Technical Guide

Castra Custom Rules: Examples & Use Cases

Learn how Castra implements custom rules into our customers' environments that expands the reach and functionality of their SIEMs.

100+

Rules Created for exabeam

Custom rules allow analysts to detect potential malicious activities that may otherwise go unnoticed using default configurations.

Threat actors use hundreds of different attack techniques and sub-techniques to compromise their targets. Many of these techniques can be combined with one another, leading to a practically unlimited number of potential attack vectors.

SIEM solutions use a system of rules to identify and categorize malicious activities. Some of these rules work directly with UEBA Modeling, and some interpret log data directly to match a specific scenario. Exabeam provides a robust platform for both use cases, allowing Castra to create comprehensive custom rulesets based on real-world user and asset behavior.

Since these rules reflect real-world behavior, they must be configured to the unique characteristics of the organization they're designed to protect. That extra step is what allows these rules to catch suspicious activity that other tools can't detect. This enhances Exabeam's capabilities beyond what boilerplate default configurations are capable of.

Every organization is unique and needs a set of custom rules designed to address its security risk profile.

The following are just some examples of the 100+ rules Castra implements in customized SIEM.

Many contain parameters that must be adjusted to meet real-world usage needs unique to the organization. Others intake data from third-party services and infrastructure.

All of them help to build a stronger and more robust security posture for growing organizations.

Image File Rules

Image file execution keys modified in registry

- Alarm: Session
- This user changed Image File Execution registry keys using a debugger or MonitorProcess option.
- This may allow the user to intercept calls to an executable file. Malicious processes may be attached to applications and launched using Image File Execution Options injections.

Image file execution keys modified in registry

- Alarm: Endpoint
- Someone modified Image File Execution registry keys using a debugger or MonitorProcess option on this endpoint.
- This may allow the user to intercept calls to an executable file on this endpoint. Malicious processes may be attached to applications and launched using Image File Execution Options injections.

Image file execution keys modified in registry

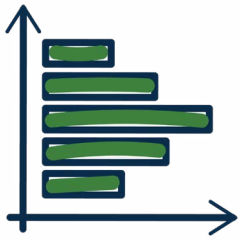
- Alarm: Asset
- Someone modified Image File Execution registry keys using a debugger or MonitorProcess option on this asset.
- This may allow the user to intercept calls to an executable file on this asset. Malicious processes may be attached to applications and launched using Image File Execution Options injections.

ClickOnce

ClickOnce application launch

- Alarm: Endpoint
- Someone used ClickOnce to rapidly deploy new applications with minimal user interaction on this endpoint.
- Utilities like ClickOnce may be signed with legitimate certificates allowing threat actors to execute malicious code while bypassing

application control solutions. This is an example of Trusted Developer Utilities Proxy Execution.



ClickOnce application launch

- Alarm: Endpoint
- Someone used ClickOnce to rapidly deploy new applications with minimal user interaction on this endpoint.
- Utilities like ClickOnce may be signed with legitimate certificates allowing threat actors to execute malicious code while bypassing application control solutions. This is an example of Trusted Developer Utilities Proxy Execution.

ClickOnce application launch on this asset

- Alarm: Asset
- Someone used ClickOnce to rapidly deploy new applications with minimal user interaction on this asset.
- Utilities like ClickOnce may be signed with legitimate certificates allowing threat actors to execute malicious code while bypassing application control solutions. This is an example of Trusted Developer Utilities Proxy Execution.

Logman

- Alarm: Endpoint
- This rule triggers when threat actors use logman to interrupt Event Tracing for Windows (ETW) on an endpoint.
- Threat actors may disable host-based sensors like ETW to prevent security teams from accurately analyzing event logs. This Indicator Blocking technique may be part of a larger, coordinated cyberattack.

- Alarm: Asset
- This rule triggers when someone invokes logman to interrupt ETW functionality on an asset.
- Threat actors may disable host-based sensors like ETW to prevent security teams from accurately analyzing event logs. This Indicator Blocking technique may be part of a larger, coordinated cyberattack.

PowerShell

PowerShell invoked for ETW interruption

- Alarm: Endpoint
- This rule activates when someone invokes a PowerShell command to interrupt ETW functionality.
- Threat actors may disable host-based sensors like ETW to prevent security teams from accurately analyzing event logs. This Indicator Blocking technique may be part of a larger, coordinated cyberattack.

PowerShell invoked for ETW interruption on this asset

- Alarm: Asset
- This rule activates when someone invokes a PowerShell command to interrupt ETW functionality on an asset.
- Threat actors may disable host-based sensors like ETW to prevent security teams from accurately analyzing event logs. This Indicator Blocking technique may be part of a larger, coordinated cyberattack.

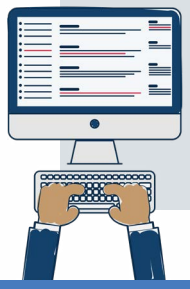
Encrypted argument detected in a PowerShell command

- Alarm: Endpoint
- This rule triggers when a user tries to include an encrypted argument in a PowerShell command.
- Threat actors may attempt to bypass detection by adding encrypted arguments in PowerShell commands. This File Obfuscation technique lets attackers use PowerShell to run malicious software and execute payloads undetected.

Encrypted argument detected in a PowerShell command

- Alarm: Endpoint
- This rule triggers when someone runs a PowerShell command that bypasses the PowerShell execution policy.
- PowerShell allows users to bypass execution policy when executing scripts from a file. Threat actors can use this Command and Scripting Interpreter technique to run malicious code without changing configurations or writing anything to disk.

A top risk for a SIEM or SOAR platform is missing that one event that helps with accurate detection.



Learn about Castra's approach to PowerShell Logging.

PowerShell Continued

Hidden argument detected in a PowerShell command

- Alarm: Endpoint
- This rule triggers when someone runs a PowerShell command that contains hidden arguments.
- Threat actors may execute PowerShell commands that contain hidden arguments. This Command and Scripting Interpreter technique allows attackers to run malicious code while bypassing detection.

Hide PowerShell History in PSReadLine

- Alarm: Asset
- This rule triggers when in response to attempts to bypass detection with malicious activity such as exfiltration software or payloads.
- PSReadLine is a PowerShell module that tracks command history across sessions. Threat actors can use this module to hide command history by deleting or editing the command history file. This is an example of Defense Evasion.

Invoke-Expression argument in a Powershell command detected

- Alarm: Endpoint
- This behavior may indicate an attempt to bypass detection with malicious activity such as exfiltration software or payloads.
- This is another example of a Command and Scripting Interpreter attack that relies on the invoke-expression cmdlet. It may be used for custom implants like CreepyDrive and CreepySnail.

NoProfile argument in a Powershell command detected

- Alarm: Endpoint
- This behavior may indicate an attempt to bypass detection with malicious activity such as exfiltration software or payloads.
- One Privilege Escalation and Persistence tactic sets PowerShell profiles to execute arbitrary scripts upon launching unless the “-NoProfile” flag is used. This custom rule triggers when that flag is detected in a PowerShell command.

NonInteractive argument in a Powershell command detected

- Alarm: Endpoint
- This behavior may indicate an attempt to bypass detection with malicious activity such as exfiltration software or payloads.
- The “-NonInteractive” flag indicates a session that shouldn’t require user input. This command has legitimate usage in continuous integration and deployment pipelines but is highly suspicious outside of that context.

Powershell Continued

Potentially malicious command in a Powershell command detected

- Alarm: Endpoint
- This rule triggers when a specific or highly suspicious Powershell command is detected.
- Castra maintains an up-to-date list of nefarious PowerShell commands and uses this rule to detect them in use.

Third Party Platforms

Castra supports a wide range of third-party security services. These include native security applications built into cloud platforms like Amazon AWS as well as cutting edge security products from Crowdstrike, SentinelOne and Palo Alto Networks, among others.

Default configurations do not scale the severity of rules according to third-party data. Usually, they simply assign a default risk score for all incoming alerts. Part of the value of customizing rules to include third-party platform data is the ability to triage alerts and assign unique risk values to individual rules.

These are some generic examples of custom rules Castra might implement on third-party platforms.

Third-Party Asset Alert

- Alarm: Asset
- A third-party security service triggered an alert associated with the asset.
- This rule would trigger whenever a third-party platform reports an alert on an asset. It includes the relevant information included in the original alert and assigns a dynamic risk score.

Third-Party User Alert

- Alarm: Session
- A third-party security service triggered an alert for a user.
- This rule would trigger whenever a third-party platform reports an alert for a user. Depending on the vendor product, it may report on an asset associated with a user, or on the session itself before assigning a dynamic risk score to it.

Third-Party Asset Alert logged on by VPN

- Alarm: Asset
- A third-party security service triggered an alert associated with a user who is logged on via VPN.

Third Party Platforms Continued

- If your cloud provider can trigger alerts for assets associated with users logged on via VPN, this custom rule allows analysts to easily identify and categorize these alerts separately. This rule may trigger a higher-severity alert than similar activity logged without a VPN.

XDR Security Alert

- Alarm: Asset
- An XDR security alert is associated with this asset.
- Your XDR solution has reported an alert on this asset. The details and severity of the alert will be included in the custom rule, along with a unique risk score.

XDR Security Alert for this User

- Alarm: Session
- An XDR security alert is associated with this user.
- Your XDR solution has reported an alert for this user. The details and severity of the alert will be included in the custom rule, along with a unique risk score.

Web Activity on a Malware Domain Detected

- Alarm: Web
- Third-party security services detected a connection to a suspected malware domain.
- This custom rule triggers when a third-party security service reports a connection to a compromised web domain. It would include all the relevant connection data included in the original alert, and a dynamic risk score calculated using that data.

Deploy Security Tools That Address Real Needs

Customization allows organizations to efficiently deploy limited resources to meet strategic objectives.

Custom rules streamline security processes and improve the outcomes of security events. As organizations implement increasingly complex IT tools and infrastructure, custom rules allow for cost-efficient coverage of key vulnerabilities in the network environment.

To learn more about 100 + customized rules that Castra offers our customers, schedule a meeting with one of our information security experts.

Schedule a meeting now.