# Threat Hunting 101

## 8 Threat Hunts You Can Do with Available Resources

LogRhythm®
The Security Intelligence Company

# TABLE OF CONTENTS

# Threat Hunting 101

## 8 Threat Hunts You Can Do with Available Resources

A hunter wishing to bring food home for his or her family first needs to decide what type of animal he or she is going to target. Every quarry requires its own unique methods that dictate when the hunter goes out, where he or she goes, what kind of weapon to carry, and a host of other considerations.

In the world of cybersecurity, it's no different. You don't just "go threat hunting." You need to have a target in mind, you need to look in the right places, and you need the right weapons.

In this white paper, we will discuss the minimum toolset and data requirements you need for successful threat hunting. We will take into account that, while some readers can devote most of their time to threat hunting, like most, you have limited time and resources for this activity. The good news is that threat hunting is flexible, and anyone can do it, regardless if you are spending just a few hours a week to full time.

Threat hunting is the process of proactively searching for malware or attackers that reside on your network. The generally accepted method is to leverage a security information and event management (SIEM) solution that centrally collects log data from disparate sources—endpoints, servers, firewalls, security solutions, antivirus (AV), and more—providing visibility into network, endpoint, and application activity that might indicate an attack.

The challenge with threat hunting is knowing what to look for. So, this white paper explores eight types of threat hunts that you can use to spot suspicious abnormalities that might be a leading or active indicator of threat activity.

First, make sure you know the kinds of log data that are necessary to threat hunts.

## Leveraging the Right Log Data for Threat Hunting

A SIEM is only as good as the data it uses, and proper threat hunting requires contextual data from a wide range of log sources. It's important to collect log data from every security-related aspect of the environment: your network (including network devices and externally facing systems), endpoints, servers (both Windows and Linux), internal applications and services, and security and authentication solutions. The following list provides an example of the specific log data sources you should consider.

**Network Devices**

- Firewalls
- Routers/Switches
- Load Balancers
- Proxies/Reverse Proxies
- VPN Systems

**External Facing Systems**

- Web Servers
- DNS Servers
- Email Proxy Systems
- Application Services
- VPN Systems
- Reverse Proxies

**Security Parameter**

- Intrusion Detection/ Prevention System
- Endpoint Security Suite
- Antivirus Management
- Email Management
- Vulnerability Scanners

**Linux Systems**

- /var/log/messages
- Audit Logs
- Host Logs
- Keylogging Logs
- Security Agent Logs
- Application Logs

**Internal Systems**

- File Servers
- Print Servers
- Email Servers
- Database Appliances
- Production Applications
- File Integrity Monitoring
- Registry Integrity Monitoring

**Windows Systems**

- System Logs
- Application Logs
- Security Logs
- PowerShell Logs
- Sysmon Logs
- Security Agent Logs
- File Integrity Monitoring
- Registry Integrity Monitoring

**Authentication Systems**

- Identity and Access Management (IAM)
- Privileged Access Management
- Policy Brokerage
- Active Directory Logs
- Kerberos Logs
- Signal Sign-on Logs (SSO)
- Multi-Factor Authentification (MFA)

Once you are centrally collecting the proper log data in your SIEM, you can begin the process of threat hunting. Start with one of the easiest and more telling indicators of threat activity: suspicious software.

# Threat Hunt No. 1

## Recognizing Suspicious Software

Attackers use locally installed malware for a number of reasons: control, persistence, automation, and data exfiltration. But for an attacker to leverage malware, it must be running as a process on the endpoint. This means that you can hunt for unusual software running on endpoints as a means to identify potential attacks.

As shown in Figure 1, there are two basic ways to identify suspicious software: by process name or by process hash. If you have an endpoint detection and response (EDR) solution in place on your endpoints, it might be able to port its log data to your SIEM solution, providing additional ways to spot suspicious software.

| Basis | Source | Subsource | Pros and Cons |
|---|---|---|---|
| Process name | Security Log | Audit process tracking: 4688 | Far easier. Nothing to install. Can be spoofed. |
| Hash | | | • Higher integrity than process names<br>• Also provides information on digital signatures<br>• Must install, maintain Sysmon<br>• Hashes far more complex to monitor<br>• New hash every time file patched<br>• .NET compiles hundreds of DLLs optimized for local system |
| | EDR | ??? | ??? |

*Figure 1. You can identify suspicious software by using either the process name or hash.*

Hunting by process name is a much easier task; all that's needed is to match the name in a log to the name of a malicious process you're looking for. But many attacks involve a spoofed process name, simply renaming the malicious executable to something known to the operating system (e.g., NOTEPAD.EXE).

Therefore, hunting based on a process hash provides a means to quickly determine whether a process is "known good." Even when a malicious executable is renamed to something known, it still produces a unique hash. The challenge with using hashes is twofold. First, you need to install and maintain the Windows Systernals tool, Sysmon, on every Windows system you want to monitor. Second, every time you patch an application or OS, you need to update the list of known-good hashes.

## Using the Process Name

Use the following steps to identify suspicious software:

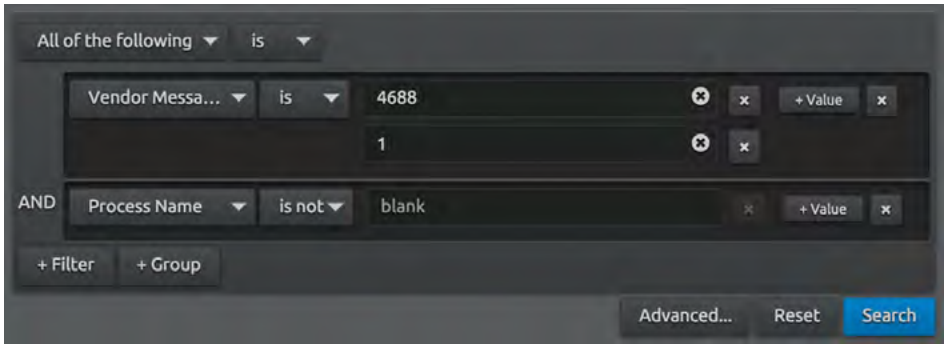
1. **Enable auditing of process tracking.** Use event ID 4688 (which includes process name, ID, command line used, and so on) from the Windows security log, or event ID 1 from the Microsoft Sysmon event log.

2. **Create an initial baseline of applications.** This step is time dependent. For example, the longer the duration selected, the more accurate the baseline.

    a. If this data is incorporated into your LogRhythm SIEM, you can use LogRhythm's WebUI Lucene query to list unique processes running across a single, or multiple systems:

```
vendorMessageId:("4688" OR "1") AND process:*
```

    b. You can also perform the same query within the LogRhythm WebUI search window:




    c. If you only have access to the windows hosts themselves, you can use a SQL statement like the following to extract a deduplicated list of process names for your baseline:

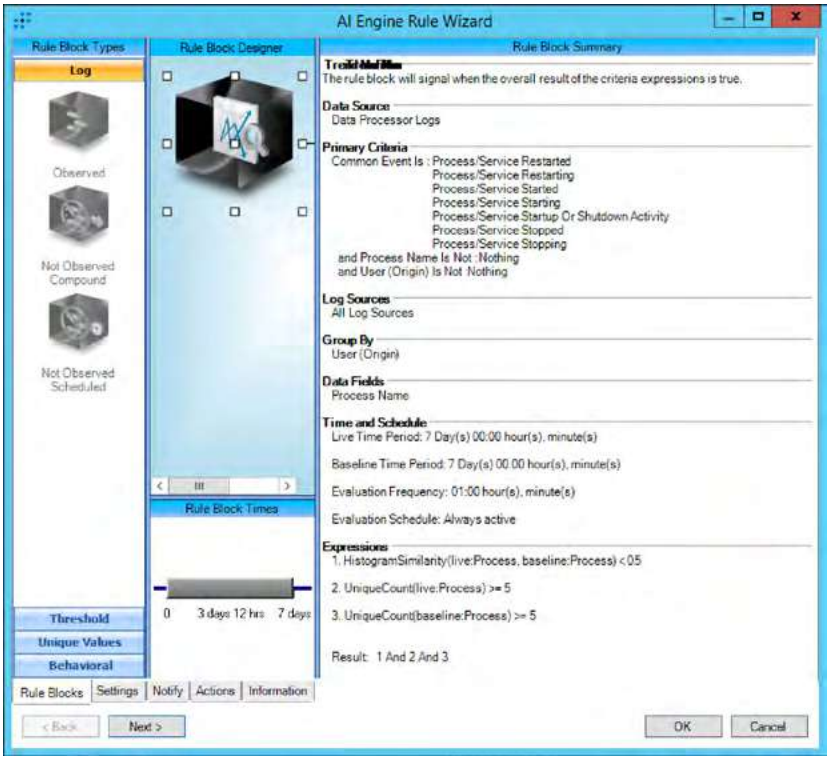
```
Select distinct ProcessName from Events where EventId=4688 OR EventId=1
```

3. Compare new processes against the baseline. Once you have a sufficiently accurate baseline, compare incoming 4688 or 1 events against that baseline. You can use these values to create a list of process names which can then be used to notify SIEM operators in the event a new process is identified. The comparison SQL statement could look something like the following:

```
If (select count(*) from Events where
ThisEventProcessName=ProcessName
and EventId=4688 OR EventId=1) = 0
```

If the process is already on the baseline, ignore the event. But if the process is new to the baseline, add it and have a notification sent to someone to investigate.

Additionally, LogRhythm currently maintains a set of helpful AI Engine rules within its out-of-the-box content. As an example, one of these rules, C2: Abnormal Process Activity, maintains a trending list of witnessed processes within a configured environment. This type of rule can greatly assist threat hunters when they witness new processes.



4. **Investigate**. Follow this simple process.

    a. The investigator needs to receive an alert, be presented a dashboard, or receive a daily report—anything that tells the investigator to focus on these processes.

    b. Next, the investigator should review each process and determine whether it appears to be a program trying to look like a common program. For example, the filenames C:\Windows\System32\d11host.exe and C:\Windows\System32\srvchost.exe look very close to the real thing, but they definitely are not part of the OS.

    c. If the filename looks suspicious, Google the process name, looking for details.

    d. Check the full filename and path on the VirusTotal website, looking for how long the file has been on the site and whether it's been reported as malicious.

    e. Potentially, sandbox the executable and see if it does anything malicious to a virtual machine.

It's important that you think about this process beyond just one global baseline. What runs on computers in the Sales department is very different from in Finance. Consider grouping computers based on departmental use within the organization to derive use-case-based baselines that accurately depict normal processes for that group.

### LogRhythm Insights: Automating Rogue Process Hunting

To ensure the accuracy of the list, the efficiency of threat-hunting suspicious processes, and the speed of notification should a rogue process be spotted, LogRhythm uses an AI Engine (AIE) rule called a Whitelist Rule Block, whereby the ProcessName value from each 4688 event is automatically added. In addition, a list of processes can be manually added to the rule.

Once a baseline is established, you can use the same process list across multiple endpoints, with the rule modified to alert the appropriate staff when a new process is spotted.

## Using a Hash

This method uses an investigation and procedure similar to those for process names. To gather process hashes, you need to install Sysmon on each system that will be baselined or continually monitored.

Be aware of a few distinct differences from process monitoring:

- **There are more hashes to investigate than program names.** Each executable has a unique hash of the binary code that makes up the file. So, when a new version of that same executable (think patches and updates) is created, so is a new hash. If you support two version of Microsoft Word, for example, you'll have two hashes of winword.exe.
- **Maintaining a whitelist of known-good hashes is more work**. Simply scanning a golden image is insufficient. You need to update the whitelist before patches hit production, likely accomplished by scanning your patched test system for new hashes. There are commercially available whitelists, but in general, they aren't updated quickly enough. You should look up hashes against VirusTotal, but you should also consider ignoring or deprioritizing a hash if it received a neutral rating and was first scanned a long time ago.

If you tackle these challenges and build a hash whitelist, you will know within minutes whenever a new binary file executes in your environment. Keep in mind, there are two scenarios when using hashes won't work:

- **Buffer overflows and related non-EXE binary code**. Remember, attackers have developed multiple ways to get binary code to run without loading an EXE, DLL, and so on. When these methods are used, hashes aren't available for comparison.
- **Scripting abuse**. Attackers that are "living off the land" might use PowerShell, WSH, or JavaScript to act. While the script executables will be evident, their intent and actions won't be.

## LogRhythm Insights: Why Hunt by Process Name?

If attackers can simply rename an executable before running it, and if hash monitoring produces far more accurate results with a low risk of alert fatigue, why would you ever do process monitoring at all? In practical application, hash monitoring is far more difficult, requiring constant updating of known-good hashes. And because most attackers simply try to name their applications to something that looks legitimate rather than spoofing an OS-specific executable, monitoring process names remains an effective way to spot a potential threat.

Maintaining an active view of hash values in your environment can be helpful in a number of instances. Simply making a WebUI widget to display hash values can save an analyst from having to perform a search during an investigation.

# Threat Hunt No. 2

## Behavior Changes

The idea of monitoring processes or hashes gives IT a one-dimensional view into what's running on a given endpoint. But when you add in other factors, such as whether a process is normal for a given user or which parent process spawned a potentially suspicious process, the monitoring becomes more about behavior of the endpoint or user. As shown in Figure 2, the same sources (i.e., Security log, Sysmon, and your EDR solution) can be used to provide detail on which user or parent process is responsible for launching a new process.

| Basis | Source | Subsource | Pros and Cons |
|---|---|---|---|
| Process name + Parent Process Name | Security Log | Audit process tracking: 4688 | Nothing to install |
| | Sysmon | 1: Process Creation | Must install Sysmon |
| | EDR | ??? | ??? |
| Username + Process Name | Security Log | Audit process tracking: 4688 | Nothing to install |
| | Sysmon | 1: Process Creation | Must install Sysmon |
| | EDR | ??? | ??? |

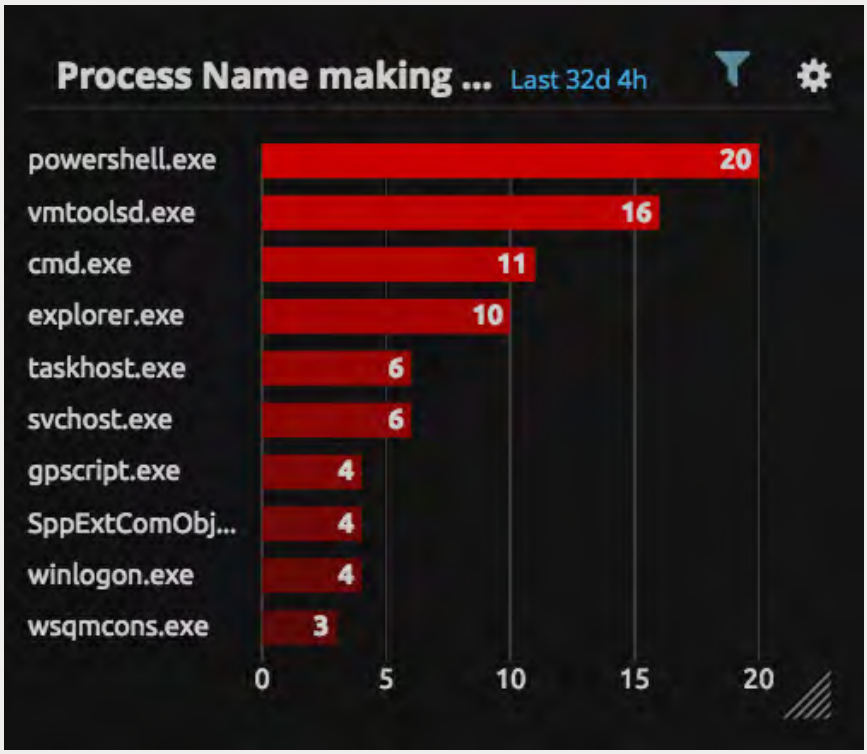
*Figure 2. By adding either the parent process or the username, processes begin to take on context useful for hunting.*

These combinations provide the necessary context to determine whether an investigation is warranted. Take the following example: In and of themselves, RDP.EXE and Microsoft Word aren't malicious at all. But as Microsoft Word launching an RDP session is abnormal, it is certainly cause for a closer look.

## LogRhythm Insights: Processes and Network Traffic

Another sign of a potentially suspicious process is one that generates network traffic. For example, you wouldn't typically expect NOTEPAD.EXE to begin communicating across the network. With attackers using filenames that mimic legitimate applications by using nearly identical naming, monitoring for the establishing of external network connections can help to spot malware droppers attempting to communicate with a command-and-control (C&C) server, or an application exfiltrating data from your network.

As shown in this figure, LogRhythm analyzes outbound connections with the process name to help spot potentially dangerous rogue applications. Notice in the example that powershell.exe is making an outbound connection, raising suspicion of its intent.



# Threat Hunt No. 3

## Scripting Abuse

Attackers trying to evade detection might avoid introducing new processes that will alert IT to their presence. Instead, they resort to scripting languages that are already available on the endpoint—in particular, PowerShell and Windows Scripting Host.

As shown in Figure 3, the simplest threat hunt is to monitor for execution of a scripting engine. The processes *cscript*, *wscript*, and *powershell* indicate the launching of a script.

Because IT is known to use scripting, you should avoid creating alert fatigue with too many false positives. As with processes in Hunt No. 1, monitoring the use of encoded scripts (a common tactic of attackers), specific script filenames, which parent process spawned the scripting engine—even adding in the dimension of the involved endpoint name or username involved—can all help to home in only on instances of scripting that indicate a potential threat.

| Basis | Source | Subsource | Pros and Cons |
|---|---|---|---|
| Execution of scripting engine | Security Log<br>Sysmon | Event ID 4688<br>Sysmon 1<br>Process name = cscript<br>wscript or powershell | • On endpoint where usually not executed<br>• For user account usually not running scripts<br>Filter out known script name |
| Encoded scripts | | | 4688 or 1 with powershell.exe and "-EncodedCommand" |
| Parent process | | | Baseline parent process names that usually kickoff scripts; look for new parents |
| Script file names | | | Baseline known script file names or implement a naming convention<br>Look for new or uncompliant script names |

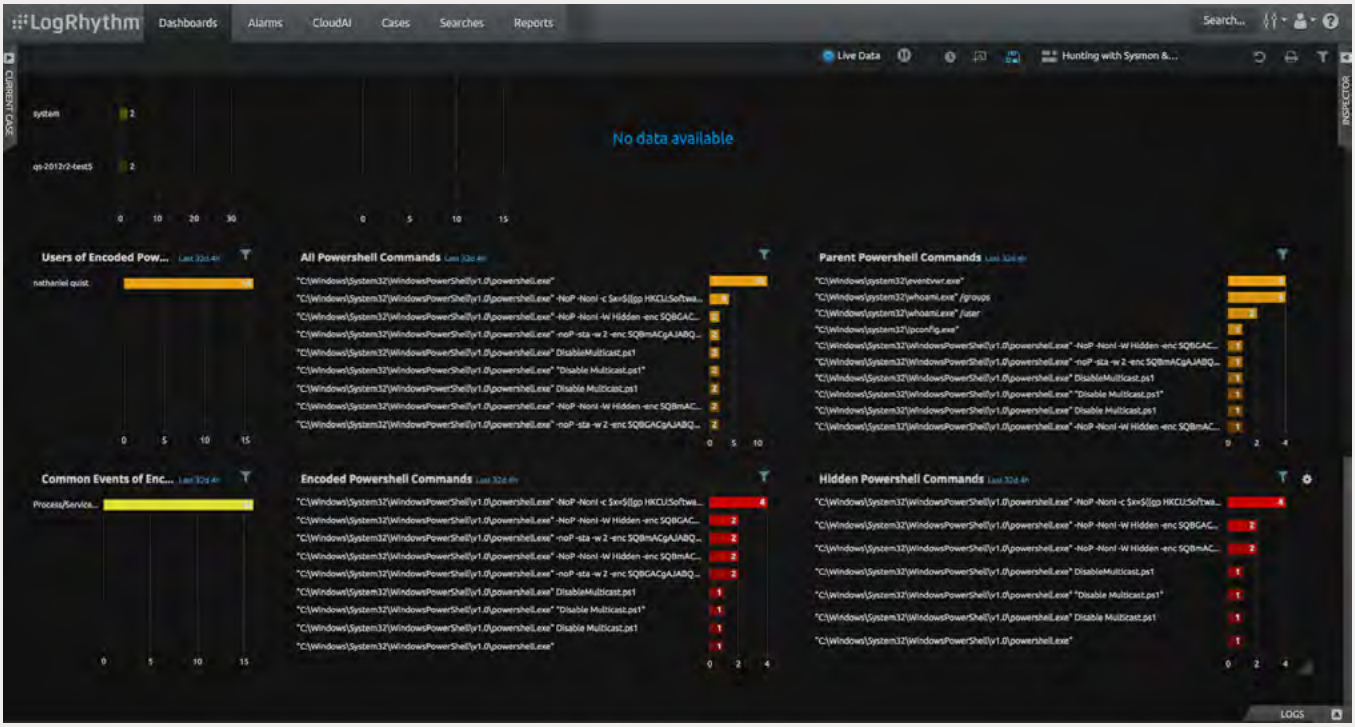
*Figure 3. Monitoring scripting engines, script filenames, and parent processes helps to spot malicious scripting.*

## LogRhythm Insights: Monitoring PowerShell

Auditing the usage of Windows Scripting Host is nearly impossible, as no logs capture what the script is doing, other than at a process level. However, PowerShell has audit logs that enable the monitoring of every command run, code block detail, and command output. LogRhythm can leverage this detail to create custom rules, actions, and views.

As shown in this view, LogRhythm can easily monitor the use of encoded PowerShell scripts (which obfuscate the actions that the script will perform), showing the users utilizing encoded scripts, the command lines used, and how often the script has been used.

These details can help provide context to determine whether the running of a script is suspect.



# Threat Hunt No. 4

## Antivirus Follow-Up

When you think about antivirus, you're likely concerned only about the number of files scanned and cleaned, or the current "safe" status of the managed endpoints. But antivirus applications can provide a lot more data that can assist with threat hunting.

Take the simple question: From where was the malware cleaned? In an expected folder like C:\Users\<User>\Downloads, it's a simple scenario of a user downloading a malicious file from the internet. But if the malware is cleaned from a folder like C:\Windows\System32, you have a potential elevated privilege issue, as administrative rights are needed to write to that folder.

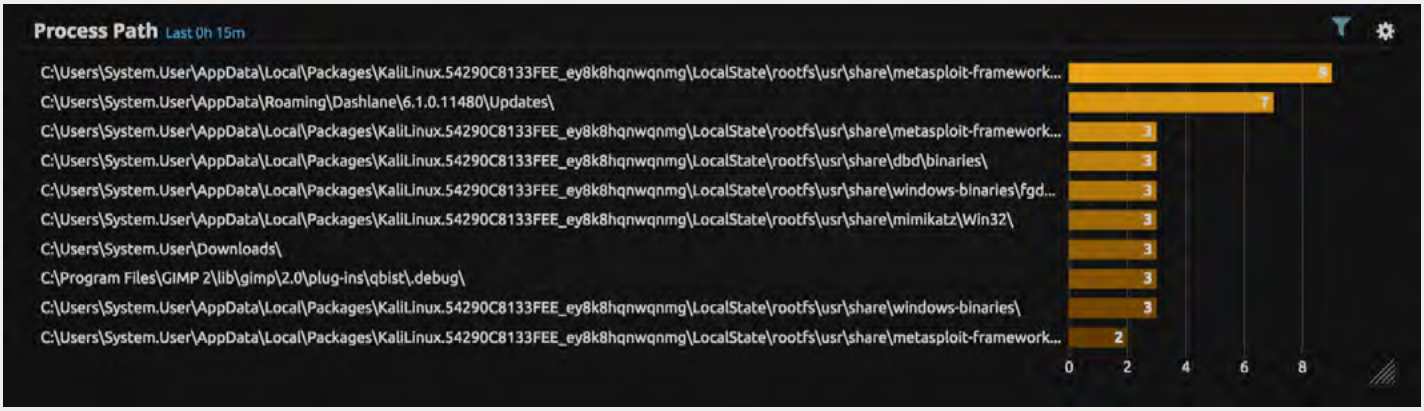
In addition, antivirus data can also be used collectively across your enterprise to better understand if and where malware is moving across your environment. So, consider antivirus log data as a viable source of post-threat intel that can help point out where network segmentation or elevated privilege issues might exist within your environment.

## LogRhythm Insights: Spotting Threat Activity from Antivirus Logs

Investigations often use forensic tools to dig into the current state of an endpoint and look for digital artifacts that indicate specific activity. But if your antivirus solution spots malware and cleans it from the endpoint, no markers are left behind.

LogRhythm takes log data from the industry's leading antivirus and EDR solutions, empowering the customization of monitoring, alerting, displaying, and reviewing of solution activity as part of your threat hunting. As shown in this view, data as simple as the file path where malware once existed (and is now cleaned or quarantined) can provide insight into specific threat activity.

As seen here, paths that include such known hacking terms as metasploit and mimikatz likely indicate that malicious tools were installed on the now-clean endpoint—and possibly used by a malicious threat actor.



# Threat Hunt No. 5

## Persistence

Once attackers have achieved some degree of control over an endpoint, they desire to retain that control, even after a reboot, logoff, or termination of a malicious process. Attackers use known methods of launching applications—Run, RunOnce, Shell, RunServices, and other keys—to make certain the malicious code that establishes their control runs each and every time the system boots up, logs on, and so on. As shown in Figure 4, both Sysmon and the Security Log can be used to determine when registry keys related to persistence are modified.

| Basis | Source | Subsource | Guidance | |
|---|---|---|---|---|
| Registry Key | Security Log | 4663 | Enable registry auditing on specific keys using group policy | Registry key on the autoruns list<br>https://www.ultimatewindowssecurity.com/webinars/register.aspx?id=1514 |
| | Sysmon | 12 - 14 | Must install Sysmon and configure which keys to monitor | Filter out known actors and keys; maintain baseline |
| Scheduled Tasks | Security | 4698 - 4702 | Enable "Other Object Access Events" Auditing<br>Watch for new scheduled task names and actors | |
| WMI Eventing | Sysmon | 19 - 21 | Watch for any instances at all or for new actors, new computer names | |
| Services | Security Log | 4697 | Enable "System Security Extension" auditing | |

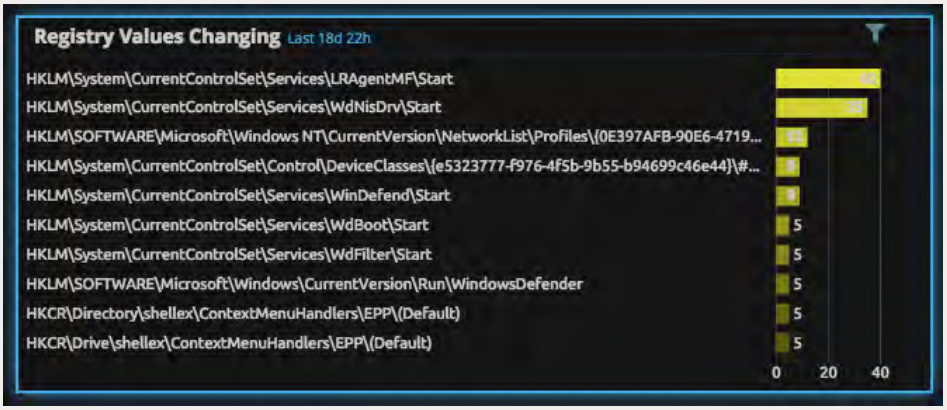
*Figure 4. Specific changes made to the OS help to indicate threat actors establishing persistence on an endpoint.*

Monitoring can be based on a baseline of users, processes, and registry keys that are normally modified. But your monitoring strategy can also simply be to watch the pertinent keys, providing as much detail as possible about who made the change and via which process.

## LogRhythm Insight: Do Users Have Admin Authority to Workstations?

One of the easiest ways to thwart attackers establishing persistence is to limit administrative access to endpoints. Whether a formal implementation of Least Privilege or simply an organizational policy that users have low-level access, this first step limits the persistence attack surface to only those users with elevated credentials.

Should an attacker attempt to establish persistence, as shown in this figure, LogRhythm can visualize the changes, providing details about the keys, user, and processes involved.




And because attackers can leverage scheduled tasks, Windows Management Instrumentation, and Windows Services, changes made to each of these parts of the Windows OS can also be equally visualized to provide a complete view of admin-level changes that denote persistence.

# Threat Hunt No. 6

## Lateral Movement

Once an attacker has established persistence on an endpoint, offering a foothold into your organization, the next step is to move from endpoint to endpoint throughout the network, until the target system containing valuable data is found. As shown in Figure 5, monitoring for unusual user/endpoint logon combinations, as well as abnormal network connections made between systems, provides an early indicator that a threat actor is attempting to move laterally within the network.

| Basis | | Source | Subsource | Guidance |
|---|---|---|---|---|
| Logon attempts: new user/ endpoint combo | | Security Log | 4624, 4625 | Baseline tuples of ComputerName and New Logon Account Name and Domain<br>Filter out new computers and users |
| Network Connections: | New endpoint/ Endpoint combo | Sysmon | 3 | New combinations of Source and Destination<br>Filter out external IPs<br>Filter out new computers<br>If DHCP address - use host name; No DNS names available on Security Log events |
| | | Security Log | 5156 | |
| | Unlikely connection combo | Sysmon | 3 | Direction outbound - Destination address should never be DHCP unless this is a systems mgt server or vulnerability scanner<br>Direction Inbound - If local computer is a workstation or source address is DHC, this is suspicious unless source is a systems mgt server of vulnerability scanner |
| | | Security Log | 5156 | |

*Figure 5. New combinations of users and endpoints might be leading indicators of a forthcoming threat action.*

Note that this method of threat hunting isn't without its challenges. Assuming your organization uses DHCP, using IP addresses as the basis for monitoring, ensuring which host is involved with a logon or connection is going to be tough. The Security Log does not provide hostname as part of event 5156, and Sysmon only captures the hostname of an endpoint if that hostname was used as part of the initial connection.

What you can do is to filter on endpoints using your DHCP range that attempt to connect with other endpoints in the same range. Generally, only systems management applications need to establish connections with endpoints, making this one way to spot suspicious movement when DHCP is in place.

# Threat Hunt No. 7

## DNS Abuse

Because virtually all internet traffic relies on DNS, attackers leverage this protocol in a number of ways to get endpoints to connect to desired "bad guy" systems rather than the intended site.

Under normal circumstances, your endpoints should talk only to the configured DNS servers with DNS request-appropriately sized communications. From a network traffic perspective, you should see only normal TCP port 53 traffic to your internal DNS servers.

As shown in Figure 6, you can monitor for DNS abuse in a number of ways. These include monitoring for DNS traffic from endpoints directly to external servers, massive amounts of DNS traffic from a single endpoint (denoting data being exfiltrated over port 53), changes made to either the DNS configuration or the hosts file, and DNS rebinding requests.

| Basis | Source | Subsource | Guidance |
|---|---|---|---|
| Bypassed DNS Server | Firewall | Varies | Outbound DNS queries from IP address other than internal DNS servers |
| Abnormally large DNS packets | | | Baseline normal range of DNS packet size |
| Changes to etc/hosts | Endpoint | Security Log File system auditing | 4663 with "etc/hosts" |
| Changes to DNS server in IP config | Endpoint | | |
| Rebinding | Endpoint | Firewall Proxy | Victim internal system visits compromised page and begins to send an API request to the external provider of the site. When the browser attempts to refresh the connection, the attack replies with a new origin address, this time an internal address. The victim browser now sends the API command to an internal system, resulting in a malicious action. |

*Figure 6. Changes in DNS traffic and configuration settings can indicate the beginning steps of a larger attack.*

## LogRhythm Insights: DNS Rebinding

DNS Rebinding is an attack that uses the client's web browser as a victim proxy. When a user visits a compromised website or ad, malicious client-side JavaScript code is passed down to the client's browser. This code contains the malicious API commands to be performed, but the malicious activity only happens once the client needs to refresh the local DNS cache. The DNS entry for the compromised site or ad is set with a very small time to live (TTL) value, causing the client to need to refresh the DNS cache to reinitialize the session. The site then points the browser to an internal IP address, at which time the malicious JavaScript code executes against a local system that would otherwise be inaccessible from the outside.

LogRhythm can easily identify DNS rebinding attacks based on their typical reliance on the REST API, which includes the presence of filetype, username, and method parameters, JavaScript filename (.JS), and as part of the URL string.

# Threat Hunt No. 8

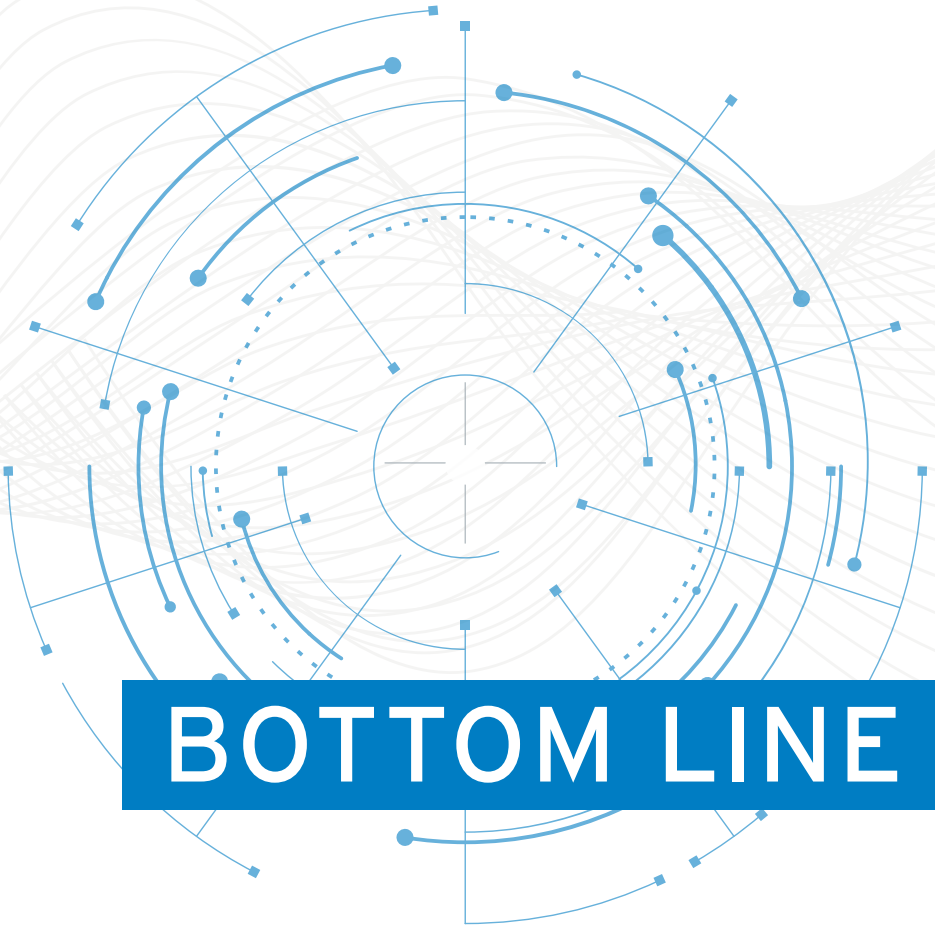
## Bait the Bad Guy

In the simplest of hunting scenarios, you can use bait to turn the predator into prey. While your intent isn't to attack the attackers, baiting an attacker expands the concept of a honeypot to include accounts, files, shares, systems, and even networks as vehicles to detect attacks without putting your production environment at risk.

In concept, you decide which aspects of the environment you want to mimic, craft a virtual environment to act as the honeypot, and make that environment accessible: open vulnerable ports, weak passwords, and so on, making it more desirable to an attacker because it appears easier to crack.

The last step is to leverage nearly all the threat-hunting methods in this paper, monitoring the honeypot environment to identify attacks before the production environment is affected.

These bait environments take quite a bit of effort to implement and maintain. And you need to make substantial effort to monitor and alert attempted attacks on the environment. Why do it, then? To keep attackers from focusing on your production environment.



# BOTTOM LINE

Not every organization can afford a layered security strategy, complete with multiple solutions in place to provide state-of-the-art protection against attack. If you can't or if you want to proactively go after threats instead of waiting on automated detection, using log data and a proper SIEM solution can give you the ability to hunt for threats.

Threat hunting allows you to spot both leading and active indicators of attacks, empowering quick responses to identified threats. By engaging in threat hunting, you can better understand where your defenses are weak, how attacks are occurring, and how to properly remediate gaps in security—thereby reducing your threat surface.

## About LogRhythm

LogRhythm is a world leader in NextGen SIEM, empowering organizations on six continents to successfully reduce risk by rapidly detecting, responding to, and neutralizing damaging cyberthreats. The LogRhythm NextGen SIEM Platform combines enterprise log management, user and entity behavior analytics (UEBA), network detection and response (NDR) and security orchestration, automation, and response (SOAR) in a single end-to-end solution. The LogRhythm platform is powered by AI and our patented Machine Data Intelligence Fabric. Its seamlessly integrated solution set is designed to deliver enterprises highest-efficacy Threat Lifecycle Management (TLM) at lowest total cost of ownership (TCO). A LogRhythm-powered security operations center (SOC) helps customers measurably secure their cloud, physical, and virtual infrastructures for both IT and OT environments. Built for security professionals by security professionals, the LogRhythm NextGen SIEM Platform has won many accolades, including being positioned as a Leader in Gartner's SIEM Magic Quadrant.

www.logrhythm.com

## About the Author

### About Randy Franklin Smith

Randy Franklin Smith is an internationally recognized expert on the security and control of Windows and AD security. Randy publishes www.UltimateWindowsSecurity.com and wrote The Windows Server 2008 Security Log Revealed—the only book devoted to the Windows security log. Randy is the creator of LOGbinder software, which makes cryptic application logs understandable and available to log-management and SIEM solutions. As a Certified Information Systems Auditor, Randy performs security reviews for clients ranging from small, privately held firms to Fortune 500 companies, national, and international organizations. Randy is also a Microsoft Security Most Valuable Professional.

1.866.384.0713 // info@logrhythm.com // 4780 Pearl East Circle, Boulder CO, 80301