Haystacks and Needles: Digging for Interest

DUSTIN DECKER – GFACT, GISF, GSEC, GCIH, GPYC, GCIA, GCWN

This paper details an examination of logs from eleven globally distributed DShield [1] honeypots for February 20th, 2024. DShield honeypots collect a substantial number of repeat alerts.

To overcome this problem, I make substantial use of DShield Manager to analyze and query data using several different formats and methods. DShield Manager is a project I authored during my internship with the Internet Storm Center as part of the SANS Institute bachelor's in applied Cybersecurity (BACS) program [2]. Project code and documentation are available on GitHub [3].

Data was gathered between October 29th, 2023, 23:04:58 through February 29th, 2024, 23:45:27. The honeypots distributed globally with the help of Microsoft Azure, were retired on March 1st, 2024, at midnight.

Just over 330 gigabytes of data on disk, including various attacks against http, ssh and other services were collected, along with full packet captures, from the following locations:

Country	Region	City
ZA	Gauteng	Johannesburg
AU	Victoria	Melbourne
HK	Hong Kong	Hong Kong
US	Florida	Jacksonville
GB	Ireland	Dublin
IN	Maharashtra	Pune
IL	Tel Aviv	Tel Aviv
KR	Sejong	Seoul
BR	Sao Paulo	Sao Paulo
SG	Singapore	Singapore
US	California	San Diego

Executive Summary of Findings:

A dangerous host was observed initiating connections to three globally distributed honeypots.

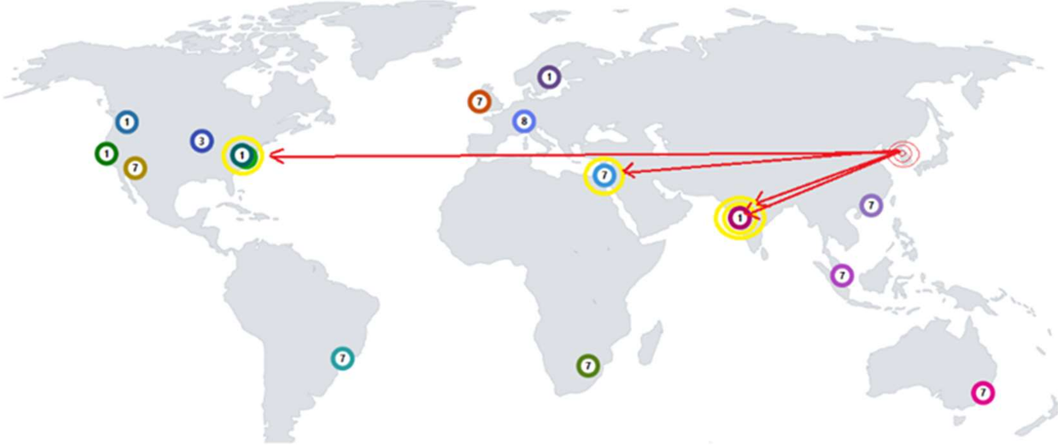
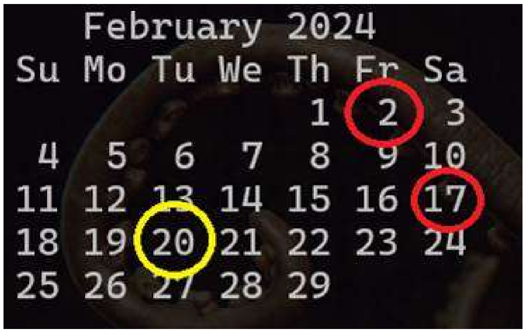


Figure 1: Global Honeypot Distribution

Host 171.238.59.73 became a focus of investigation due to the nature of a URL appearing in the "*Bottom 3 URLs*" search query on the *india24* web honeypot on February 20th, 2024.



```
    February 2024
Su Mo Tu We Th Fr Sa
             1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29
```

Figure 2: February Detections

Subsequent examination of available data revealed a series of layer 3 attempts to access TCP ports 4369 and 873, and layer 7 attempts to access two distinct URLs, across three geographic locations.

The potential for any of the services identified to be exploited if deployed and vulnerable is very high. The likelihood of services and software probed during this incident suffering a follow-on attack from the original scanner, given OSINT information provided later in this report, would be a near certainty.

Four Important Conversations

1. The first was an attempt to access a suspect URL on February 2nd at 01:32:09 UTC on the sensor eastus24.

2. The second attempted to access TCP port 4369 on sensor india24 on February 2nd at 02:00:14 UTC.

3. The third conversation was an attempt to access TCP port 873 on sensor israel24 on February 17th at 19:49:29 UTC.

4. The fourth and final conversation attempted to access a suspect URL on February 20th at 16:02:04 UTC on the sensor india24.

An overview of applications known to be associated with the suspicious URLs requested and their potential security implications are provided in the technical brief section of this document.

Executive Action Items:

- The presence of either of the products identified by the suspicious URLs in a production environment, particularly if exposed to the public Internet, represents a threat to operations. Such hosts and ports, if observed accessible from the Internet, suggest the need to audit the services and ports logged to identify the business application presenting the risk.

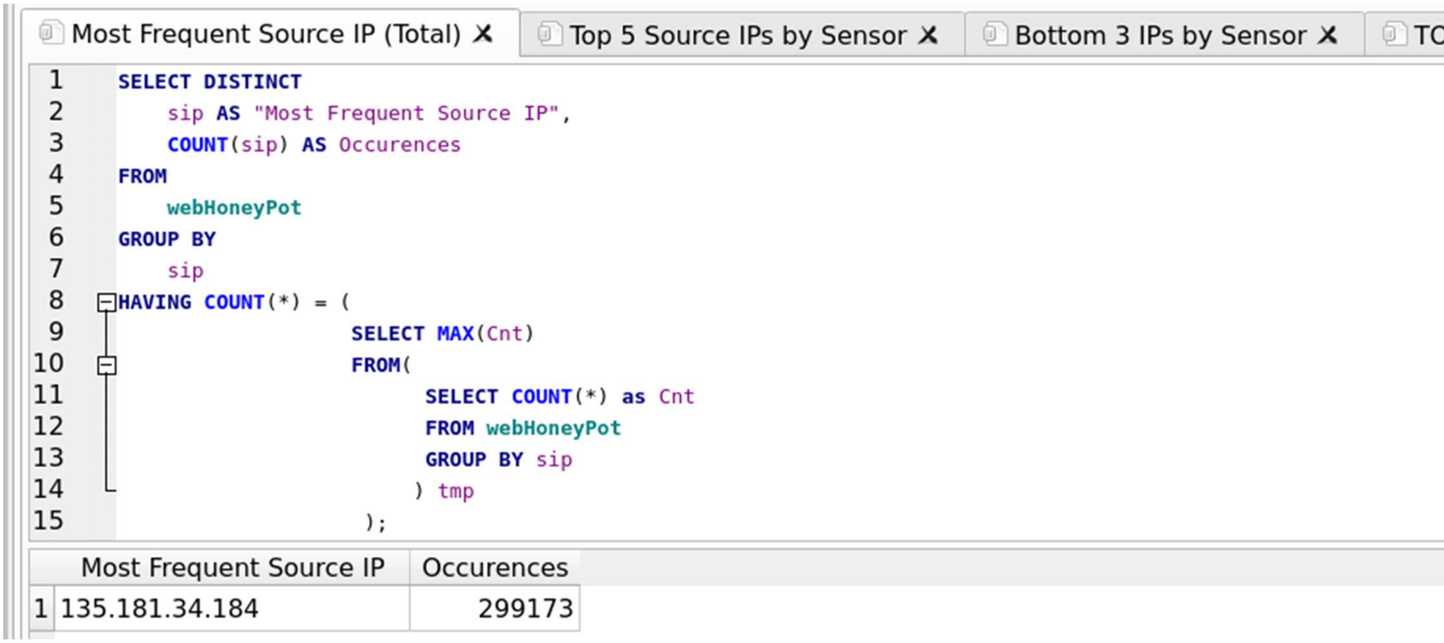
- Identified business applications need to be remediated with all current software updates and security fixes immediately. Any continued exposure to the Internet should have a significant justifiable business case for remaining. All use of such applications must also be mitigated through more a secure network deployment such as employing an application layer firewall between the system and the Internet.

Technical Brief and Analysis

A useful function of DShield Manager will import web honeypot logs for a given day, and open the webhoneypot.sqbpro project with SqliteBrowser to examine them. The project contains seven canned SQL queries for:

- Most Frequent Source IP (Total)
- Top 5 Source IPs by Sensor
- Bottom 3 IPs by Sensor
- Top 5 URLs by Sensor
- Bottom 3 URLS by Sensor
- Search Source IP
- Search for URL

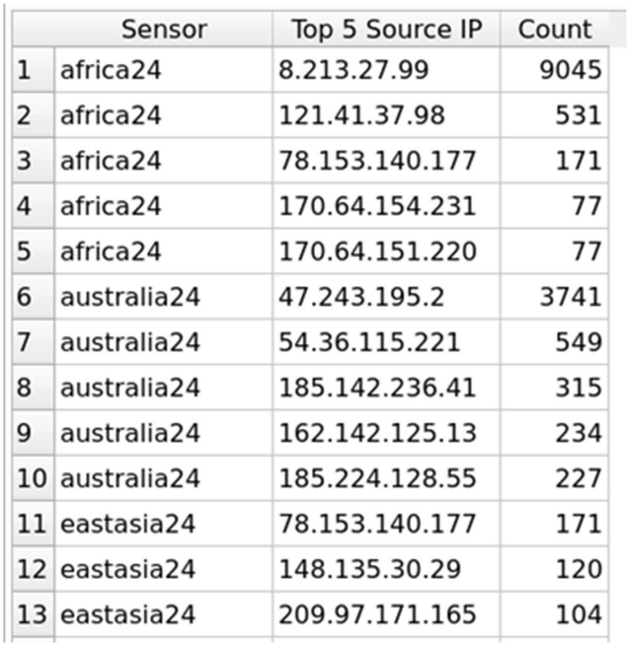
Executing the “Most Frequent Source IP (Total)” query, for example, reveals 135.181.34.184 is the most frequent Source IP with 299,173 occurrences in the database for the 20th of February.


Most Frequent Source IP (Total) ✕
Top 5 Source IPs by Sensor ✕
Bottom 3 IPs by Sensor ✕
TO

```
SELECT DISTINCT
    sip AS "Most Frequent Source IP",
    COUNT(sip) AS Occurences
FROM
    webHoneyPot
GROUP BY
    sip
HAVING COUNT(*) = (
                   SELECT MAX(Cnt)
                   FROM(
                         SELECT COUNT(*) as Cnt
                         FROM webHoneyPot
                         GROUP BY sip
                        ) tmp
                    );
```

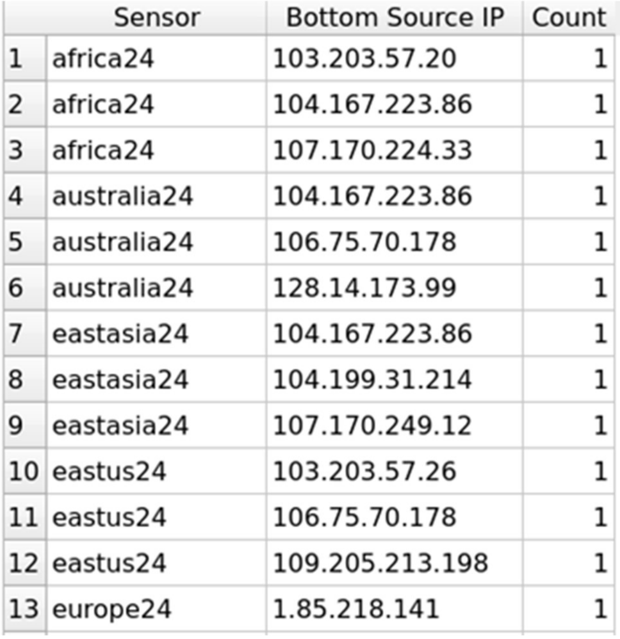
	Most Frequent Source IP	Occurences
1	135.181.34.184	299173

Figure 3: Most Frequent Source IP (Total) and Additional Query Tabs



	Sensor	Top 5 Source IP	Count
1	africa24	8.213.27.99	9045
2	africa24	121.41.37.98	531
3	africa24	78.153.140.177	171
4	africa24	170.64.154.231	77
5	africa24	170.64.151.220	77
6	australia24	47.243.195.2	3741
7	australia24	54.36.115.221	549
8	australia24	185.142.236.41	315
9	australia24	162.142.125.13	234
10	australia24	185.224.128.55	227
11	eastasia24	78.153.140.177	171
12	eastasia24	148.135.30.29	120
13	eastasia24	209.97.171.165	104

Figure 4: Top 5 Source IP by Sensor Query Results

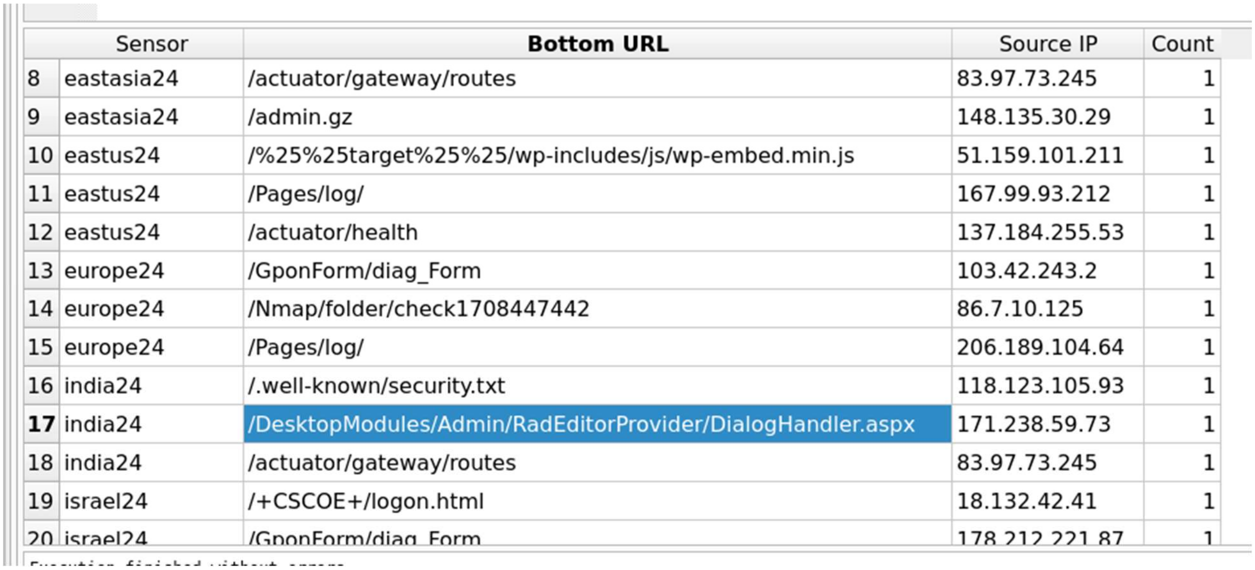


	Sensor	Bottom Source IP	Count
1	africa24	103.203.57.20	1
2	africa24	104.167.223.86	1
3	africa24	107.170.224.33	1
4	australia24	104.167.223.86	1
5	australia24	106.75.70.178	1
6	australia24	128.14.173.99	1
7	eastasia24	104.167.223.86	1
8	eastasia24	104.199.31.214	1
9	eastasia24	107.170.249.12	1
10	eastus24	103.203.57.26	1
11	eastus24	106.75.70.178	1
12	eastus24	109.205.213.198	1
13	europe24	1.85.218.141	1

Figure 5: Bottom 3 Source IP by Sensor Query Results

Executing the queries for Top and Bottom Source IP produces a count by Sensor.

The output of the query of the Bottom 3 URLs requested includes: `/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx`

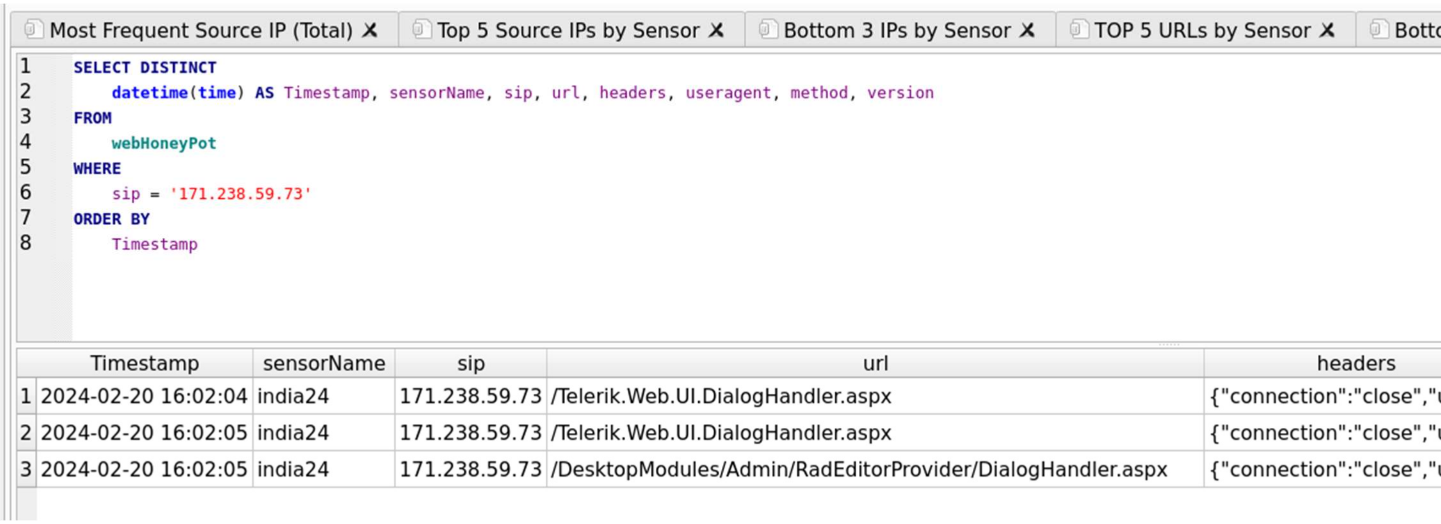


	Sensor	**Bottom URL**	Source IP	Count
8	eastasia24	/actuator/gateway/routes	83.97.73.245	1
9	eastasia24	/admin.gz	148.135.30.29	1
10	eastus24	/%25%25target%25%25/wp-includes/js/wp-embed.min.js	51.159.101.211	1
11	eastus24	/Pages/log/	167.99.93.212	1
12	eastus24	/actuator/health	137.184.255.53	1
13	europe24	/GponForm/diag_Form	103.42.243.2	1
14	europe24	/Nmap/folder/check1708447442	86.7.10.125	1
15	europe24	/Pages/log/	206.189.104.64	1
16	india24	/.well-known/security.txt	118.123.105.93	1
17	india24	/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx	171.238.59.73	1
18	india24	/actuator/gateway/routes	83.97.73.245	1
19	israel24	/+CSCOE+/logon.html	18.132.42.41	1
20	israel24	/GponForm/diag_Form	178.212.221.87	1

Execution finished without errors.

Figure 6: Bottom 3 URLs Query Output

Examining the Bottom 3 URLs by Sensor and then querying the database for the Source IP address of any URL of interest, often reveals additional communications sensors have had with the IP address and additional URLs to examine.



```
SELECT DISTINCT
    datetime(time) AS Timestamp, sensorName, sip, url, headers, useragent, method, version
FROM
    webHoneyPot
WHERE
    sip = '171.238.59.73'
ORDER BY
    Timestamp
```

	Timestamp	sensorName	sip	url	headers
1	2024-02-20 16:02:04	india24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx	{"connection":"close","
2	2024-02-20 16:02:05	india24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx	{"connection":"close","
3	2024-02-20 16:02:05	india24	171.238.59.73	/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx	{"connection":"close","

Figure 7: Search by Source IP Query Output

Output reveals additional requests for:
URL `/Telerik.Web.UI.DialogHandler.aspx`

What are these URLs?

Telerik

The URL `/Telerik.Web.UI.DialogHandler.aspx` is associated with Telerik UI for ASP.NET AJAX, a suite of UI controls for building rich ASP.NET web applications. This URL is specifically related to the DialogHandler class, which is the default dialog handler for Telerik dialogs. It requires Session State to be enabled. This handler doesn't need to physically exist as it is defined in the <httpHandlers> section in web.config.

However, it's important to note that there have been security concerns related to this URL. The Telerik.Web.UI.DialogHandler.aspx does not offer built-in authentication.

If the app is not public-facing or if a secure version of Telerik.Web.UI.dll is used then the app will be secured and an attacker won't be able to access or decrypt the handler [4] [5] [6].

DotNetNuke

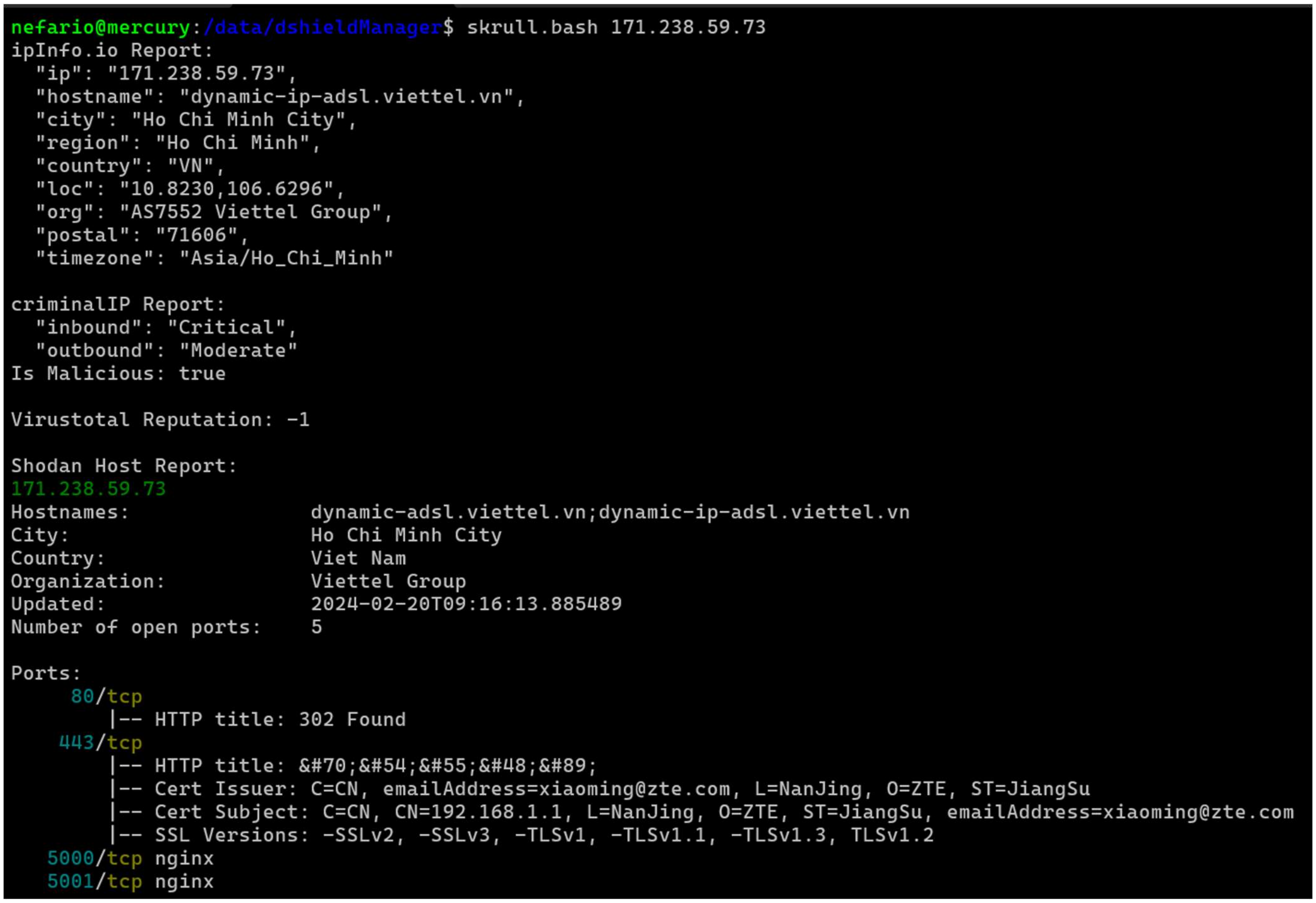
The URL /DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx is associated with the DotNetNuke (DNN) content management system.
It appears to be part of the RadEditorProvider, which is a module used for text editing within the DNN platform.

However, it's important to note that there have been security concerns related to this URL. Some users have reported instances of their DNN sites being hacked through this module. Attackers were reportedly able to upload .aspx files and other files within the DNN folder structure via this URL [7] [8] [9].

The data indicates the source IP is scanning for the presence of Telerik and DotNetNuke. While my Google-Fu hasn’t produced known exploits actively circulating in the wild, a lack of authentication in Telerik and suspected ability to upload .aspx files via DotNetNuke, would be cause for concern in a production environment. No additional connections to the URLs occur short of attempts to access them, so I consider this to be largely reconnaissance on the part of the attacker(s).

What is known about the attacker(s)?

Gatherering information from **ipinfo** [10], **criminalip** [11], **virustotal** [12] and **shodan** [13] sketches an answer the question. Further analysis will be provided in the OSINT section of this report.



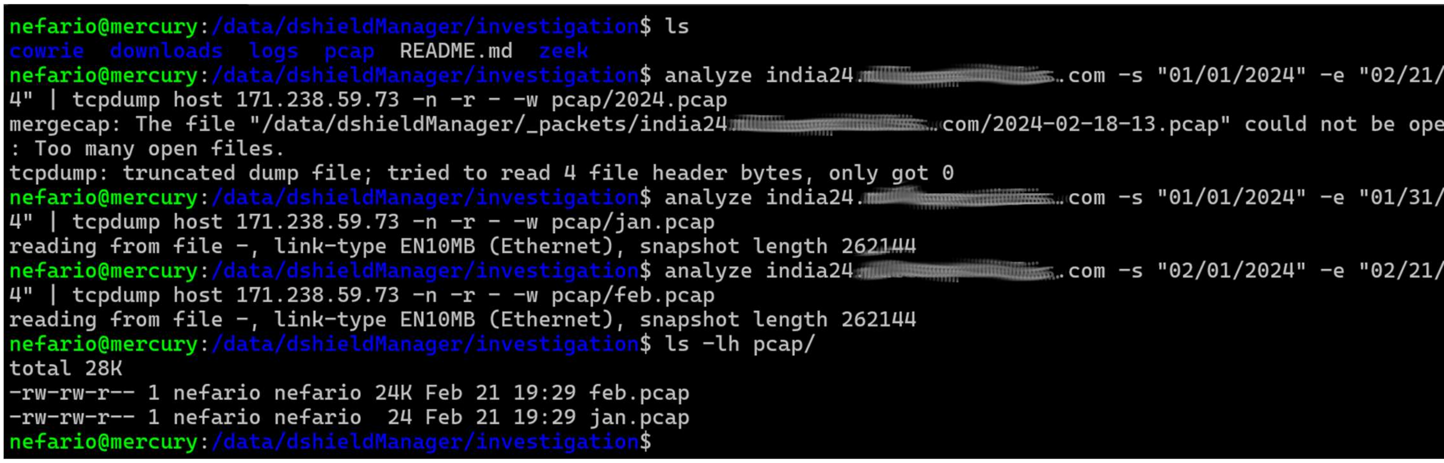
```
nefario@mercury:/data/dshieldManager$ skrull.bash 171.238.59.73
ipInfo.io Report:
  "ip": "171.238.59.73",
  "hostname": "dynamic-ip-adsl.viettel.vn",
  "city": "Ho Chi Minh City",
  "region": "Ho Chi Minh",
  "country": "VN",
  "loc": "10.8230,106.6296",
  "org": "AS7552 Viettel Group",
  "postal": "71606",
  "timezone": "Asia/Ho_Chi_Minh"

criminalIP Report:
  "inbound": "Critical",
  "outbound": "Moderate"
Is Malicious: true

Virustotal Reputation: -1

Shodan Host Report:
171.238.59.73
Hostnames:                dynamic-adsl.viettel.vn;dynamic-ip-adsl.viettel.vn
City:                     Ho Chi Minh City
Country:                  Viet Nam
Organization:             Viettel Group
Updated:                  2024-02-20T09:16:13.885489
Number of open ports:     5

Ports:
     80/tcp
        |-- HTTP title: 302 Found
    443/tcp
        |-- HTTP title: &#70;&#54;&#55;&#48;&#89;
        |-- Cert Issuer: C=CN, emailAddress=xiaoming@zte.com, L=NanJing, O=ZTE, ST=JiangSu
        |-- Cert Subject: C=CN, CN=192.168.1.1, L=NanJing, O=ZTE, ST=JiangSu, emailAddress=xiaoming@zte.com
        |-- SSL Versions: -SSLv2, -SSLv3, -TLSv1, -TLSv1.1, -TLSv1.3, TLSv1.2
   5000/tcp nginx
   5001/tcp nginx
```

Figure 8: Information Gathering Script.

Interest is in this host and any communications with it in the 2024 calendar year. Making use of the investigation folder for gathering pertinent copies of logs from cowrie, downloads, honeypot logs and packets keeps data together. It's also a scratch location for performing analysis with Zeek.

Extracting all packets for January/February of 2024 associated with the source IP employs a modified version of the `analyze` script, written and shared by Dave Hoelzer of Enclave Forensics as part of SANS SEC503: Network Monitoring and Threat Detection In-Depth [14], which he teaches.

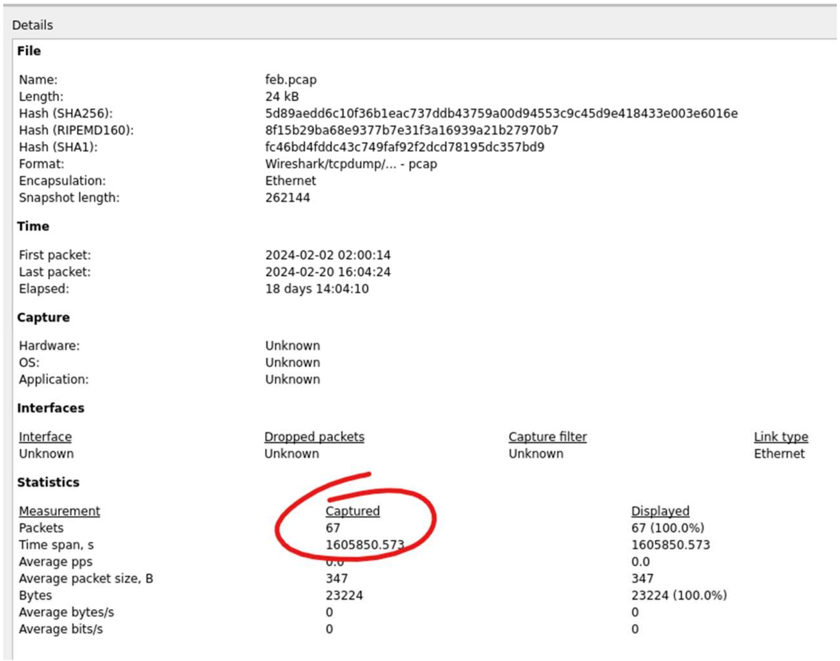
Packets are gathered by the script from the india24 sensor, piping them through `tcpdump` with a capture filter to limit packets specifically to host 171.238.59.73. Splitting it into two passes was required; one for January and another for February and saving the .pcap output avoids the error. File sizes for the two captures indicate the January capture is empty, verified with Wireshark.



```
nefario@mercury:/data/dshieldManager/investigation$ ls
cowrie  downloads  logs  pcap  README.md  zeek
nefario@mercury:/data/dshieldManager/investigation$ analyze india24.[redacted].com -s "01/01/2024" -e "02/21/
4" | tcpdump host 171.238.59.73 -n -r - -w pcap/2024.pcap
mergecap: The file "/data/dshieldManager/_packets/india24[redacted]com/2024-02-18-13.pcap" could not be ope
: Too many open files.
tcpdump: truncated dump file; tried to read 4 file header bytes, only got 0
nefario@mercury:/data/dshieldManager/investigation$ analyze india24.[redacted].com -s "01/01/2024" -e "01/31/
4" | tcpdump host 171.238.59.73 -n -r - -w pcap/jan.pcap
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
nefario@mercury:/data/dshieldManager/investigation$ analyze india24[redacted].com -s "02/01/2024" -e "02/21/
4" | tcpdump host 171.238.59.73 -n -r - -w pcap/feb.pcap
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
nefario@mercury:/data/dshieldManager/investigation$ ls -lh pcap/
total 28K
-rw-rw-r-- 1 nefario nefario 24K Feb 21 19:29 feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 19:29 jan.pcap
nefario@mercury:/data/dshieldManager/investigation$
```

Figure 9: Carving Packets for January and February

Opening the February capture file in Wireshark and examining the Capture File Properties from the Statistics Menu reports a total of 67 packets are included.



Details

File

Name:	feb.pcap
Length:	24 kB
Hash (SHA256):	5d89aedd6c10f36b1eac737ddb43759a00d94553c9c45d9e418433e003e6016e
Hash (RIPEMD160):	8f15b29ba68e9377b7e31f3a16939a21b27970b7
Hash (SHA1):	fc46bd4fddc43c749faf92f2dcd78195dc357bd9
Format:	Wireshark/tcpdump/... - pcap
Encapsulation:	Ethernet
Snapshot length:	262144

Time

First packet:	2024-02-02 02:00:14
Last packet:	2024-02-20 16:04:24
Elapsed:	18 days 14:04:10

Capture

Hardware:	Unknown
OS:	Unknown
Application:	Unknown

Interfaces

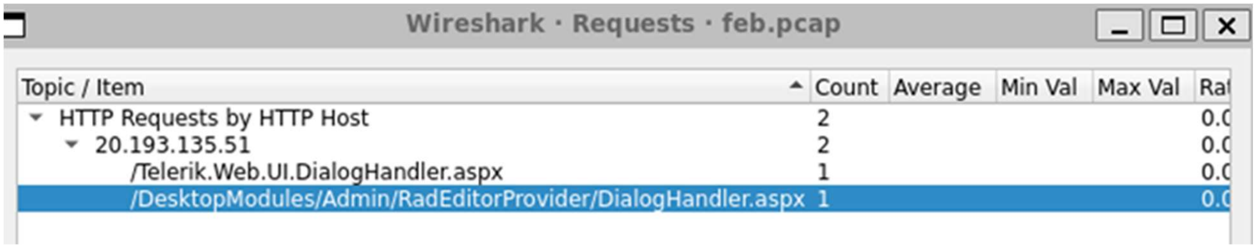
Interface	Dropped packets	Capture filter	Link type
Unknown	Unknown	Unknown	Ethernet

Statistics

Measurement	Captured	Displayed
Packets	67	67 (100.0%)
Time span, s	1605850.573	1605850.573
Average pps	0.0	0.0
Average packet size, B	347	347
Bytes	23224	23224 (100.0%)
Average bytes/s	0	0
Average bits/s	0	0

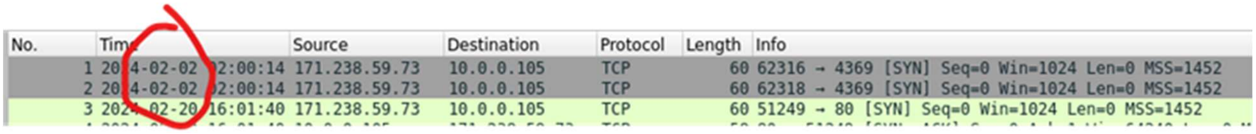
Figure 10: Wireshark Capture File Properties

Opening the Statistics Menu and selecting http, followed by Requests Output, will display all web requests contained in the .pcap file.



Topic / Item	Count	Average	Min Val	Max Val	Rat
HTTP Requests by HTTP Host	2				0.0
20.193.135.51	2				0.0
/Telerik.Web.UI.DialogHandler.aspx	1				0.0
/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx	1				0.0

Figure 11: Statistics Menu | http | Requests Output



```
No.   Time                 Source          Destination  Protocol  Length  Info
    1 20 4-02-02  2:00:14  171.238.59.73   10.0.0.105   TCP           60  62316 → 4369 [SYN] Seq=0 Win=1024 Len=0 MSS=1452
    2 20 4-02-02  2:00:14  171.238.59.73   10.0.0.105   TCP           60  62318 → 4369 [SYN] Seq=0 Win=1024 Len=0 MSS=1452
    3 2024-02-20 16:01:40  171.238.59.73   10.0.0.105   TCP           60  51249 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1452
```

Figure 12: The first two packets in the capture on February 2nd were initially overlooked in Wireshark analysis.

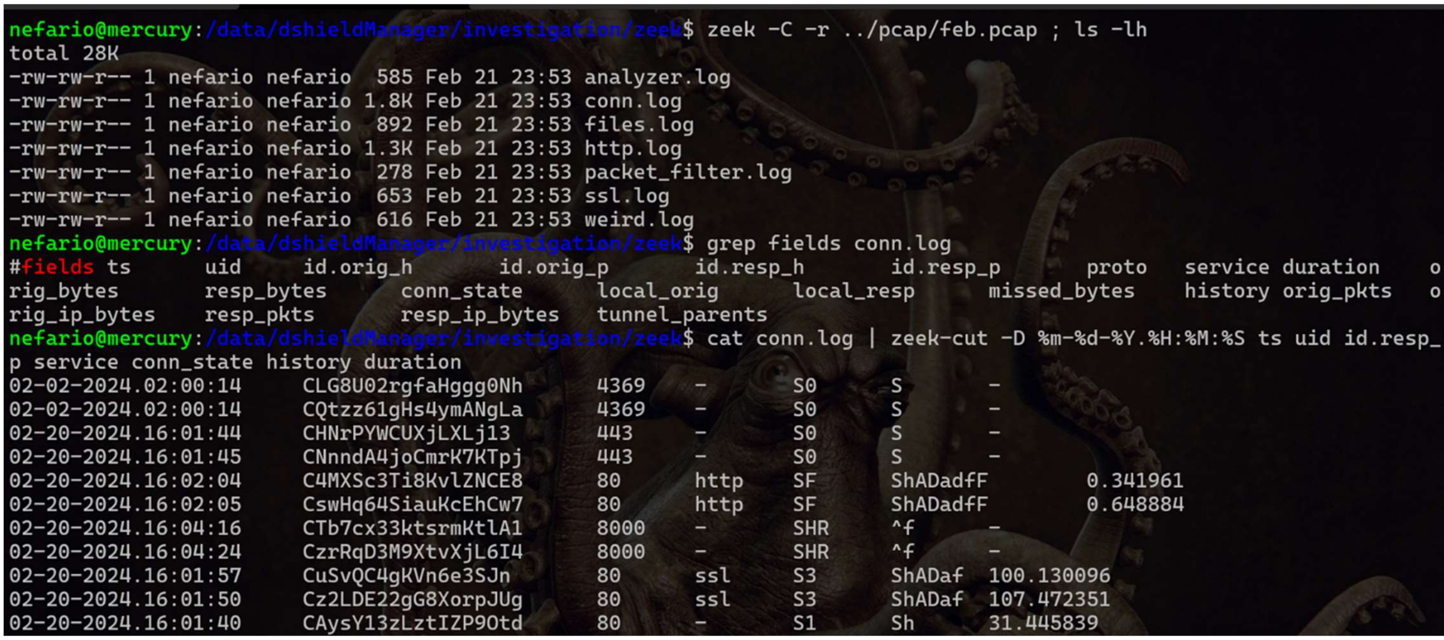
Turn now to another tool covered in-depth in SEC503, Zeek. *Zeek is a passive, open-source network traffic analyzer. Many operators use Zeek as a network security monitor (NSM) to support investigations of suspicious or malicious activity. Zeek also supports a wide range of traffic analysis tasks beyond the security domain, including performance measurement and troubleshooting* [15].

Begin moving into the zeek directory, executing `zeek` with the `-C` option to ignore checksum errors and `-r` to provide the path to read the feb.pcap file, followed by a directory listing for a look at the output files. Being interested largely in connections, focus on the `conn.log` file. It's often difficult to recall which fields each of the various zeek logs provide. Issue a quick `grep` for the line within the logfile containing fields.

Armed with these fields now pipe the log through the `zeek-cut` command to display information relevant to the investigation. The `-D %m-%d-%Y.%H:%M:%S` portion of the command formats the output of the ts (timestamp) field in a human-friendly MM-DD-YYYY.hh.mm.ss format.

```
cat conn.log | zeek-cut -D %m-%d-%Y.%H:%M:%S ts uid
id.resp_p service conn_state history duration
```

The command displays the timestamp, unique ID, responding host port, service, connection state, connection history, and duration of each connection.



```
nefario@mercury:/data/dshieldManager/investigation/zeek$ zeek -C -r ../pcap/feb.pcap ; ls -lh
total 28K
-rw-rw-r-- 1 nefario nefario  585 Feb 21 23:53 analyzer.log
-rw-rw-r-- 1 nefario nefario 1.8K Feb 21 23:53 conn.log
-rw-rw-r-- 1 nefario nefario  892 Feb 21 23:53 files.log
-rw-rw-r-- 1 nefario nefario 1.3K Feb 21 23:53 http.log
-rw-rw-r-- 1 nefario nefario  278 Feb 21 23:53 packet_filter.log
-rw-rw-r-- 1 nefario nefario  653 Feb 21 23:53 ssl.log
-rw-rw-r-- 1 nefario nefario  616 Feb 21 23:53 weird.log
nefario@mercury:/data/dshieldManager/investigation/zeek$ grep fields conn.log
#fields ts      uid     id.orig_h       id.orig_p       id.resp_h       id.resp_p       proto   service duration    o
rig_bytes       resp_bytes      conn_state      local_orig      local_resp      missed_bytes    history orig_pkts   o
rig_ip_bytes    resp_pkts       resp_ip_bytes   tunnel_parents
nefario@mercury:/data/dshieldManager/investigation/zeek$ cat conn.log | zeek-cut -D %m-%d-%Y.%H:%M:%S ts uid id.resp_
p service conn_state history duration
02-02-2024.02:00:14     CLG8U02rgfaHggg0Nh      4369    -       S0      S       -
02-02-2024.02:00:14     CQtzz61gHs4ymANgLa      4369    -       S0      S       -
02-20-2024.16:01:44     CHNrPYWCUXjLXLj13       443     -       S0      S       -
02-20-2024.16:01:45     CNnndA4joCmrK7KTpj      443     -       S0      S       -
02-20-2024.16:02:04     C4MXSc3Ti8KvlZNCE8      80      http    SF      ShADadfF        0.341961
02-20-2024.16:02:05     CswHq64SiauKcEhCw7      80      http    SF      ShADadfF        0.648884
02-20-2024.16:04:16     CTb7cx33ktsrmKtlA1      8000    -       SHR     ^f      -
02-20-2024.16:04:24     CzrRqD3M9XtvXjL6I4      8000    -       SHR     ^f      -
02-20-2024.16:01:57     CuSvQC4gKVn6e3SJn       80      ssl     S3      ShADaf  100.130096
02-20-2024.16:01:50     Cz2LDE22gG8XorpJUg      80      ssl     S3      ShADaf  107.472351
02-20-2024.16:01:40     CAysY13zLztIZP9Otd      80      -       S1      Sh      31.445839
```

Figure 13: Zeek-cut Output with Selected Fields

Make note of the first two connections from February 2nd to TCP Port 4369. Zeek brought them out of the haystack and more obvious.

The meanings of the output of *connection state* and *history* fields are listed below.

conn_state:

S0: Connection attempt seen, no reply.
S1: Connection established, not terminated.
SF: Normal establishment and termination. Note that this is the same symbol as for state S1. You can tell the two apart because for S1 there will not be any byte counts in the summary, while for SF there will be.
REJ: Connection attempt rejected.
S2: Connection established and close attempt by originator seen (but no reply from responder).
S3: Connection established and close attempt by responder seen (but no reply from originator).
RSTO: Connection established; originator aborted (sent a RST).
RSTR: Responder sent a RST.
RSTOS0: Originator sent a SYN followed by a RST, we never saw a SYN-ACK from the responder.

RSTRH: Responder sent a SYN ACK followed by a RST, we never saw a SYN from the (purported) originator.
SH: Originator sent a SYN followed by a FIN, we never saw a SYN ACK from the responder (hence the connection was “half” open).
SHR: Responder sent a SYN ACK followed by a FIN, we never saw a SYN from the originator.
OTH: No SYN seen, just midstream traffic (one example of this is a “partial connection” that was not later closed).

history:

S: The originator sent a SYN segment.
H: The responder sent a SYN ACK segment.
A: The originator sent an ACK segment.
D: The originator sent at least one segment with payload data. In this case, that was HTTP over TCP.
A: The responder replied with an ACK segment.
D: The responder replied with at least one segment with payload data.
F: The originator sent a FIN ACK segment.
F: The responder replied with a FIN ACK segment.

The initial connections from February 2nd are flagged S0 indicating a SYN was seen with no reply. This could be an ordinary SYN scan, but with our honeypot not answering on closed ports this isn’t a definite presumption.

TCP port 4369 is associated with multiple products. The Erlang port mapper daemon (EPMD) [16] is one such product. Erlang is a programming language used to build massively scalable soft real-time systems with requirements on high availability. Some of its uses are in telecoms, banking, e-commerce, computer telephony and instant messaging [17]. The description of industries making use of the product makes for a juicy target, which could explain the interest in the port.

Another product that makes use of TCP port 4369 is RabbitMQ. RabbitMQ is lightweight and easy to deploy on premises and in the cloud. It supports multiple messaging protocols and streaming. RabbitMQ can be deployed in distributed and federated configurations to meet high-scale, high-availability requirements. [18] This description also makes for a juicy target: the softer underbelly of federated systems and messaging.

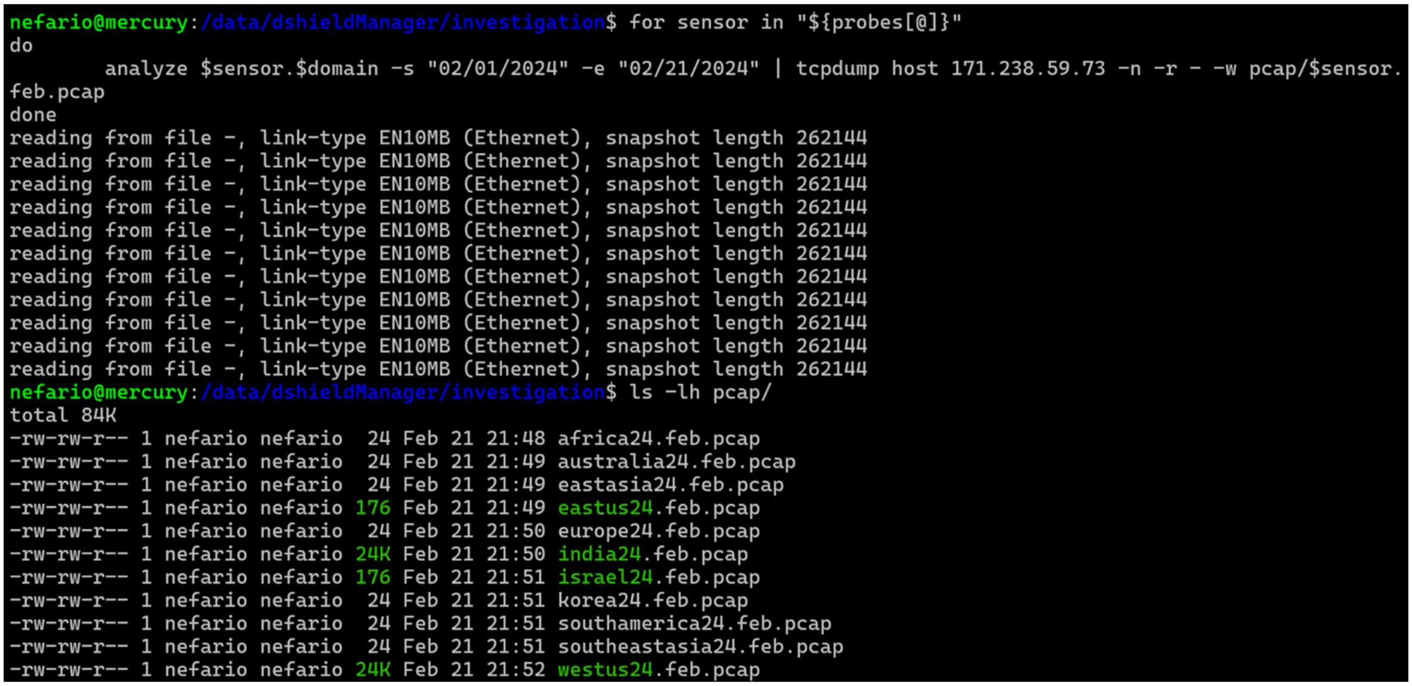
There remains a third product that makes use of TCP port 4369, albeit by virtue of an embedded instance of RabbitMQ: SolarWinds. SolarWinds Server & Application Monitor (SAM) uses the following ports:

> 4369 TCP - RabbitMQ messaging (EMPD) [19]
> 5671 TCP - RabbitMQ messaging (AMQP over TLS/SSL)
> 5672 TCP - RabbitMQ messaging (AMQP unencrypted backup port)
> 25672 TCP - RabbitMQ messaging (Erlang distribution)

`/data/dshieldManager/bin/VARS` includes a bash array of the honeypots utilized frequently in tasks or functions associated with “all sensors”. It is also available for use at the command line once sourced via the alias `dshield`.

```
probes=(africa24 australia24 eastasia24 eastus24 europe24
india24 israel24 korea24 southamerica24 southeastasia24
westus24)
```

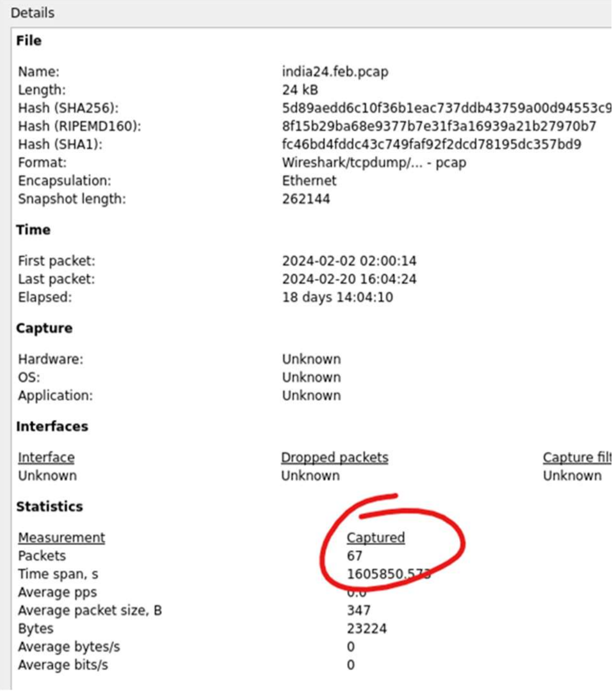
Executing a for loop wrapped around the analyze script allows fairly expeditious export of packets for each sensor for the month of February. A directory listing reveals a total of four files large enough to contain output: eastus24, india24, israel24 and westus24. Right away the file sizes suggest scripted or repetitive behavior.



```
nefario@mercury:/data/dshieldManager/investigation$ for sensor in "${probes[@]}"
do
        analyze $sensor.$domain -s "02/01/2024" -e "02/21/2024" | tcpdump host 171.238.59.73 -n -r - -w pcap/$sensor.
feb.pcap
done
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
reading from file -, link-type EN10MB (Ethernet), snapshot length 262144
nefario@mercury:/data/dshieldManager/investigation$ ls -lh pcap/
total 84K
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:48 africa24.feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:49 australia24.feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:49 eastasia24.feb.pcap
-rw-rw-r-- 1 nefario nefario 176 Feb 21 21:49 eastus24.feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:50 europe24.feb.pcap
-rw-rw-r-- 1 nefario nefario 24K Feb 21 21:50 india24.feb.pcap
-rw-rw-r-- 1 nefario nefario 176 Feb 21 21:51 israel24.feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:51 korea24.feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:51 southamerica24.feb.pcap
-rw-rw-r-- 1 nefario nefario  24 Feb 21 21:51 southeastasia24.feb.pcap
-rw-rw-r-- 1 nefario nefario 24K Feb 21 21:52 westus24.feb.pcap
```

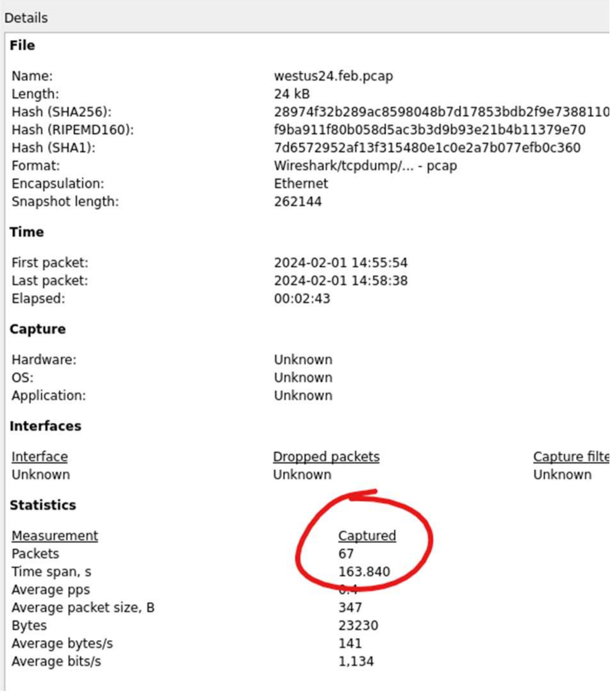
Figure 14: Executing for loop on all sensors to retrieve February packets.

Opening the two larger files with Wireshark, india24.feb.pcap and west24.feb.pcap, and examining the file properties reveals more of the same, 67 packets each.



```
Details
File
 Name:                   india24.feb.pcap
 Length:                 24 kB
 Hash (SHA256):          5d89aedd6c10f36b1eac737ddb43759a00d94553c9
 Hash (RIPEMD160):       8f15b29ba68e9377b7e31f3a16939a21b27970b7
 Hash (SHA1):            fc46bd4fddc43c749faf92f2dcd78195dc357bd9
 Format:                 Wireshark/tcpdump/... - pcap
 Encapsulation:          Ethernet
 Snapshot length:        262144
Time
 First packet:           2024-02-02 02:00:14
 Last packet:            2024-02-20 16:04:24
 Elapsed:                18 days 14:04:10
Capture
 Hardware:               Unknown
 OS:                     Unknown
 Application:            Unknown
Interfaces
 Interface               Dropped packets              Capture filt
 Unknown                 Unknown                      Unknown
Statistics
 Measurement                   Captured
 Packets                       67
 Time span, s                  1605850.573
 Average pps                   0.0
 Average packet size, B        347
 Bytes                         23224
 Average bytes/s               0
 Average bits/s                0
```

Figure 15: Wireshark Capture File Properties india24.feb.pcap



```
Details
File
 Name:                   westus24.feb.pcap
 Length:                 24 kB
 Hash (SHA256):          28974f32b289ac8598048b7d17853bdb2f9e7388110
 Hash (RIPEMD160):       f9ba911f80b058d5ac3b3d9b93e21b4b11379e70
 Hash (SHA1):            7d6572952af13f315480e1c0e2a7b077efb0c360
 Format:                 Wireshark/tcpdump/... - pcap
 Encapsulation:          Ethernet
 Snapshot length:        262144
Time
 First packet:           2024-02-01 14:55:54
 Last packet:            2024-02-01 14:58:38
 Elapsed:                00:02:43
Capture
 Hardware:               Unknown
 OS:                     Unknown
 Application:            Unknown
Interfaces
 Interface               Dropped packets              Capture filte
 Unknown                 Unknown                      Unknown
Statistics
 Measurement                   Captured
 Packets                       67
 Time span, s                  163.840
 Average pps                   0.4
 Average packet size, B        347
 Bytes                         23230
 Average bytes/s               141
 Average bits/s                1,134
```

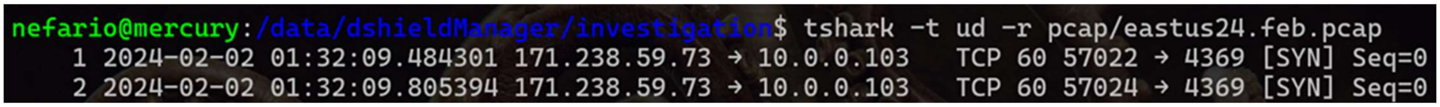
Figure 16: Wireshark Capture File Properties westus24.feb.pcap

Tshark can simplify this count from the command line:

```
for X in $(ls pcap/); do echo $X; tshark -r pcap/$X | wc -l;
done
```

What about the captures with only two packets?

Executing `tshark -t ud -r pcap/eastus24.feb.pcap` quickly reveals again an attempt to connect to TCP port 4369. Utilizing the `-t ud` flag outputs timestamps in absolute UTC format. These occurred on February 2nd with roughly thirty minutes passing between the eastus24 honeypot and the india24 honeypot occurrences.



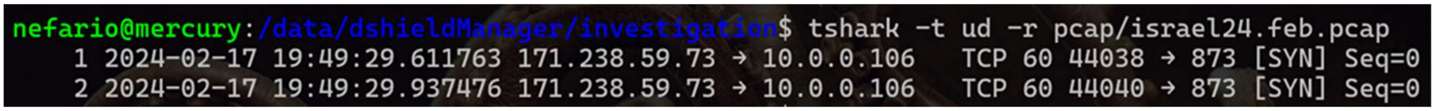
```
nefario@mercury:/data/dshieldManager/investigation$ tshark -t ud -r pcap/eastus24.feb.pcap
    1 2024-02-02 01:32:09.484301 171.238.59.73 → 10.0.0.103   TCP 60 57022 → 4369 [SYN] Seq=0
    2 2024-02-02 01:32:09.805394 171.238.59.73 → 10.0.0.103   TCP 60 57024 → 4369 [SYN] Seq=0
```

Figure 17: Tshark Output

The isreal24.feb.pcap file:

```
tshark -t ud -r pcap/israel24.feb.pcap
```

These packets are more unique, occurring on the 17th of February, bound for TCP port 873.



```
nefario@mercury:/data/dshieldManager/investigation$ tshark -t ud -r pcap/israel24.feb.pcap
    1 2024-02-17 19:49:29.611763 171.238.59.73 → 10.0.0.106   TCP 60 44038 → 873 [SYN] Seq=0
    2 2024-02-17 19:49:29.937476 171.238.59.73 → 10.0.0.106   TCP 60 44040 → 873 [SYN] Seq=0
```

Figure 18: Tshark Output

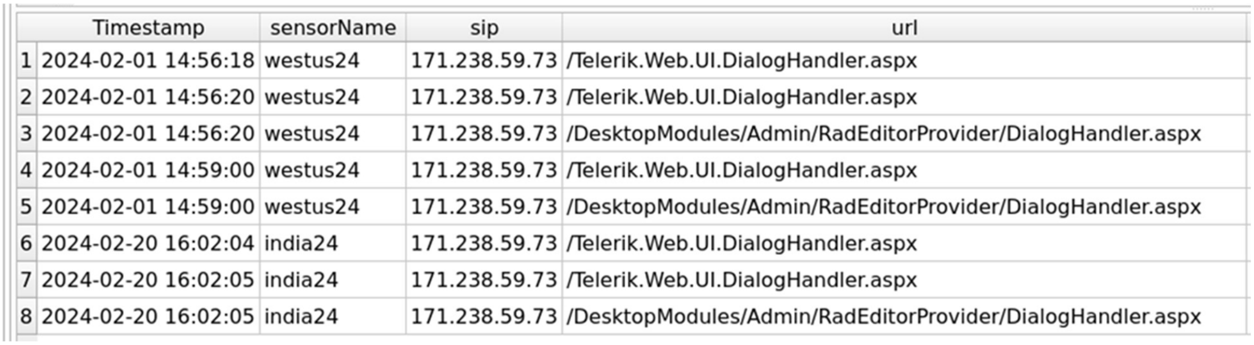
IANA associates TCP port 873 with the rsync service (as well as UDP port 873) and three well know products utilize it.

1. QNAP NAS uses the following ports:
   Web server: 80,8081 TCP and 443,8080 TCP (web admin)
   FTP/SFTP/SSH: 20,21,22 TCP and 13131 TCP (telnet)
   Remote Replication: 873,8899 TCP
   VPN server: 1723 TCP (PPTP), 1194 UDP (OpenVPN)
   CloudLink: port 20001 UDP (optional, only required for access without manual port forwarding) [20].

2. The ANTlabs InnGate devices do not require authentication for rsync sessions, which allows remote attackers to read or write to arbitrary files via TCP traffic on port 873 [21].

3. F5 BIG-IP could allow a remote attacker to execute arbitrary code on the system, caused by an error within the ConfigSync Access Control Handler component. By connecting to the rsync service on TCP port 873, an attacker could exploit this vulnerability to gain read or write access to the system and execute arbitrary code on the system with root privileges [22, 23, 24].

Searching for all of 2024 finds the February requests on sensor westus24.



	Timestamp	sensorName	sip	url
1	2024-02-01 14:56:18	westus24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx
2	2024-02-01 14:56:20	westus24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx
3	2024-02-01 14:56:20	westus24	171.238.59.73	/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx
4	2024-02-01 14:59:00	westus24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx
5	2024-02-01 14:59:00	westus24	171.238.59.73	/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx
6	2024-02-20 16:02:04	india24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx
7	2024-02-20 16:02:05	india24	171.238.59.73	/Telerik.Web.UI.DialogHandler.aspx
8	2024-02-20 16:02:05	india24	171.238.59.73	/DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx

Figure 19: Additional Requests Discovered on January 1st.

The `URL /DesktopModules/Admin/RadEditorProvider/DialogHandler.aspx` has been seen by other DShield honeypots.

A slight increase in daily submissions occurred in January 2024, creeping up in number of occurrences in February.

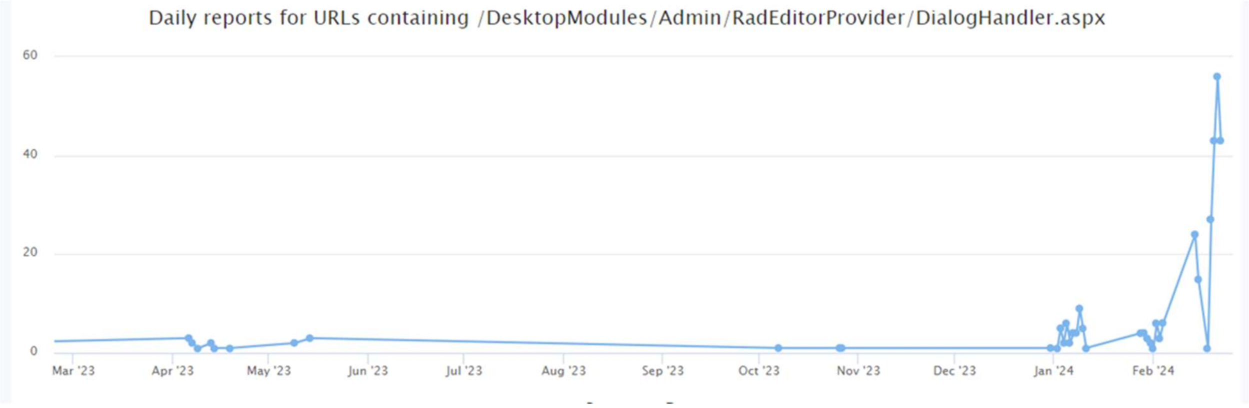


Figure 20 [25]

The URL `/Telerik.Web.UI.DialogHandler.aspx` has also been seen in the DShield network.

There was a spike in May of 2023. Its current daily reporting rate is quite low.

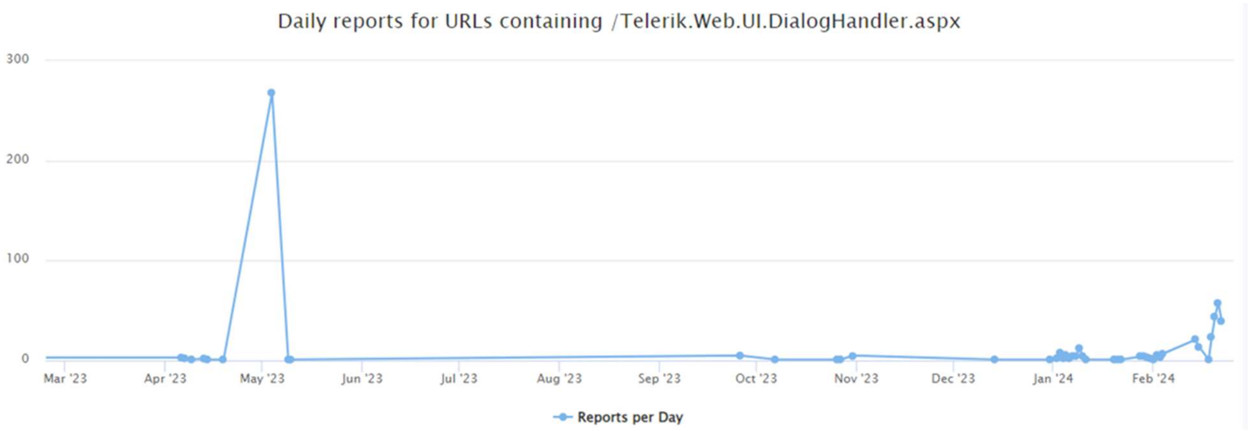


Figure 21 [25]

Historical data for TCP port 4369 looks suspiciously like dates of known SolarWinds exploitations in the wild.

Daily sources, reports and targets have fallen to consistent levels.

TCP port 873 is far more actively scanned, likely due to its association with rsync and QNAP.

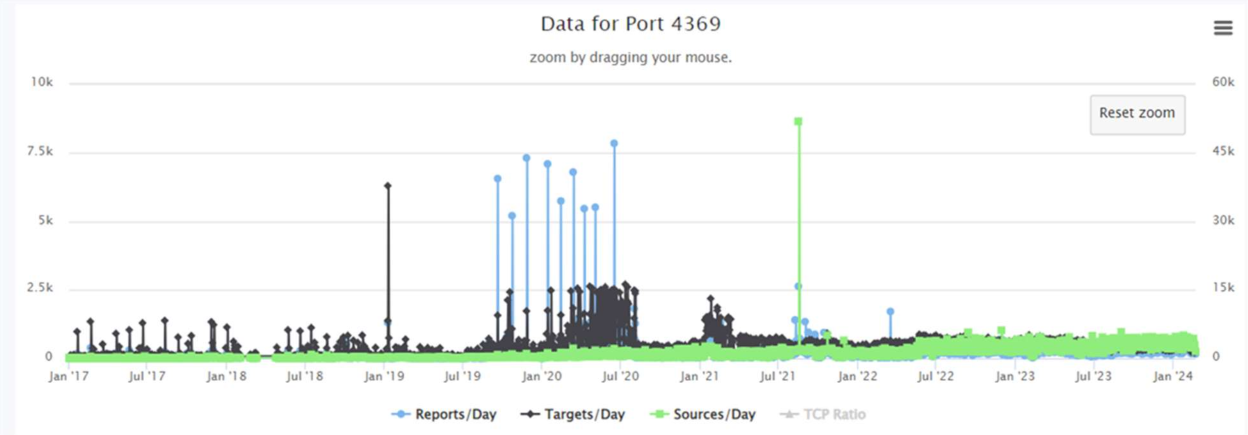


Figure 22 [25]

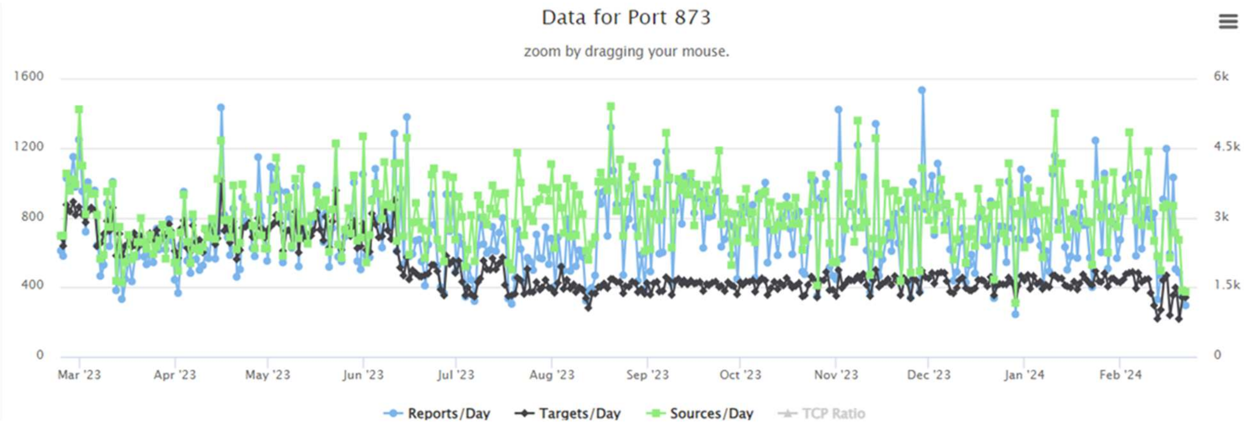


Figure 23 [25]

QNAP NAS allows installation of RabbitMQ in a Docker container. [26]

This substantiates a correlation between both the suspicious URLs and TCP Ports being scanned – it's a safe hypothesis now that the attacker is searching for vulnerable instances of Synology QNAP NAS units connected to the Internet.

OSINT

Gathering information about the attacker from publicly available sources can aid in assessing associated risks.

A good place to start really digging into a host that's likely up to no good is Virus Total. Their graphing tool is simply amazing. Beginning with a search for the IP address 171.238.59.73 in Vietnam, double-clicking the host reveals and loads associations. Double-clicking on those associated objects will in turn load their associations.

The further down the rabbit hole one chases information, the wider that funnel seems to get. In very broad strokes the honeypots have been scanned by an IP address in Vietnam that resolves hostnames through Cloudflare to some 800 or more malicious domains. The analysis has stumbled on a thriving criminal enterprise.

A snapshot of the graph will be on the next page. It is also shared on virus total allowing anyone to access it and explore [27].

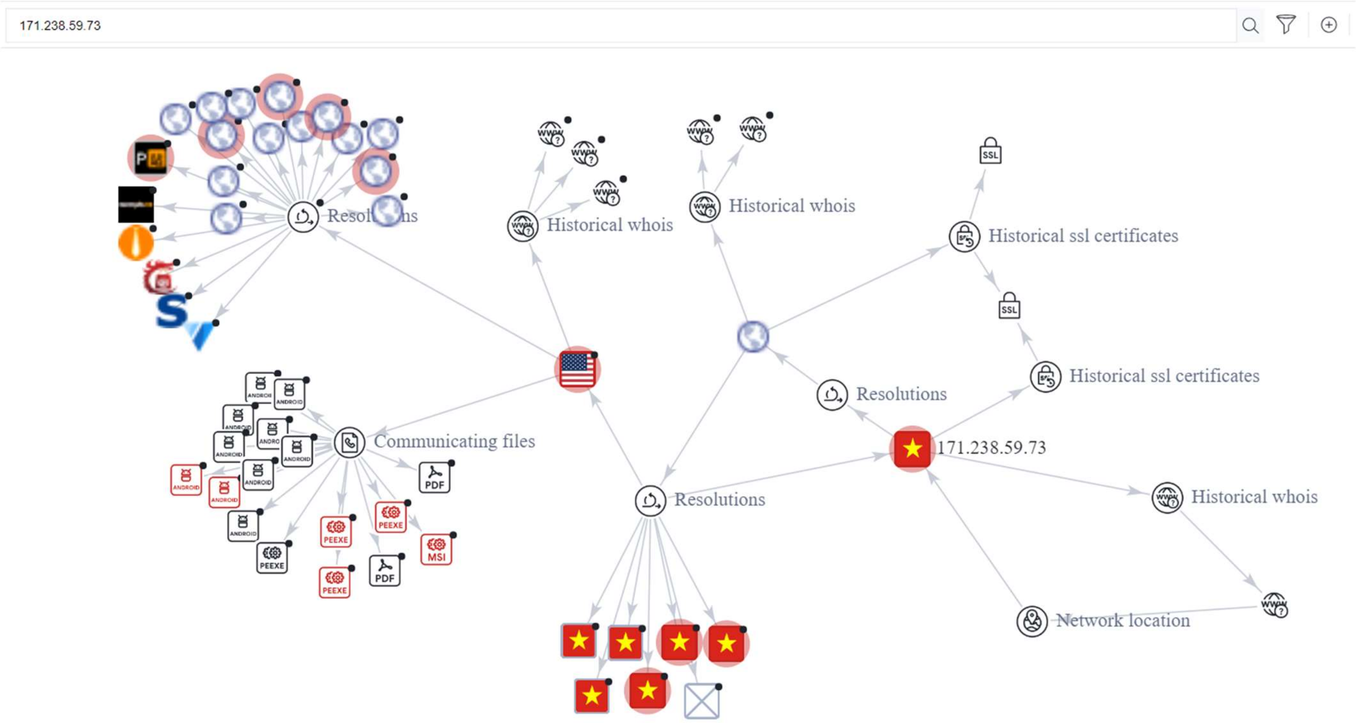
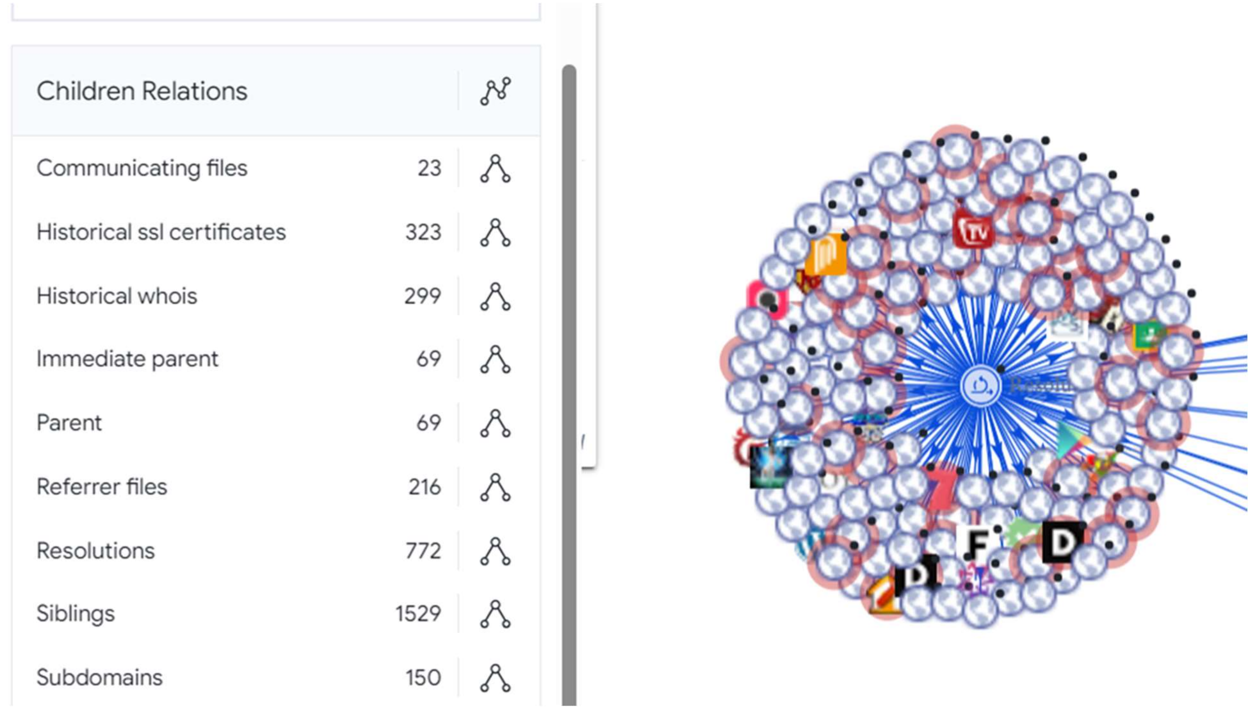


Figure 24: Virus Total Graph

The “Resolutions” node in the top left portion of the graph represents several websites that are hosting the files displayed by the “Communicating files” node, many of which are flagged as dangerous.

The initial load of the Resolutions node, however, did not load every item available because the number of child objects is immense.

Repeatedly loading the node until exceeding the Virus Total quota for the day produced the results displayed here.

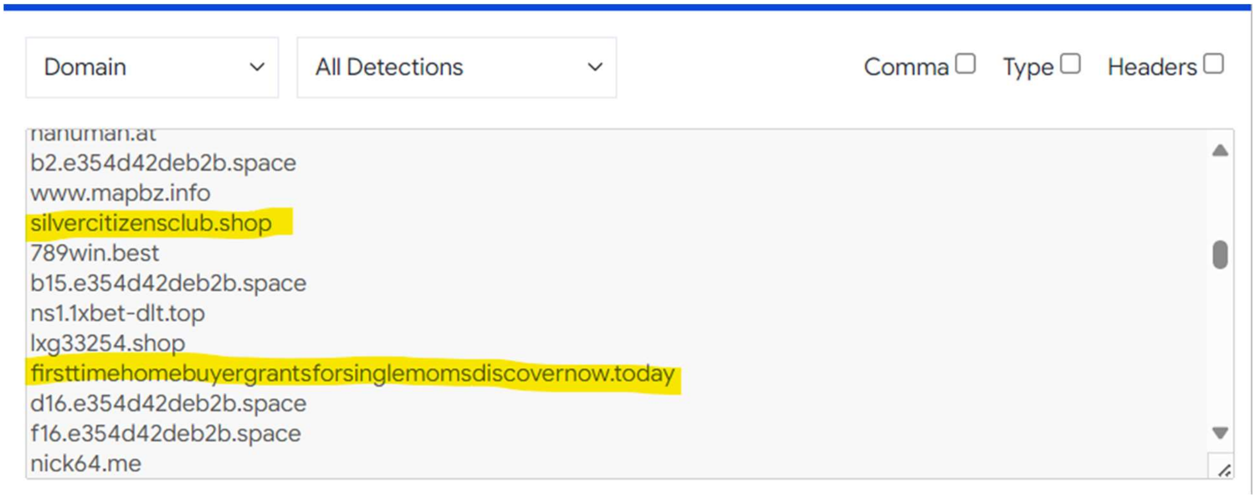


Children Relations	
Communicating files	23
Historical ssl certificates	323
Historical whois	299
Immediate parent	69
Parent	69
Referrer files	216
Resolutions	772
Siblings	1529
Subdomains	150

Figure 25: Virus Total Relationships

A single attempt to access a URL on a honeypot left breadcrumbs back to a substantial criminal enterprise!

A couple of the domains identified indicate the type of innocent people being targeted. In the highlighted examples we an presume elderly shoppers and single mothers that may have a down payment saved up to purchase a home – in other words, “something of value to steal”.



```
hanuman.at
b2.e354d42deb2b.space
www.mapbz.info
silvercitizensclub.shop
789win.best
b15.e354d42deb2b.space
ns1.1xbet-dlt.top
lxg33254.shop
firsttimehomebuyergrantsforsinglemomsdiscovernow.today
d16.e354d42deb2b.space
f16.e354d42deb2b.space
nick64.me
```

Figure 26: Dangerous hosted domains.

Parting Thoughts

My internship with the Internet Storm Center, though twenty weeks in length, passed far more quickly than I imagined it could. This should NOT have surprised me – on day one mentors warn that it passes quickly.

Not everything I witnessed crossing my honeypots rose to the level of authoring an observation paper as part of my studies, and my studies were not limited solely to the operation of the honeypots. My own journey through the BACS curriculum has been an absolute whirlwind of refreshing and reinforcing the skillsets I need to remain relevant in the final decades of my career while giving me the confidence to avoid imposter syndrome. I take pride in knowing my colleagues who hold GIAC certifications from SANS have worked just as hard to earn them as I have, and that they can be expected to provide trustworthy subject-matter-expert opinions within their respective fields. I've attended SANS training since 1999.

I am well versed in many flavors/distributions of Linux having used it in one form or another as my daily driver for a quarter century, while also having cut my teeth on products from Novell and Microsoft earning both CNE and MCSE certifications in the early 2000s. During my career I've operated my own security consulting practice, built and sold a successful wireless provider, and worked for the United States Department of Agriculture, where I witnessed one of the most famous supply chain attacks in real time.

My last formal experience with programming was a pair of 100 and 200 level courses in Java at the University of DeVry some twenty three years ago. Despite the Java my mindset has always left me clutching the terra firma of structured programming. Three courses and instructors at SANS changed that for me over the course of six short months, dramatically. SEC573: Automating Information Security with Python, taught by Mark Baggett, literally changed the way I think about code. While writing DShield Manger I often found myself reaching for Python first, regardless of the OS I'm working in.

SEC503: Network Monitoring and Threat Detection In-Depth, taught by David Hoelzer, genuinely intimidated me. I am now significantly more comfortable examining and converting data types that I have long begged others for help with. What's more, I understand what I'm looking at most of the time.

And finally, SEC505: Securing Windows and PowerShell Automation taught by Jason Fossen, brought the fullness of object oriented programming to fruition. I personally do not consider this course to be an elective – it was instead a self-imposed requisite course. If you work with Microsoft products this course is far more compelling and informative than you may expect. Most importantly, Jason opened my eyes to the just how incredibly powerful object-oriented PowerShell is!

The internship provided me with innumerable opportunities to put new skillsets to the test daily while I also found myself falling into a lifecycle of improvement of both tools and tactics on a similar schedule. I also found, in addition to outstanding mentor support, a thriving community of practitioners putting in the work and sharing results.

Completing remaining coursework and matriculating with a new degree is not the destination. It's merely a newly poured concrete renovation of the foundation of the house of my continuing security career.

References:

[1] DShield Honeypot - https://www.dshield.org/tools/honeypot/

[2] SANS BACS Program - https://www.sans.edu/cyber-security-programs/bachelors-degree/

[3] DShield Manager - https://github.com/1on1security/DShieldManager

[4] https://docs.telerik.com/devtools/aspnet-ajax/api/server/Telerik.Web.UI/DialogHandler

[5] https://docs.telerik.com/devtools/aspnet-ajax/getting-started/installation/web-config-settings-overview

[6] https://www.telerik.com/forums/telerik-web-ui-dialoghandler-aspx-what-directory

[7] https://dnncommunity.org/forums/aft/684

[8] https://www.dnnsoftware.com/answers/potential-security-issues-in-dnn-922178

[9] https://pastebin.com/L8uBeNaS

[10] ipinfo – https://ipinfo.io/

[11] CriminalIP – https://www.criminalip.io/

[12] Virus Total – https://www.virustotal.com/

[13] Shodan – https://www.shodan.io/

[14] SANS SEC503 – https://www.sans.org/cyber-security-courses/network-monitoring-threat-detection/

[15] https://docs.zeek.org/en/master/about.html#what-is-zeek

[16] https://wiki.wireshark.org/EPMD

[17] https://www.erlang.org/

[18] https://www.rabbitmq.com/

[19] https://www.speedguide.net/port.php?port=4369

[20] https://www.speedguide.net/port.php?port=873

[21] https://www.cve.org/CVERecord?id=CVE-2015-0932

[22] https://exchange.xforce.ibmcloud.com/vulnerabilities/95624

[23] https://www.exploit-db.com/exploits/34465

[24] https://www.cve.org/CVERecord?id=CVE-2014-2927

[25] https://isc.sans.edu

[26] https://blog.iannelson.uk/how-to-install-rabbitmq-server-in-docker-on-a-synology-nas/

[27] Virus Total Shared Graph – https://www.virustotal.com/graph/embed/g4ac6c9f0944c49bb91b8ad24dd8643f496fb870d8c2543168762e4799518093a?theme=light