How can I investigate the root cause of an alert?

- How can I investigate the root cause of an event?

How can I investigate the root cause of an event?

There are three building blocks of analysis available to you, while investigating an alert.

It is currently more likely that an attacker will conduct a variety of activities on a system, using a combination of in-memory techniques and process hijacking (injection or hollowing), than running an obvious “hacker.exe” program.

Memory Analytics

The first building block is found on the Response or Incident page itself. Memory analytics track the behaviors of the process in memory during continuous inspection of possible data extortion (theft or encryption) behaviors.

A memory state that indicates likely tampering is typically indicative of an issue.

It isn’t always malicious on itself, as poor programming practices or software bugs can also result in the memory state of a running program to be tampered with.

63504385.png?width=680
Here we see an instance where an Adobe Acrobat process being exhibit possible data extortion behaviors after opening a downloaded PDF, and we see the Acrobat process memory had been modified. The Acrobat.exe process on the disk has not been affected in any way.

Forensics analysis of the memory involved with the suspended Acrobat.exe will likely indicate an issue of concern - possibly an exploit in use, if not process injection or process hollowing.

For extortion responses with a Modified Memory value of True, there is a download icon in the Root Cause Analysis column to download the Memory Diff file for the response:

63602693.png?width=680
The Memory Diff File is the compiled source code of what was injected into the program. It likely contains malware and/or exploit code, for a malware or security analyst to examine. Note that this file does not contain the entire program, just the part that was altered through an exploit or attach technique like process injection. Learn more about this file here.

Process Injection Analytics

The first indicator that there may be a process injection event relevant to an automated response is the Untrusted Remote Threads column for an Incident/Response. This indicates that kernel-level behavioral analysis revealed an untrusted remote thread being observed with the process conducting data theft or ransomware-like behavior.

63471635.png?width=680
Further investigation may be performed on the Process Injection page. Untrusted Remote Threads reading True is a good indicator whether to go here.

The Root Cause Analysis column also has an icon to automatically redirect you to the Injections Page to only show the related process injections to the response

63602703.png?width=680
After clicking the injections icon, you will be taken to the Process Injections page where you will see only the related process injections to the response:

63569930.png?width=680
Here in the Process Injection Events page, we see the injector process path and arguments, and the injectee path and arguments. Usually, the arguments play a very important role in memory analytics, or indicate, for instance, what powershell commands to run. The process ID’s are present as well, to assist in tracing iterative injections done as an attacker hops from process to process, or for looking up root cause analysis in the Process Creation page.

63438879.png?width=680
How did we know rundll32.exe was really an antivirus vendor? Read on!

Process Creation Analytics

It is very rare that an attacker only uses one process during their attack. They may not even know they tools they are using are spawning new processes, and using various administrative tools on the infected system.

The Process Creation page tracks all processes that open other processes (or the same), the arguments used, and the process ID’s, or Pids, for each source and destination.

This is performed at the kernel level, meaning nothing is missed.

Note: Microsoft telemetry has some of this data, but not all. We elect to not use that data source, because we've observed that telemetry source has been seen to miss important events, or be switched off by attackers during an attack.

How good is our visibility?

Putting it altogether - an end to end investigation

How good is our visibility, and how fast can you perform Root Cause Analysis? Check this example out - an antivirus with arguably some of the highest permissions on a system, was observed trying to inject code into our Cyber Crucible software. We hope it was for investigating our software, but we still go into self-protection mode (so, this caused our software to reject the antivirus' tampering…there are no back doors allowing access to edit or use our software by external parties).

So, let's start here, with an automated response:

63569937.png?width=680
Obviously Cyber Crucible is not ransomware. We see here that a process injection attempt involving suspicious remote thread behavior, in conjunction with data extortion-like behaviors.

OK - let's look at what's going on in Process Injection Events by clicking the injections icon . 5 seconds later - we have our answer!

63471644.png?width=56463438879.png?width=680

But wait - rundll? That's not a hacker, is it? Surely, that's not a hacker.exe program. Let's do some filtering, to get our answer on the Process Creation page. We don't have a screenshot here, but process ID's are available for all responses and process injections. Let's go back to the responses page, and click the process creations icon to take us to the Process Creations page and automatically filter the grid to show the related process creations of this response.

63537184.png63438886.png?width=680

Quick and easy, we now see that the "root" (no pun intended) was WebRoot, an antivirus vendor, spawned a rundll32.exe process to load one of WebRoot's DLL's, which then tried to do *something* with Cyber Crucible's software, to force it to iterate through files like a data theft tool.

Why did we choose to show this example?

We have copious examples of attacker tradecraft, with varying levels of competence on the attackers' part. Antiviruses achieve a level of permissions on a system that require a very high level of skill, technology, and preparation by attackers. Cyber Crucbile detecting “hacker like” behaviors from a mainstream security tool, then automatically stopping it, demonstrates some of the highest levels of engineering and security excellence on our part, without getting lost in individual exploits or attacker tradecraft.