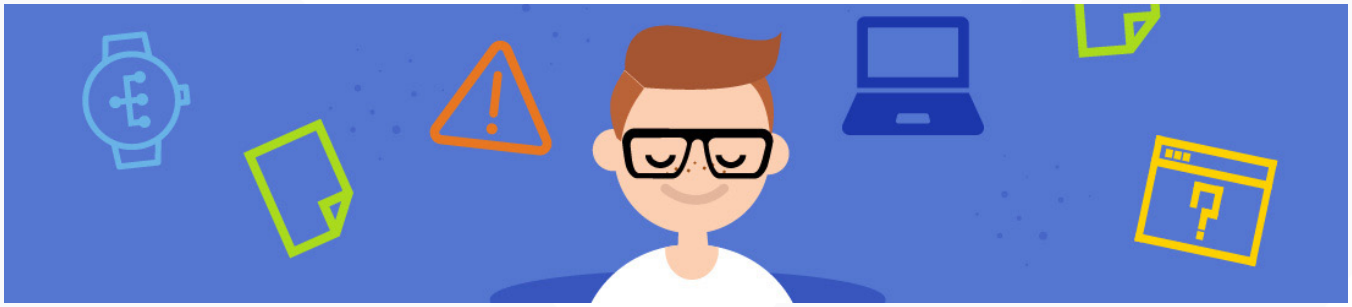

Aug 24, 2017 | Grant Oviatt

How to triage Windows endpoints by asking the right questions

Mindset over mater.

As security practitioners, it's important to remember that alerts are only the beginning, not the end, of finding evil. Alerts are simply investigative leads, not security answers. To determine if they have merit, you have to interrogate them. But without a strong process, the wealth of forensic information on the endpoint can easily overwhelm an investigator or lead them astray.

In this blog post, I'll explain the three parts of this investigative mindset and show you how to apply them when you triage endpoint alerts.

1. Know your indicators

The first question you should ask is a little obvious but it's often overlooked: what triggered the alert? As an investigator, if you can't answer that, you're going in with your eyes closed and you run the risk (or rather likelihood) that you're not going to select the right forensic artifacts. Even worse, there's a good chance you'll draw the wrong conclusions from the artifacts you do select.

One way to get the investigation off on the right foot is to create playbooks that recommend what the initial investigative step is to validate each type of detection.They should contain any IOCs (indicators of compromise) that are a part of the detection along with references to their source (whitepapers, tweets, etc.) or other related alerts. This gives analysts quick context to research the detections they encounter.

Don't have a threat intel team? Or maybe your security products don't give you any explanation when they throw an alert at you. That's OK. There's plenty of open source intelligence out there. Often a quick Google search or VirusTotal lookup for interesting file names, registry keys, or hashes can provide some context to guide your next steps.

But remember, context does not equal conclusions. Intelligence should only be used to guide your line of questioning. Be wary of making conclusions based solely on your search results.

2. Ask the right questions

I often find that inexperienced analysts pull back the same sources of evidence, regardless of their investigative lead. Usually, it's because there's no process to guide the way they triage an alert and ensure they get a complete picture.

Are you sans process? If so, I've summarized the process I use below by distilling it down into a few questions investigators can use to triage an alert, along with evidence sources that can help answer each question. Keep in mind, though, that the list of forensic evidence is in no way comprehensive.

Q: How did it get here?

Determine what took place that allowed initial access to the system (but note it's not always possible to answer this).

Evidence sources:

- Web browsing history/downloads
- IIS logs
- Service Control Manager Event Logs (EID 7045 - Service Install)
- Windows Security Logs (EID 4648 - Explicit logon attempt)
- USB Artifacts (USBSTOR, MountPoints2, MountedDevices registry keys or setupapi.log)
- Phishing Artifacts (RecentDocs and TrustedRecords registry keys and Jumplists)

Q: What does it do?

Figure out what the malware's host and network capabilities are including how it maintains persistence. This usually involves recovering a sample, performing some static/dynamic analysis, and/or uploading a sample to a sandbox.

Evidence sources:

- File acquisition
- Directory listings
- Acquire AV logs
- Acquire process memory
- Active network connections (netstat)
- Registry AutoRuns
- PowerShell Operational logs

Q: Did it execute?

If you've answered the previous question you know what the malware could do. The goal here is to see if the malware actually executed. This involves looking at execution artifacts for evidence that a particular binary launched, and searching for dropped files, registry keys created, and related evidence that would indicate the malware performed actions successfully on the system.

Evidence sources:

- Windows Prefetch
- Application Compatibility Cache (Shimcache)
- Amcache.hve
- RecentFileCache.bcf
- Registry AutoRuns
- WMI RecentlyUsedApps (RUA)

Q: Is it active?

If you learn that the malware executed, the next step is to determine if the threat is still present and running regularly on the system.

Evidence sources:

- Process listings
- Active network connections (netstat)
- DNS Cache
- Windows Prefetch

Answering these four investigative questions should be your objectives when you triage an alert. You may not be able to get good answers for all of them -- for example, determining how a piece of malware was created on a host may be impossible if it was created years ago and the forensic evidence is limited. However, you'll find that the answers you do get will quickly lead you to conclusions, and the story you can tell will be more comprehensive.

3. Understand your investigative footing

Not all alerts are equal. Depending on where the evidence came from -- a file, registry key, process event or log -- an analyst will have significantly different investigative perspectives. To illustrate, let's use the investigative questions I outlined above to show how your perspective changes with each evidence source, and how it should steer your forensic questioning.

Source #1: File

Surprisingly, the presence of a file doesn't necessarily provide a strong investigative footing. While a strange binary on a host should certainly raise suspicions, it may not have executed. Before you retrieve artifacts from the host, consult any playbooks or intelligence about the detection itself to answer the question "What does it do?". Knowing what files the malware drops and how it maintains persistence will guide the evidence an investigator seeks out to validate the threat. If available in the alert, use the NTFS timestamps for the file to establish a time window of potential activity. Below is a sample file detection using the approach I outlined above.

Example from intel sourced from: https://www.us-cert.gov/ncas/alerts/TA17-117A

Investigative Lead:

FILE DETECTION: REDLEAVES

File Name: VeetlePlayer.exe
File Path: C:\Program Files\Windows Media\VeetlePlayer.exe
Created: 1 day ago
MD5: 9d0da088d2bb135611b5450554c99672
File Size: 25704 bytes
File Description: Veetle TV Player
Signed: True

Off the bat we don't know much about this, based on the alert alone. An inexperienced analyst might be inclined to simply acquire the file. But in this case, that would lead to unfruitful results. By looking at the intel for the REDLEAVES RAT provided by the US Cert, we see that "VeetlePlayer.exe" is a legitimate binary that uses search-order hijacking to import a malicious loader DLL. This DLL then loads encoded shellcode contained in a file into memory. If an investigator simply acquired the legitimate executable, investigators would have come to the entirely wrong conclusion. Here is how I'd interrogate the evidence in this context:

Q: How did it get here?

Based on the recent creation time, we may be able to trace the initial compromise vector. Look for any suspicious logon activity prior to Windows service installation for evidence of lateral movement. Also check evidence relating to phishing documents, like RecentDocs registry keys or Jumplists.

Q: What does it do?

Using our understanding of how REDLEAVES behaves based on our existing intelligence, the next step is to validate that this is REDLEAVES. I'd validate the threat by performing a directory listing of "C:\Program Files\Windows Media\" and look for the presence of the malicious DLL (unsigned) and the shellcode file. Also, pull a Registry AutoRuns listing to see if the binary has been made persistent.

Q: Did it execute?

On a workstation, pull Windows Application Compatibility Cache and Prefetch. This will help identify if and when "VeetlePlayer.exe" has executed on the host.

Q: Is it active?

Based on the recent creation time, I'd expect this backdoor is likely active in memory. To check, I'd pull a process listing with network connection and the DNS cache on the host to validate.

Source #2: Registry

Registry alerts indicate that something has happened. Registry keys don't create themselves. So, if you're looking at an obscure key that a malware family uses, it's likely the host was infected at some point. While registry detections answer, "Did it execute?", they often leave you with less of a grasp on "What does it do?" because you're left without binary metadata. Below is an example of how I'd apply the investigative mindset to a registry detection.

Example from intel sourced from: https://www.us-cert.gov/ncas/alerts/TA14-212A

Investigative Lead:

REGISTRY DETECTION: BACKOFF

Registry Key: HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
Registry Value: Windows NT Service
Registry Data: %APPDATA%\AdobeFlashPlayer\mswinhost.exe
Last Modified: 30 days ago

From the registry detection we know that something executed to create this registry key. Now, we need to figure out if the referenced binary is evil and what it can do. By looking at the information provided by the US Cert, we can see that this detection is intended for PoS malware that creates two output files within the directory "%APPDATA%\AdobeFlashPlayer\". Given this information, here's how I would proceed with my questioning.

Q: How did it get here?

Evidence for this may be scarce based on the last modified time of the registry key. I'd look for Windows logon events around the last modified time of the registry key, specifically relating to Remote Desktop solutions based on the threat briefing.

Q: What does it do?

To figure this out, I'd acquire the referenced binary "%APPDATA%\AdobeFlashPlayer\mswinhost.exe" to see if it's still present on the host along with a directory listing of "%APPDATA%\AdobeFlashPlayer\" to validate whether output files have been created on the host.

Q: Did it execute?

A registry run key has been created, so we know something has executed on the host.

Q: Is it active?

I'd run a process listing with network connections and retrieve the host DNS cache to look for signs of current activity. Based on the last modification of the registry key, understand that this could be a long shot.

Source #3: Process

Process alerts put investigators on a relatively strong investigative footing. A process event tells you both that a binary has executed and that it's active. Plus, you get metadata about the binary. Again, before you retrieve artifacts from the host, make sure to consult any playbooks or intelligence containing information about the detection itself so you can better answer "What does it do?" and refine your line of forensic questioning. Below is an example of what that would look like.

Example can be found at: https://blog.malwarebytes.com/threat-analysis/2016/07/untangling-kovter/

Investigative Lead:

PROCESS DETECTION: KOVTER

Process Name: powershell.exe
Process Arguments: C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe iex $env:ksktr
Parent Process Name: mshta.exe
Process MD5: 097CE5761C89434367598B34FE32893B
User: CORP\Alice
Start Time: 30 minutes ago

Open source research tells us that KOVTER is "fileless" commodity malware that uses environment variables and registry data to store script interpreter commands that contain embedded shellcode. Given this information, here's how I would proceed with my questioning.

Q: How did it get here?

Pull web history for the user "Alice", along with artifacts related to any phishing attempts.

Q: What does it do?

We have a general understanding of what it might do from the open source intel. So let's pull the registry key, HKLM\System\CurrentControlSet\Control\Session Manager\Environment, containing system environment variables to see where variable "ksktr" could be storing powershell commands. Also I'd look at Registry AutoRuns to validate that persistence is still intact.

Q: Did it execute?

This one's easy. If it's a process event, that means the binary is running in memory-- so it must have executed.

Q: Is it active?

This one's also easy. If it's a process event, that means the binary is running in memory -- so it must be active. I'd check network connections for additional evidence.

Remember folks, detection is only half the battle. Good investigators are separated from great ones by the questions they ask. Hopefully this post has encouraged you to think about your own investigative mindset when you approach alerts. The key is to understand that all alerts are not made equal. Each provides unique investigative context. And by consulting intel resources before you extract forensic artifacts you'll develop a more efficient line of questioning.

--

Visit the EXE blog for more articles like this at https://expel.io/blog