SPIRION

Spirion Extensions

Running Spirion Agents in Windows Docker Containers

Table of Contents

Running Spirion Agents in Windows Docker Containers

Introduction

Spirion's on-prem and cloud-hosted platforms, Sensitive Data Manager (SDM) and Sensitive Data Platform (SDP), respectively, both control groups of centrally managed ***Spirion Agents***, which scan for sensitive data throughout unstructured and structured target repositories before sending results back to the Spirion Console (either SDM or SDP).

Agents can operate independently to complete individual searches, or they can be organized into ***Discovery Teams*** to divide a single scan into smaller, simultaneous sub-tasks distributed across multiple Spirion Agents. Adding additional discovery team agents reduces how long it takes to discover data stored in larger targets.

This extension describes one option to virtualize the Spirion Agent, giving flexibility to organizations as they scale out Discovery Teams to accelerate large discovery scans for sensitive data. While virtual machines (VMs) are often sufficient for this task, ***containers*** have gained widespread adoption as a lightweight alternative to traditional VM-based solutions.

The Spirion Agent can be provisioned as an MSI file that is easily incorporated into container deployment configurations.

Requirements

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

- The latest SDM or SDP console is accessible.
- Docker is installed on a Windows machine.
    - Install Docker Desktop on Windows.
- The following files are saved locally to the Docker host:
    - An MSI of the Spirion Agent configured for the Spirion Console.
    - The Microsoft Visual C++ 2015 Redistributable installer.

Important Notes

Software Versions

SDM Console version 11.8.2, SDP Console version 21.Q3.2.82.0 and Spirion Agent version 12.5.3 were used to perform the steps described in this document.

Docker version 4.10.1 was installed on the host system running the Spirion Agent containers. Benchmark scans were also completed for VMs using VirtualBox 6.1.28

Spirion MSI Files

If an MSI file has not been created yet, please reference the KB documentation, How to Create a Custom MSI for Windows Client. It assumes that the following is accessible for download:

- Installation media from the Spirion software portal:
  - Windows Agent installation media
  - License Key File
- Registration file obtained from the SDM or SDP console.
- The latest Spirion MSI Builder.

Process

The Spirion Agent container is configured using the following components:

- Windows Server Core base image
- Microsoft Visual C++ 2015 Redistributable
- Spirion Agent MSI

Gathering Installation Files

1. On the Docker host system, create a directory dedicated to the Spirion Agent container build environment.
2. Move the following files to the new directory:
   - ***vc_redist.x64.exe*** – obtained from Microsoft.
   - ***spirion_installer.msi*** – generated by the Spirion SDP/M Console administrator.

Creating a Dockerfile

1. Create a new file named ***Dockerfile*** in the Spirion Agent container directory that contains the following text:

```
# escape=`

#  Change the base image as appropriate for your scenario.

FROM mcr.microsoft.com/windows/servercore:ltsc2019

COPY vc_redist.x64.exe spirion_installer.msi C:\

SHELL ["powershell"]

RUN Start-Process vc_redist.x64.exe -ArgumentList '/i', 'C:\vc_redist.x64.exe',
'/quiet', '/norestart' -NoNewWindow -Wait

RUN Start-Process msiexec.exe -ArgumentList '/i', 'C:\SpirionAgent.msi',
'/quiet', '/norestart' -NoNewWindow -Wait
```

NOTE: The files highlighted above correspond to the two files moved to the Dockerfile directory per the instructions in the previous section.

Building the Docker Image

From a terminal, navigate to the Spirion Agent container directory and enter the following command to build the container image:

```
docker build -t <image_name> .
```
← INCLUDE the "." to specify current directory

NOTE: Replace "`<image_name>`" with a label that clearly references your Spirion Console environment, removing the angle brackets (</>).

For example, running "`docker build -t sdp_agent .`" would create a base image named ***sdp_agent***.

Running the Docker Image

Once built, the image can be used to initialize as many Spirion Agent containers as the Docker host can support. Please refer to the ***Docker run*** reference article for additional guidance on container runtime definitions.

Enter the following command from a terminal to build the Spirion Agent container image:

```
docker run --hostname <container_name> --name <container_name> --dns
<dns_server> -dt <image_name>
```

NOTE: Replace "`<container_name>`" with a label that clearly references the individual Spirion Agent, and specify a "`<dns_server>`" capable of resolving the URL of your Spirion Console environment. The "`<image_name>`" should match what was entered in the previous section when building the Docker image.

For example, ***SDP-DKR###*** could be incremented as containers are deployed.

Outcomes

After executing the `docker run` command, the new container will be running in the background. Typing `docker ps` into the terminal will list all running containers (or enter `docker ps -a` to view all running and stopped containers). Similarly, the Docker Desktop application will list and track the status of all containers from its GUI.

If the container is working as intended, it should automatically check into the Spirion Console, showing up as an endpoint with the hostname specified in the previous section.

Benefits

Initial benchmarking has shown that Spirion Agent containers consistently search through both structured and unstructured data faster than virtual machines scanning the same repositories. In addition to optimizing scan performance, containers also offer the following advantages:

1. **Cost-effective hardware utilization**. Free up system resources for additional workers and/or larger workloads, or simply reduce current hosting costs by decommissioning unnecessary guest VMs.
2. **Automated deployment flexibility**. Container orchestration is easily scripted (Kubernetes, Ansible, Jenkins, Docker-py, etc.) for elastic, scalable infrastructure with seamless integration options.
3. **Portability & customization**. Containers are compatible with nearly all modern operating systems and include a wealth of configuration options, initialization settings, and runtime parameters.

Verifying Agent Containers

Running containers can be interfaced with by either:

1. Inputting `docker attach <container_name>` from the terminal.

   a. **NOTE:** Press ***Ctrl-p*** followed by ***Ctrl-q*** to detach from the container.
2. In the Docker Desktop UI, selecting the ***Open In Terminal*** button for a given container.

Connectivity to the Spirion Console can be confirmed by pinging the server address from the container's terminal – routing between the containerized Spirion Agent and any target repositories can be confirmed in a similar capacity. A list of running processes is generated by the `tasklist` command, which will include entries for "idfEndpoint.exe," "idfEndpointWatcher64.exe," and "idfServicesMonitor.exe" when the Spirion Agent is running properly under default conditions.