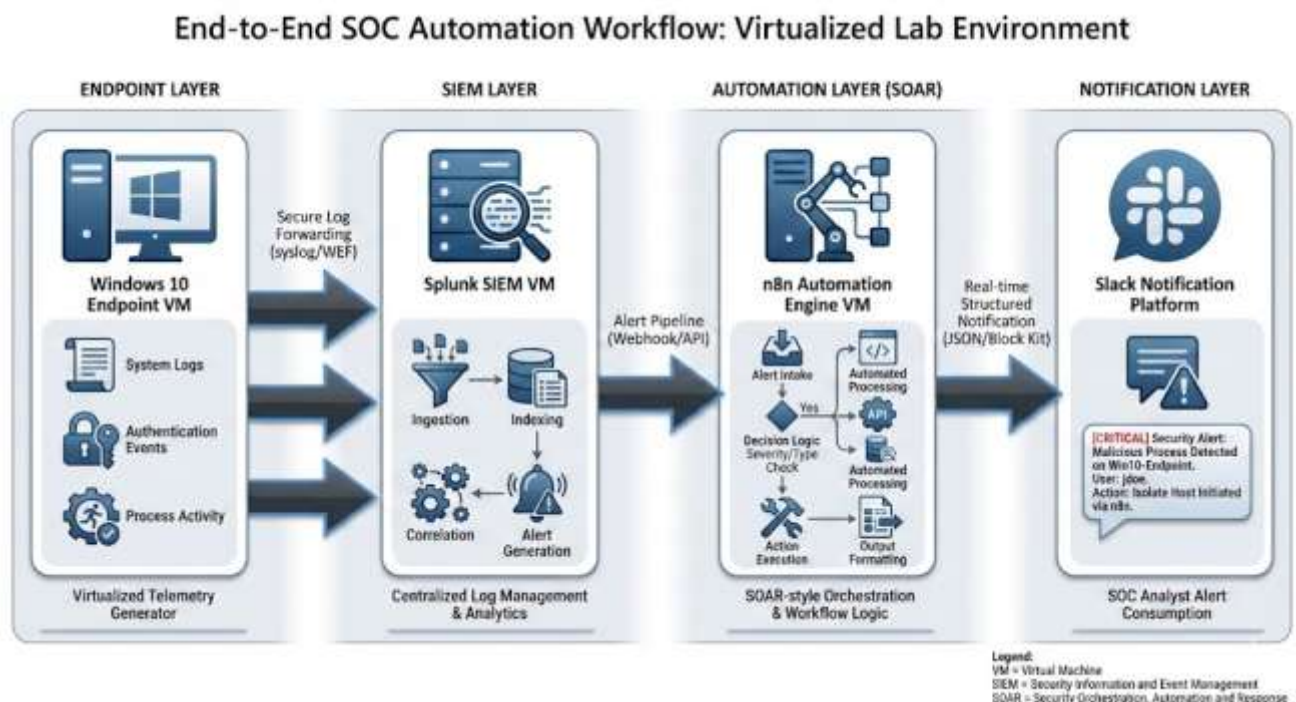


Design and Implementation of an Automated SOC Workflow

Objectives:

The objective of this lab is to demonstrate an end-to-end SOC automation workflow by integrating endpoint telemetry, SIEM detection, and automated response. A Windows 10 system generates security events that are ingested and analyzed by Splunk, where alerts are created based on defined detection logic. These alerts are then forwarded to n8n to trigger automated workflows, resulting in real-time security notifications delivered to Slack. This lab provides practical insight into SIEM–SOAR integration and automated SOC operations.



Step :1 Installing Splunk and Configuring GUI

In this step, Splunk Enterprise is installed on a dedicated virtual machine to serve as the centralized SIEM platform for the lab. After installation, the Splunk web interface is configured to enable access to the management console, verify successful deployment, and prepare the environment for log ingestion, search, and alert configuration in subsequent steps.

The initial phase of the SOC automation workflow begins with provisioning the central SIEM engine. This step involves selecting the Splunk Enterprise 10.2.0 installer specifically for a Linux environment. By choosing the Debian (.deb) package, the setup is optimized for deployment on a Linux-based server (such as Ubuntu), which will serve as the primary instance for data indexing, log analysis, and alert generation.

Choose Your Download

Splunk Enterprise 10.2.0

Index 500 MB/Day. Sign up and download now. After 60 days you can convert to a perpetual free license or purchase a Splunk Enterprise license to continue using the expanded functionality designed for enterprise-scale deployments.

Choose Your Installation Package

Windows

Linux

Mac OS

64-bit	4.x+, 5.x+, 6.x+ kernel Linux distributions	.tgz	1701.67 MB	Download Now	Copy wget link	More ▾
		.rpm	1701.65 MB	Download Now	Copy wget link	More ▾
		.deb	1291.35 MB	Download Now	Copy wget link	More ▾

The installation is performed on the Linux host by executing the Debian package manager command. This step unpacks the Splunk binaries and prepares the filesystem for the SIEM software deployment.

Command: `sudo dpkg -i splunk-10.2.0-d749cb17ea65-linux-amd64.deb`

```
safwan@safwan-VirtualBox:~$ sudo dpkg -i splunk-10.2.0-d749cb17ea65-linux-amd64.deb
```

The setup begins by acquiring the Splunk Enterprise 10.2.0 Debian package for Linux, which serves as the central log aggregator. The package is installed via the command line, followed by a transition to the dedicated splunk user environment to initialize the service from the binary directory.

- **Installer:** `splunk-10.2.0-linux-amd64.deb` (~1.3 GB).
- **Installation:** Executed using `sudo dpkg -i` to deploy files into the `/opt/splunk` directory.
- **User Management:** Switched to the service account using `sudo -u splunk bash` to ensure proper file permissions.

- **Initialization:** Navigated to /opt/splunk/bin and executed ./splunk start to trigger the license agreement and service startup.

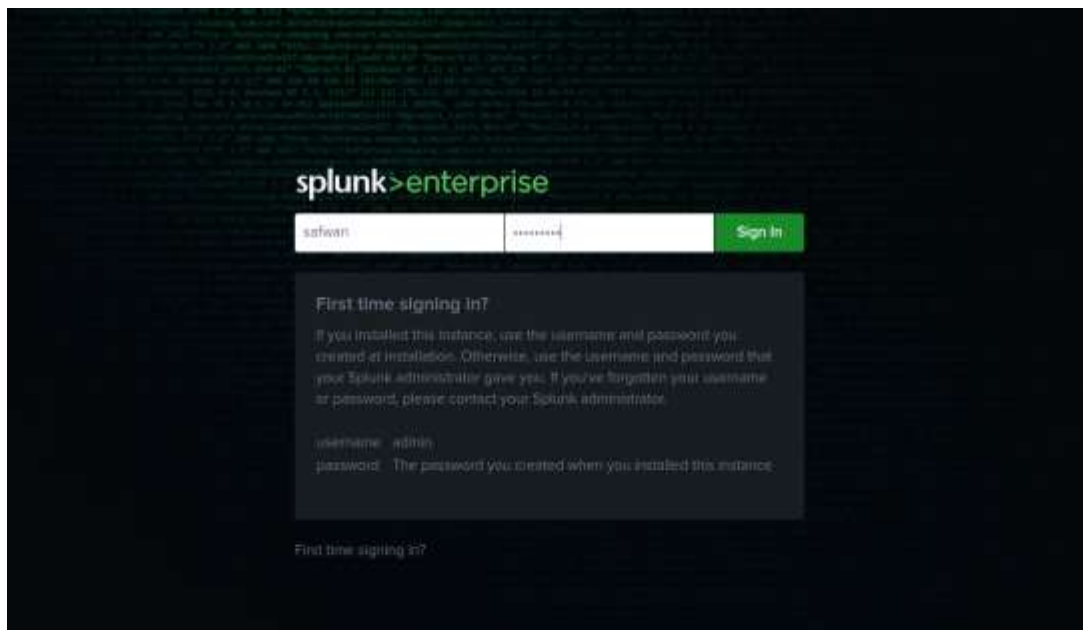
```
safwan@safwan-VirtualBox: /opt/splunk$ sudo -u splunk bash
[sudo] password for safwan:
splunk@safwan-VirtualBox: ~$ cd /opt/splunk
splunk@safwan-VirtualBox: ~$ cd bin
splunk@safwan-VirtualBox: ~/bin$ ./splunk start
```

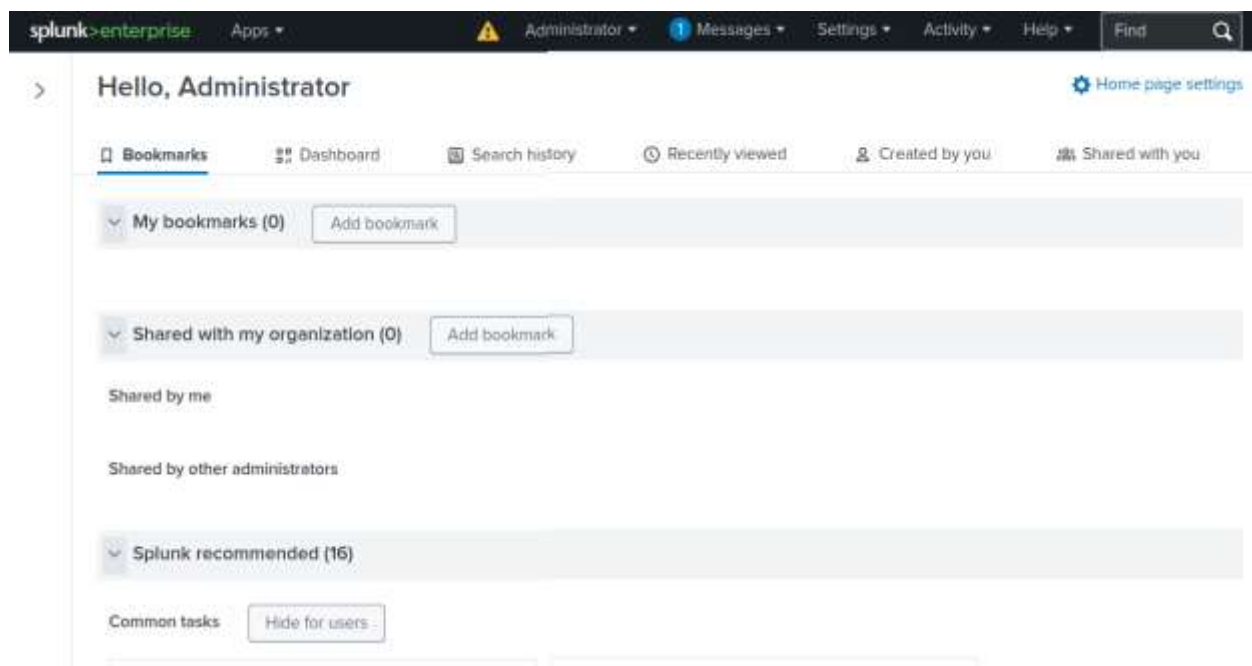
Configuration of the initial administrator username and a secure password (minimum 8 characters) to grant access to the Splunk Web interface.

```
Please enter an administrator username: safwan
Password must contain at least:
  * 8 total printable ASCII character(s).
Please enter a new password:
```

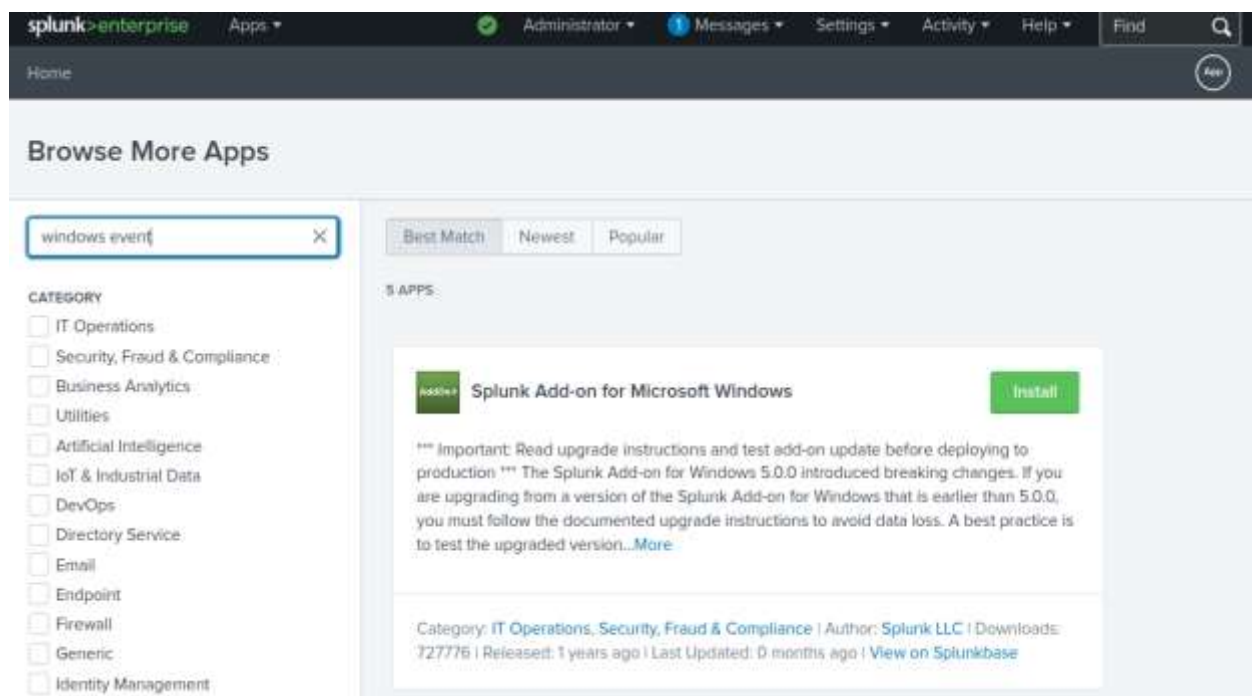
```
safwan@safwan-VirtualBox: /opt/splunk/bin$ sudo ./splunk enable boot-start -user splunk
Init script installed at /etc/init.d/splunk.
Init script is configured to run at boot.
safwan@safwan-VirtualBox: /opt/splunk/bin$
```

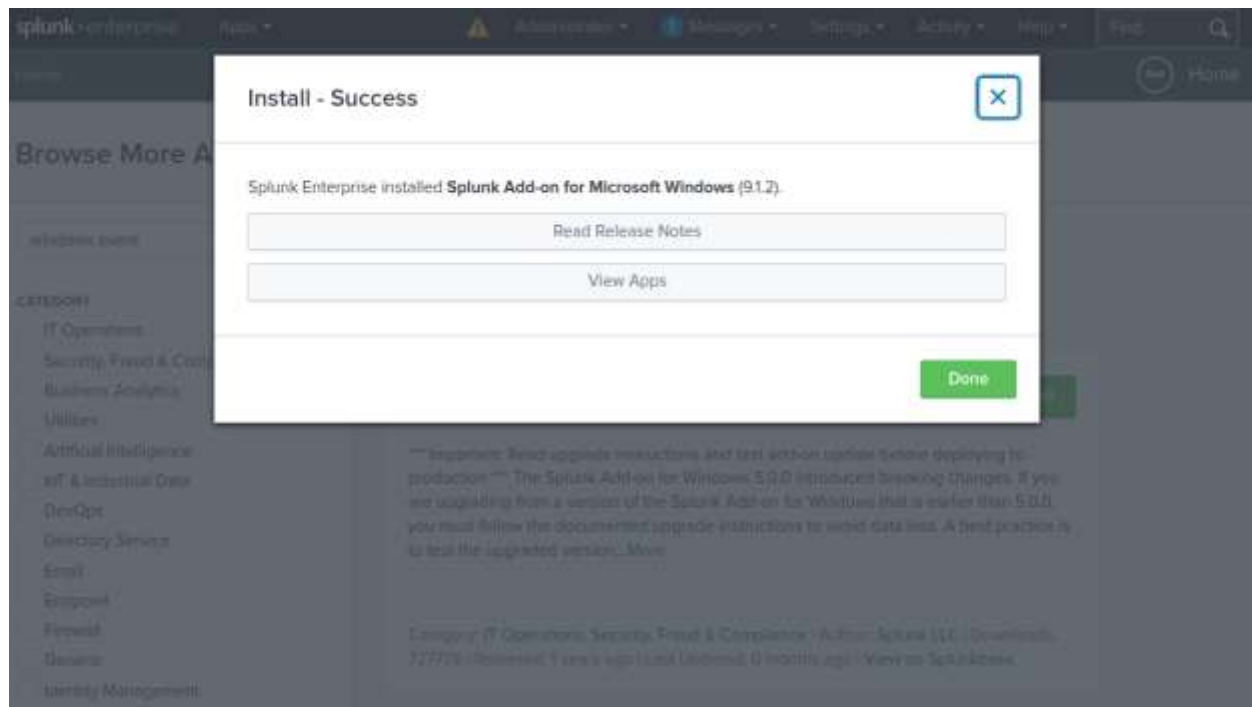
The Splunk Web interface is accessed by navigating to the host's IP address on port 8000 via a web browser. This GUI serves as the primary portal for logging in with the administrative credentials created during installation, enabling the transition from command-line setup to a visual environment for managing security events and automation tasks.





To enable endpoint telemetry ingestion, the **Splunk Add-on for Microsoft Windows** is installed via the Splunk Web GUI. This application provides the necessary knowledge objects to map Windows event logs to the Splunk CIM, ensuring that data from the Windows 10 system is correctly parsed and indexed for analysis





Part 2: Send Telemetry from Windows VM to Splunk

In this part, the Windows 10 virtual machine is configured to generate and forward system and security telemetry to the Splunk SIEM. This enables Splunk to ingest endpoint logs, validate data flow, and provide visibility into Windows activity for monitoring and detection purposes.

To enable telemetry collection from the Windows 10 VM, the Splunk Universal Forwarder 10.2.0 is selected for download. The 64-bit Windows .msi installer is chosen to ensure compatibility with modern Windows architectures. This lightweight agent is dedicated to harvesting local system logs and securely forwarding them to the Splunk indexer for analysis.

Choose Your Download

Splunk Universal Forwarder 10.2.0

Universal Forwarders provide reliable, secure data collection from remote sources and forward that data into Splunk software for indexing and consolidation. They can scale to tens of thousands of remote systems, collecting terabytes of data.

Choose Your Installation Package

Windows Linux Mac OS Free BSD Solaris AIX

32 bit	Windows 10	.msi	64.9 MB	Download Now	Copy wget link	More >
64 bit	Windows 10, 11 Windows Server 2019, 2022, 2025	.msi	166.76 MB	Download Now	Copy wget link	More >

After downloading , installation process starts. In this follow the exact same steps as per below figures displays for poper configuration.

UniversalForwarder Setup

splunk>universal forwarder

Create credentials for the administrator account. The password must contain, at a minimum, 8 printable ASCII characters.

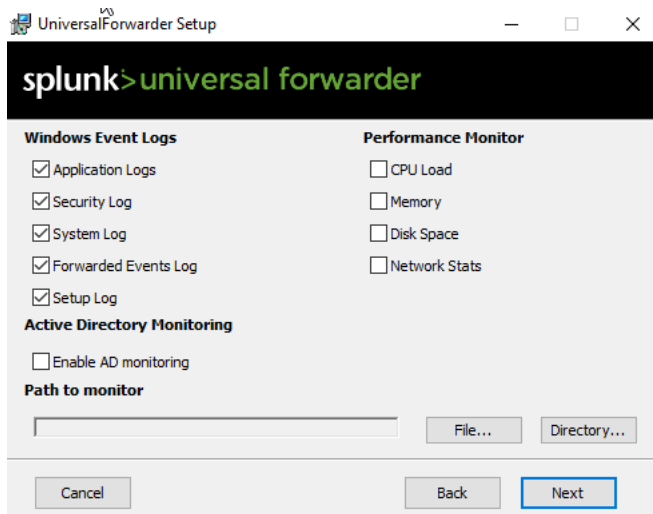
Username:

☐ Generate random password

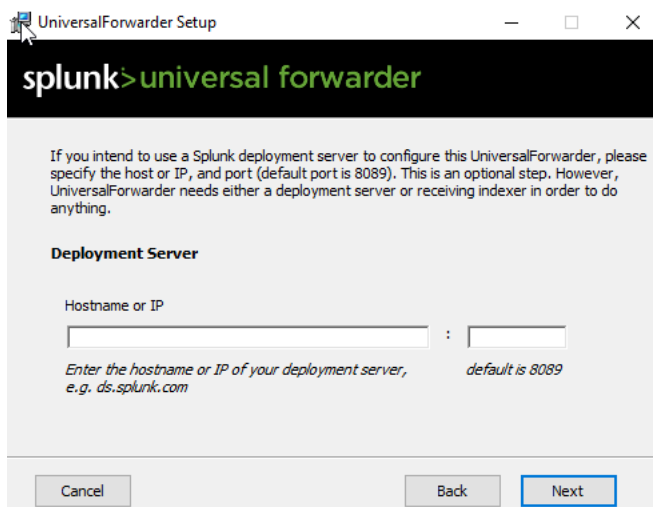
Password:

Confirm password:

[Cancel](#) [Back](#) [Next](#)



After this step Receiver Server option will come, so on that add the Splunk VM IP Address with 9997 port, as the Splunk listens for the telemetry at port 9997.



To prepare the SIEM for incoming endpoint telemetry, a receiver is configured within the Splunk Web interface. By navigating to Settings > Forwarding and receiving, a new receiving port is defined to listen for traffic from remote forwarders.

The standard **port 9997** is specified and saved, enabling the Splunk indexer to ingest data streams from the Windows VM. This configuration is vital for establishing the communication channel required for real-time log analysis and subsequent SOC automation.

The screenshot shows the Splunk Enterprise web interface. At the top is a navigation bar with the Splunk logo, 'Apps', and several menu items: 'Administrator', 'Messages', 'Settings', 'Activity', 'Help', and a search icon. Below the navigation bar, the page title is 'Add new'. Underneath, a breadcrumb trail reads 'Forwarding and receiving > Receive data > Add new'. The main content area is a white box titled 'Configure receiving'. Inside this box, there is a sub-header 'Set up this Splunk instance to receive data from forwarder(s)'. Below this, there is a label 'Listen on this port:' followed by a text input field containing the number '9997'. A small note below the input field says 'For example, 9997 will receive data on TCP port 9997.' At the bottom right of the white box are two buttons: 'Cancel' and 'Save'.

In this part add the IP Address of the Windows VM in the Source Type Description.

Input Settings

Optionally set additional input parameters for this data input as follows:

Source type

The source type is one of the default fields that the Splunk platform assigns to all incoming data. It tells the Splunk platform what kind of data you've got, so that the Splunk platform can format the data intelligently during indexing. And it's a way to categorize your data, so that you can search it easily.

The screenshot shows a form with three rows. The first row is labeled 'Source Type' and has a text input field containing 'windowslog'. Above this input field are two buttons: 'Select' and 'New'. The second row is labeled 'Source Type Category' and has a dropdown menu with 'Custom' selected. The third row is labeled 'Source Type Description' and has an empty text input field.

New Index ✕

General Settings

Index Name
Set index name (e.g., INDEX_NAME). Search using index=INDEX_NAME.

Index Data Type Events Metrics
The type of data to store (event-based or metrics).

Home Path
Hot/warm db path. Leave blank for default (\$SPLUNK_DB/INDEX_NAME/db).

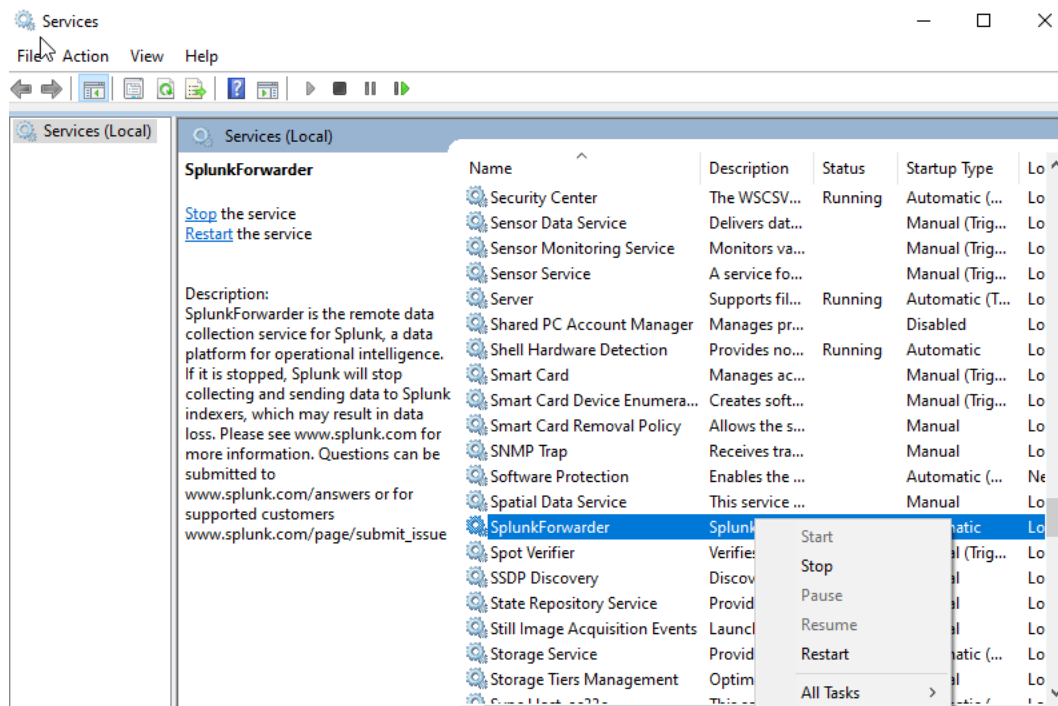
Cold Path
Cold db path. Leave blank for default (\$SPLUNK_DB/INDEX_NAME/colddb).

Thawed Path
Thawed/resurrected db path. Leave blank for default (\$SPLUNK_DB/INDEX_NAME/thaweddb).

Data Integrity Check Enable Disable
Enable this if you want Splunk to compute hashes on every slice of your data for the purpose of data integrity.

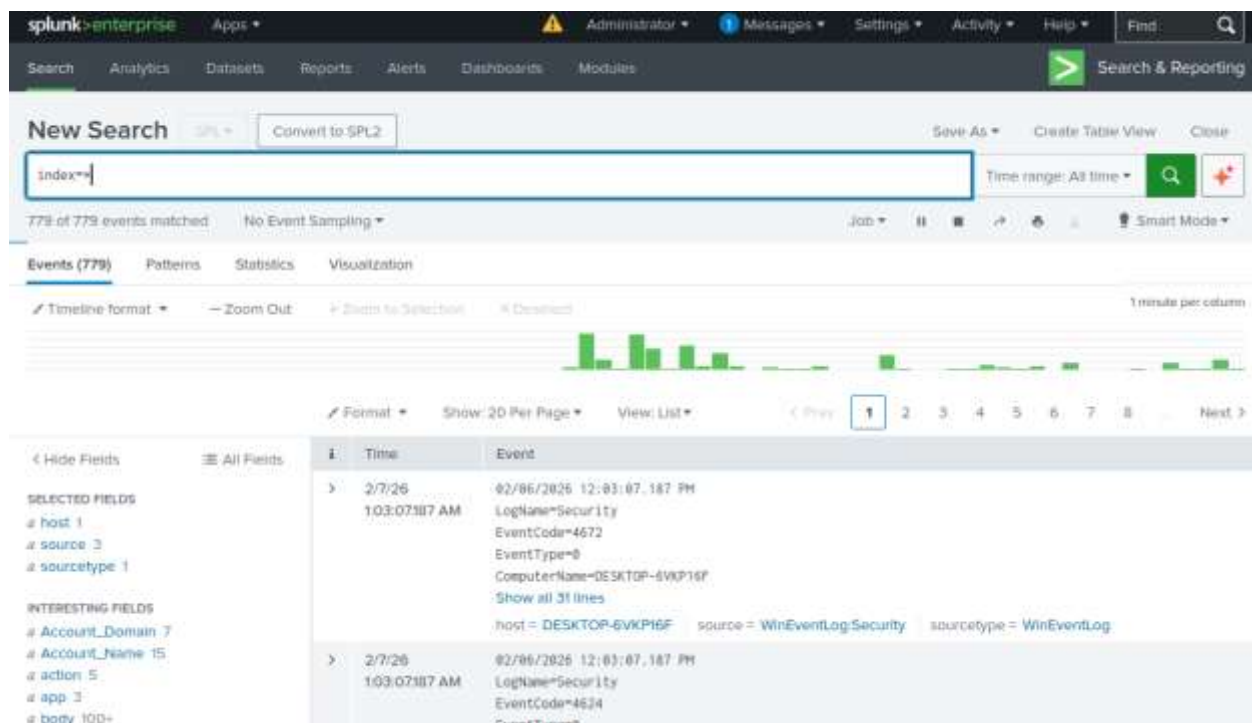
Save Cancel

Then go to the services in Windows and search for SplunkForwarder. Right click on it and restart it and ensure it should run smoothly. As this is a very crucial part of the process, if this is not configured properly the telemetry won't be sent over to the Splunk GUI.



The final stage of the telemetry setup involves verifying that the Windows event logs are successfully reaching the Splunk indexer. By executing a global search for **index=***, the Search & Reporting interface confirms the ingestion of real-time data from the Windows 10 endpoint.

- **Verified Source:** Telemetry is being received from the WinEventLog:Security source.
- **Event Confirmation:** Successful capture of critical security events, such as EventCode 4672 (Special Logon) and EventCode 4624 (An account was successfully logged on).
- **Host Identification:** The data is correctly mapped to the target Windows machine (DESKTOP-6VKP16F), indicating a healthy connection between the Universal Forwarder and the SIEM.



The screenshot displays the Splunk Search & Reporting interface. At the top, the navigation bar includes links for Search, Analytics, Datasets, Reports, Alerts, Dashboards, and Modules. The main search bar contains the query `index=*`, and the results show 779 of 779 events matched. Below the search bar, a timeline visualization shows the distribution of events over time. The search results are displayed in a table format, showing the Time and Event details for each entry.

Time	Event
2/7/26 10:07:18 AM	02/06/2026 12:03:07.187 PM LogName=Security EventCode=4672 EventType=0 ComputerName=DESKTOP-6VKP16F Show all 31 lines host = DESKTOP-6VKP16F source = WinEventLog:Security sourcetype = WinEventLog
2/7/26 10:07:18 AM	02/06/2026 12:03:07.187 PM LogName=Security EventCode=4624 EventType=0

Part 3: Installing n8n

In this part, the n8n automation platform is installed on a dedicated virtual machine. This sets up the environment for creating automated workflows that can process alerts from Splunk and trigger security responses or notifications.

Make separate directory for n8n configuration

```
safwan@safwan-VirtualBox:~$ mkdir n8n
safwan@safwan-VirtualBox:~$ cd n8n
safwan@safwan-VirtualBox:~/n8n$ sudo nano docker-compose.yaml
```

The installation of n8n is managed through Docker Compose, providing a consistent and isolated environment for the automation platform. By utilizing a YAML configuration file, the service is defined with persistent storage and specific networking parameters required for SOC orchestration.

- **Docker Image:** Utilization of the n8nio/n8n:latest image to ensure the platform remains up-to-date with the latest security features and nodes.
- **Port Mapping:** Binding the container's internal service to port 5678 on the host machine, enabling web browser access to the workflow editor.
- **Environment Variables:** Configuration of critical host details, including the Host Machine IP, protocol (HTTP), and timezone (America/Toronto) to ensure accurate log timestamps in automated responses.
- **Data Persistence:** Mapping a local volume (./n8n_data) to the container's internal storage to preserve workflows, credentials, and configuration data across service restarts.

```
GNU nano 7.2 docker-compose.yaml *
services:
  n8n:
    image: n8nio/n8n:latest
    restart: always
    ports:
      - "5678:5678"
    environment:
      - N8N_HOST= (Host Machine IP)
      - N8N_PORT=5678
      - N8N_PROTOCOL=http
      - N8N_SECURE_COOKIE=false
      - GENERIC_TIMEZONE=AMERICA/TORONTO
    volumes:
      - ./n8n_data:/home/node/.n8n
```

The `sudo docker-compose pull` command is executed to download the specific n8n image layers defined in the configuration file. This step ensures all necessary dependencies and the latest platform binaries are present on the local host before the container is officially started.

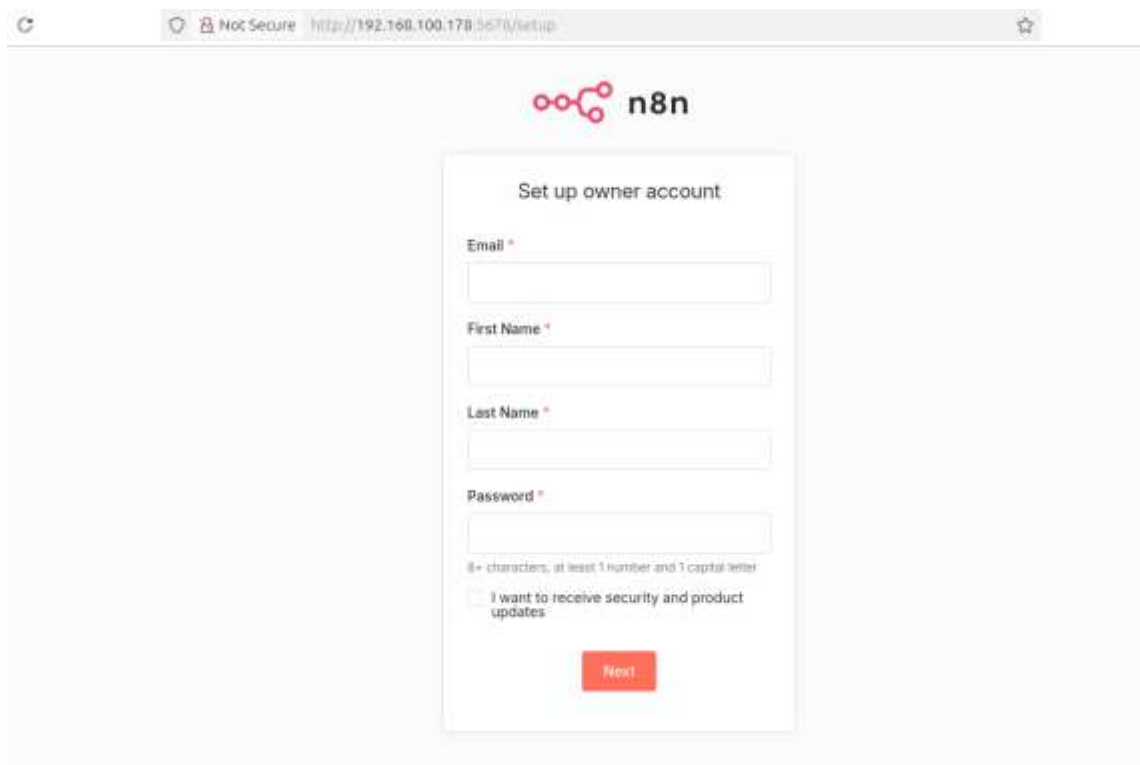
```
safwan@safwan-VirtualBox:~/n8n$ sudo docker-compose pull
```

The final step in the containerized deployment is the execution of the `sudo docker-compose up -d` command. This instruction initializes the n8n container in detached mode, allowing the automation platform to run in the background as a persistent service. Once active, the environment is ready for web-based access to begin workflow orchestration.

```
safwan@safwan-VirtualBox:~/n8n$ sudo docker-compose up -d
```

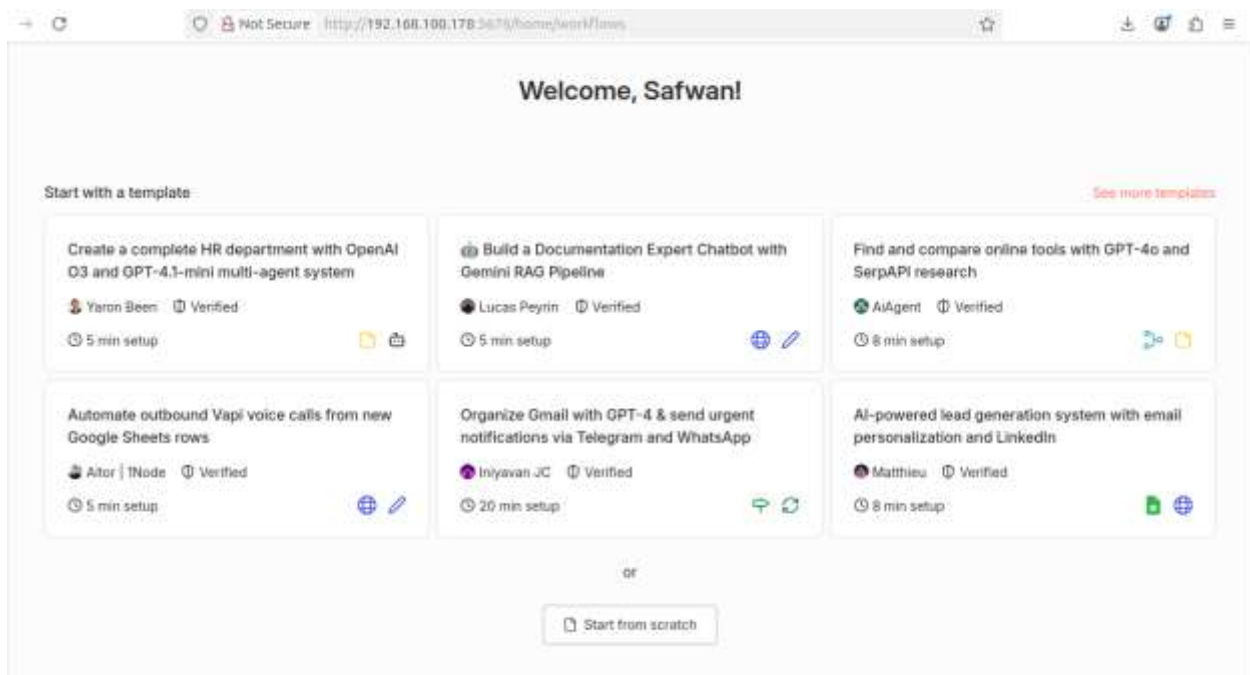
Once the container is running, the n8n web interface is accessed by navigating to the Host Machine IP on port 5678 via a web browser. This launches the initial setup wizard, which is the final step in preparing the automation environment for workflow creation.

- **Initial Access:** Navigating to `http://<Host_IP>:5678` connects the browser to the n8n service running inside the Docker container.
- **Owner Account Creation:** The "Set up owner account" screen requires the registration of an administrative user with a valid email, name, and a secure password (minimum 8 characters with numbers and capital letters).



The screenshot shows a web browser window with the address bar displaying "http://192.168.100.170:5678/setup". The page features the n8n logo at the top. Below the logo is a form titled "Set up owner account". The form contains four input fields: "Email *", "First Name *", "Last Name *", and "Password *". Below the password field, there is a note: "8+ characters, at least 1 number and 1 capital letter". At the bottom of the form, there is a checkbox labeled "I want to receive security and product updates" and a red "Next" button.

After the successful account creation the default GUI of n8n opens.



Part 4: Building Automation Workflow

In this part, an automated workflow is created in n8n to process alerts received from Splunk. The workflow defines the logic for handling security events and triggers corresponding actions, such as sending notifications to Slack, simulating a real-world SOC response process.

The first step in the automation workflow involves creating a scheduled alert in Splunk to detect specific security events. In the **Save As Alert** dialog, the search for EventCode=4625 (failed login attempts) is saved with the title "Safwan-Project". The alert is configured to run on a **Cron Schedule** of * * * * * (every minute) and is set to expire after 24 hours, ensuring continuous monitoring of the endpoint.

Save As Alert

Settings

Title: Safwan-Project

Description: Optional

Permissions: Private | Shared in App

Alert type: Scheduled | Real-time

Run on Cron Schedule ▼

Time Range: All time ▼

Cron Expression: *****
 e.g. 00 18 * * * * (every day at 6PM)
[Learn More](#)

Expires: 24 | hour(s) ▼

Cancel Save

To receive these alerts, a **Webhook** node is configured within the n8n interface. The HTTP Method is set to **POST**, and the node generates a unique **Test URL** (e.g., <http://192.168.100.178:5678/webhook-test/...>). This URL serves as the listener that will accept the JSON payload sent from Splunk whenever the alert is triggered.

Save As Alert

Trigger: Once | For each result

Throttle: ☐

Trigger Actions

+ Add Actions ▼

When triggered: ▼ Webhook Remove

URL: https://yourserver.com/foobar

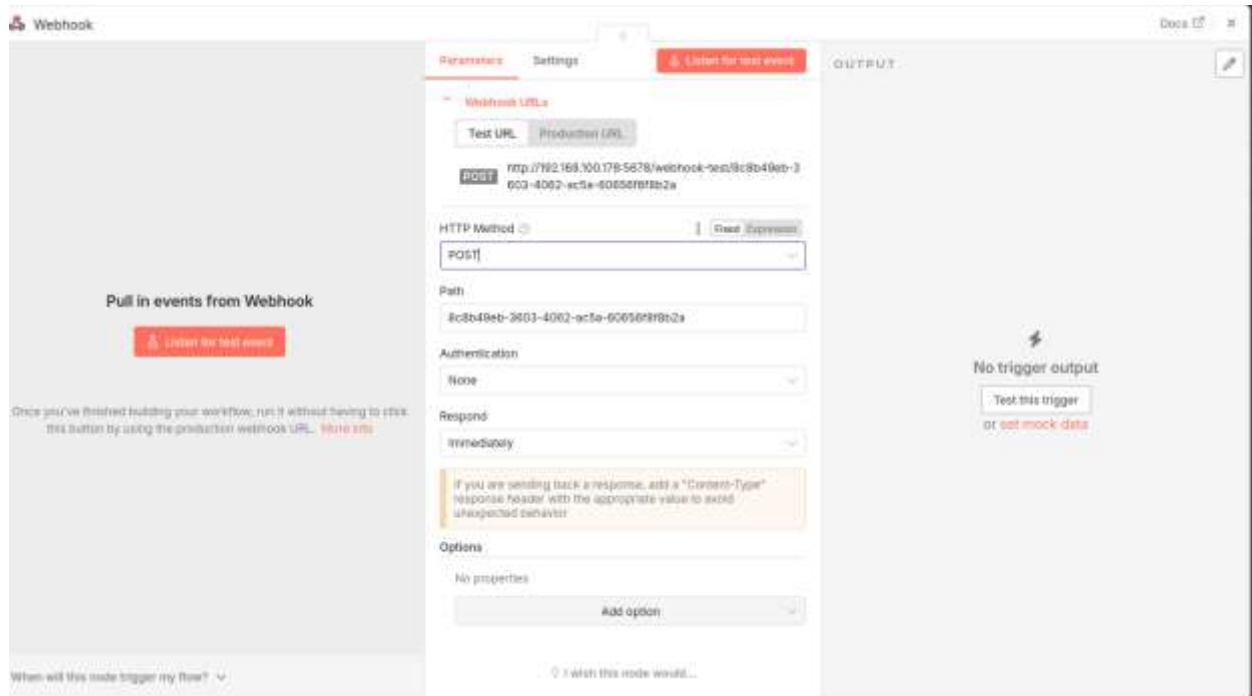
Specified URL to send JSON payload via HTTP POST (e.g., https://yourserver.com/api/webhook)

[Learn More](#)

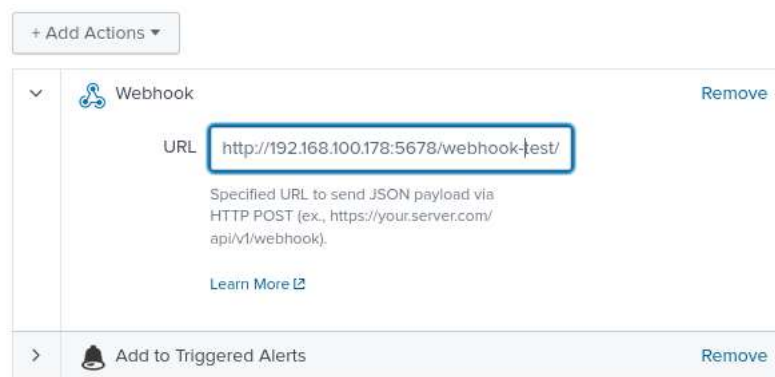
+ Add to Triggered Alerts Remove

Cancel Save

To receive these alerts, a Webhook node is configured within the n8n interface. The HTTP Method is set to POST, and the node generates a unique Test URL (e.g., <http://192.168.100.178:5678/webhook-test/...>). This URL serves as the listener that will accept the JSON payload sent from Splunk whenever the alert is triggered.

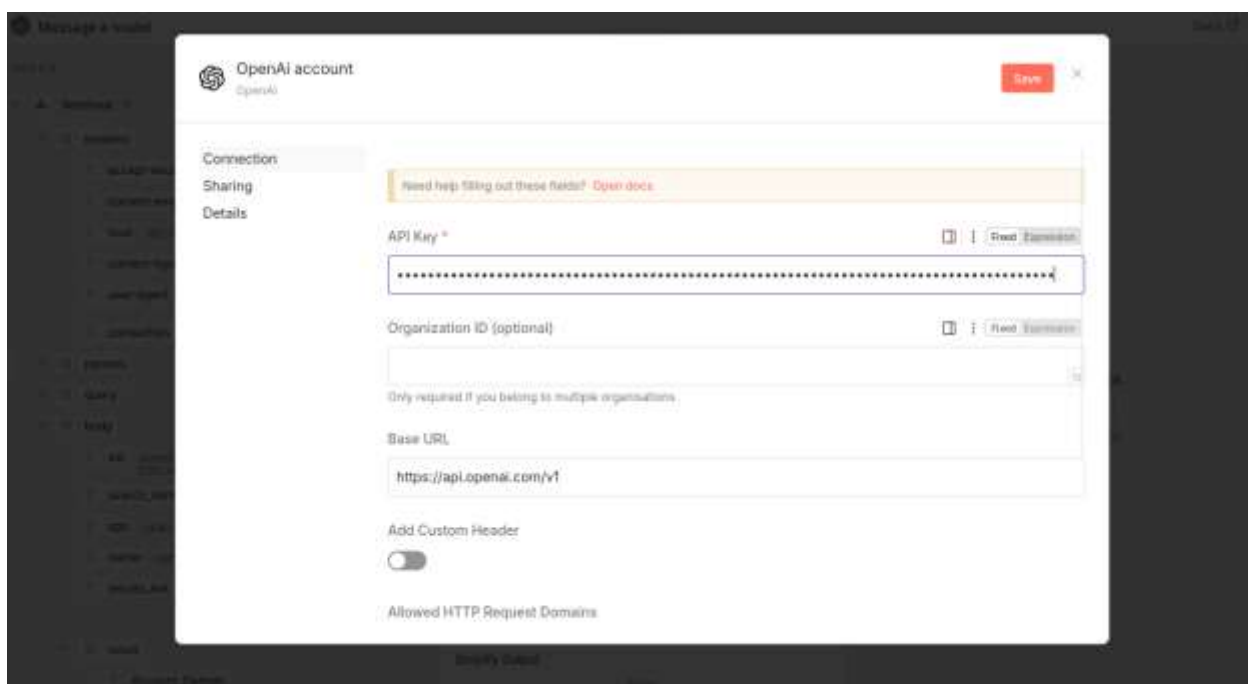


The final integration step connects the two platforms. Back in the Splunk alert settings under Trigger Actions, the "Webhook" option is selected. The unique Test URL generated by n8n is then pasted into the URL field. This configuration ensures that every time the alert fires, Splunk automatically sends the event data to the n8n automation flow for processing.

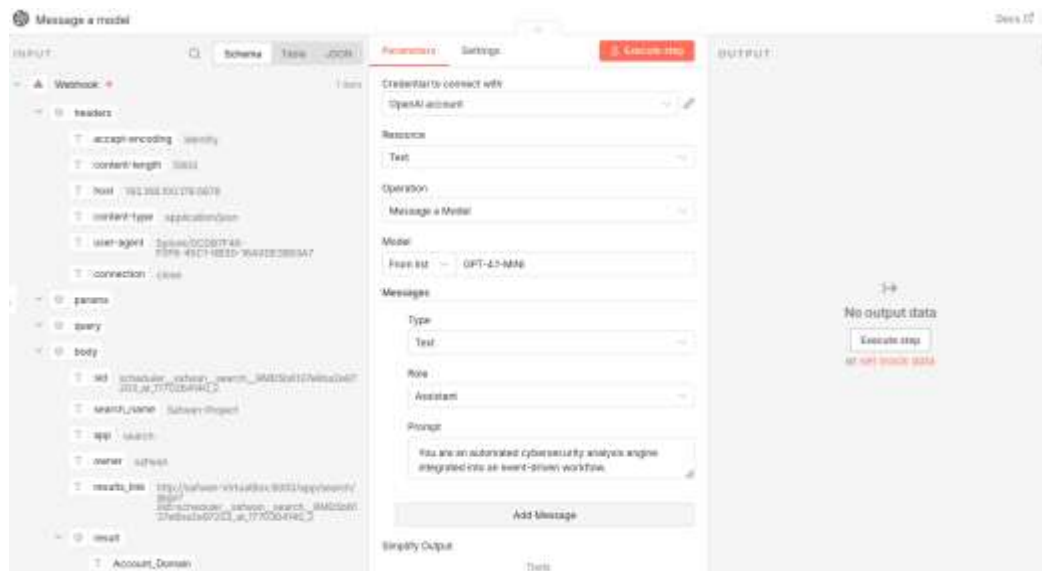




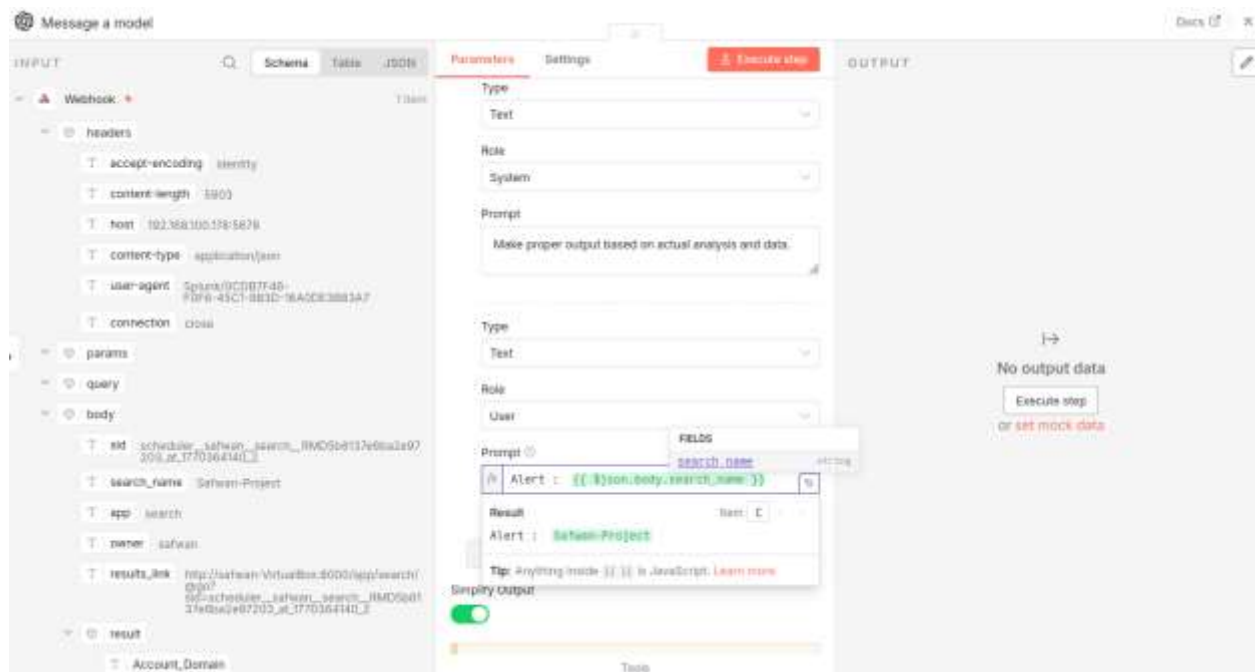
This step establishes the secure connection between the n8n automation platform and the OpenAI API. A new OpenAI account credential is created within the workflow, where the secret API Key is input and saved. This authentication allows the workflow to programmatically access GPT models for analyzing security alerts.



The OpenAI node is configured to define the AI's operational role within the SOC workflow. Under the Messages section, an "Assistant" role is created with a specific system prompt: *"You are an automated cybersecurity analysis engine integrated into an event-driven workflow."* This instruction sets the context, ensuring the model interprets subsequent data as a security analyst rather than a general-purpose chatbot.



The workflow logic is further refined by mapping incoming Splunk data to the AI's input prompt. A User role is added to the message parameters, utilizing a dynamic expression: Alert : {{ \$json.body.search_name }}. This configuration ensures that the specific title of the triggered Splunk alert (e.g., "Safwan-Project") is automatically injected into the prompt, telling the AI exactly which security event it is analyzing.



The final image demonstrates the construction of the full data payload passed to the AI model. An n8n Expression is used to stringify the entire JSON result from Splunk: `{{ JSON.stringify($json.body.result,null,2) }}`. The preview window confirms that critical telemetry—including EventCode 4625 (failed login), ComputerName (DESKTOP-6VKP16F), and timestamp data—is correctly formatted and ready to be processed by the AI for incident analysis.

The screenshot displays the n8n Expression editor with three main sections: Search previous nodes' fields, Expression, and Result.

Search previous nodes' fields: A tree view showing the structure of the data. The 'body' field is expanded, revealing 'scheduler_safwan_search_RWC36613765a2e07203_a0_070364540_2', 'search_name', 'app', 'owner', and 'results_sink'. The 'result' field is also expanded, showing a list of fields: 'Account_Domain', 'Account_Name', 'Authentication_P...', 'Caller_Computer...', 'Caller_Domain', and 'Caller_Logon_ID'.

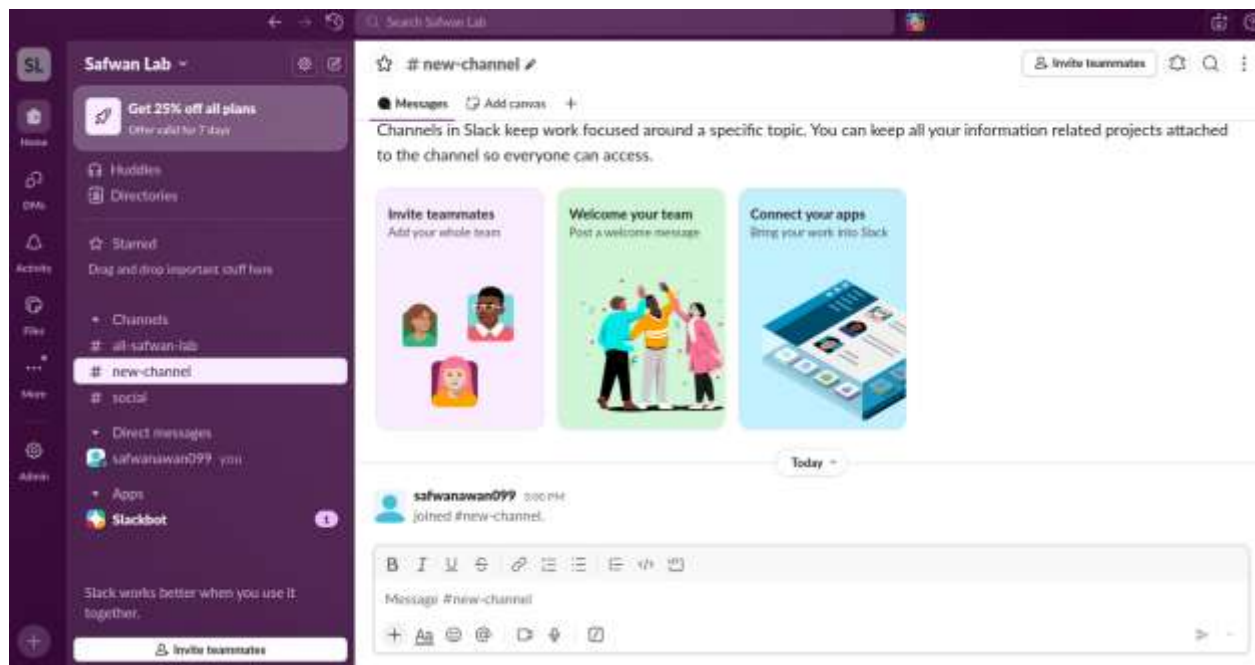
Expression: The expression `Alert : {{ $json.body.search_name }}
Alert Details : {{
JSON.stringify($json.body.result,null,2)}}}` is entered.

Result: The resulting JSON payload is displayed, showing the 'Alert' and 'Alert Details' fields. The 'Alert Details' field contains a large JSON object with various fields such as 'Account_Domain', 'Account_Name', 'Authentication_Package', 'Caller_Computer_Name', 'Caller_Domain', 'Caller_Logon_ID', 'Caller_Machine_Name', 'Caller_Process_ID', 'Caller_Process_Name', 'Caller_User_Name', 'CategoryString', 'Change_Type', 'Client_Address', 'Client_Domain', 'Client_Logon_ID', 'Client_Machine_Name', 'Client_User_Name', 'ComputerName', 'Creator_Process_Name', 'Description', 'Domain', 'Error_Code', 'EventCode', 'EventType', 'Failure_Reason', 'File_Name', 'File_Path', and 'Group_Domain'.

Configuration of Slack:

The screenshot shows the Slack sign-up page. At the top is the Slack logo. Below it is the heading "First, enter your email" followed by the text "We suggest using the email address you use at work." There is a text input field containing "name@work-email.com" and a purple "Continue" button. Below the button is a horizontal line with "OR" in the center. Underneath are two buttons: "Google" and "Apple". At the bottom, there is a paragraph of text: "By continuing, you're agreeing to our Main Services Agreement, User Terms of Service, and Slack Supplemental Terms. Additional disclosures are available in our Privacy Policy and Cookie Policy." and a link: "Already using Slack? Sign in to an existing workspace".

Click on Channels to add new channel for the smooth configuration of lab workflow.



Create a channel



Name

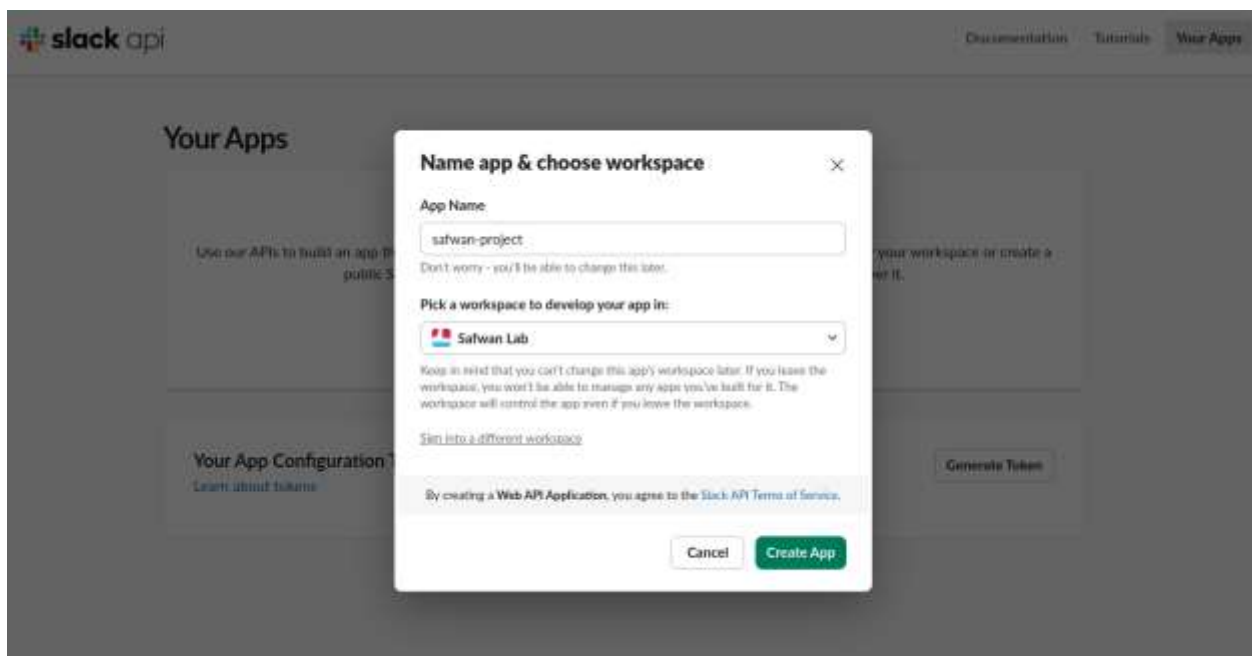
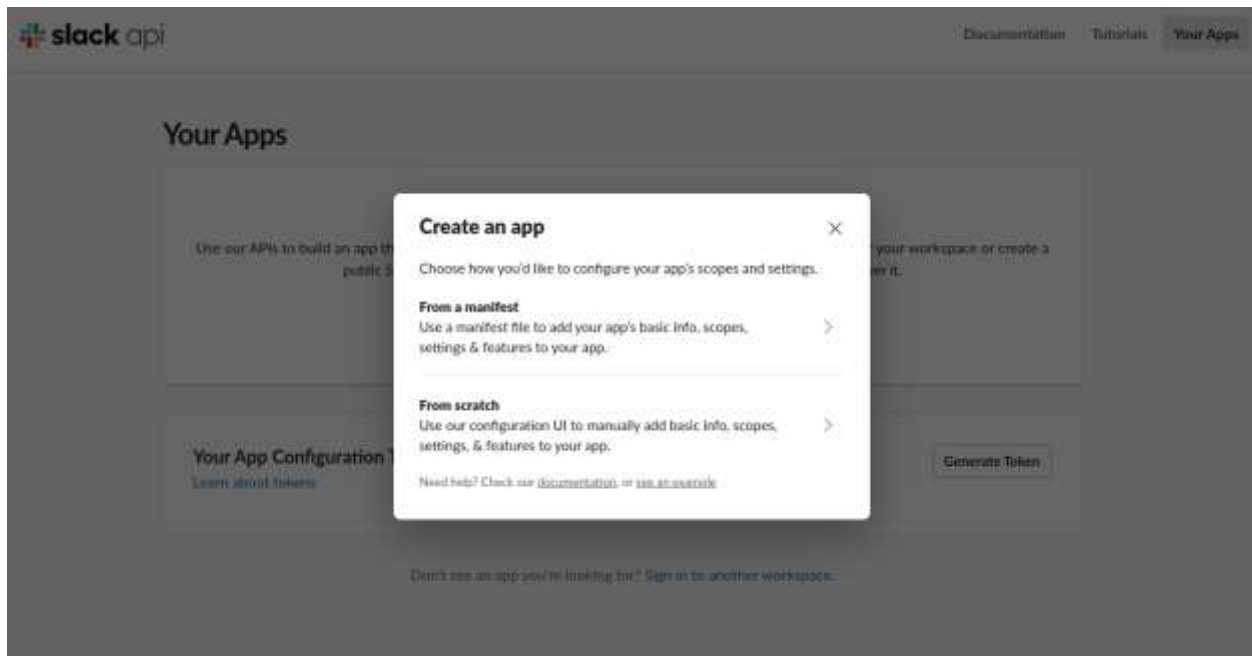
safwan-project 66

Channels are where conversations happen around a topic. Use a name that is easy to find and understand.

Step 1 of 2

Next

The Slack API dashboard by selecting **"Create New App"** from scratch. The application is given a name, such as "SIEM-Alerts," and linked to the active project workspace. This creates the bot identity that will be responsible for delivering automated notifications from the SIEM.




Go to Scopes and then add all of the below scopes in the Add and OAuth Scope section.

Scopes

A Slack app's capabilities and permissions are governed by the [scopes](#) it requests.

Bot Token Scopes














Scopes that govern what your app can access.

OAuth Scope	Description	
channels:read	View basic information about public channels in a workspace	
<div>Add an OAuth Scope</div>		

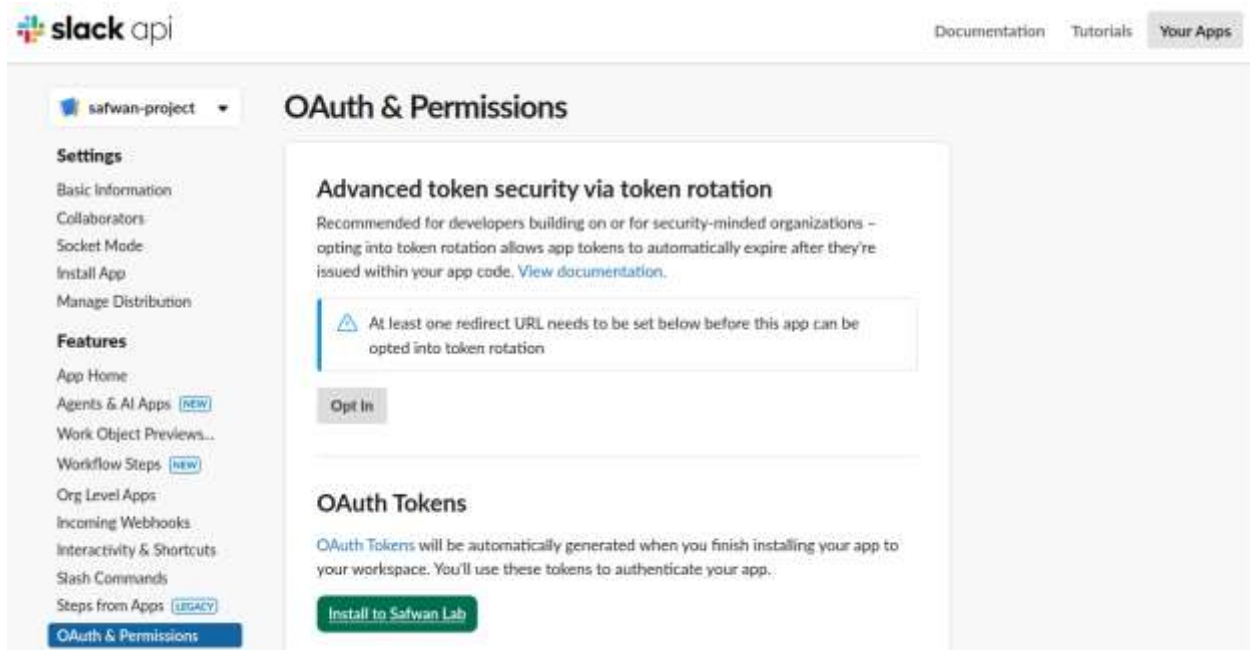
User Token Scopes

Scopes that access user data and act on behalf of users that authorize them.

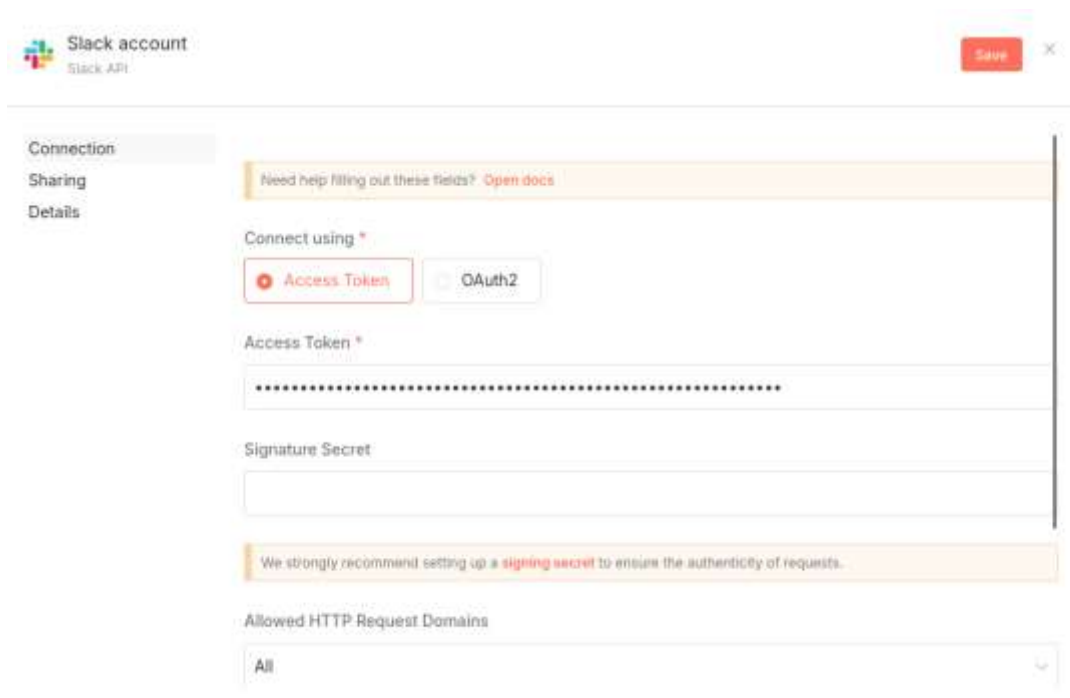
OAuth Scope	Description
You haven't added any OAuth Scopes for your User token.	
<div>Add an OAuth Scope</div>	

channels:read	View basic information about public channels in a workspace	
chat:write	Send messages as @sahwan-project	
im:read	View basic information about direct messages that "sahwan-project" has been added to	
mpim:read	View basic information about group direct messages that "sahwan-project" has been added to	
reactions:read	View emoji reactions and their associated content in channels and conversations that "sahwan-project" has been added to	
reactions:write	Add and edit emoji reactions	
usergroups:read	View user groups in a workspace	
usergroups:write	Create and manage user groups	
users:profile:read	View profile details about people in a workspace	
users:read	View people in a workspace	
files:read	View files shared in channels and conversations that "sahwan-project" has been added to	
files:write	Upload, edit, and delete files as "sahwan-project"	
groups:read	View basic information about private channels that "sahwan-project" has been added to	

Upon Saving the previous work the new instance appear. In this click on the Install to “Safwan” Lab (Your case would be change) to get the API Key.



API Key that was taken from the previous step will be putted in the Slack node of n8n.



Send a message

INPUT

Schema Table JSON

No input connected

Parameters Settings Execute stop

Credential to connect with
Slack account

Resource
Message

Operation
Send

Send Message To
Channel

Channel
From list all-safwan-lab

Message Type
Simple Text Message

Message Text
Text

Options
No properties
Add option

OUTPUT

Schema Table JSON

No output data
Execute stop
or set mock data

Send a message

INPUT

Schema Table JSON

No input connected

Parameters Settings Execute stop

Credential to connect with
Slack account

Resource
Message

Operation
Send

Send Message To
Channel

Channel
From list all-safwan-lab

Message Type
Simple Text Message

Message Text
Text

Options
No properties
Add option

OUTPUT

Schema Table JSON

time	channel	message
	C0ADT87P517	<pre>user : U0ADERNF008 type : message ts : 1700375679.212839 bot_id : B0ADB35G0JH app_id : ADACZ264XCP text : Text: "Automated with this <http://192.168.100.178:5678/workflow/ Lu9QInDwyqD15k-HL_LFo7utm_source=ml external&utm_medium=powered_by4 amp&utm_campaign=n8n-nodes- slack_slack_3ee85f5062bc770831648f3c5 8850eeef529c353890e64204c75157[n] workflow>." team : T0AERN00096 bot_profile id : B0ADB35G0JH app_id : ADACZ264XCP user_id : U0ADERNF008 name : safwan-project icons image_30 : https://a.slack-edge.com/v plugin/app/bot_30.png image_48 : https://a.slack-edge.com/v plugin/app/bot_48.png image_72 : https://a.slack-edge.com/v</pre>

This screenshot demonstrated the proper working of the complete Lab workflow. The Test shown in the slack n8n node.s


☆ # all-safwan-lab Invite teammates 🔔 🔍 ⋮

Messages Add canvas +

safwanawan099 2:58 PM Today ▾
joined #all-safwan-lab.

Only visible to you New

Slackbot LEGACY 3:27 PM
Invites have been sent! Kick back and relax while you wait for them to join.
(198 kB) ▾



safwan-project APP 3:27 PM
was added to #all-safwan-lab by safwanawan099.

safwan-project APP 3:27 PM
Test
[Automated with this n8n workflow](#)

