

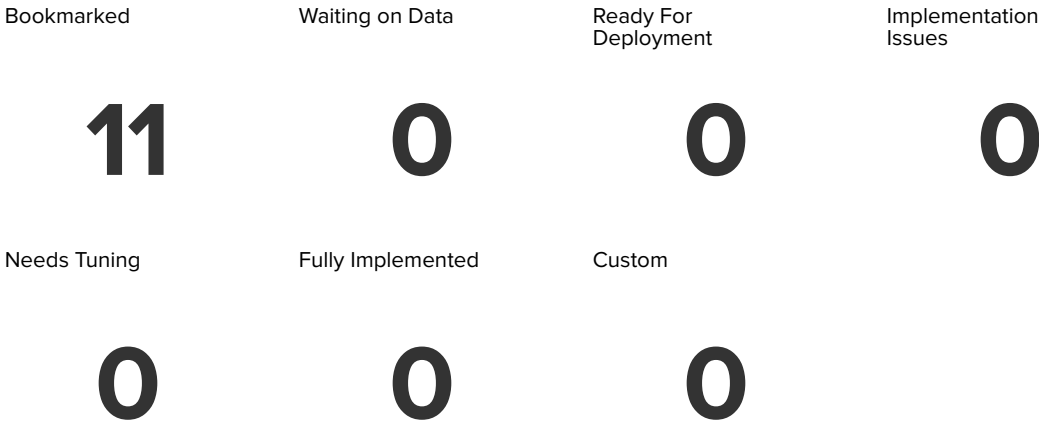


Splunk Security Content Export

Prepared 14 December 2020 via the Manage Bookmarks dashboard

Table of Contents

1. Use Case Overview
2. Data Sources
3. Use Cases for Data Sources
4. Content Detail



Data Sources Required

Data Source	Use Cases	Categories	Description
Authentication	Advanced Threat Detection: Focused on detecting advanced persistent threats, experienced and motivated attackers, and nation-state actors. Compliance: Provides assistance in ensuring that organizations are implementing all required monitoring for the regulatory environment. Security Monitoring: The foundation of security, looking for common activities from common malware and attackers.	Lateral Movement, GDPR	2 use cases selected
Endpoint Detection and Response	Advanced Threat Detection: Focused on detecting advanced persistent threats, experienced and motivated attackers, and nation-state actors.	Lateral Movement, Ransomware	5 use cases selected
Network Communication	Advanced Threat Detection: Focused on detecting advanced persistent threats, experienced and motivated attackers, and nation-state actors.	Lateral Movement	1 use cases selected
Windows Security	Advanced Threat Detection: Focused on detecting advanced persistent threats, experienced and motivated attackers, and nation-state actors. Compliance: Provides assistance in ensuring that organizations are implementing all required monitoring for the regulatory environment. Security Monitoring: The foundation of security, looking for common activities from common malware and attackers.	Lateral Movement, Ransomware, GDPR	7 use cases selected

Use Cases for Data Sources

Authentication

- First Time Logon to New Server
- Increase in # of Hosts Logged into

Network Communication

- Remote Desktop Network Traffic

Windows Security

- Authentication Against a New Domain Controller
- Detect Lateral Movement With WMI
- First Time Logon to New Server
- Increase in # of Hosts Logged into
- New AD Domain Detected
- Remote PowerShell Launches
- Significant Increase in Interactive Logons

Endpoint Detection and Response

- Detect Lateral Movement With WMI
- Remote Desktop Process Running On System
- Remote PowerShell Launches
- Remote WMI Command Attempt
- Schtasks Scheduling Job On Remote System

Remote WMI Command Attempt

Status

Bookmarked

App

Enterprise Security Content Update

Description

This search looks for wmic.exe being launched with parameters to operate on remote systems.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

<div><div>Use Case</div><div>Advanced Threat Detection</div><div>Category</div><div>Lateral Movement</div><div>Alert Volume</div><div>This search looks for wmic.exe being launched with parameters to operate on remote systems.</div><div>SPL Difficulty</div><div>None</div></div>	<div><div>Data Availability</div><div>Bad</div><div>Journey</div><div>Stage 3</div><div>MITRE ATT&CK Tactics</div><div>Execution</div><div>MITRE ATT&CK Techniques</div><div>Windows Management Instrumentation</div><div>Windows Management Instrumentation</div><div>MITRE Threat Groups</div><div>APT29</div><div>APT32</div><div>APT41</div><div>Blue Mockingbird</div><div>Chimera</div><div>Deep Panda</div><div>FIN6</div><div>FIN8</div><div>Frankenstein</div><div>Lazarus Group</div><div>Leviathan</div><div>MuddyWater</div><div>OilRig</div><div>Soft Cell</div><div>Stealth Falcon</div><div>Threat Group-3390</div><div>Wizard Spider</div><div>menuPass</div><div>Kill Chain Phases</div><div>Actions On Objectives</div><div>Data Sources</div><div>Endpoint Detection and Response</div></div>
---	--

[> Help](#)

Remote WMI Command Attempt Help

You must be ingesting data that records process activity from your hosts to populate the Endpoint data model in the Processes node. You must also be ingesting logs with both the process name and command line from your endpoints. The command-line arguments are mapped to the "process" field in the Endpoint data model.

[v Search](#)

```
| tstats `security_content_summariesonly` count values(Processes.process) as process
values(Processes.parent_process) as parent_process min(_time) as firstTime max(_time) as lastTime from
datamodel=Endpoint.Processes where Processes.process_name=wmic.exe AND Processes.process= */node* by
Processes.user Processes.process_name Processes.parent_process_name Processes.dest |
`drop_dm_object_name(Processes)` | `security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` |
`remote_wmi_command_attempt_filter`
```

[Open in Search](#)

Remote Desktop Process Running On System

Status

Bookmarked

App

Enterprise Security Content Update

Description

This search looks for the remote desktop process mstsc.exe running on systems upon which it doesn't typically run. This is accomplished by filtering out all systems that are noted in the `common_rdp_source category` in the Assets and Identity framework.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

Use Case

Advanced Threat Detection

Category

Lateral Movement

Alert Volume

This search looks for the remote desktop process mstsc.exe running on systems upon which it doesn't typically run. This is accomplished by filtering out all systems that are noted in the ``common_rdp_source category`` in the Assets and Identity framework.

SPL Difficulty

None

Data Availability

Bad

Journey

Stage 3

MITRE ATT&CK Tactics

Lateral Movement

MITRE ATT&CK Techniques

Remote Services
Remote Desktop Protocol

MITRE Threat Groups

APT1 APT3 APT39 APT41 Axiom
Blue Mockingbird Chimera Cobalt Group
Dragonfly 2.0 FIN10 FIN6 FIN8 Lazarus Group
Leviathan OilRig Patchwork Silence
Stolen Pencil TEMP.Veles Wizard Spider
menuPass

Kill Chain Phases

Actions On Objectives

Data Sources

Endpoint Detection and Response

> [Help](#)

Remote Desktop Process Running On System Help

To successfully implement this search, you must be ingesting data that records process activity from your hosts to populate the endpoint data model in the processes node. The search requires you to identify systems that do not commonly use remote desktop. You can use the included support search "Identify Systems Using Remote Desktop" to identify these systems. After identifying them, you will need to add the `"commonrdpsource"` category to that system using the Enterprise Security Assets and Identities framework. This can be done by adding an entry in the `assets.csv` file located in `SA-IdentityManagement/Lookups`.

▼ [Search](#)

```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from  
datamodel=Endpoint.Processes where Processes.process=*mstsc.exe AND  
Processes.dest_category!=common_rdp_source by Processes.dest Processes.user Processes.process |  
`security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` | `drop_dm_object_name(Processes)` |  
`remote_desktop_process_running_on_system_filter`
```

[Open in Search](#)

Schtasks Scheduling Job On Remote System

Status

Bookmarked

App

Enterprise Security Content Update

Description

This search looks for flags passed to schtasks.exe on the command-line that indicate a job is being scheduled on a remote system.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

Use Case

Advanced Threat Detection

Category

Lateral Movement

Alert Volume

This search looks for flags passed to schtasks.exe on the command-line that indicate a job is being scheduled on a remote system.

SPL Difficulty

None

Data Availability

Bad

Journey

Stage 3

MITRE ATT&CK Tactics

ExecutionPersistencePrivilege Escalation

MITRE ATT&CK Techniques

Scheduled Task/Job

Scheduled Task

MITRE Threat Groups

APT-C-36APT29APT3APT32APT33

APT39APT41BRONZE BUTLER

Blue MockingbirdChimeraCobalt Group

Dragonfly 2.0FIN10FIN6FIN7FIN8

FrankensteinGamaredon GroupMachete

MuddyWaterOilRigPatchworkRancorSilence

Soft CellStealth FalconTEMPVeles

Wizard SpidermenuPass

Kill Chain Phases

Actions On Objectives

Data Sources

Endpoint Detection and Response

> [Help](#)

Schtasks Scheduling Job On Remote System Help

You must be ingesting data that records process activity from your hosts to populate the Endpoint data model in the Processes node. You must also be ingesting logs with both the process name and command line from your endpoints. The command-line arguments are mapped to the "process" field in the Endpoint data model.

▼ [Search](#)

```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from  
datamodel=Endpoint.Processes where Processes.process_name = schtasks.exe Processes.process="*/create*"   
(Processes.process="* /s *" OR Processes.process="* /S *") by Processes.process_name Processes.process   
Processes.parent_process_name Processes.dest Processes.user | `drop_dm_object_name(Processes)` |   
`security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` |   
`schtasks_scheduling_job_on_remote_system_filter`
```

[Open in Search](#)

Remote Desktop Network Traffic

Status

Bookmarked

App

Enterprise Security Content Update

Description

This search looks for network traffic on TCP/3389, the default port used by remote desktop. While remote desktop traffic is not uncommon on a network, it is usually associated with known hosts. This search allows for whitelisting both source and destination hosts to remove them from the output of the search so you can focus on the uncommon uses of remote desktop on your network.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

Use Case

Advanced Threat Detection

Category

Lateral Movement

Alert Volume

This search looks for network traffic on TCP/3389, the default port used by remote desktop. While remote desktop traffic is not uncommon on a network, it is usually associated with known hosts. This search allows for whitelisting both source and destination hosts to remove them from the output of the search so you can focus on the uncommon uses of remote desktop on your network.

SPL Difficulty

None

Data Availability

Bad

Journey

Stage 2

MITRE ATT&CK Tactics

Lateral Movement

MITRE ATT&CK Techniques

Remote Services

Remote Desktop Protocol

MITRE Threat Groups

APT1 APT3 APT39 APT41 Axiom

Blue Mockingbird Chimera Cobalt Group

Dragonfly 2.0 FIN10 FIN6 FIN8 Lazarus Group

Leviathan OilRig Patchwork Silence

Stolen Pencil TEMP.Veles Wizard Spider

menuPass

Kill Chain Phases

Actions On Objectives

Data Sources

Network Communication

> [Help](#)

Remote Desktop Network Traffic Help

To successfully implement this search you need to identify systems that commonly originate remote desktop traffic and that commonly receive remote desktop traffic. You can use the included support search "Identify Systems Creating Remote Desktop Traffic" to identify systems that originate the traffic and the search "Identify Systems Receiving Remote Desktop Traffic" to identify systems that receive a lot of remote desktop traffic. After identifying these systems, you will need to add the "commonrdpsource" or "commonrdpdestination" category to that system depending on the usage, using the Enterprise Security Assets and Identities framework. This can be done by adding an entry in the assets.csv file located in SA-IdentityManagement/lookups.

✓ [Search](#)


```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time) as lastTime from  
datamodel=Network_Traffic where All_Traffic.dest_port=3389 AND  
All_Traffic.dest_category!=common_rdp_destination AND All_Traffic.src_category!=common_rdp_source by  
All_Traffic.src All_Traffic.dest All_Traffic.dest_port | `drop_dm_object_name("All_Traffic")` |  
`security_content_ctime(firstTime)` | `security_content_ctime(lastTime)` |  
`remote_desktop_network_traffic_filter`
```

[Open in Search](#)

First Time Logon to New Server

Status

Bookmarked

App

Splunk Security Essentials

Description

Find users who logged into a new server for the first time.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

Use Case

Advanced Threat Detection, Compliance, Security Monitoring


Category

Lateral Movement, GDPR

Security Impact

By monitoring and alerting on first time log ins to a server, you are able to detect if/when an adversary is able to escalate permissions or add new accounts to AD, or to endpoints directly. This should be a priority particularly for critical infrastructure, high-value and mission critical assets or those systems containing sensitive data. In addition to external adversary, this type of behavior can also be indicative of a potential insider threat issue, where an employee is probing their access, or potentially testing new accounts they may have created for malicious purposes.

Alert Volume

Very High 

SPL Difficulty

Medium

Data Availability

Bad

Journey

Stage 1

MITRE ATT&CK Tactics

Lateral Movement

MITRE ATT&CK Techniques

Remote Services Remote Desktop Protocol Remote Desktop Protocol

MITRE Threat Groups

APT1 APT3 APT39 APT41 Axiom Blue Mockingbird Chimera Cobalt Group Dragonfly 2.0 FIN10 FIN6 FIN8 Lazarus Group Leviathan OilRig Patchwork Silence Stolen Pencil TEMP.Veles Wizard Spider menuPass

Kill Chain Phases

Installation Actions On Objectives

Data Sources

Windows Security Authentication

> [GDPR Relevance](#)

While not explicitly required for GDPR, this capability is often seen as a part of maintaining State of the Art Security and supports GDPR requirements.

> [How to Implement](#)

Implementation of this example (or any of the First Time Seen examples) is generally very simple.

- Validate that you have the right data onboarded, and that the fields you want to monitor are properly extracted.
- Save the search.

For most environments, these searches can be run once a day, often overnight, without worrying too much about a slow search. If you wish to run this search more frequently, or if this search is too slow for your environment, we recommend leveraging a lookup cache. For more on this, see the lookup cache dropdown below and select the sample item. A window will pop up telling you more about this feature.

> [Known False Positives](#)

This is a strictly behavioral search, so we define "false positive" slightly differently. Every time this fires, it will accurately reflect the first occurrence in the time period you're searching over (or for the lookup cache feature, the first occurrence over whatever time period you built the lookup). But while there are really no "false positives" in a traditional sense, there is definitely lots of noise.

You should not review these alerts directly (except for access to extremely sensitive system), but instead use them for context, or to aggregate risk (as mentioned under How To Respond).

> How To Respond

When this search returns values, initiate your incident response process and validate the user account accessing the specific system. Determine who the system owner is and contact them. If it is authorized, document it as such and by whom. If not, the user credentials may have been used by another party and additional investigation is warranted as a first time logon to a server that was never logged into before could suggest compromised credentials gaining access to other systems.

> Help

First Time Logon to New Server Help

This example leverages the Detect New Values search assistant. Our dataset is an anonymized collection of Windows Logon events. For this analysis, we are effectively grouping by username and system name, which will give us a row for each username+systemname combination. We check if the first time that has occurred was in the last day.

SPL for First Time Login to New Server

Demo Data

`Load_Sample_Log_Data("Windows Logon Activity")`	First we pull in our demo dataset.
stats earliest(_time) as earliest latest(_time) as latest by user, anonymized_ComputerName	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
eventstats max(latest) as maxlatest	Next we calculate the most recent value in our demo dataset
where earliest > relative_time(maxlatest, "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day of the end of our demo dataset.

Live Data

index=* source="*WinEventLog:Security" (4624 OR 4647 OR 4648 OR 551 OR 552 OR 540 OR 528 OR 4768 OR 4769 OR 4770 OR 4771 OR 4768 OR 4774 OR 4776 OR 4778 OR 4779 OR 672 OR 673 OR 674 OR 675 OR 678 OR 680 OR 682 OR 683) (EventCode=4624 OR EventCode=4647 OR EventCode=4648 OR EventCode=551 OR EventCode=552 OR EventCode=540 OR EventCode=528 OR EventCode=4768 OR EventCode=4769 OR EventCode=4770 OR EventCode=4771 OR EventCode=4768 OR EventCode=4774 OR EventCode=4776 OR EventCode=4778 OR EventCode=4779 OR EventCode=672 OR EventCode=673 OR EventCode=674 OR EventCode=675 OR EventCode=678 OR EventCode=680 OR EventCode=682 OR EventCode=683)	Here we start with our basic dataset of WinSecurity authentication logs. Notably, this technique of doing the value and then the field=value can bypass some quirks around field extractions, and make searches faster for very large datasets (though that's an area of active work, and it's less true every year).
stats earliest(_time) as earliest latest(_time) as latest by user, dest	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
where earliest > relative_time(now(), "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day.

Accelerated Data

tstats summariesonly=t allow_old_summaries=t count earliest(_time) AS earliest latest(_time) AS latest from datamodel=Authentication groupby _time span=1d, Authentication.user Authentication.dest	Here, tstats is pulling in one command a super-fast count per user, per system, per day of authentications.
rename "Authentication.dest" AS dest, "Authentication.user" as user	It is usually easiest to work with data model acceleration after we've renamed the fields to something a little friendlier.
stats earliest(_time) as earliest latest(_time) as latest by user, dest	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
where earliest > relative_time(now(), "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day.

Authentication Against a New Domain Controller

Status

Bookmarked

App


Splunk Security Essentials

Description

A common indicator for lateral movement is when a user starts logging into new domain controllers.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

<div><div>Use Case</div><div>Advanced Threat Detection</div><div>Category</div><div>Lateral Movement</div><div>Security Impact</div><div>Once an attacker gains access to a network either through a compromised asset or credentials, most will attempt to then move laterally within the network targeting critical infrastructure. As domain controllers provide the physical storage for the Active Directory Domain Services (AD DS) database, in addition to providing the services and data that allow enterprises to effectively manage endpoints (servers and workstations), users, and applications. If privileged access to a domain controller is obtained by a malicious user, that adversary can modify, corrupt, or destroy the AD DS database and, along with all of the systems and accounts that are managed by Active Directory. By monitoring both successful and unsuccessful authentication attempts organizations can identify anomalies such as time of day, frequency and other suspicious patters that may indicate compromised assets or credentials.</div><div>Alert Volume</div><div>Medium </div><div>SPL Difficulty</div><div>Medium</div></div>	<div><div>Data Availability</div><div>Bad</div><div>Journey</div><div>Stage 1</div><div>MITRE ATT&CK Tactics</div><div>Lateral Movement</div><div>MITRE ATT&CK Techniques</div><div>Remote Services</div><div>Kill Chain Phases</div><div>Installation</div><div>Data Sources</div><div>Windows Security</div></div>
--	--

> How to Implement

Implementation of this example (or any of the First Time Seen examples) is generally very simple.

- Validate that you have the right data onboarded, and that the fields you want to monitor are properly extracted.
- Save the search.

For most environments, these searches can be run once a day, often overnight, without worrying too much about a slow search. If you wish to run this search more frequently, or if this search is too slow for your environment, we recommend leveraging a lookup cache. For more on this, see the lookup cache dropdown below and select the sample item. A window will pop up telling you more about this feature.

> Known False Positives

This is a strictly behavioral search, so we define "false positive" slightly differently. Every time this fires, it will accurately reflect the first occurrence in the time period you're searching over (or for the lookup cache feature, the first occurrence over whatever time period you built the lookup). But while there are really no "false positives" in a traditional sense, there is definitely lots of noise.

You should not review these alerts directly (except for high sensitivity accounts), but instead use them for context, or to aggregate risk (as mentioned under How To Respond).

> How To Respond

When this search returns values, initiate your incident response process and identify the user account accessing the specific domain controller. Contact the user and system owner about this action. If it is authorized, document that this is authorized and by whom. If not, the user credentials may have been used by another party and additional investigation is warranted to determine that lateral movement is not occurring.

> Help

Authentication Against a New Domain Controller Help

This example leverages the Detect New Values search assistant. Our dataset is a anonymized collection of Windows domain controller logon events (Event ID 4776). For this analysis, we are effectively grouping by username and domain controller name, which will give us a row for each username+domaincontrollername combination. We check if the first time that has occurred was in the last day.

SPL for Authentication Against a New Domain Controller

Demo Data

<code> `Load_Sample_Log_Data("Domain Controller Logins (Event ID 4776)")`</code>	First we pull in our demo dataset.
<code> stats earliest(_time) as earliest latest(_time) as latest by user, anonymized_DomainControllerName</code>	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
<code> eventstats max(latest) as maxlatest</code>	Next we calculate the most recent value in our demo dataset
<code> where earliest > relative_time(maxlatest, "-1d@d")</code>	We end by seeing if the earliest time we've seen this value is within the last day of the end of our demo dataset.

Live Data

<code>source="*WinEventLog:Security" index=* 4776 EventCode=4776</code>	First we start with our basic dataset of WinSecurity logs with EventCode 4776, which will only originate from a domain controller.
<code> rename ComputerName as DomainControllerName</code>	We then rename the ComputerName to DomainController name for clarity
<code> table _time DomainControllerName user</code>	Then we use table to include just the fields we're apt to care about. (Technically we need to use table for this app because we show you the intermediate results, but in production you should drop this line because it will reduce search performance.)
<code> stats earliest(_time) as earliest latest(_time) as latest by user, anonymized_DomainControllerName</code>	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
<code> where earliest > relative_time(now(), "-1d@d")</code>	We end by seeing if the earliest time we've seen this value is within the last day.

Significant Increase in Interactive Logons

Status

Bookmarked

App

Splunk Security Essentials

Description

Typically non-admin users will only interactively log into one system per day. A user who starts login into many can indicate account compromise and lateral movement. (MITRE CAR Reference)

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

Use Case

Advanced Threat Detection

Category

Lateral Movement

Security Impact

By monitoring the number of interactively logged in users to assets, security teams can identify anomalies that may indicate the compromise of an asset or credentials. A spike in users on a particular asset could be an indicate that the asset was compromised and additional system level user accounts are being created for malicious purposes, or if they are valid credentials in AD that accounts have been compromised and the adversary is testing the accounts against a particular asset or groups of assets to test and or escalate privileges to gain deeper access to critical assets and infrastructure.

Alert Volume

Low (?)

SPL Difficulty

Medium

Data Availability

Bad

Journey

Stage 1

MITRE ATT&CK Tactics

Lateral MovementPrivilege EscalationPersistence

MITRE ATT&CK Techniques

Lateral MovementValid AccountsRemote Services

MITRE Threat Groups

APT18APT28APT33APT39APT41CarbanakChimeraDragonfly 2.0FIN10FIN4FIN5FIN6FIN8LeviathanNight DragonOilRigPittyTigerSandworm TeamSilenceSoft CellSuckflyTEMP.VelesThreat Group-3390Wizard SpidermenuPass

Kill Chain Phases

InstallationActions On Objectives

Data Sources

Windows Security

> [How to Implement](#)

Implementation of this example (or any of the Time Series Spike / Standard Deviation examples) is generally pretty simple.

- Validate that you have the right data onboarded, and that the fields you want to monitor are properly extracted. If the base search you see in the box below returns results.
- Save the search to run over a long period of time (recommended: at least 30 days).

For most environments, these searches can be run once a day, often overnight, without worrying too much about a slow search. If you wish to run this search more frequently, or if this search is too slow for your environment, we recommend using a summary index that first aggregates the data. We will have documentation for this process shortly, but for now you can look at Summary Indexing descriptions such as [here](#) and [here](#).

> [Known False Positives](#)

This is a strictly behavioral search, so we define "false positive" slightly differently. Every time this fires, it will accurately a spike in the number we're monitoring... it's nearly impossible for the math to lie. But while there are really no "false positives" in a traditional sense, there is definitely lots of noise.

How you handle these alerts depends on where you set the standard deviation. If you set a low standard deviation (2 or 3), you are likely to get a lot of events that are useful only for contextual information. If you set a high standard deviation (6 or 10), the amount of noise can be reduced enough to send an alert directly to analysts.

> How To Respond

When this search returns values, initiate your incident response process and capture the event times, the user account and systems, process and other pertinent information. Contact the owners of the systems. If it is authorized behavior, document that this is authorized and by whom. If not, the user credentials may have been used by another party and additional investigation is warranted.

> Help

Significant Increase in Interactive Logons Help

This example leverages the Detect Spikes (standard deviation) search assistant. Our dataset is an anonymized collection of Windows Logon events, filtered to interactive logon types (Local: 2, RemoteInteractive: 10, Cached Local: 11). For this analysis, we are tracking the number of unique hosts the user has interactively logged into per day 'dc(host) by user _time'. Then we calculate the average, standard deviation, and the most recent value, and filter out any users where the most recent is within the configurable number of standard deviations from average.

SPL for Significant Increase in Interactive Logons

Demo Data

`Load_Sample_Log_Data("Interactive Logins")`	First we pull in our demo dataset.
bucket _time span=1d	Bucket (aliased to bin) allows us to group events based on _time, effectively flattening the actual _time value to the same day.
stats dc(dest) as count by _time user eventstats max(_time) as maxtime stats count as num_data_samples max(eval(if(_time >= relative_time(maxtime, "-1d@d"), 'count', null))) as count avg(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as avg stdev(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as stdev by user	Finally, we can count and aggregate per user, per day.
eval lowerBound=(avg-stdev*2), upperBound=(avg+stdev*2)	calculate the mean, standard deviation and most recent value
where 'count' > upperBound AND num_data_samples >=7	calculate the bounds as a multiple of the standard deviation

Live Data

index=* source="*WinEventLog:Security" Logon_Type=2 OR Logon_Type=10 OR Logon_Type=11 Logon Type TaskCategory=Logon Audit Success	First we pull in our dataset of Windows Authentication specifying Interactive logon types.
bucket _time span=1d	Bucket (aliased to bin) allows us to group events based on _time, effectively flattening the actual _time value to the same day.
stats dc(dest) as count by _time user stats count as num_data_samples max(eval(if(_time >= relative_time(maxtime, "-1d@d"), 'count', null))) as count avg(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as avg stdev(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as stdev by user	Finally, we can count and aggregate per user, per day.
eval lowerBound=(avg-stdev*2), upperBound=(avg+stdev*2)	calculate the mean, standard deviation and most recent value
where 'count' > upperBound AND num_data_samples >=7	calculate the bounds as a multiple of the standard deviation

Increase in # of Hosts Logged into

Status

Bookmarked

App

Splunk Security Essentials

Description

Find users who log into more hosts than they typically do.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

<div><div>Use Case</div><div>Advanced Threat Detection</div><div>Category</div><div>Lateral Movement</div><div>Alert Volume</div><div>Low (?)</div><div>SPL Difficulty</div><div>Hard</div></div>	<div><div>Data Availability</div><div>Bad</div><div>Journey</div><div>Stage 1</div><div>MITRE ATT&CK Tactics</div><div>Lateral Movement</div><div>MITRE ATT&CK Techniques</div><div>Remote Services</div><div>Kill Chain Phases</div><div>InstallationActions On Objectives</div><div>Data Sources</div><div>Windows SecurityAuthentication</div></div>
---	---

> How to Implement

Implementation of this example (or any of the Time Series Spike / Standard Deviation examples) is generally pretty simple.

- Validate that you have the right data onboarded, and that the fields you want to monitor are properly extracted. If the base search you see in the box below returns results.
- Save the search to run over a long period of time (recommended: at least 30 days).

For most environments, these searches can be run once a day, often overnight, without worrying too much about a slow search. If you wish to run this search more frequently, or if this search is too slow for your environment, we recommend using a summary index that first aggregates the data. We will have documentation for this process shortly, but for now you can look at Summary Indexing descriptions such as [here](#) and [here](#).

> Known False Positives

This is a strictly behavioral search, so we define "false positive" slightly differently. Every time this fires, it will accurately a spike in the number we're monitoring... it's nearly impossible for the math to lie. But while there are really no "false positives" in a traditional sense, there is definitely lots of noise.

How you handle these alerts depends on where you set the standard deviation. If you set a low standard deviation (2 or 3), you are likely to get a lot of events that are useful only for contextual information. If you set a high standard deviation (6 or 10), the amount of noise can be reduced enough to send an alert directly to analysts.

> How To Respond

When this search returns values, initiate your incident response process and identify the account name associated with the suspicious domain. Establish the time the event occurred and from what system the login attempt occurred. Contact the user and system owners to determine if it is authorized, and if so document it as such and by whom. If not, the user credentials may have been used by another party and additional investigation is warranted because compromised credentials may be used to gain access to a broad set of systems.

> Help

Increase in # of Hosts Logged into Help

This example leverages the Detect Spikes (standard deviation) search assistant. Our dataset is an anonymized collection of Windows Logon events (of all kinds). For this analysis, we are tracking the number of unique hosts the user has logged into per day 'dc(host) by user _time'. Then we calculate the average, standard deviation, and the most recent value, and filter out any users where the most recent is within the configurable number of standard deviations from average.

SPL for Increase in # of Hosts Logged into

Demo Data

`Load_Sample_Log_Data("Windows Logon Activity")`	First we pull in our demo dataset.
bucket _time span=1d	Bucket (aliased to bin) allows us to group events based on _time, effectively flattening the actual _time value to the same day.
stats dc(anonymized_ComputerName) as count by user _time eventstats max(_time) as maxtime stats count as num_data_samples max(eval(if(_time >= relative_time(maxtime, "-1d@d"), 'count', null))) as count avg(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as avg stdev(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as stdev by user	Finally, we can count and aggregate per user, per day.
eval lowerBound=(avg-stdev*2), upperBound=(avg+stdev*2)	calculate the mean, standard deviation and most recent value
where 'count' > upperBound AND num_data_samples >=7	calculate the bounds as a multiple of the standard deviation

Live Data

index=* source="*WinEventLog:Security" (4624 OR 4647 OR 4648 OR 551 OR 552 OR 540 OR 528 OR 4768 OR 4769 OR 4770 OR 4771 OR 4768 OR 4774 OR 4776 OR 4778 OR 4779 OR 672 OR 673 OR 674 OR 675 OR 678 OR 680 OR 682 OR 683)	First we pull in our Windows Security log dataset, filtering to logon Event IDs.
bucket _time span=1d	Bucket (aliased to bin) allows us to group events based on _time, effectively flattening the actual _time value to the same day.
stats dc(host) as count by user _time stats count as num_data_samples max(eval(if(_time >= relative_time(maxtime, "-1d@d"), 'count', null))) as count avg(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as avg stdev(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as stdev by user	Finally, we can count and aggregate per user, per day.
eval lowerBound=(avg-stdev*2), upperBound=(avg+stdev*2)	calculate the mean, standard deviation and most recent value
where 'count' > upperBound AND num_data_samples >=7	calculate the bounds as a multiple of the standard deviation

Accelerated with Data Models

tstats summariesonly=t allow_old_summaries=t dc(Authentication.dest) as count from datamodel=Authentication groupby _time span=1d, Authentication.user	Here, tstats is pulling in one command a super-fast count per user, per day.
rename "Authentication.user" as user stats count as num_data_samples max(eval(if(_time >= relative_time(maxtime, "-1d@d"), 'count', null))) as count avg(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as avg stdev(eval(if(_time < relative_time(maxtime, "-1d@d"), 'count', null))) as stdev by user	I usually like to rename data model fields after my tstats, as it becomes much easier to use.
eval lowerBound=(avg-stdev*2), upperBound=(avg+stdev*2)	calculate the mean, standard deviation and most recent value
where 'count' > upperBound AND num_data_samples >=7	calculate the bounds as a multiple of the standard deviation

New AD Domain Detected

Status

Bookmarked

App

Splunk Security Essentials

Description

New AD domain names in your normal domain controller logs are a symptom of many Pass the Hash tools. While some of the latest don't produce these artifacts, this remains a very valuable detection mechanism.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

<div><div>Use Case</div><div>Advanced Threat Detection, Compliance</div><div>Category</div><div>Lateral Movement</div><div>Security Impact</div><div>In Windows logs, the domain name is often reported when it's not explicitly required for the authentication. Under normal operation, the domain name reported will be totally normal, but when someone is intentionally modifying authentication (such as with Pass the Hash), you can see incorrect, or empty domains. Pass the Hash is used by attackers to move laterally within the organization, connecting to new servers. While not all Pass the Hash techniques will demonstrate this vulnerability, tracking new domains in your Windows logs is very valuable.</div><div>Alert Volume</div><div>Low (?)</div><div>SPL Difficulty</div><div>Medium</div></div>	<div><div>Data Availability</div><div>Bad</div><div>Journey</div><div>Stage 1</div><div>MITRE ATT&CK Tactics</div><div>Lateral Movement</div><div>MITRE ATT&CK Techniques</div><div>Pass the Hash</div><div>Pass the Hash</div><div>MITRE Threat Groups</div><div>APT1</div><div>APT28</div><div>APT32</div><div>Night Dragon</div><div>Soft Cell</div><div>Kill Chain Phases</div><div>Installation</div><div>Data Sources</div><div>Windows Security</div></div>
--	--

> How to Implement

Implementation of this example (or any of the First Time Seen examples) is generally very simple.

- Validate that you have the right data onboarded, and that the fields you want to monitor are properly extracted.
- Save the search.

For most environments, these searches can be run once a day, often overnight, without worrying too much about a slow search. If you wish to run this search more frequently, or if this search is too slow for your environment, we recommend leveraging a lookup cache. For more on this, see the lookup cache dropdown below and select the sample item. A window will pop up telling you more about this feature.

> Known False Positives

This is a strictly behavioral search, so we define "false positive" slightly differently. Every time this fires, it will accurately reflect the first occurrence in the time period you're searching over (or for the lookup cache feature, the first occurrence over whatever time period you built the lookup). But while there are really no "false positives" in a traditional sense, there is definitely lots of noise.

This search is designed to find older versions of Mimikatz (or other tools with similar techniques), and is not known to have any other false positives.

> How To Respond

When this search returns values, initiate your incident response process and identify the account name associated with the new domain. Determine at what time the event occurred and from what system the login attempt occurred from. Contact the user and system owner(s) to determine if it is authorized, and document that this is authorized and by whom. If not, the user credentials may have been used by another party and additional investigation is warranted to determine if a pass the hash attack has been attempted and generated this new domain in the event log.

> [Help](#)

New AD Domain Detected Help

This example leverages the Detect New Values search assistant. Our dataset is a anonymized collection of Windows logon events from a domain controller. For this analysis, we are looking for the earliest time for a domain name. We check if the first time that domain was seen was in the last day.

SPL for New AD Domain Detected

Demo Data

`Load_Sample_Log_Data(Example Pass The Hash (Legacy))`	First we pull in our demo dataset.
stats earliest(_time) as earliest latest(_time) as latest by Account_Domain, EventCode	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
eventstats max(latest) as maxlatest	Next we calculate the most recent value in our demo dataset
where earliest > relative_time(maxlatest, "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day of the end of our demo dataset.

Live Data

index=* source="*WinEventLog:Security" EventCode=4624 Authentication_Package=NTLM Type=Information Account_Name!="ANONYMOUS LOGON" Logon_Type=3 Logon_Process="NtLmSsp" NOT Security_ID!="NULL SID" Key_Length=0	This string will look in your Windows Security logs for the specific signature of Mimikatz (prior to 2017).
stats earliest(_time) as earliest latest(_time) as latest by Account_Domain, EventCode	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
where earliest > relative_time(now(), "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day.

Remote PowerShell Launches

Status

Bookmarked

App

Splunk Security Essentials

Description

It's unusual for new users to remotely launch PowerShell on another system. This will track the first time per user + host combination that powershell is remotely started.

(for most companies)

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

<div><div>Use Case</div><div>Advanced Threat Detection</div><div>Category</div><div>Lateral Movement</div><div>Security Impact</div><div>Remote execution of PowerShell is something that should rarely occur if at all within an network, and if it does should be associated with known executables or users. Therefore, unique instances of user and host combinations should be monitored and alerted upon for further investigation by security teams. This type of activity could be an indicator of compromised assets, user credentials, or a potential insider threat scenario.</div><div>Alert Volume</div><div>Low (?)</div><div>SPL Difficulty</div><div>Medium</div></div>	<div><div>Data Availability</div><div>Bad</div><div>Journey</div><div>Stage 1</div><div>MITRE ATT&CK Tactics</div><div>Lateral Movement</div><div>MITRE ATT&CK Techniques</div><div>Remote Services</div><div>Kill Chain Phases</div><div>Installation</div><div>Data Sources</div><div>Windows Security</div><div>Endpoint Detection and Response</div></div>
---	--

> How to Implement

Implementation of this example (or any of the First Time Seen examples) is generally very simple.

- Validate that you have the right data onboarded, and that the fields you want to monitor are properly extracted.
- Save the search.

For most environments, these searches can be run once a day, often overnight, without worrying too much about a slow search. If you wish to run this search more frequently, or if this search is too slow for your environment, we recommend leveraging a lookup cache. For more on this, see the lookup cache dropdown below and select the sample item. A window will pop up telling you more about this feature.

> Known False Positives

This is a strictly behavioral search, so we define "false positive" slightly differently. Every time this fires, it will accurately reflect the first occurrence in the time period you're searching over (or for the lookup cache feature, the first occurrence over whatever time period you built the lookup). But while there are really no "false positives" in a traditional sense, there is definitely lots of noise.

This type of event should only occur for remote management, or potentially some systems management tools. It would be recommended to filter out the users who typically perform these actions (or alternatively you could alter the search to look for user + sourcetype instead of user + host, so that you will be alerted the first time a user takes this action, instead of the first time a user takes it on a particular host).

> How To Respond

When this search returns values, initiate your incident response process and identify the user and system responsible for initiating this behavior. Determine the time of the event and what process and associated parent processes where triggered. Contact the user and system owner to determine if it is authorized, and document that this is authorized and by whom. If not, the user credentials may have been used by another party and additional investigation is warranted.

> [Help](#)

Remote PowerShell Launches Help

This example leverages the Detect New Values search assistant. Our dataset is a anonymized collection of process launch events (Event ID 4688) filtered for the Image and ParentImage that show up when Powershell is remotely triggered. We check the first time that's occurred per host, and then alert if that was in the last day.

SPL for Remote PowerShell Launches

Demo Data

`Load_Sample_Log_Data("Sysmon Process Launch Logs")`	First we pull in our demo dataset.
search ParentImage=*\\svchost.exe Image=*\\wsmprovhost.exe	Then we filter for where the parent process is svchost.exe and the actual process is wsmprovhost.exe, as that is what will show up for a remote Powershell launch.
eval user="Unknown"	For our demo data, we don't have a user, so we just default it as "unknown."
stats earliest (_time) as earliest latest (_time) as latest by user, host	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
eventstats max (latest) as maxlatest	Next we calculate the most recent value in our demo dataset
where earliest > relative_time (maxlatest, "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day of the end of our demo dataset.

Live Data

index=* source="*WinEventLog:Security" EventCode=4688 wsmprovhost svchost ParentImage=*\\svchost.exe Image=*\\wsmprovhost.exe	First we pull in our dataset of Windows process launch logs filtered to our common Windows executables, care of EventID 4688 documented in this app. Any other EDR solution giving process launch logs will suffice here, as well. Notably, this technique of doing the value and then the field=value can bypass some quirks around field extractions, and make searches faster for very large datasets (though that's an area of active work, and it's less true every year).
table _time user host EventCode New_Process_Name	Then we use table to include just the fields we're apt to care about. (Technically we need to use table for this app because we show you the intermediate results, but in production you should drop this line because it will reduce search performance.)
stats earliest (_time) as earliest latest (_time) as latest by user, dest	Here we use the stats command to calculate what the earliest and the latest time is that we have seen this combination of fields.
where earliest > relative_time (now(), "-1d@d")	We end by seeing if the earliest time we've seen this value is within the last day.

Detect Lateral Movement With WMI

Status

Bookmarked

App

Splunk Security Essentials

Description

This use case looks for WMI being used for lateral movement.

Content Mapping

This content is not mapped to any local saved search. [Add mapping](#)

Use Case

Advanced Threat Detection

Category

Lateral Movement, Ransomware

Alert Volume

Low (?)

SPL Difficulty

Basic

Data Availability

Bad

Journey

Stage 3

MITRE ATT&CK Tactics

Lateral MovementExecution

MITRE ATT&CK Techniques

Remote ServicesWindows Management Instrumentation

MITRE Threat Groups

APT29APT32APT41Blue MockingbirdChimeraDeep PandaFIN6FIN8FrankensteinLazarus GroupLeviathanMuddyWaterOilRigSoft CellStealth FalconThreat Group-3390Wizard SpidermenuPass

Kill Chain Phases

InstallationActions On Objectives

Data Sources

Windows SecurityEndpoint Detection and Response

> How to Implement

This use case requires Sysmon to be installed on the endpoints you wish to monitor and the Sysmon add-on installed on your forwarders and search heads.

> Known False Positives

None at the moment

> How To Respond

When this search fires, you will want to start your incident response process and investigate the actions taken by this process.

> Help

Detect Lateral Movement With WMI Help

This use case looks for WMI being used for lateral movement.

SPL for Detect Lateral Movement With WMI

Demo Data

inputlookup UC_wmi	First we load our basic demo data
search EventCode=1 Image=*wmic* CommandLine=*node* CommandLine="*process call create"	Next we look for any instances of WMIC (Windows Management Instrumentation Command-line) being launched (EventCode 1 indicates a process launch), and filter to make sure our suspicious fields are in the CommandLine string.
table _time host Image CommandLine	Then we put the data into a table because that's the easiest thing to use.

Live Data

index=* sourcetype=XmlWinEventLog:Microsoft-Windows-Sysmon/Operational EventCode=1 Image=*wmic* CommandLine=*node* CommandLine="*process call create"	First we load our Sysmon EDR (though any other process launch logs with the full command line would suffice) data. We look for any instances of WMIC (Windows Management Instrumentation Command-line) being launched (EventCode 1 indicates a process launch), and filter to make sure our suspicious fields are in the CommandLine string.
table _time host Image CommandLine	Then we put the data into a table because that's the easiest thing to use.