The first step was identifying a suspicious domain flagged by threat intelligence tools for potentially malicious activity. After the domain was marked, related indicators like file hashes, IP addresses, and URLs were gathered for further investigation.

Figure 1. URLScan.io

Indicators Panel

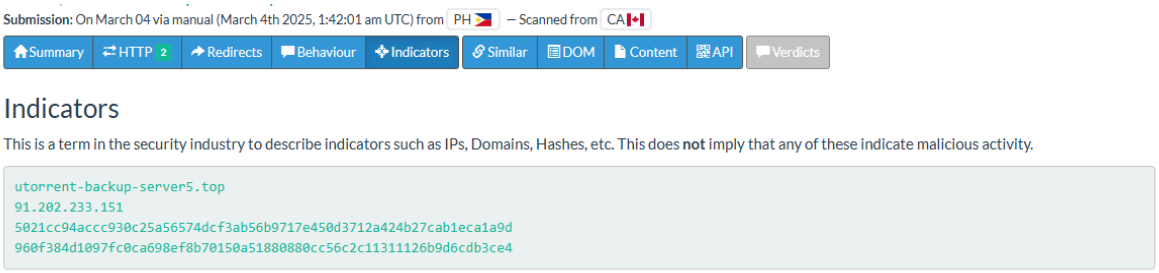

Figure 2. Indicators Page.

The file hash associated with the domain was then used as a pivot point for more analysis.

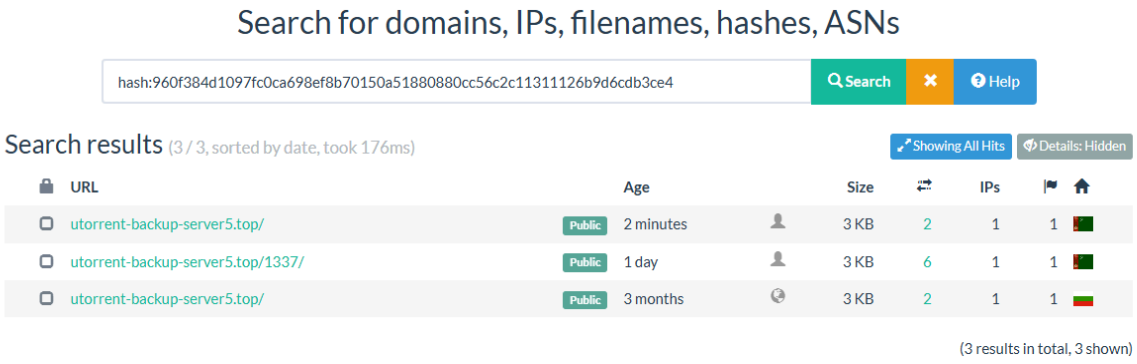

Figure 3. Search by Hash

64 hits for the same IP but different domain

Search

URL	Age	Size		IPs		
91.202.233.151	13 hours	3 KB	2	1	1	
91.202.233.151/1337Traget/1337X-1.exe	a day		1	1	1	
91.202.233.151/1337/TORRENTOLD-1.exe	a day		1	1	1	
utorrent-backup-server4.top	3 days	3 KB	2	1	1	
utorrent-backup-server4.top/1337/TORRENTOLD-1.exe	16 days		1	1	1	
utorrent-server-api.cc/1337/TORRENTOLD-1.exe	16 days		1	1	1	
update-checker-status.cc/1337/TORRENTOLD-1.exe	16 days		1	1	1	
win-network-checker.cc/1337/TORRENTOLD-1.exe	17 days		1	1	1	
fox-news-checker.cc/1337/TORRENTOLD-1.exe	17 days		1	1	1	
update-checker-status.cc/1337/	17 days	3 KB	6	1	1	

Figure 4. Search by the Other Hash

Next, attention was turned to the IP addresses linked to the domain. This helped identify any additional hosts or infrastructure connected to the malicious activity, broadening the scope of the investigation.

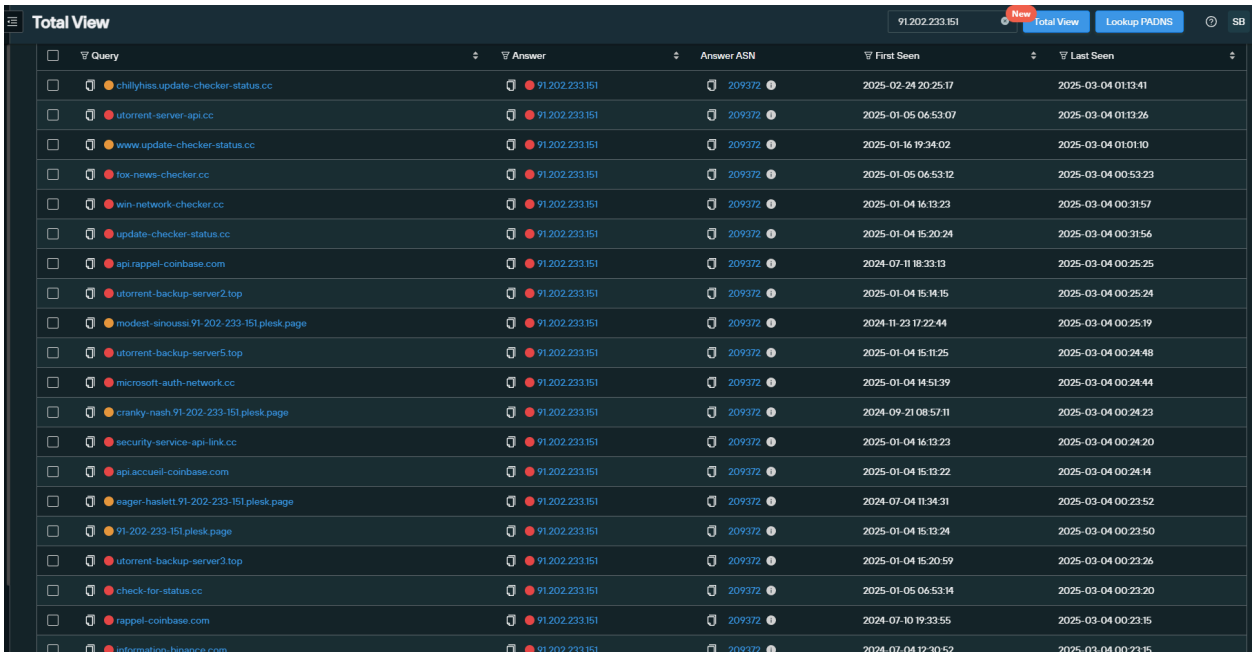

Figure 5. Check IP on SilentPush

This Python script checks the reachability and status codes of domains and their associated IP addresses. It first resolves a domain to its corresponding IP address using the `dns.resolver` module. Then, it generates random user-agent headers to mimic browser requests in the `get_robust_headers` function, which helps avoid detection by security measures. The `get_http_response` function makes HTTP and HTTPS requests to check if the domain and IP are reachable by accessing a specific directory (`/1337/`) on both the domain and its resolved IP address, capturing the HTTP status codes. The `check_domain_reachability` function handles the entire process for each domain, ensuring both the domain and IP are tested for reachability. The `check_domains_from_csv` function reads domains from a CSV file and performs these checks concurrently using `ThreadPoolExecutor` for faster execution. After the checks, the script outputs two sections: a summary of all domains with their IP addresses and status codes, and a list of reachable domains with the corresponding HTTP/HTTPS status codes. The script is designed for efficient monitoring or security operations, particularly in environments where rapid domain status assessments are necessary.

```
import dns.resolver
import requests
import pandas as pd
import random
from concurrent.futures import ThreadPoolExecutor, as_completed

def resolve_domain(domain):
```

```
    try:
        resolver = dns.resolver.Resolver()
        resolver.nameservers = ['8.8.8.8', '8.8.4.4']
        result = resolver.resolve(domain, 'A')
        return result[0].to_text()
    except (dns.resolver.NoAnswer, dns.resolver.NXDOMAIN):
        return None

# Generate headers to mimic a browser
def get_robust_headers():
    headers = {
        'User-Agent': random.choice([
            "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124
Safari/537.36",
            "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Edge/91.0.864.59 Safari/537.36",
            "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Firefox/89.0 Safari/537.36",
            "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124
Safari/537.36",
            "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0"
        ]),
        'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8',
        'Accept-Encoding': 'gzip, deflate, br',
        'Accept-Language': 'en-US,en;q=0.9',
        'Connection': 'keep-alive',
        'Upgrade-Insecure-Requests': '1',
        'Referer': 'https://www.google.com/',
        'TE': 'Trailers'
    }
    return headers


def get_http_response(url):
    try:
        headers = get_robust_headers()
        response = requests.get(url, headers=headers, timeout=10)
        return response.status_code
    except requests.exceptions.RequestException:
        return None

# Check reachability of domain and IP
# Lumma looks to be using a /1337/ directory on their domain
def check_domain_reachability(domain, reachable_domains, status_code_results):
    ip_address = resolve_domain(domain)

    if ip_address:
        http_status = get_http_response(f'http://{domain}/1337/')
        https_status = get_http_response(f'https://{domain}/1337/')
        http_ip_status = get_http_response(f'http://{ip_address}/1337/')
        https_ip_status = get_http_response(f'https://{ip_address}/1337/')

        reachable_domains.append({
            'domain': domain,
            'ip': ip_address,
            'http_domain': http_status,
            'https_domain': https_status,
            'http_ip': http_ip_status,
            'https_ip': https_ip_status
        })

        if any(status is not None for status in [http_status, https_status, http_ip_status, https_ip_status]):
            status_code_results.append({
                'domain': domain,
                'ip': ip_address,
                'http_domain': http_status,
                'https_domain': https_status,
                'http_ip': http_ip_status,
                'https_ip': https_ip_status
            })
```

```
# Read CSV and check domains
def check_domains_from_csv(csv_file):
    df = pd.read_csv(csv_file)

    reachable_domains = []
    status_code_results = []

    tasks = []
    with ThreadPoolExecutor(max_workers=20) as executor:
        for index, row in df.iterrows():
            domain = row.iloc[0]  # Use iloc to access the first column by position
            tasks.append(executor.submit(check_domain_reachability, domain, reachable_domains, status_code_results))

        for future in as_completed(tasks):
            pass

    #Read Domains
    print("\nSummary of All Domains and Their URLs:")
    if reachable_domains:
        for entry in reachable_domains:
            print(f"- Domain: {entry['domain']}, IP: {entry['ip']}")
            if entry['http_domain'] is not None:
                print(f"  HTTP URL (Domain): http://{entry['domain']}/1337/ - Status Code: {entry['http_domain']}")
            if entry['https_domain'] is not None:
                print(f"  HTTPS URL (Domain): https://{entry['domain']}/1337/ - Status Code: {entry['https_domain']}")
            if entry['http_ip'] is not None:
                print(f"  HTTP URL (IP): http://{entry['ip']}/1337/ - Status Code: {entry['http_ip']}")
            if entry['https_ip'] is not None:
                print(f"  HTTPS URL (IP): https://{entry['ip']}/1337/ - Status Code: {entry['https_ip']}")

    #Print Domains
    print("\nReachable Domains with Status Codes:")
    if status_code_results:
        for entry in status_code_results:
            print(f"- Domain: {entry['domain']}, IP: {entry['ip']}")
            if entry['http_domain'] is not None:
                print(f"  HTTP URL (Domain): http://{entry['domain']}/1337/ - Status Code: {entry['http_domain']}")
            if entry['https_domain'] is not None:
                print(f"  HTTPS URL (Domain): https://{entry['domain']}/1337/ - Status Code: {entry['https_domain']}")
            if entry['http_ip'] is not None:
                print(f"  HTTP URL (IP): http://{entry['ip']}/1337/ - Status Code: {entry['http_ip']}")
            if entry['https_ip'] is not None:
                print(f"  HTTPS URL (IP): https://{entry['ip']}/1337/ - Status Code: {entry['https_ip']}")


def main():
    csv_file = 'results.csv'  # Path to CSV
    check_domains_from_csv(csv_file)

if __name__ == '__main__':
    main()
```

Table 1. Script

List of things to optimize on the script.

*Python script needs to be fixed and could use some optimizations. Made with 95%ChatGPT and minimal human interaction. *Add a component to print the IoCs on an output file.

Table 2. ToDo List on Script

Results on example extracted csv.

```
Reachable Domains with Status Codes:
- Domain: utorrent-backup-server5.top, IP: 91.202.233.151
  HTTP URL (IP): http://91.202.233.151/1337/ - Status Code: 200
- Domain: xn--eclab-1ta.com, IP: 199.59.243.228
  HTTP URL (Domain): http://xn--eclab-1ta.com/1337/ - Status Code: 200
  HTTPS URL (Domain): https://xn--eclab-1ta.com/1337/ - Status Code: 200
  HTTP URL (IP): http://199.59.243.228/1337/ - Status Code: 400
- Domain: xn--ecolb-0qa.com, IP: 199.59.243.228
  HTTP URL (Domain): http://xn--ecolb-0qa.com/1337/ - Status Code: 200
  HTTPS URL (Domain): https://xn--ecolb-0qa.com/1337/ - Status Code: 200
  HTTP URL (IP): http://199.59.243.228/1337/ - Status Code: 400
- Domain: mp3yukle.top, IP: 38.14.102.106
  HTTP URL (Domain): http://mp3yukle.top/1337/ - Status Code: 404
  HTTP URL (IP): http://38.14.102.106/1337/ - Status Code: 403
- Domain: utorrent-backup-server5.top, IP: 91.202.233.151
  HTTP URL (IP): http://91.202.233.151/1337/ - Status Code: 200
```

Figure 6. Example Output

Content of the directory hosted on the domain.

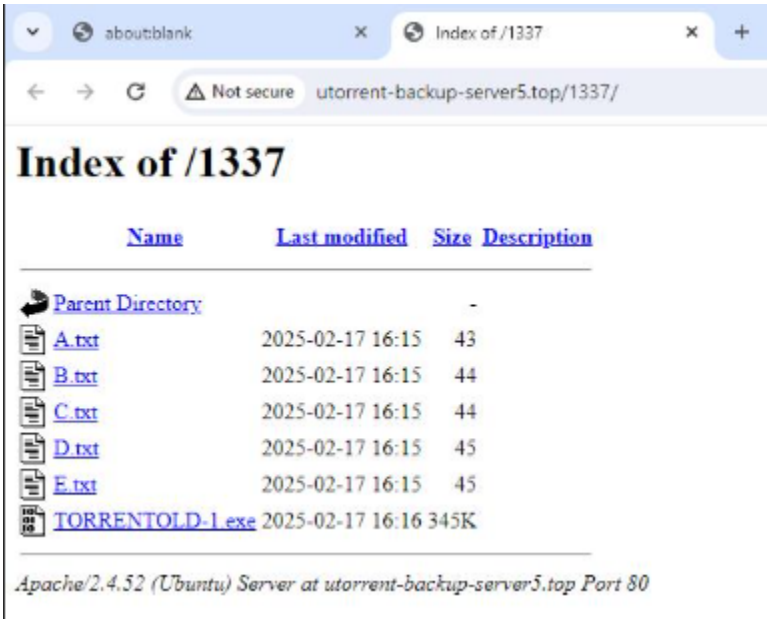

Figure 7. Landing page

MD5 Hash: 401FC7901EF8FF89309B69766FB38CCB

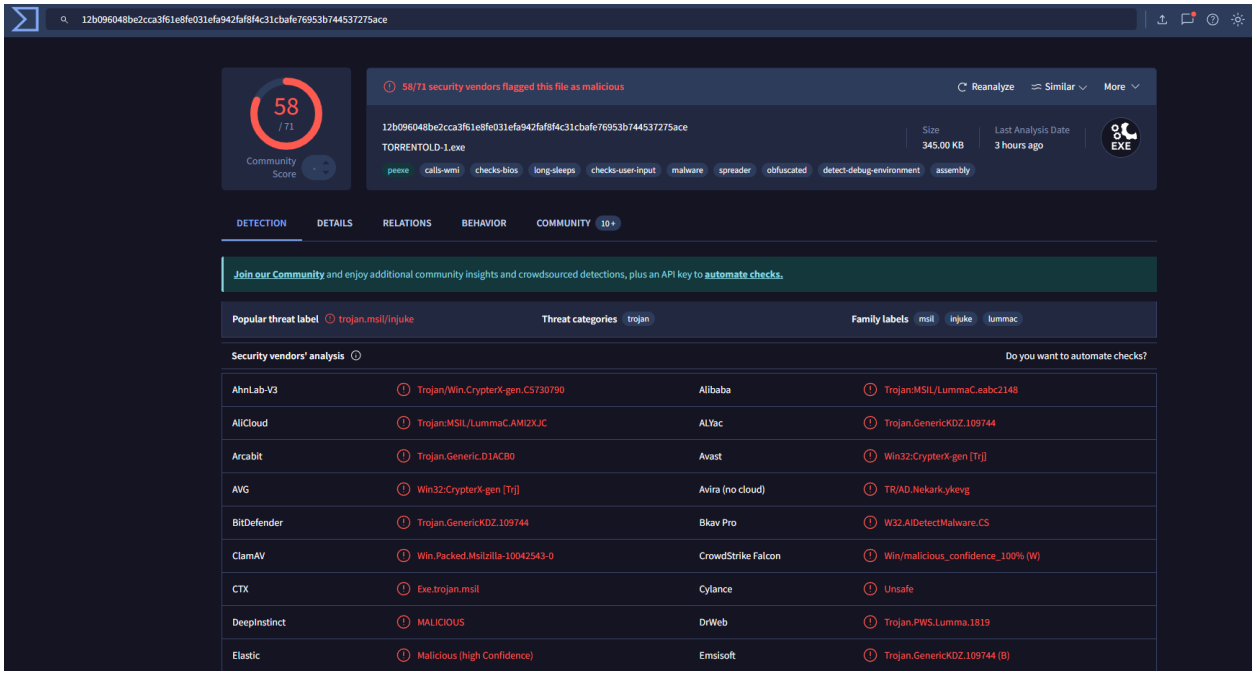

Figure 8. VT Page of the Hash