SPIRION

Spirion Extensions

Spirion Automation: Private-AI Image Redaction

Table of Contents

Spirion Automation: Private-AI Image Redaction

Introduction

When remediating sensitive data, organizations typically select one of three approaches: modifying the accessibility of the information, disposing of it entirely, or removing only the details that represent undesirable risk if left in place. Spirion features built-in actions that automate all three strategies for data protection, though these same automation policies can also engage third-party solutions specializing in advanced remediation techniques.

Private-AI is a multimodal PII solution with capabilities that include being able to redact sensitive information from documents, images, and audio recordings, featuring an API that can interact with cloud-hosted or on-prem implementations of their software. Since Spirion does not natively redact images files, integrating with Private-AI's ability to do so expands its remediation options without disrupting how the discovery process is centralized.

Common support for Python across APIs, SDKs, and Spirion scripts make it ideal for prototyping how to integrate match results from a Spirion discovery scan with meaningful functionality from third-party applications. The following information serves as reference for automating scripted API interactions with Private-AI.

Requirements

Before working on the steps outlined in this document, please confirm the following:

- The Spirion console (SDM or SDP) is up-to-date and accessible.
- A Spirion agent hosted on a Windows 10 host is online, polling to its console.
- Python is installed on the Windows host(s) running the Spirion agent(s).
    - **NOTE:** LocalSystem must be able to run the Python script called by Spirion.
- Credentials to the Private-AI API are available.

Important Notes

Software Versions

SDP console version 22.Q3.1.226.1, SDM console version 11.8.2, Spirion agent version 12.5, and Python 3.11.1 were used to perform the steps described in this document.

Private-AI API

The configuration explained below references the Private-AI API to redact sensitive information from a file location. This process was tested using a cloud-hosted instance of the de-identification service, however the solution can also run on locally deployed containers.

Process

Spirion can automate the execution of shell scripts with variables derived from scan results returned by Spirion agents. In SDM, this is handled from the Action tab of any given Workflow, and from Playbooks in SDP.

To execute a Python script, Spirion is configured to run a batch script that calls Python while passing in the Spirion match location (a file path) as a command-line argument using the "%Location%" variable.

An example Python script in a file called Spirion_CS_Private-AI-Image-Redaction.txt is provided with this document. The file extension should be changed from .txt to .py before moving it to the local directory referenced by the Windows batch file/script called by Sensitive Data Manager or Sensitive Data Platform.

Scripting the Private-AI API

Using Private-AI's Python SDK, the example script associated with this document executes the following logic:

1. Get the Spirion match result location/path.
2. Confirm the match location's file type.
3. Convert the file to a base64-encoded string.
4. API call to send the file to the Private-AI container for redaction.
5. Decode the API call's response, overwriting the original location with redacted content.

File Type Support

The following file types are currently supported for remote API redaction in Private-AI:

- TXT
- CSV*
- PDF**
- JPEG
- JPG
- TIFF
- DOCX
- XLSX

NOTE: Audio/video file extensions have been omitted from this list since they are not often targets for Spirion scans, however Private-AI also supports the redaction of M4A, MP3, MP4, and WAV files.

* CSV support requires a Private-AI container running locally in the target environment.

** PDF files are flattened to static images after being redacted.

Configuring the Spirion Agent

The location where the Python script is saved will be referenced in the batch file uploaded to the Spirion console per the instructions below.

Configuring Sensitive Data Platform (SDP)

Uploading a Script

1. From the main menu, select ***Settings***.
2. Navigate to ***Script Repository***.
3. Click the ***Actions*** button toward the top-right of SDP and select "Add Script."
4. Specify a ***name*** and ***description*** for the script.
5. Create a Windows batch file containing just the following command:

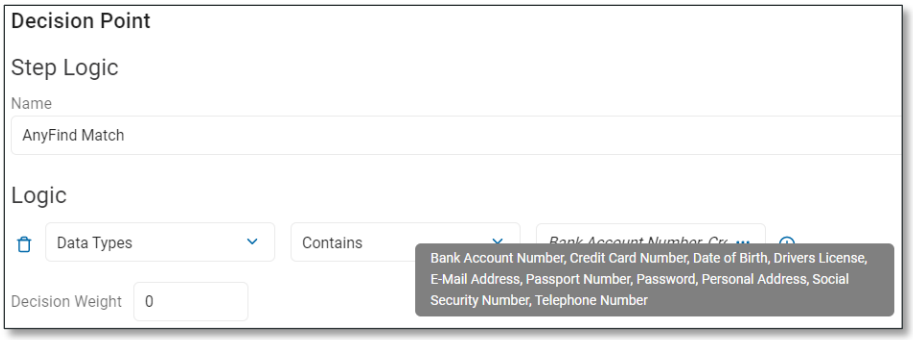
```
python c:\temp\ Spirion_CS_Private-AI-Image-Redaction.py %Location%
```

1. Click the icon next to ***Upload Script*** and upload the new batch file.
2. ***Save*** the uploaded script.

Creating a Playbook

The steps described below cover the relevant steps for automated script execution. For additional guidance, please refer to the Spirion KB Article: How to Write a Playbook.

1. From the main menu, select ***Scans***.
2. Navigate to ***Scan Playbooks***.
3. Click ***Add Playbook*** using the button from the top-right.
4. Enter a ***Name*** and ***Description*** as desired.
5. Under the ***Logic*** section of a ***Decision Point***, select the criteria that should trigger the rule.
    a. For example, specifying "Data Types" and selecting any (or all) of the built-in AnyFind search parameters would trigger this script for every location (e.g. file/document) with 1 or more of the selected data types.



6. From the ***Select Action*** pulldown menu, select "Execute Script" for the action card that corresponds to the intended decision path (yes/no) per the logic specified in the previous step.
7. Under ***Select Script***, choose the script uploaded in the previous section.
8. Define additional Playbook logic as necessary before clicking the ***Save*** icon to finalize changes.

Searching for Data

After creating the Playbook it needs to be referenced in a scan policy. This policy controls what target(s) are scanned, in this case by the Spirion agent(s) staged with the Python script. Playbook logic evaluates results from this scan and triggers the Python script when applicable.

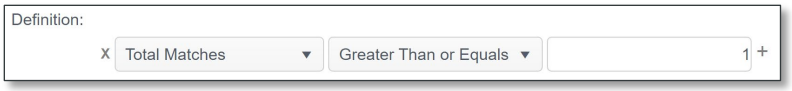
Please refer to the following documentation for additional information about scan policies: Getting Started with Scans.

Configuring Sensitive Data Manager (SDM)

Creating a Workflow

The steps described below cover the relevant steps for automated script execution. For additional guidance, please refer to the Spirion KB Article: How to Write a Workflow Rule.

1. From the ***Workflows*** tab, select "Add" from the ***Rule*** pulldown menu.
2. Enter a ***Name*** and ***Description*** as desired.
3. On the ***Definition*** tab, select the criteria that should trigger the rule.
    a. For example, specifying "Total Matches Greater Than or Equals 1" would trigger this Workflow for every scan with 1 or more match results.



4. On the ***Endpoints*** tab, select the agent(s) that will execute the Python script.
5. On the ***Actions*** tab, enable ***Perform the following remediation action*** and select "Execute script."
6. Create a Windows batch file containing just the following command:

```
python c:\temp\ Spirion_CS_Private-AI-Image-Redaction.py %Location%
```

7. Click the ellipsis button ("**…**") and upload the new batch file.
8. Finalize the creation of this Workflow by clicking ***Finish***.

Searching for Data

Once the Workflow has been synced to the Spirion agent(s), its logic will be evaluated upon the completion of any scan including by the selected endpoint(s). The most common way to initiate a scan in SDM is through a scheduled task.

Please reference the following documentation for additional information: Spirion KB Article: Getting Started with Scheduled Tasks

Outcomes

The accuracy and flexibility of Spirion search logic combined with the dynamic capabilities of its scripting engine readily incorporates advanced redaction from Private-AI into the selective automation handled by SDM Workflows and SDP Playbooks.

For instance, an image file detected as a match by Spirion would be redacted by Private-AI as indicated in the BEFORE/AFTER examples below. This redacted version of the file is written to the offending match location, thus remediating the risk of unintentionally exposed PII.

BEFORE:

AFTER:

Troubleshooting

Spirion Scan Logs

Custom script execution can be verified by auditing Spirion scan logs. In addition to confirming a successful connection to the repository environment targeted in a scan, logs will also indicate that the workflow script has been executed for any given match location:

```
"USER ACTION" "Successfully executed script (via workflow) on the file...
```

However, the statement above only indicates that the batch file ran. Testing the Python script manually is necessary if the expected outcome of the Python script is not observed

API Script Logs

The example Python script provided with this document writes logs to a user-defined location or `C:\temp` by default. Logs are configured to rotate hourly, with success and failure messages confirming the outcome of automated API requests informed by Spirion scan results.

Manual Python Verification

Console-initiated scans control the Spirion (Windows) agent using the LocalSystem account by default. Since Spirion search logs only indicate if the batch file ran properly, any Python errors will not be reported.

To verify that the Python script executes properly prior to being invoked during a scan, launch a command prompt as the LocalSystem account and call the Python script from that terminal. PsExec provides a convenient means of launching cmd.exe with the appropriate context.

Errors preventing the Python script from completing will be displayed in this terminal. Common causes of concern would be: 1) verifying that the LocalSystem account has access to Python, and 2) ensuring that any module dependencies are also accessible to LocalSystem.