Configure and Monitor Forensic Events

The **Forensics** feature set enables monitoring and alerting for possible security incidents by capturing real-time forensic events and applying user-defined rules. Specifically, it enables:

- Defining of rules to specify forensic events of interest
- Defining trigger actions for matching forensic events
- Searching for specific forensic events
- Visualizing event-generating processes and their full lineages

Warning When the **Forensics** feature is enabled, the software agents may consume additional host resources depending on the agent configuration. See the Software Agent Config section.

- Compatibility, on page 1
- Forensics Signals, on page 2
- Forensic Configuration, on page 7
- Forensic visualization, on page 20
- Fields Displayed in Forensic Events, on page 23
- Forensic Analysis - Searchable Fields, on page 28
- Search Terms in Forensic Analysis, on page 29
- Forensics alerts, on page 35
- Forensics Score, on page 37
- PCR-Based Network Anomaly Detection, on page 39
- Process Hash Anomaly Detection, on page 46

Compatibility

The forensics signals are reported by the deep visibility agents on all platforms, except Solaris. Currently, only a few forensic signals are supported for AIX. For more information, see the Forensics signals section.

Forensics information is provided through Linux kernel APIs, Audit and syslog, Windows kernel APIs, Windows events, AIX audit system, and others. In general, OS vendors guarantee compatibility within a major release. However, it is possible that APIs could differ slightly across platforms and minor releases, as OS vendors may backport features and fixes. As a result, some forensic event types might not be available on

some platforms. Also, the agent does not attempt to recover or enable any OS services that are disabled when the agent starts.

For example, there are number of forensic signals that use the Linux Audit Framework. If forensics are enabled, a deep visibility agent will insert Secure Workload audit rules into the system after the agent starts. The rule insertion requires the system to have the augenrules utility that is installed and /etc/audit/rules.d directory. If any of these prerequisites are not satisfied, Secure Workload audit rules will not be inserted. As a result, Forensics signals including File Access and Raw Socket Creation will not be reported.

If a user has enabled forensics previously and disables it, the agent removes the audit rules that are inserted by Secure Workload. On Red Hat 7.3 and CentOS 7.3, we observed an operating system bug that may impact the rule removal process. The agent removes the audit rules by: 1. Removing the taau.rules in */etc/audit/rules.d/* 2. Running *$service auditd restart*. The OS regenerates the ruleset based on the *audit.rules* and **.rules* files in */etc/audit/rules.d/*. Then auditd will load the rules into the system.

The operating system adds *-D* at the beginning of /etc/audit/rules.d/audit.rules file to clear all the rules before inserting the new ruleset. However, on Red Hat 7.3 and CentOS 7.3 machines the */etc/audit/rules.d/audit.rules* may not have *-D*. This is because the OS creates an empty */etc/audit/rules.d/audit.rules* file if this file does not exist and a default rule file in the subdirectory of */usr/share/doc/audit-<version>/* does not exist either, for example, */usr/share/doc/audit- 2.8.4/rules/10-base-config.rules* is one possible default rule location. The exact OS behavior can be observed from the RPM update script by running *$rpm -qf -scripts /etc/audit/rules.d*

In Linux, some forensics signals rely on the observation of 64-bit system calls. 32-bit Linux system calls are not supported in the current release.

Based on the Compatibility Matrix, here's a list of operating systems (OS) supported on agents:

- Linux: supported on all versions
- Windows: all supported versions
- AIX: supported on AIX 7.2 and Power8 or greater
- Solaris: supported on x86_64

Forensics Signals

The **Forensics** feature must be enabled for software agents to capture and report forensic events. The feature can be enabled in Software Agent Config. For more information, see the Software Agent Config section.

When the **Forensics** feature is enabled, the agent reports the following forensic events.

Signal	Description
Privilege Escalation	Privilege escalations, such as commands executed with sudo.
User Logon	User login events.
User Logon Failed	User login failed attempts.
Shellcode	Suspicious shell executions resembling shellcode attempts.
File Access	Accesses on sensitive files such as password files.

Signal	Description
User Account	Adding or removing user accounts.
Unseen Command	New commands that the agent has not seen. Users can use the command anomaly score to tune results based on scope. See Unseen Command for details.
Unseen Library	New library that agent have not seen process that is loaded before.
Raw Socket Creation	Processes creating raw sockets. For example, port knocking.
Binary Changed	Changes to hash values or modification times of known binaries.
Library Changed	Changes to hash values or modification times of known libraries.
Side Channel	Side channel attack attempts (Meltdown).
Follow User Logon	Descendant processes forked or executed after the login events.
Follow Process	Follow Process events report processes that match user forensic config rules based on process attributes such as binary path, command string, and others.
Network Anomaly	Anomalies in network traffic of the workload, see PCR-Based Network Anomaly Detection for more information.

Table 1: Forensic Signals Supported on AIX

Signal	Description
Privilege Escalation	Privilege escalations, such as commands executed with sudo.
Raw Socket Creation	Processes creating raw sockets. For example, port knocking.
User Account	Adding or removing user accounts.

Privilege Escalation

When the process changes its privilege from low to high, it is considered a Privilege Escalation. In Linux, this means the user-id of the process has changed from non zero to zero. There are legitimate cases such as changing the password for a normal user and other special-purpose binaries such as sudo. This event is currently not available in Windows. Privilege escalation in Windows is typically done through other mechanisms rather

than changing the privilege of the process itself, i.e., integrity level. Privilege escalations on Windows are covered by other types of forensics events, such as unseen commands or binary changes.

User Log on

User log on events including SSH, RDP, and other types of logons. Whenever available, sensors captures who, when, and how a user logs in. For example, for SSH in Linux, sensors report username, authentication type (password, public), and source IP.

User Log on Failed

Similar to User Log on events above, sensors report failed attempts to log in with similar information whenever available.

Shellcode

Shellcode events have different interpretations in Linux and Windows. In Linux, sensors identify processes running as interactive shell without a login session or terminal. (There are no good reasons for interactive shell running outside of a login session.) In this release, detection of shellcode events is limited in that it assumes the attack will utilize a shell already available in the system. If an attack uploads new binaries, sensors flag these binaries as either unseen commands or binary changes, if they replace existing binaries. In Windows, every process that is linked with the PowerShell DLL will be labeled as shellcode. Users can create rules to filter out legitimate cases.

File Access

File Access events report accesses to sensitive files, such as password files. In this release, the list of files to be monitored cannot be changed by users. In Linux, the sensor monitors write access to /etc/passwd. Sensor also monitors read and write accesses to /etc/shadow. Windows will not trigger this event in this release.

User Account

User Account events report the creation of local user accounts whenever the information is available.

Unseen Command

Unseen Command events report commands that the sensor has not seen before. Unseen command is defined as an unseen transition/edge from a parent to a child process. For example, assuming a web server (httpd) is executing a CGI script that is called abc.sh, when the sensor sees it for the first time, it will report abc.sh as unseen command. Subsequent executions of abc.sh by the web server will not result in forensic events since the sensor has seen and reported it before. If a service or process never executes any binary, an unseen command event from that service/process indicates a possible compromise. Note that sensors are stateless across restarts, so a previously seen command will be reported again after a sensor restart.

Since 3.4, for SaaS clusters, each Unseen Command event is associated with a command anomaly score ranging from 0.0 to 1.0. The lower the score, the more anomalous the transition is. The command transitions, that is, the tuples (parent command line, command line), are cross-checked for anomalous transitions among those events having the same tuple below:

- The narrowest scopes that the sensor belongs to. For example, the unseen command event is observed on workload W which belongs to the following scope lineages: `Root Scope -> A -> B -> C and Root Scope -> D -> E`. Then, the command is cross-checked among all workloads in scopes C and E (Note that C and E can be either overlapping or nonoverlapping). The anomaly score of the event is the maximum of the anomaly scores of the event regarding those 2 scopes.
- The execution path of the running process.
- The execution path of the parent process.
- The binary hash of the running process.

A score 1.0 means the same command transition having the same tuple `(narrowest scope, execution path, parent execution path, binary hash)` has been seen. A score 0.0 means such command transition with such execution path, parent execution path and binary hash of the running process has never been observed on any hosts within the same scopes. The anomaly score can be used to suppress similar unseen command alerts from firing within the same scope and reduce false positives. See Default Secure Workload Rules for an example of how this score can be used.

Note The anomaly score is only available for SaaS clusters from 3.4 and later.

Unseen Library

Unseen Library events report libraries that the sensor has not seen a process that is loaded before. An unseen library is defined as an unseen pair of binary execution path and library path. For example, an application usually loads a relatively stable list of libraries. An attacker who has access to the machine may restart the application and LD_PRELOAD malicious libraries. When the sensor sees the newly loaded malicious libraries in this application binary execution path for the first time, it reports unseen library events. Subsequent load of the malicious libraries will not result in forensic events since the sensor has seen and reported it before. Legitimate cases include application loads new libraries after upgrade or applications dynamically load new libraries. Note that sensors might report a previously seen library again after restart.

Note that this is an experimental feature and is subject to change in future releases.

Raw Socket Creation

Raw Socket Creation events are only supported on Linux in this release. Raw sockets are typically used to snoop or inject/spoof traffic. There are legitimate uses of raw sockets, such as in diagnosis tools like tcpdump, or when crafting special IP packets like ping or arp. Malicious uses include stealth scans to avoid logging by target/victim machines, malware port knocking, and so on. Secure Workload sensors also create raw sockets for collecting flow-related information. (For consistency, sensors do not suppress events that are triggered by their own flow information collection.)

Binary Changed

Binary Changed events report changes to the file contents and attributes of binaries for running processes. Sensors record the file attributes of every running process. If a process runs a binary at the same path, but with different file attributes (ctime, mtime, size, or hash), the sensor flags the process as a binary change. Legitimate cases include application upgrade.

Library Changed

Library Changed events report changes to the file contents and attributes of libraries for running processes. Sensors record the file attributes of loaded libraries. If a process loads a library at the same path, but with different file attributes (ctime, mtime, size, or hash), the sensor will flag the process with a library change. Legitimate cases include library upgrade.

Note that this is an experimental feature and is subject to change in future releases.

Side Channel

Side Channel events report running software that exploits side channel vulnerabilities. This release provides one side channel detection capability on selected Linux platform: Meltdown. See the details below for supported machine configurations. These are advanced security features and therefore disabled by default. Users should expect to see increased CPU usage when side channel reporting is enabled. The CPU quota that is configured in the UI will still be honored. If the forensic collection subprocess of the sensor determines that its CPU usage is too high for too long, it shuts down, and the parent sensor process will restart it with a small delay. Enabling this feature on old or unsupported kernels could lead to system instability. Testing in similar nonproduction environments is recommended.

This feature can be turned on/off from the agent config page in the UI and they can be turned on/off in each agent config profile.

Meltdown is a side channel attack that abuses the speculative execution and cache features in the CPU (https://meltdownattack.com/). It allows an attacker to read privileged-domain data from an unprivileged domain, for example, reading kernel memory from a user space application without ring 0 privileges. Meltdown detection currently supports CentOS 7 and Ubuntu 16.04.

Follow User Logon

Follow User Logon events report descendant processes (up to 4 levels) that are executed after a User Logon event process (SSH, RDP, and so on.). Processes reported under this Follow User Logon event are for auditing purposes and not necessary having any security events.

Follow Process

Follow Process events report processes that match user forensic config rules based on process attributes such as binary path, command string, and so on. Processes that are reported under this Follow Process event are for auditing purposes and not necessary having any security events.

Example 1: Report processes that are run by cmd.exe or powershell.exe

Event Type = Follow Process AND (Process Info - Exec Path contains cmd.exe OR Process Info - Exec Path contains powershell.exe)

Example 2: Report any processes which are created by winword.exe or excel.exe or powerpnt.exe.

Event Type = Follow Process with_ancestor (Process Info - Exec Path contains winword.exe OR Process Info - Exec Path contains excel.exe OR Process Info - Exec Path contains powerpnt.exe)

Note: Follow Process events can be tracked by one of the following process signals:

- Process Info - Exec Path

- Process Info - Command String
- Process Info - Username
- Follow Process - Parent Exec Path
- Follow Process - Parent Command String
- Follow Process - Parent Username

Forensic Configuration

Forensics feature uses intent-based configuration. Intents specify how to apply forensic profiles to inventory filters. Forensic profile consists of multiple forensic rules. Profiles in an intent are applied in order from top to bottom.

Forensic Rules

Note The maximum number of rules per root scope is 100.

Adding a Forensic Rule

This section explains how to add new forensic rules.

Before You Begin

You must login as **Site Admin**, **Customer Support** or **Scope Owner** in the system.

Procedure

Step 1 In the navigation bar on the left, click **Defend** > **Forensic Rules**.

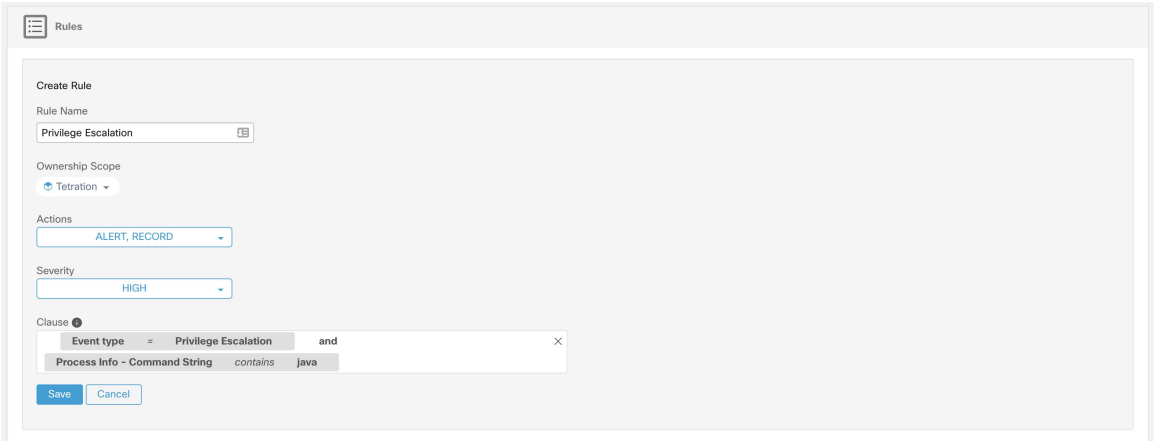
Step 2 Click **Create Rule**.

Step 3 Enter the appropriate values in the following fields.

Field	Description
Rule Name	Enter a name for the rule. Name cannot be empty.
Ownership scope	Enter an ownership scope for this rule.
Actions	Select actions when this rule is triggered. **Record** means matching security events persist for further analysis. **Alert** action means to publish matching security events to Secure Workload Alert system.
Severity	Select severity level of this rule: **LOW**, **MEDIUM**, **HIGH**, **CRITICAL** or **REQUIRES IMMEDIATE ACTION**

Field	Description
Clause	Enter a rule clause. A clause must contain security event signals from either a process forensic event or a workload event. A clause is invalid if it contains both process and workload signals.

Figure 1: Create rule



Step 4 Click **Save**.

Basic Forensic Rule Composition

A forensic rule must contain **exactly one** forensic event type (for example, **Event Type == Unseen Command**). The following optional clauses uses attributes of that event (for example, **Unseen Command - Parent Uptime**).

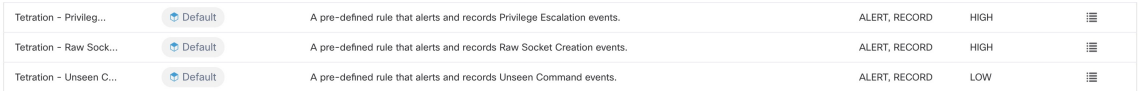
Below is an example of using **Unseen Command** event type. For more examples, see the default rules and MITRE rules.

EventType = Unseen Command and Unseen Command - Parent Uptime (microseconds) >= 60000000.

Default Secure Workload Rules

Default Secure Workload rules are provided to help the users to construct rules that are meaningful in their environment. These rules are displayed in the forensic config page and they are not editable. The rules are available in all root scopes.

Figure 2: Default Rules



The Secure Workload forensic rules:

1. **Name** Secure Workload - Privilege Escalation

   Clause EventType = Privilege Escalation and (**ProcessInfo - ExecPath** *doesn't contain* **sudo and ProcessInfo - ExecPath** *doesn't contain* **ping and Privilege Escalation Is≠ Type - Suid Binary**)

Description. This rule reports privilege escalation events that are not generated by setuid binaries. To reliably filter out the setuid binaries, it also filters out **sudo** and **ping** based on "ProcessInfo - ExecPath". Secure Workload users can also filter out other setuid binaries by defining their own rules.

2. **Name** Tetration - Unseen Command

   Clause EventType = Unseen Command and Unseen Command - Parent Uptime (microseconds) >= 60000000 or ProcessInfo - ExecPath *contains* **/bash or ProcessInfo - ExecPath** *contains* **/sh or ProcessInfo - ExecPath** *contains* **/ksh or Parent - ExecPath** *contains* **httpd or Parent - ExecPath** *contains* **apache or Parent - ExecPath** *contains* **nginx or Parent - ExecPath** *contains* **haproxy**

   Description. This rule reports unseen command events that match one of the following criteria:

   a. Process parent is alive for more than **60,000,000** microseconds.

   b. Process ExecPath contains some type of shell, for example, **/bash**, **/sh**, and **/ksh**.

   c. Process parent ExecPath contains some type of server application, for example, **httpd**, **apache**, **nginx**, and **haproxy**.

3. **Name** Tetration - Raw Socket

   Clause EventType = Raw Socket Creation and (Raw Socket - ExecPath *doesn't contain* **ping and Raw Socket - ExecPath** *doesn't contain* **iptables and Raw Socket - ExecPath** *doesn't contain* **xtables-multi**)

   Description This rule reports raw socket creation events that are not generated by **ping** and **iptables**. Secure Workload users can also filter out other binaries by defining their own rules.

4. **Name** Tetration - Network Anomaly with Unseen Command

   Clause EventType = Network Anomaly and Network Anomaly - Unseen Command Count > 3 and Network Anomaly - Non-seasonal Deviation > 0

   Description This rule reports network anomaly events that match the following criteria:

   a. There are more than 3 Unseen Command events on the same workload within 15 minutes.

   b. The Rule Attributes is greater than 0 (which also means it is greater than or equal to 6.0 because 6.0 is the minimum reported deviation for all network anomaly events).

5. **Name** Tetration - Anomalous Unseen Command

   Clause EventType = Unseen Command and Unseen Command - Anomaly - Score < 0.6

   Description This rule reports unseen command events whose anomaly score is less than 0.6. This means that only highly anomalous events whose commands do not look similar to previously observed commands are reported. The threshold 0.6 is decided based on Secure Workload's experiments on how similar commands are at different thresholds. See Unseen Command for a detailed explanation of the score.

6. **Name** Tetration - Unusual Parent of smss

   Clause EventType = Follow Process and ProcessInfo - ExecPath *contains* **smss.exe and (Follow Process - ParentExecPath** *doesn't contain* **smss.exe and Follow Process - ParentExecPath** *doesn't contain* **System)**

   Description This rule is specific for windows. This rule alerts if smss.exe has a parent that is different from another instance of smss.exe or the System process.

7. **Name** Tetration - Unusual Parent of wininit

   Clause EventType = Follow Process and ProcessInfo - ExecPath *contains* **wininit.exe and Follow Process - ParentExecPath** *doesn't contain* **smss.exe**

   Description This rule is specific for windows. This rule alerts if wininit.exe has a parent that is different from smss.exe.

8. **Name** Tetration - Unusual Parent of RuntimeBroker

   Clause EventType = Follow Process and ProcessInfo - ExecPath *contains* **RuntimeBroker.exe and Follow Process - ParentExecPath** *doesn't contain* **svchost.exe**

   Description This rule is specific for windows. This rule alerts if RuntimeBroker.exe has a parent that is different from svchost.exe.

9. **Name** Tetration - Unusual Parent of services

   Clause EventType = Follow Process and ProcessInfo - ExecPath *contains* **services.exe and Follow Process - ParentExecPath** *doesn't contain* **wininit.exe**

   Description This rule is specific for windows. This rule alerts if services.exe has a parent that is different from wininit.exe.

10. **Name** Tetration - Unusual Parent of lsaio

    Clause EventType = Follow Process and ProcessInfo - ExecPath *contains* **lsaio.exe and Follow Process - ParentExecPath** *doesn't contain* **wininit.exe**

    Description This rule is specific for windows. This rule alerts if lsaio.exe has a parent that is different from wininit.exe.

11. **Name** Tetration - Unusual Child of lsass

    Clause (EventType = Follow Process and ProcessInfo - ExecPath *doesn't contain* **efsui.exe and ProcessInfo - ExecPath** *doesn't contain* **werfault.exe**) **with ancestor Process Info - ExecPath** *contains* **lsass.exe**

    Description This rule is specific for windows. This rule alerts if lsass.exe has any descendants that are not efsui.exe or werfault.exe.

Default MITRE ATT&CK Rules

Default MITRE ATT&CK rules are provided to alert techniques from the MITRE ATT&CK Framework (https://attack.mitre.org/). There are 24 rules pertaining to adversarial behaviour and most of them are mapped to a particular MITRE technique. The complete list of the rules is below.

1. **Name** Suspicious MS Office behavior

   Clause (Event type = Follow Process and (Process Info - Exec Path doesn't contain **Windowssplwow64.exe) and (Process Info - Exec Path** *doesn't contain* **chrome.exe) and (Process Info - Exec Path** *doesn't contain* **msip.executionhost.exe) and (Process Info - Exec Path** *doesn't contain* **msip.executionhost32.exe) and (Process Info - Exec Path** *doesn't contain* **msosync.exe) and (Process Info - Exec Path** *doesn't contain* **ofccccaupdate.exe) with ancestor (Process Info - Exec Path** *contains* **winword.exe or Process Info -Exec Path** *contains* **excel.exe or Process Info -Exec Path contains powerpnt.exe)**

   Description This rule alerts and records if Microsoft Office processes (WIN-WORD.exe/EXCEL.exe/POWERPNT.exe) create any child processes. Based on our research

we have allowed a few common child processes known to be created by these MS Office binaries, to reduce the number of false positives.

2. **Name** T1015 - Accessibility features 1

   Clause Event type = Follow Process (Process Info - Exec Path *contains* **cmd.exe or Process Info -Exec Path** *contains* **powershell.exe or Process Info - Exec Path** *contains* **cscript.exe or Process Info - Exec Path** *contains* **wscript.exe) and (Follow Process - Parent Exec Path** *contains* **winlogon.exe or Follow Process - Parent Exec Path** *contains* **atbroker.exe or Follow Process - Parent Exec Path** *contains* **utilman.exe**)

   Description This rule alerts and records if any of the Accessibility features binaries (On-screen Keyboard, Magnifier, Sticky keys, and so on.) are abused and are tricked into opening cmd/powershell/cscript/wscript. The invocation of accessibility binaries is controlled by either winlogon, atbroker or utilman processes depending on from where they are invoked (from the logon screen or after a user logs in). This rule captures suspicious child processes (cmd.exe, pow- ershell.exe, cscript.exe, wscript.exe) of the accessibility processes (winlogon.exe, utilman.exe, and atbroker.exe). Use this with **T1015 - Accessibility features 2** to also catch the additional child processes of these four suspicious child processes**.

3. **Name** T1015 - Accessibility features 2

   Clause Event type = Follow Process with ancestor ((Process Info - Exec Path *contains* **cmd.exe or Process Info - Exec Path** *contains* **powershell.exe or Process Info - Exec Path** *contains* **cscript.exe or Process Info - Exec Path** *contains* **wscript.exe) and (Follow Process - Parent Exec Path** *contains* **winlogon.exe or Follow Process - Parent Exec Path** *contains* **atbroker.exe or Follow Process - Parent Exec Path** *contains* **utilman.exe))**

   Description This rule alerts and records if any of the Accessibility features binaries (On-screen Keyboard, Magnifier, Sticky keys, and so on.) are abused and are tricked into opening cmd.exe/powershell.exe/cscript.exe/wscript.exe. The invocation of accessibility binaries is controlled by either winlogon, atbroker or utilman processes depending on from where they are invoked (from the login screen or after a user logs in). This rule captures child processes of the suspicious child processes of these processes (winlogon, utilman, and atbroker). One should use this with **T1015 - Accessibility features 1** which alerts the suspicious child processes of accessibility binaries.

4. **Name** T1085 - rundll32

   Clause (Event type = Follow Process and Process Info Exec Path *does not contain* **msiexec.exe and Process Info Exec Path** *does not contain* **WindowsSystem32SystemPropertiesRemote.exe with ancestor (Process Info - Exec Path** *contains* **rundll32.exe and Follow Process - Parent Exec Path** *does not contain* **msiexec.exe and not (Process Info -command string** *contains* **Windowssystem32shell32.dll or (Process Info -command string** *contains* **Windowssyswow64shell32.dll or (Process Info -command string** *contains* **WindowsSystem32migrationWinInetPlugin.dll))**

   Description This rule alerts and records if rundll32.exe creates child processes. This binary can be called to execute arbitrary binary/dll or used by control.exe to install malicious control panel items. However, we have allowed if msiexec.exe is either the parent or the descendant of rundll32.exe. We have also permitted some of the common rundll32 commands that make use of well-known dlls.

5. **Name** T1118 - InstallUtil

   Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **installutil.exe**

   Description This rule alerts and records if InstallUtil.exe creates child processes.

6. **Name** T1121 - Regsvcs/Regasm

   Clause Event type = Follow Process and (Process Info - Exec path *does not contain* **fondue.exe or Process Info - Exec path** *does not contain* **regasm.exe or Process Info - Exec path** *does not contain* **regsvr32.exe with ancestor (Process Info - Exec Path** *contains* **regasm.exe or Process Info - Exec Path** *contains* **regsvcs.exe)**

   Description This rule alerts and records if regsvcs.exe or regasm.exe create child processes. However, we have permitted if fondue.exe/regasm.exe/regsvr32.exe is spawned by regasm.exe or regsvcs.exe to reduce the number of false positives.

7. **Name** T1127 - Trusted Developer Utilities - msbuild.exe

   Clause (Event type = Unseen Command with ancestor Process Info - Exec Path *contains* **MSBuild.exe) and (Process Info - Exec Path** *does not contain* **Tracker.exe) and (Process Info -Exec Path** *doesn't contain* **csc.exe) and (Process Info - Exec Path** *does not contain* **Microsoft Visual Studio) and (Process Info - Exec Path** *does not contain* **al.exe) and (Process Info - Exec Path** *does not contain* **lc.exe) and (Process Info - Exec Path** *does not contain* **dotnet.exe) and (Process Info - Exec Path** *does not contain* **cvtres.exe) and (Process Info - Exec Path** *does not contain* **conhost.exe) and not (Event type = Unseen Command with ancestor (Process Info - Exec Path** *contains* **Tracker.exe or Process Info - Exec Path** *contains* csc.exe or Process Info - Exec Path *contains* **Microsoft Visual Studio or Process Info - Exec Path** *contains* **al.exe or Process Info - Exec Path** *contains* **lc.exe or Process Info - Exec Path** *contains* **dotnet.exe or Process Info - Exec Path** *contains* **cvtres.exe))**

   Description This rule alerts and records if msbuild.exe creates child processes which do not belong to an allowlist of child processes it usually creates. This rule is currently Unseen Command based, as opposed to Follow Process, since Follow Process does not yet support allowing process subtrees. The current rule allows the following processes and their descendants: Tracker.exe, csc.exe, any process from "Microsoft Visual Studio" path, al.exe, lc.exe, dotnet.exe and cvtres.exe. The rule also allows conhost.exe. These processes can be seen during regular usage of MSBuild.exe (for example, compiling a project via Visual Studio). All the other descendants (not usual behavior) of MSBuild.exe are alerted.

8. **Name** T1127 - Trusted Developer Utilities - rcsi.exe

   Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **rcsi.exe**

   Description This rule alerts and records if rcsi.exe creates child processes.

9. **Name** T1127 - Trusted Developer Utilities - tracker.exe

   Clause (Event type = Unseen Command with_ancestor Process Info - Exec Path *contains* **tracker.exe) and not (Event type = Unseen Command with_ancestor Process Info - Exec Path** *contains* **MSBuild.exe)**

   Description This rule alerts and records if tracker.exe creates child processes and tracker itself is not a descendant of MSBuild.exe. Thus legitimate invocations of tracker via Visual Studio are approved, but other invocations are alerted. One limitation with the Tracker.exe and the previous MSBuild.exe rules is that if an attacker uses the MSBuild technique to create Tracker, and then make Tracker create a malicious child, it would not be alerted by either of the rules since Tracker having MSBuild as an ancestor is considered legitimate.

10. **Name** T1128 - Netsh Helper Dll

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **netsh.exe**

    Description This rule alerts and records if netsh.exe creates child processes.

11. **Name** T1136 - Create Account

    Clause Event type = User Account

    Description This rule alerts and records if a new user is created.

12. **Name** T1138 - Application Shimming

    Clause Event type = Follow Process Info - Exec Path *contains* **sdbinst.exe**

    Description This rule alerts and records if sdbinst.exe is invoked.

13. **Name** T1180 - Screensaver

    Clause Event type = Follow Process AND with ancestor Process Info - Exec Path *contains* **.scr**

    Description This rule alerts and records if a process is created with ".scr" in the exec path.

14. **Name** T1191 - CMSTP

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **cmstp.exe**

    Description This rule alerts and records if cmstp.exe creates child processes.

15. **Name** T1202 - Indirect Command Execution - forfiles.exe

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **forfiles.exe**

    Description This rule alerts and records if forfiles.exe creates child processes.

16. **Name** T1202 - Indirect Command Execution - pcalua.exe

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **pcalua.exe**

    Description This rule alerts and records if pcalua.exe creates child processes.

17. **Name** T1216 - Signed Script Proxy Execution - pubprn.vbs

    Clause Event type = Follow Process with ancestor ((Process Info - Exec Path *contains* **cscript.exe or Process Info - Exec Path** *contains* **wscript.exe) and Process Info - Command String** *contains* **.vbs and Process Info - Command String** *contains* **script)**

    Description This rule alerts and records if any vbs script is run using wscript.exe or cscript.exe, to create a new process, with a parameter "script". This technique could be used by an attacker to execute pubprn.vbs with a script parameter pointing to a malicious sct file which then gives code execution.

18. **Name** T1218 - Signed Binary Proxy Execution - msiexec.exe

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **msiexec.exe**

    Description This rule alerts and records if msiexec.exe creates child processes.

19. **Name** T1218 - Signed Binary Proxy Execution - odbcconf.exe

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **odbcconf.exe**

    Description This rule alerts and records if odbcconf.exe creates child processes.

20. **Name** T1218 - Signed Binary Proxy Execution - Register-CimProvider

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **Register-CimProvider.exe**

    Description This rule alerts and records if Register-CimProvider.exe creates child processes.

21. **Name** T1220 - XSL Script Processing - msxsl.exe

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **msxsl.exe**

    Description This rule alerts and records if msxsl.exe creates child processes.

22. **Name** T1220 - XSL Script Processing - wmic

    Clause Event type = Follow Process and (Process Info - Exec Path *contains* **wmic.exe and Process Info - Command String** *contains* **.xsl**)

    Description This rule alerts and records if an xsl script is used by wmic. This can be used to launch arbitrary binaries.

23. **Name** T1223 - Compiled HTML Files

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **hh.exe**

    Description This rule alerts and records if hh.exe creates child processes.

24. **Name** T1003 - Credential Dumping - Lsass

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **procdump.exe and Process Info - Command String** *contains* **lsass**

    Description This rule alerts and records if procdump.exe is used to dump the memory of lsass processes.

25. **Name** T1140 - Deobfuscate/Decode Files or Information

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **certutil.exe and (Process Info - Command String** *matches* **.*encode\s.* or Process Info - Command String** *matches* **.*decode\s.***

    Description This rule alerts and records if certutil.exe is used to either encode or decode a file. This technique is often used by attackers to decode their encoded payload on the victim machine.

26. **Name** T1076 - Remote Desktop Protocol

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **tscon.exe**

    Description This rule alerts and records if tscon.exe is executed. Attackers can use tscon.exe to hijack existing RDP sessions.

27. **Name** T1197 - BITS Jobs - Powershell

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **powershell.exe and Process Info - Command String** *contains* **Start-BitsTransfer**

    Description This rule alerts and records if the powershell.exe is used to run the cmdlet Start-BitsTransfer to copy/move files.

28. **Name** T1170 - MSHTA

    Clause Event type = Follow Process with ancestor Process Info - Exec Path *contains* **mshta.exe**

    Description This rule alerts and records if mshta.exe is used to run malicious HTA scripts that spawn child processes.

29. **Name** T1158 - Hidden Files and Directories

    Clause Event type = Follow Process and (Process Info - Exec Path *contains* **attrib.exe and Process Info - Command String** *contains* **+h)**

    Description This rule alerts and records if attrib.exe is used to set a file/directory as hidden.

30. **Name** T1114 - Email Collection

    Clause Event type = Follow Process (Process Info - Command String *matches* **.*.(ost|pst)(\s|"|').* or Process Info - Command String** *matches* **.*.(ost|pst)$) Process Info - Exec Path** *doesn't contain* **outlook.exe**

    Description This rule alerts and records if email files (.ost and .pst) are accessed from any other process other than outlook.exe.

31. **Name** T1070 - Indicator Removal on Host - Event Log

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **wevtutil.exe and Process Info - Command String** *matches* **.*\s(cl|clear-log)\s.***

    Description This rule alerts and records if wevtutil.exe is used to clear event logs.

32. **Name** T1070 - Indicator Removal on Host - USN

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **fsutil.exe and Process Info - Command String** *matches* **.*\susn\s.* and Process Info - Command String** *matches* **.*\sdeletejournal.***

    Description This rule alerts and records if fsutil.exe is used to delete USN journals.

33. **Name** T1053 - Scheduled Task

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **schtasks.exe and Process Info - Command String** *contains* **create**

    Description This rule alerts and records if schtasks.exe is used to create new scheduled tasks.

34. **Name** T1003 - Credential Dumping - Vaultcmd

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **vaultcmd.exe and Process Info - Command String** *matches* **.*\/list.***

    Description This rule alerts and records if vaultcmd.exe is used access Windows Credentials vault.

35. **Name** T1003 - Credential Dumping - Registry

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **reg.exe and ((Process Info - Command String** *contains* **save or Process Info - Command String** *contains* **export) and (Process Info - Command String** *contains* **hklm or Process Info - Command String** *contains* **hkey_local_machine) and (Process Info - Command String** *contains* **sam or Process Info - Command String** *contains* **security or Process Info - Command String** *contains* **system))**

    Description This rule alerts and records if reg.exe is used dump certain registry hives.

36. **Name** T1201 - Password Policy Discovery 1

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **change and Process Info - Command String** *contains* **-l**

    Description This rule alerts and records if change utility is used to list the password policy (password age policy) on a linux machine.

37. **Name** T1081 - Credentials in Files - Linux

    Clause Event type = Follow Process and (Process Info - Exec Path *contains* **cat or Process Info - Exec Path** *contains* **grep) and (Process Info - Command String** *contains* **.bash_history or Process Info - Command String** *contains* **.password or Process Info - Command String** *contains* **.passwd)**

Description This rule alerts and records if attempts are made to search for passwords stored in files on a linux machine.

38. **Name** T1081 - Credentials in Files - Windows

    Clause Event type = Follow Process and Process Info - Exec Path *contains* **findstr.exe and Process Info - Command String** *contains* **password**

    Description This rule alerts and records if attempts are made to search for passwords stored in files on a windows machine.

39. **Name** T1089 - Disabling Security Tools

    Clause Event type = Follow Process and ((Process Info - Exec Path *contains* **fltmc.exe and Process Info - Command String** *contains* **unload sysmon) or (Process Info - Exec Path** *contains* **sysmon.exe and Process Info - Command String** *contains* **/u))**

    Description This rule alerts and records if attempts are made to unload sysmon driver using fltmc.exe or sysmon.exe

Forensic profiles

Add a Profile

This section explains how to add new forensic profiles.

Before You Begin

You must log in as **Site Admin**, **Customer Support** or **Scope Owner** in the system.

Procedure

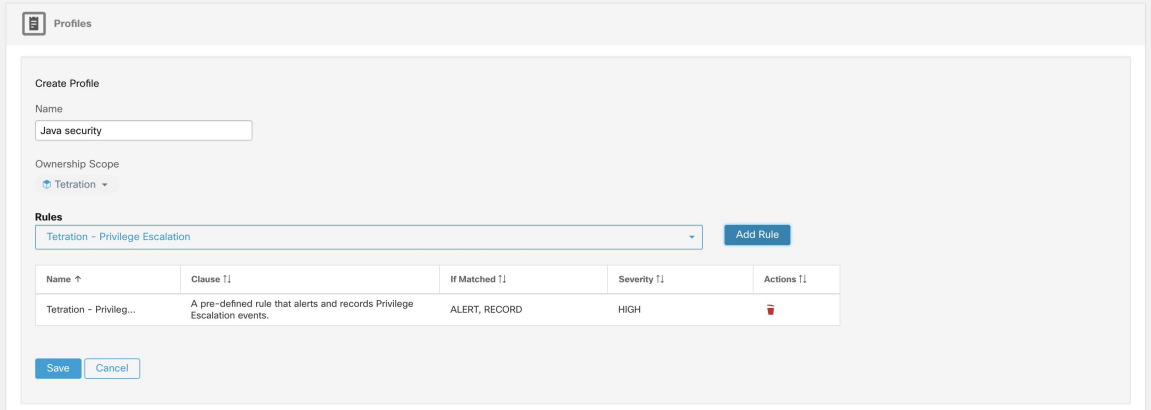
Step 1 In the navigation bar on the left, click **Defend** > **Forensic Rules**.

Step 2 Click **Create Profile**.

Step 3 Enter the appropriate values in the following fields.

Field	Description
Name	Enter a name for the profile. Name cannot be empty.
Ownership scope	Enter an ownership scope for this profile.
Rules	Add rules into this profile.

Figure 3: Create Profile



Step 4 Click **Save**.

Edit a Profile

This section explains how a user edit forensic profiles.

Before You Begin

You must login as **Site Admin**, **Customer Support** or **Scope Owner** in the system.

Procedure

Step 1 In the navigation bar on the left, click **Defend** > **Forensic Rules**.

Step 2 Find the profile you want to edit and click the **pencil** icon in the column on the right.

Step 3 Enter the appropriate values in the following fields.

Field	Description
Name	Update a name for the profile. Name cannot be empty.
Ownership scope	Update an ownership scope for this profile.
Rules	Add/remove rules into this profile.

Step 4 Click **Save**.

Clone a Profile

This section explains how a user clones forensic profiles.

Procedure

Step 1 In the navigation bar on the left, click **Defend** > **Forensic Rules**.

Step 2 Find the profile that you want to clone and click the **clone** icon in the column on the right.

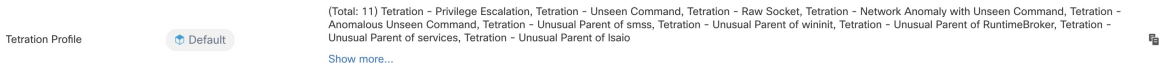
Step 3 Enter the name for the cloned profile.

Step 4 Click **Save**.

Default Profile - Secure Workload Profile

The Secure Workload profile contains 11 default forensic rules and can be added to intents. It is not editable by the user but it can be cloned. The cloned default forensic profile is editable.

Figure 4: Default profiles



Default Profile - MITRE ATT&CK Profile

The MITRE ATT&CK Profile contains 39 MITRE ATT&CK rules and can be added to intents. It is not editable by the user but it can be cloned. The cloned profile is editable. MITRE ATT&CK Profile includes the following rules:

1. Suspicious MS Office behavior
2. T1015 - Accessibility features 1
3. T1015 - Accessibility features 2
4. T1085 - rundll32
5. T1118 - InstallUtil
6. T1121 - Regsvcs/Regasm
7. T1127 - Trusted Developer Utilities - msbuild.exe
8. T1127 - Trusted Developer Utilities - rcsi.exe
9. T1127 - Trusted Developer Utilities - tracker.exe
10. T1128 - Netsh Helper Dll
11. T1136 - Create Account
12. T1138 - Application Shimming
13. T1180 - Screensaver
14. T1191 - CMSTP
15. T1202 - Indirect Command Execution - forfiles.exe
16. T1202 - Indirect Command Execution - pcalua.exe
17. T1216 - Signed Script Proxy Execution - pubprn.vbs

18. T1218 - Signed Binary Proxy Execution - msiexec.exe
19. T1218 - Signed Binary Proxy Execution - odbcconf.exe
20. T1218 - Signed Binary Proxy Execution - Register-CimProvider
21. T1220 - XSL Script Processing - msxsl.exe
22. T1220 - XSL Script Processing - wmic
23. T1223 - Compiled HTML Files
24. T1003 - Credential Dumping - Lsass
25. T1140 - Deobfuscate/Decode Files or Information
26. T1076 - Remote Desktop Protocol
27. T1197 - BITS Jobs - Powershell
28. T1170 - MSHTA
29. T1158 - Hidden Files and Directories
30. T1114 - Email Collection
31. T1070 - Indicator Removal on Host - Event Log
32. T1070 - Indicator Removal on Host - USN
33. T1053 - Scheduled Task
34. T1003 - Credential Dumping - Vaultcmd
35. T1003 - Credential Dumping - Registry
36. T1201 - Password Policy Discovery 1
37. T1081 - Credentials in Files - Linux
38. T1081 - Credentials in Files - Windows
39. T1089 - Disabling Security Tools

Change Log—Forensics

Site Admins and users with the `SCOPE_OWNER` ability on the root scope can view the change logs for each forensic rule, profile, and intent by clicking on the icon as shown below.

Figure 5: Change Log

These users can also view a list of deleted rules, profiles, and intents by clicking on the **View Deleted Rules/Profiles/Intents** link below the corresponding table.

For more information on the **Change Log**, see Change Log. Root scope owners are restricted to viewing change log entries for entities belonging to their scope.

Forensic visualization

Accessing Forensic Page

This section explains how to access forensic page.

Before You Begin

You must log in as **Site Admin**, **Customer Support** or **Scope Owner** in the system.

Procedure

Step 1 Click on **Security** link on the left panel.

Step 2 Click on **Forensics** item. Forensic page appears.

Figure 6: Security Forensic

Browsing Forensic Events

This section explains how to browse matching forensic events.

Before You Begin

You must log in as **Site Admin**, **Customer Support** or **Scope Owner** in the system and navigate to the forensic page.

Procedure

Step 1 Choose a specific range in the **Time Range Picker** at the top of the page.

Step 2 Select **Severity** drop-down.

Step 3 In **Filters**, enter filters for matching forensic events and click on **Filter Forensic Events**.

Step 4 Table of matching forensic events is updated, according to the selected time range, severity, and filters.

Note Forensic events are visible under the root scope level and will not be visible upon switching to sub/child scopes.

Inspecting a Forensic Event

This section explains how to inspect forensic events.

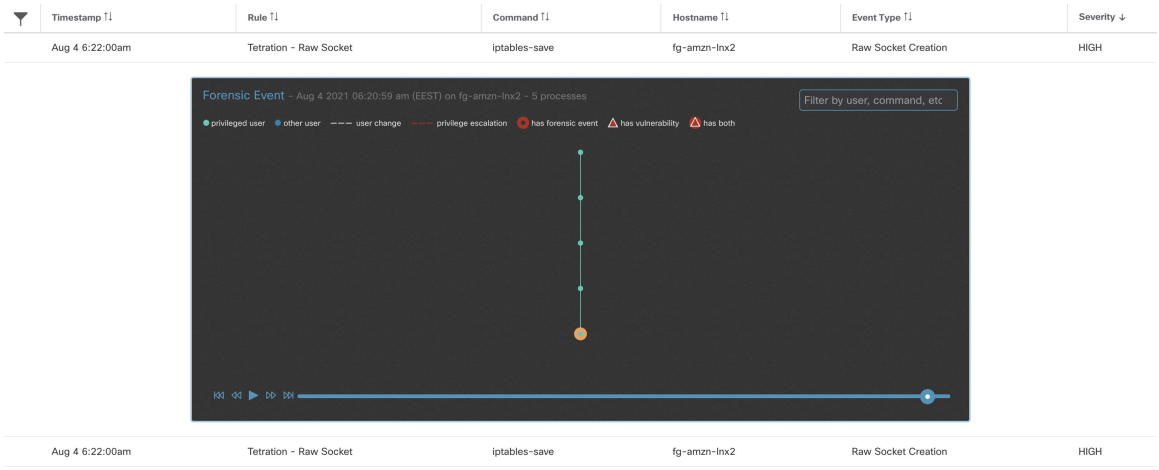
Before You Begin

You must log in as **Site Admin**, **Customer Support** or **Scope Owner (Root Scope)** in the system.

Procedure

Step 1 Click on the event to be inspected. The **Process detail** pane appears.

Figure 7: Forensic Event Table



Step 2 On lineage tree, click on process to be inspected for details.

Figure 8: Forensic Process Details

Fields Displayed in Forensic Events

Each Forensic Event has several fields which provide useful data. There are a few fields common to all the different types of forensic events, and there are a few fields unique to a particular forensic event.

Below is a list of the fields that are part of the UI. The first table describes the fields common to all forensics event, followed by a table that describes process information that is displayed with each alert and then the tables with unique fields per forensic event. Some of the fields may be present in multiple tables, because of the way the data is stored and exported.

Common Fields

Field	Description
Bin attr ctime	Changed time in linux/ Create time in windows of the binary

Field	Description
Bin attr hash	Sha256 hash of the binary
Bin attr mtime	Modified time of the binary
Bin attr name	Name of the binary on the file system
Bin attr size	Size of the binary on the file system
Bin exec path	Full path of the binary
Cmdline	Full command line of the process that gets executed
Event time usec	Time (in microseconds) when this event is observed

Process Info

Field	Description
Process ID	Process ID of the process
Parent Process ID	Process ID of the parent of the process
User	User that executed the process
Execution path	Full path of the binary that corresponds to the process.
Start time	Time when the process was started
Full command	Full command line of the process that gets executed

Privilege Escalation

Field	Description
Parent cmdline	Full command line of the parent of the process
Parent exe	Full path of the parent of the process
Parent Uptime (microseconds)	Time since the parent of the process was executed
Parent Username	User that executed the parent of the process
Types bitmap suid binary	Indicates whether the binary has the suid bit set

User Logon

Field	Description
Auth type password	Indicates password authentication

Field	Description
Auth type pubkey	Indicates key based authentication
Type login ssh	Indicates that a user logged in via ssh
Type login win batch	Indicates windows batch login (Type 4, eg schtasks)
Type login win cached	Indicates logon via cached credentials (Type 11, CachedIntetractive)
Type login win interactive	Indicates interactive logon (Type 2, eg RDP)
Type login win network cleartext	Indicates logon via ssh (Type 8)
Type login win network	Indicates network login (Type 3, eg Psexec)
Type login win new cred	Indicates the usage of new credentials (Type 9, eg Runas command)
Type login win remote interactive	Indicates remote logon (Type 10, eg RDP)
Type login win service	Indicates that a service was started by SCM (Type 5)
Type login win unlock	Indicates that the workstation was unlocked (Type 7)
Src IP	The source IP from which the login event was generated
Src Port	The source port from which the login event was generated
Username	Username associated with the log in event

User Logon Failed

Field	Description
Auth type password	Indicates password authentication
Auth type pubkey	Indicates key based authentication
Type login ssh	Indicates that a user logged in via ssh
Type login win batch	Indicates windows batch login (Type 4, eg schtasks)
Type login win cached	Indicates logon via cached credentials (Type 11, CachedIntetractive)
Type login win interactive	Indicates interactive logon (Type 2, eg RDP)
Type login win network cleartext	Indicates logon via ssh (Type 8)
Type login win network	Indicates network login (Type 3, eg Psexec)

Field	Description
Type login win new cred	Indicates the usage of new credentials (Type 9, eg Runas command)
Type login win remote interactive	Indicates remote logon (Type 10, eg RDP)
Type login win service	Indicates that a service was started by SCM (Type 5)
Type login win unlock	Indicates that the workstation was unlocked (Type 7)
Src IP	The source IP from which the login event was generated
Src Port	The source port from which the login event was generated
Username	Username associated with the log in event

Shellcode

Field	Description
Signal sources bitmap cmd as sh no tty	Indicates that a shell process has no tty that is associated with it
Signal sources bitmap powershell	Indicates that the process has powershell dll loaded (System.Management.Automation)

File Access

Field	Description
File	Full path of the file that was accessed
Perm read perm	Indicates that the file had Read permission
Perm read write perm	Indicates that the file had Read and Write permissions
Perm write perm	Indicates that the file had Write permission

User Account

Field	Description
Username	Username of the user that was created
Ops acct add	Indicates that a new account was added

Unseen Command

Field	Description
Anomaly - Score	Score (0 to 1.0) indicating how frequently the command line was seen previously, lower score implies that the command is more anomalous
Anomaly - Similarity - High	True if the anomaly score is larger than 0.8 and is smaller than 1
Anomaly - Similarity - Medium	True if the anomaly score is larger than 0.6 and is smaller than or equal to 0.8
Anomaly - Similarity - Low	True if the anomaly score is larger than 0 and is smaller than or equal to 0.6
Anomaly - Similarity - Seen	True if the anomaly score is 1, i.e. the same command has been seen before
Anomaly - Similarity - Unique	True if the anomaly score is 0, i.e. the command has never been seen before
Parent cmdline	Full command line of the parent process
Parent exepath	Binary path of the parent process
Parent uptime	Time since the parent process was executed
Parent username	Username of the user that executed the parent process
Sensor uptime	Uptime of the sensor

Unseen Library

Field	Description
Lib Path	The full path of the library file that was previously not associated to the process

Raw Socket Creation

Field	Description
Exe Path	Full path of the process that created the raw socket

Library Changed

Field	Description
Library changed name	The full path of the Library that was changed

Side Channel

Field	Description
Signal sources bitmap meltdown	Indicates the use of Meltdown exploit

Follow User Logon

Field	Description
Username	Username that executed the process

Follow Process

Field	Description
Parent cmdline	Full command line of the parent process
Parent exepath	Binary path of the parent process
Parent uptime usec	Time since the parent process was executed
Parent username	Username of the user that executed the parent process
Time since last changed usec	Time elapsed between the process start time and its binary file change time (mtime)
Username	Username of the user that executed the process

Network Anomaly

For more information, see Forensic Rules for Network Anomaly Events for the list of attributes associated with Network Anomaly events.

Forensic Analysis - Searchable Fields

The below tables describe searchable fields on the Forensics Analysis page search bar.

Miscellaneous Fields

Field	Description
Forensic Rule Name	Events labeled by a particular forensic rule
Hostname	Events from a particular hostname
Sensor ID	Events from a particular Sensor
Severity	Events of a particular severity

Search Terms in Forensic Analysis

Common Fields

These fields are common to various event types. They have the prefix “Event name - Event”, for example, “Binary Changed - Binary Attribute - CTime (epoch nanoseconds)”

Field	Description
Binary Attribute - CTime (epoch nanoseconds)	Changed time in linux/ Create time in windows of the binary
Binary Attribute - Hash	Sha256 hash of the binary
Binary Attribute - MTime (epoch nanoseconds)	Modified time of the binary
Binary Attribute - Filename	Name of the binary on the file system
Binary Attribute - Size (bytes)	Size of the binary on the file system
Event Binary Path	Full path of the binary
Command Line	Full command line of the process that gets executed

Binary Changed

There are no other search terms other than the ones described in “Common Fields” table.

File Access

File Access search terms have the prefix “File Access - “, for example, “File Access - Filename”

Field	Description
Filename	Full path of the file that was accessed
Is = Permission - Read	Indicates that the file had Read permission

Field	Description
Is = Permission - ReadWrite	Indicates that the file had Read and Write permissions
Is = Permission - Write	Indicates that the file had Write permission

Follow Process

Follow Process search terms have the prefix “Follow Process - “, for example, “Follow Process - Parent Command Line”

Field	Description
Parent Command Line	Full command line of the parent process
Parent Exec Path	Binary path of the parent process
Parent Uptime (microseconds)	Time since the parent process was executed
Parent Username	Username of the user that executed the parent process
Process Start Time Since Last File Changed (microseconds)	Time that is elapsed between process start and the most recent (corresponding) file change
Username	Username that are associated with the process being followed

Follow User Logon

Follow User Logon search terms have the prefix "Follow User Logon - ", for example, "Follow User Logon - Username"

Field	Description
Username	Username that is associated with a process

Ldap

Ldap search terms have the prefix “Ldap - “, for example, “Ldap - Department”

Field	Description
Department	AMS Ldap user department associated with the process username (if available)
Description	AMS Ldap user description associated with the process username (if available)
Username	AMS Ldap username associated with the process (if available)

Library Changed

Library Changed search terms have the prefix “Library Changed - “, for example, “Library Changed - Department”

Field	Description
Lib Filename	The full path of the Library that was changed

Privilege Escalation

Privilege Escalation search terms have the prefix “Privilege Escalation - “, for example, “Privilege Escalation - Parent Com- mand Line”

Field	Description
Parent Command Line	Full command line of the parent of the process
Parent Exec Path	Full path of the parent of the process
Parent Uptime (microseconds)	Time since the parent of the process was executed
Parent Username	User that executed the parent of the process
Type - Suid Binary	Indicates whether the binary has the suid bit set

Process Info

Process Info search terms have the prefix “Process Info - “, for example, “Process Info - Binary Hash”

Field	Description
Binary Hash	Hash of the binary associated with the process
Command String Tokenized	Tokenized command line of the process
Command String	Full command line of the process
Exec Path	Full path of the binary that corresponds to the process

Raw Socket

Raw Socket search terms have the prefix “Raw Socket - ”, for example, “Raw Socket - Exec Path”

Field	Description
Exec Path	Full path of the process that created the raw socket

Shellcode

Shellcode search terms have the prefix “Shellcode - ”, for example, “Shellcode - Source - Not From Login”

Field	Description
Source - Not From Login	Indicates that a shell process has no tty that is associated with it
Source - Powershell	Indicates that the process has powershell dll loaded (System.Management.Automation)

Side Channel

Side Channel search terms have the prefix “Shellcode - ”, for example, “Shellcode - Source - Meltdown”

Field	Description
Source - Meltdown	Indicates the use of Meltdown exploit

Unseen Command

Unseen Command search terms have the prefix “Unseen Command - ”, for example, “Unseen Command - Anomaly - Similarity - High”

Field	Description
Anomaly - Score	Score (0 to 1.0) indicating how frequently the command line was seen previously, lower score implies that the command is more anomalous
Anomaly - Similarity - High	True if the anomaly score is larger than 0.8 and is smaller than 1
Anomaly - Similarity - Medium	True if the anomaly score is larger than 0.6 and is smaller than or equal to 0.8
Anomaly - Similarity - Low	True if the anomaly score is larger than 0 and is smaller than or equal to 0.6
Anomaly - Similarity - Seen	True if the anomaly score is 1, i.e. the same command has been seen before
Anomaly - Similarity - Unique	True if the anomaly score is 0, i.e. the command has never been seen before
Parent Cmdline	Full command line of the parent process
Parent Exepath	Binary path of the parent process
Parent Uptime	Time since the parent process was executed

Field	Description
Parent Username	Username of the user that executed the parent process
Sensor Uptime	Uptime of the sensor
Anomaly - Latest Similar Commands	5 latest previously observed command which are similar to the command of the event

Unseen Library

Unseen Library search terms have the prefix “Unseen Library - ”, for example, “Unseen Library - Lib Filename”

Field	Description
Lib Filename	The full path of the library file that was previously not associated to the process

User Account

User Account search terms have the prefix “User Account - ”, for example, “User Account - Account Name”

Field	Description
Account Name	Username of the user that was created
Operation - Add Account	Indicates that a new account was added

User Logon

User Logon search terms have the prefix “User Logon - ”, for example, “User Logon - Auth Type - Password”

Field	Description
Auth Type - Password	Indicates password authentication
Auth type - Pubkey	Indicates key based authentication
Login Type - Login Via SSH	Indicates that a user logged in via ssh
Login Type - Windows Login Batch	Indicates windows batch login (Type 4, eg schtasks)
Login Type - Windows Login Cached	Indicates logon via cached credentials (Type 11, CachedIntetractive)
Login Type - Windows Login Interactive	Indicates interactive logon (Type 2, eg RDP)
Login Type - Windows Network Cleartext	Indicates logon via ssh (Type 8)
Login Type - Windows Network	Indicates network login (Type 3, eg Psexec)

Field	Description
Login Type - Windows Login New Credential	Indicates the usage of new credentials (Type 9, eg Runas command)
Login Type - Windows Login Remote Interactive	Indicates remote logon (Type 10, eg RDP)
Login Type - Windows Login Service	Indicates that a service was started by SCM (Type 5)
Login Type - Windows Login Unlock	Indicates that the workstation was unlocked (Type 7)
Source IP	The source IP from which the login event was generated
Source Port	The source port from which the login event was generated
Username	Username associated with the log in event

User Logon Failed

User Logon Failed search terms have the prefix “User Logon Failed - “, e.g., “User Logon Failed - Auth Type - Password”

Field	Description
Auth Type - Password	Indicates password authentication
Auth type - Pubkey	Indicates key based authentication
Login Type - Login Via SSH	Indicates that a user logged in via ssh
Login Type - Windows Login Batch	Indicates windows batch login (Type 4, eg schtasks)
Login Type - Windows Login Cached	Indicates logon via cached credentials (Type 11, CachedIntetractive)
Login Type - Windows Login Interactive	Indicates interactive logon (Type 2, eg RDP)
Login Type - Windows Network Cleartext	Indicates logon via ssh (Type 8)
Login Type - Windows Network	Indicates network login (Type 3, eg Psexec)
Login Type - Windows Login New Credential	Indicates the usage of new credentials (Type 9, eg Runas command)
Login Type - Windows Login Remote Interactive	Indicates remote logon (Type 10, eg RDP)
Login Type - Windows Login Service	Indicates that a service was started by SCM (Type 5)
Login Type - Windows Login Unlock	Indicates that the workstation was unlocked (Type 7)
Source IP	The source IP from which the login event was generated

Field	Description
Source Port	The source port from which the login event was generated
Username	Username associated with the log in event

Forensics alerts

Forensic events can be found in the Secure Workload Alert System if their matching rules contain an **Alert** action.

Accessing Forensic Alerts

This section explains how to access forensic alerts.

Before You Begin

- Login into the system as a **Site Admin**, **Customer Support** or **Scope Owner**.
- Turn on alerts for **Forensics** alert source.

Procedure

Step 1 From the navigation pane, choose **Configure Alerts**.

Step 2 Alert page appears.

Checking Alert Details

Before You Begin:

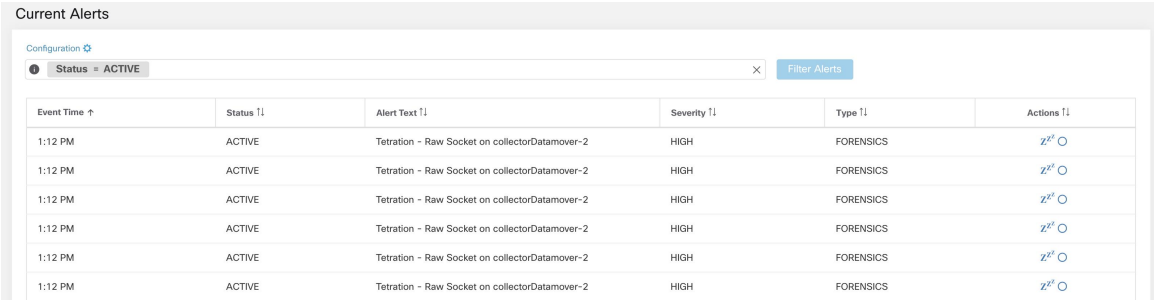
You must log in as **Site Admin**, **Customer Support** or **Scope Owner** in the system.

Procedure

Step 1 From the alert page, click on the alert to be checked.

Step 2 Click on profile/rule to see the details of the matching forensic profile/rule. If the matching profile/rule is updated after alerts are raised, there will be a warning indicator.

Figure 9: Forensic Alert Page



In addition, you can snooze or include/exclude an alert. Refer to the section Current Alerts for more details.

External Integration

Forensics alerts can be sent to external monitoring tools such as syslog. The forensics alert is sent in JSON format. The JSON field definitions are defined in the section “Fields Displayed in Forensic Events” above.

A sample JSON Kafka output is shown below:


```
  {
    "severity": "HIGH",
    "tenant_id": 0,
    "alert_time": 1595573847156,
    "alert_text": "Tetration - Anomalous Unseen Command on collectorDatamover-1",
    "key_id":
"d89f926cddc7577553eb8954e492528433b2d08e:5efcfdf5497d4f474f1707c2:5efcfdf6497d4f474f1707d6:20196:CMD_NOT_SEEN",

    "alert_id": "/Alerts/5efcfdf5497d4f474f1707c2/DataSource{location_type='TETRATION',
location_name='forensics', location_grain='MIN',
root_scope_id='5efcfdf5497d4f474f1707c2'}/db10d21631eebefc3b8d3aeaba5a0b1b45f4259e85b591763d7eaee9161ca076",

    "root_scope_id": "5efcfdf5497d4f474f1707c2",
    "type": "FORENSICS",
    "event_time": 1595573795135,
    "alert_details": "{\"Sensor
Id\":\"d89f926cddc7577553eb8954e492528433b2d08e\",\"Hostname\":\"collectorDatamover-1\",\"Process
 Id\":20196,\"scope_id\":\"5efcfdf5497d4f474f1707c2\",\"forensic\":{\"Unseen
Command\":\"true\",\"Unseen Command - Sensor Uptime (microseconds)\":\"34441125356\",\"Unseen
 Command - Parent Uptime (microseconds)\":\"35968418683\",\"Unseen Command - Parent
Username\":\"root\",\"Unseen Command - Parent Command Line\":\"svlogd -tt
/local/logs/tetration/efe/ \",\"Unseen Command - Parent Exec Path\":\"/sbin/svlogd\",\"Unseen
 Command - Anomaly - Score\":\"0\",\"Unseen Command - Anomaly - Similarity -
Unique\":\"true\",\"Process Info - Command String\":\"gzip \",\"Process Info - Exec
Path\":\"/bin/gzip\"},\"profile\":{\"id\":\"5efcfdf6497d4f474f1707e4\",\"name\":\"Tetration

Profile\",\"created_at\":1593638390,\"updated_at\":1593638390,\"root_app_scope_id\":\"5efcfdf5497d4f474f1707c2\"},\"rule\":{\"id\":\"5efcfdf6497d4f474f1707d6\",\"name\":\"Tetration
 - Anomalous Unseen
Command\",\"clause_chips\":\"[{\\\"type\\\":\\\"filter\\\",\\\"facet\\\":{\\\"field\\\":\\\"event_type\\\",\\\"title\\\":\\\"Event

type\\\",\\\"type\\\":\\\"STRING\\\"},\\\"operator\\\":{\\\"label\\\":\\\"\\u003d\\\",\\\"type\\\":\\\"eq\\\"},\\\"displayValue\\\":\\\"Unseen
 Command\\\",\\\"value\\\":\\\"Unseen
Command\\\"},{\\\"type\\\":\\\"filter\\\",\\\"facet\\\":{\\\"field\\\":\\\"forensic_event_cmd_not_seen_data_cmdline_anomaly_info_score\\\",\\\"title\\\":\\\"Unseen
 Command - Anomaly -
Score\\\",\\\"type\\\":\\\"NUMBER\\\",\\\"operator\\\":{\\\"label\\\":\\\"\\u003d\\\",\\\"type\\\":\\\"1\\\",\\\"displayValue\\\":\\\"0\\\",\\\"value\\\":\\\"0\\\"}\",\"created_at\":1593638390,\"updated_at\":1595598,\"root_app_scope_id\":\"5efcfdf5497d4f474f1707c2\"}"
```

```
}
```

The value in alert_details is itself an escaped JSON string whose content for the above alert can be seen below:

```
{
  "Sensor Id": "d89f926cddc7577553eb8954e492528433b2d08e",
  "Hostname": "collectorDatamover-1",
  "Process Id": 20196,
  "scope_id": "5efcfdf5497d4f474f1707c2",
  "forensic": {
    "Unseen Command": "true",
    "Unseen Command - Sensor Uptime (microseconds)": "34441125356",
    "Unseen Command - Parent Uptime (microseconds)": "35968418683",
    "Unseen Command - Parent Username": "root",
    "Unseen Command - Parent Command Line": "svlogd -tt /local/logs/tetration/efe/ ",
    "Unseen Command - Parent Exec Path": "/sbin/svlogd",
    "Unseen Command - Anomaly - Score": "0",
    "Unseen Command - Anomaly - Similarity - Unique": "true",
    "Process Info - Command String": "gzip ",
    "Process Info - Exec Path": "/bin/gzip"
  },
  "profile": {
    "id": "5efcfdf6497d4f474f1707e4",
    "name": "Tetration Profile",
    "created_at": 1593638390,
    "updated_at": 1593638390,
    "root_app_scope_id": "5efcfdf5497d4f474f1707c2"
  },
  "rule": {
    "id": "5efcfdf6497d4f474f1707d6",
    "name": "Tetration - Anomalous Unseen Command",
    "clause_chips":
"[{\"type\":\"filter\",\"facet\":{\"field\":\"event_type\",\"title\":\"Event
type\",\"type\":\"STRING\"},\"operator\":{\"label\":\"=\",\"type\":\"eq\"},\"displayValue\":\"Unseen
 Command\",\"value\":\"Unseen
Command\"},{\"type\":\"filter\",\"facet\":{\"field\":\"forensic_event__cmd_not_seen_data__cmdline_anomaly_info__score\",\"title\":\"Unseen
 Command - Anomaly -
Score\",\"type\":\"NUMBER\"},\"operator\":{\"label\":\"<\",\"type\":\"lt\"},\"displayValue\":\"0.6\",\"value\":\"0.6\"}]",

    "created_at": 1593638390,
    "updated_at": 1595539498,
    "root_app_scope_id": "5efcfdf5497d4f474f1707c2"
  }
}
```

The details of the forensic events are included in the field forensic. For the list of attributes of the forensic events, see Fields Displayed in Forensic Events. These attributes are also shown in the alert details in the UI.

Forensics Score

Where to See Forensic Score

Security Dashboard:

Figure 10: Forensics Score Section in Security Dashboard

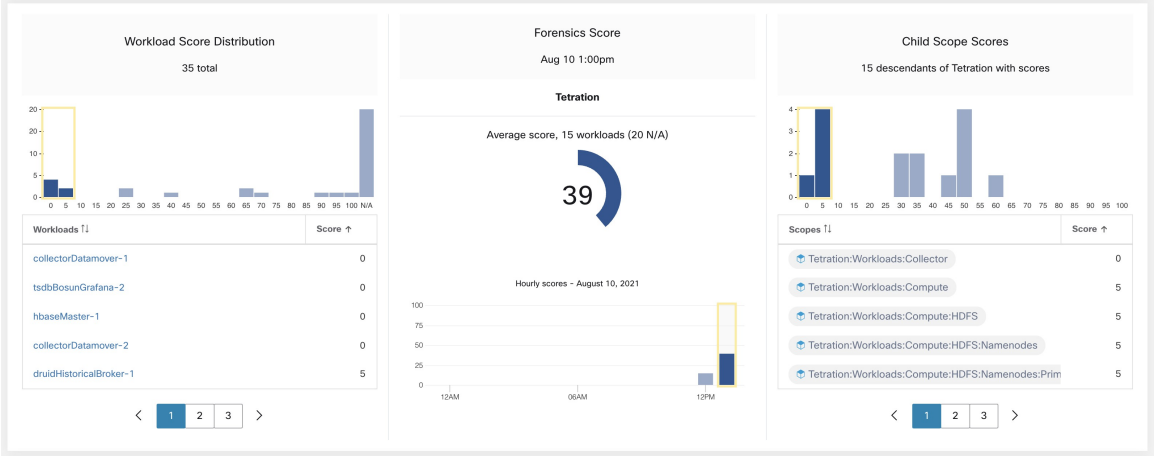


Figure 11: Forensics Score Details Section in Security Dashboard

9 Forensic Events

Timestamp	Rule	Command	Hostname	Event Type	Severity
Aug 10 1:00:00pm	Tetration - Unseen Command	/bin/sh (ps	zookeeper-1	Unseen Command	LOW
Aug 10 1:00:00pm	Tetration - Unseen Command	/bin/bash /usr/bin/atopd	zookeeper-1	Unseen Command	LOW
Aug 10 1:00:00pm	Tetration - Unseen Command	/bin/sh (/usr/sbin/ntpq	zookeeper-1	Unseen Command	LOW
Aug 10 1:00:00pm	Tetration - Unseen Command	/bin/bash /etc/rc.d/init.d/atop	zookeeper-1	Unseen Command	LOW
Aug 10 1:00:00pm	Tetration - Unseen Command	/bin/bash ulimit	zookeeper-1	Unseen Command	LOW
Aug 10 1:01:00pm	Tetration - Unseen Command	/bin/bash /etc/cron.hourly/0anacro	zookeeper-1	Unseen Command	LOW
Aug 10 1:01:00pm	Tetration - Unseen Command	/bin/bash /usr/bin/run-parts	zookeeper-1	Unseen Command	LOW
Aug 10 1:18:00pm	Tetration - Anomalous Unseen Con	bash /usr/hdp/current/zookeeper-c	zookeeper-1	Unseen Command	HIGH
Aug 10 1:22:00pm	Tetration - Anomalous Unseen Con	pickup	zookeeper-1	Unseen Command	HIGH

How the Forensic Score is Calculated

For each Workload, we compute a Forensics Score. A Workload's Forensics Score is derived from the Forensic Events observed on that Workload based on the profiles that are enabled for this scope. A score of 100 means that no Forensic Events were observed via configured rules in enabled profiles, and a score of 0 means that there is a Forensic Event detected that requires immediate action. The Forensics Score for a Scope is the average Workload score within that Scope. Forensics Score for a given hour is a minimum of all scores within that hour.

- A Forensic Event with the severity REQUIRES IMMEDIATE ACTION reduces the Score for the entire Scope to zero.
- A Forensic Event with the severity CRITICAL reduces workload's score with the weight of 10.

- A Forensic Event with the severity HIGH reduces workload's score with the weight of 5.
- A Forensic Event with the severity MEDIUM reduces workload's score with the weight of 3.
- A Forensic Event with the severity LOW doesn't contribute to the Forensics Score. This is recommended for new rules where the quality of the signal is still being tuned and is likely to be noisy.

For example, a workload has 3 forensic events that match 2 rules with *CRITICAL* severity, 1 rule with *HIGH* severity, 1 rule with *LOW*, respectively. The forensic score for that workload is: 100 - 1 * 10 - 1 * 5 - 1 * 0 = 85.

The Forensics Scores are N/A for workloads in which the Forensics feature is not enabled.

How to Improve Forensic Score

Tuning your Forensics Score can be done by adjusting the Forensic Rules enabled. Creating rules that are less noisy will give you a more accurate score. Acting upon and preventing legitimate Forensic Events (events that are evidence of an intrusion or other bad activity) is another good way to improve your Forensics Score.

Caveats

- Forensics Score details show all forensic events within that hour. That means Forensic Score details may show forensic events other than the ones used for computing forensic score.
- Forensics Score is currently available for Deep Visibility and Enforcement sensors.

PCR-Based Network Anomaly Detection

Network Anomaly feature detects abnormally large amounts of data flowing into or out of the workloads based on the concept of Producer Consumer Ratio (PCR). The PCR is defined as:

```
Egress app byte count - Ingress app byte count
  PCR = ------------------------------------------
        Egress app byte count + Ingress app byte count
```

The value of PCR is in the [-1.0, 1.0] range where:

- PCR = 1.0 means the workload purely sends data out.
- PCR = -1.0 means the workload purely receives data.
- PCR = 0.0 means the workload has balanced amounts of data in and data out.

Similar to other Forensics features, you can use the intent-based configuration to configure the Network Anomaly events you want to record and/or alert. Detected Network Anomaly events from workloads are exported every 5 minutes and are matched against configured rules 5 minutes later. As a result, new Network Anomaly events are only observed on the UI every 5 minutes with delay of up to 10 minutes from the time of the event.

Note In 3.2 and 3.1 versions of Secure Workload software, Network Anomaly detection was known as Data Leak detection.

Forensic Rules for Network Anomaly Events

Refer to Forensic Configuration on how to add forensic rules.

Rule Attributes

This section explains the details of the attributes to define a Network Anomaly related rule. The simplest Network Anomaly rule is:

```
Event Type = Network Anomaly
```

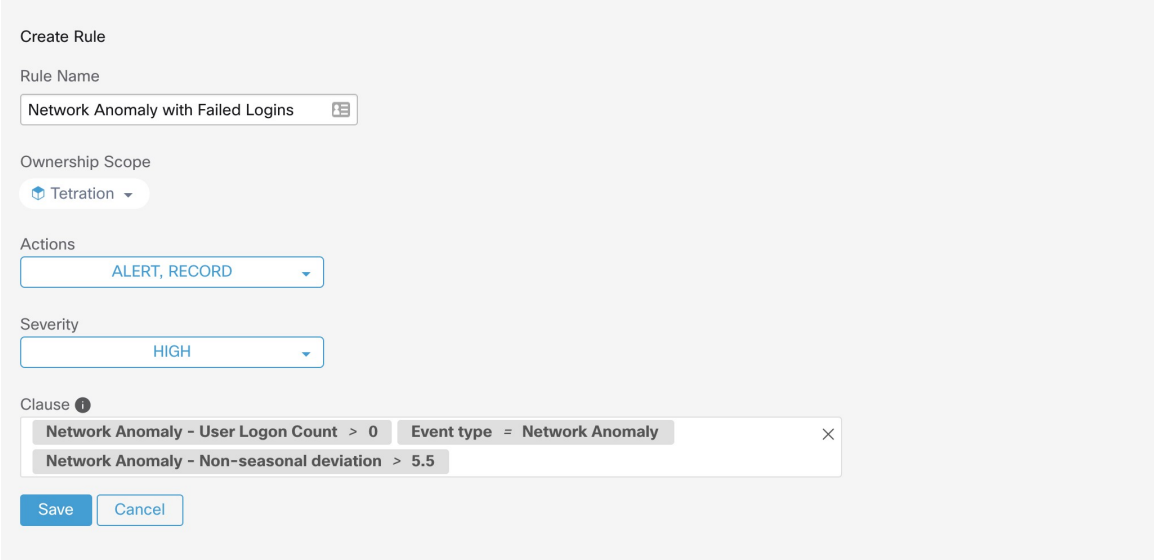
Other attributes in the Network Anomaly event to refine the rules for your data centers:

Table 2: Rule Attributes in Network Anomaly event

Attribute	Description
Host Name	The host name of the workload emitting this event.
Timestamp (epoch milliseconds)	The timestamp (in milliseconds) of the event.
PCR Deviation	The deviation of PCR from the mean at the event time as a multiple of historical standard deviation.
Non-seasonal Deviation	This is the PCR deviation after removing the seasonality pattern (for example, by cron-jobs). The value of Non-seasonal Deviation is always larger than or equal to 6.0.
PCR	The Producer-Consumer Ratio.
EIR	The Egress Ingress Ratio, which is the ratio between the total Egress App Byte Count and the Ingress App Byte Count.
Egress App Byte Count	The egress application byte count, which is the total byte count of packet contents (excluding headers) flowing out of the work-load.
Ingress App Byte Count	The ingress application byte count, which is the total byte count of packet contents (excluding headers) flowing into the work-load.
Protocol	The protocol for which the PCR time series is calculated. Currently, the supported protocols are TCP, UDP, and Aggregate. Aggregate PCR is calculated based on the total sum of TCP, UDP, and ICMP byte counts.

Attribute	Description
User Logon Count	The number of user login events on the workload within approximately the last 15 minutes. This is the count of the User Logon events regardless of whether there are matched rules. To know the details of the User Logon events, you must define rules to record the events for workloads of interests and view them on the Forensics Analysis page.
User Logon Failed Count	The number of users logon failed events on the workload within approximately the last 15 minutes. This is the count of the User Logon failed events regardless of whether there are matched rules. To know the details of the User Logon Failed events, you must define rules to record the events for workloads of interests and view them on the Forensics Analysis page.
Unseen Command Count	The number of unseen command events on the workload within approximately the last 15 minutes. This is the count of the Unseen Command events regardless of whether there are matched rules. To know the details of the Unseen Command events, you must define rules to record the events for workloads of interests and view them on the Forensics Analysis page.
Date Time (UTC) - Year	The year of the event time.
Date Time (UTC) - Month	The month of the event time (1, 2, . . .).
Date Time (UTC) - Day	The day of a month of the event time (1, 2, . . .).
Date Time (UTC) - Hour	The hour of a day of the event time (1, 2, . . . , 24).
Date Time (UTC) - Minute	The minute of an hour of the event time (1, 2, . . . , 60).
Date Time (UTC) - Second	The second of minute of the event time (1, 2, . . . , 60).
Date Time (UTC) - Day of Week	The day of a week of the event time (0-7, for Monday to Sunday).

Figure 12: Defining Forensic Rules for Network Anomaly Events



Below are some sample rules:

Listing 7.10.1.1.1: Detects network anomalies for UDP only.

```
Event Type = Network Anomaly AND Network Anomaly Is = Protocol - UDP
```

Listing 7.10.1.1.2: Detects large deviation after removing seasonal pattern (if detected), with a threshold on the egress app byte count for a subset of workloads whose names contain *sensitiveDataServer*.

```
Event Type = Network Anomaly AND Network Anomaly - Non-seasonal Deviation > 10.0)
AND Network Anomaly - Egress App Byte Count > 1000000
AND Network Anomaly - Host Name CONTAINS sensitiveDataServer
```

Listing 7.10.1.1.3: Detects Network Anomaly events on workloads with unseen command events except the Network Anomaly events happen from 7.30AM UTC to 7.35AM UTC everyday.

```
Event Type = Network Anomaly AND Network Anomaly - Unseen Command Count > 0
AND ( Network Anomaly - Date Time (UTC) - Hour != 7
OR Network Anomaly - Date Time (UTC) - Minute < 30 OR Network Anomaly - Date Time (UTC)
- Minute > 35 )
```

Rule Actions

Action	Description
RECORD	The matched events contribute to the Network Anomaly Score and can be found via the Security Dashboard or the Workload Profile Page / Network Anomaly Tab.
ALERT	The matched events shows up in the Alerts Page and the chosen Alert Publishers.

The next section describes in more detail where to find detected Network Anomaly events in the UI.

Where to See Network Anomaly Events

Note Network Anomaly events are *not* currently shown on Forensics Analysis page. You can find Network Anomaly events on the following pages.

- **Security Dashboard:** Network Anomaly events that match rules with **RECORD** action can be found in the Network Anomaly score section in the Security Dashboard. If there are workloads with nonbest (less than 100) scores, clicking on the workload name, you are able to view the PCR time series and the Network Anomaly events on that workload. On the right side of each row of the Network Anomaly event table, you can see action links that can help you search for flows and other forensic events around the time of the corresponding Network Anomaly event. See Network Anomaly Latency for known delay in Network Anomaly score reporting.

Figure 13: Network Anomaly Score in Security Dashboard

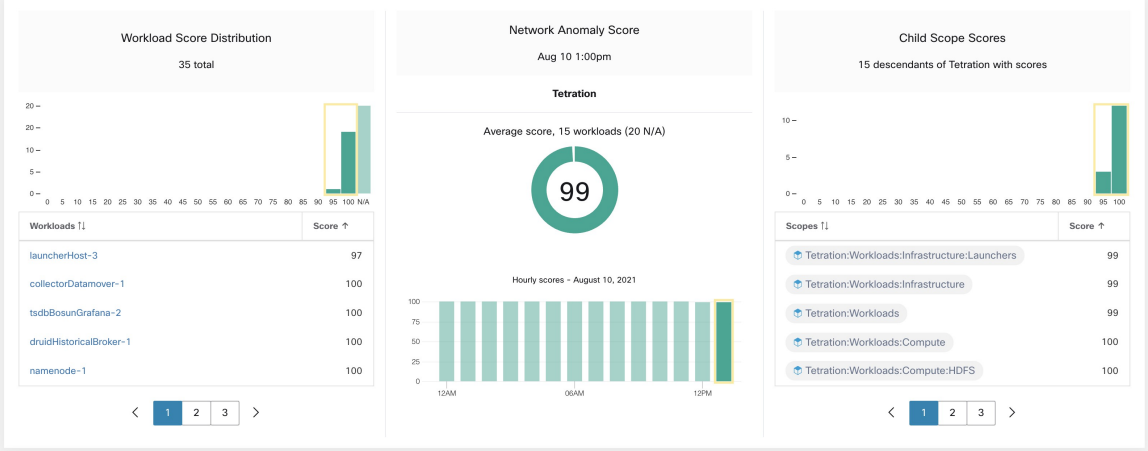
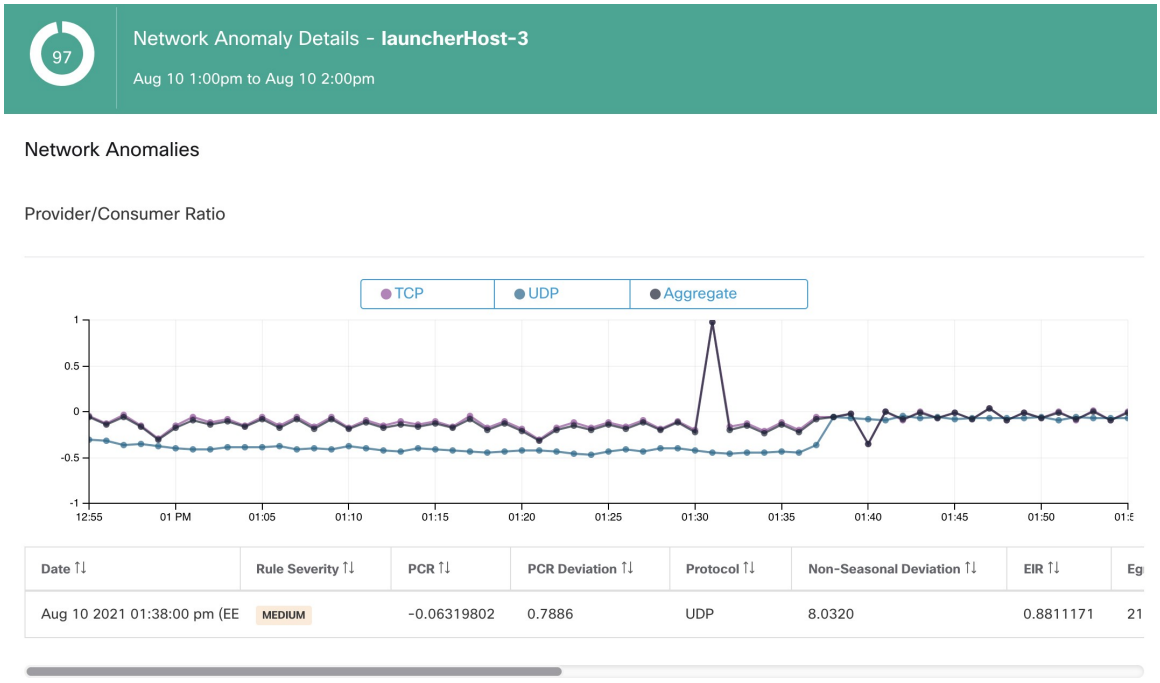
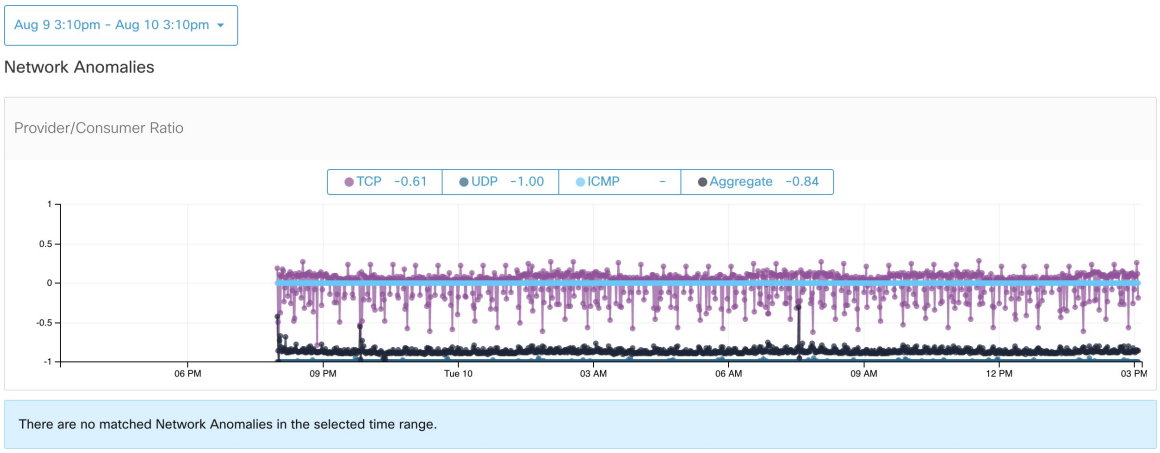


Figure 14: Network Anomaly score in Security Dashboard Drilled-Down by Workload



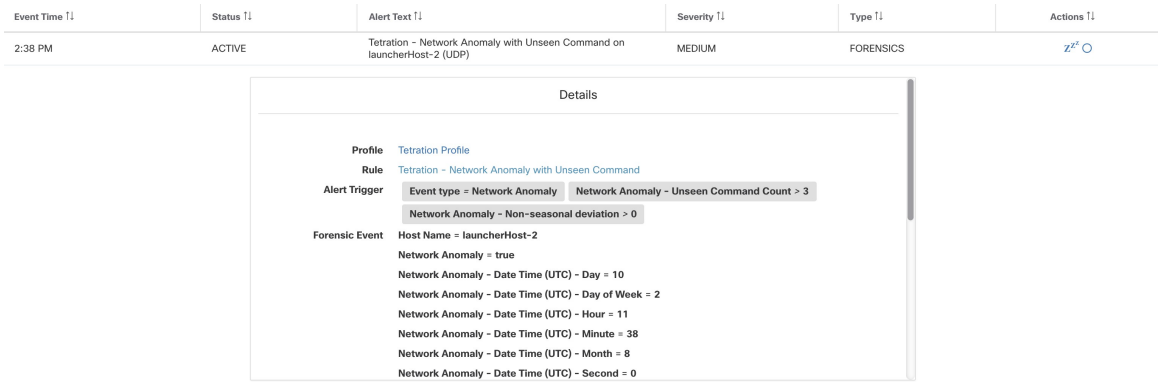
- Workload Profile Page / Network Anomaly Tab**:** on this page, you can see the PCR time series graph and the Network Anomaly events that match rules with **RECORD** action. What you can see that on this page is similar to what you find by clicking on the workload name in the security dashboard.

Figure 15: Network Anomaly Tab on Workload Profile Page



- **Alerts:** If the Network Anomaly rule is configured with **ALERT** action, the matched events is displayed on the Alerts Page and are also available via Alert Publisher.

Figure 16: Network Anomaly Alert



Rule Severities and Network Anomaly Scores

The Network Anomaly Score is computed similarly to the Forensics Score. For each Workload we compute a Network Anomaly Score. The Network Anomaly Score of a Workload is derived from the Network Anomaly Events observed on that Workload based on the profiles that are enabled for this scope. A score of 100 means no Network Anomaly Events were observed via configured rules in enabled profiles. A score of 0 means there is a Network Anomaly Event detected that requires immediate action.

- A Network Anomaly Event with the severity REQUIRES IMMEDIATE ACTION reduces the Score for the entire Scope to 0.
- A Network Anomaly Event with the severity CRITICAL reduces workload's score with the impact of 10.
- A Network Anomaly Event with the severity HIGH reduces workload's score with the impact of 5.
- A Network Anomaly Event with the severity MEDIUM reduces workload's score with the impact of 3.
- A Network Anomaly Event with the severity LOW doesn't contribute to the Network Anomaly Score. This is recommended for new rules where the quality of the signal is still being tuned and is likely to be noisy.

For each workload, the total impact score is aggregated every 5 minutes to compute the score of that workload within those 5 minutes.

For workloads without Network Anomaly enabled sensor types, the Network Anomaly scores are N/A.

PCR Data and Network Anomaly Events Retention

PCR data and Network Anomaly events are kept for 7 days.

Network Anomaly Latency

Network Anomaly scores reported in the security dashboard have 5-minute delays. For instance, the score of a workload for the hour 10:00 a.m-10:59 a.m is based on Network Anomaly events happen from 9:55 a.m to 10:54 a.m.

Caveats

- Old Data Leak events remain as Data Leak events instead of Network Anomaly events.
- Network Anomaly detection per protocol is a new feature in 3.3 and the protocol is not set in old Data Leak events.

Process Hash Anomaly Detection

As the name suggested, this feature detects process hash anomalies by assessing the consistency of process binary hashes across the system. The motivation of this feature is as follows. Imagine that you have a farm of Apache web servers that are cloned from the same setup configuration (for example, those servers are deployed from the same automation scripts). You can expect that the hashes of httpd binaries on all servers are the same. If there is a mismatch, it is an anomaly and might worth a further investigation.

Formally, we define a *process group* as the set of processes across workloads in the same rootscope that have the same combination of executable binary path, OS version, and package info (if applicable)1.

Note Package information is included since 3.4 release; in the previous releases, the process group is defined based on the combination of executable binary path and only the OS version.

In the above example, if all Apache web servers are running httpd 2.4.43 on CentOS 7.7 and in the same rootscope, then the corresponding process group is the set of processes (across all servers) that have the same combination: binary path of `/usr/sbin/ httpd` & OS version of `CentOS-7.7` & package version of `httpd-2.4.43`. It is expected that the hashes of all binaries in the same process group are the same, and an anomaly will appear if any mismatch is detected.

Besides detecting anomalous process hashes, this feature also detects process hashes that appear in a Flagged list File Hashes that are uploaded. The motivation is that you may have a list of known malware hashes, and want to know if a process associated with any of these hashes is run.

To reduce false alarms, we use the National Software Reference Library's Reference Data Set (RDS) provided by NIST, also called NIST RDS dataset as a Benign list; a benign hash is considered "safe" (see the Analyze Threat Intelligence Reports section on how to enable NIST RDS dataset). Also, you can see the File Hashes section to upload from your own hash Benign list.

In addition to the NIST RDS dataset, we also curate Secure Workload **Hash Verdict** service. When you enable the service, if any known malware hash shows up, it is detected as a malicious hash. However, if the hash is known and legit, then it is also marked as benign in the anomaly analysis. Due to the large dataset and fast updates that cover all known and legit process hashes that can be used to either approve or red flag processes running on a workload, Secure Workload **Hash Verdict** is only available through Secure Workload Cloud. To ensure that Secure Workload **Hash Verdict** service is accessible from your appliance, see Automatic Threat Intelligence Updates.

Output of this feature is a security score known as **process hash score**. This score is calculated and output hourly. Like all other security scores, a higher process hash score is better. In particular, for a process hash:

- Hash score of 0 means that the hash is flagged or malicious.
- Hash score of 100 means that the hash is either benign, or consistent across workloads (no mismatch)

- Hash score from 1-99 means that the hash is considered anomalous (that is, there is some mismatch)

The process hash score of a workload is the minimum process hash score of all hashes observed in that workload, with 0 meaning there is a flagged or malicious process hash in the system, and 100 meaning there is no hash anomaly observed in the system.

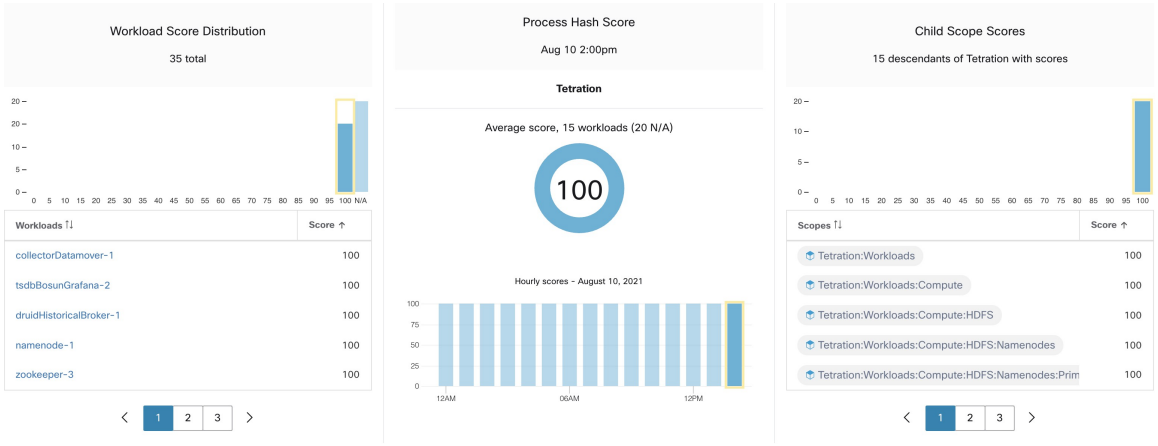
How to Enable Process Hash Feature

Process hash feature is enabled by default on deep visibility agents and enforcement agents; no forensic config is needed. If there are such agents in your system, you should begin to see scores within 2 hours after the system starts.

Where to See Process Hash Score

- **Security Dashboard:**

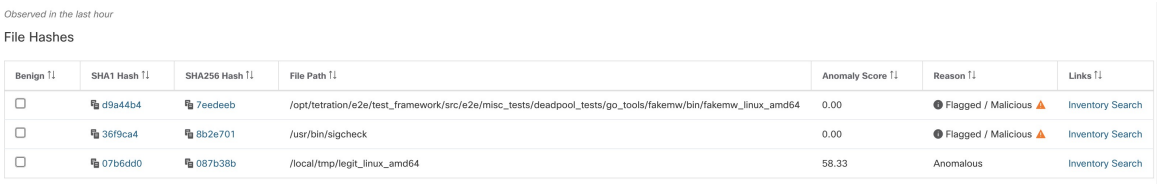
Figure 17: Process Hash Score Section in Security Dashboard



Process Hash Score section in Security Dashboard

- **Workload Profile Page** / File Hashes Tab:

Figure 18: File Hashes Tab on Workload Profile Page



File Hashes tab in Workload Profile page

How the Process Hash Score is Calculated

For each process hash, we compute a score as follows:

1. If the hash is flagged or malicious, `score = 0`
2. Else, if hash is benign, `score = 100`
3. Else, if hash is an anomaly, `score` is in the range of `[1, 99]`, the higher the better.
4. Else, `score = 100`

The logic for calculating score in (3) is that we first calculate the minority score of the hash (which is one minus the population ratio of that hash in workload population under the same rootscope), then map it to range `[0.0, 1.0]` using an information function `-log2(x)` if the minority score of the hash is above 0.5, then map the score again to a range `[1.0, 99.0]`. Let us take the above example of the Apache web server farm and consider the hash of `httpd`. Below are some scenarios:

- Suppose that `httpd` has two hash values (`h1` and `h2`) across 1000 servers in the farm: `h1` in 1 server, `h2` in the rest 999 servers. In this case:
  - `population_ratio(h1) = 0.001, population_ratio(h2) = 0.999`. Then:
  - `minority_score(h1) = 0.999, minority_score(h2) = 0.001`. Then:
  - `score(h1) = -log2(0.999) * 98 + 1 = 1.14;`
  - Since `minority_score(h2) < 0.5, h2` is not considered an anomaly, hence `score(h2) = 100`.
- Suppose that `httpd` has two hash values (`h1` and `h2`) across 10 servers in the farm: `h1` in 1 server, `h2` in the rest 9 servers. In this case:
  - `population_ratio(h1) = 0.1, population_ratio(h2) = 0.9`. Then:
  - `minority_score(h1) = 0.9, minority_score(h2) = 0.1`. Then:
  - `score(h1) = -log2(0.9) * 98 + 1 = 15.90;`
  - Since `minority_score(h2) < 0.5, h2` is not considered an anomaly, hence `score(h2) = 100`.
- Suppose that `httpd` has two hash values (`h1` and `h2`) across 2 servers in the farm: `h1` in one server, `h2` in the other. In this case:
  - `population_ratio(h1) = population_ratio(h2) = 0.5`. Then:
  - `minority_score(h1) = minority_score(h2) = 0.5`. Then:
  - `score(h1) = score(h2) = -log2(0.5) * 98 + 1 = 99.0`. This is the highest score for any hash that is considered an anomaly.
- Suppose that `httpd` has only one hash value (`h1`) across all servers. In this case, `minority_score(h1) = 0.0 < 0.5;` hence it is not considered an anomaly, and `score(h1) = 100`.

Finally, the process hash score of a workload is the minimum process hash score of all that hashes observed in that workload.

Additional information about the `-log2(x)` information function is found here.

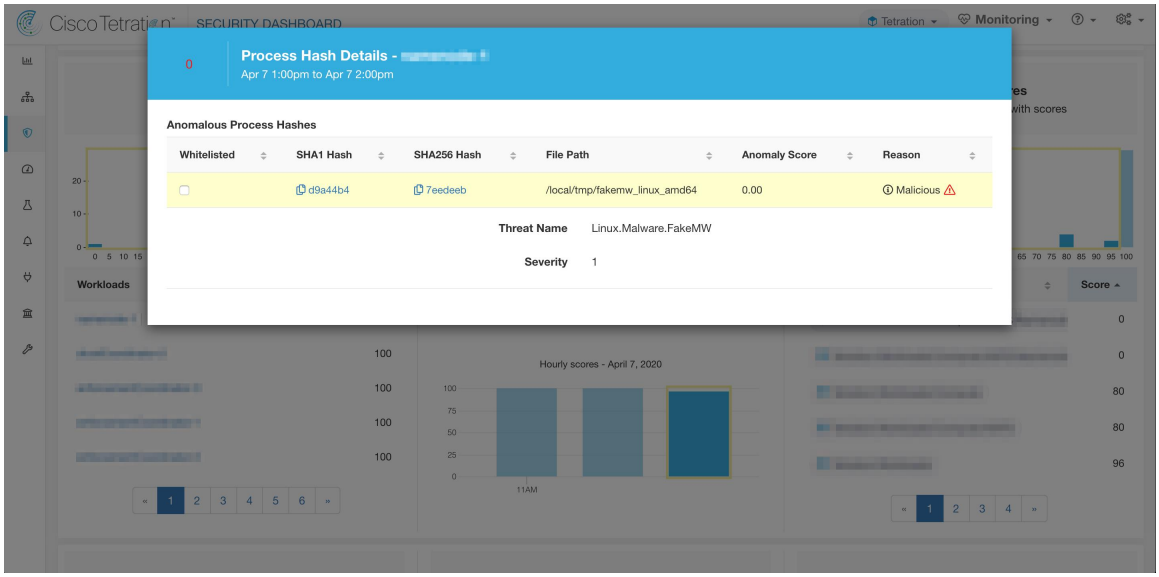
How to Improve Process Hash Score

The process hash score of 0 on a workload means that a flagged or malicious process hash has shown up in that workload; preventing such process to run again improves the score. A positive process hash score less than 100 means that there is a process hash anomaly across your system; it is not malicious but worth a further investigation. After a careful investigation, if the hash is concluded to be safe, adding it to your Benign list will also improve the score. User can mark anomalous hashes as 'benign' by clicking on the Benign checkbox in the File Hashes / Process Hash Details page or by uploading a Benign list via OpenAPI.

Threat Info Details

As mentioned earlier, if Secure Workload the Hash Verdict service is enabled, any known malware hash when showing up would be flagged as malicious. In that case, more threat information of the malicious hash (gathered via our threat intelligence platform) will be provided. Currently the additional threat data include *threat name* and *severity*. Threat name is the name of the threat, while severity is a value from 1-5 to indicate how severe the threat is, where 1 means the least and 5 means the most severe.

Figure 19: User Can Click on the Row of Malicious Hash to View Its Threat Info Details



Caveats

- Process hash analysis task is run once per hour, but it may take up to 2 hours for the expected scores/results to show in the security dashboard depending on the action. For examples:
  - If you upload your hash Flagged list and a process hash in that list shows up, it may take up to 1 hour for the score to be reflected in the security dashboard.
  - If you remove a hash from your Flagged list, it may take up to 2 hours for it to be cleared and the score is reflected in the security dashboard.
- Retention:
  - Detailed results from process hash analysis are kept for at least 7 days.

- File Hashes tab in Workload Profile page only shows process hash details analyzed in the last hour.
- Previous versions of deep visibility and enforcement agents, and AnyConnect endpoints only report SHA256 hash values. Thus, matching against SHA1 hash Flagged/Benign list is not supported for those agents.
- Process hash score is calculated regarding a particular rootscope. If a workload belongs to multiple rootscopes, the process hash score of that workload is the minimum score across all rootscopes that it belongs to.
- To further reduce the false alarms in process hash anomaly analysis, we also mark all Secure Workload agent binaries as benign according to their file paths. This mechanism happens only when these hashes do not appear in any user-defined hash list, or are not flagged by Secure Workload Hash Verdict service.