Solution Guide: Incident Analysis with Kibana

Instructions: Investigating the Incident

Even though you already know what you did to exploit the target, analyzing the logs is still valuable. It will teach you: - What your attack looks like from a defender's perspective.

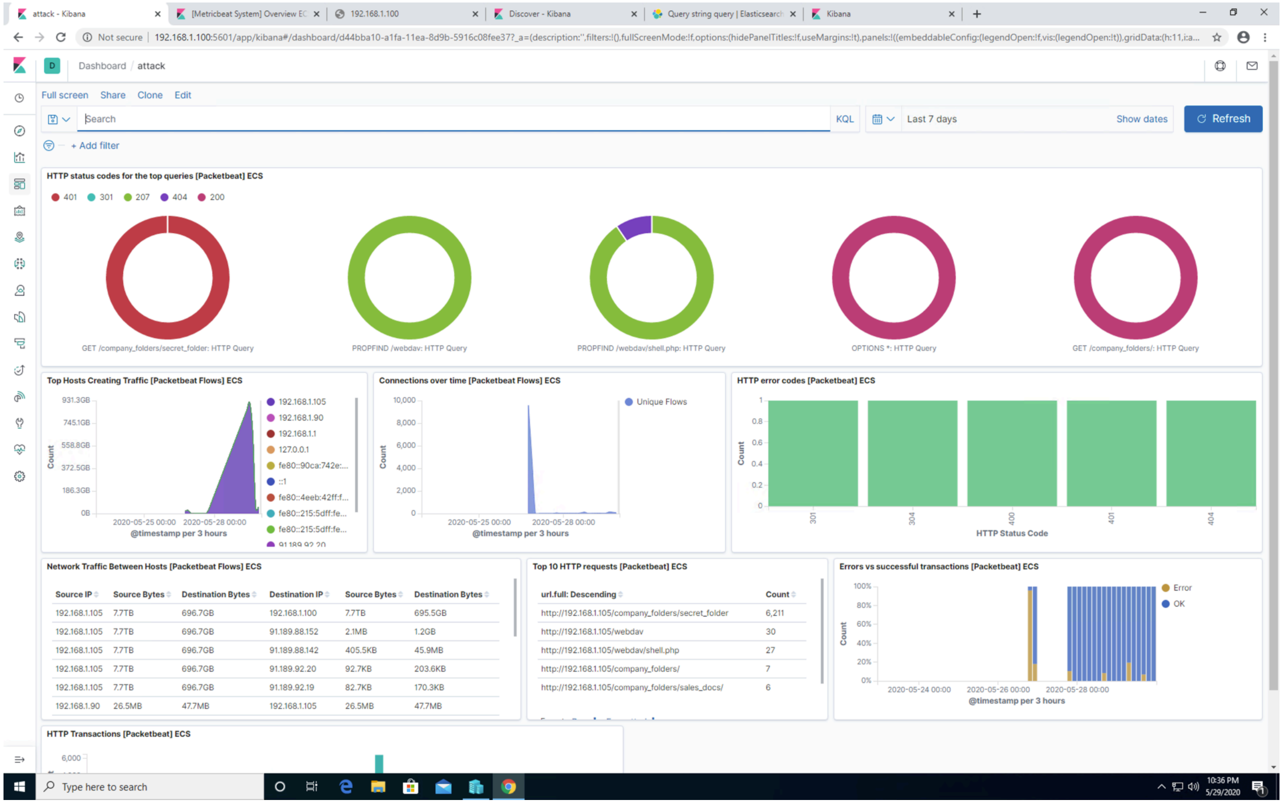
- How stealthy or detectable your tactics are.
- Which kinds of alarms and alerts SOC and IR professionals can set to spot attacks like yours while they occur, rather than after.
- While going through the solution file, please note that the IP addresses here need to be replaced your machine's IP addresses.

Double-click the Google Chrome icon on the Windows host's desktop to launch Kibana. If it doesn't load as the default page, navigate to http://192.168.1.105:5601.

Start by creating a Kibana dashboard using the pre-built visualizations. Navigate to your home page, then scroll down to **Visualize and Explore Data** then **Dashboard**.

Click on **Create dashboard** in the upper left hand side. On the new page click on **Add an existing** to add the following existing reports: - `HTTP status codes for the top queries [Packetbeat] ECS` - `Top 10 HTTP requests [Packetbeat] ECS` - `Network Traffic Between Hosts [Packetbeat Flows] ECS` - `Top Hosts Creating Traffic [Packetbeat Flows] ECS` - `Connections over time [Packetbeat Flows] ECS` - `HTTP error codes [Packetbeat] ECS` - `Errors vs successful transactions [Packetbeat] ECS` - `HTTP Transactions [Packetbeat] ECS`

Your final dashboard should look similar to:


attack - Kibana
[Metricbeat System] Overview EC
192.168.1.100
Discover - Kibana
Query string query | Elasticsearch
Kibana
Not secure
192.168.1.100:5601/app/kibana#/dashboard/d44bba10-a1fa-11ea-8d9b-5916c08fee37?_a=(description:'',filters:!(),fullScreenMode:!f,options:(hidePanelTitles:!f,useMargins:!t),panels:!((embeddableConfig:(legendOpen:!f,vis:(legendOpen:!t)),gridData:(h:11,i:a...
Dashboard / attack
Full screen
Share
Clone
Edit
Search
KQL
Last 7 days
Show dates
Refresh
+ Add filter
HTTP status codes for the top queries [Packetbeat] ECS
401
301
207
404
200
GET /company_folders/secret_folder: HTTP Query
PROPFIND /webdav: HTTP Query
PROPFIND /webdav/shell.php: HTTP Query
OPTIONS *: HTTP Query
GET /company_folders/: HTTP Query
Top Hosts Creating Traffic [Packetbeat Flows] ECS
931.3GB
745.1GB
558.8GB
372.5GB
186.3GB
0B
Count
2020-05-25 00:00
2020-05-28 00:00
@timestamp per 3 hours
192.168.1.105
192.168.1.90
192.168.1.1
127.0.0.1
fe80::90ca:742e:...
::1
fe80::4eeb:42ff:f...
fe80::215:5dff:fe...
fe80::215:5dff:fe...
91.189.92.20
Connections over time [Packetbeat Flows] ECS
10,000
8,000
6,000
4,000
2,000
0
Count
2020-05-25 00:00
2020-05-28 00:00
@timestamp per 3 hours
Unique Flows
HTTP error codes [Packetbeat] ECS
1
0.8
0.6
0.4
0.2
0
Count
301
304
400
401
404
HTTP Status Code
Errors vs successful transactions [Packetbeat] ECS
100%
80%
60%
40%
20%
0%
Count
2020-05-24 00:00
2020-05-26 00:00
2020-05-28 00:00
@timestamp per 3 hours
Error
OK
HTTP Transactions [Packetbeat] ECS
6,000
Type here to search
10:36 PM
5/29/2020

Network Traffic Between Hosts [Packetbeat Flows] ECS

Source IP	Source Bytes	Destination Bytes	Destination IP	Source Bytes	Destination Bytes
192.168.1.105	7.7TB	696.7GB	192.168.1.100	7.7TB	695.5GB
192.168.1.105	7.7TB	696.7GB	91.189.88.152	2.1MB	1.2GB
192.168.1.105	7.7TB	696.7GB	91.189.88.142	405.5KB	45.9MB
192.168.1.105	7.7TB	696.7GB	91.189.92.20	92.7KB	203.6KB
192.168.1.105	7.7TB	696.7GB	91.189.92.19	82.7KB	170.3KB
192.168.1.90	26.5MB	47.7MB	192.168.1.105	26.5MB	47.7MB

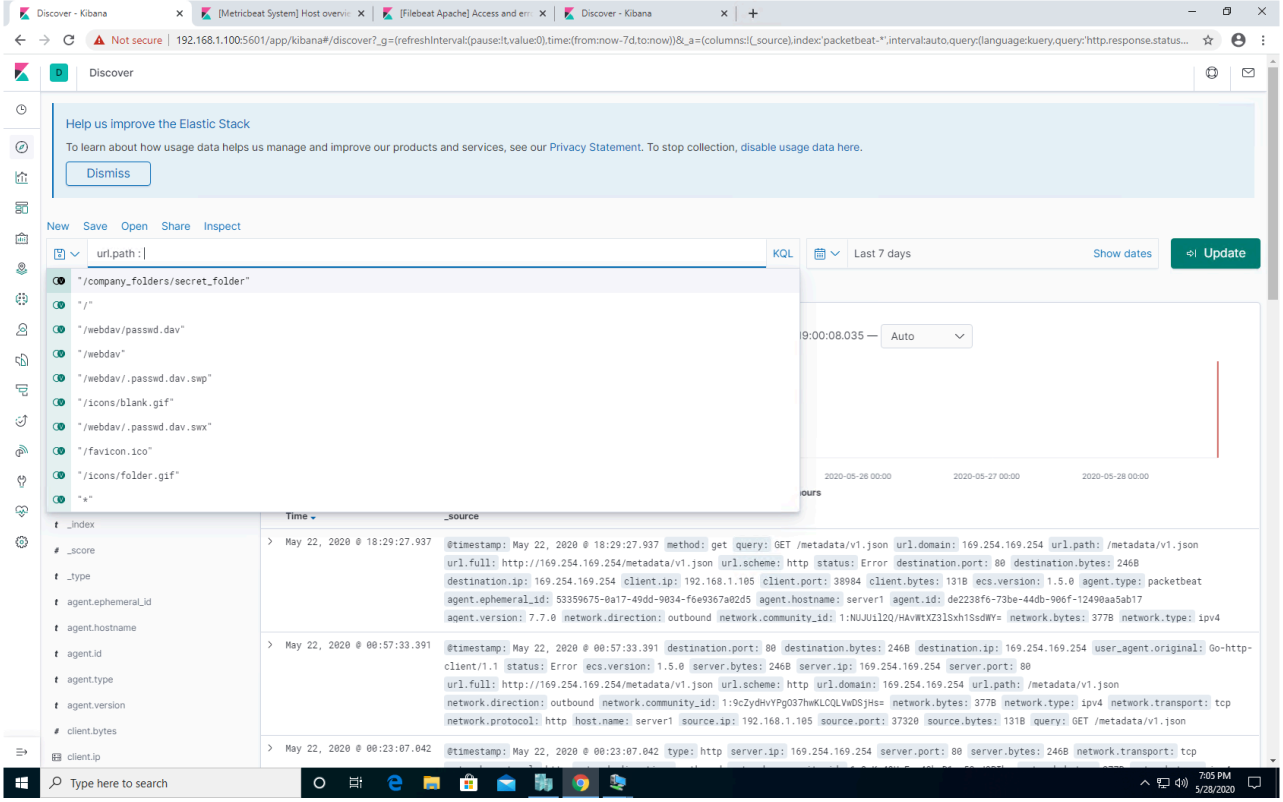
Top 10 HTTP requests [Packetbeat] ECS

url.full: Descending	Count
http://192.168.1.105/company_folders/secret_folder	6,211
http://192.168.1.105/webdav	30
http://192.168.1.105/webdav/shell.php	27
http://192.168.1.105/company_folders/	7
http://192.168.1.105/company_folders/sales_docs/	6

Next, get familiar with running search queries in the `Discover` screen with Packetbeat. - On the Discover page, locate the search field. - Start typing `source` and notice the suggestions that come up. - Search for the `source.ip` of your attacking machine. - Use `AND` and `NOT` to further filter you search and look for communications between your attacking machine and the victim machine. - Other things to look for: - `url` - `status_code` - `error_code`

Some helpful searches include

- `http.response.status_code : 200`
- `url.path: /company_folders/secret_folder/`
- `source.port: 4444`
- `destination.port: 4444`
- `NOT source.port: 80 and NOT source.port: 443`



192.168.1.100:5601/app/kibana#/discover?_g=(refreshInterval:(pause:!t,value:0),time:(from:now-7d,to:now))&_a=(columns:!(_source),index:'packetbeat-*',interval:auto,query:(language:kuery,query:'http.response.status...

Help us improve the Elastic Stack

To learn about how usage data helps us manage and improve our products and services, see our Privacy Statement. To stop collection, disable usage data here.

Dismiss

url.path : |

Last 7 days

```
"/company_folders/secret_folder"
"/"
"/webdav/passwd.dav"
"/webdav"
"/webdav/.passwd.dav.swp"
"/icons/blank.gif"
"/webdav/.passwd.dav.swx"
"/favicon.ico"
"/icons/folder.gif"
"*"
```

```
t _index
# _score
t _type
t agent.ephemeral_id
t agent.hostname
t agent.id
t agent.type
t agent.version
# client.bytes
  client.ip
```

```
Time ▾                       _source
> May 22, 2020 @ 18:29:27.937   @timestamp: May 22, 2020 @ 18:29:27.937 method: get query: GET /metadata/v1.json url.domain: 169.254.169.254 url.path: /metadata/v1.json
                                url.full: http://169.254.169.254/metadata/v1.json url.scheme: http status: Error destination.port: 80 destination.bytes: 246B
                                destination.ip: 169.254.169.254 client.ip: 192.168.1.105 client.port: 38984 client.bytes: 131B ecs.version: 1.5.0 agent.type: packetbeat
                                agent.ephemeral_id: 53359675-0a17-49dd-9034-f6e9367a02d5 agent.hostname: server1 agent.id: de2238f6-73be-44db-906f-12490aa5ab17
                                agent.version: 7.7.0 network.direction: outbound network.community_id: 1:NUJUil2Q/HAvWtXZ3lSxh1SsdWY= network.bytes: 377B network.type: ipv4
> May 22, 2020 @ 00:57:33.391   @timestamp: May 22, 2020 @ 00:57:33.391 destination.port: 80 destination.bytes: 246B destination.ip: 169.254.169.254 user_agent.original: Go-http-
                                client/1.1 status: Error ecs.version: 1.5.0 server.bytes: 246B server.ip: 169.254.169.254 server.port: 80
                                url.full: http://169.254.169.254/metadata/v1.json url.scheme: http url.domain: 169.254.169.254 url.path: /metadata/v1.json
                                network.direction: outbound network.community_id: 1:9cZydHvYPgO37hwKLCQLVwDSjHs= network.bytes: 377B network.type: ipv4 network.transport: tcp
                                network.protocol: http host.name: server1 source.ip: 192.168.1.105 source.port: 37320 source.bytes: 131B query: GET /metadata/v1.json
> May 22, 2020 @ 00:23:07.042   @timestamp: May 22, 2020 @ 00:23:07.042 type: http server.ip: 169.254.169.254 server.port: 80 server.bytes: 246B network.transport: tcp
```

After you create your dashboard and become familiar with the search syntax, use these tools to answer the questions below:

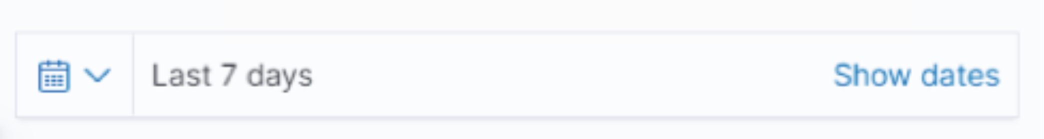
1. Identify the Offensive Traffic

Identify the traffic between your machine and the web machine:

- Staring with a few searches in the 'Discover' area, we can find some interesting interactions.
- Run `source.ip: 192.168.1.90 and destination.ip: 192.168.1.105` in which the source IP is your Kali machine and your destination machine is your web server.
- Run `url.path: /company_folders/secret_folder/`.

When did the interaction occur?

- You know when the interaction happened so we will need to change the timeline that Kibana is searching to see that time period:

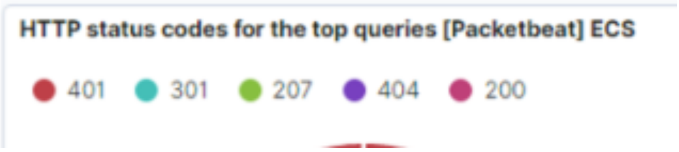


Last 7 days

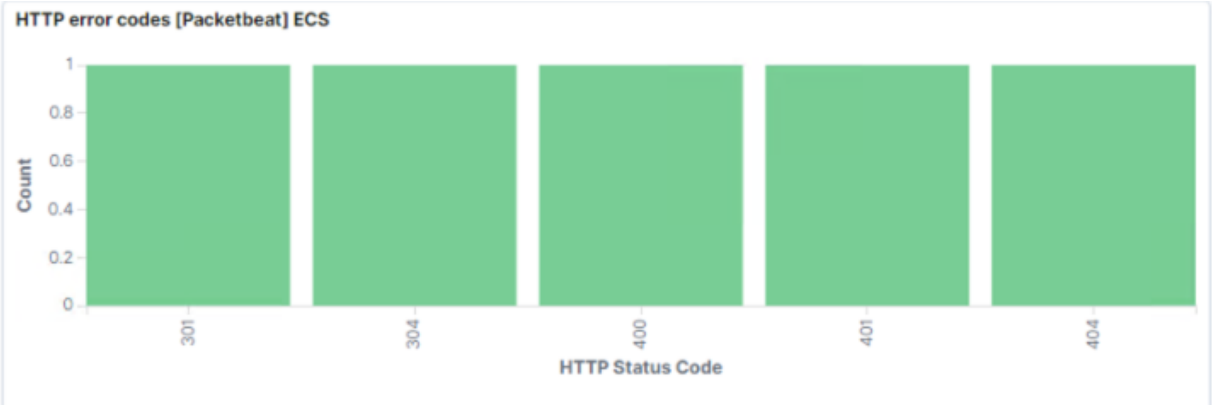
In your dashboard, look through the different panels and use this data to look through the results and notice the following interactions:

What responses did the victim send back?

- On our dashboard, we can see the top responses in the `HTTP status codes for the top queries [Packetbeat] ECS`

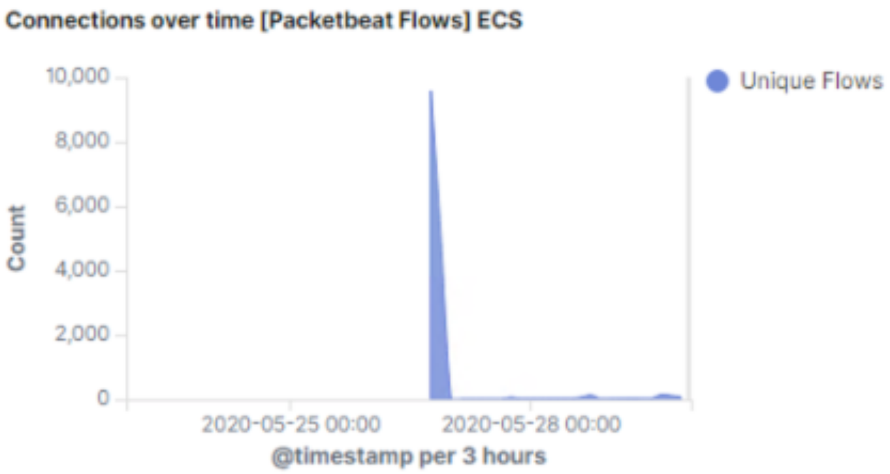


- We can see `401`, `301`, `207`, `404` and `200` as the top responses.
- We can also see with the `HTTP Error Codes [Packetbeat] ECS` panel:

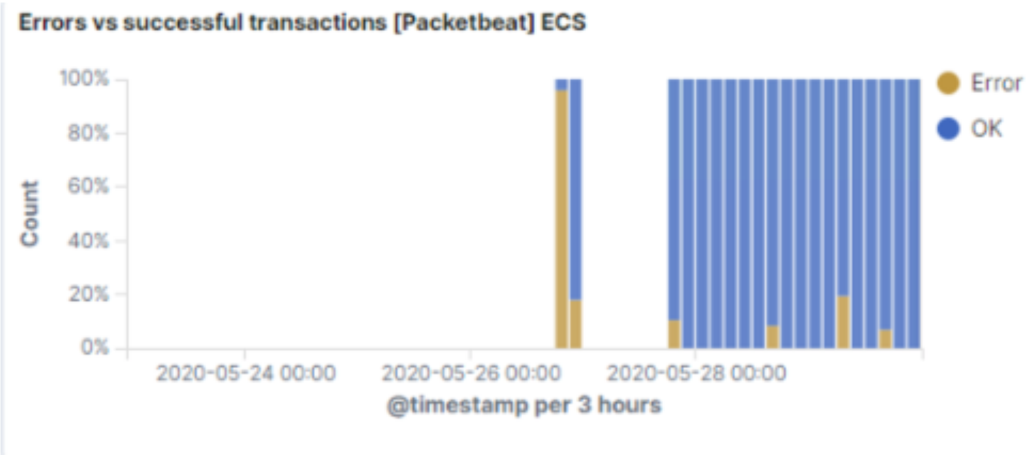


What data is concerning from the Blue Team perspective?

- We can see a connection spike in the `Connections over time [Packetbeat Flows] ECS`



- We can also see a spike in errors in the `Errors vs successful transactions [Packetbet] ECS`

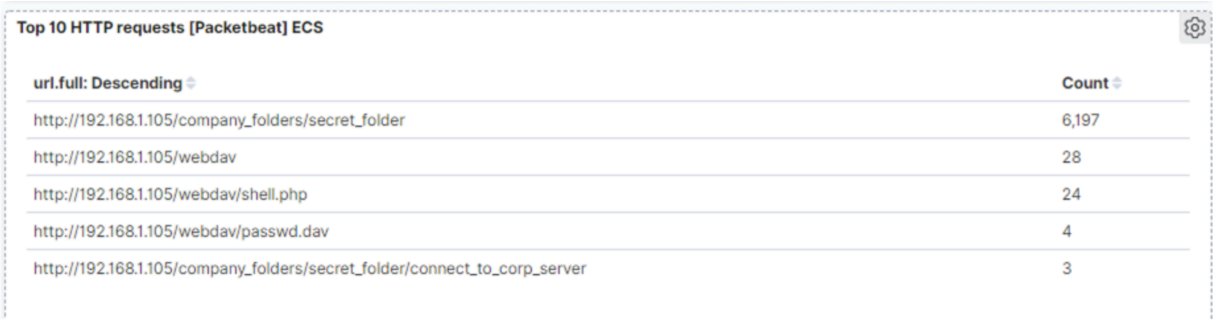


2. Find the Request for the Hidden Directory

In your attack, you found a secret folder. Let's look at that interaction between these two machines.

How many requests were made to this directory? At what time and from which IP address(es)?

- On the dashboard you built, a look at your `Top 10 HTTP requests [Packetbeat] ECS` panel:



Top 10 HTTP requests [Packetbeat] ECS

url.full: Descending	Count
http://192.168.1.105/company_folders/secret_folder	6,197
http://192.168.1.105/webdav	28
http://192.168.1.105/webdav/shell.php	24
http://192.168.1.105/webdav/passwd.dav	4
http://192.168.1.105/company_folders/secret_folder/connect_to_corp_server	3

- In this example we can see that this folder was requested `6,197` times.

Which files were requested? What information did they contain?

- We can see in the same panel that the file `connect_to_corp_server` was requested `3` times.

What kind of alarm would you set to detect this behavior in the future?

- We could set an alert that goes off for any machine that attempts to access this directory or file.

Identify at least one way to harden the vulnerable machine that would mitigate this attack.

- This directory and file should be removed from the server all together.

3. Identify the Brute Force Attack

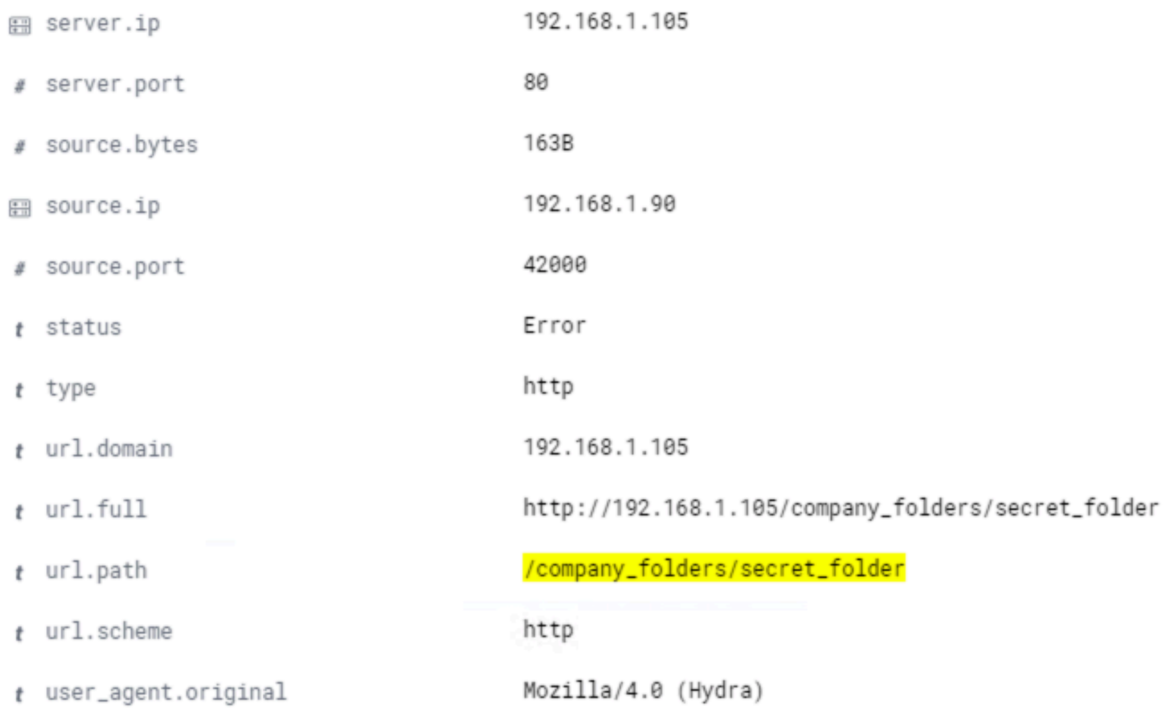
After identifying the hidden directory, you used Hydra to brute-force the target server. Answer the following questions:

Can you identify packets specifically from Hydra?

- Yes, if you are using the search function `url.path: /company_folders/secret_folder/` will show you a few conversations involving this folder.

- In the `Discovery` page, search for: `url.path: /company_folders/secret_folder/`.

Look through the results and notice that `Hydra` is identified under the `user_agent.original` section:

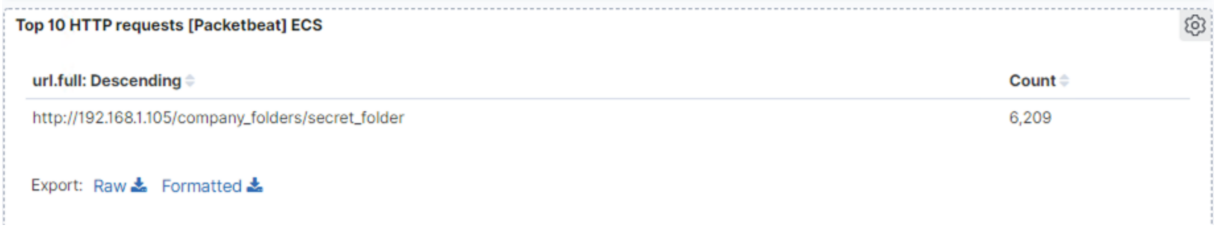


server.ip	192.168.1.105
server.port	80
source.bytes	163B
source.ip	192.168.1.90
source.port	42000
status	Error
type	http
url.domain	192.168.1.105
url.full	http://192.168.1.105/company_folders/secret_folder
url.path	/company_folders/secret_folder
url.scheme	http
user_agent.original	Mozilla/4.0 (Hydra)

How many requests were made in the brute-force attack? How many requests had the attacker made before discovering the correct password in this one?

- In the `Top 10 HTTP requests [Packetbeat] ECS` panel, we can see that the password protected `secret_folder` was *requested* `6209` times, but the file inside that directory was only requested `3` times. So, out of `6209` requests, only `3` were successful.

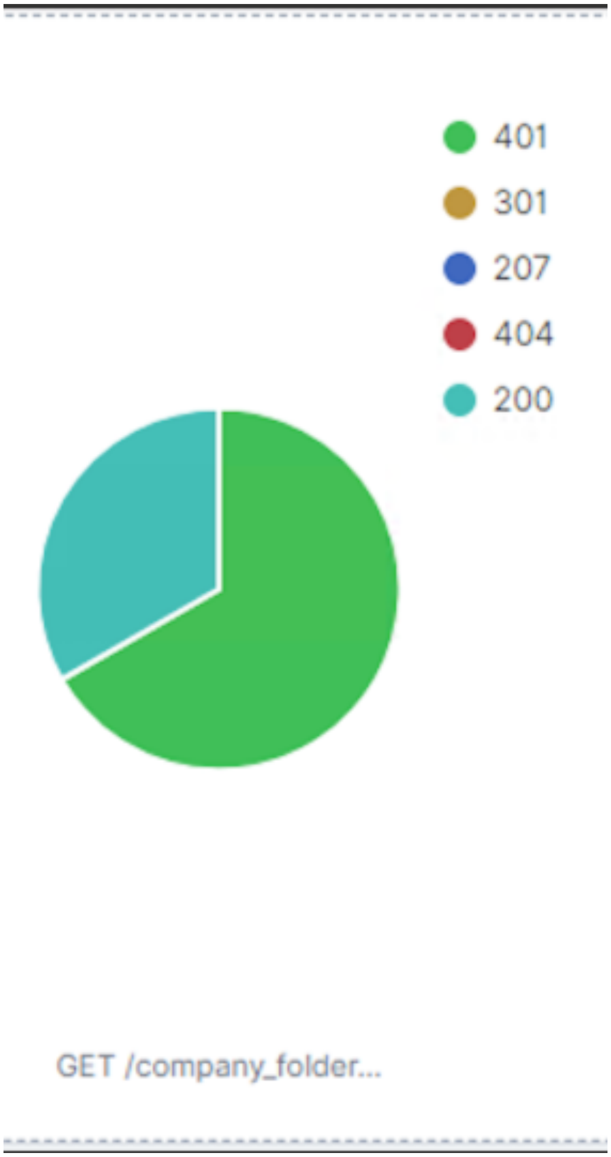
  Note: Your results will differ.

  Top 10 HTTP requests [Packetbeat] ECS

url.full: Descending	Count
http://192.168.1.105/company_folders/secret_folder	6,209

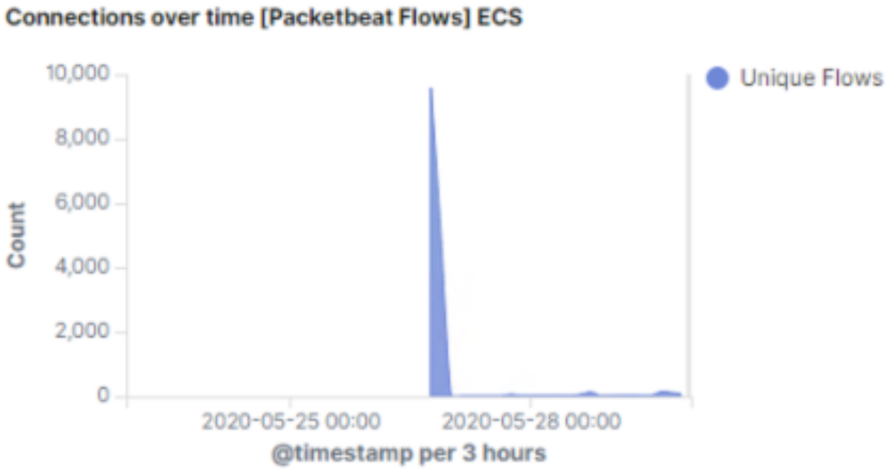
Take a look at the `HTTP status codes for the top queries [Packetbeat] ECS` panel:



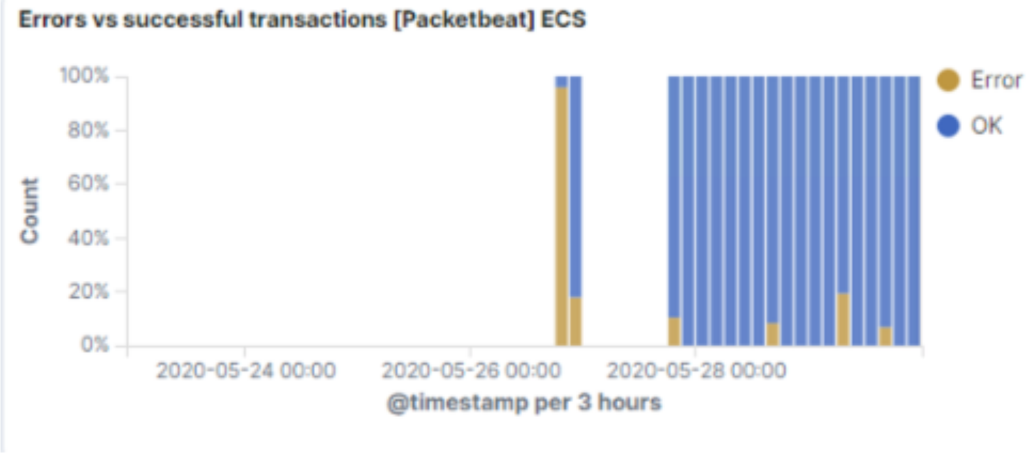
- You can see on this panel the breakdown of `401 Unauthorized` status codes as opposed to `200 OK` status codes.

- We can also see the spike in both traffic to the server and error codes.

- We can see a connection spike in the `Connections over time [Packetbeat Flows] ECS`



- We can also see a spike in errors in the `Errors vs successful transactions [Packetbet] ECS`



These are all results generated by the brute force attack with Hydra.

What kind of alarm would you set to detect this behavior in the future and at what threshold(s)?

- We could set an alert if `401 Unauthorized` is returned from any server over a certain threshold that would weed out forgotten passwords. Start with `10` in one hour and refine from there.

- We could also create an alert if the `user_agent.original` value includes `Hydra` in the name.

Identify at least one way to harden the vulnerable machine that would mitigate this attack.

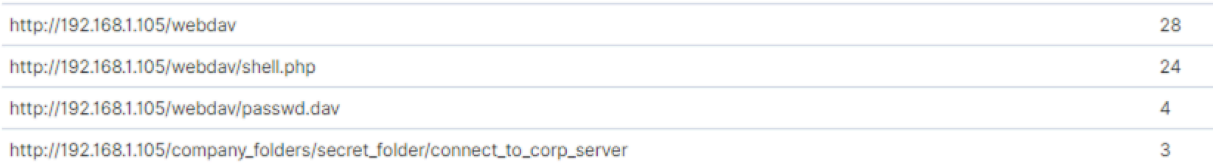
- After the limit of 10 `401 Unauthorized` codes have been returned from a server, that server can automatically drop traffic from the offending IP address for a period of 1 hour. We could also display a lockout message and lock the page from login for a temporary period of time from that user.

4. Find the WebDav Connection

Use your dashboard to answer the following questions:

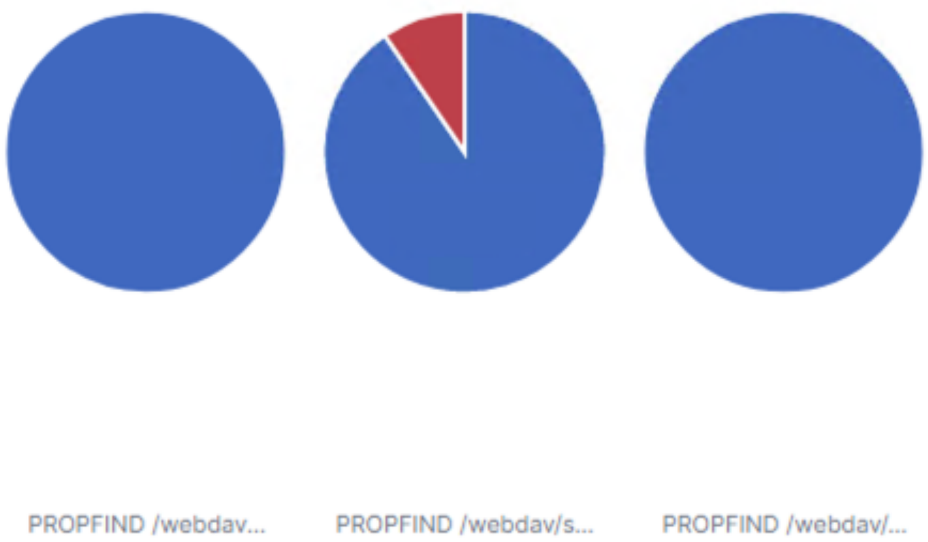
How many requests were made to this directory?

- We can again see in the `Top 10 HTTP requests [Packetbeat] ECS` panel that the webdav folder was directly connected and files inside were accessed.



http://192.168.1.105/webdav	28
http://192.168.1.105/webdav/shell.php	24
http://192.168.1.105/webdav/passwd.dav	4
http://192.168.1.105/company_folders/secret_folder/connect_to_corp_server	3

- We can also see it in the pie charts:



Which file(s) were requested?

- We can see the passwd.dav file was requested as well as a file named `shell.php`

What kind of alarm would you set to detect such access in the future?

- We can create an alert anytime this directory is accessed by a machine *other* than the machine that should have access.

Identify at least one way to harden the vulnerable machine that would mitigate this attack.

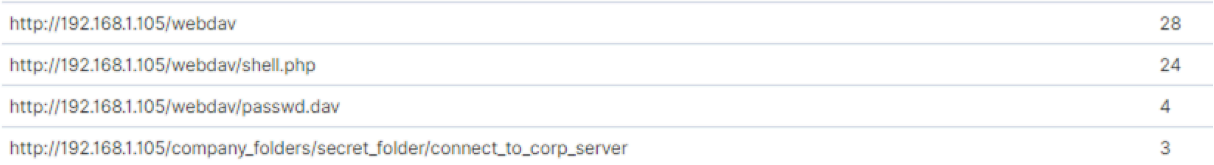
- Connections to this shared folder should not be accessible from the web interface.

- Connections to this shared folder could be restricted by machine with a firewall rule.

5. Identify the Reverse Shell and meterpreter Traffic

To finish off the attack, you uploaded a PHP reverse shell and started a meterpreter shell session. Answer the following questions: Can you identify traffic from the meterpreter session?

- First, we can see the `shell.php` file in the `webdav` directory on the `Top 10 HTTP requests [Packetbeat] ECS` panel.



http://192.168.1.105/webdav	28
http://192.168.1.105/webdav/shell.php	24
http://192.168.1.105/webdav/passwd.dav	4
http://192.168.1.105/company_folders/secret_folder/connect_to_corp_server	3

- Remember that your meterpreter session ran over port `4444`. Port `4444` is the *default* port used for meterpreter and the port used in all of their documentation. Because of this, many attackers forget to change this port when conducting an attack. You can construct a search query to find these packets.

- `source.ip: 192.168.1.105 and destination.port: 4444`

What kinds of alarms would you set to detect this behavior in the future?

- We can set an alert for any traffic moving over port `4444.`

- We can set an alert for any `.php` file that is uploaded to a server.

Identify at least one way to harden the vulnerable machine that would mitigate this attack.

- Removing the ability to upload files to this directory over the web interface would take care of this issue.

:warning: Important Checkpoint :warning:
At this time, you should have completed the following steps:
Step 1: Identify the Offensive Traffic.
Step 2: Find the Request for the Hidden Directory.
Step 3: Identify the Brute Force Attack.
Step 4: Find the WebDav Connection.
Step 5: Identify the Reverse Shell and meterpreter Traffic.

To complete the next part of the project, you should take screen shots that represent each of the issues listed in preparation of compiling them into a report.