

# Hunting in the Dark

## HTCIA 2015

Ryan Kazanciyan, Chief Security Architect

September 2, 2015



# whoami

- Chief Security Architect for Tanium
- Former Technical Director & incident response leader at Mandiant
- Instructor for Black Hat, LEO
- Contributing author: “Incident Response & Computer Forensics, 3rd Ed.” (2014)



# Motivations

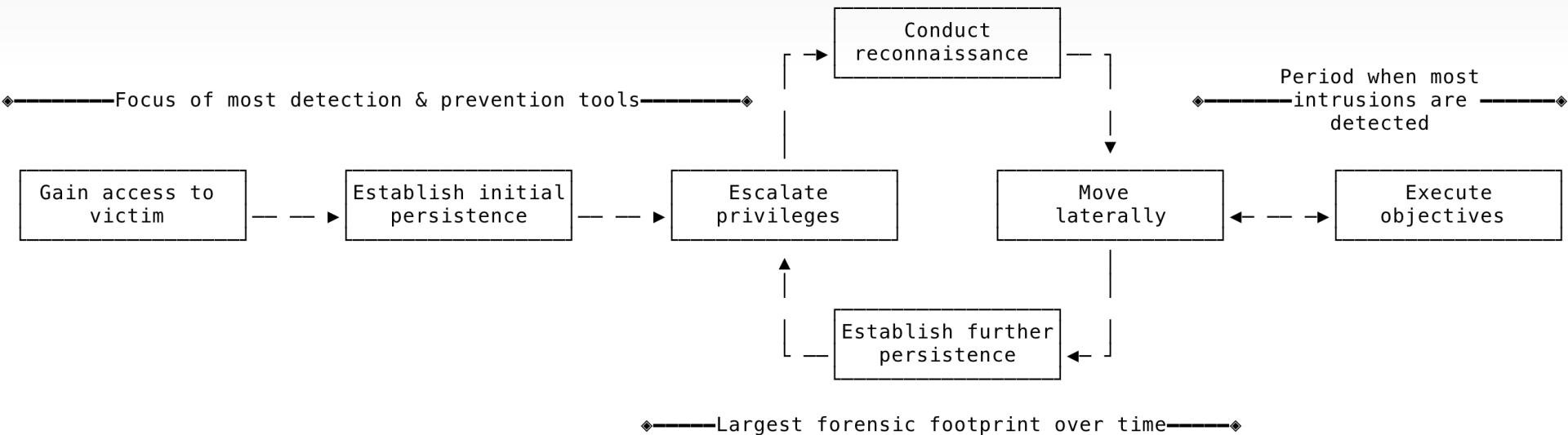
- Investigating at **enterprise-scale**
- Building **repeatable** analysis tasks for both **proactive** and **reactive** hunting
- Finding evidence of compromise with **minimal leads**
- **Fully scoping** an incident as efficiently as possible

# Areas of focus

- Endpoint-centric approach
- Evidence available **by default** on Windows Vista / Server 2008 and later
- Techniques – not specific tools – for search, stacking, outlier analysis, and data reduction

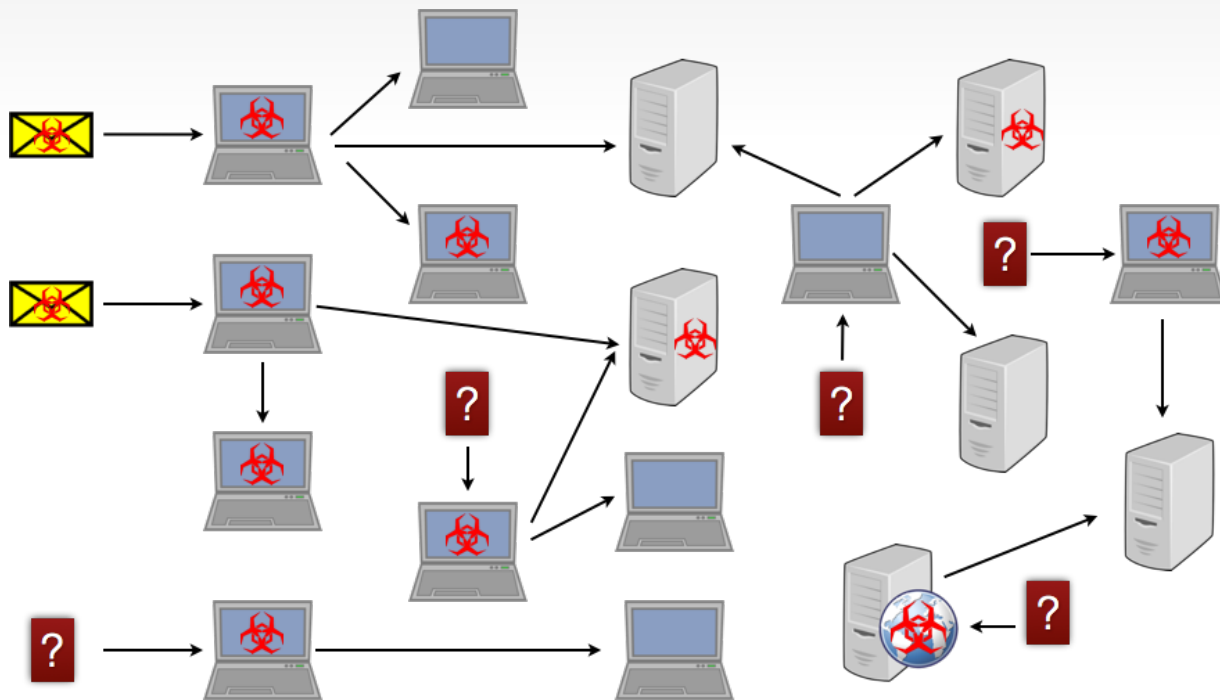
The forensic “footprint” of an incident  
is primarily shaped and defined  
by *post-compromise* activity

# When are targeted attacks detected?



Why is “hunting” difficult...  
especially when investigating  
tens or hundreds of thousands  
of systems?

# Incidents are non-linear

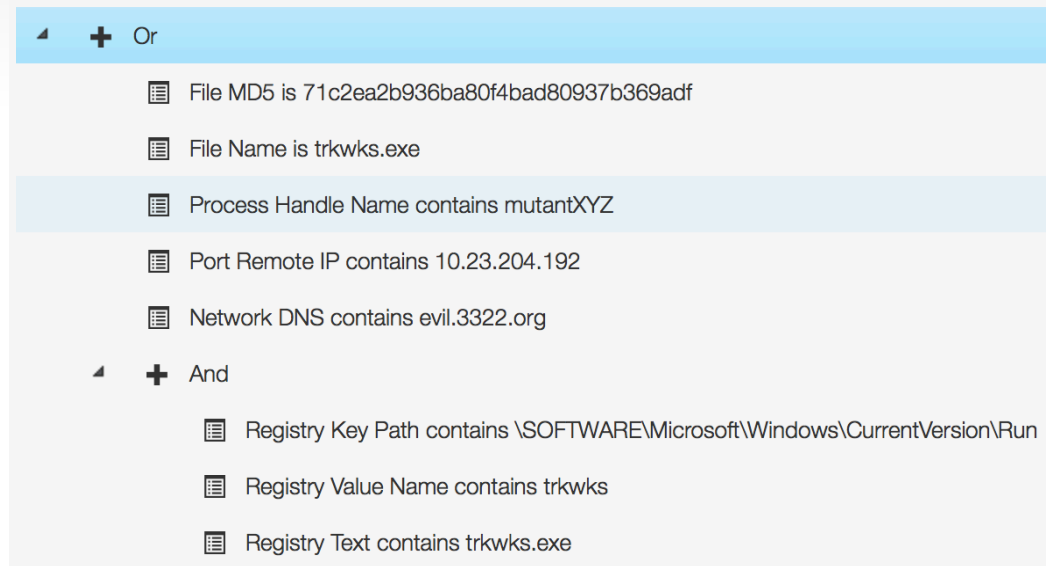


# Why?

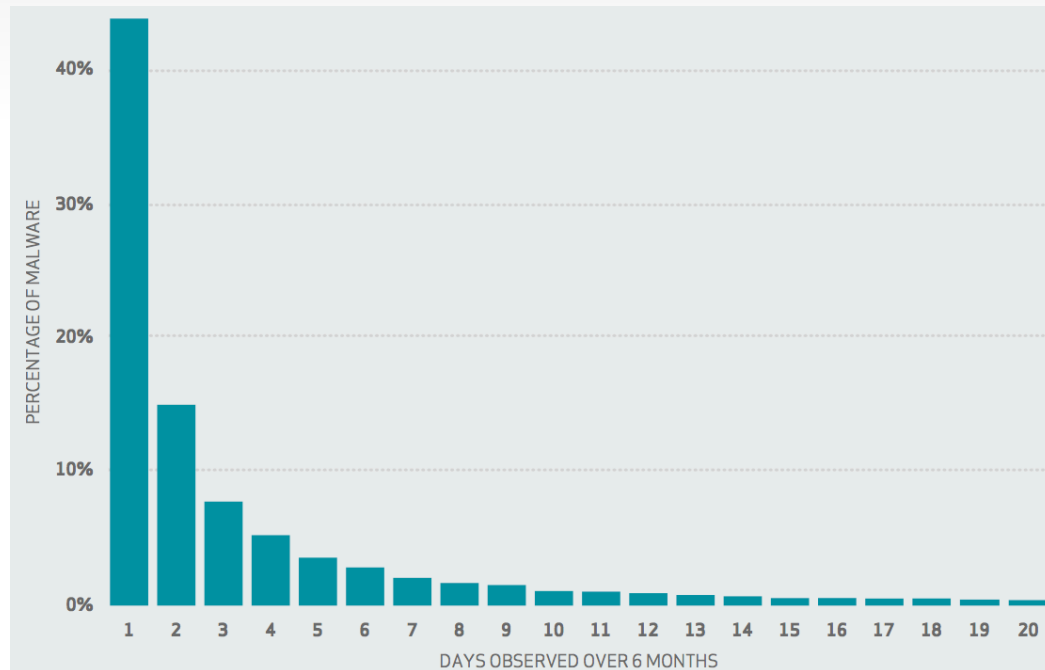
- Targeted intrusions often begin with opportunistic compromises
- Attackers can be erratic & unpredictable when operating in an unfamiliar environment
- Evidence is often incomplete or insufficient

# IOCs can be brittle

- Easy to build high-fidelity IOCs (may yield high false-negatives)
- Hard to build robust IOCs (may yield higher false-positives)
- Large environments = more noise = more false positives



# Lifespan of malware-specific IOCs



Source: Verizon DBIR 2015

# 95%

OF MALWARE TYPES  
SHOWED UP FOR LESS  
THAN A MONTH, AND  
FOUR OUT OF FIVE  
DIDN'T LAST BEYOND  
A WEEK.

# Endpoints are noisy

“How many unique PE files (EXEs, DLLs, drivers) executed or loaded on servers and end-user systems?”

- Different OS versions and add-ons
- User-installed applications
- Random / GUID file names & paths
- Temporary artifacts of software installers
- Updates & patches

# Endpoints are noisy

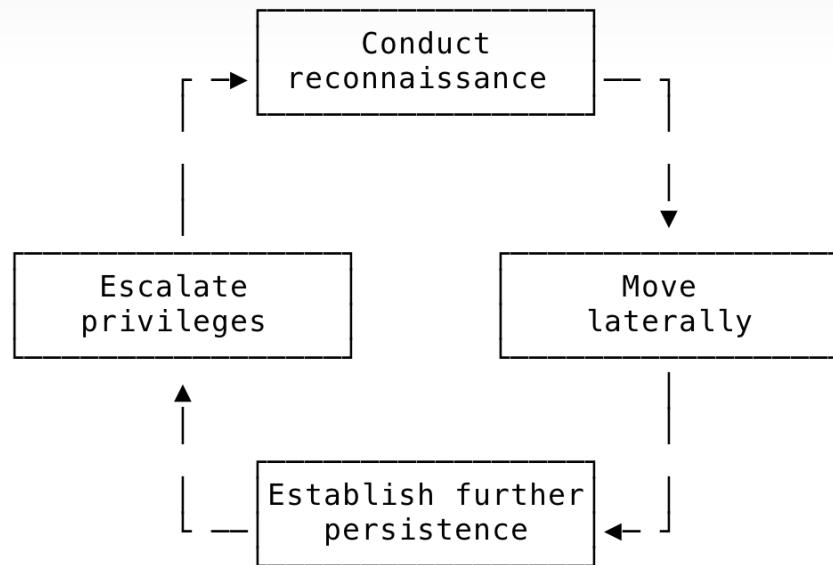
“How common is logon activity by privileged accounts across end-user systems and servers,?”

- Automated maintenance and administration scripts
- Troubleshooting tasks and tools
- Service and application accounts
- Remnants of legacy IT operations
- Misunderstood native OS behavior

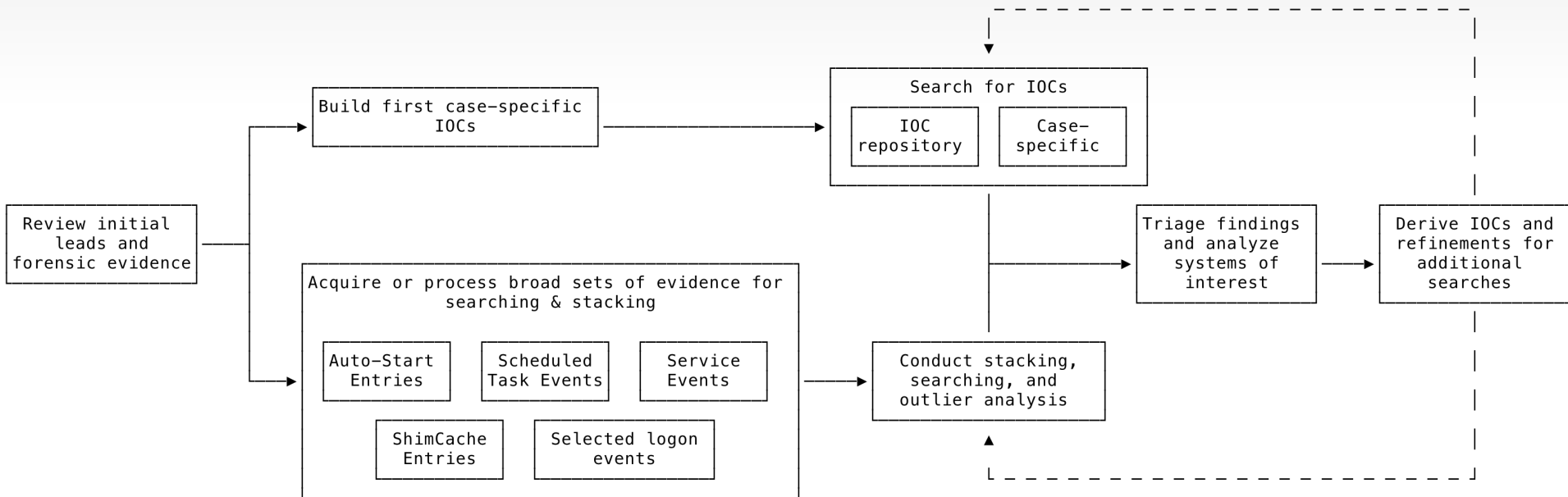
What analysis techniques can  
overcome the limitations of  
searching for “known bad”...  
in large and intrinsically noisy  
environments

# Focusing on the core of an intrusion

- What fundamental techniques are common across nearly all intrusions?
- What evidence do they leave behind?



# Overall methodology



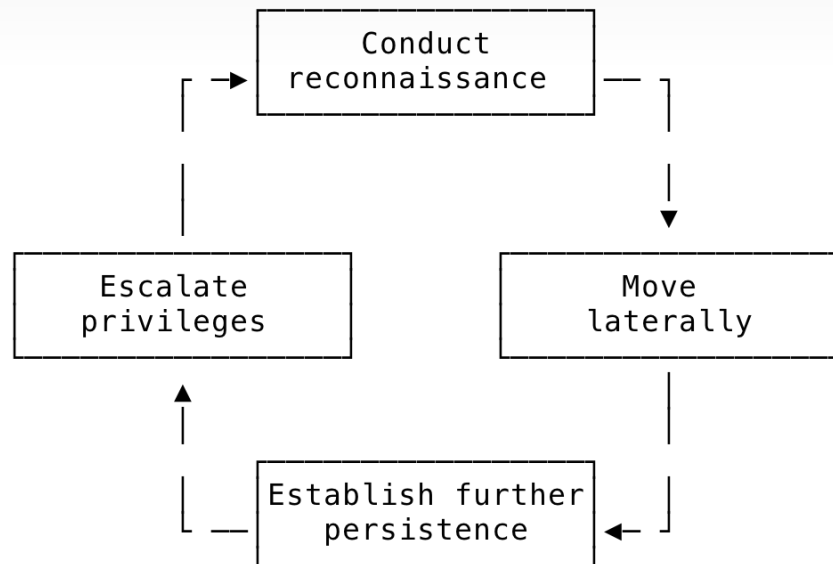
# Focus for this presentation

- Hunting for rogue scheduled tasks
- Working with ShimCache evidence at-scale
- Analyzing service events
- Bonus round!

# Hunting for rogue scheduled tasks

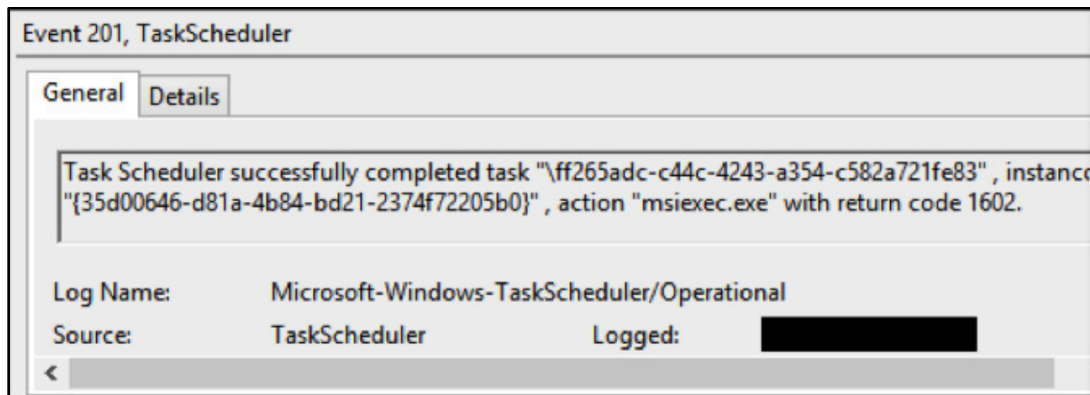
# Why scheduled tasks?

- Move laterally
  - Execute command on remote system
- Escalate privileges
  - Easy way for Administrator to run command as SYSTEM
- Establish persistence
  - Recurring tasks



# Example: Duqu 2.0

“In addition to creating services to infect other computers in the LAN, attackers can also use the Task Scheduler to start ‘msiexec.exe’ remotely. The usage of Task Scheduler during Duqu infections for lateral movement was also observed with the 2011 version...”



Source:

[https://securelist.com/files/2015/06/The\\_Mystery\\_of\\_Duqu\\_2\\_0\\_a\\_sophisticated\\_cyber\\_espionage\\_actor\\_returns.pdf](https://securelist.com/files/2015/06/The_Mystery_of_Duqu_2_0_a_sophisticated_cyber_espionage_actor_returns.pdf)

# Task Scheduler Operational Log

- Windows Vista, Server 2008 and later
- Microsoft-Windows-TaskScheduler/Operational.evtx
- Events to harvest
  - 106 – Task Registered
  - 129 – Created Task Process
  - 200 – Action Started
  - 201 – Action Completed

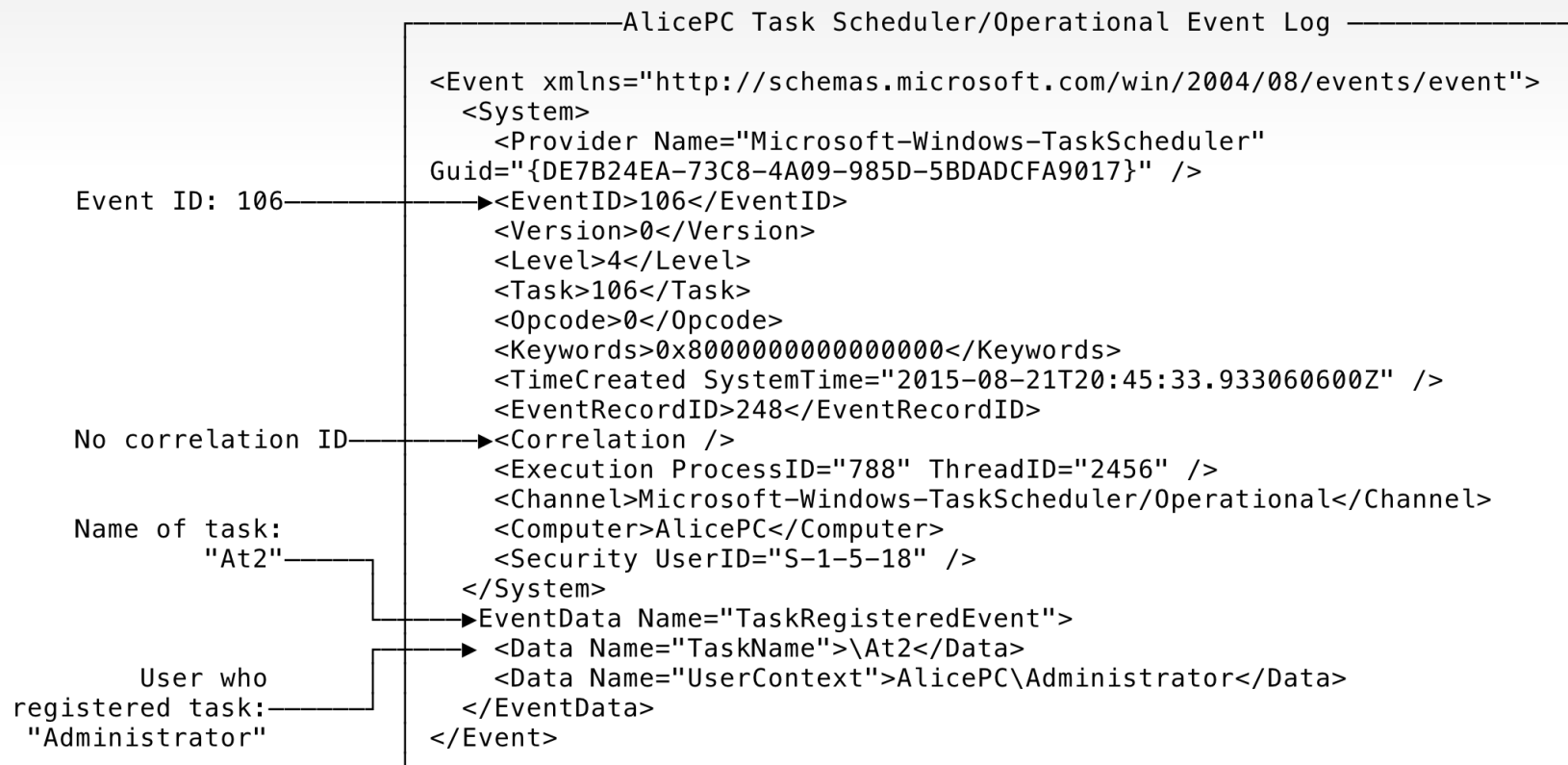
# Finding malicious **unnamed** tasks

Attacker Shell

```
C:\temp>at \\AlicePC 16:46 c:\dump.exe
```

- Initial filter:
  - TaskName contains \At
- Key event fields:
  - UserContext
  - ActionName

# Task registration event



# Task action start event

AlicePC Task Scheduler/Operational Event Log

```
<Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
  <System>
    <Provider Name="Microsoft-Windows-TaskScheduler"
      Guid="{DE7B24EA-73C8-4A09-985D-5BDADCFA9017}" />
    <EventID>200</EventID>
    <Version>0</Version>
    <Level>4</Level>
    <Task>200</Task>
    <Opcode>1</Opcode>
    <Keywords>0x8000000000000000</Keywords>
    <TimeCreated SystemTime="2015-08-21T20:45:59.999490900Z" />
    <EventRecordID>253</EventRecordID>
    <Correlation ActivityID="{0DEB2232-85CB-4780-AD9D-3E97488F703C}" />
    <Execution ProcessID="1540" ThreadID="1100" />
    <Channel>Microsoft-Windows-TaskScheduler/Operational</Channel>
    <Computer>AlicePC</Computer>
    <Security UserID="S-1-5-18" />
  </System>
  <EventData Name="ActionStart">
    <Data Name="TaskName">\At2</Data>
    <Data Name="ActionName">C:\dump.exe</Data>
    <Data
      Name="TaskInstanceId">{0DEB2232-85CB-4780-AD9D-3E97488F703C}</Data>
    </EventData>
</Event>
```

Event ID: 200

Correlation  
Activity ID equal  
to Task Action  
Instance ID

Path to what executed

# Tasks to run batch scripts

Attacker Shell

```
C:\temp>at \\AlicePC 16:46 c:\test.bat
```

"Path" contains "cmd.exe" –  
better to stack on ActionName  
from EID 200

EID 200 (Action Started)

```
...  
<EventData Name="ActionStart">  
  <Data Name="TaskName">\At20</Data>  
  <Data Name="ActionName">c:\test.bat</Data>  
  <Data Name="TaskInstanceId">{C8312018-4B96-44BA-90B1-CB875D3BA15B}</Data>  
</EventData>  
...
```

EID 129 (Created Task Process)

```
...  
<EventData Name="CreatedTaskProcess">  
  <Data Name="TaskName">\At20</Data>  
  <Data Name="Path">C:\Windows\SYSTEM32\cmd.exe</Data>  
  <Data Name="ProcessID">3916</Data>  
  <Data Name="Priority">7</Data>  
</EventData>  
...
```

# Stack and search workflow

Stack by User

```
User, Count
...
CORP\ITAdmin,9761
CORP\InfosecAdmin,2240
CORP\DomainAdmin,77
CORP\EBachman,5
AlicePC\Administrator,3
ACMEDC01\Administrator,2
CORP\RHendricks,1
...
```

Stack by ActionName

```
Path, Count
...
C:\Program Files (x86)\Google\Update\GoogleUpdate.exe,1862
c:\temp\patchFix.exe,310
...
c:\test.bat,15
c:\dump.bat,7
c:\temp\1.bat,3
...
```

Summarized "At" Task Events

```
Hostname, Timestamp, EID, UserContext, ActionName
...
AlicePC,2015-08-15 16:44:33,106,AlicePC\Administrator,N/A
AlicePC,2015-08-15 16:45:00,200,N/A,c:\test.bat
...
BobPC,2015-08-15 16:47:18,106,CORP\ITAdmin,N/A
BobPC,2015-08-15 16:49:00,200,N/A,c:\dump.exe
...
AcmeDC01,2015-06-15 09:44:00,200,N/A,c:\temp\patchFix.exe
AcmeDB02,2015-06-15 09:51:00,200,N/A,c:\temp\patchFix.exe
AcmeEX02,2015-06-15 09:37:00,200,N/A,c:\temp\patchFix.exe
...
```

Remember: must match EID 106 with EID 200 events  
by proximity and sequencing – no correlation ID!

# Blind spot: Tasks that run “cmd.exe /c [args]”

Attacker Shell

```
C:\temp>at \\AlicePC 16:46 cmd.exe /c "c:\test.bat"
```

EID 200 (Action Started)

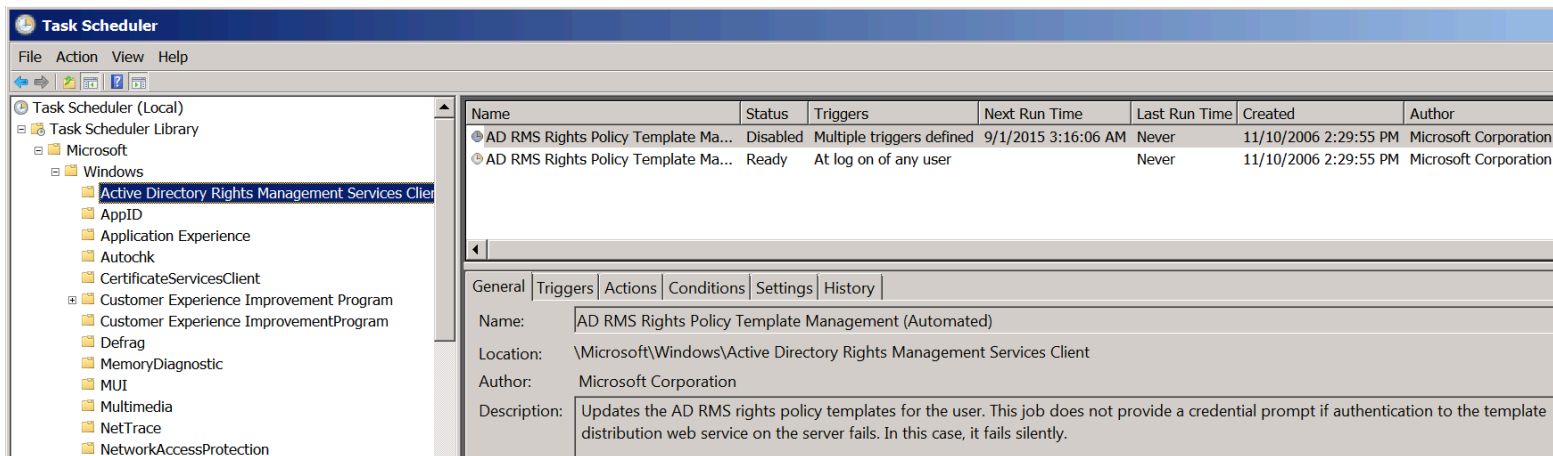
```
...
<EventData Name="ActionStart">
  <Data Name="TaskName">\At21</Data>
  <Data Name="ActionName">cmd.exe</Data>
  <Data Name="TaskInstanceId">{CA265597-3F0D-44E4-9BE1-DC307F34FC14}</Data>
</EventData>
...
```

EID 129 (Created Task Process)

```
...
<EventData Name="CreatedTaskProcess">
  <Data Name="TaskName">\At21</Data>
  <Data Name="Path">cmd.exe</Data>
  <Data Name="ProcessID">2018</Data>
  <Data Name="Priority">7</Data>
</EventData>
...
```

# Finding malicious **named** tasks

- User-created tasks (schtasks.exe) stored in:  
`%SYSTEMROOT%\system32\Tasks`
- Built-in Windows tasks stored in:  
`%SYSTEMROOT%\system32\Tasks\Microsoft`



# Named scheduled task events

Attacker Shell

```
C:\> schtasks /Create /TN TestJob /TR C:\dump.exe /ST 18:15 /SC ONCE
```

EID 106

```
<EventData Name="TaskRegisteredEvent">  
  <Data Name="TaskName">TestJob</Data>  
  <Data Name="UserContext">AlicePC\Administrator</Data>  
</EventData>
```

EID 200

```
<EventData Name="ActionStart">  
  <Data Name="TaskName">TestJob</Data>  
  <Data Name="ActionName">C:\dump.exe</Data>  
  <Data Name="TaskInstanceId">{B9BC3E1A-E137-409C-828F-A607504C94C6}</Data>  
</EventData>
```

EID 129

```
<EventData Name="CreatedTaskProcess">  
  <Data Name="TaskName">TestJob</Data>  
  <Data Name="Path">C:\dump.exe</Data>  
  <Data Name="ProcessID">1204</Data>  
  <Data Name="Priority">5</Data>  
</EventData>
```

Stackable fields:  
TaskName,  
ActionName,  
UserContext

# Blind spot: Task paths

- TaskName defines the path to the job file
- By default, tasks are placed in  
`%systemroot%\system32\tasks\`
- Attacker with Administrator privileges can create tasks in  
`%systemroot%\system32\tasks\Microsoft\ [...]`
- If stacking on TaskName these may be harder to spot!

Attacker Shell

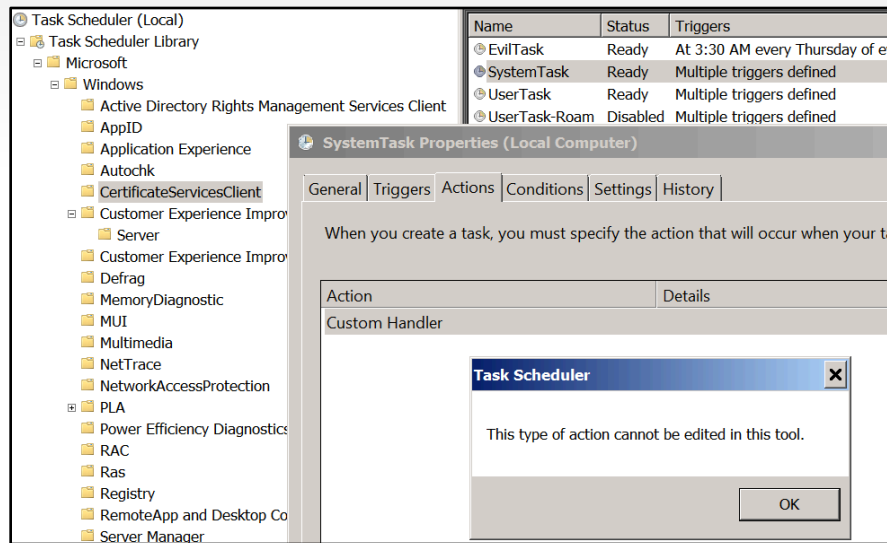
```
C:\> schtasks /Create /TN Microsoft\Windows\CertificateServicesClient\UserTaskRoam /TR C:\dump.exe /ST 18:15 /SC
```



`C:\WINDOWS\system32\tasks\Microsoft\Windows\CertificateServicesClient\UserTaskRoam.job`

# Blind spot: COM handler tasks

- Actions need not be an executable path!
- Note that `ActionName` is just a string for many OS-native Tasks
  - Cannot edit in Task Viewer UI
- These tasks invoke a COM object



EID 200

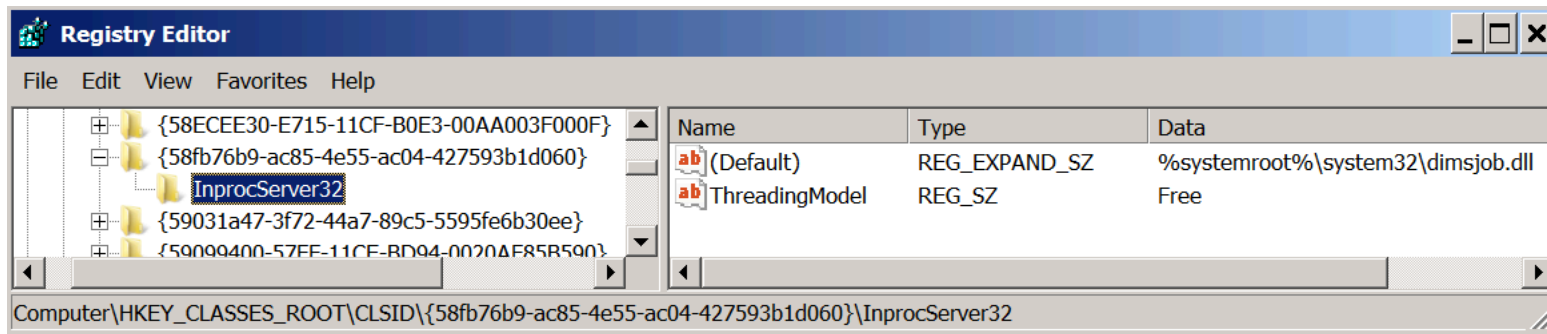
```
<EventData Name="ActionStart">
  <Data Name="TaskName">\Microsoft\Windows\CertificateServicesClient\SystemTask</Data>
  <Data Name="ActionName">Certificate Services Client Task Handler</Data>
  <Data Name="TaskInstanceId">{A9D9ED23-7D59-4D64-817F-206F18CF6A75}</Data>
</EventData>
```

# Mapping COM handler to associated DLL

SystemTask.job

```
<Task version="1.3"
xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
...
  <Actions Context="LocalSystem">
    <ComHandler>
      <ClassId>{58fb76b9-ac85-4e55-ac04-427593b1d060}</ClassId>
      <Data>SYSTEM</Data>
    </ComHandler>
  </Actions>
...
</Task>
```

CLSID →



# Attacker limitations

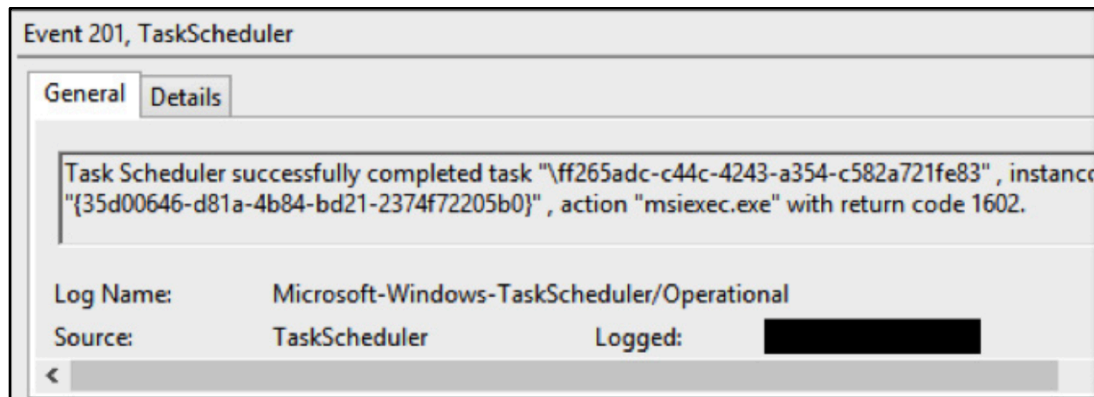
- Must import task configuration XML file if using COM

```
schtasks /Create /XML c:\EvilTask.xml  
/TN Microsoft\Windows\CertificateServicesClient\EvilTask
```

- Cannot modify existing tasks without breaking hash
  - Stored in the registry
  - Stuxnet exploited weak task hashing algorithm in older versions of Windows

# Revisiting our example: Duqu 2.0

- How common are tasks with `ActionName="msiexec.exe"`
- Could you have found this proactively, without any leads?



Source:

[https://securelist.com/files/2015/06/The\\_Mystery\\_of\\_Duqu\\_2\\_0\\_a\\_sophisticated\\_cyber\\_espionage\\_actor\\_returns.pdf](https://securelist.com/files/2015/06/The_Mystery_of_Duqu_2_0_a_sophisticated_cyber_espionage_actor_returns.pdf)

# Summary: Task hunting

- TaskName
  - Easy to filter for “\At” jobs, but not all will be malicious
  - Attacker can hide by choosing a known-good TaskName
- ActionName
  - Easy to stack on action paths and file names
  - Legitimate and evil tasks invoking “cmd.exe /c” will blend together
  - Rogue tasks that load COM objects may be hard to find at-scale without additional leads
- UserContext
  - Only paired with TaskName in event logs
  - Useful as an additional filtering criteria

# Other approaches

- EID 203 (Action Failed)
  - Attackers often screw up task syntax
- Harvesting JOB files
  - Small, easy to parse
  - One-time jobs may not persist indefinitely
- Searching process & command line history (if collected) for usage of `at.exe` and `schtasks.exe`

```
----- Sysmon Event Log -----
...
Source:           Microsoft-Windows-Sysmon
Date:             8/31/2015 3:51:35 PM
Event ID:         1
Task Category:    Process Create (rule: ProcessCreate)
Level:            Information
Keywords:
User:             SYSTEM
Computer:         AlicePC.acme.com
Description:
Process Create:
UtcTime: 2015-08-31 19:51:35.150
ProcessGuid: {33fd9a8b-b047-55e4-0000-0010c9074601}
ProcessId: 7032
Image: C:\Windows\System32\schtasks.exe
CommandLine: schtasks /Create /S AlphaWeb /XML
c:\users\administrator\desktop\TestTask.xml
/TN:Microsoft\Windows\TestTaskRemote2 /U:AlicePC\Administrator
CurrentDirectory: C:\Windows\system32\
User: AlicePC\Administrator
LogonGuid: {33fd9a8b-67a8-55e3-0000-00203c6d6800}
LogonId: 0x686d3c
...
```

# Working with ShimCache evidence at-scale

# ShimCache 101

- “Application compatibility cache” or “AppCompatCache”
- Tracks compatibility data for PE files, scripts
  - Created, modified, and / or executed within the scope of cache history
- Cache locations
  - `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatibility\AppCompatCache`
  - `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\AppCompatCache\AppCompatCache`
- Reference:  
[https://dl.mandiant.com/EE/library/Whitepaper\\_ShimCacheParser.pdf](https://dl.mandiant.com/EE/library/Whitepaper_ShimCacheParser.pdf)

# ShimCache 101

\$SI Last Modified,  
**not** created or  
executed

\*XP only

\*XP / 2k3

\* >= Vista, 2k8

| Last Modified     | Last Update* | Path                             | File Size* | Exec Flag* |
|-------------------|--------------|----------------------------------|------------|------------|
| 07/14/09 01:41:26 | N/A          | C:\Windows\System32\mmcshext.dll | N/A        | False      |
| 11/21/10 03:24:02 | N/A          | C:\Windows\system32\mstsc.exe    | N/A        | False      |
| 07/14/09 01:39:29 | N/A          | C:\WINDOWS\SYSTEM32\reg.exe      | N/A        | True       |
| 08/25/14 23:36:48 | N/A          | C:\Windows\pd.cmd                | N/A        | False      |
| 05/04/15 06:25:57 | N/A          | C:\Windows\PSEXESVC.exe          | N/A        | True       |
| 05/04/15 06:26:14 | N/A          | C:\Windows\PreDeploy.cmd         | N/A        | False      |

Most recent



Least recent

Adjacent  
entries often  
reflect files  
executed or  
created /  
updated in  
sequence

*Processed ShimCache excerpt from Server 2008 system*

# How much data?

- Up to 1024 entries in Vista, Server 2008 and later
- Averages lower: cleared by updates, patches
- Too noisy to stack by Full Path

```
04/19/11 11:09:16,N/A,c:\59c21276cd24c40bae86\install.exe,N/A,True
04/07/15 17:06:26,N/A,C:\Users\ryankaz\AppData\Local\Temp\{EC7A4BA8-3A3E-4BC2-910E-2B2B4CD6B4E7}~setup\vc_redist_x64.exe,N/A,True
07/14/09 01:39:02,N/A,C:\Windows\System32\cscript.exe,N/A,True
07/12/09 10:55:40,N/A,c:\9628c2ff8935c77372219bd5df26cd\install.exe,N/A,True
04/07/15 17:06:26,N/A,C:\Users\ryankaz\AppData\Local\Temp\{EC7A4BA8-3A3E-4BC2-910E-2B2B4CD6B4E7}~setup\vc_redist_x86.exe,N/A,True
04/07/15 17:06:24,N/A,C:\Users\ryankaz\AppData\Local\Temp\vmware-ryankaz\00004045\setup64.exe,N/A,True
04/07/15 17:06:24,N/A,C:\Users\ryankaz\AppData\Local\Temp\vmware-ryankaz\00004045\setup.exe,N/A,True
04/07/15 17:01:32,N/A,C:\Users\ryankaz\AppData\Local\Temp\upgrader.exe,N/A,True
04/07/15 17:01:32,N/A,C:\Users\ryankaz\AppData\Local\Temp\unattend.cmd,N/A,False
```

*Processed ShimCache excerpt from Server 2008 system*

# PsExec evidence in ShimCache

Attacker Shell

```
C:\temp>net use y: \\Victim\C$ /u:Administrator
```

```
C:\temp>copy dump.exe y:\dump.exe
```

```
C:\temp>psexec \\Victim c:\dump.exe
```

victim\_shimcache.csv

```
Last Modified, Last Update, Path, File Size, Exec Flag
```

```
...
```

```
Victim,08/24/15 14:40:13,N/A,c:\dump64.dll,N/A,False
```

```
Victim,11/21/10 03:24:02,N/A,c:\dump.exe,N/A,True
```

```
Victim,08/24/15 14:40:11,N/A,C:\Windows\PSEXESVC.exe,N/A,True
```

```
...
```

# Analysis workflow: Adjacent entries

## 1. Parse shimcache from each system's registry

—shimcache\_alicepc.csv—

```
Last Modified, Last Update, Path, File Size, Exec Flag
...
AlicePC,11/21/10 03:24:19 N/A C:\Windows\system32\net1.exe N/A True
AlicePC,07/14/09 01:14:27 N/A C:\Windows\system32\net.exe N/A True
AlicePC,07/14/09 01:14:21 N/A C:\Windows\system32\HOSTNAME.EXE N/A True
AlicePC,07/14/09 01:14:45 N/A C:\Windows\system32\whoami.exe N/A True
...
```

## 2. Aggregate and merge results, ensuring cache entry order is preserved

—shimcache\_merged.csv —

```
...
AlicePC,11/21/10 03:24:19 N/A C:\Wind|ws\system32\net1.exe N/A True
AlicePC,07/14/09 01:14:27 N/A C:\Wind|ws\system32\net.exe N/A True
AlicePC,07/14/09 01:14:21 N/A C:\Wind|ws\system32\HOSTNAME.EXE N/A True
AlicePC,07/14/09 01:14:45 N/A C:\Windows\system32\whoami.exe N/A True
...
BobPC,07/14/09 01:40:53 N/A C:\Windows\System32\filemgmt.dll N/A False
BobPC,07/14/09 01:39:19 N/A C:\Windows\system32\mmc.exe N/A True
BobPC,07/14/09 01:39:37 N/A C:\Windows\system32\services.exe N/A True
...
```

## 3. Search & return adjacent lines to matches

→ `grep -i -B 1 psaxesvc shimcache_merged.csv`

# Stacking paths adjacent to PsExec

Stack paths and jump to in-context locations for search hits

## Stack by Unique Path

```
Path,Count
C:\Windows\PSEXESVC.exe,18037
C:\temp\hostInv.bat,9144
C:\Windows\system32\scmcheck.cmd,1082
C:\temp\upgrader.exe,511
...
C:\test.bat,13
c:\dump2.bat,9
C:\dump.exe,7
C:\windows\temp\updater.exe,3
C:\psremote.cmd,4
C:\Users\Alice\AppData\Local\Temp\install.exe,1
...
```

—"dump.exe"—>

## AlicePC\_shimcache.csv

```
Last Modified, Last Update, Path, File Size, Exec Flag
...
Victim,11/21/10 03:24:02,N/A,c:\dump.exe,N/A,True
Victim,08/24/15 14:40:11,N/A,C:\Windows\PSEXESVC.exe,N/A,True
...
```

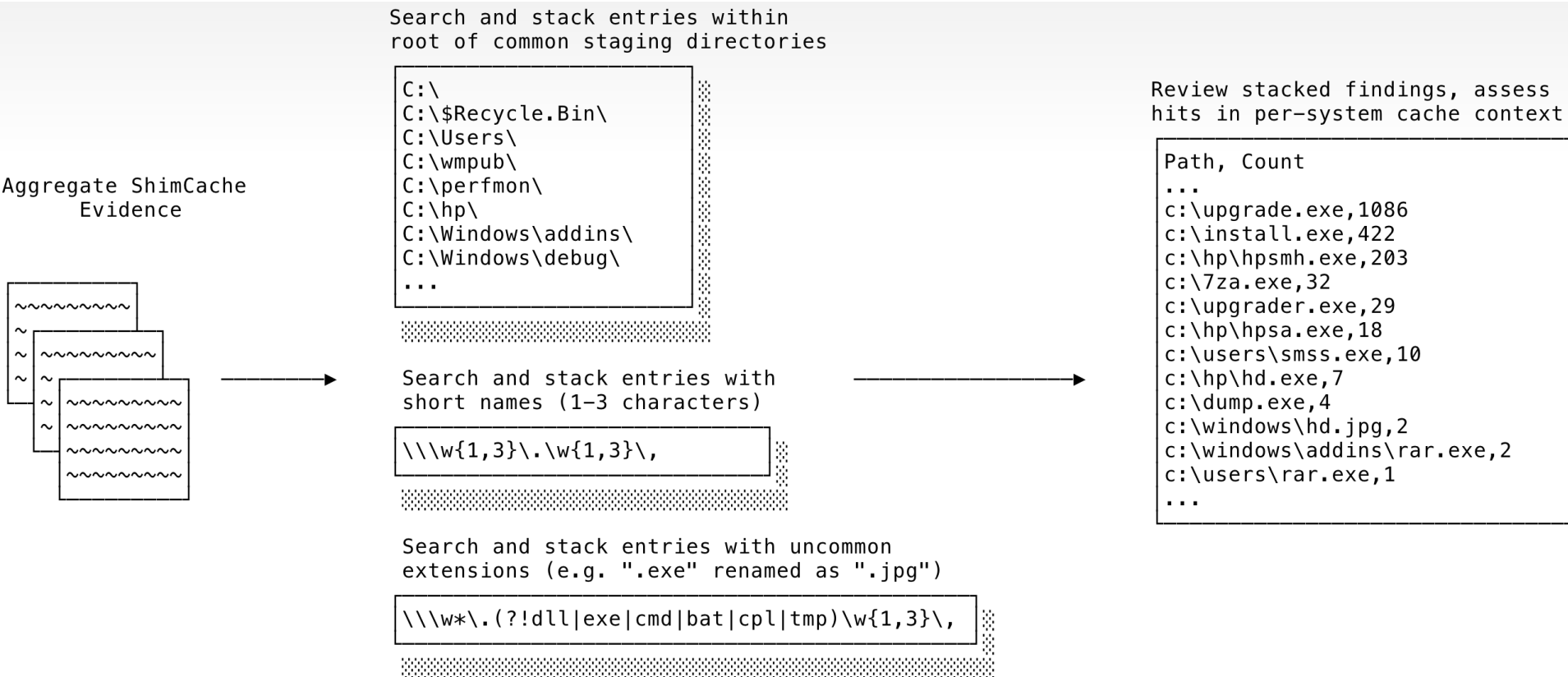
## BobPC\_shimcache.csv

```
Last Modified, Last Update, Path, File Size, Exec Flag
...
Victim,11/21/10 03:24:02,N/A,c:\dump.exe,N/A,True
Victim,08/24/15 14:43:52,N/A,C:\Windows\PSEXESVC.exe,N/A,True
...
```

## ChrisPC\_shimcache.csv

```
Last Modified, Last Update, Path, File Size, Exec Flag
...
Victim,11/21/10 03:24:02,N/A,c:\dump.exe,N/A,True
Victim,06/14/15 09:02:35,N/A,C:\Windows\PSEXESVC.exe,N/A,True
...
```

# Path searches and outlier analysis



# ShimCache analysis gotcha's

- Path stacking in `%systemroot%` and `%systemroot%\system32\` is noisy, difficult
- Incomplete data – outliers may not be true outliers
- Not all entries have executed
- Adjacent entries may be unrelated
- New entries only serialized upon reboot

# Other shim databases

- RecentFileCache.bcf
  - File paths only
  - Cleared by ProgramDataUpdater daily (or more often)
  - Replaced by AmCache in Windows 8 and later
- AmCache.hve
  - Windows 8 and later (limited footprint on Win 7)
  - Includes SHA-1 hashes, version metadata
  - More entries, slower to parse

## References:

- <http://binaryforay.blogspot.com/2015/07/amcacheparser-reducing-noise-finding.html>
- <http://www.swiftforensics.com/2013/12/amcache-hehve-in-windows-8-goldmine-for.html>
- <http://www.swiftforensics.com/2013/12/amcache-hehve-part-2.html>
- <https://github.com/williballenthin/python-registry/blob/master/samples/amcache.py>

# Other native evidence that can track execution

| Source                            | Full Path | Cmd-Line Args            | Parent Process | User | Timestamps                                | Other Evidence Captured                                                              | Availability & Scope                                  |
|-----------------------------------|-----------|--------------------------|----------------|------|-------------------------------------------|--------------------------------------------------------------------------------------|-------------------------------------------------------|
| Prefetch Files                    | Yes       | N/A                      | N/A            | N/A  | First & last run, add'l runtimes on Win 8 | Run count, list of files accessed w/in first 10 sec                                  | Workstations only; rolls at 128 entries               |
| Process Auditing (Security EVTX)  | Yes       | Optional, Win 7 / 2K8 R2 | PID only       | Yes  | Process start, process end                | Associated logon session GUID                                                        | Must be enabled by audit policy                       |
| AppLocker Events (AppLocker EVTX) | Yes       | N/A                      | N/A            | Yes  | Process start                             | Can track EXE, scripts, MSI, DLL loads                                               | Must be enabled by audit policy                       |
| Task Events (Task Scheduler EVTX) | Yes       | No                       | No             | Yes  | Task & process start & finish             | Task creation, task name, PID                                                        | Enabled by default; Vista & 2k8 onward                |
| ShimCache                         | Yes       | N/A                      | N/A            | N/A  | File last modified, cache last updated    | Tracks EXE, DLL, batch, VBS even for files that did not run but were present on disk | Default; history varies by OS, ~1,000 entries         |
| UserAssist (Per-user reg key)     | Yes       | No                       | No             | No   | No                                        | Application name and version data;                                                   | Default; only tracks EXEs ran in interactive sessions |
| MUICache (Per-user reg key)       | Yes       | N/A                      | N/A            |      | Last run time                             | Run count                                                                            |                                                       |

# Analyzing service events

# Service events

- Why services?
  - Remain a popular persistence mechanism for long-running malware
  - Can serve as a loader for short-lived tools
    - PsExec Service
    - Windows Credential Editor (WCE)
- What events?

A service was installed in the system.

Service Name: svcmsi\_32@  
Service File Name: msixec /i %systemroot%\temp\tmp129deb0.tmp /q PROP=fa5d78ba-f17  
Service Type: user mode service  
Service Start Type: demand start  
Service Account: LocalSystem

Duqu 2.0 installation as Windows service

Source:

[https://securelist.com/files/2015/06/The\\_Mystery\\_of\\_Duqu\\_2\\_0\\_a\\_sophisticated\\_cyber\\_espionage\\_actor\\_returns.pdf](https://securelist.com/files/2015/06/The_Mystery_of_Duqu_2_0_a_sophisticated_cyber_espionage_actor_returns.pdf)

# Service creation event (e.g. “sc create”)

User that created service is in event properties, not message

Service name and parameters in event message

EID 7045

Log Name: System  
Source: Service Control Manager  
Date: 8/26/2015 9:19:13 AM  
Event ID: 7045  
Task Category: None  
Level: Information  
Keywords: Classic  
User: AlicePC\Administrator  
Computer: AlicePC.acme.com  
Description: A service was installed in the system.

...

<EventData>  
    <Data Name="ServiceName">TestService</Data>  
    <Data Name="ImagePath">c:\windows\system32\evil.exe</Data>  
    <Data Name="ServiceType">user mode service</Data>  
    <Data Name="StartType">auto start</Data>  
    <Data Name="AccountName">LocalSystem</Data>  
  </EventData>  
  ...

Note: EID 7035 (service sent start control) not audited in Vista / 2k8 or later

50

Copyright 2015 Tanium Inc. All rights reserved.

 **TANIMUM™**

# Other examples

## EID 7045 - PsExec Service

```
...
<EventData>
  <Data Name="ServiceName">PSEXESVC</Data>
  <Data Name="ImagePath">%SystemRoot%\PSEXESVC.exe</Data>
  <Data Name="ServiceType">user mode service</Data>
  <Data Name="StartType">demand start</Data>
  <Data Name="AccountName">LocalSystem</Data>
</EventData>
...
```

## EID 7045 - Pmem Memory Acquisition

```
...
<EventData>
  <Data Name="ServiceName">pmem</Data>
  <Data Name="ImagePath">C:\Windows\TEMP\pmeA05.tmp</Data>
  <Data Name="ServiceType">kernel mode driver</Data>
  <Data Name="StartType">demand start</Data>
  <Data Name="AccountName"></Data>
</EventData>
...
```

## EID 7045 - Windows Credential Editor

```
...
<EventData>
  <Data Name="ServiceName">WCESERVICE</Data>
  <Data
Name="ImagePath">C:\Users\ADMINI~1\AppData\Local\Temp\5\8c8b
d470-4be0-4864-835e-53c85dea2a7b.exe -S</Data>
  <Data Name="ServiceType">user mode service</Data>
  <Data Name="StartType">demand start</Data>
  <Data Name="AccountName">LocalSystem</Data>
</EventData>
...
```

# Stacking service creation events

- “Who created which services?”
- “When and where?”
- `ServiceName + ImagePath + User` from EID 7045
  - Remember `AccountName` is the service context, not creator
  - `ImagePath` includes arguments
- Use time and hostname to further sub-filter

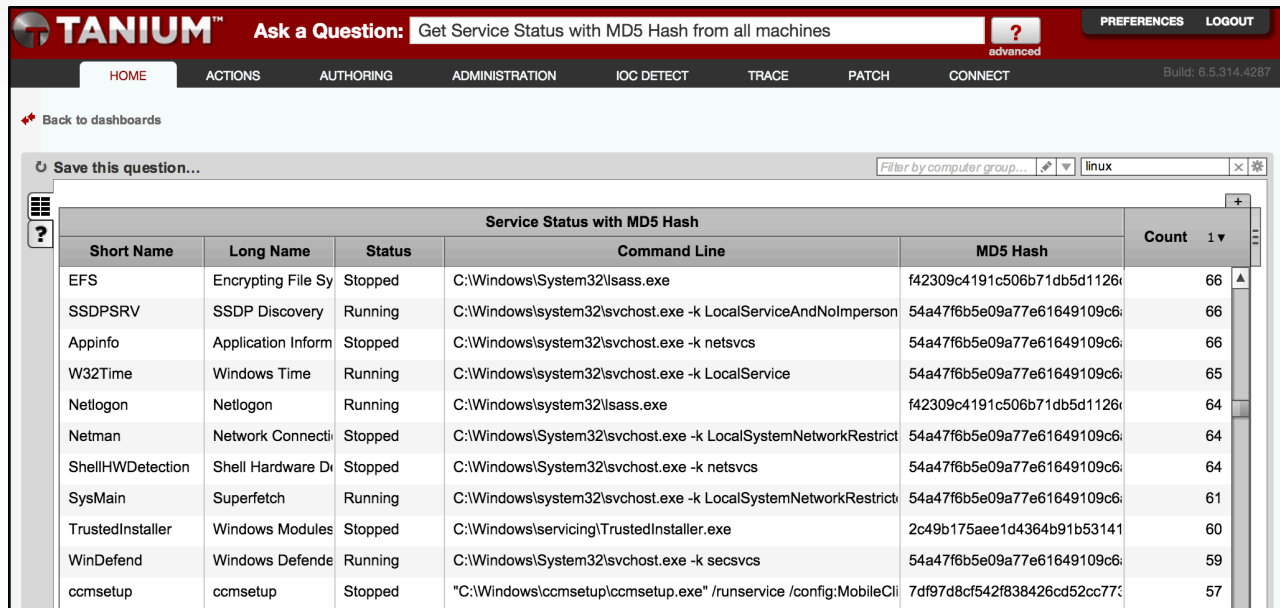
*Example / Case Study: Harvesting PsExec service events*

# Service event gotcha's

- Attackers can install services without calling `CreateService`
  - Avoids generating event log entry
  - Still may leave evidence in registry
- Many 3<sup>rd</sup> party applications install services
- Service start & stop events (7036) too frequent, noisy for outlier analysis

# Service configuration stacking

- Short name
- Long name
- ImagePath
- MD5 hash



The screenshot shows the Tanium console interface. At the top, there's a search bar with the text "Ask a Question: Get Service Status with MD5 Hash from all machines". Below the search bar, there's a navigation menu with tabs: HOME, ACTIONS, AUTHORIZING, ADMINISTRATION, IOC DETECT, TRACE, PATCH, and CONNECT. The main content area displays a table titled "Service Status with MD5 Hash". The table has columns for Short Name, Long Name, Status, Command Line, MD5 Hash, and Count. The table lists various services and their corresponding MD5 hashes and counts.

| Short Name       | Long Name          | Status  | Command Line                                                     | MD5 Hash                     | Count |
|------------------|--------------------|---------|------------------------------------------------------------------|------------------------------|-------|
| EFS              | Encrypting File Sy | Stopped | C:\Windows\System32\lsass.exe                                    | f42309c4191c506b71db5d1126c  | 66    |
| SSDPSRV          | SSDP Discovery     | Running | C:\Windows\system32\svchost.exe -k LocalServiceAndNoImperson     | 54a47f6b5e09a77e61649109c6   | 66    |
| Appinfo          | Application Inform | Stopped | C:\Windows\system32\svchost.exe -k netsvcs                       | 54a47f6b5e09a77e61649109c6   | 66    |
| W32Time          | Windows Time       | Running | C:\Windows\system32\svchost.exe -k LocalService                  | 54a47f6b5e09a77e61649109c6   | 65    |
| Netlogon         | Netlogon           | Running | C:\Windows\system32\lsass.exe                                    | f42309c4191c506b71db5d1126c  | 64    |
| Netman           | Network Connecti   | Stopped | C:\Windows\System32\svchost.exe -k LocalSystemNetworkRestrict    | 54a47f6b5e09a77e61649109c6   | 64    |
| ShellHWDetection | Shell Hardware D   | Stopped | C:\Windows\System32\svchost.exe -k netsvcs                       | 54a47f6b5e09a77e61649109c6   | 64    |
| SysMain          | Superfetch         | Running | C:\Windows\system32\svchost.exe -k LocalSystemNetworkRestrict    | 54a47f6b5e09a77e61649109c6   | 61    |
| TrustedInstaller | Windows Modules    | Stopped | C:\Windows\servicing\TrustedInstaller.exe                        | 2c49b175aee1d4364b91b53141   | 60    |
| WinDefend        | Windows Defende    | Running | C:\Windows\System32\svchost.exe -k secsvcs                       | 54a47f6b5e09a77e61649109c6   | 59    |
| ccmsetup         | ccmsetup           | Stopped | "C:\Windows\ccmsetup\ccmsetup.exe" /runservice /config:MobileCli | 7df97d8cf542f838426cd52cc77c | 57    |

# Service configuration stacking

- Add ServiceDLL path (registry) where present
- Add signature data and hash look-ups (e.g. VirusTotal)

| Count  | Service Name | Long Name                         | ImagePath                                                        | ImagePath MD5 | ImagePath Signed? | ServiceDLL                       | ServiceDLL MD5 | ServiceDLL Signed? |
|--------|--------------|-----------------------------------|------------------------------------------------------------------|---------------|-------------------|----------------------------------|----------------|--------------------|
| 20,344 | AeLookupSvc  | Application Experience            | C:\Windows\system32\svchost.exe -k netsvcs                       | 8f078ae4...   | Yes - Microsoft   | C:\Windows\System32\aelupsvc.dll | 4b78b431...    | Yes - Microsoft    |
| 21,196 | ALG          | Application Layer Gateway Service | C:\Windows\System32\alg.exe                                      | 3290d694...   | Yes - Microsoft   | N/A                              | N/A            | N/A                |
| 20,085 | AppMgmt      | Application Management            | C:\Windows\system32\svchost.exe -k netsvcs                       | 8f078ae4...   | Yes - Microsoft   | C:\Windows\System32\appmgmts.dll | 4aba3e75...    | Yes - Microsoft    |
| 8      | AppMgmt      | Application Management            | C:\Windows\system32\svchost.exe -k netsvcs                       | 8f078ae4...   | Yes - Microsoft   | C:\Windows\System32\appmgmt.dll  | c7f0a8be...    | No - unsigned      |
| 16,973 | AudioSrv     | Windows Audio                     | C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted | 8f078ae4...   | Yes - Microsoft   | C:\Windows\System32\Audiosrv.dll | f23fef6d...    | Yes - Microsoft    |
| 13     | AudioSrv     | Window Audio Service              | C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted | 8f078ae4...   | Yes - Microsoft   | C:\Windows\System32\Audiosrv.dll | af88c2eb...    | No - unsigned      |
| 9      | iSCSI        | iSCSI Devices Management          | C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted | 8f078ae4...   | No - unsigned     | C:\Windows\System32\iscsidsc.dll | bb5b4ba7...    | No - unsigned      |

# Bonus round (Other examples)

# WMI event consumers

- Covert, obscure persistence mechanism
- Used by SEADUKE / SEADADDY
  - <https://live.paloaltonetworks.com/t5/Articles/Unit-42-Technical-Analysis-Seaduke/ta-p/62743>
  - <https://github.com/pan-unit42/iocs/blob/master/seaduke/decompiled.py>
- Non-default WMI event filters and consumers are rare
  - Easy to enumerate with PowerShell
  - Data is perfect for stacking!

# Stacking WMI event consumers

Administrator: Windows PowerShell

```
PS C:\> Get-WMIObject -Namespace root\Subscription -Class __EventConsumer
```

Stack on Name, Class,  
CommandLineTemplate,  
ScriptFileName

| Count | __Class                  | Name                   | CommandLineTemplate          | ScriptFileName |
|-------|--------------------------|------------------------|------------------------------|----------------|
| 25191 | NTEventLogConsumer       | SCM Event Log Consumer | N/A                          | N/A            |
| 24852 | CommandLineEventConsumer | BVTConsumer            | cscript KernCap.vbs          | N/A            |
| 2     | CommandLineEventConsumer | EvilConsumer456        | C:\Windows\System32\evil.exe | N/A            |

# Alternative lateral movement

- PowerShell and Windows Remote Management (WinRM) increasingly popular
- Rely on Windows network authentication
  - NTLM
  - Kerberos
- Generate additional logon events during remote access
  - Low volume
  - Infrequently used by most users
  - Easy to harvest / search and spot anomalies
  - May persist beyond security event logs



# WinRM / PSRemoting logon events

Attacker Shell

```
C:\> winrs -r VictimPC.acme.com c:\windows\evil.exe
```

or

Attacker Shell

```
C:\> Invoke-Command -ComputerName AlphaDB {c:\windows\evil.exe}
```

EID 169

```
Log Name:      Microsoft-Windows-WinRM/Operational
Source:        Microsoft-Windows-WinRM
Date:          9/1/2015 3:19:28 AM
Event ID:      169
Task Category: User authentication
Level:         Information
Keywords:      Security, Server
User:          NETWORK SERVICE
Computer:      VictimPC.acme.com
Description:   User VictimPC\Administrator authenticated successfully using NTLM authentication
```

Event Xml:

```
...
<EventData>
  <Data Name="username">VictimPC\Administrator</Data>
  <Data Name="authenticationMechanism">NTLM</Data>
</EventData>
...
```

User and authentication protocol (NTLM, Kerberos) recorded in event data

# Conclusion

# Next steps

- Pick one of these techniques and practice!
- Learn the “noise” of your own environment
- Incorporate into red-vs-blue team exercises
- Ensure endpoint tools enable rapid search and harvesting
  - Current-state evidence (OS artifacts “at rest”, in memory)
  - Historical activity (logs, look-back databases)

# Further reading: Evolving attack techniques

- Modern Active Directory Attacks, Detection, & Prevention
  - <https://adsecurity.org/wp-content/uploads/2015/08/BlackHat-USA-2015-Metcalf-RedvsBlue-ModernActiveDirectoryAttacksDetectionandProtection-Final.pdf>
- Investigating PowerShell Attacks
  - <https://www.blackhat.com/docs/us-14/materials/us-14-Kazanciyan-Investigating-Powershell-Attacks-WP.pdf>
- Abusing WMI to Build a Persistent, Asynchronous, and File-less Backdoor
  - <https://www.blackhat.com/docs/us-15/materials/us-15-Graeber-Abusing-Windows-Management-Instrumentation-WMI-To-Build-A-Persistent%20Asynchronous-And-Fileless-Backdoor-wp.pdf>

## Further reading: Logging and monitoring

- Windows Logging Cheat Sheet
  - <http://sniperforensicstoolkit.squarespace.com/storage/logging/Windows%20Logging%20Cheat%20Sheet%20v1.1.pdf>
- Spotting the Adversary with Windows Event Log Monitoring
  - [https://www.nsa.gov/ia/files/app/spotting\\_the\\_adversary\\_with\\_windows\\_event\\_log\\_monitoring.pdf](https://www.nsa.gov/ia/files/app/spotting_the_adversary_with_windows_event_log_monitoring.pdf)

Thank you!

ryan.kazanciyan [at] tanium.com  
@ryankaz42

