was an existing suspicion that an employee is 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 Tokyo (United 803 IAD->NRT) due to an exceedingly large sum of cash in his luggage that had not been properly declared. He also had a ticket from Tokyo to Singapore (All Nippon Airways 7051 NRT->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.

© 2017 Lewes Technology Consulting, LLC, 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.

Discuss within class (or write down on your own) 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: squid.conf

- Contents of `/etc/squid/squid.conf`
  - See Notes page
- Default except for:
  - Added `useragent_log` directive
  - Added logging of HTTP query strings


```
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl to_localhost dst 127.0.0.0/8
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

http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports

http_access allow localhost
http_access allow all

icp_access allow all

http_port 3128
```

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

```
hierarchy_stoplist cgi-bin ?

access_log /var/log/squid/access.log squid
useragent_log /var/log/squid/useragent.log
referer_log /var/log/squid/referer.log

acl QUERY urlpath_regex cgi-bin \?
cache deny QUERY

refresh_pattern ^ftp:           1440    20%     10080
refresh_pattern ^gopher:        1440    0%      1440
refresh_pattern .               1440    75%     4320 ignore-no-cache ignore-
private

acl apache rep_header Server ^Apache
broken_vary_encoding allow apache

visible_hostname testvm

coredump_dir /var/spool/squid

strip_query_terms off
```


© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: access.log Analysis (1)

```
$ cd /cases/for572/demo-01/Demo-01_source_evidence/var/log/squid/
$ calamaris -a access.log
...
# Request-destinations by 2nd-level-domain
destination                    request      %   hit-%    Byte      %   hit-%
------------------------------ -------- ------ ------ -------- ------ ------
*.google.com                       213  12.41   3.76  6599403  33.29   0.78
*.mnn.com                          174  10.13   0.00  1130407   5.70   0.00
*.blogsmithmedia.com               102   5.94   0.00   301205   1.52   0.00
...
*.yahoo.com                         36   2.10   0.00   177014   0.89   0.00
*.pointroll.com                     30   1.75   0.00   361350   1.82   0.00
...
*.2mdn.net                          24   1.40   4.17   381528   1.92   0.22
*.facebook.com                      21   1.22   0.00    94259   0.48   0.00
*.wikimedia.org                     20   1.16   0.00   221682   1.12   0.00
*.tsa.gov                           19   1.11   0.00   314270   1.59   0.00
other: 104 2nd-level-domains       521  30.34   0.77  5122270  25.84   0.32
------------------------------ -------- ------ ------ -------- ------ ------
Sum                               1717 100.00   2.04 19822828 100.00   0.40
```


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!

First, we see the period of activity covered by the logs. In this case, about 23 minutes on June 12, 2012. In the scenario, we have access to the host-based forensic triage data, so we would want to ensure these values are consistent with those findings.

The `calamaris` tool provides a number of report sections, but we've restricted the view here to just the second-level domain summary section. He can clearly see that the employee accessed Google quite a bit, but also seemed to use Yahoo!, though much less frequently. There is also a tsa.gov result, which may or may not be relevant given the circumstances of the employee's apprehension.

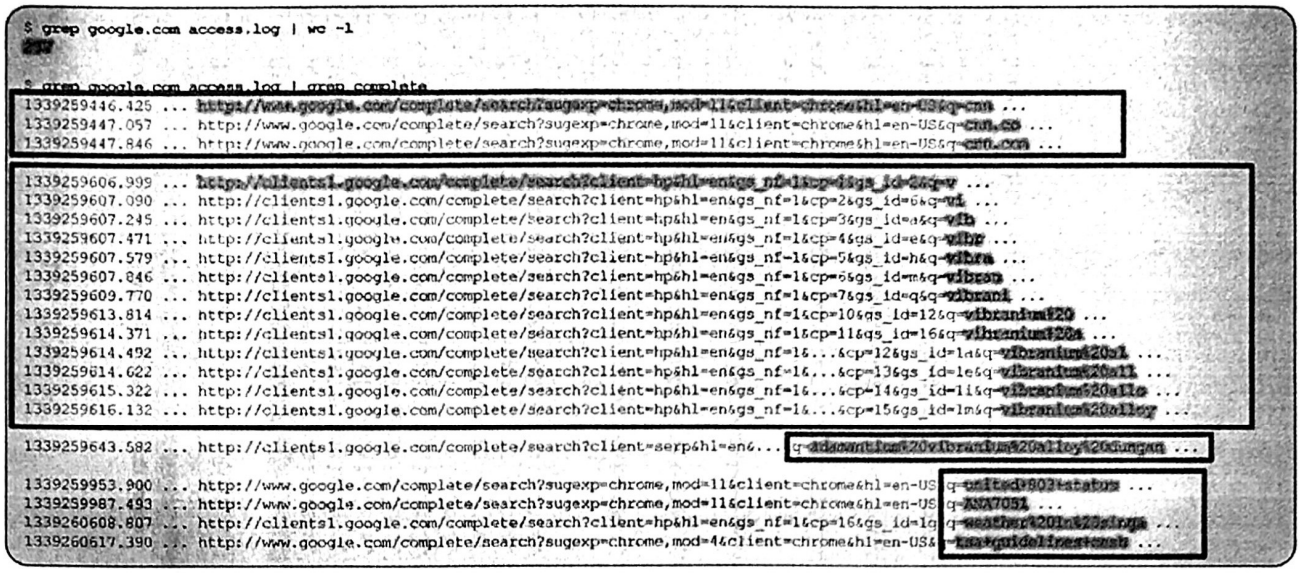
© 2017 Lewes Technology Consulting, LLC, Mat Oldham

```
$ cd /cases/for572/demo-01/Demo-01_source_evidence/var/log/squid/
$ calamaris -a access.log
...
# Request-destinations by 2nd-level-domain
```

destination	request	%	hit-%	Byte	%	hit-%
*.google.com	213	12.41	3.76	6599403	33.29	0.78
*.mnn.com	174	10.13	0.00	1130407	5.70	0.00
*.blogsmithmedia.com	102	5.94	0.00	301205	1.52	0.00
...						
*.yahoo.com	38	2.10	0.00	177014	0.89	0.00
*.pointroll.com	30	1.75	0.00	361350	1.82	0.00
...						
*.2mdn.net	24	1.40	4.17	381528	1.92	0.22
*.facebook.com	21	1.22	0.00	94259	0.48	0.00
*.wikimedia.org	20	1.16	0.00	221682	1.12	0.00
*.nsa.gov	19	1.11	0.00	314270	1.59	0.00
other: 104 2nd-level-domains	521	30.34	0.77	5122270	25.84	0.32
Sum	1717	100.00	2.04	19822828	100.00	0.40

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: access.log Analysis (2)



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. The diagram here shows a selection from the 61 refined results, but the pattern of queries is distinct. In the first set, we see the employee typing an URL in the browser's Location bar. Google's search autocomplete feature gives us a keylogger-like record of the employee's activity! (Good thing we have those signed Acceptable Use Policy documents, isn't it?)

The second block of records has a different structure—they are the results of the employee's Google searches within the browser content window, rather than the Location bar. Again, we see the query being built a few characters at a time.

The last five records were also built a few characters at a time, but we've abbreviated them here to save page space. Given the context of the employee's apprehension, the presence of flight status for two different flights (United 803 is IAD->NRT, ANA 7051 is NRT->SIN), and research on "tsa guidelines cash" are definitely significant.

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 a letter at a time.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

```
$ grep google.com access.log | wc -l
237

$ grep google.com access.log | grep complete
1339259446.425 ... http://www.google.com/complete/search?sugexp=chrome,mod=11&client=chrome&hl=en-US&q=cnn ...
1339259447.057 ... http://www.google.com/complete/search?sugexp=chrome,mod=11&client=chrome&hl=en-US&q=cnn.co ...
1339259447.846 ... http://www.google.com/complete/search?sugexp=chrome,mod=11&client=chrome&hl=en-US&q=cnn.com ...

1339259606.999 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=1&gs_id=2&q=v ...
1339259607.090 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=2&gs_id=6&q=vi ...
1339259607.245 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=3&gs_id=a&q=vib ...
1339259607.471 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=4&gs_id=e&q=vibr ...
1339259607.579 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=5&gs_id=h&q=vibra ...
1339259607.846 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=6&gs_id=m&q=vibran ...
1339259609.770 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=7&gs_id=q&q=vibrani ...
1339259613.814 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=10&gs_id=12&q=vibranium%20 ...
1339259614.371 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=11&gs_id=16&q=vibranium%20a ...
1339259614.492 ... http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&...&cp=12&gs_id=1a&q=vibranium%20a1
...
1339259614.622 ...
http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&...&cp=13&gs_id=1e&q=vibranium%20all ...
1339259615.322 ...
http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&...&cp=14&gs_id=1i&q=vibranium%20allo ...
1339259616.132 ...
http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&...&cp=15&gs_id=1m&q=vibranium%20alloy ...

1339259643.582 ...
http://clients1.google.com/complete/search?client=serp&hl=en&...&q=adamantium%20vibranium%20alloy%20dungan ...

1339259953.900 ... http://www.google.com/complete/search?sugexp=chrome,mod=11&client=chrome&hl=en-US&q=united+803+status
...
1339259987.493 ... http://www.google.com/complete/search?sugexp=chrome,mod=11&client=chrome&hl=en-US&q=ANA7051 ...
1339260608.807 ...
http://clients1.google.com/complete/search?client=hp&hl=en&gs_nf=1&cp=16&gs_id=1q&q=weather%20in%20singa ...
1339260617.390 ... http://www.google.com/complete/search?sugexp=chrome,mod=4&client=chrome&hl=en-US&q=tsa+guidelines+cash
...
```

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: Create Derivative Log Files

- Create derived evidence files
  - Human-readable timestamps in working copy of log file makes human analysis much easier

```
$ cat access.log | awk '{$1=strftime("%F %T", $1, 1); print $0}' > ⏎
  ~/humantime_access.log
$ cat referer.log | awk '{$1=strftime("%F %T", $1, 1); print $0}' > ⏎
  ~/humantime_referer.log
```

```
$ cat access.log | squidtime > ~/humantime_access.log
$ cat referer.log | squidtime > ~/humantime_referer.log
```

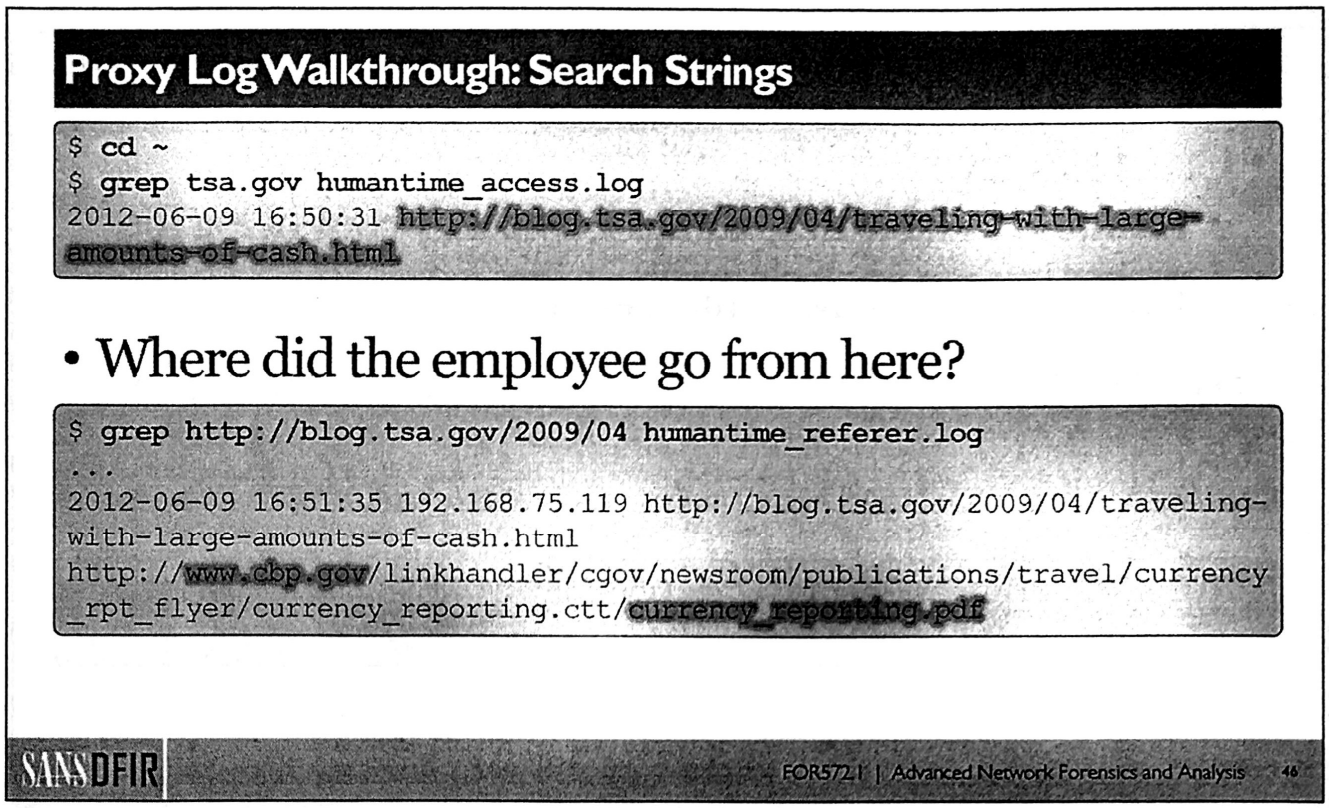

It seems we'll be doing some manual parsing of the log files, so it probably makes sense to create derived files containing the same evidence—only with human readable timestamps.

```
$ cat access.log | awk '{$1=strftime("%F %T", $1, 1); print $0}' > ⏎
  ~/humantime_access.log
$ cat referer.log | awk '{$1=strftime("%F %T", $1, 1); print $0}' > ⏎
  ~/humantime_referer.log
```

Note that on your SIFT Workstation VM, the `awk` portion of this command has been aliased and can be invoked simply by using"`squidtime`".

```
$ cat access.log | squidtime > ~/humantime_access.log
$ cat referer.log | squidtime > ~/humantime_referer.log
```

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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

```
2012-06-09 16:50:31 http://blog.tsa.gov/2009/04/traveling-with-large-
amounts-of-cash.html
```

We can visit this page to see what is there, but let's also use the `referer.log` file to determine if the employee clicked any links on that TSA Blog page:

```
# grep http://blog.tsa.gov/2009/04 humantime_referer.log
2012-06-09 12:51:35 192.168.75.119 http://blog.tsa.gov/2009/04/traveling-
with-large-amounts-of-cash.html
http://www.cbp.gov/linkhandler/cgov/newsroom/publications/travel/currency_rpt
_flyer/currency_reporting.ctt/currency_reporting.pdf
```

The format for the referer.log file is: timestamp (shown here after awk conversion), client IP address, referring page, and clicked link. This means that the employee loaded the "`currency_reporting.pdf`" file, which was hosted on the `cbp.gov` (Customs and Border Protection) site. Given the circumstances, this is definitely a relevant finding!

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: Yahoo! Access

- Why was Yahoo! in the calamaris report?

```
$ grep yahoo.com humantime_access.log
...
2012-06-09 16:47:59 ... GET http://mail.yahoo.com/ ...
2012-06-09 16:48:00 ... CONNECT login.yahoo.com:443 ...
2012-06-09 16:48:22 ... GET https://us.mg5.mail.yahoo.com/neo/launch?.rand=[redacted] ...

2012-06-09 16:49:31 ... POST http://us.mg5.mail.yahoo.com/ws/mail/v2.0/jsonrpc?appid=YahooMailNeo&m=[redacted]...
2012-06-09 16:49:31 ... GET http://us.mg5.mail.yahoo.com/neo/darla/php/fc.php?trace=[redacted]tID=2&d=0&f=978500224&...
```

- How can you confirm these observations against the actions that could have caused them?

It would seem that the employee didn't search using Yahoo!, but rather logged into Yahoo! Mail. Recall that we're not proxying SSL traffic, so all we see from the login reflects the "CONNECT" method. This simply means that the proxy passed along the request without inspection, it cannot decrypt the end-to-send SSL session the employee's workstation made with the Yahoo! Mail server.

However, at the time this evidence was collected, Yahoo! Mail only used SSL for login—it reverted back to plaintext for most of its later operations. Although this behavior is decreasingly used, it is still seen on many sites—even very large ones. In this case, we can see the third entry clearly shows the full GET request, which launched the Yahoo! Mail web application itself.

Following the other log entries, we see another that may prove to be interesting—note the "`SendMessage`" POST and "`send_confirm`" string in the final two log entries, respectively. Could the employee have sent out an e-mail containing the intellectual property that we suspect he is leaking?

One way to confirm this would be to establish a "control" capture. By observing the URL patterns during a controlled experiment, we can confirm the nature of our observations. In this case, using a Yahoo! Mail account to send a message would show consistent results with that of our log entries above.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

```
$ grep yahoo.com humantime_access.log
...
2012-06-09 16:47:59 ... GET http://mail.yahoo.com/ ...

2012-06-09 16:48:00 ... CONNECT login.yahoo.com:443 ...

2012-06-09 16:48:22 ... GET http://us.mg5.mail.yahoo.com/neo/launch?.rand=5tmcco1u121vn ...


2012-06-09 16:49:31 ... POST
http://us.mg5.mail.yahoo.com/ws/mail/v2.0/jsonrpc?appid=YahooMailNeo&m=SendMessage&...
2012-06-09 16:49:31 ... GET
http://us.mg5.mail.yahoo.com/neo/darla/php/fc.php?trace=send_confirm&tID=2&d=0&f=978500224&...
```

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: Data Dumping?

- What other suspicious actions are logged?

```
$ less humantime_access.log

2012-06-09 16:47:18 ... TCP_MISS/200 6148 GET http://pastebin.com/ ...

2012-06-09 16:47:49 ... TCP_MISS/302 505 POST http://pastebin.com/post.php ...

2012-06-09 16:47:49 ... TCP_MISS/200 25403 GET http://pastebin.com/aCSeZJVD ...
```


After looking through the logs, we see that immediately before logging into Yahoo! Mail, the employee posted some data to the popular data dumping site `pastebin.com`. What we see in these proxy log entries is the network equivalent to security camera video tape of someone walking out the door with what may be some very valuable 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. In this case, the URL is `http://pastebin.com/aCSeZJVD`. This can be 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 ten minutes, or an hour, day, or month. In this case, the site indicates the paste is no longer available, so we *might* be out of luck…

More on that later.

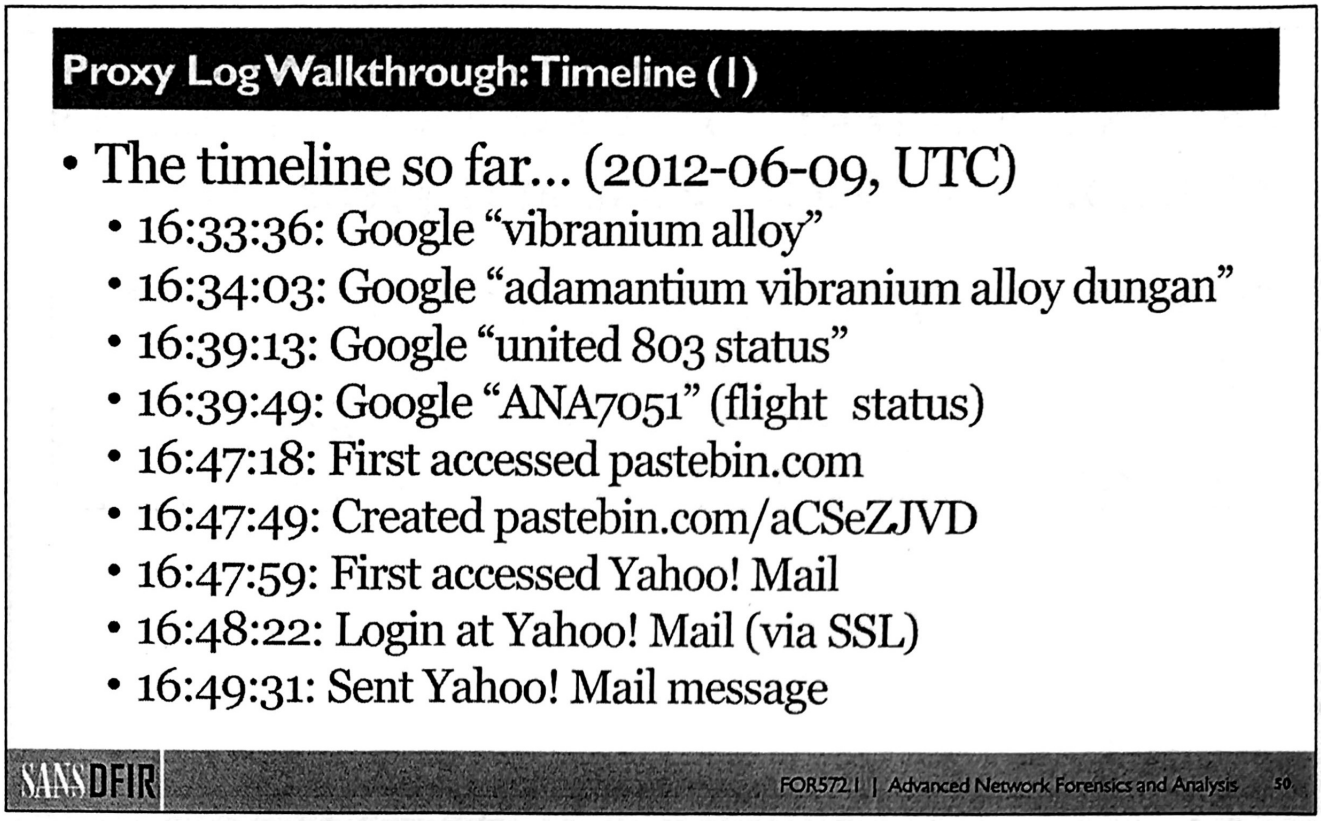
```
$ less humantime_access.log

2012-06-09 16:47:18 ... TCP_MISS/200 6148 GET http://pastebin.com/ ...

2012-06-09 16:47:49 ... TCP_MISS/302 505 POST http://pastebin.com/post.php
...

2012-06-09 16:47:49 ... TCP_MISS/200 25403 GET http://pastebin.com/aCSeZJVD
...
```

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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 timestamps were converted to UTC and all events occurred June 9, 2012.

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 that 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 Tokyo, then Tokyo to Singapore. Singapore is the nearest International airport to Madripoor, the location for the corporate base of operations of the victim company's chief competition.

Approximately 8 minutes later, the employee created a pastebin "paste" with contents not yet identified. The paste was no longer accessible.

After creating the paste, it appears the employee logged into a Yahoo! Mail account. Because Yahoo! only encrypts the login process, but not the user's session, we could see that the employee sent an e-mail message.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Log Walkthrough: Timeline (2)

- 16:50:09: Google for "weather in singapore"
- 16:50:17: Google "tsa guidelines cash"
- 16:50:31: Viewed TSA Blog re: flying with cash
- 16:51:36: Viewed CBP PDF re: reporting carried currency

The employee then checked for the weather in Singapore, which was also the destination for the second flight he previously researched.

Finally, the employee searched for and read about TSA and CBP guidelines and regulations regarding carrying large amounts of cash via air travel and across international borders.

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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Aside: Commercial Proxy Log Analysis

- Commercial products often have different, proprietary log formats
  - Extraction can be a (computer) science project!
  - Look for the rules before interpreting!
- Talk with admins to see if verbose logs are already available

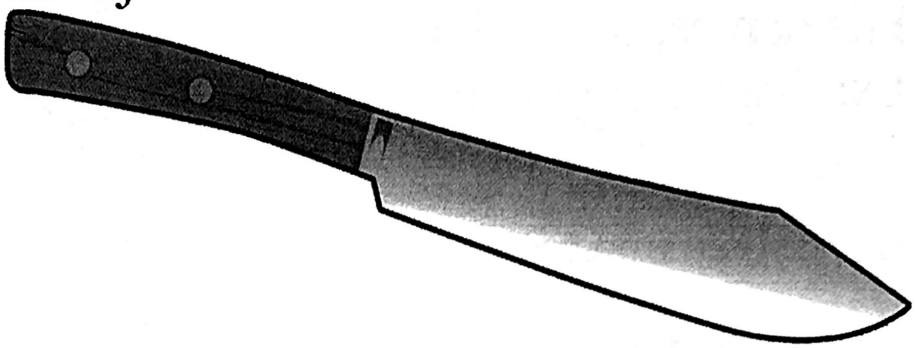
We mentioned that commercial proxy servers are not always as easy to "get into" as an open source proxy like Squid. That doesn't mean that the data isn't available in a useful format, just that it takes some work to get there. For example, enterprises that use the Blue Coat proxy server can batch upload logs via FTP or export through additional third-party connectors like Arcsight and Websense. 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
```


© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Cache Extraction

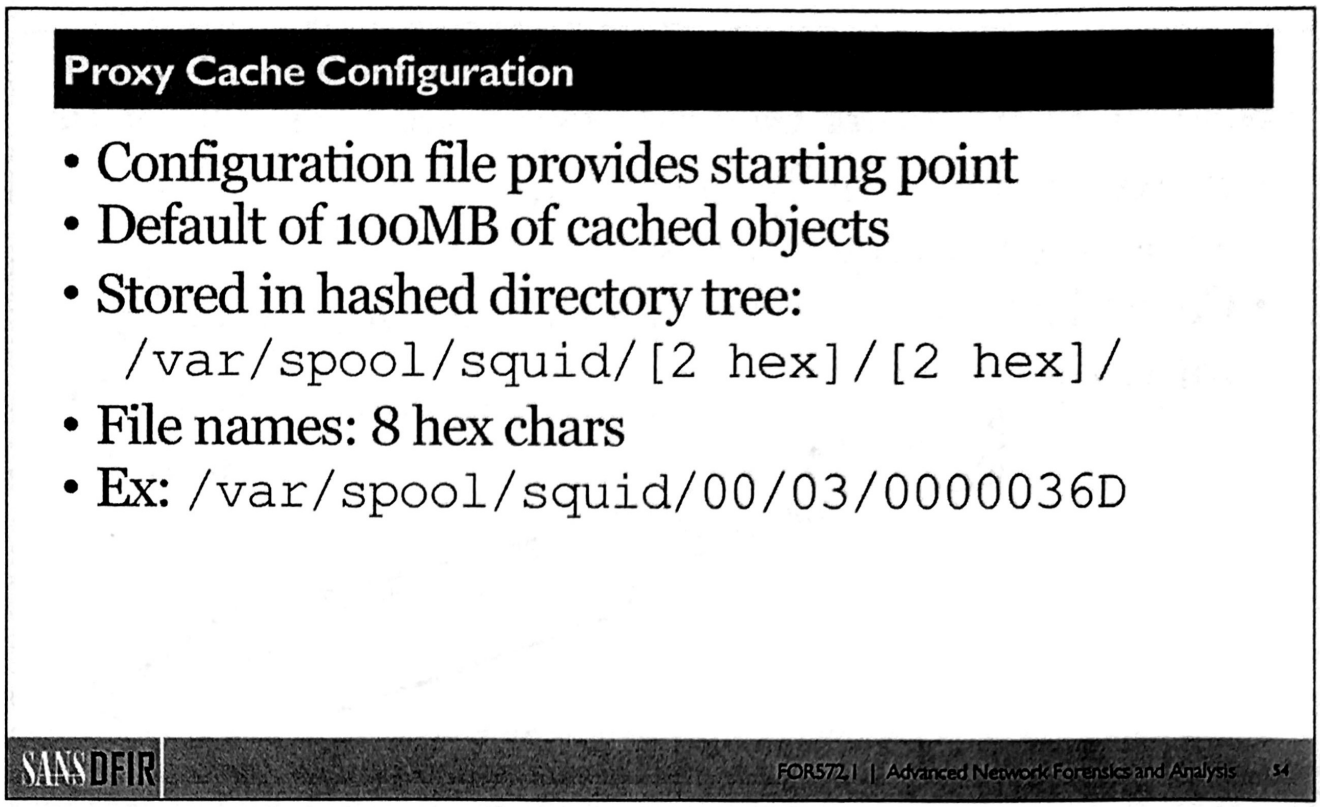
- Proxy servers usually cache data
  - Speeds up subsequent accesses
- Cached items can be deconstructed—or "carved"—into original objects



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.

© 2017 Lewes Technology Consulting, LLC, 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.

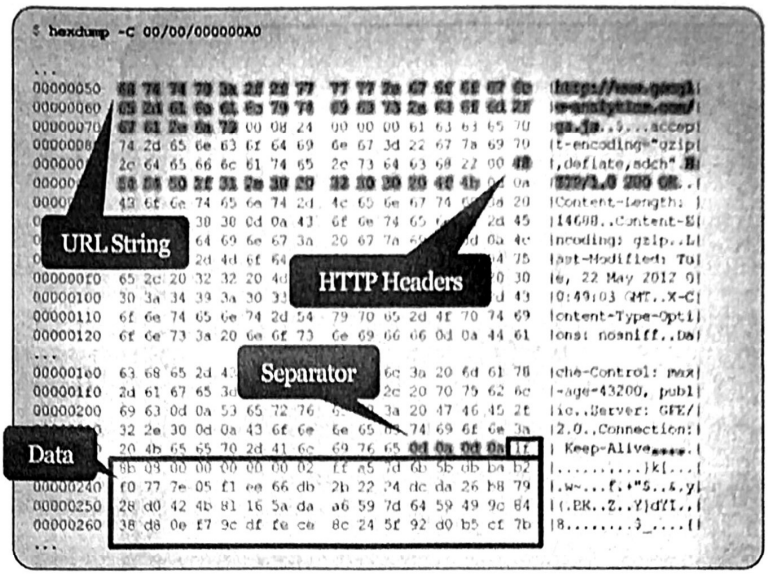
The cache directory seems straightforward, and in most cases, it is left as the default. Note, however, that there can be more than one cache directory specified. In such a case, the server "load balances" cached content among the multiple cache directories. If recovering the cache from a proxy server, 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 allocate for the cache from the baseline of just 100MB.

The cached files themselves are stored in a form such as "`/var/spool/squid/00/03/0000036D`". Unfortunately, such filenames do not give any insight to the origin, file type, size, time, or anything else we might like to see.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid Cache File Structure



- URL where retrieved
- All HTTP headers from transaction
- Header-Data separator (\r\n\r\n)
- Data as delivered!

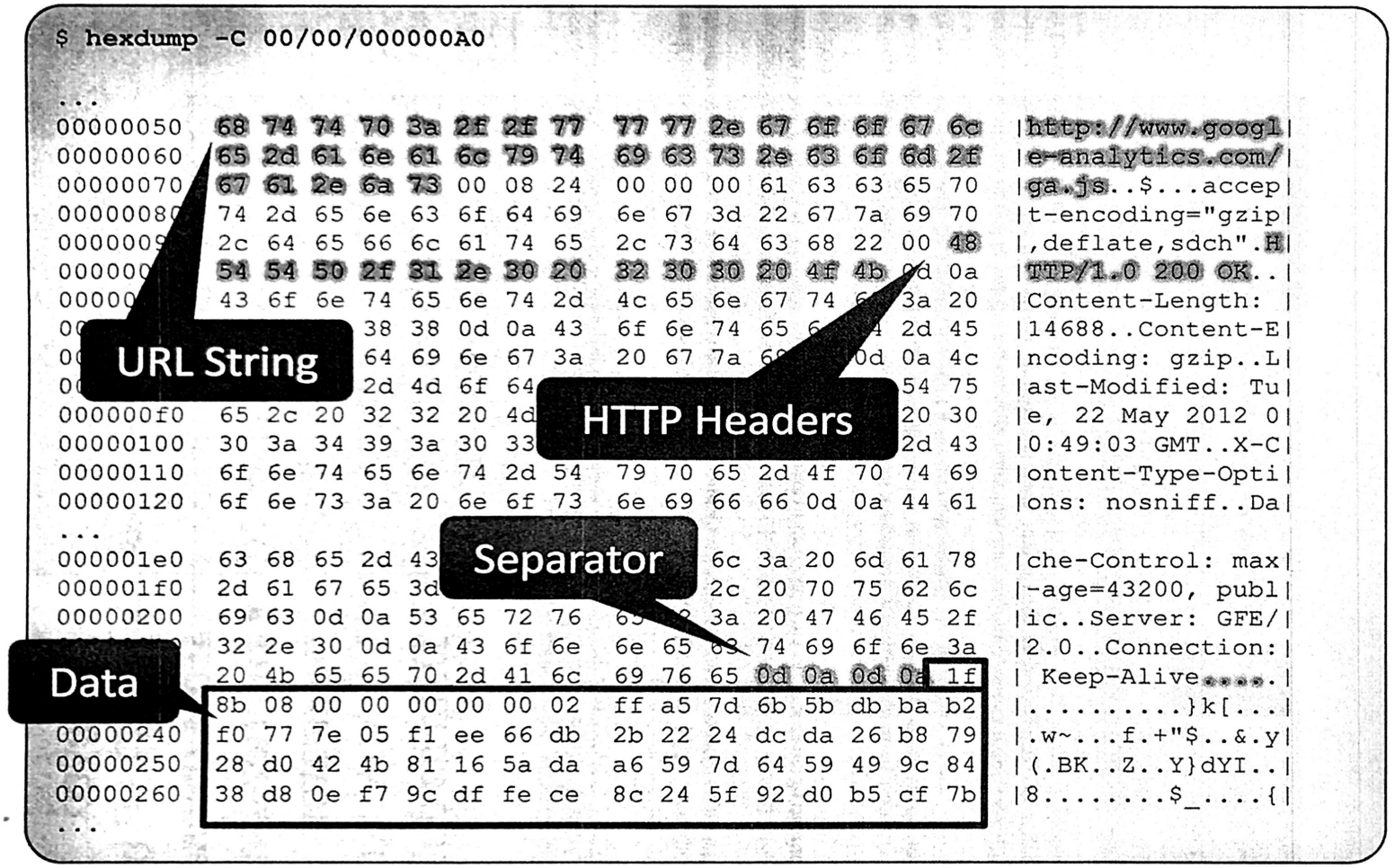
Although each cache file is generally recognized as a binary file, there are enough printable ASCII components that it is easy enough to visually parse. On a small enough scale, manual carving is a very reasonable approach.

The first recognizable field is the Cache ID number. This is an MD5 hash value derived from several components of the request. It is NOT, however, the MD5 hash of the contents of the data! This value appears in the records from the `store.log` file as well.

The URL string is easily visible, as it is prefixed with a 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, 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 http://www.google-analytics.com *
00/00/0000009F
00/00/000000A0
```


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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Squid Cache Objects: Quick Look (2)

```
$ strings 00/00/000000A0
http://www.google-analytics.com/ga.js
accept-encoding="gzip,deflate,sdch"
HTTP/1.0 200 OK
Content-Length: 14688
Content-Encoding: gzip
Last-Modified: Tue, 22 May 2012 00:49:03 GMT
Date: Sat, 09 Jun 2012 11:09:25 GMT
Expires: Sat, 09 Jun 2012 23:09:25 GMT
Content-Type: text/javascript
Age: 19339
Cache-Control: max-age=43200, publicServer: GFE/2.0
Connection: Keep-Alive
Y}dYI&vg2r$B_~
...
```


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, and even "magic file value" of the image file contained in the cache object are displayed.

The URL will be easily recognizable in a cache object file, so you could combine `strings` with `grep` and `head -1` to retrieve a list of URLs cached:

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


© 2017 Lewes Technology Consulting, LLC, 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?

```
$ less humantime_access.log

2012-06-09 16:47:18 ... TCP_MISS/200 6148 GET http://pastebin.com/ ...

2012-06-09 16:47:49 ... TCP_MISS/302 505 POST http://pastebin.com/post.php ...

2012-06-09 16:47:49 ... TCP_MISS/200 25403 GET http://pastebin.com/aCSe2JVD ...
```


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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Walkthrough: Find Objects by URL

- Identify the cache object that corresponds to the known paste URL
  - "railgrep" for objects containing the unique paste ID

```
$ cd /cases/for572/demo-01/Demo-01_source_evidence/var/spool/squid/
$ grep -rail aCSeZJVD *
var/spool/squid/00/03/000003A6
var/spool/squid/00/03/000003A5
```

- Review contents of each file
  - `var/spool/squid/00/03/000003A5` is an internal Squid file to optimize caching—we want `000003A6`

To start, let's look for any cache object that contains the believed URL of the paste the employee created. Using `grep`, this results in two candidate files: `000003A6` and `000003A5`. It is outside the scope of our material, but the "`00003A5`" file is an internally generated object, which Squid uses to optimize caching objects that allow different encoding mechanisms such as compression.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Walkthrough: Determine Object Content

```
$ strings var/spool/squid/00/03/000003A6
http://pastebin.com/aCSeZJVD
accept-encoding="gzip,deflate,sdch"
HTTP/1.1 200 OK
Server: nginx
Date: Sat, 09 Jun 2012 16:47:49 GMT
Content-Type: text/html; charset=utf-8
Connection: close
Set-Cookie: cookie_key=3; expires=Sat, 07-Jul-2012 16:47:49 GMT; path=/;
  domain=.pastebin.com
Set-Cookie: realuser=1; expires=Sun, 10-Jun-2012 16:47:49 GMT; path=/
Content-Encoding: gzip
Vary: Accept-Encoding
```

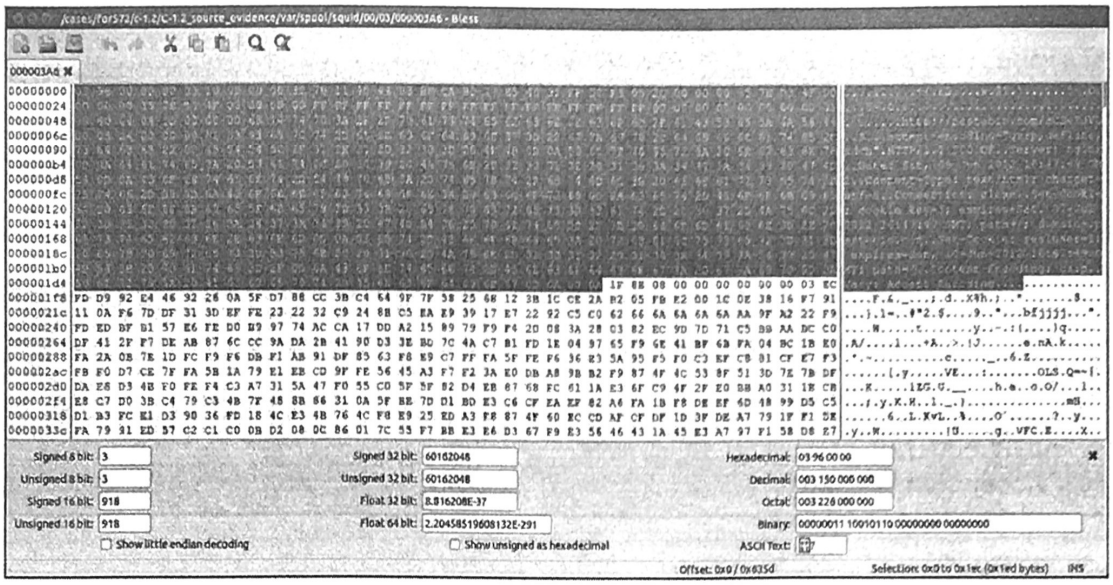

Looking at the actual cached object file, we see headers that are consistent with HTTP-delivered web pages— Content-Type, Cookies, etc. The rest of the file doesn't contain anything that resembles a web page, though.

The key here is the "Content-Encoding" header. In this case, the pastebin.com web server compressed the content before returning it. This is a common and effective way for servers to be more efficient in their bandwidth utilization.

We can still carve the content, though.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Walkthrough: Remove Headers



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`.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Walkthrough: Content Type

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

```
$ file /tmp/proxy_cache_extraction
/tmp/proxy_cache_extraction: gzip compressed data, from Unix
```

  - 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 just straight gzip-compressed data—as expected based on the HTTP headers in the proxy file itself.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Walkthrough: Examine Content (1)

```
$ zcat /tmp/proxy_cache_extraction | less
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
        <head>
                <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
                <title>data as requested - Pastebin.com</title>
                <link rel="shortcut icon" href="/favicon.ico" />
<link href="http://pastebin.com/cache/css/text.css" rel="stylesheet" type="text/css" />
   <link href="/i/main.css" rel="stylesheet" type="text/css" />

                <script type="text/javascript" src="/js/jquery-1.4.4.js"></script>
                <script type="text/javascript" src="/js/main.js"></script>
                <meta property="fb:page_id" content="150549571626327" />
                <meta property="og:title" content="data as requested - Pastebin.com" />
            <meta property="og:url" content="http://pastebin.com/aCSeZJVD" />
            <meta property="og:image" content="http://pastebin.com/i/fb.jpg" />
            <meta property="og:site_name" content="Pastebin" />
...
```


To see the content, just uncompress the contents and view. We can clearly see this is consistent with the content of a web page, which is equivalent to what the user would have seen in their browser using the "View Source" feature. By comparing the results to a test load of a pastebin.com web page, or by manually paging through the contents, we can find the part of the page that should contain the data on the pastebin page that the user sent out of the environment.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Walkthrough: Examine Content (2)

```
<form class="paste_form" id="myform" method="post" action="/post.php">
            <div class="textarea_border">
                  <textarea id="paste_code" class="paste_code" name="paste_code"
onkeydown="return catchTab(this,event)">vjvumzUiCNSzq1vyrhyNgQgJ9KC3Sjtj3juIbK+P2mdFDh
n459+nx9FEE2ybWzE91QZkPMvVvbI37ZOoBDEsP3I/Q5Iz0Pp58jlsiY8sfer3TE23r4DdNTgHj87VRLUNG6r4t
HLIbUXxSJ/WsRI91VmlK795DihIBLrQ38/5ouEv46bgvCWuogVRndVufIMeGZFPX4t/Vsj1096tguylKtcWBXyE
K2O754dchaI1XF0MdyRBBnj8HHf1bBmP77Hhak61JFNsSVRonNRqhcae//1SjGbCcZ6/JR7EtHwaj+KeLHdKVr9
cYSMVM5GBrOpdZGkJ4dKg1PfpKnWIKG2ZfHwtU+i/Yvc7liwjg6Aav1SQUp0ciLrxV0uxgtT4fk89ts9qKIKIwD
4csQ1wxxVpqs18nH6sSTGeYXFVDm3RBDjQ5bpGKqdd3aoC1H7ylIBJ1XE2FPi1b0r7aEDSz+JiWwkXV6nq+lBLy
jm3IVCrZJcAN8ZZH5BGWqeoatN5Ev1o7HL+jpsUEpd8Fj1LbOq4Gm8kkWqxle+Yr3jKktZ4qerolZeCVM8khtah
NO/VtKbQyQZqUw89wOtd7n717Ly7HSwNCLtnDnVrQrE4Fao4uoQwtYHAWt7Om6JZ1t54uUIl2zdb1zjdw0fbmW1
FD8Xk8pvEGaS2IwZHJHIpipl46Sb7Te+OcAVFhMokyMZmltLJzKLQ8FfosrOYGylMG8rwDziH91GT3nCSc02PVt
0LMCq3yeFYyuwx4BRKAj53a8CwOJ4UM3/WY7Q08jh+Bh9avFfW2qaSS/yZ8392oj7P/KLE/9y/vzcLjVufKbKvX
jYLSmmLxC5D0pyhiNfr1wU3pqTGY7cMNP24DfbUol06PgqqG27kIR9W5EUuPQSSrgMhUw8pSwffXJk9dzCkEaSi
74bJDqwOS0cTZxbxB0M/MUN7wCRSQfXGG4gX2s5qwoygPuRcY4M+L1PeadEHbUVPDBULvE/MfLItBPfcHDlgu/v
cCZaAbW7jUKU6w/xs2oj7iRmRmME0v/6zvi/VDbcTe/CsGbVh2Qwn220F1ZQbzayVSPUJq3qKfe3FbcGENX3f1N
CMjh8Seuu+yKyh8abKqFBWwVb8XGGQwAh1OsQpVIVkdRhdx4QI8r1C8IWqDlX5FzGvus5XPn1YJRTYGVUHiezrq
7X4wUEb0zt0u2T2JEiF/S74arJA64ptFsqd8fVcUyAwSUR3cpkx/3Ww34Ov3B4KzCIjQZIztgKjoNueE0/EDkyV
...
```


Paging down the contents of the paste, we find some HTML form elements that appear to contain base64-encoded data. We can confirm the behavior of the pastebin.com site by creating another paste, and verifying that it does, in fact re-display the contents of that paste in a similar HTML form element.

Based on proxy cache extraction, we have successfully recovered what may be the data our rogue employee leaked from his workstation before his attempted flight to Madripoor. However, even after base64-decoding the contents of the paste, there is no identifiable file type or data. We'll need to hand this off to our reverse engineering team to determine what the paste may have included.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Proxy Extraction Homework: What was Pasted?

- From the evidence provided, determine what appears to have been uploaded to Pastebin
- Does it appear actionable?
- What would you do next?

We've come quite a long way in this walkthrough. However, the question about what exactly had been uploaded via the Pastebin URL is still unknown. Based on what we have accomplished so far, consider this optional homework as a follow-on to our walkthrough.

- What was uploaded? Provide hash checksum and characterize its content.
- Do you feel this is an actionable finding? Why or why not?
- What would you do next?

Bring your results to class tomorrow for discussion.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

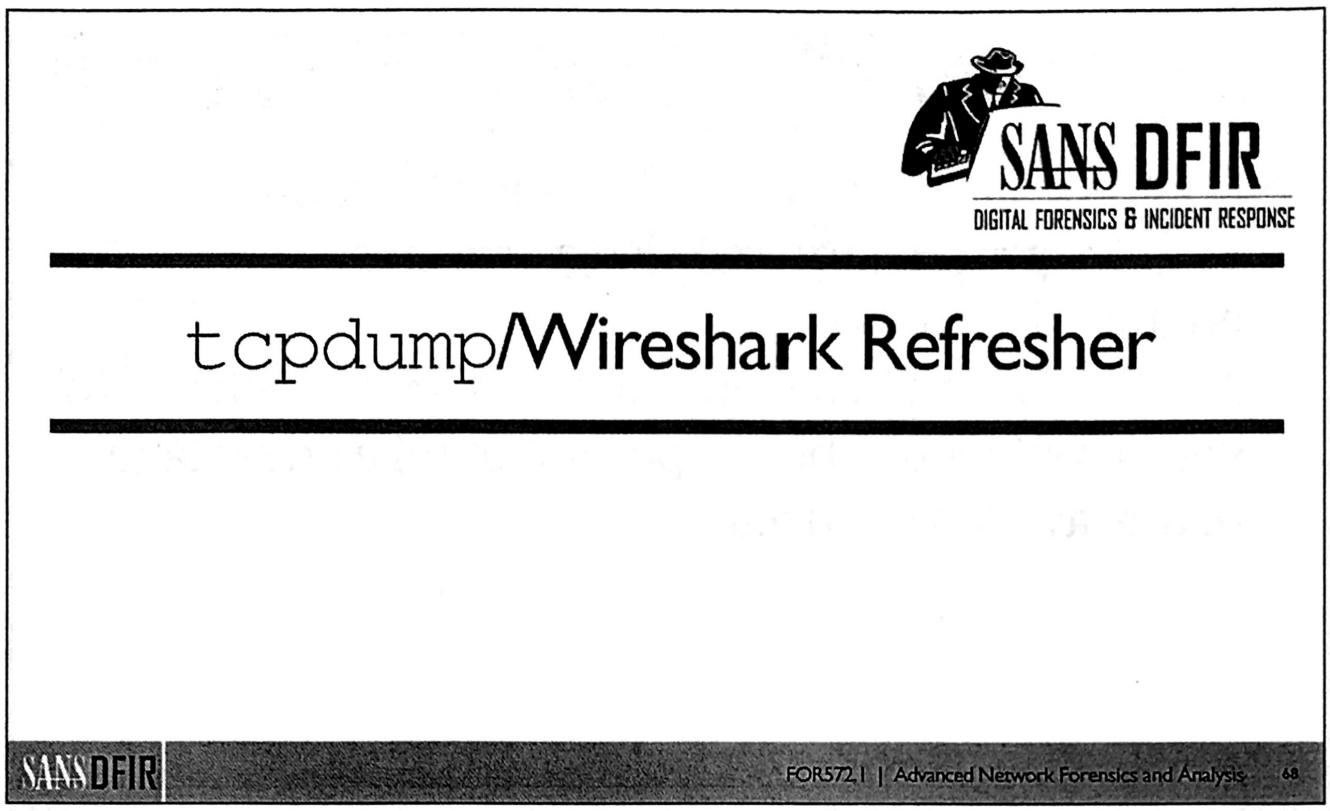
Proxy Extraction: "It's Complicated"

- Commercial products can hinder extraction
- Require proprietary knowledge, reverse engineering, or both
- May be easier to use alternate sources of evidence to get to root truths, but important to take advantage of any available format

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Back to Basics ... (1)

- `tcpdump`: Most widely used capture tool
  - Open-source, cross-platform
  - Command line-based
  - Based on libpcap
    - Uses Berkeley Packet Filter (BPF) syntax
    - Display details in terminal or save to pcap file on disk
    - Read from network or existing pcap file
    - Proprietary tools often read from/export to pcap files

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 very 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`

© 2017 Lewes Technology Consulting, LLC, 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 very 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 is pre-loaded 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 one's self 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 that 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).

One very useful feature is that Wireshark is also distributed with "tshark", which is a console application that performs all the same features as the GUI variant. This is 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`

© 2017 Lewes Technology Consulting, LLC, 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!

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

- 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)

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

PCAP File Format (2)

pcap packet/frame header

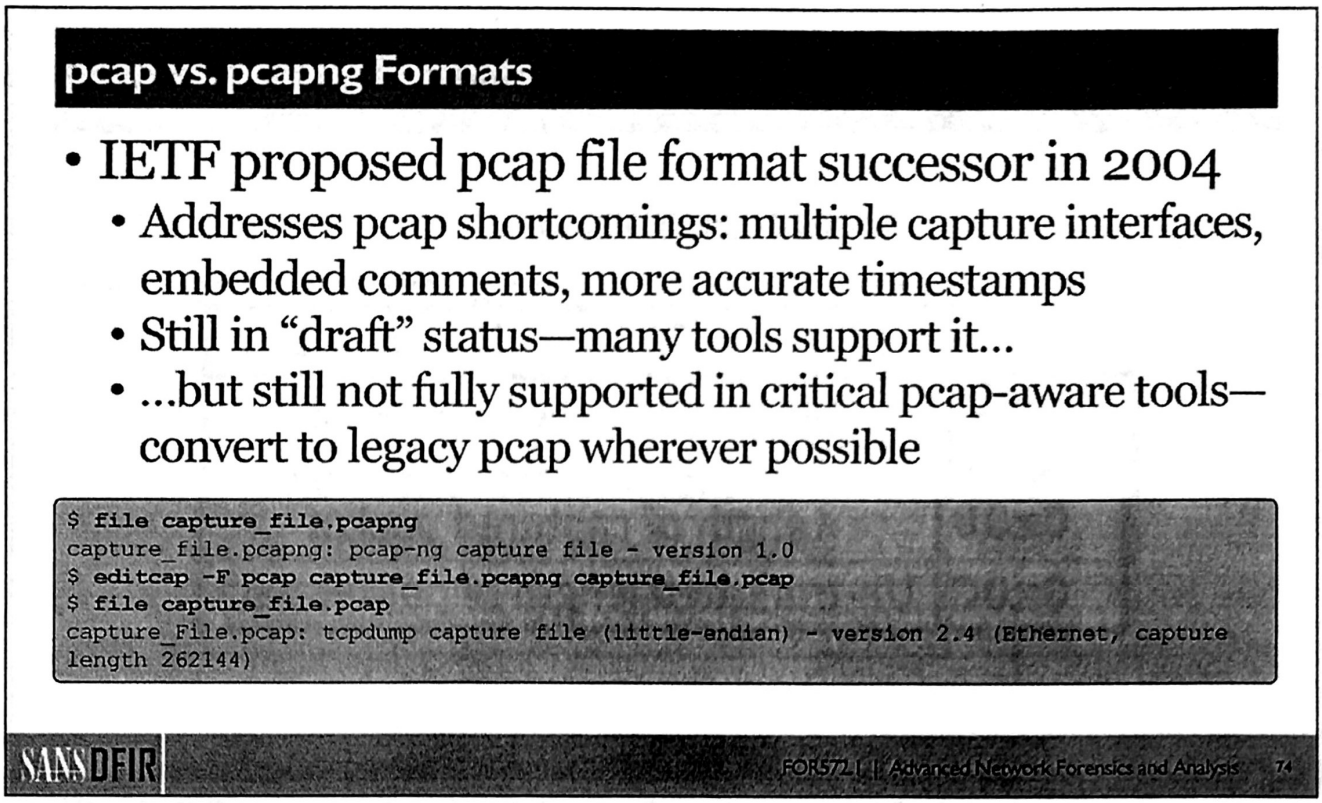
	0	1	2	3
0x00	Time stamp, seconds value			
0x04	Time stamp, microseconds value			
0x08	Length of captured packet			
0x0C	Un-truncated length of packet data			

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 very useful data for an analyst as well:

- 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 then 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)

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



Because the pcap file format has its shortcomings, the Internet Engineering Task Force devised a successor format named "pcapng", or "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 include[2]:

- 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. There are a number of resources that 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/gie91`
[3] `http://for572.com/fxa32`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Fundamental tcpdump Usage

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

```
$ sudo tcpdump -n -s 0 -i eth0 -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 eth0, 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 eth0`: Capture from "`eth0`" 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 re-acquainted 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 eth0 -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 eth0`	Capture from device "eth0". 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.

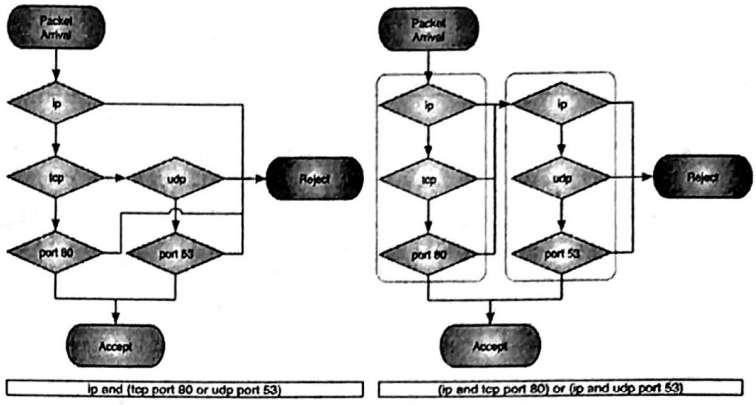
We'll examine the BPF in more detail next, but it is always best practice to enclose the BPF in single or double quotes to prevent the shell from interpreting spaces, parenthesis and other reserved characters.

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

BPF Primitives

- Several primitives and logical combination
  - Common: `ip`, `tcp`, `udp`, `icmp`, `host`, `ether`, `net`, `port`
  - Qualifiers: `src`, `dst`
  - Logic: `and`, `or`, `not`, `()`
  - Less common:
    - `vlan`, `portrange`, `gateway`, byte offsets: `ip[9:1] == 0x06`



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 are 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 "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, 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, 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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

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 parenthesis 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 is not.)

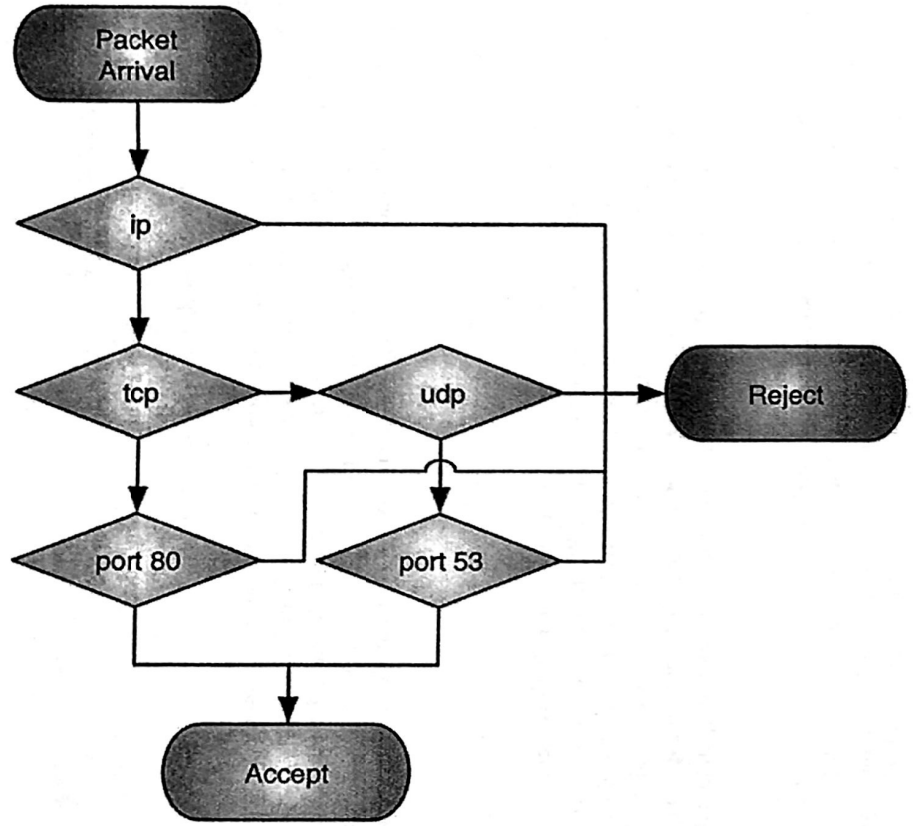
Again, these are just a few of the more common and helpful BPF primitives—there are many others that you may find helpful in this class's exercises and your daily jobs. A few of the more clever of the lot are below, but you should absolutely take a look at the man pages to become better acquainted with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

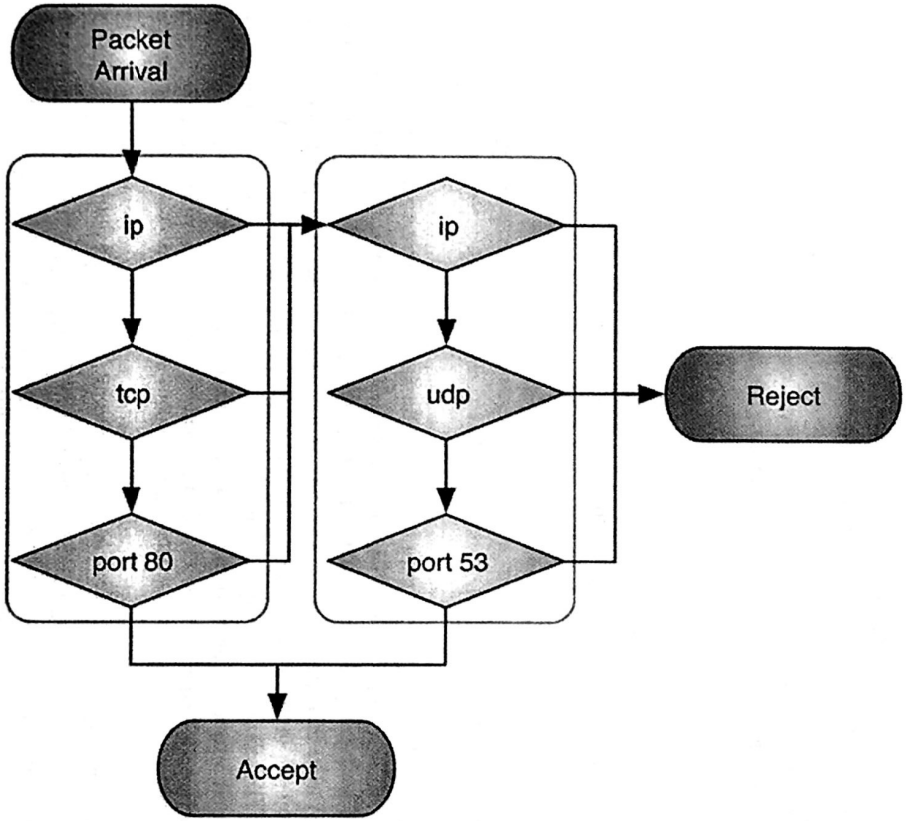
References:

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

pcap-filter(7)

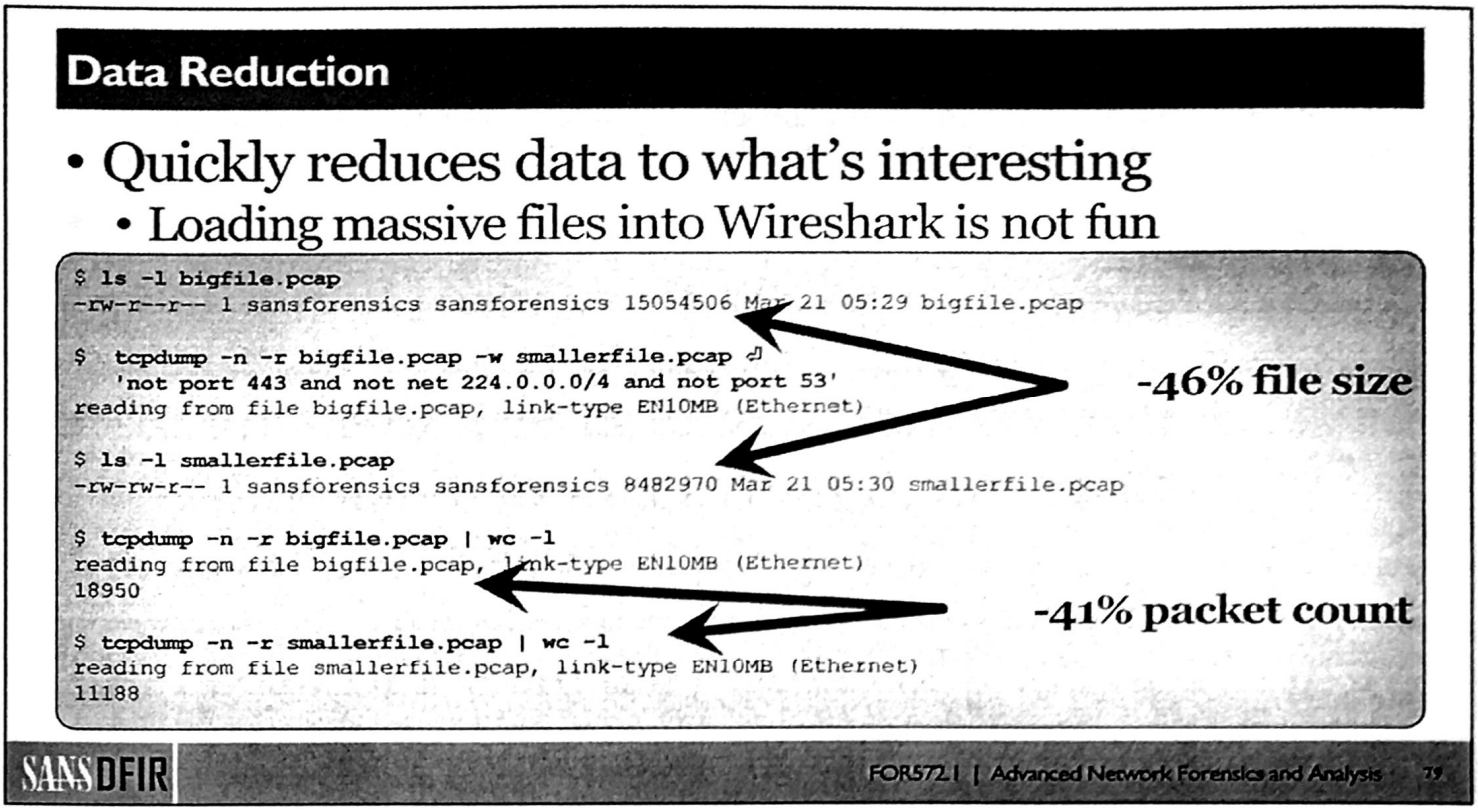


ip and (tcp port 80 or udp port 53)



(ip and tcp port 80) or (ip and udp port 53)

© 2017 Lewes Technology Consulting, LLC, 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 load the data 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.

```
$ ls -l bigfile.pcap
-rw-r--r-- 1 sansforensics sansforensics 15054506 Mar 21 05:29 bigfile.pcap

$  tcpdump -n -r bigfile.pcap -w smallerfile.pcap ⏎
   'not port 443 and not net 224.0.0.0/4 and not port 53'
reading from file bigfile.pcap, link-type EN10MB (Ethernet)

$ ls -l smallerfile.pcap
-rw-rw-r-- 1 sansforensics sansforensics 8482970 Mar 21 05:30
smallerfile.pcap

$ tcpdump -n -r bigfile.pcap | wc -l
reading from file bigfile.pcap, link-type EN10MB (Ethernet)
18950

$ tcpdump -n -r smallerfile.pcap | wc -l
reading from file smallerfile.pcap, link-type EN10MB (Ethernet)
11188
```

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

In this case, we are removing some traffic that is 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%.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

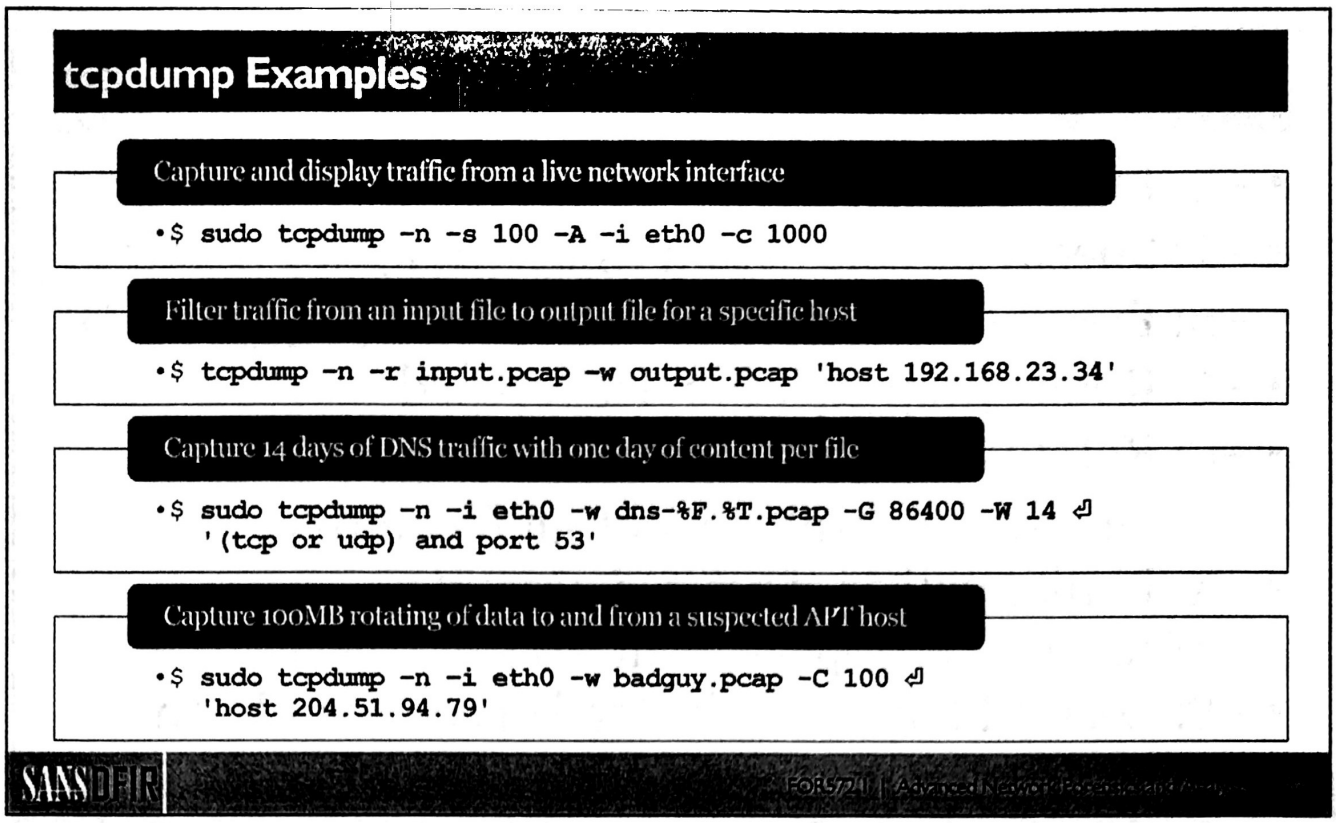
Useful tcpdump Options

- `-r`: Read from pcap file / `-w`: Write to pcap file
- `-i`: Specify interface from which to capture traffic
- `-n`: Prevent DNS resolution for all IPs (CRITICAL!)
- `-C`: Rotate pcap after file size reached
- `-G`: Rotate pcap after number of seconds (needs timestamp format in output filename)
- `-W`: Limit number of rotated pcap files
  - With "`-C`": Ring capture; With "`-G`": Max count
- `-F`: Load BPF from file instead of command line

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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

The first command monitors the network interface identified as `eth0` 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 and application data to the console.
`-i eth0`	Use `eth0` as the network interface for capture.
`-c 1000`	Capture the first 1000 packets from the network interface.

The second option 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

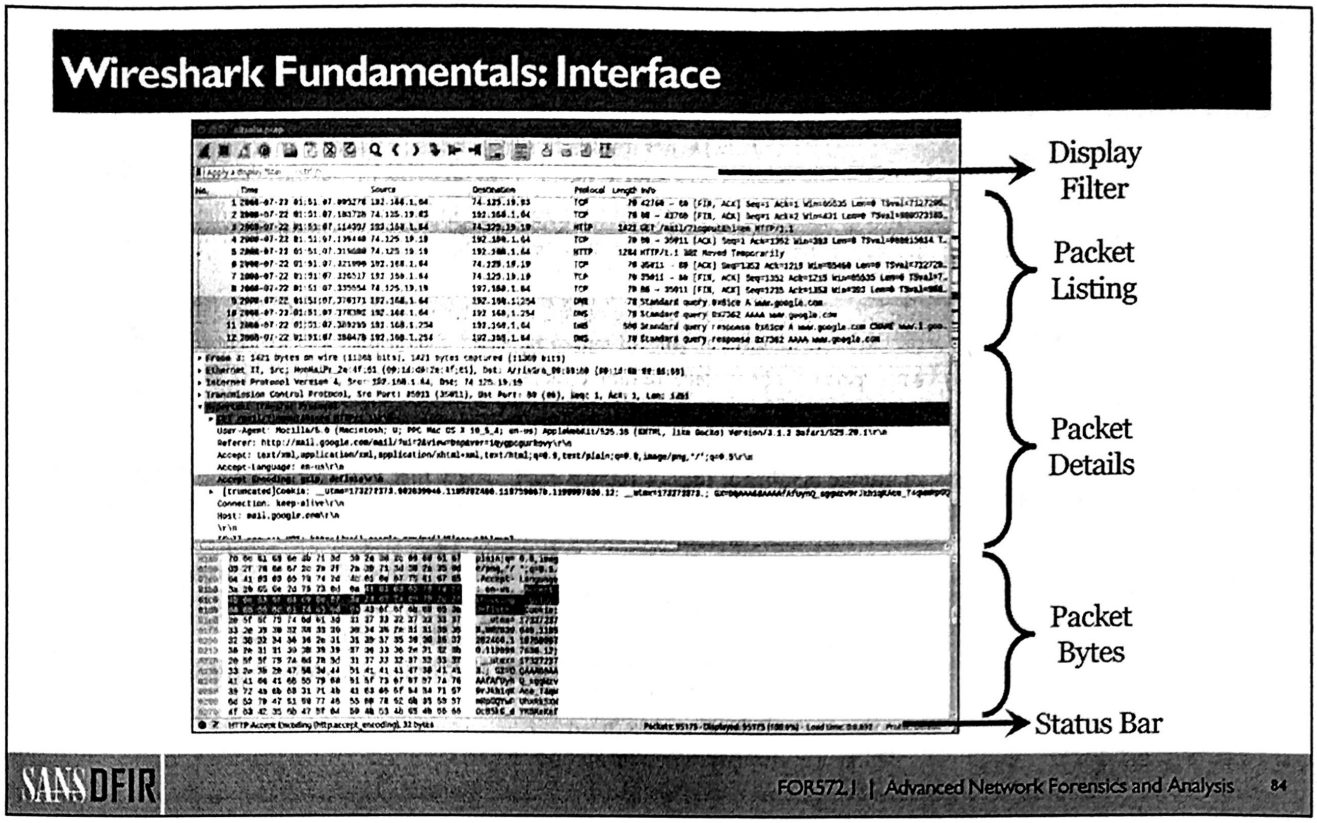
The third option captures 14 days of DNS traffic and then exits. Each 24-hour period has a separate file (24 hours * 60 minutes * 60 seconds = 86400 seconds). The BPF limits output to just TCP/UDP traffic on port 53.

`-n`	Do not resolve IP addresses to hostnames or ports to protocols.
`-i eth0`	Use eth0 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 option 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 eth0`	Use eth0 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.

© 2017 Lewes Technology Consulting, LLC, 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 as well. The "Info" column can also be helpful in that many of Wireshark's protocol decoders will provide a very 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 each and every field within the packet—from the IP and TCP or UDP headers, all the way to the layer seven 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.

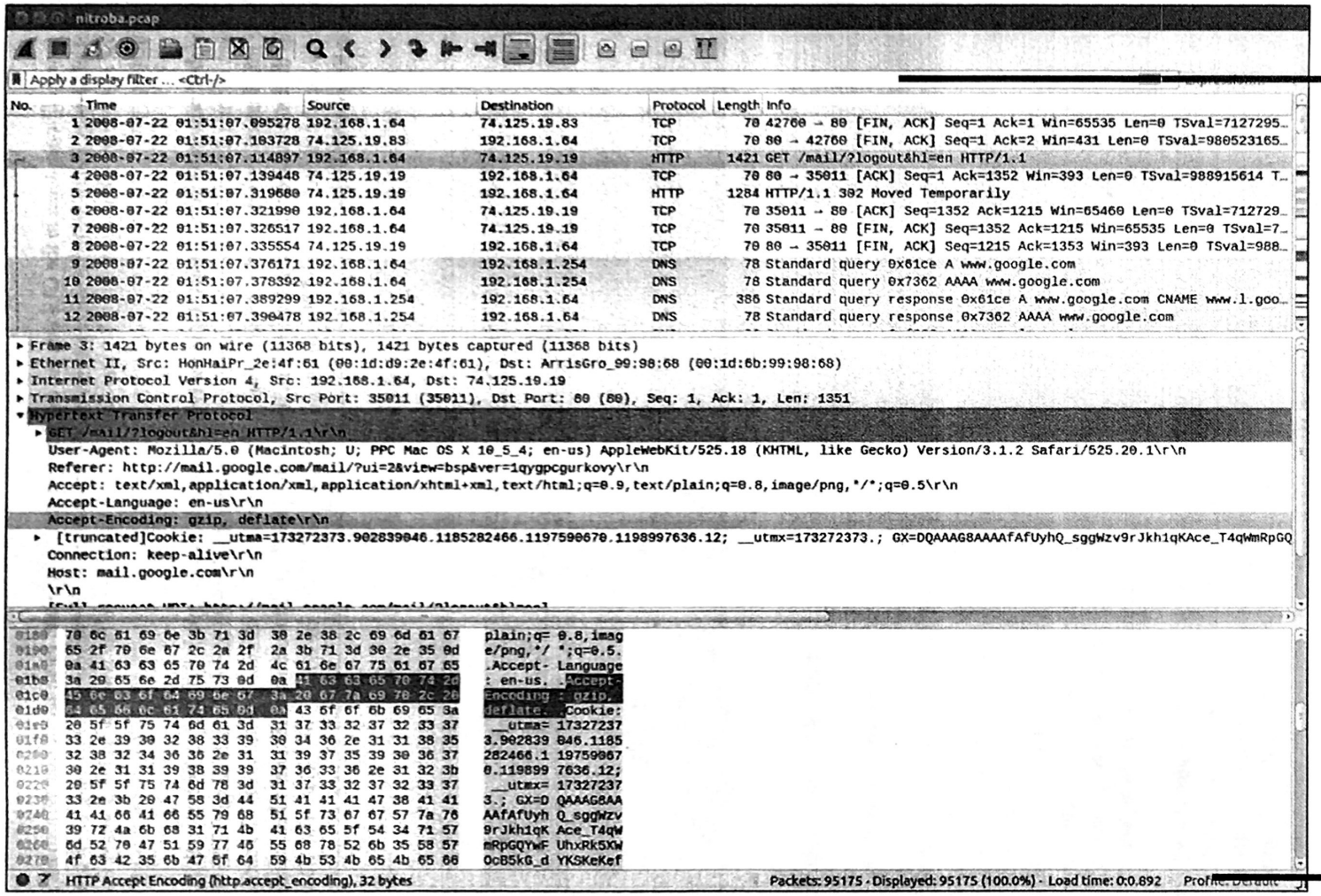
Finally, the Packet Bytes pane contains a hex dump-style listing of all bytes contained for that packet as stored in the pcap file or as they transited the network. Although at first this may seem unnecessarily complicated when compared to the Packet Details pane, having the raw data often helps an analyst 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!

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

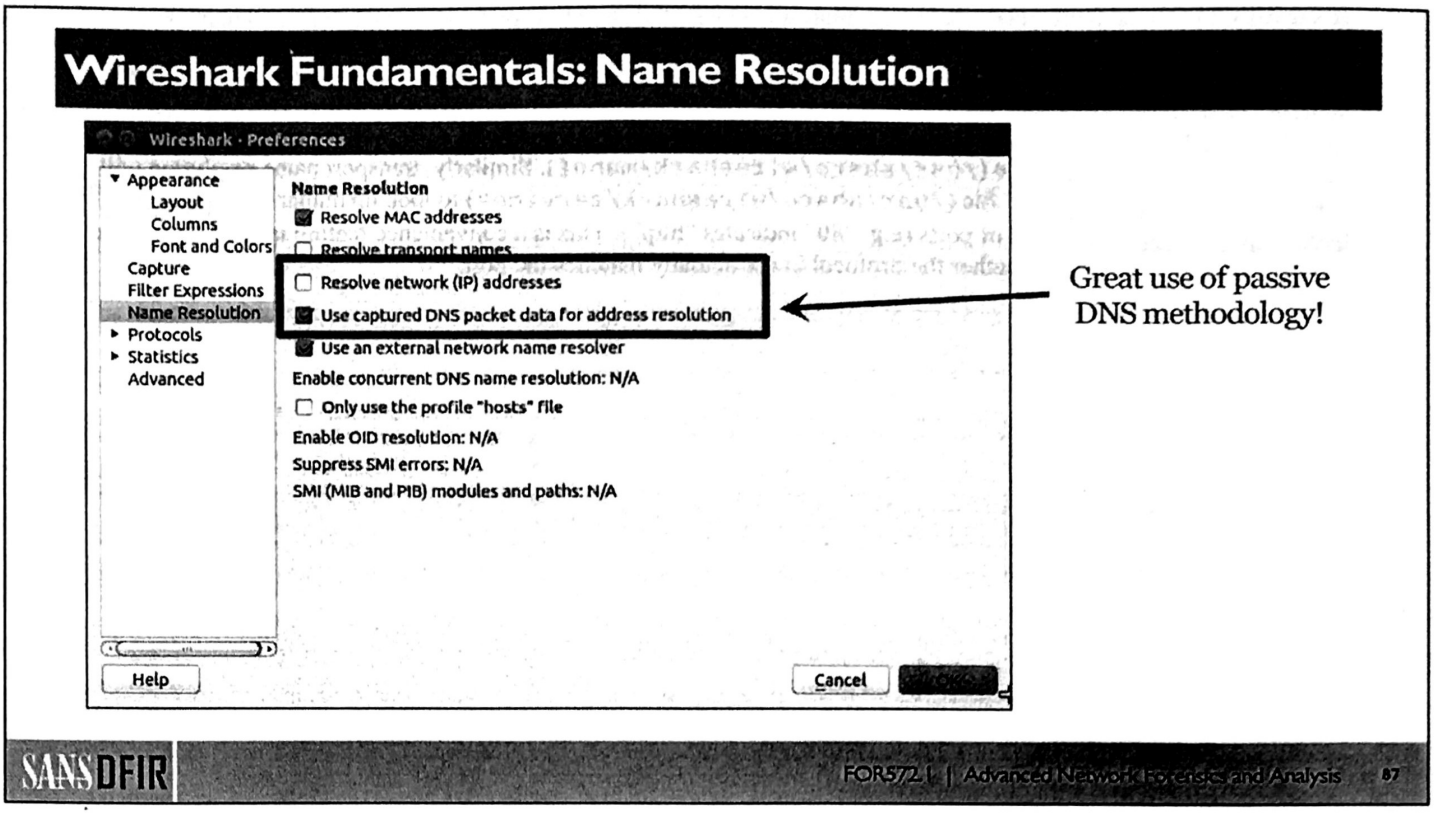
There are two additional items in the GUI that are extremely important to note. First, recall that Wireshark provides a robust set of display filters for us to use in narrowing down the number of packets we're looking at. We will take a closer look at these in a few pages, but the Filter Toolbar, seen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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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.

The first one deals primarily with OPSEC, or 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!

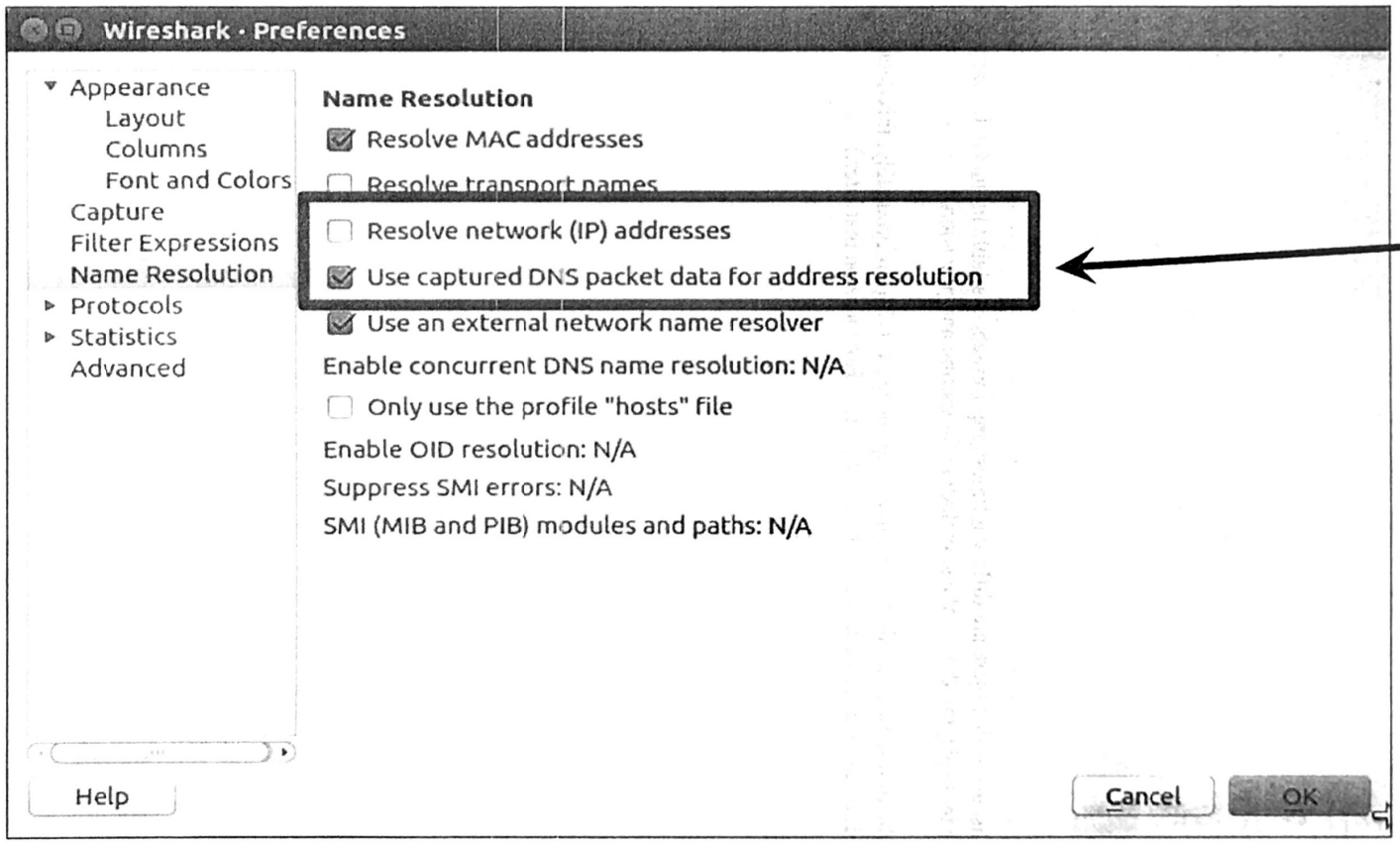
Although this may seem far-fetched or even a little bit paranoid, it has occurred in the wild, and adversaries are known to advance their capabilities quite quickly. Because best practice is to minimize unnecessary network traffic while reasonably maximizing OPSEC posture, this option should almost never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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

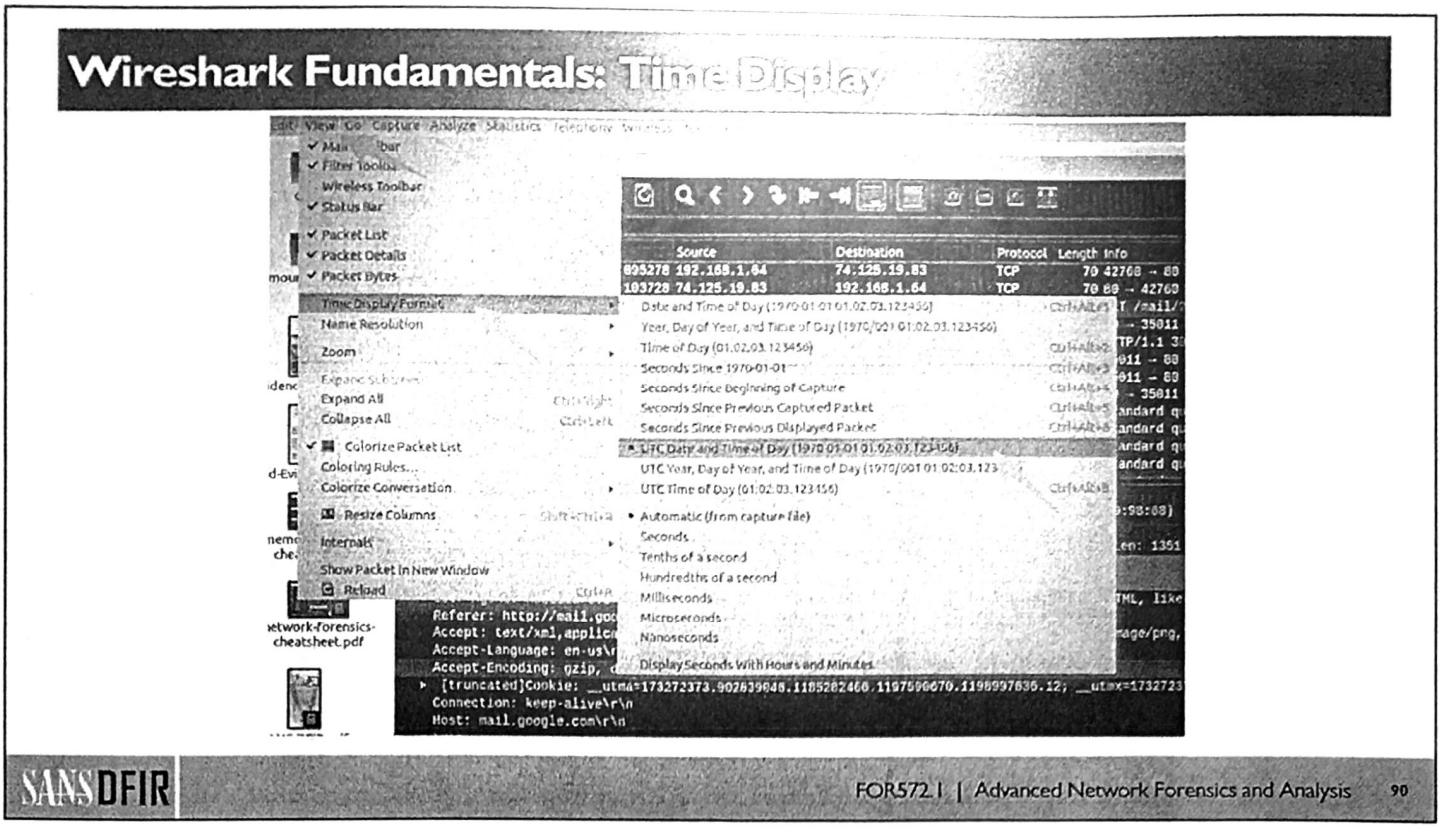
It's usually a personal preference whether an analyst enables MAC and transport name resolution, and neither causes Wireshark to generate its own network traffic or any OPSEC concerns. These both look up their data from files on the local system. MAC name resolution means that Wireshark will look up 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



Great use of passive DNS methodology!

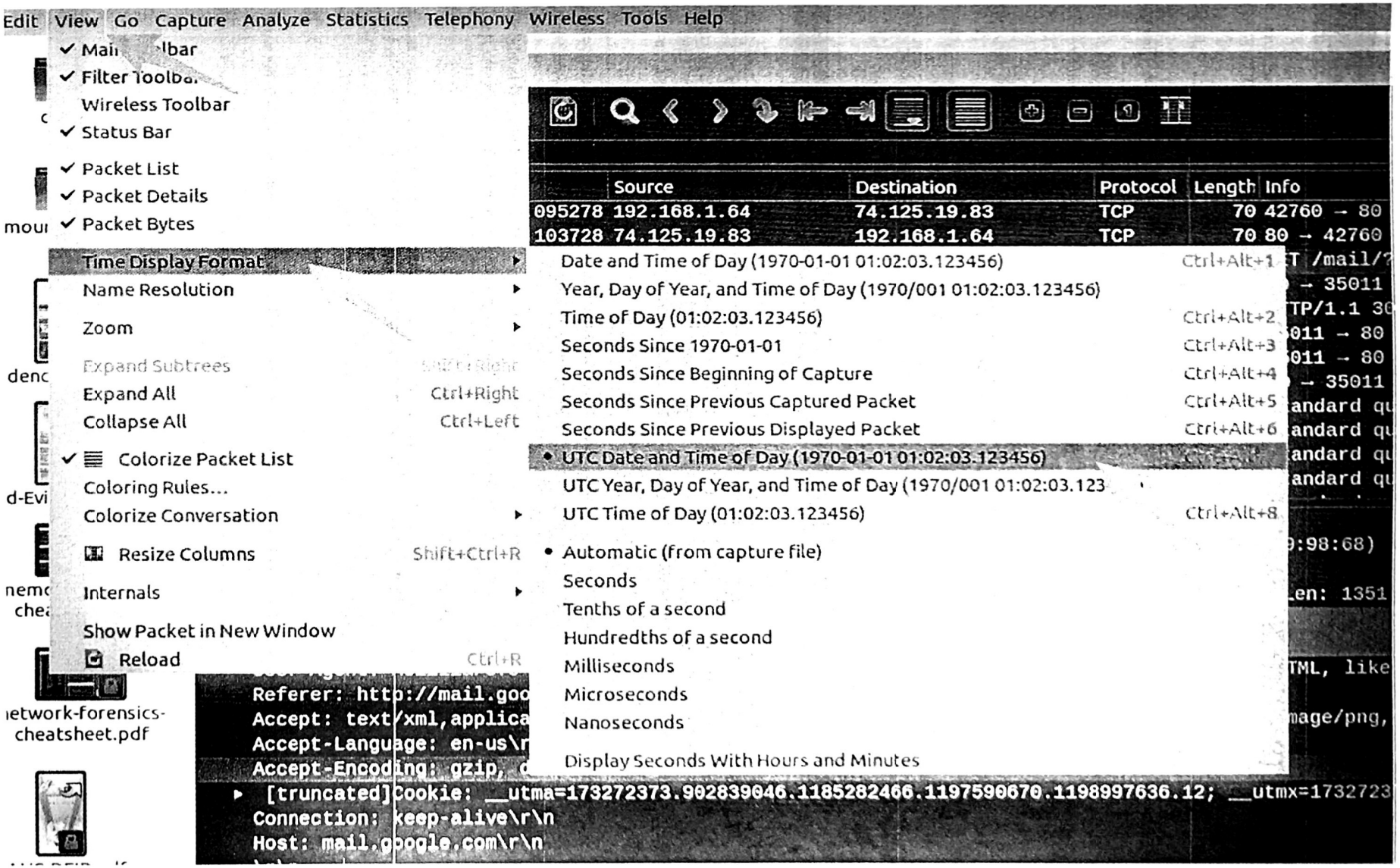
© 2017 Lewes Technology Consulting, LLC, Mat Oldham



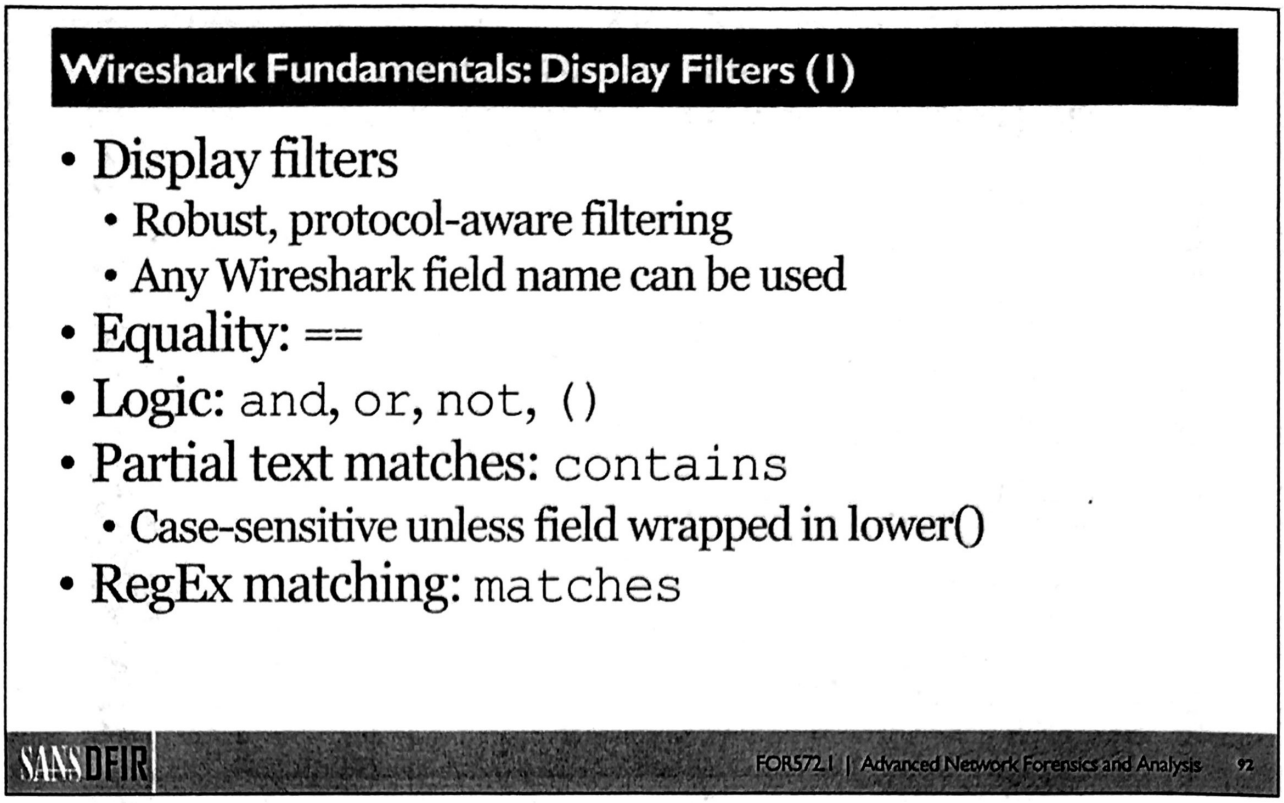
Previously, we discussed that there are different timestamp formats available in the GUI. You can see here the options you have in displaying these values, including formatting and precision.

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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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 tradeoff 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"'
```


© 2017 Lewes Technology Consulting, LLC, Mat Oldham

The power of a well-crafted display filter becomes even more evident when looking for DNS replies containing more than 5 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"'
```

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:

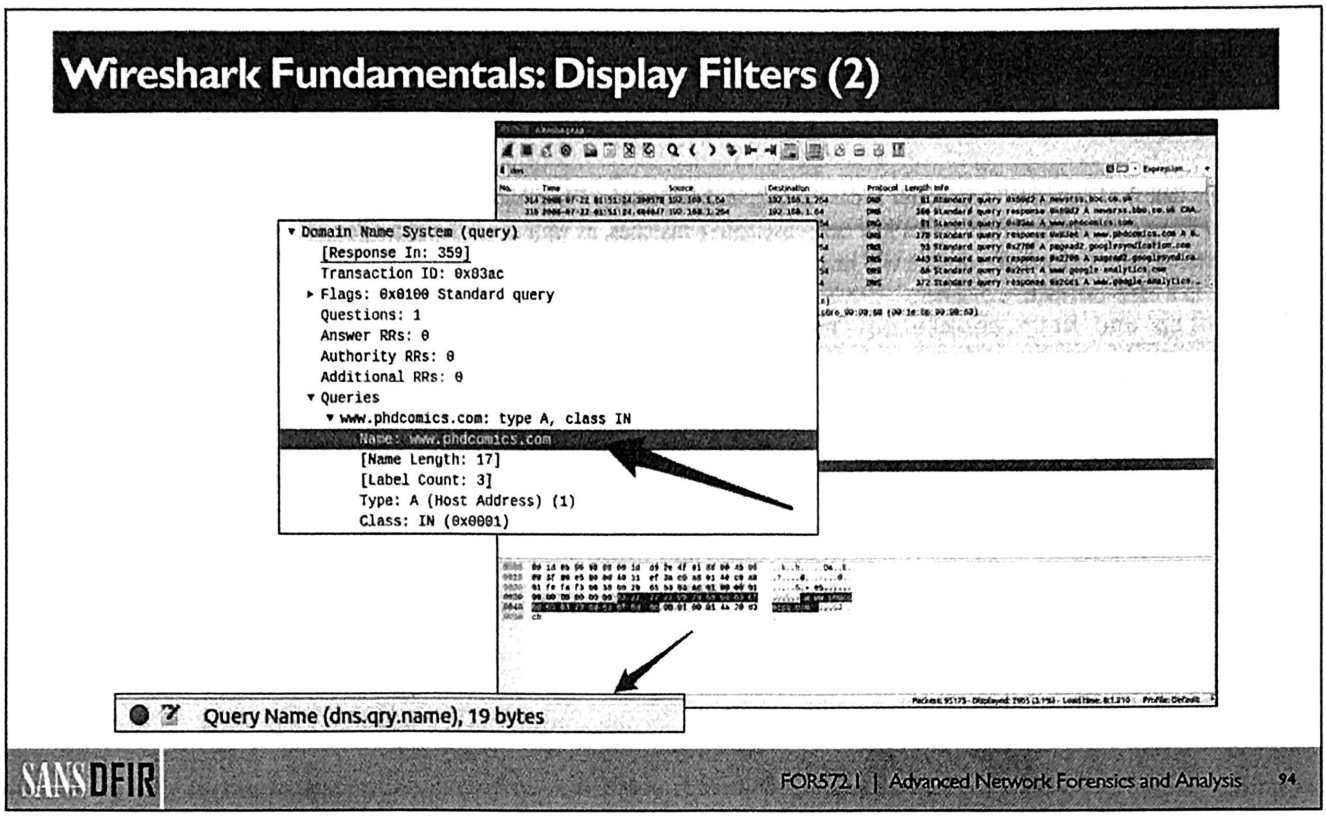
```
'http and lower(http.cookie) contains "username"'
```

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

References:

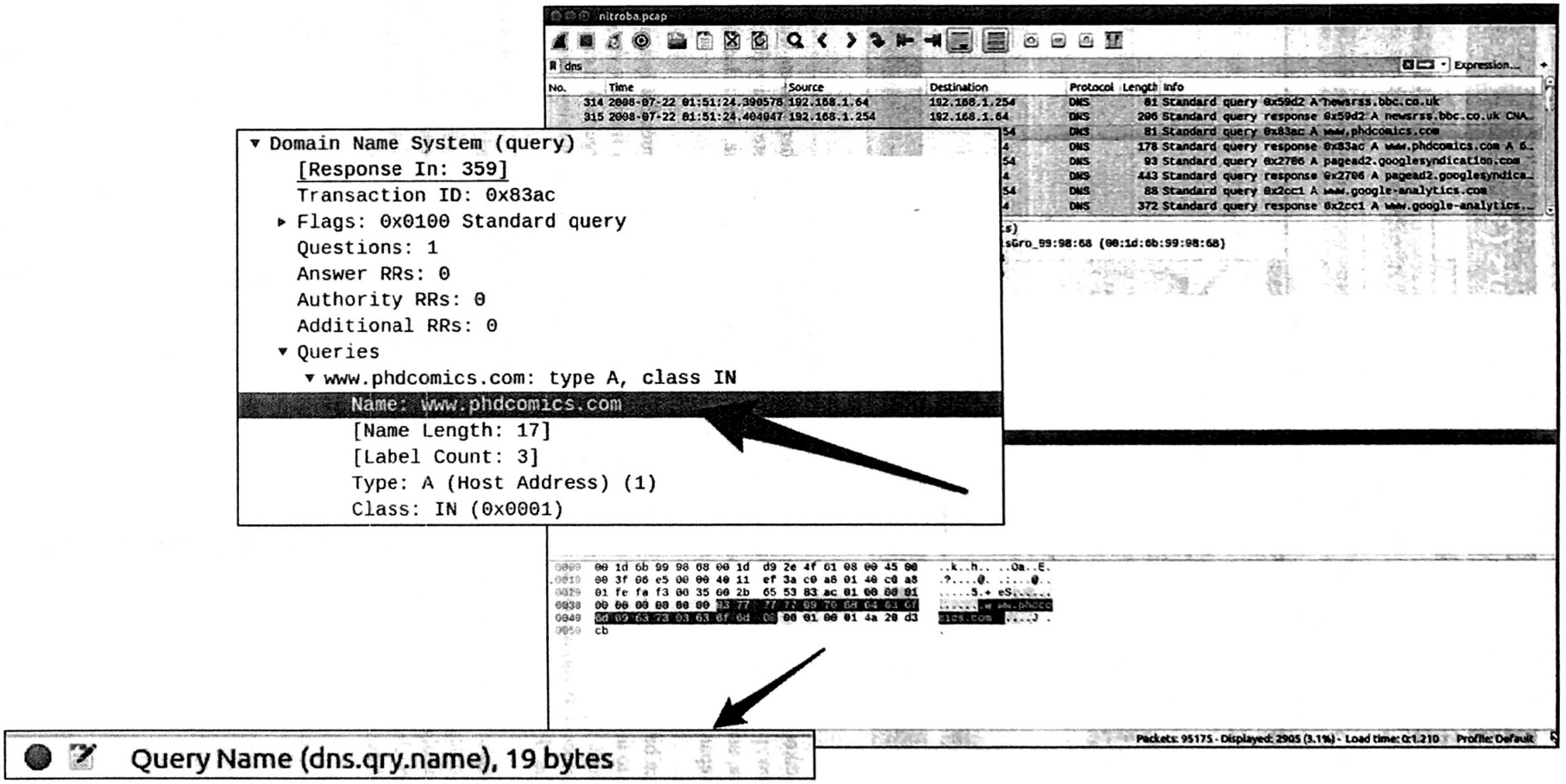
http://for572.com/-52to
http://for572.com/g2ayh
wireshark-filter(4)

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

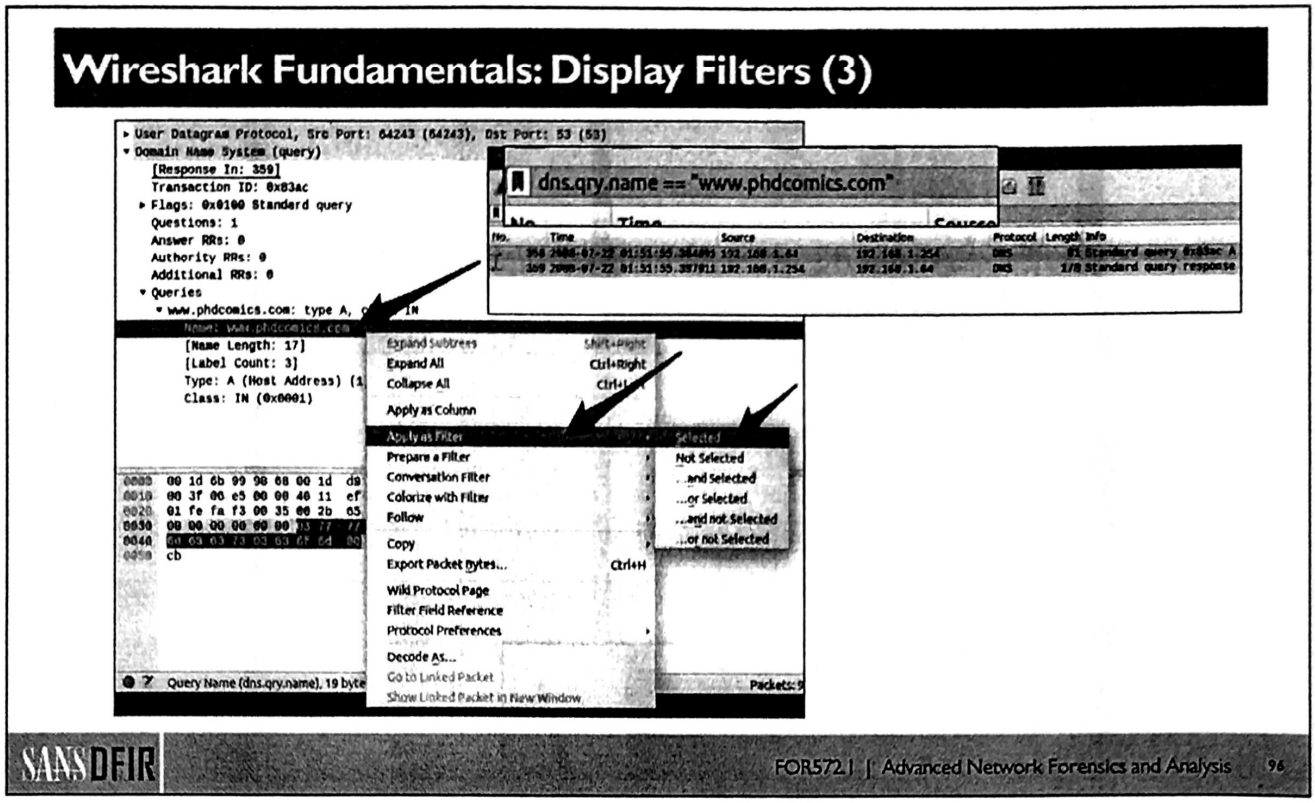


Take the example above. The Packet Details pane shows a basic DNS query for the A record associated with the hostname "`www.phdcomics.com`". By looking at the left side of the status bar, we can see that the DNS decoder uses the name "`dns.qry.name`" to denote this field, and that in this case, the field 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, Mat Oldham

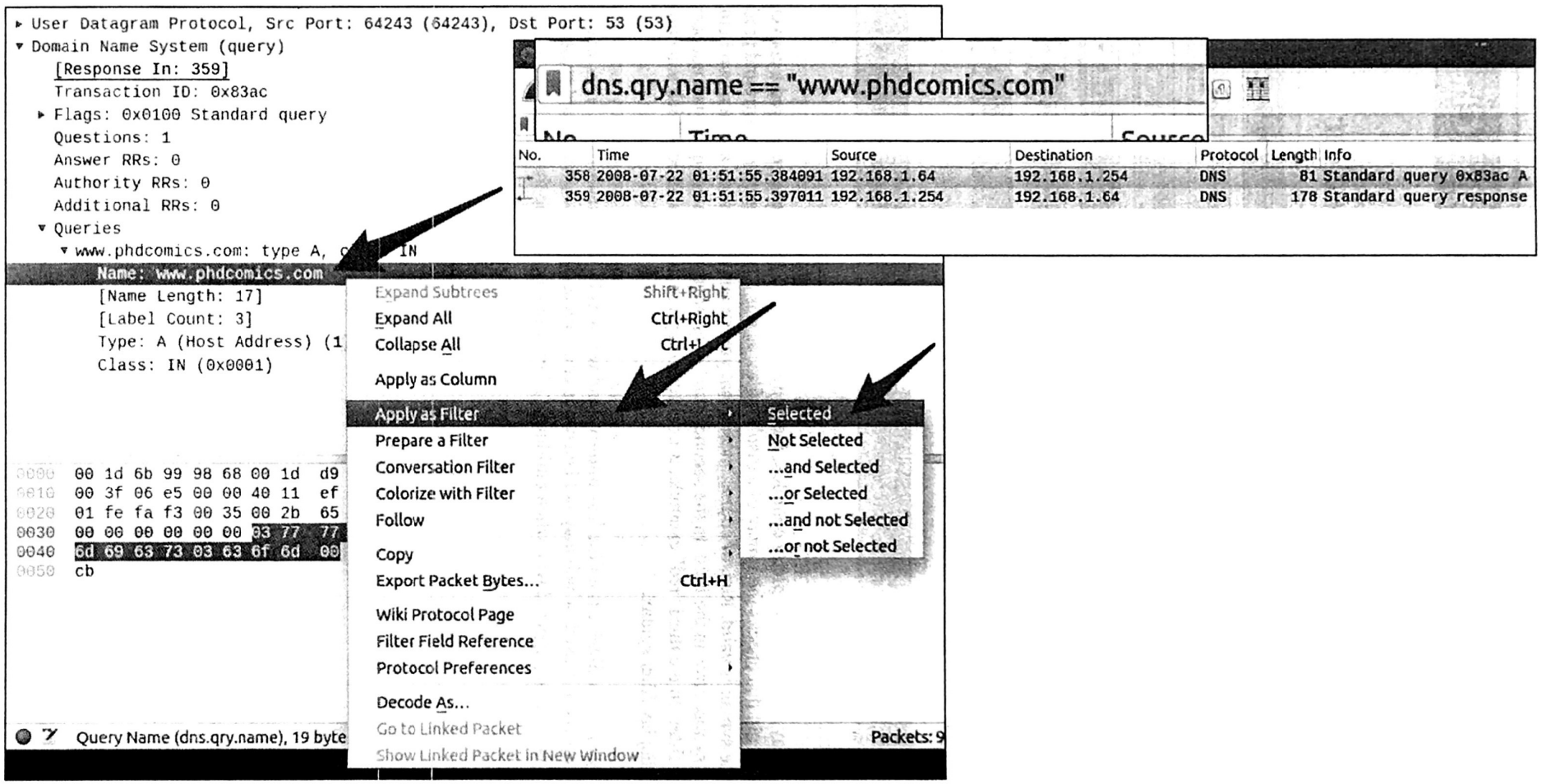


Here, we've taken that field name (`dns.qry.name`) and crafted a simple display filter: packets with an exact match of "`www.phdcomics.com`" in the field. Wireshark allows the analyst to create display filters manually, or, as shown here, use the GUI to build them interactively. Simply right-click a field name, then select the "Apply as Filter" or "Prepare as a Filter" submenus and one of the resulting popup 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 a very elegant solution, it makes walking through the raw bytes of a DNS packet a rather unpleasant experience. Wireshark decodes first, filters second—saving the analyst's precious brain cells in the process.

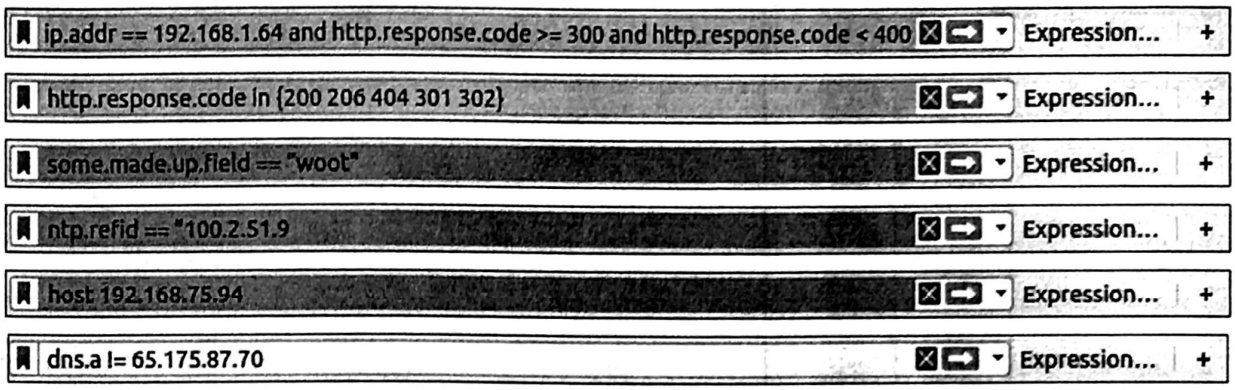
© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Wireshark Fundamentals: Display Filter Syntax Checking (1)

- Display filters
  - Real-time green/red/yellow syntax checker



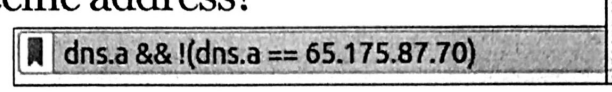
Because the display filters can get pretty complex, the Wireshark developers have employed a convenient "stoplight" visual aid that checks the syntax of a display filter as it is typed. A green background predictably indicates that the syntax is valid. Red, of course, means that you've either specified 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.

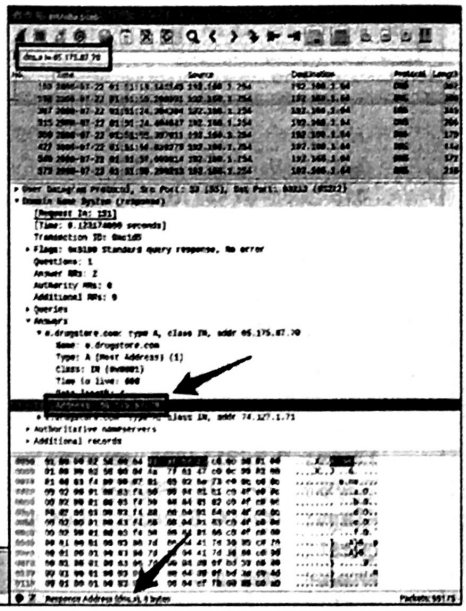
However, the "yellow" indicator is a little less intuitive and warrants further examination. This will trigger any time when the "!=" operator is used, which may not behave as common computer science would suggest. It specifically behaves in a somewhat unexpected way when building a filter with a field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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Wireshark Fundamentals: 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?

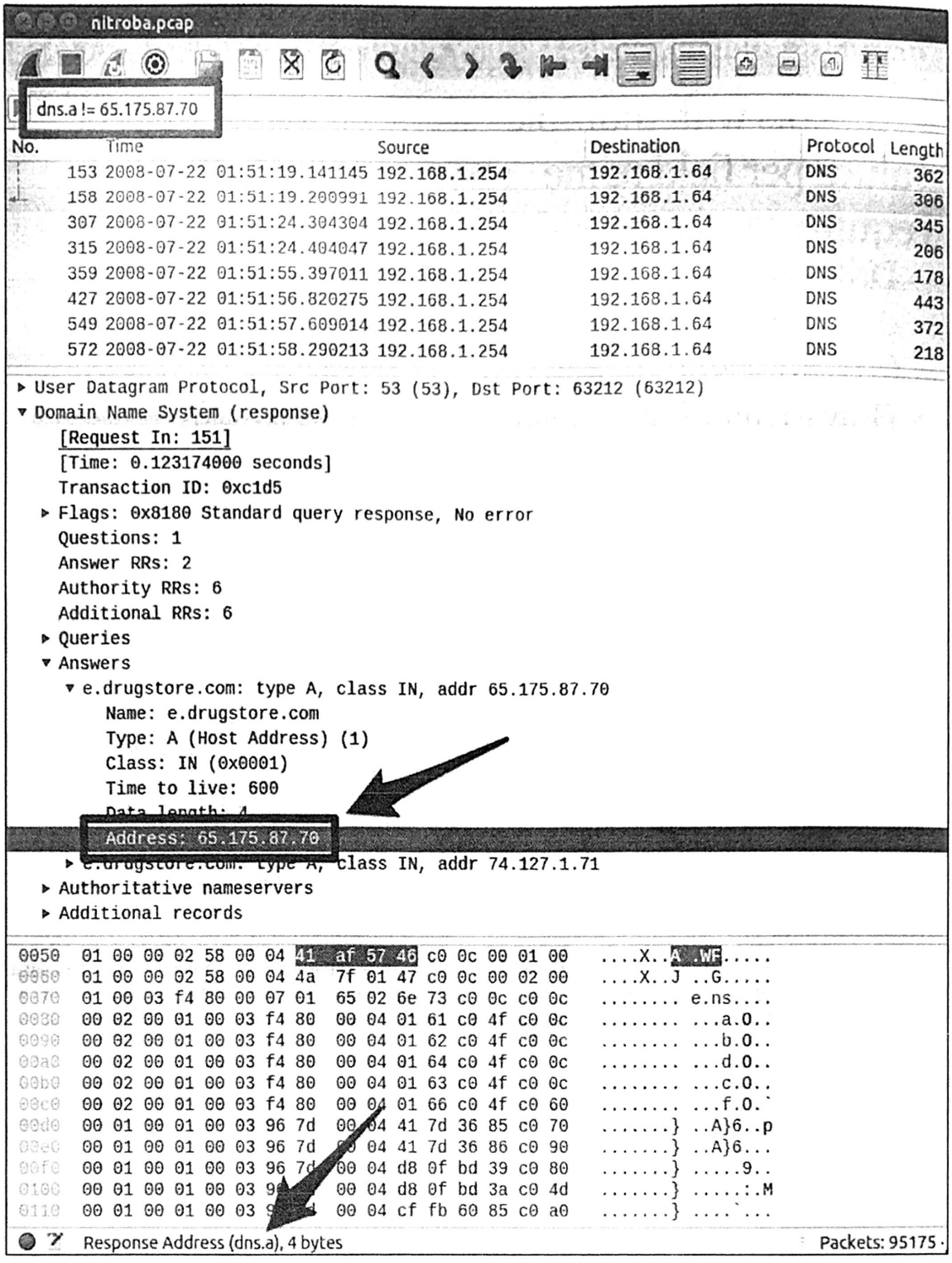




The screenshots above demonstrate the nuance of the "!=" operator and the resulting yellow syntax checker status. The larger screenshot on the right side shows a display filter of "`dns.a != 65.175.87.70`", but the contents of the DNS response data clearly show that IP address is present. This is because 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, and not a logic checker. Just because a display filter validates as "green", doesn't mean it will match the packets you want.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

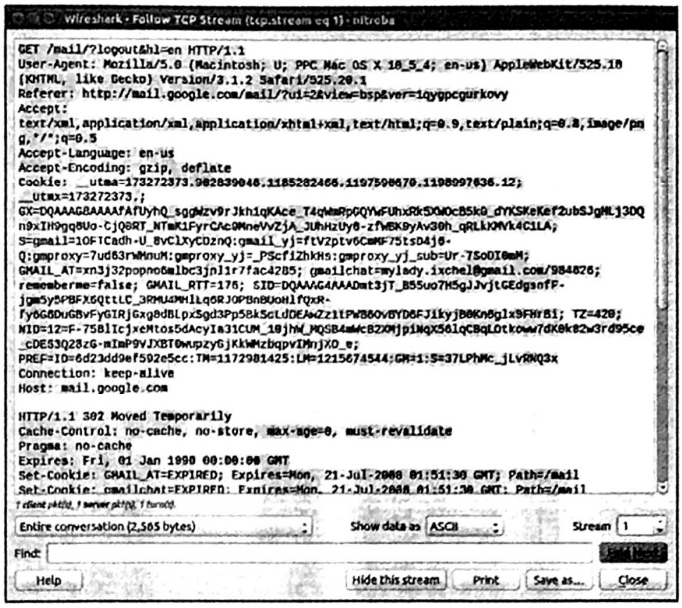



```
dns.a && !(dns.a == 65.175.87.70)
```


© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Wireshark Fundamentals: Follow TCP Stream

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



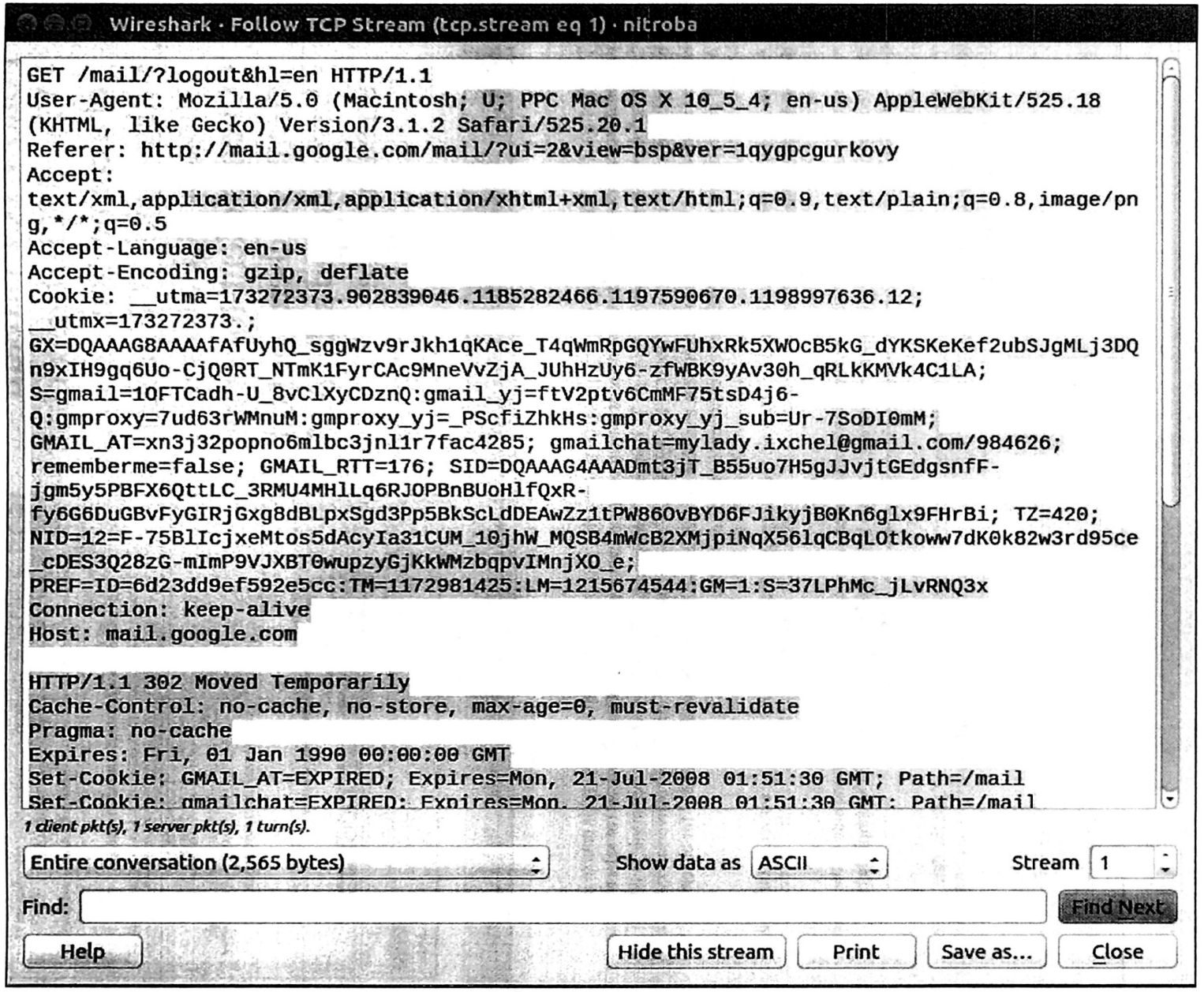
The "Follow TCP Stream" feature is very powerful and one that you will use extensively during this course and your real-world cases. The feature is activated by selecting a packet, then right-clicking the packet (or selecting the "Analyze" menu item) and selecting "Follow TCP Stream". Wireshark will pop up a window that displays one or both sides of a TCP conversation in ASCII or hexdump-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.

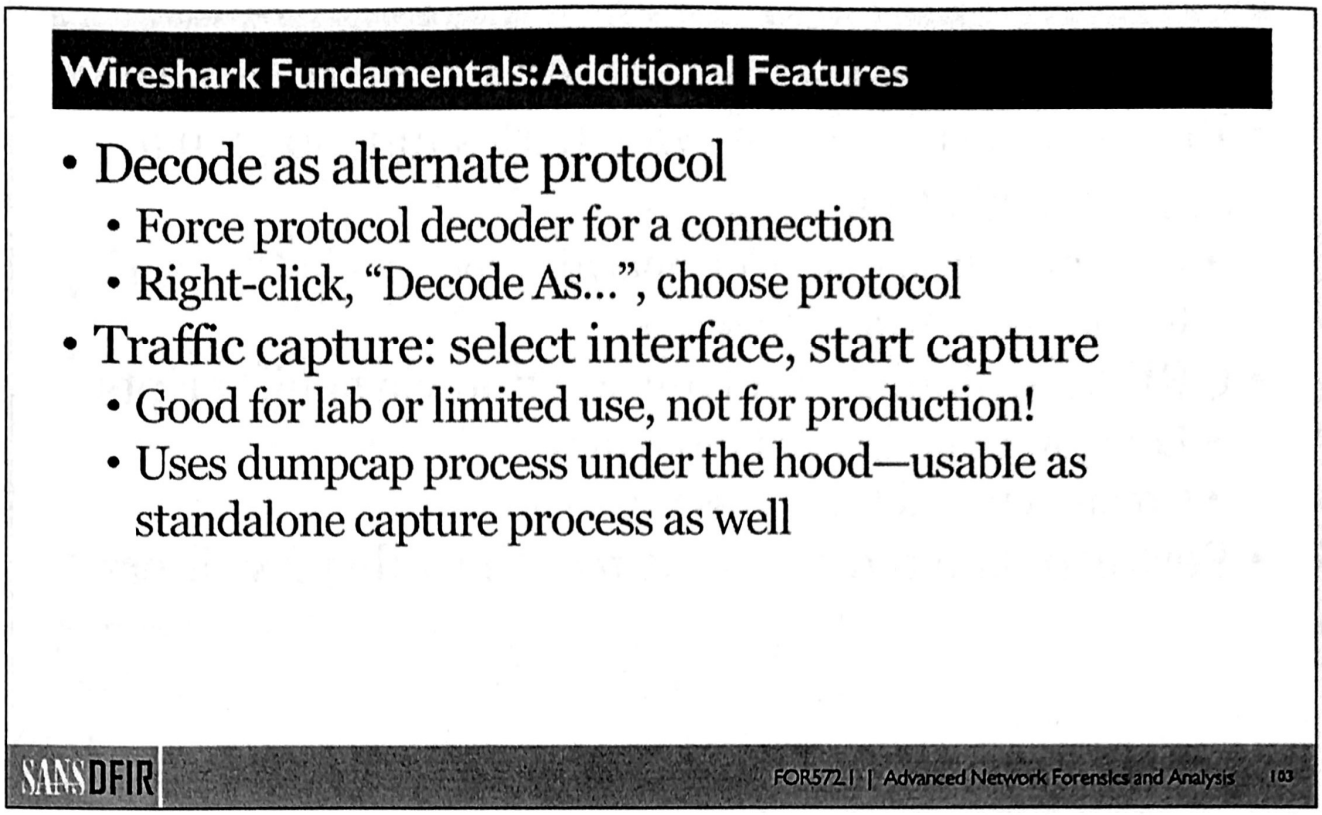
Note the dropdown labeled "Entire Conversation" in this screenshot can be used to select just one side or the other of the conversation. This is helpful when examining a stream that is less distinct than the HTTP example here in terms of block of client-to-server and server-to-client transmission.

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, Mat Oldham

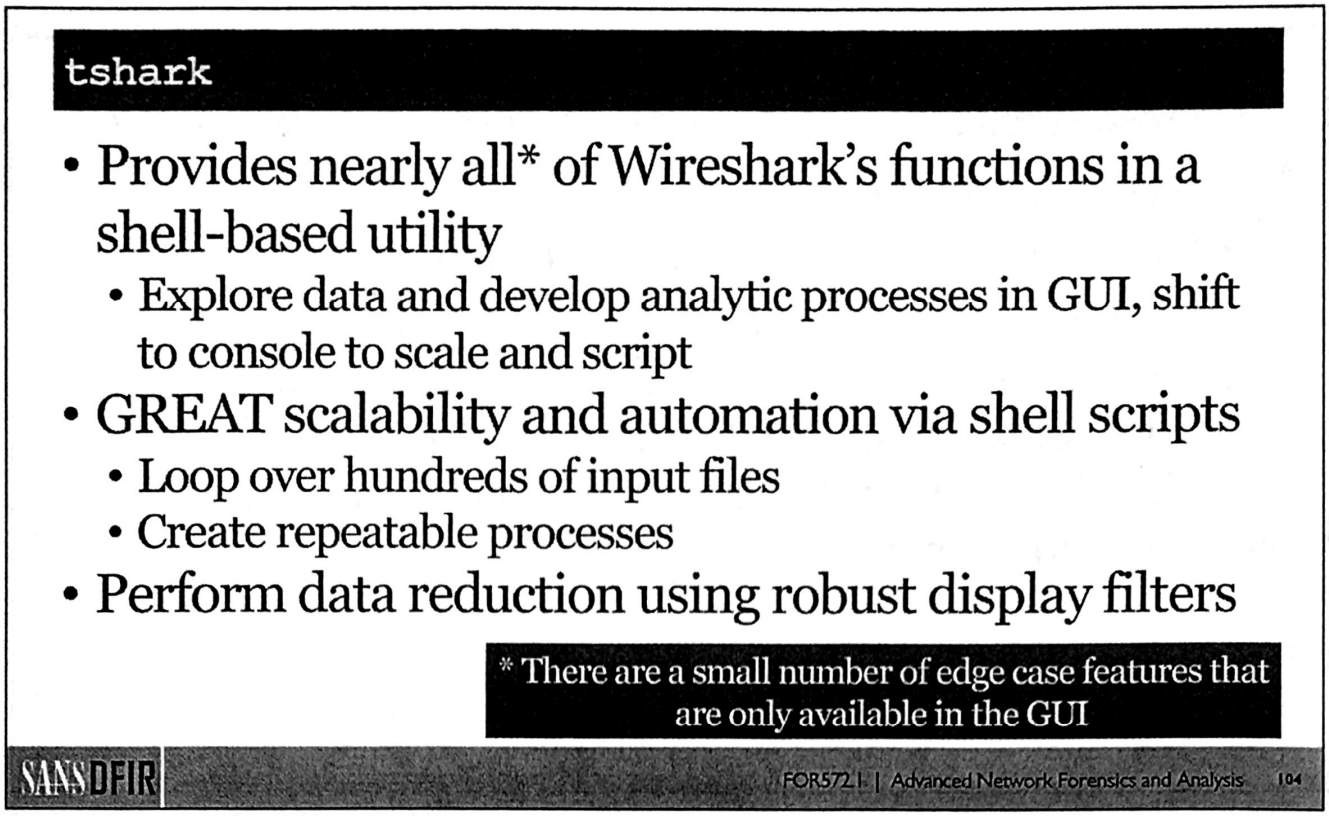


Wireshark is a very powerful and exceedingly useful tool. There are countless features that could benefit you during your investigations, and there is no way for us to fully cover them all here. The wiki-based documentation is quite extensive, and can be a great resource for even Wireshark veterans. 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`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

tshark Options

- `-r`: Read from pcap file / `-w`: Write to pcap file
- `-n`: Prevent DNS resolution for all IPs (CRITICAL!)
- `-Y`: Display filter to use (enclose in single-ticks)
- `-T`: Output mode: `text`, `fields`, `pdml`, others
- `-e`: With "`-T fields`", select fields to display (use multiple times)
- `-G`: Display glossary reports (Use "`-G ?`" for available options)

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 as we discussed with `tcpdump` and should also become an automatic flag!

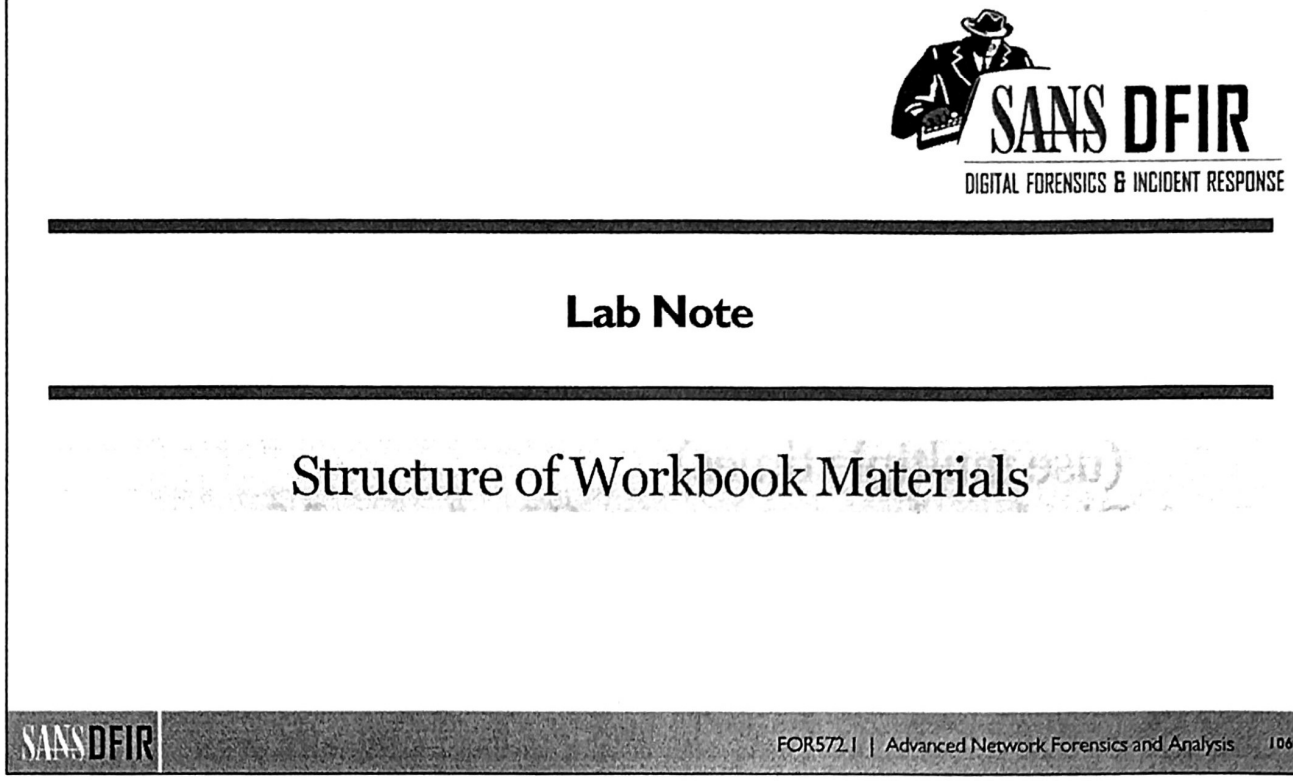
`-Y` Apply a display filter using the same syntax as with the GUI version of Wireshark. Because these may include characters that would cause special handling in the shell (parenthesis, spaces, etc.), it's best to enclose the filter string in single quotes.

`-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 entire class, and you'll soon see it's 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.

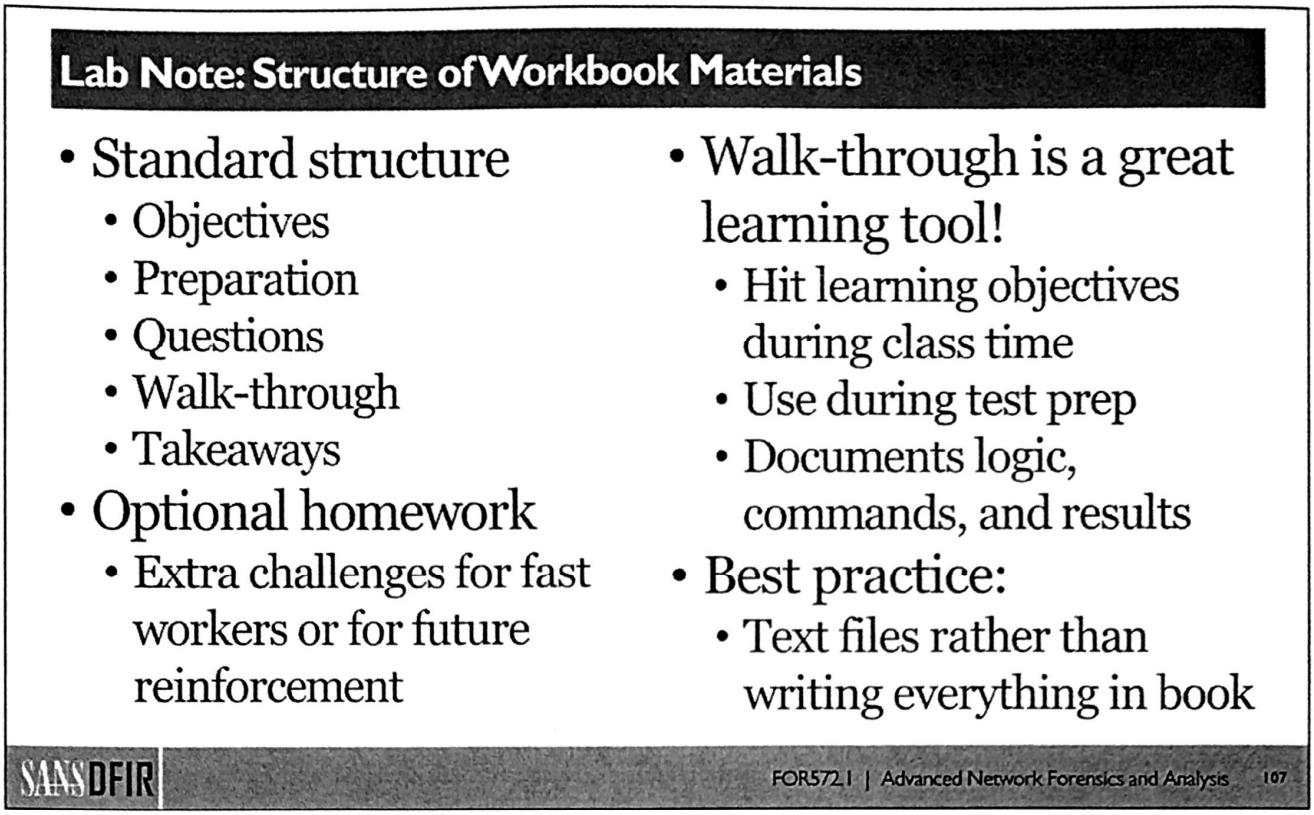
`-G` If you are unable to use a the Wireshark interface to determine field names, or simply want to perform a broader search, the built-in glossary can be helpful. For example, the command "`tshark -G fields`" will display a list of both the machine- and human-readable formats for each field that `tshark` can decode. This is most useful in conjunction with "`grep`" to isolate the field(s) of interest, because the entire list of over 155,000 fields is a bit unwieldy!

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

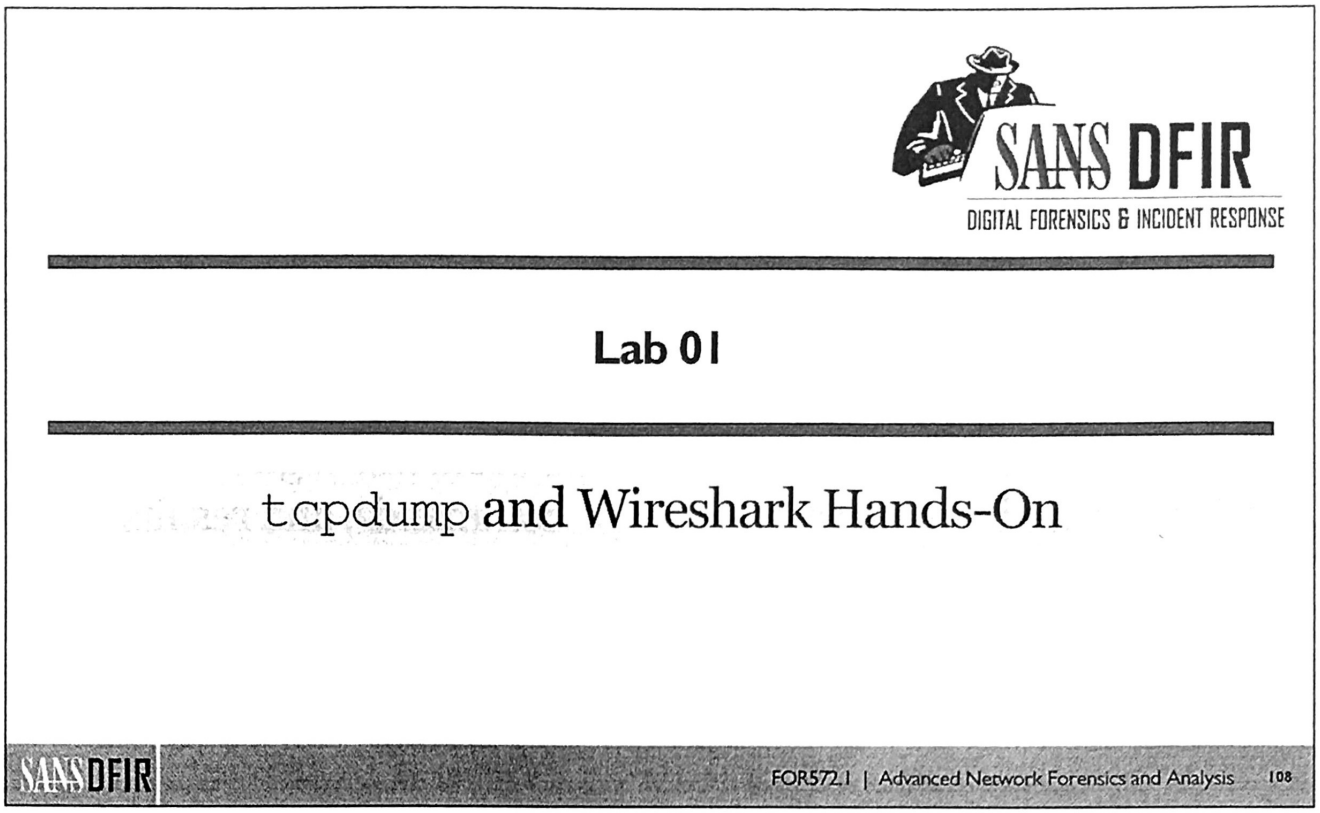


Because this is the first hands-on lab in the class, let's take a moment to explain the overall structure for each of the exercises you'll use during the class. These labs both reinforce classroom material as well as introduce some new tools and concepts. However, they all follow a common structure:

- 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 Walk-Through" sections reflect identical content, with the exception being that the walk-through 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 walk-throughs. Some students prefer to take the challenge head on and avoid peeking at any of the commands—which is great! Most will use the walk-throughs when they hit a roadblock, then shift back to the "Questions" section and keep plugging away. Those that are not as comfortable with the command line prefer to use the walk-through 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.)

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Lab 01 Objectives: tcpdump and Wireshark Hands-On

- Re-familiarize yourself with the basic operations of two of our core tools
  - Use `tcpdump` to capture traffic being sent across the network in real time
  - Use `tcpdump` to store network traffic to a pcap file
  - Open pcap file in Wireshark and use basic display filters
- Become familiar with `tshark` functionality

This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Lab 01 Takeaways: tcpdump and Wireshark Hands-On

- `tcpdump`: Observe/capture network traffic
  - Creating pcap files is primary means of storing evidence
  - `tcpdump` and `editcap` are effective tools to reduce large packet captures
- Wireshark brings powerful decoding capabilities
  - Display filters allow filtering at higher layers than BPFs
- `tshark`: All of Wireshark's power in the shell
  - Scalable and repeatable analytics with "`-T fields`"
- BPFs are MUCH faster to filter on layers 3-4—use `tcpdump` instead of `tshark` for data reduction

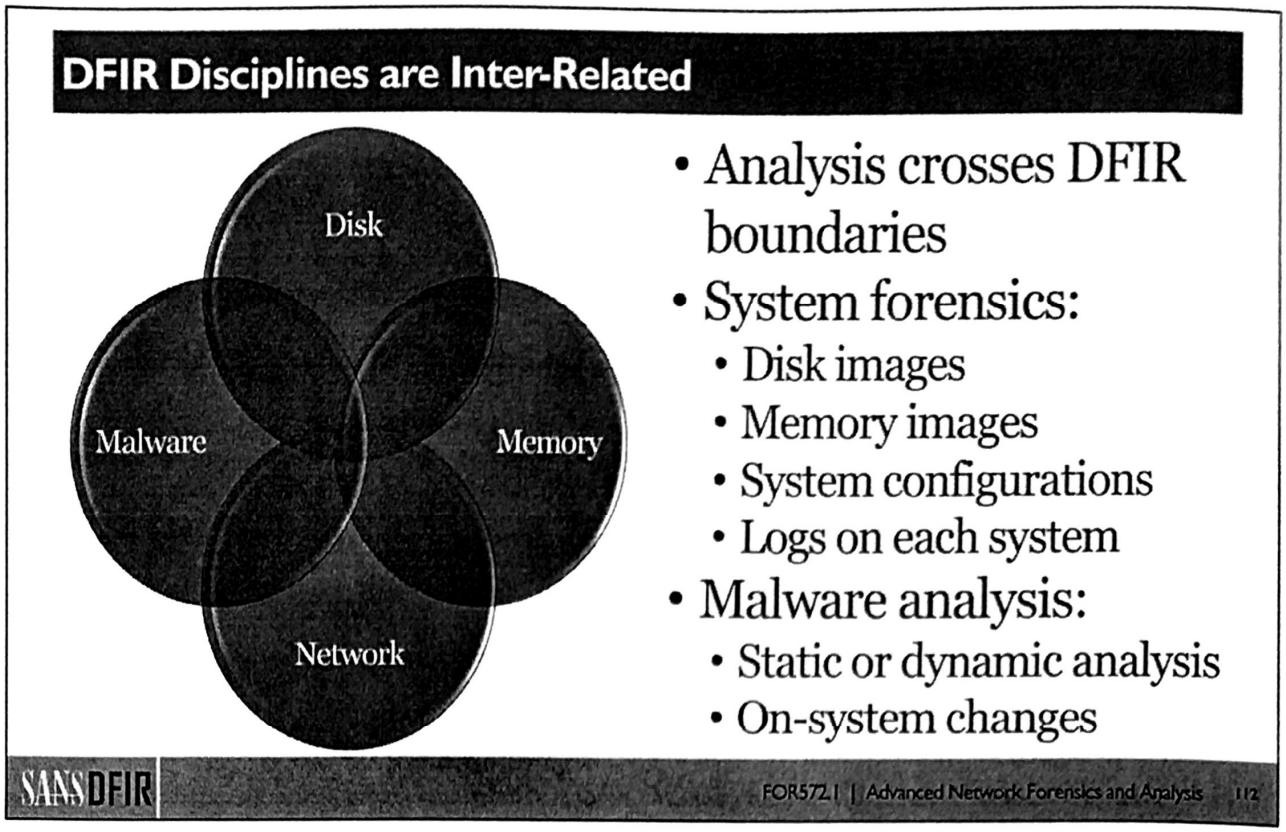
This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Network Evidence Acquisition

This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, 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 FOR408, 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 sub-discipline 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.

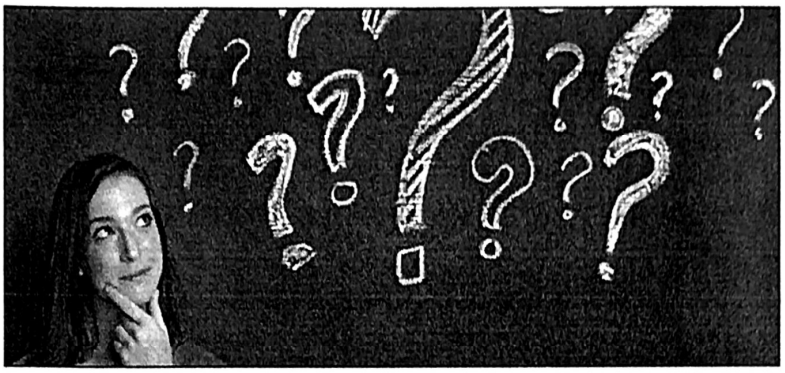
References:

`http://for572.com/for408`
`http://for572.com/for585`
`http://for572.com/for518`
`http://for572.com/for508`
`http://for572.com/for526`
`http://for572.com/for610`

© 2017 Lewes Technology Consulting, LLC, 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?

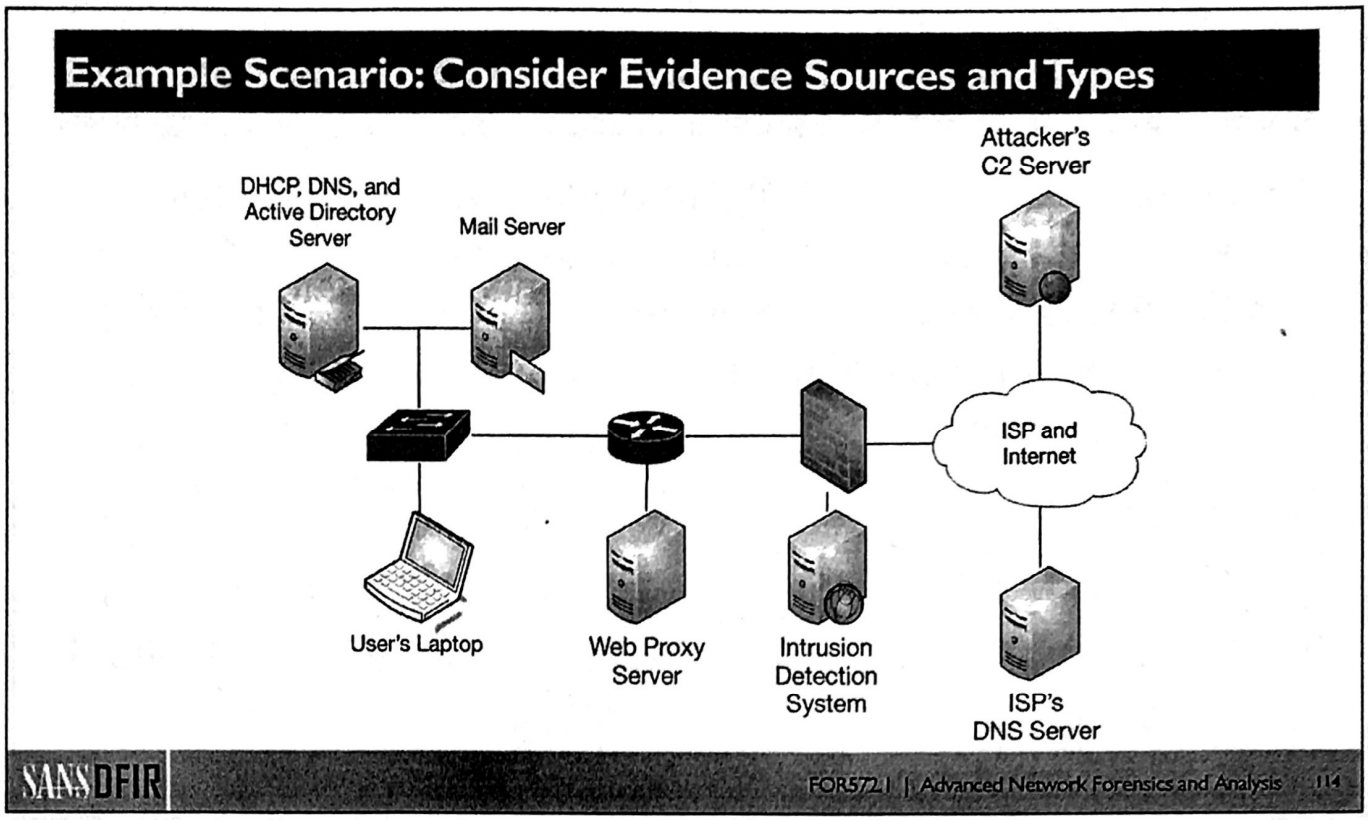


- How did malware get onto the system?
- What were other machines doing at that time?

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 task 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 to learn 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 in 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



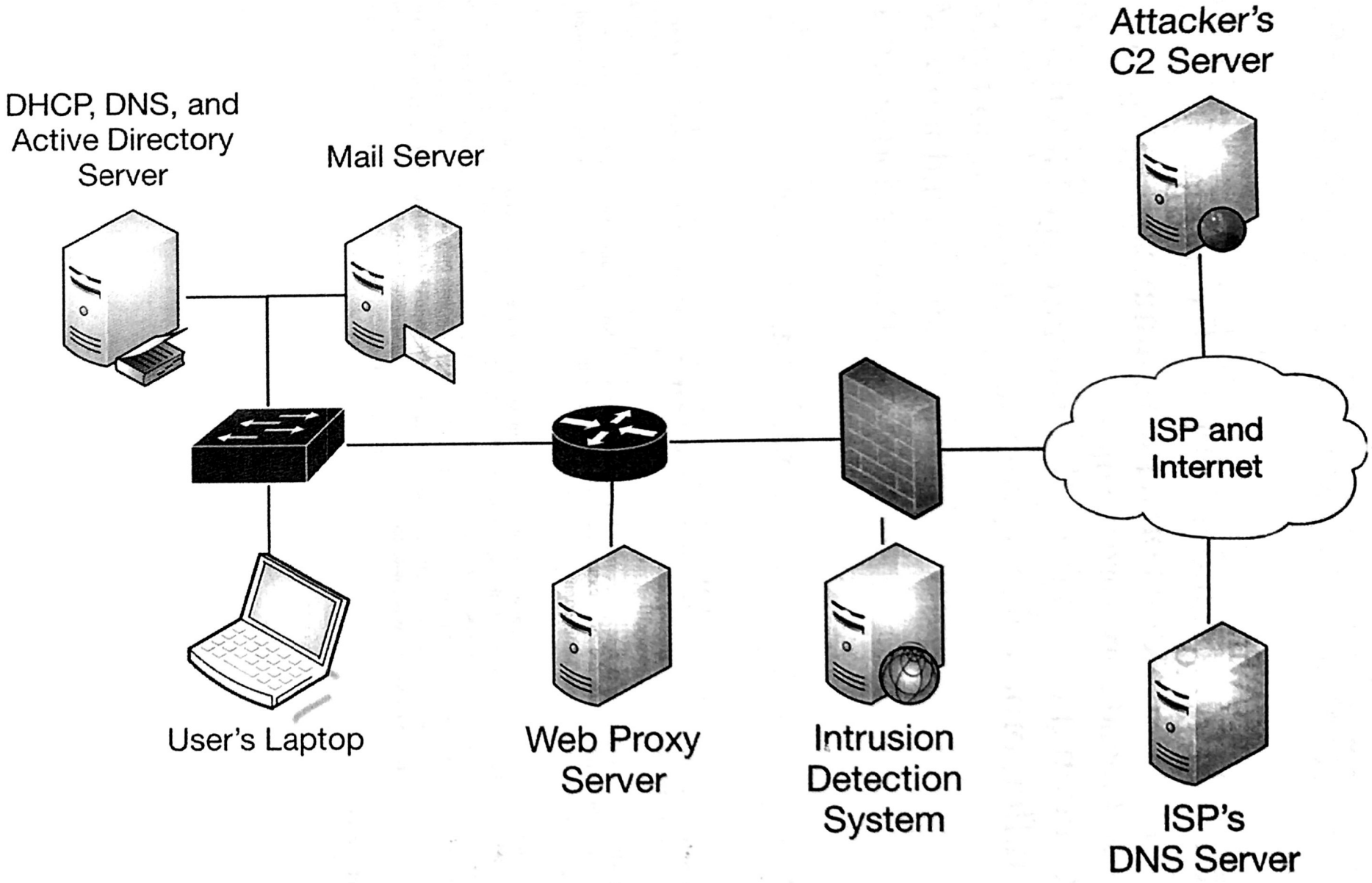
Consider the systems involved in a common but potentially devastating event—the phishing email.

In this example, the user has just connected their system (a laptop) to the office network and has clicked on a link in an e-mail 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 e-mail
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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Evidence Types: pcap Files

- `tcpdump`/windump generated
- Uses de facto standard libpcap file format
- Typical file extensions: pcap dump cap
  - Not mandatory—use file magic!
- Contain the data from the interface to which the sniffer/protocol analyzer was connected

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 which ever 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, however 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`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Evidence Types: Logs

- Excellent corroborating evidence
  - Seek Artifacts of Communication!
- Careful handling—easy to edit
- Require parsing and searching
- Collectable from a large number of devices
- May not go back far enough
- May not have sufficient fidelity of data
- Time synchronization is critical

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 in the course of 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 are 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").

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Evidence Types: NetFlow

- Cisco proprietary term: NetFlow
  - Versions: v5 (most common), v7, v9
- Open IETF standard: Internet Protocol Flow Information EXport
  - Based on NetFlow protocol (v9)
- Tallies packets sharing common characteristics
  - Same hosts, ports, and protocol
- Records volume, timing, and count of packets
- For our purposes: SFlow, JFlow, etc. are equivalent

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 RFC5470, RFC5101, RFC5102, and RFC5103. (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 is connection meta data that is much smaller than the data itself. The biggest problem with high-bandwidth networks is that pcaps rapidly become unwieldy and difficult to move and slow to analyze.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

How Do We Acquire? Switches (1)

- Enterprise switches: port mirroring
  - Cisco's trade name: SPAN port
  - A "software tap" that duplicates packets and sends the copies out to a specified port
  - Identify specific ports or VLANs to monitor
  - Some hardware allows more granular source specification

Let's take just a few moments to re-familiarize 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.

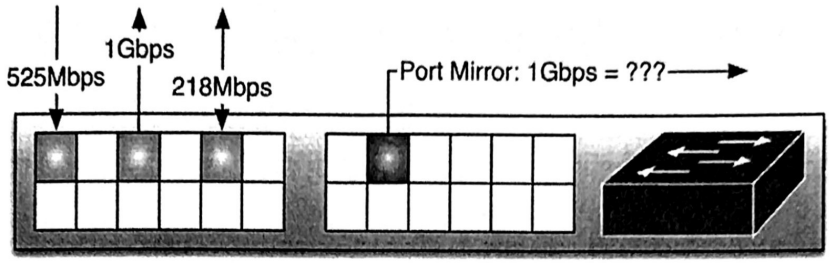
This is not a comprehensive review of all methods to capture traffic. We will not cover the more exotic technologies worthy of Hollywood movies (nuclear-powered undersea cable taps), or more recent ugly-yet-functional solutions (such as using a "dumb" hub to interrupt a link of interest). Instead, we'll 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.

© 2017 Lewes Technology Consulting, LLC, 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



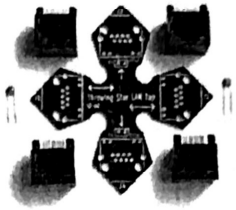
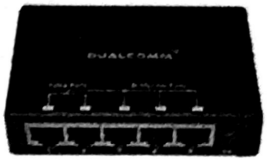
Perhaps the greatest benefit to using port mirroring is that the platform already exists in most enterprises—it's merely a software configuration tweak to activate the feature. The traffic of interest is automatically pushed to the designated port in addition to its existing flow. In environments where 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 of 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.

© 2017 Lewes Technology Consulting, LLC, 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

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 re-aggregate the traffic to a comprehensive, full-duplex stream. Other taps will perform the aggregation, presenting a single full-duplex monitoring port to which the monitoring hardware can connect through 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 re-combined after the capture is complete.

Another useful feature that a "regenerating" tap provides is multiple monitoring ports. This category of tap will make multiple copies of each packet, so more than one monitor or capture device can be used at once. Such a setup is especially helpful when an intrusion detection system is in use at all times, but the security teams want to provide a pre-positioned tap point for an as-needed traffic capture solution.

The Ninja Throwing Star Network Tap[1] pictured in the center is an interesting hardware variation on the 100Mbps split direction network tap. The Dualcomm company offers several affordable tap platforms that operate at 100Mbps[2] or 1Gbps.[3] Depending on your budget and operational requirements, these may bear consideration for a tactical "Incident Response kit".

References:

[1] `http://for572.com/sby16`
[2] `http://for572.com/yu6-t`
[3] `http://for572.com/0s6n7`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

How Do We Acquire? Taps (2)

- Pro: Single-purpose, highly engineered
  - Network traffic is not dropped
  - Redundant and fail-safe
- Con: Installation process and cost
  - Installing requires downtime
  - Cost can be very high, limiting pre-positioning

The main benefit is that with a purpose-built device such as a tap is that they are engineered to do their jobs 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!

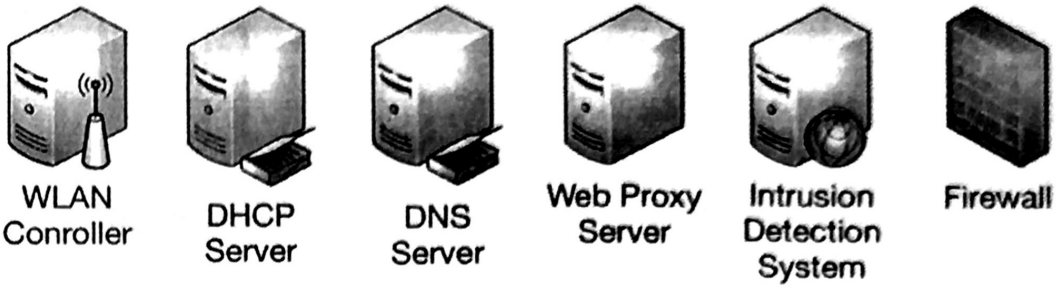
Of course with great power come large price tags. The major drawback on taps is the cost—it's not uncommon for a high-grade device to cost several thousand dollars. At that price point, scattering a few dozen taps around a network environment is not an option for most environments. This leads us to another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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

OSI Layer 7 Sources

- Many platforms generate logs
- All corroborate observed activity
- May not provide deep knowledge of logged events
- Aggregation needed for effective analysis



SANSDFIR

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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Internal NetFlow Data

- Data flow information
- Less detailed than pcaps
- Smaller than pcaps so can be retained longer

Router

Firewall

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] or `nprobe`[2]. (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 is an easy way to scope a compromise, evaluate whether newly identified indicators identify potential attack traffic in past collections, and can 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.

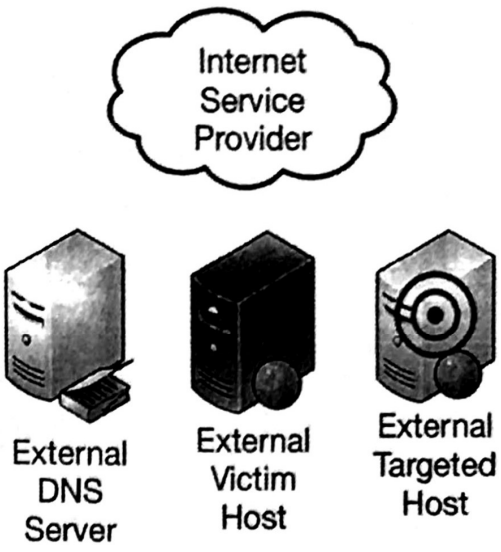
References:

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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

External Sources

- ISP or outsourced Internet DNS services
- ISPs sometimes retain NetFlow data
- Other targets or victims



Other sources of evidence can be external to the organization. These are more common that 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`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

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

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Network Data Collection Planning: Ideal Case (2)

- Pros
  - Collect evidence before you need it: Continuous Monitoring aka SANS SEC511
  - Easy to activate without downtime, ensures strategic focus on IR/forensic processes
- Cons
  - High cost for "maybe" requirements, can be limiting if problems don't occur where capture devices are, possible proprietary storage formats

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Network Data Collection Planning: Acceptable Case

- Standby hardware and good plans
  - Pre-build 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.

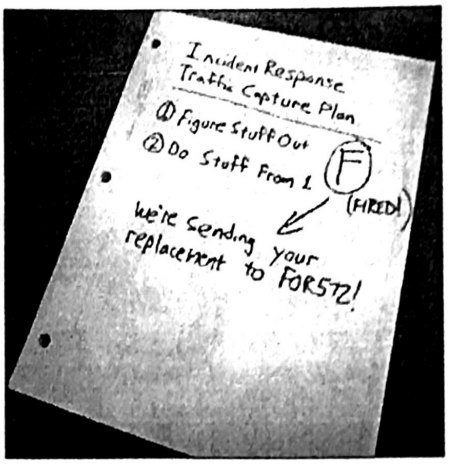
Deployment of these tactical capture platforms can sometimes be more complicated, however. If downtime to a production system is required before it can start collecting packets, then business and mission impacts may outweigh the immediacy of placing it into operation. (I have personally seen victims 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.

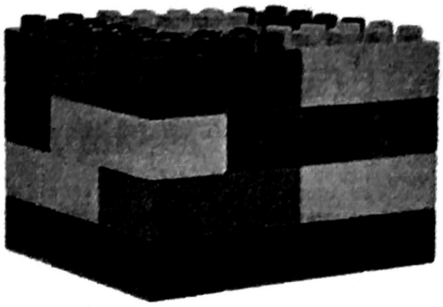
© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Network Data Collection Planning: Worst and Realistic Cases

- Case never:



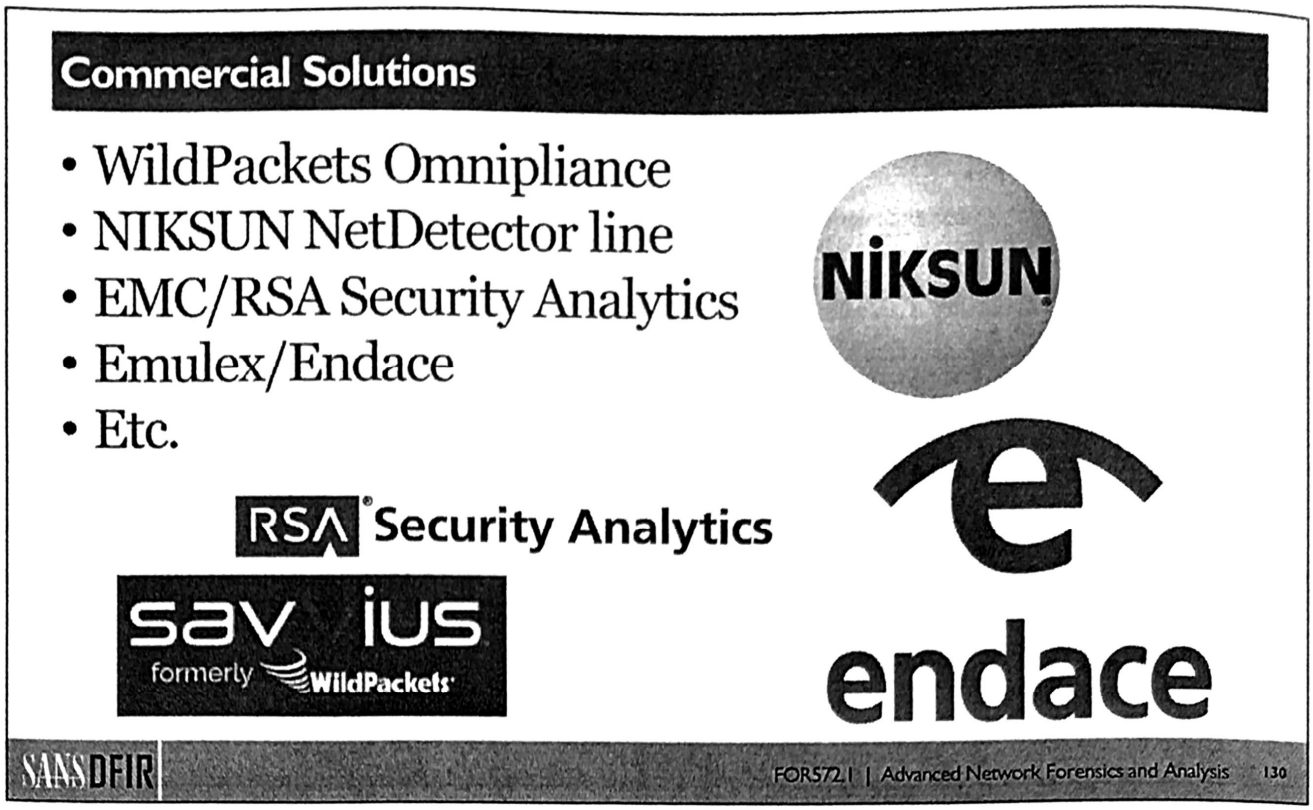
- Realistic case:
  - Hybrid solution



One design case that we should **never** pursue, of course 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.

© 2017 Lewes Technology Consulting, LLC, 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.

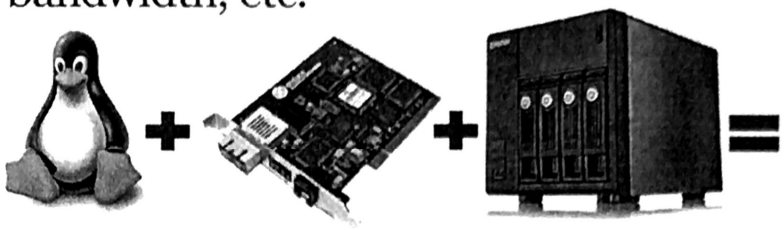
We'll spend a great deal more time on commercial tools later in the course, but these can all provide network capture functionality, including full packet content in most cases. Many such solutions are marketed as "network flight data recorders", which is a reasonable analogy. At the core of each type of device is the ability to acquire and store packets. A variety of additional analytic functions are also available, depending on the specific platform.

- Savvius (formerly WildPackets) Omnipliance TL Network Analysis & 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`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Home Grown 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.



Build your own lower-cost platform

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 very simple task if you consider the specific requirements the task demands.

The tradeoff, 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" applies here![1])

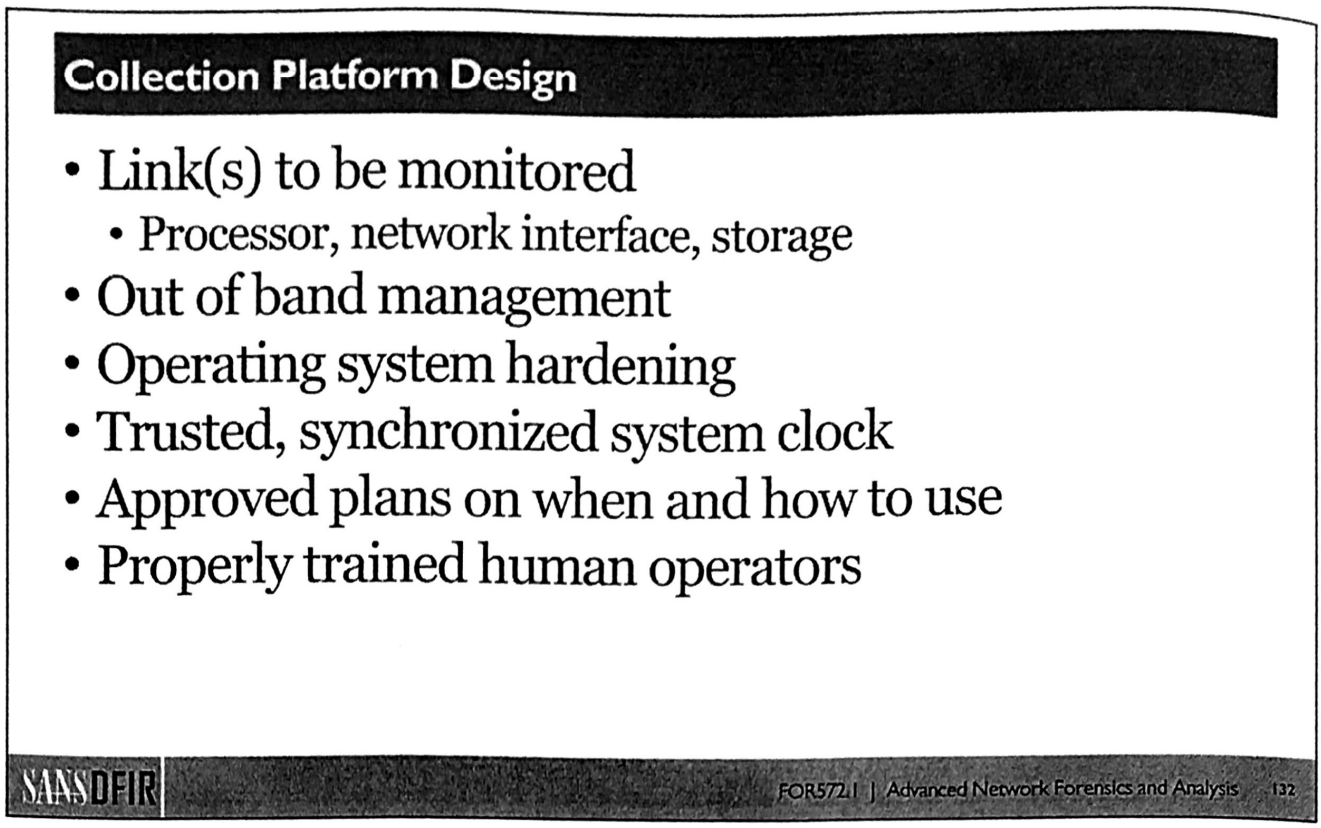
That said, a home-grown platform can be tailored perfectly to your specific requirements. Before building such a platform, we of course want to consider our requirements—both technical and non-technical. We'll discuss a few of these in the following slides.

The following SANS Gold Certification paper may prove to be a useful reference if you need one for a lab or even an operational environment: `http://for572.com/i59jl`

References:

[1] `http://for572.com/-xman`

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



Of course 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.

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.

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 very 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 that will be tasked to carry out the capture activity should be trained on both the equipment and the plan.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

What to Capture (1)

- Standard "Computer Science answer"
  - It depends!!
- Highly dependent on numerous factors
  - Business objectives
  - Administrative and legal constraints
  - External regulatory requirements
  - Technical limitations

Regardless of what platforms, software, or hardware used to acquire network data, there is still a fundamental question about what to capture with all these toys. We quickly arrive at the fact 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 multibillion 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 surely 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 e-mail communications, it would be inconsistent with this policy to capture some or all e-mail-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 indeed 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 simply accept a policy that limits that retention to a few days or weeks. (This applies to almost every type of evidence we will discuss in this course—not just pcap data.)

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

What to Capture (2)

- Consider the big picture—what evidence is available from other parts of the network
  - HTTP proxy logs and cache
  - Passive DNS logs
  - Logs, logs, logs
  - NetFlow collectors

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.

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—what else is already available that might minimize the amount a packet capture solution would need to pull from the wire.

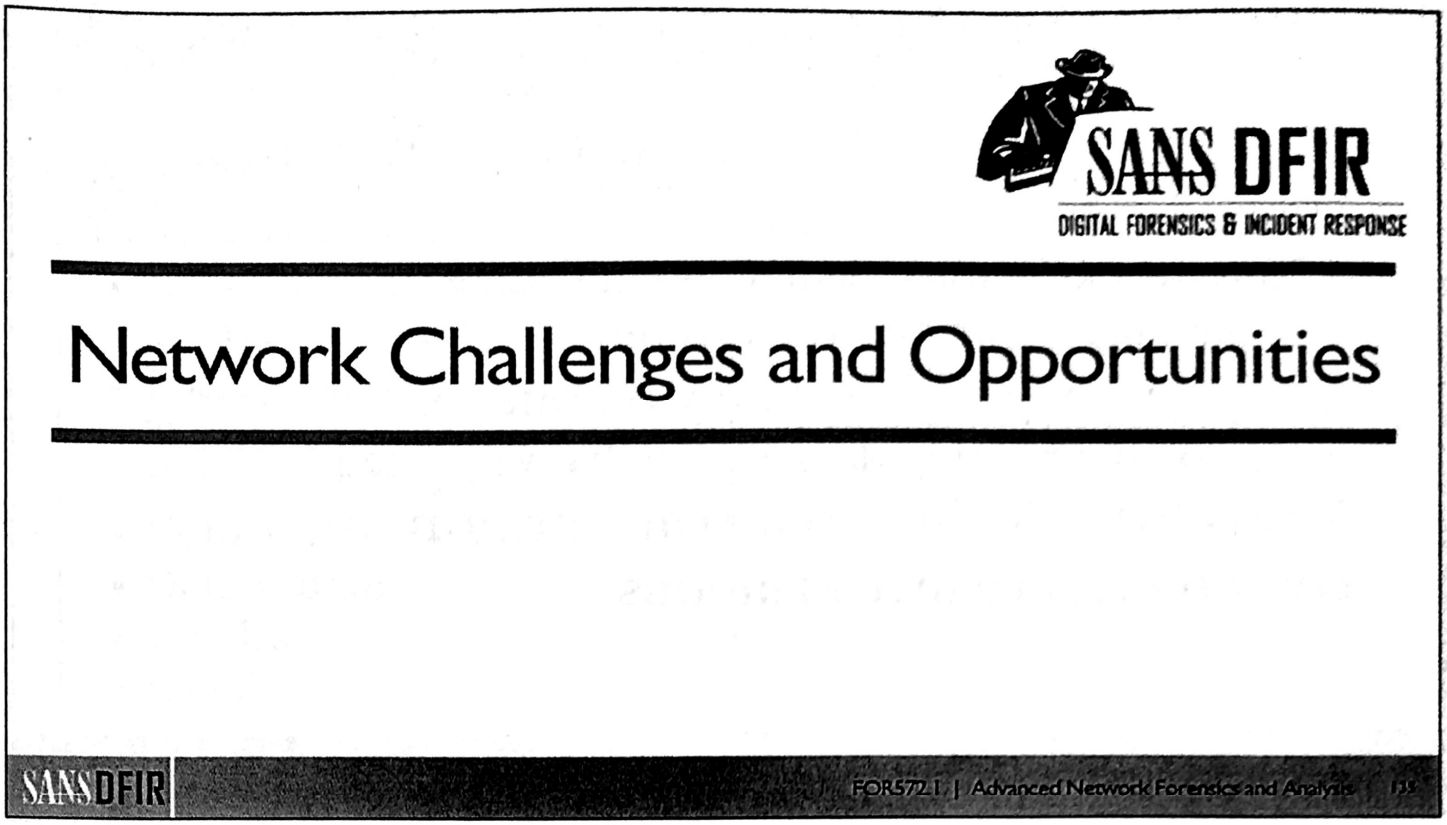
For example, consider the vast amount of evidence available and sound findings made during the proxy walk-through earlier in this section. Without touching a single pcap file, we were able to clearly identify a subject's web activity, potentially recovering the data he or she uploaded to a suspicious 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.

Although we don't cover DNS in detail until later in the course, 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.

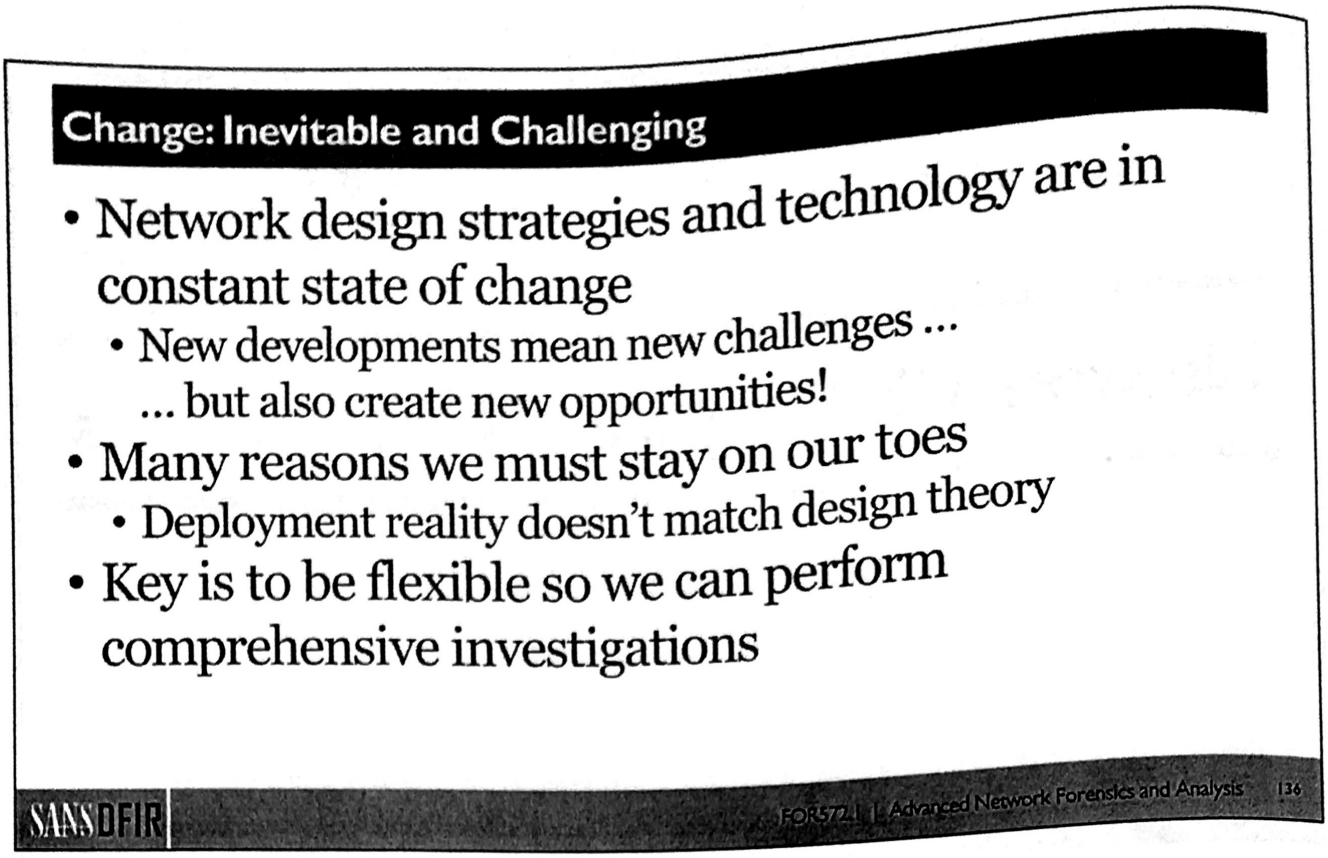
NetFlow collections can also provide excellent visibility, especially for encrypted communications.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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

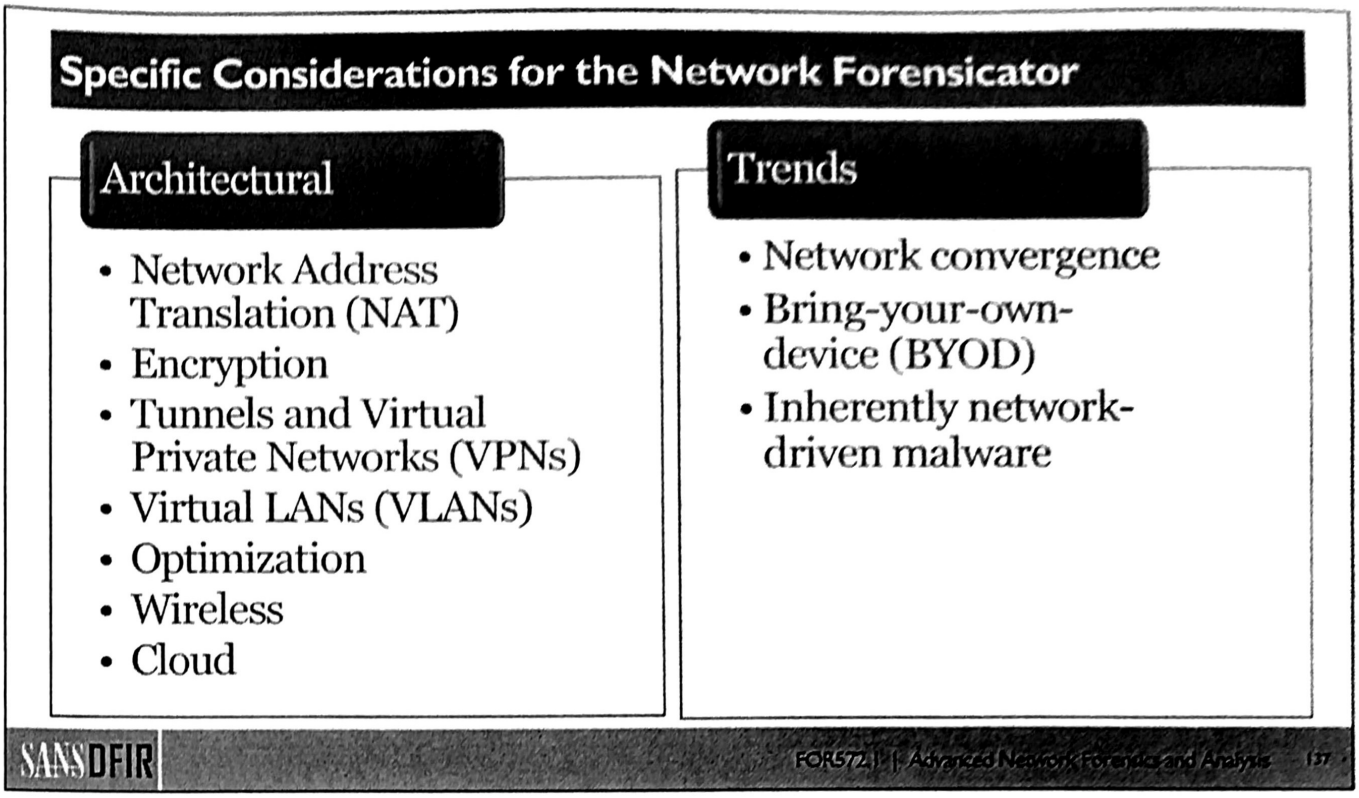
We work in a field that has and will continue to undergo constant change. As the network becomes central to so much more of modern business processes and our daily lives, the constant change is perhaps even more a part of our work as network investigators. With each new technology that hits the market, 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 pre-determined sequence of buttons and menu items in a pre-selected 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. But, 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 is in our ability to handle the unexpected. This game is often won or lost on an investigator's ability to seize upon unexpected opportunities.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



We'll take a look at a number of technologies, features, or situations that might be discovered during a network-based investigation. Although these may necessitate a change in the investigative approach, they should not always be seen as a barrier to our progress. They will certainly present challenges, but also can provide additional opportunities, and may even be a significant benefit in the end. Remember: Success often rides upon the ability to capitalize on unexpected opportunities!

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Architecture: NAT (1)

- Maps "inside" IP(s) to "outside" IP(s)
  - Changes the evidence observed
  - Position of observation matters
  - Often a workaround for proper network integration
- Some capture technologies account for this
  - NetFlow on routers
- Often must correlate through other means
  - Timestamp + source port
  - Proxy log containing GET requests

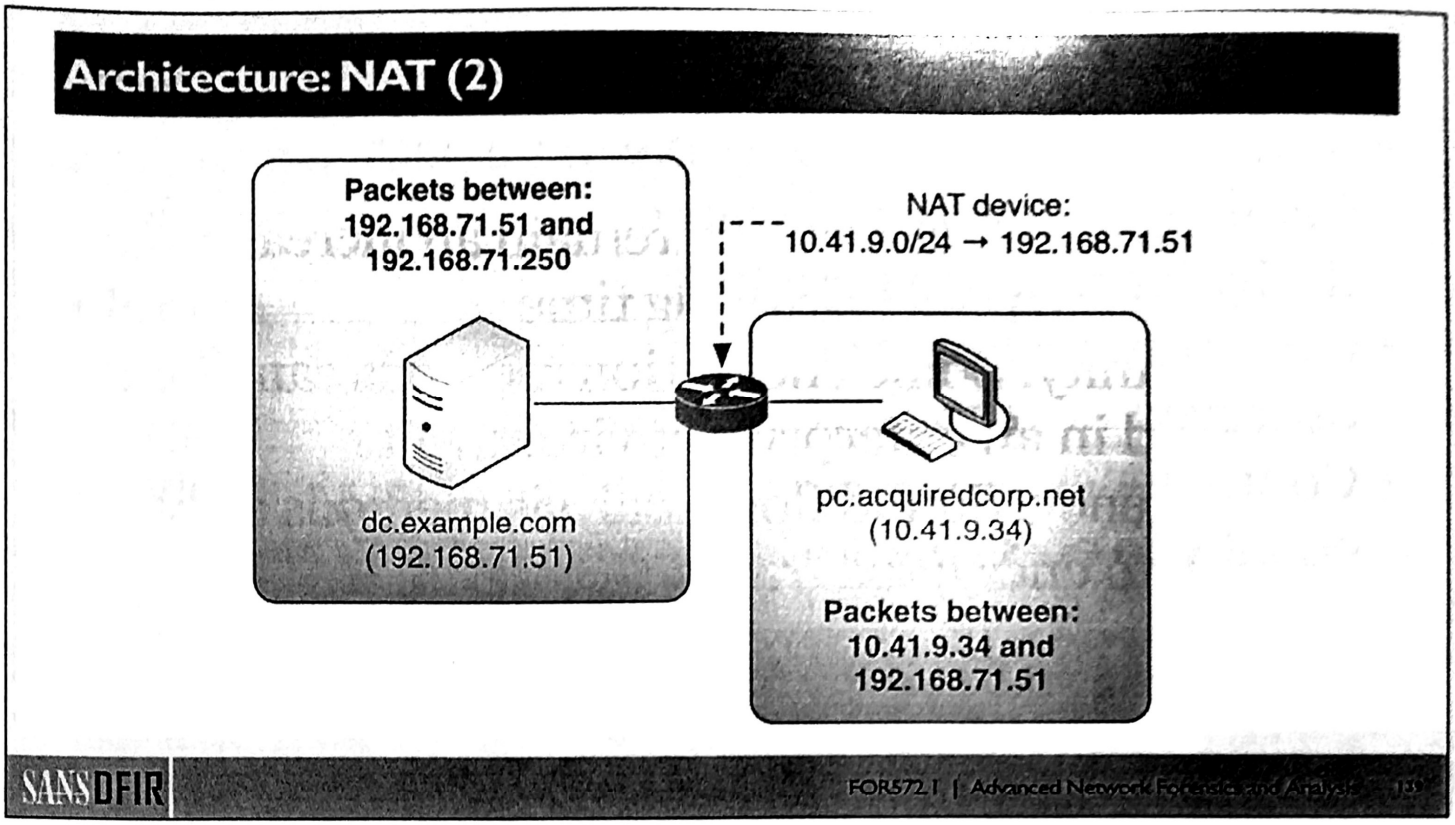
The original scope of the Internet's addressing structure seemed limitless. Who could imagine more than four billion endpoint devices online back in 1980, when RFC 760 (superseded by RFC 791[1]) defined that 32 bits would be used for an IP address? Today, of course, the IP exhaustion is a serious problem 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 (RFC1918[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 is commonly seen is after network merges, which often occur after one business acquires another.

References:

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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



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 for 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. Re-consider 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 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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Architecture: Encryption

- May be effectively impossible to inspect
- Challenge: This is and will remain an increasingly difficult challenge for a long time
- Opportunity: Some encryption methods can be observed in an enterprise environment
- Contingency: Sound flow analysis methods apply equally to encrypted traffic

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. We discuss encryption in much greater detail later in the class. There are some very sound, robust methods that can be used to help re-claim some ground in this arena. In a typical enterprise environment, there are commercial and free solutions that can be used to enable plaintext inspection and capture of most SSL-encrypted traffic.

From a law enforcement perspective, capturing SSL-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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Architecture: Tunnels and VPNs (1)

- Centralized oversight for large pools of decentralized users
- Protocol-over-protocol encapsulation often used to bridge multiple offices/sites
  - GRE, IPv6, SOCKS, SSH, higher-layer protocols

SANS DFIR

In a pure tunneling setup, one stream of network traffic is fully encapsulated as the payload within another stream of network traffic. This can occur over purpose-designed tunneling protocols such as GRE, or through re-purposed 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. Even consumer-grade devices such as NAS or perimeter routers commonly provide this feature.

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.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Architecture: Tunnels and VPNs (2)

- Challenge: Incorrect monitor placement only sees encapsulated or encrypted flows
- Opportunity: Proper monitor placement has visibility of many users' traffic
- Gotcha: VPN from inside to outside?
  - Surfbouncer, VPN-to-home

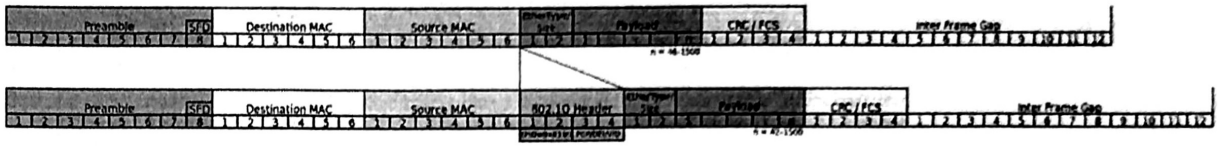
Given that relatively minor hurdle to clear, a VPN means that most or all of the remote users' traffic can be centrally monitored or captured. When cross-checked against the VPN authentication logs, it becomes trivial to associate traffic of interest to a source system, user account, and possibly a location.

An interesting consideration for VPN traffic is that is doesn't always start with remote users connecting to the "home office". Many commercial services provide VPN-like connectivity including encryption and obfuscation so users within the environment can connect to the Internet without the pesky security team's prying. The combination of consumer devices that provide VPN servers and dynamic DNS services means that users may start to grow accustomed to connecting back to their home network resources when they are away. By fully understanding an enterprise's VPN architecture, this activity will be easy to spot and act upon.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Architecture: VLANs

- Combine multiple logical network segments onto one physical segment via pseudo-encapsulation
- Challenge: VLAN-aware tools, oversaturation
- Opportunity: Observe/capture multiple segments with single platform



Virtual LAN, or VLAN, is a technology that tags IP traffic so multiple distinct segments can be transmitted over a single physical link. Often referred to by its IEEE protocol identifier of 802.1Q, or "dot1q", VLANs have become almost ubiquitous even in smaller networks.

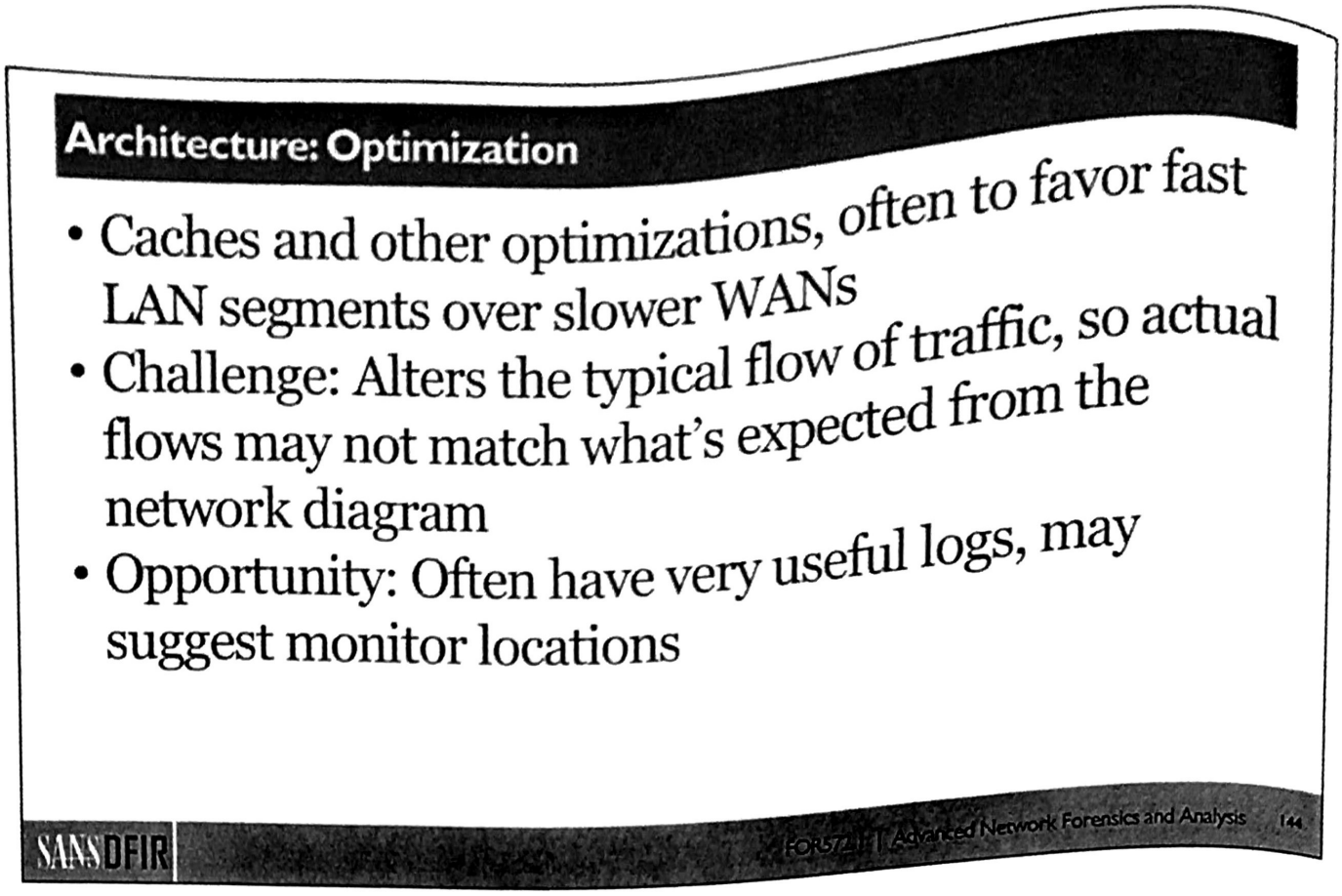
Each VLAN is identified by an integer between 1 and 4,095. (VLAN 1 is typically reserved by the switching hardware.) The VLAN ID is contained in a four-byte field inserted into the Ethernet frame after the source MAC address field. A link over which multiple tagged VLAN streams is sent is called a "trunk". However, as you may imagine, inserting bytes into the middle of an established protocol can cause parsing problems!

The largest issue with analyzing a VLAN trunk is that the tools must be VLAN aware. A tool that expects a protocol field to start and end at a certain byte offset is easily confused by an extra four bytes in the header! Fortunately, libpcap tools should easily handle tagged VLAN traffic via the 'vlan' BPF primitive. Another possible pitfall is oversaturation, particularly when acquiring with port mirroring. Mirroring a VLAN that spans multiple ports is subject to the same limitations as discussed previously.

VLANs can be helpful to us in that a single traffic capture platform can acquire from multiple logical segments when placed on the trunk. As long as a tap is used or oversaturation concerns are addressed, and the toolset can handle the tagged traffic, this technology can greatly simplify a capture deployment while improving the investigator's visibility within the environment.

Image Credit: `http://for572.com/9g24m` (Bill Stafford, Creative Commons Attribution Share-Alike 3.0 Unported)

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



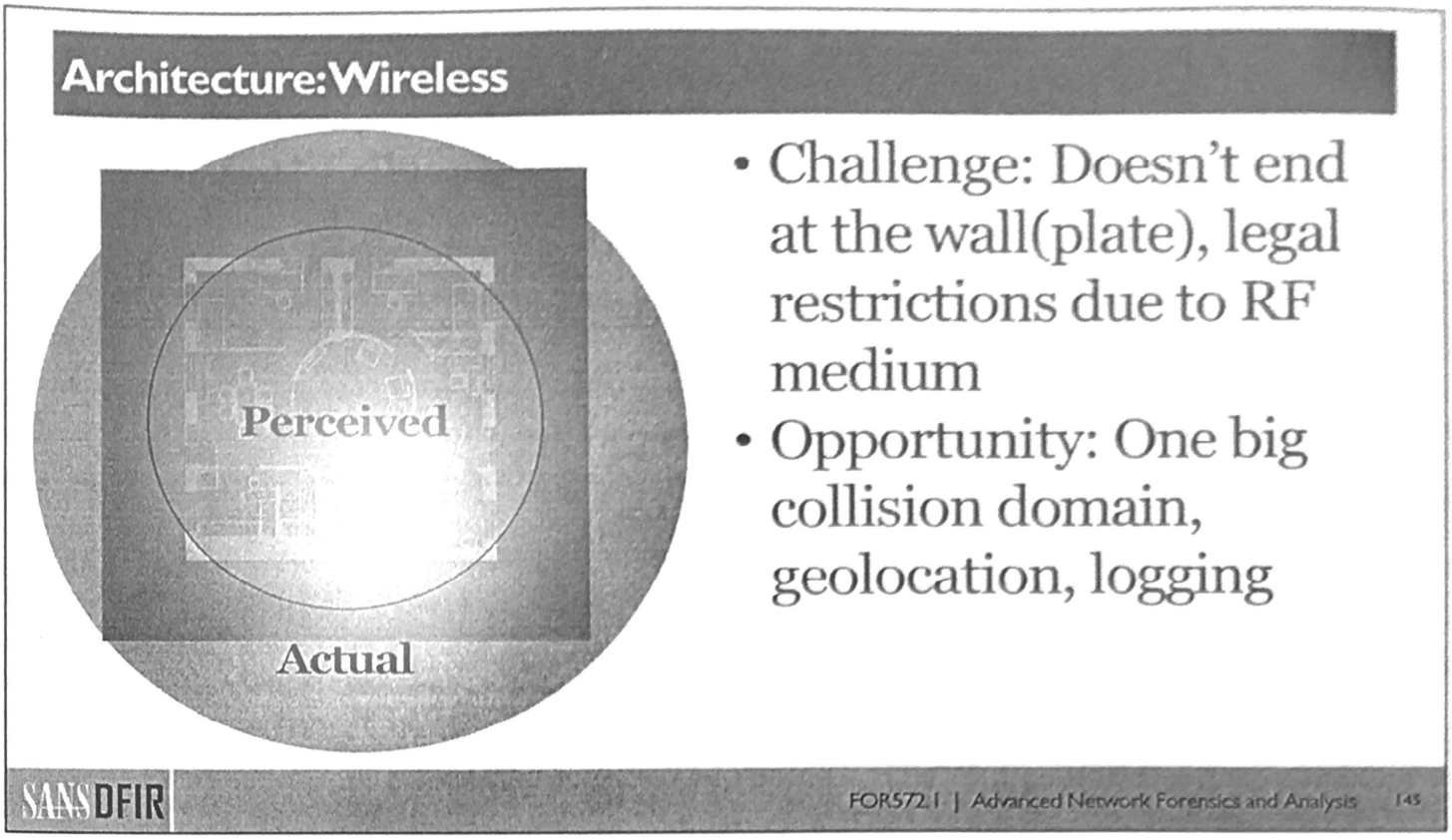
A common way to improve speed and network utilization is to deploy network optimization devices. Perhaps the most common of these is the web proxy server. Of course, this is a device that keeps local copies of HTTP-sourced objects, ensuring identical content is only pulled from a remote server once, 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 e-mail, 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 such 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.

© 2017 Lewes Technology Consulting, LLC, 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 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 further away than intended. This means that even physically "secured" wireless deployments certainly extend past their designed service areas.

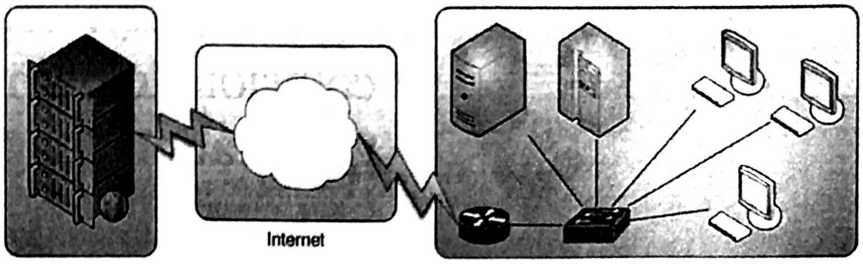
Another consideration, mentioned briefly before, is that legal requirements often change significantly when entering the wireless realm. These legal stipulations and restrictions vary greatly between jurisdictions. Because this is not a law class, we'll just say that you must get appropriate legal guidance 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 for a device of interest. Finally, most large-scale wireless deployments use hardware that keeps very good logs, which can be extremely useful during an investigation, as we'll see later.

© 2017 Lewes Technology Consulting, LLC, 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 very 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.

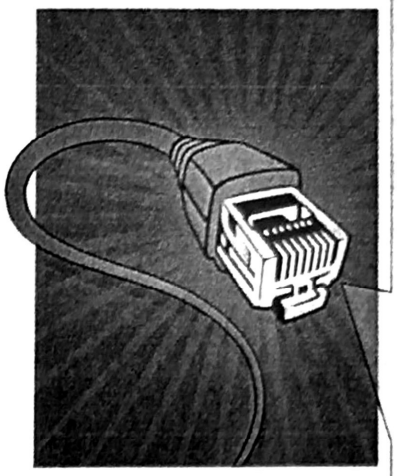
Another hurdle to overcome when investigating an incident involving a hosted service is that the model is new and undergoing rapid change. Although the forensic community is starting to address the cloud trend, each service has the potential to be completely different than anything previously seen or 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) is easily 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.

© 2017 Lewes Technology Consulting, LLC, 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
...
???

SANS DFIR | FOR572.1 | Advanced Network Forensics and Analysis 147

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.

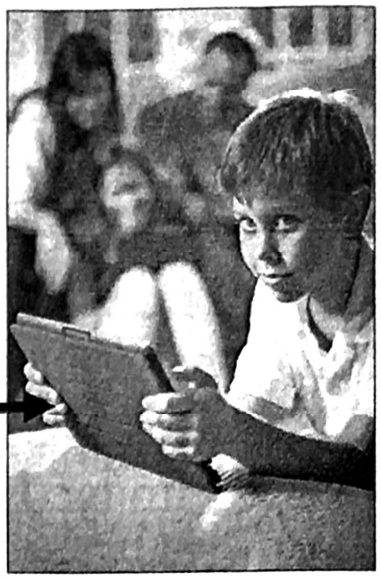
We're seeing a huge shift to using the network for more and more business communications. Voice over IP (VoIP) technology and devices are converging the data and voice networks. Storage Area Networks (SANs) and iSCSI are using Ethernet as a foundation to perform storage-related tasks. Backups, remote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

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Trend: BYOD

- Relying on 100% "company-procured" assets is becoming less feasible
- Challenge: Legality, infinite device possibilities



(Corporate e-mail account)

SANS DFIR | FOR572.1 | Advanced Network Forensics and Analysis 148

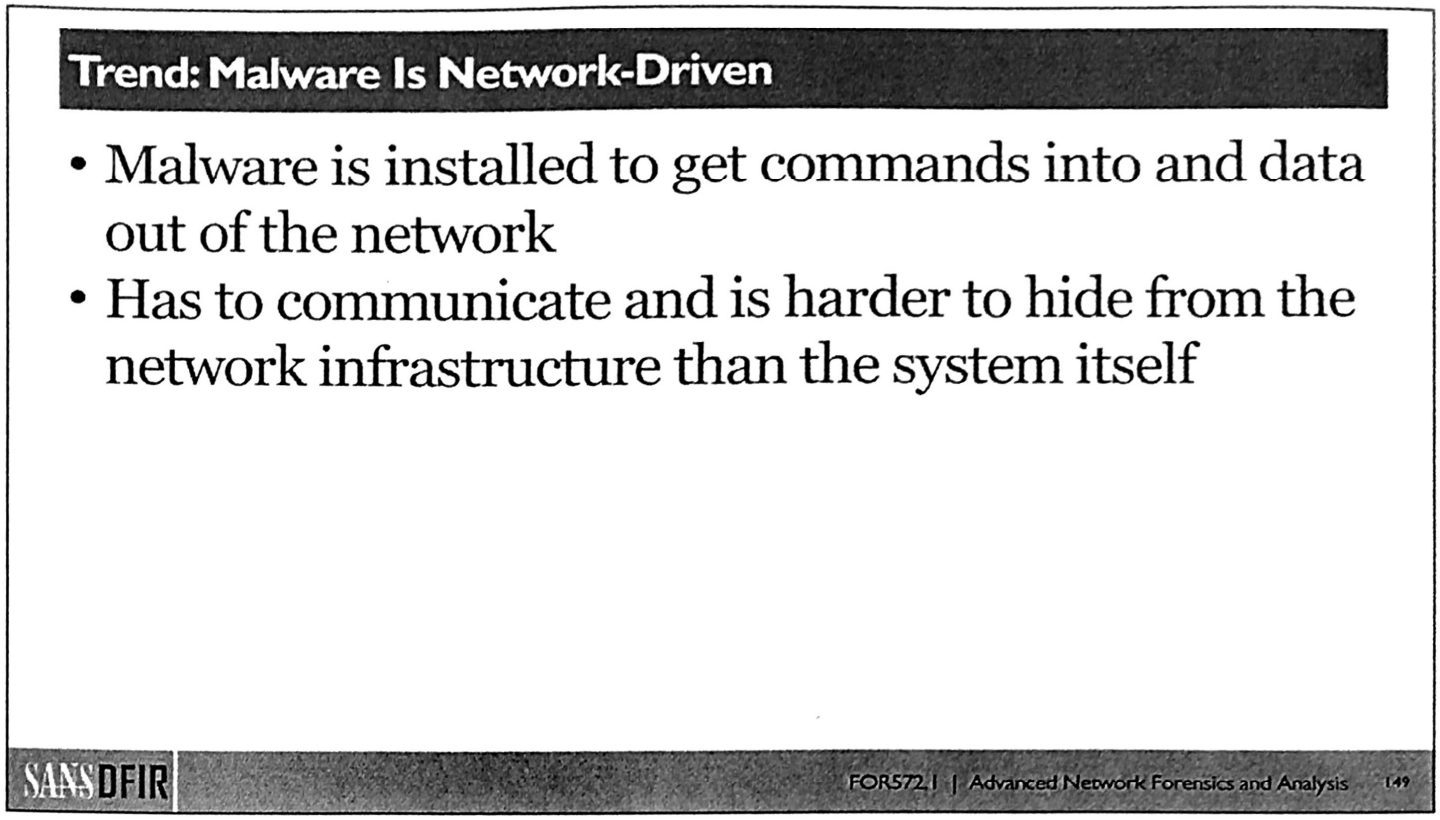
As anyone that works as or with a system administrator will tell you, the "Bring Your Own Device" trend is in full effect. Users are no longer content to rely solely on company-provided computing platforms, even if it means paying for the hardware themselves. Some will point to the new technology allowing workers to do their jobs better/faster than they can do with corporate resources. Of course there may be some truth to that, but a major component of this "requirement" is the need for the newest shiny toy and simplicity of carrying fewer devices.

The reason BYOD is even an option for many users is that the network provides access to the business data and functions. The most basic example being e-mail access from a user's own device. Today, however, this extends to many additional business processes through the use of hosted services, remote access technologies, and more.

The security community is just starting to address the BYOD trend through new policies and procedures. However, with such a heavy reliance on the network for these functions, the need to account for BYOD in our investigations is inevitable. The legal framework for using personally-owned devices in business operations is still being developed and will certainly be challenged. Although the legal issues are obviously outside the scope of this course, be sure to work closely with inside council or the appropriate legal resources to ensure new or adapted processes are in line with local laws and policies.

Another challenge is the speed at which the technology market churns out new products. Although a homogenous operating environment is ideal from a management perspective, there are likely hundreds of different devices in use in a BYOD enterprise, which complicates investigation where personally-owned devices are involved.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

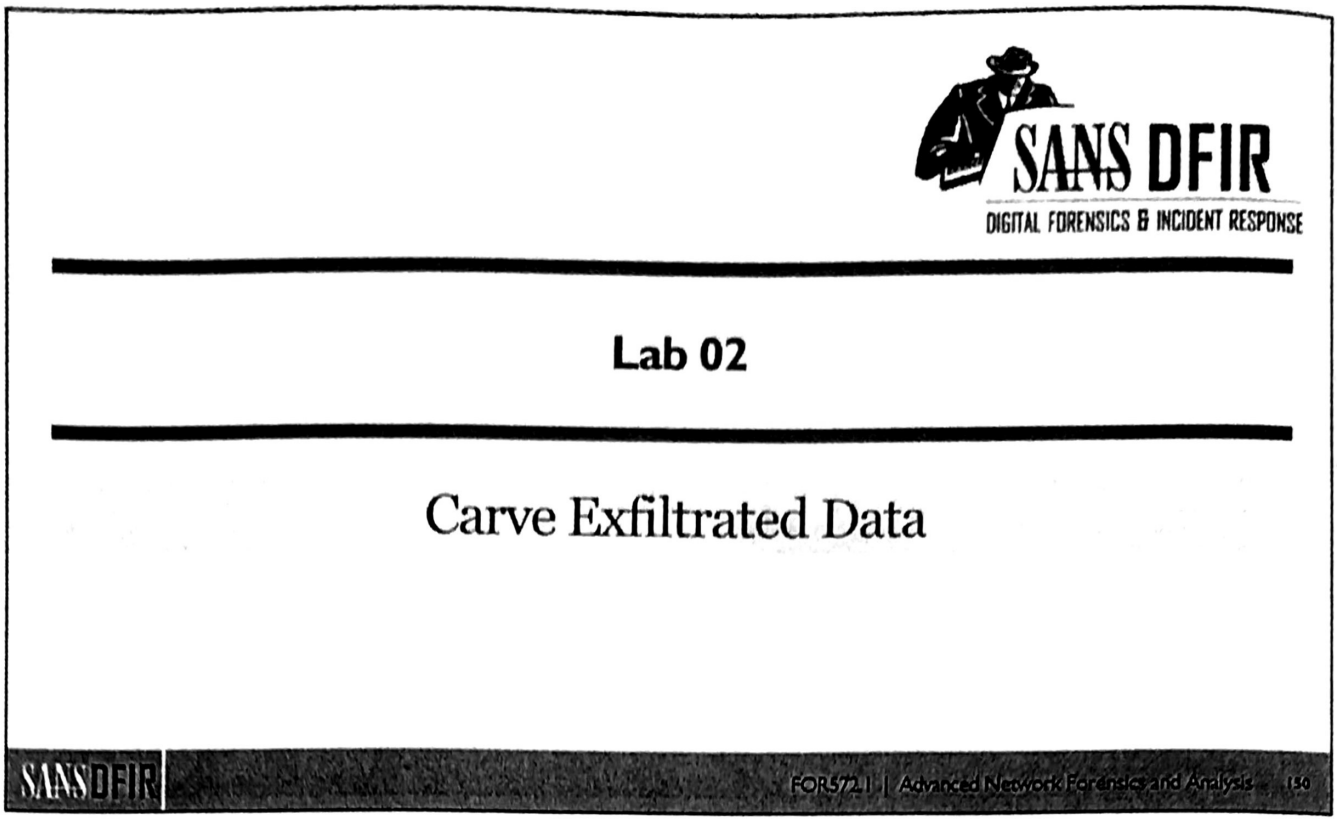


Lastly, we have to consider that the fundamental nature of malware is to accept commands from its master and provide data of some kind in return. Hollywood spy movies notwithstanding, malware generally communicates using the network. Although there are some very good methods for malware to "hide" from 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 very extensively. We should seize upon this weakness while it still exists!

© 2017 Lewes Technology Consulting, LLC, Mat Oldham



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Lab 02 Objectives: Carve Exfiltrated Data

- Identify and isolate relevant TCP stream from a full-content pcap file
- Extract payload data from TCP stream
- Analyze reconstructed data to establish investigative value

This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Lab 02 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

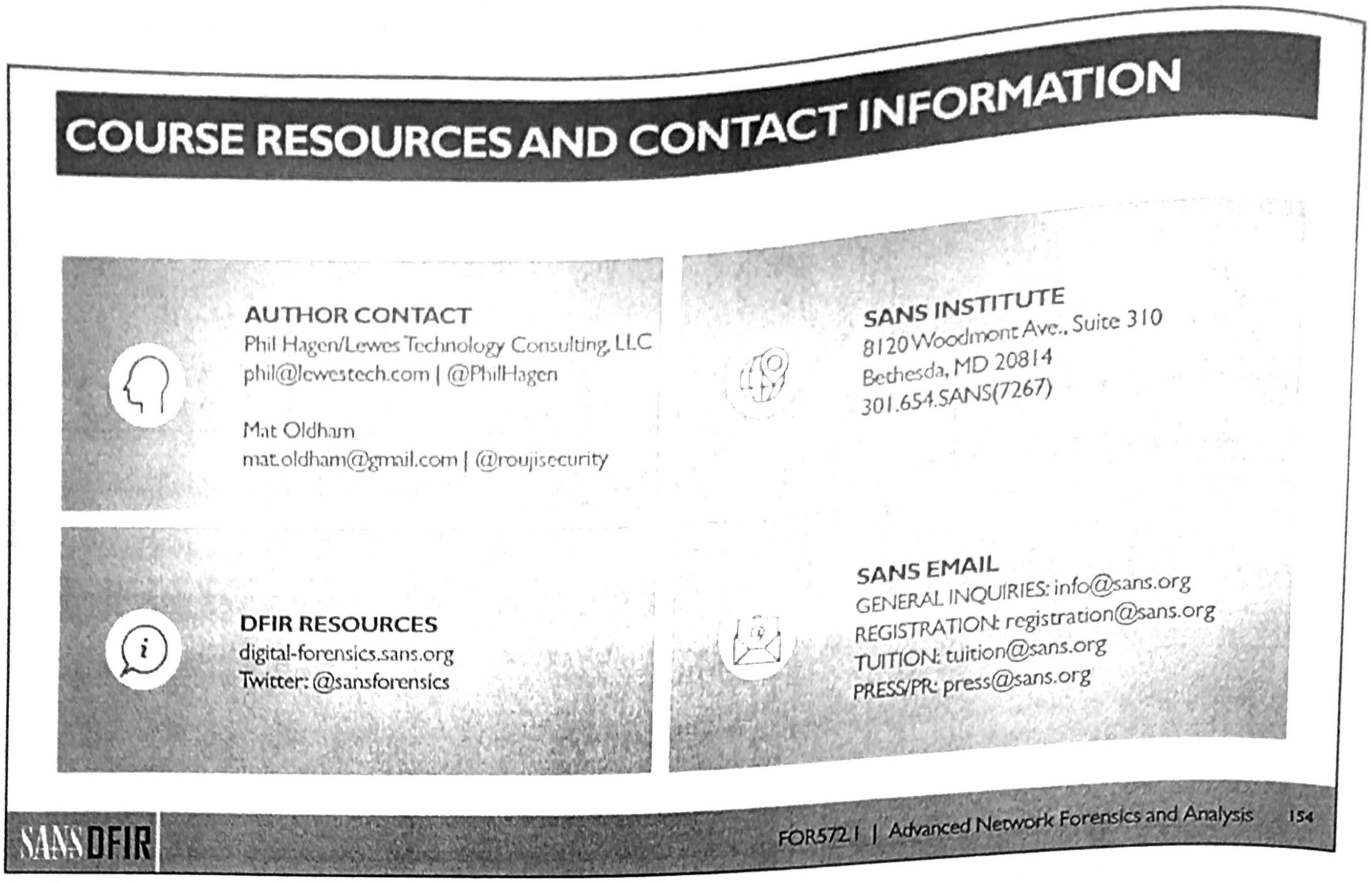
This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

Off the Disk and Onto the Wire

© 2017 Lewes Technology Consulting, LLC and Mat Oldham | All Rights Reserved | Version # FOR572_C01_01

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



This page intentionally left blank.

© 2017 Lewes Technology Consulting, LLC, Mat Oldham

"As usual, SANS courses pay for themselves by Day 2. By Day 3, you are itching to get back to the office to use what you've learned."
Ken Evans, Hewlett Packard Enterprise - Digital Investigation Services

SANS Programs
sans.org/programs

GIAC Certifications
Graduate Degree Programs
NetWars & CyberCity Ranges
Cyber Guardian
Security Awareness Training
CyberTalent Management
Group/Enterprise Purchase Arrangements
DoDD 8140
Community of Interest for NetSec
Cybersecurity Innovation Awards

Search SANSInstitute

SANS Free Resources
sans.org/security-resources

- E-Newsletters
  NewsBites: Bi-weekly digest of top news
  OUCH!: Monthly security awareness newsletter
  @RISK: Weekly summary of threats & mitigations
- Internet Storm Center
- CIS Critical Security Controls
- Blogs
- Security Posters
- Webcasts
- InfoSec Reading Room
- Top 25 Software Errors
- Security Policies
- Intrusion Detection FAQ
- Tip of the Day
- 20 Coolest Careers
- Security Glossary

SANS Institute
8120 Woodmont Avenue | Suite 310
Bethesda, MD 20814
301.654.SANS(7267)
info@sans.org