Automating Security with a Chat Interface

Owner Jonas De Weerdt

Company Information ActWise NV
Veldkant 33 A
2550 Kontich
BE 0779 808 239

1. Preface

In this document I will be explaining what my internship assignment is and what I created here at Actwise. With this document I will try to prove to you that I am ready to be sent onto the job market. This document will explain everything about the flows that I have created but also my thought process on certain decisions I have made. How they work in a detailed technical matter and how to use them in a more user guide format.

Copyright 2024 Jonas De Weerdt – all rights reserved.

Reproduction or disclosure to third parties of this document, or any part thereof, is only permitted with the prior and express written permission of Jonas De Weerdt. Any person, other than the authorized holder, obtaining possession of this document, by finding or otherwise, should send it, together with his name and address, in a sealed envelope to the address given above.

Table of Contents

1. Preface 2
3. Intro 5
4. Thought process 6
5. Local PAM environment 6
    5.1. CyberArk Vault Server 7
        5.1.1. What is a CyberArk Vault? 7
        5.1.2. Key Functions of a CyberArk Vault: 7
        5.1.3. Creating and Managing a Vault: 7
        5.1.4. Accessing the Vault: 7
    5.2. CyberArk Component Server 7
        5.2.1. PVWA (Privileged Web Access): 7
        5.2.2. PSM (Privileged Session Manager): 8
        5.2.3. CPM (Central Policy Manager): 8
    5.3. Domain Controller Server 8
        5.3.1. What is a Domain Controller? 8
        5.3.2. Key Functions of a Domain Controller: 9
        5.3.3. CyberArk Integration with Domain Controllers: 9
    5.4. Privileged Threat Analytics Server 9
        5.4.1. What is PTA? 9
        5.4.2. How Does PTA Work? 9
        5.4.3. Key Functions of PTA: 9
        5.4.4. Installation and Configuration: 10
        5.4.5. Benefits of PTA: 10
    5.5. Slack Url Verification 11
        5.5.1. Unverified Source: 11
        5.5.2. Auth-Challenge: 12
        5.5.3. Event: 12
    5.6. Discord Bot 13
6. Flows user guide 14
    6.1. Temporary Access 15
    6.2. Panic Button 16
    6.3. Confirming Request 17
    6.4. Password Recovery 18
7. Flows technical explanation 20
    7.1. Temporary Access 20
        7.1.1. Python code 20

7.2. Panic Button 21

7.2.1. Python code 21

7.3. Confirming Request 22

7.3.1. Python code 22

7.4. Password Recovery 23

7.4.1. Python Code 23

8. Sources 26

3. Intro

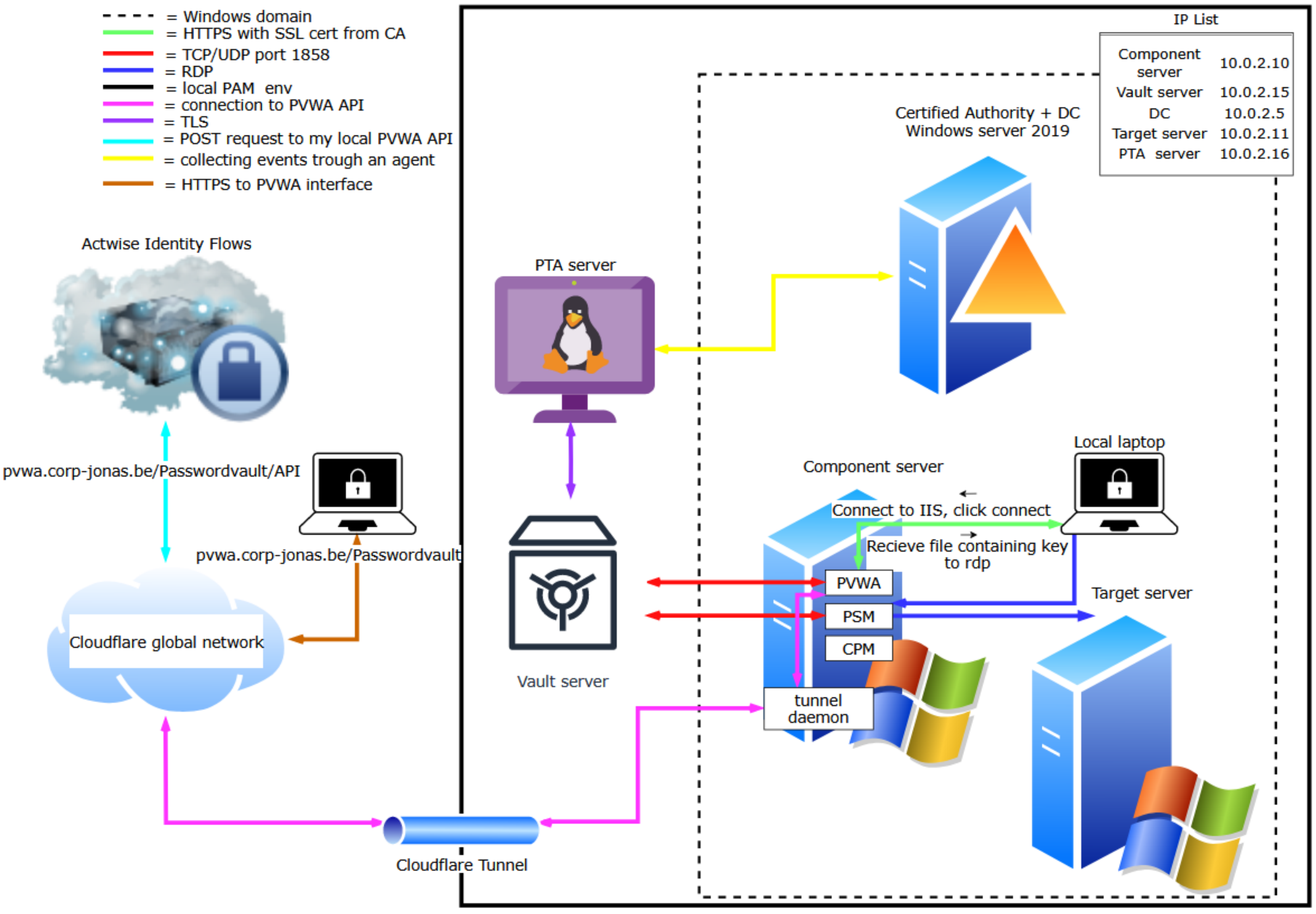
My internship at Actwise was great because I got to experience a different side of security that we did not really cover in school. What I am talking about is PAM (Privileged Access Management), which is a security strategy that focuses on managing and controlling access to privileged accounts and sensitive information within an organization. But my internship assignment wasn't just PAM. I had to learn how to use the service Identity Flows. CyberArk provides both of those products. CyberArk is a global leader in Identity Security. CyberArk helps organizations secure access to critical business data and infrastructure, protect a distributed workforce, and accelerate business in the cloud. My internship assignment was to use CyberArk Identity Flows in conjunction with a chat interface of my choosing to streamline certain security actions.

4. Thought process

In the beginning it was all about figuring out what exactly I was meant to do. So I asked a lot of questions to my colleagues and my internship mentor Jelle Verreth. The first realization I did was to create the entire local PAM environment. This will be explained completely later. After this I could finally start to create some Flows. I had decided to use Slack as my chat interface because it was already integrated with Flows. I will explain how that should have worked later. After encountering issues with the Slack integration, I had contact with Patrik Horemans a CyberArk employee here in Belgium. He told me that our system was outdated meaning that certain features did not work as intended or weren't even implemented yet. I also had contact with developers from Slack to ask if there was any way to change the response time-out time. They told me that it was 3 seconds and unchangeable. After this I made the decision to step away from Slack and start looking for other chat interfaces. I looked at teams and discord. Ultimately, I chose discord because it gave me the possibility to create a bot coded in python, meaning I had a lot more freedom in how I wanted my logic to work since I am not bound to predetermined functions within Flows.

5. Local PAM environment

I will explain every part of this environment in detail.



5.1. CyberArk Vault Server

5.1.1. WHAT IS A CYBERARK VAULT?

- A CyberArk Vault is a critical component of CyberArk's Privileged Access Management (PAM) solution. It serves as a secure repository for managing and safeguarding privileged credentials, secrets, and other sensitive information.
- Think of it as a digital safe where you can securely store and manage access to critical assets such as administrative passwords, SSH keys, API tokens, and more.

5.1.2. KEY FUNCTIONS OF A CYBERARK VAULT:

- **Credential Storage**: The Vault securely stores privileged credentials (such as passwords, SSH keys, and certificates) in encrypted form.
- **Access Control**: It allows you to define fine-grained access permissions for users and applications. Only authorized individuals can retrieve or manage stored credentials.
- **Secret Rotation**: The Vault supports automatic rotation of secrets (e.g., changing passwords periodically) to enhance security.

5.1.3. CREATING AND MANAGING A VAULT:

- To create a Vault, administrators use the PrivateArk Administrative Client. Once created, they can create Safes (logical containers) within the Vault.
- Safes allow you to organize and compartmentalize credentials based on business needs.
- Vault administrators allocate Safes and manage access permissions for users.

5.1.4. ACCESSING THE VAULT:

- Users access the Vault through the PrivateArk Client installed on their workstations.
- They establish a connection to the Vault, which allows them to retrieve credentials from Safes.
- The Vault connection properties include authentication methods (e.g., username/password, certificate-based).

5.1.5. HOW IT IS CONFIGURED IN MY ENVIRONMENT

- My vault is configured to use TLS to send secured data to my PTA server. It also has an SSL certificate from my Certificate Authority. This removed the self-signed log warning.

5.2. CyberArk Component Server

5.2.1. PVWA (PRIVILEGED WEB ACCESS):

- The PVWA is a web-based interface that allows authorized users to interact with the CyberArk Vault securely via a web browser.
- **Functionality**:
  - **User Authentication**: Users log in to the PVWA using their credentials (username and password).
  - **Access to Secrets**: Users can retrieve privileged credentials (such as passwords, SSH keys) stored in the Vault.
  - **Request Access**: Users can request access to specific secrets through the PVWA.

    - **Session Management**: PVWA enables users to establish sessions (e.g., RDP, SSH) directly from the interface using retrieved credentials.
    - **API Access**: The PVWA has a REST API that can control everything within the PAM environment.
- **My environment**: To make this API reachable from the internet I use Cloudflare tunnels. The Tunnel daemon that is installed on the component server creates an encrypted tunnel between the PVWA webserver and Cloudflare's nearest data center, all without opening any public inbound ports. This is required because CyberArk Identity Flows requires a connection to my local PAM environment.

5.2.2. PSM (PRIVILEGED SESSION MANAGER):

- PSM is responsible for managing privileged sessions (e.g., SSH, RDP) to target systems.
- **Functionality**:
    - **Session Isolation**: PSM isolates privileged sessions from the user's workstation, reducing the risk of credential exposure.
    - **Session Recording**: PSM records all session activities for auditing and compliance.
    - **Credential Injection**: PSM injects credentials into the session transparently.
    - **Access Control**: PSM enforces access controls and session policies.
- **My environment**:
    - **HTML5Gateway**: I have configured the html5Gateway as a side project. This service makes it possible to open the RDP sessions in your browser so you do not have to download the constant RDP files.

5.2.3. CPM (CENTRAL POLICY MANAGER):

- CPM manages the automatic rotation of privileged account passwords.
- **Functionality**:
    - **Password Rotation**: CPM periodically changes passwords for privileged accounts (e.g., service accounts, database accounts).
    - **Credential Management**: CPM interacts with target systems to update credentials.
    - **Policy Enforcement**: CPM enforces password rotation policies defined by administrators.
    - **Integration with PVWA and PSM**: CPM coordinates password changes with PVWA and PSM.
- **Security Considerations**: CPM ensures that passwords are regularly rotated to reduce the risk of unauthorized access.

5.3. Domain Controller Server

5.3.1. WHAT IS A DOMAIN CONTROLLER?

A domain controller is a critical component in Microsoft's Active Directory (AD) infrastructure. It serves as the central authority for managing user authentication, access control, and other directory services.

Think of it as the heart of an organization's network, responsible for maintaining user accounts, security policies, and domain-wide settings.

5.3.2. KEY FUNCTIONS OF A DOMAIN CONTROLLER:

User Authentication: When a user logs in to their workstation or any other network resource, the domain controller verifies their credentials (username and password).

Access Control: It manages permissions and access rights for users and groups. For example, it determines who can access specific files, folders, or printers.

Directory Services: The domain controller maintains a directory database containing information about users, computers, groups, and organizational units (OUs).

Security Policies: It enforces security policies (such as password complexity rules) across the network.

DNS Services: Domain controllers also function as DNS servers, resolving domain names to IP addresses.

5.3.3. CYBERARK INTEGRATION WITH DOMAIN CONTROLLERS:

- CyberArk integrates with domain controllers to enhance privileged access management (PAM) and secure privileged accounts.
- **Active Directory Integration**:
    - CyberArk's Remote Access solution can integrate with your organization's Active Directory.
    - This integration allows users to authenticate using their AD credentials when accessing privileged accounts through CyberArk.
    - Configuration involves specifying domain context, domain controllers, LDAPS settings, and more.

5.4. Privileged Threat Analytics Server

5.4.1. WHAT IS PTA?

- PTA, short for Privileged Threat Analytics, is a crucial feature within CyberArk's suite of tools.
- Its primary purpose is to detect high-risk accounts and activities related to privileged access.
- PTA works by continuously monitoring privileged sessions and identifying any suspicious actions that might indicate a cyber-attack.

5.4.2. HOW DOES PTA WORK?

PTA operates as follows:

- **Session Monitoring**: It actively observes privileged sessions (such as administrative logins, SSH sessions, or database connections).
- **Behaviour Analysis**: PTA analyses user behaviour during these sessions.
- **Anomaly Detection**: It looks for deviations from normal behaviour patterns.
- **Alerts and Reporting**: When it detects unusual or risky actions, PTA generates alerts or reports for security teams to investigate further.

5.4.3. KEY FUNCTIONS OF PTA:

- **User Behaviour Profiling**: PTA builds profiles of typical user behaviour based on historical data.

- **Risk Scoring**: It assigns risk scores to users and sessions, highlighting potential threats.
- **Privileged Session Monitoring**: PTA watches over privileged sessions in real-time.
- **Threat Detection**: It identifies suspicious activities, such as unauthorized access attempts, unusual commands, or lateral movement.
- **Insider Threat Detection:** PTA helps detect insider threats by analysing privileged user behaviour.
- **Integration with SIEM**: PTA can integrate with Security Information and Event Management (SIEM) systems for centralized monitoring and correlation.

5.4.4. INSTALLATION AND CONFIGURATION:

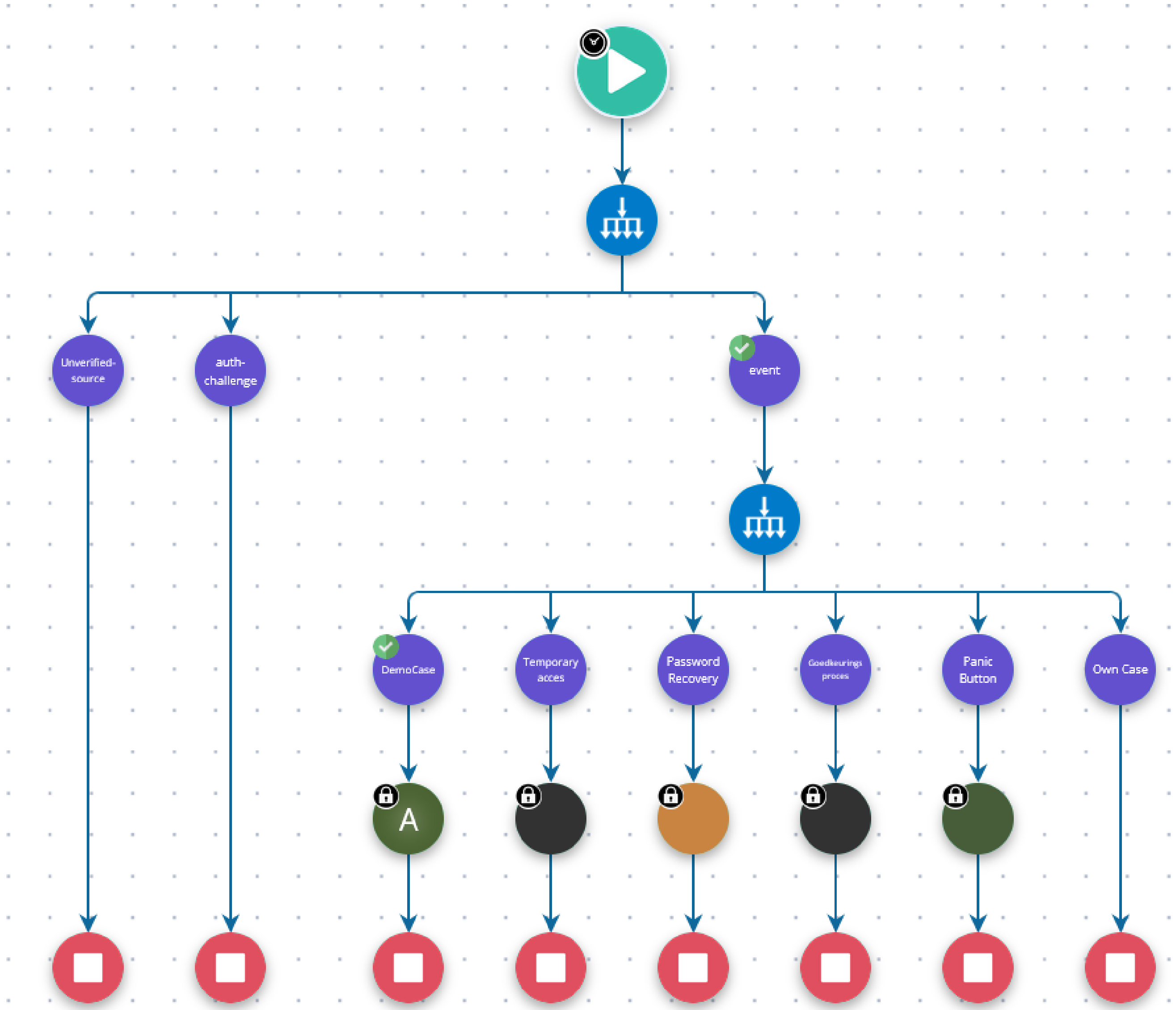
- To set up PTA, you'll need to:
  - **Install the PTA Server**: This involves deploying the PTA software on a compatible operating system (such as Red Hat Enterprise Linux).
  - **Configure Trusted Connections:** Establish connections between the PTA server and the CyberArk Vault.
  - **Manage Certificates**: Import SSL certificates for secure communication.

5.4.5. BENEFITS OF PTA:

- **Early Threat Detection**: PTA helps identify potential security incidents before they escalate.
- **Reduced Attack Surface**: By monitoring privileged sessions, it minimizes the risk of unauthorized access.
- **Compliance and Auditing**: PTA assists with compliance requirements by tracking privileged activity.
- **Insight into Insider Threats**: It sheds light on risky behaviour by privileged users.

5.5. Slack Events API connector

The Flow depicted here illustrates the steps necessary to receive events from the Slack Events API, the events get processed by the branch and then other Flows will be started with API calls to have a clear overview. This is how it was supposed to work but due to the continued problems that I was having with the slack integration I had to stop developing this part of the project and switch over to writing a Discord bot.



5.5.1. UNVERIFIED SOURCE

This path is followed if the token in the request is not equal to the token from my Slack App.

5.5.2. AUTH-CHALLENGE

Upon encountering a new endpoint, the system initiates an authentication challenge to verify that it is a working endpoint.

5.5.3. EVENT

After successful authentication, the slack Events API will send pre-specified events, and these will be handled by the branch to select what process will be started.

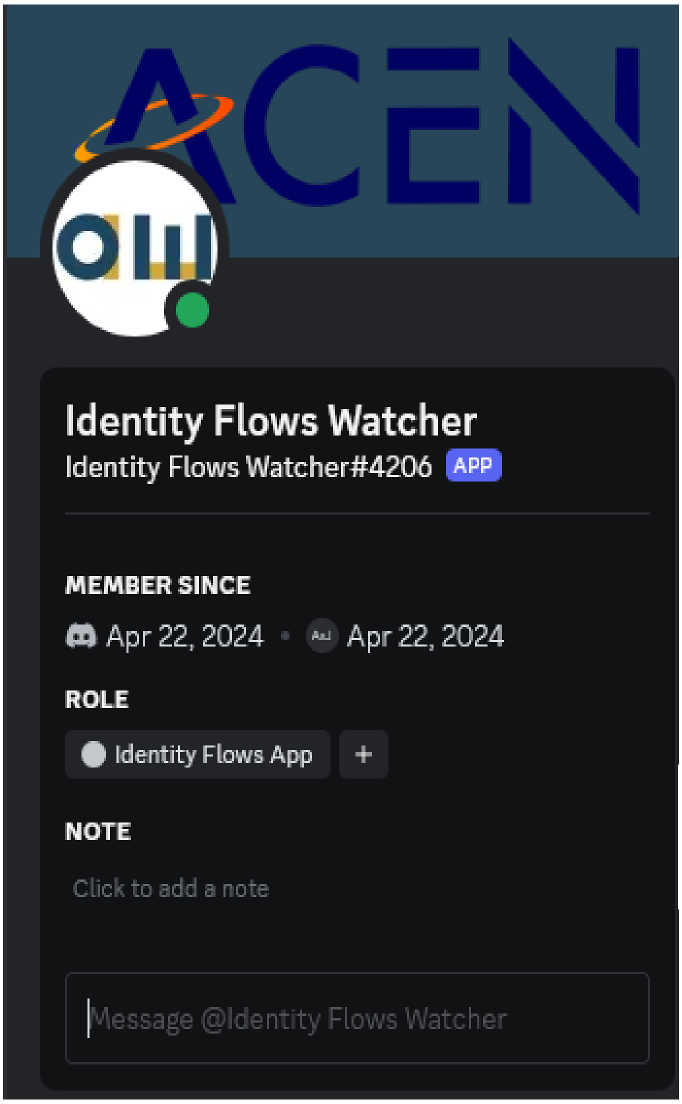
- Temporary access: This path will be selected when the message contains some keywords like: ‘Temp acc’, ’Temporary access’, ’Temporary account’, etc…
- Password Recovery: This path will be selected when these keywords are present: ‘Password recovery’, ’Wachtwoordherstel’, ’Password recov’, etc…
- Etc.

5.5.4. PROBLEMS

The slack integration came with many challenges which in the end made it impossible to do exactly what I wanted/needed. For example:

- Response time-out because of outdated Identity Flows version
- Custom-responses not working for the same reason
- The Events API expects a response within 3 seconds which was too short to do any logic with.
- Events API constantly sending requests even though I returned a “no-retry” header

5.6. Discord Bot



Identity Flows Watcher

Identity Flows Watcher#4206 APP

MEMBER SINCE

Apr 22, 2024 • Apr 22, 2024

ROLE

Identity Flows App +

NOTE

Click to add a note

Message @Identity Flows Watcher

This Discord bot was written in python and created to function as a constant watcher to be able to react to any message that gets send in the entire server. I changed to this instead of slack because slack could not work with CyberArk Flows before the update. After the update it did work but at this point, I already had invested a lot of time in the Discord bot which is why I decided to continue working on that instead of starting all over again on the slack workflows. I did create one use case using Slack to show that it is possible on both chat interfaces. To use this bot, you need to run the python code on a system that has a connection to the internet.

It also needs to be added to a guild or server so it can function and has the right permissions. This can be done via clicking the following link and allowing it access to a server that you own.

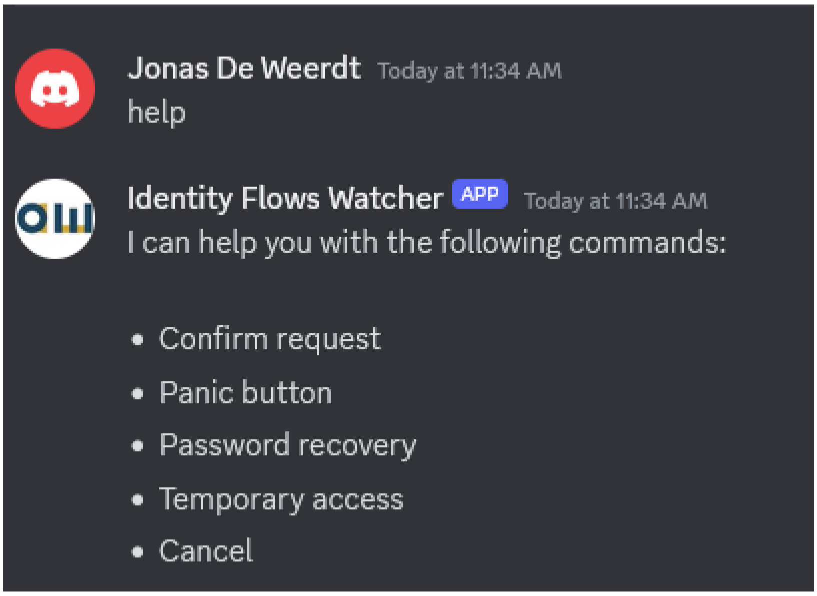
https://discord.com/oauth2/authorize?client_id=1231925449399992351&permissions=76864&scope=bot

I will show the python code later in this document. The discord bot send the data received from user interaction to the Flows using a webhook API call.

6. Flows user guide

This will be where I will list and explain the flows in a user manual way.

Every flow will be started with commands in discord because of this I created a function that lets me type help in the chat to receive a list of the possible commands. When you have selected a command, you can type it in the chat and then the flow will be started. That part I will explain for every single flow separately.

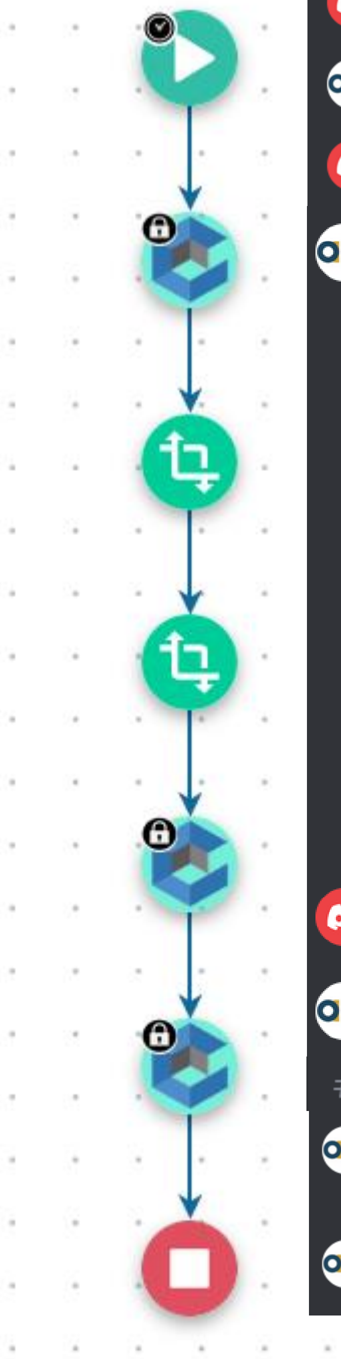


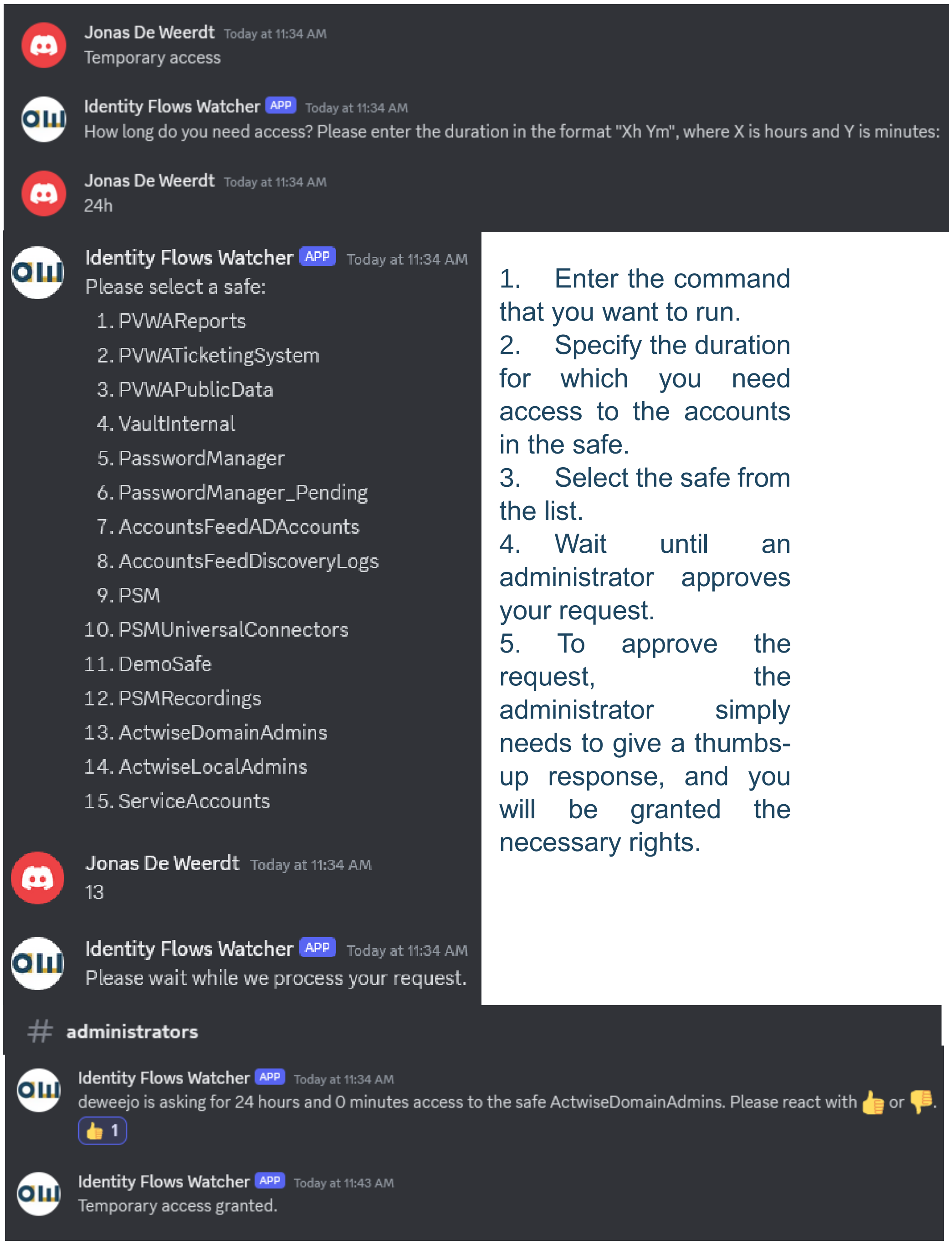
Jonas De Weerdt Today at 11:34 AM
help

Identity Flows Watcher APP Today at 11:34 AM
I can help you with the following commands:

- Confirm request
- Panic button
- Password recovery
- Temporary access
- Cancel

6.1. Temporary Access





Jonas De Weerdt Today at 11:34 AM
Temporary access

Identity Flows Watcher APP Today at 11:34 AM
How long do you need access? Please enter the duration in the format "Xh Ym", where X is hours and Y is minutes:

Jonas De Weerdt Today at 11:34 AM
24h

Identity Flows Watcher APP Today at 11:34 AM
Please select a safe:

1. PVWAReports
2. PVWATicketingSystem
3. PVWAPublicData
4. VaultInternal
5. PasswordManager
6. PasswordManager_Pending
7. AccountsFeedADAccounts
8. AccountsFeedDiscoveryLogs
9. PSM
10. PSMUniversalConnectors
11. DemoSafe
12. PSMRecordings
13. ActwiseDomainAdmins
14. ActwiseLocalAdmins
15. ServiceAccounts

Jonas De Weerdt Today at 11:34 AM
13

Identity Flows Watcher APP Today at 11:34 AM
Please wait while we process your request.

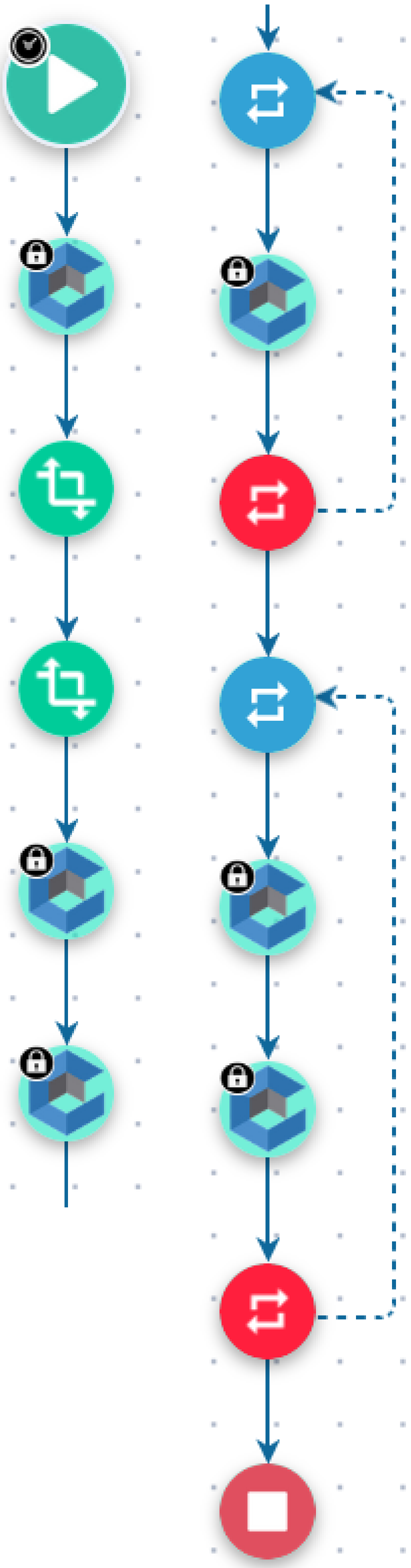
# administrators

Identity Flows Watcher APP Today at 11:34 AM
deweejo is asking for 24 hours and 0 minutes access to the safe ActwiseDomainAdmins. Please react with 👍 or 👎.
👍 1

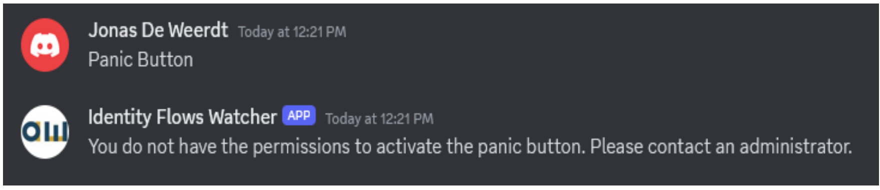
Identity Flows Watcher APP Today at 11:43 AM
Temporary access granted.

1. Enter the command that you want to run.
2. Specify the duration for which you need access to the accounts in the safe.
3. Select the safe from the list.
4. Wait until an administrator approves your request.
5. To approve the request, the administrator simply needs to give a thumbs-up response, and you will be granted the necessary rights.

6.2. Panic Button



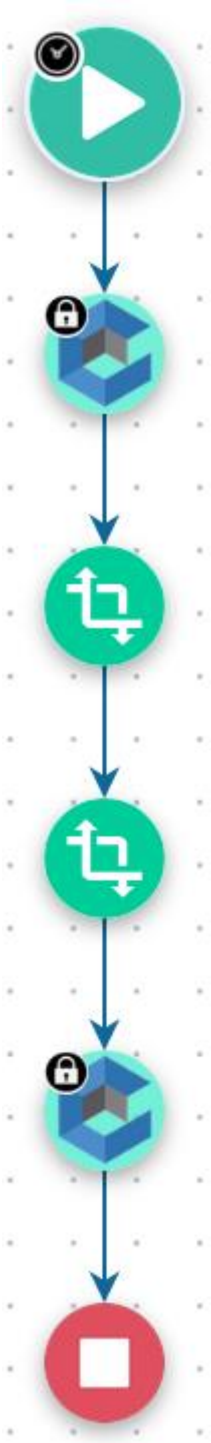
This Flow primarily utilizes CyberArk flows for its actions. I will provide a more detailed technical explanation further in the document. The panic button can only be triggered in the administrators' channel. If it's triggered elsewhere, you will receive the following response:



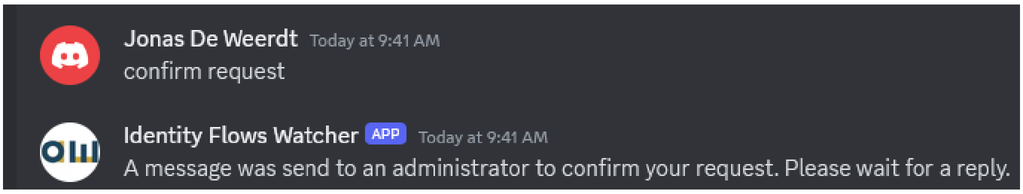
Jonas De Weerdt Today at 12:21 PM
Panic Button

Identity Flows Watcher APP Today at 12:21 PM
You do not have the permissions to activate the panic button. Please contact an administrator.

6.3. Confirming Request



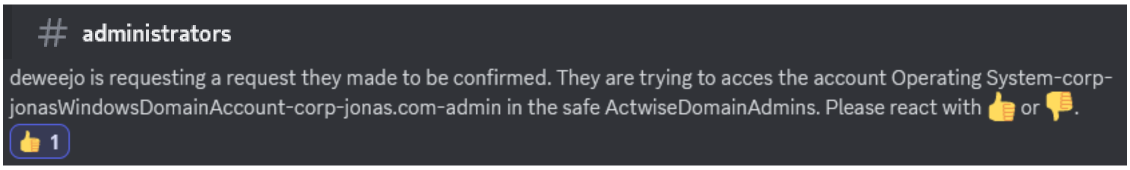
First, I type in the command I want to run. And the bot searches for requests using the local PAM API.



Jonas De Weerdt Today at 9:41 AM
confirm request

Identity Flows Watcher APP Today at 9:41 AM
A message was send to an administrator to confirm your request. Please wait for a reply.

If it finds any requests a message will be send to the Administrators to confirm the request.

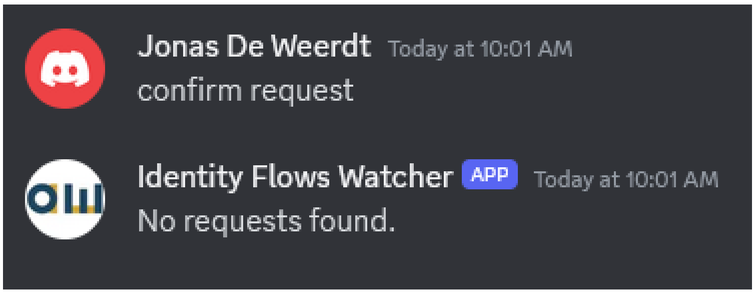


# administrators

deweejo is requesting a request they made to be confirmed. They are trying to acces the account Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-admin in the safe ActwiseDomainAdmins. Please react with 👍 or 👎.

👍 1

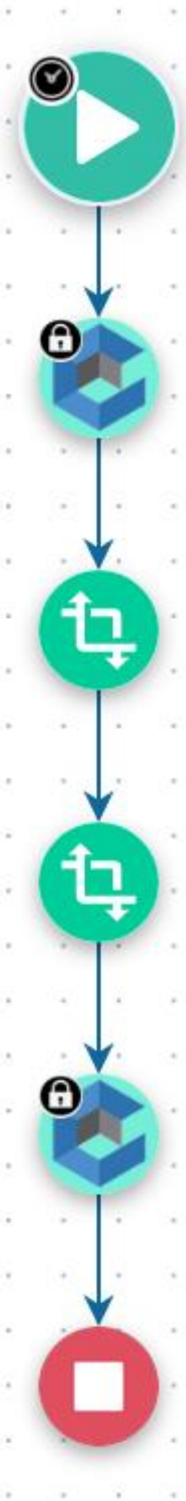
If there are no requests found the bot will respond with this.



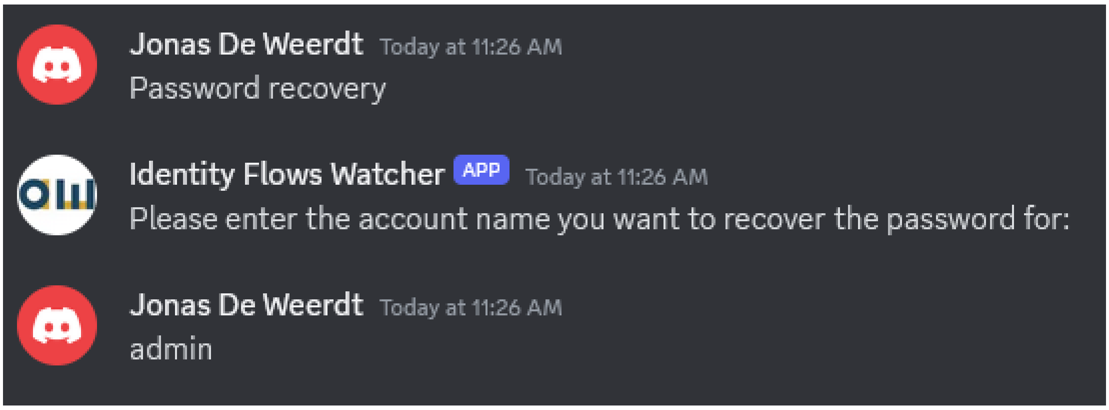
Jonas De Weerdt Today at 10:01 AM
confirm request

Identity Flows Watcher APP Today at 10:01 AM
No requests found.

6.4. Password Recovery



First, I type in the command I want to run. Then the bot asks me what account I want to recover the password for.

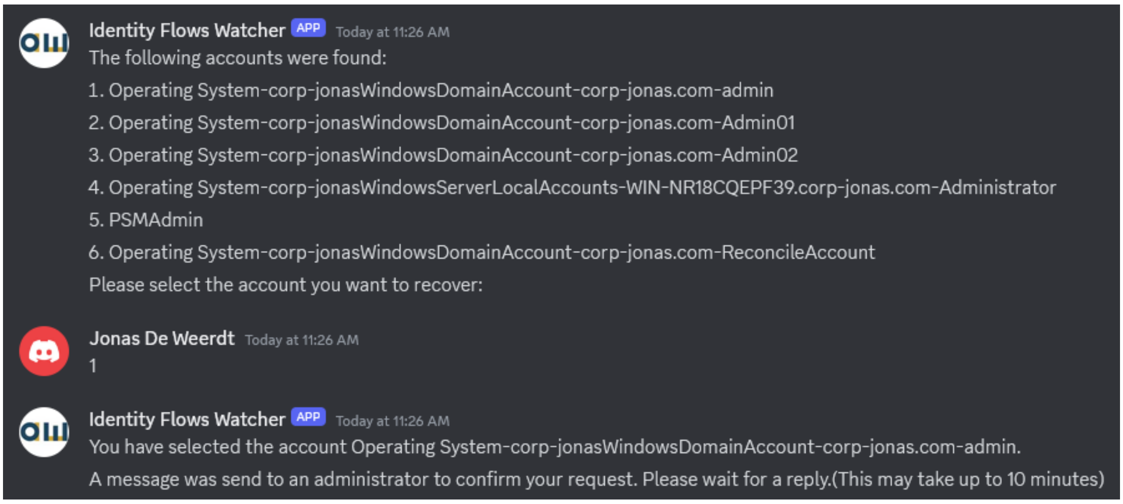


Jonas De Weerdt Today at 11:26 AM
Password recovery

Identity Flows Watcher APP Today at 11:26 AM
Please enter the account name you want to recover the password for:

Jonas De Weerdt Today at 11:26 AM
admin

The bot then searches for all accounts containing that name and displays them as a list. After choosing the desired account the bot will ask the Administrators if the request is authorized.

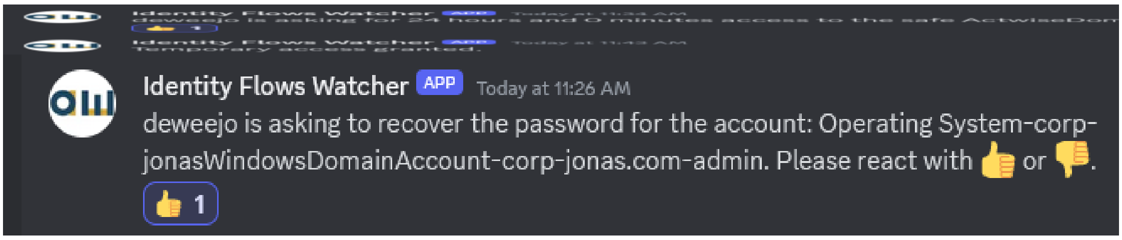


Identity Flows Watcher APP Today at 11:26 AM
The following accounts were found:
1. Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-admin
2. Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-Admin01
3. Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-Admin02
4. Operating System-corp-jonasWindowsServerLocalAccounts-WIN-NR18CQEPF39.corp-jonas.com-Administrator
5. PSMAdmin
6. Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-ReconcileAccount
Please select the account you want to recover:

Jonas De Weerdt Today at 11:26 AM
1

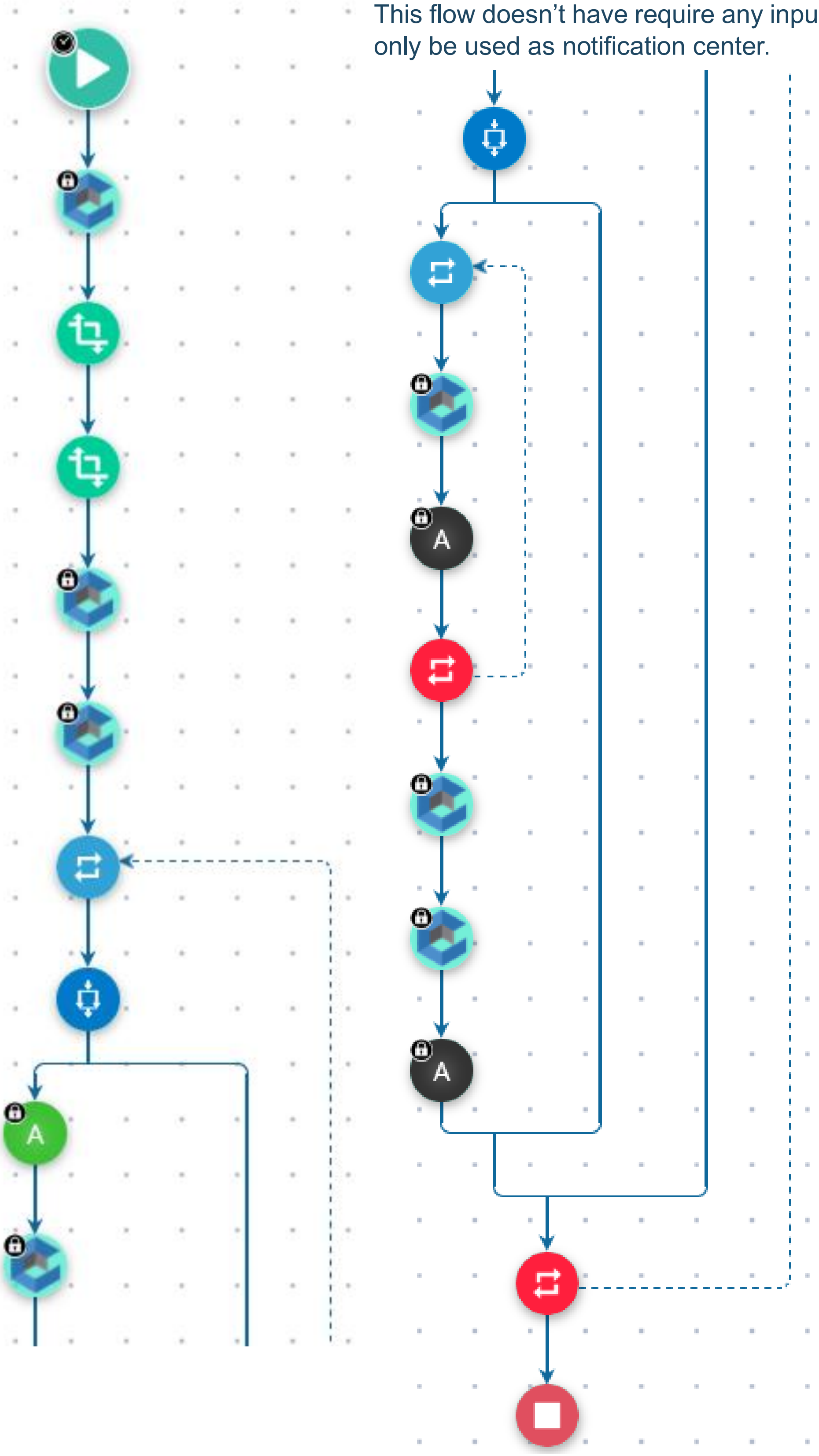
Identity Flows Watcher APP Today at 11:26 AM
You have selected the account Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-admin.
A message was send to an administrator to confirm your request. Please wait for a reply.(This may take up to 10 minutes)

After an Administrator has responded to the message the account gets recovered.



Identity Flows Watcher APP Today at 11:26 AM
deweejo is asking to recover the password for the account: Operating System-corp-jonasWindowsDomainAccount-corp-jonas.com-admin. Please react with 👍 or 👎.
👍 1

6.5. Suspicious Activity Detection

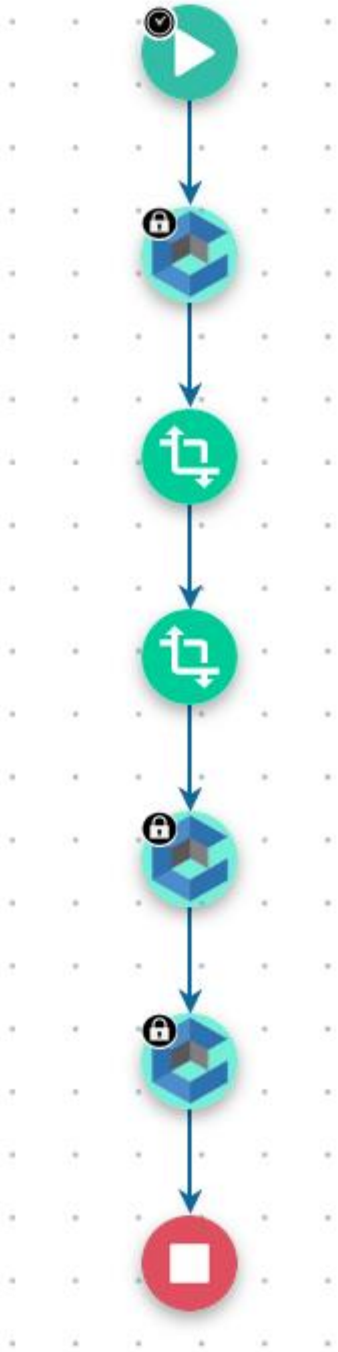


This flow doesn't have require any input from discord it will only be used as notification center.

7. Flows technical explanation.

This is the part where I will explain in detail what every component does.

7.1. Temporary Access



All the logic for this Flow is performed in Python.

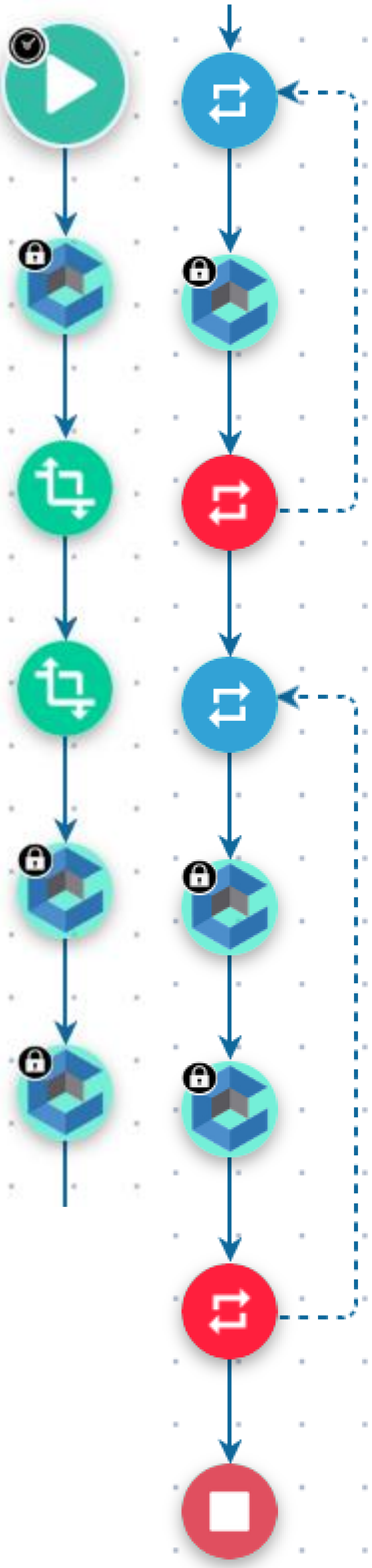
1. **Logon Call**: This initial API call is used to authenticate and obtain a session token. The session token allows subsequent API calls to be made on behalf of the authenticated user.
2. **Response Transformation**: After making the logon call, the server responds with a session token. This token has parentheses around it which had to be removed to use the token as an authorization method.
3. **Safe Member deleting**: The safe member needs to be removed so the user is no longer a member of the safe. I need to do this to be able to change how long it has access to the contents of the safe.
4. **Adding a user as a safe member**: After removing it I can add it again with a changed membership expiration date.

All the data that is used in these API calls comes from communication with the discord bot. The bot also checks by asking an administrator if the user has the rights to perform this action.

7.1.1. PYTHON CODE

Microsoft Word
Document

7.2. Panic Button



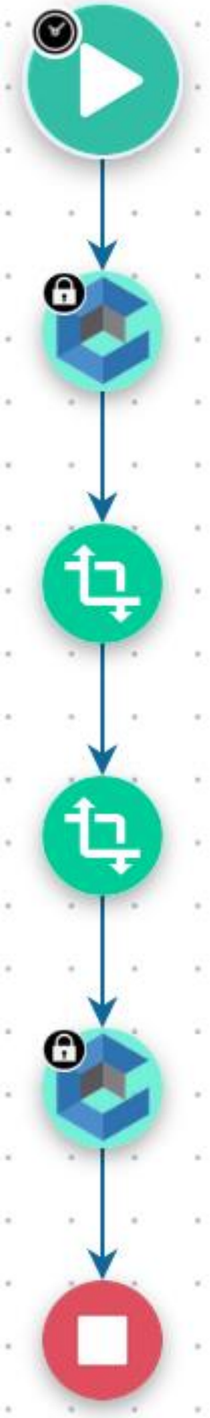
1. **Logon sequence**: This is the exact same in all the flows so I will not repeat it every time.
2. **Get live sessions and accounts**: These are the next 2 API calls and will as the name suggest get all existing accounts and sessions that are currently opened.
3. **Loop over all live sessions**: A loop is created to terminate all the live sessions to make sure there is no rogue connections. To do this I had to add my API Logon user to a group called session terminators.
4. **Loop over privileged accounts**: A loop to check-in (When an account is used it gets checked in so that no other user can use it at the same time. I configured this for the domain admins.) all accounts in the domain admins safe and change the password immediately.

All the data that is used in these API calls comes from communication with the discord bot.

7.2.1. PYTHON CODE

Microsoft Word
Document

7.3. Confirming Request



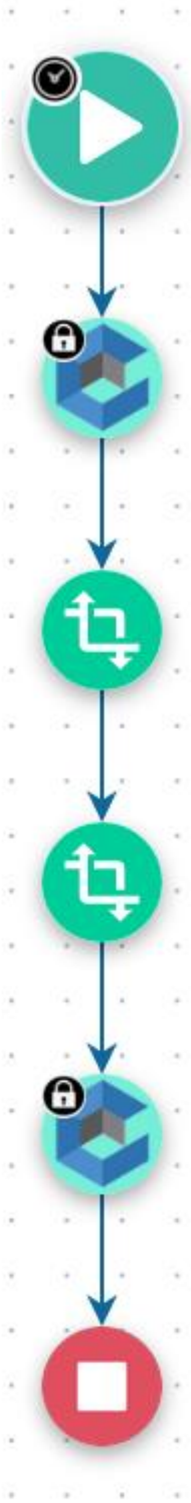
All the logic for this Flow is performed in python.

1. **Logon sequence**: All actions performed to authorize the API Calls.
2. **Confirm request**: This is an API call that just confirms the incoming request using data received from discord.

7.3.1. PYTHON CODE

Microsoft Word
Document

7.4. Password Recovery



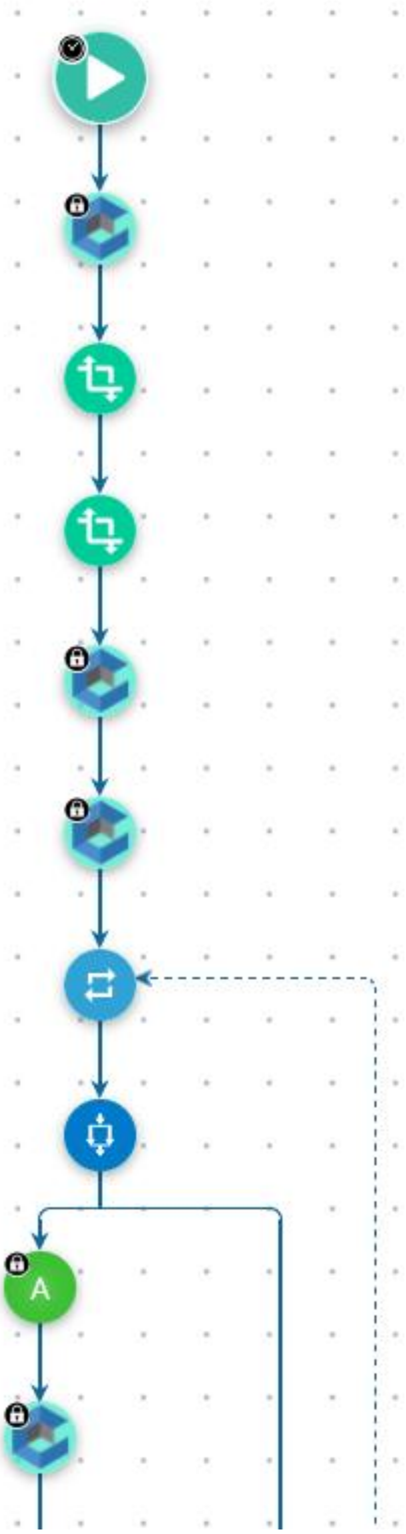
All the logic for this Flow is performed in Python.

1. **Logon sequence**: All actions performed to authorize the API Calls.
2. **Reconcile Password**: This API call triggers the CPM to immediately reconcile the password of the account that is associated with the account ID. The account ID is received from communication with the discord bot.

7.4.1. PYTHON CODE

Microsoft Word
Document

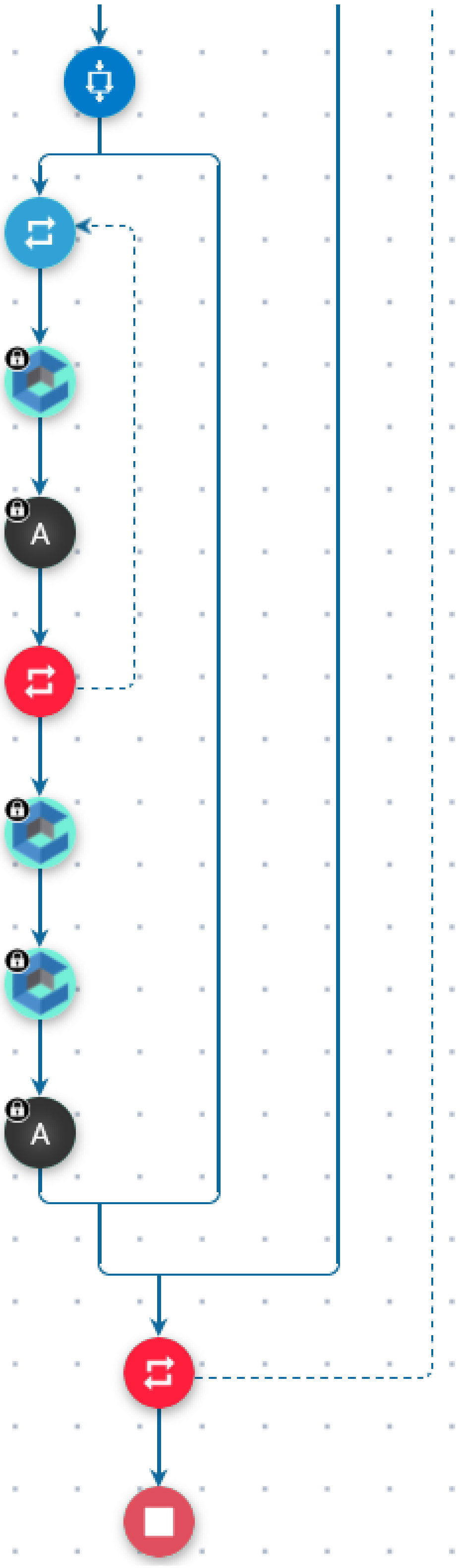
7.5. Suspicious Activity Detection



No logic for this Flow is performed in Python.

1. **Logon sequence**: All actions performed to authorize the API Calls.
2. **Get security events + live sessions**: The next 2 API calls are to get all events and live sessions to use later in the flow.
3. **Loop security events**: Loop over the security events and check if there are any OPEN alerts. If this is true send a message to Discord saying that a Security event was created and change the event to CLOSED.

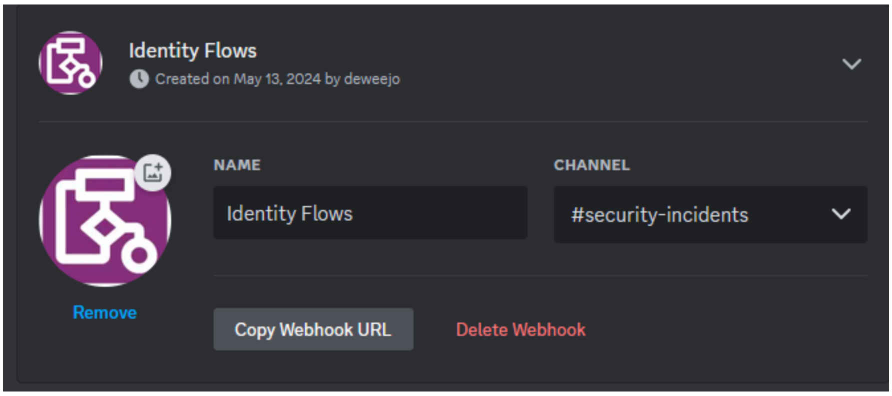
Scroll further down to find the continuation of the guide.



4. **Loop over live sessions**: If there are any live sessions. Loop over every session and terminate it while also sending a message in discord telling the security specialist that a live session was terminated because an account was breached.
5. **Check in and reconcile password**: The account gets checked in after terminating the live session to make sure the CPM can reconcile its password and then reconcile the password. After this happened successfully sends a Discord message saying the account was successfully reconciled and no further action is required.

7.5.1. PYTHON CODE

For this Flow no python code had to be made but I did have to add a webhook to one of my channels for the notifications that get send to Discord.



Identity Flows
Created on May 13, 2024 by deweejo

NAME: Identity Flows
CHANNEL: #security-incidents

8. Sources

CyberArk Docs. (n.d.). https://docs.cyberark.com/pam-self-hosted/Latest/en/Content/Resources/_TopNav/cc_Home.htm

Tunnel | Zero Trust App Connector | CloudFlare. (n.d.). https://www.cloudflare.com/products/tunnel/

Microsoft. (2024). *Bing Copilot* (may 15 version) [Large language model]. https://www.bing.com/chat

- Can you explain in a clear and structured way how the CyberArk Vault server works and what it does?
- you can also access the vault using the PVWA could you add that?
- Could you explain the component server in the same way. The component server holds the PVWA server, the PSM server and the CPM server.
- can you also explain what the domain controller does in this environment?
- Could you also explain the PTA server. And mention what exactly its use is?