SEC530 | DEFENSIBLE SECURITY ARCHITECTURE AND ENGINEERING

530.3

Network-Centric Security

SANS

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

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 SANS INSTITUTE FOR THE COURSEWARE. YOU AGREE THAT THIS AGREEMENT IS ENFORCEABLE LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY YOU.

With the CLA, 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 SANS Institute to 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 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 SANS INSTITUTE, AND THAT 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 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 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 SANS Institute.

If any provision of this CLA is declared unenforceable in any jurisdiction, then such provision shall be deemed to be severable from this CLA and shall not affect the remainder thereof. An amendment or addendum to this CLA may accompany this Courseware.

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

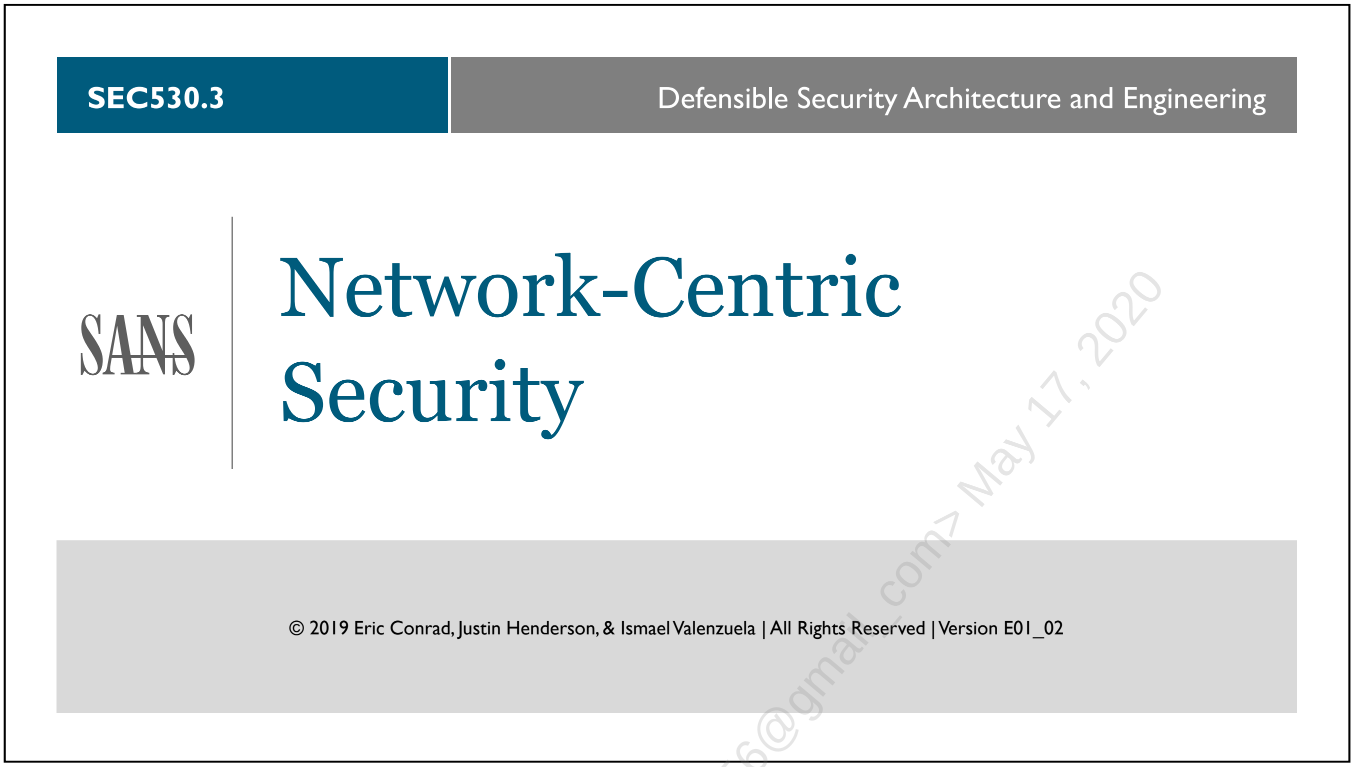
AirDrop, AirPort, AirPort Time Capsule, Apple, Apple Remote Desktop, Apple TV, App Nap, Back to My Mac, Boot Camp, Cocoa, FaceTime, FileVault, Finder, FireWire, FireWire logo, iCal, iChat, iLife, iMac, iMessage, iPad, iPad Air, iPad Mini, iPhone, iPhoto, iPod, iPod classic, iPod shuffle, iPod nano, iPod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

PMP and PMBOK are registered marks of PMI.

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

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

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



SEC530.3

Defensible Security Architecture and Engineering

Network-Centric Security

© 2019 Eric Conrad, Justin Henderson, & Ismael Valenzuela | All Rights Reserved | Version E01_02

Welcome to SEC530.3, Network-Centric Security!

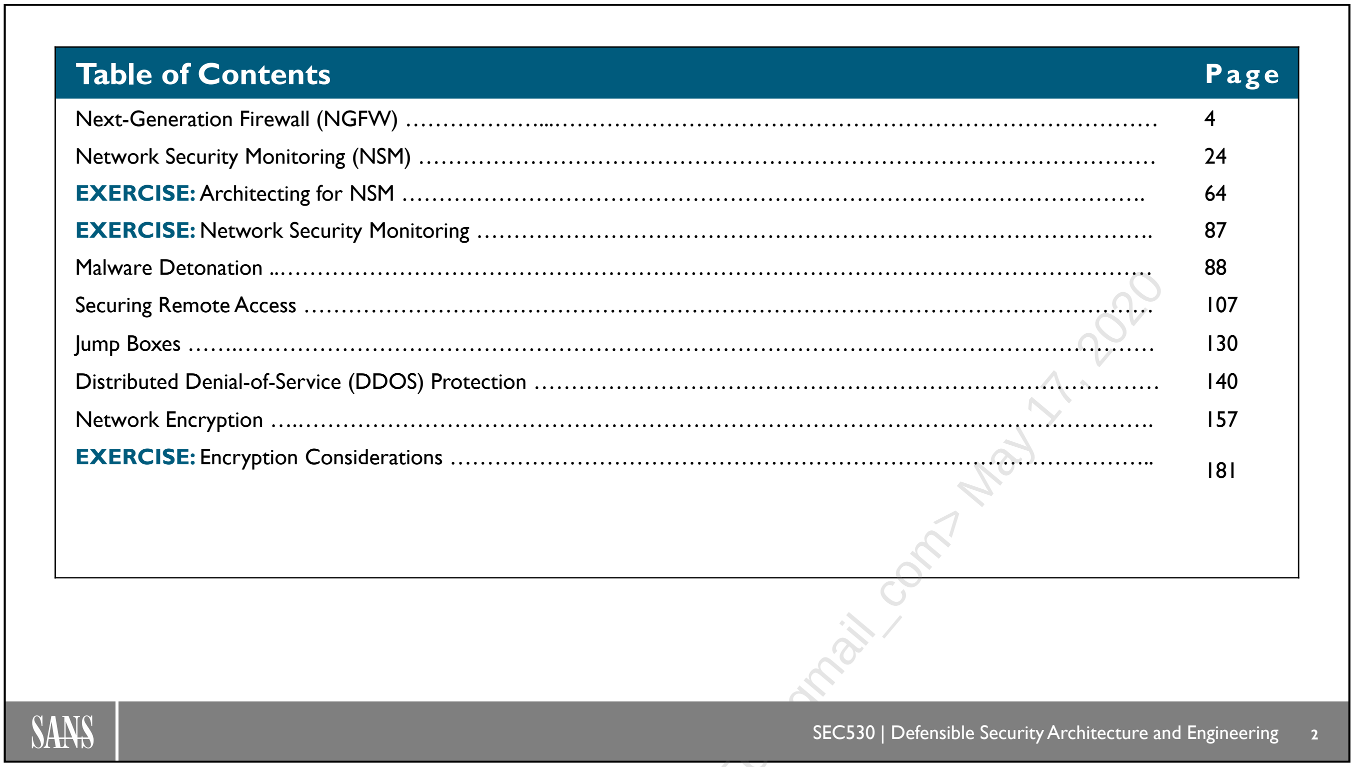


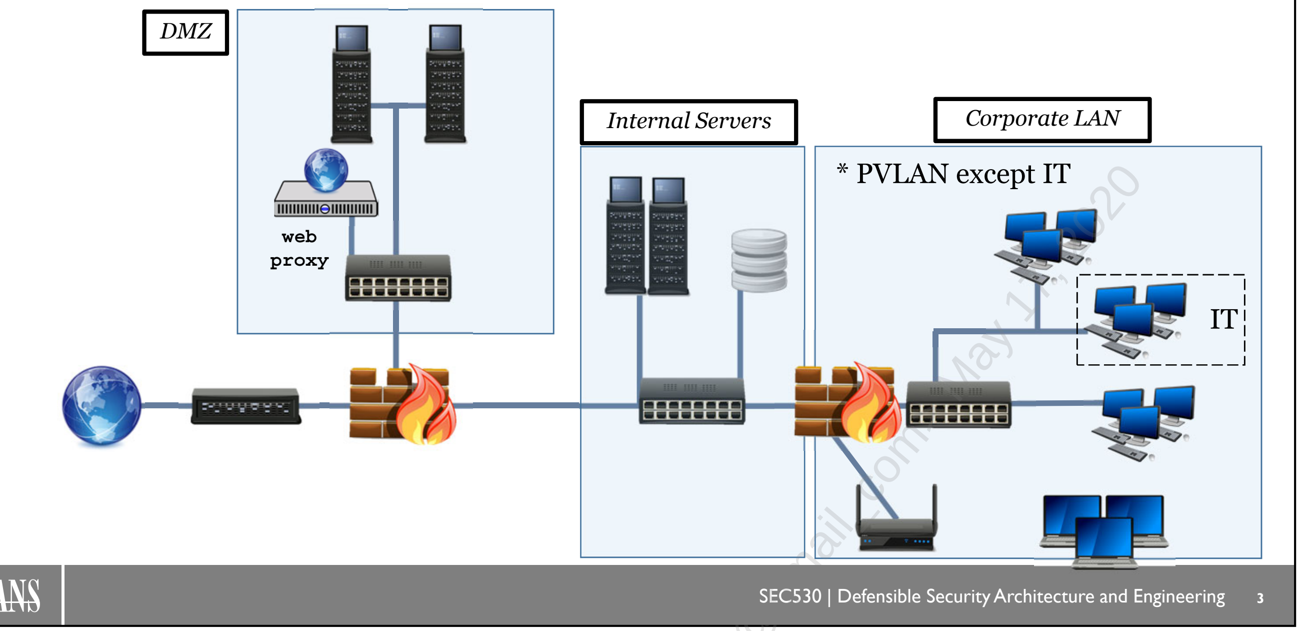
Table of Contents

Page

530.3 Table of Contents

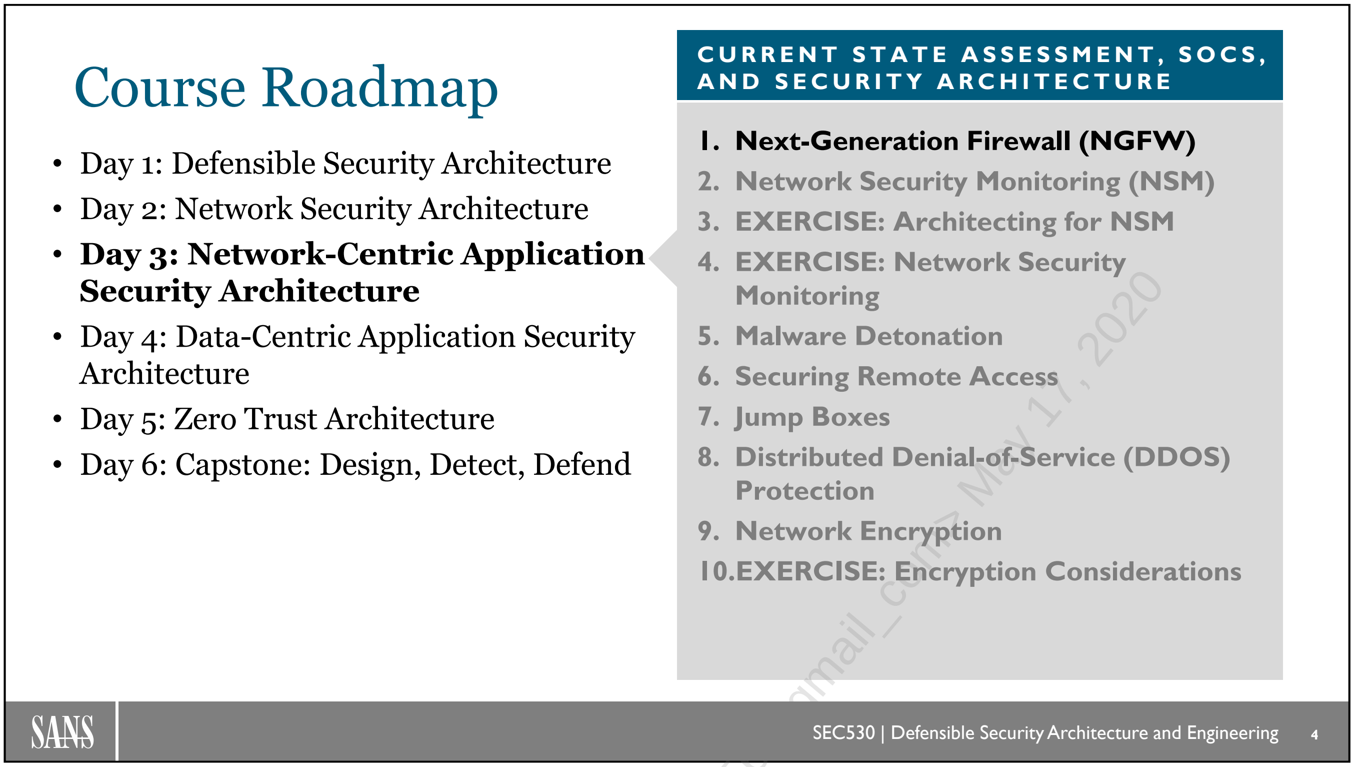
This table of contents outlines our plan for 530.3.

Case Study: Tyrrell Corporation



Case Study: Tyrrell Corporation

This diagram reflects the network architecture based on book two's ending. In book three you will be building upon this design.



Course Roadmap

- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

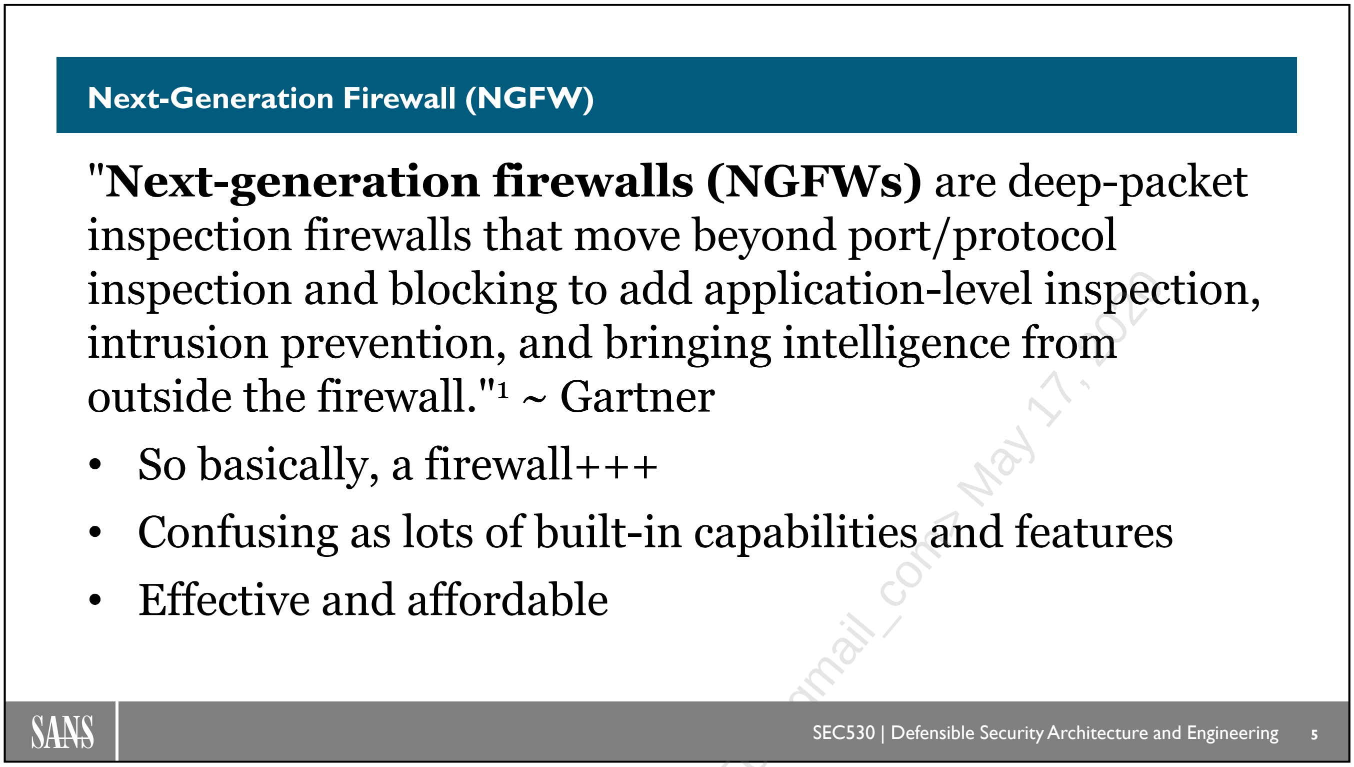
1. **Next-Generation Firewall (NGFW)**
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

SANS

SEC530 | Defensible Security Architecture and Engineering 4

Course Roadmap

The next section covers Next-Generation Firewalls (NGFWs).



Next-Generation Firewall (NGFW)

"**Next-generation firewalls (NGFWs)** are deep-packet inspection firewalls that move beyond port/protocol inspection and blocking to add application-level inspection, intrusion prevention, and bringing intelligence from outside the firewall."[1] ~ Gartner

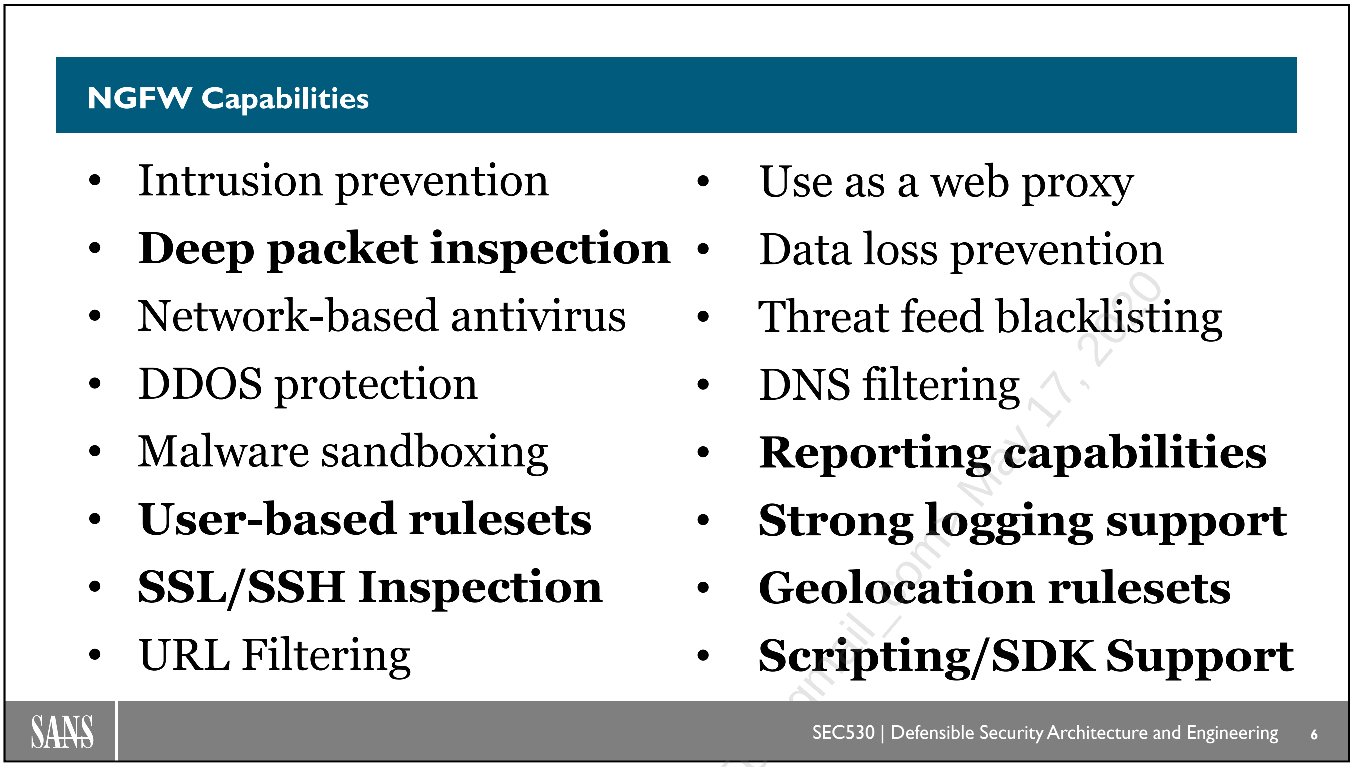
- So basically, a firewall+++
- Confusing as lots of built-in capabilities and features
- Effective and affordable

Next-Generation Firewall (NGFW)

Firewalls have come a long way since their initial creation. Compared to traditional layer four firewalls Next-Generation Firewalls operate at layer 7. By operating at layer seven modern firewalls can inspect the network payload of new or existing connections allowing significant granularity of what is or is not allowed.

Layer seven visibility allows firewalls to filter on application-level details. The confusion is how application-level analysis is applied. Modern firewalls come with multiple services that analyze network payloads but how each operates varies.

[1] https://www.gartner.com/it-glossary/next-generation-firewalls-ngfws



NGFW Capabilities

- Intrusion prevention
- **Deep packet inspection**
- Network-based antivirus
- DDOS protection
- Malware sandboxing
- **User-based rulesets**
- **SSL/SSH Inspection**
- URL Filtering
- Use as a web proxy
- Data loss prevention
- Threat feed blacklisting
- DNS filtering
- **Reporting capabilities**
- **Strong logging support**
- **Geolocation rulesets**
- **Scripting/SDK Support**

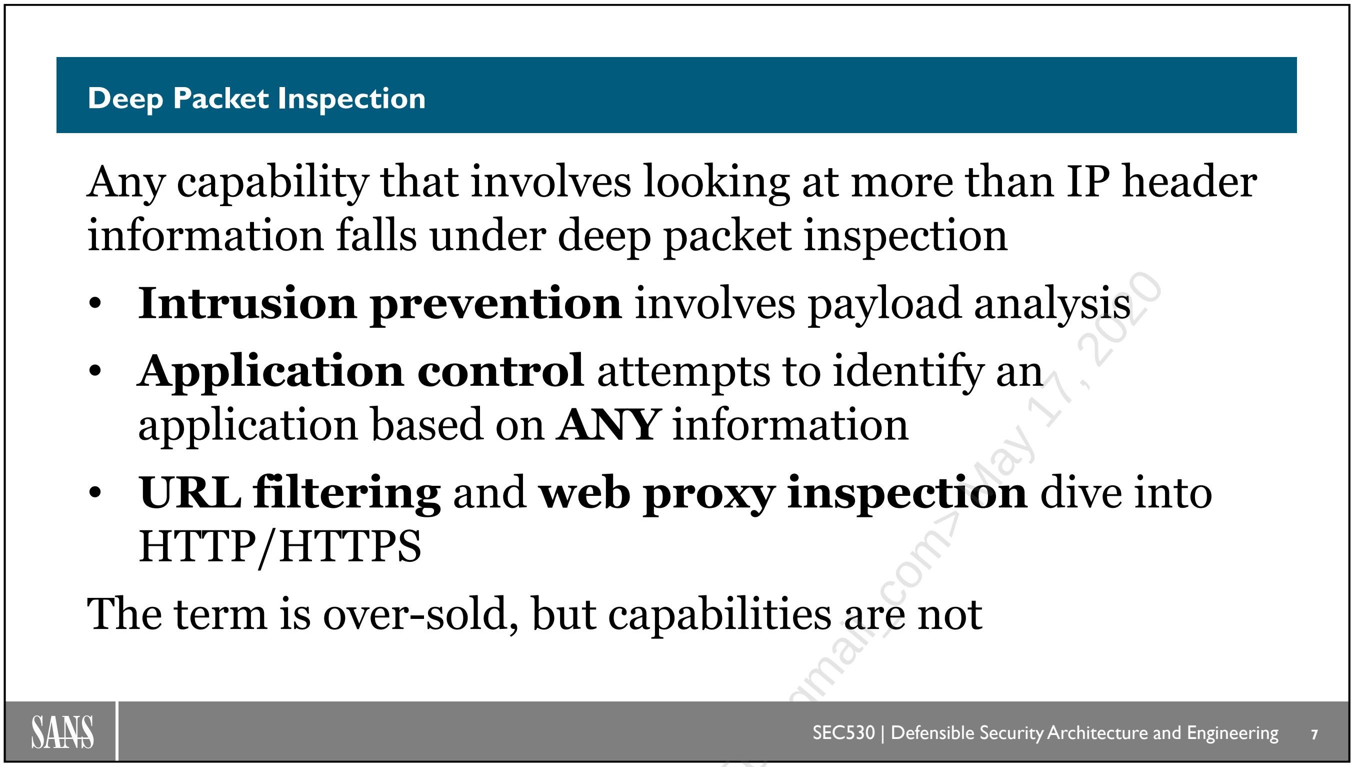
NGFW Capabilities

Some of the more common layer seven filtering capabilities of a next-generation firewall include intrusion prevention, network-based antivirus, and URL filtering. However, new security services come out on a routine basis. For example, SSL Inspection allows intentional man-in-the-middle encryption and decryption of traffic. Malware sandboxing allows you to take files and analyze them in an automated virtual machine environment.

Along with tremendous security services, next-generation firewalls may include advanced reporting, troubleshooting capabilities, and tight integration with user authentication systems such as Active Directory. Authentication integration alone can severely enhance firewall rulesets by limiting access by users or user groups. Think about it; even a traditional outbound rule can limit what ports can be used to a specific security group.

[1] https://www.paloaltonetworks.com/resources/datasheets/product-summary-specsheet

[2] https://www.fortinet.com/content/dam/fortinet/assets/data-sheets/FortiGate_FortiWiFi_50E_Series.pdf



Deep Packet Inspection

Any capability that involves looking at more than IP header information falls under deep packet inspection

- **Intrusion prevention** involves payload analysis
- **Application control** attempts to identify an application based on **ANY** information
- **URL filtering** and **web proxy inspection** dive into HTTP/HTTPS

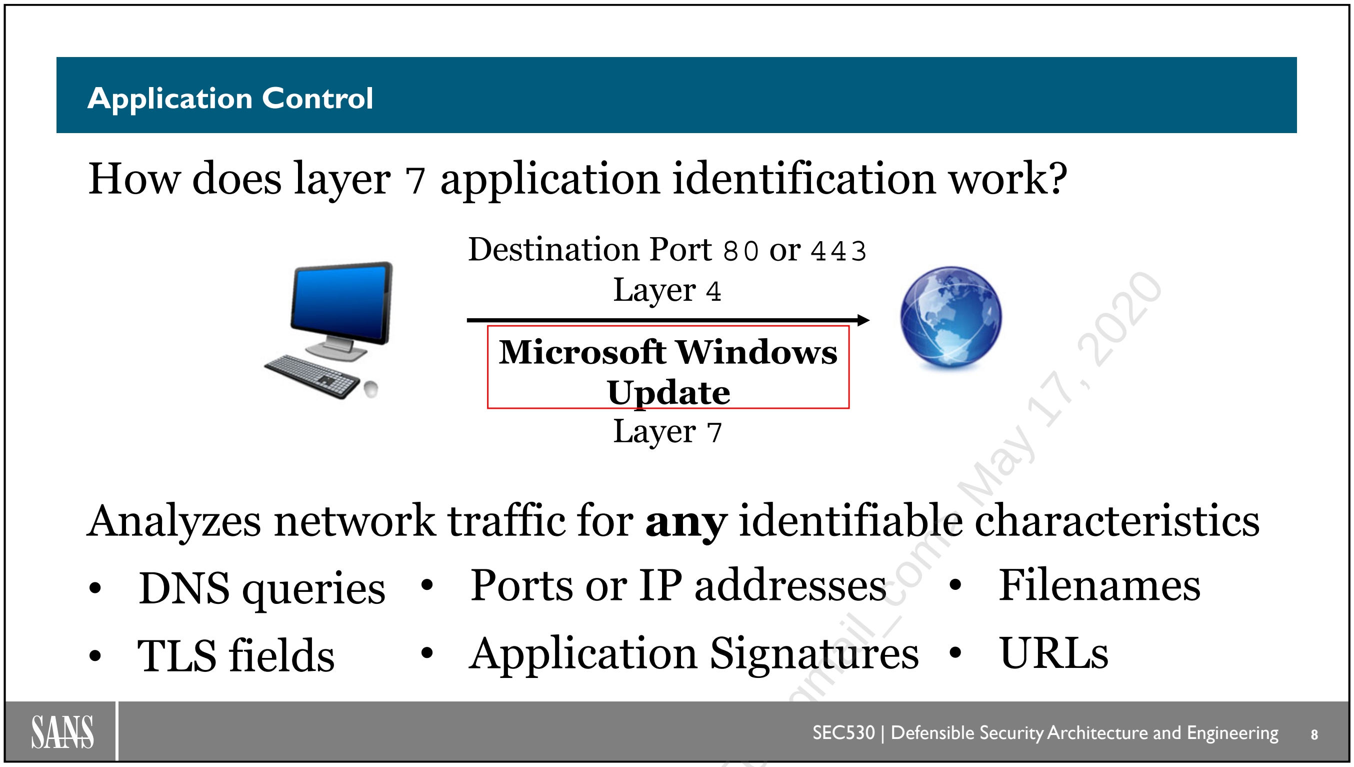
The term is over-sold, but capabilities are not

SANS SEC530 | Defensible Security Architecture and Engineering 7

Deep Packet Inspection

Deep Packet Inspection is an overused term often without proper context. A simple definition is the capability to look at header and payload information. Traditional firewalls can only look at layer 3 and 4 information in the packet header but cannot read layer 7 contents. Modern firewalls can examine the entire contents of the packet to make filtering decisions.

The form of payload analysis is what separates one security feature from another. For example, an Intrusion Prevention service applies signatures to the payload of the packet to identify malicious activity. URL filtering analyzes HTTP specific fields to identify hostnames, MIME types, and other web related data. Application control analyzes payload data to identify common patterns or use cases to identify which application is in use.



Application Control

How does layer 7 application identification work?

Destination Port 80 or 443
Layer 4

Microsoft Windows Update
Layer 7

Analyzes network traffic for **any** identifiable characteristics

- DNS queries
- TLS fields
- Ports or IP addresses
- Application Signatures
- Filenames
- URLs

Application Control

One of the most mysterious services of next-generation firewalls is application control. One of the most mysterious services of next-generation firewalls is application control. Although it may seem to be a black box, it is not that complicated.

Under the hood, applications get identified by analyzing various characteristics such as DNS names, TLS field data, ports and IP addresses, and application-specific patterns. The singular use or combination of these characteristics is what identifies a given application.

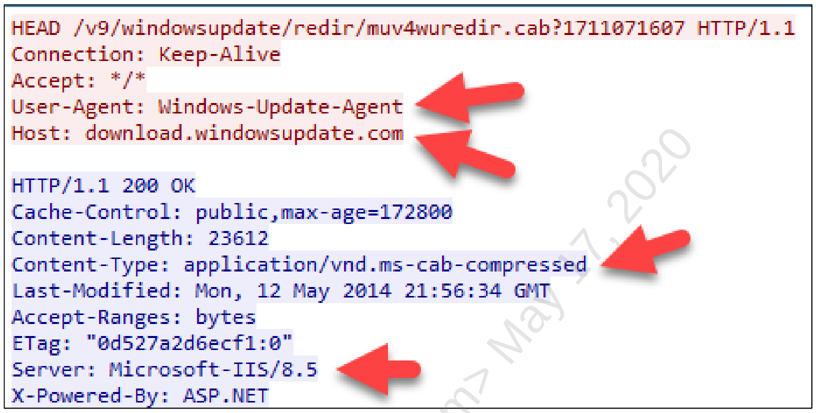
[1] https://www.paloaltonetworks.com/resources/techbriefs/app-id-tech-brief

Windows Update Example

Multiple means to identify Windows Updates

- DNS
- HTTP
- HTTPS

Not all are needed



```
HEAD /v9/windowsupdate/redir/muv4wuredir.cab?1711071607 HTTP/1.1
Connection: Keep-Alive
Accept: */*
User-Agent: Windows-Update-Agent
Host: download.windowsupdate.com

HTTP/1.1 200 OK
Cache-Control: public,max-age=172800
Content-Length: 23612
Content-Type: application/vnd.ms-cab-compressed
Last-Modified: Mon, 12 May 2014 21:56:34 GMT
Accept-Ranges: bytes
ETag: "0d527a2d6ecf1:0"
Server: Microsoft-IIS/8.5
X-Powered-By: ASP.NET
```

HTTP Host Name: download.windowsupdate.com

SSL/TLS Server Name: www.update.microsoft.com

Windows Update Example

This slide shows a request from a Windows 7 client checking for updates over Windows Update. The slide image includes the HTTP information for this request. While application control signatures vary from vendor to vendor, it is simple to see how the application for Windows Update may function.

For example, the DNS domain and HTTP host field in this example use download.windowsupdate.com. On top of that, the request is for a .cab file type and has an HTTP User-Agent of Windows-Update-Agent. Combining these characteristics may be enough to identify the application Windows Update is in use. However, more characteristics are available if necessary. For example, the server is using IIS and is using the ASP.NET server-side language while delivering a Content-Type of application/vnd.ms-cab-compressed.

Because application identification may require multiple characteristics to achieve high fidelity there can be some concerns from a security standpoint. Multiple network packets can get authorized because enough details need to be analyzed to identify an application. Allowing multiple packets to identify an application means data can accidentally be allowed in or out of a network. If multiple packets are allowed, an attacker can exfiltrate data by sending data over a few packets at a time and then rotating to a new network session.

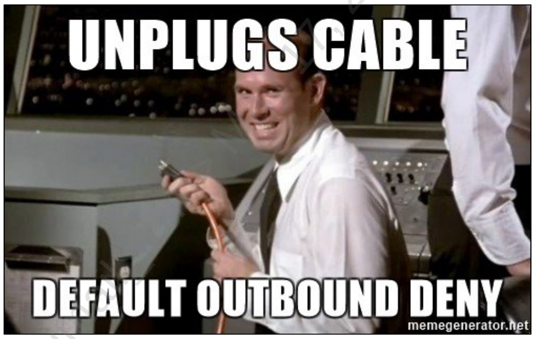
NGFW Rulesets

NGFW should allow all authorized connections

- Allow authorized ports, applications, and countries
- Achieve fine-grain control by allowing by subnet or user
- And deny everything else

This is not a simple task

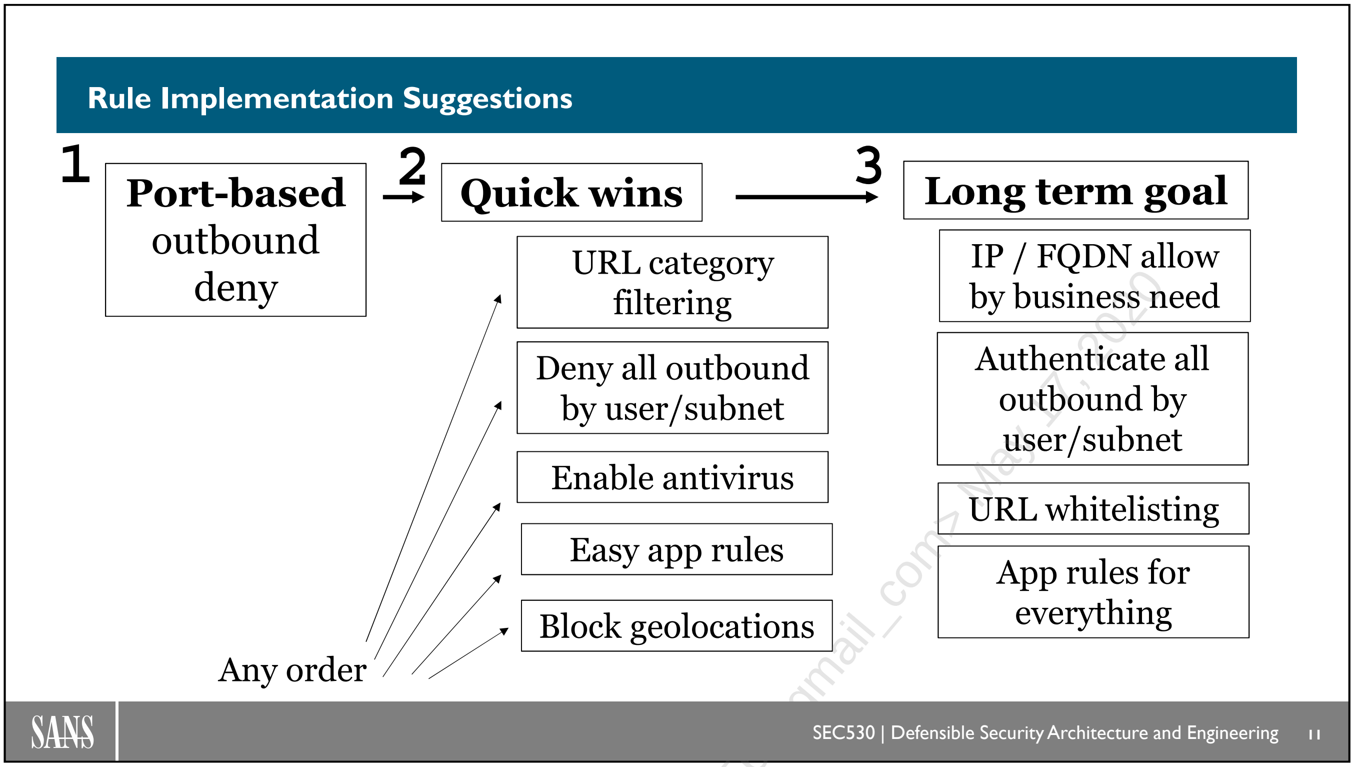
- Many do not have a default outbound deny for ports
- Where to begin?



NGFW Rulesets

Both a traditional and next-generation firewall's goal is only to allow authorized connections. Allowing only authorized connections on a traditional firewall is difficult. The difficulty of whitelisting authorized connections is significantly harder for a next-generation firewall because of all the granular control and added security services. Also, vendor implementation and administration of security services are different from one vendor to another.

The next couple slides recommend possible methods on getting to authorized only connections on a next-generation firewall.



Rule Implementation Suggestions

1 **Port-based** outbound deny

2 **Quick wins**

- URL category filtering
- Deny all outbound by user/subnet
- Enable antivirus
- Easy app rules
- Block geolocations

Any order

3 **Long term goal**

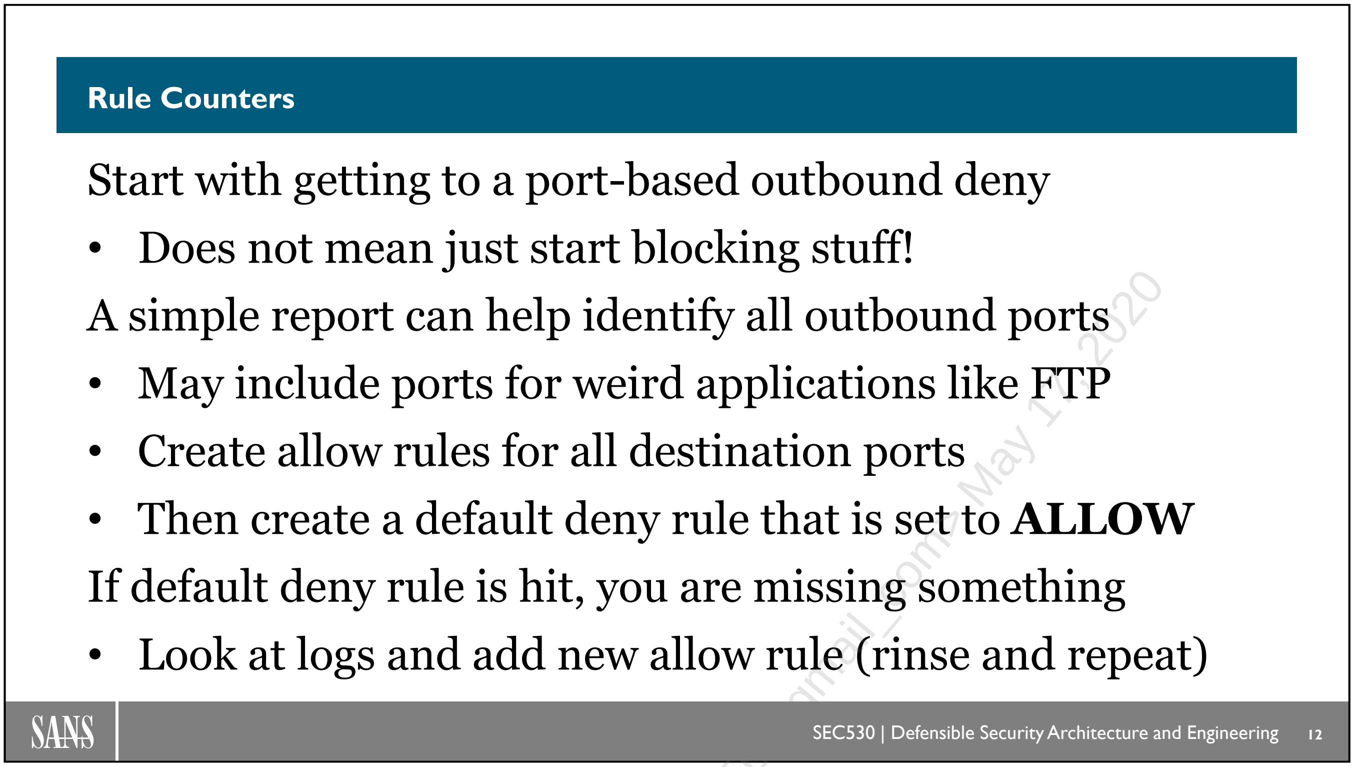
- IP / FQDN allow by business need
- Authenticate all outbound by user/subnet
- URL whitelisting
- App rules for everything

Rule Implementation Suggestions

Locking down ports is a great first step for next-generation firewalls. When packets can be quickly dropped based on the port number instead of payload contents the volume of traffic that requires more CPU-intensive layer 7 checks is minimized. Why perform antivirus checks or application identification on ports that are not necessary?

After locking down ports, it is a good idea to focus on quick win implementations. Quick wins include high-level URL category filters, denying outbound access to subnets or users that do not need internet access, as well as a few other quick wins. The goal is to increase security without lots of manual labor rapidly.

After establishing hygiene and quick wins, organizations should weigh the security benefits of locking down rules more granularly vs organizational maturing. Organizational security posture varies from organization to organization, so some techniques are not practical for everyone.



Rule Counters

Start with getting to a port-based outbound deny

- Does not mean just start blocking stuff!

A simple report can help identify all outbound ports

- May include ports for weird applications like FTP
- Create allow rules for all destination ports
- Then create a default deny rule that is set to **ALLOW**

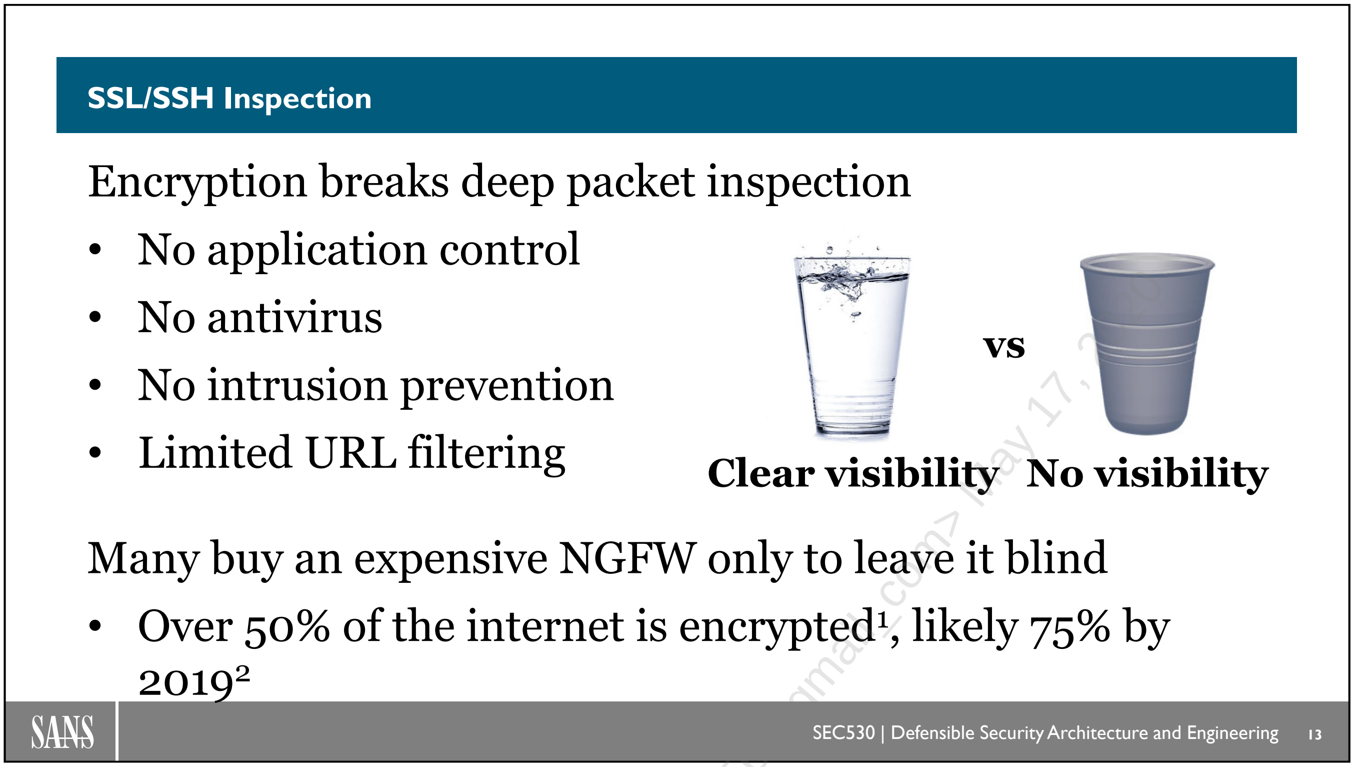
If default deny rule is hit, you are missing something

- Look at logs and add new allow rule (rinse and repeat)

Rule Counters

Using firewall rule counters is important. A rule counter increments each time a firewall rule has a match. Effectively, rule counters provide a simple reporting mechanism to identify if a rule triggers.

As a defender, rule counters are critical for monitoring and locking down rule sets. Initially, a rule can be put in place as an allow rule. If the allow rule triggers, then traffic is occurring that may be unauthorized or malicious. If the rule does not trigger over a set period, then it can be flipped to deny.



SSL/SSH Inspection

Encryption breaks deep packet inspection

- No application control
- No antivirus
- No intrusion prevention
- Limited URL filtering

Many buy an expensive NGFW only to leave it blind

- Over 50% of the internet is encrypted[1], likely 75% by 2019[2]

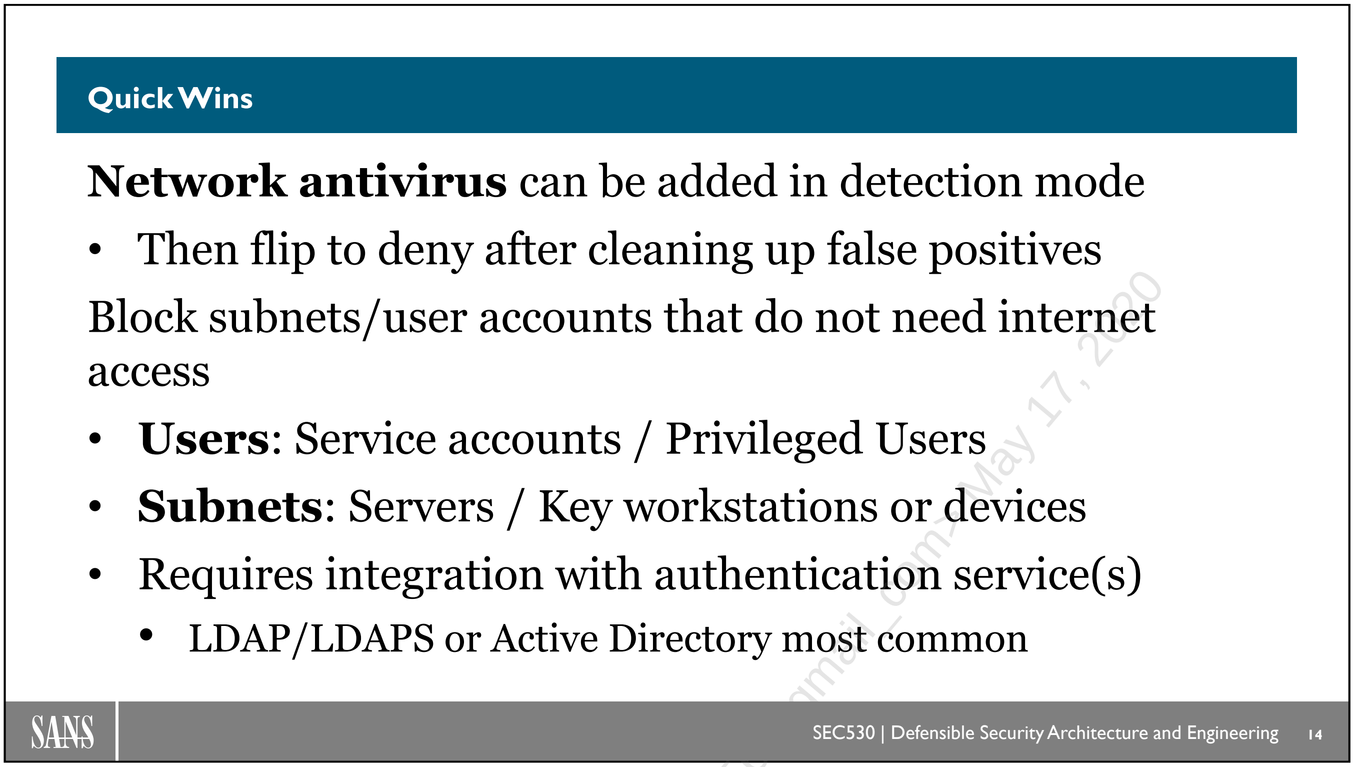
SSL/SSH Inspection

Before moving on to deep packet inspection rules, consider enabling SSL/SSH inspection. At this point over 50% of the internet uses encryption[1]. By 2019 it is estimated that 75% of the internet will utilize encryption[2]. Without SSL inspection a next-generation firewall is unable to perform deep packet inspection web traffic that uses SSL or TLS encryption.

Put a different way, a next-generation firewall without SSL Inspection is 50% less effective. To make up for this modern next-generation firewalls will perform SSL certificate inspection. While this is better than nothing, it still leaves you blind to the packet payload when compared to SSL inspection. Politics and international privacy laws may prevent enabling SSL Inspection, but it is worth having the discussions and trying to enable it. Without SSL Inspection, application control, antivirus, intrusion prevention, and many other security services are considerably less effective.

[1] https://www.eff.org/deeplinks/2017/02/were-halfway-encrypting-entire-web

[2] https://www.nsslabs.com/company/news/press-releases/nss-labs-predicts-75-of-web-traffic-will-be-encrypted-by-2019/



Quick Wins

Network antivirus can be added in detection mode

- Then flip to deny after cleaning up false positives

Block subnets/user accounts that do not need internet access

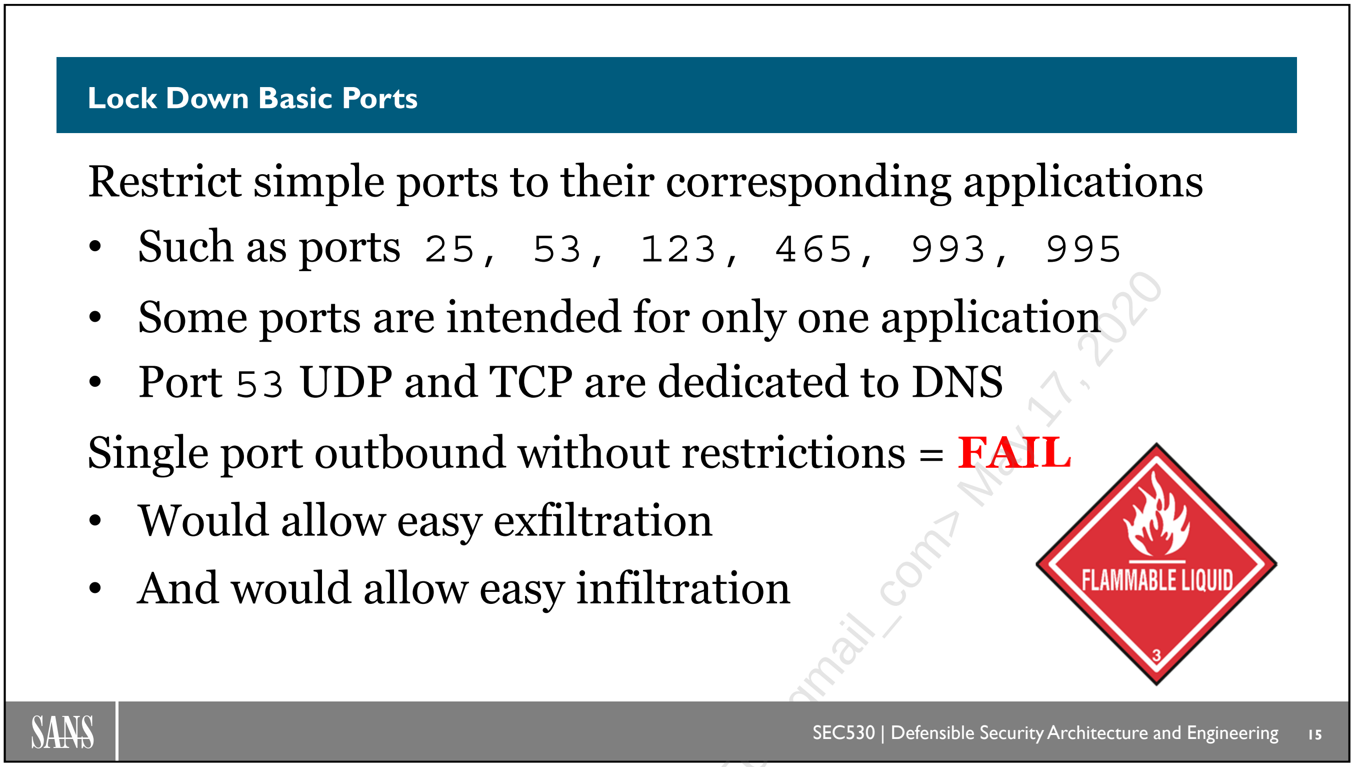
- **Users**: Service accounts / Privileged Users
- **Subnets**: Servers / Key workstations or devices
- Requires integration with authentication service(s)
  - LDAP/LDAPS or Active Directory most common

SANS SEC530 | Defensible Security Architecture and Engineering 14

Quick Wins

Antivirus is a blacklisting solution that blocks known bad files. Because it is blocks known bad files it has a low false positive rate. Therefore, enabling antivirus is a quick win. However, one should consider enabling antivirus in the detect only mode before blocking mode.

Another quick win is composing a list of privileged user accounts and service accounts that have no business accessing the internet. Adds these users and groups to your firewall and block internet access for them. Most next-generation firewalls support integration with authentication services such as Active Directory or LDAP. When integrating with authentication services try to reduce the account privilege to the minimum possible. Vendors may request domain administrator rights but under the hood may only need standard user access. Also, consider blocking internet access to key subnets such as critical server assets. If internet access is not required, then internet access should be blocked.



Lock Down Basic Ports

Restrict simple ports to their corresponding applications

- Such as ports `25, 53, 123, 465, 993, 995`
- Some ports are intended for only one application
- Port `53` UDP and TCP are dedicated to DNS

Single port outbound without restrictions = **FAIL**

- Would allow easy exfiltration
- And would allow easy infiltration

FLAMMABLE LIQUID
3

SANS

SEC530 | Defensible Security Architecture and Engineering 15

Lock Down Basic Ports

Application control is a powerful capability but maintaining a whitelist of applications is difficult. One area organizations can experience a quick win is by applying application control to ports that are basic. For example, port 53 is often only for DNS. Port 25 is often dedicated to SMTP. Therefore, application control is simple to apply to both DNS and SMTP. Early application control should focus on more simple ports and applications such as these below:

Port	Application
25	SMTP
53	DNS
123	NTP
465	SMTP over TLS
993	Secure IMAP
995	Secure POP3

Rather than diving into ports 80 and 443, which have a tremendous amount of applications, we recommend aiming for some quick wins. This can be accomplished by starting application control on several basic applications and ports.

Geolocation Blocking

An organization may not have business in other countries

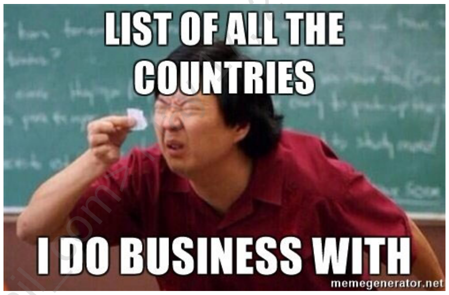
- Yet default settings allow connections to and from them

Traffic has legitimate reasons to be in certain countries

- Content delivery networks (CDN)
- Cloud hosting
- Business partners

Block specific countries

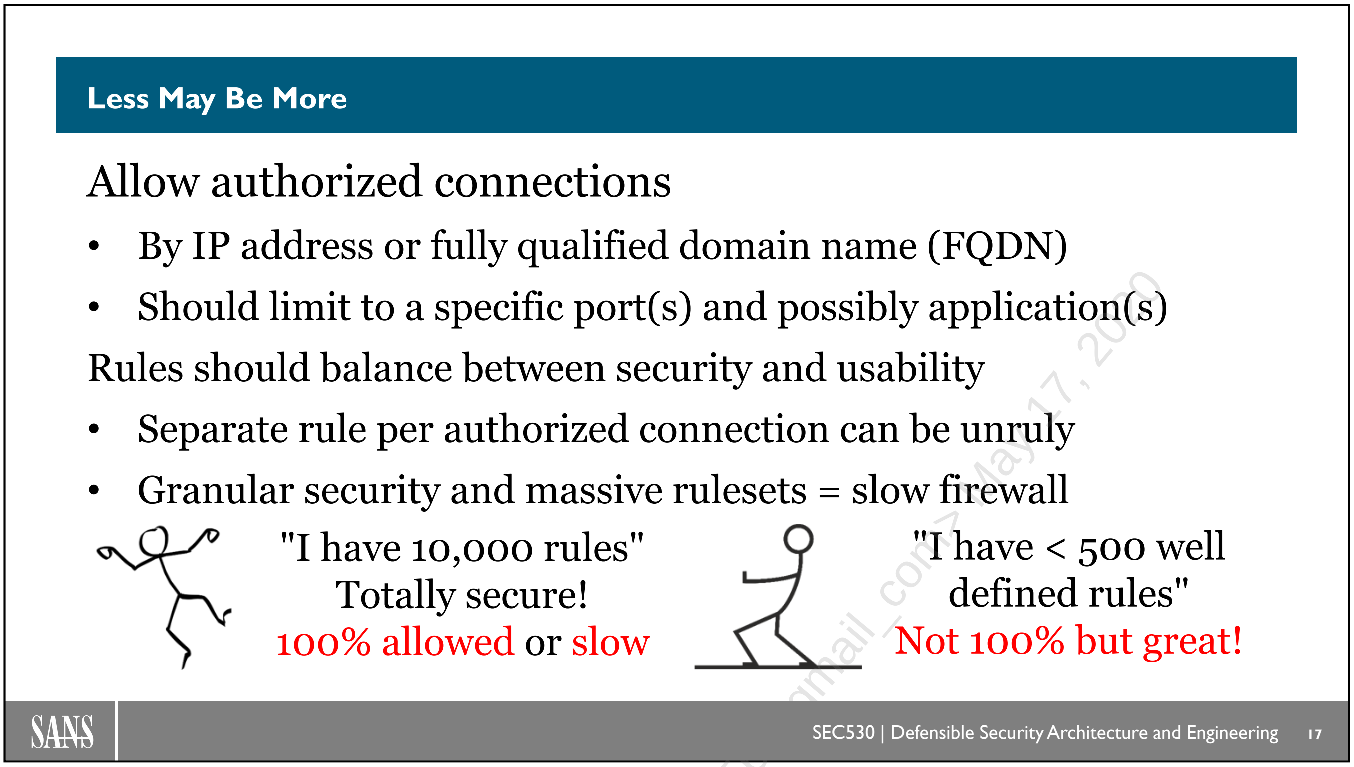
- Add FQDN or IP exceptions as necessary



Geolocation Blocking

If your organization does not do business with organizations in foreign countries, then it may be a good idea to block all connections to or from a given country. By blocking foreign countries, the attack surface of your organization shrinks. The main purpose is not to stop an advanced adversary but rather to minimize automated reconnaissance or attacks. Bypassing geolocation filtering is as simple as registering a virtual private server such as in Amazon and attacking from it.

The problem with blocking foreign countries is that certain websites are outside your country. Often these are from content delivery networks, one-off business partners, and cloud hosting. However, an entire country does not need authorization simply because of a few exceptions. Instead, authorize a specific website or IP address above the rule blocking the country. Now the site is allowed, but the entire country is not.



Less May Be More

Allow authorized connections

- By IP address or fully qualified domain name (FQDN)
- Should limit to a specific port(s) and possibly application(s)

Rules should balance between security and usability

- Separate rule per authorized connection can be unruly
- Granular security and massive rulesets = slow firewall

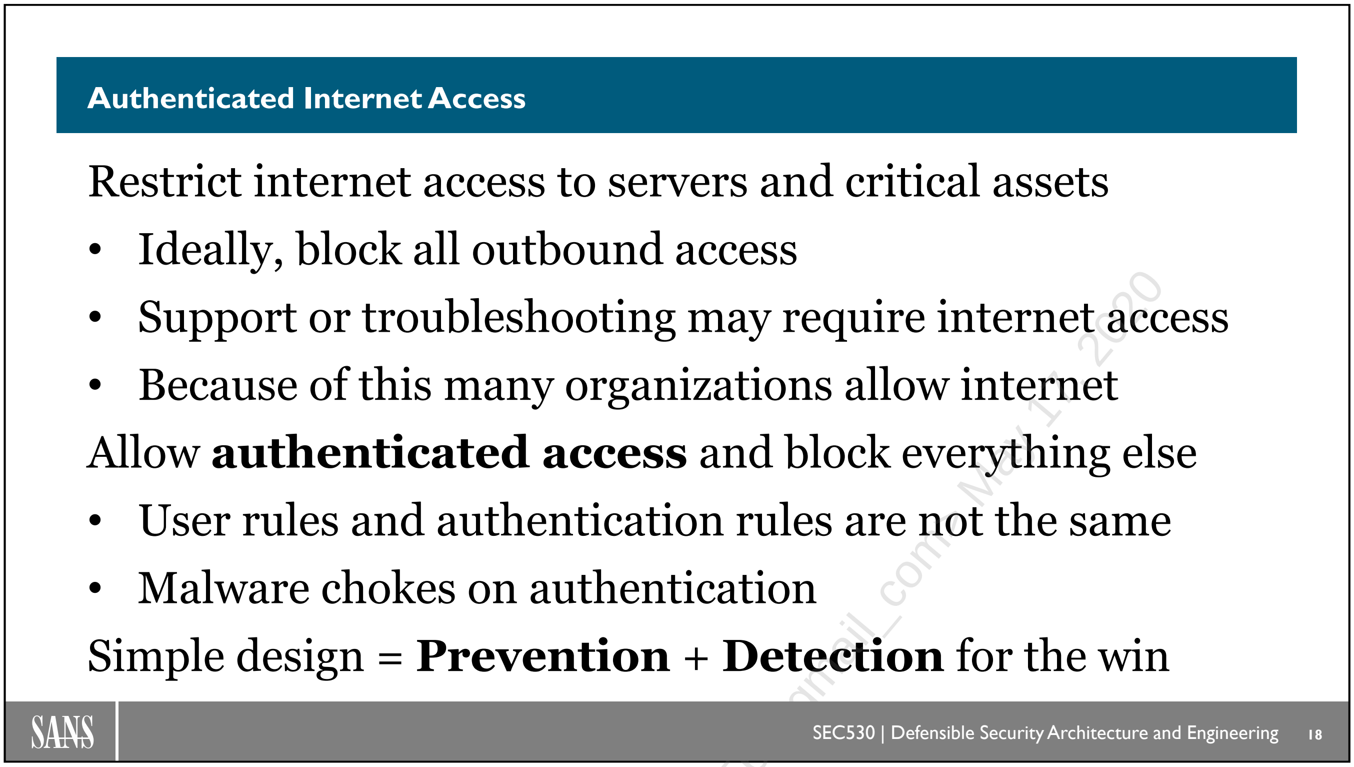
"I have 10,000 rules"
Totally secure!
100% allowed or slow

"I have < 500 well defined rules"
Not 100% but great!

Less May Be More

One needs to be careful when creating firewall rules. Some security professionals feel having a rule for every specific condition is more secure. However, the results of thousands of rules often surface as confusing, time-consuming, and less secure. A balance between the number of rules and security is required.

One method to balance security and management is grouping similar concepts together. For example, a rule called authorized business sites may place similar restrictions on authorized sites or business partners. These restrictions may include antivirus, URL filtering, and SSL inspection. Because the sites found in the rule are similar, it becomes easier to tune the rule.



Authenticated Internet Access

Restrict internet access to servers and critical assets

- Ideally, block all outbound access
- Support or troubleshooting may require internet access
- Because of this many organizations allow internet

Allow **authenticated access** and block everything else

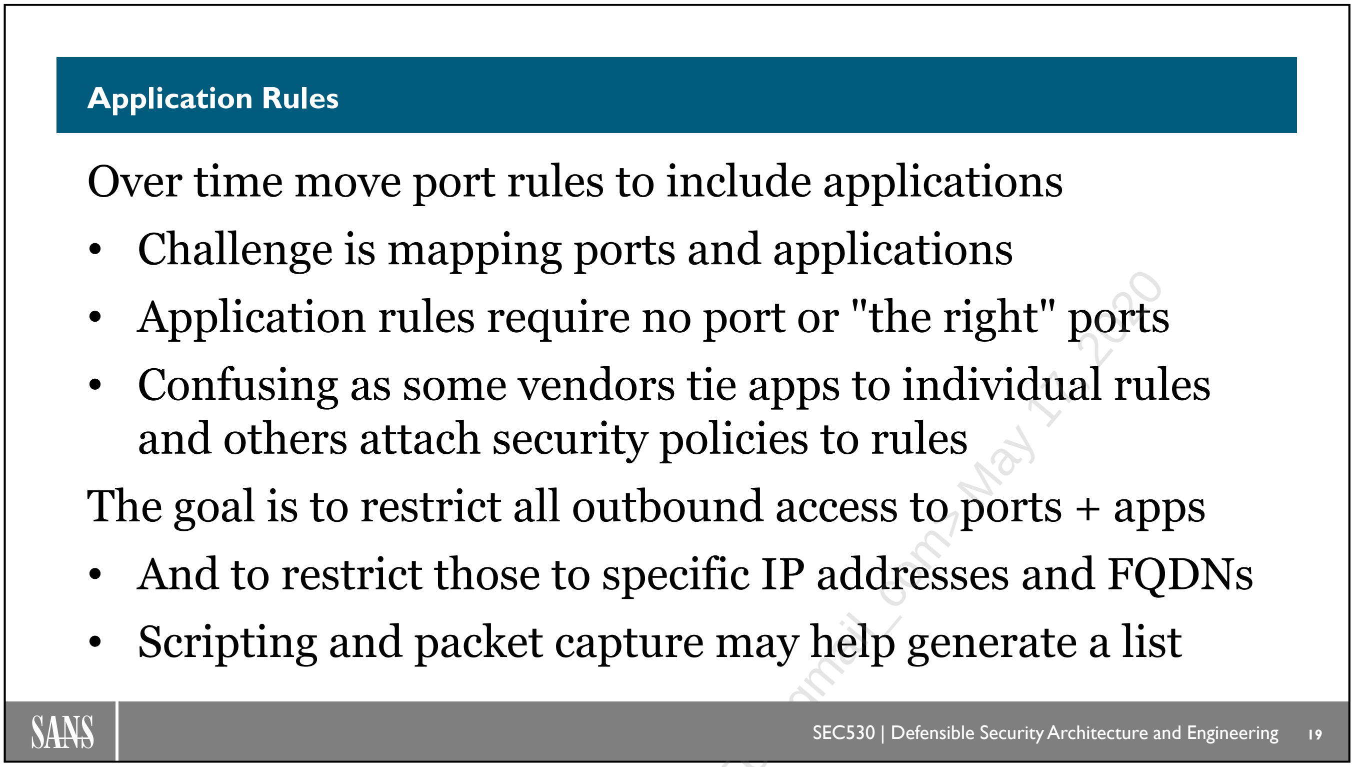
- User rules and authentication rules are not the same
- Malware chokes on authentication

Simple design = **Prevention** + **Detection** for the win

Authenticated Internet Access

Limiting or eliminating outbound access to servers or key critical assets is tough for organizations. Often network restrictions are not put in place because of the difficulty it creates for supporting systems. However, it is possible to allow easy support access while implementing security controls that block malware actions. Authentication firewall rules, in particular, are powerful.

An authentication rule can prevent all internet access by default but generate an authentication popup when internet access is required. A system user simply authenticates and then internet access is authorized. Malware would be unaware of the popup or would likely be unaware of proper credentials to bypass this restriction. Thus, prevention and detection of unauthorized use are achieved.



Application Rules

Over time move port rules to include applications

- Challenge is mapping ports and applications
- Application rules require no port or "the right" ports
- Confusing as some vendors tie apps to individual rules and others attach security policies to rules

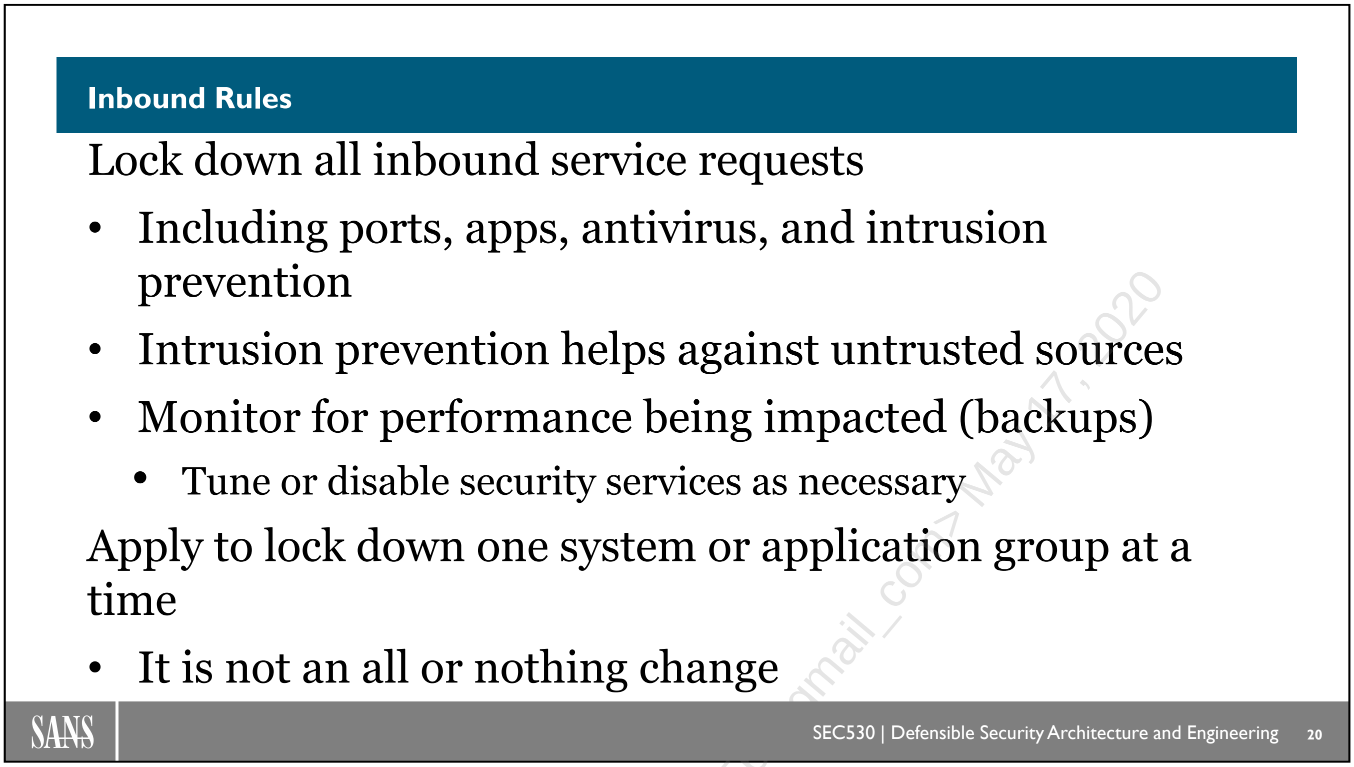
The goal is to restrict all outbound access to ports + apps

- And to restrict those to specific IP addresses and FQDNs
- Scripting and packet capture may help generate a list

Application Rules

Overtime firewall rules should include applications. Application control is especially important for outbound connections. A mature next-generation firewall only allows connections that are to authorized ports and applications. However, getting to a whitelist of ports and applications is time-consuming and difficult.

One method to minimize the effort of getting to full whitelisting is using centralized reporting such as through a commercial interface SIEM solution or scripting. These reporting tools allow for the generation of applications and ports that have exceptions. The task then becomes finding out which exceptions are authorized vs unauthorized. If an unauthorized connection is found but the system generating the connection cannot be changed, then a firewall rule should be set up that does not have logging enabled to hide the exception.



Inbound Rules

Lock down all inbound service requests

- Including ports, apps, antivirus, and intrusion prevention
- Intrusion prevention helps against untrusted sources
- Monitor for performance being impacted (backups)
  - Tune or disable security services as necessary

Apply to lock down one system or application group at a time

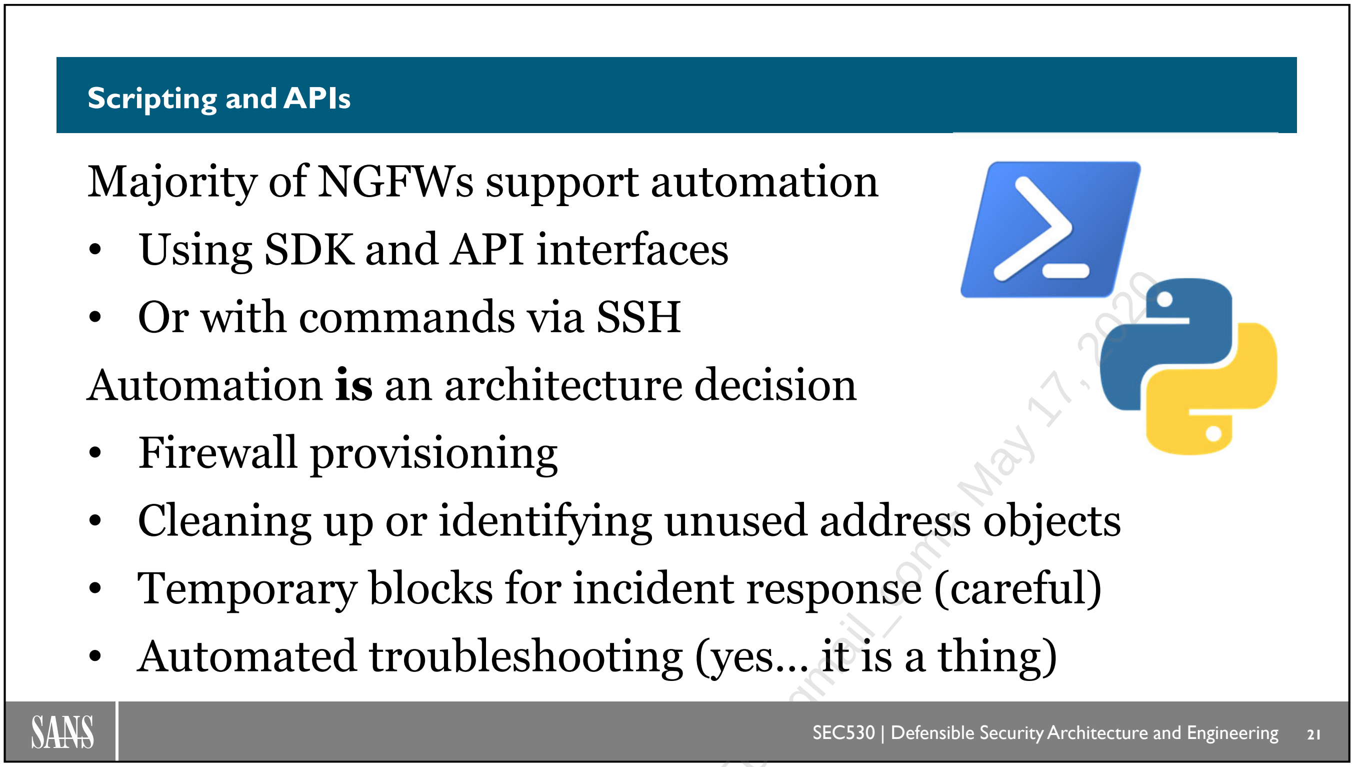
- It is not an all or nothing change

SANS SEC530 | Defensible Security Architecture and Engineering 20

Inbound Rules

Inbound connections should occur only over specific ports and applications. Specifically, each inbound connection is to a service your organization has. Application control and port rules tend to be easy to implement. However, network antivirus, intrusion prevention, and other services need consideration as these extra protection services can have dire performance consequences.

Some services need extra protection due to their sensitivity or use cases. For example, a web server is a commonly attacked surface. As such, a web server may benefit from intrusion prevention and other web-based security services. Other services may only need basic protection. For example, a backup server may not respond well to added security measures. Intrusion prevention may tank bandwidth and likely would not benefit an encrypted backup service anyway. A defender needs to understand the implications behind each service and evaluate what extra security services are necessary.



Scripting and APIs

Majority of NGFWs support automation

- Using SDK and API interfaces
- Or with commands via SSH

Automation **is** an architecture decision

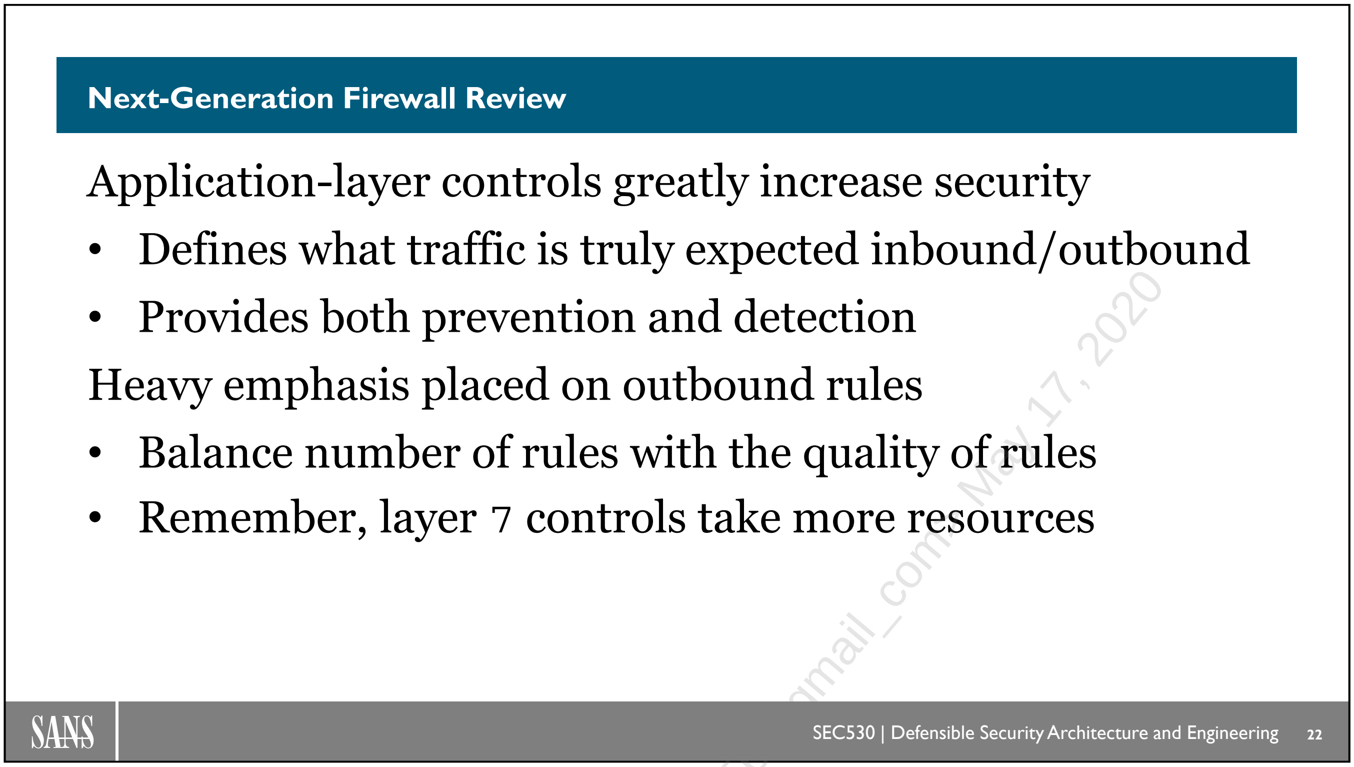
- Firewall provisioning
- Cleaning up or identifying unused address objects
- Temporary blocks for incident response (careful)
- Automated troubleshooting (yes… it is a thing)

SANS SEC530 | Defensible Security Architecture and Engineering 21

Scripting and APIs

Almost every next-generation firewall has under-the-hood APIs or SSH support. By having APIs and SSH capabilities, a next-generation firewall supports full automation. Scripting support means automatic firewall rule provisioning, automatic scripting response capabilities, and even help with troubleshooting firewall connections.

Automation is a key skill a defender needs in his or her toolbelt. Some of the most secure operations automate small tasks that free up significant amounts of time. Some organizations even use the API or SSH support to provision new firewalls or migrate between vendors.



Next-Generation Firewall Review

Application-layer controls greatly increase security

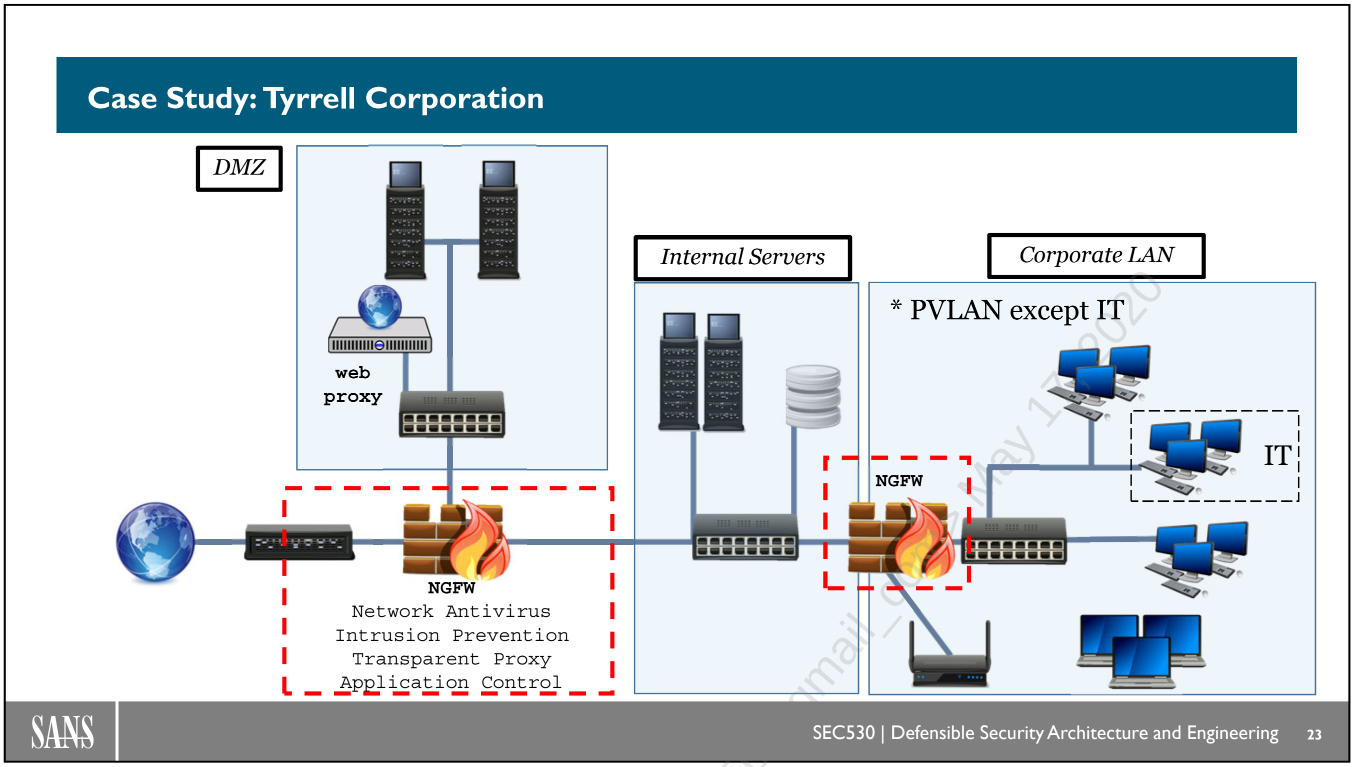
- Defines what traffic is truly expected inbound/outbound
- Provides both prevention and detection

Heavy emphasis placed on outbound rules

- Balance number of rules with the quality of rules
- Remember, layer 7 controls take more resources

Next-Generation Firewall Review

The difference between a traditional layer four firewall and next-generation firewalls is vast. Fortunately, next-generation firewalls are affordable and becoming a security staple. The trick is finding the right balance between quick win implementations and full whitelisting.



Case Study: Tyrrell Corporation

DMZ

web proxy

Internal Servers

Corporate LAN

* PVLAN except IT

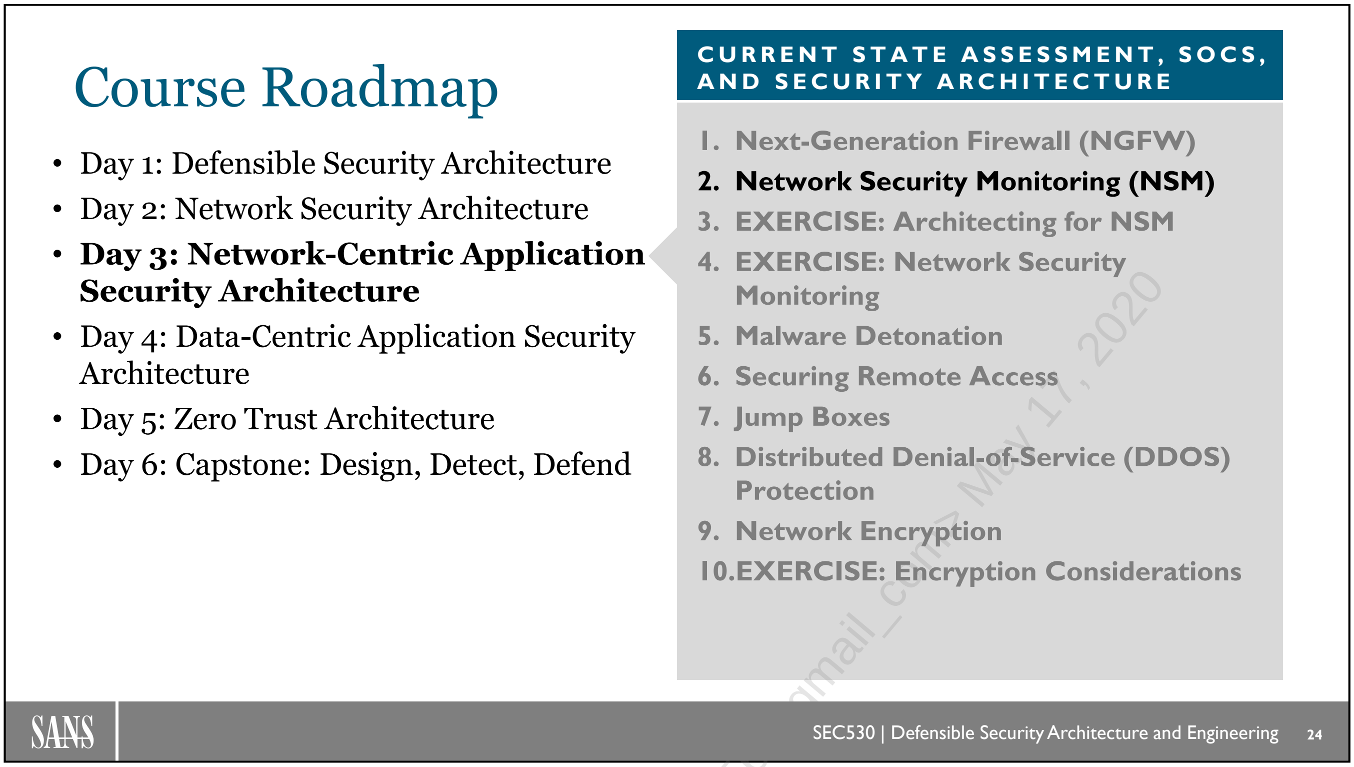
IT

NGFW

NGFW
Network Antivirus
Intrusion Prevention
Transparent Proxy
Application Control

Case Study: Tyrrell Corporation

This diagram shows an adjustment in that the firewall is now a Next-Generation Firewall. In this example, the edge firewall is a NGFW and so is the internal firewall.



Course Roadmap

- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

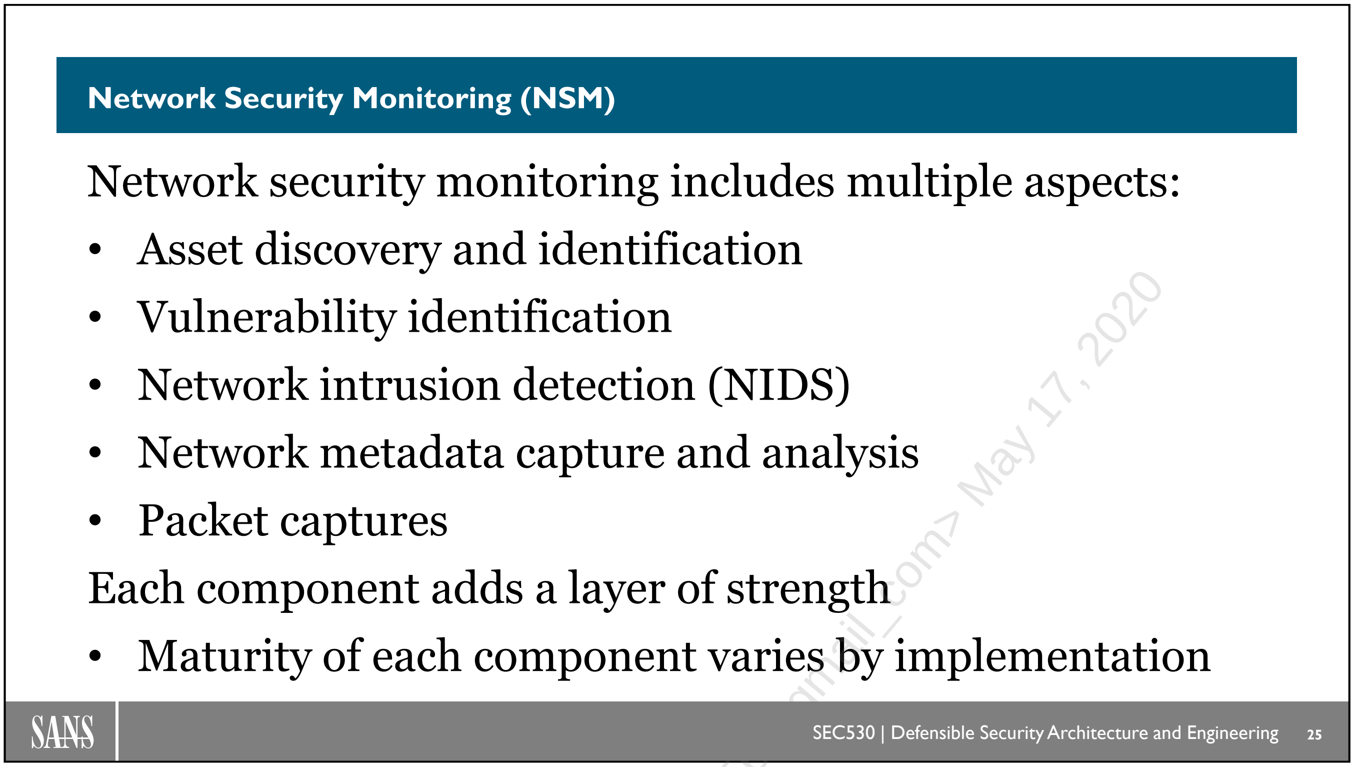
1. Next-Generation Firewall (NGFW)
2. **Network Security Monitoring (NSM)**
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

SANS

SEC530 | Defensible Security Architecture and Engineering 24

Course Roadmap

The next section covers Network Security Monitoring (NSM).



Network Security Monitoring (NSM)

Network security monitoring includes multiple aspects:

- Asset discovery and identification
- Vulnerability identification
- Network intrusion detection (NIDS)
- Network metadata capture and analysis
- Packet captures

Each component adds a layer of strength

- Maturity of each component varies by implementation

Network Security Monitoring (NSM)

Network security monitoring deals with data in motion and is more than intrusion detection. Network monitoring is now an evolution of network security monitoring or NSM. NSM comprises of related capabilities to provide a complete picture. For example, a single solution may generate alerts about a possible attack, but the solution also captures additional context around the alert. The context can include DNS and SSL logs and possibly full packet captures.

By switching from the traditional find evil mentality into context enrichment and data gathering, network security monitoring provides a complete toolset. NSM helps organizations learn more about their environments while providing capabilities to look for alerts and anomalies.

Alert Driven Workflows vs. Data Driven Workflows

- Most security operations teams live in an alert driven world
- Alerts provide only the **initial** point for an investigation, but often additional context is needed to determine what to do next
- NSM provides additional data needed to "pull a thread" (go, hunt, explore) vs reactively waiting for an alert

SEC530 | Defensible Security Architecture and Engineering 26

Alert Driven Workflows vs. Data Driven Workflows

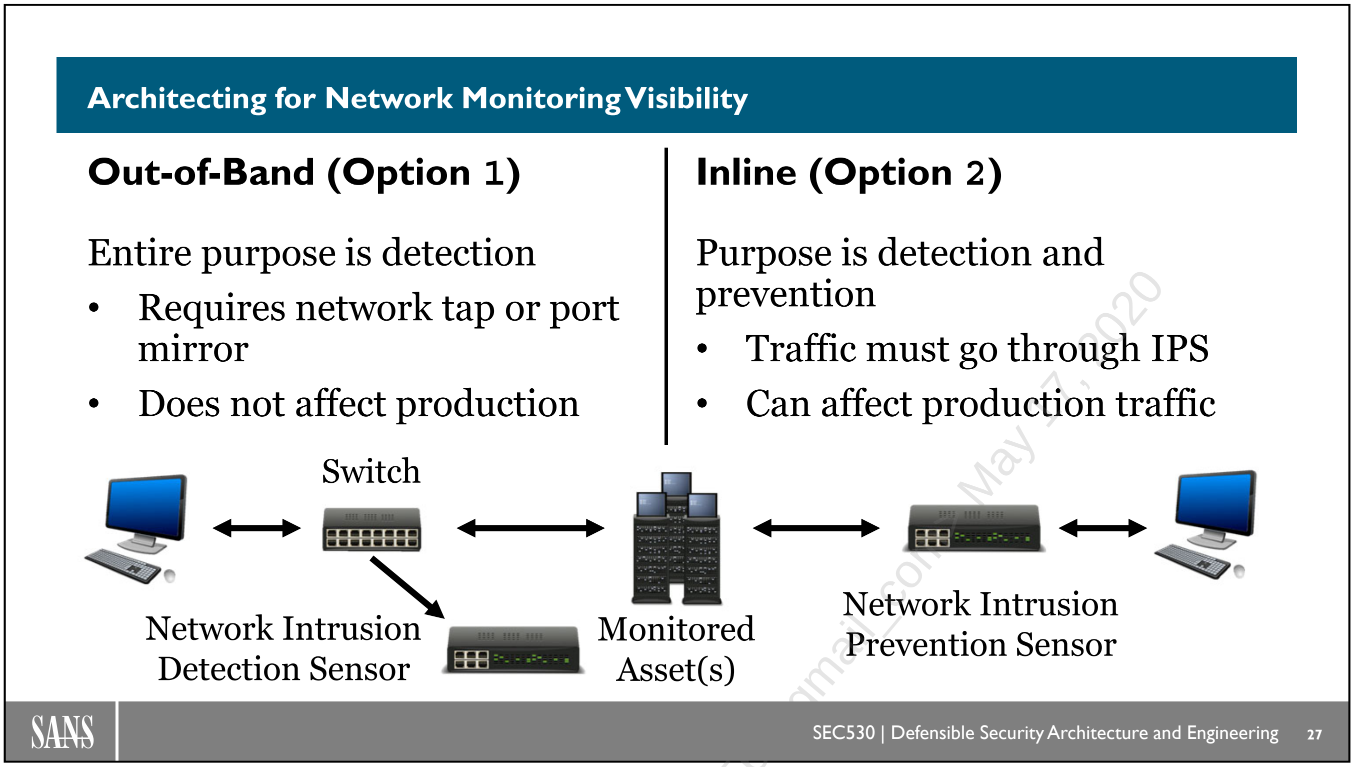
One of the main characteristics of NSM solutions is that they are not alert centric but data centric. What does that mean? Take as an example a traditional signature based IDS like Snort. When the contents of a packet trigger a rule signature, an alert is generated. The alert typically contains information on the single packet that triggered that rule and, typically, little else. We usually say that this is like looking at the world from the bottom of a well, or in other words, having tunnel vision[1]. This is often all the data you have to make a decision on what to do next: is it a false positive? Is it a real incident? In what phase of the attack are we? Is it Command and Control (C2) or exfiltration? Maybe lateral movement? What other data has been exchanged between these two systems? If this system is compromised, what other systems has it been talking to? Answering these questions is critical to triage a potential incident during the initial phases of an investigation and requires additional data.

NSM implements a data-centric philosophy that allows organizations to answer those questions, helping to discover, explore, and hunt for the presence of the adversary on the network.

In his blog post[2], Anton Chuvakin (Gartner) talks about a team manager that went as far as to say, "we don't hire alert responders here." He meant to say that in his team he doesn't want people to wait for alerts, but to go and explore, "hunt" for insights rather than "gather" alerts. Starting from a hypothesis, a "thread to pull", a question rather than an alert is characteristic of this newer way of approaching security monitoring. NSM implements this approach.

[1] Tunnel vision - https://pixabay.com/photos/tunnel-tube-tunnel-vision-light-336693/

[2] https://blogs.gartner.com/anton-chuvakin/2013/05/20/alert-driven-vs-exploration-driven-security-analysis/



Architecting for Network Monitoring Visibility

Out-of-Band (Option 1)

Entire purpose is detection

- Requires network tap or port mirror
- Does not affect production

Inline (Option 2)

Purpose is detection and prevention

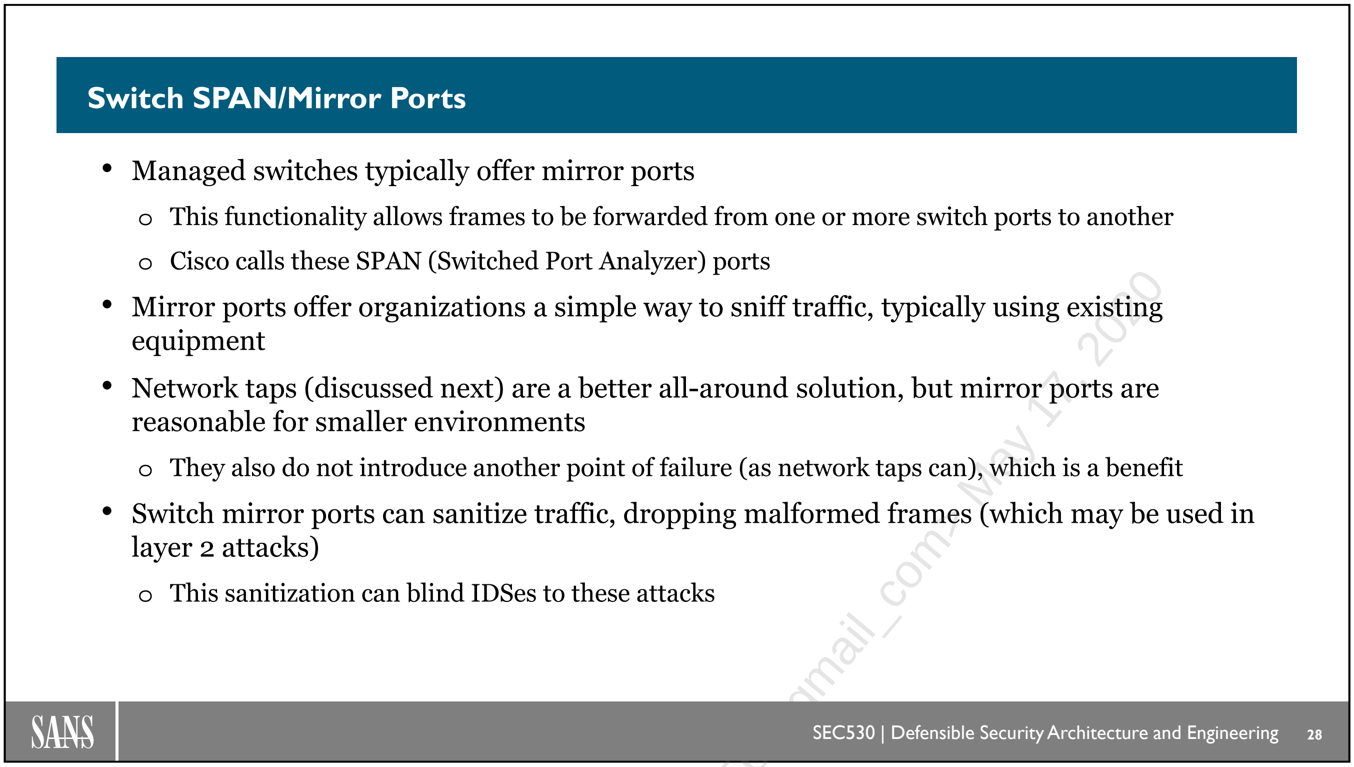
- Traffic must go through IPS
- Can affect production traffic

Switch
Network Intrusion Detection Sensor
Monitored Asset(s)
Network Intrusion Prevention Sensor

Architecting for Network Monitoring Visibility

The main point to be aware of in the context of network security monitoring is that network visibility is required. Gaining visibility is done by mirroring traffic to a dedicated network sensor via switch port mirroring or implementing a network tap. Mirroring traffic allows a network sensor to perform network security monitoring in detect only mode. Detect only simply means the sensor does not have prevention capabilities.

Another method is putting a network sensor inline between devices or networks. When inline, a network sensor can detect and prevent attacks. However, by forcing network traffic through the sensor, there is a risk of accidental denial of service and additional points of failure. Because of the points of failure, inline network sensors run with far fewer rules enabled compared to a detection only sensor. It is possible to run an inline network sensor with rules set to prevention and rules set for detection.



Switch SPAN/Mirror Ports

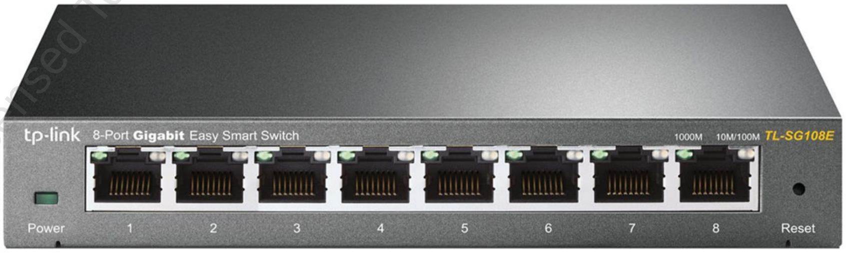
- Managed switches typically offer mirror ports
  - This functionality allows frames to be forwarded from one or more switch ports to another
  - Cisco calls these SPAN (Switched Port Analyzer) ports
- Mirror ports offer organizations a simple way to sniff traffic, typically using existing equipment
- Network taps (discussed next) are a better all-around solution, but mirror ports are reasonable for smaller environments
  - They also do not introduce another point of failure (as network taps can), which is a benefit
- Switch mirror ports can sanitize traffic, dropping malformed frames (which may be used in layer 2 attacks)
  - This sanitization can blind IDSes to these attacks

Switch SPAN/Mirror Ports

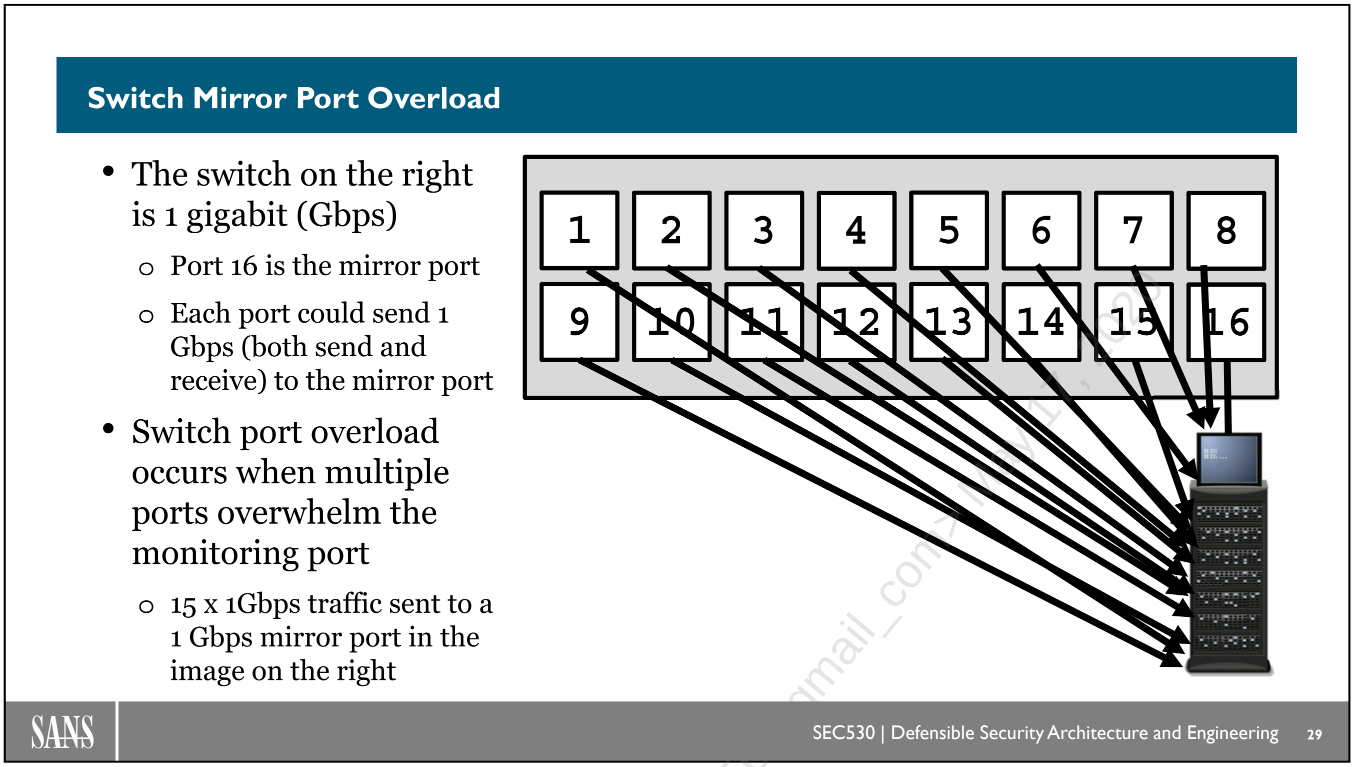
Switch mirror ports offer organizations (that have already deployed managed switches) an inexpensive way to gain network visibility. The offer many upsides, including price (the functionality is often already purchased) and simplicity. They are already in place, so they do not present an additional point of failure, as network taps do.

Inexpensive managed switches for SOHO (Small Office/Home Office) use have plummeted in price: the TP-Link 8-port gigabit switch shown below[1] costs under US $40. The switch offers a port mirror, QoS, VLANs, and more.

This switch is not recommended for enterprise use but shown to illustrate how inexpensive this technology has become.



[1] https://www.amazon.com/dp/B00K4DS5KU/ref=twister_B06XDLVVF6



Switch Mirror Port Overload

- The switch on the right is 1 gigabit (Gbps)
  - Port 16 is the mirror port
  - Each port could send 1 Gbps (both send and receive) to the mirror port
- Switch port overload occurs when multiple ports overwhelm the monitoring port
  - 15 x 1Gbps traffic sent to a 1 Gbps mirror port in the image on the right

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

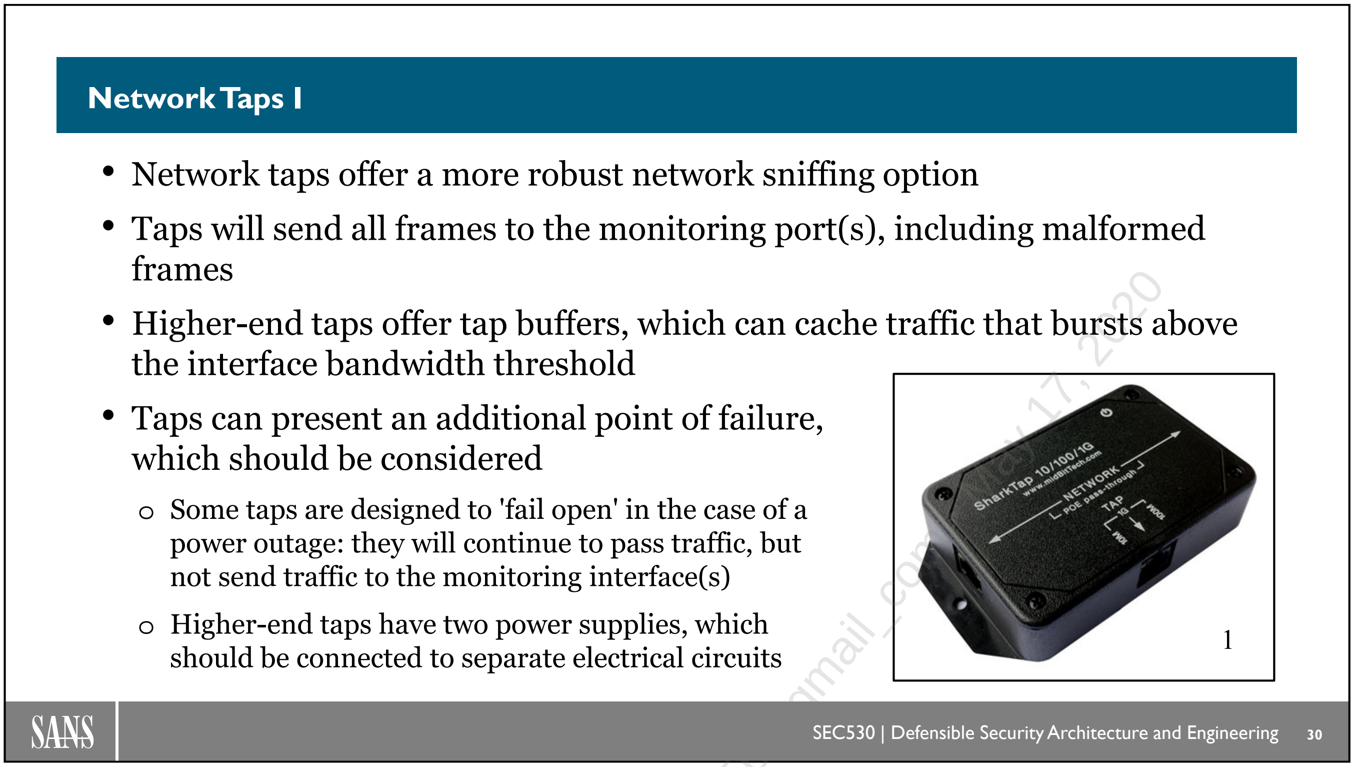
Switch Mirror Port Overload

Port overload is very common. The course authors have seen clients span 15 of 16 (and even 31 of 32) one-gigabit ports to a one-gigabit mirror port. What happens when you pour 31 gallons of water into a 1-gallon bucket? Your feet get wet.

A one Gbps switch allows one gig simultaneously on both duplexes (send and receive).

Always check the port utilization on the monitoring ports of IDSes, full packet capture devices, etc. Most of these devices use Linux. In that case, 'netstat --statistics' ('-s' does the same) will show dropped packets:

```
$ netstat -s
Ip:
    175483 total packets received
    88 with invalid addresses
    0 forwarded
    0 incoming packets discarded
    175156 incoming packets delivered
    105127 requests sent out
    416 outgoing packets dropped
    79 dropped because of missing route
```



Network Taps I

- Network taps offer a more robust network sniffing option
- Taps will send all frames to the monitoring port(s), including malformed frames
- Higher-end taps offer tap buffers, which can cache traffic that bursts above the interface bandwidth threshold
- Taps can present an additional point of failure, which should be considered
  - Some taps are designed to 'fail open' in the case of a power outage: they will continue to pass traffic, but not send traffic to the monitoring interface(s)
  - Higher-end taps have two power supplies, which should be connected to separate electrical circuits

1

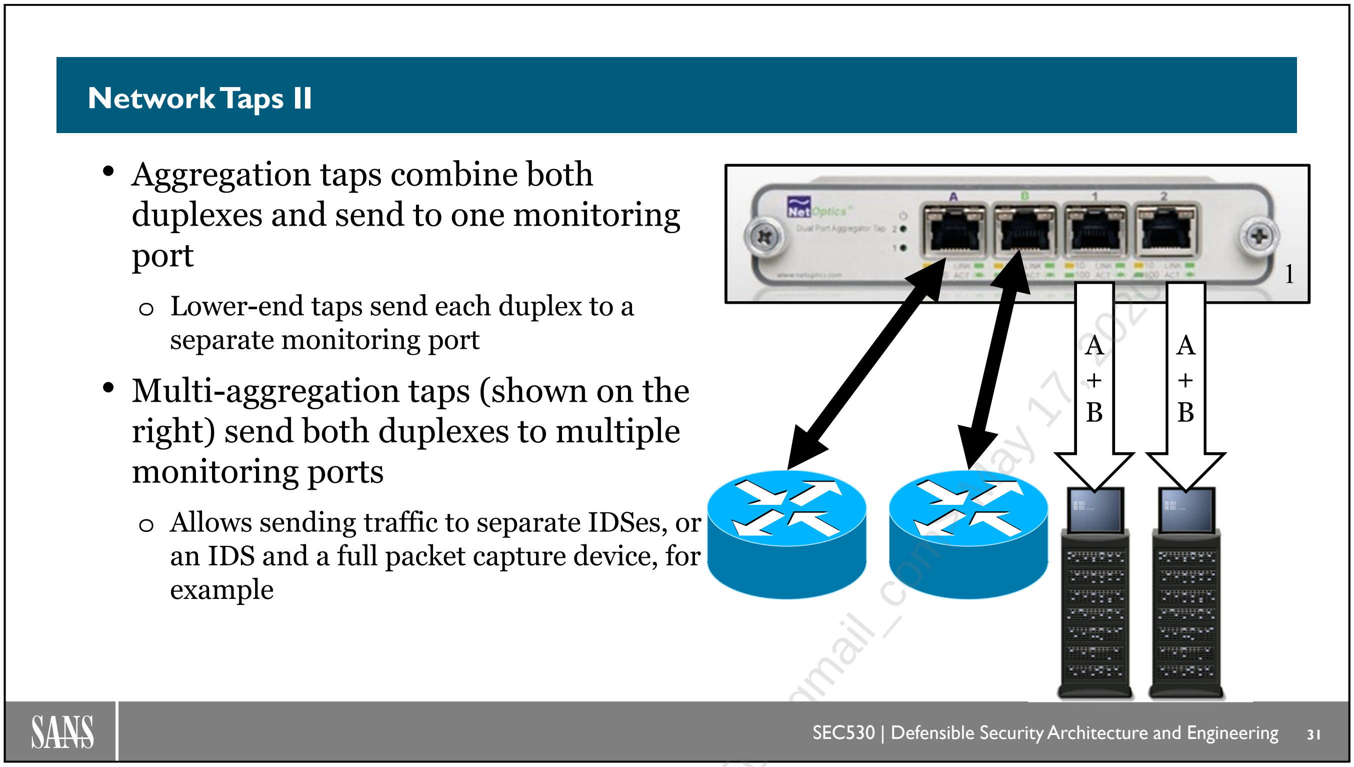
Network Taps I

As noted previously: switch mirror ports offer advantages for smaller environments, especially those that already contain managed switches that support this functionality.

Taps offer a better way to mirror traffic, due to support for tap buffers, and also because they will forward all frames to the monitoring port(s). Tap buffers are critical in cases where the monitoring interface is overloaded with traffic.

The tap shown above is the Gigabit SharkTap, which costs less than US $180 on amazon.com.

[1] http://www.midbittech.com/index.html



Network Taps II

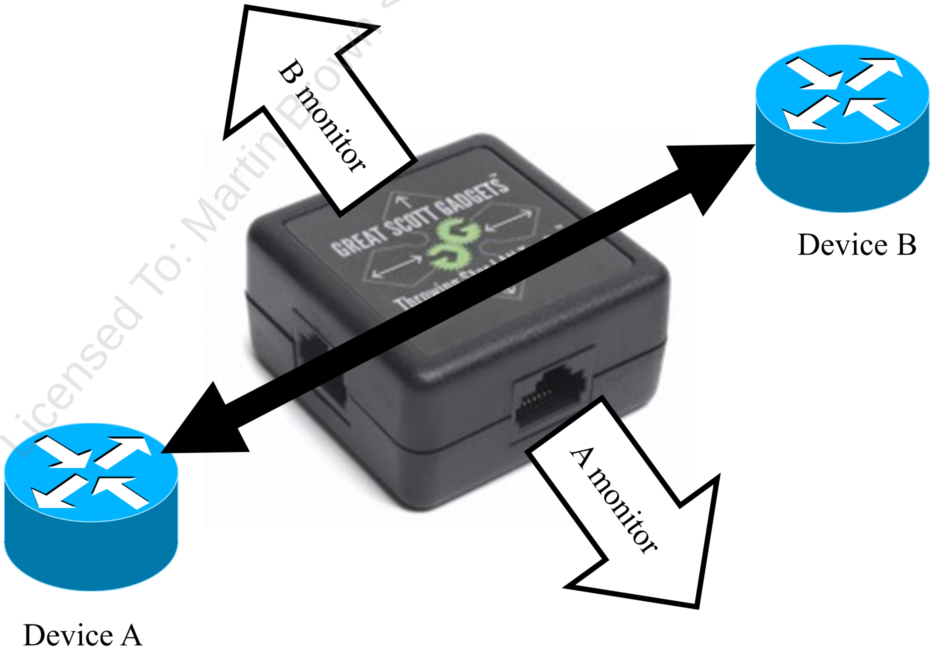
- Aggregation taps combine both duplexes and send to one monitoring port
  - Lower-end taps send each duplex to a separate monitoring port
- Multi-aggregation taps (shown on the right) send both duplexes to multiple monitoring ports
  - Allows sending traffic to separate IDSes, or an IDS and a full packet capture device, for example

SANS

SEC530 | Defensible Security Architecture and Engineering 31

Network Taps II

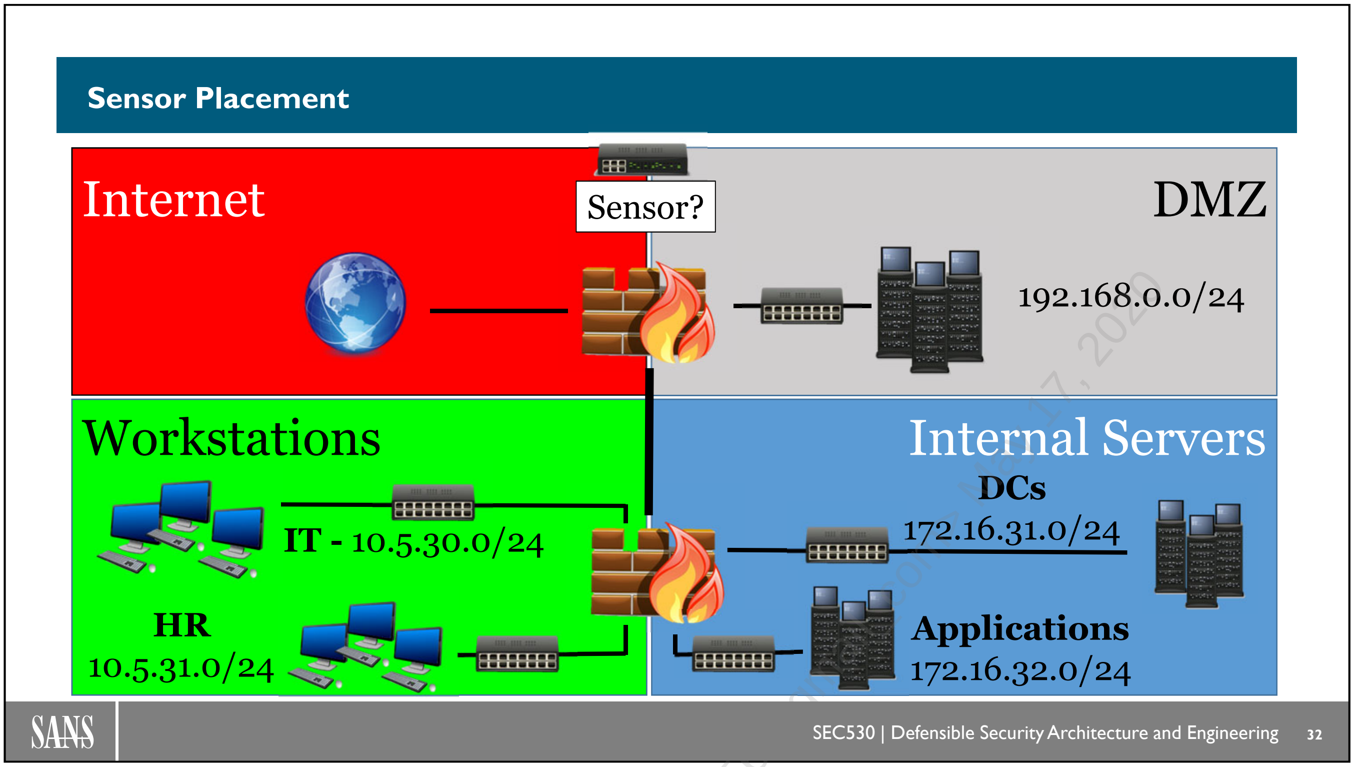
Low-end taps (like the US $40 Throwing Star LAN Tap shown below[2]) do not aggregate the two duplexes (sender and receiver). That device has four ports: device A, device B, A traffic to monitor port, and B traffic to monitor port.



[1] http://www.nextgigsystems.com/net_optics/10_100BaseT_Dual_Port_Agg.html

[2] https://greatscottgadgets.com/throwingstar/

Sensor Placement

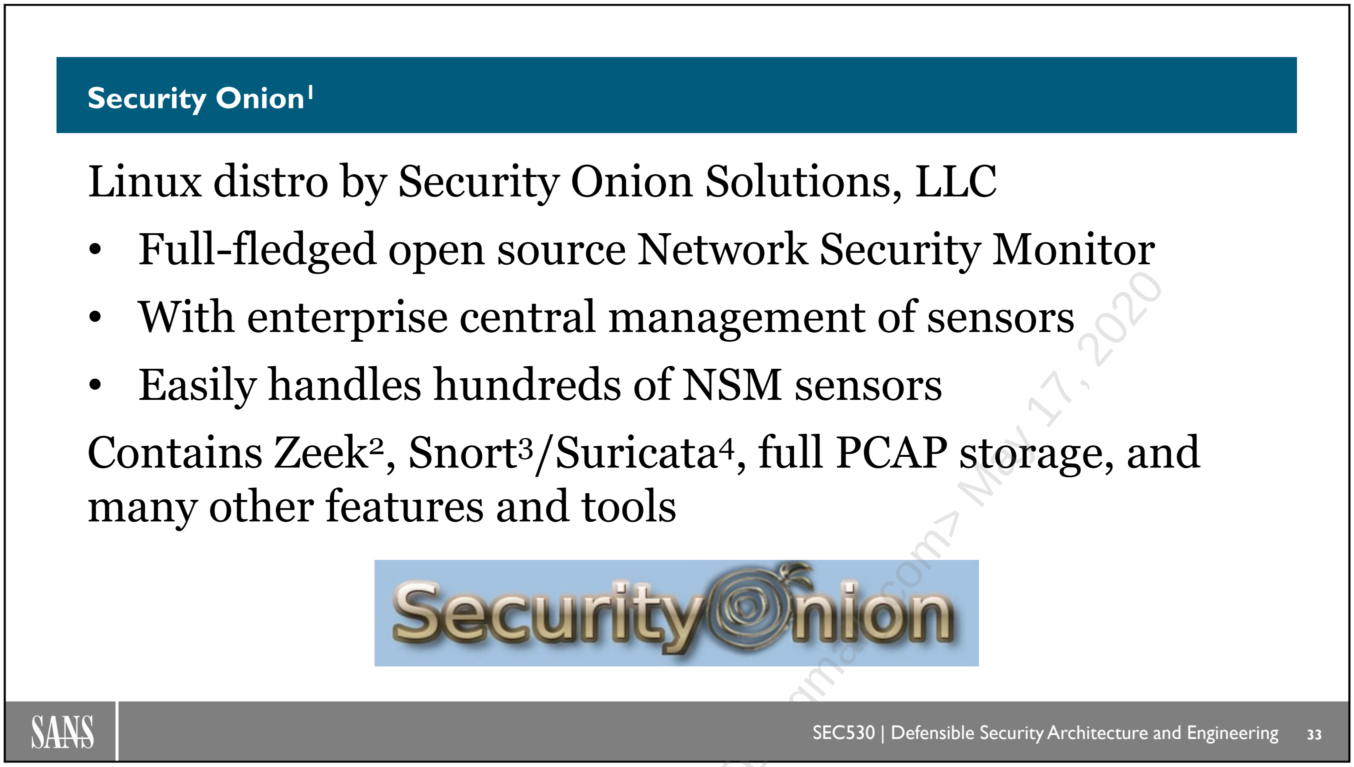


Sensor Placement

Network monitoring is dependent on the location of network sensors. The value of the data and alerts a sensor generates requires proper placement. The diagram above shows a network with segmentation. The question to consider is where sensors should be placed in the above diagram.

A network sensor between the internet and edge firewall provides visibility into external attacks as well as outbound traffic from an internal system. However, external attacks will happen and are not practical to investigate. Also, the internal traffic would be visible but not with internal IP addresses. Instead, the external IP of the edge firewall would show due to NAT. An external-facing sensor can be helpful for research and for reporting how often an organization is attacked to management.

A more actionable approach is to deploy network sensors on the inside of the environment. Each subnet that is visible to a sensor provides outbound traffic monitoring. Outbound monitoring is a key element as it can identify command and control, malware activity, and many other attacks such as phishing. Internal monitoring is also important but usually overlooked. Internal monitoring requires sensor customization and tuning to pull off but can identify anomalies and pivoting activity.



Security Onion[1]

Linux distro by Security Onion Solutions, LLC

- Full-fledged open source Network Security Monitor
- With enterprise central management of sensors
- Easily handles hundreds of NSM sensors

Contains Zeek[2], Snort[3]/Suricata[4], full PCAP storage, and many other features and tools

SecurityOnion

SANS

SEC530 | Defensible Security Architecture and Engineering 33

Security Onion[1]

Security Onion is one of the most well put together open source distros out there. Kali[2] is considered the red team distro while Security Onion[1] is considered the blue team distro. In this author's opinion, Security Onion can stand up to—and even beat—many commercial solution alternatives. It supports centralized management of sensors and has been deployed successfully in environments with over 800 remote sensors. It also has gone toe-to-toe with the best of breed IDS solutions, and Security Onion has outperformed many of them.

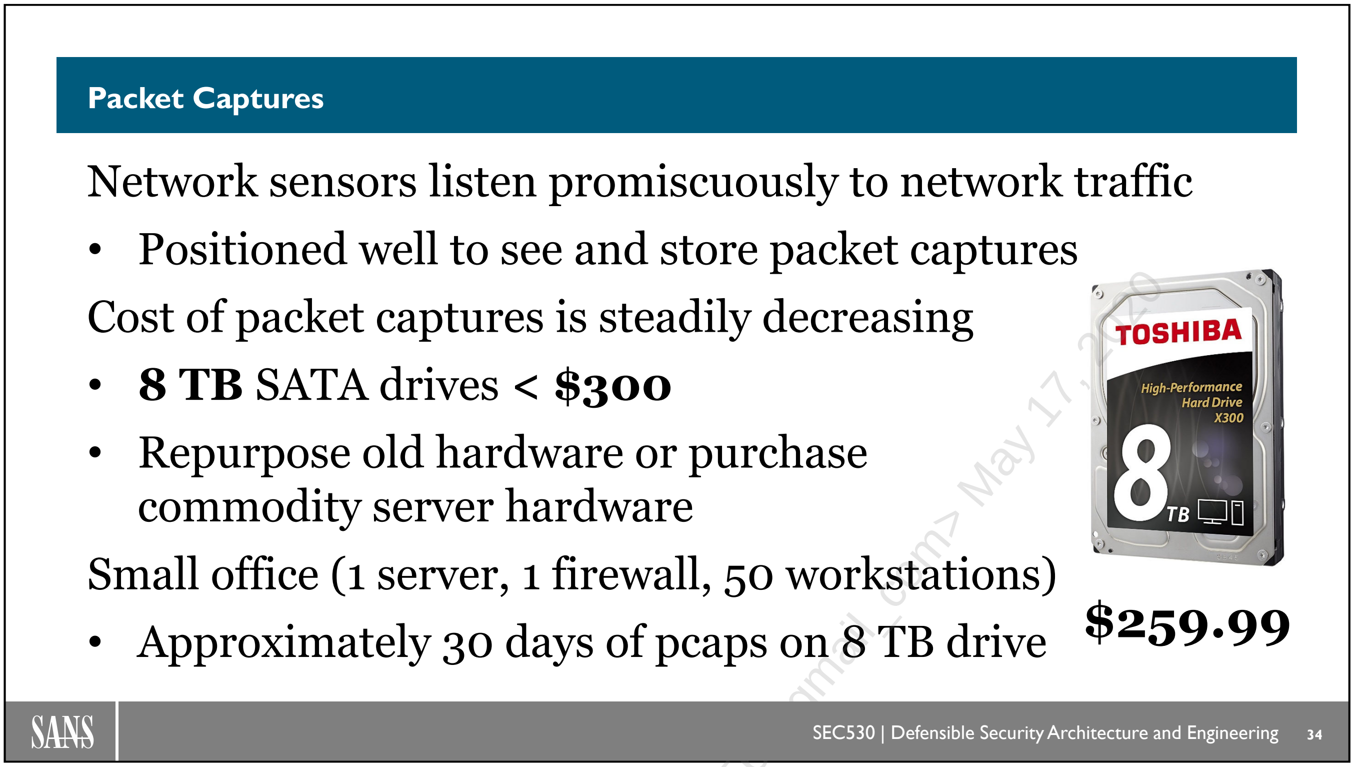
If you are looking for a quick and easy way to deploy multiple Zeek sensors that natively supports central management, then you probably are looking for Security Onion. On top of that, it natively supports running either Snort[3] or Suricata[4] which are open source IDS solutions as well as having the capability to save and rotate full PCAP storage should you want to store PCAPs. You can also purchase professional support or installation for Security Onion.

[1] https://securityonion.net/

[2] https://www.bro.org/zeek.html

[3] https://snort.org/

[4] https://suricata-ids.org/



Packet Captures

Network sensors listen promiscuously to network traffic

- Positioned well to see and store packet captures

Cost of packet captures is steadily decreasing

- **8 TB** SATA drives < **$300**
- Repurpose old hardware or purchase commodity server hardware

Small office (1 server, 1 firewall, 50 workstations)

- Approximately 30 days of pcaps on 8 TB drive

TOSHIBA
High-Performance
Hard Drive
X300
8
TB

$259.99

SANS SEC530 | Defensible Security Architecture and Engineering 34

Packet Captures

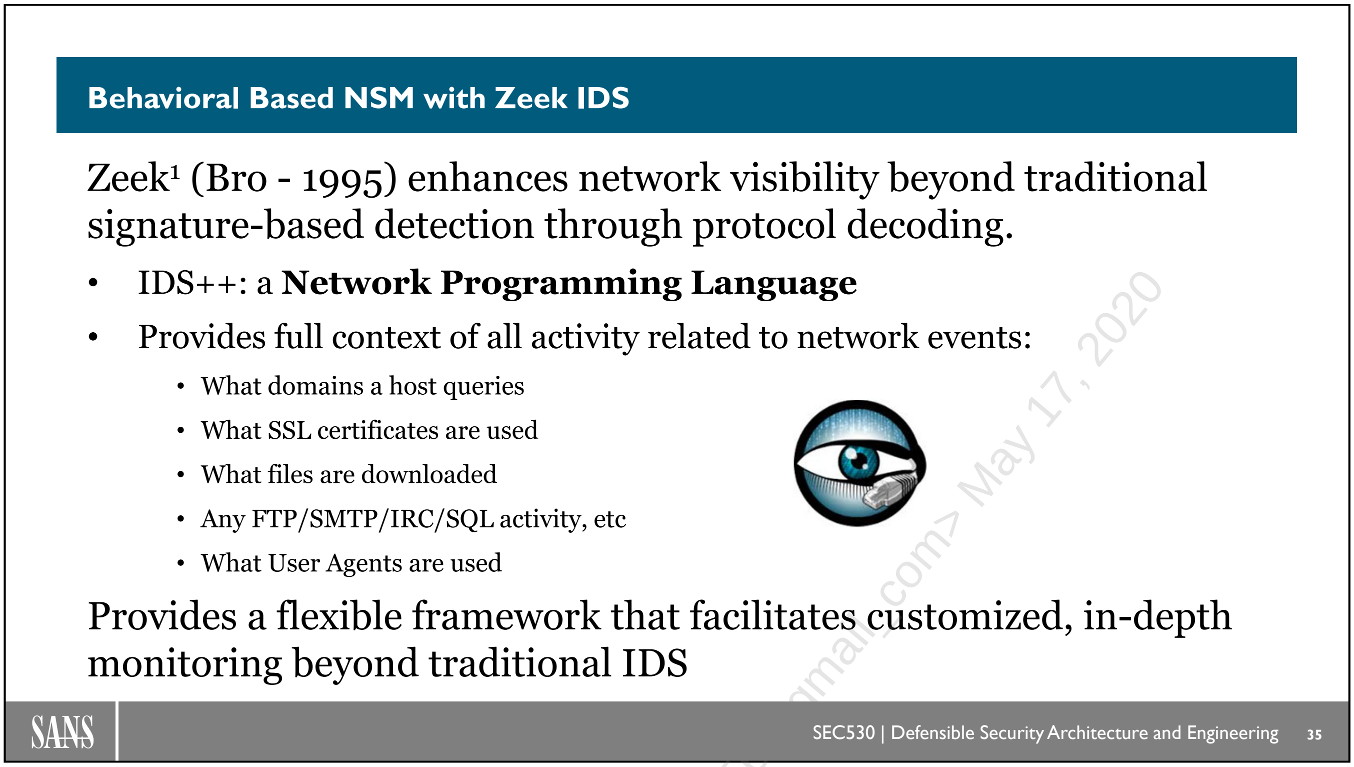
Modern network security monitoring solutions offer the ability to keep packet captures. The value in storing packet captures is they act as a historical representation of exactly what happened on the network. While packet captures have immense forensic value, packet captures also provide tremendous value in investigating alerts. Alerts are often nondescript and provide nominal information. Having additional information such as packet capture helps disprove or approve investigative theories.

The problem with packet captures is they can be large. An example of this is a remote location with a single server, firewall, and fifty workstations consuming 300 GB of network bandwidth day. With consumption of 300 GB of the day, an 8 TB hard drive would only be able to hold roughly 26 days' worth of packet captures. The good news is hard drive costs keep decreasing. Today, an 8 TB hard drive is available for less than three hundred dollars.

The example office mentioned is an actual production environment. That same environment with basic filtering added to the packet captures went from 26 days' worth of data to 1 years' worth of data. Packet capturing and filtering is available with various packet capturing solutions such as Security Onion[1] and Moloch[2].

[1] https://securityonion.net/

[2] https://github.com/aol/moloch



Behavioral Based NSM with Zeek IDS

Zeek[1] (Bro - 1995) enhances network visibility beyond traditional signature-based detection through protocol decoding.

- IDS++: a **Network Programming Language**
- Provides full context of all activity related to network events:
  - What domains a host queries
  - What SSL certificates are used
  - What files are downloaded
  - Any FTP/SMTP/IRC/SQL activity, etc
  - What User Agents are used

Provides a flexible framework that facilitates customized, in-depth monitoring beyond traditional IDS

SANS

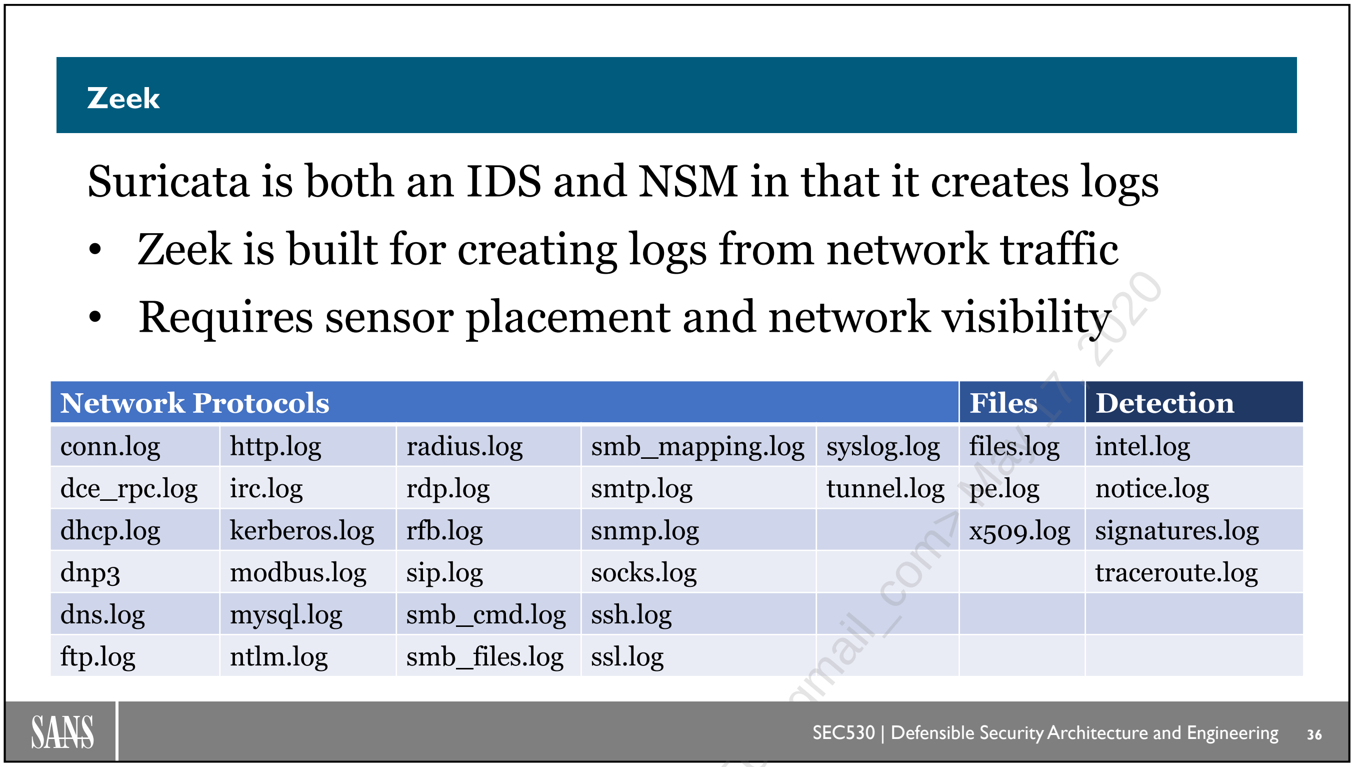
SEC530 | Defensible Security Architecture and Engineering 35

Behavioral Based NSM with Zeek IDS

Zeek[1] is fundamentally different from the typical IDS you may know. Zeek is, in fact, an open source real-time network analysis framework that implements the NSM philosophy. Zeek incorporates a very powerful network programming language that can accommodate a wide range of detection approaches. Note that Zeek is the new name for the long-established Bro system, originally developed by Vern Paxson in 1995. While most of the documentation in this class will refer to Zeek, you will notice that parts of the system retain the "Bro" name, and it also often appears in the documentation and distributions.

Zeek capabilities allow for data driven workflows based on event driven traffic analysis. Instead of focusing on providing alerts for anomalous or malicious traffic, it logs events and records of summarized connections. These network traffic events are predefined in the Zeek software, but they can also be customized by the user. Unlike an IDS, it's up to the analyst to decide whether the data collected is relevant to an investigation or not.

[1] https://www.bro.org/zeek.html



Zeek

Suricata is both an IDS and NSM in that it creates logs

- Zeek is built for creating logs from network traffic
- Requires sensor placement and network visibility

Network Protocols					Files	Detection
conn.log	http.log	radius.log	smb_mapping.log	syslog.log	files.log	intel.log
dce_rpc.log	irc.log	rdp.log	smtp.log	tunnel.log	pe.log	notice.log
dhcp.log	kerberos.log	rfb.log	snmp.log		x509.log	signatures.log
dnp3	modbus.log	sip.log	socks.log			traceroute.log
dns.log	mysql.log	smb_cmd.log	ssh.log			
ftp.log	ntlm.log	smb_files.log	ssl.log			

SANS

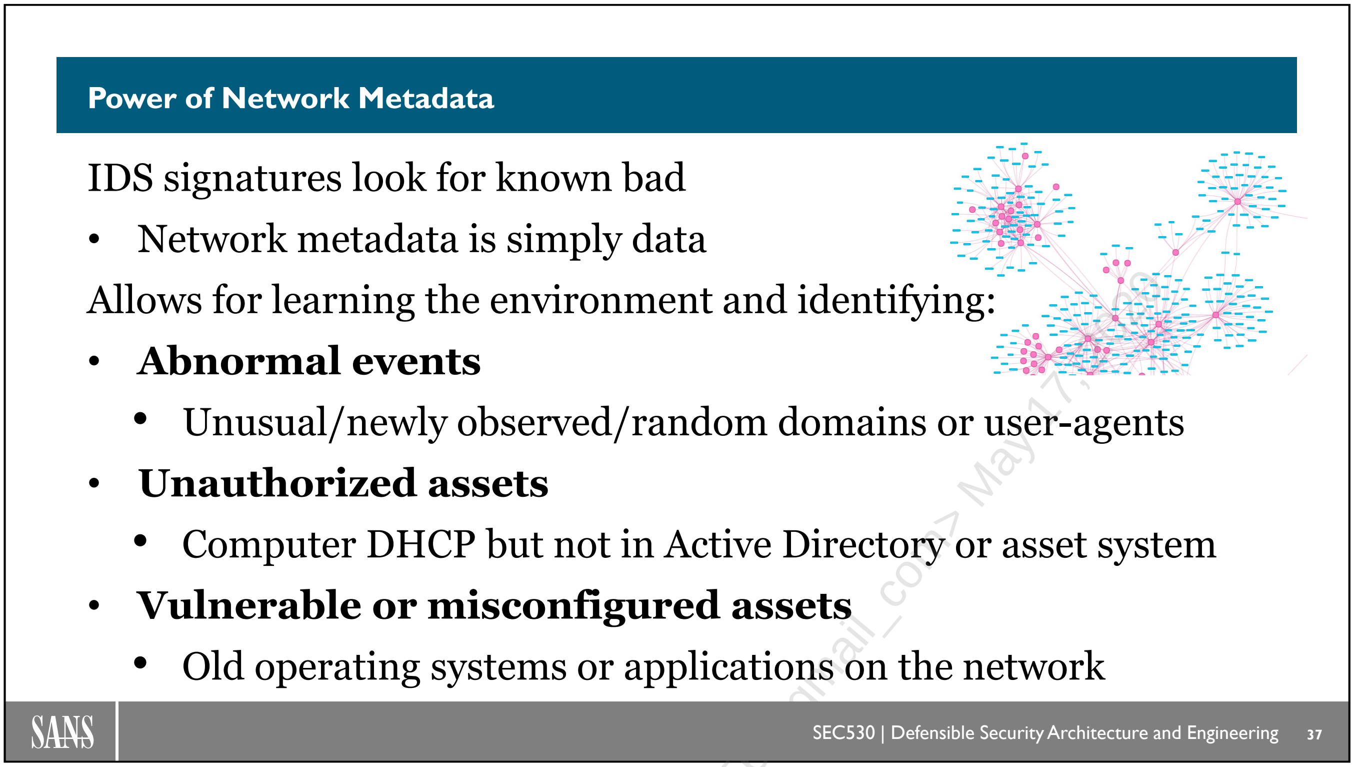
SEC530 | Defensible Security Architecture and Engineering 36

Zeek

Zeek is a network monitoring solution that generates logs about what is observed. Effectively, Zeek is a network metadata logging utility. As seen in the slides above, the number of logs Zeek generates is vast. In fact, this slide does not capture all of the logs available. Do not underestimate the value of Zeek.

Think of just one log for a second. For example, http.log includes logs of any HTTP request to or from a web service. Instead of having to configure logging on IIS 6, 7, 8, 8.5, Apache, Nginx, Squid proxy, Tomcat, and all the other HTTP sources an environment has you simply deploy Zeek and give it network visibility. Now, production systems do not need to be changed, and a consistent log format is generated. The Zeek log stays consistent and works even when HTTP services or upgraded or changed for new services. If you switched outbound requests from Squid to BlueCoat, you would still have all the logs via Zeek. If an unknown system is making outbound requests via HTTP Zeek would capture it and generate a log.

Having access to this kind of logs provides more power and capabilities than even an IDS or IPS. Think about it for a second. An IDS/IPS is mainly a blacklisting device that looks for known bad. Logs from Zeek are simply data about what is going on in an environment. Thus it allows one to learn his or her environment, look for anomalies, and find unauthorized activity. This, of course, requires one to collect and analyze this kind of data.



Power of Network Metadata

IDS signatures look for known bad

- Network metadata is simply data

Allows for learning the environment and identifying:

- **Abnormal events**
  - Unusual/newly observed/random domains or user-agents
- **Unauthorized assets**
  - Computer DHCP but not in Active Directory or asset system
- **Vulnerable or misconfigured assets**
  - Old operating systems or applications on the network

SANS SEC530 | Defensible Security Architecture and Engineering 37

Power of Network Metadata

So, what can be done using network logs? After all the logs are simply data about what occurred, not an alert. Using this data allows an organization to learn about themselves. What encryption algorithms do they use? What systems connect to specific systems. What HTTP user-agents do they use? What domains does the organization access? Studying this data and then using scripts or a SIEM to put rules around this allows for manual or automated alerting.

For example, looking at all the new domains observed to an organization is likely to include domains used for phishing or drive-by malware. Knowing the list is going to be smaller tricks like frequency analysis or whois creation data lookup can be performed with tools like freq.py or freq_server.py[1] and domains_stats.py[2]. Depending on the organization, the dhcp.log can be analyzed for hostnames that do not match the organizations naming standard or MAC address list of company devices. Other logs such as software.log can be used to find vulnerable operating systems or applications through passive identification.

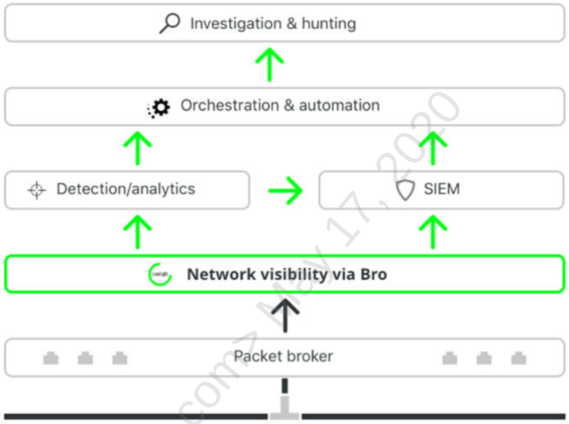
The point is, the limit on how useful the data can be is based on the team using the data. Zeek data is one of the number one data sources if not the number one for network monitoring and alerting. For example, connection logs can be crawled back in time to look for connections that reoccur on frequent intervals the same way command and control activity. DNS logs can be inspected for new domain requests or randomly named domains or cousin domains.

[1] https://github.com/MarkBaggett/MarkBaggett/tree/master/freq

[2] https://github.com/MarkBaggett/domain_stats

Zeek & Network Visibility in the Enterprise

- Zeek is to the network what Sysmon[1] is to the endpoint
- Adds a layer in the overall logs + endpoint + network + cloud + mobile **visibility** stack



Source: Corelight[2]
https://www.corelight.com/products/solutions

Zeek & Network Visibility in the Enterprise

From a defensible security architecture perspective, Zeek allows us to add visibility to the network layer in the same fashion that Sysmon allows us to add visibility to the endpoint. It's of such significance that even Gartner has created a new definition for this type of solutions, called Network Traffic Analysis or NTA.

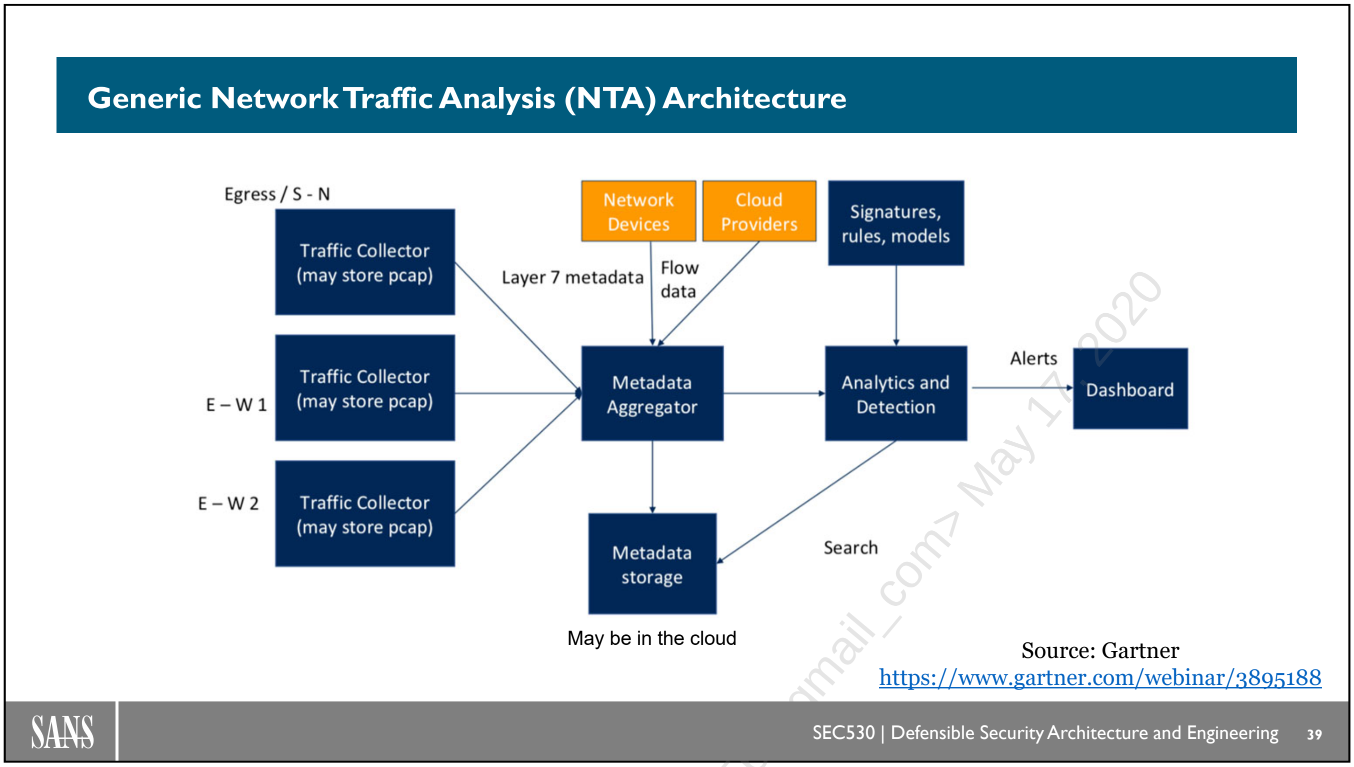
The metadata collected by NTA solutions like Zeek can be consumed by detection or analytics products to build supervised or unsupervised models, statistics, rules, etc., or by SIEM devices to produce signals. These can be at the same time consumed by other solutions that automate or orchestrate certain response, speeding up investigations and incident response workflows.

Zeek is offered commercially as a supported product by the company Corelight.

[1] Sysmon https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon

[2] NTA https://blogs.gartner.com/anton-chuvakin/2018/09/20/nta-the-other-ids/

[3] Corelight https://www.corelight.com/products/solutions



Generic Network Traffic Analysis (NTA) Architecture

Egress / S - N

Traffic Collector (may store pcap)

E – W 1

Traffic Collector (may store pcap)

E – W 2

Traffic Collector (may store pcap)

Network Devices

Cloud Providers

Signatures, rules, models

Layer 7 metadata

Flow data

Metadata Aggregator

Analytics and Detection

Alerts

Dashboard

Metadata storage

Search

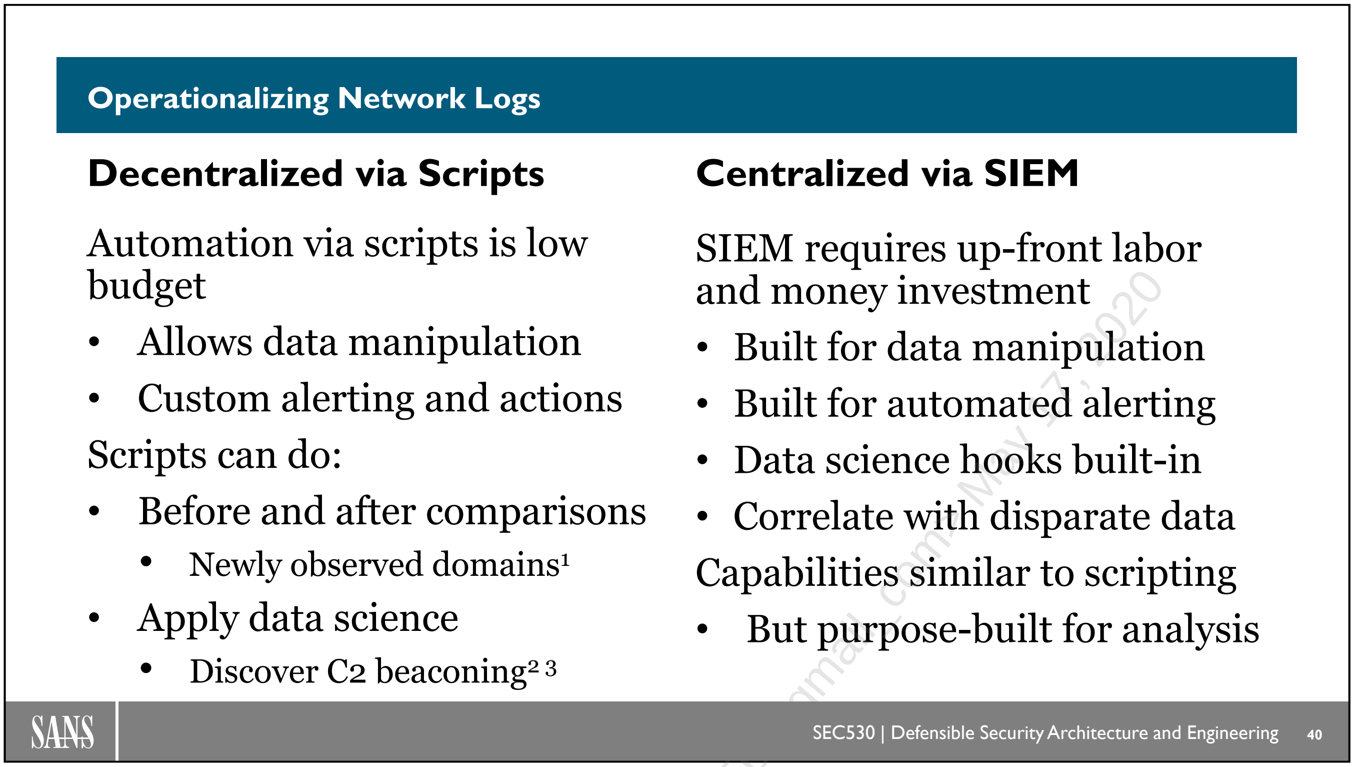
May be in the cloud

Source: Gartner
https://www.gartner.com/webinar/3895188

Generic Network Traffic Analysis (NTA) Architecture

A NTA architecture should support specific use cases such as:

- Command and Control (C2) to known and unknown bad (IDS can't do the latter)
- Ransomware
- Exfiltration and data theft
- Lateral movement by the attacker (requires east-west sensors)
- Compromised devices
- Internal access abuse
- Other network misconfigurations (troubleshooting)



Operationalizing Network Logs

Decentralized via Scripts

Automation via scripts is low budget

- Allows data manipulation
- Custom alerting and actions

Scripts can do:

- Before and after comparisons
  - Newly observed domains[1]
- Apply data science
  - Discover C2 beaconing[2 3]

Centralized via SIEM

SIEM requires up-front labor and money investment

- Built for data manipulation
- Built for automated alerting
- Data science hooks built-in
- Correlate with disparate data

Capabilities similar to scripting

- But purpose-built for analysis

SANS

SEC530 | Defensible Security Architecture and Engineering 40

Operationalizing Network Logs

Using Zeek or Suricata metadata requires finding a method to analyze the data on mass. Scripting works well but requires scripting knowledge and developing methods for analyzing massive amounts of data that is likely distributed across multiple platforms. Writing scripts that work across distributed sensors works well on platforms like Security Onion. For example, Johannes Ullrich has a bash script that crawls across Security Onion sensors to find new domains that were accessed in the last 24 hours. Other frameworks like RITA[2] or Real Intelligence Threat Analytics consumes Zeek data to find command and control beaconing. Since scripts are used almost anything can be done. However, scripting does not scale as well as other solutions.

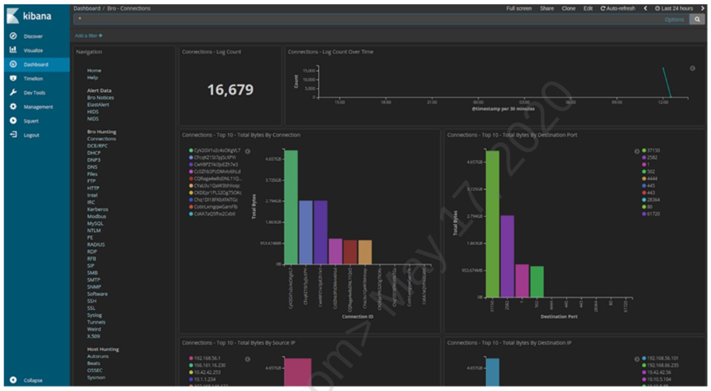
Sending Zeek or Suricata logs into a SIEM provides a lot of capabilities. First, data is easy to analyze and operationalize. Now, jumping from an IDS alert over to connection logs, DNS query logs, and SSL information is easy. On top of that data can be enriched with a SIEM. For example, the DNS query field of sec530.com can be used to add additional data like creation_date of 2017-11-01 dynamically, and it can be dynamically tagged as being a frequently accessed domain or not. Also, a SIEM can on the fly detect if this is a newly observed domain. Enriching the log allows automatic analysis and provides improved criteria for automated alerts. The latest release of Security Onion includes the Elastic Stack for just such purposes. Security Onion is now an NSM platform and SIEM.

Network logs are so important that SEC555: SIEM with Tactical Analytics[4] spends an entire day on collecting, enriching, and using Zeek or Suricata logs to find malicious or unauthorized occurrences.

[1] https://github.com/Security-Onion-Solutions/security-onion/wiki/DNSAnomalyDetection
[2] https://github.com/ocmdev/rita
[3] https://github.com/austin-taylor/flare
[4] https://www.sans.org/course/siem-with-tactical-analytics

Security Onion Network + Endpoint Visibility

- New Elastic Stack and Kibana dashboards in Security Onion allow for easy pivoting between network and host data (analytical pivoting).

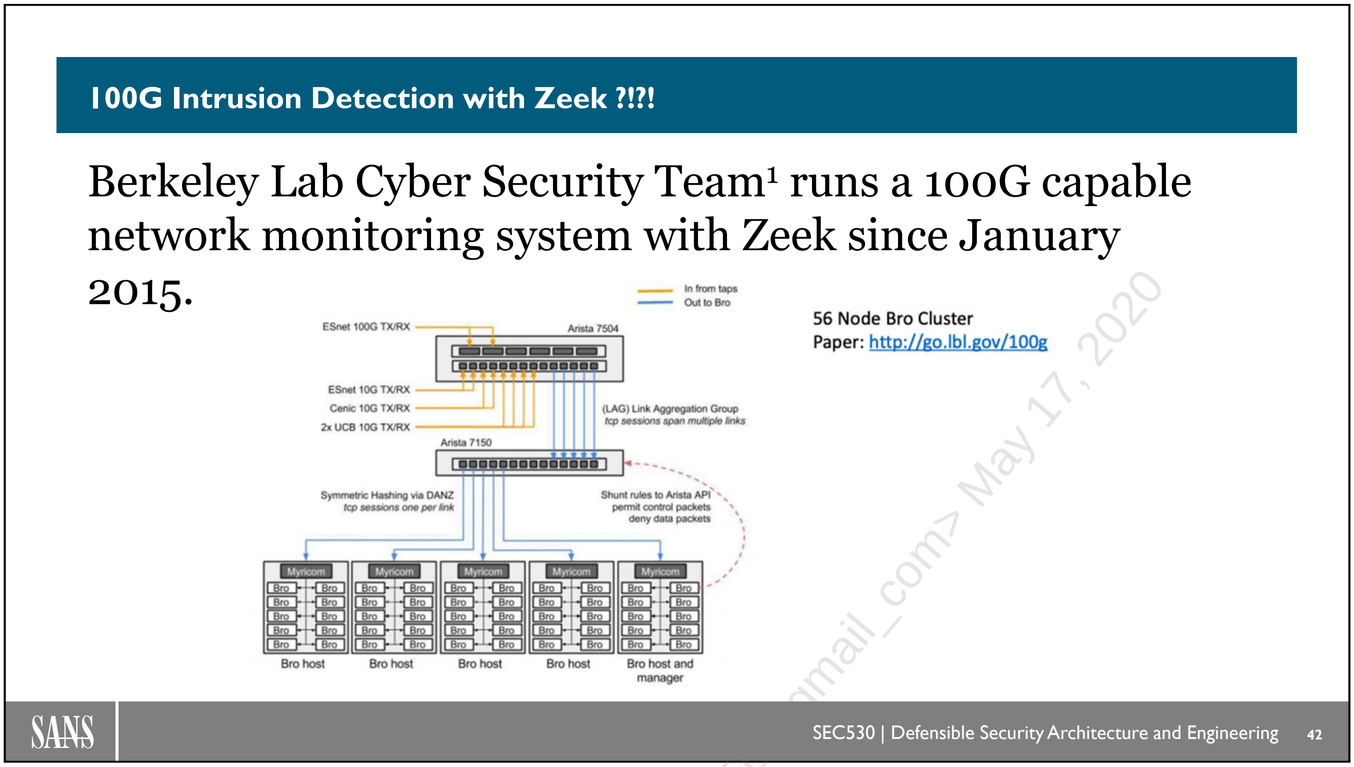


Security Onion Network + Endpoint Visibility

In 2017, Security Onion decided to incorporate the Elastic Stack into its distribution, adding parsers for not only NSM data through Zeek, but also for host visibility through Sysmon among others. This allows for seamless pivoting[2] between network and host data.

[1] https://blog.securityonion.net/2017/07/towards-elastic-on-security-onion.html

[2] https://sqrrl.com/cyber-incident-investigation-training-reducing-evidence-abstraction/



100G Intrusion Detection with Zeek ?!?!

Berkeley Lab Cyber Security Team[1] runs a 100G capable network monitoring system with Zeek since January 2015.

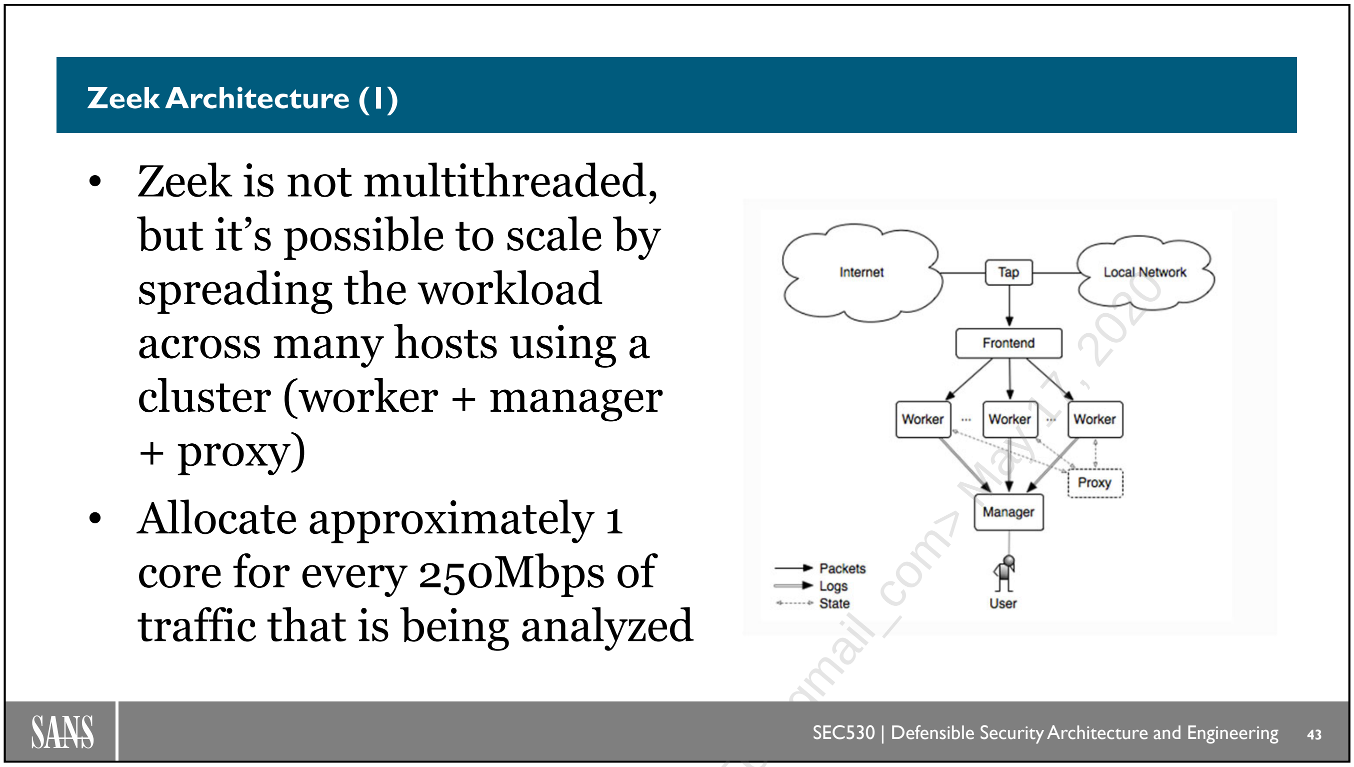
56 Node Bro Cluster
Paper: http://go.lbl.gov/100g

100G Intrusion Detection with Zeek ?!?!

While 100Gbps links have become more prevalent these days, security operations' ability to maintain network monitoring at this traffic volume has not always kept pace, as many commercial offerings fail to support monitoring at this speed. After extensive evaluation, deployment and testing, the Berkeley Lab Cyber Security Team brought their 100G capable network monitoring system online in January 2015. They created a technical document to help other security teams, architects, engineers and Zeek enthusiasts learn about the components and configuration that they used to create this monitoring system.

The clustering capabilities of Zeek allow for this type or highly efficient architecture.

[1] https://www.zeek.org/brocon2018/slides/Ed_and_Daniel._Managing_Bro_Deployments.pdf



Zeek Architecture (1)

- Zeek is not multithreaded, but it's possible to scale by spreading the workload across many hosts using a cluster (worker + manager + proxy)
- Allocate approximately 1 core for every 250Mbps of traffic that is being analyzed

Zeek Architecture (1)
Zeek is not multithreaded, so once the limitations of a single processor core are reached the only option currently is to spread the workload across many cores, or even many physical computers. The cluster deployment scenario for Zeek is the current solution to build these larger systems. The tools and scripts that accompany Zeek provide the structure to easily manage many Zeek processes examining packets and doing correlation activities but acting as a singular, cohesive entity.

Zeek has different functional components: the worker, the manager and the proxy. Together, they form a Zeek cluster. In the diagram shown above, the tap splits the packet stream in order to make a copy available for inspection. This could be a span port on a switch or an optical splitter on fiber networks. The frontend flow distributor could be hardware (i.e. cFlow or cVu device from cPacket) or software based (like PF_RING), but in any case, it's not part of the Zeek system.

The workers, as the name suggests, do the heavy-lifting by sniffing the traffic, reassembling it into strings and doing the protocol analysis on them. This would be the equivalent to a sensor in a traditional IDS. The fastest memory and CPU core speed you can afford is recommended since all of the protocol parsing and most analysis will take place here. There are no particular requirements for the disks in workers since almost all logging is done remotely to the manager, and normally very little is written to disk.

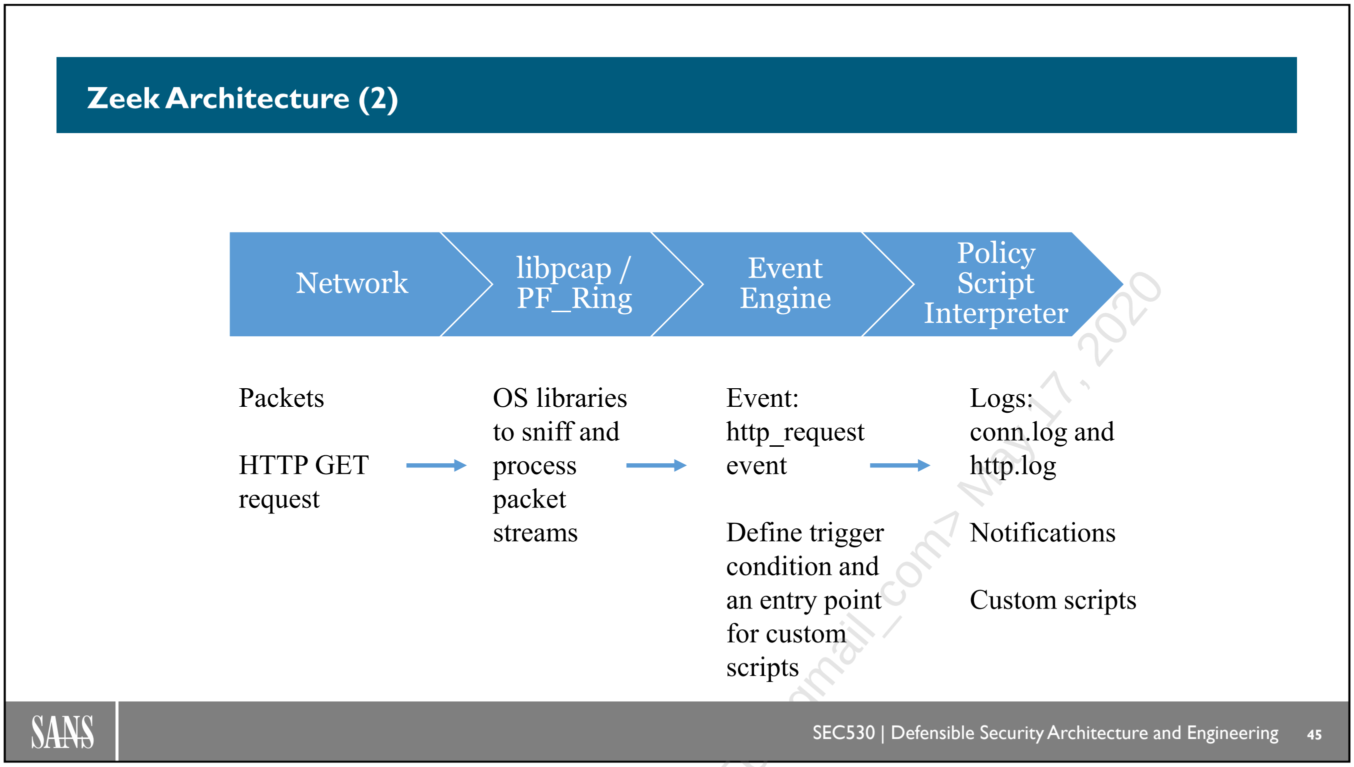
The manager is the central log and notice collector, creating a single log for all the traffic collected by the workers, de-duplicating notices and acting as a chokepoint. Finally, the proxy is a Zeek process that manages the synchronized state. Variables can be synchronized across connected Zeek processes automatically. Proxies help the workers by alleviating the need for all of the workers to connect directly to each other.

In a standalone deployment, a worker, manager and proxy are setup on the same host. In a cluster architecture, a worker or a number of workers can be setup in separate hosts depending on the traffic volume and the type of hardware available for network collection. A rule of thumb commonly followed is to allocate approximately 1 core for every 250Mbps of traffic that is being analyzed. The frontend flow distributor allows this model to scale.

For example, if your traffic peaks around 2Gbps (combined) and you want to handle traffic at peak load, you may want to have 8 cores available (2048 / 250 == 8.2). If the 250Mbps estimate works for your traffic, this could be handled by 2 physical hosts dedicated to being workers with each one containing a quad-core processor.

This document describes the Bro cluster[1] architecture and how to configure it:

[1] https://docs.zeek.org/en/stable/cluster/



Zeek Architecture (2)

Network	libpcap / PF_Ring	Event Engine	Policy Script Interpreter
Packets HTTP GET request	OS libraries to sniff and process packet streams	Event: http_request event Define trigger condition and an entry point for custom scripts	Logs: conn.log and http.log Notifications Custom scripts

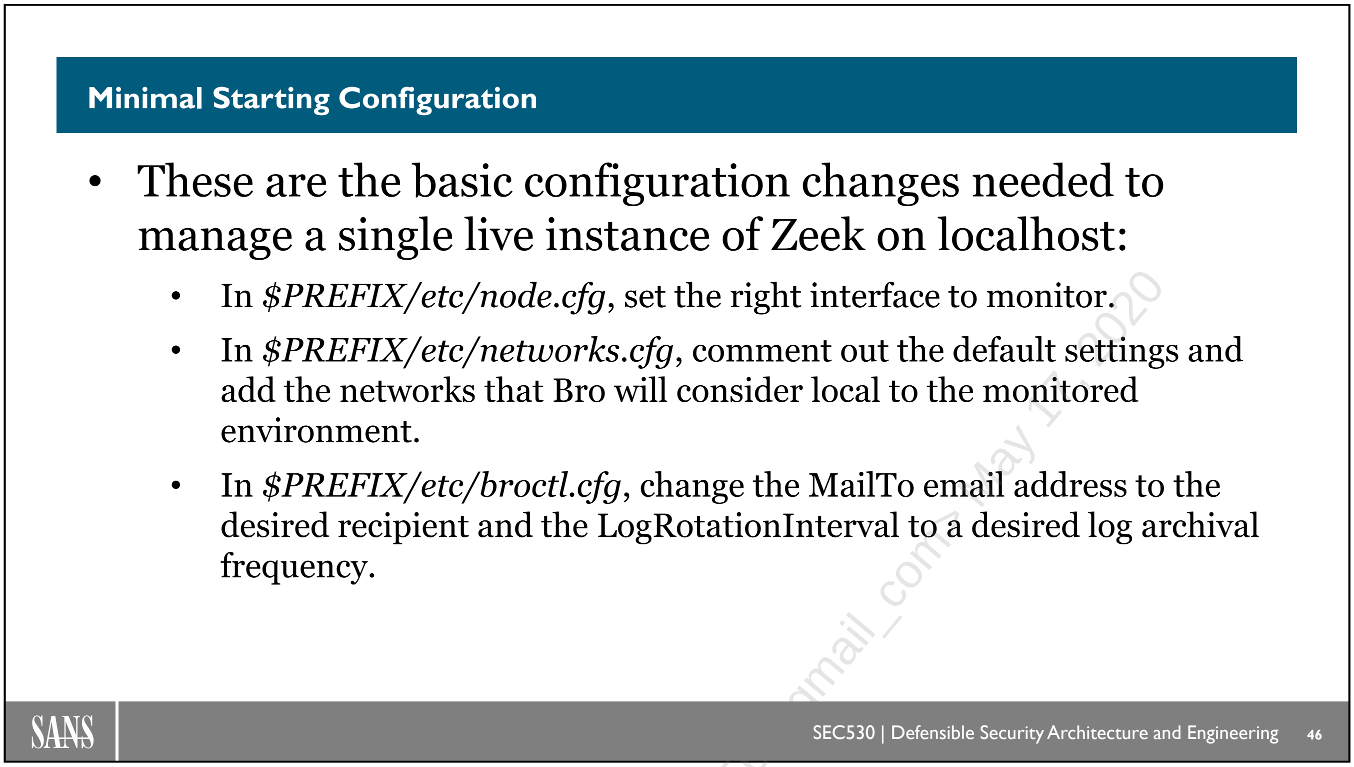
Zeek Architecture (2)

Architecturally, Zeek is layered into two major components. Its *event engine* (or *core*) reduces the incoming packet stream into a series of higher-level *events*. These events reflect network activity in policy-neutral terms, i.e., they describe *what* has been seen, but not *why*, or whether it is significant. For example, every HTTP request on the wire turns into a corresponding http_request event that carries with it the involved IP addresses and ports, the URI being requested, and the HTTP version in use. The event, however, does not convey any further *interpretation*, e.g., of whether that URI corresponds to a known malware site.

Such semantics are instead derived by Zeek's second main component, the *script interpreter*, which executes a set of *event handlers* written in Zeek's custom scripting language. These scripts can express a site's security policy, i.e., what actions to take when the monitor detects different types of activity. More generally they can derive any desired properties and statistics from the input traffic. Zeek's language comes with extensive domain-specific types and support functionality; and, crucially, allows scripts to maintain state over time, enabling them to track and correlate the evolution of what they observe across connection and host boundaries. Bro scripts can generate real-time alerts and also execute arbitrary external programs on demand, e.g., to trigger an active response to an attack.

Zeek architecture also relies heavily on the libpcap library to sniff and filter packets, very much like Snort or Suricata. PF_RING can be used to improve the performance of the sniffing process acting as a host load balancing, directing traffic streams to different cores where there are multiple Zeek workers sharing the same host.

[1] https://docs.zeek.org/en/stable/intro/index.html#architecture



Minimal Starting Configuration

- These are the basic configuration changes needed to manage a single live instance of Zeek on localhost:
  - In *$PREFIX/etc/node.cfg*, set the right interface to monitor.
  - In *$PREFIX/etc/networks.cfg*, comment out the default settings and add the networks that Bro will consider local to the monitored environment.
  - In *$PREFIX/etc/broctl.cfg*, change the MailTo email address to the desired recipient and the LogRotationInterval to a desired log archival frequency.

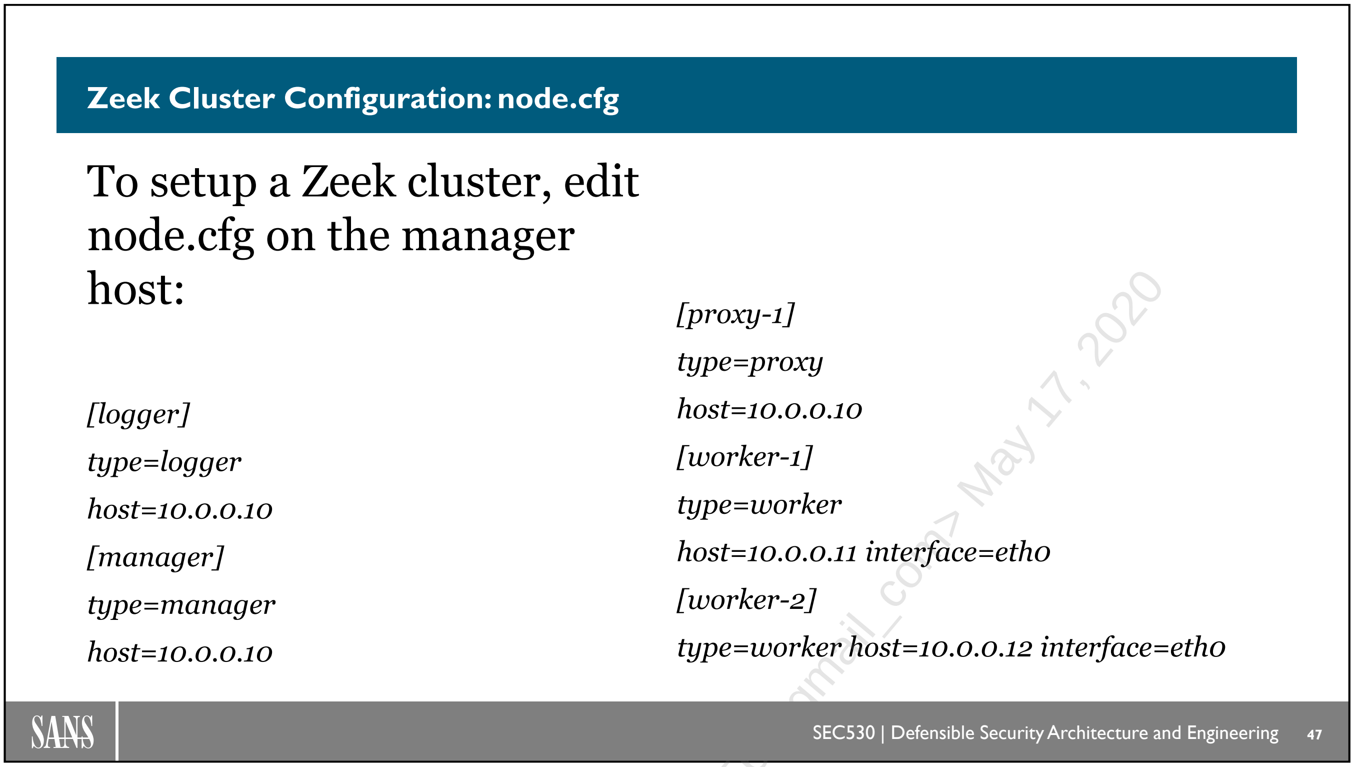
SEC530 | Defensible Security Architecture and Engineering 46

Minimal Starting Configuration

These are the basic configuration changes to make for a minimal BroControl installation that will manage a single Zeek instance on the localhost:

- In $PREFIX/etc/node.cfg, set the right interface to monitor.
- In $PREFIX/etc/networks.cfg, comment out the default settings and add the networks that Bro will consider local to the monitored environment.
- In $PREFIX/etc/broctl.cfg, change the MailTo email address to the desired recipient and the LogRotationInterval to the desired log archival frequency.

The main entry point for the default analysis configuration of a standalone Bro instance managed by BroControl is the $PREFIX/share/bro/site/local.bro script.



Zeek Cluster Configuration: node.cfg

To setup a Zeek cluster, edit node.cfg on the manager host:

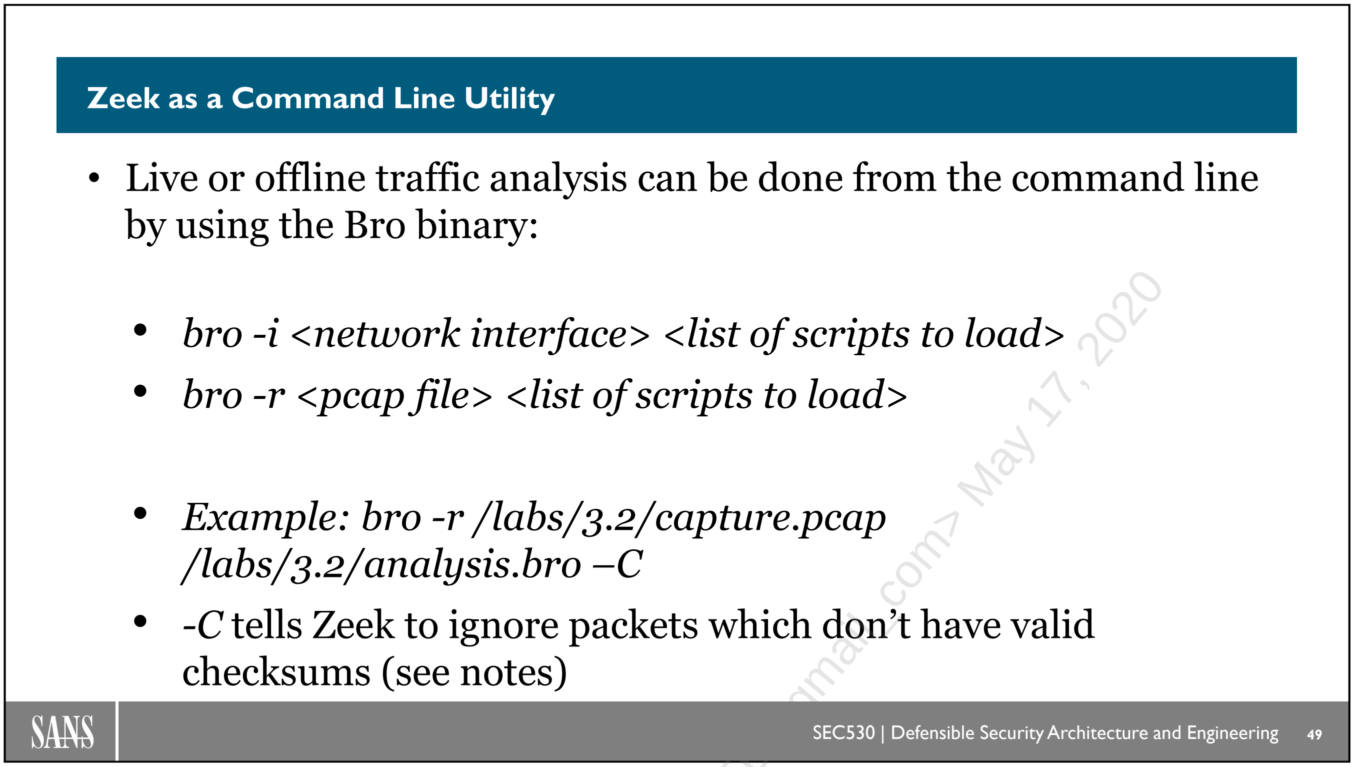
```
[logger]
type=logger
host=10.0.0.10
[manager]
type=manager
host=10.0.0.10
```

```
[proxy-1]
type=proxy
host=10.0.0.10
[worker-1]
type=worker
host=10.0.0.11 interface=eth0
[worker-2]
type=worker host=10.0.0.12 interface=eth0
```

Zeek Cluster Configuration: node.cfg
Edit the BroControl node configuration file, <prefix>/etc/node.cfg to define where logger, manager, proxies, and workers are to run. For a cluster configuration, you must comment-out (or remove) the standalone node in that file, and either uncomment or add node entries for each node in your cluster (logger, manager, proxy, and workers). For example, if you wanted to run five Bro nodes (two workers, one proxy, a logger, and a manager) on a cluster consisting of three machines, your cluster configuration would look like this:

```
[logger]
type=logger
host=10.0.0.10

[manager]
type=manager
host=10.0.0.10

[proxy-1]
type=proxy
host=10.0.0.10

[worker-1]
type=worker
host=10.0.0.11
interface=eth0

[worker-2]
type=worker
host=10.0.0.12
interface=eth0
```

For a complete reference of all options that are allowed in the node.cfg file, and how to install Zeek on all machines in the cluster see https://docs.zeek.org/en/stable/configuration/



Zeek as a Command Line Utility

- Live or offline traffic analysis can be done from the command line by using the Bro binary:
  - *bro -i <network interface> <list of scripts to load>*
  - *bro -r <pcap file> <list of scripts to load>*
  - *Example: bro -r /labs/3.2/capture.pcap /labs/3.2/analysis.bro –C*
  - *-C* tells Zeek to ignore packets which don't have valid checksums (see notes)

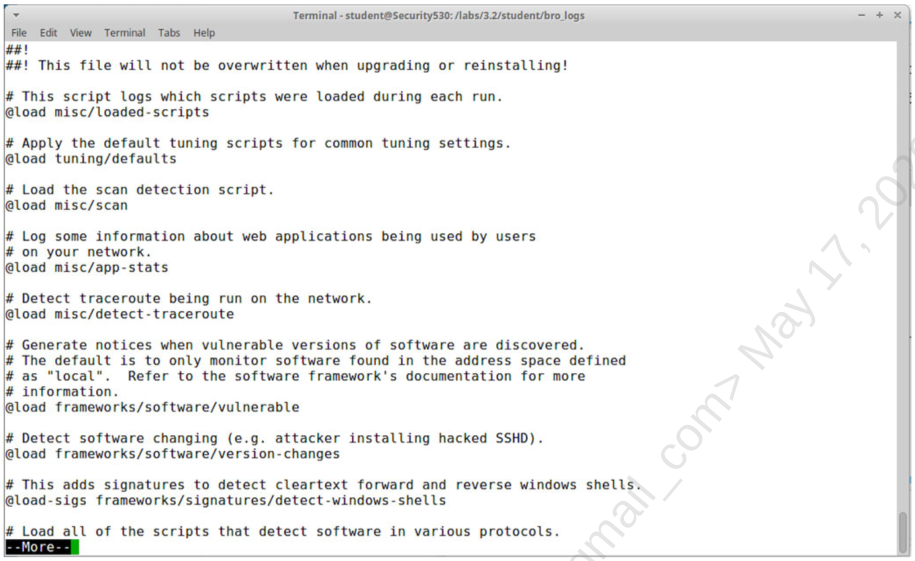
SANS SEC530 | Defensible Security Architecture and Engineering 49

Zeek as a Command Line Utility

Normally, Zeek's event engine will discard packets which don't have valid checksums. This can be a problem if one wants to analyze locally generated/captured traffic on a system that offloads checksumming to the network adapter. In that case, all transmitted/captured packets will have bad checksums because they haven't yet been calculated by the NIC. Thus such packets will not undergo analysis defined in Zeek policy scripts as they normally would. Bad checksums in traces may also be a result of some packet alteration tools.

By using the –C option, Zeek will ignore bad checksums.

Loading Zeek[1] Scripts - A Peek of analysis.bro


Terminal - student@Security530: /labs/3.2/student/bro_logs
File Edit View Terminal Tabs Help

```
##!
##! This file will not be overwritten when upgrading or reinstalling!

# This script logs which scripts were loaded during each run.
@load misc/loaded-scripts

# Apply the default tuning scripts for common tuning settings.
@load tuning/defaults

# Load the scan detection script.
@load misc/scan

# Log some information about web applications being used by users
# on your network.
@load misc/app-stats

# Detect traceroute being run on the network.
@load misc/detect-traceroute

# Generate notices when vulnerable versions of software are discovered.
# The default is to only monitor software found in the address space defined
# as "local".  Refer to the software framework's documentation for more
# information.
@load frameworks/software/vulnerable

# Detect software changing (e.g. attacker installing hacked SSHD).
@load frameworks/software/version-changes

# This adds signatures to detect cleartext forward and reverse windows shells.
@load-sigs frameworks/signatures/detect-windows-shells

# Load all of the scripts that detect software in various protocols.
--More--
```

Loading Zeek[1] Scripts - A Peek of analysis.bro

Virtually all of the output generated by Zeek is, in fact, generated by Zeek scripts. Zee scripts effectively notify Zeek that should there be an event of a type we define, then let us have the information about the connection so we can perform some function on it.

More information on Zeek scripts can be found here:

[1] https://docs.zeek.org/en/stable/examples/scripting/

These are the contents of the file analysis.bro found on your SEC530 Virtual Machine under **/labs/3.2/analysis.bro**

##! Local site policy. Customize as appropriate.
##!
##! This file will not be overwritten when upgrading or reinstalling!

This script logs which scripts were loaded during each run.
@load misc/loaded-scripts

Apply the default tuning scripts for common tuning settings.
@load tuning/defaults

Load the scan detection script.
@load misc/scan

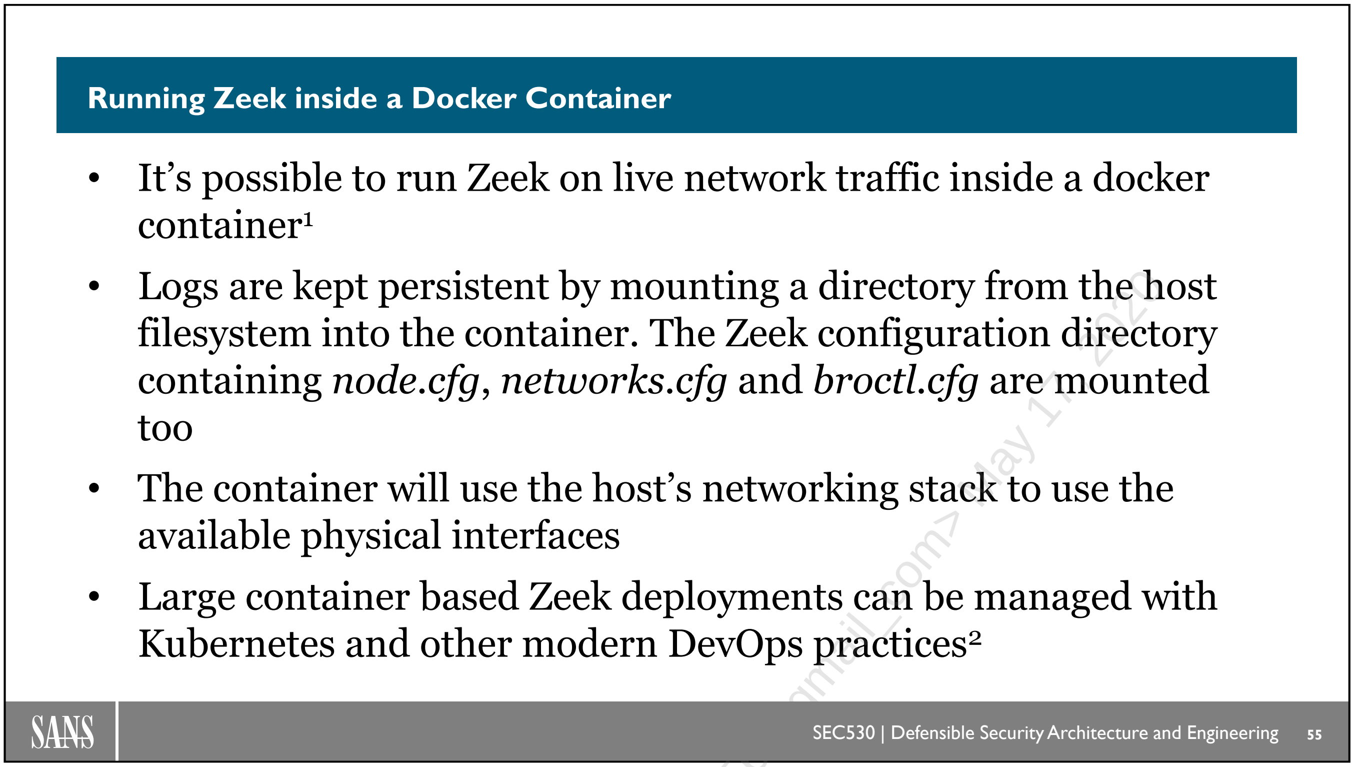
Log some information about web applications being used by users
on your network.
@load misc/app-stats

```
# Detect traceroute being run on the network.
@load misc/detect-traceroute

# Generate notices when vulnerable versions of software are discovered.
# The default is to only monitor software found in the address space defined
# as "local".  Refer to the software framework's documentation for more
# information.
@load frameworks/software/vulnerable

# Detect software changing (e.g. attacker installing hacked SSHD).
@load frameworks/software/version-changes

# This adds signatures to detect cleartext forward and reverse windows shells.
@load-sigs frameworks/signatures/detect-windows-shells

# Load all of the scripts that detect software in various protocols.
@load protocols/ftp/software
@load protocols/smtp/software
@load protocols/ssh/software
@load protocols/http/software
# The detect-webapps script could possibly cause performance trouble when
# running on live traffic.  Enable it cautiously.
#@load protocols/http/detect-webapps

# This script detects DNS results pointing toward your Site::local_nets
# where the name is not part of your local DNS zone and is being hosted
# externally.  Requires that the Site::local_zones variable is defined.
@load protocols/dns/detect-external-names

# Script to detect various activity in FTP sessions.
@load protocols/ftp/detect

# Scripts that do asset tracking.
@load protocols/conn/known-hosts
@load protocols/conn/known-services
@load protocols/ssl/known-certs

# This script enables SSL/TLS certificate validation.
@load protocols/ssl/validate-certs

# This script prevents the logging of SSL CA certificates in x509.log
@load protocols/ssl/log-hostcerts-only

# Uncomment the following line to check each SSL certificate hash against the IC
SI
# certificate notary service; see http://notary.icsi.berkeley.edu .
# @load protocols/ssl/notary

# If you have libGeoIP support built in, do some geographic detections and
# logging for SSH traffic.
@load protocols/ssh/geo-data
# Detect hosts doing SSH bruteforce attacks.
@load protocols/ssh/detect-bruteforcing
# Detect logins using "interesting" hostnames.
@load protocols/ssh/interesting-hostnames
```

```
# Detect SQL injection attacks.
@load protocols/http/detect-sqli

#### Network File Handling ####

# Enable MD5 and SHA1 hashing for all files.
@load frameworks/files/hash-all-files

# Detect SHA1 sums in Team Cymru's Malware Hash Registry.
@load frameworks/files/detect-MHR

# Uncomment the following line to enable detection of the heartbleed attack. Ena
bling
# this might impact performance a bit.
# @load policy/protocols/ssl/heartbleed

# Turn on JSON logs
@load policy/tuning/json-logs

# File Extraction
@load /labs/3.2/bro_modules/file-extraction

redef FTP::default_capture_password = T;
redef HTTP::default_capture_password = T;
[/labs/3.2/student/bro_logs]$ bro -r /home/student/dnstunnel.pcap /labs/3.2/analysis.bro -C^C
[/labs/3.2/student/bro_logs]$ less analysis.bro
analysis.bro: No such file or directory
[/labs/3.2/student/bro_logs]$ cat /labs/3.2/analysis.bro | more
##! Local site policy. Customize as appropriate.
##!
##! This file will not be overwritten when upgrading or reinstalling!

# This script logs which scripts were loaded during each run.
@load misc/loaded-scripts

# Apply the default tuning scripts for common tuning settings.
@load tuning/defaults

# Load the scan detection script.
@load misc/scan

# Log some information about web applications being used by users
# on your network.
@load misc/app-stats

# Detect traceroute being run on the network.
@load misc/detect-traceroute

# Generate notices when vulnerable versions of software are discovered.
# The default is to only monitor software found in the address space defined
# as "local".  Refer to the software framework's documentation for more
# information.
@load frameworks/software/vulnerable

# Detect software changing (e.g. attacker installing hacked SSHD).
@load frameworks/software/version-changes
```

```
# This adds signatures to detect cleartext forward and reverse windows shells.
@load-sigs frameworks/signatures/detect-windows-shells

# Load all of the scripts that detect software in various protocols.
[/labs/3.2/student/bro_logs]$ cat /labs/3.2/analysis.bro
##! Local site policy. Customize as appropriate.
##!
##! This file will not be overwritten when upgrading or reinstalling!

# This script logs which scripts were loaded during each run.
@load misc/loaded-scripts

# Apply the default tuning scripts for common tuning settings.
@load tuning/defaults

# Load the scan detection script.
@load misc/scan

# Log some information about web applications being used by users
# on your network.
@load misc/app-stats

# Detect traceroute being run on the network.
@load misc/detect-traceroute

# Generate notices when vulnerable versions of software are discovered.
# The default is to only monitor software found in the address space defined
# as "local".  Refer to the software framework's documentation for more
# information.
@load frameworks/software/vulnerable

# Detect software changing (e.g. attacker installing hacked SSHD).
@load frameworks/software/version-changes

# This adds signatures to detect cleartext forward and reverse windows shells.
@load-sigs frameworks/signatures/detect-windows-shells

# Load all of the scripts that detect software in various protocols.
@load protocols/ftp/software
@load protocols/smtp/software
@load protocols/ssh/software
@load protocols/http/software
# The detect-webapps script could possibly cause performance trouble when
# running on live traffic.  Enable it cautiously.
#@load protocols/http/detect-webapps

# This script detects DNS results pointing toward your Site::local_nets
# where the name is not part of your local DNS zone and is being hosted
# externally.  Requires that the Site::local_zones variable is defined.
@load protocols/dns/detect-external-names

# Script to detect various activity in FTP sessions.
@load protocols/ftp/detect
```

```
# Scripts that do asset tracking.
@load protocols/conn/known-hosts
@load protocols/conn/known-services
@load protocols/ssl/known-certs

# This script enables SSL/TLS certificate validation.
@load protocols/ssl/validate-certs

# This script prevents the logging of SSL CA certificates in x509.log
@load protocols/ssl/log-hostcerts-only

# Uncomment the following line to check each SSL certificate hash against the ICSI
# certificate notary service; see http://notary.icsi.berkeley.edu .
# @load protocols/ssl/notary

# If you have libGeoIP support built in, do some geographic detections and
# logging for SSH traffic.
@load protocols/ssh/geo-data
# Detect hosts doing SSH bruteforce attacks.
@load protocols/ssh/detect-bruteforcing
# Detect logins using "interesting" hostnames.
@load protocols/ssh/interesting-hostnames

# Detect SQL injection attacks.
@load protocols/http/detect-sqli

#### Network File Handling ####

# Enable MD5 and SHA1 hashing for all files.
@load frameworks/files/hash-all-files

# Detect SHA1 sums in Team Cymru's Malware Hash Registry.
@load frameworks/files/detect-MHR

# Uncomment the following line to enable detection of the heartbleed attack. Enabling
# this might impact performance a bit.
# @load policy/protocols/ssl/heartbleed

# Turn on JSON logs
@load policy/tuning/json-logs

# File Extraction
@load /labs/3.2/bro_modules/file-extraction

redef FTP::default_capture_password = T;
redef HTTP::default_capture_password = T;[/labs/3.2/student/bro_logs]$
```



Running Zeek inside a Docker Container

- It’s possible to run Zeek on live network traffic inside a docker container[1]
- Logs are kept persistent by mounting a directory from the host filesystem into the container. The Zeek configuration directory containing *node.cfg*, *networks.cfg* and *broctl.cfg* are mounted too
- The container will use the host’s networking stack to use the available physical interfaces
- Large container based Zeek deployments can be managed with Kubernetes and other modern DevOps practices[2]

Running Zeek inside a Docker Container
If you’re interested in running Zeek inside a Docker container on a single host, Draconyx has written a step by step guide describing how to build an image container and its dependencies:

This repository on GitHub contains a Dockerfile of Zeek too:

https://github.com/blacktop/docker-bro

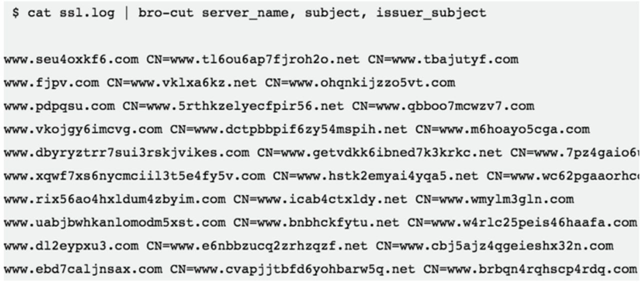
At BroCon2018, Daniel Lohin and Ed Sealing from Sealing Technologies shared their research on implementing Bro using modern DevOps tools including Docker, Kubernetes, and Jenkins.

If you’re looking at managing Bro inside of containers to efficiently process traffic and integration with other tools such as Elasticsearch using Kubernetes, and how to build a Continuous Integration and Continuous Development (CI/CD) pipeline using Jenkins, check out their presentation here:

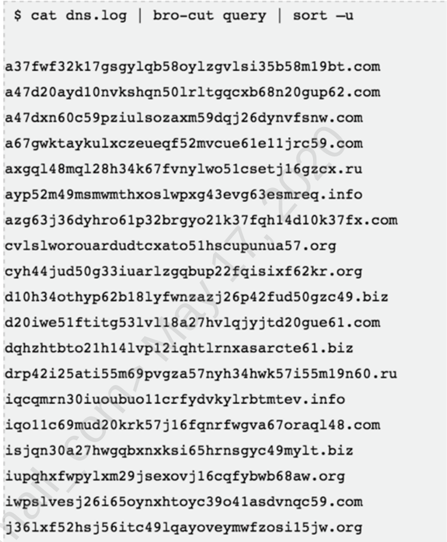
[1] http://www.draconyx.net/articles/running-bro-on-live-network-traffic-in-a-docker-container.html
[2] https://www.zeek.org/brocon2018/slides/Ed_and_Daniel._Managing_Bro_Deployments.pdf

Zeek Use Cases: Spotting the C2

- 'Pulling a thread' with X.509 certificates and DNS logs
- Samples available at
  - https://github.com/aboutsecurity/Bro-samples



```
$ cat ssl.log | bro-cut server_name, subject, issuer_subject

www.seu4oxkf6.com CN=www.tl6ou6ap7fjroh2o.net CN=www.tbajutyf.com
www.fjpv.com CN=www.vklxa6kz.net CN=www.ohqnkijzzo5vt.com
www.pdpqsu.com CN=www.5rthkzelyecfpir56.net CN=www.qbboo7mcwzv7.com
www.vkojgy6imcvg.com CN=www.dctpbbpif6zy54mspih.net CN=www.m6hoayo5cga.com
www.dbyryztrr7sui3rskjvikes.com CN=www.getvdkk6ibned7k3krkc.net CN=www.7pz4gaio6
www.xqwf7xs6nycmciil3t5e4fy5v.com CN=www.hstk2emyai4yqa5.net CN=www.wc62pgaaorhc
www.rix56ao4hxldum4zbyim.com CN=www.icab4ctxldy.net CN=www.wmylm3gln.com
www.uabjbwhkanlomodm5xst.com CN=www.bnbhckfytu.net CN=www.w4rlc25peis46haafa.com
www.dl2eypxu3.com CN=www.e6nbbzucq2zrhzqzf.net CN=www.cbj5ajz4qgeieshx32n.com
www.ebd7caljnsax.com CN=www.cvapjjtbfd6yohbarw5q.net CN=www.brbqn4rqhscp4rdq.com
```



```
$ cat dns.log | bro-cut query | sort –u

a37fwf32k17gsgylqb58oylzgvlsi35b58m19bt.com
a47d20ayd10nvkshqn50lrltgqcxb68n20gup62.com
a47dxn60c59pziulsozaxm59dqj26dynvfsnw.com
a67gwktaykulxczeueqf52mvcue61e11jrc59.com
axgql48mql28h34k67fvnylwo51csetj16gzcx.ru
ayp52m49msmwmthxoslwpxg43evg63esmreq.info
azg63j36dyhro61p32brgyo21k37fqh14d10k37fx.com
cvlslworouardudtcxato51hscupunua57.org
cyh44jud50g33iuarlzgqbup22fqisixf62kr.org
d10h34othyp62b18lyfwnzazj26p42fud50gzc49.biz
d20iwe51ftitg53lvl18a27hvlqjyjtd20gue61.com
dqhzhtbto21h14lvp12iqhtlrnxasarcte61.biz
drp42i25ati55m69pvgza57nyh34hwk57i55m19n60.ru
iqcqmrn30iuoubuo11crfydvkylrbtmtev.info
iqo11c69mud20krk57j16fqnrfwgva67oraql48.com
isjqn30a27hwgqbxnxksi65hrnsgyc49mylt.biz
iupqhxfwpylxm29jsexovj16cqfybwb68aw.org
iwpslvesj26i65oynxhtoyc39o41asdvnqc59.com
j361xf52hsj56itc49lqayoveymwfzosi15jw.org
```

Zeek Use Cases: Spotting the C2

Let's go through some use cases where network metadata can provide tremendous value at a very little cost. Some of the traffic shown above was captured by one of the course authors during a threat hunting exercise conducted in the Middle East. After collecting full packet captures for a few hours, the traffic was analyzed with Zeek on the author's laptop. A couple of command lines like the ones shown above were enough to uncover a network infected with the Zeus Gameover variant.

Looking at the length of the domains requested (right image) it's possible to observe a pattern. To do so, we can cut out the TLDs (com, info, net…) and then calculate the length of each of the strings.

$ cat dns.log | bro-cut query | sort -u | cut -d . -f1 > domains-withoutTLD $ for i in `cat domains-withoutTLD`; do echo "${#i}"; done | sort –u

34
35
36
37
38
39
40
41
42
43

These strings are within a close range of 34 to 43 characters long. Casualty? Not really, the so-called ZeuS Gameover is known for implementing P2P and Domain Generation Algorithm (DGA) communications to determine the current Command and Control (C&C) domain. When these bots can't communicate with its botnet via P2P, DGA is used. The domain names generated by ZeuS Gameover consist of a string with a length of 32 to 48 chars and one of the following TLDs: ru, com, biz, net or org. The list contains over 1000 domains and changes every 7 days, based on the current date.

Again, parsing SSL or TLS traffic with Zeek (left image) provides us with metadata that shows X.509 certificate fields like server_name, subject and issuer_subject filled with random strings, a common tactic employed by malware authors to evade signature detection.

Bro_cut used in both screenshots, extracts specific columns from the ASCII based logs produced by Zeek on standard input.

You can download these sample pcaps from one from this repository:

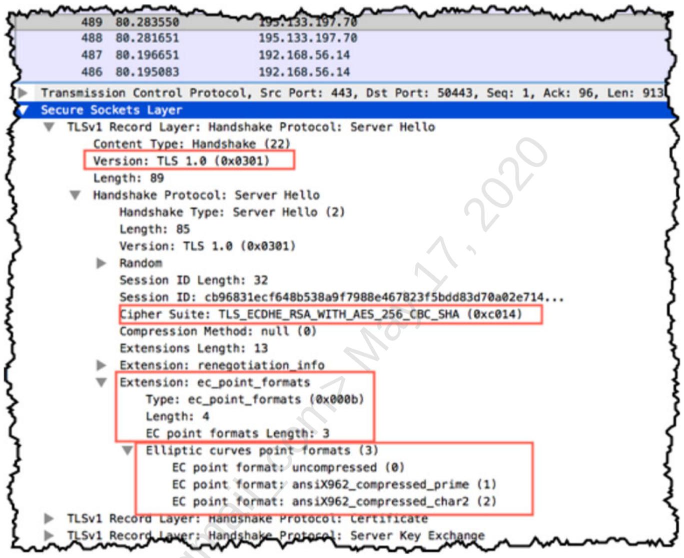
https://github.com/aboutsecurity/Bro-samples

A full write-up of this analysis is available here:

http://blog.opensecurityresearch.com/2014/03/identifying-malware-traffic-with-bro.html

Zeek Use Cases: TLS Magic with JA3 (1)

- JA3 is a technique for creating SSL client fingerprints from the pre-encryption handshakes of the SSL protocol
- Malicious self-signed certificates often have unique JA3 fingerprints
- They can be matched against known bad JA3 fingerprints too



```
      489  80.283550          195.133.197.70
      488  80.281651          195.133.197.70
      487  80.196651          192.168.56.14
      486  80.195083          192.168.56.14
▶ Transmission Control Protocol, Src Port: 443, Dst Port: 50443, Seq: 1, Ack: 96, Len: 913
▼ Secure Sockets Layer
   ▼ TLSv1 Record Layer: Handshake Protocol: Server Hello
        Content Type: Handshake (22)
        Version: TLS 1.0 (0x0301)
        Length: 89
      ▼ Handshake Protocol: Server Hello
           Handshake Type: Server Hello (2)
           Length: 85
           Version: TLS 1.0 (0x0301)
         ▶ Random
           Session ID Length: 32
           Session ID: cb96831ecf648b538a9f7988e467823f5bdd83d70a02e714...
           Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
           Compression Method: null (0)
           Extensions Length: 13
         ▶ Extension: renegotiation_info
         ▼ Extension: ec_point_formats
              Type: ec_point_formats (0x000b)
              Length: 4
              EC point formats Length: 3
            ▼ Elliptic curves point formats (3)
                 EC point format: uncompressed (0)
                 EC point format: ansiX962_compressed_prime (1)
                 EC point format: ansiX962_compressed_char2 (2)
   ▶ TLSv1 Record Layer: Handshake Protocol: Certificate
   ▶ TLSv1 Record Layer: Handshake Protocol: Server Key Exchange
```

Zeek Use Cases: TLS Magic with JA3 (1)

John B. Althouse, Jeff Atkinson, and Josh Atkins, developed JA3 (a tool that named after their shared initials ”JA”), a technique for creating SSL client fingerprints from the pre-encryption handshakes of the SSL protocol.

JA3 fingerprints can be generated with either a Zeek script or customizing an instance of Suricata.

As previously discussed, SSL certificate values are often created or changed by malicious actors, usually with “self-signed” certificates that will have unique JA3 fingerprint values. By bringing those values into a SIEM or other analysis tools, you can hunt for unusual JA3 fingerprints or even match them against a threat list of known-bad JA3 fingerprints.

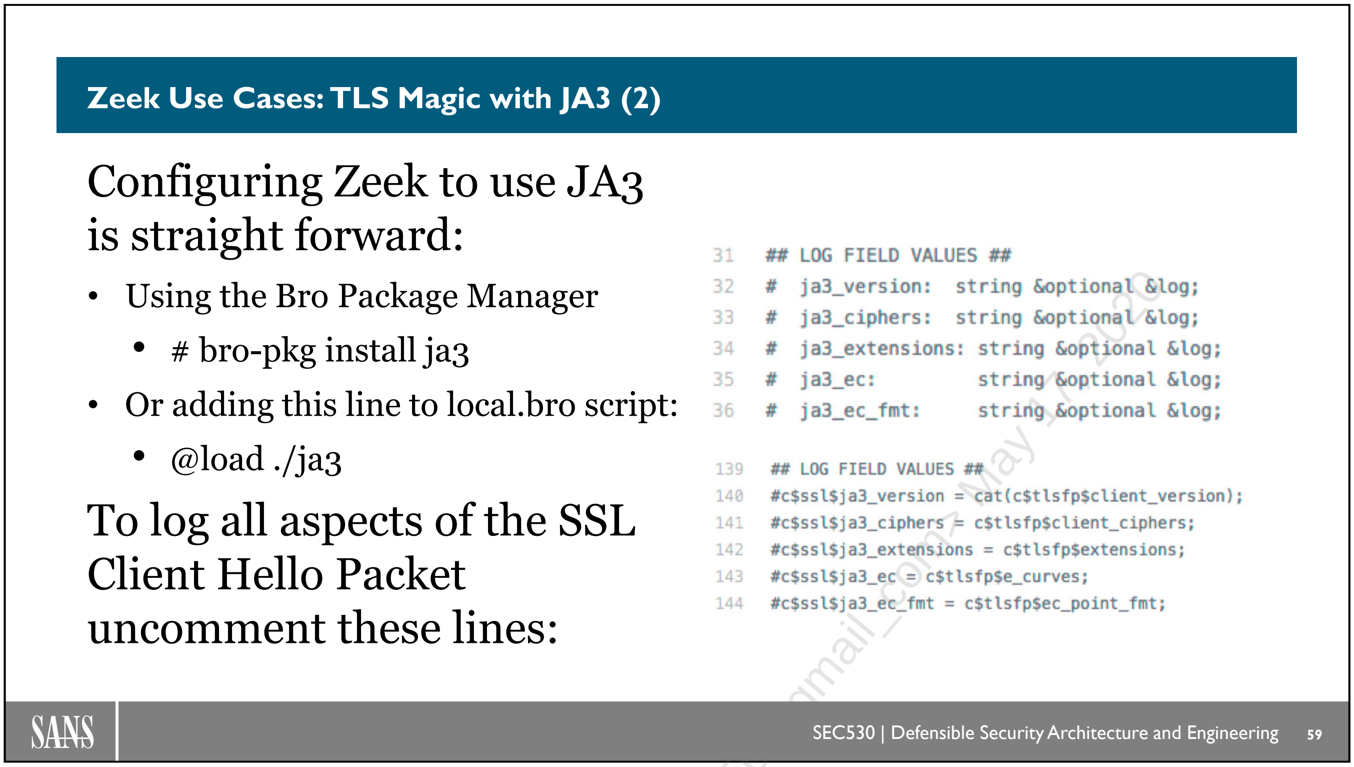
Notice how the fields highlighted in red above are the fields that Bro (and Suricata) use to create the JA3 signature hashes. The JA3 Bro script then takes those fields and concatenates them with fields separated by a comma and values within the field, separated by a hyphen. That string is hashed with md5 which results in the ja3 SSL client fingerprint. The results are output to the bro_ssl log file.

The JA3 code is hosted in https://github.com/salesforce/ja3

There is a list of JA3 hashes and clients that they've been associated with, available here https://github.com/trisulnsm/trisul-scripts/blob/master/lua/frontend_scripts/reassembly/ja3/prints/ja3fingerprint.json

Steve Brant from Splunk has a great blog post on how to leverage JA3 with Zeek and SIEM here:

https://www.splunk.com/blog/2017/12/18/configuring-ja3-with-bro-for-splunk.html



Zeek Use Cases: TLS Magic with JA3 (2)

Configuring Zeek to use JA3 is straight forward:

- Using the Bro Package Manager
  - # bro-pkg install ja3
- Or adding this line to local.bro script:
  - @load ./ja3

To log all aspects of the SSL Client Hello Packet uncomment these lines:

```
## LOG FIELD VALUES ##
#  ja3_version:  string &optional &log;
#  ja3_ciphers:  string &optional &log;
#  ja3_extensions: string &optional &log;
#  ja3_ec:         string &optional &log;
#  ja3_ec_fmt:     string &optional &log;

## LOG FIELD VALUES ##
#c$ssl$ja3_version = cat(c$tlsfp$client_version);
#c$ssl$ja3_ciphers = c$tlsfp$client_ciphers;
#c$ssl$ja3_extensions = c$tlsfp$extensions;
#c$ssl$ja3_ec = c$tlsfp$e_curves;
#c$ssl$ja3_ec_fmt = c$tlsfp$ec_point_fmt;
```

SANS

SEC530 | Defensible Security Architecture and Engineering 59

Zeek Use Cases: TLS Magic with JA3 (2)
Utilizing the JA3 Bro config is reasonably straightforward. If you're running Bro >=2.5 or a Bro product like Corelight, you can install by using the Bro Package Manager and this one simple command:

# bro-pkg install ja3

For everyone else, download the files to bro/share/bro/site/ja3 and add this line to your local.bro script:

@load ./ja3

By default ja3.bro will only append JA3 to the ssl.log. However, if you would like to log all aspects of the SSL Client Hello Packet, uncomment the following lines in ja3.bro

```
# ja3_version: string &optional &log;
# ja3_ciphers: string &optional &log;
# ja3_extensions: string &optional &log;
# ja3_ec: string &optional &log;
# ja3_ec_fmt: string &optional &log;
...
#c$ssl$ja3_version = cat(c$tlsfp$client_version);
#c$ssl$ja3_ciphers = c$tlsfp$client_ciphers;
#c$ssl$ja3_extensions = c$tlsfp$extensions; #c$ssl$ja3_ec = c$tlsfp$e_curves;
#c$ssl$ja3_ec_fmt = c$tlsfp$ec_point_fmt;
```

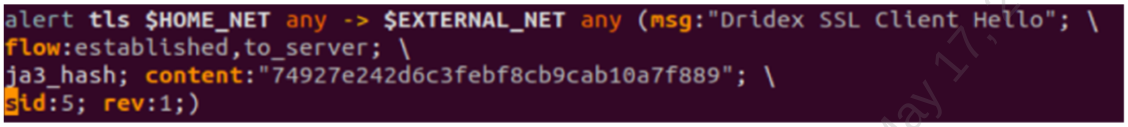
The same changes can be made in ja3s.bro as well.

More information can be found on the JA3 GitHub: https://github.com/salesforce/ja3/tree/master/bro

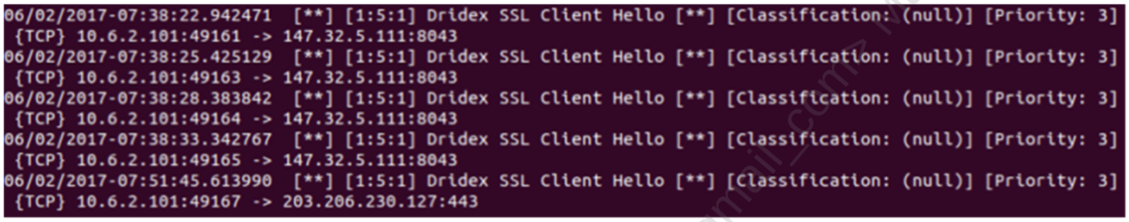
A good example of how to use

Suricata & TLS Magic with JA3

- Suricata can also make good use of network metadata (in this case JA3 hashes) even with a portless signature
- Detects Dridex[1] SSL Client Hello on a non-conventional TLS port



```
alert tls $HOME_NET any -> $EXTERNAL_NET any (msg:"Dridex SSL Client Hello"; \
flow:established,to_server; \
ja3_hash; content:"74927e242d6c3febf8cb9cab10a7f889"; \
sid:5; rev:1;)
```

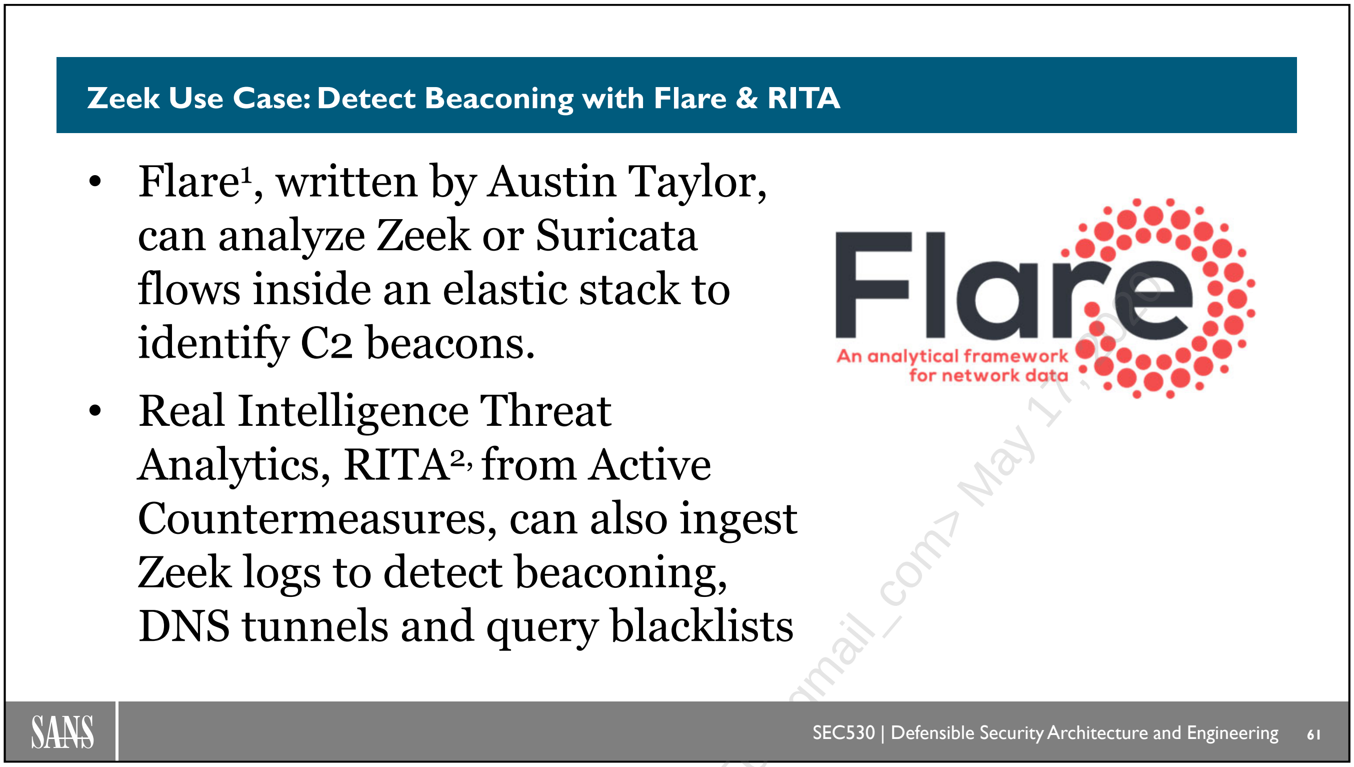


```
06/02/2017-07:38:22.942471  [**] [1:5:1] Dridex SSL Client Hello [**] [Classification: (null)] [Priority: 3]
 {TCP} 10.6.2.101:49161 -> 147.32.5.111:8043
06/02/2017-07:38:25.425129  [**] [1:5:1] Dridex SSL Client Hello [**] [Classification: (null)] [Priority: 3]
 {TCP} 10.6.2.101:49163 -> 147.32.5.111:8043
06/02/2017-07:38:28.383842  [**] [1:5:1] Dridex SSL Client Hello [**] [Classification: (null)] [Priority: 3]
 {TCP} 10.6.2.101:49164 -> 147.32.5.111:8043
06/02/2017-07:38:33.342767  [**] [1:5:1] Dridex SSL Client Hello [**] [Classification: (null)] [Priority: 3]
 {TCP} 10.6.2.101:49165 -> 147.32.5.111:8043
06/02/2017-07:51:45.613990  [**] [1:5:1] Dridex SSL Client Hello [**] [Classification: (null)] [Priority: 3]
 {TCP} 10.6.2.101:49167 -> 203.206.230.127:443
```

Suricata & TLS magic with JA3

As mentioned earlier, Suricata can also leverage the JA3 technique. As documented in this blog post by Trustwave, the researcher was able to spot the Dridex malware C2 with a portless signature. Notice how no ports are specified in the rule, yet Suricata is able to identify the malicious traffic going out on a non-conventional TLS port by the JA3 hash.

[1] https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/inspecting-encrypted-network-traffic-with-ja3/



Zeek Use Case: Detect Beaconing with Flare & RITA

- Flare[1], written by Austin Taylor, can analyze Zeek or Suricata flows inside an elastic stack to identify C2 beacons.
- Real Intelligence Threat Analytics, RITA[2], from Active Countermeasures, can also ingest Zeek logs to detect beaconing, DNS tunnels and query blacklists

Flare
An analytical framework
for network data

SANS

SEC530 | Defensible Security Architecture and Engineering 61

Zeek Use Case: Detect Beaconing with Flare & RITA
Flare is a free open-source cyber analytic framework intended to make identifying malicious behavior, such as C2 beaconing in networks, as simple as possible, written by SANS Instructor Austin Taylor.

Written in Python, it is designed for rapid prototyping and development of behavioral analytics leveraging Elasticsearch and any IDS system like Bro, Snort or Suricata to identify periodic activity. The flow data inside Elastic Stack (ES) is what Flare uses to identify beacons.

After you have created a small subset of interesting traffic to investigate, you can pivot to Kibana to visualize your findings. A full write up with an example is available at:

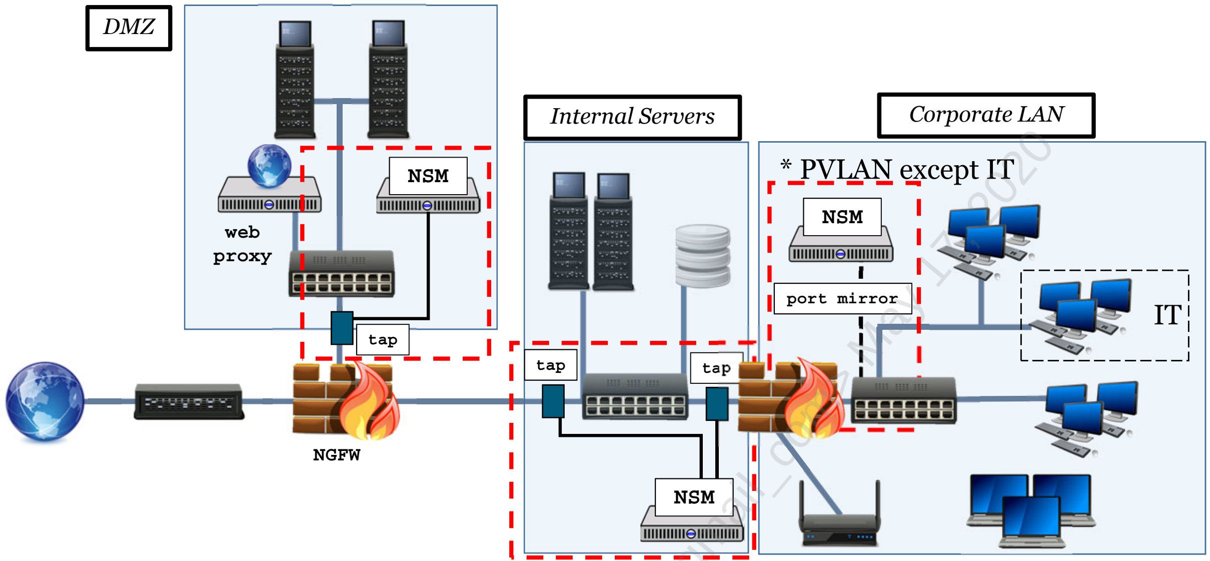
RITA is a similar project from Active countermeasures, led by SANS Instructor John Strand, that has interesting features like such as:

- **Beaconing Detection**: Search for signs of beaconing behavior in and out of your network
- **DNS Tunneling Detection** Search for signs of DNS based covert channels
- **Blacklist Checking**: Query blacklists to search for suspicious domains and hosts

A full description of RITA's capabilities and the code is available here:
https://github.com/activecm/rita/releases/tag/v1.0.1

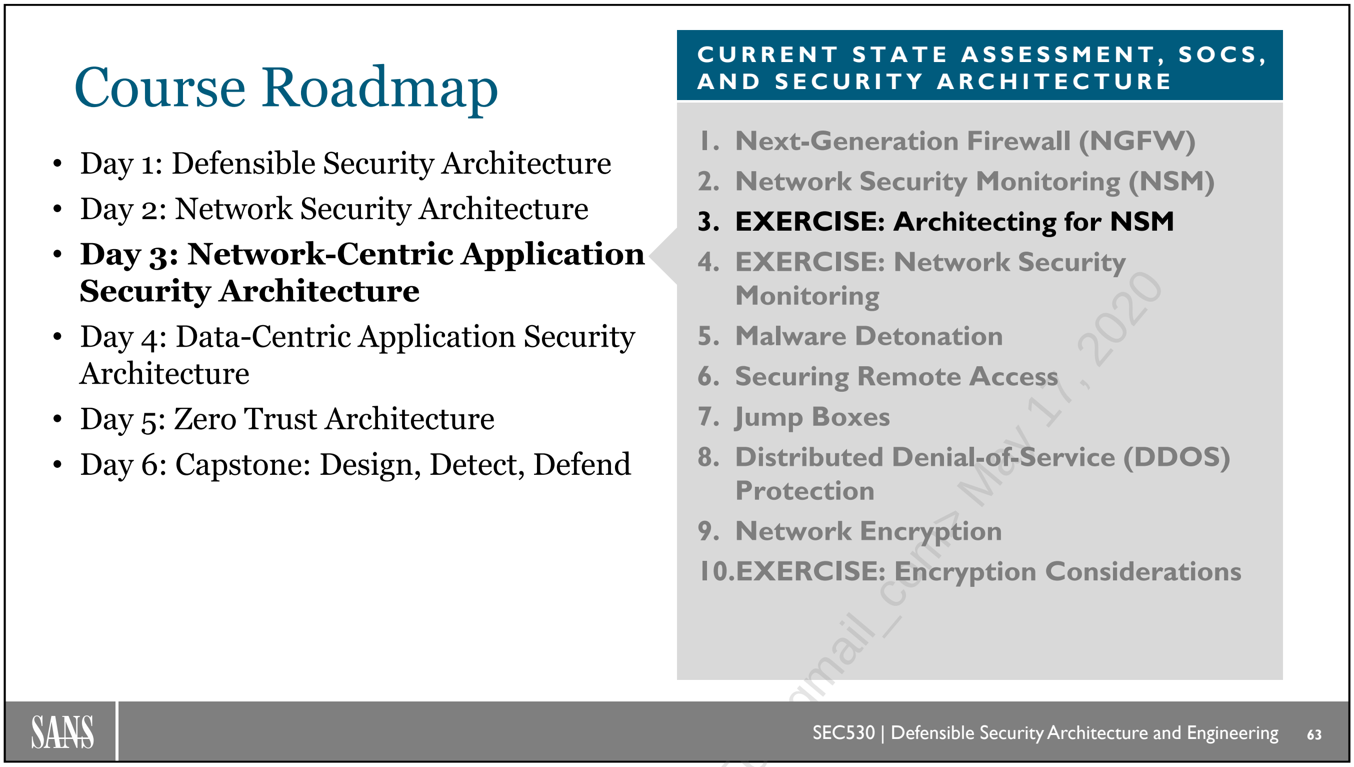
[1] http://www.austintaylor.io/detect/beaconing/intrusion/detection/system/command/control/flare/elastic/stack/2017/06/10/detect-beaconing-with-flare-elasticsearch-and-intrusion-detection-systems/
[2] https://github.com/austin-taylor/flare
[3] https://www.blackhillsinfosec.com/projects/rita/

Case Study: Tyrrell Corporation



Case Study: Tyrrell Corporation

This diagram shows the addition of multiple Network Security Monitor sensors to the Tyrrell Corporation architecture. In this example, a sensor is in the DMZ with a tap sending a copy of the traffic to it. Another sensor is within the internal server zone using network taps to receive traffic. Also, a sensor is in the corporate LAN zone with network traffic being received from a port mirror.



Course Roadmap

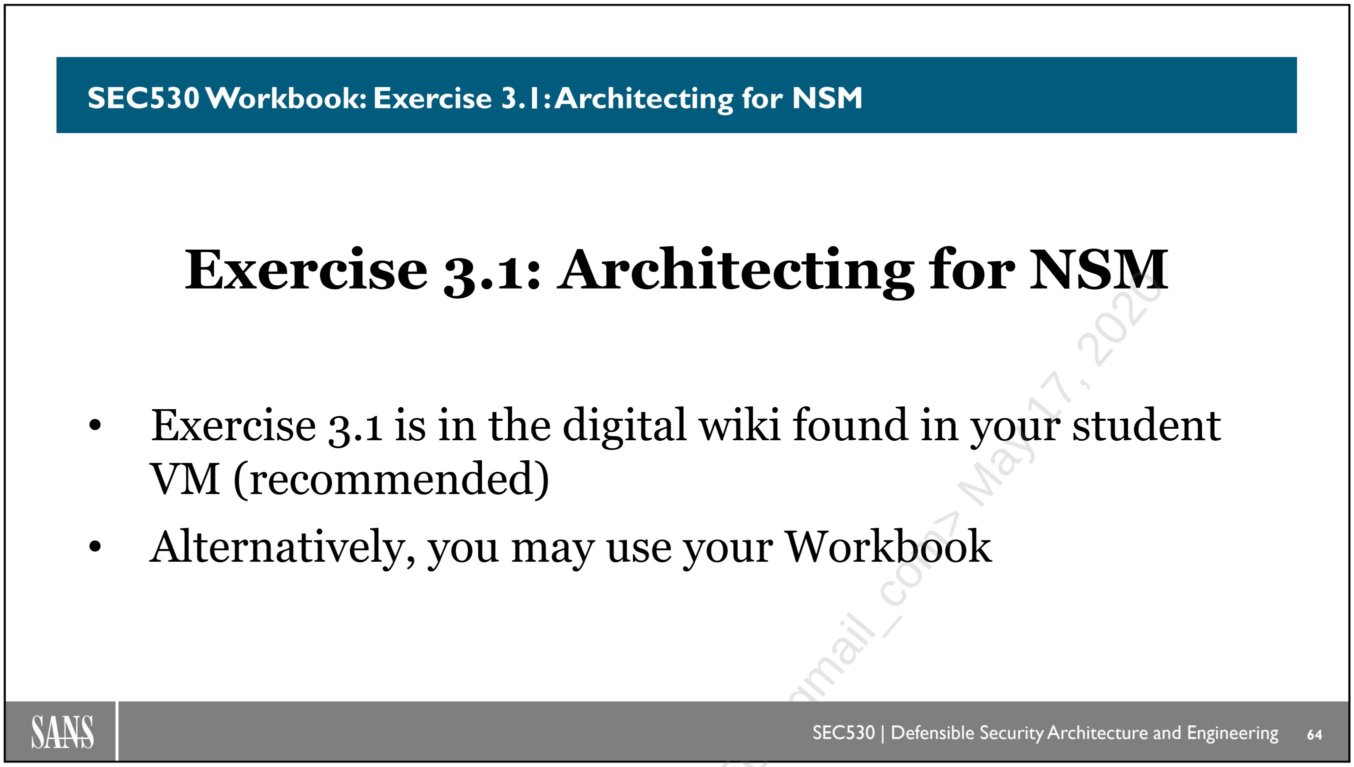
- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. **EXERCISE: Architecting for NSM**
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

Course Roadmap

We will now perform a lab on architecting for network security monitoring.



SEC530 Workbook: Exercise 3.1: Architecting for NSM

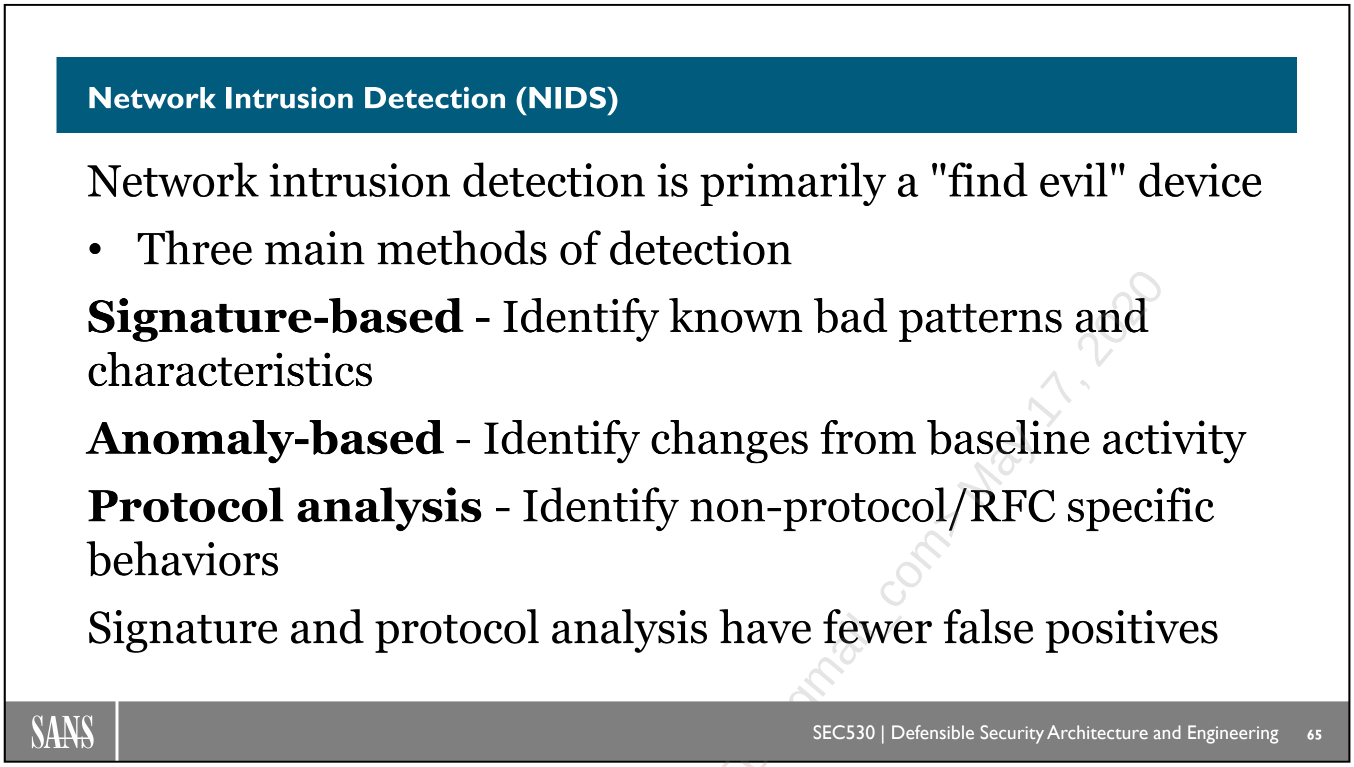
Exercise 3.1: Architecting for NSM

- Exercise 3.1 is in the digital wiki found in your student VM (recommended)
- Alternatively, you may use your Workbook

SANS

SEC530 | Defensible Security Architecture and Engineering 64

Please open the digital wiki or your lab workbook and access exercise 3.1.



Network Intrusion Detection (NIDS)

Network intrusion detection is primarily a "find evil" device

- Three main methods of detection

Signature-based - Identify known bad patterns and characteristics

Anomaly-based - Identify changes from baseline activity

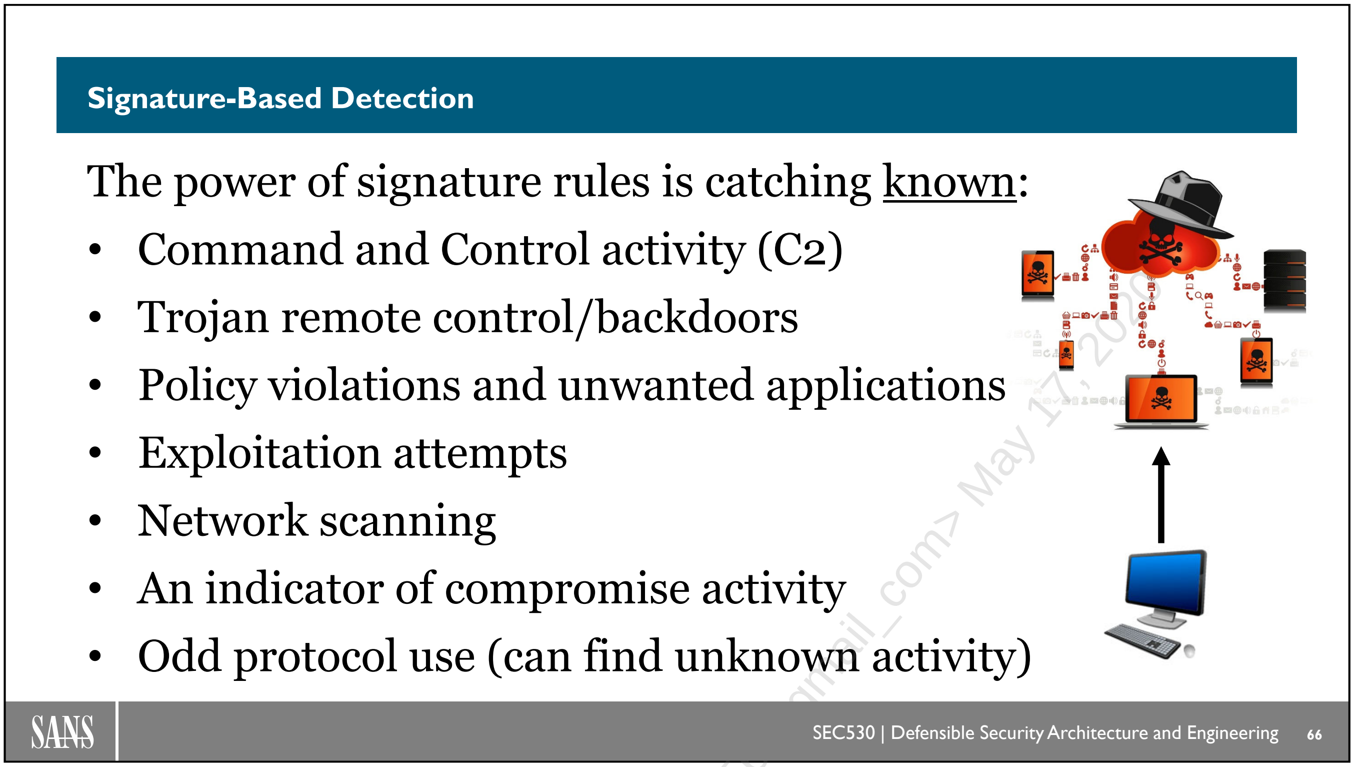
Protocol analysis - Identify non-protocol/RFC specific behaviors

Signature and protocol analysis have fewer false positives

Network Intrusion Detection (NIDS)

The most common network sensor is a network intrusion detection sensor or NIDS. NIDS is purpose-built to find and alert the presence of blacklisted payloads or anomalies. Anomaly detection in an IDS is usually in the form of protocols not conforming to common behaviors or RFC requirements. Anomaly detection can also be from automatic baselining or statistical calculations, but historically automatic baselining has not been mature. Instead, the core of NIDS is signature-based rules.

Rules in a NIDS specifically look for network connections and specific layer-7 payload information. If a match is found, a NIDS reacts. Most of the time an alert is generated but a NIDS can also block traffic or kick off short-term packet captures to gather additional information.



Signature-Based Detection

The power of signature rules is catching <u>known</u>:

- Command and Control activity (C2)
- Trojan remote control/backdoors
- Policy violations and unwanted applications
- Exploitation attempts
- Network scanning
- An indicator of compromise activity
- Odd protocol use (can find unknown activity)

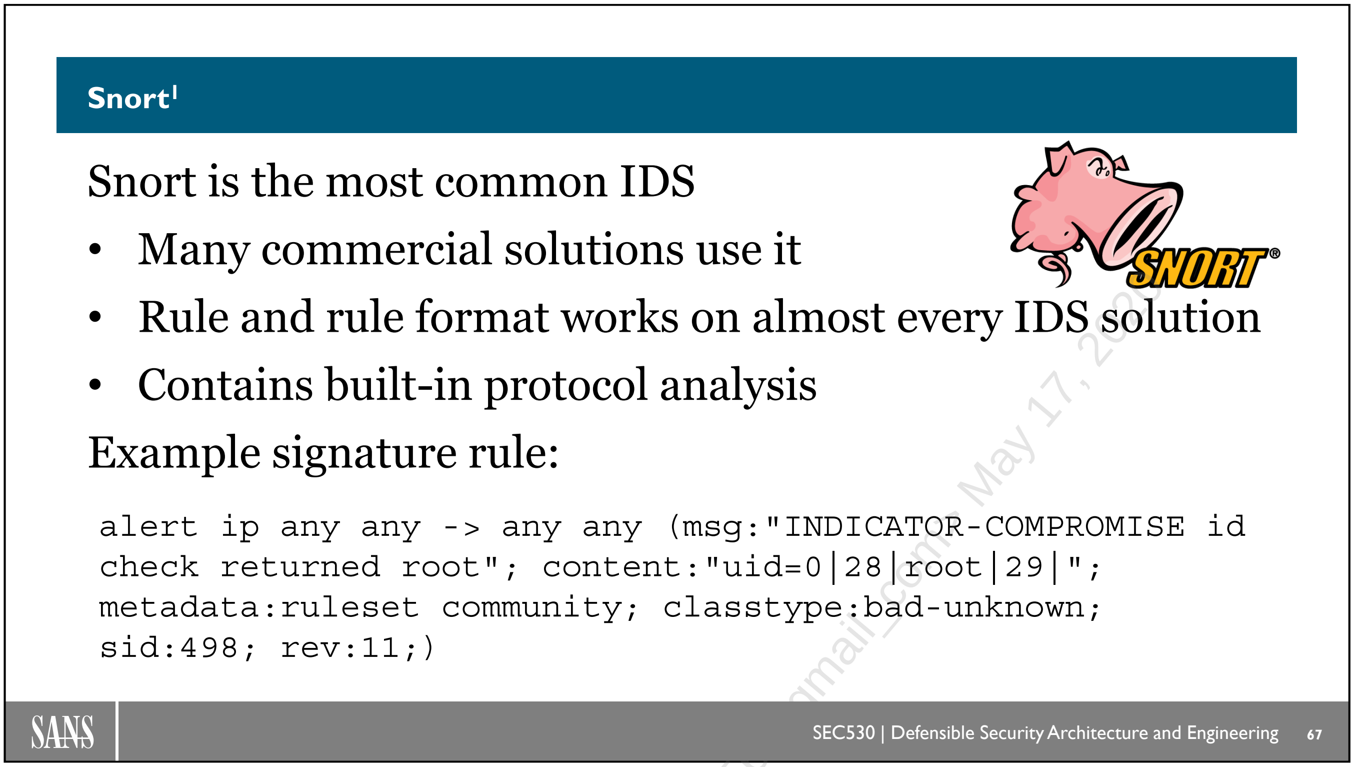
SANS

SEC530 | Defensible Security Architecture and Engineering 66

Signature-Based Detection

NIDS comes pre-packaged with many signatures. These signatures fall within specific functional categories. For example, POLICY specifies a signature is identifying a potential corporate policy violation such as Dropbox use. Some signatures may cause a lot of false positives. Other signatures may be high fidelity and highly actionable. For example, C2 or INDICATOR-COMPROMISE signatures call out that a system is most likely compromised and either phoning home to a command and control server or showing signs of post compromise activity.

Knowing what rule categories are available and in use factors into a successful NIDS deployment. How these signatures are applied matters as well. For example, a network scanning rule triggering from an external IP is unauthorized but normal activity. The internet is an untrusted network and routinely attacks and scans organizations. However, network scanning internal on a network is more likely to be unauthorized and worthy of investigation.



Snort[1]

Snort is the most common IDS

- Many commercial solutions use it
- Rule and rule format works on almost every IDS solution
- Contains built-in protocol analysis

Example signature rule:

```
alert ip any any -> any any (msg:"INDICATOR-COMPROMISE id
check returned root"; content:"uid=0|28|root|29|";
metadata:ruleset community; classtype:bad-unknown;
sid:498; rev:11;)
```

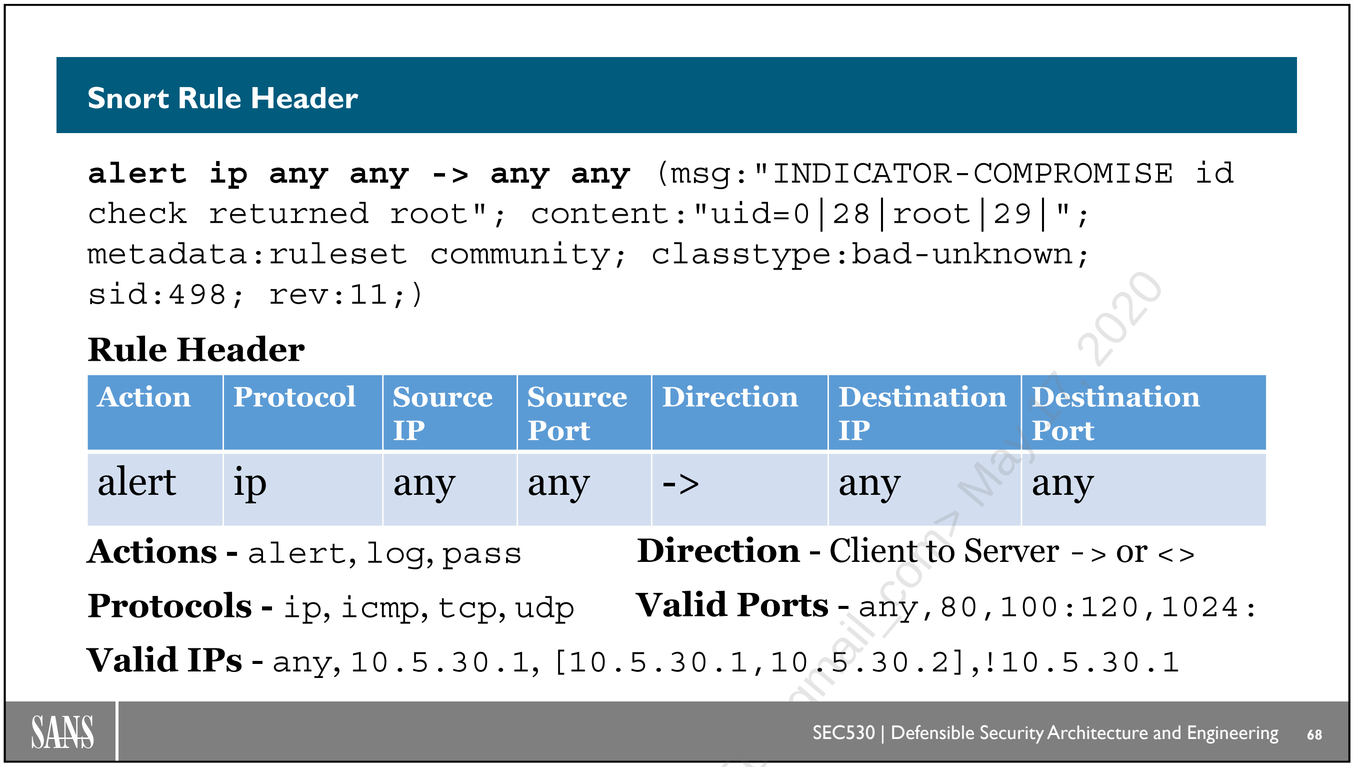
Snort[1]

Snort is the de facto NIDS available. Snort has long-running roots in the security community. It is used commercially as well as an open source offering. Both free rule subscriptions are available as well as commercial subscriptions[2]. Online blogs, documentation, and video tutorials are available. In some circles, stating you do not like Snort is a sure method to get someone to help you find the door.

The Snort rule format is accepted by most major IDS solutions, or there are converter utilities to translate from Snort syntax to commercial solution. In fact, many next-generation firewalls have such converter utilities because their intrusion prevention engines are similar enough to Snort that rules can be converted to or from a specific implementation. The Snort rule format will be covered at a high level over the next slides.

[1] https://www.snort.org/

[2] https://www.snort.org/products#rule_subscriptions



Snort Rule Header

```
alert ip any any -> any any (msg:"INDICATOR-COMPROMISE id
check returned root"; content:"uid=0|28|root|29|";
metadata:ruleset community; classtype:bad-unknown;
sid:498; rev:11;)
```

Rule Header

Action	Protocol	Source IP	Source Port	Direction	Destination IP	Destination Port
alert	ip	any	any	->	any	any

Actions - alert, log, pass

Direction - Client to Server -> or <>

Protocols - ip, icmp, tcp, udp

Valid Ports - any, 80, 100:120, 1024:

Valid IPs - any, 10.5.30.1, [10.5.30.1,10.5.30.2], !10.5.30.1

Snort Rule Header

The Snort rule header has seven parts as identified in the table in this slide. A description of these is below.

Actions - Actions tell Snort what to do if a match is found. Alert will generate an alert. The log will log the match but not create an alert. Pass tells Snort to ignore the match similar to a whitelist. There are also special actions available such as activate and dynamic that provide support for rule chaining.

Protocols - Protocols tell snort what network protocol to analyze.

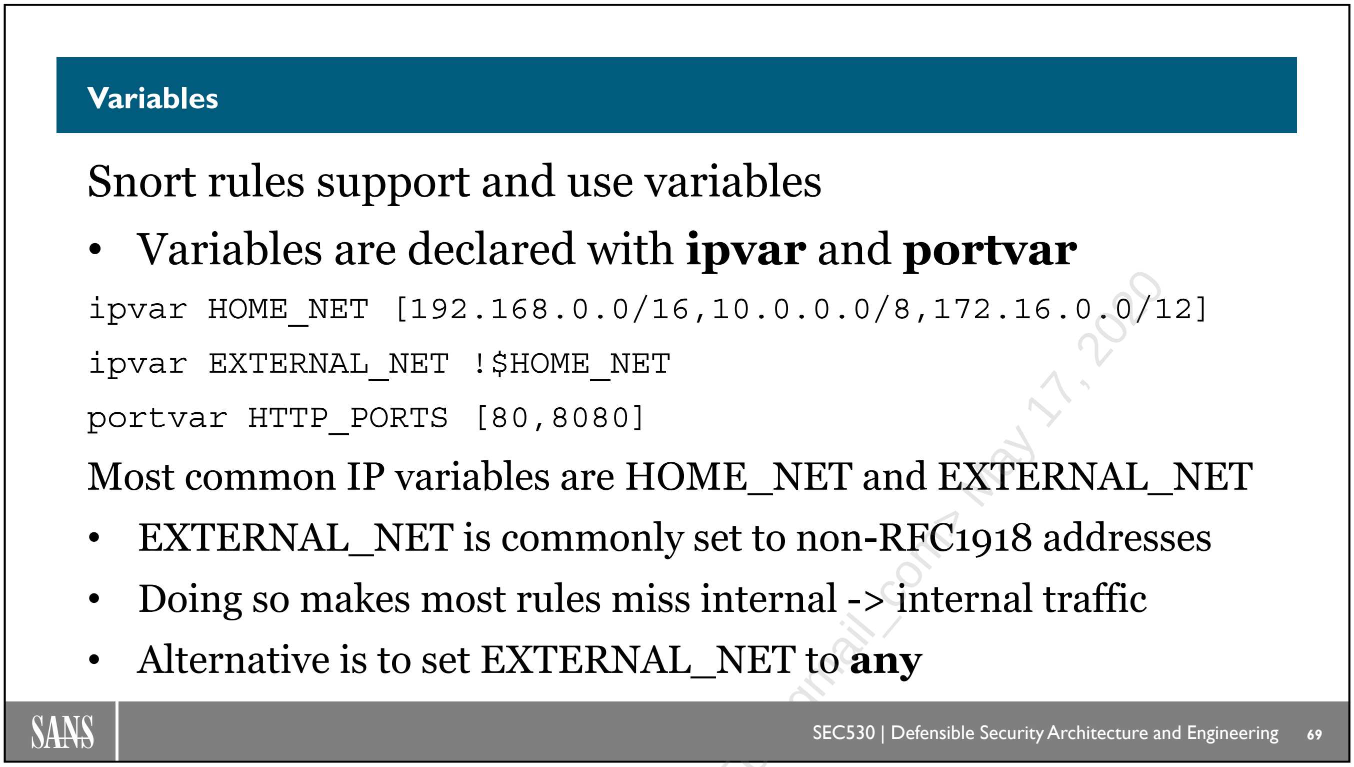
Source IP - The Source IP specifies what IP address, addresses, or subnets to include or exclude for analysis in regard to source addresses. The exclamation mark specifies something should be excluded. The use of square brackets allows for multiple addresses to be set.

Source Port - The Source Port specifies what port, ports, or range of ports a rule should include or exclude in regard to source ports. A colon can be used to specify a range or to include all ports above a given port number.

Direction - Direction specifies if Snort should analyze traffic going from the source IP addresses to the destination IP addresses or analyze based on any combination of source or destination addresses. Snort does not support <-

Destination IP - The Destination IP specifies what IP address, addresses, or subnets to include or exclude for analysis in regard to destination addresses. The syntax is the same as Source IP.

Destination Port - The Destination Port specifies what port, ports, or range of ports a rule should include or exclude in regard to destination ports. The syntax is the same as Source Port.



Variables

Snort rules support and use variables

- Variables are declared with **ipvar** and **portvar**

```
ipvar HOME_NET [192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]
ipvar EXTERNAL_NET !$HOME_NET
portvar HTTP_PORTS [80,8080]
```

Most common IP variables are HOME_NET and EXTERNAL_NET

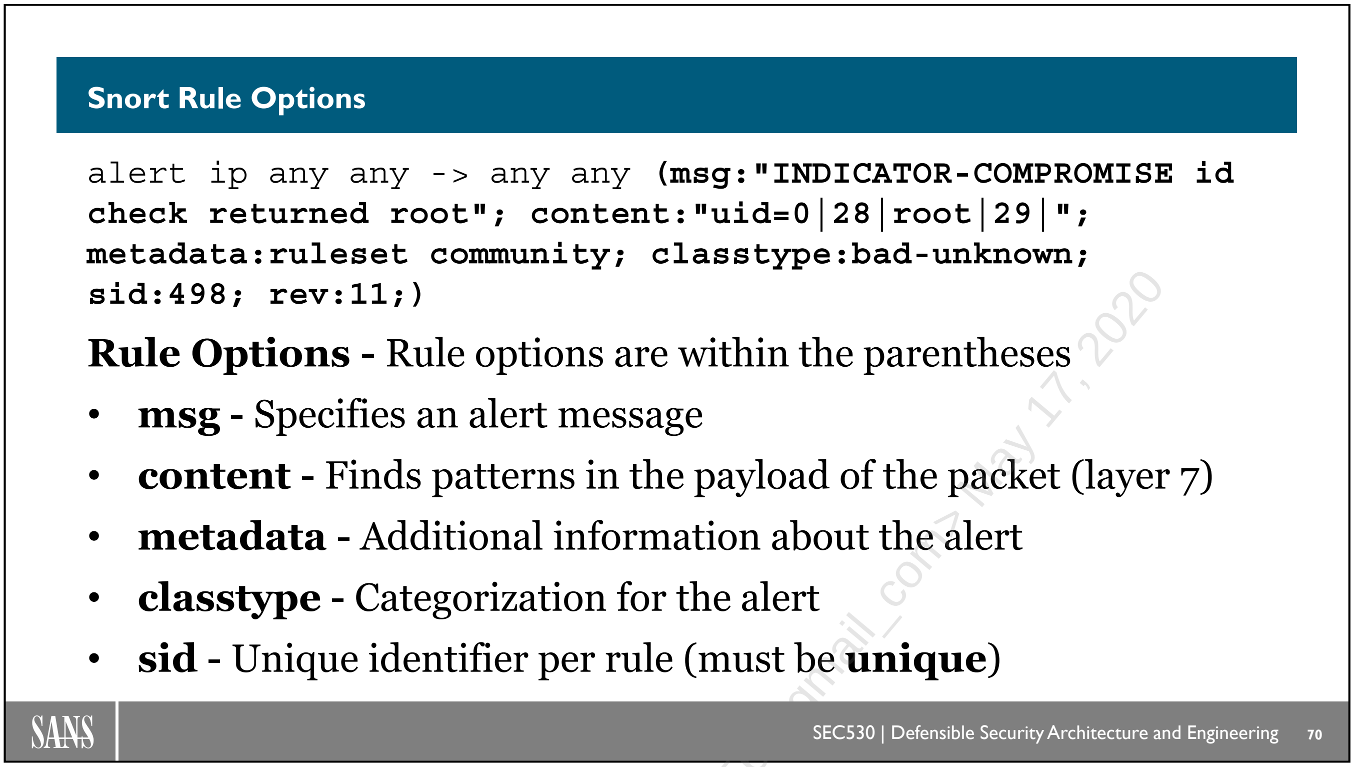
- EXTERNAL_NET is commonly set to non-RFC1918 addresses
- Doing so makes most rules miss internal -> internal traffic
- Alternative is to set EXTERNAL_NET to **any**

Variables

Inside Snort's configuration file variables can be declared. IP address variables are declared using ipvar, and port variables are declared with portvar. Using variables make it easier to write rules and maintain consistency. For example, if you had web servers on port 80 and 8080 but later added port 8081, you would have to update every rule to have the additional port. However, if a portvar was used you simply would update the variable, and all the rules will be updated upon a restart of Snort.

The default variables often need to be tuned. These include but are not limited to HOME_NET, EXTERNAL_NET, DNS_SERVERS, HTTP_SERVERS, and SQL_SERVERS. The default behavior for HOME_NET is to include all RFC1918 private addresses. EXTERNAL_NET defaults to !$HOME_NET, or non-RFC1918 private IP addresses. Since many of Snort rules use the variable $EXTERNAL_NET, it may be worth changing EXTERNAL_NET to any. By setting EXTERNAL_NET to any rules will monitor and alert on actions from internal devices to other internal devices. Using any requires more work maintaining rules but provides significantly more security. Other variables like HTTP_SERVERS should be set according to system assets. Variables like HTTP_SERVERS default to $HOME_NET. This leads to false positives. For example, a workstation could accidentally trigger a web server related rule even though it is not a web server.

[1] https://www.snort.org/faq/readme-variables



Snort Rule Options

```
alert ip any any -> any any (msg:"INDICATOR-COMPROMISE id
check returned root"; content:"uid=0|28|root|29|";
metadata:ruleset community; classtype:bad-unknown;
sid:498; rev:11;)
```

Rule Options - Rule options are within the parentheses

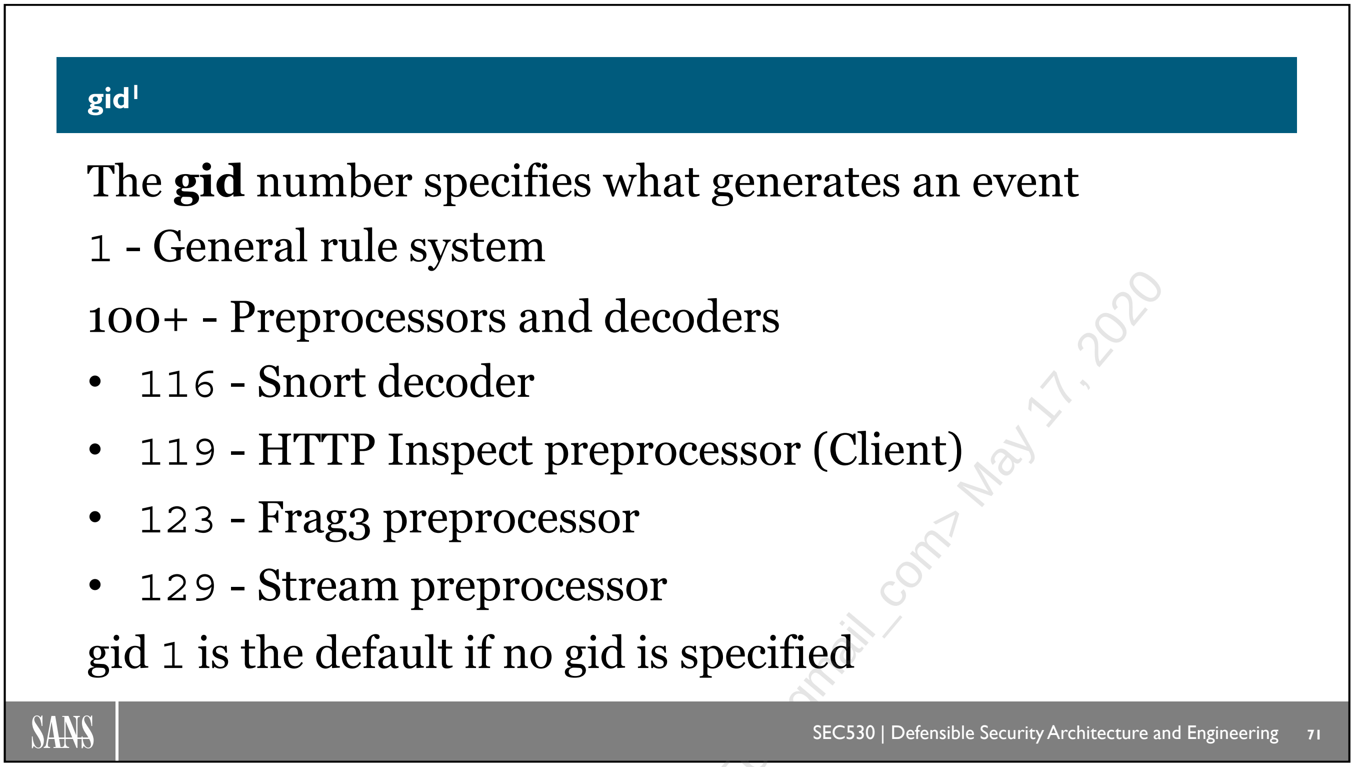
- **msg** - Specifies an alert message
- **content** - Finds patterns in the payload of the packet (layer 7)
- **metadata** - Additional information about the alert
- **classtype** - Categorization for the alert
- **sid** - Unique identifier per rule (must be **unique**)

Snort Rule Options

The power in a Snort signature is in the rule options. Rule options specify information about the rule and more importantly what layer-7 information to look for. The rule options portion of this slide captures some of the more common options.

The sample rule in the slide is simply looking for "uid=0(root)". The () are specified in hexadecimal characters since they are special characters. The content section simply requires surrounding hex matches with the pipe character. The content match behavior can be changed by specifying a content modifier such as nocase or http_header. In the example of nocase, payload matches would not be case sensitive and match whether something is uppercase or lowercase. In the example of http_header, the content match would only look in an HTTP header. There are many content modifiers available.

A well-written Snort rule will include a classtype, a well-defined msg field, and ideally metadata with additional information about the rule. All rules require a sid.



gid[1]

The **gid** number specifies what generates an event

`1` - General rule system

100+ - Preprocessors and decoders

- `116` - Snort decoder
- `119` - HTTP Inspect preprocessor (Client)
- `123` - Frag3 preprocessor
- `129` - Stream preprocessor

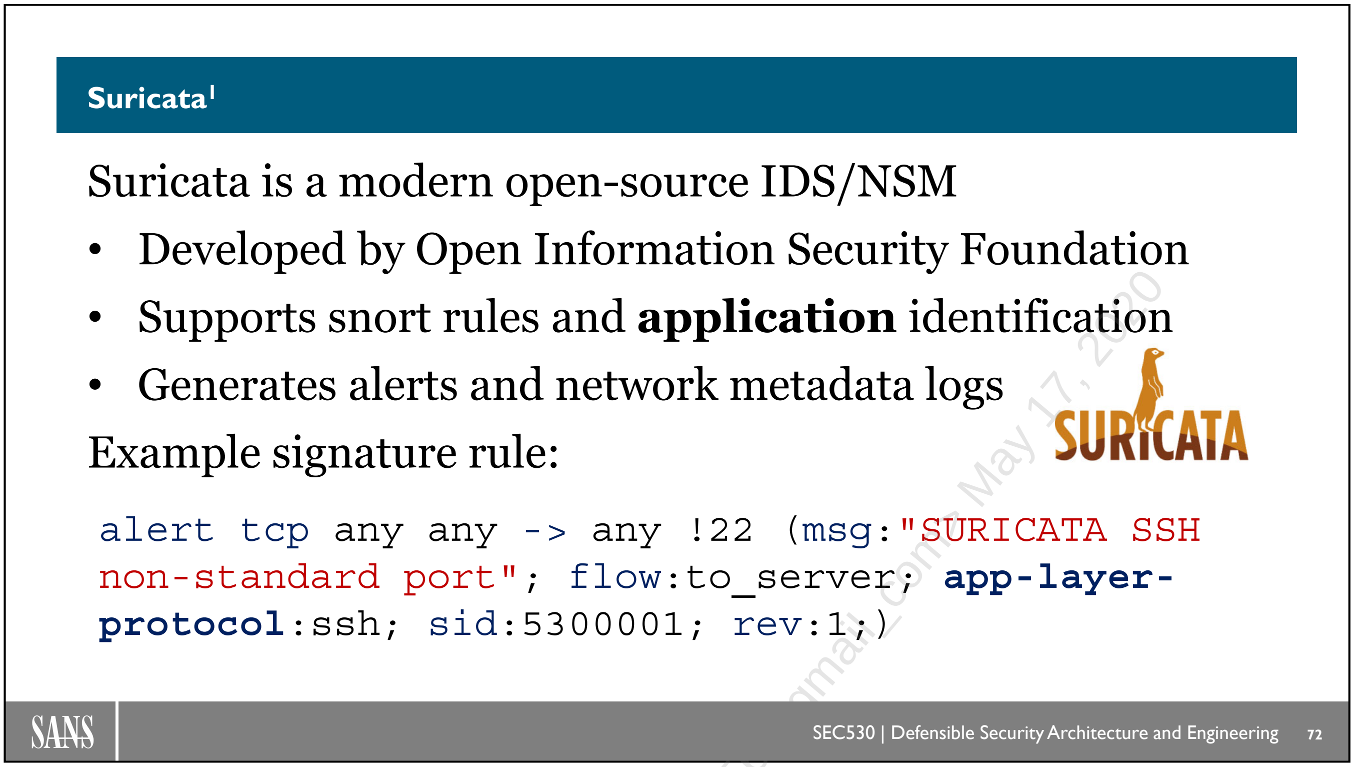
gid `1` is the default if no gid is specified

gid[1]

Snort comes with built-in capabilities to decode and process payload information. These come in the form of a decoder or preprocessor. Because alerts can trigger from a standard Snort rule, decoded data, or protocol preprocessing a gid is necessary to identify what generated an event. The most common gid number is one. A gid of one is the default when a gid is not specified and stands for a general Snort rule.

A gid at or above 100 involves decoders and preprocessors such as those found in this slide. These are special capabilities built-in to Snort that are rarely modified by an organization.

[1] http://manual-snort-org.s3-website-us-east-1.amazonaws.com/node31.html



Suricata[1]

Suricata is a modern open-source IDS/NSM

- Developed by Open Information Security Foundation
- Supports snort rules and **application** identification
- Generates alerts and network metadata logs

Example signature rule:

```
alert tcp any any -> any !22 (msg:"SURICATA SSH
non-standard port"; flow:to_server; app-layer-
protocol:ssh; sid:5300001; rev:1;)
```

Suricata[1]

Suricata is the new NIDS on the block. That being said, it has been in existence since 2010. Adoption of Suricata has been slow primarily because Snort is amazing, and folks are comfortable with it. However, Suricata natively works with Snort rules and also supports major enhancements such as better processor support, application identification in rules, and network metadata extraction. Do not underestimate these capabilities.

Application identification alone is worth the jump from Snort. This slide shows a simple Snort rule. Except, this rule would not work in Snort because snort does not support application layer identification. This simple rule looks for the SSH application being used on any non-standard port. Suricata does this with ease by using the **app-layer-protocol** rule option. On top of that, network metadata logging is a huge boon to security.

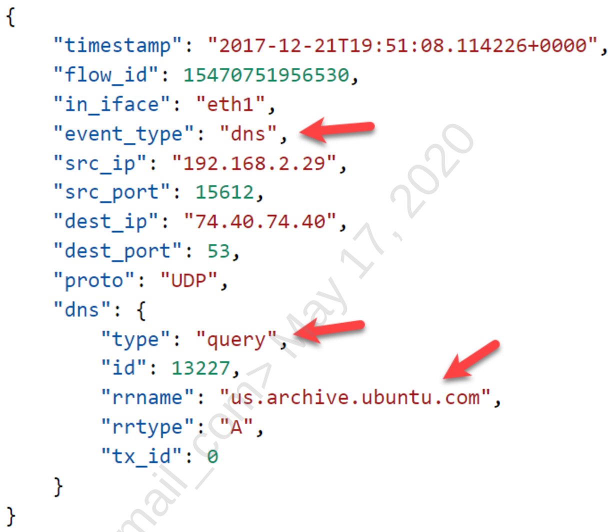
[1] https://suricata-ids.org

Metadata Logging

Suricata creates network logs

- DNS
- Bidirectional flow data
- HTTP
- SMTP
- TLS and more

Means you have IDS alerts plus **context**



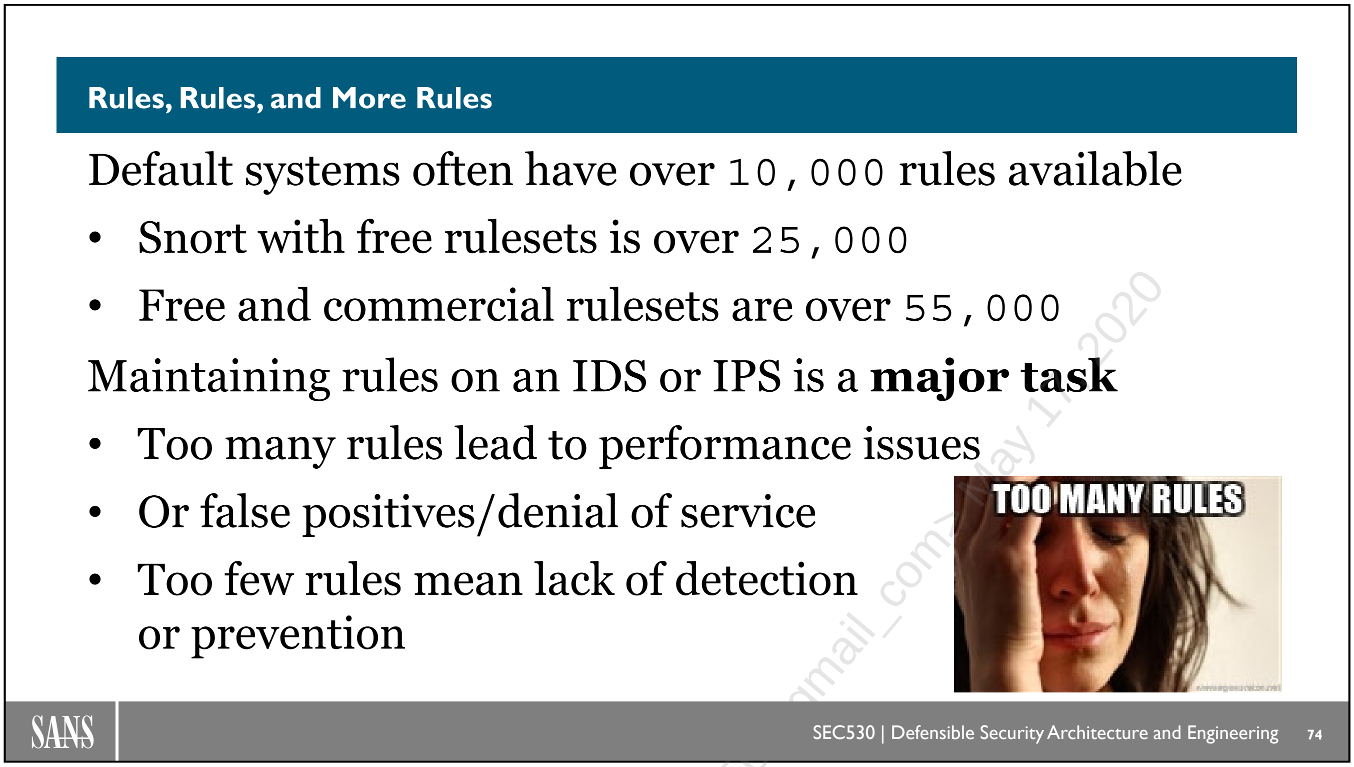
```
{
    "timestamp": "2017-12-21T19:51:08.114226+0000",
    "flow_id": 15470751956530,
    "in_iface": "eth1",
    "event_type": "dns",
    "src_ip": "192.168.2.29",
    "src_port": 15612,
    "dest_ip": "74.40.74.40",
    "dest_port": 53,
    "proto": "UDP",
    "dns": {
        "type": "query",
        "id": 13227,
        "rrname": "us.archive.ubuntu.com",
        "rrtype": "A",
        "tx_id": 0
    }
}
```

Metadata Logging

Having alerts plus context is a core NSM concept. Think about this for a second, you have an IDS alert, and now you need to investigate it. What DNS names were involved? Were any files downloaded? Was encryption used? How many bytes were transferred between the victim and attacker and how long was the connection? An alert will not answer all of these questions. However, if you have network metadata logs, you can answer all of these questions.

The definition of metadata is simply data about data. In regard to Suricata, the metadata is detailed information about a network connection. For example, the Suricata log in this slide contains network metadata extracted from DNS traffic observed over the network. In effect, Suricata has generated a DNS log by observing traffic. This ability to extract metadata from the network is extremely powerful as it allows defenders to collect and analyze massive amounts of data without having to reconfigure production systems.

The image in this slide shows a Suricata DNS query log. It shows the client of 192.168.2.29 asked the DNS server at 74.40.74.40 for the IP address associated with the A record of us.archive.ubuntu.com. Suricata also would log the response from the DNS server. This level of context is a huge differentiator between an IDS and an NSM sensor. An IDS generates alerts. An NSM sensor generates alerts, network logs, and more.



Rules, Rules, and More Rules

Default systems often have over 10,000 rules available

- Snort with free rulesets is over 25,000
- Free and commercial rulesets are over 55,000

Maintaining rules on an IDS or IPS is a **major task**

- Too many rules lead to performance issues
- Or false positives/denial of service
- Too few rules mean lack of detection or prevention

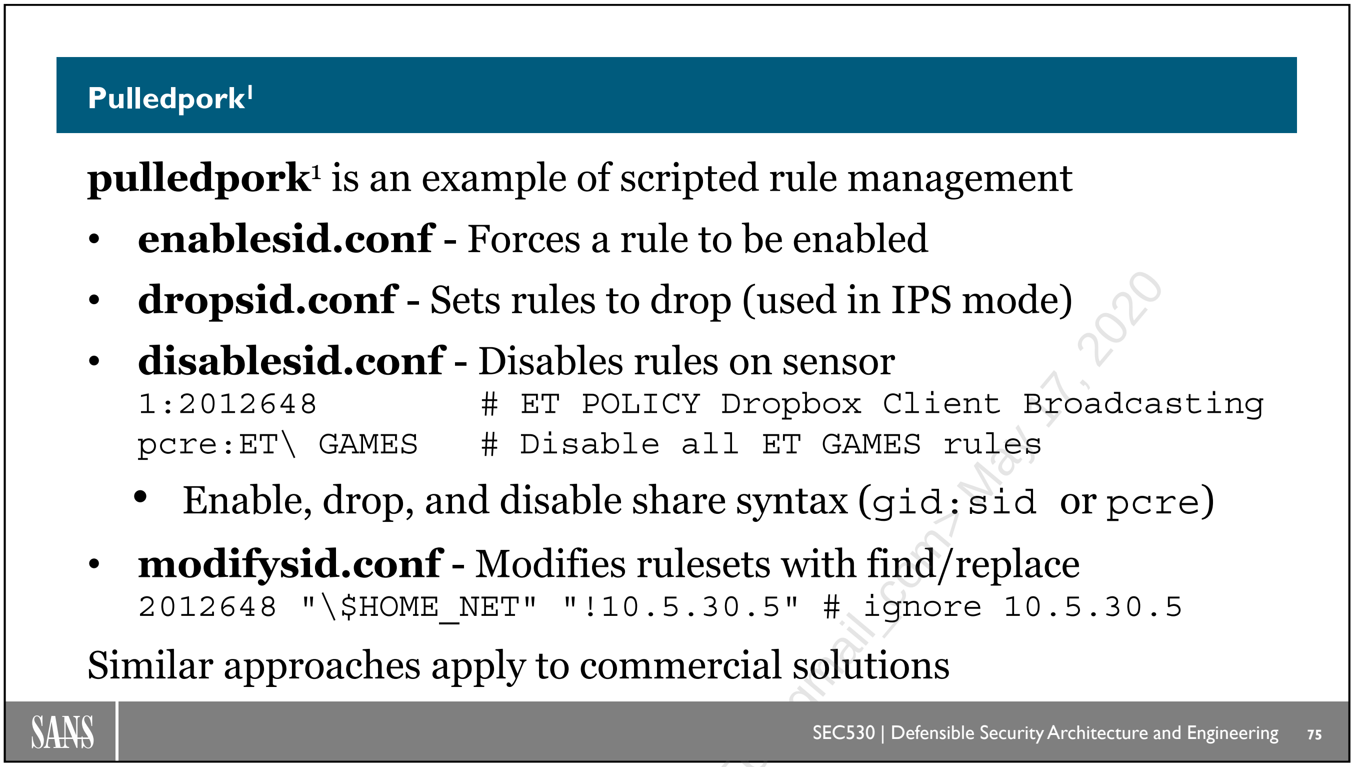
TOO MANY RULES

SANS

SEC530 | Defensible Security Architecture and Engineering 74

Rules, Rules, and More Rules

A NIDS requires lots of labor to tune rules properly, and a successful NIDS deployment requires rule tuning. Leaving all the default rule sets enabled generates a ton of false positives and can also cause performance bottlenecks that crash the system. Eliminating rules is a balancing act. Too many rules and false positives increase and performance tanks. Too few rules and alerts will not generate. While it is hard to give a definitive answer one rule of thumb is to disable enough rules so that performance is acceptable and your analysts that can be reviewing alerts can keep up with all alerts that generate within a 24-hour time period. If more alerts are generated that can be investigated in a 24-hour period; then the rules likely are not tuned.



Pulledpork[1]

pulledpork[1] is an example of scripted rule management

- **enablesid.conf** - Forces a rule to be enabled
- **dropsid.conf** - Sets rules to drop (used in IPS mode)
- **disablesid.conf** - Disables rules on sensor

```
1:2012648        # ET POLICY Dropbox Client Broadcasting
pcre:ET\ GAMES   # Disable all ET GAMES rules
```

  - Enable, drop, and disable share syntax (`gid:sid` or `pcre`)

- **modifysid.conf** - Modifies rulesets with find/replace

```
2012648 "\$HOME_NET" "!10.5.30.5" # ignore 10.5.30.5
```

Similar approaches apply to commercial solutions

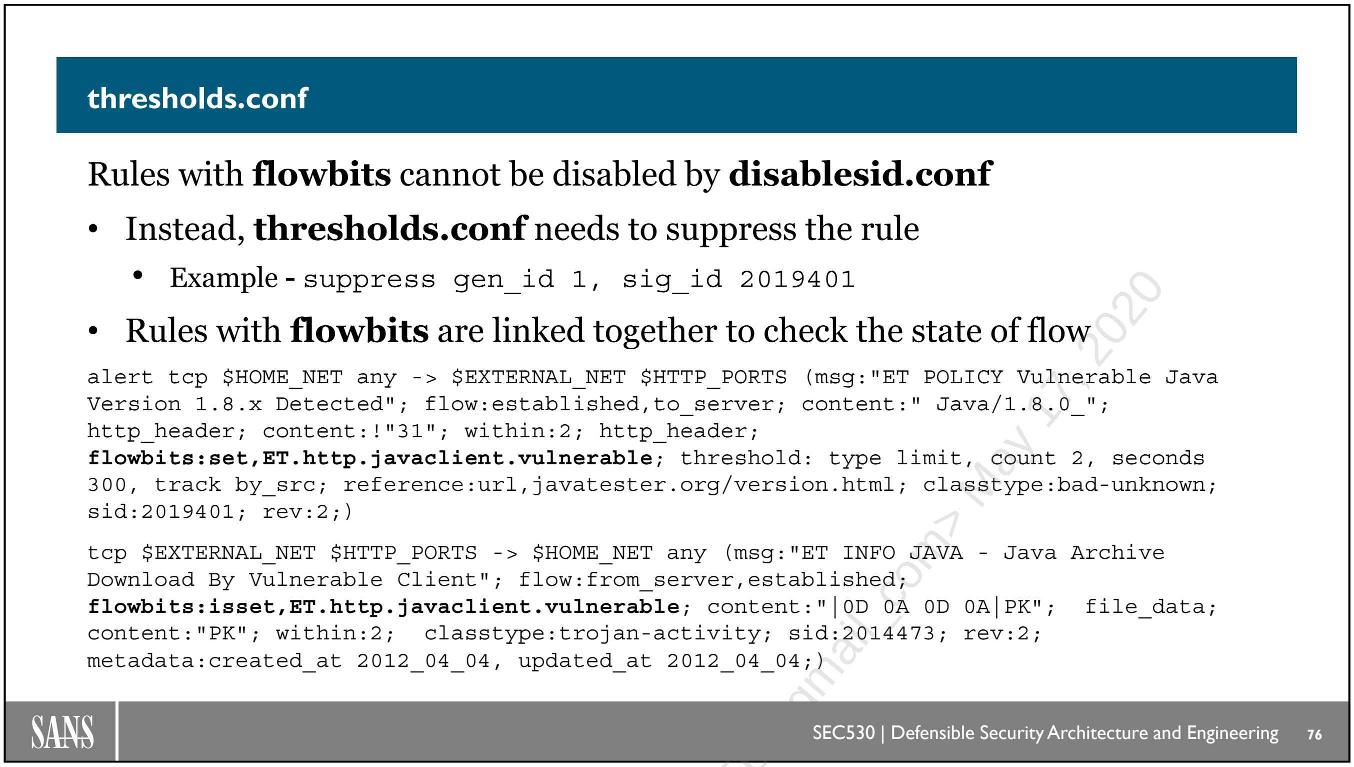
Pulledpork[1]

Regardless of if you are using Snort, Suricata, or a commercial NIDS, rule management is a must. To accomplish this in a more automated fashion on Snort or Suricata use pulledpork[1]. Pulledpork is a script that automates the download of new signatures as well as what to do with them as well as existing signatures. It uses a combination of unique signature IDs or sids, and regex to disable, change, or enable rules.

Examples on this slide are based on Security Onion's implementation of pulledpork[1]. Even if you are not using pulledpork, it is important to understand what it is doing under the hood as commercial rule management systems operate similarly. In effect, pulledpork is downloading rules on a reoccurring schedule. It then uses configuration settings found in the configuration files mentioned in the slide to automatically modify or disable rules. Snort or Suricata then load the downloaded and modified rulesets.

This slide describes the four main configuration files for modifying the status of rules. The enablesid.conf, dropsid.conf, and disablesid.conf all share the same syntax. Either a rule gid and sid are specified in gid:sid format or pcre is used to specify a regex to match rules based on the rule msg. The modifysid.conf format is different. It only works for gid one rules, and then it takes a string to find and then the string to replace it with.

[1] https://github.com/shirkdog/pulledpork



thresholds.conf

Rules with **flowbits** cannot be disabled by **disablesid.conf**

- Instead, **thresholds.conf** needs to suppress the rule
  - Example - `suppress gen_id 1, sig_id 2019401`
- Rules with **flowbits** are linked together to check the state of flow

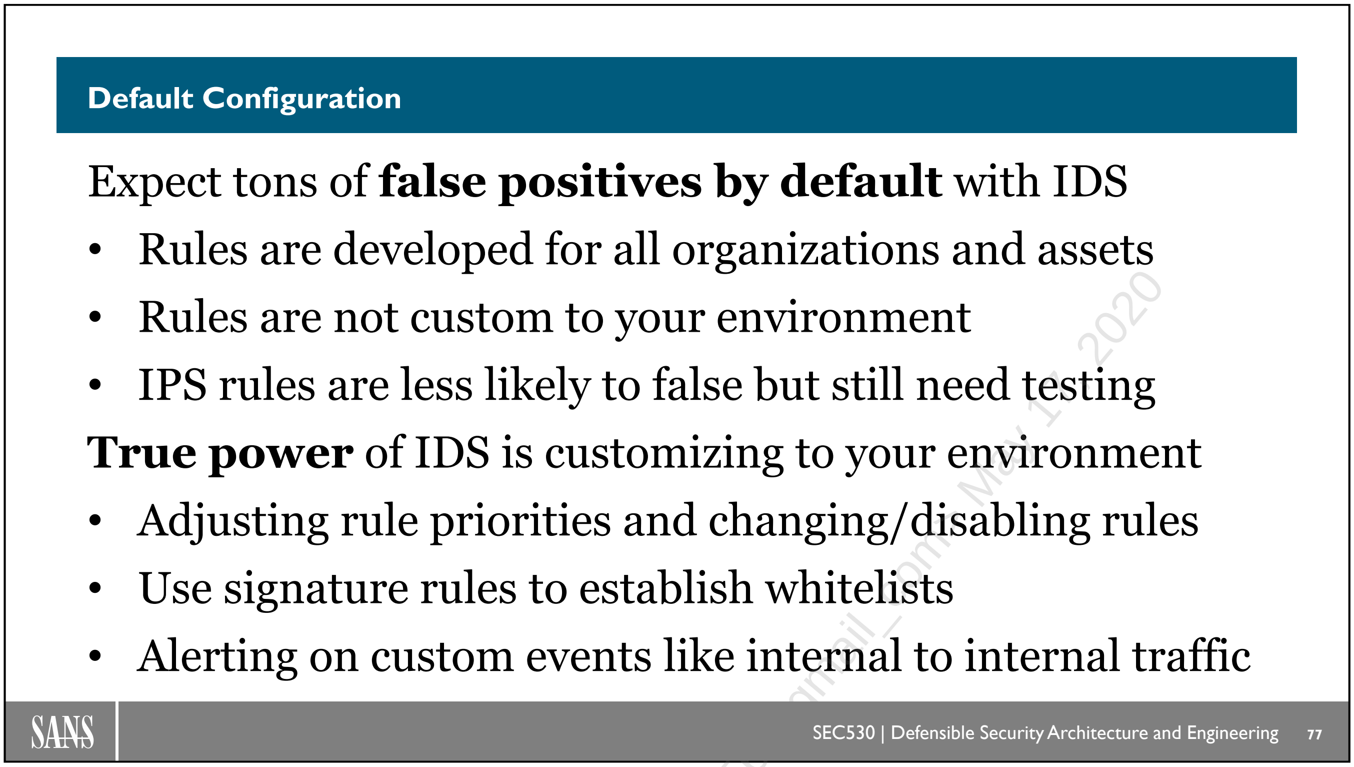
```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS (msg:"ET POLICY Vulnerable Java
Version 1.8.x Detected"; flow:established,to_server; content:" Java/1.8.0_";
http_header; content:!"31"; within:2; http_header;
flowbits:set,ET.http.javaclient.vulnerable; threshold: type limit, count 2, seconds
300, track by_src; reference:url,javatester.org/version.html; classtype:bad-unknown;
sid:2019401; rev:2;)

tcp $EXTERNAL_NET $HTTP_PORTS -> $HOME_NET any (msg:"ET INFO JAVA - Java Archive
Download By Vulnerable Client"; flow:from_server,established;
flowbits:isset,ET.http.javaclient.vulnerable; content:"|0D 0A 0D 0A|PK";  file_data;
content:"PK"; within:2;  classtype:trojan-activity; sid:2014473; rev:2;
metadata:created_at 2012_04_04, updated_at 2012_04_04;)
```

thresholds.conf

Flowbits are used by an IDS to share the state of a connection flow with another rule. Basically, it allows linking multiple rules together. For this to work a rule must first set a flowbit. In this slide, the first rule sets a flowbit called ET.http.javaclient.vulnerable if the rule matches network traffic. The second rule on the slide is triggered but only if ET.http.javaclient.vulnerable flowbit is set from the first rule. This is what the flowbits:isset setting is for. Flowbits allow chaining rules together. However, because rules are chained together rules that utilize flowbits must be handled with thresholds.conf instead of modifysid.conf or disablesid.conf.

The thresholds.conf file is needed to disable alerting of rules using flowbits. The examples in this slide show a java rule with a sid of 2019401 that sets the flowbit called ET.http.javaclient.vulnerable. The second rule with a sid of 2014473 is one of the multiple java rules that checks to see if the ET.http.javaclient.vulnerable flowbit is set.

Normally, a rule would be disabled in disabledsid.conf, but because of flowbits a rule cannot be disabled in disablesid.conf unless all other associated rules with the same flowbit are disabled. This is likely not desired. Instead, a suppression rule can be added to thresholds.conf. This has the desired effect of disabling alerts from a specific rule without breaking other rules.



Default Configuration

Expect tons of **false positives by default** with IDS

- Rules are developed for all organizations and assets
- Rules are not custom to your environment
- IPS rules are less likely to false but still need testing

True power of IDS is customizing to your environment

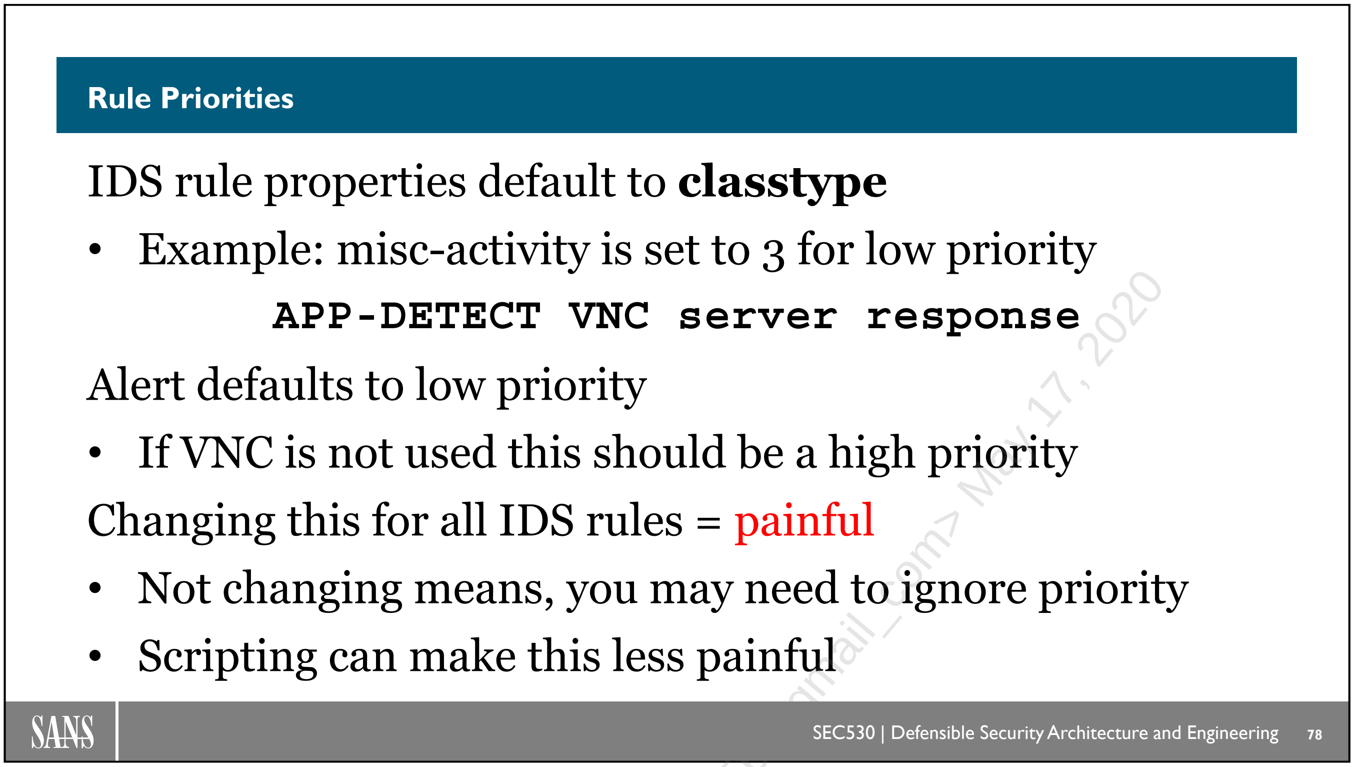
- Adjusting rule priorities and changing/disabling rules
- Use signature rules to establish whitelists
- Alerting on custom events like internal to internal traffic

SANS

Default Configuration

The rules that come with an IDS are tailored to work in every organization. This means that rules are not tailored to your organization and thus every NIDS is misconfigured by default. Default rules cause a lot of false positives and lack the specificity necessary to catch adversaries and unauthorized activity. If you are thinking about purchasing a NIDS and getting immense value out of a default configuration, then NIDS is probably not the right purchase. NIPS systems are less likely to have false positives as vendors cannot afford for rules to block activity accidentally. This means that NIPS products tend to have much fewer rules enabled and even then, those rules tend to be more specific. In turn, a NIPS is much more likely to let evil through with a default configuration.

The power of a NIDS/NIPS comes with tuning it. Some minor adjustments or rule creation can add some powerful detection and prevention capabilities. Some of the changes to consider are adjusting rule priorities, using signature rules in a whitelisting fashion, and creating custom alerts based on internal flows.



Rule Priorities

IDS rule properties default to **classtype**

- Example: misc-activity is set to 3 for low priority

```
APP-DETECT VNC server response
```

Alert defaults to low priority

- If VNC is not used this should be a high priority

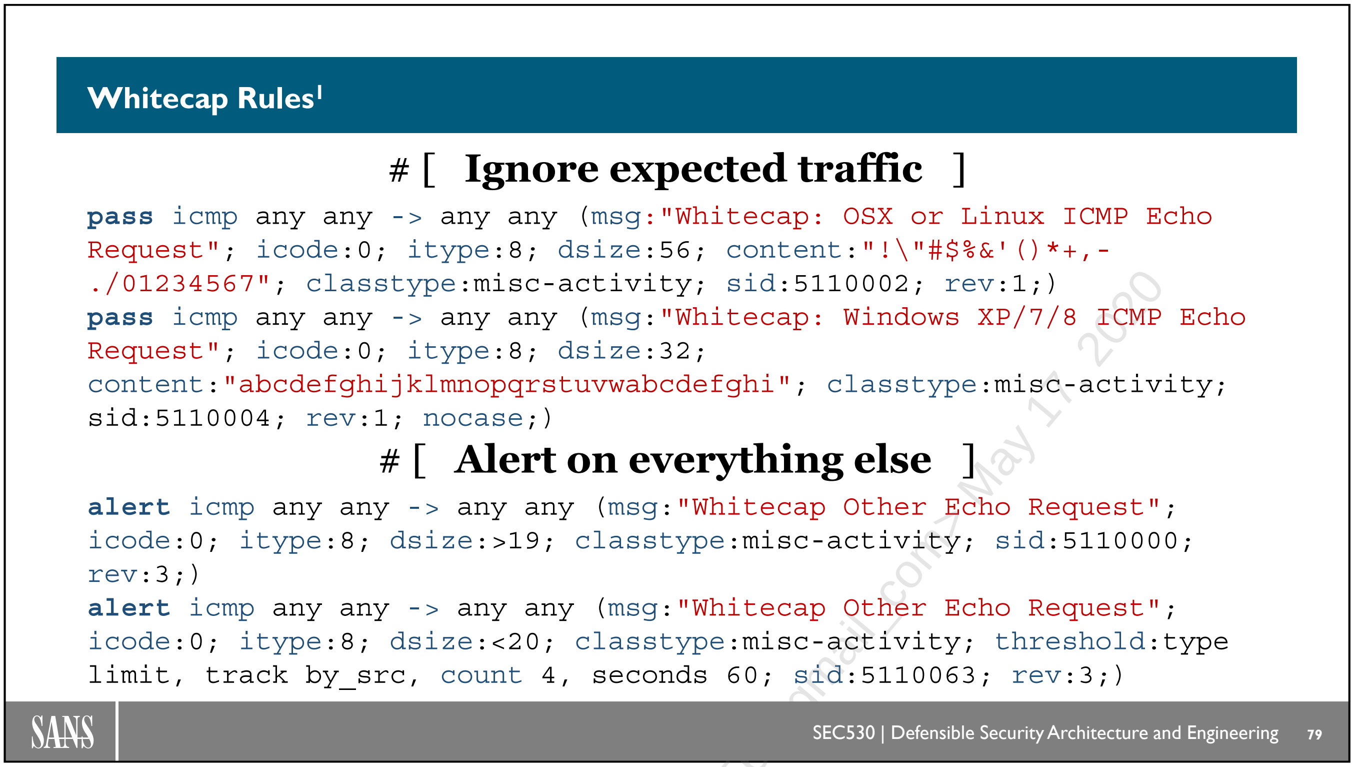
Changing this for all IDS rules = painful

- Not changing means, you may need to ignore priority
- Scripting can make this less painful

Rule Priorities

One thing new analysts have trouble with is that they want to prioritize alerts by the priority specified. For example, APP-DETECT VNC server response has a priority of low, so a new analyst may skip it, and instead, focus on something that has a priority of high. Yet, the alert that has a priority of high may actually be insignificant. This happens because IDS systems use a default classification scheme if an alert does not specify a priority, which most do not. Realistically, either the alert priority field should be ignored, or it should be regularly updated.

Scripting or using tools like pulledpork can greatly aid in reprioritizing rules. For example, modifysid.conf can be used to dynamically change the priority on rules based on regex match or sid numbers. The problem is more so of going through all the rules. A default system may have upwards of sixty thousand rules. Rule priority updates may be more feasible by updating alerts that are received that have an incorrect priority.



Whitecap Rules[1]

# [Ignore expected traffic]

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

# [Alert on everything else]

```
alert icmp any any -> any any (msg:"Whitecap Other Echo Request";
icode:0; itype:8; dsize:>19; classtype:misc-activity; sid:5110000;
rev:3;)
alert icmp any any -> any any (msg:"Whitecap Other Echo Request";
icode:0; itype:8; dsize:<20; classtype:misc-activity; threshold:type
limit, track by_src, count 4, seconds 60; sid:5110063; rev:3;)
```

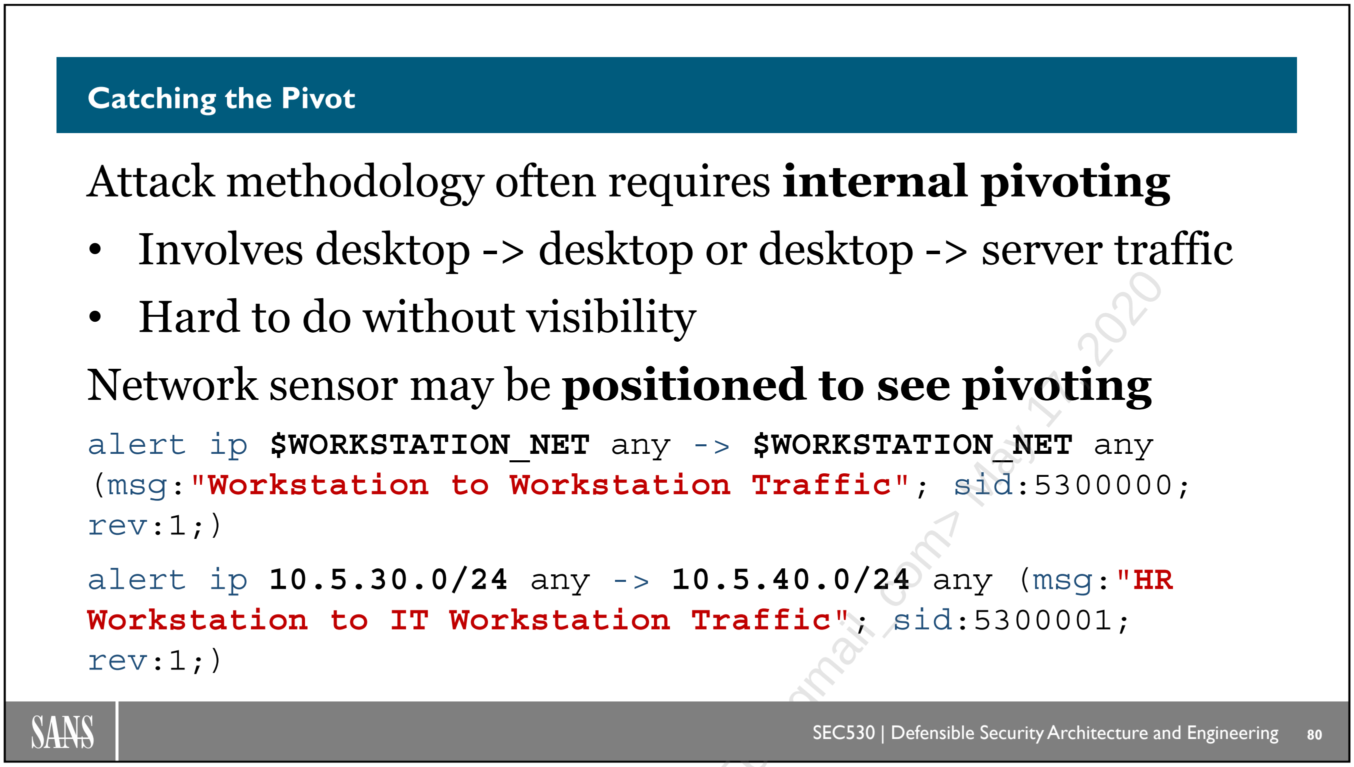
Whitecap Rules[1]

A NIDS signature is primarily a blacklisting mechanism. However, because actions like pass exist, rules are able to be written in a whitelist fashion. For example, Eric Conrad came up with the concept of using Snort rules to whitelist ICMP traffic and then Justin Henderson worked with him to apply the technique in production environments. These rules are called Whitecap rules.

To whitelist authorized ICMP activity rules similar to the top two rules in this slide are applied. These pass rules will ignore valid ICMP such as a ping from a Linux or Windows box. Then all other ICMP activity not matching a whitelist rule will cause an alert. To reduce false positives the last rule on the slide will allow up to four ping packets that have a data size less than 20 bytes within a minute before alerting. This would allow unknown ping payloads without generating massive amounts of false positives.

For a complete listing of whitelist rules visit the NSM GitHub page[1].

[1] https://github.com/sans-blue-team/NSM



Catching the Pivot

Attack methodology often requires **internal pivoting**

- Involves desktop -> desktop or desktop -> server traffic
- Hard to do without visibility

Network sensor may be **positioned to see pivoting**

```
alert ip $WORKSTATION_NET any -> $WORKSTATION_NET any
(msg:"Workstation to Workstation Traffic"; sid:5300000;
rev:1;)

alert ip 10.5.30.0/24 any -> 10.5.40.0/24 any (msg:"HR
Workstation to IT Workstation Traffic"; sid:5300001;
rev:1;)
```

Catching the Pivot

Mirroring or tapping network traffic to a sensor is an easy process. Because the process is simple to perform, a single IDS sensor is capable of having visibility into multiple segments or layers of an organization. By having this much network visibility, a NIDS may be the best place to look for unauthorized pivoting activity. Thus, even if a NIDS is deployed out-of-band without prevention capabilities, the tactical advantage is still a huge win.

The sample rules in the slide show how simple it would be to find workstations talking to other workstations or specific subnets reaching out to subnets they should not be communicating with. Keep in mind, the rules above are likely only going to see traffic crossing a router. However, it is possible to see layer 2 connections if a switch is capable of port mirroring all desktop ports to an IDS. Some organizations use 10 Gb fiber links from distributed closet switches to port mirror the entire 1 Gb switch across a 10 Gb port to a NIDS. Mirroring traffic like this works because even though a 1 Gb switch is capable of doing 1 Gb full duplex across 48 ports the aggregate traffic usually will not exceed 10 Gb half duplex across the fiber port. Mirroring an entire switch across a large uplink only works if the switch you have is not limited to mirroring a small number of ports.

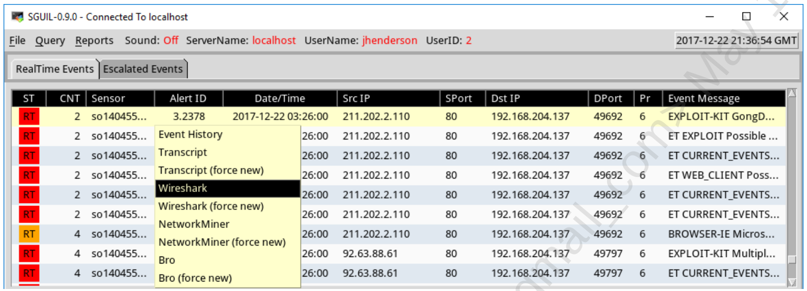
Alert Investigation

IDS/IPS solutions have investigation workflows

- Alerts should be analyzed
- Results and findings should be recorded

Security Onion makes use of Sguil[1], Squert[2], and Kibana[3]

- All provide support for pivoting from alert to pcap



ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	2	so140455...	3.2378	2017-12-22 03:26:00	211.202.2.110	80	192.168.204.137	49692	6	EXPLOIT-KIT GongD...
RT	2	so140455...		26:00	211.202.2.110	80	192.168.204.137	49692	6	ET EXPLOIT Possible ...
RT	2	so140455...		26:00	211.202.2.110	80	192.168.204.137	49692	6	ET CURRENT_EVENTS...
RT	2	so140455...		26:00	211.202.2.110	80	192.168.204.137	49692	6	ET WEB_CLIENT Poss...
RT	2	so140455...		26:00	211.202.2.110	80	192.168.204.137	49692	6	ET CURRENT_EVENTS...
RT	2	so140455...		26:00	211.202.2.110	80	192.168.204.137	49692	6	ET CURRENT_EVENTS...
RT	4	so140455...		26:00	211.202.2.110	80	192.168.204.137	49692	6	BROWSER-IE Micros...
RT	4	so140455...		26:00	92.63.88.61	80	192.168.204.137	49797	6	EXPLOIT-KIT Multipl...
RT	4	so140455...		26:00	92.63.88.61	80	192.168.204.137	49797	6	ET CURRENT_EVENTS...

Alert Investigation

A NIDS system is based on generating alerts. Therefore, an alert investigation interface is required and needs to be routinely utilized. All NIDS have supported alert investigations. Commercial NIDS have built-in workflows, and Snort has multiple open source solutions that integrate with it. For example, Security Onion has three built-in interfaces that can be used. Sguil[1] is the most commonly used with Security Onion, Squert[2] provides an easy to use the web interface, and Kibana[3] is part of the new Elastic Stack open source SIEM integration and allows pivoting between alerts and other log data.

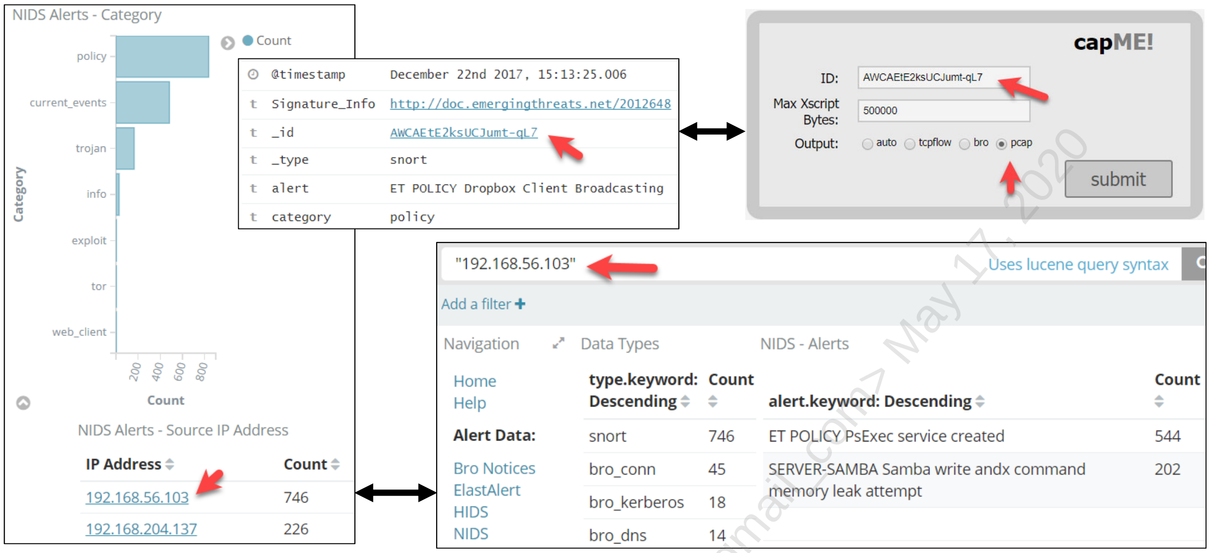
These interfaces are important to help analysts investigate alerts. A key component of that is the ability to pivot to other data. For example, Security Onion supports pivoting from alerts to packet capture data. Commercial interfaces support similar capabilities. Another component that is equally important is the ability to classify alerts after an investigation. Alert classification helps identify how many attacks are occurring and what they are. Classification also helps identify bad rules that need to be cleaned up.

[1] http://bammv.github.io/sguil/index.html

[2] http://www.squertproject.org/

[3] https://www.elastic.co/products/kibana

SIEM Integration - Kibana



@timestamp	December 22nd 2017, 15:13:25.006
t Signature_Info	http://doc.emergingthreats.net/2012648
t _id	AWCAEtE2ksUCJumt-qL7
t _type	snort
t alert	ET POLICY Dropbox Client Broadcasting
t category	policy

capME!

ID: AWCAEtE2ksUCJumt-qL7

Max Xscript Bytes: 500000

Output: ○ auto ○ tcpflow ○ bro ● pcap

submit

NIDS Alerts - Source IP Address

IP Address	Count
192.168.56.103	746
192.168.204.137	226

"192.168.56.103"

Uses lucene query syntax

Add a filter +

Navigation

Home
Help
Alert Data:
Bro Notices
ElastAlert
HIDS
NIDS

Data Types

type.keyword: Descending	Count
snort	746
bro_conn	45
bro_kerberos	18
bro_dns	14

NIDS - Alerts

alert.keyword: Descending	Count
ET POLICY PsExec service created	544
SERVER-SAMBA Samba write andx command memory leak attempt	202

SIEM Integration - Kibana

Integrating network logs with a SIEM allows for simpler workflow orchestration. At its simplest, a SIEM provides a central location for storing and searching log data. By having the data centrally collected and searchable, an analyst gains efficiency by being able to search across disparate data sources that are related.

A SIEM also provides the ability to pivot from key data to external systems. For example, the images in this slide show multiple fields linking off to third-party solutions or other dashboards with related information. The _id field when clicked pivots to a third-party solution called CapMe[1] which retrieves full packet capture data from the related network session. IP addresses in the left image link off to a dashboard showing log data related to the IP address that is clicked. In this case, it shows what Zeek logs are available for the given IP address as well as what alerts pertain to the IP.

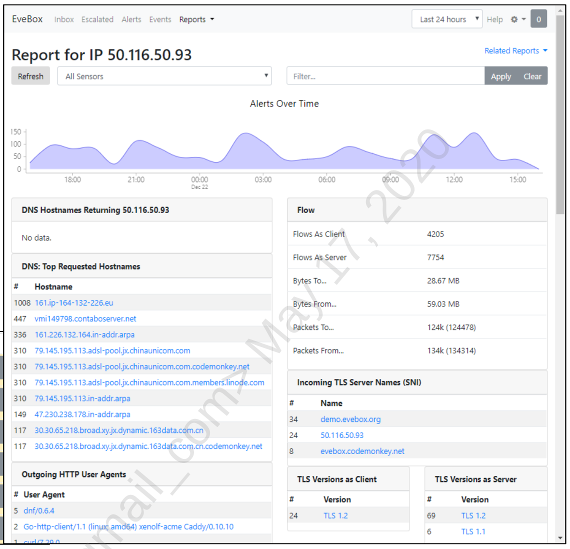
By having dashboards and third-party products integrated through a SIEM analysis becomes less frustrating and less time-consuming.

[1] https://github.com/Security-Onion-Solutions/securityonion-capme

Evebox[1]

Evebox is designed for Suricata

- Provides a web based alert and event management tool
- Uses alerts and metadata logs



Report for IP 50.116.50.93

DNS Hostnames Returning 50.116.50.93

No data.

DNS: Top Requested Hostnames

#	Hostname
1008	161.ip-164-132-226.eu
447	vmi149798.contaboserver.net
336	161.226.132.164.in-addr.arpa
310	79.145.195.113.adsl-pool.jx.chinaunicom.com
310	79.145.195.113.adsl-pool.jx.chinaunicom.com.codemonkey.net
310	79.145.195.113.adsl-pool.jx.chinaunicom.com.members.linode.com
310	79.145.195.113.in-addr.arpa
149	47.230.238.178.in-addr.arpa
117	30.30.65.218.broad.xy.jx.dynamic.163data.com.cn
117	30.30.65.218.broad.xy.jx.dynamic.163data.com.cn.codemonkey.net

Outgoing HTTP User Agents

#	User Agent
5	dnf/0.6.4
2	Go-http-client/1.1 (linux; amd64) xenolf-acme Caddy/0.10.10
1	curl/7.29.0

Flow

Flows As Client	4205
Flows As Server	7754
Bytes To...	28.67 MB
Bytes From...	59.03 MB
Packets To...	124k (124478)
Packets From...	134k (134314)

Incoming TLS Server Names (SNI)

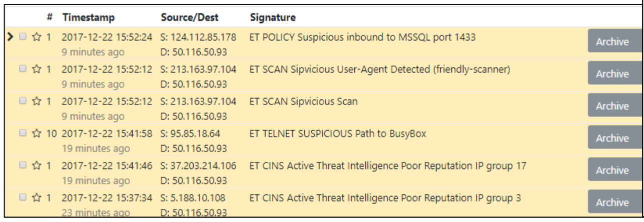
#	Name
34	demo.evebox.org
24	50.116.50.93
8	evebox.codemonkey.net

TLS Versions as Client

#	Version
24	TLS 1.2

TLS Versions as Server

#	Version
69	TLS 1.2
6	TLS 1.1

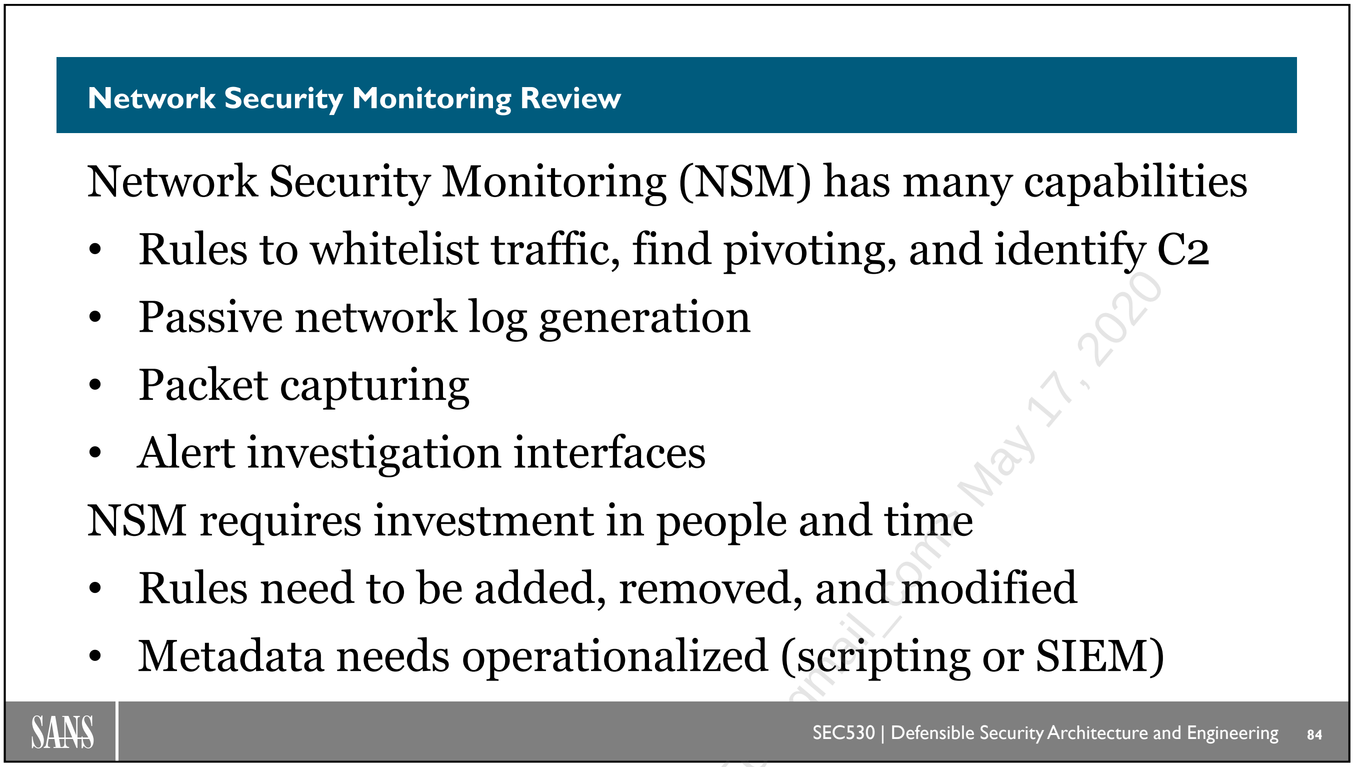


#	Timestamp	Source/Dest	Signature
1	2017-12-22 15:52:24 9 minutes ago	S: 124.112.85.178 D: 50.116.50.93	ET POLICY Suspicious inbound to MSSQL port 1433
1	2017-12-22 15:52:12 9 minutes ago	S: 213.163.97.104 D: 50.116.50.93	ET SCAN Sipvicious User-Agent Detected (friendly-scanner)
1	2017-12-22 15:52:12 9 minutes ago	S: 213.163.97.104 D: 50.116.50.93	ET SCAN Sipvicious Scan
10	2017-12-22 15:41:58 19 minutes ago	S: 95.85.18.64 D: 50.116.50.93	ET TELNET SUSPICIOUS Path to BusyBox
1	2017-12-22 15:41:46 19 minutes ago	S: 37.203.214.106 D: 50.116.50.93	ET CINS Active Threat Intelligence Poor Reputation IP group 17
1	2017-12-22 15:37:34 23 minutes ago	S: 5.188.10.108 D: 50.116.50.93	ET CINS Active Threat Intelligence Poor Reputation IP group 3

Evebox[1]

One more alert investigation interface worth mentioning is Evebox[1]. Evebox is designed entirely for Suricata with quick and easy integration. Evebox can directly read the eve.json output Suricata uses, or it can hook into an Elasticsearch database. The beauty of Evebox is how simple it is to install and its interface. The interface simplifies analysis by allowing built-in correlation of Suricata alerts with Suricata metadata. For organizations that are not using Security Onion or Zeek Evebox provides a quick win.

[1] https://evebox.org/



Network Security Monitoring Review

Network Security Monitoring (NSM) has many capabilities

- Rules to whitelist traffic, find pivoting, and identify C2
- Passive network log generation
- Packet capturing
- Alert investigation interfaces

NSM requires investment in people and time

- Rules need to be added, removed, and modified
- Metadata needs operationalized (scripting or SIEM)

SANS

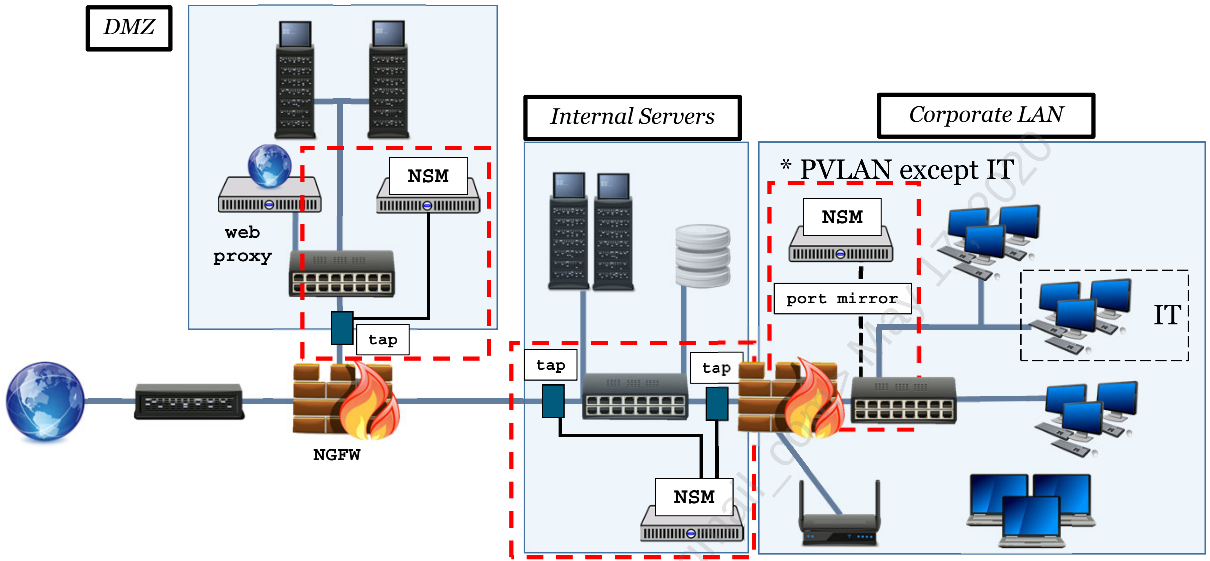
SEC530 | Defensible Security Architecture and Engineering 84

Network Security Monitoring Review

Gaining visibility into a network seems difficult. However, there are multiple commercial and open source solutions available to gain visibility into a network easily. Security Onion[1], in particular, is one of the most well put together open source enterprise platforms for installing, maintaining, and operationalizing network data. Regardless of the solution, network monitoring is a key component for any network defender.

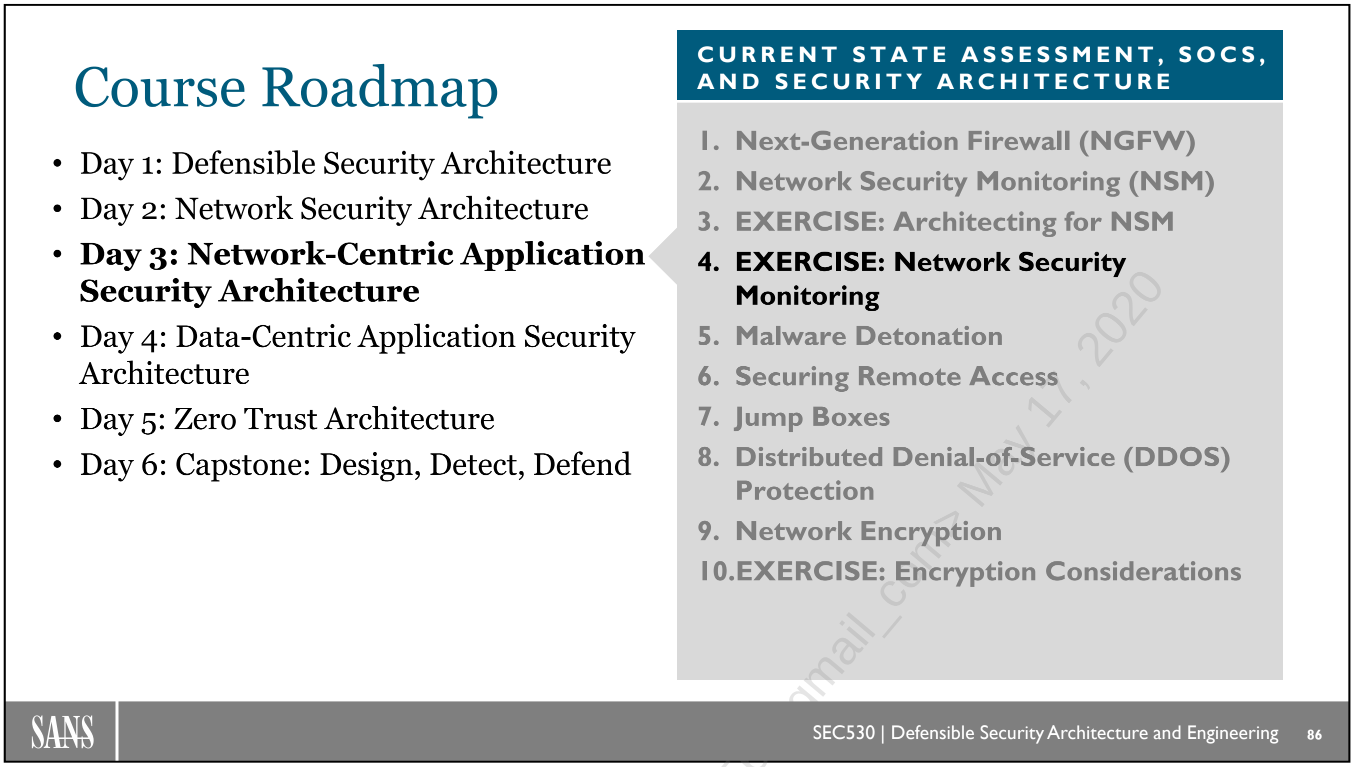
[1] https://www.sans.org/course/siem-with-tactical-analytics

Case Study: Tyrrell Corporation



Case Study: Tyrrell Corporation

This diagram shows the addition of multiple Network Security Monitor sensors to the Tyrrell Corporation architecture. In this example, a sensor is in the DMZ with a tap sending a copy of the traffic to it. Another sensor is within the internal server zone using network taps to receive traffic. Also, a sensor is in the corporate LAN zone with network traffic being received from a port mirror.



Course Roadmap

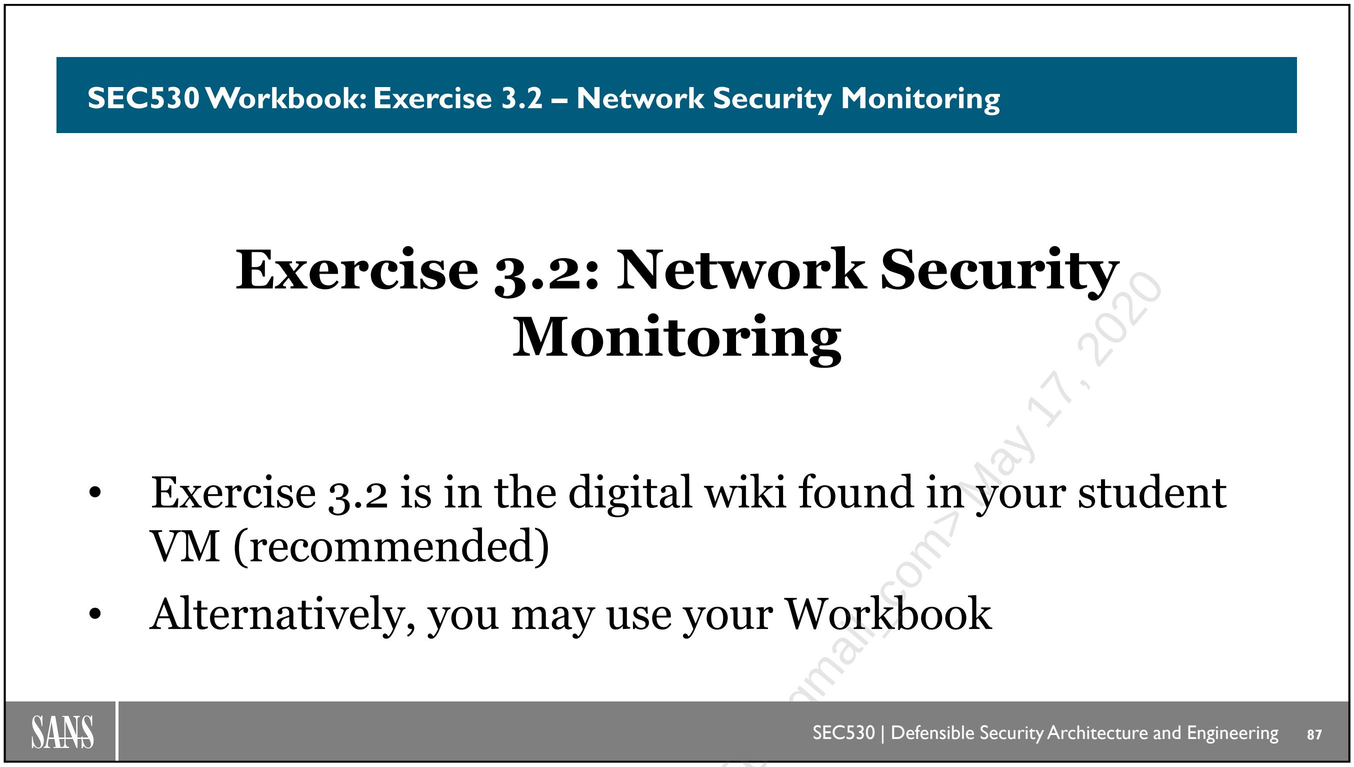
- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. **EXERCISE: Network Security Monitoring**
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

Course Roadmap

We will now perform a lab on looking holistically at Network Security Monitoring data.



SEC530 Workbook: Exercise 3.2 – Network Security Monitoring

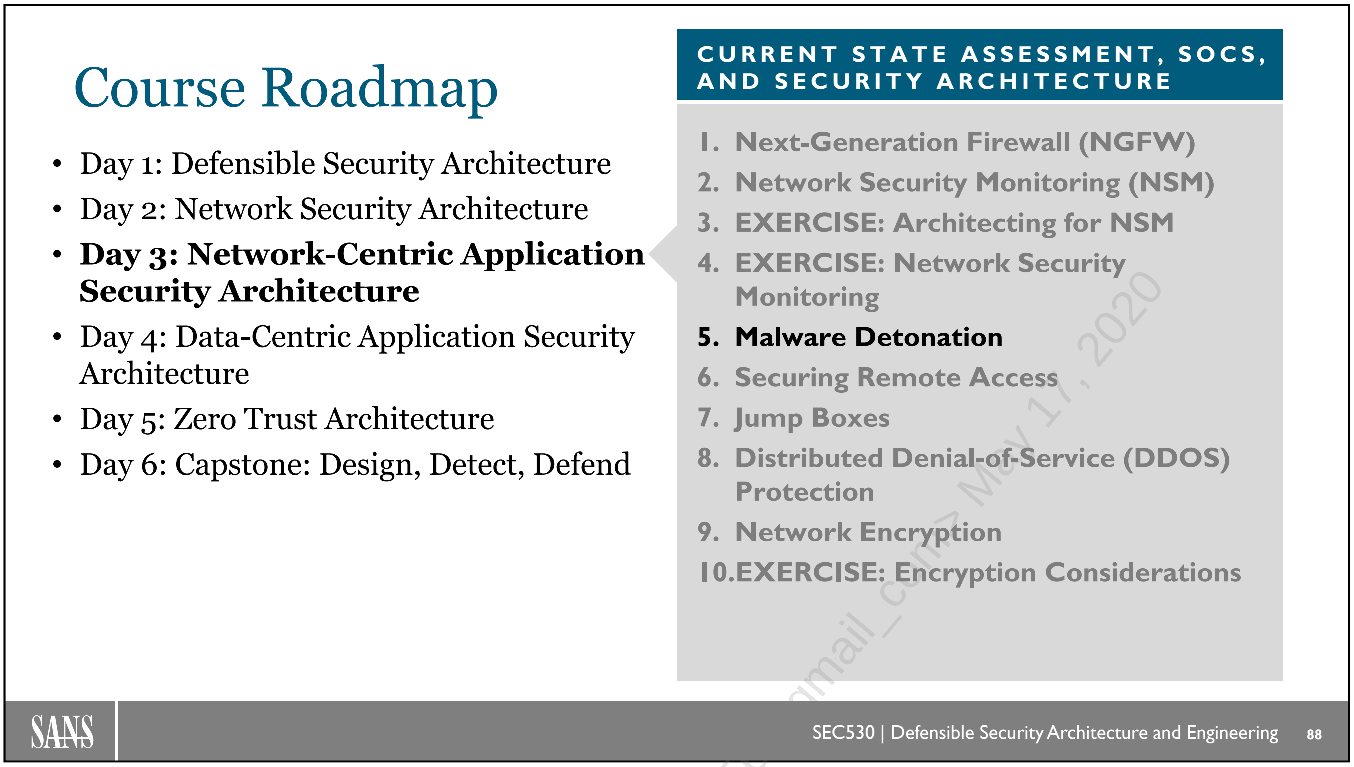
Exercise 3.2: Network Security Monitoring

- Exercise 3.2 is in the digital wiki found in your student VM (recommended)
- Alternatively, you may use your Workbook

SANS

SEC530 | Defensible Security Architecture and Engineering 87

Please open the digital wiki or your lab workbook and access exercise 3.2.



Course Roadmap

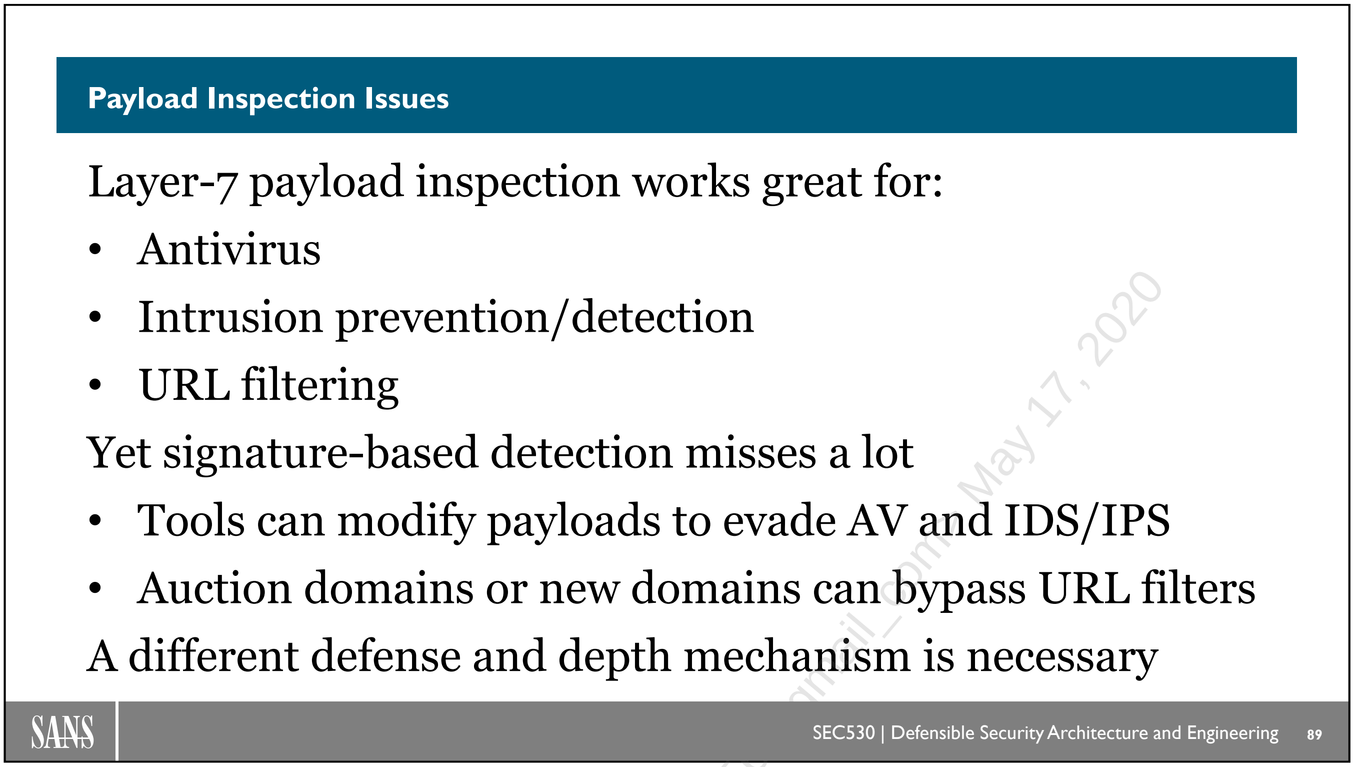
- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. **Malware Detonation**
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

Course Roadmap

The next section covers Malware Detonation.



Payload Inspection Issues

Layer-7 payload inspection works great for:

- Antivirus
- Intrusion prevention/detection
- URL filtering

Yet signature-based detection misses a lot

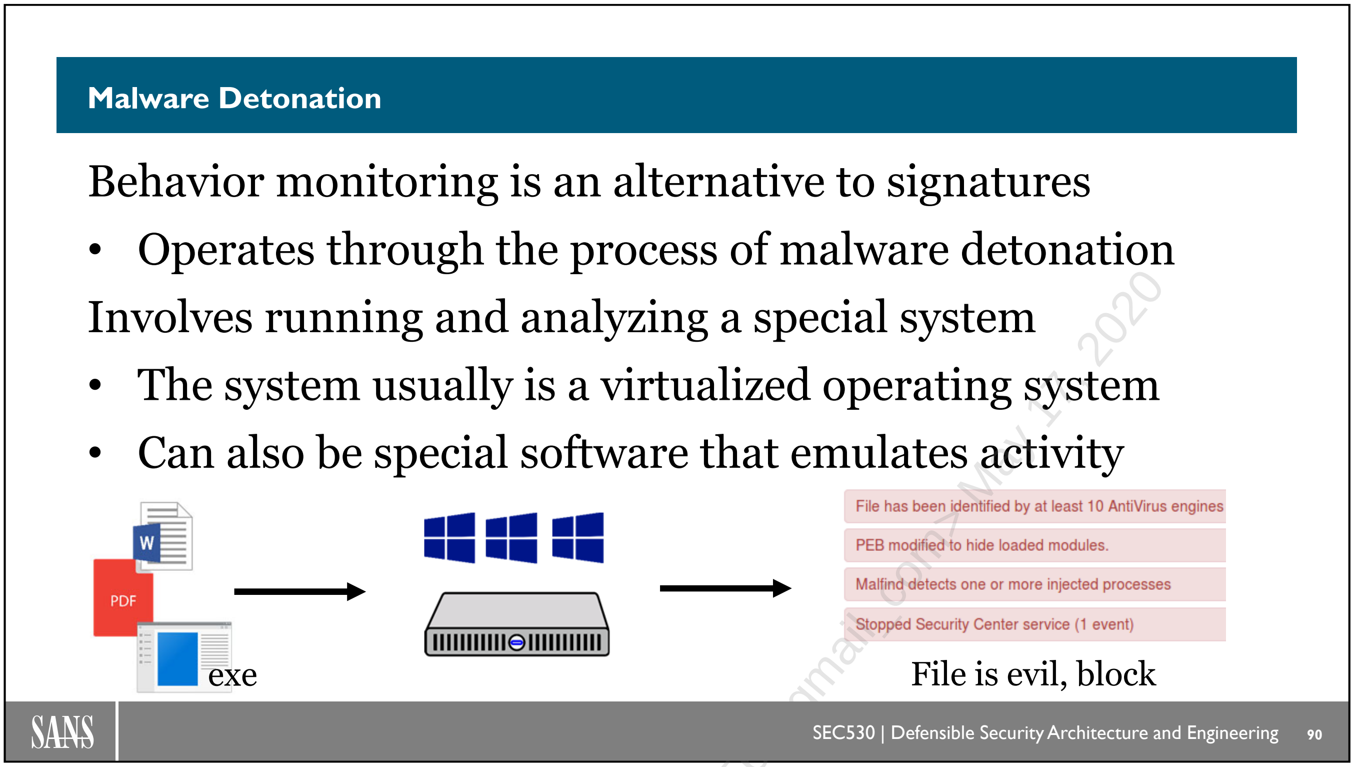
- Tools can modify payloads to evade AV and IDS/IPS
- Auction domains or new domains can bypass URL filters

A different defense and depth mechanism is necessary

Payload Inspection Issues

Many of the previous solutions work primarily using layer four through seven inspections. However, even the layer seven inspections can also fall short. Signatures look for specific things of interest. The problem is evasion tools are prevalent and easy to use making signature matching flawed. Given enough time it is possible to evade any signature-based tool.

All-in-all this represents the normal attack/defense cycle and why defense in depth is so critical.



Malware Detonation

Behavior monitoring is an alternative to signatures

- Operates through the process of malware detonation

Involves running and analyzing a special system

- The system usually is a virtualized operating system
- Can also be special software that emulates activity

File has been identified by at least 10 AntiVirus engines

PEB modified to hide loaded modules.

Malfind detects one or more injected processes

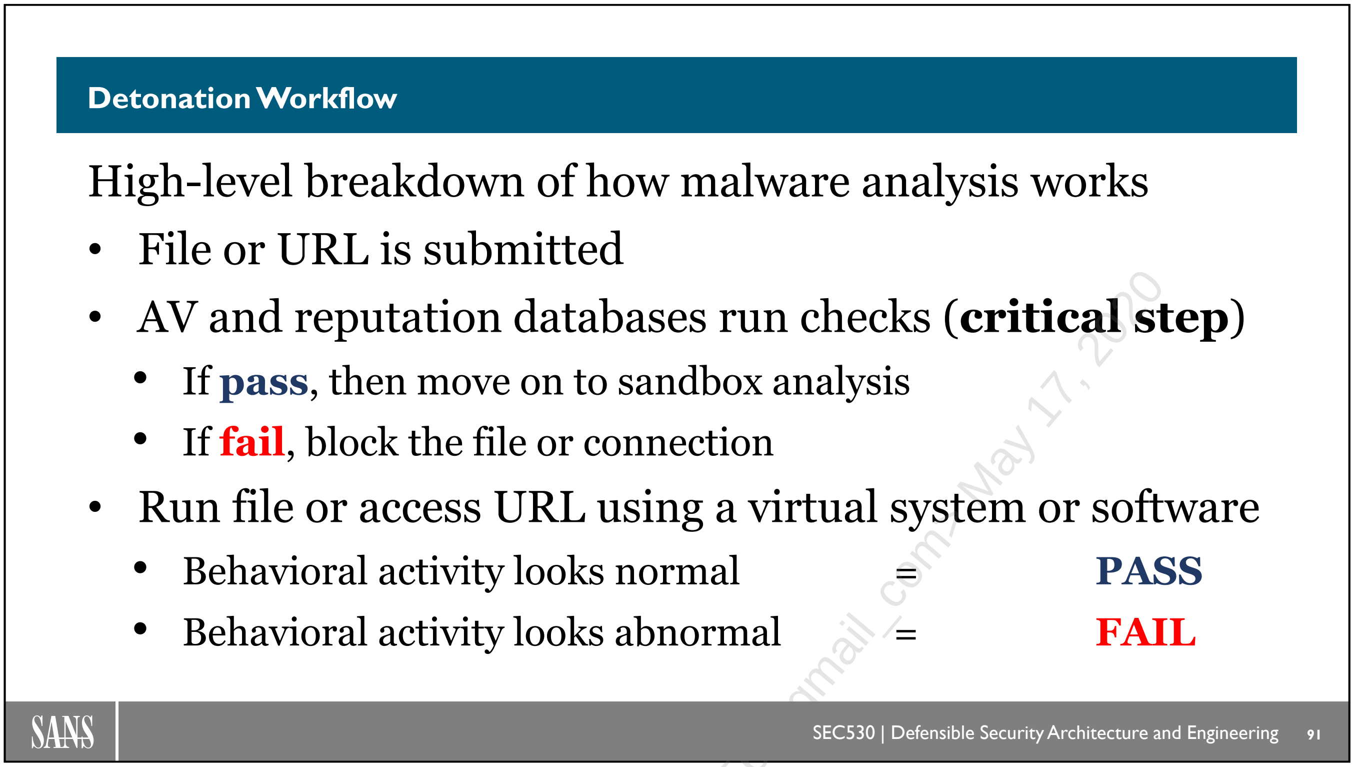
Stopped Security Center service (1 event)

File is evil, block

Malware Detonation

An alternative to signature-based rules is behavioral analytics. Behavioral analytics work by studying the actions and results resulting from something occurring. One method of performing automatic behavior analysis is using malware detonation devices. A malware detonation device functions by running potential malware in a special sandbox environment. Typically, the sandbox environment is a virtual machine farm.

Potential malware goes into the detonation system. Virtual machines and special analysis software monitor the execution of the specimen and then analyze the results. If the behaviors recorded show signs of malice, the specimen is marked as malicious and action is taken.



Detonation Workflow

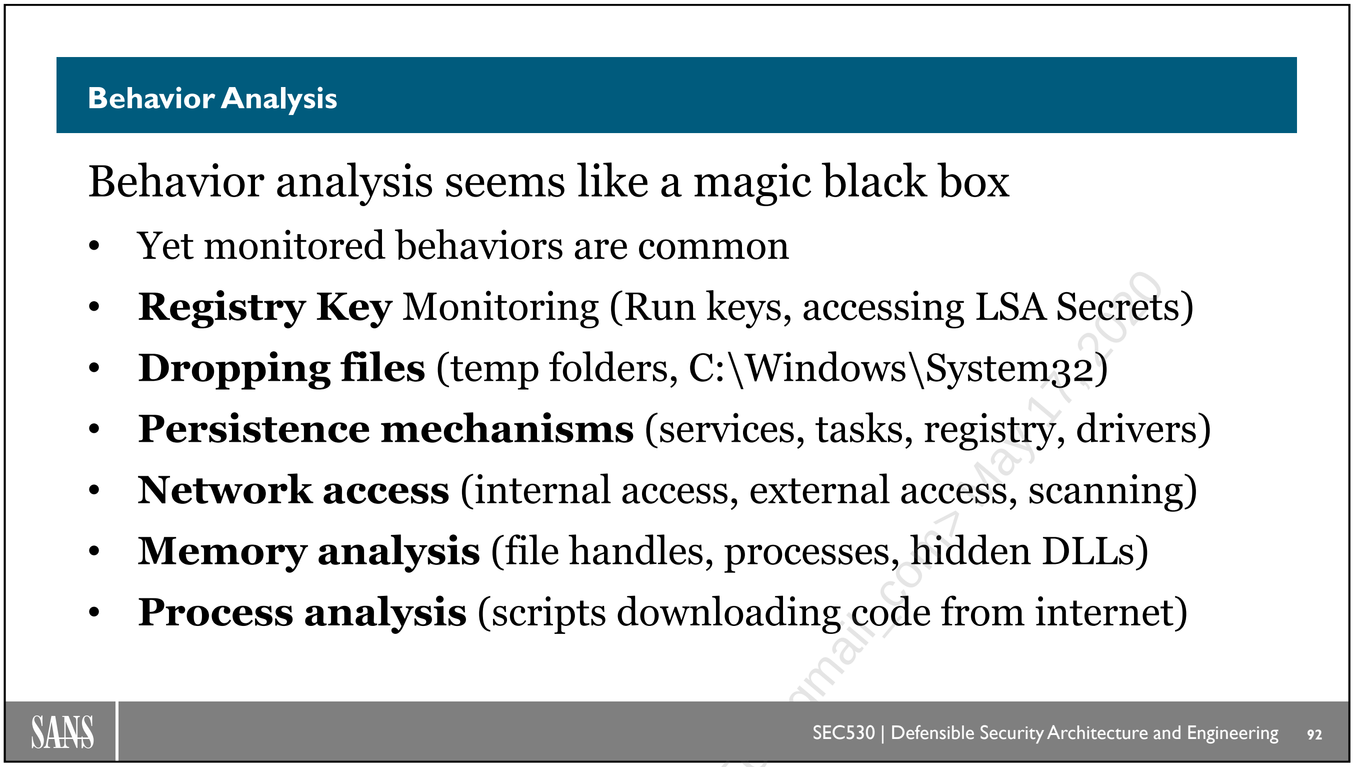
High-level breakdown of how malware analysis works

- File or URL is submitted
- AV and reputation databases run checks (**critical step**)
  - If **pass**, then move on to sandbox analysis
  - If **fail**, block the file or connection
- Run file or access URL using a virtual system or software
  - Behavioral activity looks normal = **PASS**
  - Behavioral activity looks abnormal = **FAIL**

Detonation Workflow

Malware detonation devices do not analyze every potential specimen. The reason for not analyzing everything is performance. Behavioral analysis is an intense process that requires lots of CPU, memory, and disk IO. Also, analyzing every specimen is not practical.

Files or URLs are first checked to see if they are known bad actors via antivirus signatures, reputation databases, and any other means of identifying known bad items. Then, files or URLs that are not identified as known bad get submitted for malware detonation. The result is an optimized performance without the risk of a true negative. Files and URLs only are analyzed as necessary.



Behavior Analysis

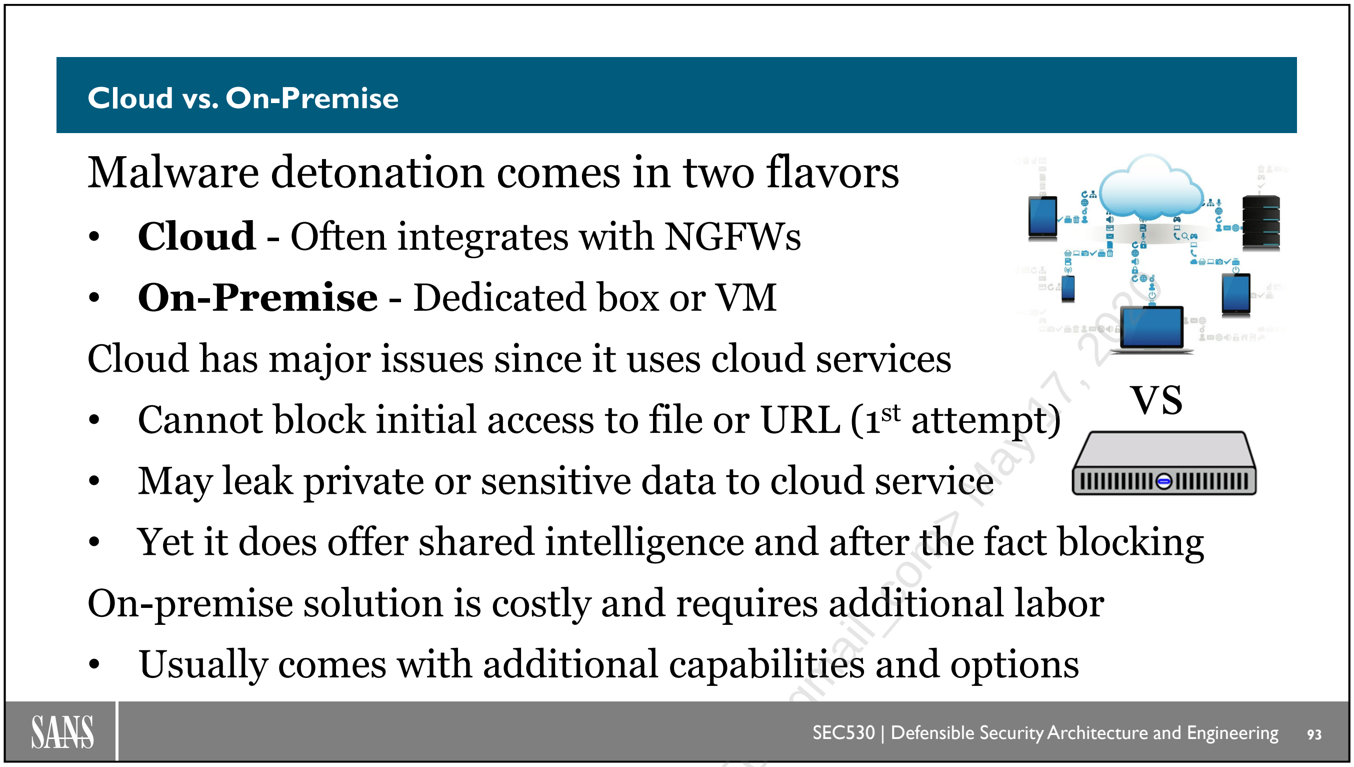
Behavior analysis seems like a magic black box

- Yet monitored behaviors are common
- **Registry Key** Monitoring (Run keys, accessing LSA Secrets)
- **Dropping files** (temp folders, C:\Windows\System32)
- **Persistence mechanisms** (services, tasks, registry, drivers)
- **Network access** (internal access, external access, scanning)
- **Memory analysis** (file handles, processes, hidden DLLs)
- **Process analysis** (scripts downloading code from internet)

Behavior Analysis

Malware detonation devices seem like magical black boxes. However, all malware detonation does is analyze the results of launching a specimen or visiting a URL. The actions taken can be normal such as Microsoft Word opening a document to highly abnormal such as Microsoft Word launching powershell.exe which then reaches out to the internet to download additional code, becomes SYSTEM, and then accesses the SAM database.

Analysis of actions taken is automated and easy to follow. The behaviors monitored are traditional malware analysis steps. What registry keys and files were accessed, modified, or created? Was persistence established? What remote connections were made and where to? What processes are involved and are the processes doing anything out of the norm?



Cloud vs. On-Premise

Malware detonation comes in two flavors

- **Cloud** - Often integrates with NGFWs
- **On-Premise** - Dedicated box or VM

Cloud has major issues since it uses cloud services

- Cannot block initial access to file or URL (1st attempt)
- May leak private or sensitive data to cloud service
- Yet it does offer shared intelligence and after the fact blocking

On-premise solution is costly and requires additional labor

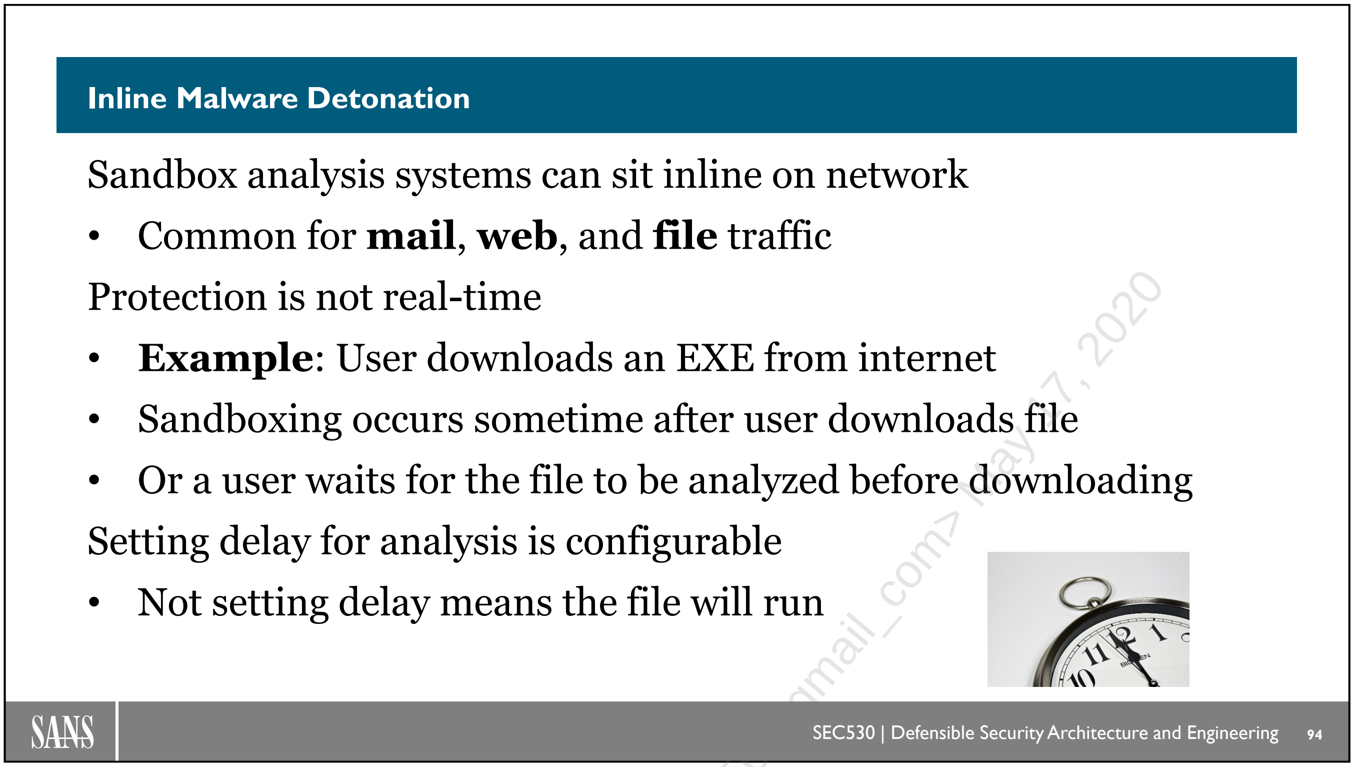
- Usually comes with additional capabilities and options

SANS | SEC530 | Defensible Security Architecture and Engineering 93

Cloud vs. On-Premise

Many NGFWs and modern IDS solutions integrate with cloud-based malware detonation services. The capability to have content traversing a firewall or IDS sent for malware detonation is an awesome capability. However, a cloud-based detonation service is not the same as an on-premise solution. The main difference is that cloud-based integration must upload content to the cloud and wait for results. Uploading to the cloud has a couple of key issues. First, sensitive company content such as PDFs and document files may be uploaded to the cloud service, if this data is made available to all customers of the cloud analysis solution, this can lead to an unintended data breach. The main issue with cloud integration is that the first time a specimen is observed it cannot be delayed and blocked as necessary.

On-premise offsets the weaknesses of cloud integration by allowing faster analysis, custom virtual machines, and not sharing files or URLs submitted. The downside of on-premise is that it is more expensive and requires labor to set up properly. A huge advantage of on-premise solutions is the support of inline blocking. Inline blocking capabilities mean the first time a specimen is observed that it can be analyzed and blocked before an end user can open it. Another major consideration for using an on-premise solution is that submissions to a cloud service can lead to an adversary knowing their content has been identified as malicious. This identification is possible if the content is submitted to a system that shares reports on file hashes or any other content metadata.



Inline Malware Detonation

Sandbox analysis systems can sit inline on network

- Common for **mail**, **web**, and **file** traffic

Protection is not real-time

- **Example**: User downloads an EXE from internet
- Sandboxing occurs sometime after user downloads file
- Or a user waits for the file to be analyzed before downloading

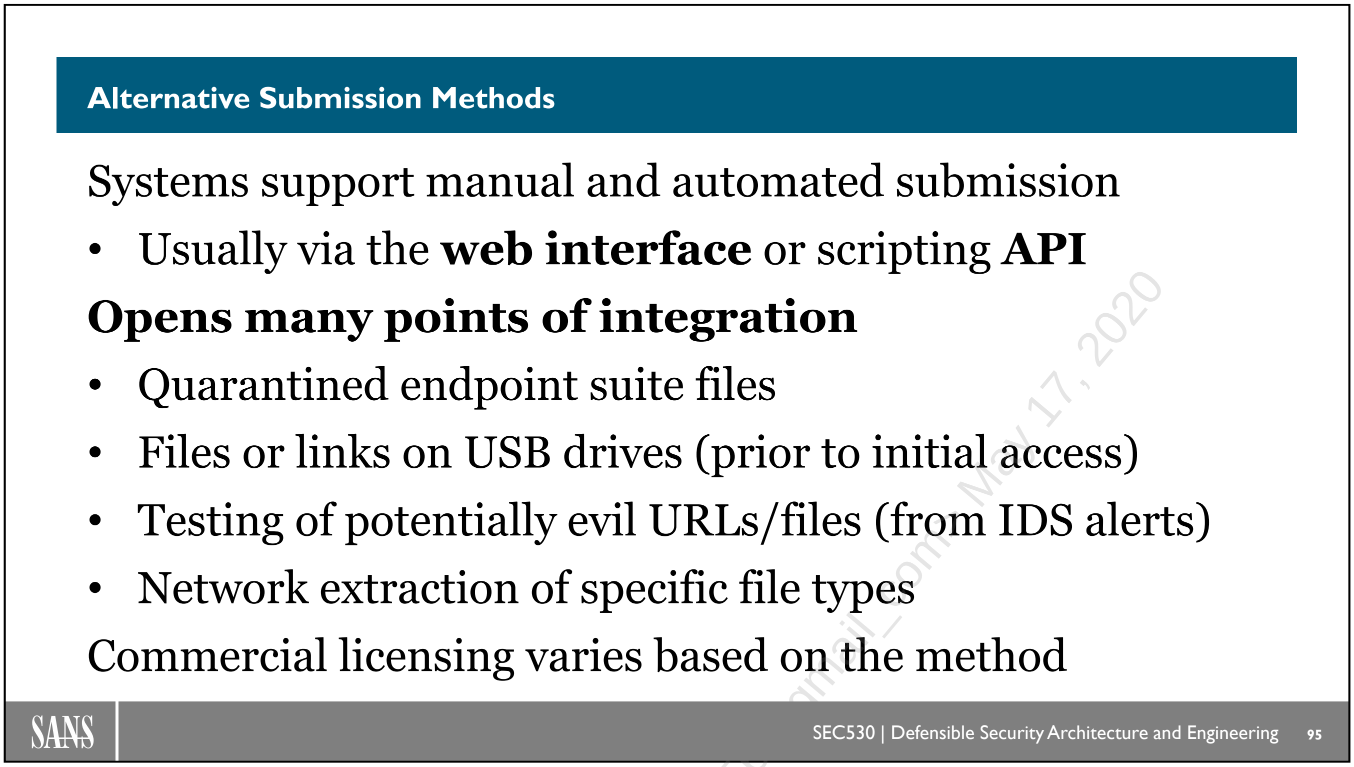
Setting delay for analysis is configurable

- Not setting delay means the file will run

Inline Malware Detonation

For inline malware detonation to work in block mode, a delay option must be enabled. Depending on the malware detonation system you are using this option may or may not exist. A delay is required to allow full analysis of a specimen prior to the endpoint system receiving it. For example, if a user is downloading an executable from the internet one of two things can happen. First, the file download could be allowed as normal. This runs the risk that the end user will run it prior to knowing if the contents of the file are malicious or not. A second option is enforcing a delay. With a delay enforced the end system must wait for analysis to complete prior to having access to a file. With a delay, if the file is found to be malicious, then the end user is notified, and the file is never sent to the endpoint. If the file is benign, then it is transferred after the analysis completes.

When enabling enforced delay detonation, make sure end users are aware of the change prior to it taking place. Ideally, a message will be displayed notifying why the file is taking longer to download than normal. If the detonation system integration does not support this, educate end users why downloads are taking longer than normal. Delays operate a little different depending on the transport mechanism such as mail through Exchange, web-based access via a proxy, or file transfers over SMB.



Alternative Submission Methods

Systems support manual and automated submission

- Usually via the **web interface** or scripting **API**

Opens many points of integration

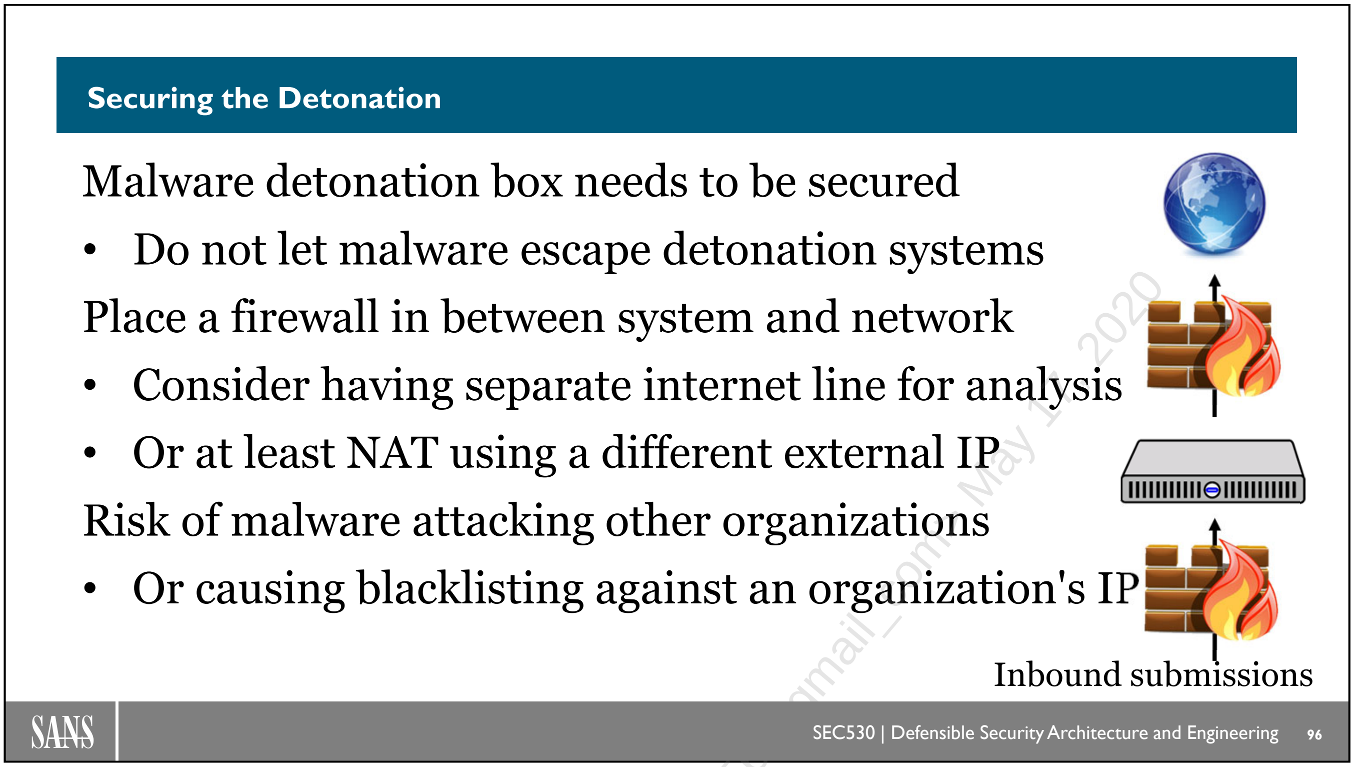
- Quarantined endpoint suite files
- Files or links on USB drives (prior to initial access)
- Testing of potentially evil URLs/files (from IDS alerts)
- Network extraction of specific file types

Commercial licensing varies based on the method

Alternative Submission Methods

Malware detonation devices support multiple methods of sample submission. The most common is automatic submission through network monitoring. As traffic is observed, items are pulled out of the network and analyzed. Two alternative solutions are manual submission via a web interface or through an API. A web interface is great for manually testing something. Yet the full power of a malware detonation system is operationalized via the API. Hooking into the API allows for automated analysis and protection for non-inline risks. For example, scripts could automatically submit files from a newly plugged in a USB drive and potentially delay the user until the analysis is complete. A remote location that cannot afford an inline malware detonation system can have files remotely submitted to a central appliance or cloud system.

Some commercial malware detonation boxes allow unlimited inline analysis but limit the number of times the API interface can be used within a set time period. The count and time limitation is most often true for cloud integration from an NGFW. However, if inline capture and analysis is unlimited, an organization still has a method to submit samples automatically. Instead of submitting files over the API interface stand up a dedicated network file share that uses a cleartext file share protocol and then have scripts copy the files to the server. Since traffic must traverse the firewall to this file server, the inline analysis mechanism will be triggered.



Securing the Detonation

Malware detonation box needs to be secured

- Do not let malware escape detonation systems

Place a firewall in between system and network

- Consider having separate internet line for analysis
- Or at least NAT using a different external IP

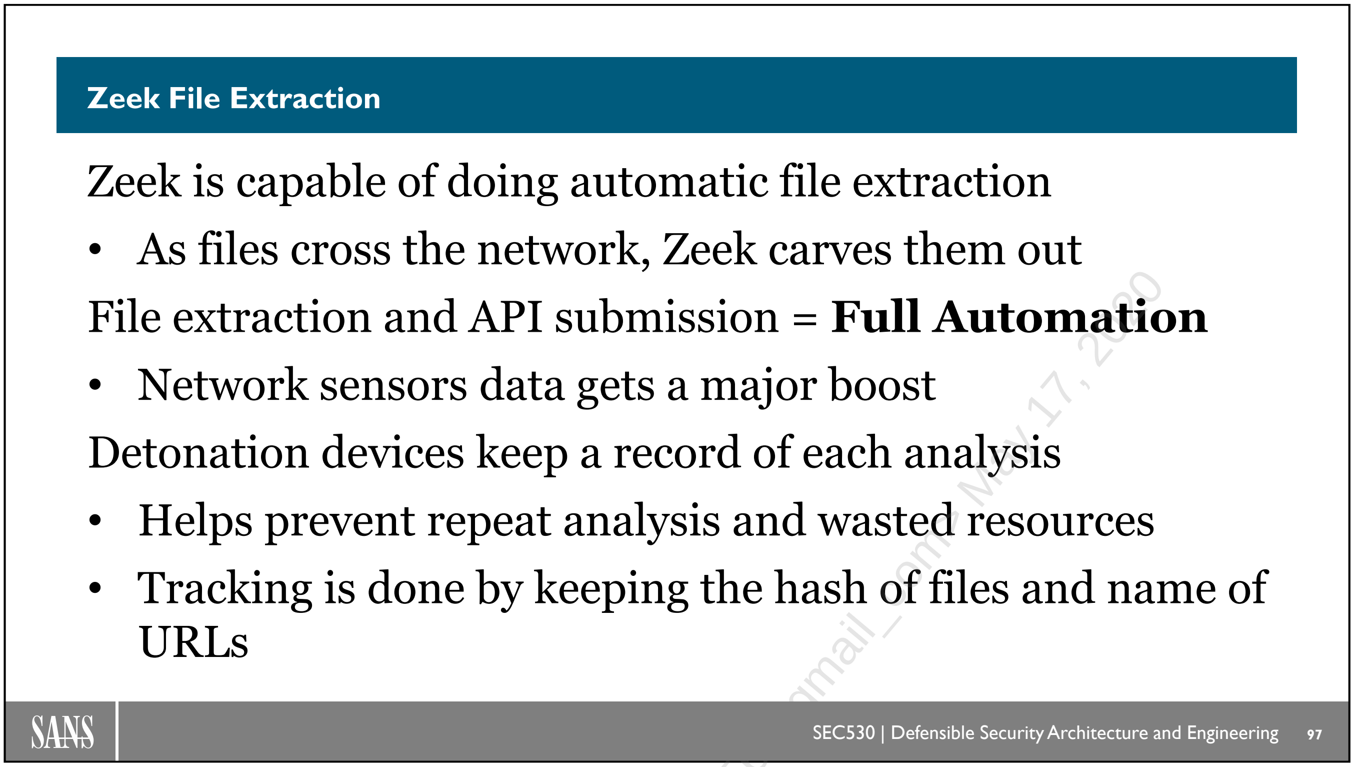
Risk of malware attacking other organizations

- Or causing blacklisting against an organization's IP

Securing the Detonation

The intent behind malware detonation is to run a potentially malicious specimen. Because the specimen is malicious, it needs to be handled with care. A good practice to follow is to place a malware detonation service behind a firewall. The firewall should apply restrictions preventing malware within the virtual machine sandbox systems from coming into the internal network. The firewall should also restrict what all can be done outbound to the internet. Should a piece of malware reach out and infect other organizations there would be liability issues. However, not having internet access at all may cause the malware to hide.

If internet access is provided to detonation virtual machines, then consider using a dedicated internet line or external IP address for malware analysis. The reason for a dedicated line or IP is that the traffic from malware analysis out to the internet could cause an organization to be temporarily or permanently blocked. By placing the malware detonation services on a separate line or IP, any potential blocks would be segmented to just the malware analysis systems.



Zeek File Extraction

Zeek is capable of doing automatic file extraction

- As files cross the network, Zeek carves them out

File extraction and API submission = **Full Automation**

- Network sensors data gets a major boost

Detonation devices keep a record of each analysis

- Helps prevent repeat analysis and wasted resources
- Tracking is done by keeping the hash of files and name of URLs

SANS SEC530 | Defensible Security Architecture and Engineering 97

Zeek File Extraction

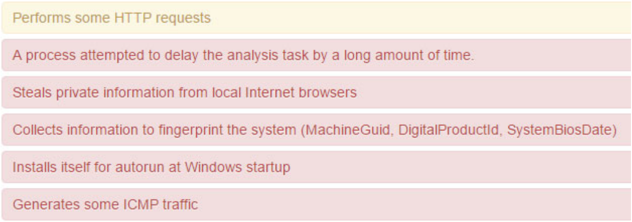
One way to boost your defense is to integrate automatic file carving with malware detonation systems. For example, Zeek supports automatic file carving. File carving means that as a file is transferred across the network, it will be extracted from the network and saved to disk. Add some simple scripting to file extraction, and now automatic malware detonation of files from remote or internal networks is possible.

When writing scripts to integrate with malware detonation APIs, try to build out so that files are only submitted if they have not been analyzed. To do this, prior to sending a file query the API to see if the hash of the file has been previously analyzed. This question is a quick yes or no and can be used only to transmit a file if it has not been seen. Only transferring the file if it is a new file keeps internal bandwidth optimized and helps to lower resource consumption of the detonation device.

malwr.com[1]

malwr.com[1] provides a free malware detonation box

- Great for ad hoc or triage analysis of alerts
- Safe(r) way to launch potentially malicious URLs or files
- Helps with the urge to open phishing email contents



Performs some HTTP requests

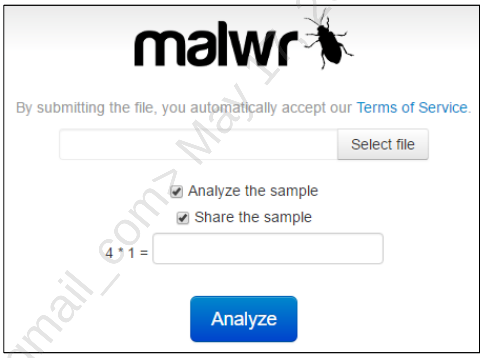
A process attempted to delay the analysis task by a long amount of time.

Steals private information from local Internet browsers

Collects information to fingerprint the system (MachineGuid, DigitalProductId, SystemBiosDate)

Installs itself for autorun at Windows startup

Generates some ICMP traffic



By submitting the file, you automatically accept our Terms of Service.

☑ Analyze the sample

☑ Share the sample

4 * 1 =

malwr.com[1]

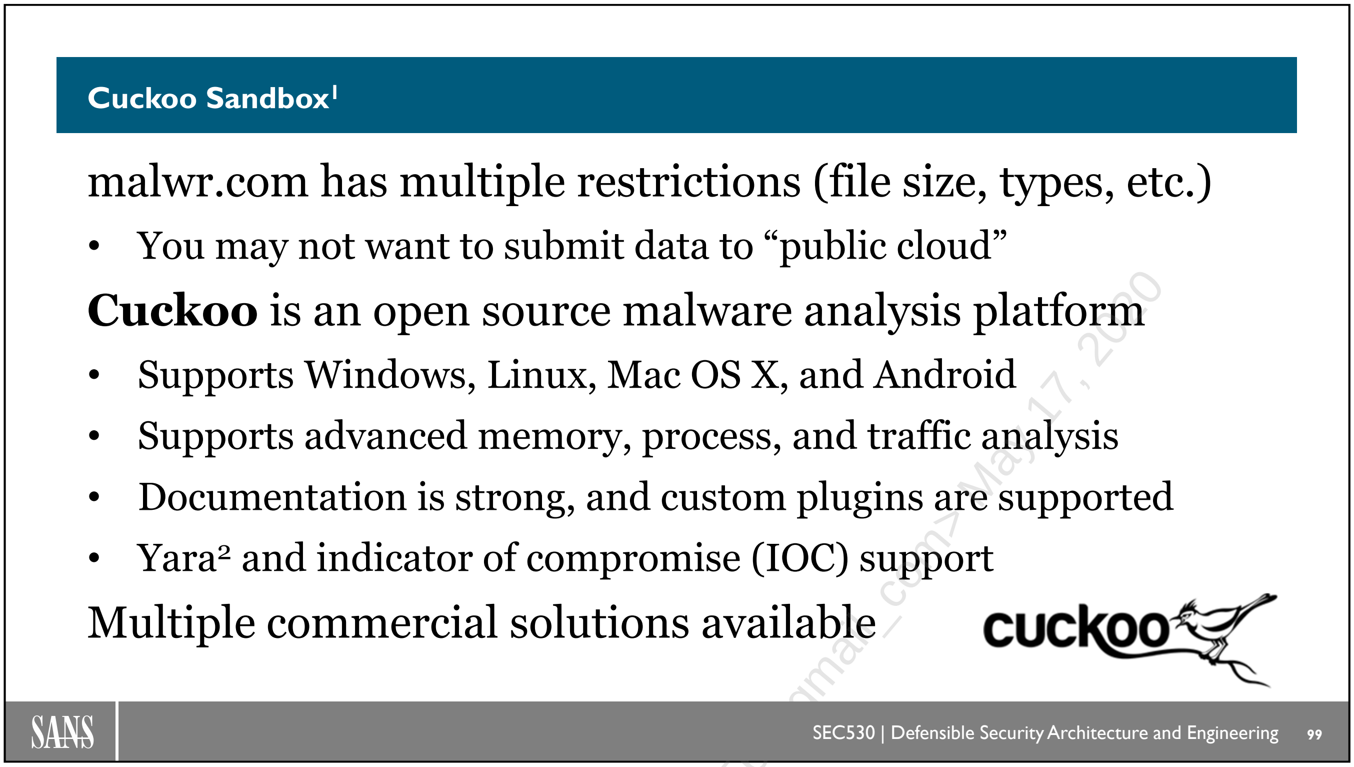
Having a billion antivirus engines scan something only gets you so far. Unfortunately, there are multiple free and easy methods to bypass antivirus. The Veil Framework[2] is an example of this, and there are many others. Running suspected malware is dangerous, but to really study something you need to run it.

Malwr.com[1] is an online site that will accept uploads of potentially malicious files. They are then executed in a virtual environment and monitored. The end report identifies if any of the behaviors were abnormal or potentially malicious. The color code text on this slide is a sample report of a malicious file. Another free alternative to malwr.com[1] is Hybrid Analysis[3].

[1] https://malwr.com

[2] https://www.veil-framework.com/

[3] https://www.hybrid-analysis.com/



Cuckoo Sandbox[1]

malwr.com has multiple restrictions (file size, types, etc.)

- You may not want to submit data to "public cloud"

Cuckoo is an open source malware analysis platform

- Supports Windows, Linux, Mac OS X, and Android
- Supports advanced memory, process, and traffic analysis
- Documentation is strong, and custom plugins are supported
- Yara[2] and indicator of compromise (IOC) support

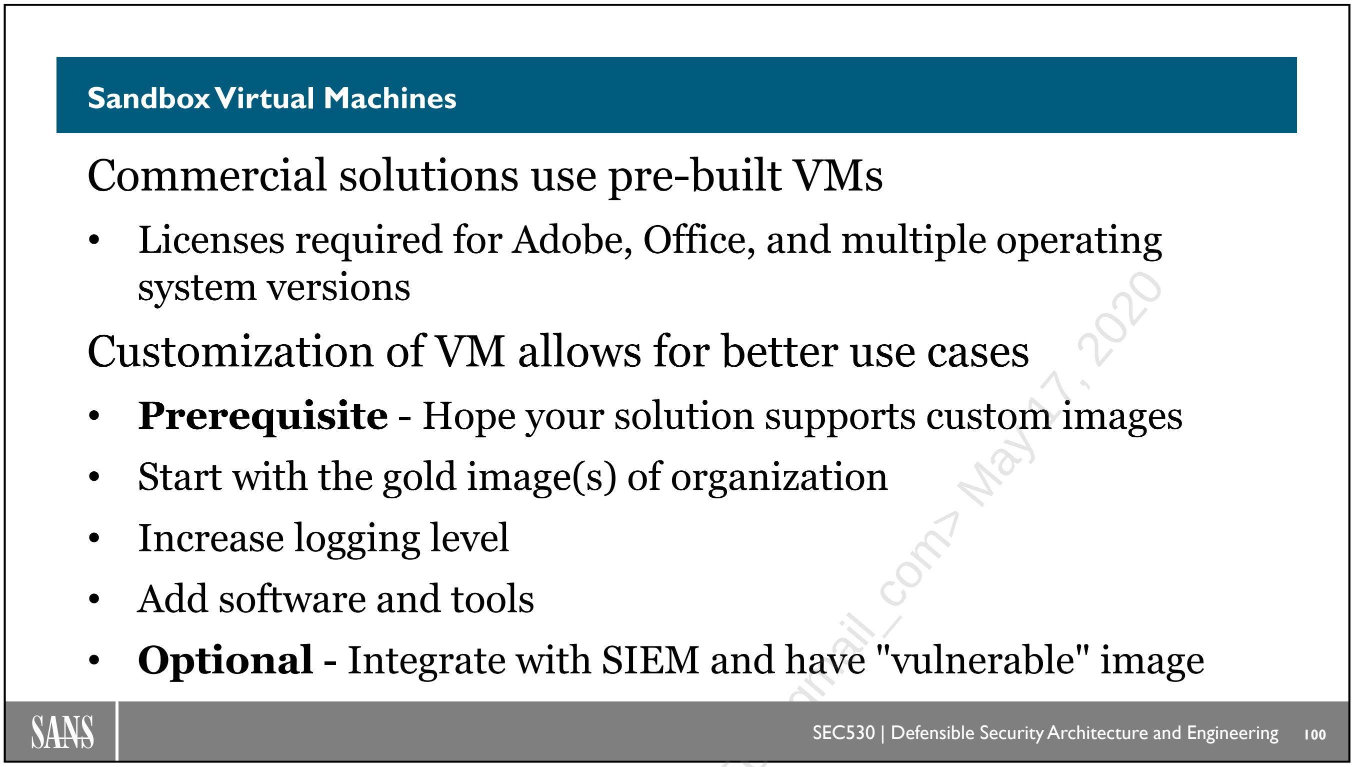
Multiple commercial solutions available

Cuckoo Sandbox[1]

An organizational policy may prevent uploading of potentially sensitive files outside the environment. To get around this, a malware analysis lab needs to be on-premise and tightly controlled with network firewalls. Malwr.com uses Cuckoo Sandbox[1] which is available to deploy on-premise. It is an open source malware analysis lab. Deploying on-premise also removes many of the restrictions malwr.com has, i.e., it will only analyze small files and certain file types. For instance, Android applications can be analyzed.

[1] https://cuckoosandbox.org/

[2] https://virustotal.github.io/yara/



Sandbox Virtual Machines

Commercial solutions use pre-built VMs

- Licenses required for Adobe, Office, and multiple operating system versions

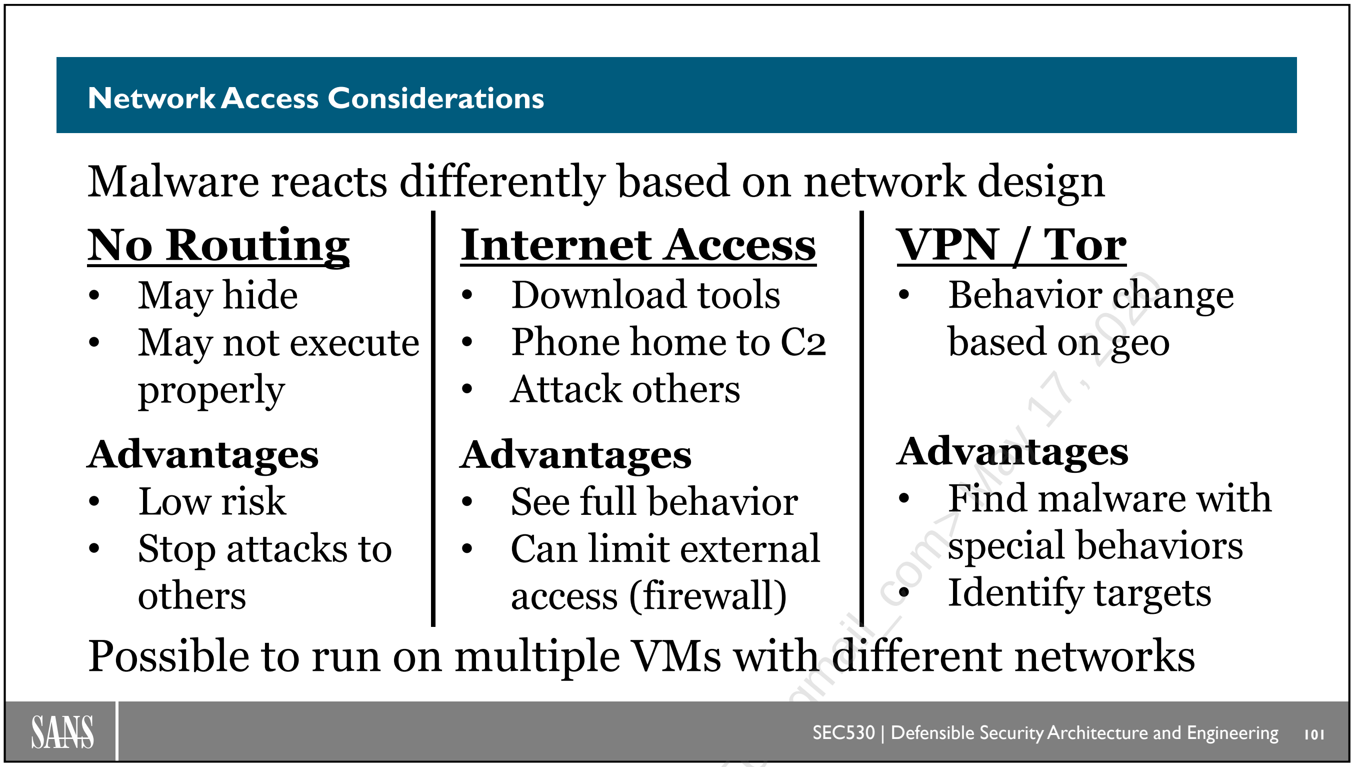
Customization of VM allows for better use cases

- **Prerequisite** - Hope your solution supports custom images
- Start with the gold image(s) of organization
- Increase logging level
- Add software and tools
- **Optional** - Integrate with SIEM and have "vulnerable" image

Sandbox Virtual Machines

The virtual machines that detonation files and URLs are important. These VMs are designed to identify abnormal behaviors. However, without the proper software and operating systems, this is not possible. For example, a malicious PDF submitted to a virtual machine without Adobe Reader is unlikely to be flagged as malicious. The same malicious PDF may only flag on specific versions of Adobe Reader. Thus, maintaining proper sandbox VMs is important. The downside to this is some commercial solutions do not allow you to customize your VM images and/or require additional licenses purchased to have things like Adobe Reader or Microsoft Office. A general recommendation is to, if possible, upload a custom VM that uses your organizations gold image or software. By doing so, malware is analyzed against your current running set.

Older versions of Windows are usually recommended as it gives that malware more opportunity to exhibit full behavior. Yet, at the same time, if Windows 10 is used and it blocks the malware from running that behavior in itself can be used to mark the specimen as malicious. Mainly, it all comes down to what level of behavior monitoring your detonation system has. If you can use a customized virtual machine, it may also be worth cranking up the logging level and shipping the logs off to a SIEM. Just a few days' worth of this data can be used for some interesting analytics. Supervised machine learning can be used against this sample set as you have a population pool of malicious vs benign data sets. These datasets can be used to identify new characteristics in logs to detect evil.



Network Access Considerations

Malware reacts differently based on network design

No Routing

- May hide
- May not execute properly

Advantages

- Low risk
- Stop attacks to others

Internet Access

- Download tools
- Phone home to C2
- Attack others

Advantages

- See full behavior
- Can limit external access (firewall)

VPN / Tor

- Behavior change based on geo

Advantages

- Find malware with special behaviors
- Identify targets

Possible to run on multiple VMs with different networks

SANS

SEC530 | Defensible Security Architecture and Engineering 101

Network Access Considerations

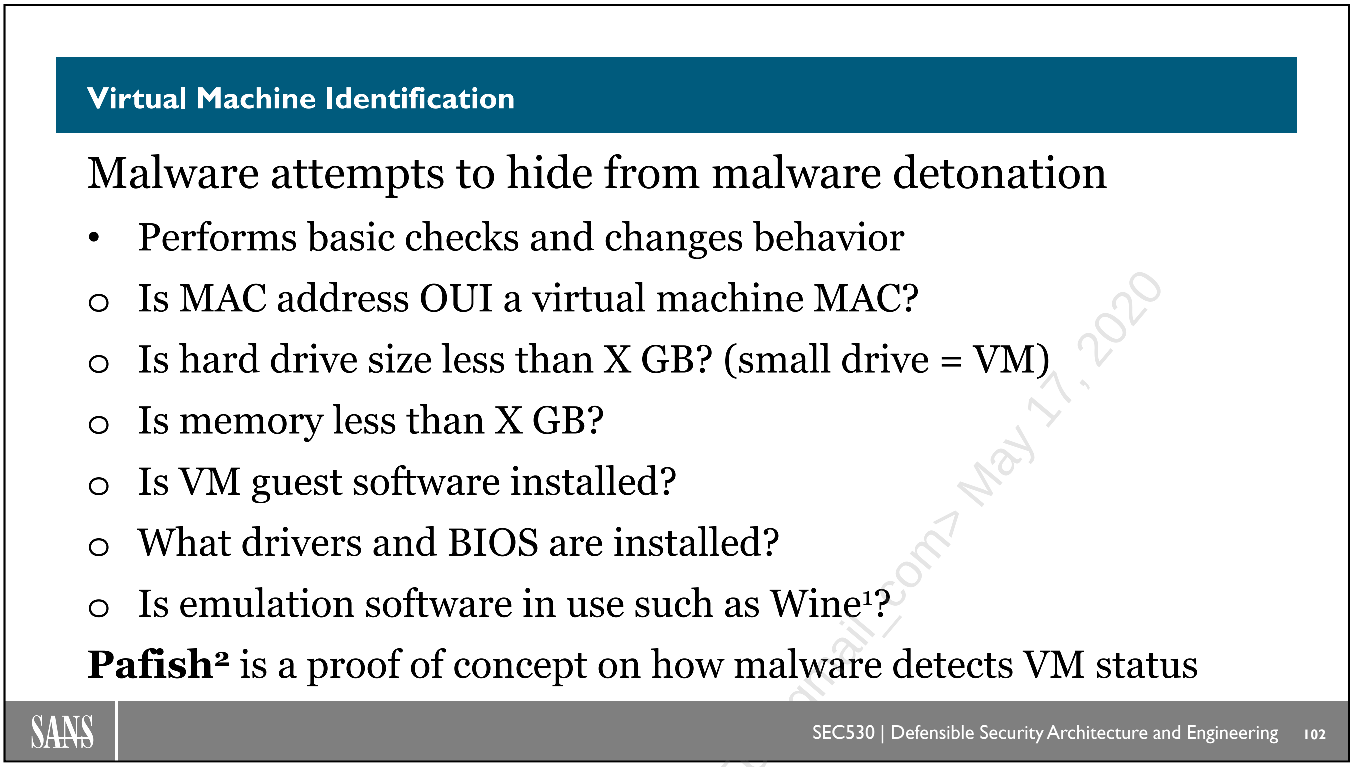
In order for malware detonation to work the specimen must be run. However, detecting evil is not that simple. Depending on surrounding circumstances malware may change behavior. For example, if no internet access is provided, then malware cannot download stage two payloads. Without these downloads, no further behaviors can be observed making it more likely to have the sample marked as benign. Yet without internet, there is no risk of malware attacking other organizations. In most cases, internet access is desired with some level of restrictions applied to it. With internet access malware is more likely to perform as intended.

However, some malware is advanced in that it will only run under extremely specific circumstances. For example, some malware will only run if it detects the external IP address it is communicating on belongs to specific geolocation or ASN. If you work for an organization that is in multiple countries, it may be worth having malware samples use a detonation system with multiple points of the outbound internet. For this VPN tunnels are used to allow internet access to come from different geolocations or ASNs.

Internet access can also be set up to emulate access. This access is often referred to as a fake internet design and uses tools like inetsim[1] and fakedns[2] to make malware think connections and services are actually available. These services will resolve all domain requests and attempt to establish connections so that malware thinks it has internet access. The goal behind this is to increase the likelihood of seeing true malware behavior.

[1] http://www.inetsim.org/

[2] https://github.com/Crypt0s/FakeDns



Virtual Machine Identification

Malware attempts to hide from malware detonation

- Performs basic checks and changes behavior
  - Is MAC address OUI a virtual machine MAC?
  - Is hard drive size less than X GB? (small drive = VM)
  - Is memory less than X GB?
  - Is VM guest software installed?
  - What drivers and BIOS are installed?
  - Is emulation software in use such as Wine[1]?

Pafish[2] is a proof of concept on how malware detects VM status

Virtual Machine Identification

Just like some forms of malware can detect what country it is running in malware can also be configured to detect if the system it is being run on is a virtual machine. A proof of concept tool called pafish[2] exists that analyzes the system it is run on to see if it is a virtual machine. This tool and similar capabilities are now within malware in the wild. This slide has multiple examples of how a system can be identified as a virtual machine. Some are specific such as looking at the drivers or BIOS version while others are educated guesses such as if the hard drive size is small. Modern computers today do not have hard drives below a threshold such as 60 or 100 GB.

The bad news is that this method of identifying if a system is a virtual machine is easy. The good news is that technologies like virtual desktops and virtualized terminal servers are becoming so prevalent that malware authors have to choose to hide from virtualized systems and potentially never run or disable the virtual machine identification checks.

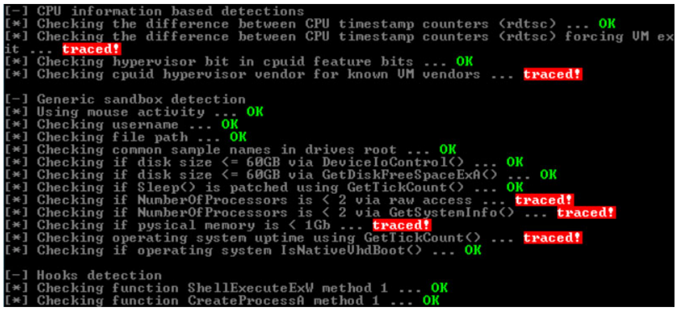
[1] https://www.winehq.org/

[2] https://github.com/a0rtega/pafish

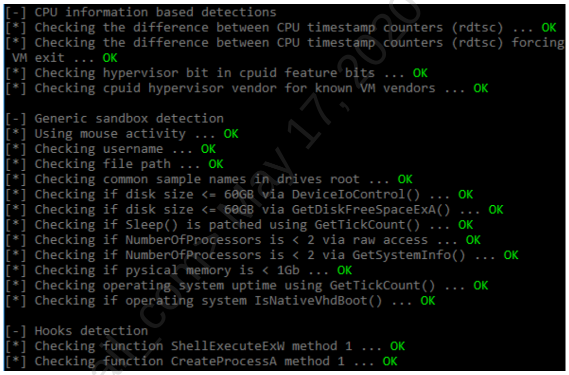
Virtual Machine Masking

Antivmdetection[1] and **VMCloak**[2] hide virtual status

- Modify sandbox images to mask they are VMs



```
[-] CPU information based detections
[*] Checking the difference between CPU timestamp counters (rdtsc) ... OK
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing VM ex
it ... traced!
[*] Checking hypervisor bit in cpuid feature bits ... OK
[*] Checking cpuid hypervisor vendor for known VM vendors ... traced!

[-] Generic sandbox detection
[*] Using mouse activity ... OK
[*] Checking username ... OK
[*] Checking file path ... OK
[*] Checking common sample names in drives root ... OK
[*] Checking if disk size <= 60GB via DeviceIoControl() ... OK
[*] Checking if disk size <= 60GB via GetDiskFreeSpaceExA() ... OK
[*] Checking if Sleep() is patched using GetTickCount() ... OK
[*] Checking if NumberOfProcessors is < 2 via raw access ... traced!
[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... traced!
[*] Checking if pysical memory is < 1Gb ... traced!
[*] Checking operating system uptime using GetTickCount() ... traced!
[*] Checking if operating system IsNativeVhdBoot() ... OK

[-] Hooks detection
[*] Checking function ShellExecuteExW method 1 ... OK
[*] Checking function CreateProcessA method 1 ... OK
```



```
[-] CPU information based detections
[*] Checking the difference between CPU timestamp counters (rdtsc) ... OK
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing
 VM exit ... OK
[*] Checking hypervisor bit in cpuid feature bits ... OK
[*] Checking cpuid hypervisor vendor for known VM vendors ... OK

[-] Generic sandbox detection
[*] Using mouse activity ... OK
[*] Checking username ... OK
[*] Checking file path ... OK
[*] Checking common sample names in drives root ... OK
[*] Checking if disk size <= 60GB via DeviceIoControl() ... OK
[*] Checking if disk size <= 60GB via GetDiskFreeSpaceExA() ... OK
[*] Checking if Sleep() is patched using GetTickCount() ... OK
[*] Checking if NumberOfProcessors is < 2 via raw access ... OK
[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo() ... OK
[*] Checking if pysical memory is < 1Gb ... OK
[*] Checking operating system uptime using GetTickCount() ... OK
[*] Checking if operating system IsNativeVhdBoot() ... OK

[-] Hooks detection
[*] Checking function ShellExecuteExW method 1 ... OK
[*] Checking function CreateProcessA method 1 ... OK
```

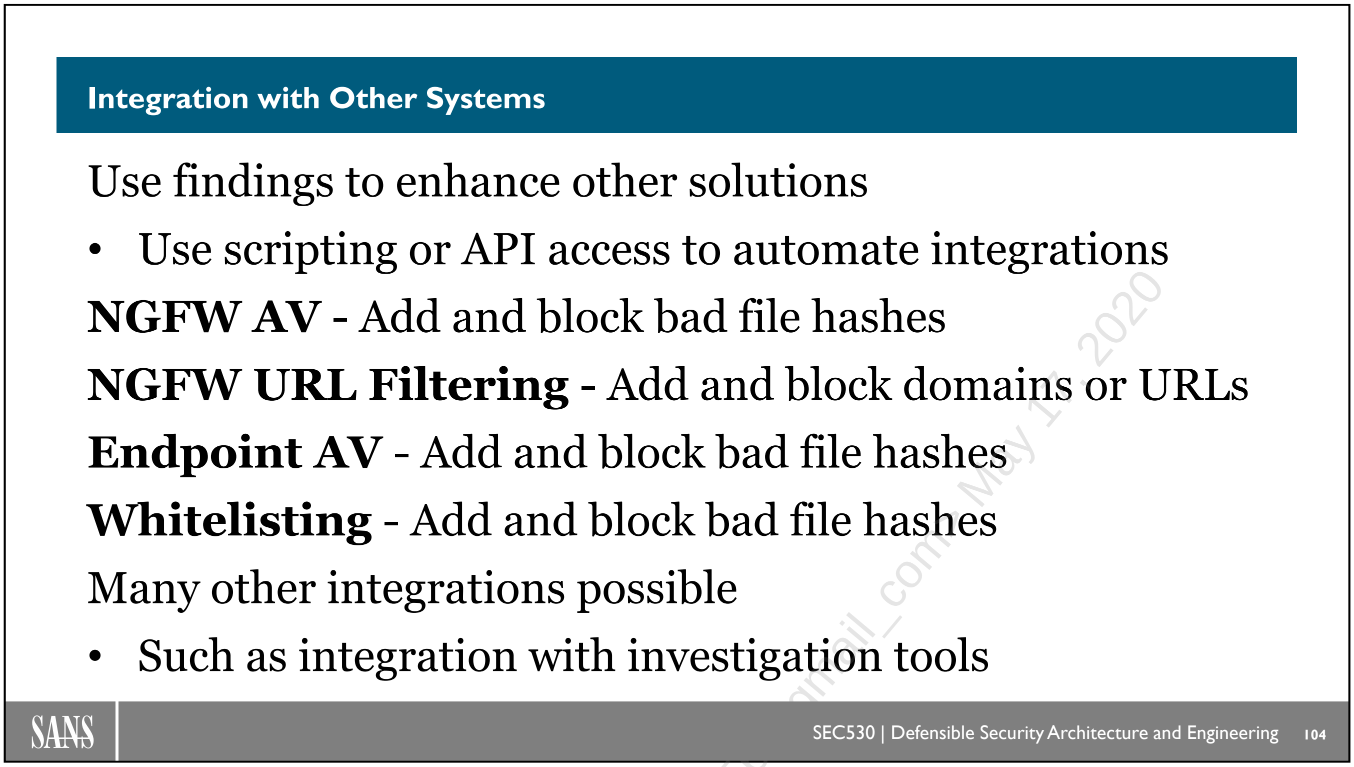
↑ From this

To this ↗

Virtual Machine Masking

Just because adversaries can check to see if a system is a virtual machine does not mean defenders cannot do anything about it. In truth, multiple methods exist to get around virtual machine identification checks. Two such methods are using antivmdetection[1] or vmcloak[2]. Both of these projects use scripts to modify a virtual machine so that it does not look like a virtual machine. Some of the items are recommendations such as disk sizing and memory sizing, but most are registry keys and file modifications to trick tools like pafish.

The two pictures in this slide are pafish being run on a standard virtual machine and pafish being run on a virtual machine that uses antivmdetection to hide the fact that it is a virtual machine.



Integration with Other Systems

Use findings to enhance other solutions

- Use scripting or API access to automate integrations

NGFW AV - Add and block bad file hashes

NGFW URL Filtering - Add and block domains or URLs

Endpoint AV - Add and block bad file hashes

Whitelisting - Add and block bad file hashes

Many other integrations possible

- Such as integration with investigation tools

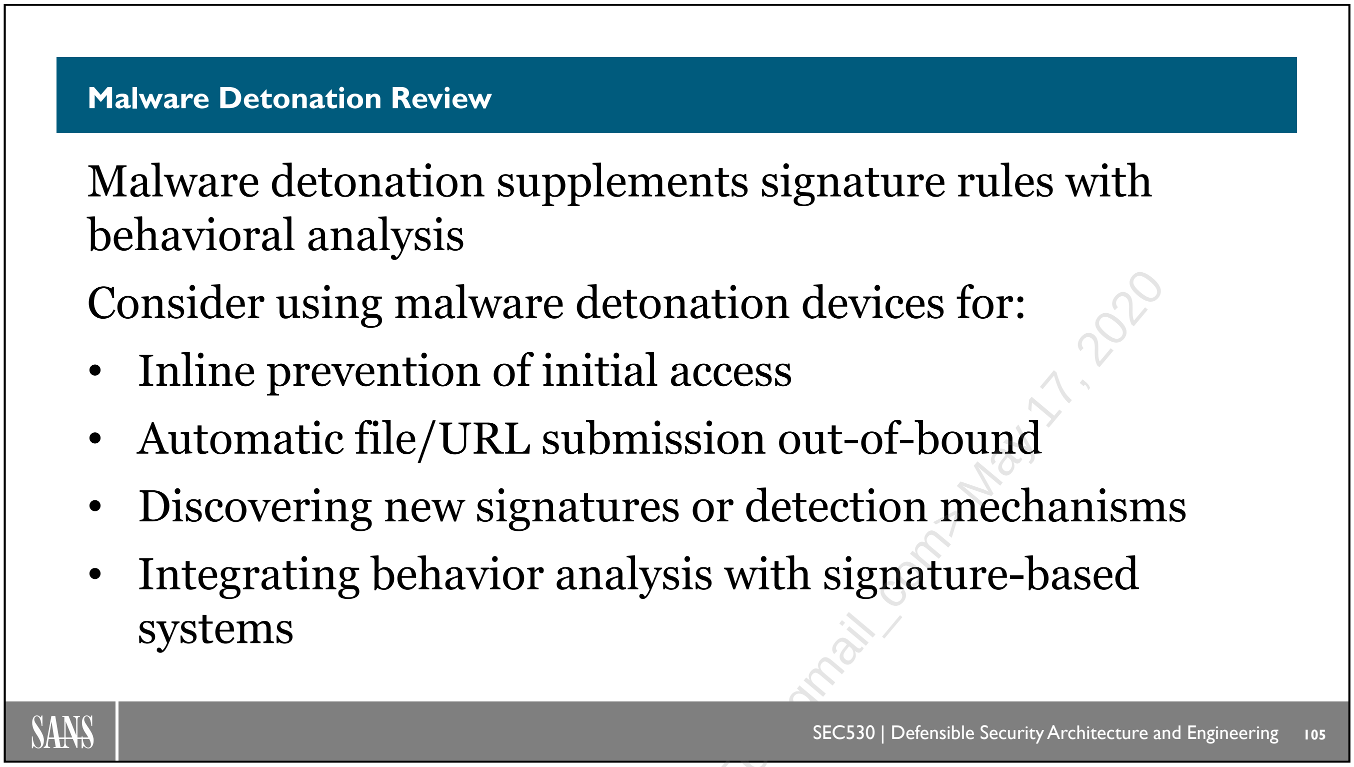
Integration with Other Systems

One of the largest advantages a malware detonation system provides is the ability to find malicious behaviors and then block access to files or URLs across multiple systems. If an organization purchases multiple devices from the same vendor, then integration is likely automatic. For example, an NGFW, an endpoint antivirus suite, will often integrate and block items found as malicious from a malware detonation system that is from the same vendor. Auto-blocking should be applied with caution. Without whitelisting authorized sites, it is possible to see malware connecting to legitimate sites and block them along with the actual evil sites.

However, if you own a malware detonation service or device you can build integration into the same type of products. For example, a simple script could take all the file hashes found as evil in a malware detonation system and use an API or SSH to place them inside an NGFW or endpoint suite. In fact, even if no endpoint protection was owned a script could create a group policy and block the hashes of known bad files using AppLocker[1] or Software Restriction Policies[2] which are built into Windows for free. A small script can go a long way in securing your organization.

[1] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/dd759117(v=ws.11)

[2] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/hh831534(v=ws.11)



Malware Detonation Review

Malware detonation supplements signature rules with behavioral analysis

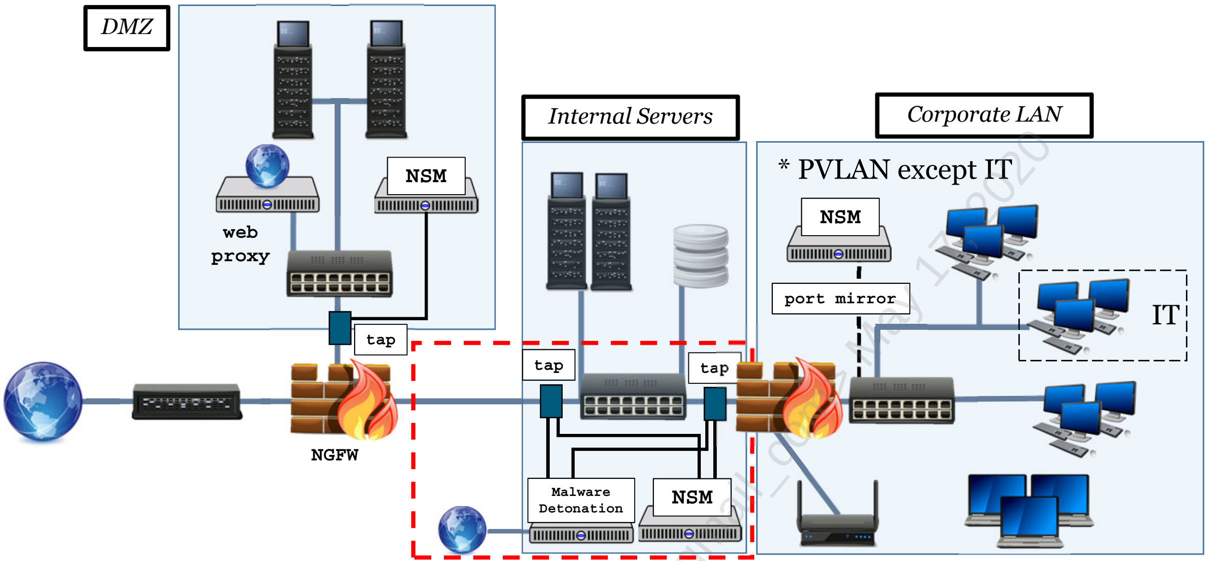
Consider using malware detonation devices for:

- Inline prevention of initial access
- Automatic file/URL submission out-of-bound
- Discovering new signatures or detection mechanisms
- Integrating behavior analysis with signature-based systems

Malware Detonation Review

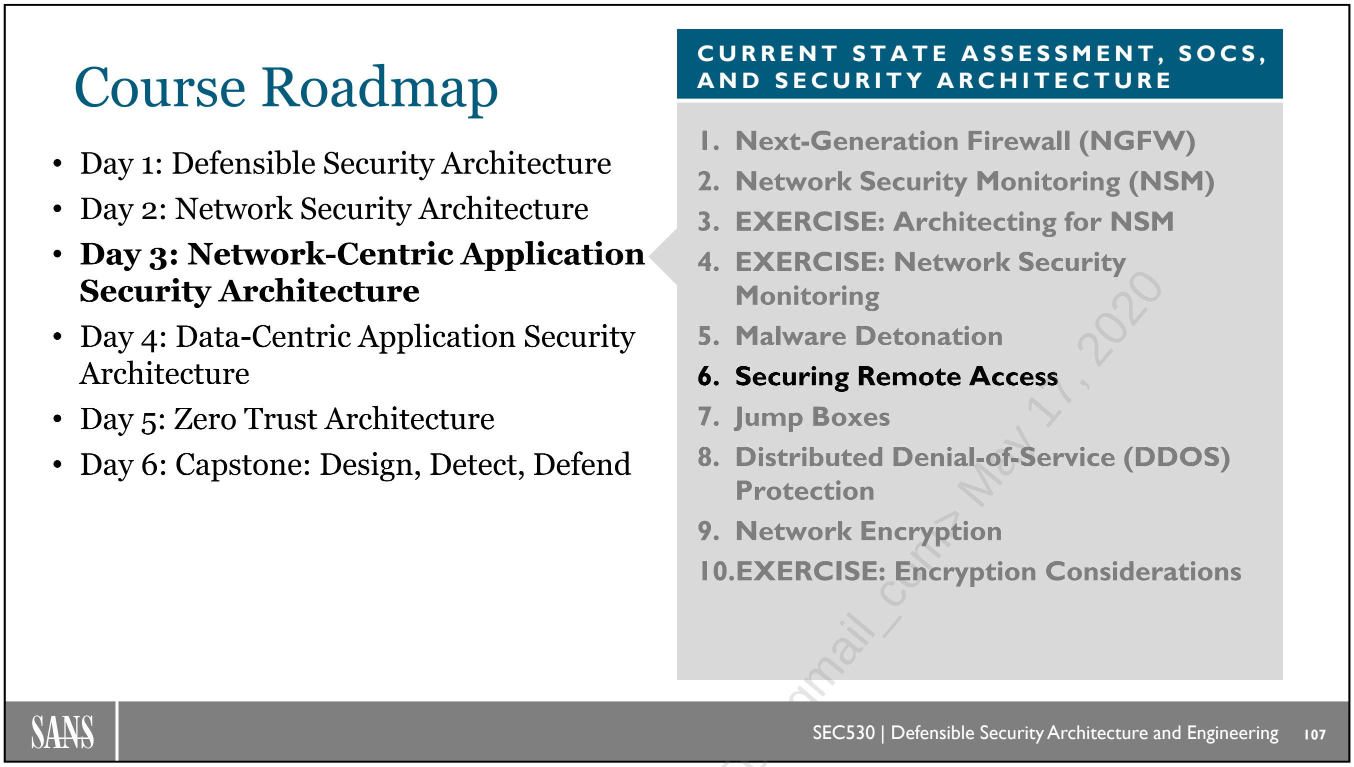
Malware detonation devices and services are incredibly powerful. Because signatures are not necessary, a detonation system is capable of identifying and finding abnormal or evil specimens. Upon finding bad files or URLs, a malware detonation box can automatically block and detect subsequent requests and even integrate with other systems to extend the value of a malware detonation system.

Case Study: Tyrrell Corporation



Case Study: Tyrrell Corporation

This diagram shows the addition of a malware detonation device to the Tyrrell Corporation architecture. In this example, content can be submitted to the malware detonation device as well as automatically extracted through network traffic visible from network taps.



Course Roadmap

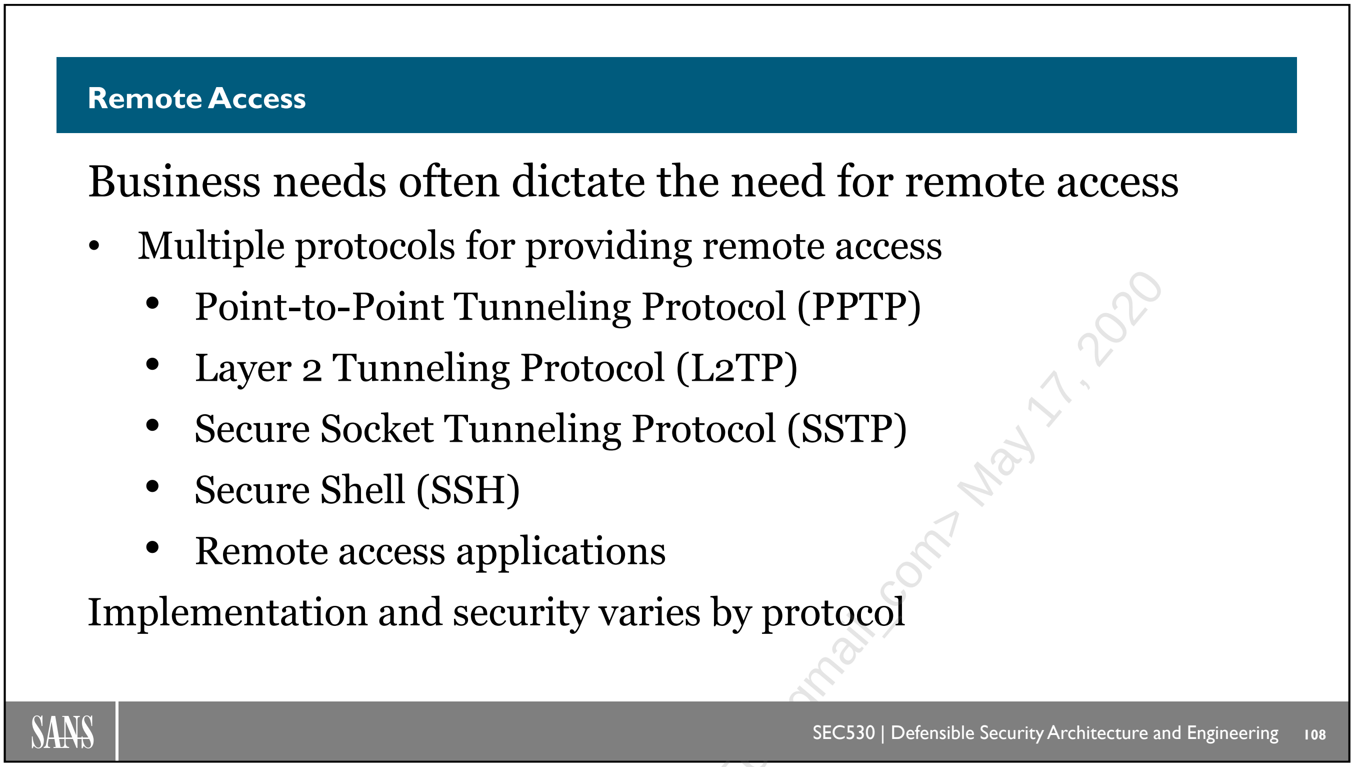
- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. **Securing Remote Access**
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

Course Roadmap

The next section covers Securing Remote Access.



Remote Access

Business needs often dictate the need for remote access

- Multiple protocols for providing remote access
  - Point-to-Point Tunneling Protocol (PPTP)
  - Layer 2 Tunneling Protocol (L2TP)
  - Secure Socket Tunneling Protocol (SSTP)
  - Secure Shell (SSH)
  - Remote access applications

Implementation and security varies by protocol

SANS

SEC530 | Defensible Security Architecture and Engineering 108

Remote Access

In today's modern world, clients and mobile devices need remote access into an organization. How this is achieved varies dramatically in implementation and, as a result, security. This module will be talking about some of the more common forms of client-based VPNs and remote application methods.

Point-To-Point Tunneling Protocol (PPTP)

PPTP was developed in the 90s for dial-up connections

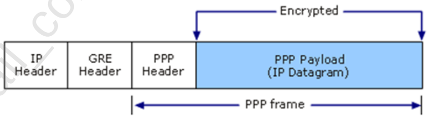
- Windows Server 2016 supports PPTP VPN services
- Tunnels IP over Point-to-Point (PPP) frames

Uses MSCHAP, MSCHAPv2, or EAP-TLS for authentication

- Both MSCHAP versions can be brute forced in hours
- EAP requires full PKI deployment

Encryption is with Microsoft Point-to-Point Encryption (MPPE)

- Uses 40 or 128-bit RC4
- RC4 has no authentication



Point-To-Point Tunneling Protocol (PPTP)

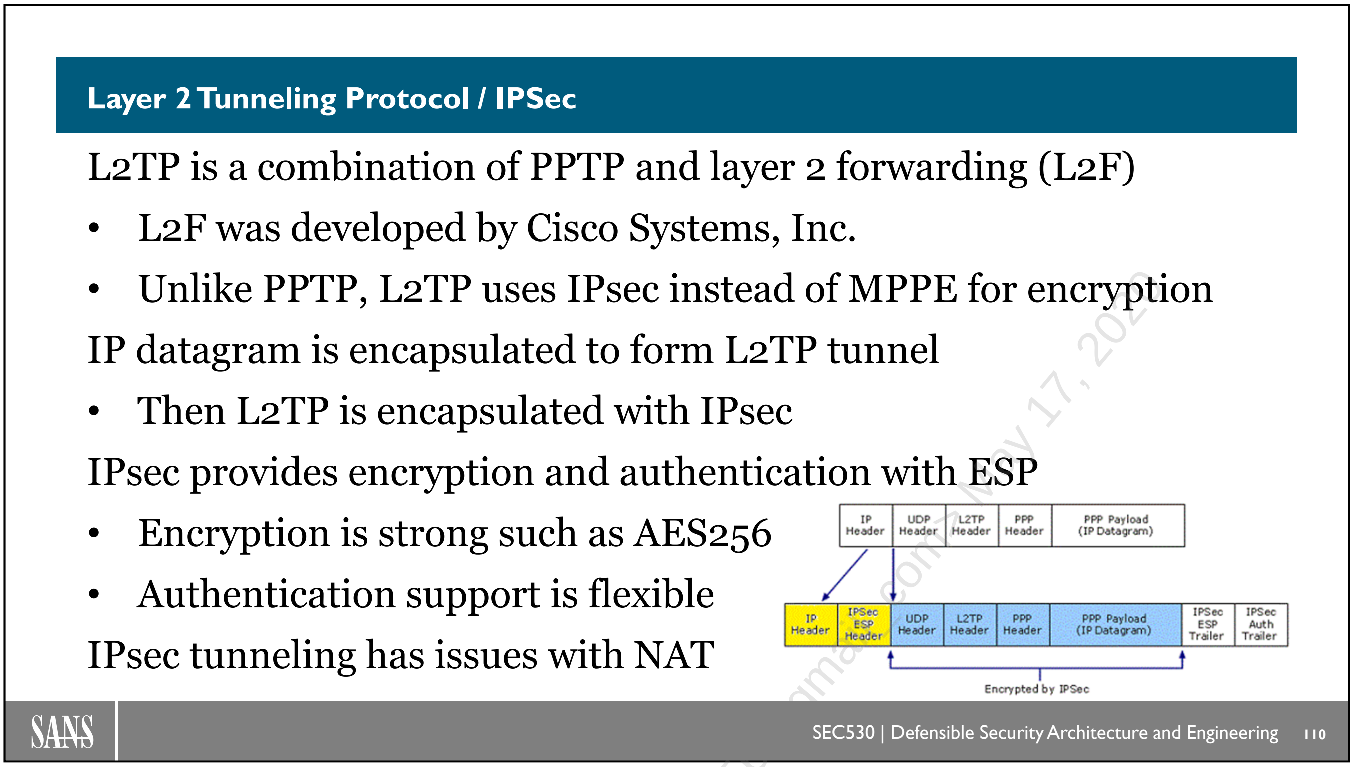
One of the earliest VPN capabilities is still supported today, and that is PPTP. It tunnels IP datagrams over point-to-point frames. PPTP is old, yet it is still supported as a VPN server even on the latest Microsoft operating systems like Server 2016. However, PPTP makes a great example of just because it is easy to set up and use does not mean you should.

PPTP uses primarily MSCHAP or MSCHAP version 2 for authentication. Both are extremely weak by modern standards and can be brute forced in a matter of hours even on a standard workstation. For any level of secure authentication, EAP needs to be enforced. Yet EAP requires full public key infrastructure (PKI) integration. Even with strong authentication, PPTP uses either a 40-bit or 128-bit RC4 encryption cipher through Microsoft's Point-to-Point Encryption (MPPE) algorithm. This algorithm has no authentication or verification that data is coming from the expected source. It is vulnerable to a technique called bit-flipping that can allow an attacker to inject data arbitrarily.

Microsoft recommends using L2TP or another modern VPN client. The point is PPTP, not a secure VPN standard due to the weakness of how it handles authentication and encryption.

[1] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc958045(v=technet.10)

[2] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc771298(v=ws.10)



Layer 2 Tunneling Protocol / IPSec

L2TP is a combination of PPTP and layer 2 forwarding (L2F)

- L2F was developed by Cisco Systems, Inc.
- Unlike PPTP, L2TP uses IPsec instead of MPPE for encryption

IP datagram is encapsulated to form L2TP tunnel

- Then L2TP is encapsulated with IPsec

IPsec provides encryption and authentication with ESP

- Encryption is strong such as AES256
- Authentication support is flexible

IPsec tunneling has issues with NAT

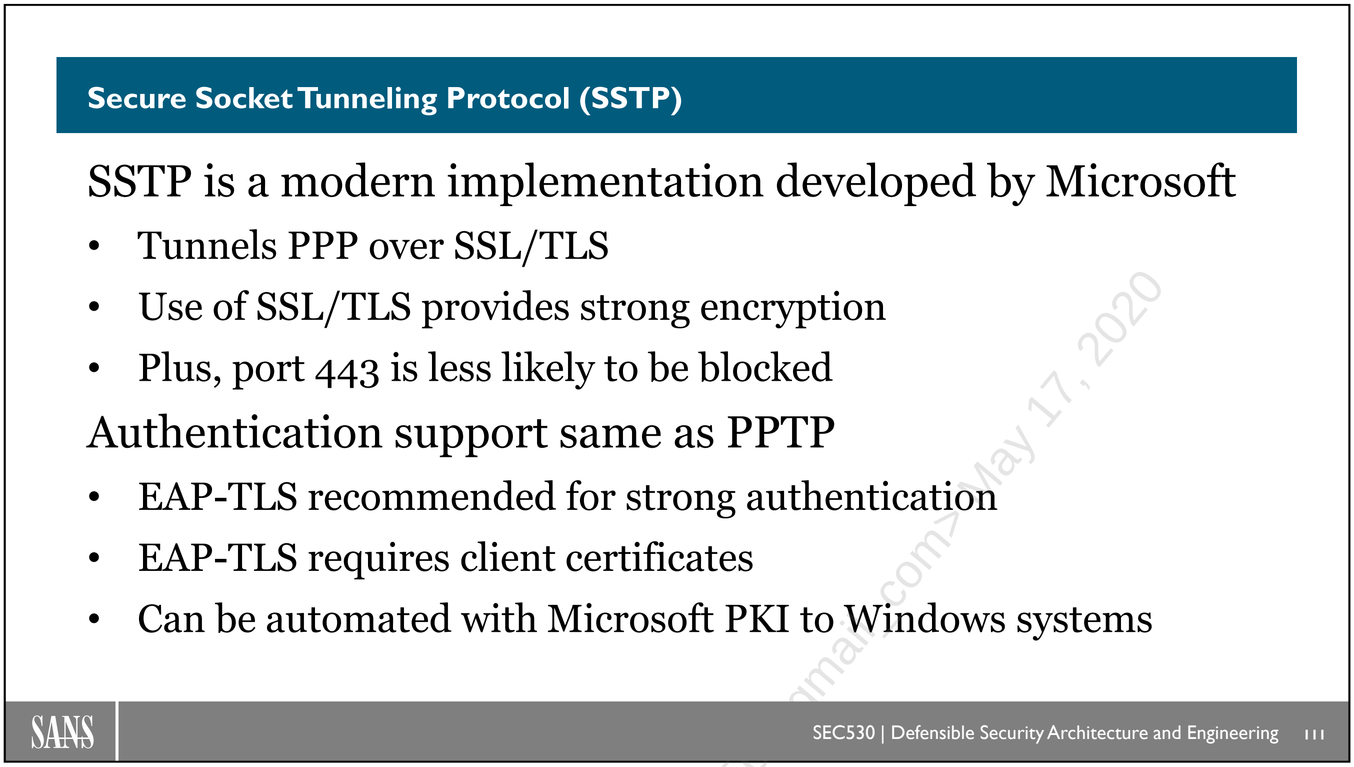
SANS | SEC530 | Defensible Security Architecture and Engineering 110

Layer 2 Tunneling Protocol / IPSec

L2TP is a combination of Microsoft's PPTP protocol and Cisco Systems, Inc. implementation of layer 2 forwarding (L2F). L2TP operates by encapsulating IP datagrams into an L2TP header. Technically, L2TP does not require the use of IPSec but to be used securely it should then encapsulate the L2TP header and IP datagram using IPSec encapsulating security payload (ESP). ESP provides strong authentication and encryption and has the added benefit that the extra encapsulation hides the contents of the original IP datagram including the IP addresses and ports in use.

L2TP is not the fastest VPN solution due to the double encapsulation, but it is common and provides strong protection. Also, the use of IPSec with ESP has issues with certain network address translation implementations (NAT). The NAT issues are because the IP datagram is not visible, and NAT modifies the IP addresses in the IPSec's IP header.

[1] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc977622(v=technet.10)



Secure Socket Tunneling Protocol (SSTP)

SSTP is a modern implementation developed by Microsoft

- Tunnels PPP over SSL/TLS
- Use of SSL/TLS provides strong encryption
- Plus, port 443 is less likely to be blocked

Authentication support same as PPTP

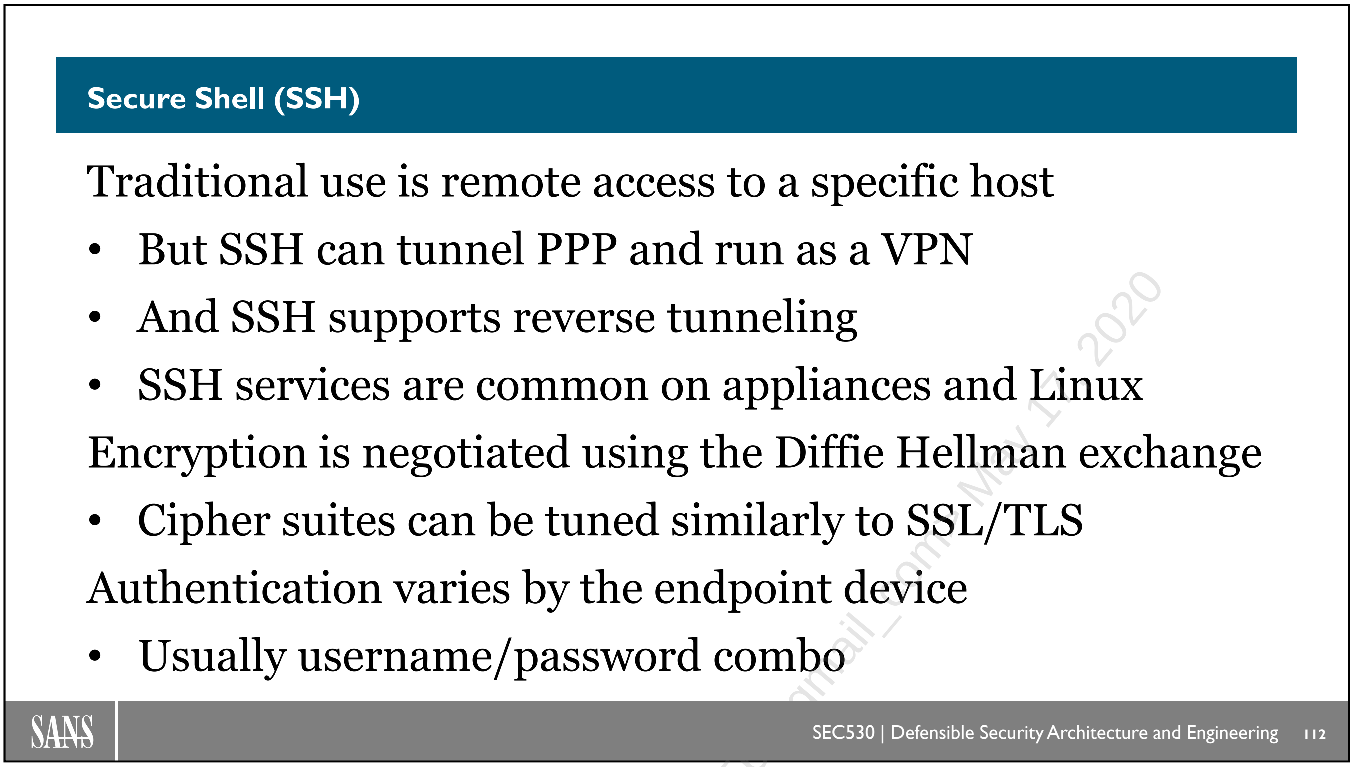
- EAP-TLS recommended for strong authentication
- EAP-TLS requires client certificates
- Can be automated with Microsoft PKI to Windows systems

Secure Socket Tunneling Protocol (SSTP)

Microsoft developed SSTP to be a secure replacement for PPTP. SSTP solves two primary issues. First, the authentication and encryption of PPTP are unacceptable. Second, VPN connectivity often uses ports that are blocked. To resolve both issues, the SSTP protocol uses port 443 with SSL/TLS, similar to a web server. With SSTP, PPP is still used to tunnel IP datagrams, but the tunneling is over SSL/TLS.

Technically, SSTP still supports MSCHAP and MSCHAPv2 for authentication. However, since a certificate is already used by the SSTP server, it is much easier to implement EAP-TLS for strong authentication.

[1] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc731352(v=ws.10)



Secure Shell (SSH)

Traditional use is remote access to a specific host

- But SSH can tunnel PPP and run as a VPN
- And SSH supports reverse tunneling
- SSH services are common on appliances and Linux

Encryption is negotiated using the Diffie Hellman exchange

- Cipher suites can be tuned similarly to SSL/TLS

Authentication varies by the endpoint device

- Usually username/password combo

SANS

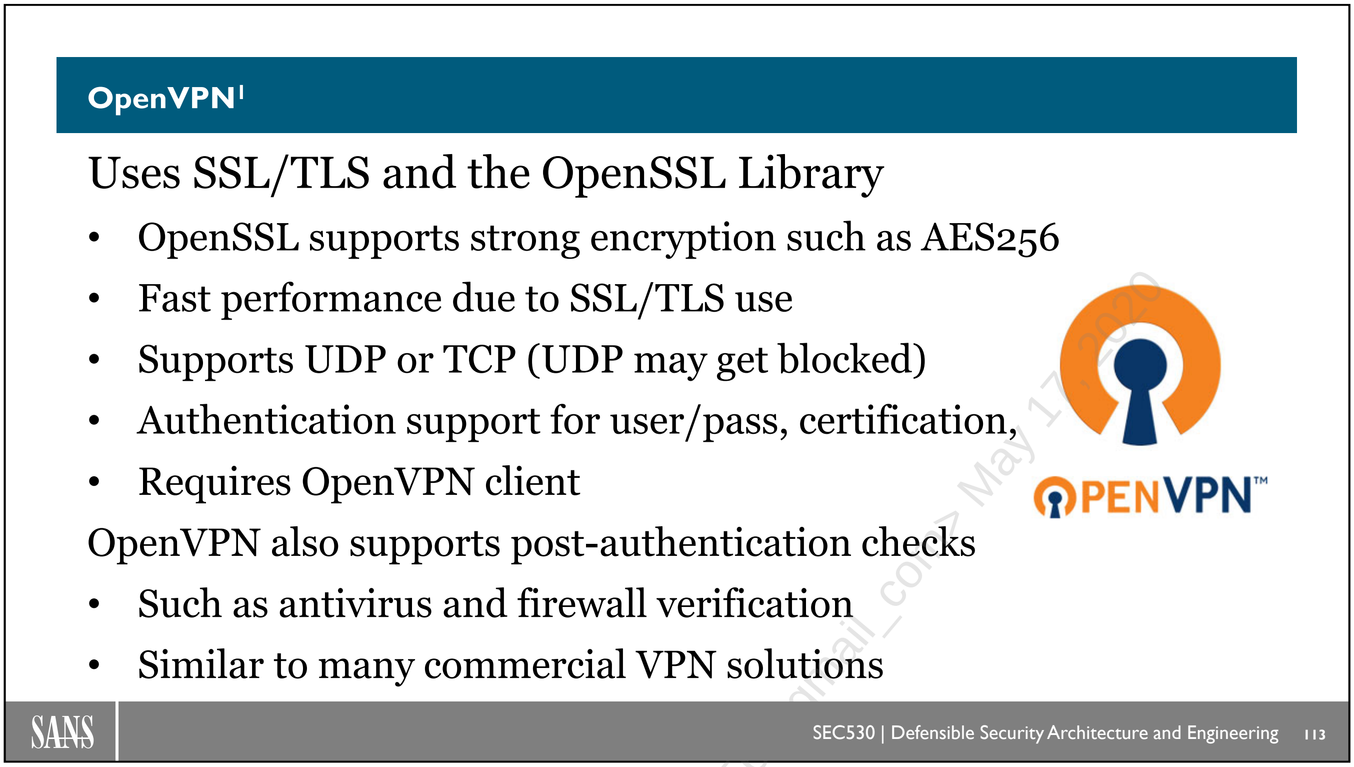
SEC530 | Defensible Security Architecture and Engineering 112

Secure Shell (SSH)

Secure shell is commonly associated with remote access to a specific host. However, SSH can be used as a VPN client by tunneling IP datagrams over PPP using SSH. SSH is not common for VPN access and may not be the most appropriate protocol for a VPN, but it is possible. If anything, the fact that SSH can be used as a VPN client speaks to the danger of a misconfigured SSH server.

Remember back to the attack of the RV slide; SSH was used to allow unlimited access from multiple individuals into the servers.

[1] https://wiki.archlinux.org/index.php/VPN_over_SSH



OpenVPN[1]

Uses SSL/TLS and the OpenSSL Library

- OpenSSL supports strong encryption such as AES256
- Fast performance due to SSL/TLS use
- Supports UDP or TCP (UDP may get blocked)
- Authentication support for user/pass, certification,
- Requires OpenVPN client

OpenVPN also supports post-authentication checks

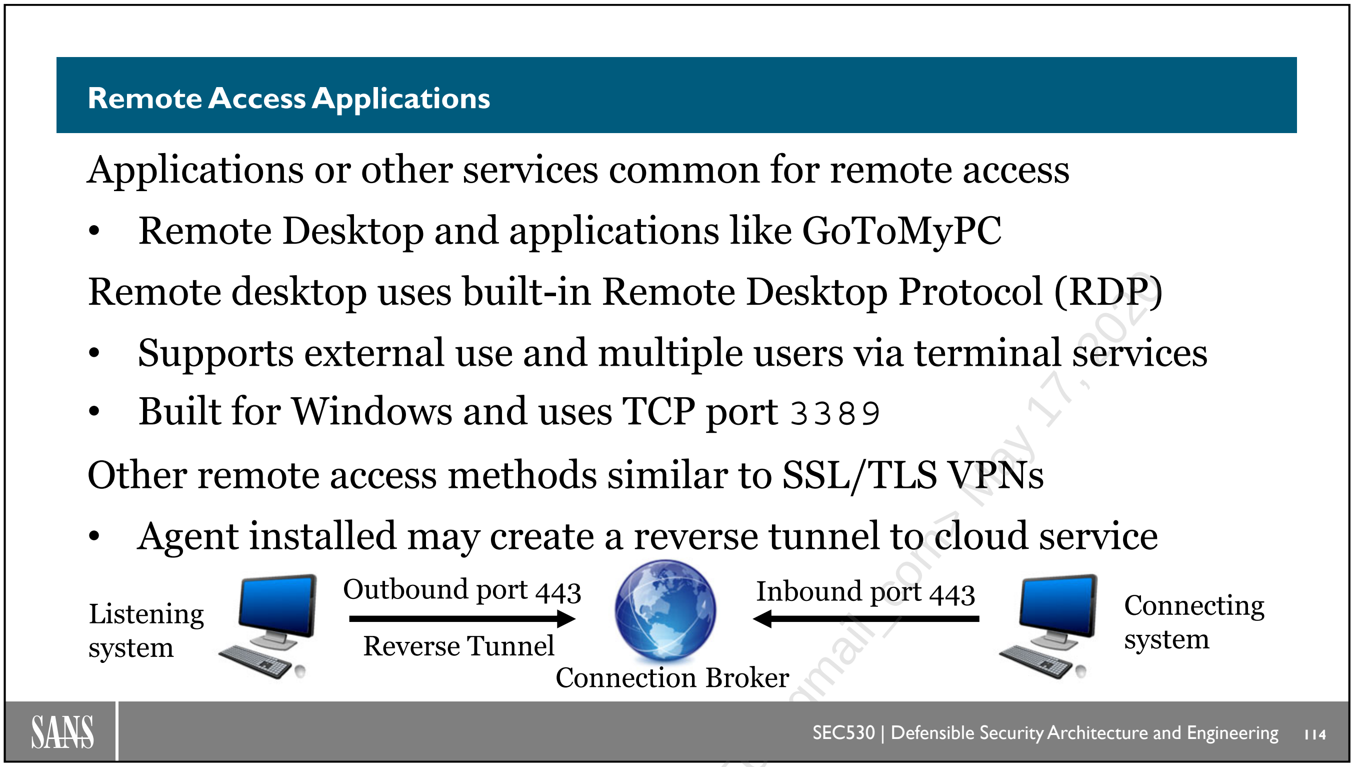
- Such as antivirus and firewall verification
- Similar to many commercial VPN solutions

OpenVPN[1]

OpenVPN is a trending VPN solution for both commercial and open source implementations. OpenVPN tunnels traffic over SSL/TLS but also uses capabilities in the OpenSSL library to maximum resiliency and performance. A standard OpenVPN configuration will attempt to use UDP first and then if UDP is blocked will fail back to TCP port 443. By utilizing UDP, OpenVPN optimizes the user experience and performance.

The main downside to OpenVPN is that it requires the installation of an OpenVPN client. The solution supports strong authentication, strong encryption, and advanced capabilities such as post-authentication checks.

[1] https://openvpn.net/



Remote Access Applications

Applications or other services common for remote access

- Remote Desktop and applications like GoToMyPC

Remote desktop uses built-in Remote Desktop Protocol (RDP)

- Supports external use and multiple users via terminal services
- Built for Windows and uses TCP port 3389

Other remote access methods similar to SSL/TLS VPNs

- Agent installed may create a reverse tunnel to cloud service

Listening system

Outbound port 443

Reverse Tunnel

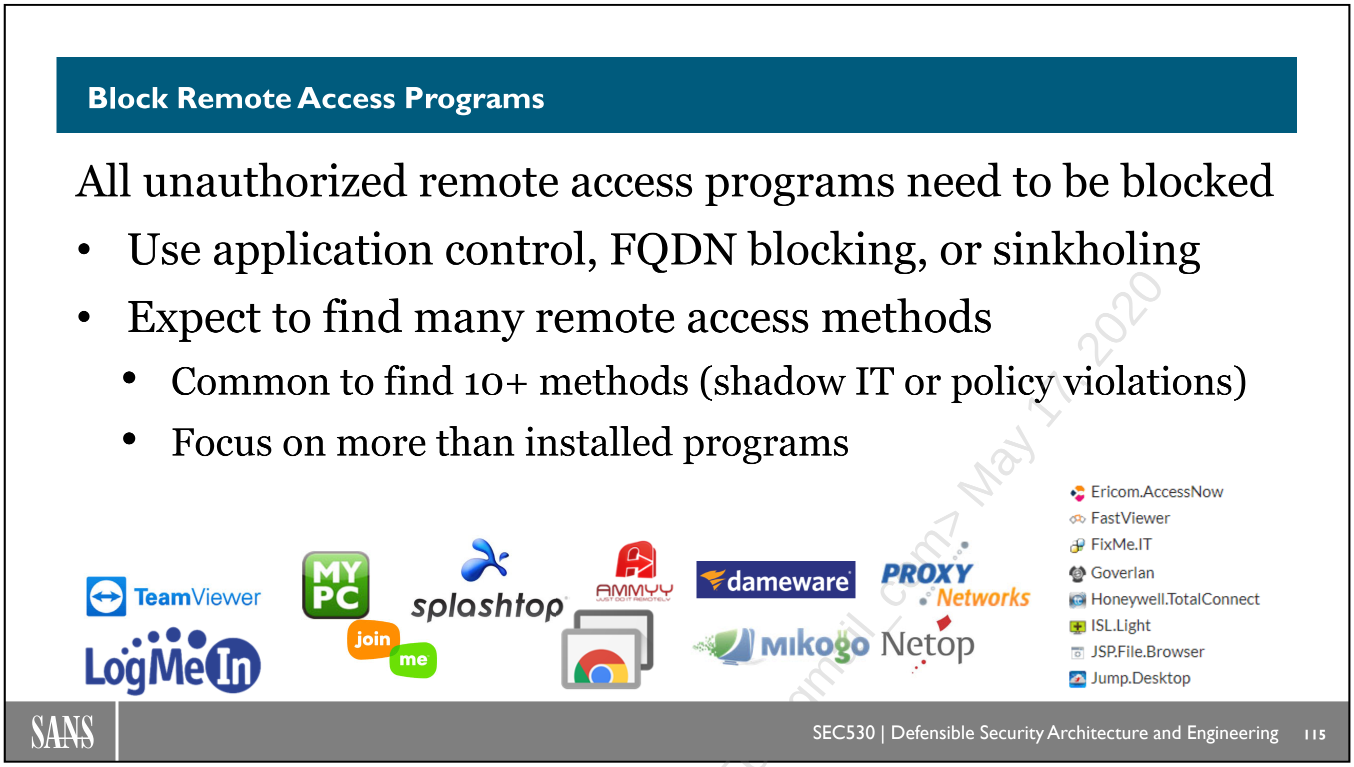
Connection Broker

Inbound port 443

Connecting system

Remote Access Applications

The client-based VPN solution is not the only means of remote access. Remote desktop protocol/terminal services is a standard Windows implementation that can be used for both internal and external access. Even more common is the use of third-party programs providing remote access using reverse tunneling. These reverse tunneling applications bypass internal protection mechanisms and can undermine security.



Block Remote Access Programs

All unauthorized remote access programs need to be blocked

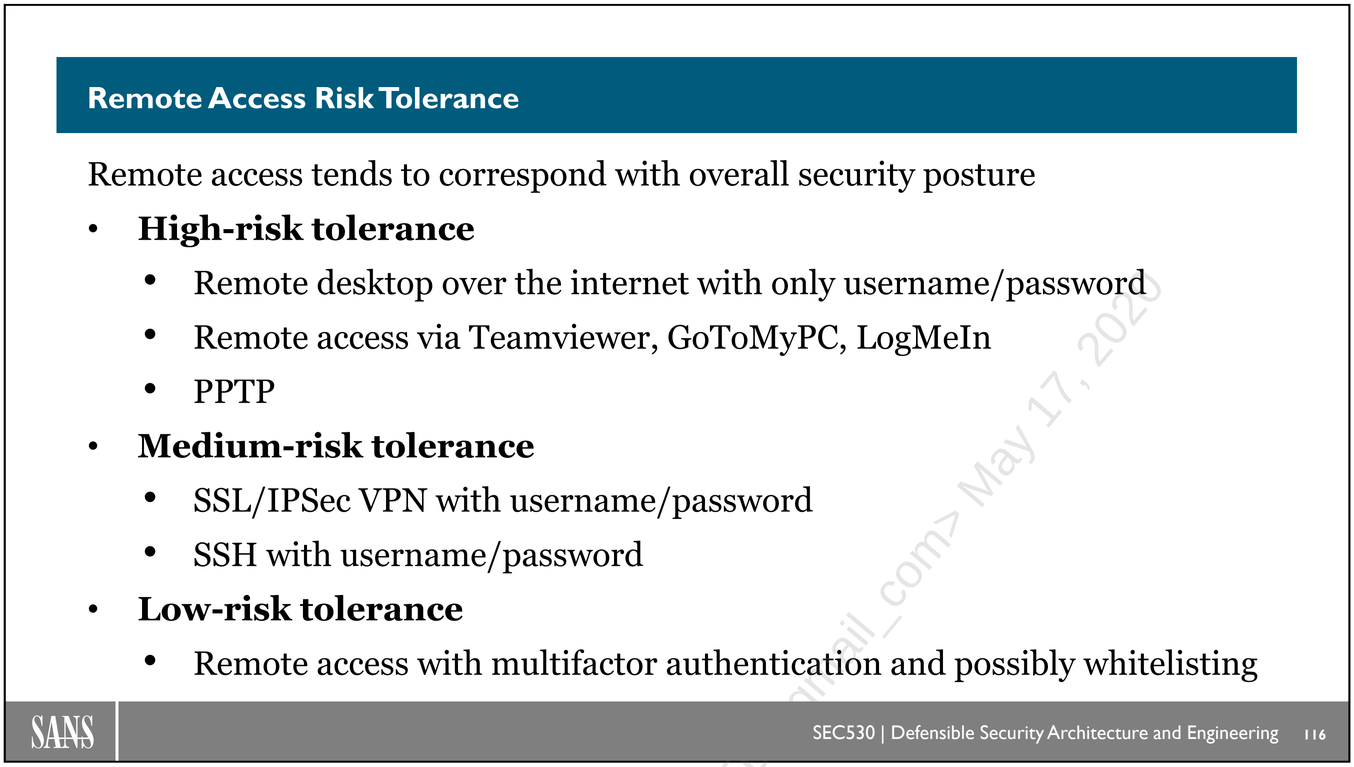
- Use application control, FQDN blocking, or sinkholing
- Expect to find many remote access methods
  - Common to find 10+ methods (shadow IT or policy violations)
  - Focus on more than installed programs

SANS SEC530 | Defensible Security Architecture and Engineering 115

Block Remote Access Programs

Organizations of any size need to choose the remote access methods that are authorized and enforce them. Enforcement includes the authorized applications and explicitly blocking all others. Failure to block other remote access solutions almost guarantees that employees will use them. It is extremely common even on a small to the midsize organization to have over ten methods of external remote access in play.

These applications do not have to be installed to function. Some are simply executables that can be run in a portable fashion. The best method to block these is with application control on a firewall. These applications tend to favor SSL/TLS, so SSL Inspection provides the best method of filtering them out. DNS filtering and SSL certificate inspection also can have large success.



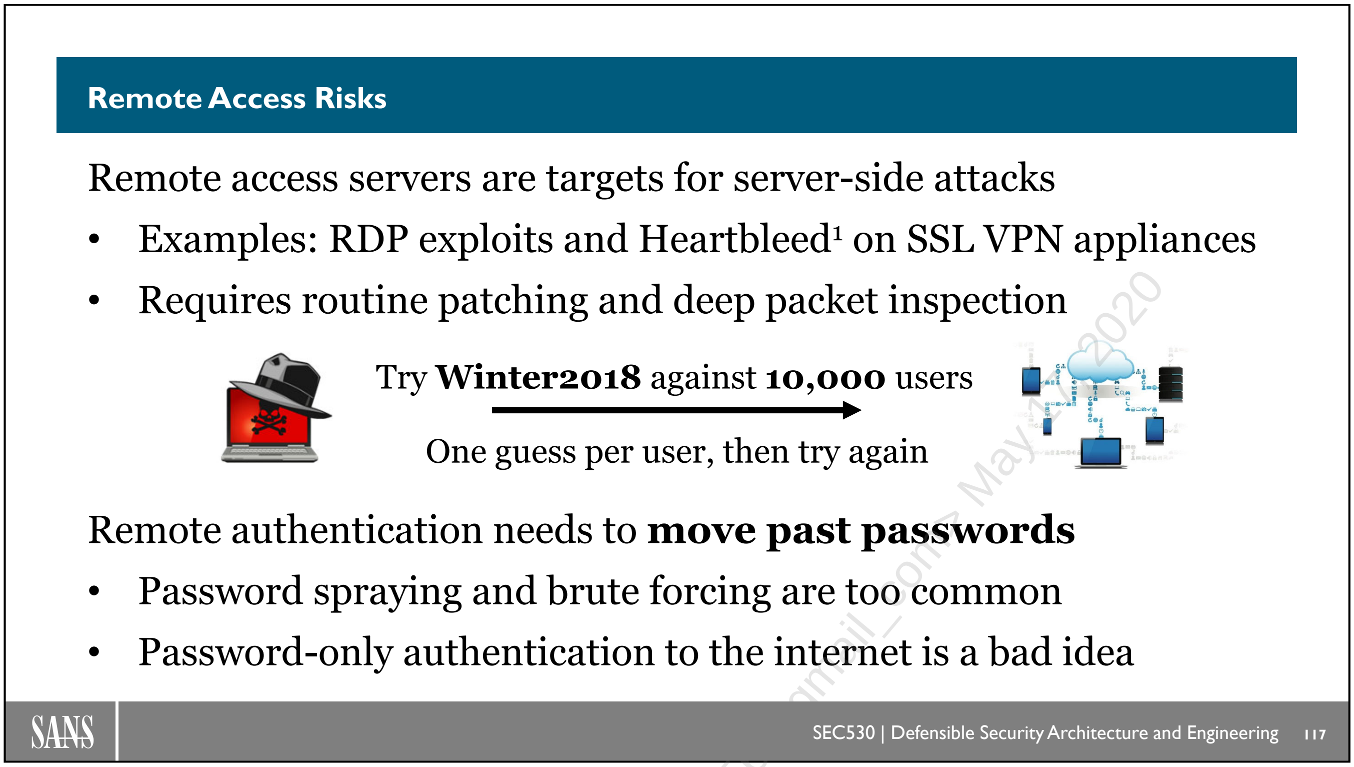
Remote Access Risk Tolerance

Remote access tends to correspond with overall security posture

- **High-risk tolerance**
  - Remote desktop over the internet with only username/password
  - Remote access via Teamviewer, GoToMyPC, LogMeIn
  - PPTP
- **Medium-risk tolerance**
  - SSL/IPSec VPN with username/password
  - SSH with username/password
- **Low-risk tolerance**
  - Remote access with multifactor authentication and possibly whitelisting

Remote Access Risk Tolerance

How an organization allows remote access more often than not correlates to its overall security posture. Weak controls and security implementations manifest as deeper issues internally. High security with multifactor authentication and restrictions on what, where, and when tend to spill over to internal implementations.



Remote Access Risks

Remote access servers are targets for server-side attacks

- Examples: RDP exploits and Heartbleed[1] on SSL VPN appliances
- Requires routine patching and deep packet inspection

Try **Winter2018** against **10,000** users

One guess per user, then try again

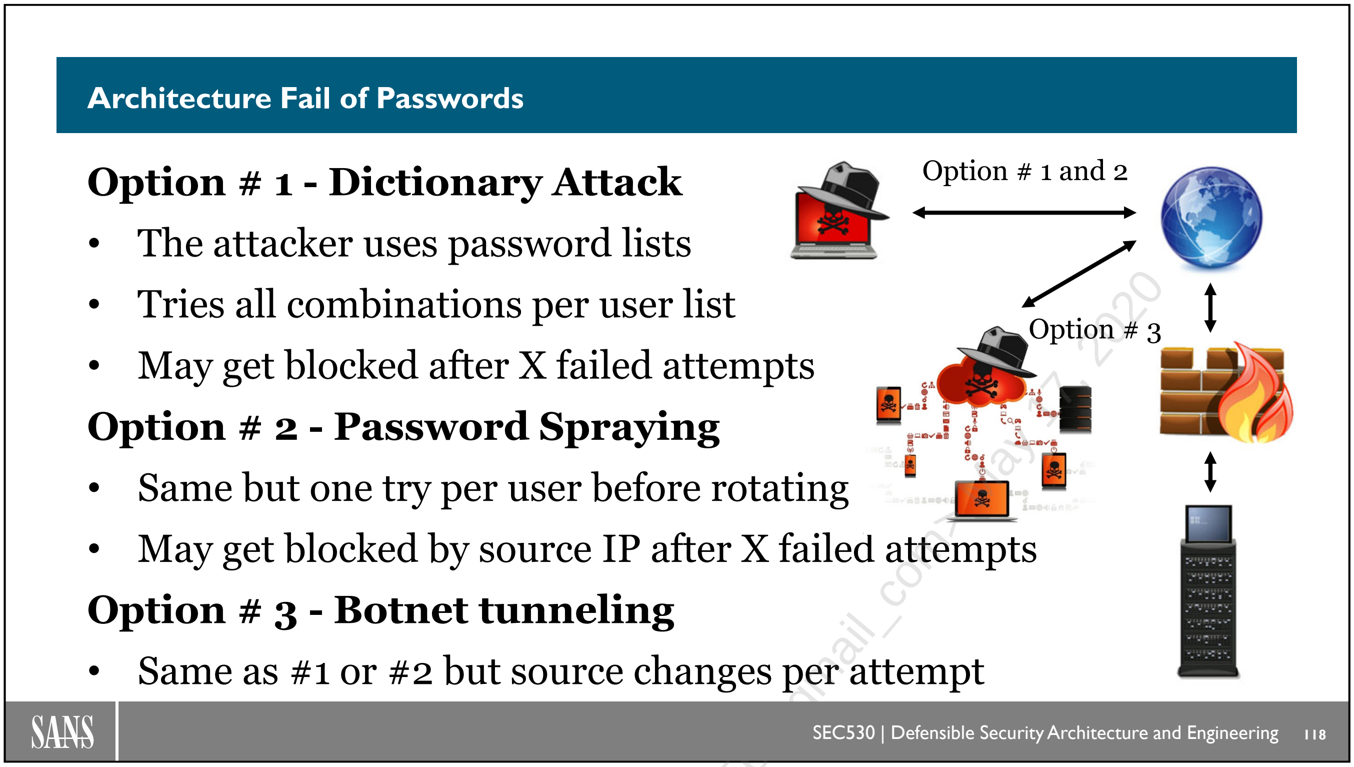
Remote authentication needs to **move past passwords**

- Password spraying and brute forcing are too common
- Password-only authentication to the internet is a bad idea

Remote Access Risks

How authentication is handled presents one of the number one server-side attack vectors today. Exploits still crop up from time to time such as Heartbleed, but username and password guessing still is a viable and successful technique for hackers and penetration testers. Password attempt restrictions are beneficial but multifactor should be a standard for external access.

[1] http://heartbleed.com/



Architecture Fail of Passwords

Option # 1 - Dictionary Attack

- The attacker uses password lists
- Tries all combinations per user list
- May get blocked after X failed attempts

Option # 2 - Password Spraying

- Same but one try per user before rotating
- May get blocked by source IP after X failed attempts

Option # 3 - Botnet tunneling

- Same as #1 or #2 but source changes per attempt

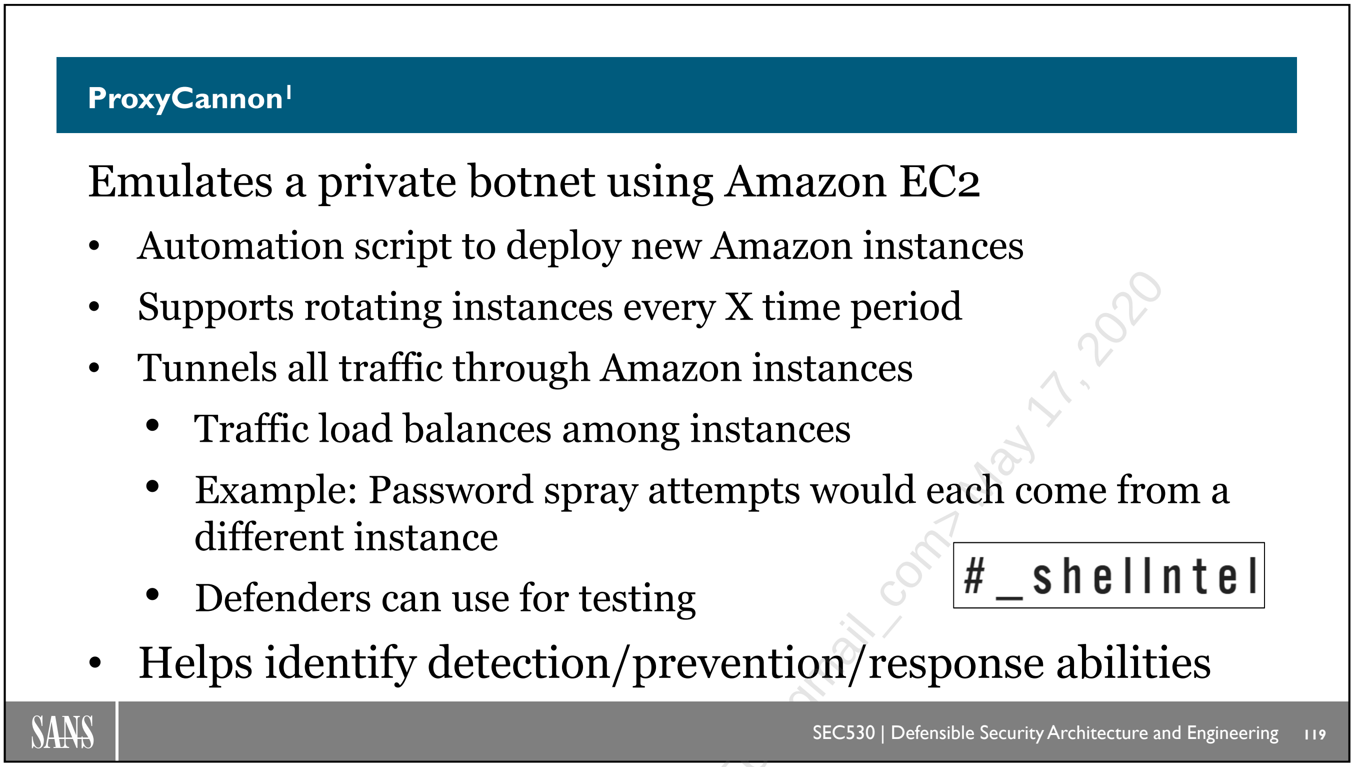
Architecture Fail of Passwords

The two most common forms of password guessing are dictionary attacks and password spraying. A dictionary attack uses a password list and attempts to log in to accounts by trying each possible password in the list. Password guessing tools may also try common permutations of the passwords in the list. The main problem for attackers is that systems often will temporarily or permanently lock out a system after too many failed login attempts.

A more successful technique involves using a password list but guessing each password for every user before trying another password. Many systems would not block this attack because the source IP attempting logons is not repeatedly failing to login to a specific account. However, password spraying should be easy to find as it is an enormously high volume of failed logins from a specific IP.

A third option involves password guessing but by use of a botnet. By using a botnet, password guesses come from multiple source IP addresses and possibly from different geolocations. A botnet password attack makes attribution and prevention significantly more difficult.

[1] https://www.shellntel.com/blog/2015/9/9/update-creating-your-own-private-botnet-for-scanning



ProxyCannon[1]

Emulates a private botnet using Amazon EC2

- Automation script to deploy new Amazon instances
- Supports rotating instances every X time period
- Tunnels all traffic through Amazon instances
  - Traffic load balances among instances
  - Example: Password spray attempts would each come from a different instance
  - Defenders can use for testing
- Helps identify detection/prevention/response abilities

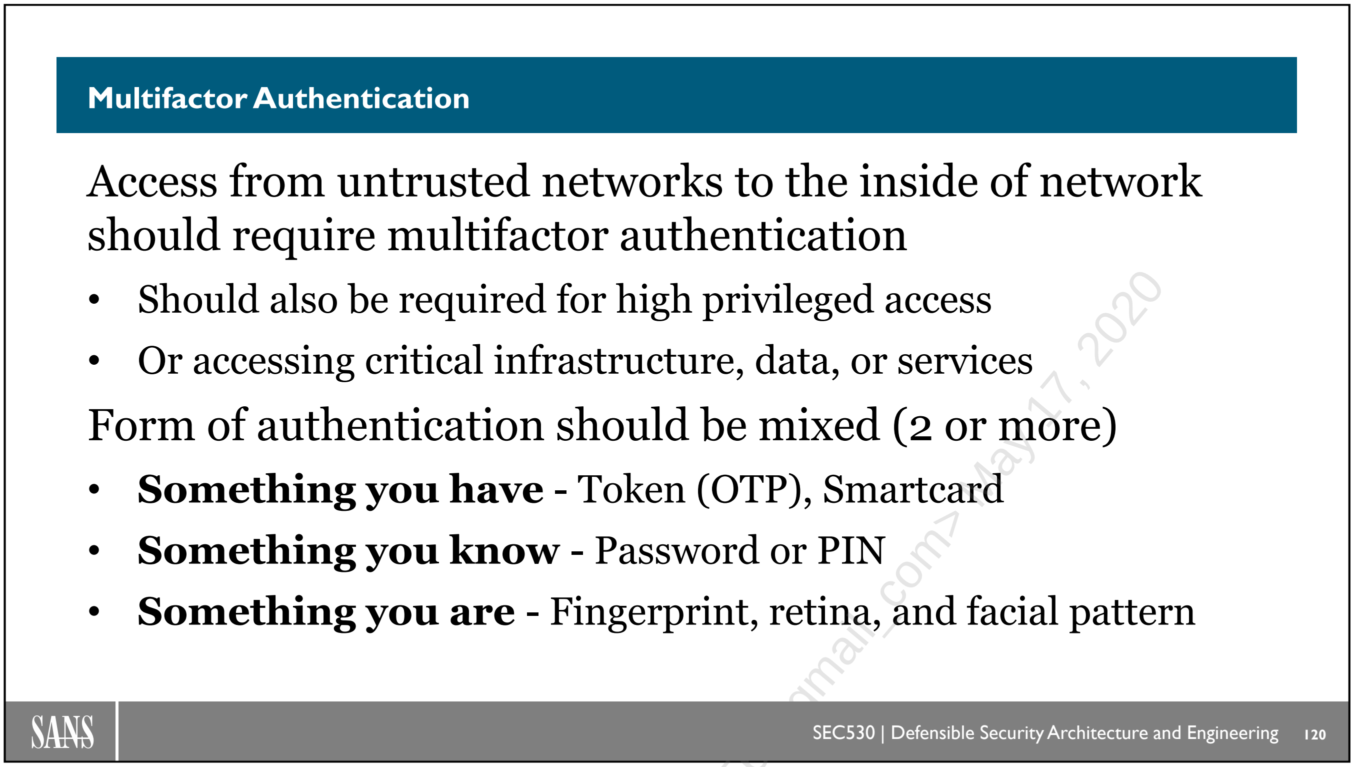
#_shellntel

ProxyCannon[1]

To drive home that username and password authentication is a horrible idea consider an open source tool called ProxyCannon. ProxyCannon interfaces with Amazon EC2 to automate the deployment of Amazon instances and dynamically load balances traffic through these instances for network-based functions. The tool is designed for penetration testers to simulate attacks from a botnet. As blue teamers, we can use this tool to test our abilities to detect network scans or password guessing that rotates through multiple IP addresses and geolocations.

Including setting up a new Amazon EC2 user, ProxyCannon can be set up in less than an hour. Thought exercise - What would you do to detect or block Botnet? Can you even block it?

[1] https://www.shellntel.com/blog/2015/9/9/update-creating-your-own-private-botnet-for-scanning



Multifactor Authentication

Access from untrusted networks to the inside of network should require multifactor authentication

- Should also be required for high privileged access
- Or accessing critical infrastructure, data, or services

Form of authentication should be mixed (2 or more)

- **Something you have** - Token (OTP), Smartcard
- **Something you know** - Password or PIN
- **Something you are** - Fingerprint, retina, and facial pattern

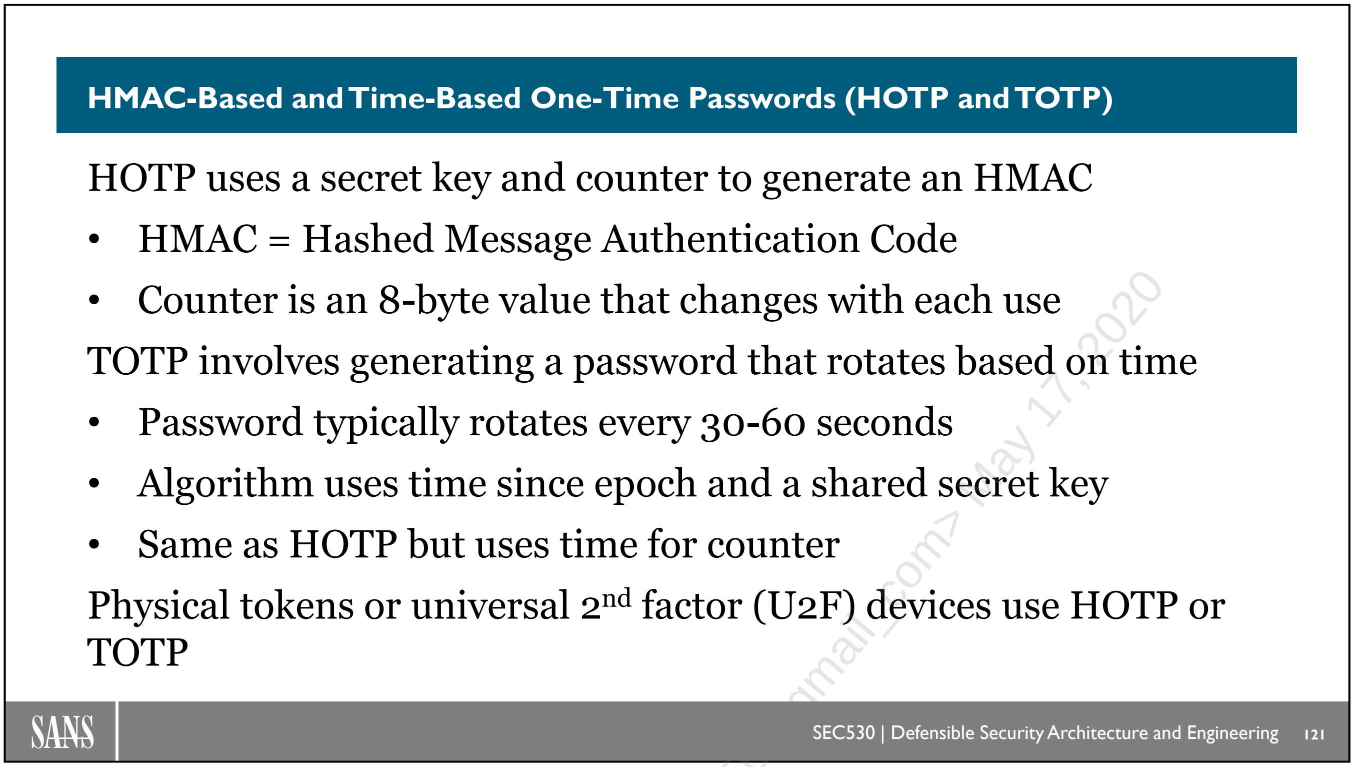
SANS

SEC530 | Defensible Security Architecture and Engineering 120

Multifactor Authentication

Authentication should include more than one form of authentication. The forms of authentication fall into high-level categories of something you know, something you have, and something you are. By using more than one form of authentication, the risk of credential compromise is significantly reduced.

At a minimum, multifactor authentication should be considered for all external authentication entry points. This includes VPN access and organization public interfaces such as Microsoft Outlook Web Access.



HMAC-Based and Time-Based One-Time Passwords (HOTP and TOTP)

HOTP uses a secret key and counter to generate an HMAC

- HMAC = Hashed Message Authentication Code
- Counter is an 8-byte value that changes with each use

TOTP involves generating a password that rotates based on time

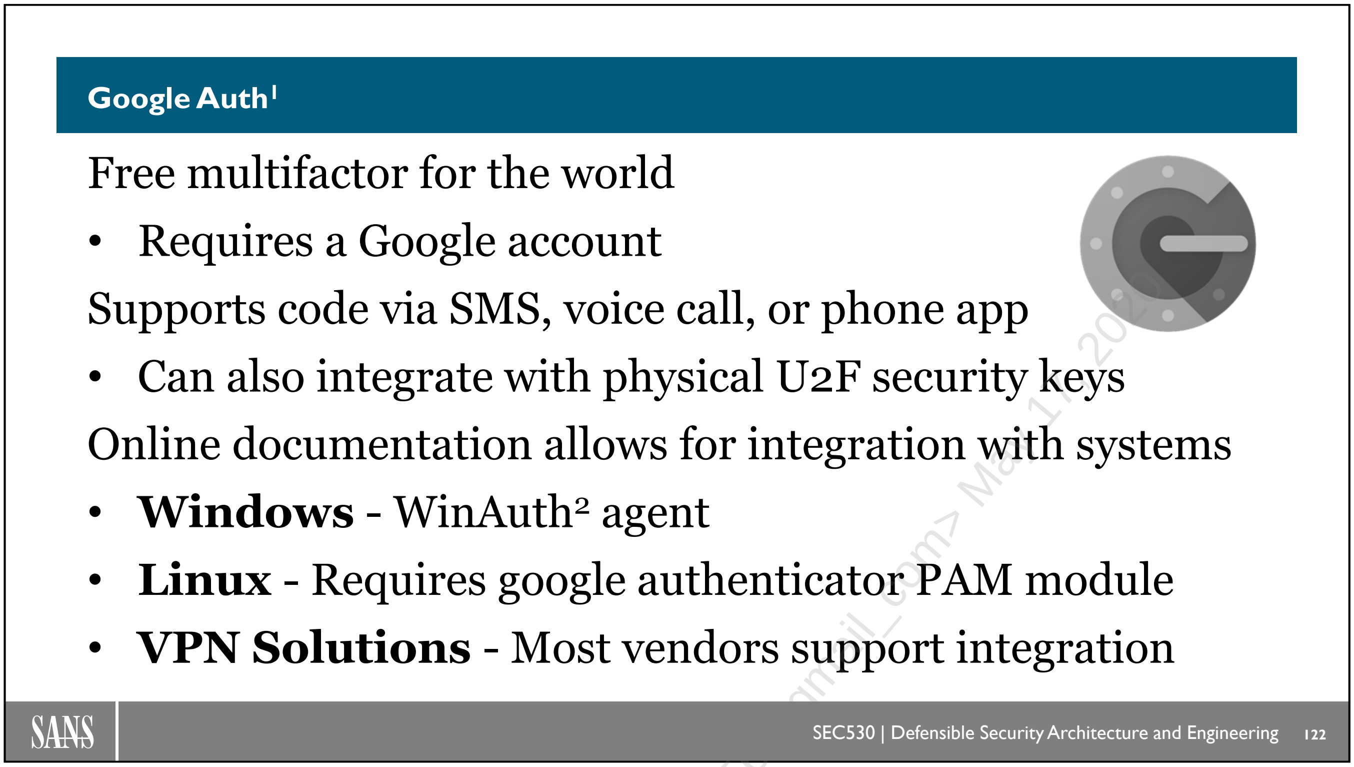
- Password typically rotates every 30-60 seconds
- Algorithm uses time since epoch and a shared secret key
- Same as HOTP but uses time for counter

Physical tokens or universal 2nd factor (U2F) devices use HOTP or TOTP

HMAC-Based and Time-Based One-Time Passwords (HOTP and TOTP)

Many hardware and software token solutions use either HOTP or TOTP. These provide a hashed message authentication code (HMAC) that rotates with the use or by time. These solutions represent some of the easiest methods to enforce multifactor authentication.

[1] https://tools.ietf.org/html/rfc4226



Google Auth[1]

Free multifactor for the world

- Requires a Google account

Supports code via SMS, voice call, or phone app

- Can also integrate with physical U2F security keys

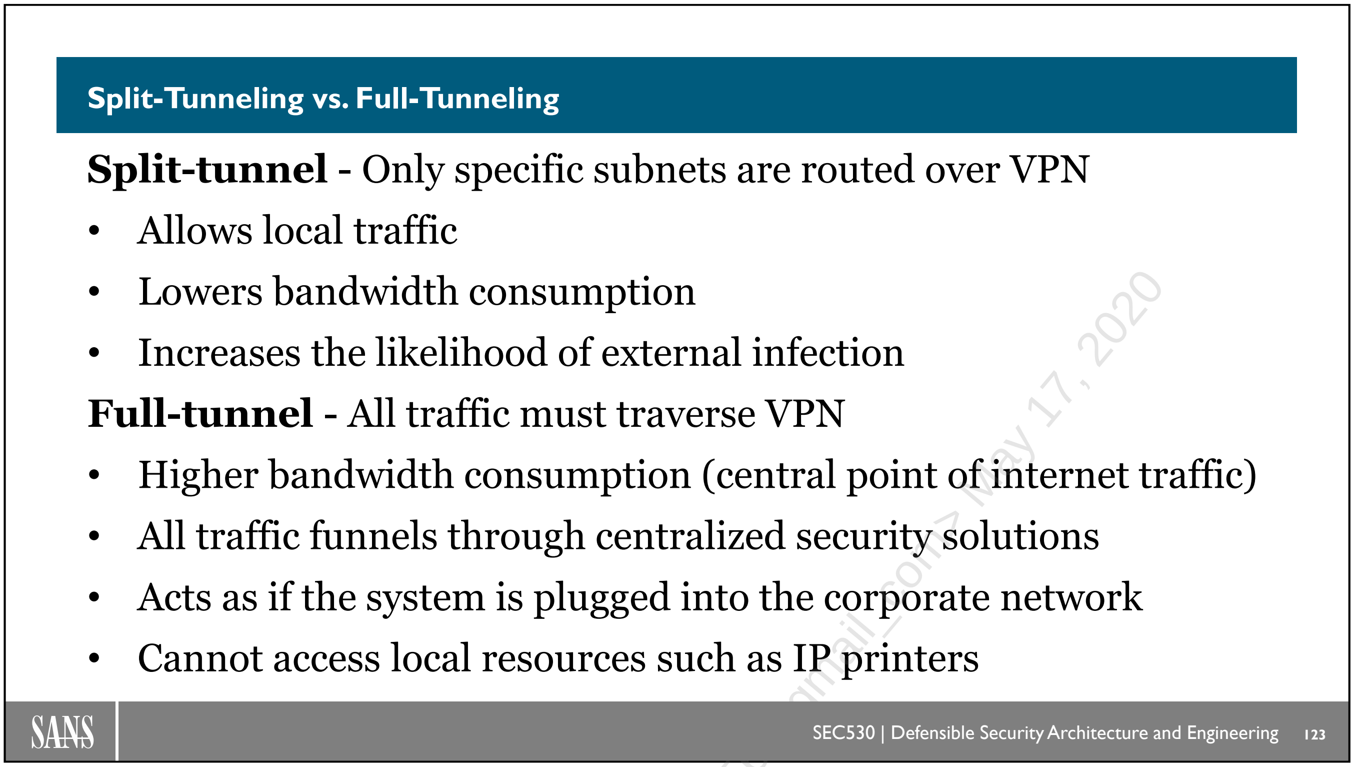
Online documentation allows for integration with systems

- **Windows** - WinAuth[2] agent
- **Linux** - Requires google authenticator PAM module
- **VPN Solutions** - Most vendors support integration

Google Auth[1]

Google auth is worth calling out as it is a **free solution that provides multifactor authentication** via hardware/software tokens, text messaging, voice calling, or phone applications. Google auth is extremely well documented and has support for almost all **commercial VPN solutions**. Google Auth also has documentation for building custom integrations although many integrations have publicly available solutions such as WinAuth for dual factor authentication on Windows platforms.

[1] https://www.google.com/landing/2step/#tab=how-it-works



Split-Tunneling vs. Full-Tunneling

Split-tunnel - Only specific subnets are routed over VPN

- Allows local traffic
- Lowers bandwidth consumption
- Increases the likelihood of external infection

Full-tunnel - All traffic must traverse VPN

- Higher bandwidth consumption (central point of internet traffic)
- All traffic funnels through centralized security solutions
- Acts as if the system is plugged into the corporate network
- Cannot access local resources such as IP printers

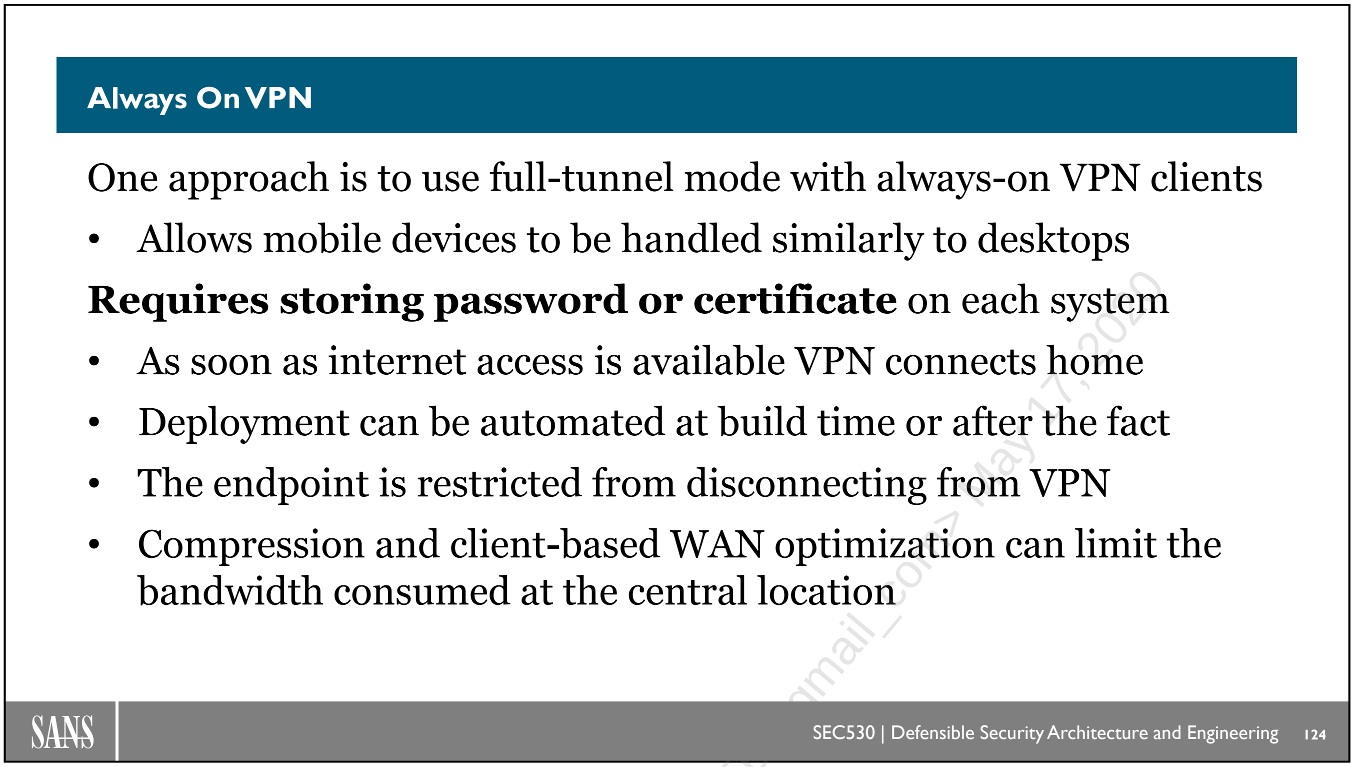
SANS

SEC530 | Defensible Security Architecture and Engineering 123

Split-Tunneling vs. Full Tunneling

When a VPN connects, a client routing gets configured for either split-tunneling or full-tunneling mode. Split-tunneling is more common and involves routing traffic destined to specific subnets over the VPN only. With split-tunneling internet access and local access are allowed directly and do not involve traversing the VPN. Full-tunneling involves forcing all traffic over the VPN. Full-tunneling often will break access to local devices such as network printers. All traffic must traverse the VPN.

Full-tunneling has significant security ramifications as now central monitoring and control is achieved by funneling all traffic through an organization's internet.



Always On VPN

One approach is to use full-tunnel mode with always-on VPN clients

- Allows mobile devices to be handled similarly to desktops

Requires storing password or certificate on each system

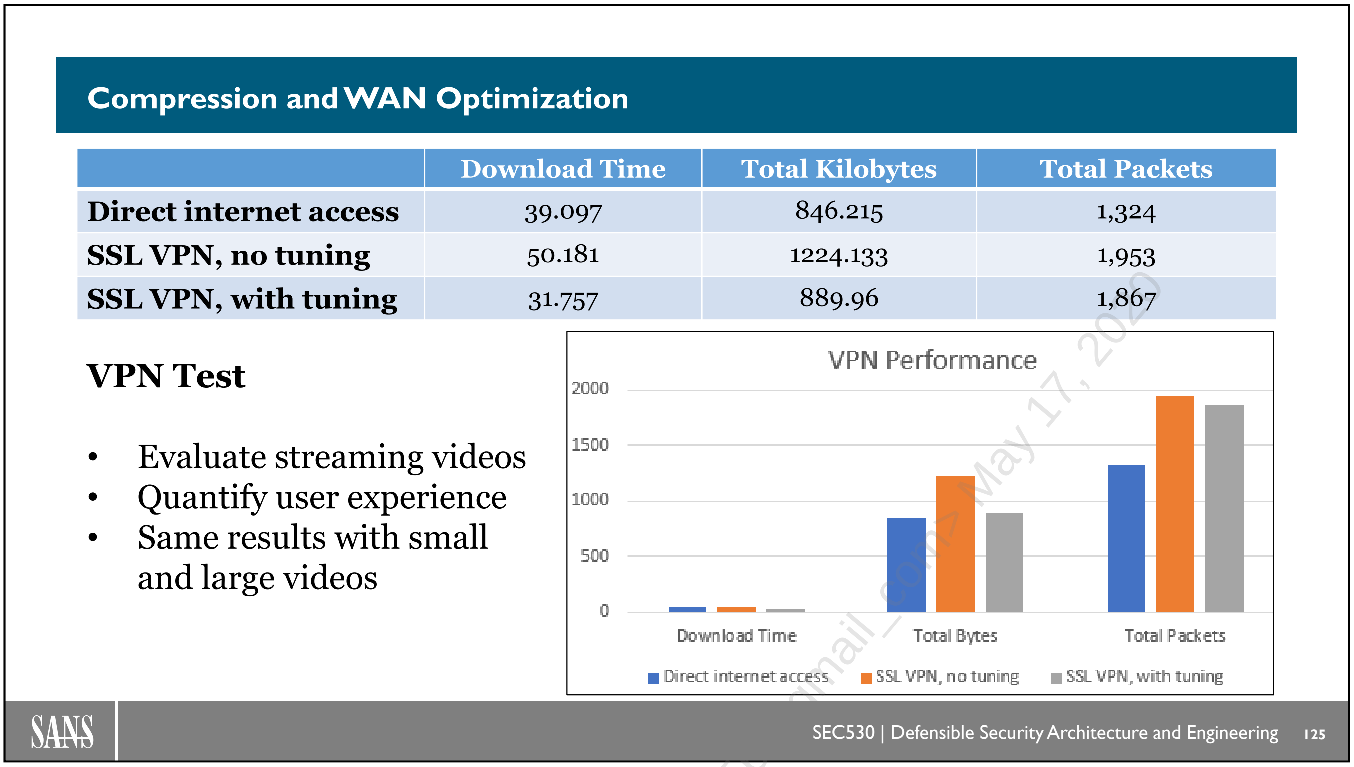
- As soon as internet access is available VPN connects home
- Deployment can be automated at build time or after the fact
- The endpoint is restricted from disconnecting from VPN
- Compression and client-based WAN optimization can limit the bandwidth consumed at the central location

SANS SEC530 | Defensible Security Architecture and Engineering 124

Always On VPN

Modern VPN solutions offer always-on ability. Always-on means, the VPN will automatically turn on and stay enabled anytime internet access is available. The use of always on enables laptops and mobile devices to be managed more as if they were traditional desktops. Patching and security controls can then be implemented the same, and central security devices are able to protect all assets.

The downside to always on VPN is that additional bandwidth needs to be purchased and performance tuning and high availability become critical.



Compression and WAN Optimization

	Download Time	Total Kilobytes	Total Packets
Direct internet access	39.097	846.215	1,324
SSL VPN, no tuning	50.181	1224.133	1,953
SSL VPN, with tuning	31.757	889.96	1,867

VPN Test

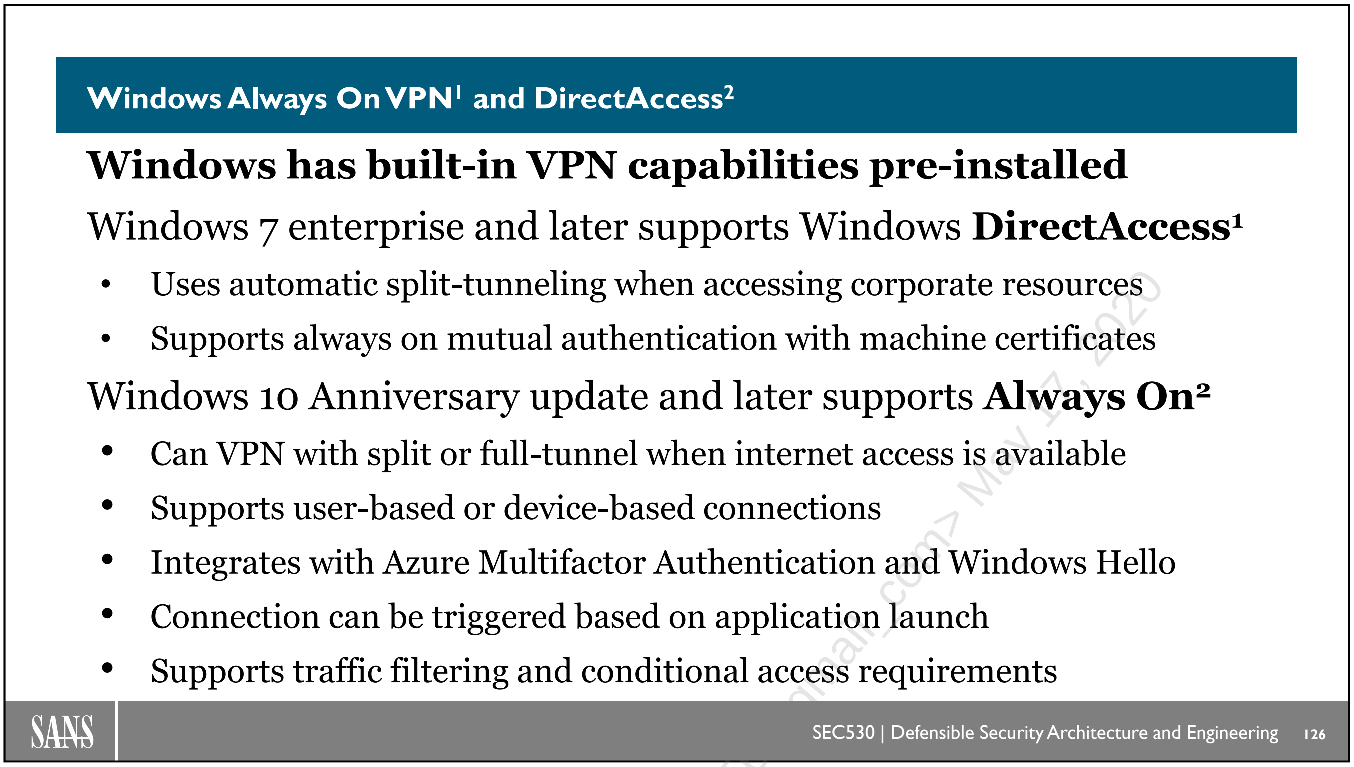
- Evaluate streaming videos
- Quantify user experience
- Same results with small and large videos

VPN Performance
2000
1500
1000
500
0
Download Time
Total Bytes
Total Packets
■ Direct internet access
■ SSL VPN, no tuning
■ SSL VPN, with tuning

Compression and WAN Optimization

This slide demonstrates testing performed under three scenarios. In all of the scenarios, MP4 videos were streamed over the internet. Under the first scenario, videos were accessed directly over the internet without going through a VPN. The second scenario involved streaming videos over a traditional SSL VPN with default configuration. The last scenario involved the same SSL VPN but with tuning compression and WAN optimization techniques. Oddly enough, the user experience was best over the SSL VPN that was tuned as fewer bytes were transferred and the download time was the smallest.

The data represented in the slide represents a single streaming video, but the results were the same for small and massive videos. Obviously, the results will vary by application and contents accessed, but the point is a well-tuned SSL VPN can be capable of improving or equally the performance of direct internet access even when tunneling traffic.



Windows Always On VPN[1] and DirectAccess[2]

Windows has built-in VPN capabilities pre-installed

Windows 7 enterprise and later supports Windows **DirectAccess[1]**

- Uses automatic split-tunneling when accessing corporate resources
- Supports always on mutual authentication with machine certificates

Windows 10 Anniversary update and later supports **Always On[2]**

- Can VPN with split or full-tunnel when internet access is available
- Supports user-based or device-based connections
- Integrates with Azure Multifactor Authentication and Windows Hello
- Connection can be triggered based on application launch
- Supports traffic filtering and conditional access requirements

SANS

SEC530 | Defensible Security Architecture and Engineering 126

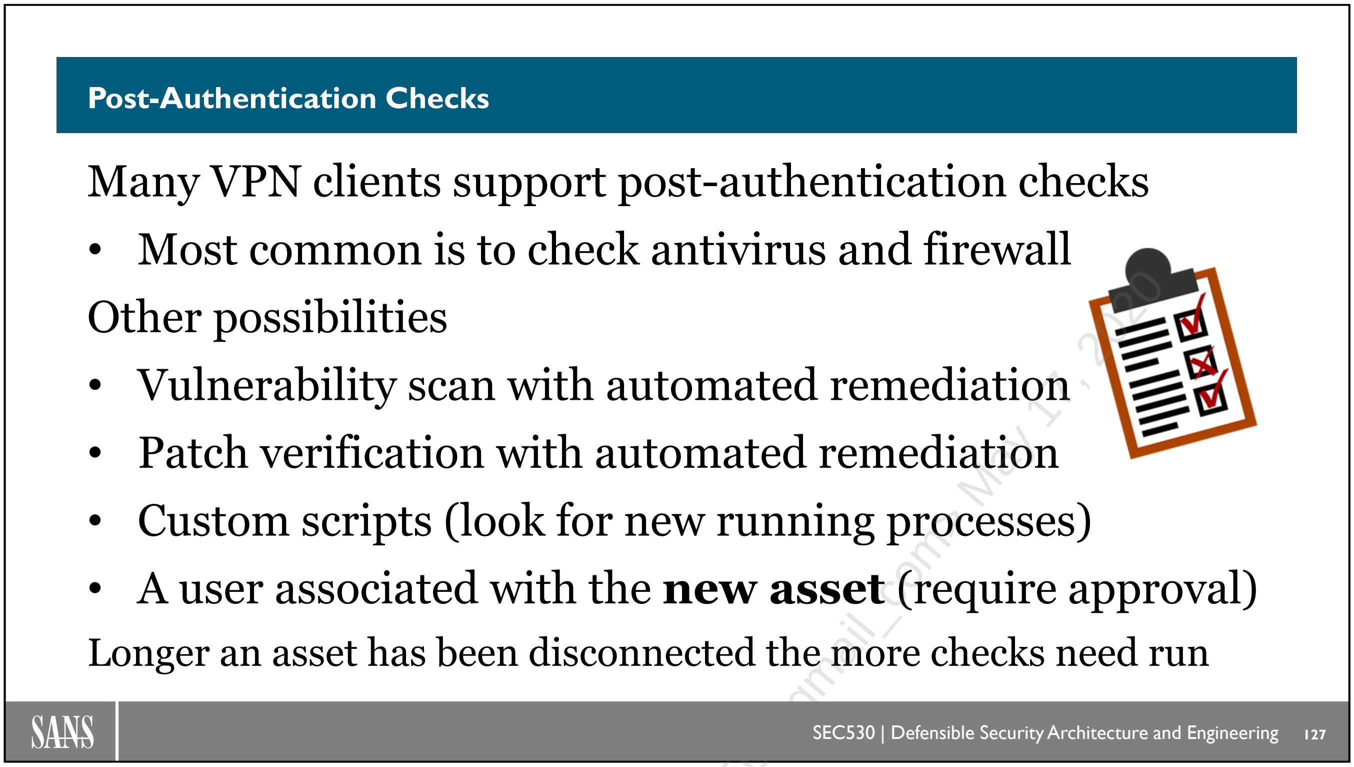
Windows Always On VPN[1] and DirectAccess[2]

Windows 7 enterprise and later operating systems support DirectAccess. DirectAccess provides split-tunnel dynamic access to corporate resources. DirectAccess uses DNS and SSL/TLS streaming to identify if access is required via tunneling and if so dynamically connects as needed. However, DirectAccess requires enterprise versions of Windows to work.

A more recent addition to Windows 10 Anniversary update or later operating systems are Microsoft Always On. Always On supports split-tunneling or full-tunneling automatically using user-based or device-based certificates. The solution supports multifactor authentication options as well as more advanced configurations. Always On is free so long as you have a valid server license and deploy Windows 10 Anniversary update or later. It does not require enterprise licensing.

[1] https://docs.microsoft.com/en-us/windows-server/remote/remote-access/vpn/always-on-vpn/deploy/always-on-vpn-deploy

[2] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn464273(v=ws.11)



Post-Authentication Checks

Many VPN clients support post-authentication checks

- Most common is to check antivirus and firewall

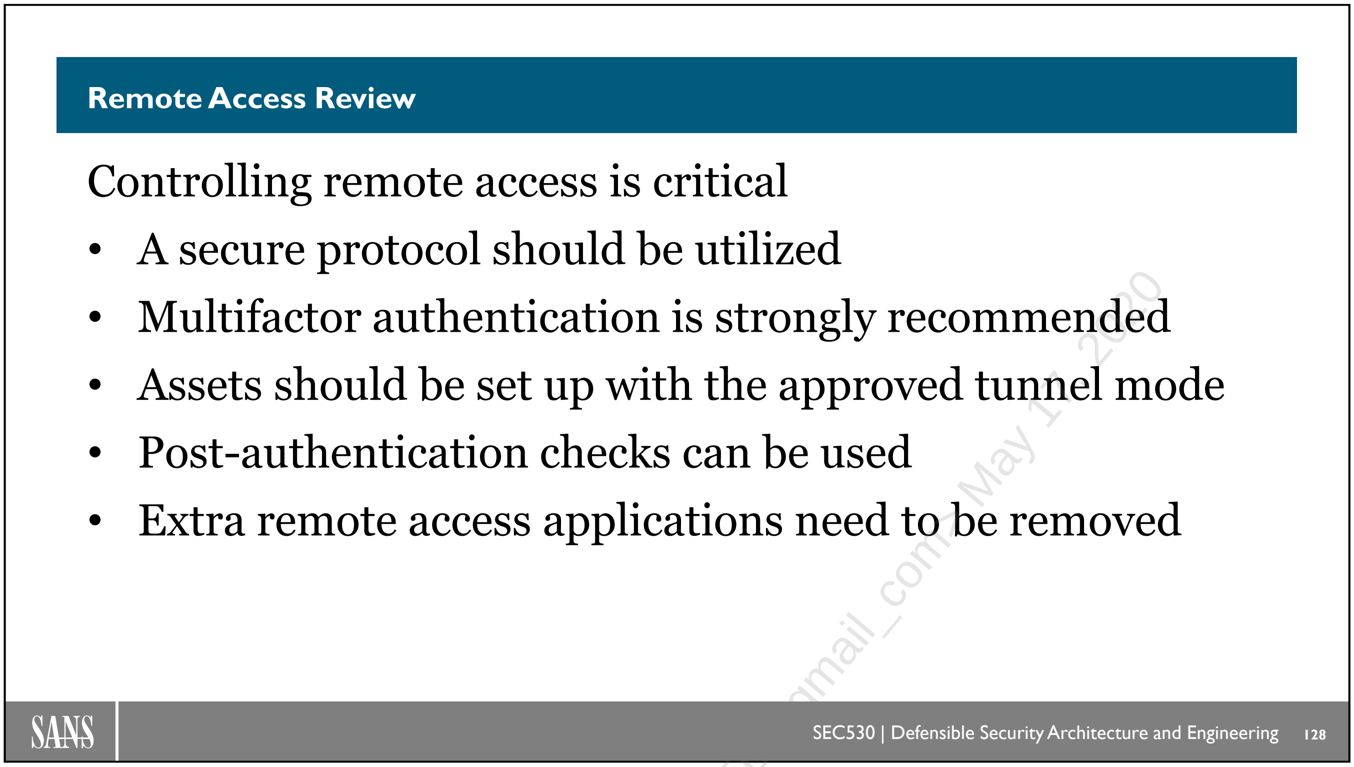
Other possibilities

- Vulnerability scan with automated remediation
- Patch verification with automated remediation
- Custom scripts (look for new running processes)
- A user associated with the **new asset** (require approval)

Longer an asset has been disconnected the more checks need run

Post-Authentication Checks

VPN solutions often support post-authentication checks. These checks can be used to verify a system has antivirus installed with current definitions, that a host-based firewall is on, as well as any other requirements. These checks can often include automatic remediation and place clients in various subnets depending on their risk. The implementation is similar to network access control (NAC).



Remote Access Review

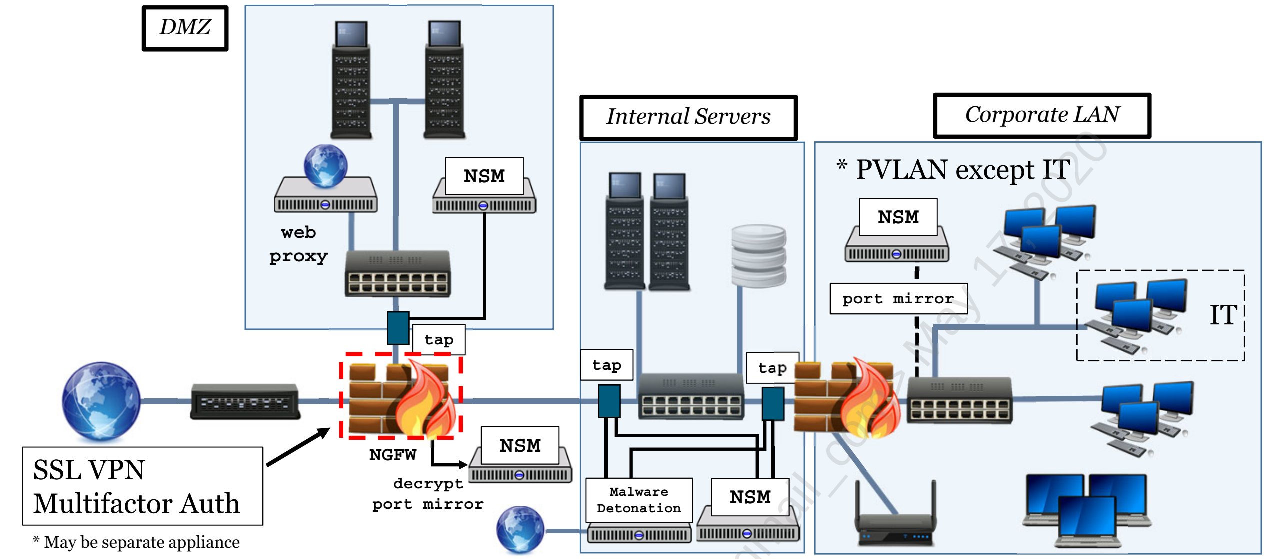
Controlling remote access is critical

- A secure protocol should be utilized
- Multifactor authentication is strongly recommended
- Assets should be set up with the approved tunnel mode
- Post-authentication checks can be used
- Extra remote access applications need to be removed

Remote Access Review

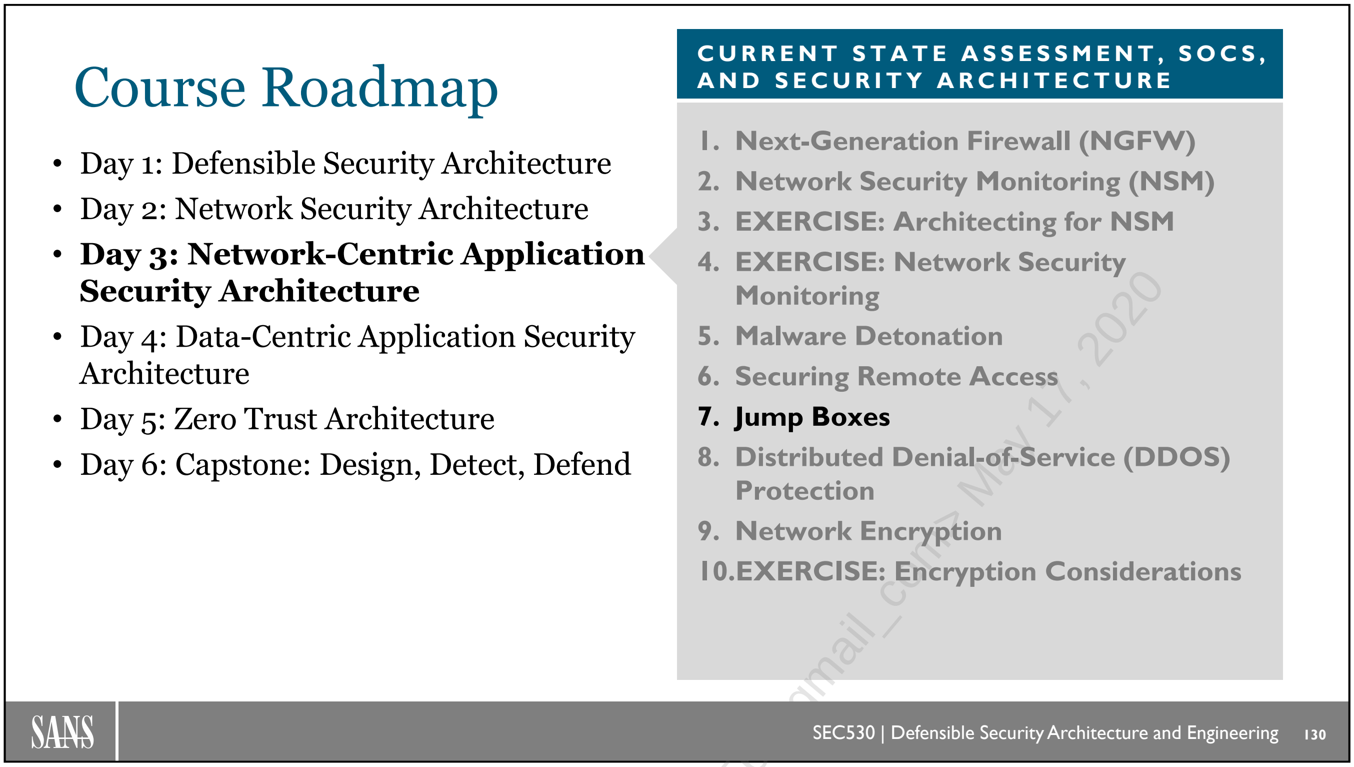
Remote access to an organization is a critical architecture component that must be designed with security in mind. If employees and vendors can remote into an organization however they see fit, an organization should expect trouble. A secure method of remote access should include multifactor authentication and strong encryption. VPN clients should be selected and always on, and full-tunneling should be considered for extra security. All unauthorized remote access solutions should be blocked.

Case Study: Tyrrell Corporation



Case Study: Tyrrell Corporation

This diagram shows the addition of a SSL VPN with multifactor authentication requirements to the Tyrrell Corporation architecture. In this example, users can remotely connect into the environment using the SSL VPN with multifactor authentication.



Course Roadmap

- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

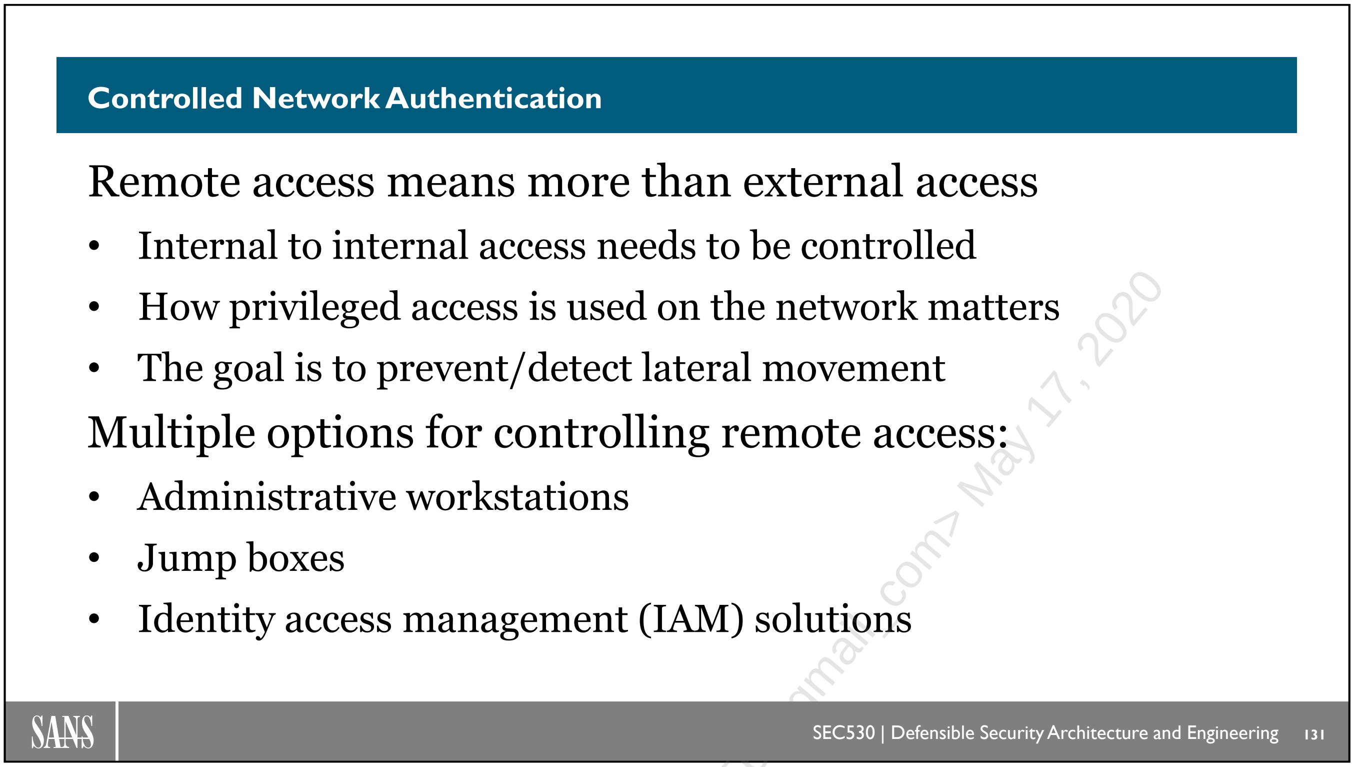
1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. **Jump Boxes**
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. EXERCISE: Encryption Considerations

SANS

SEC530 | Defensible Security Architecture and Engineering 130

Course Roadmap

The next section covers Jump Boxes.



Controlled Network Authentication

Remote access means more than external access

- Internal to internal access needs to be controlled
- How privileged access is used on the network matters
- The goal is to prevent/detect lateral movement

Multiple options for controlling remote access:

- Administrative workstations
- Jump boxes
- Identity access management (IAM) solutions

Controlled Network Authentication

Remote access needs to include more than just external to internal. How an organization handles internal remote authentication and access plays a significant factor in preventing and detecting unauthorized lateral movement. Organizations should consider using administrative workstations, jump boxes, and identity access management (IAM) solutions.

Common Attacks with Lateral Movement

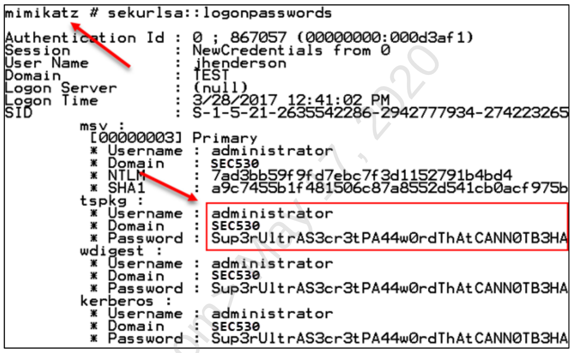
Primary issue around remote access is credential theft and re-use

- Malware/Adversaries highly successful at stealing credentials
- Credential on disk or in RAM

Cleartext password retrieval

- Mimikatz[1]
- LSA Secrets
- Saved passwords

Hash Dumping/Token Smuggling



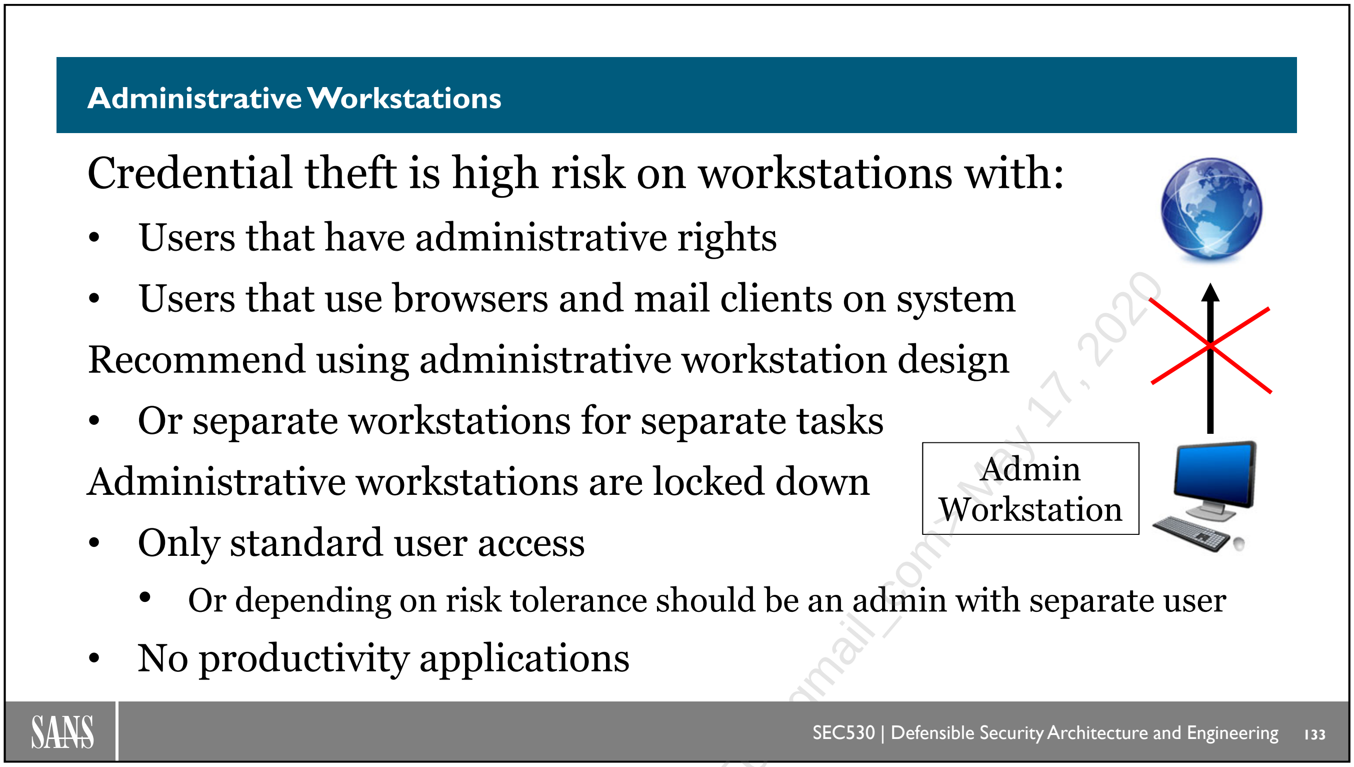
```
mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 867057 (00000000:000d3af1)
Session           : NewCredentials from 0
User Name         : jhenderson
Domain            : TEST
Logon Server      : (null)
Logon Time        : 3/28/2017 12:41:02 PM
SID               : S-1-5-21-2635542286-2942777934-274223265
        msv :
         [00000003] Primary
         * Username : administrator
         * Domain   : SEC530
         * NTLM     : 7ad3bb59f9fd7ebc7f3d1152791b4bd4
         * SHA1     : a9c7455b1f481506c87a8552d541cb0acf975b
        tspkg :
         * Username : administrator
         * Domain   : SEC530
         * Password : Sup3rUltrAS3cr3tPA44w0rdThAtCANN0TB3HA
        wdigest :
         * Username : administrator
         * Domain   : SEC530
         * Password : Sup3rUltrAS3cr3tPA44w0rdThAtCANN0TB3HA
        kerberos :
         * Username : administrator
         * Domain   : SEC530
         * Password : Sup3rUltrAS3cr3tPA44w0rdThAtCANN0TB3HA
```

Common Attacks with Lateral Movement

In almost every major breach or penetration test credential theft and reuse plays a key part. Exploitation works, but it is far more common for credential use to decide the fate of an organization. Tools like Mimikatz exist that can steal credentials from RAM. Other tools can retrieve passwords, password hashes, or password tokens from a system.

Because so many tools are trivial to use, sensitive or privileged access internally requires additional controls.

[1] https://github.com/gentilkiwi/mimikatz



Administrative Workstations

Credential theft is high risk on workstations with:

- Users that have administrative rights
- Users that use browsers and mail clients on system

Recommend using administrative workstation design

- Or separate workstations for separate tasks

Administrative workstations are locked down

- Only standard user access
  - Or depending on risk tolerance should be an admin with separate user
- No productivity applications

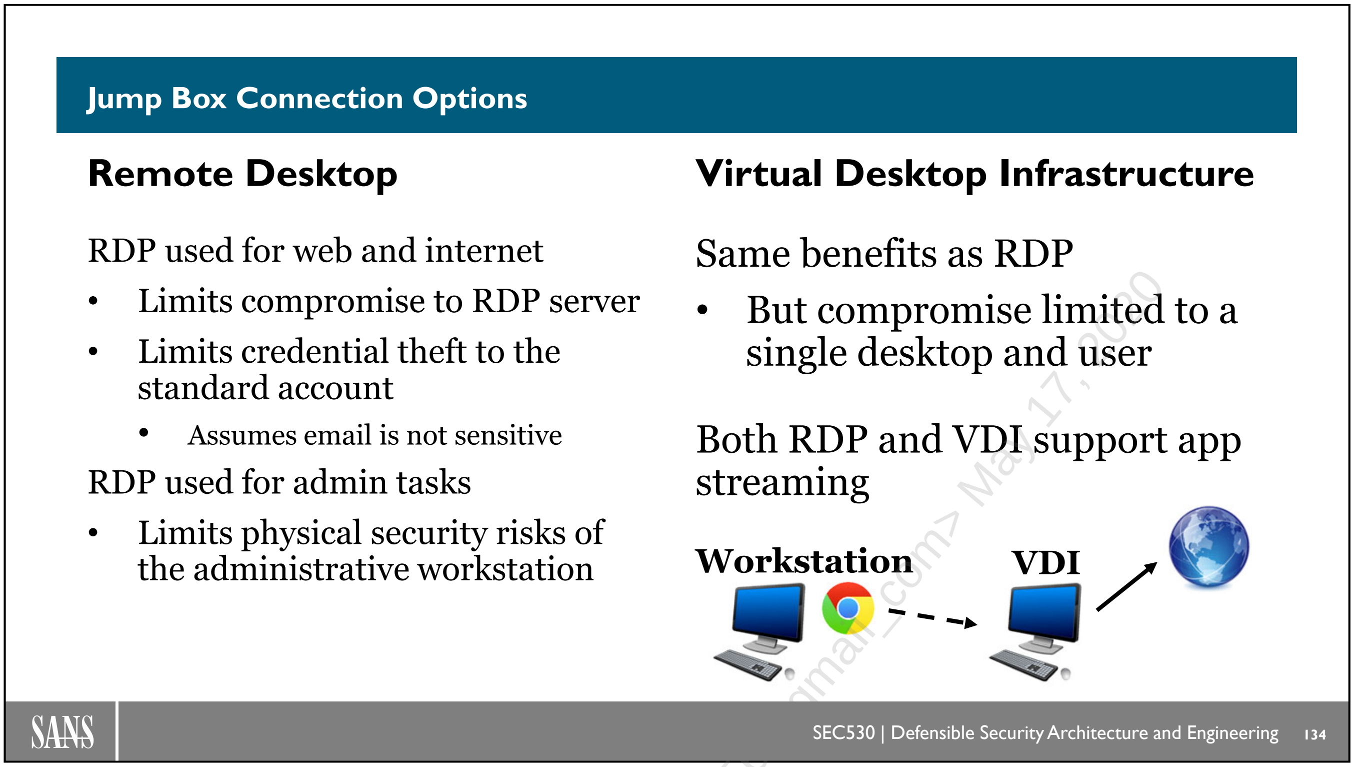
Administrative Workstations

Workstations that have administrative rights run the most risk of credential theft because many of the tools require special rights or privileges to work. Workstations that are used to browse the internet or check email are at high risk of compromise. Combine the two together, and there is a high risk of compromise and credential theft.

One method to lower the risk is to use separate physical workstations for various tasks. However, this solution is expensive and inconvenient. Fortunately, alternative solutions exist that can bolster security. The key is separating key rights and privileges from high-risk use.

[1] https://blogs.technet.microsoft.com/askpfeplat/2016/03/14/secure-administrative-workstations/

[2] https://docs.microsoft.com/en-us/windows-server/identity/securing-privileged-access/privileged-access-workstations



Jump Box Connection Options

Remote Desktop

RDP used for web and internet

- Limits compromise to RDP server
- Limits credential theft to the standard account
    - Assumes email is not sensitive

RDP used for admin tasks

- Limits physical security risks of the administrative workstation

Virtual Desktop Infrastructure

Same benefits as RDP

- But compromise limited to a single desktop and user

Both RDP and VDI support app streaming

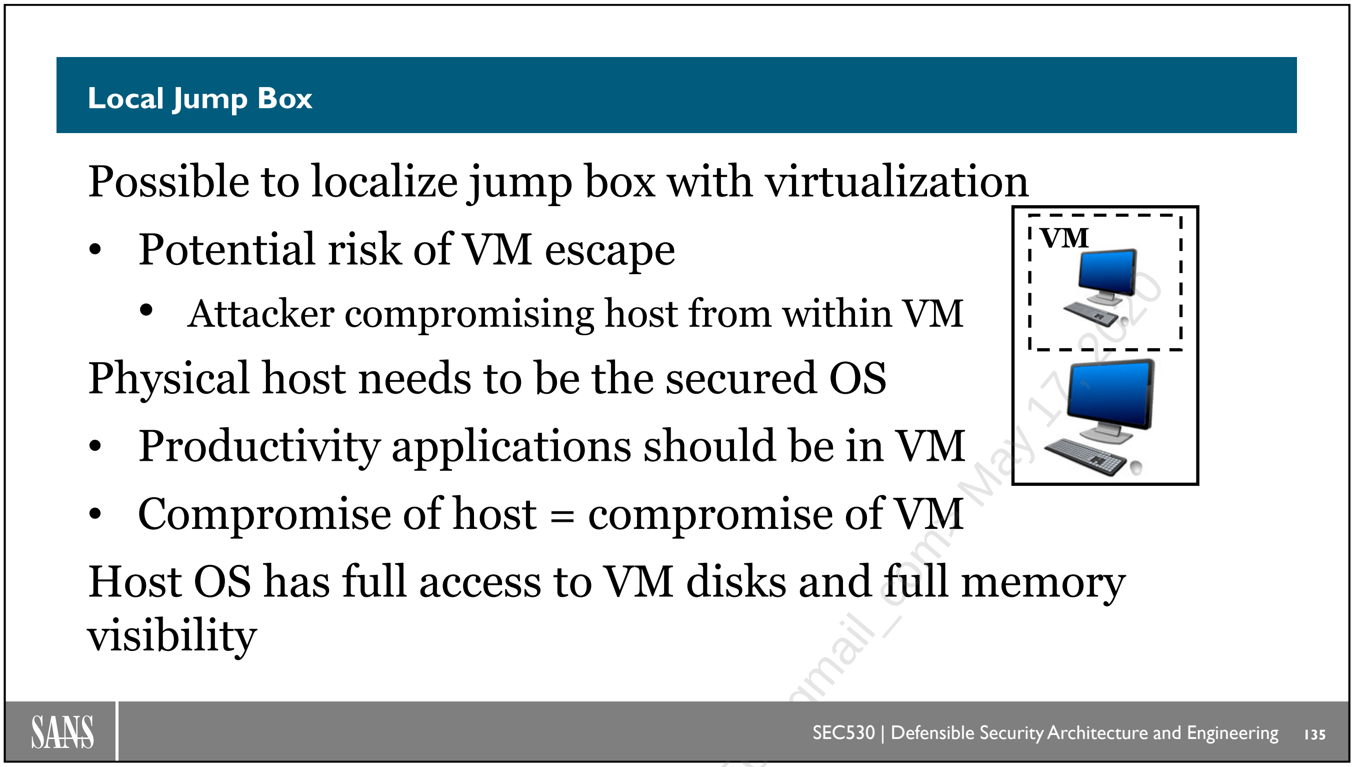
Jump Box Connection Options

Instead of having two physical workstations employees can utilize remote desktop or virtual desktop infrastructure. For example, an end user's desktop can be locked down and secured and then either RDP or VDI can be used with a separate user account to access the internet and email. Then either the locked down workstation can be used for administrative tasks or another RDP or VDI system can be used for those purposes. By pushing the purpose and level of access to different systems, the risk is dramatically reduced. The systems used for internet access may still get compromised, but the credentials on those systems are limited. Plus, the internet access systems can have additional controls around them such as firewall rules preventing internal access from those systems.

Both terminal services and virtual desktop infrastructure support application streaming. Application streaming involves making an application seem like it is installed locally, but the application is actually running on a dedicated server or workstation. Streaming the application to desktops can help limit risks without the need for users to use two GUI desktop interfaces.

[1] https://docs.microsoft.com/en-us/microsoft-desktop-optimization-pack/appv-v4/how-to-configure-the-application-virtualization-streaming-servers

[2] https://docs.microsoft.com/en-us/windows-server/identity/securing-privileged-access/privileged-access-workstations



Local Jump Box

Possible to localize jump box with virtualization

- Potential risk of VM escape
  - Attacker compromising host from within VM

Physical host needs to be the secured OS

- Productivity applications should be in VM
- Compromise of host = compromise of VM

Host OS has full access to VM disks and full memory visibility

SEC530 | Defensible Security Architecture and Engineering 135

Local Jump Box

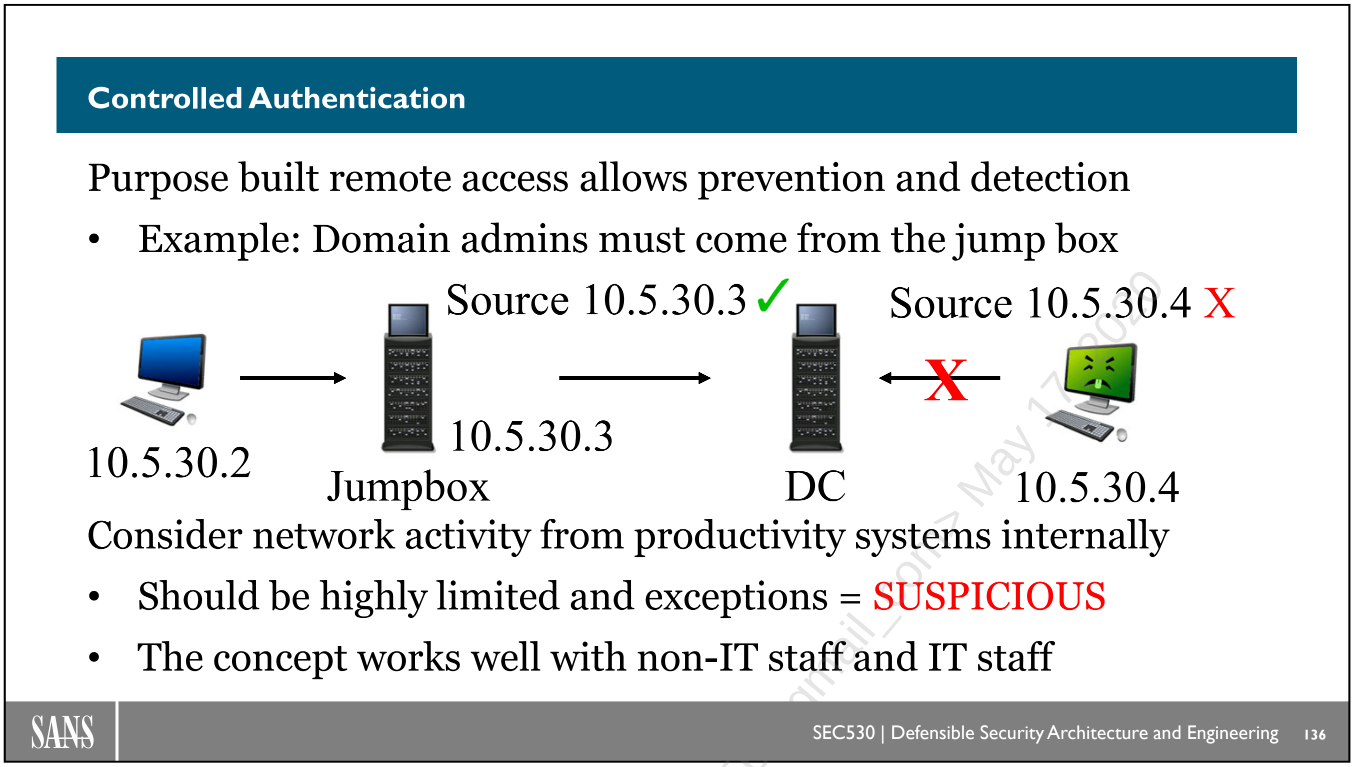
Virtualization software such as Virtualbox[1] or VMware Workstation/Fusion[2] can be used to implement a local version of jump boxes. The solution involves using the host operating system for administrative or business tasks and using a local virtual machine for productivity access. With this design, compromise will likely be limited to the local VM. While it is possible for an attacker to escape the virtual machine to attack the host, it is much less likely to happen compared to having a user run both administrative tasks and productivity applications directly on one system.

The key design for this is that the productivity applications are not allowed or ran on the host operating system. Compromise of the host operating system would result in compromise of any virtual machines due to the ability to read the disk and memory of the host.

[1] https://www.virtualbox.org/wiki/Downloads

[2] https://www.vmware.com/products/workstation-pro.html

[3] https://docs.microsoft.com/en-us/windows-server/identity/securing-privileged-access/privileged-access-workstations



Controlled Authentication

Purpose built remote access allows prevention and detection

- Example: Domain admins must come from the jump box

Source 10.5.30.3 ✓

Source 10.5.30.4 X

10.5.30.2

Jumpbox

10.5.30.3

DC

10.5.30.4

Consider network activity from productivity systems internally

- Should be highly limited and exceptions = SUSPICIOUS
- The concept works well with non-IT staff and IT staff

SANS

SEC530 | Defensible Security Architecture and Engineering 136

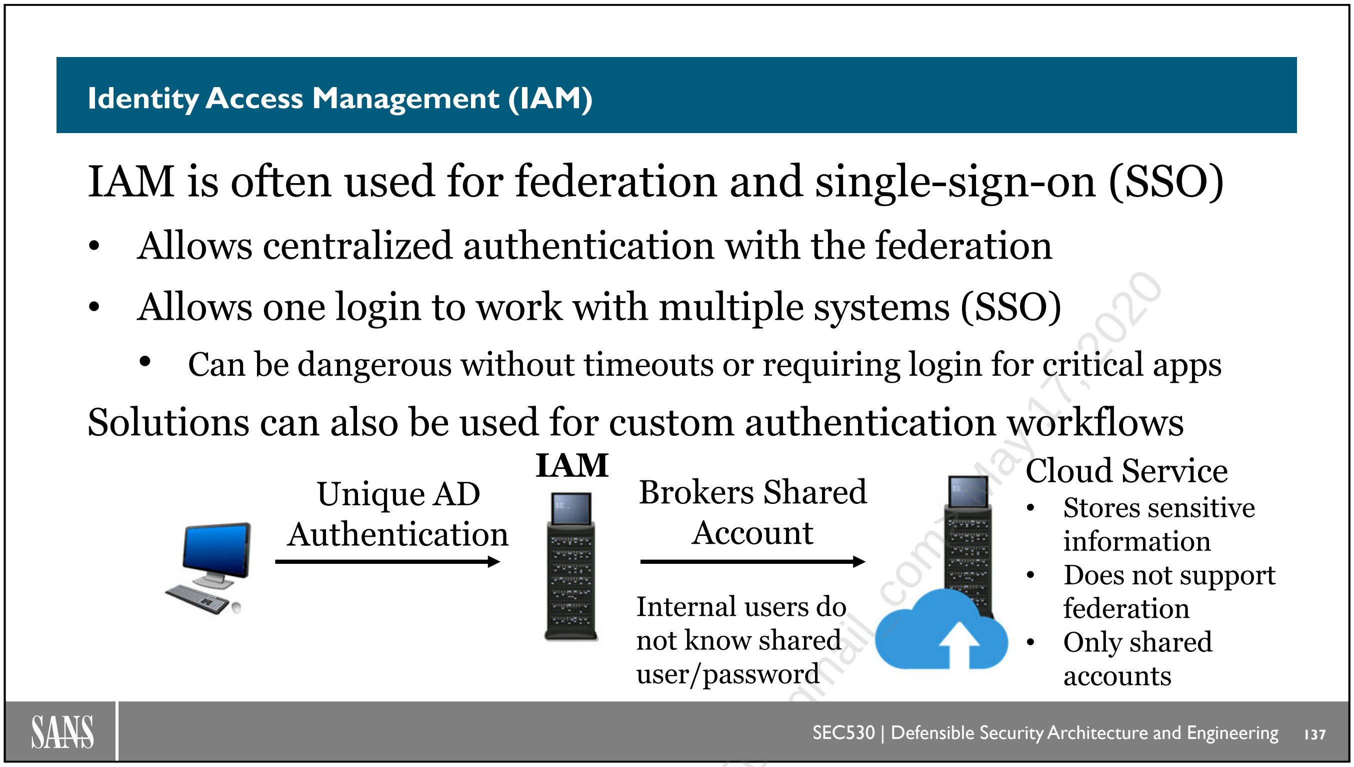
Controlled Authentication

One thing that can greatly aid in monitoring privileged accounts is to force policy for how access is handled. For example, having a policy that states the use of domain administrator use can only come from a jump box or specific machines or subnets is a great idea. This allows controls and detection rules to be used to look for all attempted access outside this authentication flow.

For example, domain administrator logons sourcing from a jump box would be validated based on user account, login type, and source of the login type. This could be RDP sessions, PowerShell commands, or anything that domain administrator access requires. The control does not limit using domain administrator credentials on other boxes but instead controls the flow of access.

For example, if someone needed to RDP into a domain controller and policy dictates domain administrator access should source from a jump box, they would need to first RDP into the jump box and then from the jump box, RDP into the domain controller. If someone tried to RDP from any other system outside the jump box to the domain controller, this would be red flagged. Controlled authentication works best when using local Windows or Linux settings to restrict logins. For example, in Windows Login Rights1 can limit who can log in to a system and how the login is allowed.

[1] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc976701(v=technet.10)



Identity Access Management (IAM)

IAM is often used for federation and single-sign-on (SSO)

- Allows centralized authentication with the federation
- Allows one login to work with multiple systems (SSO)
  - Can be dangerous without timeouts or requiring login for critical apps

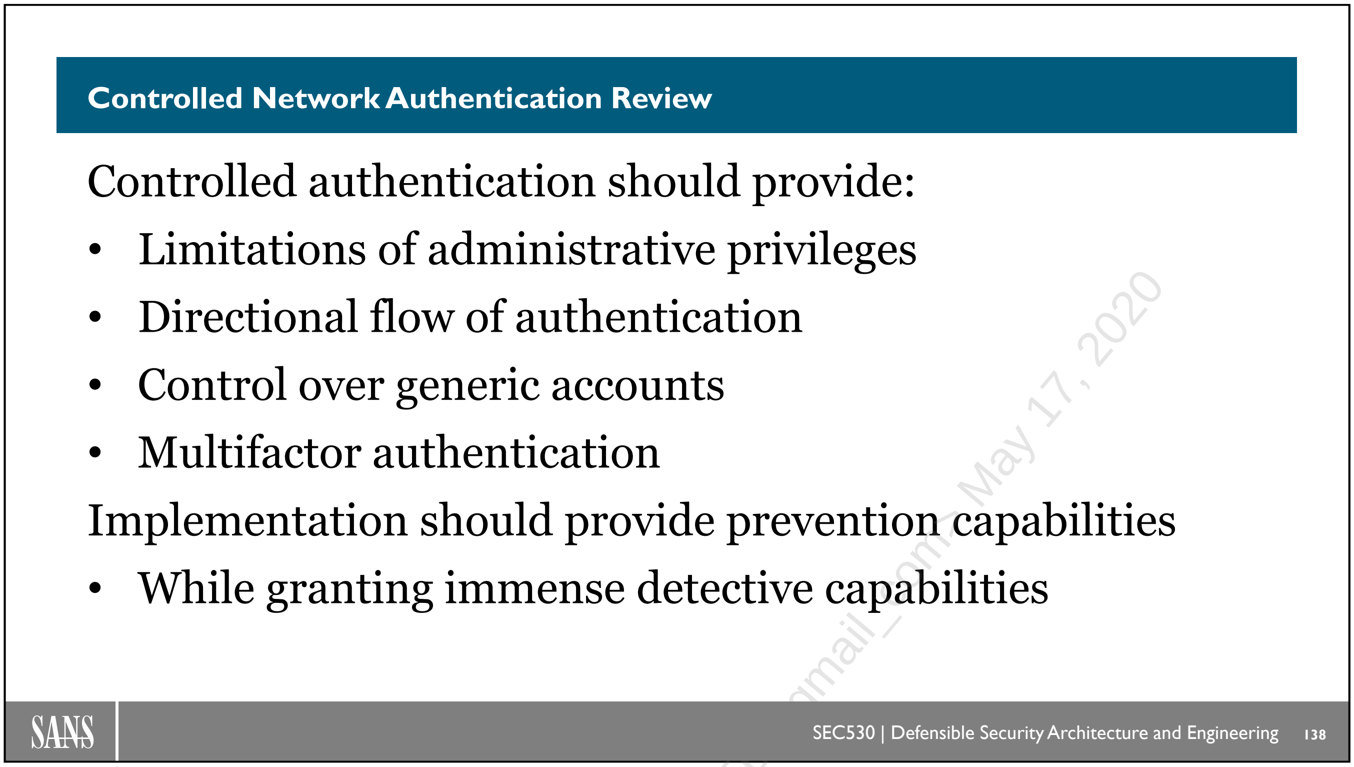
Solutions can also be used for custom authentication workflows

- Stores sensitive information
- Does not support federation
- Only shared accounts

Identity Access Management (IAM)

IAM is designed to control how, when, and where authentication is handled for enterprise assets. It provides single-sign-on and federated authentication for ease of use. The key to a secure implementation is the use of multifactor authentication and login timeouts due to a single account login allowing access to multiple resources.

On top of IAM's traditional deployments, IAM can be used for advanced custom workflows. For example, assume a web site must be used for business purposes but it does not allow centralized accounts, does not have an API for account creation, and only contains a single or a few local accounts. Assume further that the site is used for something that mandates unique account use such as HIPAA or PCI. How would an organization secure the site and be compliant? While this sounds like a scenario that should never happen, it is a commonplace. One method to handle this is to use an IAM solution to handle unique user authentication to the IAM service and then have the IAM login to the back-end or cloud-hosted service with the local shared account(s). By brokering the connection and mandating a successful unique login an audit log is generated proving who did what while minimizing the overhead of manually creating accounts for every user.



Controlled Network Authentication Review

Controlled authentication should provide:

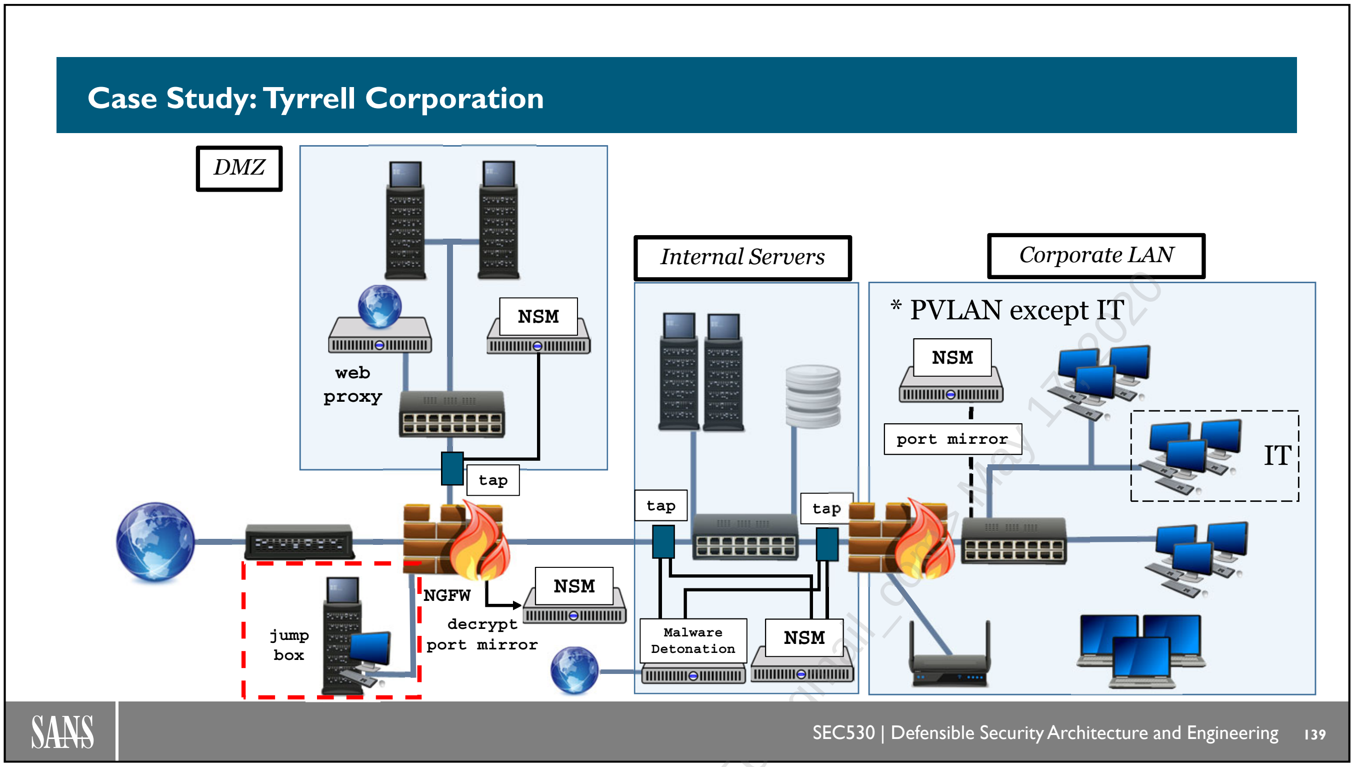
- Limitations of administrative privileges
- Directional flow of authentication
- Control over generic accounts
- Multifactor authentication

Implementation should provide prevention capabilities

- While granting immense detective capabilities

Controlled Network Authentication Review

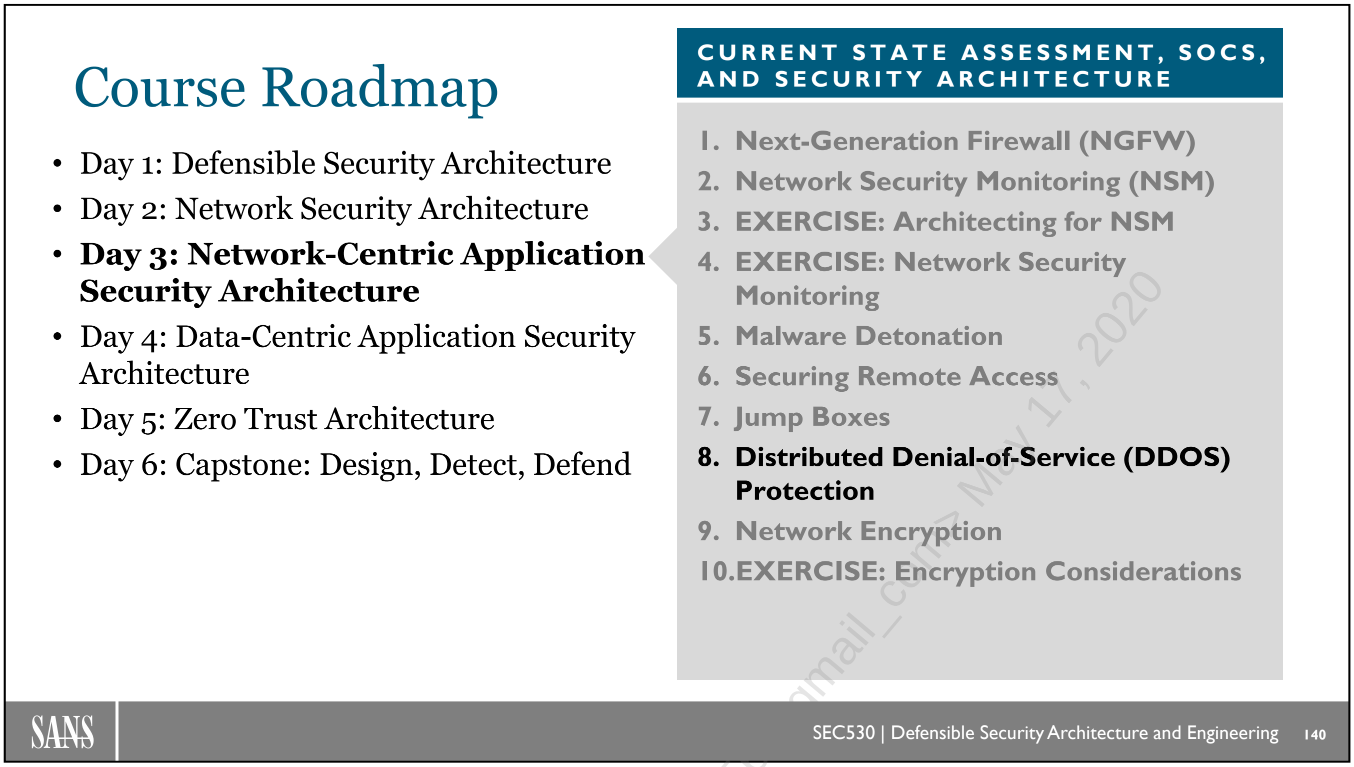
Placing controls around remote authentication is important for both external and internal systems and access. Privilege separation is necessary to secure sensitive accounts whether due to their administrative powers or the criticality of the data the accounts can access. Also, by controlling the flow of authentication additional preventative and detective capabilities are born.



Case Study: Tyrrell Corporation

Case Study: Tyrrell Corporation

This diagram shows the addition of remote access controls to mitigate the risk of unauthorized lateral movement to the Tyrell Corporation architecture. To do so, we will make use of administrative workstations, jump boxes, and identity access management (IAM) solutions.



Course Roadmap

- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

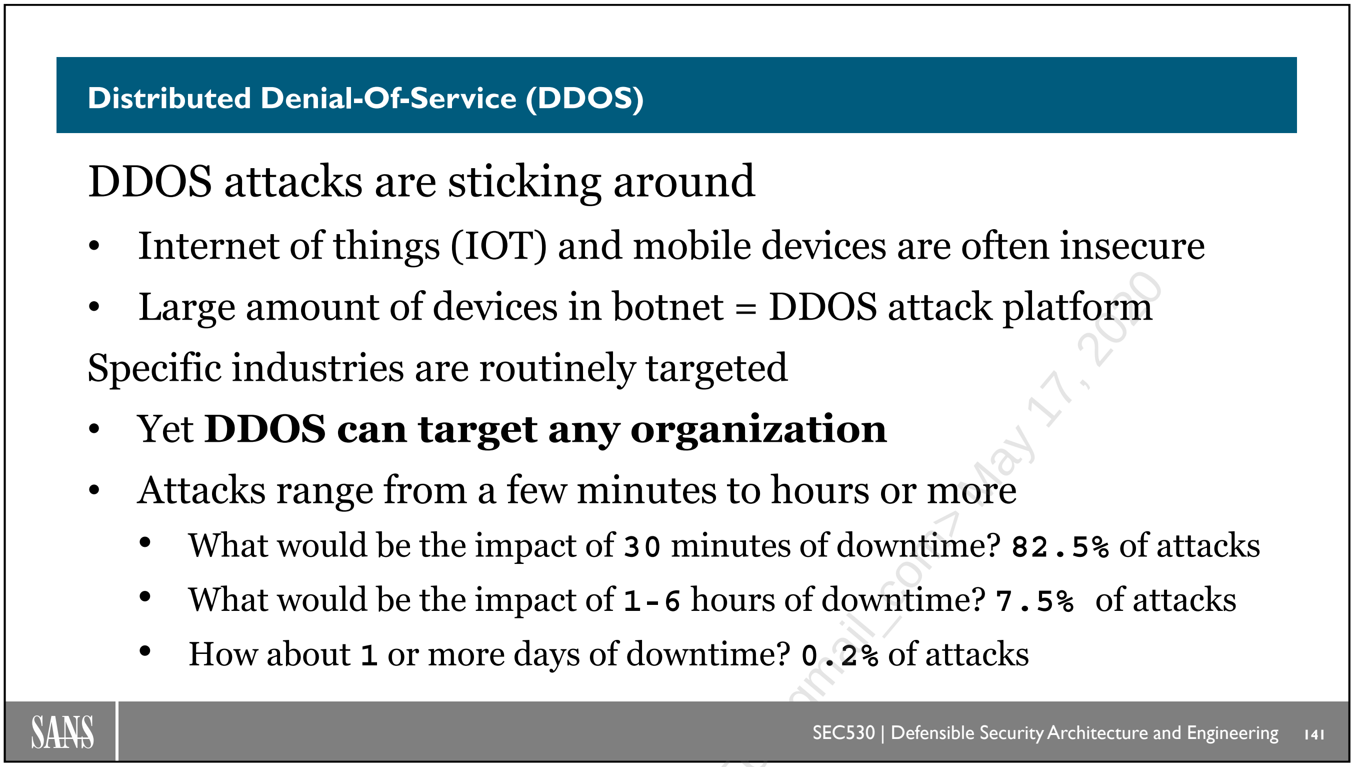
1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. **Distributed Denial-of-Service (DDOS) Protection**
9. Network Encryption
10. EXERCISE: Encryption Considerations

SANS

SEC530 | Defensible Security Architecture and Engineering 140

Course Roadmap

The next section covers Distributed Denial-of-Service (DDOS) protection.



Distributed Denial-Of-Service (DDOS)

DDOS attacks are sticking around

- Internet of things (IOT) and mobile devices are often insecure
- Large amount of devices in botnet = DDOS attack platform

Specific industries are routinely targeted

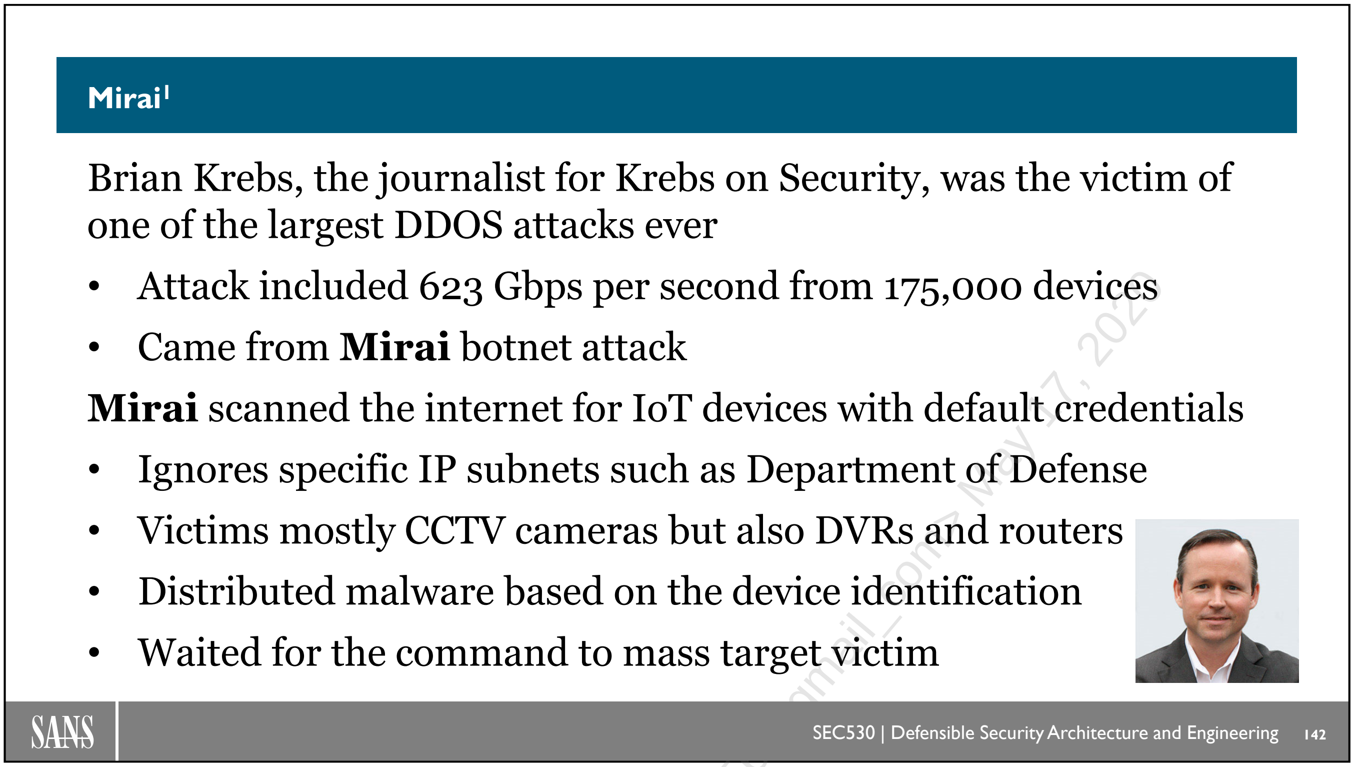
- Yet **DDOS can target any organization**
- Attacks range from a few minutes to hours or more
  - What would be the impact of `30` minutes of downtime? `82.5%` of attacks
  - What would be the impact of `1-6` hours of downtime? `7.5%` of attacks
  - How about `1` or more days of downtime? `0.2%` of attacks

Distributed Denial-Of-Service (DDOS)

Distributed denial-of-service (DDOS) is an attack that focuses on disrupting service. Attackers use DDOS to demand ransom or to attempt to destroy an organization. DDOS is more prevalent in certain industries such as service providers, gaming, and financial institutions, but it also is used to target businesses of any industry or size.

There are some truths organizations need to understand. Botnets are not going away, and if anything, they are becoming larger and easier to obtain due to the internet of things. These botnets are for hire or personal use and make DDOS something that is likely to become more prevalent.

[1] https://www.incapsula.com/ddos-report/ddos-report-q2-2017.html



Mirai[1]

Brian Krebs, the journalist for Krebs on Security, was the victim of one of the largest DDOS attacks ever

- Attack included 623 Gbps per second from 175,000 devices
- Came from **Mirai** botnet attack

Mirai scanned the internet for IoT devices with default credentials

- Ignores specific IP subnets such as Department of Defense
- Victims mostly CCTV cameras but also DVRs and routers
- Distributed malware based on the device identification
- Waited for the command to mass target victim

SANS SEC530 | Defensible Security Architecture and Engineering 142

Mirai[1]

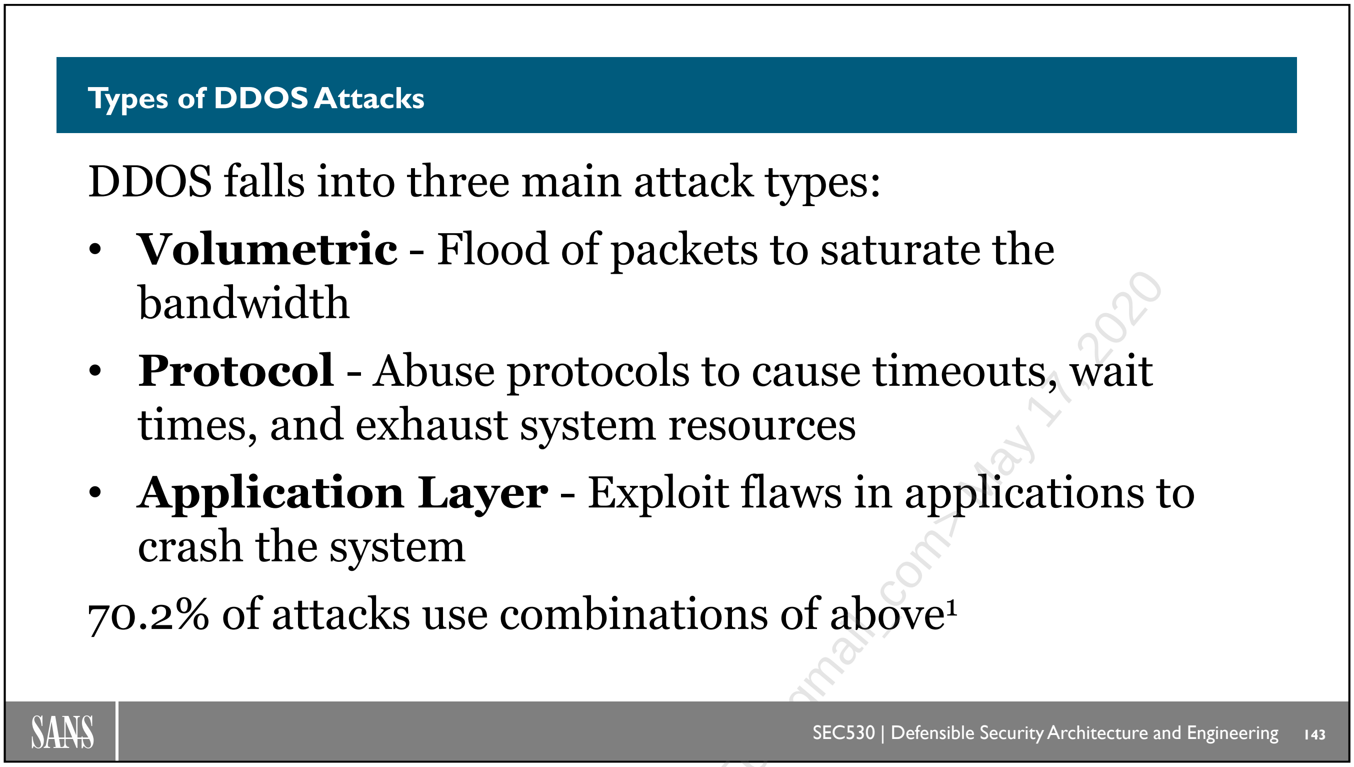
An example of a recent and modern DDOS attack is Mirai. The Mirai botnet targeted multiple organizations as well as the well-known journalist Brian Krebs. Oddly enough, Brian Krebs has suffered one of the largest DDOS attacks recorded in an attack that sustained up to 623 Gbps.

The Mirai botnet works by scanning for the internet of things devices such as CCTV cameras and DVRs. If default credentials are found, the botnet delivers a malicious payload to obtain control of the device and then hardens the device to keep out other malware. Mirai then waits for the command to attack a target such as Brian Krebs. Mirai uses a combination of multiple denials of service techniques were 32.8% are volumetric, 39.8%, are TCP state exhaustion, and 34.5% are application-layer.

[1] https://www.incapsula.com/blog/malware-analysis-mirai-ddos-botnet.html

[2] https://krebsonsecurity.com/2017/12/mirai-iot-botnet-co-authors-plead-guilty/#more-41717

[3] https://www.usenix.org/system/files/conference/usenixsecurity17/sec17-antonakakis.pdf



Types of DDOS Attacks

DDOS falls into three main attack types:

- **Volumetric** - Flood of packets to saturate the bandwidth
- **Protocol** - Abuse protocols to cause timeouts, wait times, and exhaust system resources
- **Application Layer** - Exploit flaws in applications to crash the system

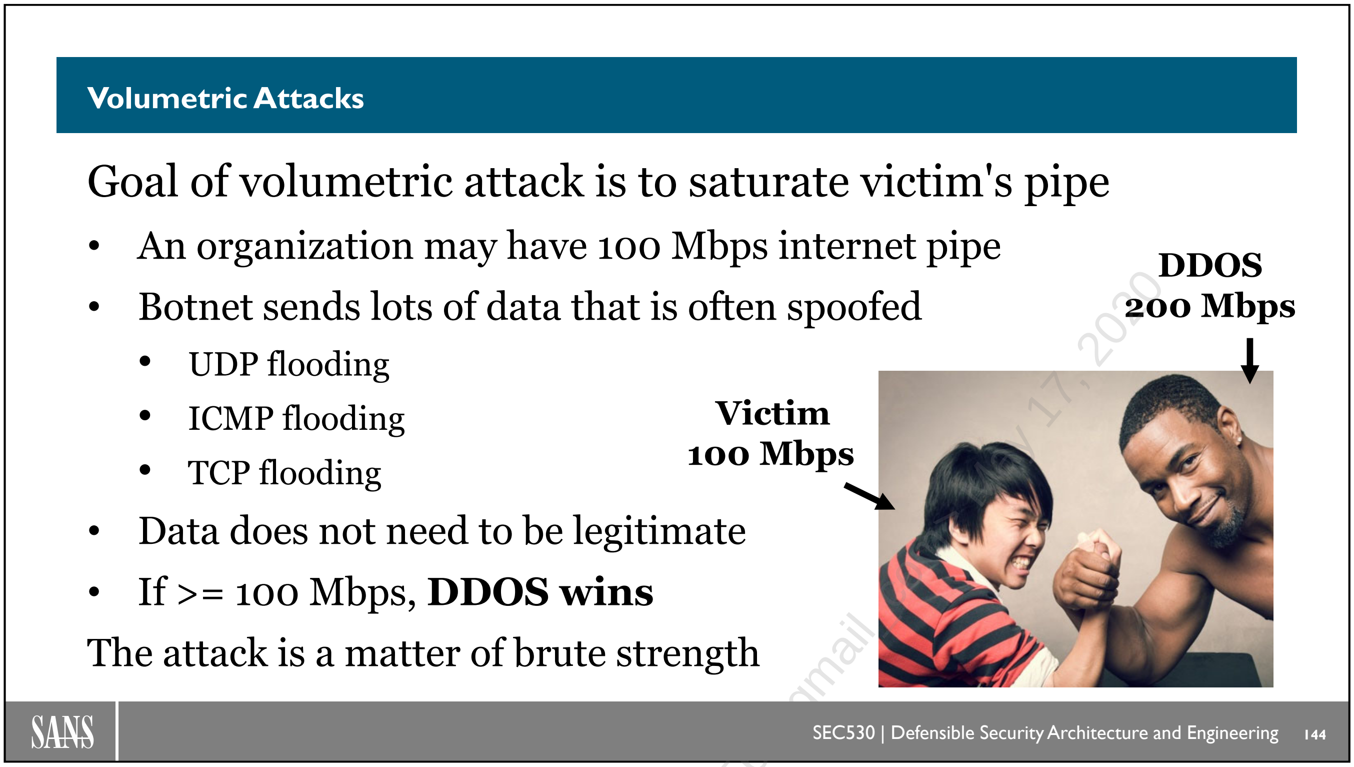
70.2% of attacks use combinations of above[1]

Types of DDOS Attacks

DDOS attacks fall into three categories: volumetric, protocol, and application layer. Volumetric attacks involve high amounts of traffic to saturate bandwidth to the point new connections cannot be made. Protocol attacks involve abusing traditional protocols like TCP so that systems end up waiting to complete a transaction and ends up with resource exhaustion and crashing. Application layer involves attacks that exploit a weakness specific to a given application.

[1] https://www.incapsula.com/ddos-report/ddos-report-q3-2017.html

[2] https://www.incapsula.com/ddos/ddos-attacks/



Volumetric Attacks

Goal of volumetric attack is to saturate victim's pipe

- An organization may have 100 Mbps internet pipe
- Botnet sends lots of data that is often spoofed
  - UDP flooding
  - ICMP flooding
  - TCP flooding
- Data does not need to be legitimate
- If >= 100 Mbps, **DDOS wins**

The attack is a matter of brute strength

Victim
100 Mbps

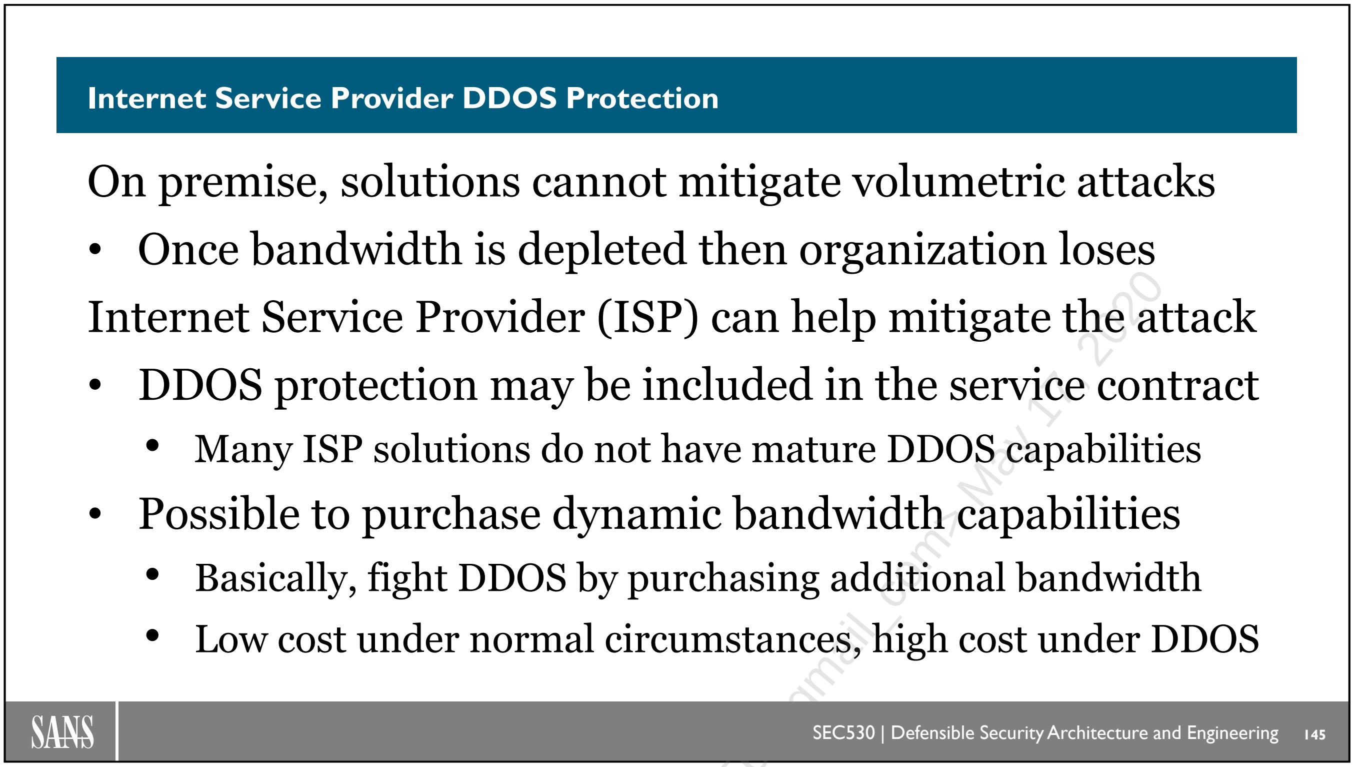
DDOS
200 Mbps

SANS

SEC530 | Defensible Security Architecture and Engineering 144

Volumetric Attacks

A volumetric attack is one of the simplest to understand. Basically, whoever can handle the most bandwidth wins. This would be similar to two individuals arm wrestling. The strength of one can overpower the other. Another way of thinking of this is the weight restriction on chairs. A kid's seat is rated for a specific amount of weight. Normally, kids can take turns sitting on the chair, and everything is fine. But if an adult sits on the chair, it can break and then no kids are able to use the chair anymore.



Internet Service Provider DDOS Protection

On premise, solutions cannot mitigate volumetric attacks

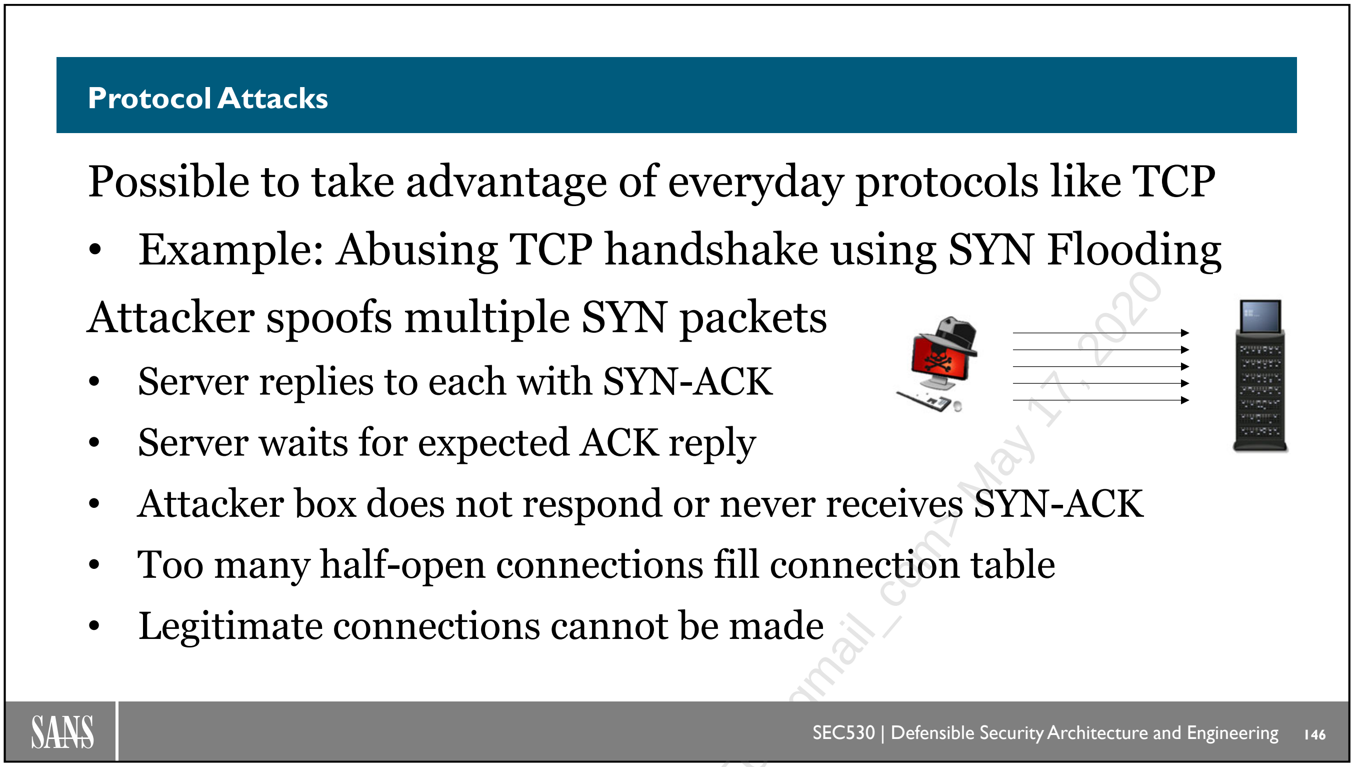
- Once bandwidth is depleted then organization loses

Internet Service Provider (ISP) can help mitigate the attack

- DDOS protection may be included in the service contract
  - Many ISP solutions do not have mature DDOS capabilities
- Possible to purchase dynamic bandwidth capabilities
  - Basically, fight DDOS by purchasing additional bandwidth
  - Low cost under normal circumstances, high cost under DDOS

Internet Service Provider DDOS Protection

One way to handle volumetric attacks is to partner with your internet service provider (ISP). Some ISPs have the ability to provide DDOS protection within a contract. Usually, this is fairly rudimentary protection. An alternative way the ISP can be utilized is purchasing a plan that allows dynamically increasing maximum bandwidth as necessary. Having a near unlimited internet pipe can help fight network saturation, but under DDOS the costs may be astronomical.



Protocol Attacks

Possible to take advantage of everyday protocols like TCP

- Example: Abusing TCP handshake using SYN Flooding

Attacker spoofs multiple SYN packets

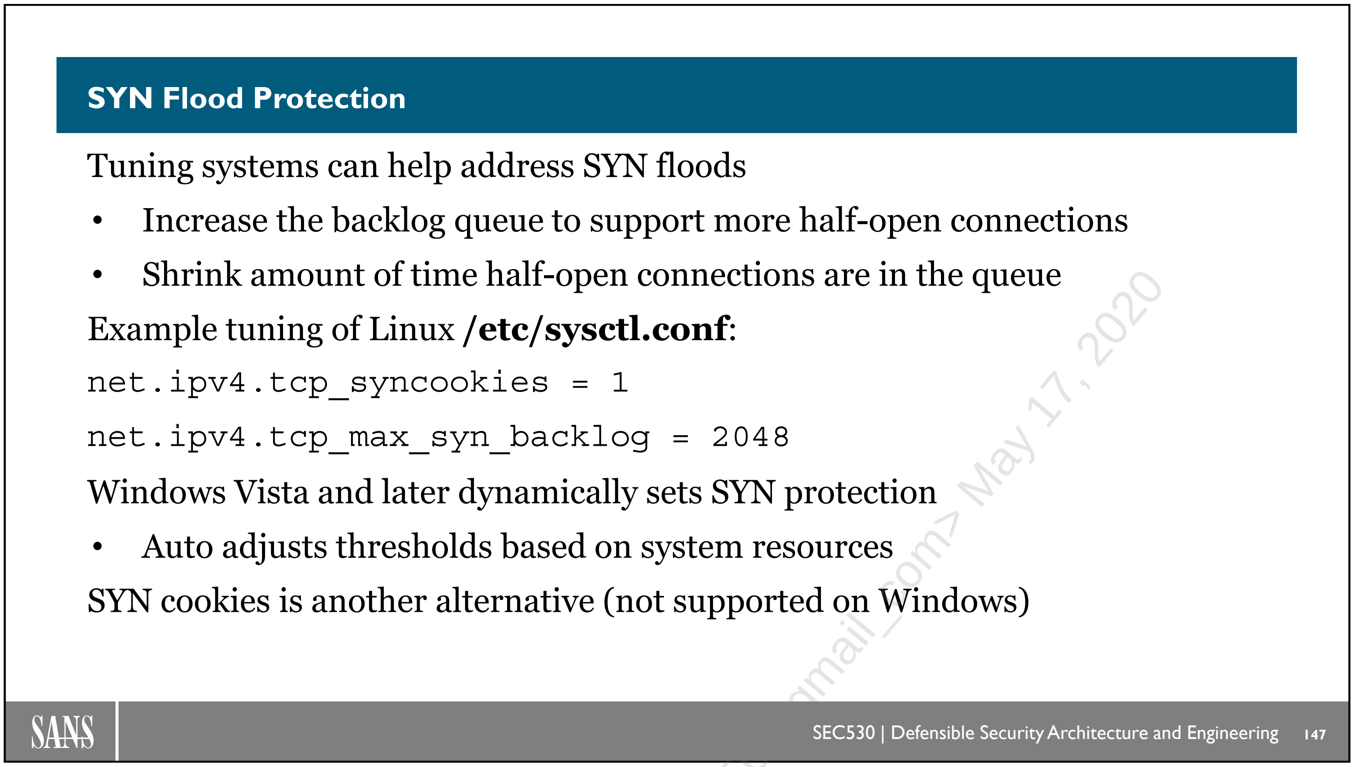
- Server replies to each with SYN-ACK
- Server waits for expected ACK reply
- Attacker box does not respond or never receives SYN-ACK
- Too many half-open connections fill connection table
- Legitimate connections cannot be made

Protocol Attacks

Protocol attacks focus on abusing the way protocols work. For example, an SYN flood sends multiple spoofed SYN packets to a victim. An example of this would be an attacker with an IP address of 5.30.0.1 sending a bunch of SYN packets to 1.0.3.5 but in a way that the source IP address of the packets is a fake IP such as 8.8.8.8. When 1.0.3.5 receives the packets, they appear from 8.8.8.8 instead of 5.30.0.1.

Under normal conditions, a TCP handshake involves three packets: an SYN packet from a client, an SYN/ACK packet from the server back to the client, and then a client replying with an ACK back to the server. However, with a syn flood the attacker never responds and likely never receives the server's SYN/ACK.

The goal is to saturate the victim's connection table by filling it with half-open connections. Once the session table is full, no new connections are accepted.



SYN Flood Protection

Tuning systems can help address SYN floods

- Increase the backlog queue to support more half-open connections
- Shrink amount of time half-open connections are in the queue

Example tuning of Linux **/etc/sysctl.conf**:

```
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 2048
```

Windows Vista and later dynamically sets SYN protection

- Auto adjusts thresholds based on system resources

SYN cookies is another alternative (not supported on Windows)

SANS

SEC530 | Defensible Security Architecture and Engineering 147

SYN Flood Protection

There are multiple strategies for protecting against SYN floods that vary from system to system. The most common methods involve increasing the session table so that it can handle more connections before filling. Windows does this dynamically based on the number of resources a system has available.

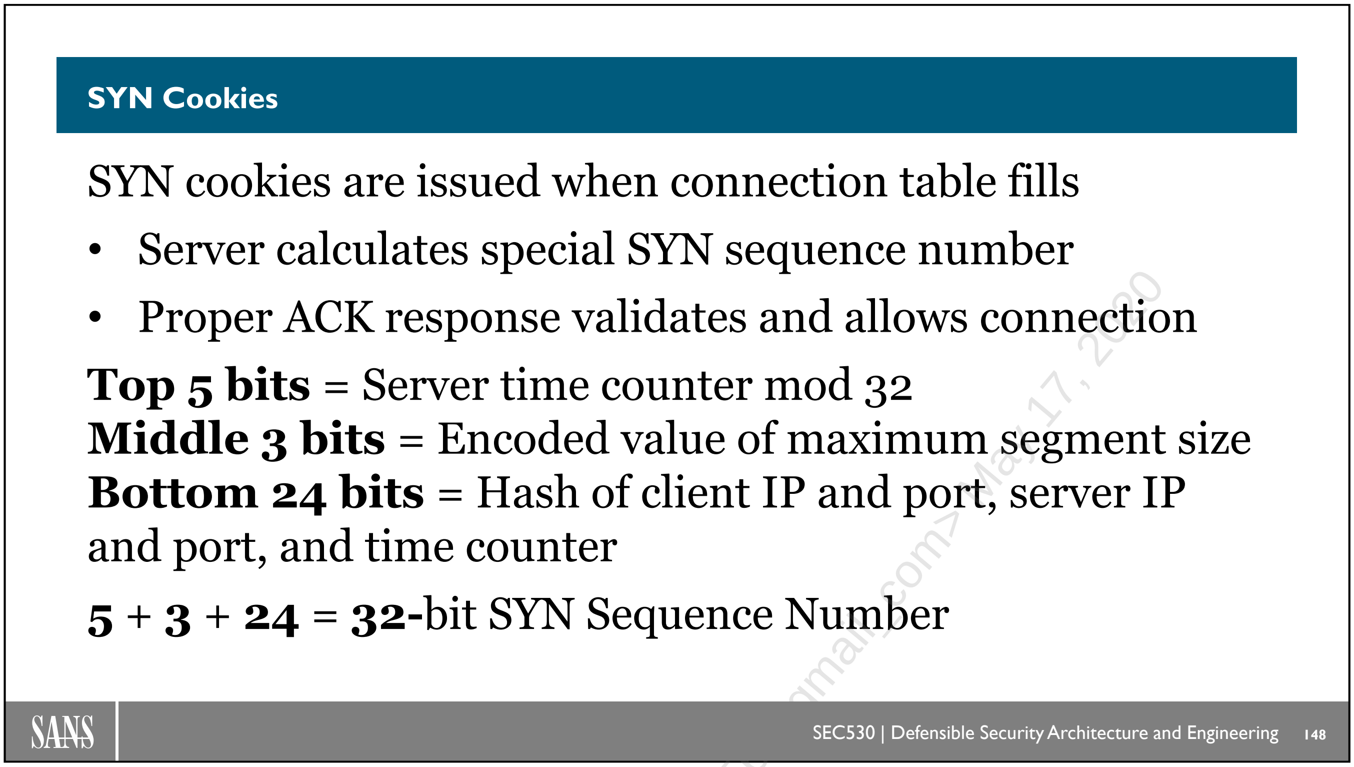
Examples on tuning Linux:

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 2048 > /proc/sys/net/ipv4/tcp_max_syn_backlog
echo 3 > /proc/sys/net/ipv4/tcp_synack_retries
```

Then add the settings to **/etc/sysctl.conf** to make permanent:

```
net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_max_syn_backlog = 2048
net.ipv4.tcp_synack_retries = 3
```

[1] https://www.ndchost.com/wiki/server-administration/hardening-tcpip-syn-flood



SYN Cookies

SYN cookies are issued when connection table fills

- Server calculates special SYN sequence number
- Proper ACK response validates and allows connection

Top 5 bits = Server time counter mod 32
Middle 3 bits = Encoded value of maximum segment size
Bottom 24 bits = Hash of client IP and port, server IP and port, and time counter

5 + **3** + **24** = **32-**bit SYN Sequence Number

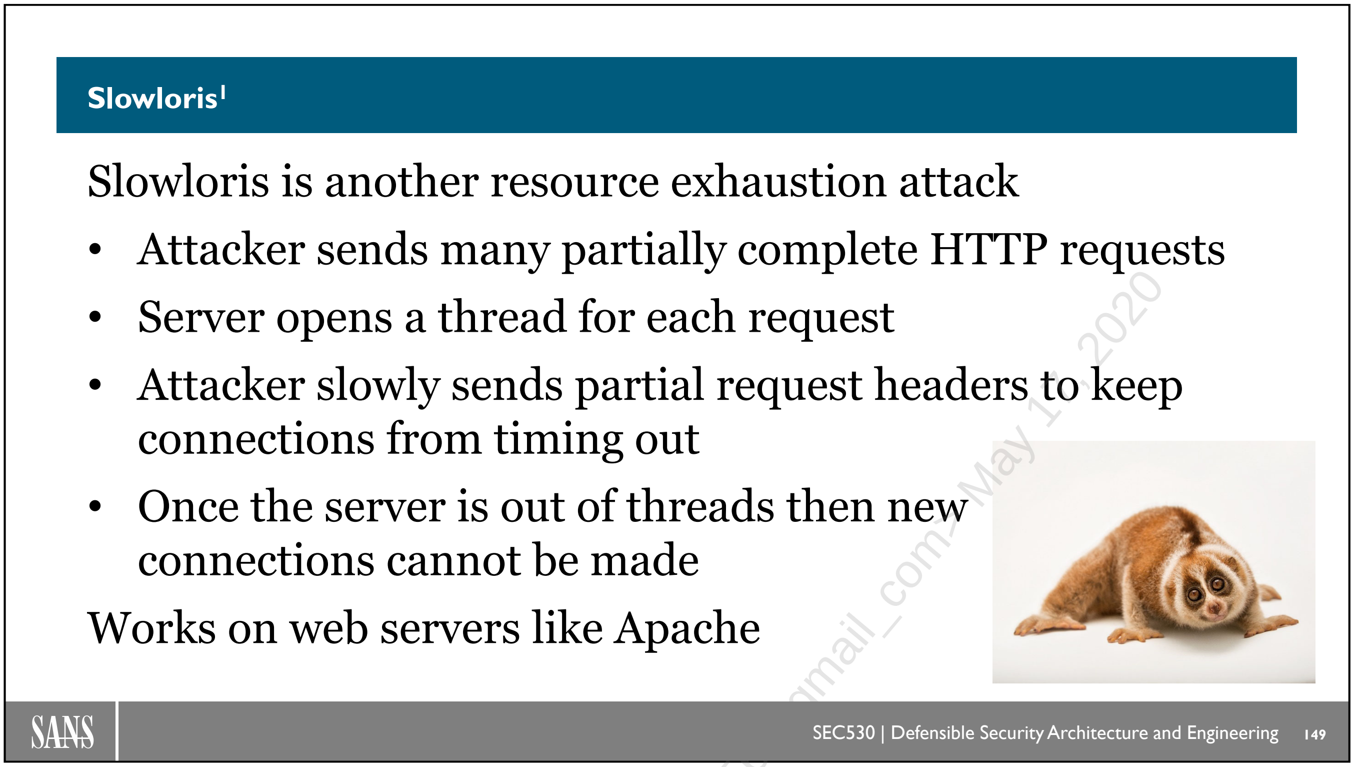
SYN Cookies

A different method for protecting against SYN flooding is using an SYN cookie. An SYN cookie does nothing until a system's connection table is full. Once full, the server begins to respond to SYN packets with an SYN/ACK that has an SYN sequence number that is a special cryptographic hash. If a system receives the server's SYN/ACK and sends back an SYN/ACK where the ACK is the server's sequence number plus one, then the server knows the request came from a legitimate connection and the connection is allowed. The connection works even though the normal session table is full.

While it is possible for an attacker to guess the special SYN sequence number to spoof replies it would likely take millions or billions of packets to guess correctly. However, the point of SYN cookies is not to keep an attacker from connecting but to keep an attacker from performing a denial of service that prevents legitimate connections. Because SYN cookies do not use the TCP session table, servers are able to reply to initial SYN packets with their own SYN/ACK packet with the special cryptographic hash. If a correct response is received, then the connection is allowed. Thus, an SYN flood-based denial-of-service attack would not be successful.

A summary of how the SYN cookie is calculated is represented in the slide. The output will always be 32-bits to represent an SYN sequence number. Also, the sequence number follows RFC compliance for TCP.

[1] https://cr.yp.to/syncookies.html



Slowloris[1]

Slowloris is another resource exhaustion attack

- Attacker sends many partially complete HTTP requests
- Server opens a thread for each request
- Attacker slowly sends partial request headers to keep connections from timing out
- Once the server is out of threads then new connections cannot be made

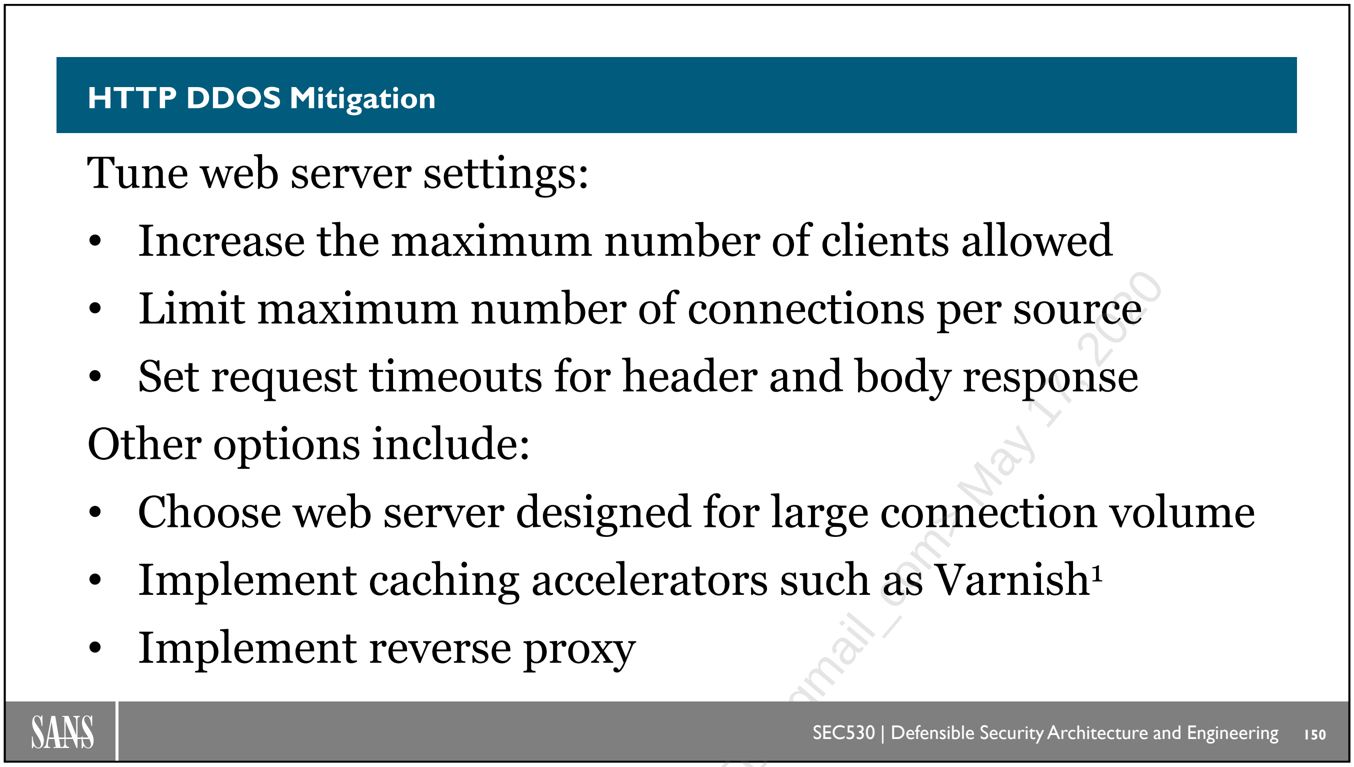
Works on web servers like Apache

Slowloris[1]

Another example of protocol exhaustion is an attack called Slowloris. The Slowloris attack abuses HTTP by creating many HTTP requests and sending small header requests to keep the connections open until a web server can no longer accept new connections. This attack is old but still works on web servers today.

The process for pinning up multiple threads on a web server occurs by making many connections and keeping them alive. An example of this would be making a thousand connections to a web server. Then every couple of seconds, say five seconds, performing another web request with the current thousand connections. However, every one second, a new set of a thousand connections is made. Basically, a thousand sockets are established per second and then every five seconds per socket an action is taken to keep it going. Ultimately, the increasing rate of web connections on a web server can cause it to run out of resources.

[1] https://www.corero.com/resources/ddos-attack-types/slowloris-attack.html



HTTP DDOS Mitigation

Tune web server settings:

- Increase the maximum number of clients allowed
- Limit maximum number of connections per source
- Set request timeouts for header and body response

Other options include:

- Choose web server designed for large connection volume
- Implement caching accelerators such as Varnish[1]
- Implement reverse proxy

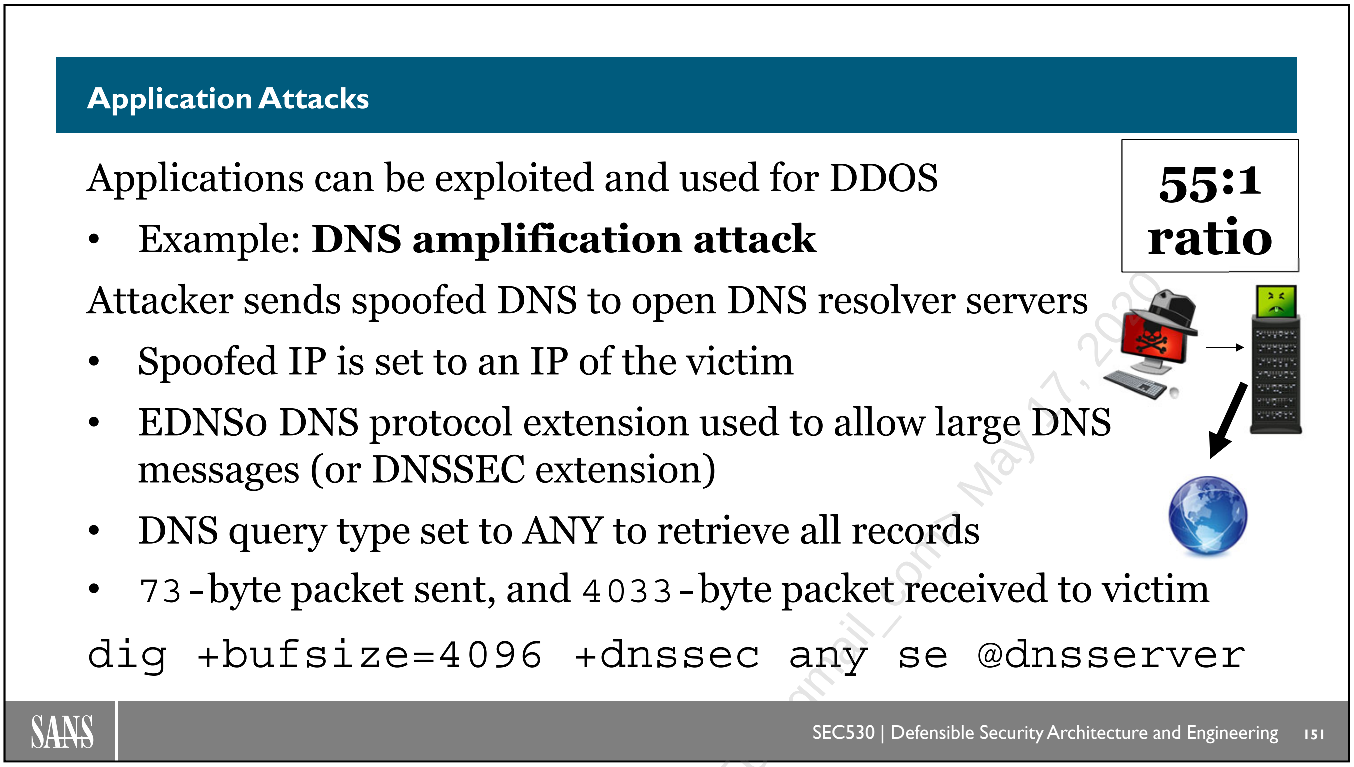
SANS

SEC530 | Defensible Security Architecture and Engineering 150

HTTP DDOS Mitigation

Mitigation techniques for HTTP DDOS attacks are similar to SYN flooding protection. For example, increasing the maximum number of connections can help a web server hold up against an attack. Other solutions include using a web server designed to handle many connections like Nginx or installing a caching accelerator like Varnish. One of the stronger defenses would be to implement a purpose-built reverse proxy such as a web application firewall.

[1] https://varnish-cache.org/



Application Attacks

Applications can be exploited and used for DDOS

- Example: **DNS amplification attack**

Attacker sends spoofed DNS to open DNS resolver servers

- Spoofed IP is set to an IP of the victim
- EDNS0 DNS protocol extension used to allow large DNS messages (or DNSSEC extension)
- DNS query type set to ANY to retrieve all records
- `73`-byte packet sent, and `4033`-byte packet received to victim

```
dig +bufsize=4096 +dnssec any se @dnsserver
```

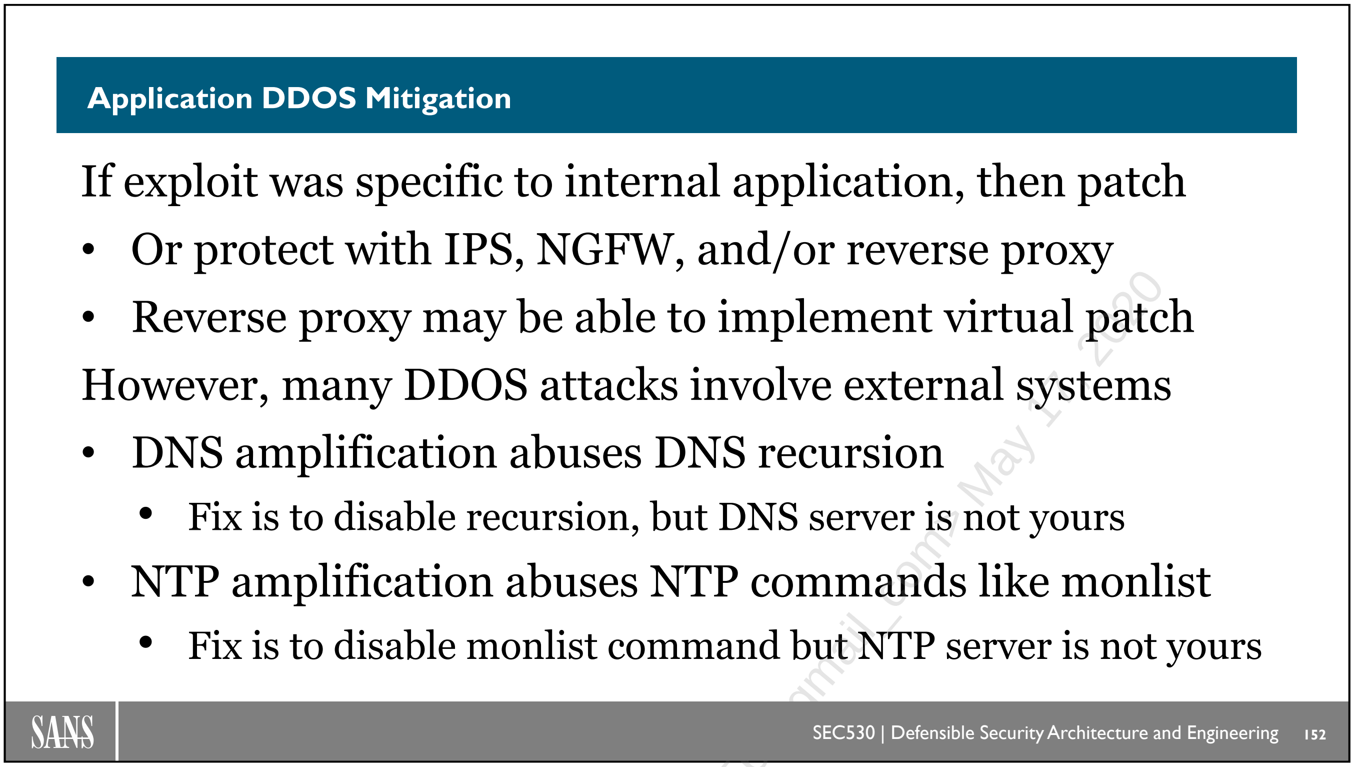
55:1 ratio

SEC530 | Defensible Security Architecture and Engineering 151

Application Attacks

Application-based DDOS attacks exploit or abuse applications. This type of attack ranges from small amounts of packets that exploit a flaw and crashes a service directly to techniques like amplification attacks. DNS amplification uses publicly available DNS servers that allow external recursion requests to spoof a DNS request. The source IP is spoofed to that of the victim, and the request is a small DNS query that is expected to have a large response. Amplification represents taking a small request and turning it into a large response. The example dig command when run in a lab created a 73-byte request and received a 4,033-byte response. If this was a spoofed DNS attack, it would have an amplification ratio of about 55 to 1.

[1] https://dnscurve.org/amplification.html



Application DDOS Mitigation

If exploit was specific to internal application, then patch

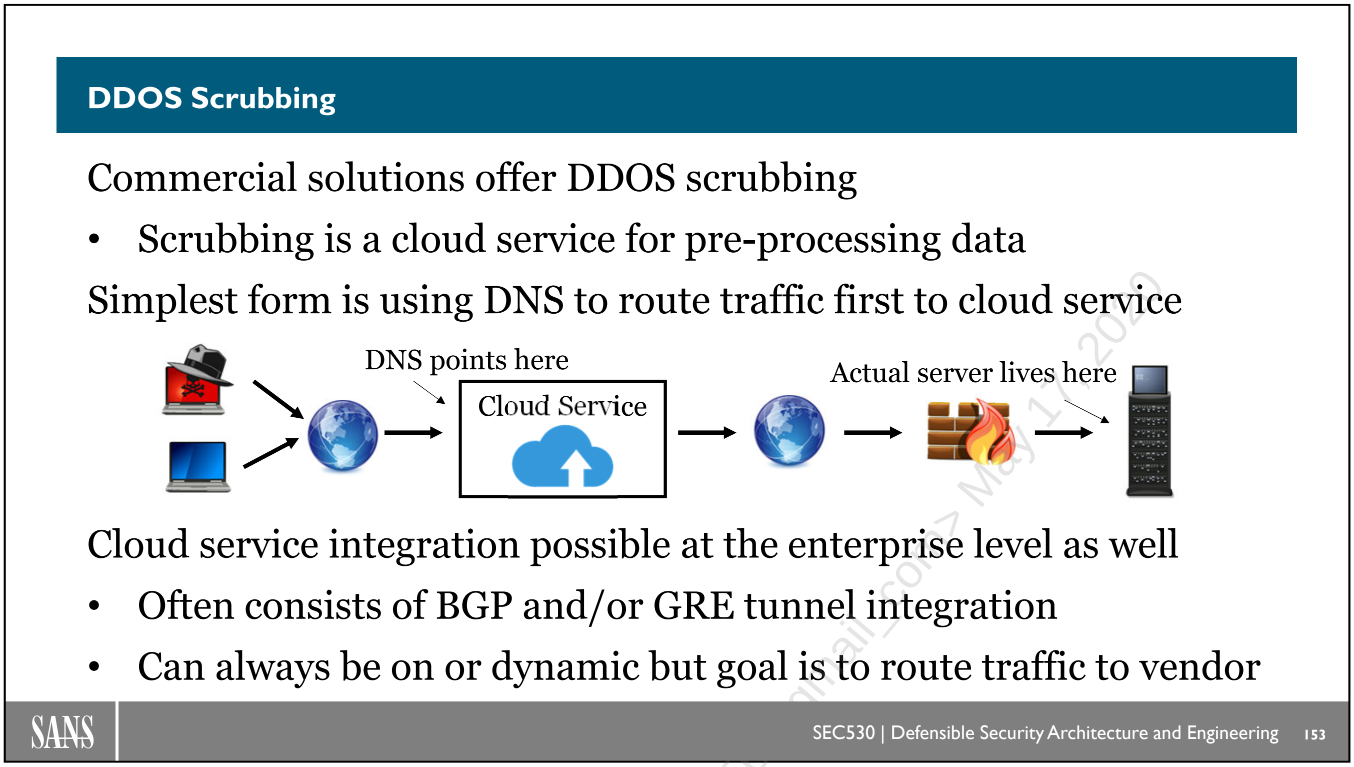
- Or protect with IPS, NGFW, and/or reverse proxy
- Reverse proxy may be able to implement virtual patch

However, many DDOS attacks involve external systems

- DNS amplification abuses DNS recursion
  - Fix is to disable recursion, but DNS server is not yours
- NTP amplification abuses NTP commands like monlist
  - Fix is to disable monlist command but NTP server is not yours

Application DDOS Mitigation

Mitigating application DDOS attacks is near impossible. If the application being attacked is within your control, then patches or security devices can be put in place. If the application is not in your control, which is highly probable, that nothing can be done. For example, DNS and NTP amplification attacks both use servers not owned by the victim organization. Therefore, nothing can be done directly to mitigate them. In the case of amplification attacks, your best bet may be to use cloud scrubbing or have enough bandwidth to sustain the attack.



DDOS Scrubbing

Commercial solutions offer DDOS scrubbing

- Scrubbing is a cloud service for pre-processing data

Simplest form is using DNS to route traffic first to cloud service

DNS points here

Cloud Service

Actual server lives here

Cloud service integration possible at the enterprise level as well

- Often consists of BGP and/or GRE tunnel integration
- Can always be on or dynamic but goal is to route traffic to vendor

DDOS Scrubbing

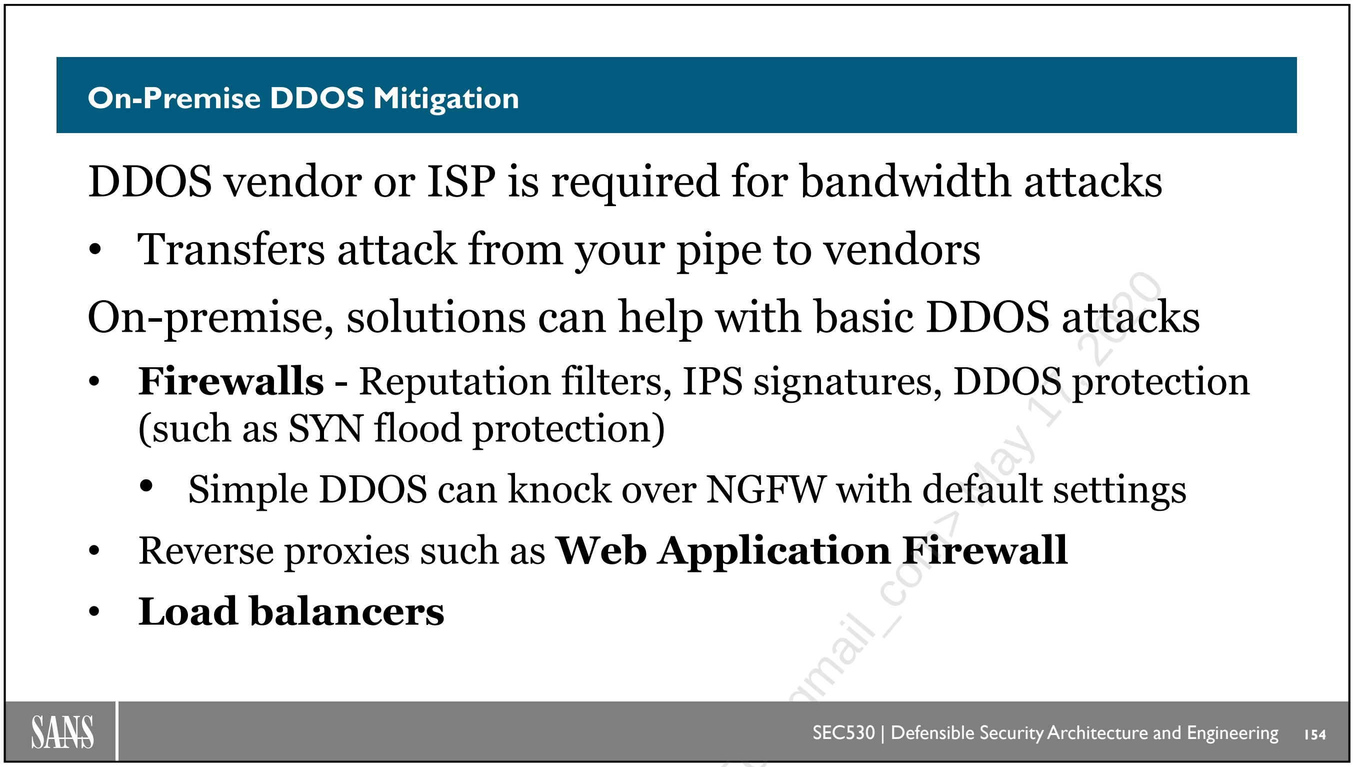
Most security controls can be purchased as either cloud or on-premise solutions. However, when it comes to DDOS attacks, a major issue is bandwidth. In order to handle DDOS attacks that saturate bandwidth the attack needs to be mitigated before hitting an organizations internet pipe. One of the best ways to do this is using cloud-based DDOS protection services. These services are sometimes referred to as DDOS scrubbing centers as traffic is cleaned or removed before being sent to an organization's systems.

A simple example of this would be setting the DNS record to a DDOS solution. The DDOS solution would receive all requests and then hand the legitimate ones to the real service it is protecting. The danger with DNS based protection is if the adversary finds the public servers real IP address than the adversary can still DDOS the server. DDOS site protection can be purchased for around $20 a month per site.

More advanced solutions can provide similar DDOS protection for an entire organization. These solutions usually involve BGP and GRE routing or some other form of routing traffic first through the DDOS scrubbing solution.

[1] https://devcentral.f5.com/articles/scrubbing-away-ddos-attacks

[2] https://www.incapsula.com/ddos/ddos-mitigation-services.html



On-Premise DDOS Mitigation

DDOS vendor or ISP is required for bandwidth attacks

- Transfers attack from your pipe to vendors

On-premise, solutions can help with basic DDOS attacks

- **Firewalls** - Reputation filters, IPS signatures, DDOS protection (such as SYN flood protection)
  - Simple DDOS can knock over NGFW with default settings
- Reverse proxies such as **Web Application Firewall**
- **Load balancers**

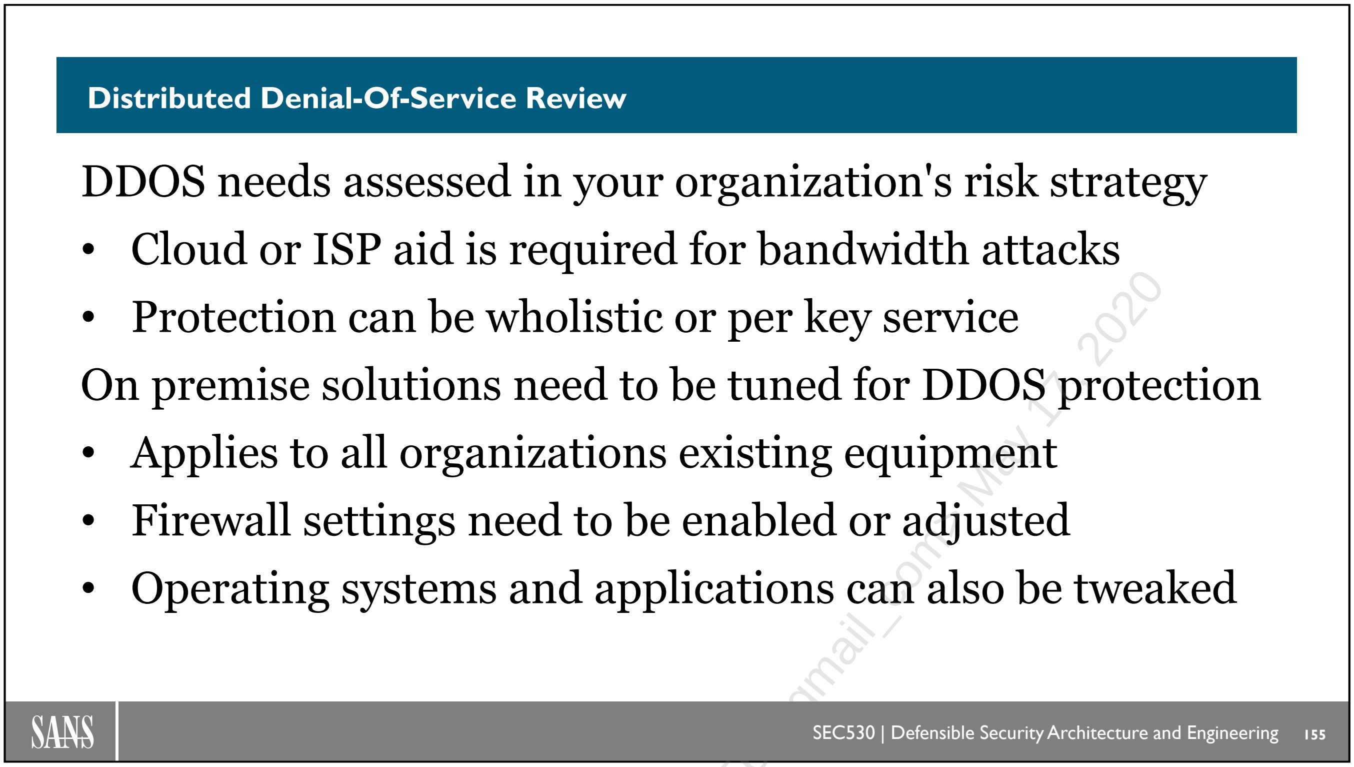
SANS

SEC530 | Defensible Security Architecture and Engineering 154

On-Premise DDOS Mitigation

DDOS attacks are best handled by dedicated solutions like DDOS scrubbing centers or ISPs. But internal systems should be more resilient against basic DDOS attacks. On-premise solutions such as NGFWs, web application firewalls, and load balancers can be tuned to properly handle DDOS attacks such as protocol or application DDOS attacks.

By default, many on-premise solutions do not have DDOS protection enabled. For example, most NGFWs have DDOS protection capabilities, but those capabilities are either disabled by default or poorly tuned. In fact, most NGFWs can be taken offline by extremely small and simple to perform DDOS attacks. A single workstation could potentially crash a firewall. For example, a workstation could successfully deploy attacks like SYN floods or other TCP exhaustion attacks. Considering the fact that organizations typically do not do DDOS tests, they may be unaware their true risk level is actually higher than they estimated, and beyond their own acceptable levels.



Distributed Denial-Of-Service Review

DDOS needs assessed in your organization's risk strategy

- Cloud or ISP aid is required for bandwidth attacks
- Protection can be wholistic or per key service

On premise solutions need to be tuned for DDOS protection

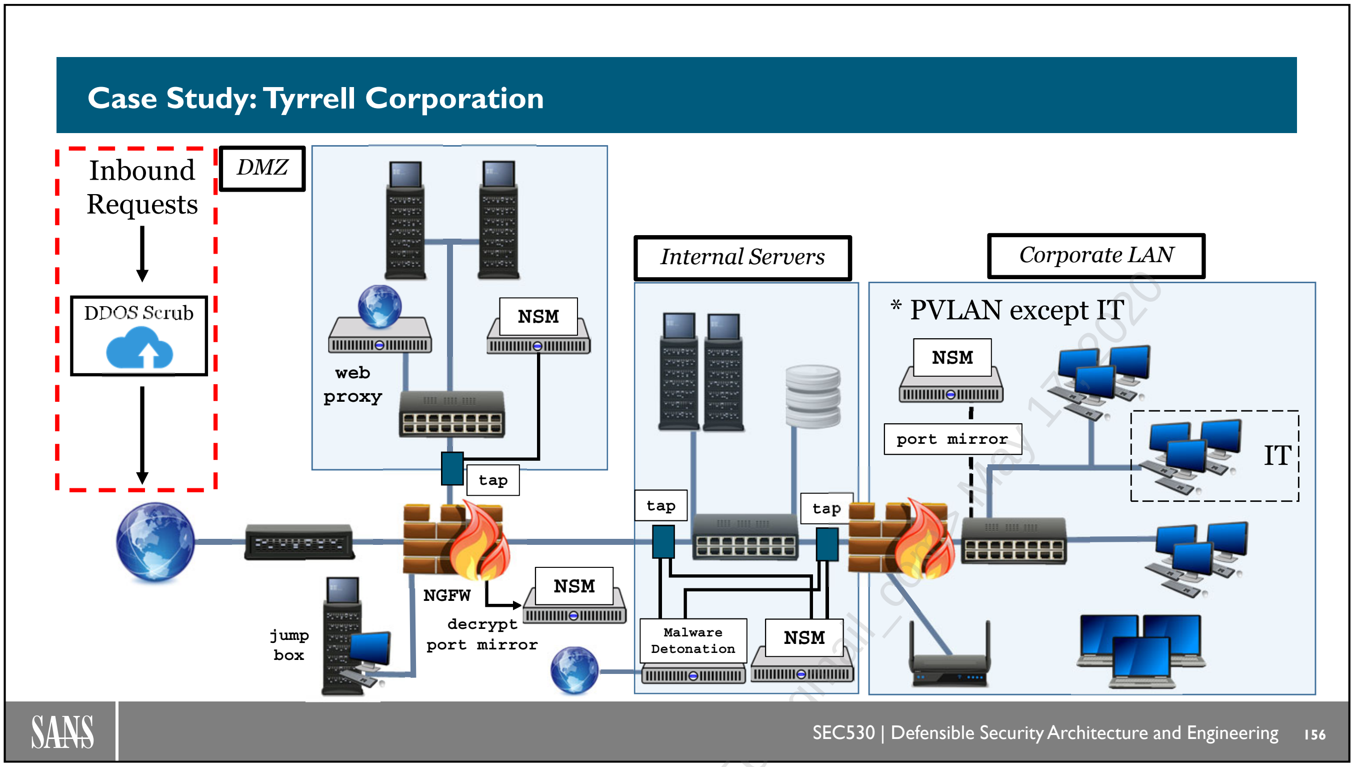
- Applies to all organizations existing equipment
- Firewall settings need to be enabled or adjusted
- Operating systems and applications can also be tweaked

SANS

SEC530 | Defensible Security Architecture and Engineering 155

Distributed Denial-Of-Service Review

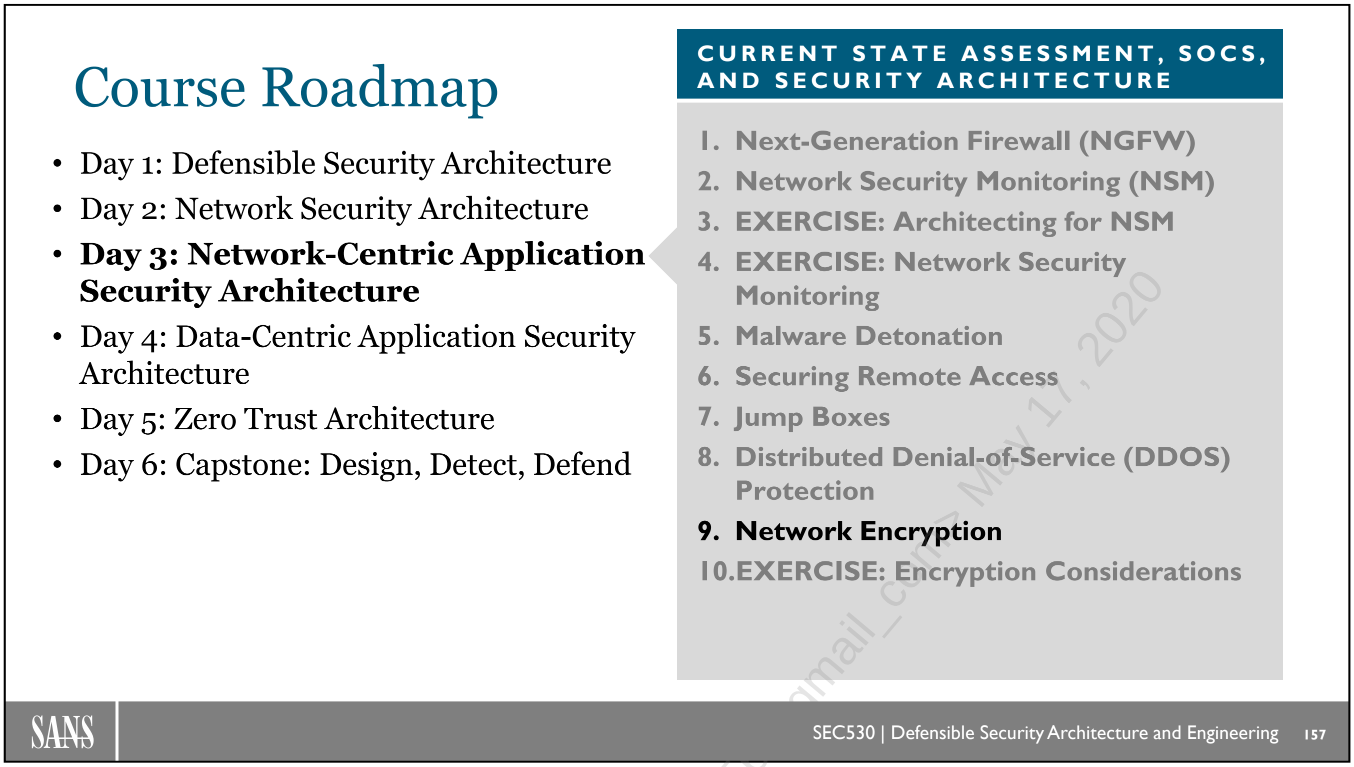
All organizations need to assess the risk of a DDOS attack. Depending on the risk tolerance level organizations can simply tune their existing solutions or implement additional controls such as cloud-based DDOS scrubbing.



Case Study: Tyrrell Corporation

Case Study: Tyrrell Corporation

This diagram shows the addition of DDOS protection to the Tyrrell Corporation architecture. In this example, requests to inbound services such as web services in the DMZ are routed through a DDOS scrubbing cloud service.



Course Roadmap

- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

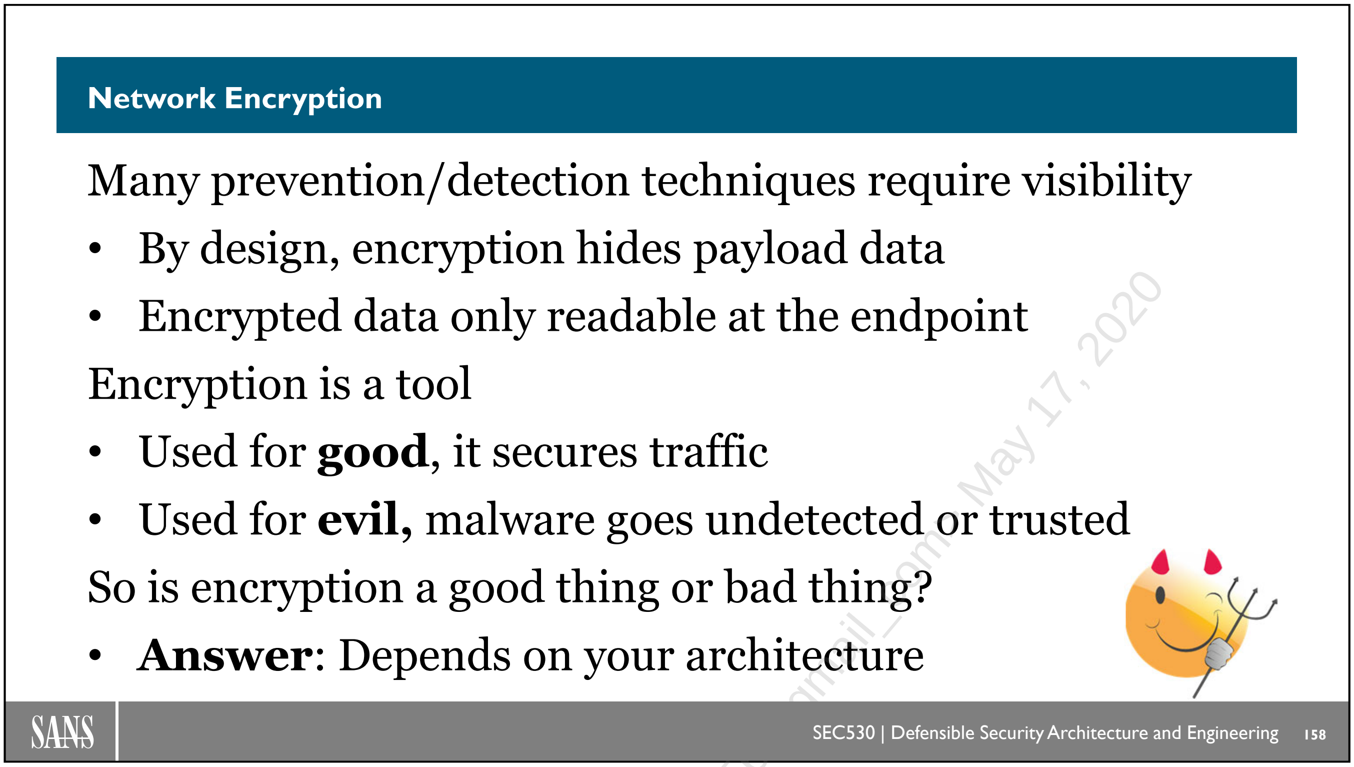
CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. **Network Encryption**
10. EXERCISE: Encryption Considerations

SANS

Course Roadmap

The next section covers Network Encryption.



Network Encryption

Many prevention/detection techniques require visibility

- By design, encryption hides payload data
- Encrypted data only readable at the endpoint

Encryption is a tool

- Used for **good**, it secures traffic
- Used for **evil,** malware goes undetected or trusted

So is encryption a good thing or bad thing?

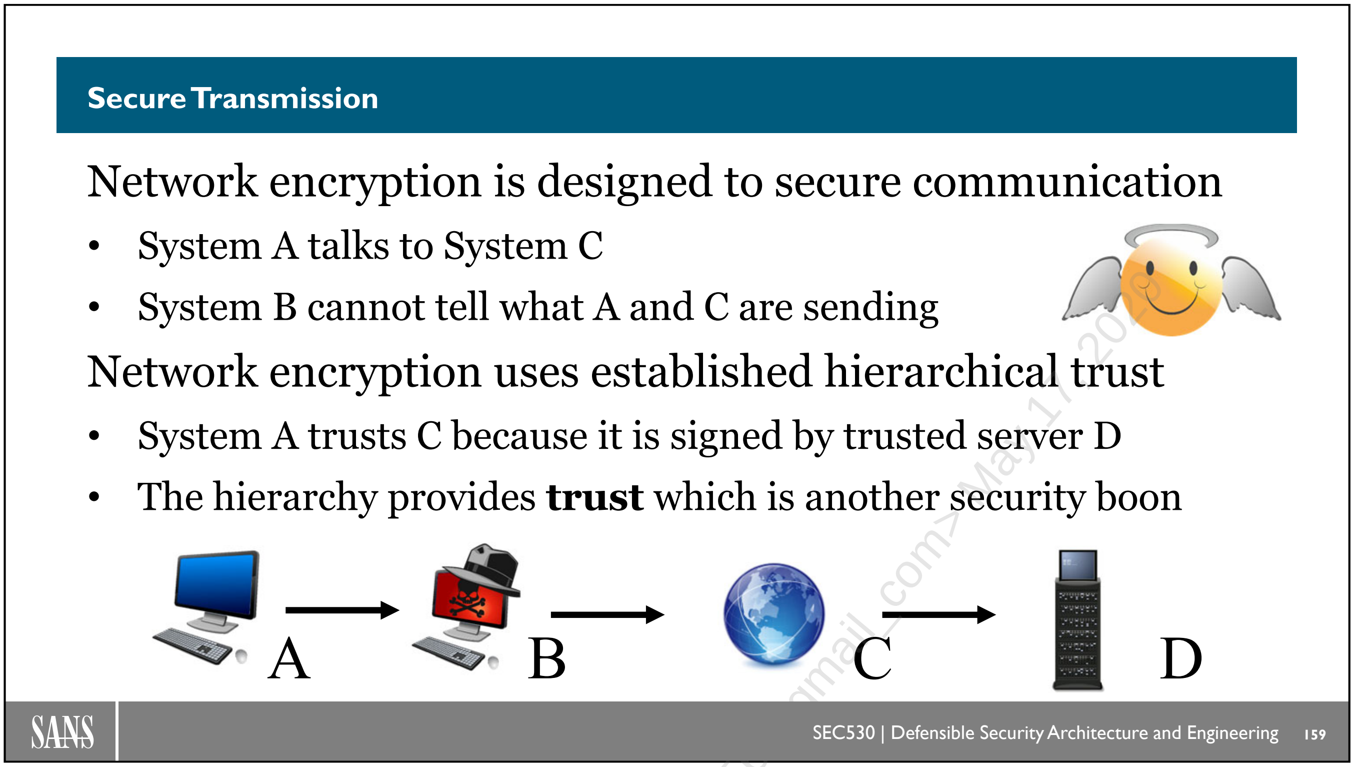
- **Answer**: Depends on your architecture

SEC530 | Defensible Security Architecture and Engineering 158

Network Encryption

Many of the previous solutions work primarily using layer four through seven inspection. However, even layer seven inspection can also fall short. Signatures look for specific things of interest. The problem is evasion tools are prevalent and easy to use making signature matching flawed. Given enough time it is possible to evade any signature-based tool.

All-in-all this represents the normal attack/defense cycle where attackers adapt to new tricks to bypass defenses and defenders then adapt their abilities to prevent or detect the new attack techniques. This cycle is why defense in depth is so critical.



Secure Transmission

Network encryption is designed to secure communication

- System A talks to System C
- System B cannot tell what A and C are sending

Network encryption uses established hierarchical trust

- System A trusts C because it is signed by trusted server D
- The hierarchy provides **trust** which is another security boon

SANS

SEC530 | Defensible Security Architecture and Engineering 159

Secure Transmission

Encryption brings two major features in regard to transporting data securely over an untrusted network. First, network encryption keeps unauthorized parties from reading data or modifying data. Secondly, network encryption provides non-repudiation and trust. Specifically, network encryption uses a trust to verify a system is an expected system. Trust is implemented using public key infrastructure.

For example, when a client accesses a website protected with TLS the client will verify the certificate issued to the web server is issued from a trusted certificate authority. If the certificate is from a valid trusted certificate authority, then the site is considered the expected and trusted site. If an adversary attempts to redirect a system to a different web server, the certificate will not match and may be from an untrusted certificate authority.

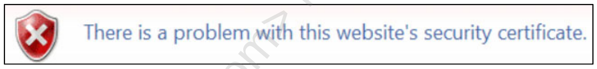
Historically, certificates from attackers or malware were unlikely to be trusted. However, SSL certificates can be obtained for free and be trusted by all modern operating systems and browsers.

Certificate Validation

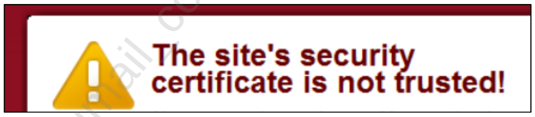
Operating systems and software *should* validate certificates

- Is the certificate expired?
- Is the certificate digitally signed by a trusted authority?
- Process handled by certificate revocation list (CRL) or online certificate status protocol (OCSP)

Adversary unlikely to generate a trusted certificate for another organization



There is a problem with this website's security certificate.



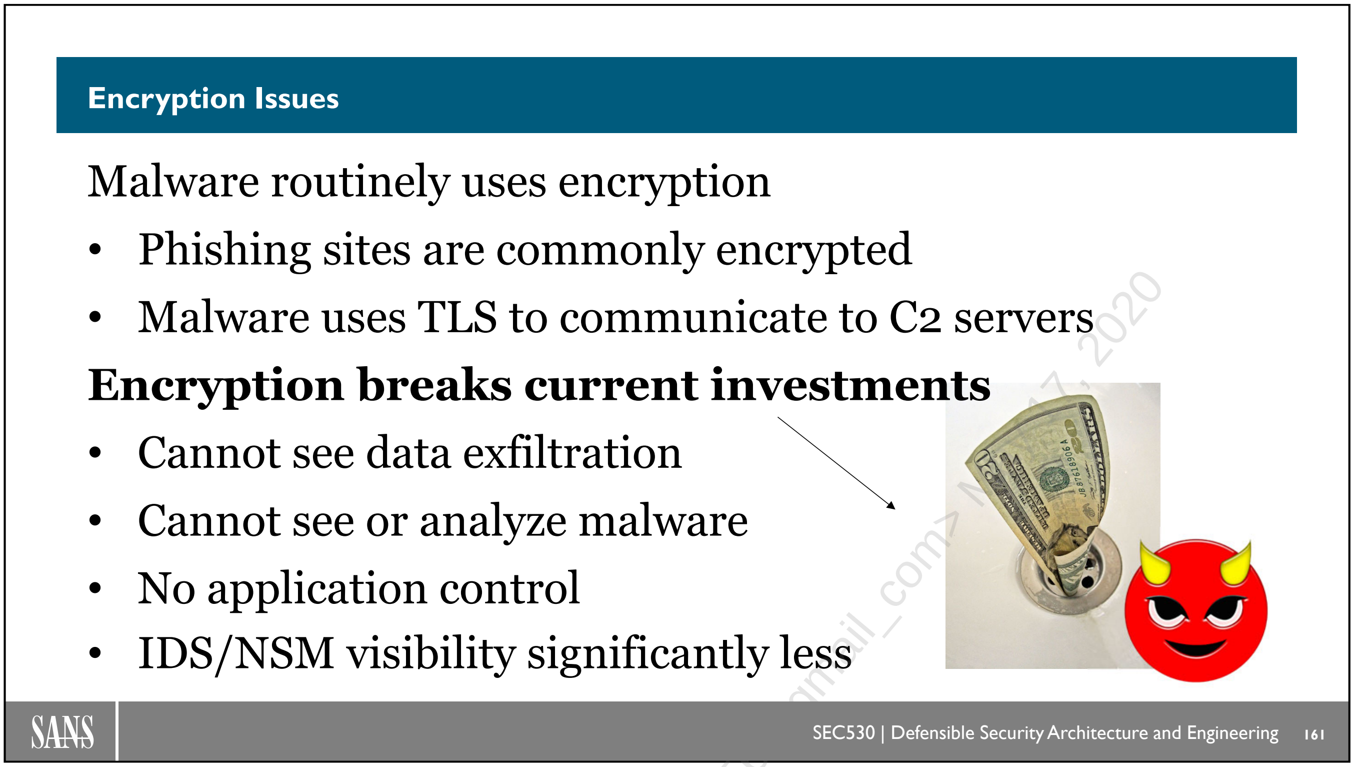
The site's security certificate is not trusted!

Certificate Validation

Certificates have expiration dates and are issued by a certificate authority. Client systems use both the expiration date and either certificate revocation lists or the online certificate status protocol to verify a certificate is valid. Certificates may be revoked by the issuing certificate authority for various reasons such as the certificate is no longer in use, the certificate has been compromised, or any other reason.

Because of certificate validation and how trust is established, creating or stealing a certificate of a legitimate business is unlikely. However, stealing and using certificates has happened in the past such as with Stuxnet[1].

[1] https://www.wired.com/2014/11/countdown-to-zero-day-stuxnet/



Encryption Issues

Malware routinely uses encryption

- Phishing sites are commonly encrypted
- Malware uses TLS to communicate to C2 servers

Encryption breaks current investments

- Cannot see data exfiltration
- Cannot see or analyze malware
- No application control
- IDS/NSM visibility significantly less

SANS

SEC530 | Defensible Security Architecture and Engineering 161

Encryption Issues

A major oversight of businesses today is the lack of consideration of the impact network encryption has in an environment. For example, both command and control communication and phishing websites routinely utilize encryption. Yet, organizations have technologies implemented to crack open the encrypted traffic or do a high-level analysis of the certificates or encryption algorithms in use. The issue is typically not lack of capabilities but rather lack of tuning for visibility.

Without visibility into encrypted data, attacks run rampant and existing security technologies are blind. IDS, NGFW, web proxies, malware detonation systems, and more are ineffective if encrypted traffic is observed without the capability to see what is present. At a minimum, certificates should be inspected to potentially find malicious use based on the domain name, certificate subject, certificate issuer, or algorithms an encrypted session is using. Certificate inspection is possible using Zeek with port mirroring or a tap as well as in most modern Next-Generation Firewalls.

Leveraging Encryption

Industry push is to encrypt everything

- Concept **increases risk** rather than reducing it
- Encryption should be deliberate and calculated

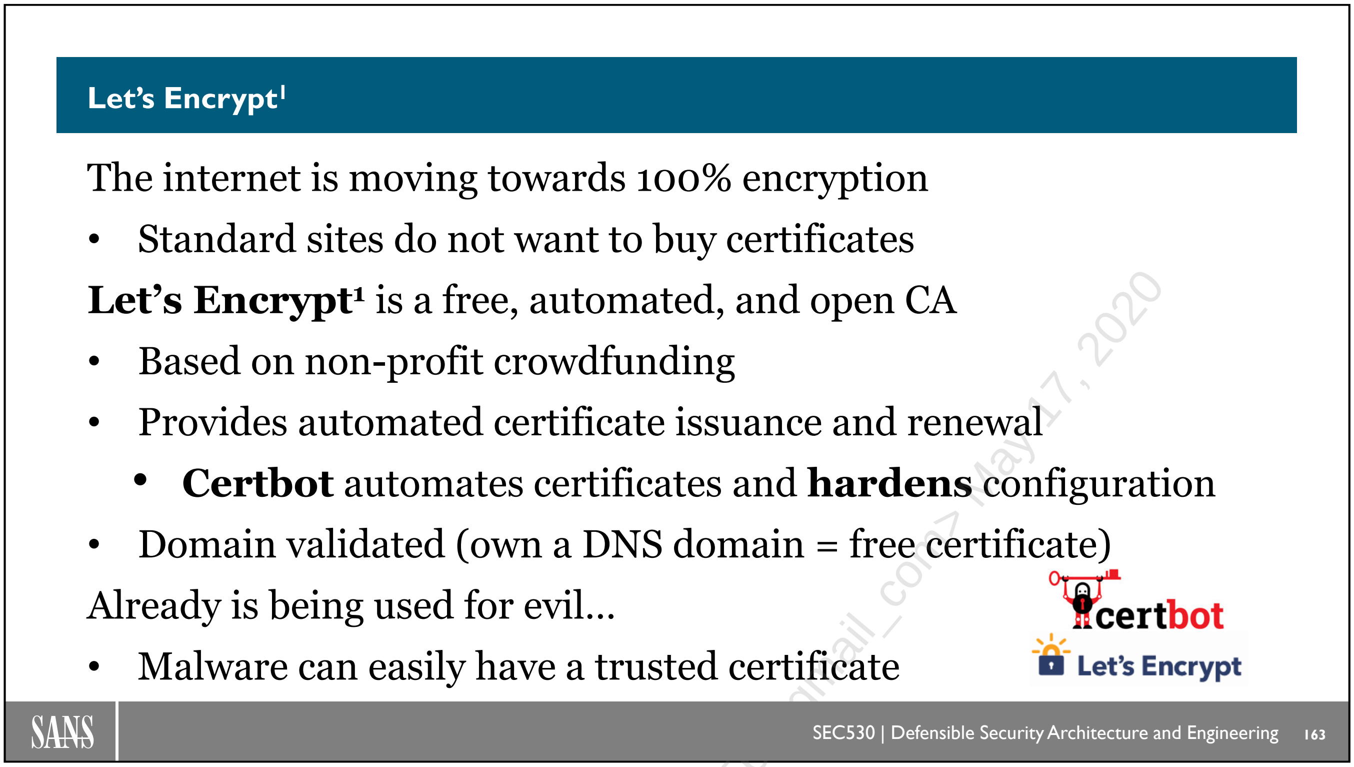
Encryption considerations:

- Is end-to-end encryption necessary?
- Is encryption implemented securely?
- Is inspection allowed?
- Is strong crypto support forced?

Leveraging Encryption

The push to encrypt all network traffic is continuing. However, web traffic has dramatically been shifting from HTTP traffic to HTTPS. Encryption is a good thing. However, too much encryption or encryption without thought or planning is not. The shift towards encrypting all things has in many cases allowed attacks to go undetected thus increasing the risk to organizations.

The encryption of increasing amounts of web traffic comes with tradeoffs. On the one hand, traffic cannot be intercepted and modified to insert malicious material, which is good for defenders. On the other hand, we can no longer monitor our networks if we are unable to understand the content of the traffic on it.



Let's Encrypt[1]

The internet is moving towards 100% encryption

- Standard sites do not want to buy certificates

Let's Encrypt[1] is a free, automated, and open CA

- Based on non-profit crowdfunding
- Provides automated certificate issuance and renewal
  - **Certbot** automates certificates and **hardens** configuration
- Domain validated (own a DNS domain = free certificate)

Already is being used for evil…

- Malware can easily have a trusted certificate

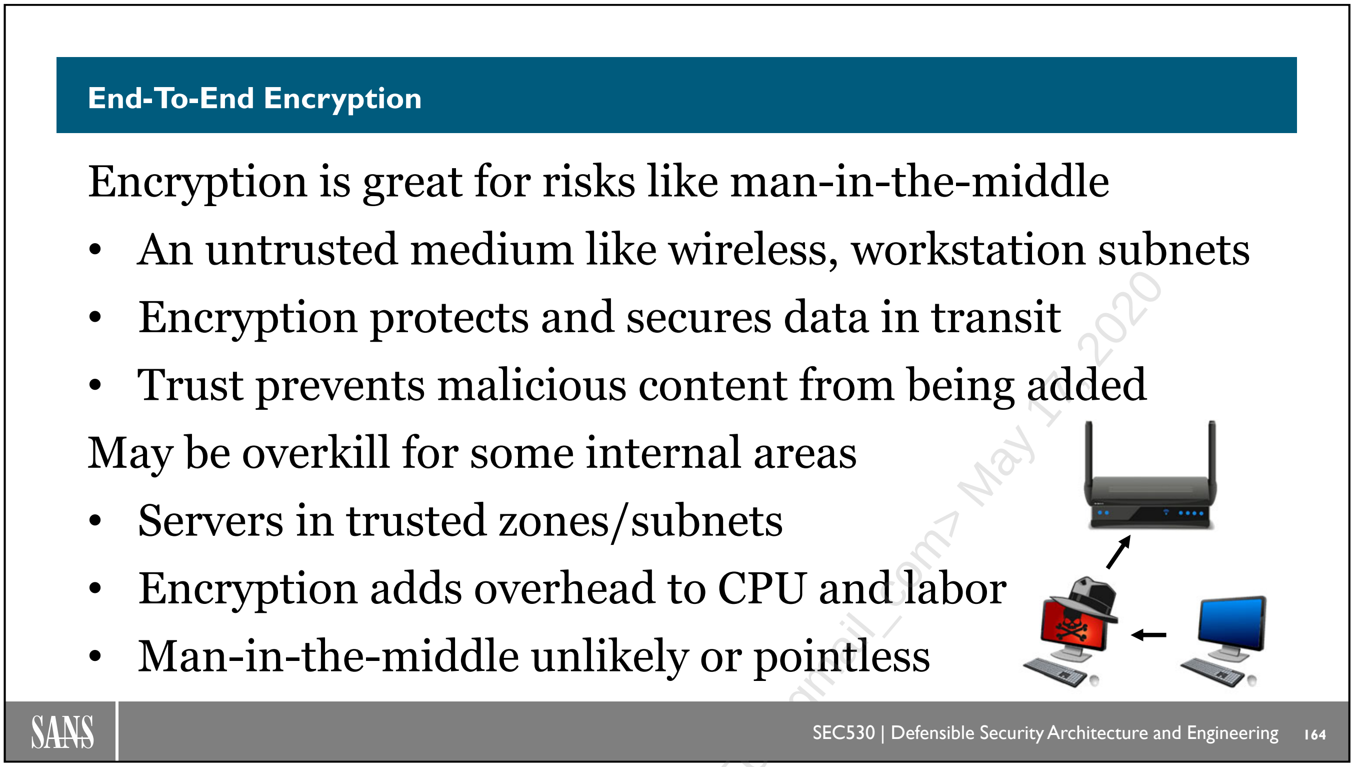
Let's Encrypt[1]

One of the major holdbacks in shifting to 100% website encryption is the cost of a certificate. Previously, these ran about $30 a site. Let's Encrypt[1] is an organization that was developed to eliminate this cost by providing free and automated certificates. It does this by allowing a client called cerbot to issue and renew certificates automatically. These certificates are valid for exactly 90 days.

Arguably, this increases security as it introduces encryption and helps mitigate man-in-the-middle attacks. However, it also decreases security as it allows malware and phishing campaigns to use trusted certificates. Because of the automation, there is a high likelihood that malware and adversaries will start using free certificates such as the ones through this company. In fact, there already have been multiple occurrences of this that you can read about online[2].

[1] https://letsencrypt.org/

[2] http://blog.trendmicro.com/trendlabs-security-intelligence/lets-encrypt-now-being-abused-by-malvertisers/



End-To-End Encryption

Encryption is great for risks like man-in-the-middle

- An untrusted medium like wireless, workstation subnets
- Encryption protects and secures data in transit
- Trust prevents malicious content from being added

May be overkill for some internal areas

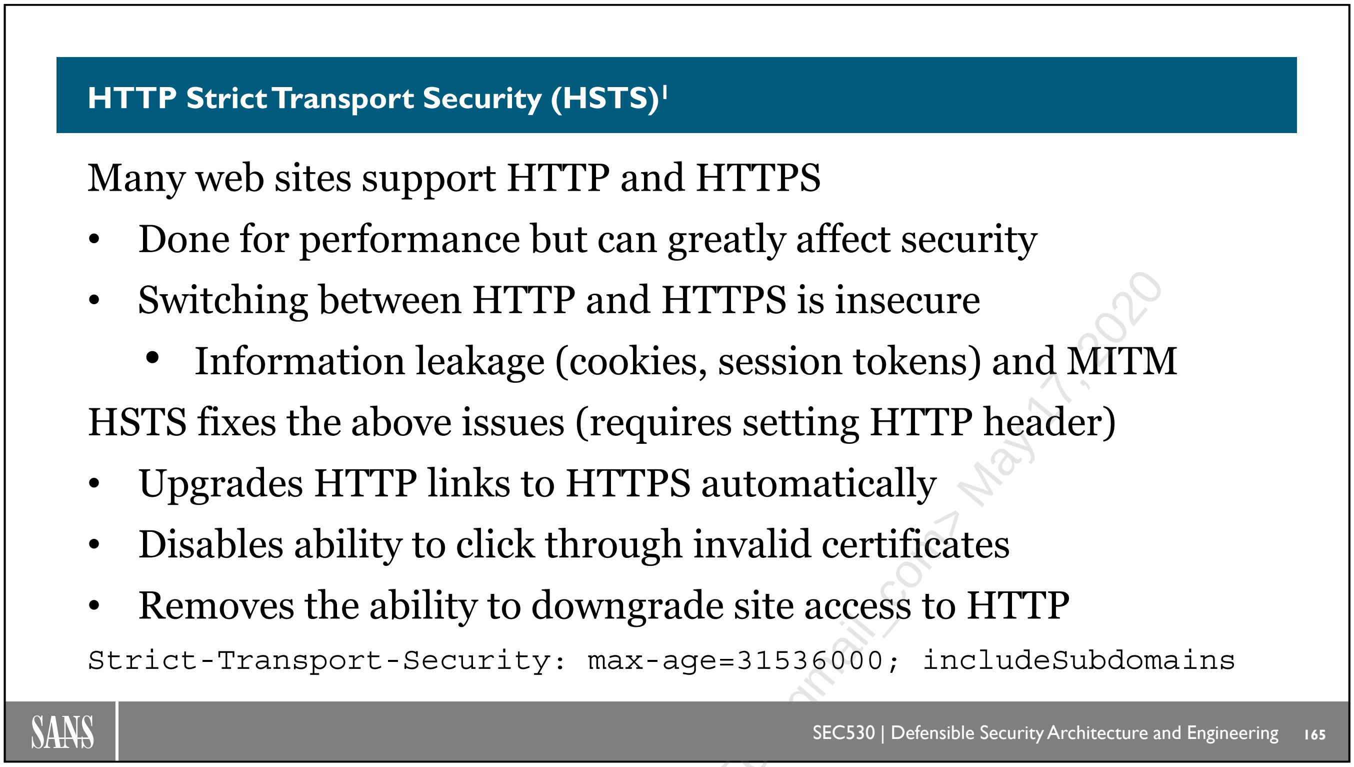
- Servers in trusted zones/subnets
- Encryption adds overhead to CPU and labor
- Man-in-the-middle unlikely or pointless

SEC530 | Defensible Security Architecture and Engineering 164

End-To-End Encryption

Encryption is highly effective at preventing man-in-the-middle attacks and protecting the integrity of traffic between two systems. However, implementing encryption is confusing and has multiple components that all need properly configured. PKI, or public key infrastructure, is an advanced technical skill. As a result, organizations may spend significant amounts of time implementing encryption, even in some cases where the risk it is addressing is low.

For example, two servers in the same layer-2 subnet or even in a layer-3 arrangement that talk to each other on a critical datacenter switch are not at high risk of man-in-the-middle. In this scenario, an attacker would have to compromise the data center switch or a server that has access to the network of one of the servers. If such actions were successfully performed the man-in-the-middle attack is likely unnecessary as the attacker already has significant access. And yet, organizations will spend hours or even days configuring the server to server encryption enforcement. The result is labor time lost, additional CPU overhead added, and a minor risk mitigated.



HTTP Strict Transport Security (HSTS)[1]

Many web sites support HTTP and HTTPS

- Done for performance but can greatly affect security
- Switching between HTTP and HTTPS is insecure
  - Information leakage (cookies, session tokens) and MITM

HSTS fixes the above issues (requires setting HTTP header)

- Upgrades HTTP links to HTTPS automatically
- Disables ability to click through invalid certificates
- Removes the ability to downgrade site access to HTTP

```
Strict-Transport-Security: max-age=31536000; includeSubdomains
```

HTTP Strict Transport Security (HSTS)[1]

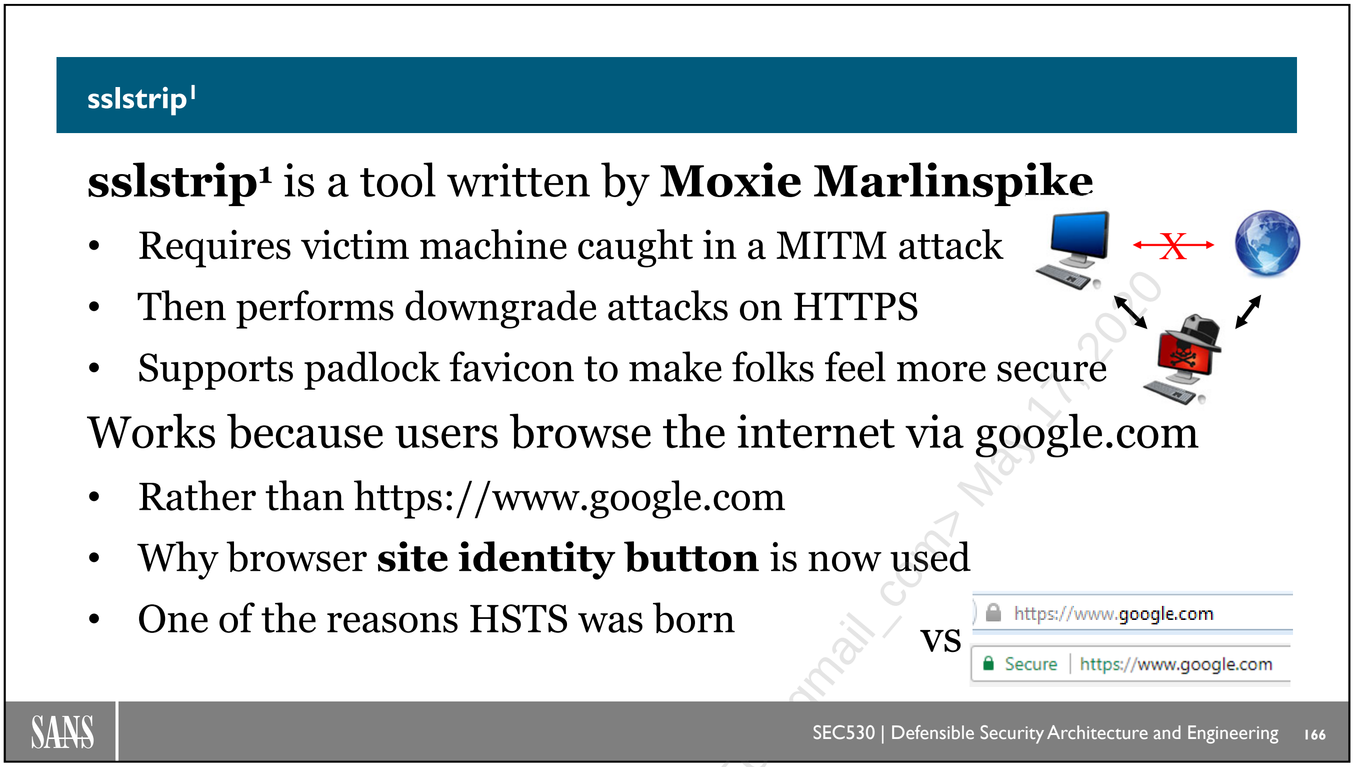
The internet still is composed of both HTTP and HTTPS sites. In fact, many sites still support both. The problem with sites utilizing both protocols is that adversaries can use HTTP to steal information or attempt to downgrade connections from HTTPS to HTTP to perform man-in-the-middle attacks. A common attack tool called sslstrip[2] is an example of downgrading a victim to HTTP to man-in-the-middle them and steal sensitive information.

Fortunately, web servers can be configured to use HTTP Strict Transport Security or HSTS. HSTS simply requires a web server to add a header to requests that enable HSTS. The example setting found on the slide enables HSTS for the site being accessed as well as all subdomains. The includeSubdomains is an optional parameter. The max-age of 31536000 notifies the client to remember the site uses HSTS for one year which is 31536000 seconds. The beauty and simplicity of HSTS is that it automatically upgrades all connection attempts to HTTP to HTTPS and removes the end user's capability of clicking through an invalid certificate. Thus, certificate validation is forced, and attackers are no longer able to downgrade HTTPS.

HSTS requires that an end user first accesses a site. Once the HSTS header is observed encryption is enforced for the max-age timeframe. The concern with HSTS is that an attacker may be able to exploit systems going to a website for the first time. While this does not happen frequently, first-time access is possible.

[1] https://https.app.cloud.gov/hsts/

[2] https://github.com/moxie0/sslstrip



sslstrip[1]

sslstrip[1] is a tool written by **Moxie Marlinspike**

- Requires victim machine caught in a MITM attack
- Then performs downgrade attacks on HTTPS
- Supports padlock favicon to make folks feel more secure

Works because users browse the internet via google.com

- Rather than https://www.google.com
- Why browser **site identity button** is now used
- One of the reasons HSTS was born

https://www.google.com

vs

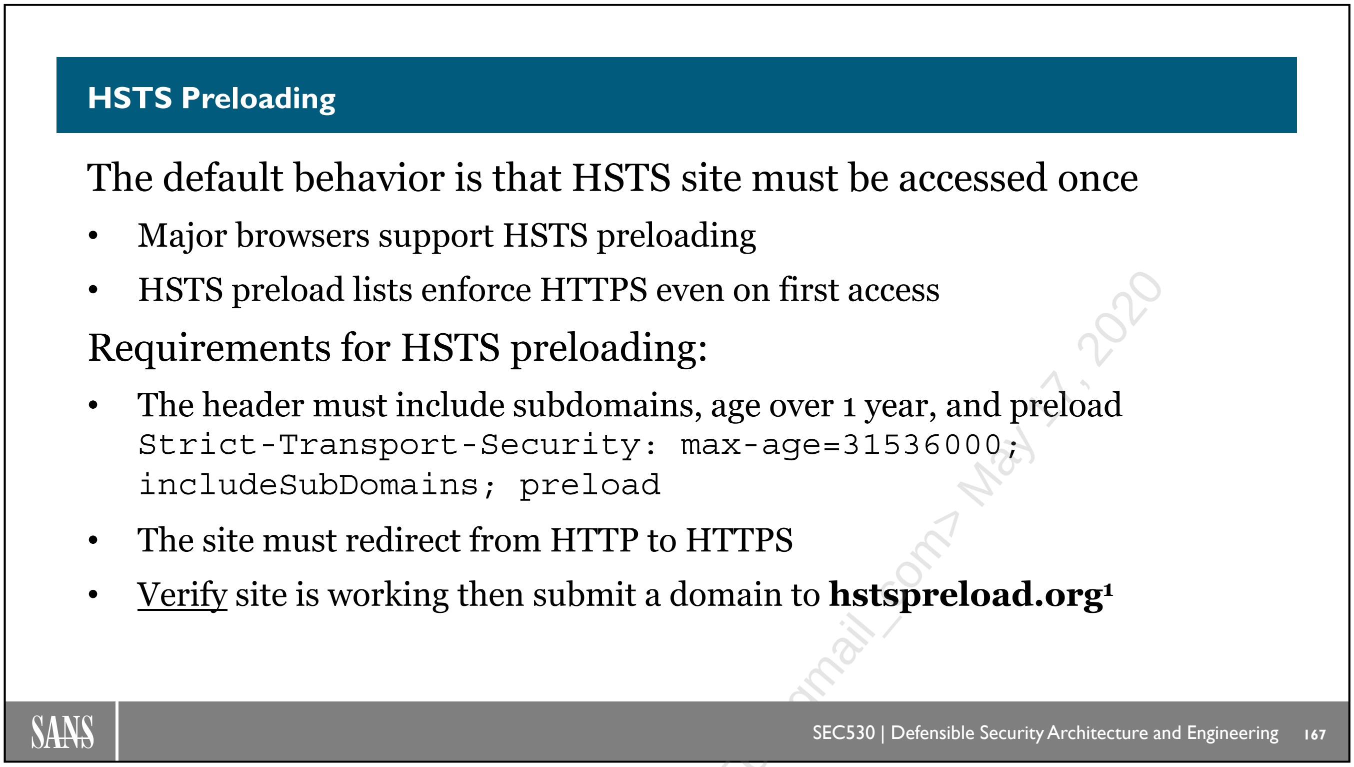
Secure | https://www.google.com

sslstrip[1]

A demonstration of why HSTS is so important can be achieved by using sslstrip. This tool requires a victim machine be controlled via man-in-the-middle such as ARP or DNS poisoning to work. Once a victim's traffic is being controlled sslstrip waits for a request such as examplebank.com, google.com, or facebook.com. The problem with the request is that it does not include https:// by default. Normally, an end user requests a site via a DNS name, and the web server redirects the user to the secured SSL/TLS site. However, sslstrip intercepts the request and initiates an SSL/TLS connection to the server but presents the victim's browser the same content but over HTTP.

Now all user input is captured as HTTP is not encrypted. To further entice the user, sslstrip can mask the favicon presented as a security lock icon. The lock tricks users into thinking the site uses HTTPS. Because of tools like sslstrip browsers like Google Chrome have changed from a favicon to a site identity button that more clearly identifies the use of HTTPS.

[1] https://github.com/moxie0/sslstrip



HSTS Preloading

The default behavior is that HSTS site must be accessed once

- Major browsers support HSTS preloading
- HSTS preload lists enforce HTTPS even on first access

Requirements for HSTS preloading:

- The header must include subdomains, age over 1 year, and preload
  ```
  Strict-Transport-Security: max-age=31536000;
  includeSubDomains; preload
  ```
- The site must redirect from HTTP to HTTPS
- Verify site is working then submit a domain to **hstspreload.org**[1]

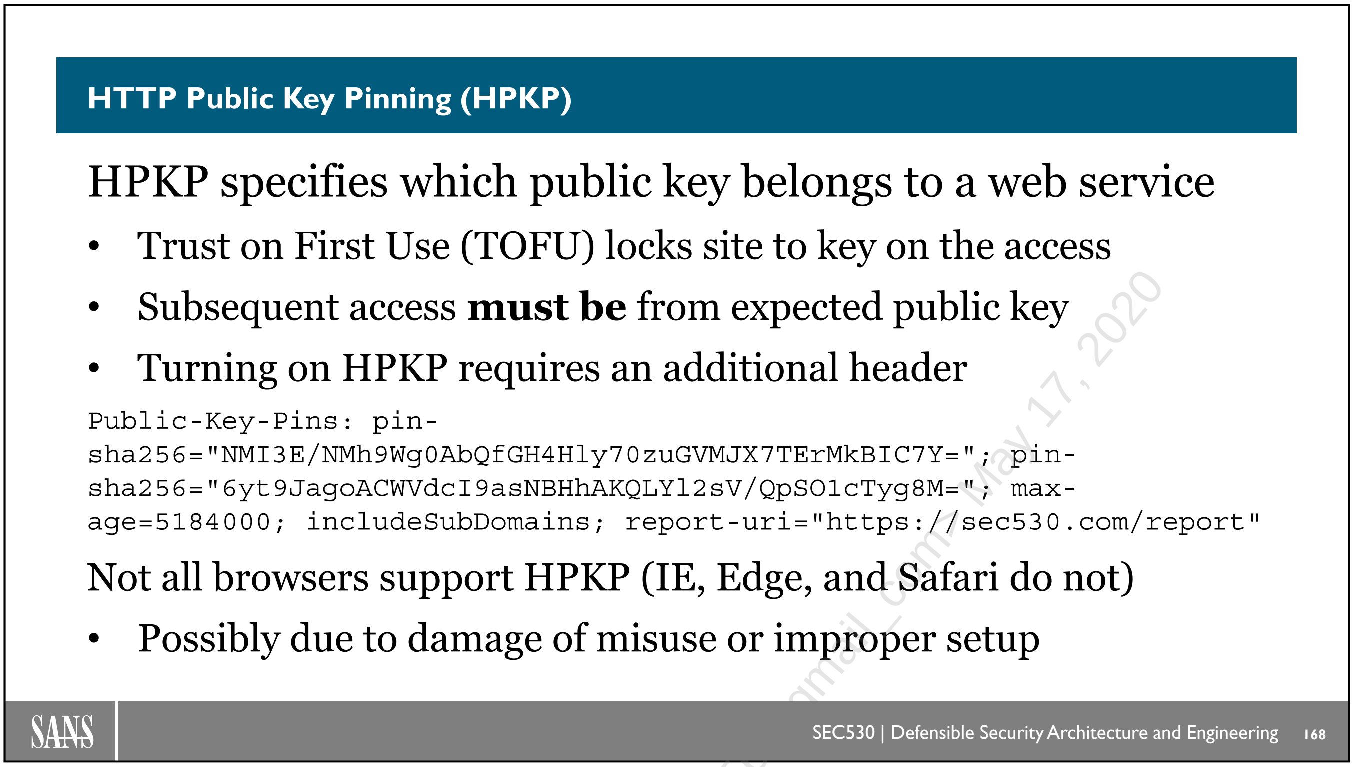
HSTS Preloading

HSTS is an incredibly strong protection mechanism. But the requirement that a server is accessed once before it takes affect greatly reduces HSTS's effectiveness. Fortunately, there is a way to get around the first-time access requirement for sites under an organization's control. The fix is to add the preload option to the end of the HSTS header. The preload option only works if includeSubDomains is also present.

The inclusion of includeSubDomains and preload makes a web server eligible to be added to the list of HSTS preload sites. The list is maintained by major organizations such as Google. Any domain within the preload list enforces HSTS regardless of first-time access. Effectively, HSTS is pre-enabled by preload. Sites with the preload option in their header may be automatically added to the HSTS preload list, but the best way to be added is to go to hstspreload.org and to submit the site manually. The manual submission also includes error checks to verify a server is properly set up for HSTS.

All major browsers submit HSTS preloading. Older browsers do not.

[1] https://hstspreload.org



HTTP Public Key Pinning (HPKP)

HPKP specifies which public key belongs to a web service

- Trust on First Use (TOFU) locks site to key on the access
- Subsequent access **must be** from expected public key
- Turning on HPKP requires an additional header

```
Public-Key-Pins: pin-
sha256="NMI3E/NMh9Wg0AbQfGH4Hly70zuGVMJX7TErMkBIC7Y="; pin-
sha256="6yt9JagoACWVdcI9asNBHhAKQLYl2sV/QpSO1cTyg8M="; max-
age=5184000; includeSubDomains; report-uri="https://sec530.com/report"
```

Not all browsers support HPKP (IE, Edge, and Safari do not)

- Possibly due to damage of misuse or improper setup

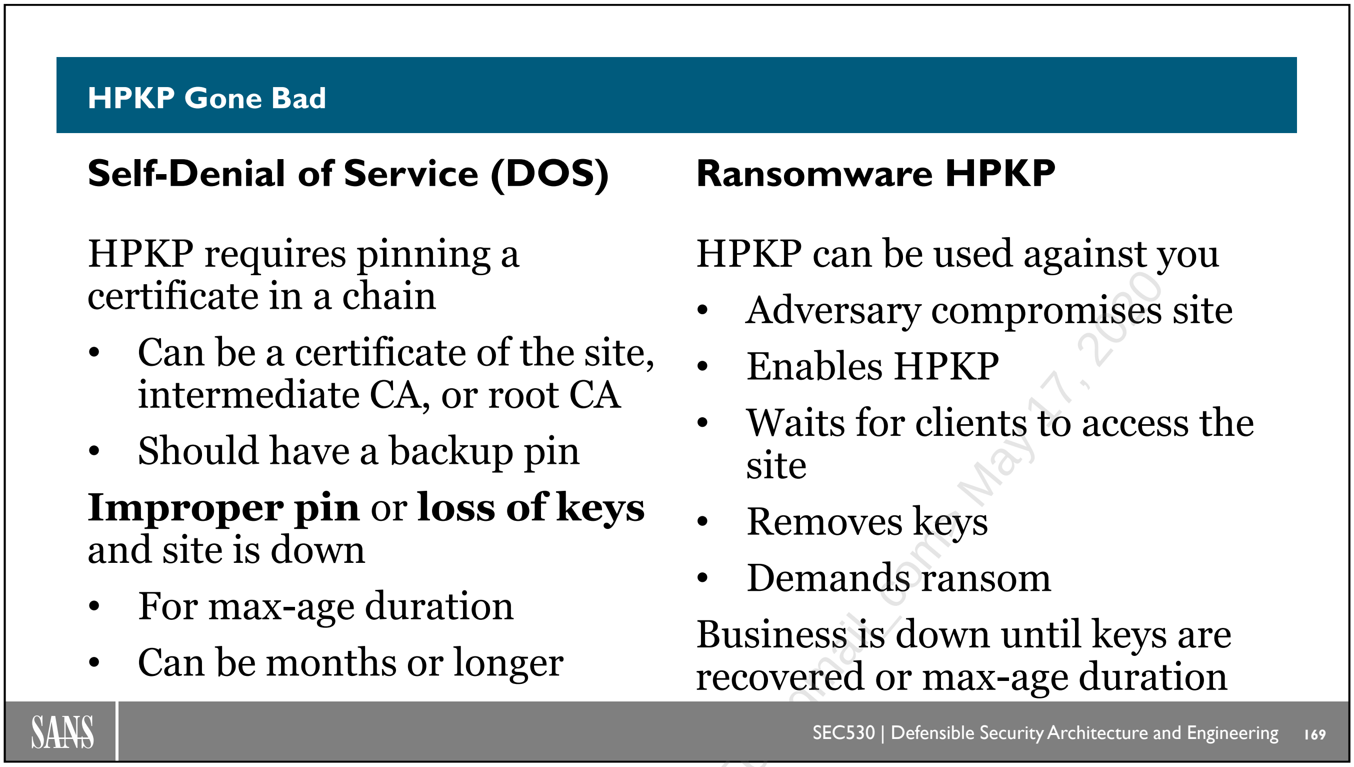
HTTP Public Key Pinning (HPKP)

HSTS enforces the use of HTTPS. However, HSTS still is not perfect. An adversary who can create a trusted certificate for a site can abuse victim machines and intercept user-supplied data. While significantly more difficult to pull off compared to HTTP downgrade attacks it is possible. HPKP solves this by pinning the excepted certificate public key to the site. With HPKP even a legitimate certificate for a site will throw an error into a browser that a certificate is not trusted if it does not match the pinned public key.

HPKP works by providing a header on a web server that includes the public key expected to be used, how long the HPKP setting should be enforced, and a required backup public key. The duration HPKP is to be enforced set by max-age, and the public keys that are enforced are specified by pin-sha256. Similar to HSTS, includeSubDomains can be specified to force a public key for all subdomains. HPKP also supports the ability to notify site owners of improper public key use via the report-uri. Violations are recorded via a JSON payload sent from the application that observed the public key violation. The header of Public-Key-Pins enforces HPKP, and a header of Public-Key-Pins-Report-Only enables HPKP in report-only mode. Report-only mode does not enforce HPKP but will generate a log.

Only certain browsers support HPKP. The reason for this is discussed on the next slide.

[1] https://scotthelme.co.uk/hpkp-cheat-sheet/



HPKP Gone Bad

Self-Denial of Service (DOS)

HPKP requires pinning a certificate in a chain

- Can be a certificate of the site, intermediate CA, or root CA
- Should have a backup pin

Improper pin or **loss of keys** and site is down

- For max-age duration
- Can be months or longer

Ransomware HPKP

HPKP can be used against you

- Adversary compromises site
- Enables HPKP
- Waits for clients to access the site
- Removes keys
- Demands ransom

Business is down until keys are recovered or max-age duration

HPKP Gone Bad

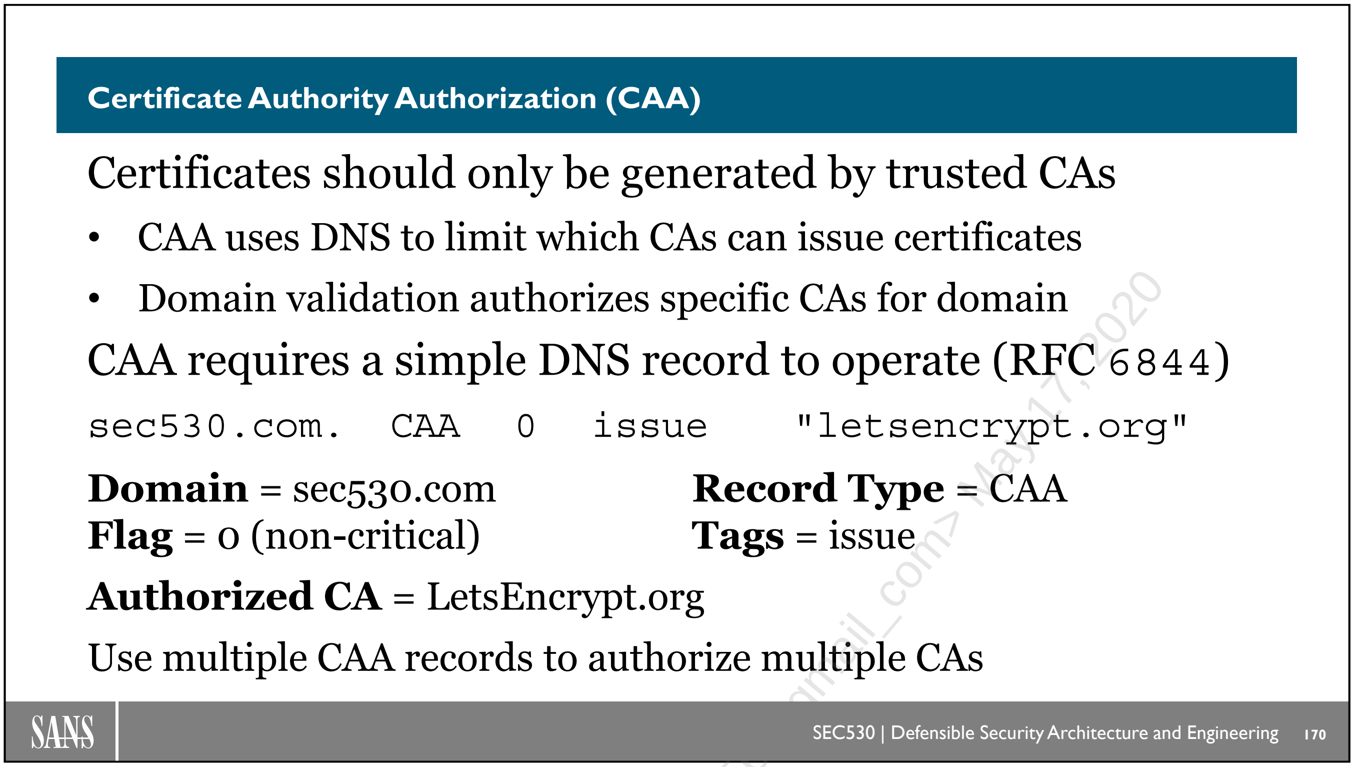
HPKP has serious security implications for both defenders and adversaries. When properly implemented HPKP locks a server to the exact certificate expected. When implemented or abused, HPKP becomes highly destructive. Multiple companies have implemented HPKP only to find out they improperly set the public key or lost the public key. The result is self-denial of service. The site protected by HPKP is down for the duration max-age was set to or until an organization can get major browsers to implement a workaround. The workaround method still requires end users to receive a browser patch. So effectively, a site can be down for months if HPKP is misconfigured.

Adversaries who understand HPKP can also use it for nefarious purposes. Assume an adversary gains access to an organization's web server. The adversary can either use the existing SSL cert or create a new SSL certificate and add an HPKP header using the certificate. The attacker then sits back and waits for clients to access the website. As clients access the site, they began enforcing HPKP. Finally, the adversary removes the SSL keys, and now the website will not load for any certificate except the keys the adversary has. The adversary can now demand ransom or leave the company to its fate.

The use of a reverse proxy can help secure a site's keys. By storing private keys in a reverse proxy, the attacker would have to compromise the reverse proxy to access the keys which is much less likely to happen as most attacks against a web service compromise the backend web server itself.

At a minimum, the organization's enforcing HPKP need to backup all keys.

[1] https://news.netcraft.com/archives/2016/03/22/secure-websites-shun-http-public-key-pinning.html



Certificate Authority Authorization (CAA)

Certificates should only be generated by trusted CAs

- CAA uses DNS to limit which CAs can issue certificates
- Domain validation authorizes specific CAs for domain

CAA requires a simple DNS record to operate (RFC 6844)

```
sec530.com.  CAA  0  issue   "letsencrypt.org"
```

Domain = sec530.com **Record Type** = CAA
Flag = 0 (non-critical) **Tags** = issue

Authorized CA = LetsEncrypt.org

Use multiple CAA records to authorize multiple CAs

Certificate Authority Authorization (CAA)

DNS can also be used to support proper certificate use. Specifically, certificate authority authorization or CAA records should be created to enforce which certificate authorities are authorized to create certificates for a given domain. A CAA record type includes a domain, a flag to identify if a record is critical or non-critical, a tag to identify the purpose, and the CAA record value which is usually set to the name of an authorized certificate authority.

Multiple certificate authorities can be authorized by adding multiple CAA records to a domain. The tag of issue authorizes standard certificates, and a tag of issuewild authorizes both standard and wildcard certificates. A special tag of iodef can be used to report invalid certificate requests to the domain owner. If the iodef tag is used either an email address or web server address can be specified such as below:

```
sec530.com.    CAA 0 iodef mailto:someone@sec530.com
sec530.com.    CAA 0 iodef "certissues.sec530.com"
```

Currently not all certificate authorities enforce CAA records.

[1] https://www.ssl.com/article/certification-authority-authorization-caa/

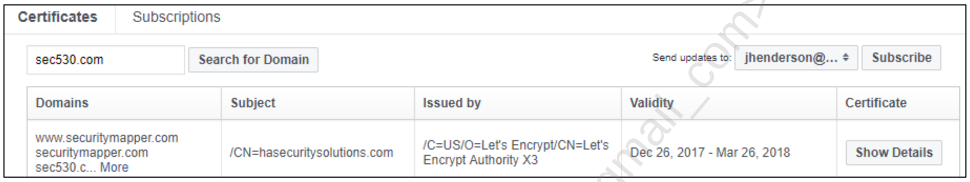
Certificate Transparency Monitoring

Google is pushing for certificate issuance transparency

- Means a public log is available per certificate issued
- The goal is to require support for certificate transparency

Provides near real-time notification of new certificates

- Vendors offer commercial integration and monitoring[1]
- Facebook provides free certificate transparency monitoring[2]



sec530.com

Send updates to: jhenderson@...

Domains	Subject	Issued by	Validity	Certificate
www.securitymapper.com securitymapper.com sec530.c... More	/CN=hasecuritysolutions.com	/C=US/O=Let's Encrypt/CN=Let's Encrypt Authority X3	Dec 26, 2017 - Mar 26, 2018	Show Details

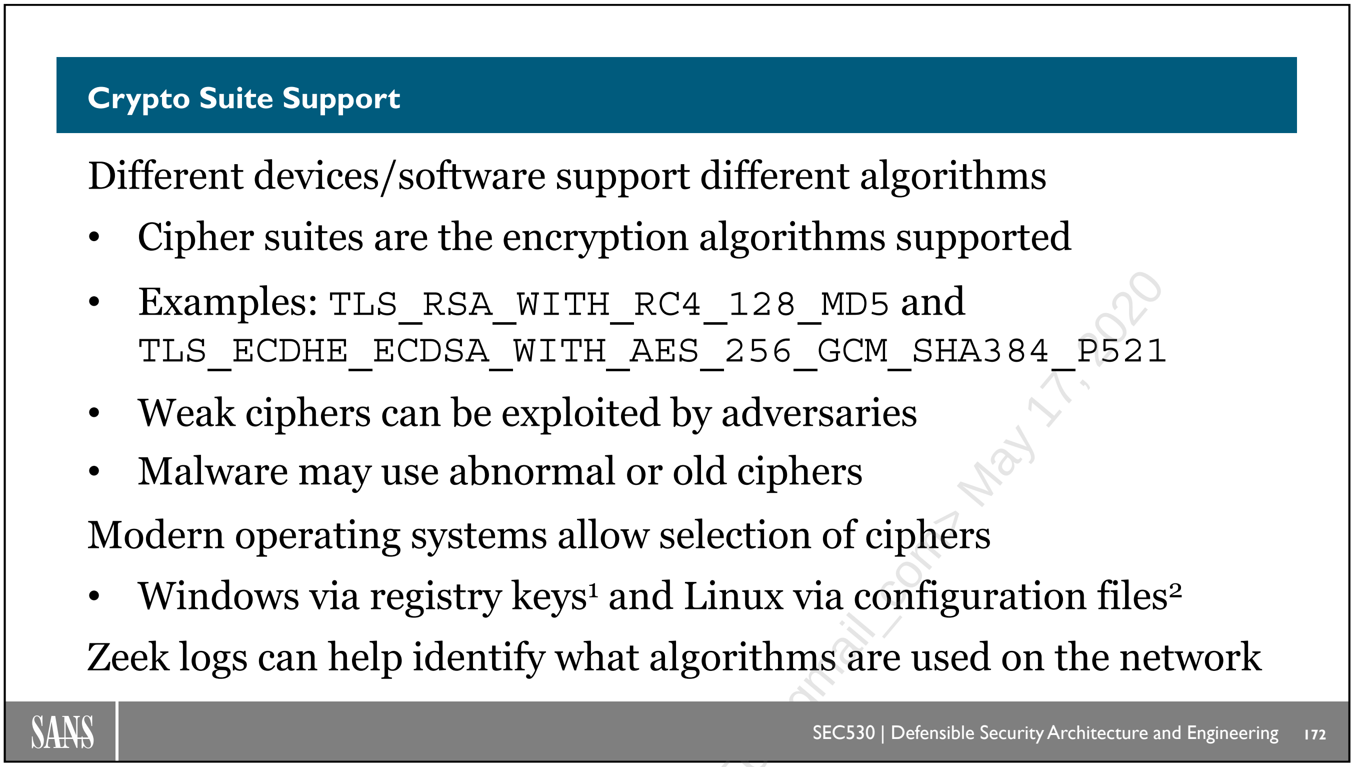
Certificate Transparency Monitoring

Another security implementation around certificate issuance is the concept of certificate transparency. Certificate transparency is the concept of any time a certificate is issued; it must create a public log. The log provides anyone with the ability to see what certificates have been issued for a given domain, which certificate authority generated them, and details about the certificates issued. The information is stored in a signed certificate timestamp (SCT). The SCT is usually added as an X.509v3 extension on the certificate.

The existence of certificate transparency provides near real-time detection capabilities of certificate misuse. Commercial solutions like CertSpotter or free solutions like Facebook certificate transparency monitoring can be subscribed to and provide automatic notification of new certificates issued for a domain. Google is pushing strongly for certificate transparency. To help force certificate transparency, Google Chrome will check a site's certificate of compliance with certificate transparency. Failure to comply will result in page errors or notifications and potentially being blocked moving forward.

[1] https://sslmate.com/certspotter/

[2] https://developers.facebook.com/tools/ct/



Crypto Suite Support

Different devices/software support different algorithms

- Cipher suites are the encryption algorithms supported
- Examples: `TLS_RSA_WITH_RC4_128_MD5` and `TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P521`
- Weak ciphers can be exploited by adversaries
- Malware may use abnormal or old ciphers

Modern operating systems allow selection of ciphers

- Windows via registry keys[1] and Linux via configuration files[2]

Zeek logs can help identify what algorithms are used on the network

Crypto Suite Support

Servers utilizing encryption are only as strong as the encryption used. The strength of encryption is based on the algorithms being used and the protocol such as Secure Sockets Layer (SSL) or Transport Layer Security (TLS). Systems like web servers should be configured to use strong encryption. Modern systems should all use a version of TLS. However, the algorithms supported also need to be tuned. The algorithms used are referred to as cipher suites. Cipher suites are the list of supported encryption and hashing algorithms a system supports and are used to negotiate encryption.

Cipher suites supported by a system or service are controlled by either configuration files, registry keys, or group policy. For example, an Apache service on Linux can be configured to only use certain cipher suites by using a global configuration file or a site-specific file such as /etc/apache2/sites-enabled/ssl_site.conf. A Windows system can be configured by editing the group policy found at Computer Configuration -> Administrative Templates -> Network -> SSL Configuration Settings -> SSL Cipher Suite Order.

Selecting strong cipher suites will cause older devices and software to be unable to establish encrypted sessions. For encryption to work, both client and server must negotiate to cipher suites supported by both systems. The default cipher suites supported on many systems allow for weak encryption ciphers to be selected but supports many devices. However, a modern web server typically is not concerned with old clients such as Windows 95 or even Windows XP.

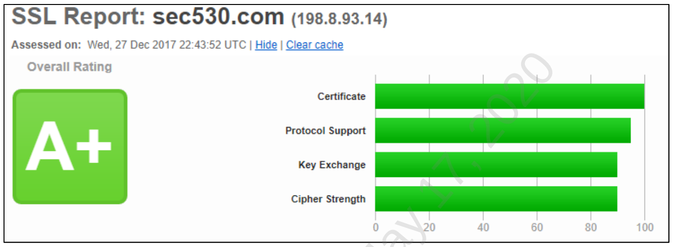
[1] https://github.com/EnclaveConsulting/SANS-SEC505/blob/master/SEC505-Scripts.zip

[2] https://www.linuxjournal.com/content/cipher-security-how-harden-tls-and-ssh

Qualys SSL Labs[1]

Web servers need to be hardened for proper encryption

- Disable SSL support
- Fine tune cipher suites
- Enable proper headers
  - Such as HSTS and HPKP
- Harden SSL/TLS settings



SSL Report: sec530.com (198.8.93.14)

Assessed on: Wed, 27 Dec 2017 22:43:52 UTC | Hide | Clear cache

Overall Rating

A+

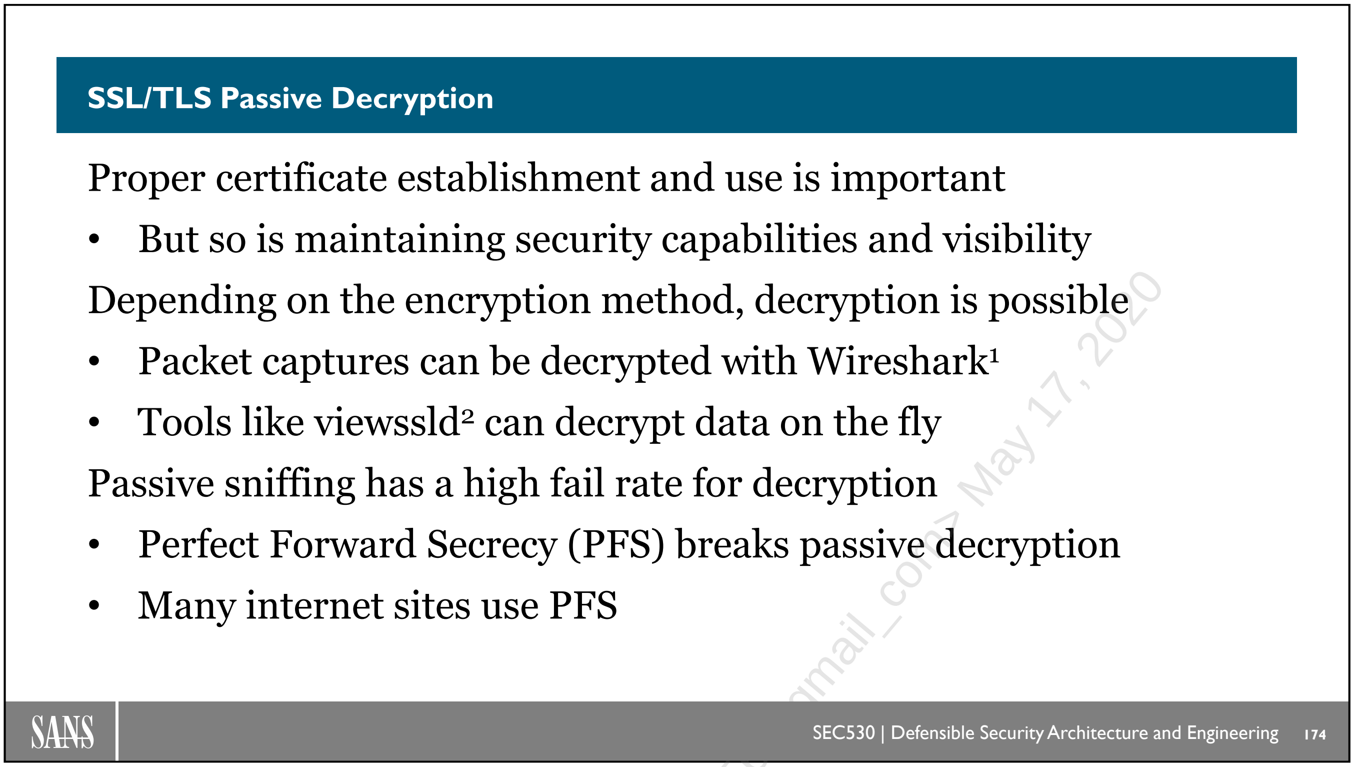
Qualys SSL Labs[1] scans and grades SSL/TLS settings

- Recommendations provided to improve the score

Qualys SSL Labs[1]

Qualys provides a free online solution called Qualys SSL Labs. This solution will scan a domain's SSL implementation and provide a report card. The report will contain an overall ranking similar to a school grade of A+ through F and also will provide specific details on where things need to be changed and where things are properly set.

The documentation provided in the report is highly detailed and can be used to harden a web server's SSL/TLS configuration. Examples of settings it will recommend are disabling SSL compression, disabling session tickets, enabling HSTS, and honoring cipher suite orders.



SSL/TLS Passive Decryption

Proper certificate establishment and use is important

- But so is maintaining security capabilities and visibility

Depending on the encryption method, decryption is possible

- Packet captures can be decrypted with Wireshark[1]
- Tools like viewssld[2] can decrypt data on the fly

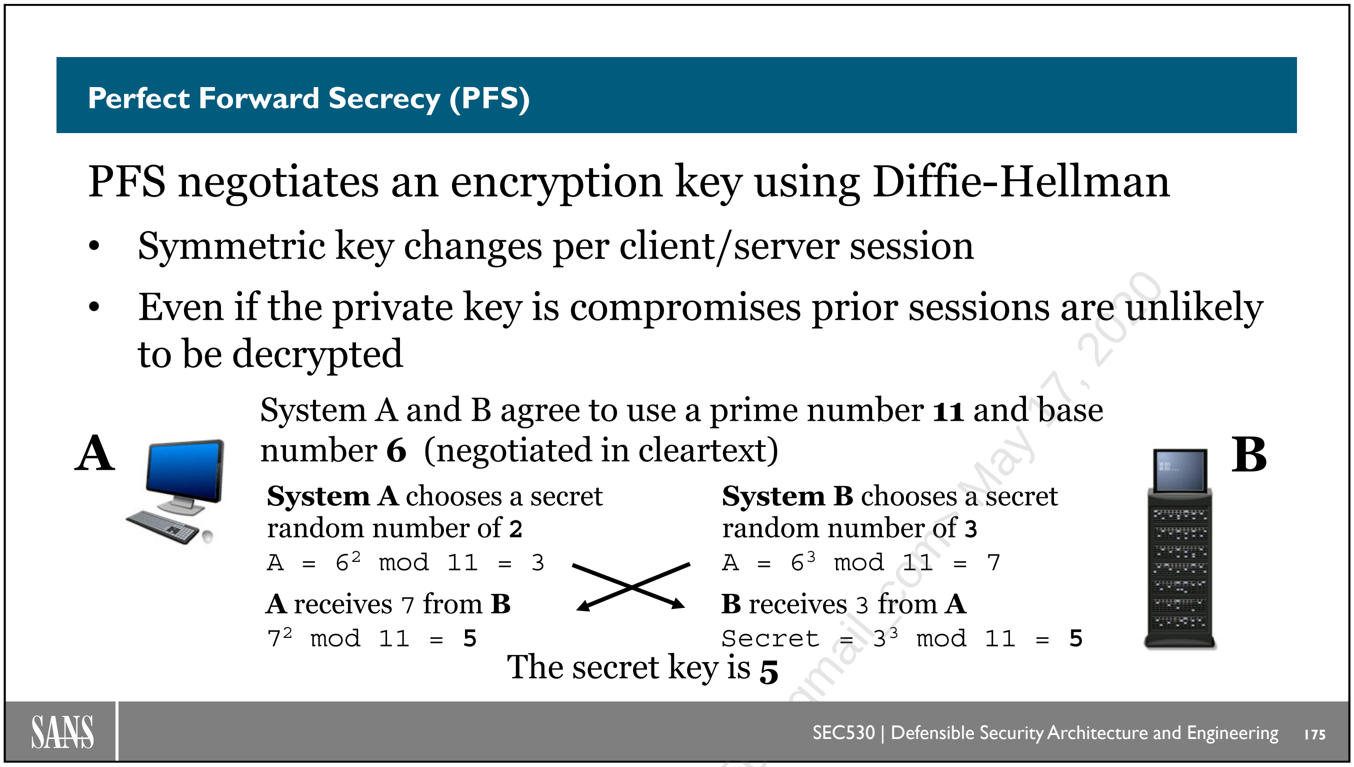
Passive sniffing has a high fail rate for decryption

- Perfect Forward Secrecy (PFS) breaks passive decryption
- Many internet sites use PFS

SSL/TLS Passive Decryption

Decrypting SSL/TLS traffic that has been captured is possible, but it is not practical. For example, if Wireshark was used to capture HTTPS traffic and the private key of the web server was accessible then Wireshark can use the private key to decrypt the HTTPS session. However, this requires the private key, and it also requires that certain configurations such as perfect forward secrecy (PFS) are not in use.

These requirements mean that an IDS, NGFW, or any passive monitoring technology would require all possible private keys to be provided and that encryption uses weak cipher suites. Since many sites accessed are not owned by the organization these conditions will never be possible. Therefore, passive automated decryption is not a viable solution.



Perfect Forward Secrecy (PFS)

PFS negotiates an encryption key using Diffie-Hellman

- Symmetric key changes per client/server session
- Even if the private key is compromises prior sessions are unlikely to be decrypted

System A and B agree to use a prime number **11** and base number **6** (negotiated in cleartext)

System A chooses a secret random number of **2**

$A = 6^2 \bmod 11 = 3$

A receives 7 from **B**

$7^2 \bmod 11 = \mathbf{5}$

System B chooses a secret random number of **3**

$A = 6^3 \bmod 11 = 7$

B receives 3 from **A**

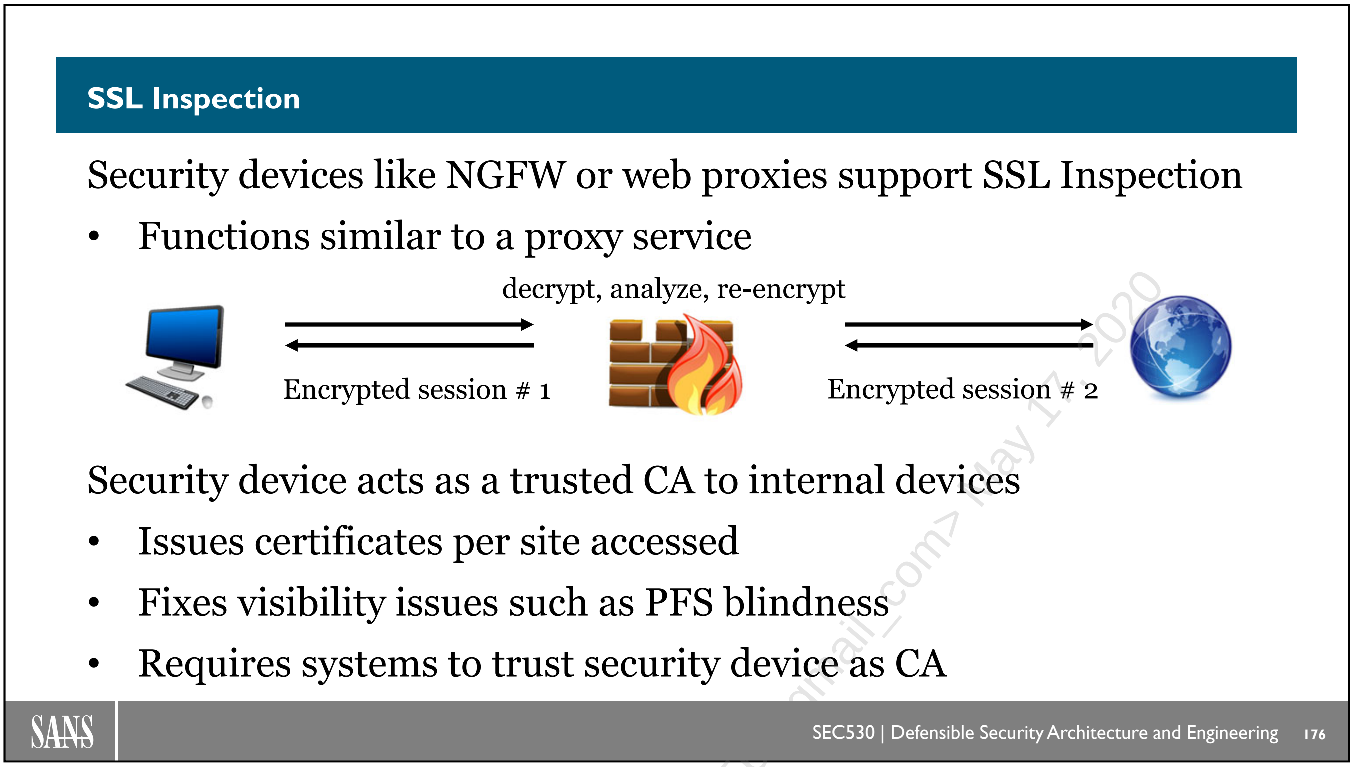
$\text{Secret} = 3^3 \bmod 11 = \mathbf{5}$

The secret key is **5**

Perfect Forward Secrecy (PFS)

Perfect forward secrecy is an implementation of Diffie-Hellman that enforces a new symmetric key per client/server session. Key negotiation is done using a prime number and a base number that both are negotiated and agreed upon by the client and server in cleartext. Even though these numbers are shared in cleartext, a mathematic calculation is performed with these numbers to generate a shared key that is only known by the client and server. The math uses exponents and modulus calculations and is demonstrated in this slide.

The example in the slide uses small numbers to simplify the demonstration. Normally, the prime number, base number, and random numbers are large numbers. The large number use makes it computationally improbable that an adversary can brute force the secret key negotiated.



SSL Inspection

Security devices like NGFW or web proxies support SSL Inspection

- Functions similar to a proxy service

decrypt, analyze, re-encrypt

Encrypted session # 1

Encrypted session # 2

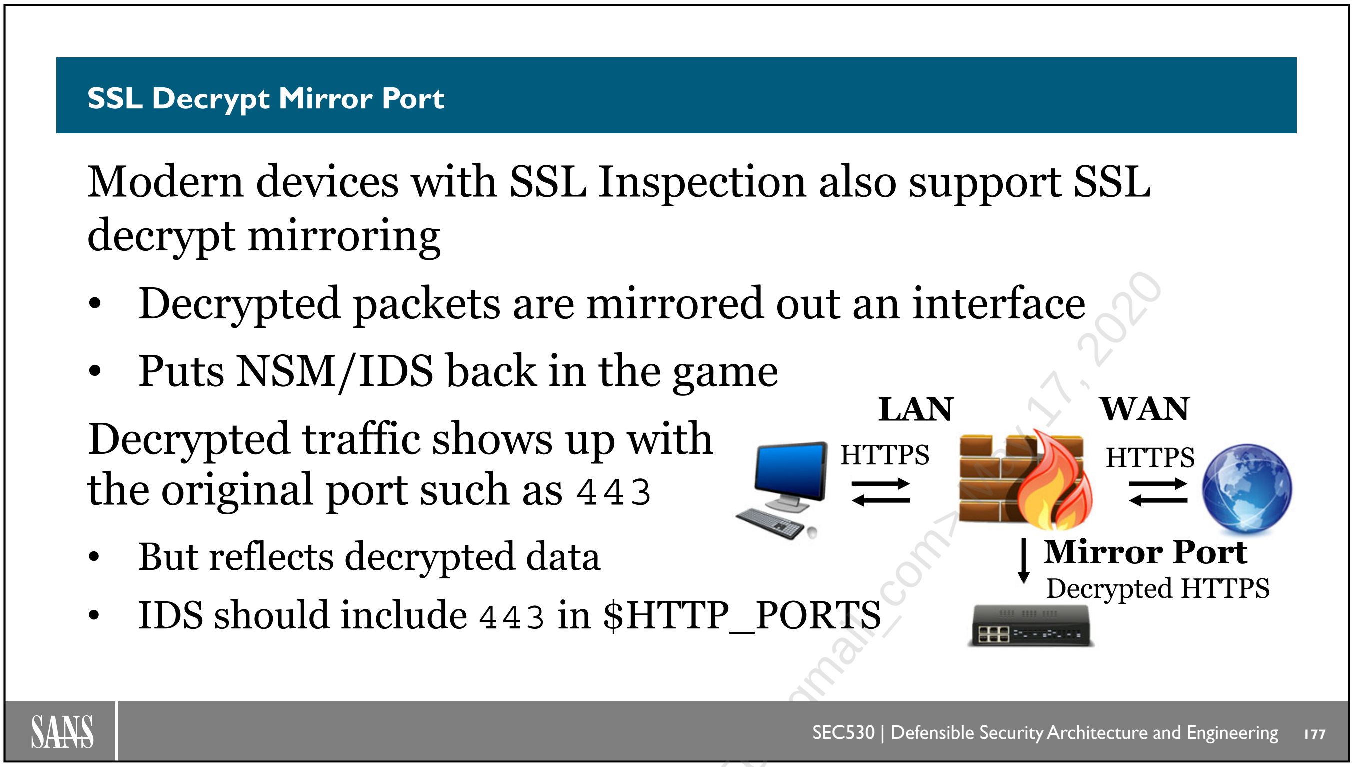
Security device acts as a trusted CA to internal devices

- Issues certificates per site accessed
- Fixes visibility issues such as PFS blindness
- Requires systems to trust security device as CA

SSL Inspection

SSL inspection involves acting as a proxy for SSL/TLS connections. It functions by initiating an SSL session with the requested services, receiving the expected data, decrypting it and then re-encrypting using a generated certificate. The re-encrypted data is delivered to the initial system requesting the connection. In order for this to work the certificates generated by the SSL inspection device must be trusted by the internal asset. The beauty of SSL inspection is it works even for strong algorithms that use perfect forward secrecy.

The most important aspect of SSL inspection is that now there is full visibility into the traffic. The visibility includes everything including usernames, passwords, and other sensitive information. As a result, there are political and ethical concerns with SSL inspection. However, without SSL inspection technologies like AV, IDS/IPS, malware detonation, and more cannot function and become expensive paperweights. Of the many political hurdles to try and overcome choose SSL inspection. It is worth the fight.



SSL Decrypt Mirror Port

Modern devices with SSL Inspection also support SSL decrypt mirroring

- Decrypted packets are mirrored out an interface
- Puts NSM/IDS back in the game

Decrypted traffic shows up with the original port such as `443`

- But reflects decrypted data
- IDS should include `443` in $HTTP_PORTS

LAN
HTTPS
WAN
HTTPS
Mirror Port
Decrypted HTTPS

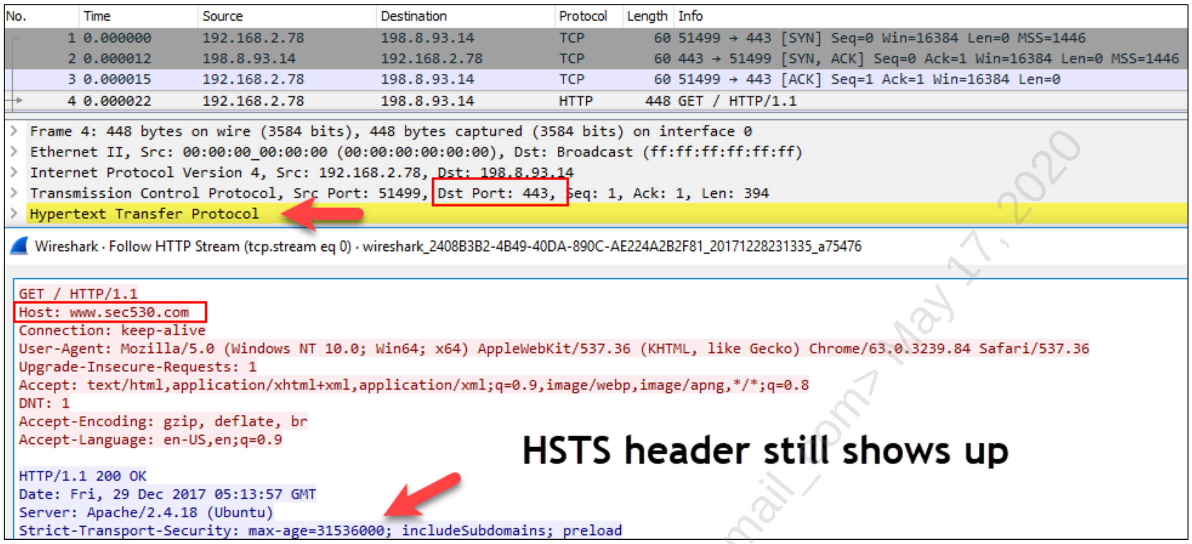
SSL Decrypt Mirror Port

Modern NGFWs support both SSL inspection and SSL decrypt mirror ports. An SSL decrypt mirror port takes all the traffic decrypted by SSL inspection and sends it to another device. Effectively, a decrypt mirror port is like a tap or port mirror that only contains decrypted data from SSL inspection. This SSL decrypt mirror port then enables other security devices such as IDS/IPS and malware detonation systems the ability to analyze passive traffic in real-time, even if it was decrypted. The mirror can also be useful for troubleshooting.

When using the decrypt mirror, it is important to configure the back-end system to identify normal HTTPS ports as HTTP. The reason for this is traffic will appear from port 443 and other TLS ports, but the traffic is decrypted. An example of this would be configuring an IDS to include 443 in the $HTTP_PORTS variable. In truth, the solution is not that simple. In reality, you probably would want either a dedicated IDS configured to monitor the decrypted data or a highly tuned IDS that will only analyze 443 traffic as cleartext if it came in from a specific interface. Both are easily achievable with Security Onion.

While not all NGFWs support SSL decrypt mirroring, the solution is inexpensive and supported by multiple vendors. For example, an entry-level Fortigate firewall costs around $600 without any discount and can support SSL inspection and SSL decrypt mirroring. Even at $600, the device is capable of handling 250 Mbps of SSL inspected traffic.

SSL Decrypt Example



```
No.  Time      Source        Destination   Protocol Length Info
   1 0.000000  192.168.2.78  198.8.93.14   TCP        60 51499 → 443 [SYN] Seq=0 Win=16384 Len=0 MSS=1446
   2 0.000012  198.8.93.14   192.168.2.78  TCP        60 443 → 51499 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1446
   3 0.000015  192.168.2.78  198.8.93.14   TCP        60 51499 → 443 [ACK] Seq=1 Ack=1 Win=16384 Len=0
   4 0.000022  192.168.2.78  198.8.93.14   HTTP      448 GET / HTTP/1.1

> Frame 4: 448 bytes on wire (3584 bits), 448 bytes captured (3584 bits) on interface 0
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Internet Protocol Version 4, Src: 192.168.2.78, Dst: 198.8.93.14
> Transmission Control Protocol, Src Port: 51499, Dst Port: 443, Seq: 1, Ack: 1, Len: 394
> Hypertext Transfer Protocol
```

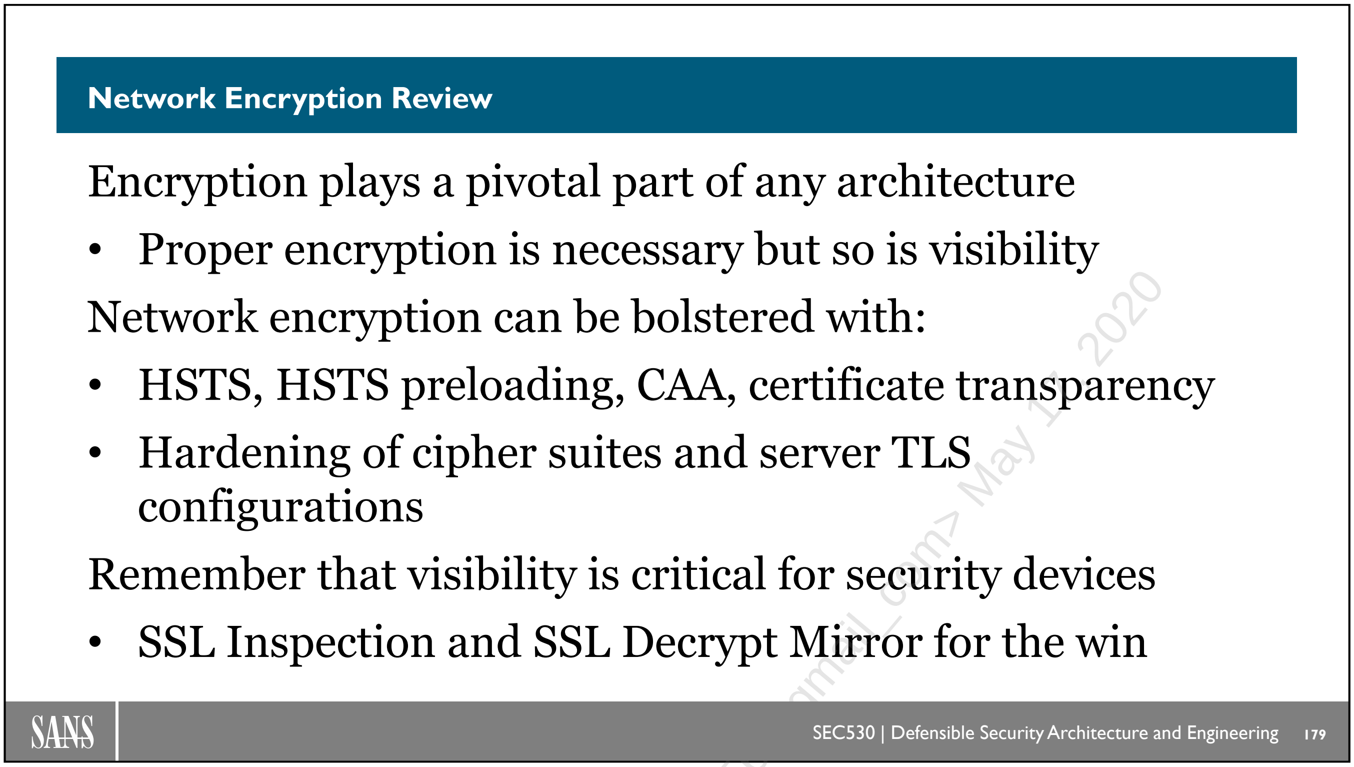
Wireshark · Follow HTTP Stream (tcp.stream eq 0) · wireshark_2408B3B2-4B49-40DA-890C-AE224A2B2F81_20171228231335_a75476

```
GET / HTTP/1.1
Host: www.sec530.com
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.84 Safari/537.36
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
DNT: 1
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9

HTTP/1.1 200 OK
Date: Fri, 29 Dec 2017 05:13:57 GMT
Server: Apache/2.4.18 (Ubuntu)
Strict-Transport-Security: max-age=31536000; includeSubdomains; preload
```

HSTS header still shows up

SSL Decrypt Example

This slide is an example of capturing data from an SSL decrypt mirror port using Wireshark. In this picture, traffic to https://www.sec530.com is shown in cleartext as if it was traditional HTTP. Also, the server response even shows the HSTS header enforcing HTTPS.



Network Encryption Review

Encryption plays a pivotal part of any architecture

- Proper encryption is necessary but so is visibility

Network encryption can be bolstered with:

- HSTS, HSTS preloading, CAA, certificate transparency
- Hardening of cipher suites and server TLS configurations

Remember that visibility is critical for security devices

- SSL Inspection and SSL Decrypt Mirror for the win

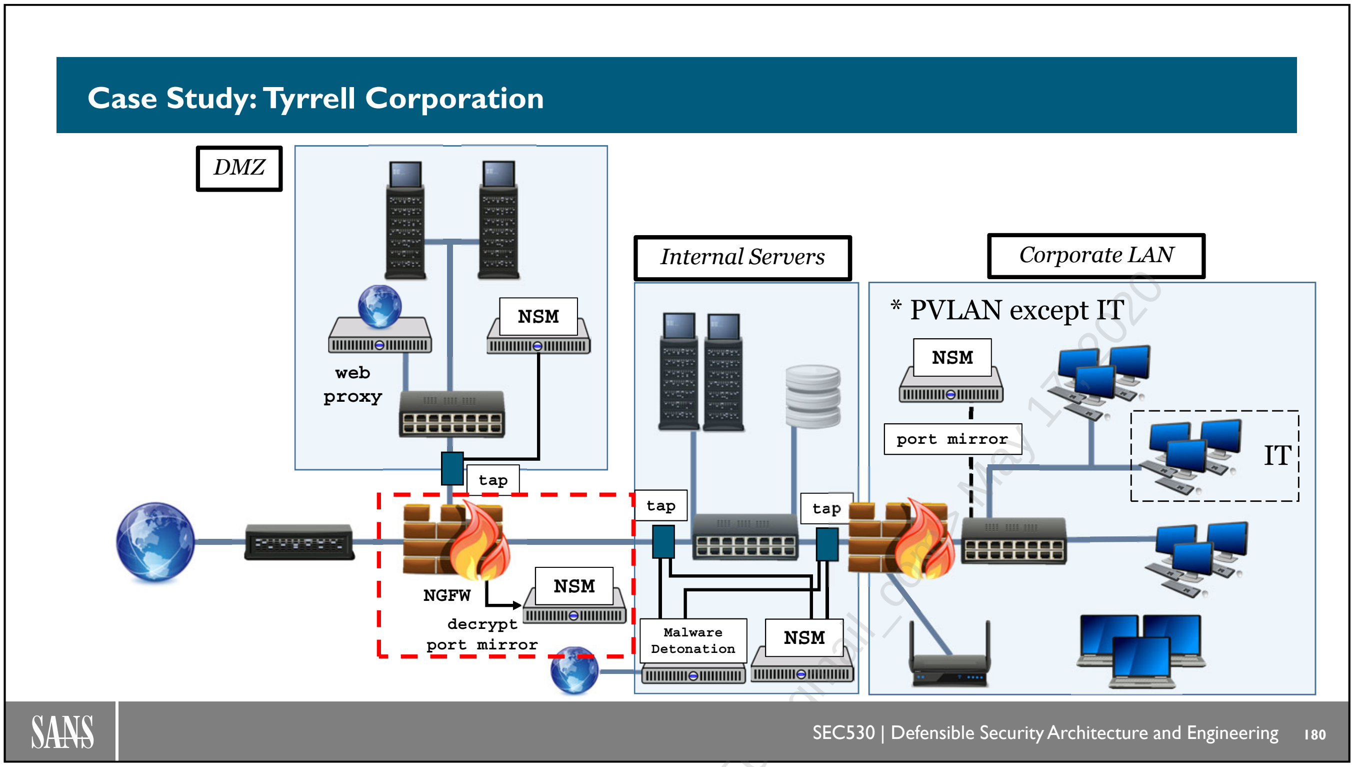
SANS

SEC530 | Defensible Security Architecture and Engineering 179

Network Encryption Review

How and why network traffic encryption is implemented matters. When improperly configured, adversaries use downgrade attacks or exploit weaknesses in implementations. By implementing technologies like HSTS, HPKP, CAA, and certificate transparency multiple capabilities for prevention and detection are achieved.

Equally important is the consideration for using SSL inspection and SSL decrypt mirroring. Without SSL inspection large portions of adversary traffic will be unobserved. If an organization is having trouble approving SSL inspection, reach out to other organizations to find out how they were able to get it approved. Organizations have had success getting SSL inspection approved even in countries where the response is automatically no. Another approach is asking the organization if they feel the antivirus, intrusion prevention, and application control capabilities they purchased in their X dollar NGFW are important. If they answer yes, ask them why it is acceptable to have those features disabled on over 50% of traffic.



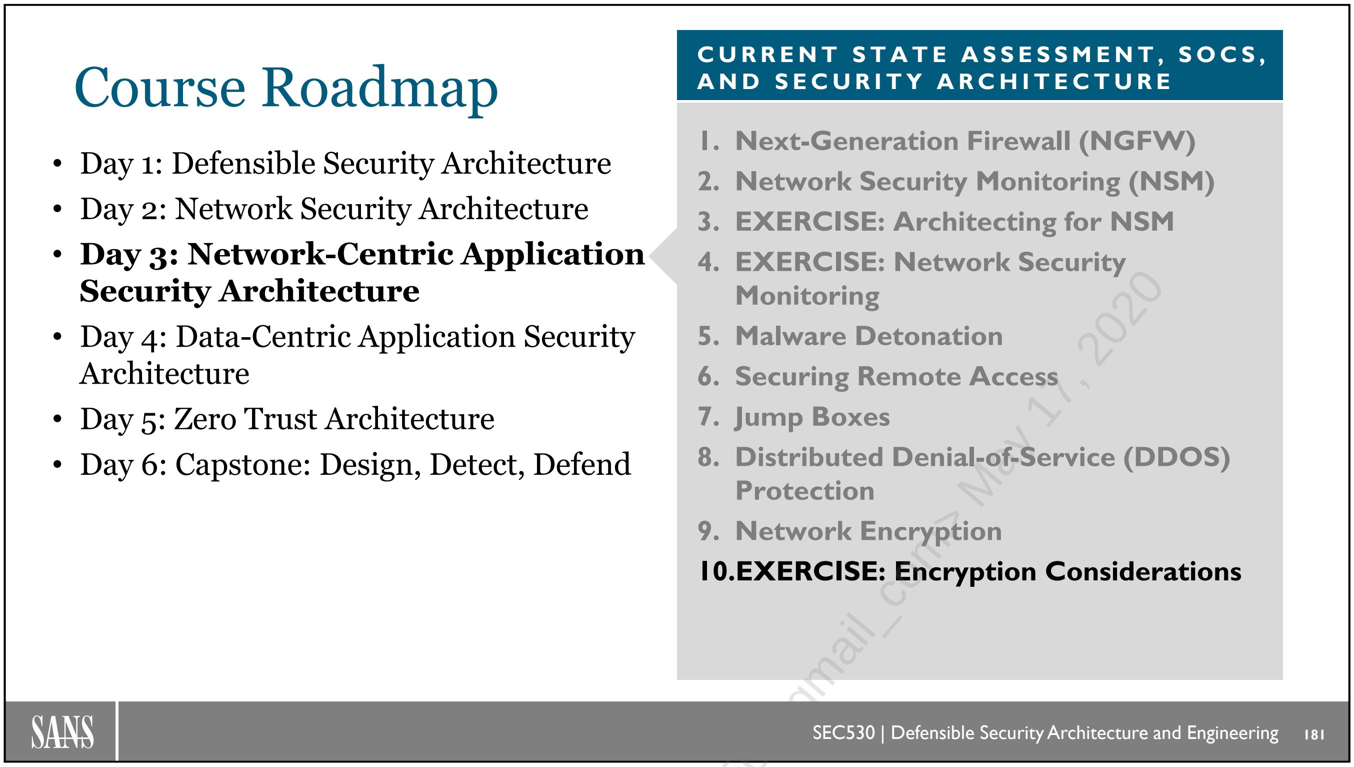
Case Study: Tyrrell Corporation

SANS

SEC530 | Defensible Security Architecture and Engineering 180

Case Study: Tyrrell Corporation

This diagram shows the addition of a decrypt port mirror and Network Security Monitor sensor to the Tyrrell Corporation architecture. In this example, this allows for TLS traffic to be decrypted for the NGFW to inspect data and then a copy of the decrypted data to be sent to a Network Security Monitor.



Course Roadmap

- Day 1: Defensible Security Architecture
- Day 2: Network Security Architecture
- **Day 3: Network-Centric Application Security Architecture**
- Day 4: Data-Centric Application Security Architecture
- Day 5: Zero Trust Architecture
- Day 6: Capstone: Design, Detect, Defend

CURRENT STATE ASSESSMENT, SOCS, AND SECURITY ARCHITECTURE

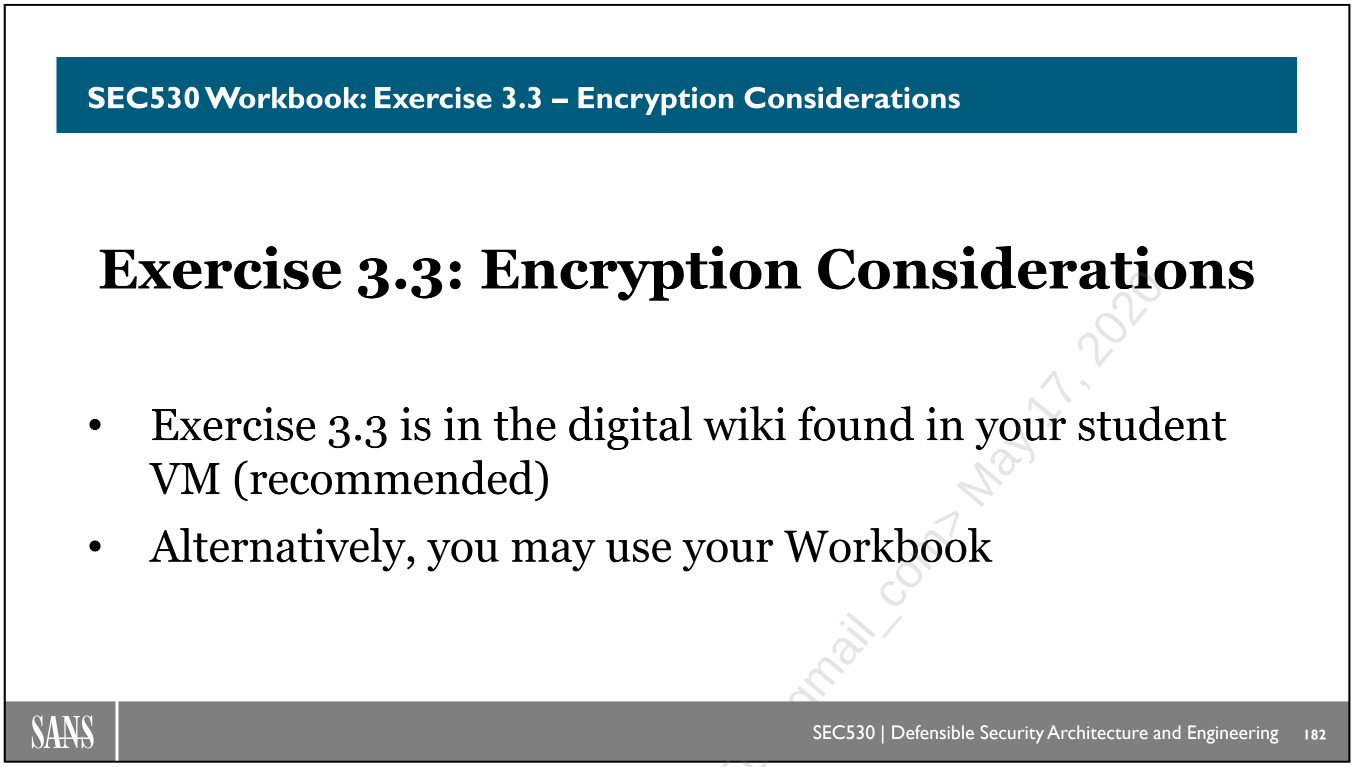
1. Next-Generation Firewall (NGFW)
2. Network Security Monitoring (NSM)
3. EXERCISE: Architecting for NSM
4. EXERCISE: Network Security Monitoring
5. Malware Detonation
6. Securing Remote Access
7. Jump Boxes
8. Distributed Denial-of-Service (DDOS) Protection
9. Network Encryption
10. **EXERCISE: Encryption Considerations**

SANS

SEC530 | Defensible Security Architecture and Engineering 181

Course Roadmap

We will now have a lab on handling network encryption as a defender.



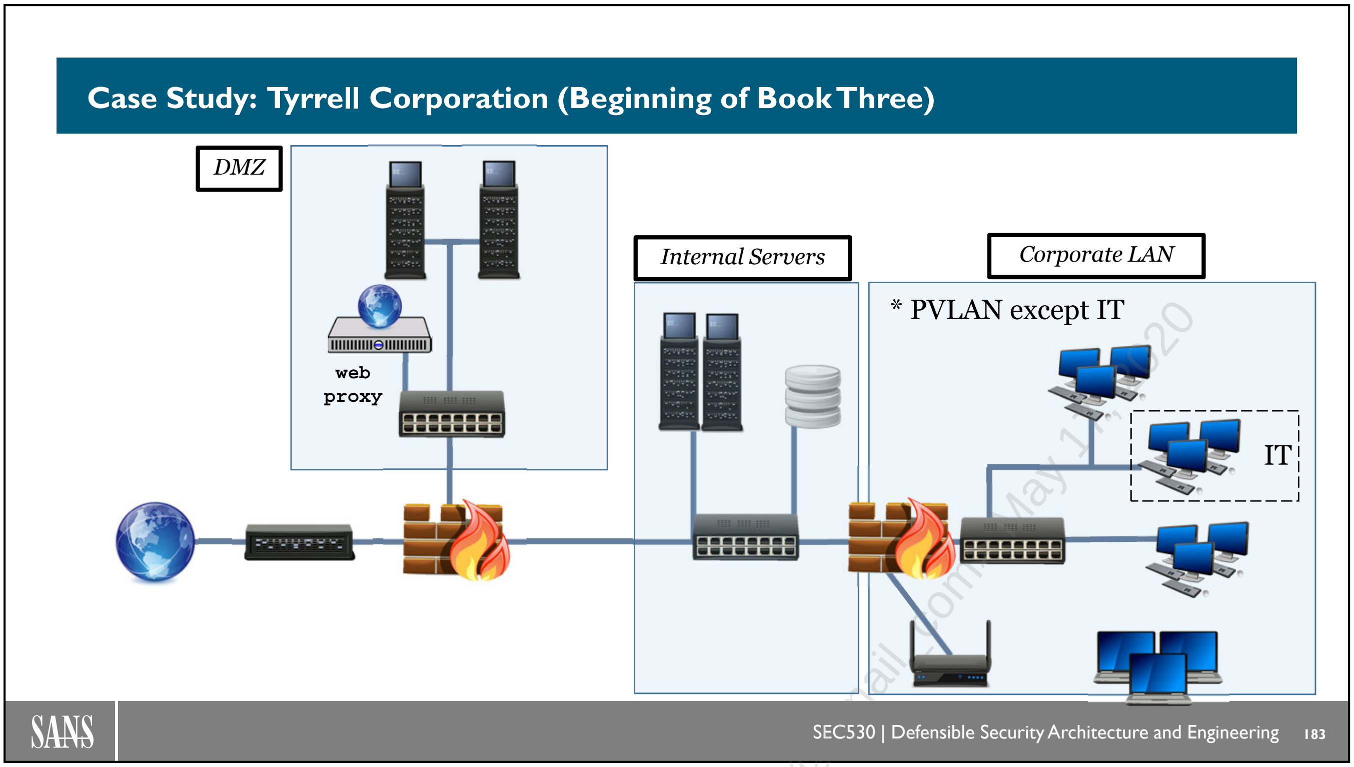
SEC530 Workbook: Exercise 3.3 – Encryption Considerations

Exercise 3.3: Encryption Considerations

- Exercise 3.3 is in the digital wiki found in your student VM (recommended)
- Alternatively, you may use your Workbook

SEC530 | Defensible Security Architecture and Engineering 182

Please open the digital wiki or your lab workbook and access exercise 3.3.



Case Study: Tyrrell Corporation (Beginning of Book Three)

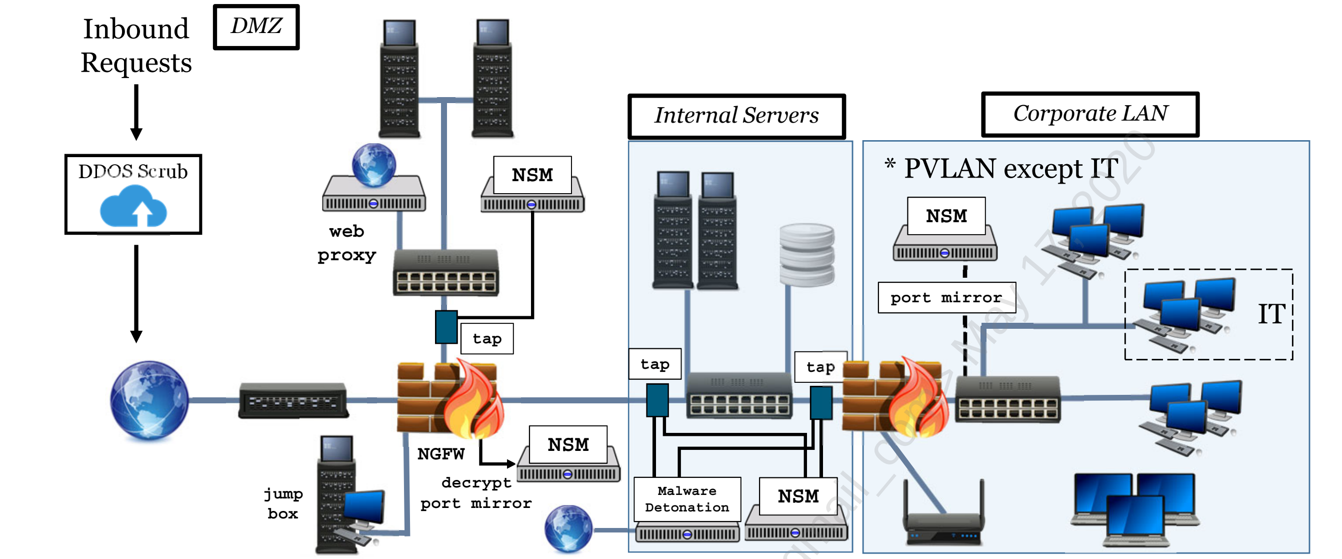
SANS

SEC530 | Defensible Security Architecture and Engineering 183

Case Study: Tyrrell Corporation (Beginning of Book Three)

This diagram represents the Tyrrell Corporation's network design at the beginning of book three.

Case Study: Tyrrell Corporation (End of Book Three)



Case Study: Tyrrell Corporation (End of Book Three)

This diagram represents the Tyrrell Corporation's network design at the end of book three.