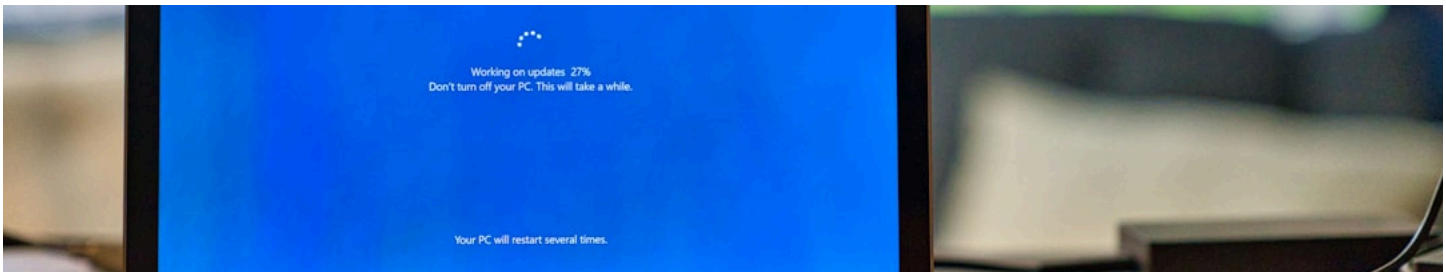

dotnet Crawlers

We have dotnet specific crawlers. These are for hard to reach places, Windows Systems using external crawlers, where Java (JVM) is more trouble than it is worth.

Our dotnet Crawlers run as windows services. They operate similarly to their Java equivalents. At present there is only one crawler that has been ported to dotnet. That crawler is the Microsoft Fileshare crawler.

A file-share needs to be mapped to the server where this crawler is installed as a service. A typical mapping (UNC) looks like `\\server\share-name\folders`, where the folder(s) are optional. Our dotnet Microsoft Filecrawler expects that share to be a local share of the server it runs on.

Installing dotnet

Our crawler has been created using Microsoft dotnet Framework 8.0. This is a slightly older version of the Framework that is compatible with older Microsoft servers. The oldest server we have tested this on is a Microsoft 2016 server. However, any Windows version that can have this 8.0 Framework installed can run this service.

To install, open PowerShell and type

```
winget install Microsoft.DotNet.Runtime.8
```

It is possible that `winget` isn't installed on your Windows service, in that case you will have to Google and download Microsoft's dotnet Framework 8.0 as an MSI from their website, either SKD or Runtime will do.

Verify installation

```
dotnet --list-runtimes
```

Download the dotnet crawlers

The dotnet (Framework 8.0) crawlers can be downloaded from:

64 bit Windows: https://dataset.simsage.co.uk/data/software/dotnet-crawler-x64.zip

32 bit Windows: https://dataset.simsage.co.uk/data/software/dotnet-crawler-x86.zip

Testing and Installing the service

It is a good idea to run our code from the console first as a quick test all is good to go. You need to get some details from SimSage before you do this. All these parameters are required, unless they are marked as `[optional]`.

Parameter	Use

-server	`-server https://yourserver.simsage.ai/api` The location of your SimSage server, must be contactable from your location, try `ping yourserver.simsage.ai`. As long as the name resolves to an IP address you should be ok.
-crawler	`-crawler file` This parameter must be present. For now, only `file` is supported for this crawler.
-org	`-org xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` The organisation ID of your organisation
-kb	`-kb xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` The knowledge-base ID of your system
-sid	`-sid xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx` The security ID of your knowledge-base
-aes	`-aes xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx` The Platform AES key shared with SimSage to encrypt / decrypt data.
-source	`-source 1` The source ID of your external Microsoft Fileshare crawler. This crawler must be set to external mode.
-encryption	[optional] Do not use, currently not supported.
-noselfsigned	[optional] Indicate that this crawler will not allow a self signed certificate for its communications (default is allow)
-exitwhendone	[optional] Quit the program once crawling has completed. Default is to go to sleep and re-run the crawler afterwards.
-disablecache	[optional] Disable the local files seen cache. This cache is enabled by default.

Run your crawler for a quick test from the command line, press ^C once you see it communicate. Make sure you use the values of your own server. You must run as administrator with elevated privileges due to us accessing the Event Viewer logs.

The parameters used below in these examples are all made up. You will need to get these values from your SimSage instance, or ask the SimSage team for help.

```
# go into the folder you've extracted your SimSage dotnet crawler
Crawlers.exe /debug -server https://localhost:8080/api -crawler file -org xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -kb xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -sid xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -source 5 -aes xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Install this Crawler as a service, once you've verified the crawler can communicate and is operational. You must use `sc.exe` (Microsoft's service controller) to install our service.

```
# use the same values as above
sc.exe create "SimSageCrawlerService-1" binPath= "\"C:\Path\To\Your\Crawlers.exe\" -server http://localhost:8080/api -crawler file -org xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -kb xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -sid xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx -source 5 -aes xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" DisplayName= "SimSage FileCrawler Service 1" start= auto
```

You can use the Even Viewer to see how the crawler is performing. Use the service control panel to stop and start this service.

NB. If you see any errors like "**Unhandled exception. System.IO.FileLoadException: Could not load file or assembly 'C:\...\Crawlers.dll'. Operation did not complete successfully because the file contains a virus or potentially unwanted software. (0x800700E1)**" you will need to add these files to Microsoft Windows Defender. This means the Virus Scanner is taking exception with the SimSage dotnet code.

Remove the service

Use the following command if you need/want to remove this service. Stop the service using the service control panel. Then run

```
sc.exe delete "SimSageCrawlerService-1"
```