
SOC Analyst
Event Analysis
THINKCYBER

Table of Contents

Basic Events

Event Analysis is a critical process for SOC analysts, involving the collection, examination, and interpretation of logs and alerts from various sources such as firewalls, endpoint devices, network infrastructure, and operating systems. It enables analysts to identify patterns, detect anomalies, and investigate suspicious activities that could indicate cyber threats. The ability to analyze events in real-time or through historical log reviews helps SOC analysts respond promptly to incidents, mitigate risks, and prevent security breaches. Event analysis is deeply integrated with **SIEM (Security Information and Event Management) systems**, which aggregate and correlate logs from multiple sources, providing centralized visibility and automated alerts. SIEM systems enhance the efficiency of event analysis by applying threat intelligence, filtering noise, and detecting complex attack patterns through correlation rules. This synergy between **event analysis** and SIEM allows SOC analysts to streamline incident detection, conduct root cause analysis, and generate actionable insights, strengthening an organization's overall security posture.

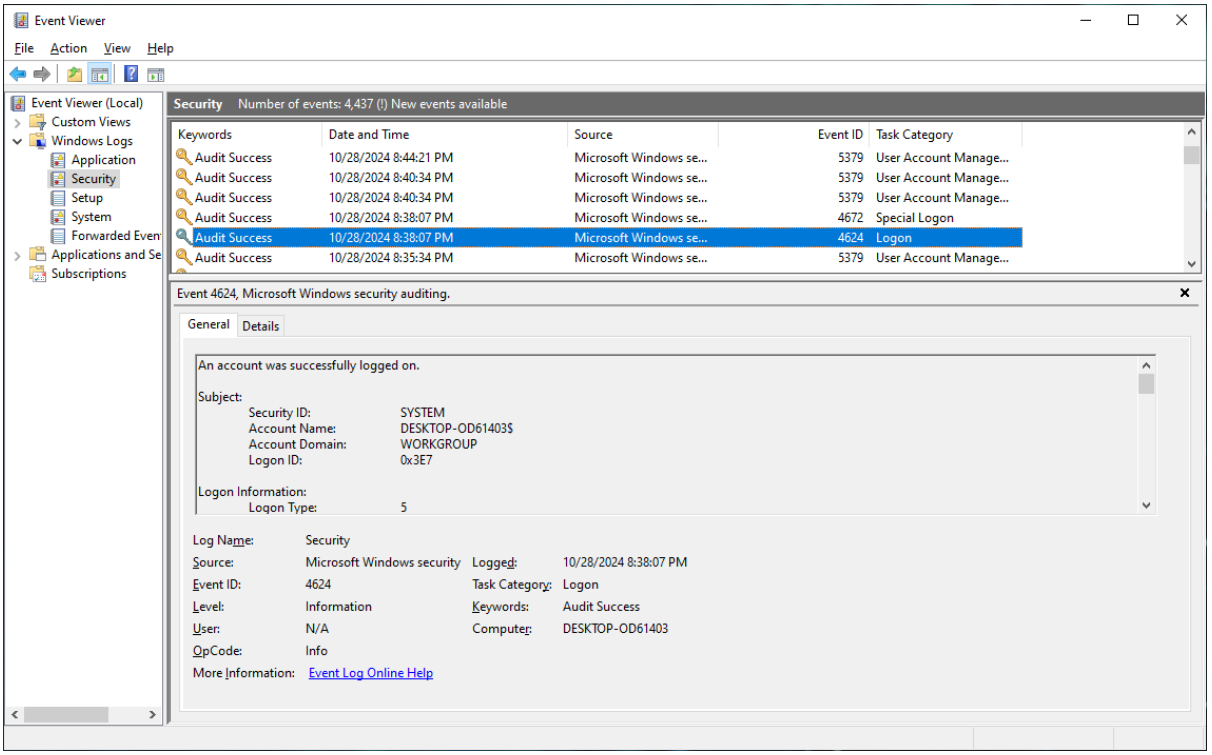
About the Windows Event Log

The Windows Event Viewer is a tool that helps monitor system events and activities. Logs are categorized under different sections (like **Security**, **Application**, and **System**) and can be used to investigate incidents, such as login attempts, service failures, or errors in the system.

Analyzing Event Logs

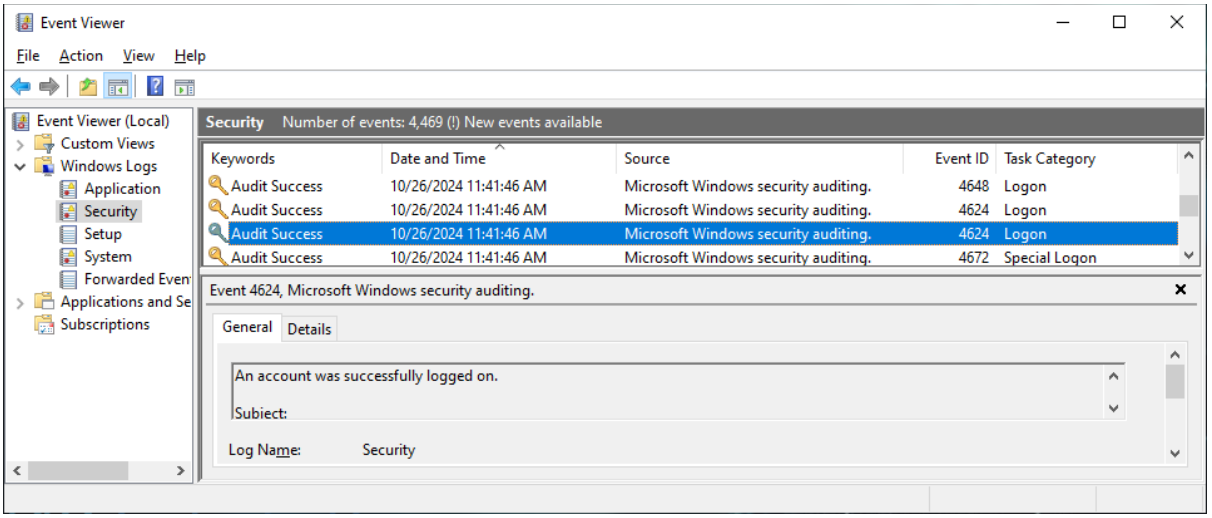
Security Events

Security events are a cornerstone of log analysis, providing visibility into authentication, access control, and privilege changes. These logs are crucial for detecting unauthorized activities, lateral movement, or privilege escalation attempts within a network.



Login Attempts (Success and Failure) – Event ID 4624, 4625

Successful logins generate **Event ID 4624**, which includes details such as the user account, login type, and IP address. **Event ID 4625** logs failed login attempts, often serving as an early indicator of brute-force attacks or unauthorized access attempts. By analyzing these events, SOC analysts can detect patterns such as repeated failed attempts or logins from unusual locations, helping to mitigate potential threats.




```
An account was successfully logged on.

Subject:
        Security ID:            SYSTEM
        Account Name:           DESKTOP-OD61403$
        Account Domain:         WORKGROUP
        Logon ID:               0x3E7

Logon Information:
        Logon Type:             2
        Restricted Admin Mode:  -
        Virtual Account:        No
        Elevated Token:         No

Impersonation Level:            Impersonation

New Logon:
        Security ID:            DESKTOP-OD61403\Malware
        Account Name:           Malware
        Account Domain:         DESKTOP-OD61403
        Logon ID:               0x83916
        Linked Logon ID:        0x838E1
        Network Account Name:   -
        Network Account Domain: -
        Logon GUID:             {00000000-0000-0000-0000-000000000000}

Process Information:
        Process ID:             0x6f4
        Process Name:           C:\Windows\System32\svchost.exe

Network Information:
        Workstation Name: DESKTOP-OD61403
        Source Network Address: 127.0.0.1
        Source Port:            0
```

```
Detailed Authentication Information:
        Logon Process:          User32
        Authentication Package: Negotiate
        Transited Services:  -
        Package Name (NTLM only):  -
        Key Length:             0

This event is generated when a logon session is created. It is generated on the computer that was accessed.

The subject fields indicate the account on the local system which requested the logon. This is most commonly a service such as the Server service, or a local process such as Winlogon.exe or Services.exe.

The logon type field indicates the kind of logon that occurred. The most common types are 2 (interactive) and 3 (network).

The New Logon fields indicate the account for whom the new logon was created, i.e. the account that was logged on.

The network fields indicate where a remote logon request originated. Workstation name is not always available and may be left blank in some cases.

The impersonation level field indicates the extent to which a process in the logon session can impersonate.

The authentication information fields provide detailed information about this specific logon request.
        - Logon GUID is a unique identifier that can be used to correlate this event with a KDC event.
        - Transited services indicate which intermediate services have participated in this logon request.
        - Package name indicates which sub-protocol was used among the NTLM protocols.
        - Key length indicates the length of the generated session key. This will be 0 if no session key was requested.
```

The following **Event ID 4624** represents a successful logon on a Windows system. This event is crucial in security monitoring, as it provides detailed information about the logon session, the process initiating the logon, and the source of the request. Understanding these details helps SOC analysts track access activities, detect anomalies, and correlate multiple events for potential security incidents.

The **Subject** section indicates the account or system process that requested the logon. In this example, the **Security ID** shows it is the SYSTEM account, meaning a system process initiated the login. The **Account Name** field reveals the name of the system where the event occurred, in this case, DESKTOP-OD61403, part of the WORKGROUP domain. **Logon ID** (0x3E7) is a unique identifier representing the session and can be used to correlate related events across logs.

The **Logon Information** section provides the **Logon Type**, which is set to 2, meaning it was an interactive logon (physically at the machine). **Impersonation Level** is listed as Impersonation, indicating the process can impersonate the user's security context to a limited degree.

The **New Logon** fields describe the account that successfully logged on, named **Malware** on the local system. This could potentially indicate a suspicious or test account if it is not part of normal operations. The **Logon GUID** is present but empty, often indicating that it was not tied to a Key Distribution Center (KDC) authentication.

Process Information reveals that the process initiating the logon is **svchost.exe**, a legitimate Windows process that hosts services. If this process behaves unusually, it could indicate compromise or misuse.

The **Network Information** section shows that the request originated from **localhost (127.0.0.1)**, which means it was initiated from the same machine. This could imply a local service login rather than a remote connection.

The **Detailed Authentication Information** section provides data on the **logon process (User32)** and the **authentication package (Negotiate)**, which handles both NTLM and Kerberos protocols.

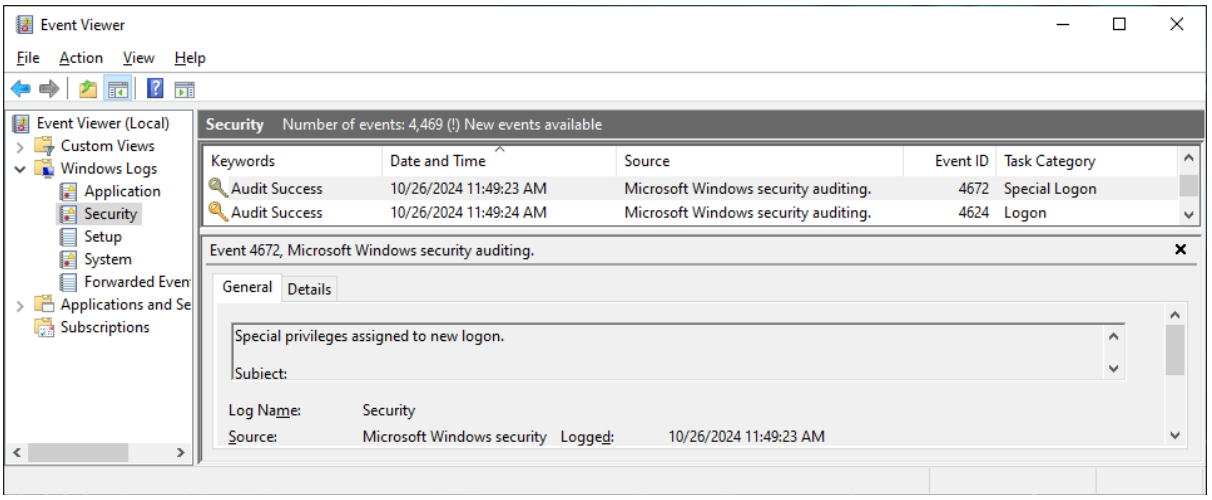
Below is a table summarizing the different logon types:

Logon Type	Description	Example
2	Interactive Logon	User logs in at the physical keyboard
3	Network Logon	User accesses a shared folder remotely
4	Batch Logon	Scheduled task runs automatically
5	Service Logon	Windows service starts with a specific user account
7	Unlock	User unlocks the workstation after being idle
8	NetworkCleartext Logon	Remote login where password is sent in plaintext
9	NewCredentials Logon	"RunAs" feature used to run a process with different credentials
10	Remote Interactive Logon	User logs in using Remote Desktop (RDP)
11	Cached Interactive Logon	User logs in with cached credentials (e.g., offline login)

This event log highlights important elements such as the process initiating the logon, the type of logon, and the user or service involved. It is crucial for analysts to monitor these events for suspicious behavior, such as logons occurring outside normal hours, from unexpected accounts, or initiated by processes that do not align with regular operations.

Privilege Use and Escalation – Event ID 4672, 4673

Monitoring privilege use is essential to identify unauthorized access to high-level functions. **Event ID 4672** records when special privileges, such as administrative rights, are used. **Event ID 4673** logs attempts to perform privileged operations, even if they are unsuccessful. These events are critical in detecting insider threats and privilege escalation attempts that may indicate an ongoing attack.



```
Special privileges assigned to new logon.

Subject:
        Security ID:            SYSTEM
        Account Name:           SYSTEM
        Account Domain:         NT AUTHORITY
        Logon ID:               0x3E7

Privileges:             SeAssignPrimaryTokenPrivilege
                        SeTcbPrivilege
                        SeSecurityPrivilege
                        SeTakeOwnershipPrivilege
                        SeLoadDriverPrivilege
                        SeBackupPrivilege
                        SeRestorePrivilege
                        SeDebugPrivilege
                        SeAuditPrivilege
                        SeSystemEnvironmentPrivilege
                        SeImpersonatePrivilege
                                        SeDelegateSessionUserImpersonatePrivilege
```

The **Event ID 4672** shown in your screenshot indicates that **special privileges have been assigned to a new logon**. This event is critical because it provides insight into privileged activities that could affect system security. When high-level privileges are granted, they allow the account to perform tasks beyond regular user capabilities, such as debugging processes, changing system settings, or loading drivers. Monitoring these events helps detect potential misuse, privilege escalation attempts, and insider threats.

Key Fields and Information in Event 4672

The **Subject** section identifies the account that was assigned special privileges.

- **Security ID**: SYSTEM
- **Account Name**: SYSTEM
- **Account Domain**: NT AUTHORITY
- **Logon ID**: 0x3E7

This tells us that the **SYSTEM account**, a highly privileged local account used by the operating system, initiated the session. The **Logon ID (0x3E7)** can be used to correlate this event with other related events, such as logon events (like Event ID 4624), to track the actions of this session.

Why Event ID 4672 is Important

This event is a **red flag** if it occurs unexpectedly, especially if assigned to non-administrative users or during off-hours. It signals that a logon session was granted special privileges, which could allow dangerous operations such as system-level changes, driver installations, or impersonation. Monitoring these events helps detect **privilege escalation attacks**, where an attacker gains higher access levels to perform unauthorized actions.

Use Case in Security Monitoring

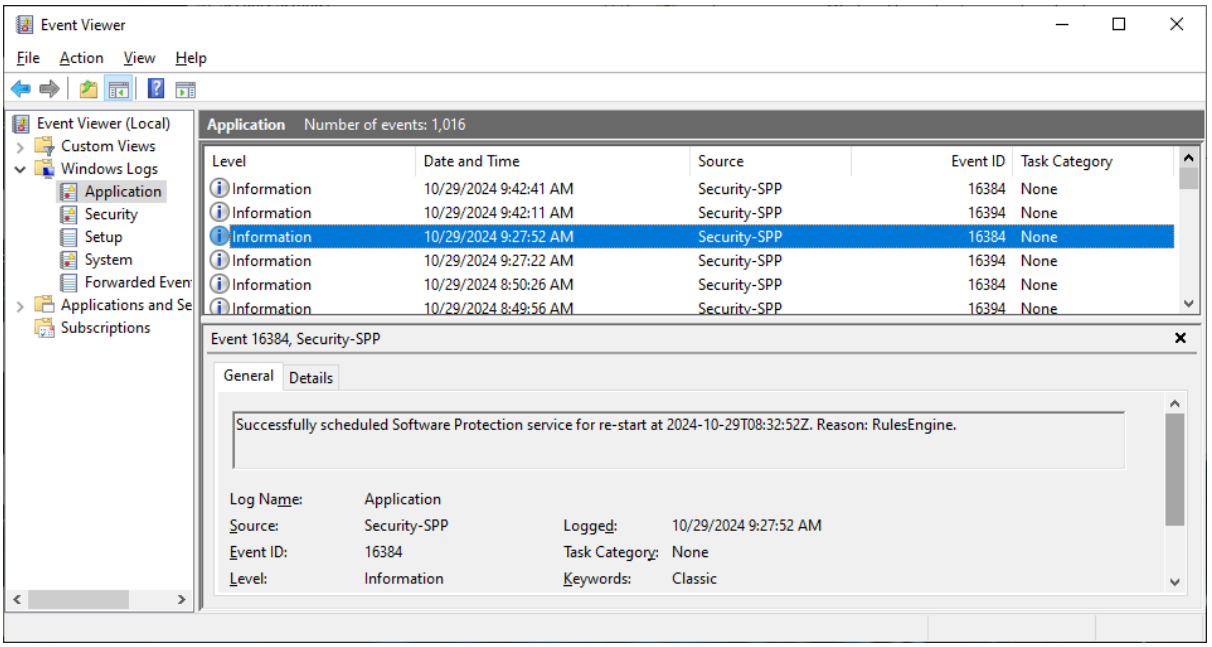
- **Correlate with Logon Events**: Use the **Logon ID (0x3E7)** to trace back to the originating **logon event (e.g., Event ID 4624)**. This provides context about when and how the logon session was initiated.
- **Detect Suspicious Activity**: If these privileges are assigned to **unexpected accounts** (like regular users), it could indicate an insider threat or compromised credentials.
- **Monitor for Privilege Escalation**: Privileges such as **SeDebugPrivilege** and **SeImpersonatePrivilege** are highly sought after by attackers trying to escalate their privileges and gain deeper control over the system.

Group Membership Changes – Event ID 4728, 4732

Changes to group memberships can grant or restrict access to sensitive data. **Event ID 4728** logs when a user is added to a group, while **Event ID 4732** records removals. Unauthorized group membership changes are often signs of an attacker elevating access privileges to perform malicious actions. Continuous monitoring of these events helps prevent unauthorized access and ensures the integrity of user permissions.

Application Events

Application logs capture events related to software performance and functionality, providing insights into potential issues with critical applications such as databases, antivirus solutions, or other enterprise tools.



Software Crash Analysis

Application crashes can result from software bugs, compatibility issues, or resource constraints. Analyzing these logs helps SOC analysts identify the root cause and prevent future occurrences. Patterns in crash logs, such as crashes triggered by specific user actions or processes, may also indicate exploitation attempts through application vulnerabilities.

Database Connection Errors

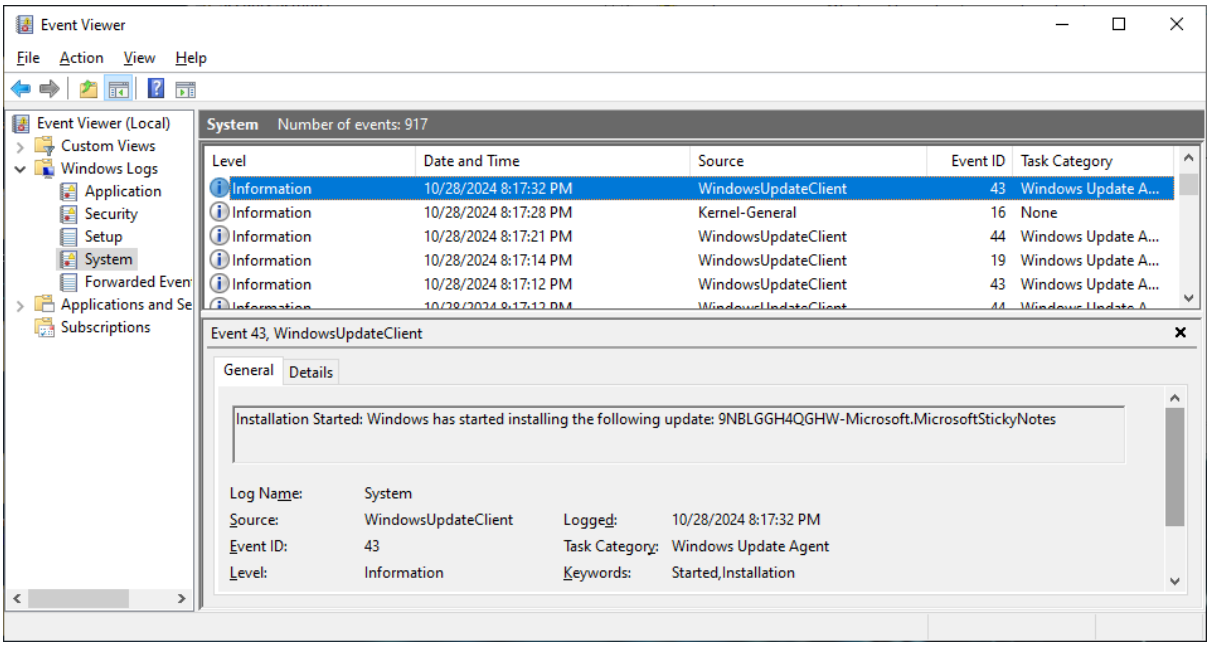
Logs that record database connection failures can point to issues such as configuration errors, network connectivity problems, or attempts to access the database using incorrect credentials. In some cases, repeated connection failures may indicate a brute-force attack targeting the database. Monitoring these logs helps maintain data availability and detect unauthorized access attempts.

Monitoring Antivirus Updates

Antivirus tools generate logs during updates, scans, and detections. Analyzing these logs helps identify whether antivirus solutions are functioning correctly and up to date. Gaps in antivirus updates can leave systems exposed to new threats, and SOC analysts need to monitor for such events to ensure continuous protection.

System Events

System logs capture events at the operating system level, including hardware malfunctions, OS crashes, and service failures. These logs are essential for identifying potential system weaknesses and maintaining stability.



Device Driver Errors

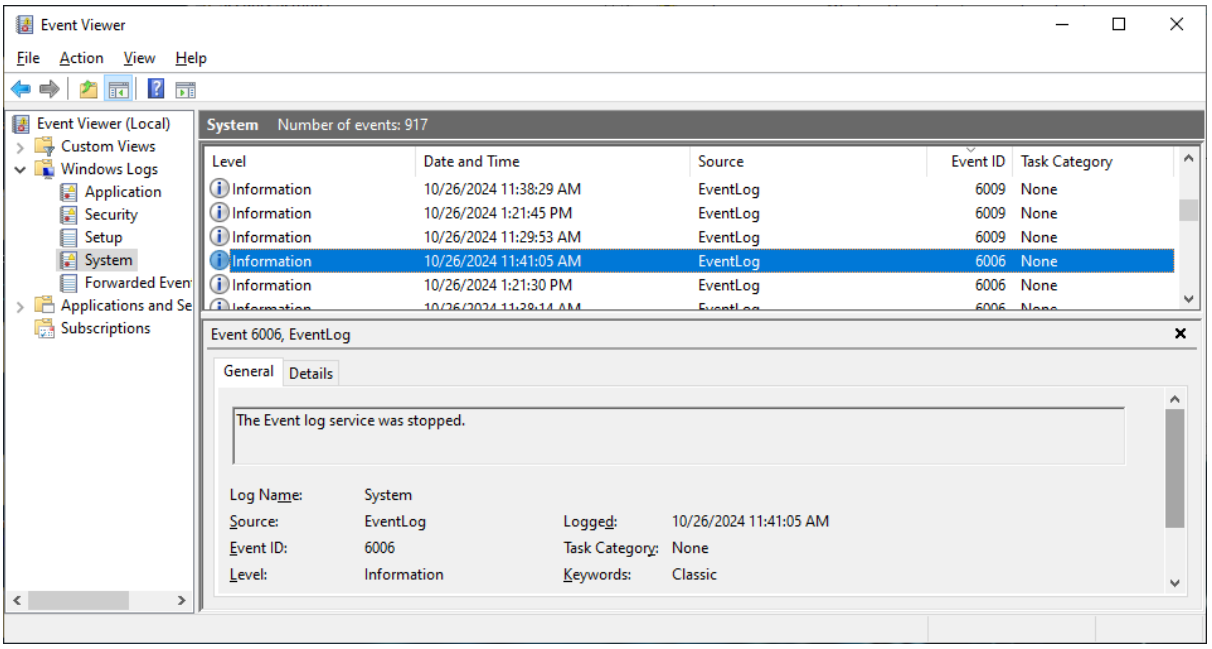
Device drivers facilitate communication between the operating system and hardware components. Errors in these drivers can cause system instability, degraded performance, or service interruptions. Monitoring these logs helps detect failing hardware or driver incompatibilities that require remediation to maintain operational efficiency.

Kernel Crashes or Unexpected Reboots – Event ID 6008

Kernel crashes, also called **blue screen errors** in Windows, indicate critical failures in the operating system. **Event ID 6008** logs unexpected reboots, providing information about crashes or power failures. SOC analysts can use these logs to investigate the cause of system instability and identify whether a crash was due to hardware failure or malicious activity.

Service Failures

Services running on a system provide core functionality, and their failures can disrupt business operations. Logs related to service failures help SOC analysts identify dependencies, restart critical services, and investigate the root causes of disruptions. Recurring service failures may point to deeper issues such as resource exhaustion, malware infections, or configuration errors.



Searching and Filtering Logs

Effectively searching and filtering logs is a crucial skill for SOC analysts and administrators. Large volumes of logs can make it challenging to identify critical events without the proper techniques. In this chapter, you will learn how to efficiently search for relevant events using **PowerShell commands** and the **Event Viewer**.

Techniques to Find Critical Events Amidst Large Logs

Logs contain valuable data, but without efficient search techniques, important events may go unnoticed. The key to successful log analysis lies in **identifying patterns and focusing on specific event types** such as logon attempts, failed authentication, and policy changes. Searching by **event ID** is particularly effective since each event type is associated with a unique identifier. For example, you can search for **Event ID 4625** to find failed logon attempts or **Event ID 4672** to locate privilege use. Combining filters, such as **date ranges, event sources, and keywords**, allows you to narrow the scope and pinpoint relevant data quickly.

Another useful technique is **log correlation**, where you identify patterns across multiple events. For example, detecting a failed logon event (4625) followed shortly by a successful one (4624) from the same IP address may indicate a brute-force attempt. Filtering logs effectively reduces the noise and ensures that analysts focus on high-priority events.

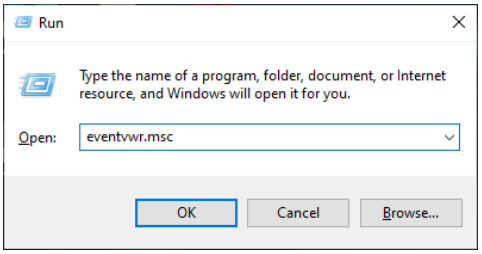
Using the Event Viewer to Filter Logs

The **Event Viewer** is an essential tool in Windows for monitoring and analyzing system and security events. It provides a graphical interface that allows administrators and SOC analysts to filter large volumes of logs and focus on specific events of interest, such as failed login attempts or policy changes. Filtering logs helps to reduce noise and ensures that critical activities are not missed during investigations.

Opening the Event Viewer

To begin, open the Event Viewer:

1. Press **Windows Key + R** to open the **Run** dialog box.



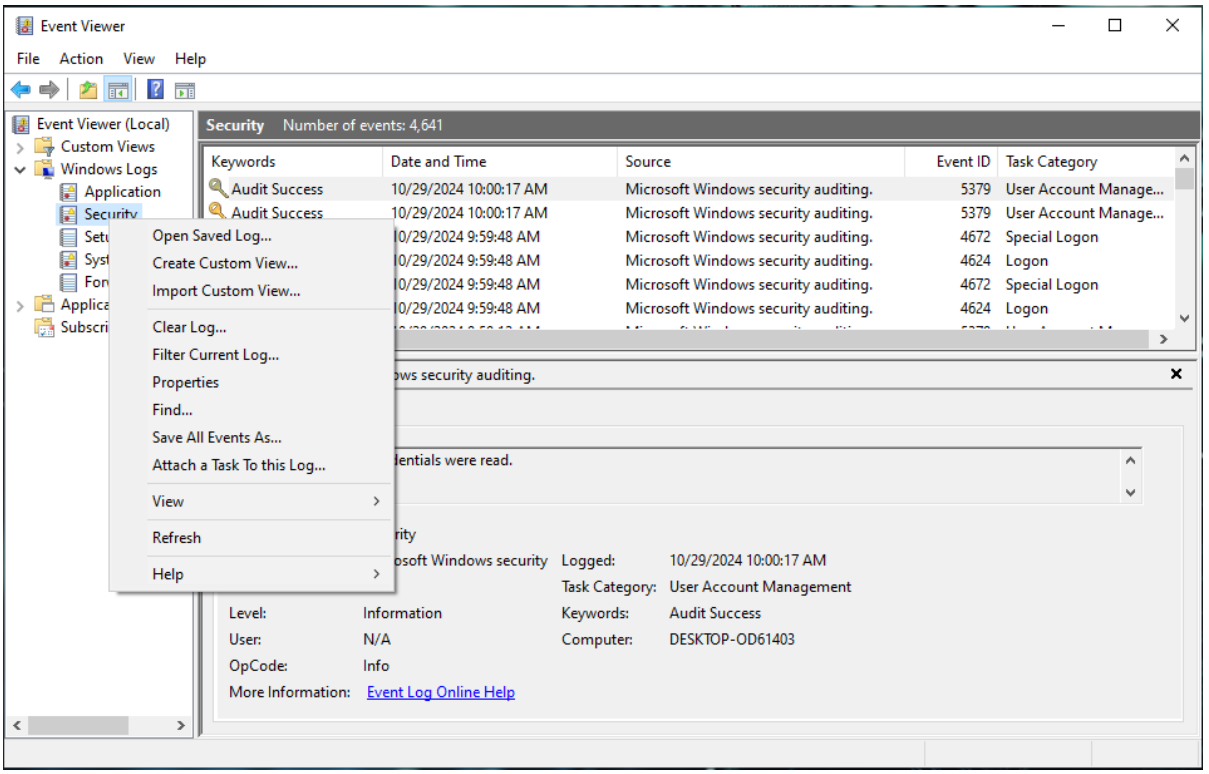
2. Type **eventvwr.msc** and press **Enter**.
3. The Event Viewer window will open, displaying various log categories under **Windows Logs**: **Application, Security, Setup, System, and Forwarded Events**.

Each log type serves a different purpose. In this chapter, we will primarily focus on the **Security log**, which records authentication events, access attempts, and privilege use.

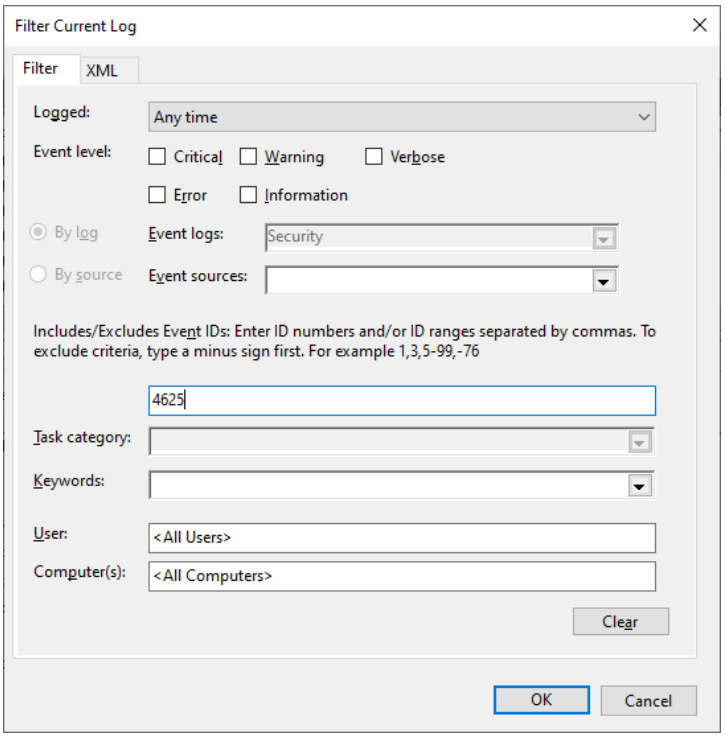
Filtering Logs by Event ID

Filtering logs by **Event ID** allows you to quickly locate specific types of events. For example, **Event ID 4624** represents successful logins, while **Event ID 4625** indicates failed login attempts.

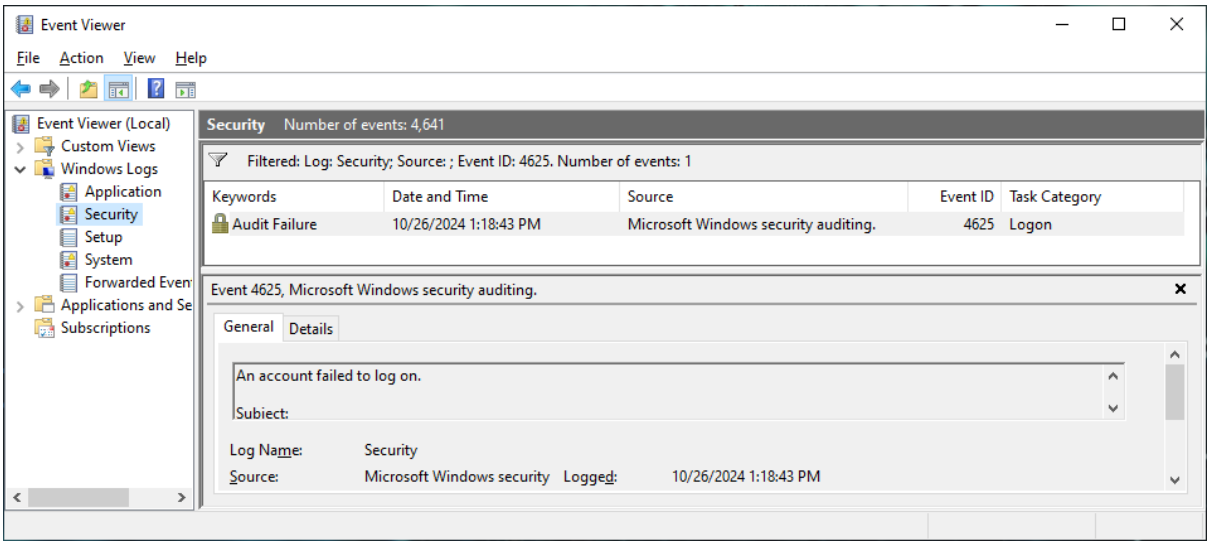
1. In the Event Viewer, navigate to **Windows Logs** and click on **Security**.
2. Right-click on the **Security log** and select **Filter Current Log**.



3. In the **Event IDs** field, enter 4625 to search for all failed login attempts.



4. Click **OK** to apply the filter.

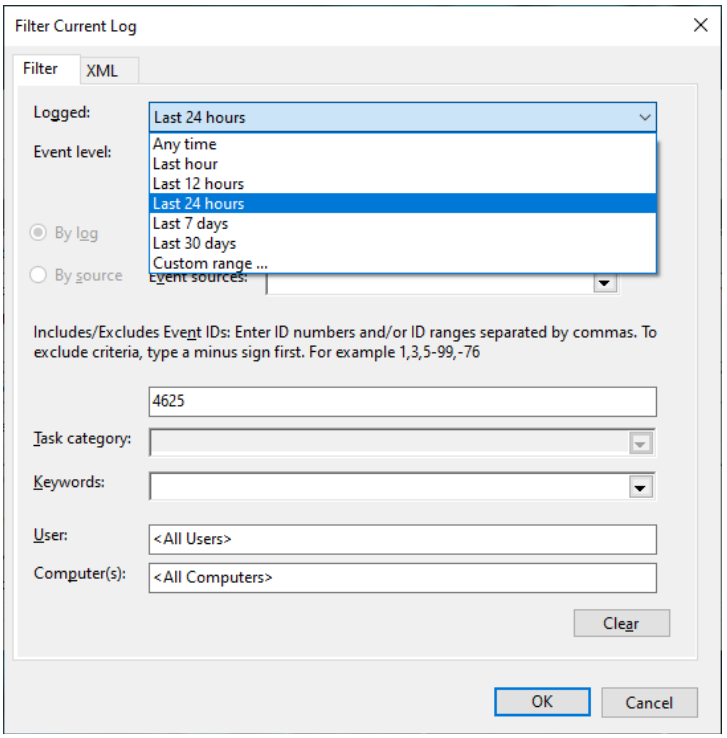


The Security log will now display only **failed login attempts**. Review the filtered events to investigate suspicious behavior, such as repeated login failures from the same account or IP address.

Filtering Logs by Date and Time

If you need to investigate events that occurred within a specific time frame, filtering by **date and time** will help narrow your search.

1. Right-click on the **Security log** and select **Filter Current Log**.
2. In the **Logged** dropdown, select a predefined time range, such as **Last 24 hours**. Alternatively, choose **Custom range** to specify an exact start and end date.



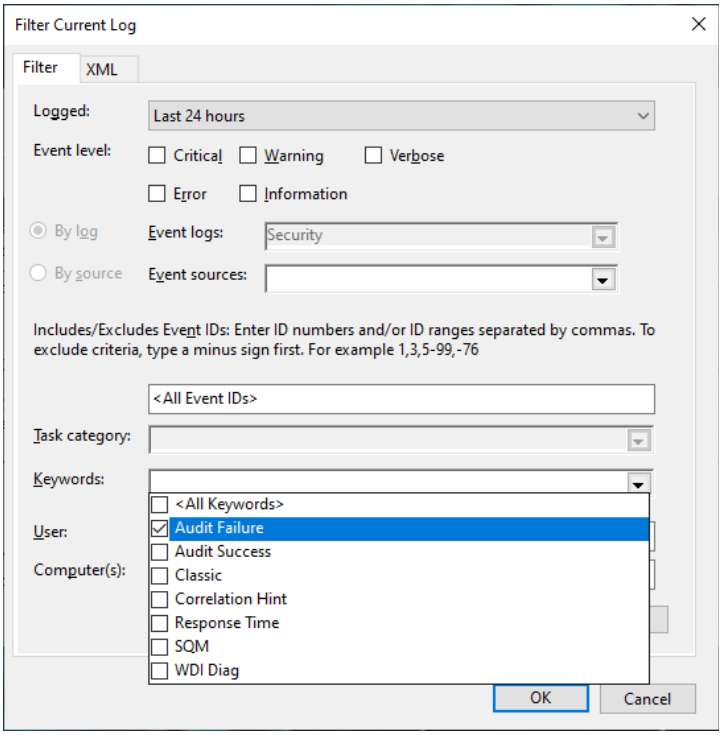
3. Click **OK** to apply the filter.

This filter will show only the events that occurred within the selected time range, helping you focus on recent activities or incidents that happened during a particular window.

Filtering Logs by Keywords

The **Keyword** field allows you to filter for specific terms within event messages. This is useful if you are looking for events related to particular activities, such as remote desktop logins or administrative actions.

1. Right-click on the **Security log** and select **Filter Current Log**.
2. In the **Keywords** field, choose the relevant selection. For example, type **"Audit Failure"** to search for remote desktop logins.



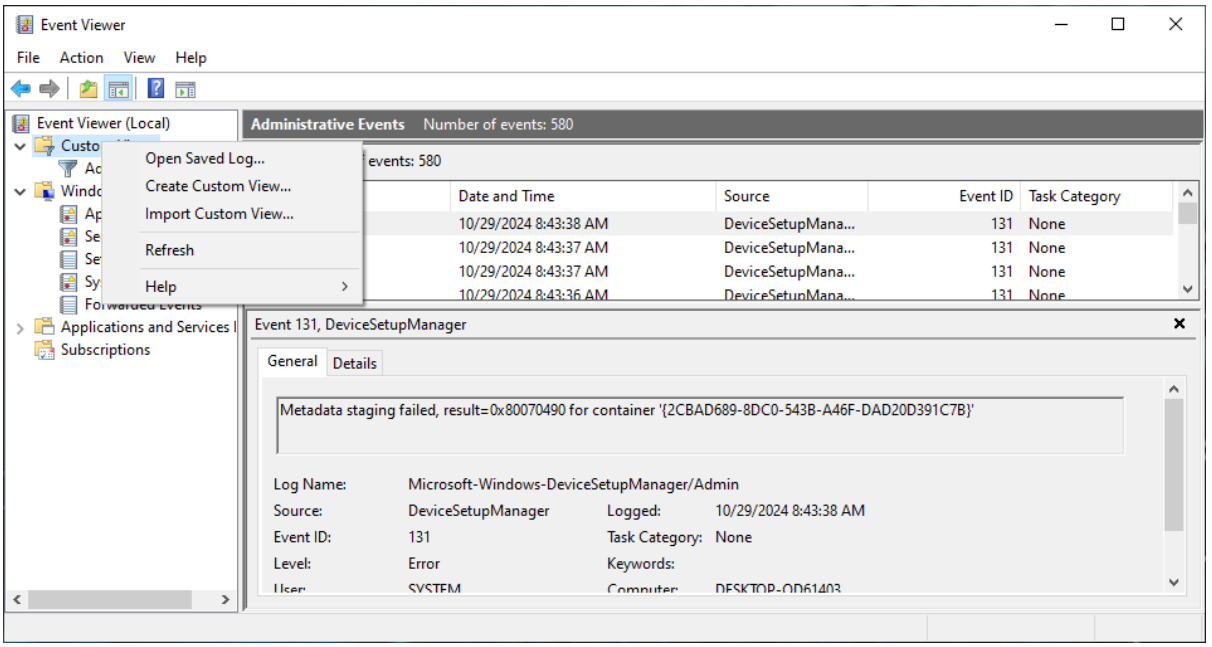
3. Click **OK** to apply the filter.

The Event Viewer will now display only events containing the keyword **Audit Failure** in their messages, helping you investigate failure activities.

Creating a Custom View

Creating a **custom view** allows you to save a filter configuration for future use. This is particularly useful if you need to monitor the same type of events regularly.

1. In the Event Viewer, click on **Create Custom View** from the left-hand panel.



2. In the **Custom View Wizard**, select the log types you want to monitor, such as **Security**.
3. Enter the **Event IDs** you wish to filter. For example, type 4624, 4625 to include both successful and failed logins.
4. Set the **time range** to focus on recent events, or leave it open to include all logs.
5. Click **OK** and give your custom view a name, such as **Login Events**.
6. Click **Save** to create the custom view.

You can now access this custom view anytime to monitor login events without having to recreate the filter.

Exporting and Importing Filters

If you need to share filters with other analysts or apply the same filter on multiple machines, you can **export and import custom views**.

To export a custom view:

1. Right-click on the custom view and select **Export Custom View**.
2. Save the view as an **XML file**.

To import a custom view:

1. Click on **Import Custom View** from the right-hand panel.
2. Select the XML file containing the custom view and click **Open**.

This functionality makes it easy to standardize log monitoring across different systems and teams.

Saving and Exporting Filtered Logs

After applying a filter, you may want to **save the filtered logs** for further analysis or reporting.

1. Click on **Save Filter to Custom View** from the right-hand panel to save your filter configuration.
2. Alternatively, select the filtered events, right-click, and choose **Save All Events As** to export the logs.
3. Save the logs as an **EVTX** file for use with the Event Viewer or as a **CSV** file for analysis in Excel.

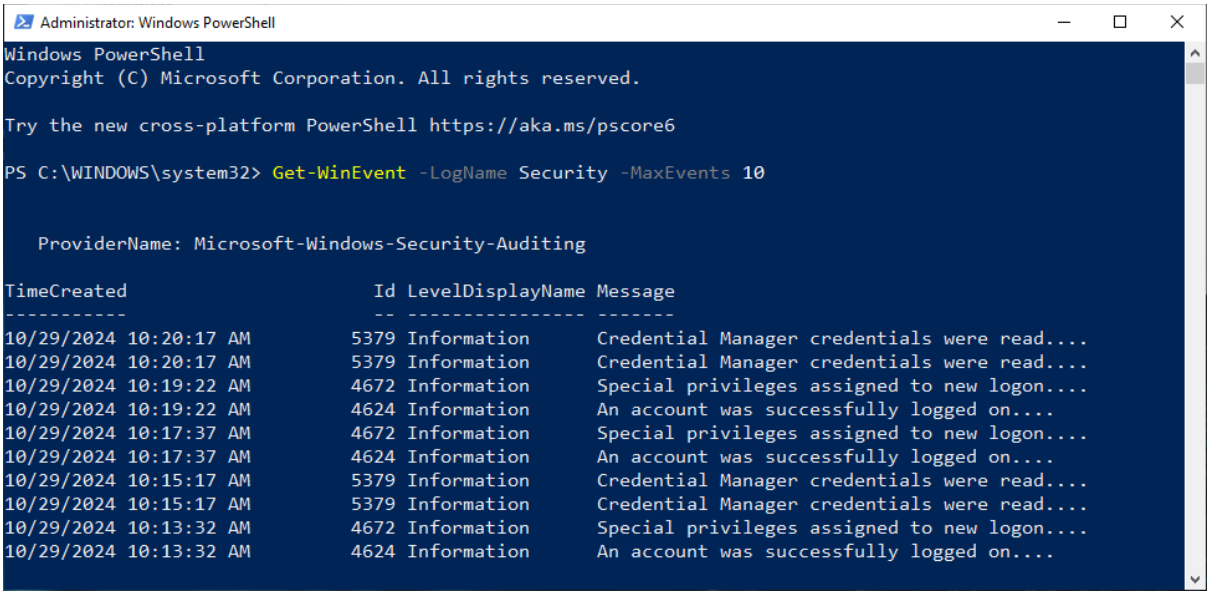
Using PowerShell to Search and Filter Logs

PowerShell provides a powerful way to search through logs programmatically, offering flexibility that goes beyond the Event Viewer. The **Get-WinEvent** command is particularly useful for filtering logs by **event ID, source, date, or keywords**. In this section, you will learn how to use PowerShell to query specific events efficiently.

To list the most recent security logs, use the following command:

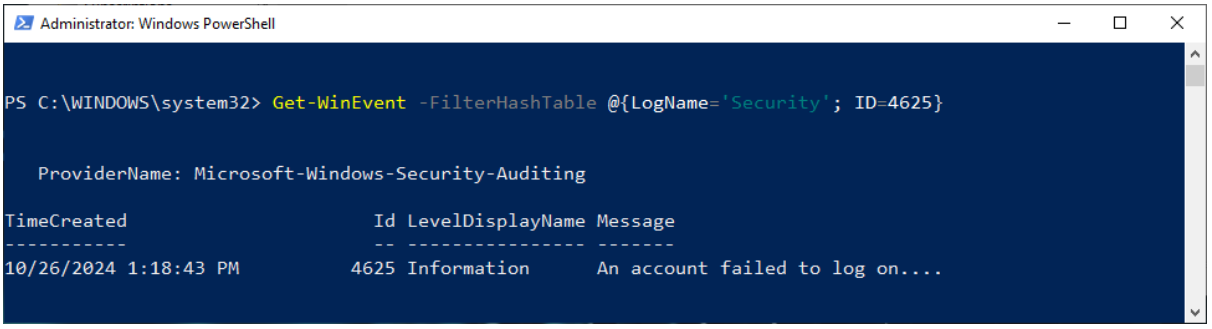
```
Get-WinEvent -LogName Security -MaxEvents 10
```

This command retrieves the latest ten events from the **Security** log, helping you quickly review recent activities.



If you want to search for all failed logon attempts (Event ID 4625), use:

```
Get-WinEvent -FilterHashTable @{LogName='Security'; ID=4625}
```

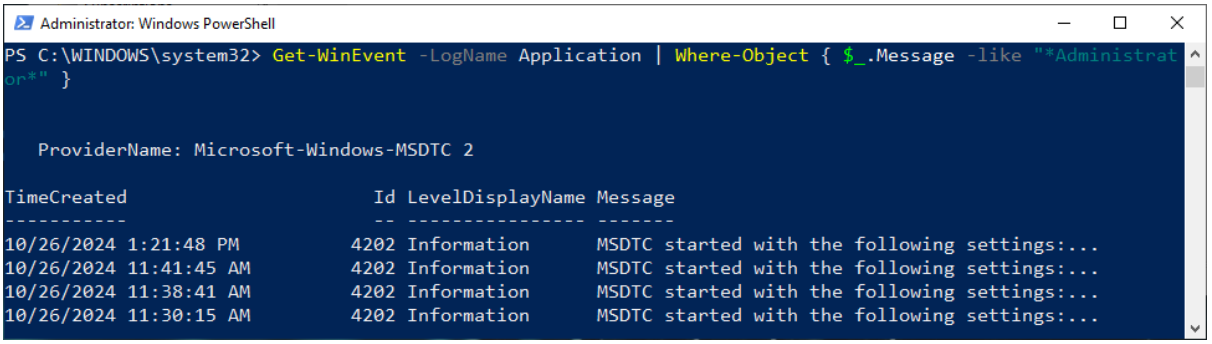


This filter retrieves only events with the **ID 4625**, showing failed login attempts.

To search for events containing specific **keywords**, such as “Administrator,” within the **Application** log:

```
Get-WinEvent -LogName Application | Where-Object { $_.Message -like "*Administrator*" }
```

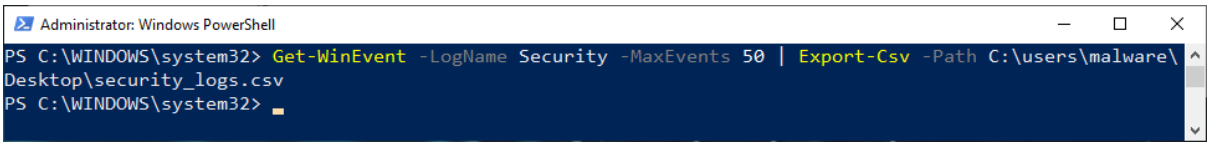
This command returns events that contain the word “Administrator” in their message, allowing you to locate specific activities or errors involving that account.



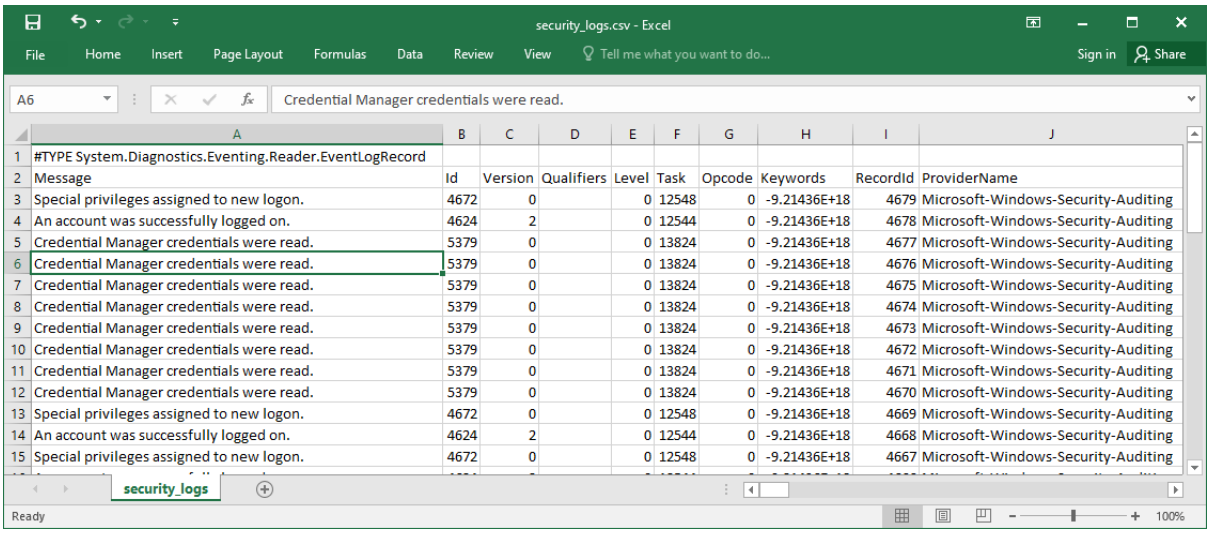
Save the Output

Saving the output to a **CSV file** allows for further analysis in tools like Excel:

```
Get-WinEvent -LogName Security -MaxEvents 50 | Export-Csv -Path
C:\users\malware\Desktop\security_logs.csv
```

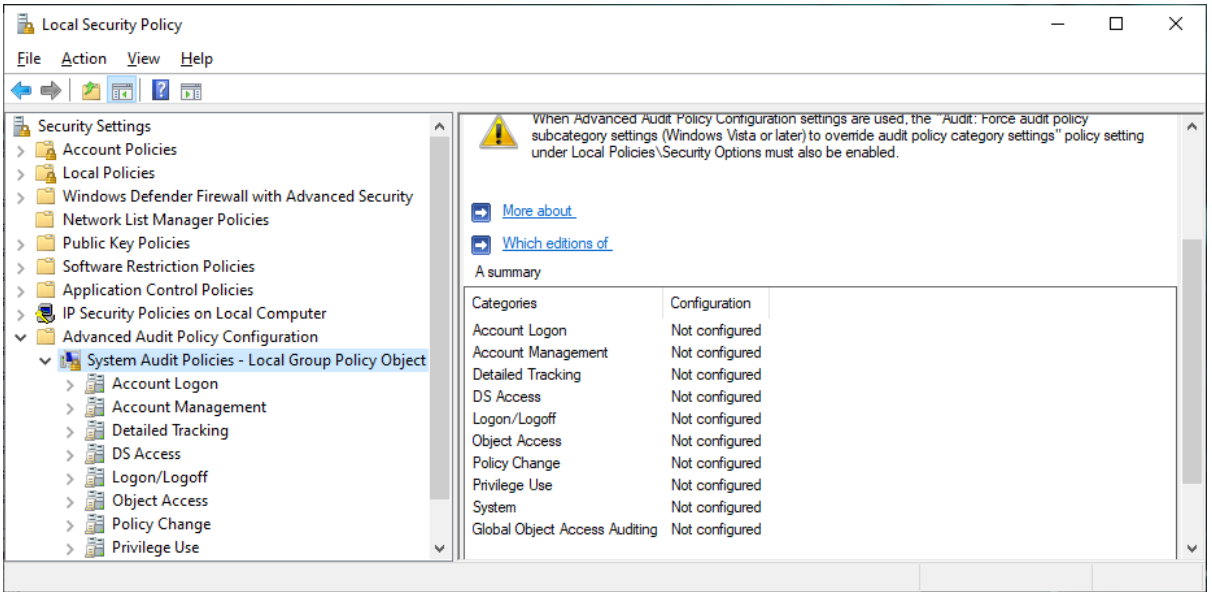


This command exports the latest 50 security events to a CSV file for external review.



Local Security Policy

The **Local Security Policy (secpol.msc)** is a management console in Windows that allows administrators to configure and manage **security policies** on a local machine or within a domain. This tool provides more granular control over settings related to **account policies, audit policies, and security configurations**. While **Event Viewer** captures logs related to system and security events, it is important to understand that the **Event Viewer reflects the outcomes of the configurations** made within the Local Security Policy. In other words, the Event Viewer is used to **monitor and view the effects** of policy settings, but not to modify or enforce them.



How Local Security Policy Works

The Local Security Policy tool allows administrators to configure the following:

- **Account Policies**: Includes password policies, lockout policies, and Kerberos settings to control user authentication behavior.
- **Local Policies**: Enforces rules regarding user rights assignments, security options, and audit policies for logging activity.
- **Audit Policies**: Controls which actions and events are logged by the system. These policies are essential for ensuring that key security events, such as failed logins or privilege escalations, are recorded.

The **Advanced Audit Policy Configuration** section provides **detailed control over what gets logged**. Administrators can specify individual categories and subcategories to fine-tune what events are tracked. For example, you can enable audits for:

- **Account Logon** events to track user login activity.
- **Policy Change** events to monitor when changes are made to critical security settings.
- **Privilege Use** to detect when users attempt to use elevated privileges.

How Event Viewer Relates to Local Security Policy

The **Event Viewer** is a monitoring tool that **displays the results of the policies configured through secpol.msc**. When an audit policy is enabled within the Local Security Policy, events generated based on those rules are captured and displayed in the Event Viewer. For example:

- If **Audit Logon Events** is enabled, every login attempt (successful or failed) will appear in the **Security** log of the Event Viewer.
- If **Audit Policy Changes** is active, any modifications to the security settings will generate a log entry.

The **Event Viewer cannot modify these policies**; it is purely a **log viewing tool**. All configuration and policy management must be performed through **secpol.msc** or Group Policy in a domain environment. The Event Viewer acts as a **window into the logs** that result from applying these policies, allowing SOC analysts and administrators to monitor compliance, detect incidents, and investigate suspicious activities.

Using Advanced Audit Policies to Improve Monitoring

Advanced Audit Policy Configuration allows organizations to go beyond basic auditing by breaking policies into **more specific subcategories**. This enables better visibility and reduces noise in the logs. For example, instead of enabling a broad "Logon/Logoff" category, administrators can selectively enable auditing for **Remote Desktop logins** or **failed logon attempts**. These detailed logs are then displayed in the Event Viewer, helping analysts focus on critical events without getting overwhelmed by unnecessary information.

By enabling **appropriate audit policies** and monitoring the resulting logs through the Event Viewer, organizations can maintain a strong security posture, ensure compliance, and detect unauthorized activities promptly.

Local Policies vs. Advanced Audit Policies

Windows provides two key mechanisms to manage security and audit settings: **Local Policies** and **Advanced Audit Policies**. While both frameworks allow administrators to enforce security rules, they differ in **scope, granularity, and control** over what gets logged and how security events are managed.

Local Policies

Local Policies, configured through the **Local Security Policy (secpol.msc)** or Group Policy, offer **high-level security settings** to manage aspects of a computer's behavior. These settings control user rights, security options, and general audit policies.

Local Policies include three primary categories:

- **Audit Policies**: Enable or disable event logging for critical activities such as logon attempts, account management changes, and policy modifications. These policies broadly categorize events, meaning they **track a wide range of activities** without much specificity.
- **User Rights Assignment**: Controls what actions users or groups can perform on a system, such as logging on locally, shutting down the system, or accessing the computer over the network.

- **Security Options**: Includes settings related to password policies, account lockout policies, and other system security behaviors.

Example: Enabling the **Audit Logon Events** policy in Local Policies will log every logon and logoff event in the Event Viewer, but it doesn't allow for deeper filtering, such as distinguishing between remote and interactive logons.

Advanced Audit Policies

Advanced Audit Policies provide **fine-grained control** over what events are logged. Introduced in Windows Vista and later, these policies allow administrators to **select specific activities** to audit within broader categories, reducing unnecessary logs and helping focus on important events. Advanced Audit Policies are accessible through **secpol.msc** under **Advanced Audit Policy Configuration**.

Advanced Audit Policies break down each audit category into **subcategories**, enabling more detailed monitoring. For example:

- **Logon/Logoff Events**: Can be divided into **Logon**, **Logoff**, **Special Logon**, **Account Lockout**, and more.
- **Account Management**: Tracks changes to user accounts, but subcategories allow monitoring of specific actions like **User Creation** or **Password Reset**.

Using Advanced Audit Policies, administrators can target specific activities (e.g., **only failed RDP login attempts**) and avoid logging every logon event. This **granularity improves visibility** while reducing noise in the logs, making incident detection more efficient.

Differences Between Local Policies and Advanced Audit Policies

Aspect	Local Policies	Advanced Audit Policies
Control	High-level control over broad categories	Fine-grained control with subcategories
Logging Granularity	Less specific, captures a wider range of activities	Highly specific, logs targeted actions
Configuration Scope	Set at a high level for events like "Logon"	Can filter events (e.g., RDP logons vs. local logons)
Management Tool	Local Security Policy (secpol.msc) or Group Policy	Advanced Audit Policy Configuration (secpol.msc)
Policy Overlap	Provides basic auditing policies	Overrides local policies if both are configured
Use Case	Good for basic auditing needs	Useful for organizations needing detailed logging

How They Work Together

Advanced Audit Policies offer **better control** than Local Policies. When both policies are enabled, the **Advanced Audit Policies take precedence**, ensuring more specific activities are logged based on the finer settings configured. For example, if both local and advanced policies enable logon auditing, the system will follow the advanced policy, allowing you to capture only remote logons instead of all logons.

Windows Services Events

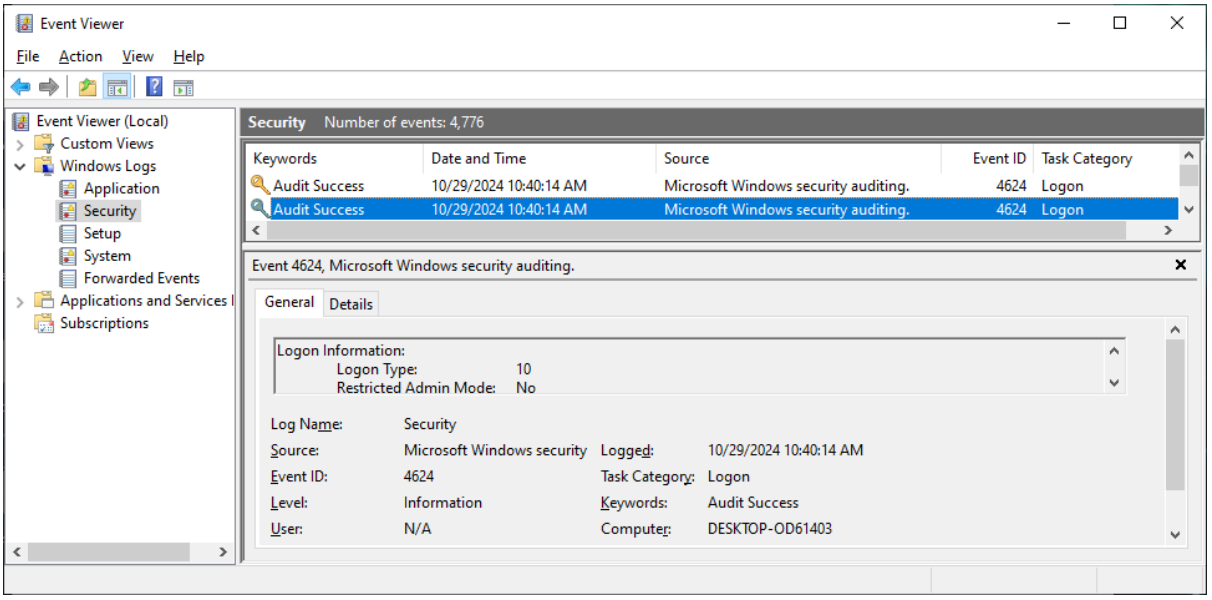
Remote Connections and System Monitoring Using Event Viewer

Remote connections, particularly through protocols like **Remote Desktop Protocol (RDP)**, are a cornerstone of modern system administration. However, they also represent a common attack vector for malicious actors attempting to gain unauthorized access to systems. Monitoring these remote sessions using **Event Viewer** helps detect suspicious activity, identify security risks such as brute force attacks, and understand session behavior.

Tracking RDP Connections Using Event Viewer

Tracking RDP connections requires monitoring key events that indicate when users log in and out of the system. In **Windows Security logs**, **Event ID 4624** represents a successful logon, while **Event ID 4634** marks the logoff. When using RDP, these events include **Logon Type 10**, which indicates a remote interactive session. Understanding and tracking these events is essential to detect both legitimate use and potential misuse of remote access.

1. Open **Event Viewer** by pressing **Windows + R**, typing eventvwr.msc, and pressing **Enter**.
2. Navigate to **Windows Logs > Security**.
3. In the right-hand pane, select **Filter Current Log** and enter 4624 in the **Event IDs** field. Click **OK** to apply the filter.
4. Review the **Logon Type** field within each event. Look specifically for **Logon Type 10**, which indicates an RDP session.



5. Note the **Account Name** and **Source Network Address** fields to understand who accessed the system and from where.

After identifying successful RDP logons, repeat the process using **Event ID 4634** to track logoffs. By correlating these events using the **Logon ID** field, you can calculate session durations and identify abnormal behavior, such as extremely long or unusually short sessions.

Detecting Unusual Login Attempts and IP-based Anomaly Detection

One of the most common ways attackers exploit RDP is through **brute force attacks**, where they repeatedly attempt to guess user passwords. **Event ID 4625** represents failed login attempts and can indicate such an attack if there are many failed attempts within a short period. Detecting suspicious login attempts involves reviewing **source IP addresses** and identifying patterns that deviate from normal behavior.

To filter for failed login attempts:

1. In the **Security** log, click **Filter Current Log** and enter 4625 as the **Event ID**.
2. Review the **Source Network Address** field for any **unexpected IP addresses**.
3. Check the **Account Name** field to identify which account was targeted. If a high-value account, such as an administrator, is the target, it could indicate a targeted attack.

Look for patterns such as multiple failed attempts from the same IP address followed by a **successful logon (Event ID 4624)**. This sequence could indicate that a brute force attempt has succeeded.

Additionally, you can perform **geolocation checks** on IP addresses to determine whether they originate from unusual or unexpected regions. If you notice frequent attempts from IP addresses outside your organization's operating region, it could signal an attack from a **botnet or unauthorized actor**.

Investigating Brute Force Attacks Using Event Correlation

Brute force attacks are often characterized by repeated **failed login attempts** followed by a sudden **successful login**. In these cases, it is critical to correlate **Event ID 4625** (failed login) with **Event ID 4624** (successful login) to detect compromised accounts.

1. Filter for **Event ID 4625** in the **Security log** and look for multiple failed attempts targeting the same user account.
2. Next, filter for **Event ID 4624** and review the timeframes and **Logon ID** values to identify any successful logins occurring shortly after the failed attempts.
3. If you find a match, this may indicate a **compromised account**. Further investigation should include reviewing other related events, such as privilege escalations or unusual account activity.

Analyzing the **Source Network Address** field for these events provides insight into the location of the attempts. If all failed attempts come from one IP address and the subsequent successful login originates from a different IP, it could indicate the involvement of multiple actors or a pivot through compromised devices.

Analyzing Session Duration Using Event Viewer

Tracking the **duration of remote sessions** helps identify abnormal or suspicious activity. Unusually long sessions could indicate an unauthorized user performing malicious activities, while very short sessions might suggest testing or probing behavior.

To analyze session durations:

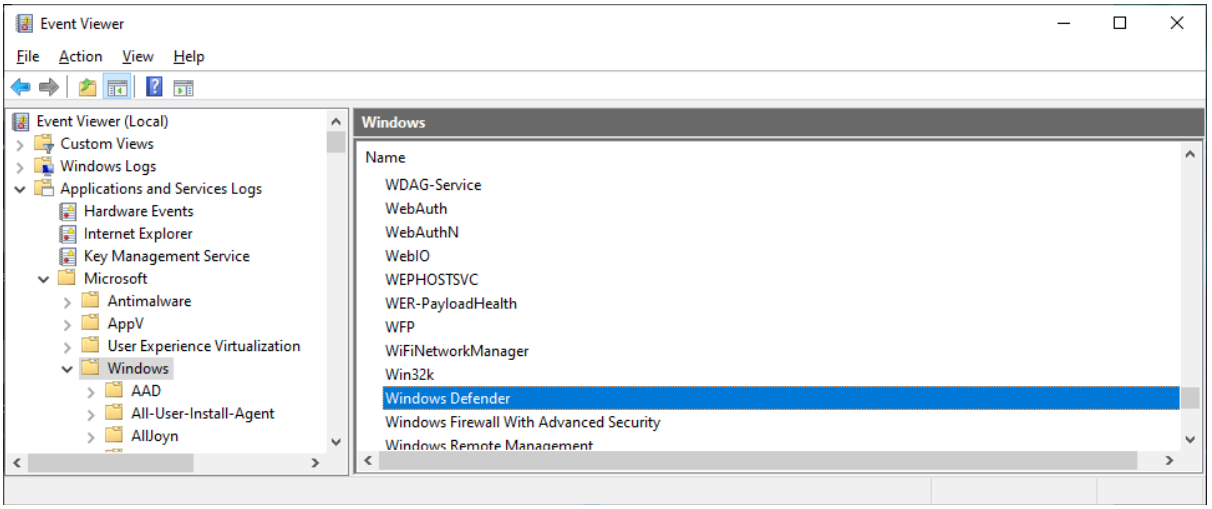
1. Open the **Security log** and filter for **Event ID 4624** (logon). Take note of the **Logon ID** and **TimeCreated** values.
2. Next, filter for **Event ID 4634** (logoff) and find the matching **Logon ID** to identify when the session ended.
3. Calculate the session duration by comparing the **TimeCreated** values of the logon and logoff events.

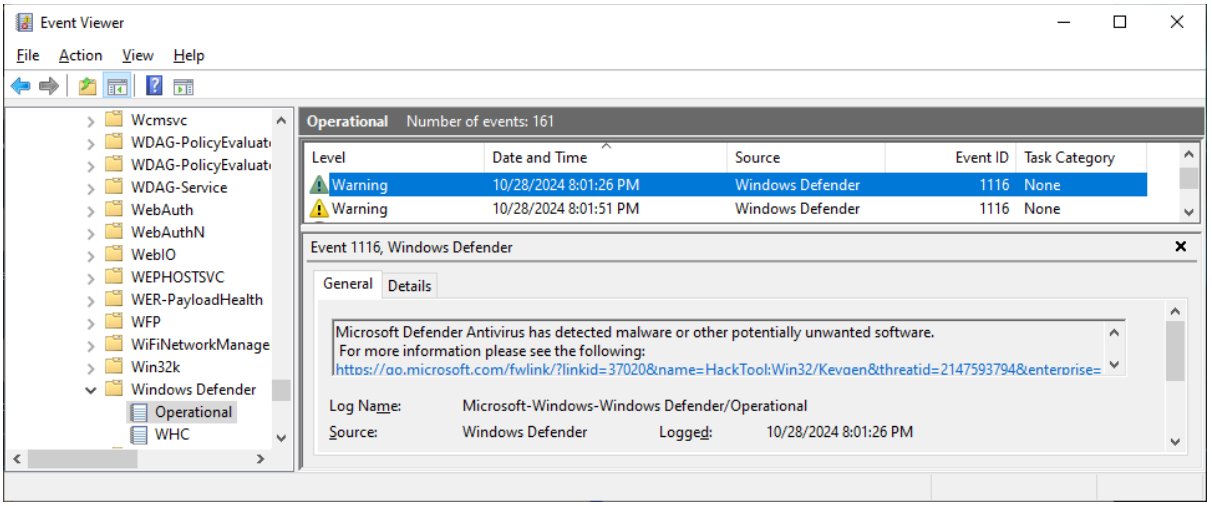
For continuous monitoring, you can create a **custom view** in Event Viewer that captures both **logon and logoff events**. This makes it easier to track session durations and detect anomalies without manually filtering the logs each time.

Monitoring Windows Defender for Malware Detection

Windows Defender provides **real-time protection** against malware, and it logs significant events such as malware detections, quarantines, and removals. **Event ID 1116** records malware detection and the action taken by Defender.

1. In the Event Viewer, navigate to **Applications and Services Logs > Microsoft > Windows > Windows Defender > Operational**.





2. Filter the log by **Event ID 1116** to view recent malware detection events.
3. Review the **File Path** and **Threat Name** fields to determine what was detected and where it was found.
4. If malware was quarantined or removed, further investigation is needed to ensure no other parts of the system were compromised.

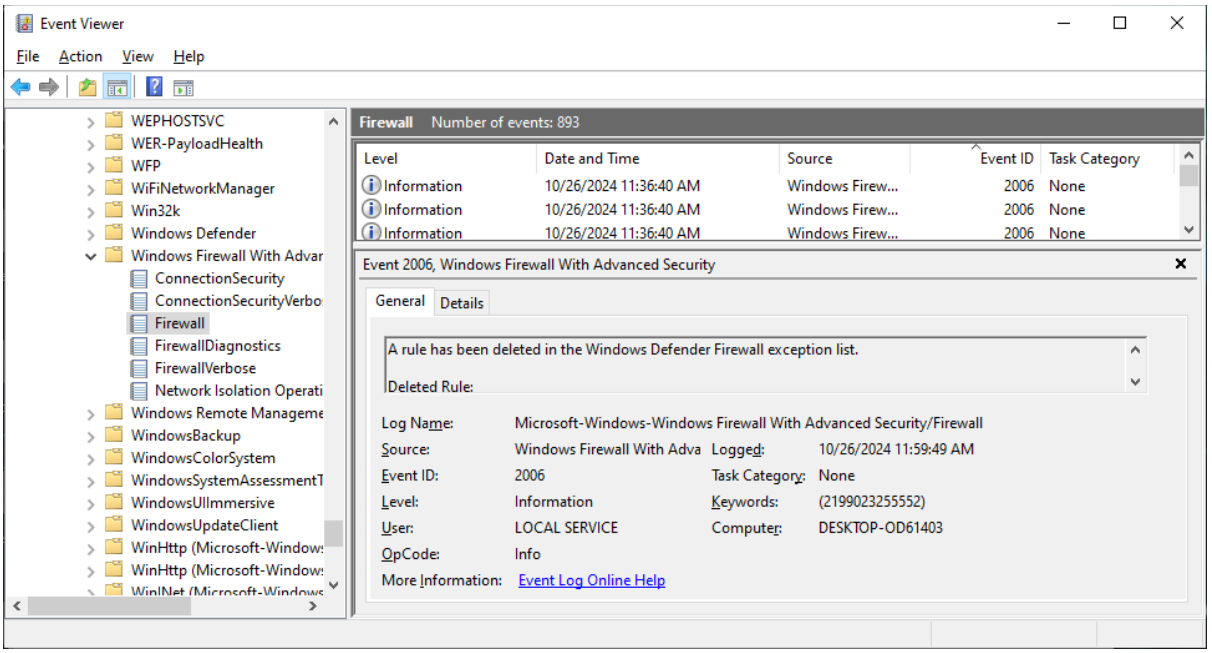
Configuring Real-time Monitoring Alerts in Windows Defender

Enabling **real-time monitoring alerts** ensures that administrators are notified immediately when malware is detected. Alerts can be configured through the **Windows Security app** or with **Group Policy** for enterprise environments.

In addition to configuring alerts, it is essential to **analyze false positives**—legitimate files that were mistakenly flagged as malicious. This can be done by cross-referencing the detected file with **VirusTotal** or similar threat intelligence platforms.

Monitoring Windows Firewall Logs for Traffic and Rule Changes

Windows Firewall plays a critical role in controlling **inbound and outbound network traffic**. Monitoring the firewall for **rule changes and traffic anomalies** helps detect unauthorized modifications or malicious connections.



Event ID 2004 records any **rule changes** in the Windows Firewall. Investigating these changes is essential to ensure that new rules were not added by an attacker attempting to allow unauthorized traffic.

To view firewall logs:

1. Open **Event Viewer** and navigate to **Applications and Services Logs > Microsoft > Windows > Windows Firewall with Advanced Security > Firewall**.

2. Filter by **Event ID 2004** to view recent rule changes.
3. Investigate any unexpected changes to firewall rules, such as newly allowed ports or IP addresses.

Firewall logs also help identify **blocked traffic**. By reviewing blocked connections, you can detect attempts to access the system from **untrusted IPs**. If necessary, correlate the firewall logs with the **Security log** to determine whether the blocked traffic aligns with any suspicious login attempts or malware activity.

Investigating Inbound and Outbound Traffic Anomalies

Monitoring **outbound traffic** is just as important as tracking inbound connections. Outbound connections to suspicious IP addresses or domains could indicate that malware is trying to **exfiltrate data** or establish a **command-and-control (C2) connection**.

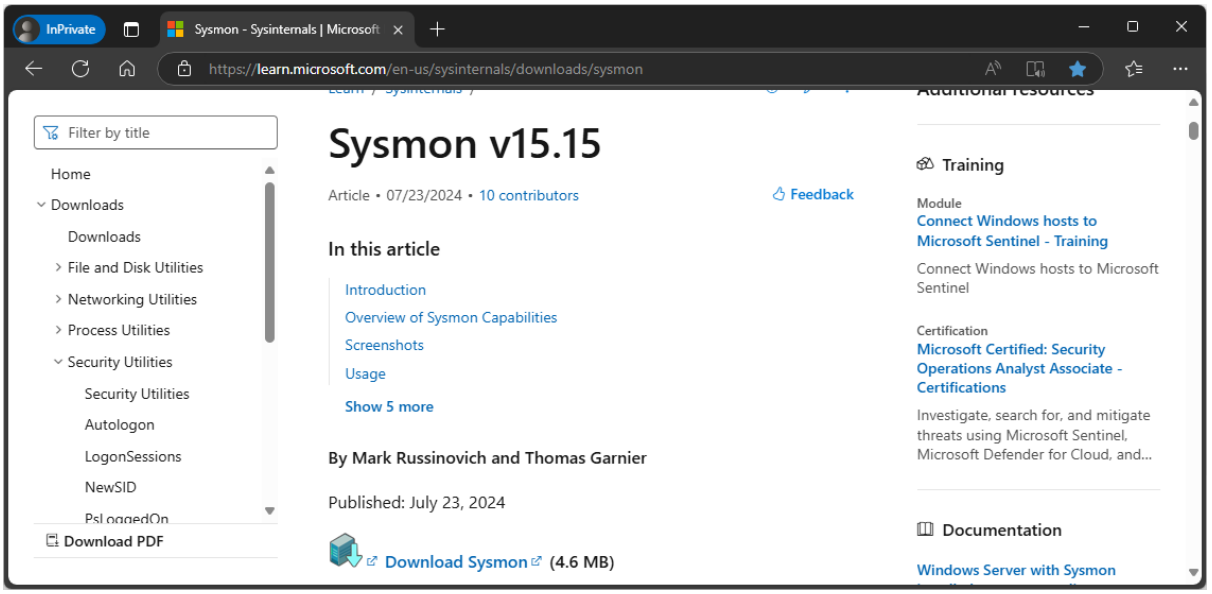
1. In the **Firewall log**, search for any outbound connections that were **blocked or allowed**.
2. Cross-reference the IP addresses with threat intelligence sources to identify known malicious IPs.
3. If suspicious outbound connections are detected, investigate the processes responsible for initiating those connections by checking the **Process Name** field in the event logs.

Effectively monitoring **remote connections**, **Windows Defender alerts**, and **firewall logs** using the Event Viewer is essential for maintaining system security. Tracking **RDP logons and logoffs** helps detect unauthorized access and ensures that session durations are within normal limits. Detecting **brute force attacks** through failed logon attempts and correlating them with successful logons provides insight into potential compromises. **Windows Defender logs** offer visibility into malware detection and quarantine events, while **firewall logs** help monitor network traffic and rule changes.

Sysmon

Installing and Configuring Sysmon

Sysmon (System Monitor) is a powerful Windows tool that enhances event logging by providing detailed information about system activities, such as process creation, network connections, and file modifications. It is essential for security monitoring, as it offers insights beyond the standard Windows event logs, making it a valuable tool for threat detection, forensic analysis, and incident response.



Understanding Sysmon's Purpose and Usage

Sysmon is part of the **Microsoft Sysinternals suite** and provides detailed event logging for **process creation, network connections, registry modifications, and file access**. Unlike standard Windows event logs, Sysmon captures additional context, such as **command-line parameters** used to launch processes, **network connection details**, and relationships between parent and child processes. These insights are crucial for identifying advanced threats like **malware, lateral movement, and privilege escalation**. Sysmon events are logged in the **Applications and Services Logs > Microsoft > Windows > Sysmon/Operational** section of the Event Viewer.

Sysmon is particularly useful for:

- Detecting **malware** by identifying suspicious process launches
- Monitoring **network activity** for unusual outbound connections or C2 (Command and Control) traffic
- **Correlating events** across different system activities to build a comprehensive view of potential security incidents

Installing Sysmon without an XML File

While installing **Sysmon with an XML configuration** is the recommended practice for advanced logging control, it is possible to install Sysmon **without an XML file**. In this configuration, Sysmon starts with **default settings**, which enable basic logging but may result in more verbose logs since there are no specific filters in place. This is useful for quick deployments or when you are testing Sysmon before deploying it with a refined configuration.

Purpose of Installing Sysmon Without an XML File

Installing Sysmon without an XML configuration allows you to:

- Quickly deploy Sysmon to **monitor basic system activities**.
- Capture **all process creation events and network connections** without filtering.
- Test how Sysmon operates before committing to a custom XML template.
- Review the default events Sysmon generates and adjust your configuration later.

Steps to Install Sysmon

1. **Download Sysmon**

   Visit the **Microsoft Sysinternals** website and download the latest Sysmon release.

2. **Open Command Prompt as Administrator**

   Press **Windows + R**, type cmd, and press **Enter**. In the Command Prompt window, type:

3. **Run the Installation Command**

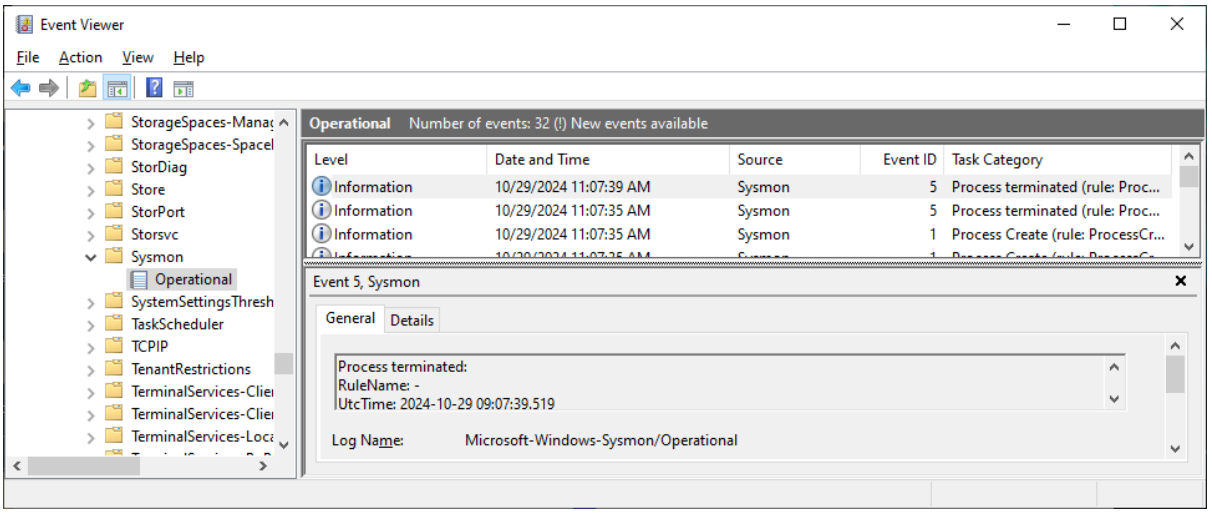
   Install Sysmon with default settings by running:

```
sysmon -accepteula -i
```

The **-accepteula** parameter accepts the license agreement, and the -i parameter installs Sysmon without specifying a configuration file.

4. **Verify the Installation**

   Open **Event Viewer** and navigate to **Applications and Services Logs > Microsoft > Windows > Sysmon/Operational**. You should start seeing **Sysmon events** generated based on the default configuration.



Default Sysmon Behavior Without XML

When installed without an XML file, Sysmon logs **all process creation and network connection events**. However, because no filtering is applied, the logs may contain **a large number of benign events**. Below are some key event types you can expect with a default installation:

- **Event ID 1**: Process Creation – Logs every new process launched, including command-line arguments.
- **Event ID 3**: Network Connection – Captures every network connection attempt made by processes.
- **Event ID 5**: Process Terminated – Logs when a process ends.
- **Event ID 7**: Image Loaded – Records when a DLL or other executable file is loaded into a process.
- **Event ID 8**: CreateRemoteThread – Indicates remote thread creation, often associated with **process injection attacks**.

Installing Sysmon with XML Templates

Installing and configuring Sysmon with an **XML configuration template** allows you to define which events are captured and how they are logged. Microsoft provides a **base XML template**, but it is highly recommended to **customize the configuration** to match your organization's security needs.

Step-by-Step Installation of Sysmon

1. Download the Sysmon executable from the **Microsoft Sysinternals** website.
2. Open a **Command Prompt** as Administrator.
3. Navigate to the directory where you downloaded Sysmon.
4. Install Sysmon using the following command:

```
sysmon -accepteula -i sysmonconfig-export.xml
```

This command installs Sysmon with the provided **XML configuration**. The -i flag indicates installation, and sysmonconfig-export.xml is the template used to define the logging behavior.

To verify that Sysmon is running, check the **Applications and Services Logs > Microsoft > Windows > Sysmon/Operational** section in the Event Viewer.

Best Practices for Reducing Noise in Logs

Sysmon can generate a large number of events, especially on busy systems, which can make it challenging to identify relevant incidents. Applying best practices for **noise reduction** helps streamline event analysis:

- **Exclude common processes**: Avoid logging benign processes like **explorer.exe** unless you suspect malicious activity.
- **Use specific conditions**: Configure filters to capture **only high-risk activities**. For example, monitor PowerShell usage by filtering for **-EncodedCommand** in the command line.
- **Leverage existing templates**: Use well-maintained community templates (like the **SwiftOnSecurity Sysmon config**) as a starting point.
- **Monitor critical accounts**: Focus on activities performed by high-privilege accounts or administrative tools.

Correlating Sysmon Events with Security Incidents

Sysmon events provide context that can be correlated with other **security events**, such as **failed logon attempts (Event ID 4625)** or **malware detections (Windows Defender Event ID 1116)**. Correlating these events helps build a **timeline of an incident**, providing valuable insights for forensic analysis.

Analyzing Network Events in Sysmon

Event ID 3 in Sysmon logs **network connections** initiated by processes, including details about IP addresses, ports, and protocols. Monitoring these events helps detect **C2 (Command and Control) traffic** and other malicious connections.

1. Open **Event Viewer** and navigate to **Sysmon/Operational**.
2. Filter the log by **Event ID 3** to view recent network connections.
3. Review the **DestinationIp** and **DestinationPort** fields to identify unusual outbound connections.
4. Check the **Image** field to see which process initiated the connection. If a non-networking tool like notepad.exe is making network connections, it could indicate process injection.

Detecting Unusual Outbound Connections

Unusual outbound connections, especially to **external IPs or uncommon ports**, may indicate **data exfiltration** or malware communication. Monitor connections initiated by **high-risk processes** such as **powershell.exe** or **cmd.exe**.

- If outbound traffic is detected on ports typically associated with **C2 traffic** (e.g., 8080, 443), investigate further.
- Compare the **destination IP addresses** with threat intelligence feeds to determine if they are associated with known malicious actors.

Cross-referencing Sysmon Logs with Firewall Events

Sysmon network events can be **correlated with Windows Firewall logs** to detect blocked traffic or unauthorized rule changes. For example, if Sysmon logs show an outbound connection attempt that was blocked by the firewall, it could indicate malicious activity.

1. Open **Event Viewer** and filter the **Sysmon log** by **Event ID 3** to find the network connection event.
2. Cross-reference the connection attempt with **Windows Firewall logs** to determine whether it was blocked or allowed.
3. Investigate any **discrepancies between the firewall logs and Sysmon events** to identify potential misconfigurations or evasion attempts.

Mastering Event Analysis

Understanding event analysis is a fundamental skill for SOC analysts and system administrators. Effective log monitoring allows you to detect unauthorized activities, investigate incidents, and secure your environment.

Understanding Key Event IDs: 4624, 4625, and 4634

What is Event ID 4624 in Windows logs?

Event ID 4624 signifies a **successful logon**. It contains details about the username, the source of the logon (such as a local machine or remote IP), and the **logon type** (e.g., interactive or remote). This event is essential for tracking authorized access and understanding login patterns.

How does Event ID 4625 help in security monitoring?

Event ID 4625 logs **failed logon attempts**. These events serve as an early warning sign of **brute-force attacks** or attempts to gain unauthorized access. Multiple failed attempts, especially from the same IP address or targeting high-privilege accounts, indicate a potential security threat.

What is the significance of Event ID 4634?

Event ID 4634 logs **user logoffs**, marking the end of a session. It helps analysts monitor when a session begins and ends. By correlating **logon events (4624)** with **logoff events (4634)** using the **Logon ID field**, you can calculate the **session duration** and detect abnormal behavior, such as extended or unusually short sessions.

Identifying and Tracking Remote Desktop Protocol (RDP) Connections

What does Logon Type 10 represent, and why is it important?

Logon Type 10 indicates a **Remote Desktop Protocol (RDP) session**. Monitoring these logons is critical because RDP is a frequent target for attackers attempting to gain remote access. Filtering by **Event ID 4624** with **Logon Type 10** allows you to track remote sessions and investigate suspicious activities.

How can you detect unusual login attempts using IP-based anomaly detection?

Detecting unusual logins involves tracking the **source IP addresses** in **RDP logon events**. If logins occur from unfamiliar or foreign IPs, it may indicate malicious activity. Use geolocation tools to identify the origin of IPs and compare them against known safe addresses. Monitoring changes in login behavior helps prevent unauthorized access.

What is a brute-force attack, and how can it be identified through logs?

A brute-force attack involves attempting many passwords in rapid succession to guess a user's credentials. This type of attack generates multiple **failed logon events (4625)**. If a successful logon (4624) occurs after many failed attempts, it may indicate that the attacker has succeeded. Monitoring these sequences helps detect compromised accounts early.

Effective Use of Event Viewer for Log Analysis

How does filtering by event ID help in log analysis?

Filtering by **event ID** allows analysts to focus on specific activities, such as failed logins (4625) or successful logons (4624). This technique reduces noise in the logs and ensures that relevant events are not overlooked. For example, filtering by **ID 4625** highlights failed logon attempts, making it easier to spot brute-force attacks or unauthorized access attempts.

What are custom views, and how can they improve log monitoring?

Custom views in **Event Viewer** allow you to **save filters** for frequently monitored events. This makes it easy to access relevant logs without manually filtering each time. For example, you can create a custom view to monitor **login events**, filtering by **Event IDs 4624 and 4625**. This streamlines your workflow, making log analysis more efficient.

How can session duration be calculated using logon and logoff events?

To calculate session duration, **correlate Event ID 4624 (logon)** with **Event ID 4634 (logoff)** using the **Logon ID** field. The difference between the timestamps of these events reveals the duration of the session. Unusually long or short sessions may indicate suspicious behavior, such as unauthorized use or testing of access.

Enhancing Security through Log Correlation and Monitoring Tools

What is the importance of monitoring Logon Type 5 events?

Logon Type 5 indicates a **service logon**, where system services authenticate to perform scheduled tasks. Monitoring these events ensures that only authorized services are running. Unauthorized service logons can indicate misuse of **service accounts** for malicious purposes, such as maintaining persistence on a compromised system.

What strategies can be used to detect unusual login attempts?

Unusual login attempts can be detected by **analyzing source IPs** and **monitoring login patterns**. Look for repeated failed attempts (Event ID 4625) targeting sensitive accounts, especially from **foreign or unexpected IP addresses**. Integrating **threat intelligence feeds** helps identify known malicious IPs.

How can Event Viewer logs be correlated with firewall logs?

Correlating **Windows security logs** with **firewall logs** helps build a comprehensive view of an incident. For example, if a **failed login attempt (4625)** corresponds with blocked inbound traffic in the firewall, it may indicate that the attacker attempted remote access. Identifying such patterns allows you to respond proactively to threats.

Reducing Noise in Logs and Improving Efficiency

What are some best practices for noise reduction in event analysis?

Reducing noise ensures that analysts can focus on **critical events**. Some best practices include **excluding common benign events**, such as routine logons from trusted users, creating **custom views** to highlight relevant activities, and **correlating events** to identify patterns of malicious behavior. Using targeted filtering based on **logon type, event ID, or IP address** helps streamline analysis.

How can Windows Defender's Event ID 1116 assist in malware detection?

Event ID 1116 indicates that **Windows Defender** has detected malware and recorded the action taken, such as **quarantine or removal**. Reviewing these events helps ensure that detected threats are contained. Investigating the **file path and threat name** allows analysts to determine whether further remediation is needed.

What is the role of real-time monitoring alerts in log analysis?

Real-time monitoring alerts notify administrators of **critical events** as soon as they occur. For example, an alert for **failed logon attempts (4625)** can indicate an ongoing brute-force attack. Configuring alerts for **malware detections, logon attempts, and firewall rule changes** ensures timely responses to security incidents.

Practical Scenarios and Best Practices for Event Analysis

Understanding how to analyze logs and correlate events is essential for incident response. Suppose you detect **multiple failed logon attempts (4625)** from the same IP address followed by a **successful login (4624)**. This sequence suggests a brute-force attack that succeeded, and immediate action is required to block the account and investigate further. Similarly, if you notice an **unusual RDP login (Logon Type 10)** from an unfamiliar IP address, further checks are needed to determine if the login was authorized.

Creating **custom views** for high-priority events, such as failed logins and privilege escalations, improves efficiency by allowing you to focus only on relevant activities. **Log correlation** with other data sources, such as firewall logs, adds context to your analysis. For instance, identifying both **blocked firewall traffic** and **failed logon attempts** helps confirm attempted intrusions.