Phishing With Phineas (Again)

Steroid Boosted Hack Recreation

Georgios Karantzas && Constantinos Patsakis

Disclaimer

All experiments were Performed prior to April 2022 – All responsibilities lie with the Authors and not the EU, current or future employers.

Experiments took place at a given time under specific circumstances and specific software versions. Reproduction is not guaranteed.

The authors notified only the corresponding EDR vendors of several flaws in their detection mechanisms and logic.
Not all these issues are presented in this work.

This work was supported by the European Commission under the Horizon 2020 Program (H2020), as part of the projects CyberSec4Europe (https://www.cybersec4europe.eu) (Grant Agreement no. 830929) and LOCARD (https://locard.eu) (Grant Agreement no. 832735).
The content of this article does not reflect the official opinion of the European Union. Responsibility for the information and views expressed therein lies entirely with the authors.
Moreover, the views do not reflect in any case the views of later employers of the people who conducted this research.

Brief Introduction

About The Experiment

A few years ago, a vigilante hacker under the name “Phineas Phisher” conducted a series of high-profile attacks including hacking into a company that, among others, was developing and selling spyware to government agencies named “Hacking Team”. We decided to recreate his hack on the `Cayman Bank` from start to finish but with elements that modern APT engagements pos.

Side-Note:
Malware will always stand out in traffic-less environment ,however, we still did it ☺.

Final Goal: Reach the secure zone and access a virtual SWIFT panel.

Multiple rounds of tests and BETA versions included.

Environment Similar to the Breach but Hardened

Combat-Proven Defenses

100% Dedication Towards Objectives

State of The Art Malware

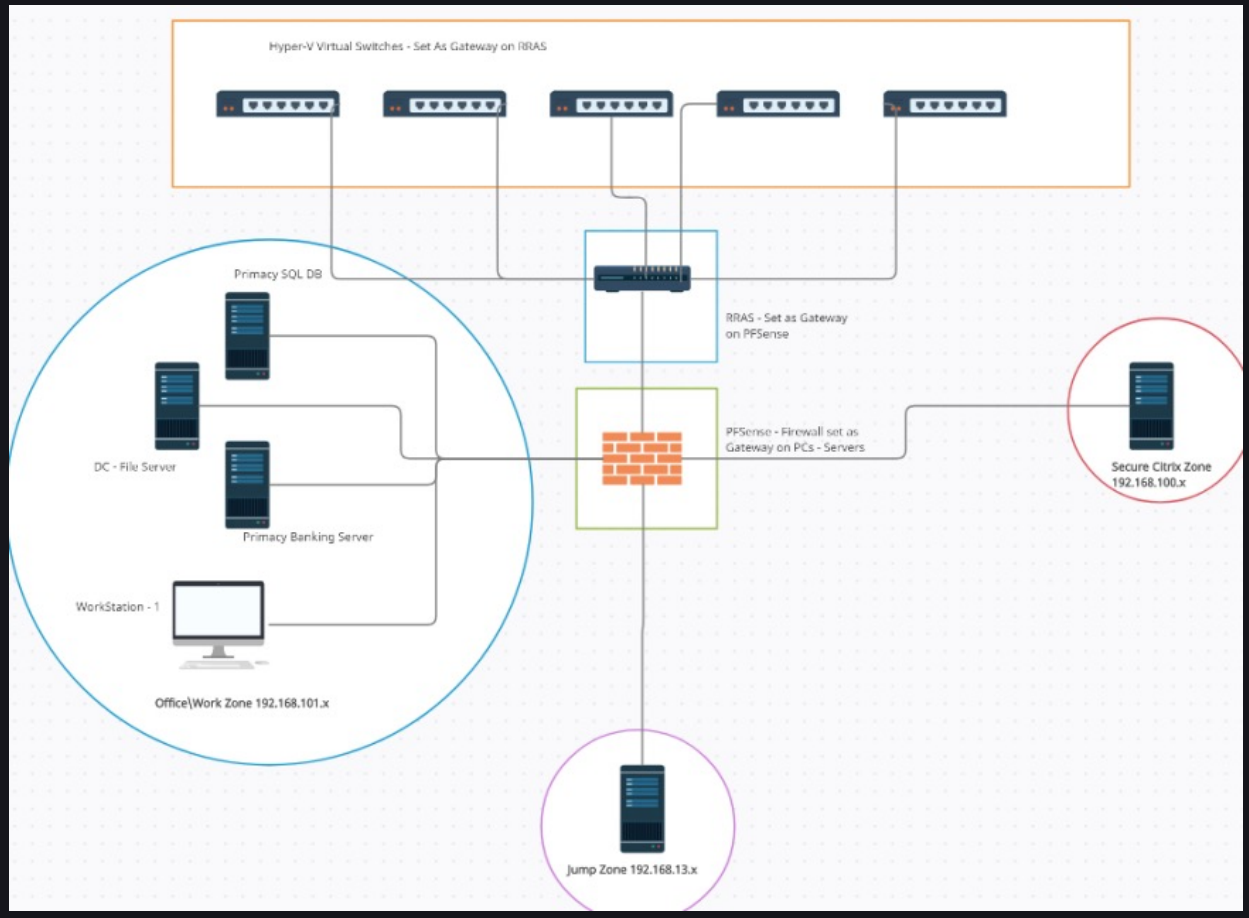
Network Setup

Description

The setup of the environment was created by recreating the Sherwood leak's target network and reconfiguring it as well as extending it with new machines on a Hyper-V server.

Important aspects:

- Jump Server Credential Guard
- Network Segmentation Of The Secure Zone.
- Squid Proxy and ClamAV.



Endpoint Protection Details

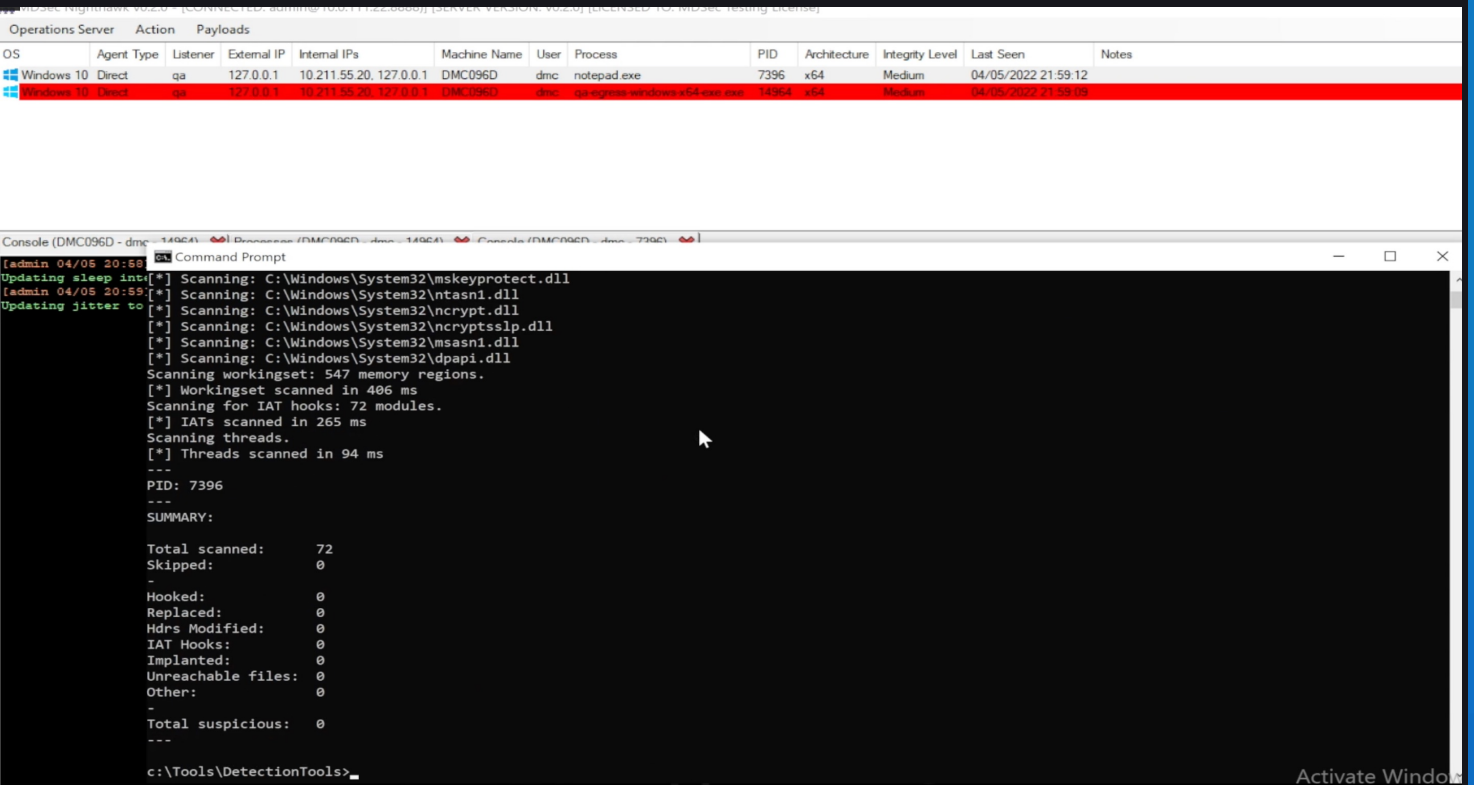
Defensive Model

Sentinel One Singularity with extended aggressive policy without custom STAR rules. In this study, we used versions 21.6.4.423 to 21.7.4.1043, including several tests on BETA features and updates pushed after responsible disclosure.

Fortinet EDR with the normal and effective policy which is set to block for more than 90% of the attack footprints supported. We must note that due to lack day-to-day traffic on the lab VMs we set the solution directly to Protection and not Learning mode which is used to adapt to the network's activity without causing false positives. The versions used in this study are 5.0.2.261 to 5.0.2.335.

Firewalling and proxying with additional **Clam AV** was performed with **PF Sense** 2.5.0-RELEASE. SSL Inspection was enabled as well.
The goal was some minimal hardening at least rather than focusing on network aspects such as NDR solutions or deception.

Offensive Capabilities Nighthawk C2 0.1



Nighthawk is a C2 framework developed by MDSec designed by hardcore red teamers for hardcore red teamers with stealth, configurability, and feature rich- ness in-mind. Malleability is among the top priorities and assists in avoiding attributing specific behaviors to Nighthawk making it harder to track down.

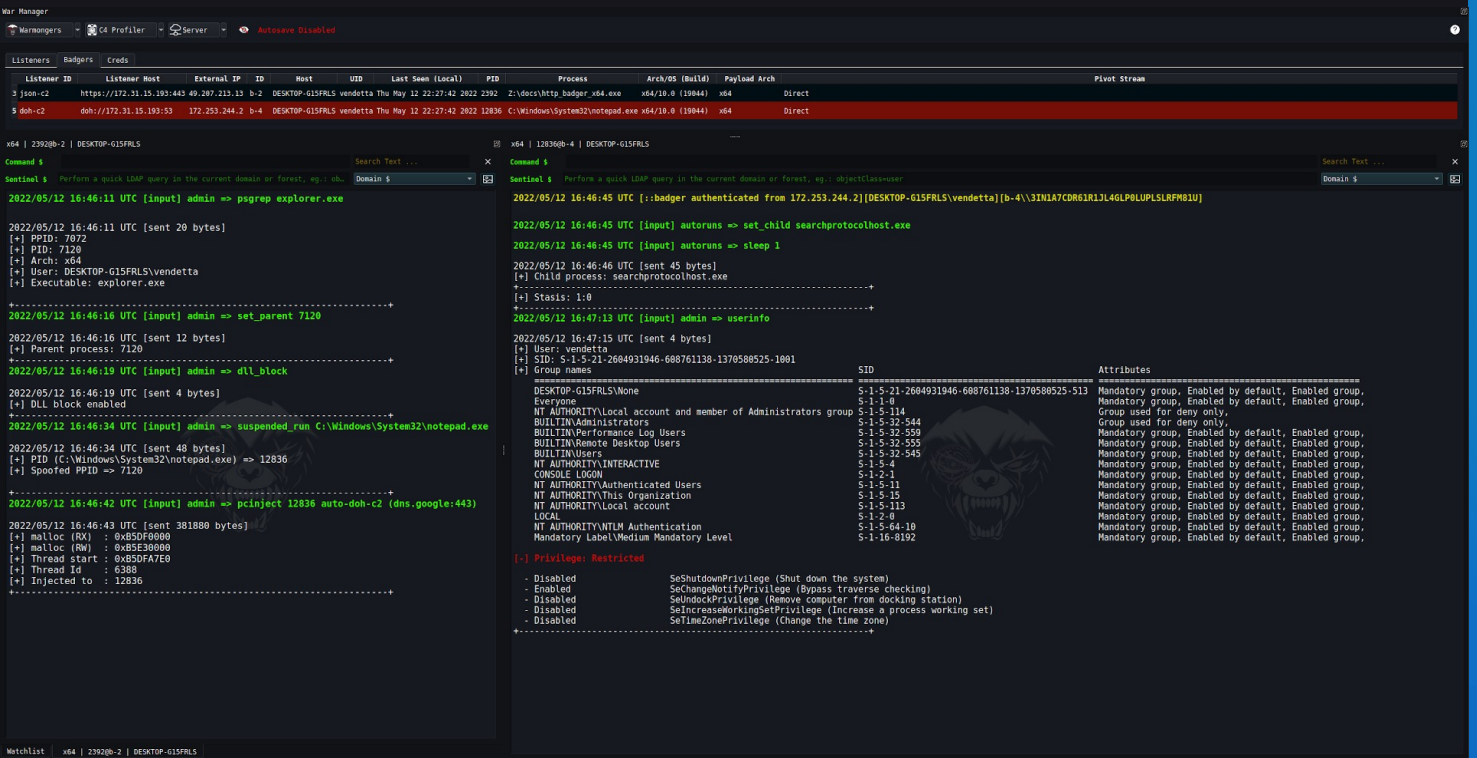
Features we assessed included:

- ROP-based system call unhooking and later full DLL unhooking which comes “by-design” therefore, it makes the operator’s life easier.

It also includes other useful features like:

- Thread Stack Spoofing
- In-memory hiding via heap-based encryption as well as it usually avoids several tools (depending on the case) that will scrap through the memory of a process for abnormal indicators such as Moneta and pe-sieve.
- Customized process injection methods.
- Universal usage of unhooked system calls.
- Network-callback and traffic related options to guarantee undercover beaconing.

Offensive Capabilities BRC4 0.7-0.9



Brute Ratel C4, by Dark Vortex, is one of the most ambitious attempts we have seen in the industry.
It is marketed as:

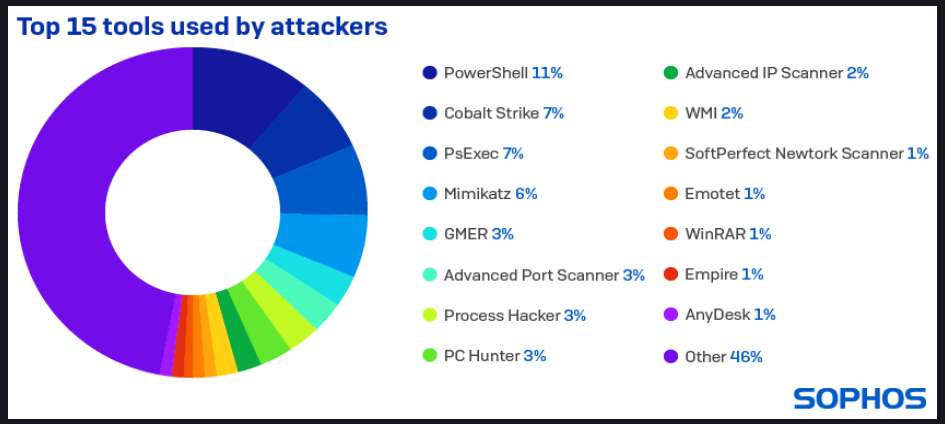
- A low-cost alternative to Cobalt Strike with less well-known indicators.
- A more opsec and user friendly as well as adaptable product.

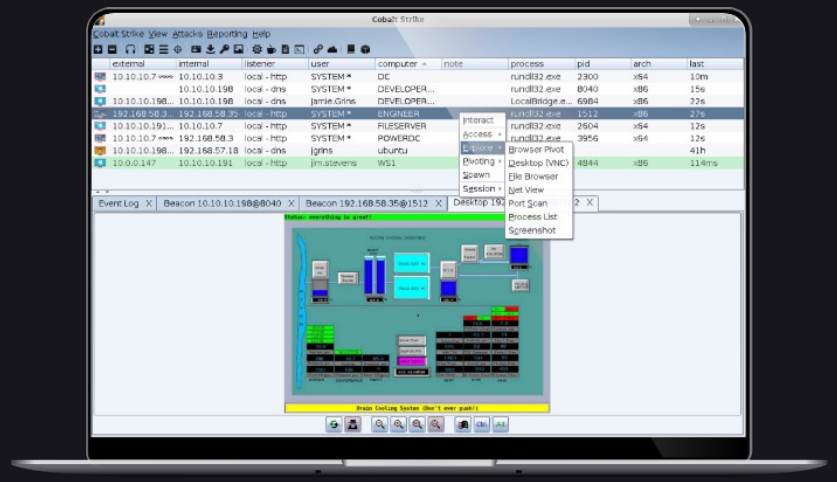
Caveats Include:

- Attributed a few months after the experiments to malicious campaigns by Palo Alto Unit42.
- Questionable vetting and pre-sales process.
- Borrowed code from open-source tools without much editing results in shared IOCs as well.

Some of the high- lights included:

- The custom plugin called LDAP sentinel.
- The customized reflective loader.
- The BOF files.
- The easily configurable TTPs ranging from the network communication to process injections and more.
- Anti-Analysis capabilities.
    - *A gap was created due the impact of anti-analysis on the correct functionality from version to version and this was something noticed empirically. Such issues should be taken into consideration by all OST developers as they affect the universal ability to operate against diverse environments. This applies as a principle to EDRs regarding the detection-performance-accuracy relationship.*





Offensive Capabilities Cobalt Strike 4.4

Whether you are a Red Teamer or a threat actor , CS4.4 is the major tool used for stage 2 operations. Our research has as a goal to **break that norm by exploring alternatives.**

A significant part of the work is done by the community judging from the plug-ins in the forms of BOFs, Reflective DLLs, various kits etc.
The latest version at the time of the writing of this section is 4.4 and it is the one we are using in our experiments.
Cobalt Strike is not that opsec safe anymore, yet you can always:

- Implement several security features on your own and embed them on your loader like LockdExe and Shellcode Fluctuation to avoid shellcode in-memory scanning or Shellycoat to remove hooks by refreshing ntdll's text section.
- You can also try to avoid a few IOCs and be stealthier by using tools like the Sleep Mask kit or the Artifact Kit which was originally created to obfuscate behavior of artifacts produced by CS and how they handled shellcode to evade sandboxes.

In our case, those will not be helpful as many detections nowadays are multi-layered and generic.
In many cases, TTPs such as stage-0 implants are tailoring to the host such as using specific processes and matching profiles for C2 communication are used.

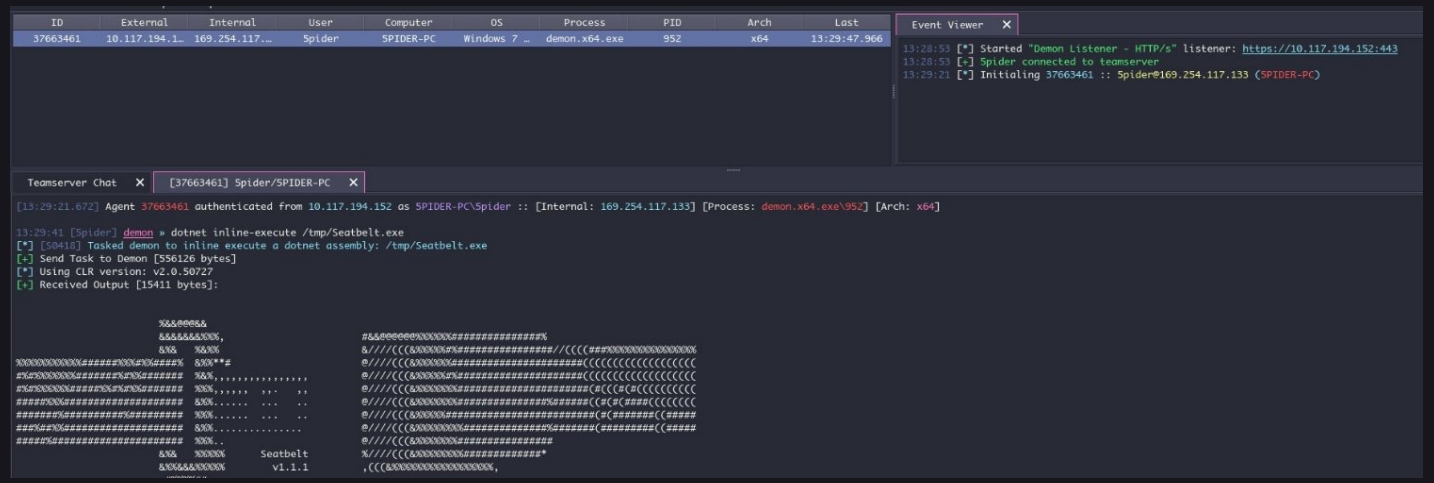
Offensive Capabilities Oyabun

Oyabun, by Red Code Labs, is a newly created tool by Red Code labs with a somewhat more generic and limited scope when it comes to its usage. It is a multi-platform Golang based toolkit that has proven to be highly effective against modern defences to pass some initial barriers. Basic capabilities include situational awareness and deployment of additional malware. The authors attempt to solve bugs that may be present by placing a lot of effort on a new product whose delivery is mostly non-modular and unstaged/static. The tool was not designed to complete the mission but rather serve as a limited stage-0 which happened successfully.

Features Include:

- Ngrok Integration.
- Go-native reverse shell through TCP Dialer.
- Discovery Features.
- Keylogging.

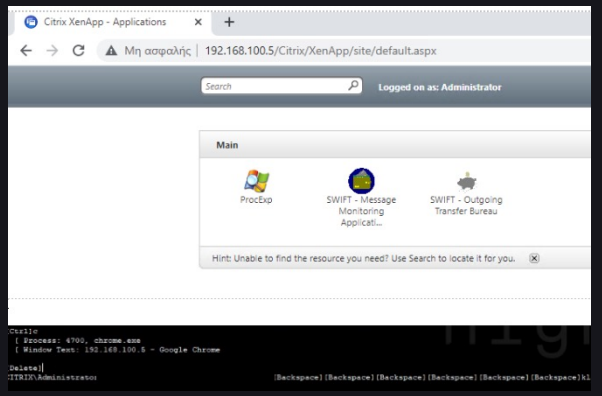
Offensive Capabilities Havoc

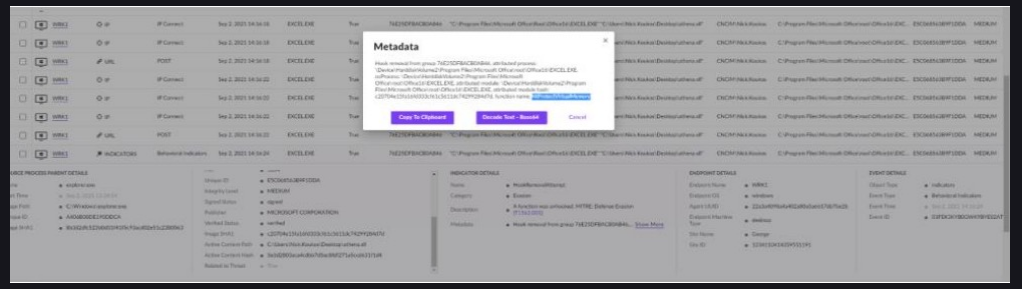


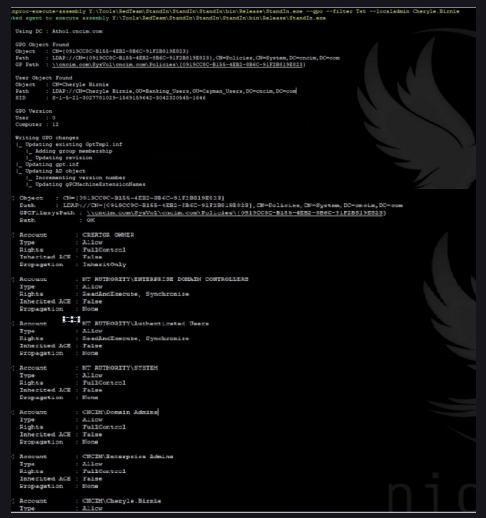
Havoc represents the category of malware that is not a commercial product rather, it was developed by an aspiring, young security researcher named Paul Ungur who is still a student.
We therefore wish to demystify this kind of tool and demonstrate the capabilities of non-corporate software developed with stealth, simplicity and stability in mind. Referring to the previous slide, we would attribute Havoc to the category “Other”-”Unknown” that represented the 46% of malware mentioned in the Sophos’ research.

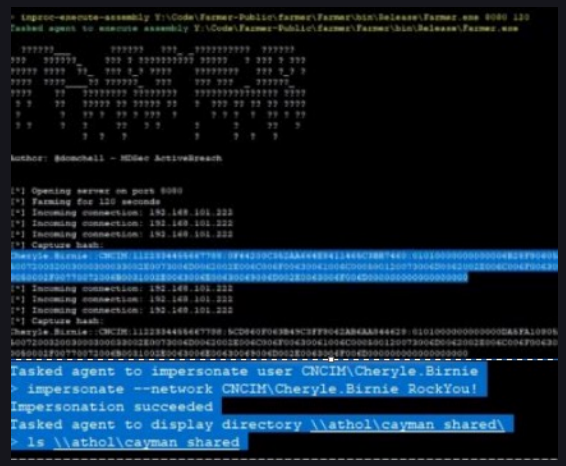
Havoc was built targeting the vast majority of endpoint solutions and therefore it had a few IOCs by design.
We assisted Paul to transition the software to a more suitable condition for this scenario by contributing slightly to the development process. The network communication is performed through the TCP protocol’s sockets with AES encrypted content and capability of sleeps during which no command will be fetched.

With a bit more customization through some basic amount of effort , Havoc’s flexible codebase was easily changed to cover a much wider defensive landscape and complete the mission.

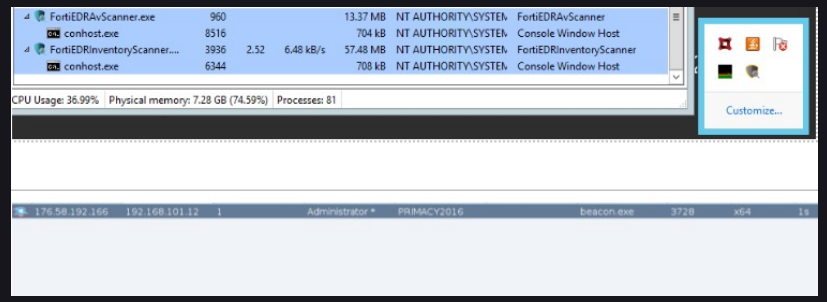


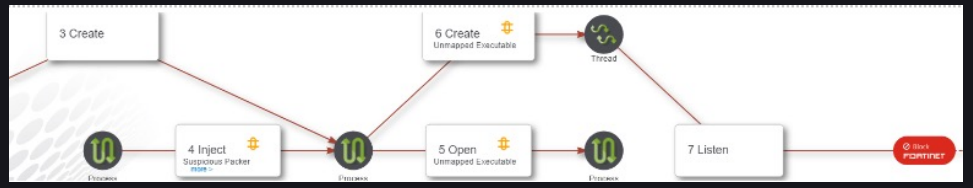


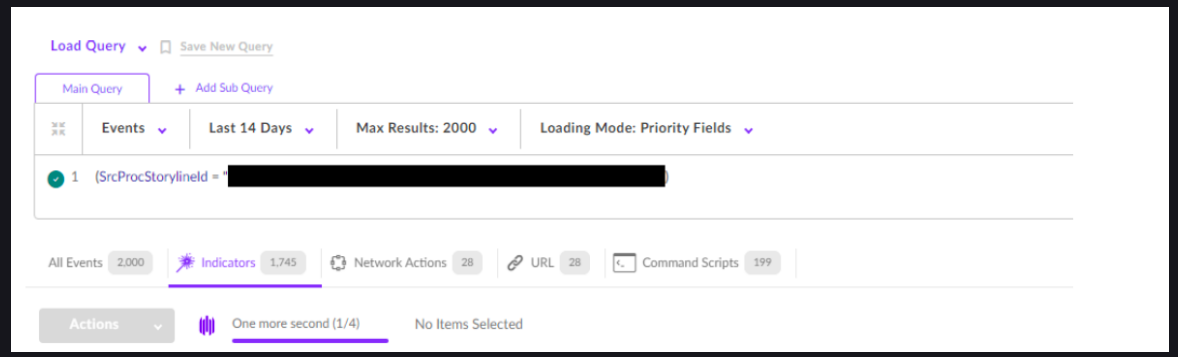


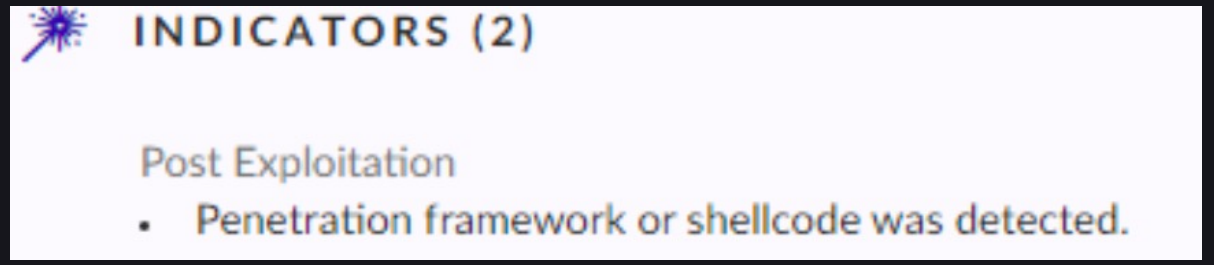


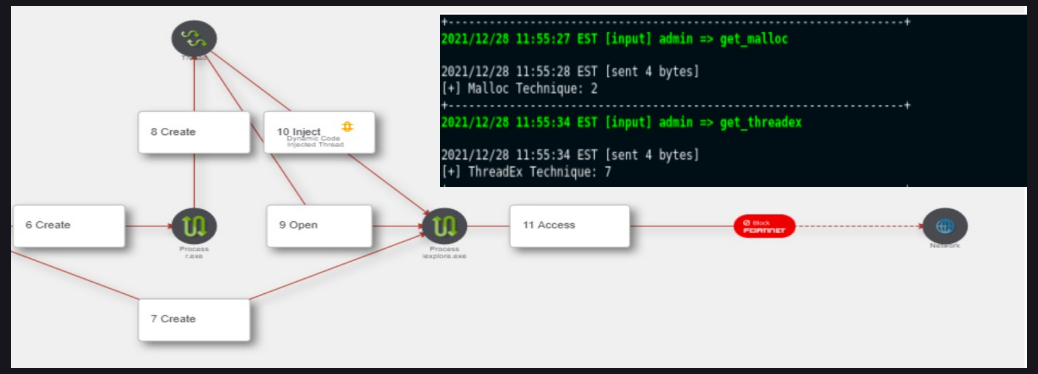
Experimentation Examples – Scenario





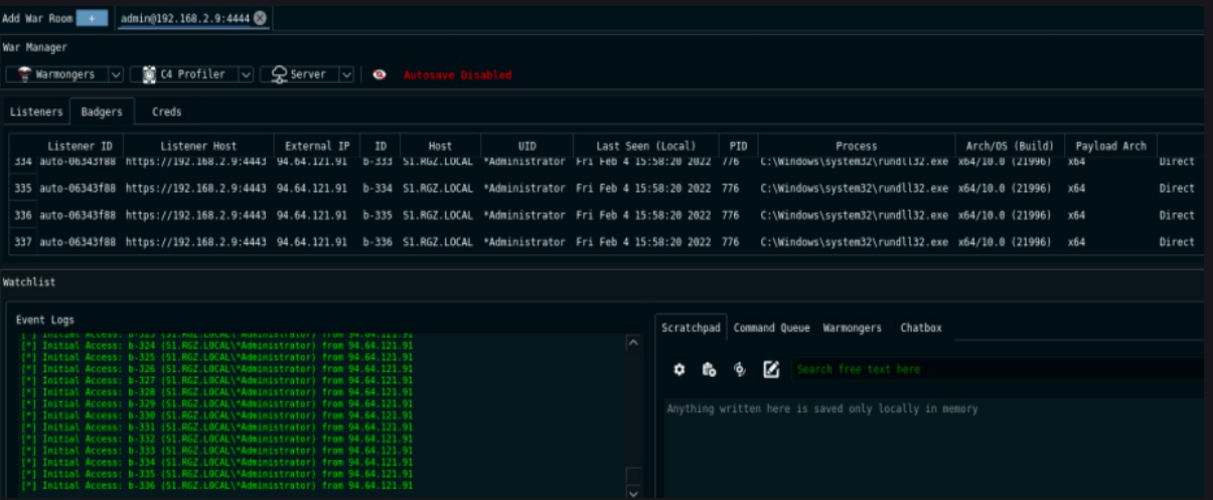


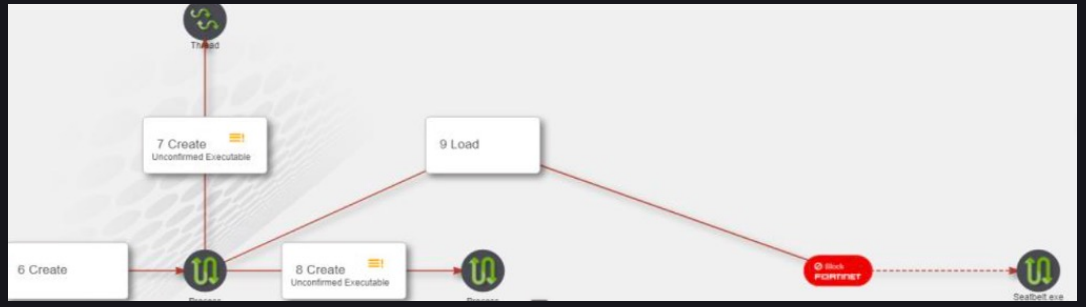




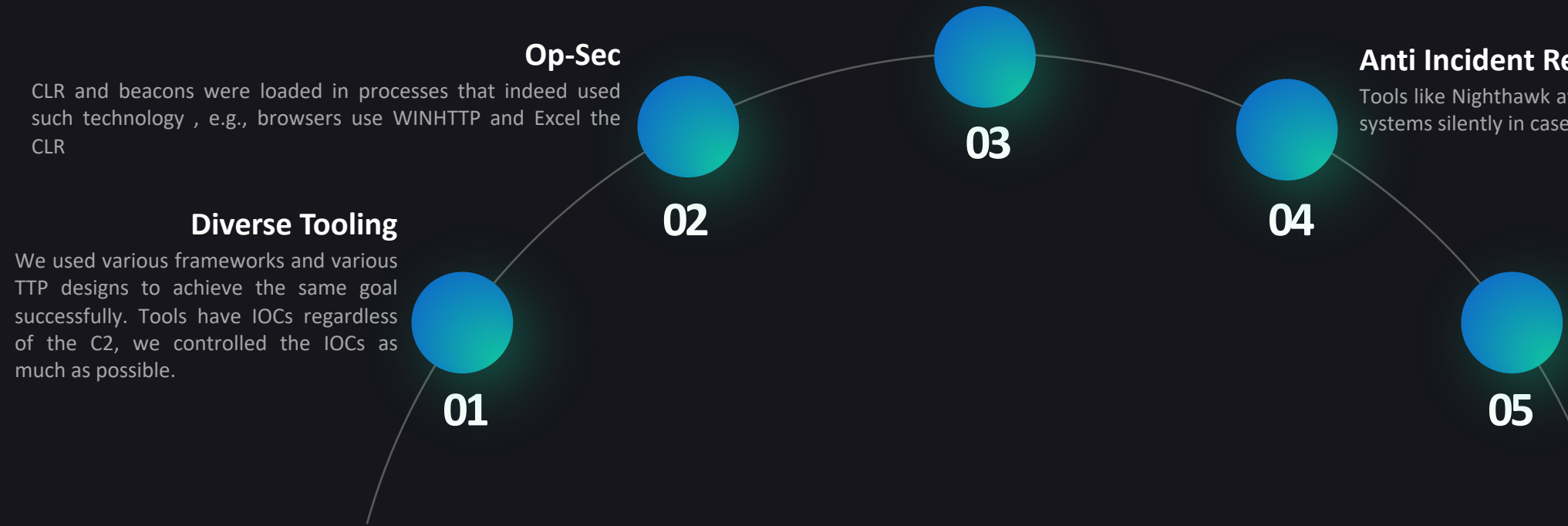
Upon Developer's "Challenge"

Experimentation Examples – BRC4





Tales From The Reproduction



Using The Kernel

We had to use kernel drivers for credential extraction from Physical Memory, immobilizing the EDR using ProcExp's Driver and running tools but even injecting into LSASS and dumping from within.

03

Op-Sec

CLR and beacons were loaded in processes that indeed used such technology , e.g., browsers use WINHTTP and Excel the CLR

02

Anti Incident Response

Tools like Nighthawk avoid in memory scans and can persist on systems silently in case of raised red flags.

04

Diverse Tooling

We used various frameworks and various TTP designs to achieve the same goal successfully. Tools have IOCs regardless of the C2, we controlled the IOCs as much as possible.

01

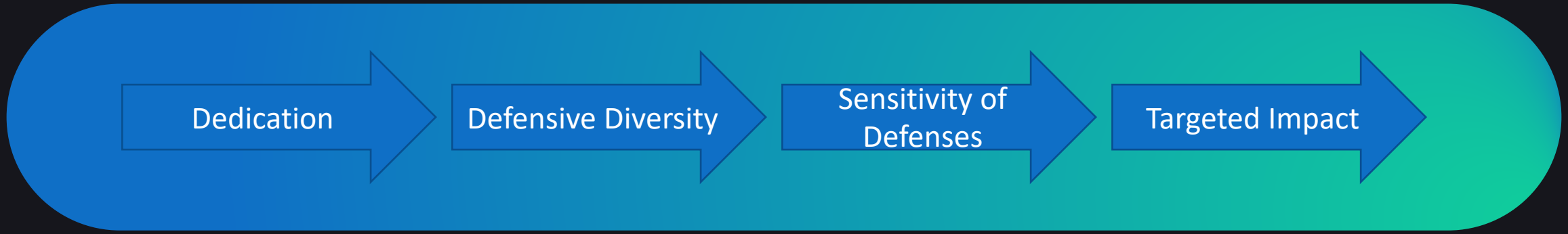
Discovery Of Vendor Specific Bugs

We avoided detection by leveraging bugs unique to each vendor's architecture, defeating certain protections universally.

05

Why This Experiment? Why Not MITRE?

Differences with Existing Scenario-Based Assessments



1. Months of work were devoted.
2. Important findings occurred and were targeted towards the specific environments.
3. Real threat actors compromise “at will”.
4. What worked on Host A would be detected by 90% on Host B due to the detection approach differences (“Closed Circuit” Protections vs Memory Inspection).
5. Casual evasion such as API unhooking would be obsolete.
6. BETA testing was included.
7. One mistake was enough.
8. Many ways tested to achieve the same goal.
9. Detections were easy to trigger.
10. We were self-aware and had in mind the context of the process and monitoring changes across processes.

TTP Highlights

Some Abused Techniques

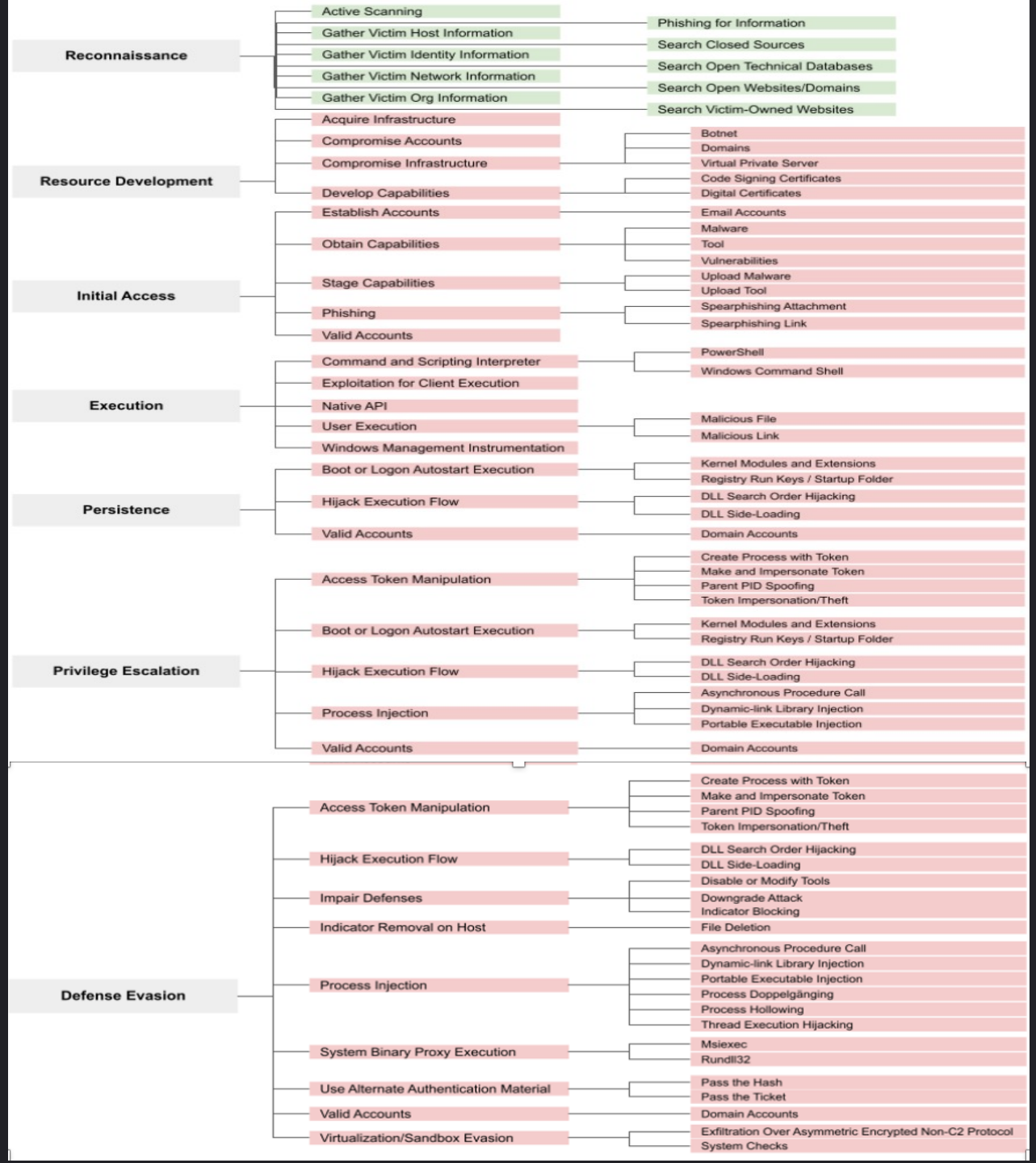
- XLL Files
- Vulnerable Driver Abuse
- Customized Injections
- MSI Files

- Custom Per-Vendor Bugs
- DLL Sideloading
- Credential Theft via Kernel Dumps & Injections
- PowerShell & CLR Reflection

- Reflective PE Loading
- Blending In to Avoid C2 Detections (Image Loads & Network Traffic & Assembly Loads)

What Happened?

TTP Mapping



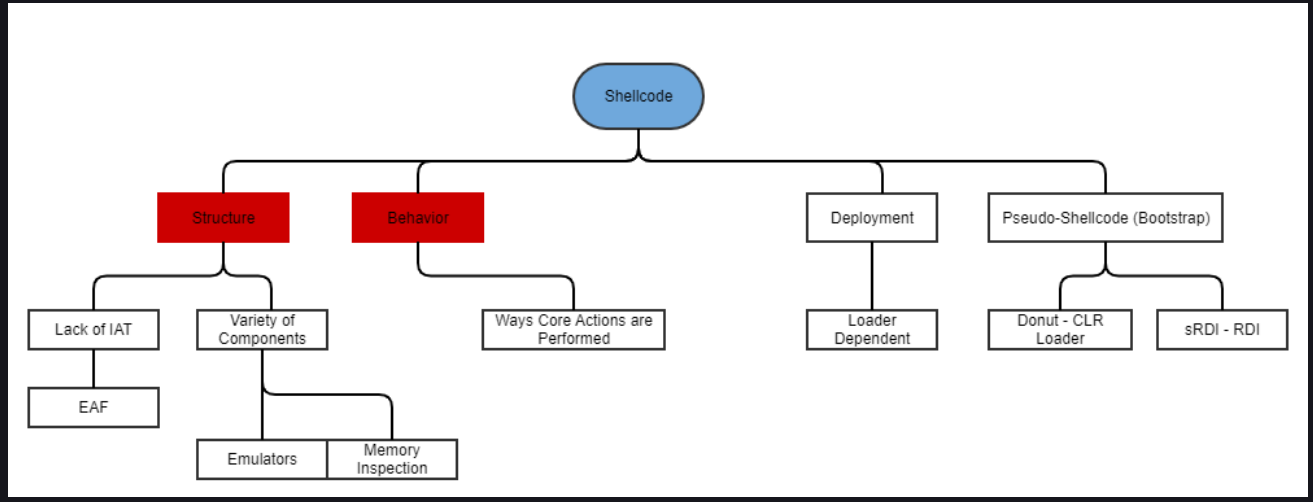
Points Of Failure for Dynamic Code

To PIC or not to PIC?

Shellcode or PIC is one of the most common attack vectors, being the default delivery method for many C2s.
Points of detection may include:

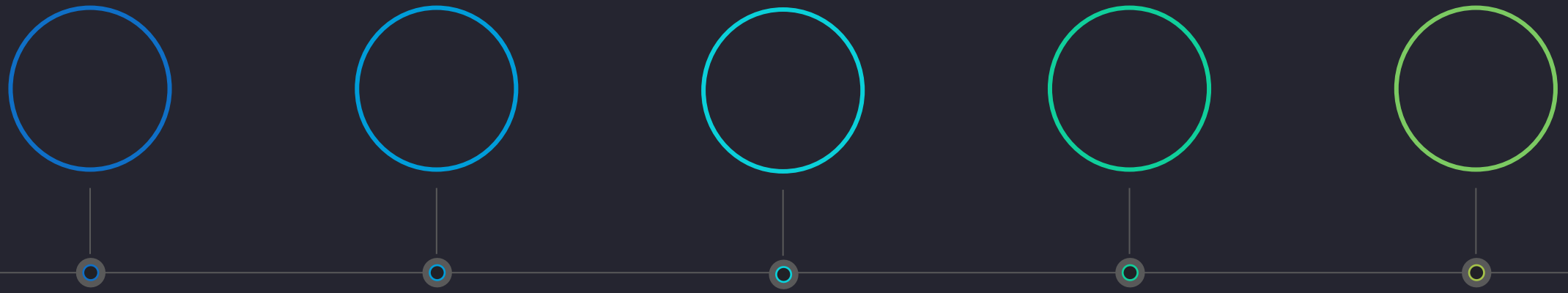
- In memory format.
- IAT absence.
- Behaviour and goal.
- Deployment.

In many case defences will attempt to classify shellcode as an entity itself. False positives however are very common to exist.



Brief Timeline Description

A Brief Timeline Of The Attack



Initial Access

Initial access via XLL and MSI (incl. Persistence), Injecting into a Free AV solution and DLL sideloading.

Spread

Usage of Farmer and malicious shortcuts to harvest NetNTLMv2.

Impersonate & Move

Token impersonation , GPO abuse (got SYSTEM via .exe) , SOCKS proxying and RDP\WMI for lateral movement.

Steal

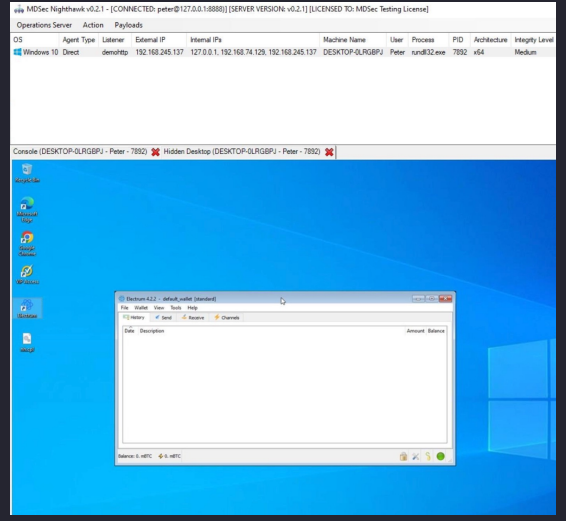
We performed credential dumping via WinPMEM and used keyloggers across the network. Internal monologue was used avoiding downgrade and detection.

Jump Around

On the last steps we utilized a bug (now fixed) related to lateral movement and all of our malicious actions were disregarded.

Post-Experiment Impact

C2 Improvements Nighthawk



Hidden RDP

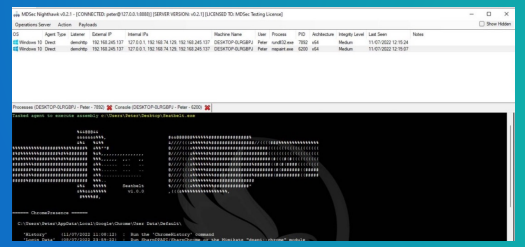
A slick way to control and monitor the targets

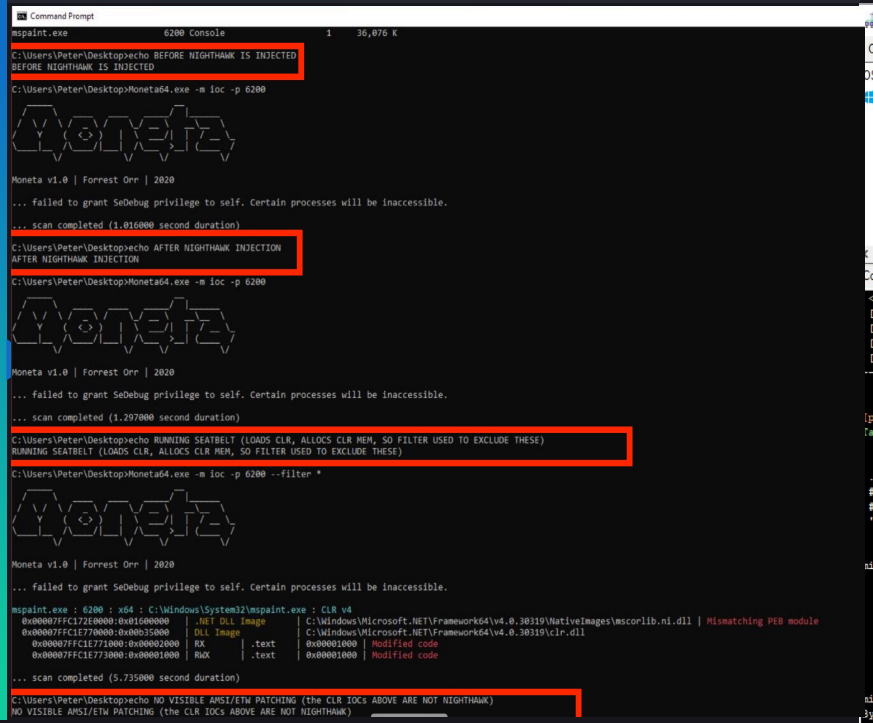
Stealthy PE Loading

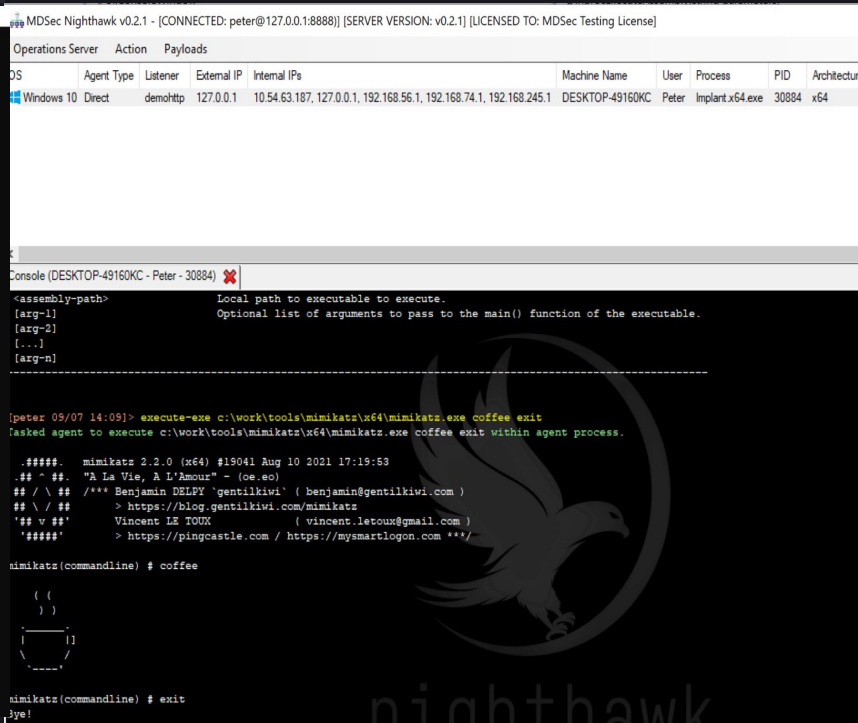
Running EXEs just like normally with improved stealth.

Memory Evasions

Evading patch detections and memory scanning.







Post-Experiment Impact

C2 Improvements Havoc

Malleability

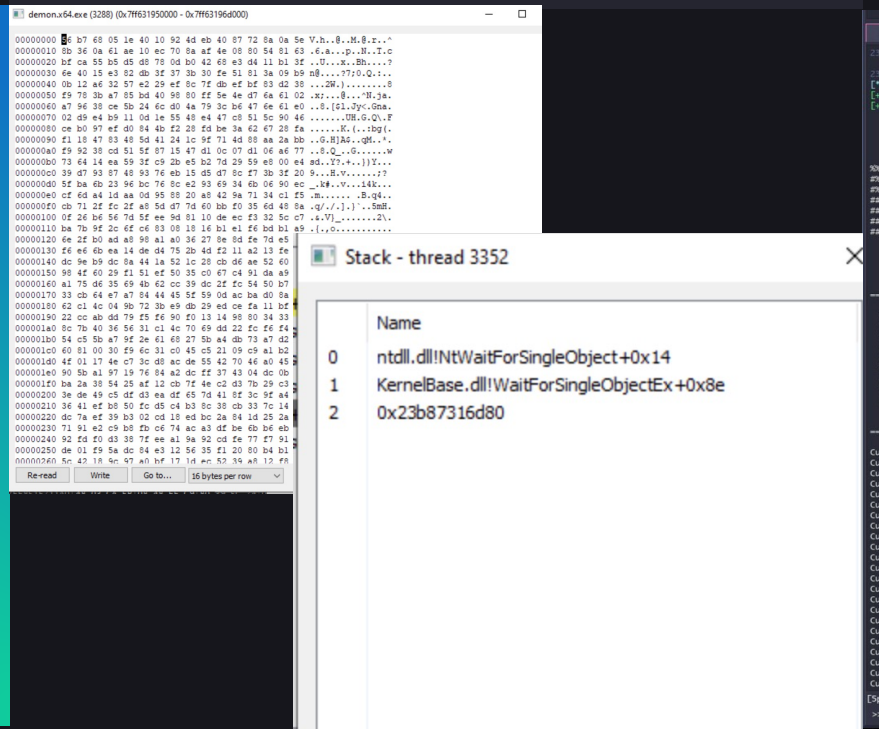
More control in the hands of the user.

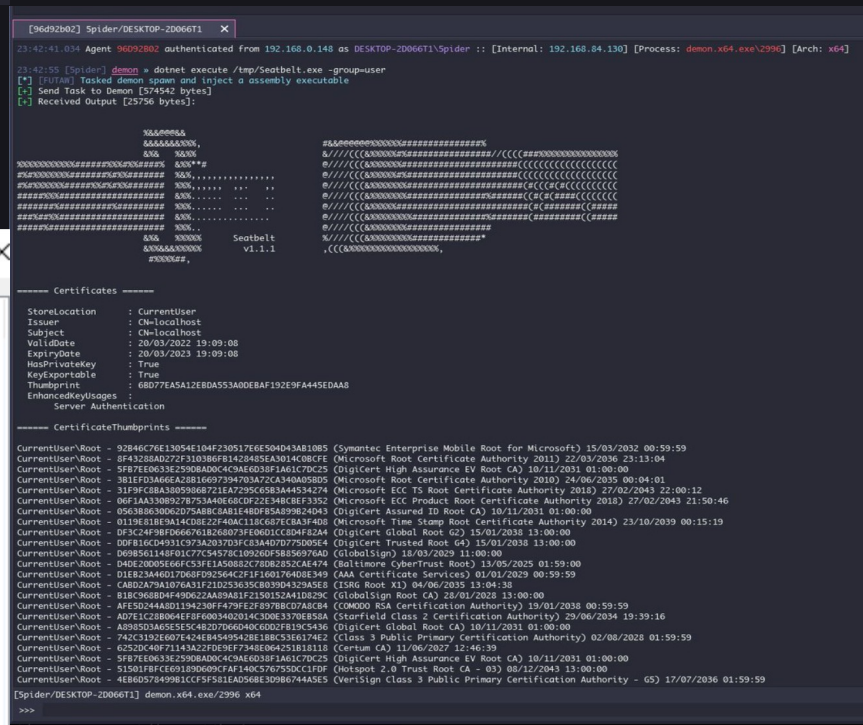
Extending Common Features

e.g. Fork & Run Execute Assembly with output consumption via anonymous pipes.

Sleep Based Evasions

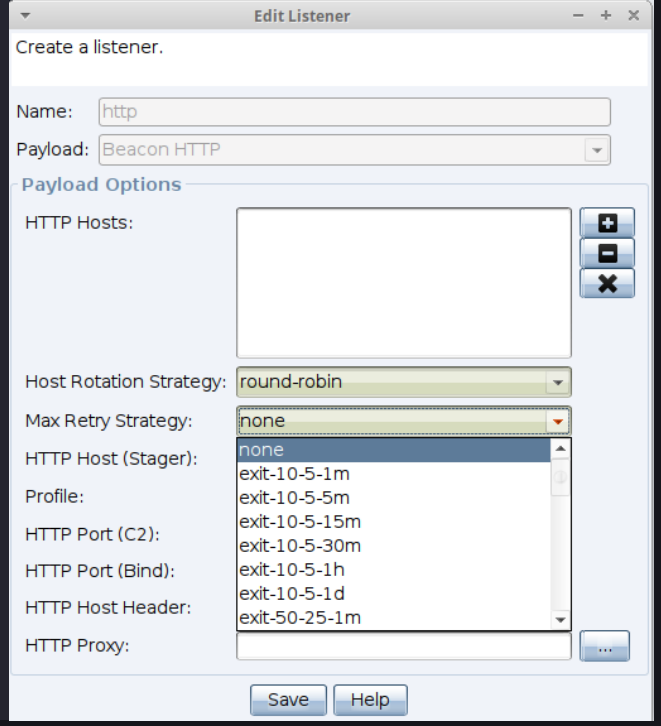
ROP Chain based image encryption & thread stack spoofing.





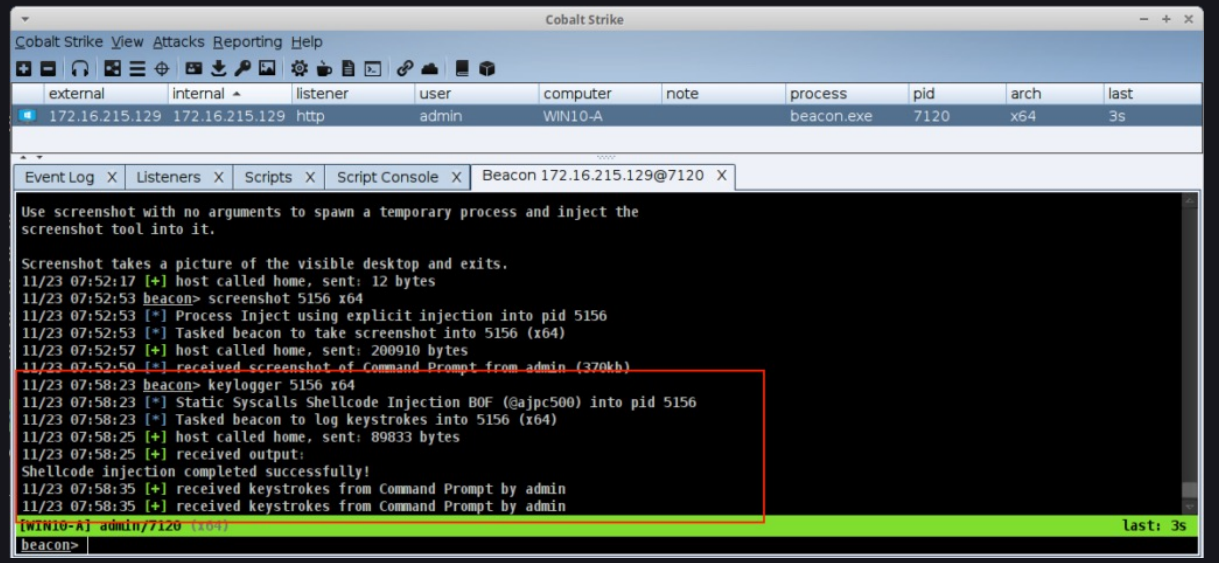
Post-Experiment Impact

C2 Improvements Cobalt Strike



Various Improvements

- Thread Stack Spoofing Arsenal Kit Update
- Licensing and Watermarking
- Increased Price
- Malleability Improved – e.g. Fork & Run Customizations
- Evasion and Stability updates - e.g. Max Beacon Retries



Post-Experiment Impact

C2 Improvements BRC4

- C2 Communication Improvements (e.g. DNSoverHTTPs, Slack)
- Sleep Encryption Mechanisms (with or without ROP)
- WMI over DCOM Lateral Movement
- Hook Evasion Attempts
- Built-In Kerberoast
- “Crisis Monitor” Updates

Various Improvements

Authors cannot be responsible for inacurracies as the information was taken from the blog posts of the creator.

Acknowledgements

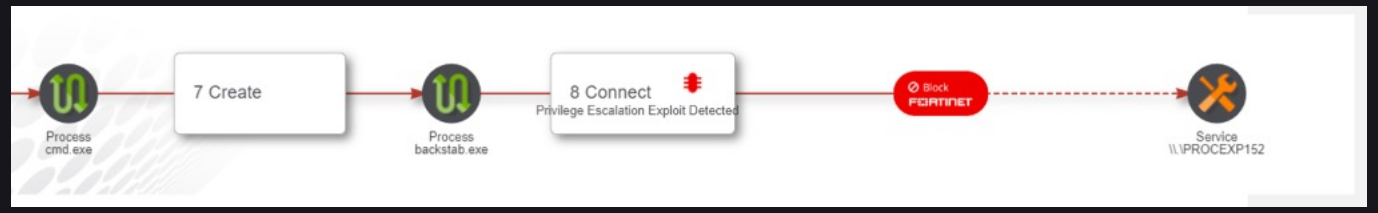
Vendors Acknowledged:

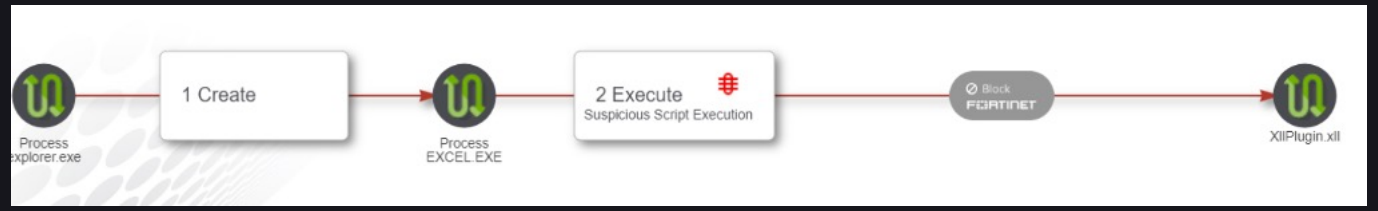
- Singularity XDR's Technical Team acknowledged and fixed in a timeframe of several months that were invested in rigorous reviews and consulting\negotiating with appropriate departments aspects such as public impact and criticality of information as well as alternative ways to get credited:
    - Rigorous BETA testing of several features that pushed the scenario to the extreme.
    - Architectural Misses
    - Product Specific Bugs\Issues
    - Common Misses
    - Feedback and Support from our Team led by Dr. Patsakis (as Academic Lead) and Georgios Karantzas (as Technical Lead)
    - Information will be kept private for client safety reasons.

- FortiEDR Team acknowledged and fixed in a timeframe of almost a month that was invested in pushing detections, fixes and informing the CTO:
    - Testing of both the production version and some insight on detections pushed in BETA
    - Feedback over Architectural Misses & Bug fixes that would possibly assist in ensuring future prevention of cases such as those later described in PA's blog.
    - Product Specific Issues
    - Common Misses
    - Feedback and Support from our Team led by Dr. Patsakis (as Academic Lead) and Georgios Karantzas (as Technical Lead)
    - Information will be kept private for client safety reasons.

Thanking Note:

- MDSec
- Paul Ungur
- Redcode Labs





Mission Accomplished?

Outcome

Brute Ratel

Unstable.
Operationally unsafe.

Detected by various traps including targeted detections against evasion mechanisms.

Minimal utilization.
Lack of Delivery Flexibility.

Havoc

Autonomous Scenario Completion.
Generic codebase modifications needed.
Payload delivery adapted to the scenario.
Basic features.
Modified by-hand.

Some generic detections occurred but were combated via generic op-sec improvements.

Cobalt Strike

Utilized only as a secondary tool on systems with removed protections.
Lack of Delivery Flexibility.
High stability.

Highly detected by various traps on-sight.

- Targeted PIC Detections
- Memory Inspection
- Generic Detections
- Deployment and malicious files (EMU)

Nighthawk

Autonomous Scenario Completion.
Operational Stability.
Malleable configuration and modifications.
Payload delivery adapted to the scenario.
Complex features.
Lateral movement eased.

Detections were mostly targeted towards unsafe operator actions and configurations.

Get in Touch With Us ☺!

University Of Piraeus

gck.kara@gmail.com && kpatsak@unipi.gr

@GeKarantzas & @kpatsak

https://www.cs.unipi.gr/kpatsak

Thank You