METASPONSE CREW AID

3 February 2022

DEPLOYMENT CONSIDERATIONS

When planning for host collection using Metasponse, ensure the following:

- When operating Metasponse from the MIP, If Metasponse is hosted on a separate host, operators will ensure that the MIP time is synced with the Metasponse host so that scheduled job requests being sent from the MIP to the Metasponse server will make chronological sense to both hosts and will be executed correctly at the appropriate time
- Metasponse artifacts look malicious and throw Suricata Alert
    - SMB, WMI, Reg Keys, .bat file in temp
    - Need TTP on how to recognize this behavior and not mischaracterize
- MP network allows one or more transport mechanisms available in Metasponse (WMI & SMB, SSH, MPT, Remote Scheduled Tasks & SMB)
    - Ensure host supports script execution from location Metasponse is configured to drop collection scripts (default: Linux - /tmp, Windows- %SYSTEMROOT%\Temp)
        - HBSS HIPS and/or SELinux policy may block execution from Metasponse working directory.
- Is Metasponse in the same subnet as target Windows hosts? Make sure the following rules (as specified in the Metasponse SMBTransport.pdf and WMITransport.pdf documetation) in the target Windows hosts' Windows Defender Firewall specify a scope that includes the IP address that Metasponse will communicate with them from (usually the DIP's point of presence in the Mission Partner network):
    - Scope:
        - Some defaults may limit scope to "Local subnet"
    - Rules:
        - **`File & Print Sharing`**
        - every **`Windows Management Instrumentation`** rule, including (WMI-In)

TEMPLATES

- Create a Template
  - Follow the steps to create a new job from above up until adding target addresses
  - Without populating the target addresses field, and after validating that all other information is accurate, scroll down on the left side of the page until you see the “Save Job as Template” button; click it to create a template
- Clone a Template
  - Navigate to [Windows Server IP]:5003 (must be in Chrome browser)
  - Click on the 🗎 Templates navbar item at the top
  - Locate the name of the template you would like to clone and click the ⧉ clone button located in the same line on the far right

CONFIGURE METASPONSE DEFAULTS (HIT THE SAVE BUTTON AFTER EACH ONE)

- Navigate to [Windows Server IP]:5003 in Chrome
- Click on the ⚙ Settings navbar item at the top
- Set ElasticSearch Server URL
- Set Privilege Escalation to “On”
- Set Plugin Debug to True
- Set Logging Level to Error
- Set Pickup Delay to 600
- Set Pickup Timeout to 20
- Set Maximum Pickup File Size to “3220000000” (3.00gb)
- Leave all other settings with their default values

METASPONSE – USAGE

CREATE AND RUN A NEW JOB

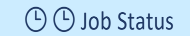
- From any Metasponse page, click the + New Job navbar item
- Give the job a name
    - Typically during execution, this name should be given to you by a crew lead and refer to a date, set of collectors, and target list
    - For example, if today is June 21st, and you are running all collectors against a target list labeled “W1”, your job would be titled “621.all.w1”
    - Consistent job names enables quick lookup of job results within Kibana and will help validate collection results
- Under 📷 📷 Collectors, select the collectors as indicated by your crew commander
    - For example, if you were running all collectors against a set of Windows targets, you would select all collectors labeled “cpt_[collector type]_windows”, HashFilesystem, and RAIR
    - Do not combine two OS types in one job or template (e.g. cpt_system_windows and cpt_system_solaris cannot run at the same time)
- Under Transports, select the transports appropriate for the systems which your crew commander has assigned you to target
    - For Windows systems, select SMB and WMI
    - For Unix systems (including RedHat and Solaris), select SSH
- Under 🔍 🔍 Analyzers, select the ElasticSearch Analyzer
- Under ⚙ Authentication

  - If using SMB and WMI, populate the Domain, Password, and Username fields, ensuring the Privilege Escalation field is set to “On”.
  - If using SSH, populate the SSH SU User (likely root), SU Password (likely root password), Unix Username, and Unix Password fields, ensuring the Privilege Escalation is set to “On” and the SU Enabled box is checked.
- Under ⚙ Collection
  - See HashFilesystem under the Collector Special Cases section below
- Under ⚙ Logging
  - Ensure Logging Level is set to “error”
  - Ensure Plugin Debug box is checked
- Under ⚙ Miscellaneous
  - See the Collector Special Cases section below
- Under ⚙ Pickup
  - Ensure the Maximum Pickup File Size field is set to >= 3 Gigabytes
  - Ensure the Pickup Delay is set to >= 600 minutes
  - Ensure the Pickup Timeout is set to >= 20 Minutes
- Under ⚙ Deployment
  - Populate the Target Addresses field with the target IPs assigned from your crew commander (do not use hostnames)
- If the server has been configured correctly according to this guide, all other default values should be left as they are
- To run the job
  - Ensure target list is accurate using a two-person integrity check
  - After receiving the go-ahead from your crew commander, click the ▶ Schedule button to start the job

COLLECTOR SPECIAL CASES

- Some collectors have additional configuration options to extend functionality
- HashFilesystem
    - Under ⚙ Collection
        - The HashFS: Check Signature field toggles signature checking on digitally signed files
        - The HashFS: Discover field toggles hashing files on physically attached storage drives
        - The HashFS: Unix Root Directories and HashFS: Windows Root Directories fields allow specification of directories to run hashing from; add or remove directories to expand or limit the hashing capability respectively
    - Under ⚙ Filter
        - The HashFS: Hash All Filetypes field allows specification of the specific file types to do hashing on within the directories specified in the Root Directories fields; by default several file types are already prepopulated. To expand hashing to every file type, remove all entries and replace with a single “*” entry

MONITOR JOB PROGRESS

- Metaview
  - While a job is running, information can be gathered on the current job status through the web browser
  - On the job page after the job has been started (you should see a ⚙ Job Control panel on the left and a  🕒 🕒 Job Status panel on the right), replace "jobs" in the URL with "metaview"
  - e.g. for a job titled "621.all.w1" with Metasponse running on a Windows Server with hostname "win2k12r2"
    http://win2k12r2:5003/jobs/621.all.w1
    can be changed to
    http://win2k12r2:5003/metaview/621.all.w1
  - From here, two collections are of particular interest
    - The "hosts" collection is a quick way to view which target IPs succeeded and which failed
    - The "job_log" collection gives a list of detailed messages which can highlight a number of interest items
      - e.g. the "job_log" will show why SMB, WMI, or SSH may fail to authenticate with the exact error, why data may not have ingested into ElasticSearch, etc
- Get-TerrainStatus
  - Once you see the Successes column under the Plugin Statistics panel on the job page populate with 1s (particularly for the Transports), job status can be queried on the remote targets
  - To query a job, open the Metasponse Provider shortcut from the desktop of the Windows Server that the Metasponse server is installed on, and change directories into the job name for which you are interested
  - The following command will list the contents of a directory called "data" within the job directory on the remote system:

```
> Get-Item . | Get-TerrainStatus
```

  - How to interpret this data:

- **Name**: the name of the file
- **Count**: the number of target systems this file resides on; for most files, this number will equal the number of targets for all selected job collectors when the job is complete
- **PresentRatio**: the fraction of hosts this file resides on (i.e. 75% for a job with four target hosts would mean 3/4 target hosts have the file present within the “data” directory)
- **AverageFileSize**: the average size in bytes of all the file across all targets. This is the **best indicator for job status**; if this size is

growing for any file, that means the associated collector is still gathering information on at least one remote host

        - **MinFileSize**: the size of the smallest file on the targets
        - **MaxFileSize**: the size of the largest file on the targets

    - Additionally, a file named **`scripts_done.txt`** will be written to the data folder once all selected collectors have finished executing; if the Count field for this file equals the number of targets for this job, then it is complete
    - In order to continually monitor the progress using Get-TerrainStatus as above or set up a simple script in Metasponse Provider:

```
While(1){ Get-Item . | Get-TerrainStatus; Sleep 60; Echo
 “-------------------------------” }
```

    - These queries can only be run from the Windows Server that the Metasponse server is installed on. Do not attempt to query job status from a local Metasponse provider instance as pfSense will not allow this connection.

- Plugin Log
    - Information within the Plugin Log shows output on STD_OUT and STD_ERR from the executing collector scripts
    - Please first refer to the section titled Read and Copy Bin Files From Metasponse Job for steps on reading the content of the **`plugin-log.txt`** file
    - The output of each script will be both start and end with a string of “=====================”; if the log indicates that a script has started running but only see one of these strings, this indicates that this script is still running on the target system

JOB PICKUP

- Gathering collected information
  - Once determined that no further scripts are being executed on a target using the methods mentioned above and approval has been given by the crew commander, click ⭡ Begin Pickup from the job page
  - Pickup is complete when the ✎✎ Job Data panel containing the ▤▤ Analysis button appears
  - Click ▤▤ Analysis to view the results in the raw JSON format under the Miscellaneous panel
  - Pickup may take a long time depending on the number of remote systems and the collectors run. For jobs with particularly large amounts of data, information on the job page regarding pickup may seem to get stuck. In many of these cases, all data has been retrieved from the remote host, but the pickup completion is awaiting ElasticSearch ingestion. To verify:
    - Use Get-TerrainStatus to verify no files remain on the target systems as described in the above section
    - Open the Discover tab in Kibana and navigate to the ms-* index
      - Click “Add a filter +” at the top, with “job_name” as the field, “is” as the operator, and [job name] for the value
      - Take note of the hit count listed at the top of the page, and refresh a couple of times. If the number is growing, pickup is still waiting on ElasticSearch ingestion

ARTIFACT CLEANUP

- As long as all collection scripts have finished executing on the target system, Metasponse finishes pickup by removing all artifacts from the remote system
- If pickup is started before all scripts have finished executing, these scripts will continue to write to files within the data directory and will not be removed until system reboot.
    - Do not begin pickup early without approval of the crew commander
    - If pickup must be conducted early, ensure artifacts are removed from the target system through contingency means or direct coordination with the mission partner
- Do not abort jobs that have successfully connected/authenticated to target systems. This will not remove any artifacts, and all artifacts will remain until system reboot or removed through contingency means or direct coordination with the mission partner

EXPORT PREVIOUSLY COLLECTED INFO FROM METASPONSE TO ELASTICSEARCH

- In some cases, you may need to export information gathered from a previously run Metasponse job (e.g. a job did not successfully complete pickup)
- Before initiating the export, verify that the job name does not show up when searching for it in Kibana
  - Open Kibana and navigate to Discover
  - Click “Add a filter +” at the top left
  - Add “job_name” under fields, “is” for the operator, and [job name] for the value
  - If any data shows up from this query, the data has already been ingested into ElasticSearch
- After verifying data does not exist, copy only the name of the job from which you would like to export information, then create a new job (do not clone the previous)
- Name the new job the name of the previous job with “.repeat” appended to the end
- Do not select any collectors or transports
- Under ✳ Other, select “Task: Elasticsearch Export”
- Under the Analysis section, verify the “ElasticSearch: Server URL” is correct and enter the previously copied job name into the “Job Name” field with a single line
  - If you need to export data from multiple jobs, you can add each job name as an individual line to the next field instead
- Click the ⟳ Check button to clear any potential errors
- Click the ▶ Schedule button to start the export
- Exports can be monitoring using the “metaview” method described in the Monitor Job Progress above

READ AND COPY BIN FILES FROM METASPONSE JOB

- For offline analysis, it is beneficial to configure a local instance of the Metasponse provider to be able to query previously conducted job info (this will not work to query job status—Get-TerrainStatus queries the external system directly from the system it’s run from, so pfSense will not allow this; please continue to use the Metasponse provider from Windows Server 2012 for these queries)
    - Run Metasponse installer with version >= 1.5.7 on local Windows VM and reboot
    - Start the Metasponse UI as administrator locally using the desktop shortcut
    - Navigate to localhost:5003 in Chrome
    - Click on the ⚙ Settings navbar item at the top right
    - Under “Database Options,” change “Address” from “localhost” to [Windows Server IP]
    - Restart the local Metasponse UI
- Once a local instance has been configured, open the Metasponse provider using the desktop shortcut from your local Windows VM and change directories into the job name in which you are interested
- To read a bin file (such as the plugin log for debugging purposes):

```
> dir bin | ? { $_.Name -eq “plugin-log.txt”} |
Get-BinContent -Encoding asci
```

- To copy bin files (such as all .evtx files):

```
> dir bin | ? { $_.Name.Contains(”.evtx”) } |
Copy-BinFile -Destination
C:\Users\assessor\Desktop\evtx
```

    - This drops all files into directories named with the copied file’s originating IP