BSIDES DFW

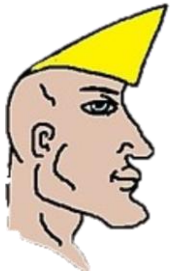
Purple Teaming: Red Team Advice and Why You Should Do It Too

UNIT 43

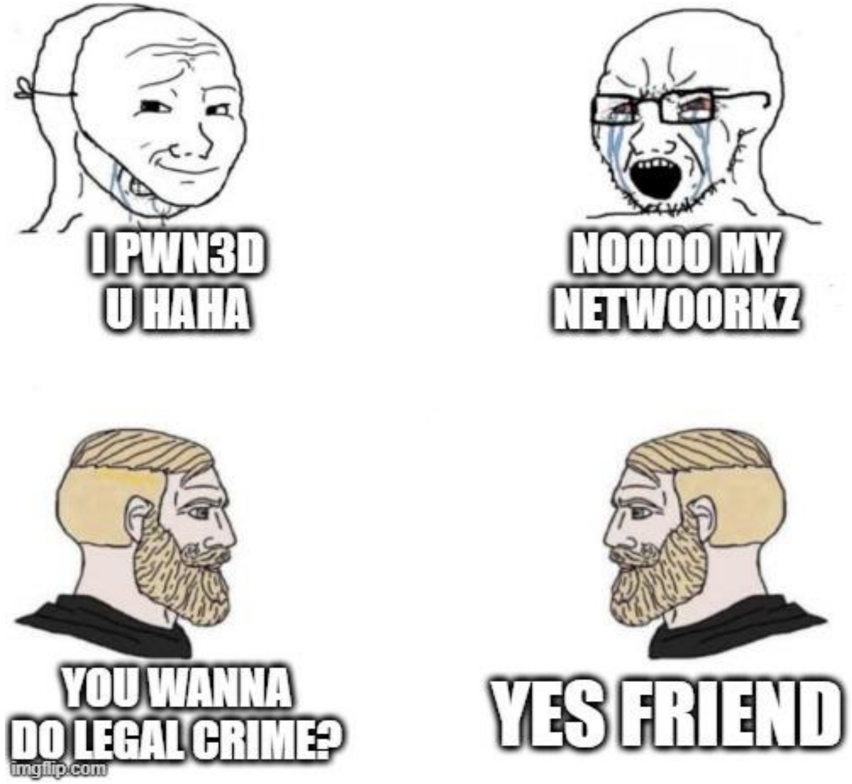
whoami
Cody Read (@ewbysec)

- ~8 years previous IT experience (Desktop Support/Systems Administration/Security)
- CTF’ing for ~8 years
- Started Red Teaming professionally in early 2022
- Focused on adversary simulation and offensive security tooling (Primarily in C)
- Author of some basic Cobalt Strike BOFs and other POCs
- Threat Intelligence enjoyer and Purple Teamer
- **Delinquent**

Overview



- Introduction to Purple Teaming
  - Co-workers, not adversaries!
  - It’s natural, why aren’t you doing it yet???
  - How can I get started???
  - Methodology
  - How it benefits the company, and you!
- Purple Teaming Endpoint Protection
  - Red Team TTPs…a lot of them
- Purple Teaming Network Detection
  - More Red Team TTPs…snooze!



Co-workers, not adversaries!

- Blue and Red have typically been quite adversarial over the years
  - If you're internal, you both have the responsibility of increasing the security posture of the org
  - If you're external…you both have the responsibility of increasing the security posture of the org
  - Why not make it an enjoyable experience??? Learn together and build each other up!

- Very few things have a greater impact on security posture than PT
  - My humble opinion, don't @ me
  - Live testing and changes (of course not pushed to PROD without a change…right!?!)
  - Collaboration, brainstorming, adversary replication, custom detection

It’s natural, why aren’t you doing it yet???

- This thought process is mostly for the internal folk like myself
  - You should want to vigorously test your security products, especially as a red teamer!
  - In my experience both in IT and offensive security, out-of-the-box products rarely do what you think they do and require tailoring to your environment.
  - This requires effort from both red and blue. Some red can do the blue side, and some blue can do the red side, but neither can really go deep into their “opposing” fields without working 80 hours a week, so help each other!

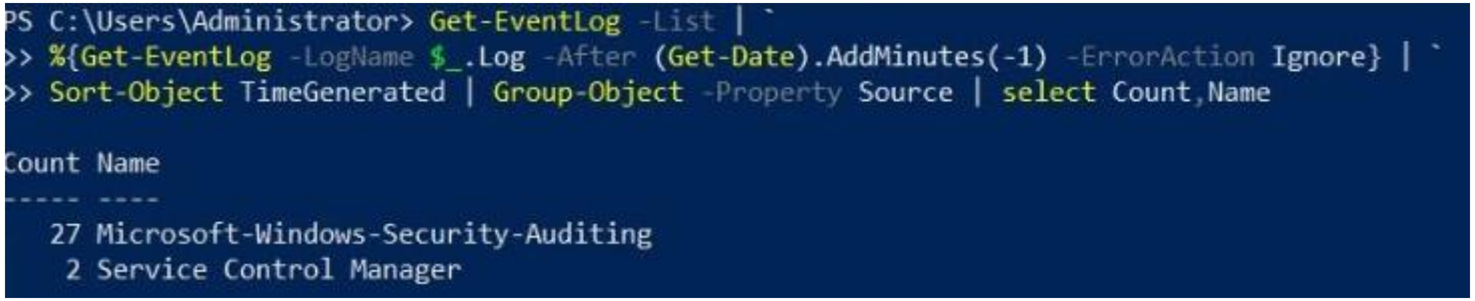
- Keeping secrets never helped anyone
  - On either side, no one should be keeping secrets
  - The threat landscape changes so often and so much, tradecraft is always evolving
  - Provide value, document findings, and propose R&D time to keep up
    - This is **extremely** important, for your sanity and personal time, as well as your org
    - As POCs become harder to find, and private exploits continue to be used, make your own!

How can I get started???

- Time and eventually resources will be your biggest problem
- You and your blue team colleague MUST be interested and dedicated
- As red, be interested in blue
  - Read threat intel reports. Start with open source (twitter is great for this)
  - Expand your knowledge past your silo to cover as much ground as possible
- As blue, be interested in red
  - Again, read threat intel reports. Start with open source, then eventually enterprise
  - Look up common tooling on GitHub, read it and understand how it works
- Start small, keep it at 1-2 hours a week at first
  - Do the thing
  - Document you doing thing
  - Provide value in detecting the thing
  - Get more time and resources to do more things

How can I get started??? pt. 2

- Close the barn door FIRST, then work on the holes in the roof
  - Detect what happens when someone literally just runs the tool
    - Baseline out-of-the-box C2 detection
      - Default network profiles, heartbeats, API usage, etc.
    - Port scanning
      - Detect massive amounts of port connection attempts
      - Tailored detection for popular Windows ports
    - Bloodhound / LDAP enumeration
      - Detect massive amounts of LDAP queries from a single endpoint to your DC / grouped DCs in your log aggregator
    - etc
    - Sweet one liner to get you started log hunting for detection when testing (credit John Dwyer, IBM X-Force)



```
PS C:\Users\Administrator> Get-EventLog -List | `
>> %{Get-EventLog -LogName $_.Log -After (Get-Date).AddMinutes(-1) -ErrorAction Ignore} | `
>> Sort-Object TimeGenerated | Group-Object -Property Source | select Count,Name

Count Name
----- ----
   27 Microsoft-Windows-Security-Auditing
    2 Service Control Manager
```

How can I get started??? Red Team Edition

- https://book.hacktricks.xyz/
  - Amazing site for most things offensive, and defensive too!
  - Where I go most of the time to get started
- https://www.ired.team/
  - Probably the best site for red team specific stuff
  - TTPs, code to get you started on your own POCs, etc.
- https://maldevacademy.com/
  - The best malware dev site available
  - Modern techniques and frequent updates
- https://institute.sektor7.net/
  - A close 2nd to MalDevAcademy
  - I personally learned with Sektor7
  - MDE/MDI/Evasion are great courses
  - Unfortunately most if not all content in those courses are in MalDevAcademy, and more
- Not required, but you're going to find that POCs become harder and harder to find…so make your own!
- Highly recommend C (language of the gods) and MalDevAcademy!

Methodology

- Get friendly with your vendors
  - This is **SUPER** important. We have good relations with a few of our vendors and contacts past tier 1 support that we can get technical with
- Set goals and strategize
  - What do you care about more? Initial access? Post-exploitation? Etc.
  - Intelligence reports help a lot here. Align your PT sessions with your threat model
- Start with default configurations
- Increment evasion one technique at a time to bypass, build detection for it
- It’s worth testing EVERYTHING. Don’t assume anything is working fine.
- Document findings, detection wins and losses, and what you did about it in tools like VECTR (docs.vectr.io)

How Purple Teaming benefits the company...and YOU

- There's nothing like collaboration between red and blue. The company benefits massively from consistent, targeted testing and detection based on your threat model
- Everyone grows and becomes better in their respective fields
  - Red teamers benefit from understanding how detection works, major plus if you can create your own detection for a specific technique
  - Blue teamers benefit from understanding how specific attacks and offensive tooling work, and seeing them used live along with various evasion
  - Some of the best red teamers I know engage in purple teaming and even create detection for the tooling they release, because ultimately the best understand things at such a deep level that they will create new ways to get what they need for an operation
- Great for review time
  - Individual contributions are very easy to point out
  - Less bureaucracy, more action = more results attributed by you

UNIT 43

BSIDES DFW

Purple Teaming: Endpoint Protection

- Preface: I’m a bit biased towards EDR/XDR (Endpoint/Extended Detection and Response) evasion, malleable C2 (Command-and-Control) profiles for network evasion, etc.

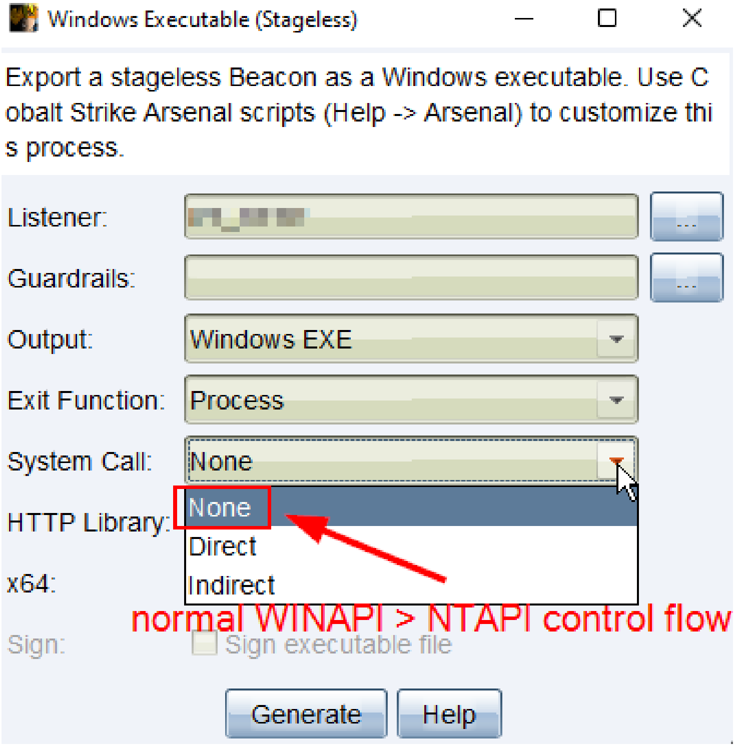
- This isn’t EVERYTHING you should test but could be a good starting point depending on what you value at first.

- Not everything will include a link for detection, but articles are out there and you know the technique!

Implants: WINAPI/NTAPI Usage

- Your EDR probably has detection for this already, still worth testing EVERYTHING!
- Building a Cobalt Strike beacon out-of-the-box will achieve this level of testing, some C2s give you the option (Havoc, Brute Ratel, also Cobalt Strike as of v4.8, etc.)

- Reading Opportunities
  - EDR Userland API Hooking
  - WINAPI vs. NTAPI
  - API Control Flow in Windows
  - Unhooking EDR Userland Hooks



Windows Executable (Stageless)

Export a stageless Beacon as a Windows executable. Use Cobalt Strike Arsenal scripts (Help -> Arsenal) to customize this process.

Listener: [redacted]

Guardrails:

Output: Windows EXE

Exit Function: Process

System Call: None

HTTP Library:

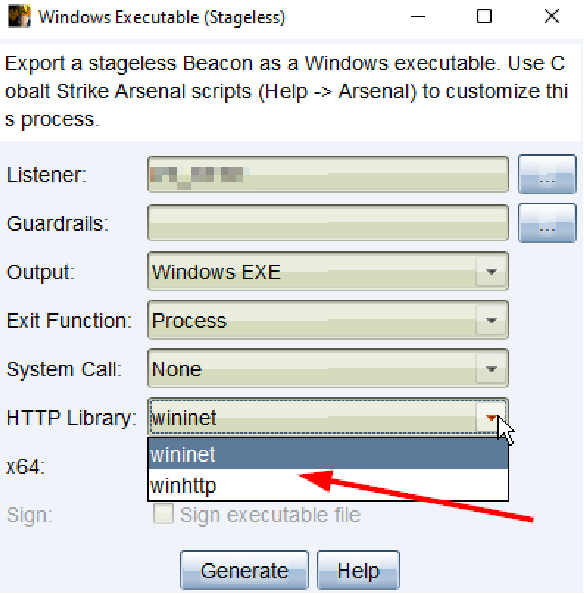
x64:

Sign: □ Sign executable file

normal WINAPI > NTAPI control flow

Implants: Beaconing

- Definitely a detection point on the endpoint side, not just network
- Certain Windows libraries can **potentially** be an IOC of beacon activity, such as WinHTTP which is typically used for server applications. WinINET was designed for user applications, so it looks a lot more natural.
  - Depends heavily on the applications in your environment
  - Cobalt Strike now gives the option for WinHTTP OR WinINET, Brute Ratel uses WinINET, Havoc uses WinHTTP



Windows Executable (Stageless)

Export a stageless Beacon as a Windows executable. Use Cobalt Strike Arsenal scripts (Help -> Arsenal) to customize this process.

Listener: [redacted]

Guardrails:

Output: Windows EXE

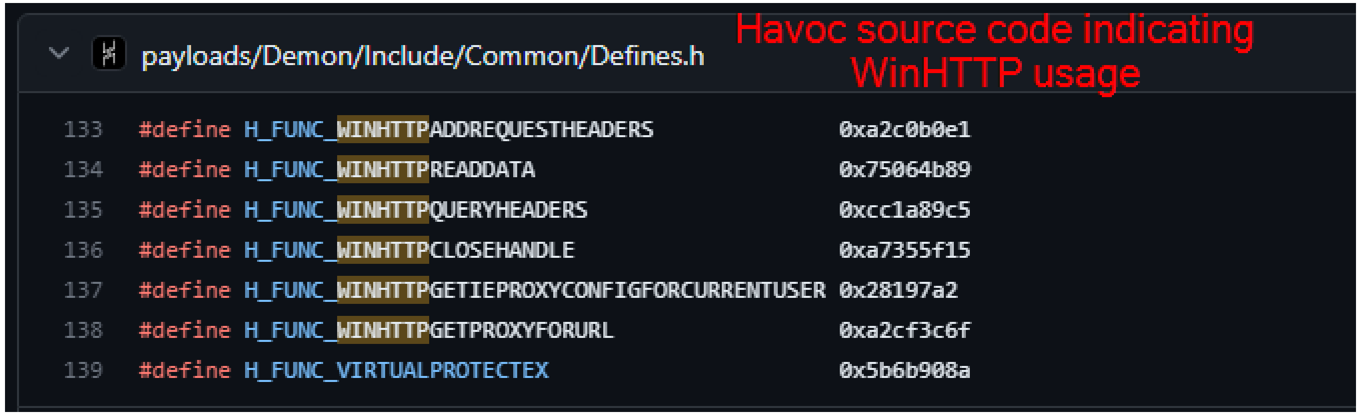
Exit Function: Process

System Call: None

HTTP Library: wininet

x64:

Sign: □ Sign executable file



payloads/Demon/Include/Common/Defines.h

```
#define H_FUNC_WINHTTPADDREQUESTHEADERS                  0xa2c0b0e1
#define H_FUNC_WINHTTPREADDATA                           0x75064b89
#define H_FUNC_WINHTTPQUERYHEADERS                       0xcc1a89c5
#define H_FUNC_WINHTTPCLOSEHANDLE                        0xa7355f15
#define H_FUNC_WINHTTPGETIEPROXYCONFIGFORCURRENTUSER     0x28197a2
#define H_FUNC_WINHTTPGETPROXYFORURL                     0xa2cf3c6f
#define H_FUNC_VIRTUALPROTECTEX                          0x5b6b908a
```

Havoc source code indicating WinHTTP usage

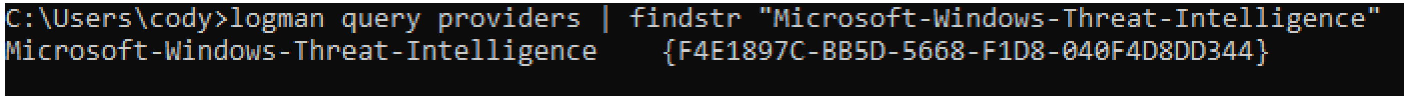
Implants: Unhooking/Evading EDR in Userland

- As mentioned before, your EDR most likely is hooking commonly abused API functions
- A couple of popular ways to get around this
  - Unhooking (restoring the hooked function via reading clean .dll from disk or patching jmp)
  - Direct/Indirect Syscalls
  - Direct Syscalls are probably also caught by your EDR, it’s well-known tradecraft.
    - Easy to detect as the program’s own and/or thread’s start address will point to the memory location of the Syscall rather than the .dll it came from
  - Indirect Syscalls are a bit tricker and may not be caught, but can make the API control flow appear more natural

- Reading Opportunities
  - https://www.ired.team/offensive-security/defense-evasion/bypassing-cylance-and-other-avs-edrs-by-unhooking-windows-apis
  - https://www.ired.team/offensive-security/defense-evasion/retrieving-ntdll-syscall-stubs-at-run-time
  - https://fool.ish.wtf/2022/11/detecting-indirect-syscalls.html

Implants: Event Tracing for Windows (ETW) Patching/Hooking

- May or may not be available with your EDR or security suite
  - ETW is there, but maybe not the threat intelligence provider which is what you’d leverage
  - If it’s there, is your security suite configured to use it?



```
C:\Users\cody>logman query providers | findstr "Microsoft-Windows-Threat-Intelligence"
Microsoft-Windows-Threat-Intelligence    {F4E1897C-BB5D-5668-F1D8-040F4D8DD344}
```

  - Amazing source for building custom detection
- Unfortunately easily bypassable via in-process assembly patching ETW Trace/Write Event calls, hardware breakpoints, etc.

- https://whiteknightlabs.com/2021/12/11/bypassing-etw-for-fun-and-profit/
- https://github.com/Mr-Un1k0d3r/AMSI-ETW-Patch
- https://ethicalchaos.dev/2022/04/17/in-process-patchless-amsi-bypass/
  - AMSI focused but same concept (hardware breakpoints)

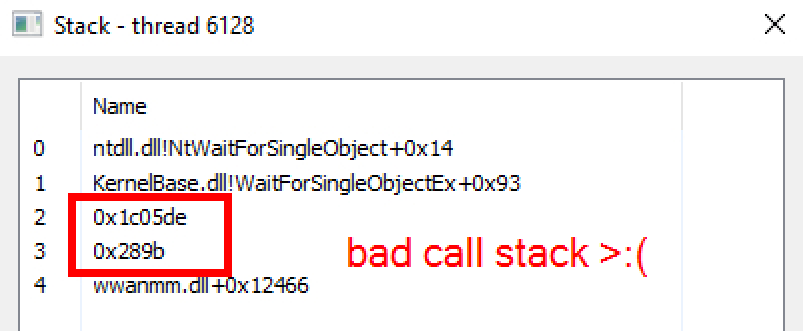
Implants: AMSI Patching/Hooking

- Just another layer of defense, easy to bypass
- Signature based
- Modify your tools
  - Only need a few functions from PowerView? Strip it down to said functions and dependencies
  - Classic rename of mimikatz to mimidogz
- Assembly patch the AMSI check
  - https://github.com/Mr-Un1k0d3r/AMSI-ETW-Patch
- Hardware breakpoints
  - https://ethicalchaos.dev/2022/04/17/in-process-patchless-amsi-bypass/
- https://amsi.fail/ ftw!

- https://www.mdsec.co.uk/2018/06/exploring-powershell-amsi-and-logging-evasion/
- https://0xdarkvortex.dev/red-team-ttps-part-1-amsi-evasion/

Implants: Call Stack Spoofing

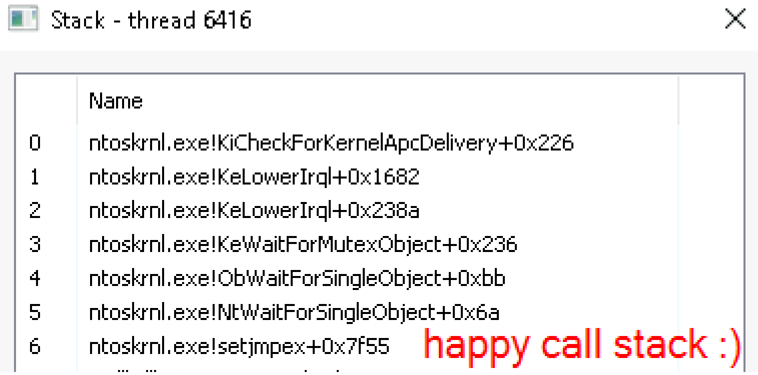
- Everything you need to know, amazing talk by Alessandro Magnosi (Klezvirus)
  - https://www.youtube.com/watch?v=dl-AuN2xsbg
  - x33fcon 2023
- Depending on the implant's execution technique, call stack spoofing is required
  - Ex. Remember with direct syscalls, the call stack points to memory regions containing the Syscall itself



Stack - thread 6128

	Name
0	ntdll.dll!NtWaitForSingleObject+0x14
1	KernelBase.dll!WaitForSingleObjectEx+0x93
2	0x1c05de
3	0x289b
4	wwanmm.dll+0x12466

bad call stack >:(



Stack - thread 6416

	Name
0	ntoskrnl.exe!KiCheckForKernelApcDelivery+0x226
1	ntoskrnl.exe!KeLowerIrql+0x1682
2	ntoskrnl.exe!KeLowerIrql+0x238a
3	ntoskrnl.exe!KeWaitForMutexObject+0x236
4	ntoskrnl.exe!ObWaitForSingleObject+0xbb
5	ntoskrnl.exe!NtWaitForSingleObject+0x6a
6	ntoskrnl.exe!setjmpex+0x7f55

happy call stack :)

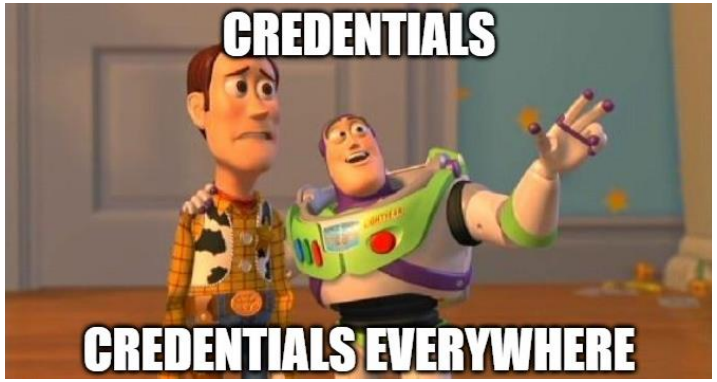
- https://securityintelligence.com/x-force/reflective-call-stack-detections-evasions/
- https://labs.withsecure.com/publications/spoofing-call-stacks-to-confuse-edrs

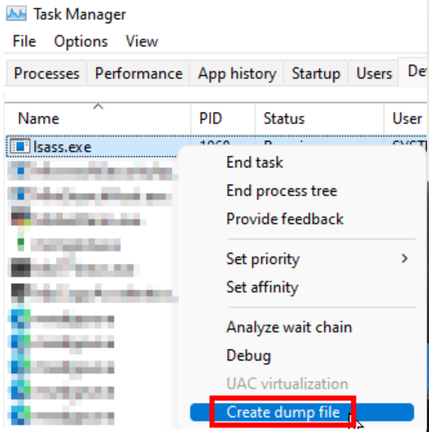
Implants: Process Injection

- Not the most OPSEC friendly on a red team engagement (remote process tampering)
- Primitives can be removed to make it less OPSEC negative
  - Hard to control, requires custom tooling (from what I've discovered)
- Shameless plug
  - https://github.com/ewby/Mockingjay_BOF (credit Security Joes for original article)
    - Allocation primitive removed
  - https://github.com/ewby/ThreadlessInject_BOF (WIP, still good. Credit CCob for original POC)
    - Execution/thread creation primitive removed
- APC injection is a very worthy area to test
  - Remote process injection but in a different way
  - Doesn't rely on the usual process injection API chain, instead waking up sleeping threads
  - https://attack.mitre.org/techniques/T1055/004/
  - https://www.ired.team/offensive-security/code-injection-process-injection/apc-queue-code-injection

Implants: Memory Dumping

- LSASS **should** be covered by most EDR but should be tested anyways!
- Outlook/email clients, browsers, etc.
  - All the credentials!
  - More “modern” tradecraft
  - Not under as much scrutiny as other “secrets” processes
- Most if not all C2 have a generic memdump command
  - Some use basic MiniDumpWriteDump API combos
  - Some use proprietary methods
  - Some straight up run mimikatz
- Advanced Technique Alert: Just call the user
  - Have them click “THAT” button in task manager
  - Mostly funny for the report
  - You should still detect this

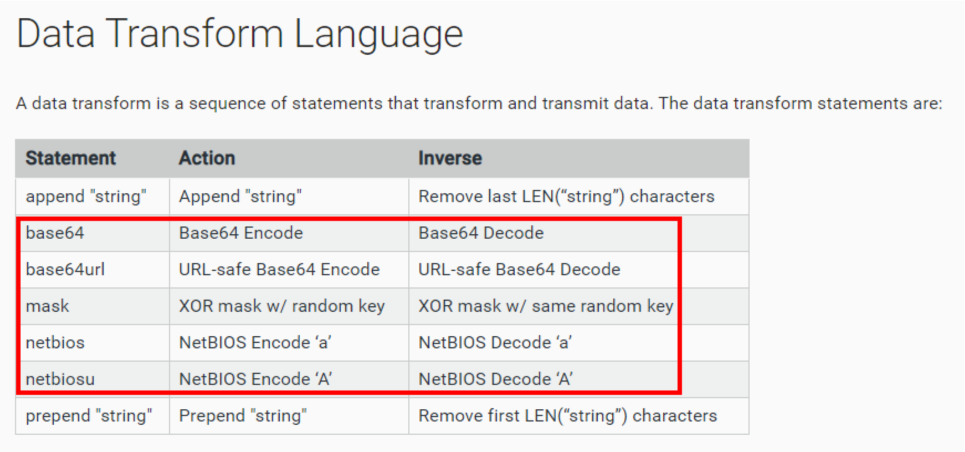




Purple Teaming: Network Detection

Command and Control: Default C2 Network Profiles

- Every C2 has a default configuration, we can at least catch this
  - https://github.com/rsmudge/Malleable-C2-Profiles
  - “Unknown” C2 is much harder to detect
    - Tools like “Burp2Malleable” help with this
      - https://github.com/CodeXTF2/Burp2Malleable
- If your org has the ability to decrypt traffic (depending on privacy laws of course), the body/headers containing beacon metadata are very detectable
  - Some C2s are just base64 encoded + keyed to the server
  - Some C2s have encoding options
- This is where you get to have fun and test them!
- Look up your threat model
- Find what your threat actors are using



Data Transform Language

A data transform is a sequence of statements that transform and transmit data. The data transform statements are:

Statement	Action	Inverse
append "string"	Append "string"	Remove last LEN(“string”) characters
base64	Base64 Encode	Base64 Decode
base64url	URL-safe Base64 Encode	URL-safe Base64 Decode
mask	XOR mask w/ random key	XOR mask w/ same random key
netbios	NetBIOS Encode ‘a’	NetBIOS Decode ‘a’
netbiosu	NetBIOS Encode ‘A’	NetBIOS Decode ‘A’
prepend "string"	Prepend "string"	Remove first LEN(“string”) characters

Command and Control: Heartbeat Detection

- More of a follow up to the previous slide, but super important to point out

- Every C2, when not tasked, has a heartbeat
  - "Hmmm, no task? Check with you in a few!"

- Depending on # of operators, sleep + jitter time, etc., this can be an area of detection

- If TA's are lazy and leave their beacons unattended while sleeping or away, you're (probably) protected or at least raising incidents

- I've tested across multiple versions of a couple C2s, different victims, etc.
  - Seems to be pretty consistent
  - Might not be enough testing on my end but definitely worth exploring

Command and Control: Algorithmic Detection with RITA

- https://github.com/activecm/rita
- Created by Active Countermeasures under Black Hills Infosec
- Quite the tool to go up against as a Red Teamer
- Seemingly resilient to sophisticated attacker infrastructure
- THE C2 network detection tool
  - Trust me, your firewalls aren’t saving you

- Difficult and scary to configure
- Everything is a beacon
  - I’m looking at you Microsoft
- Given enough time and effort, this tool will be one of the security backbones of your org
- Very cheap enterprise features and support with AC-Hunter

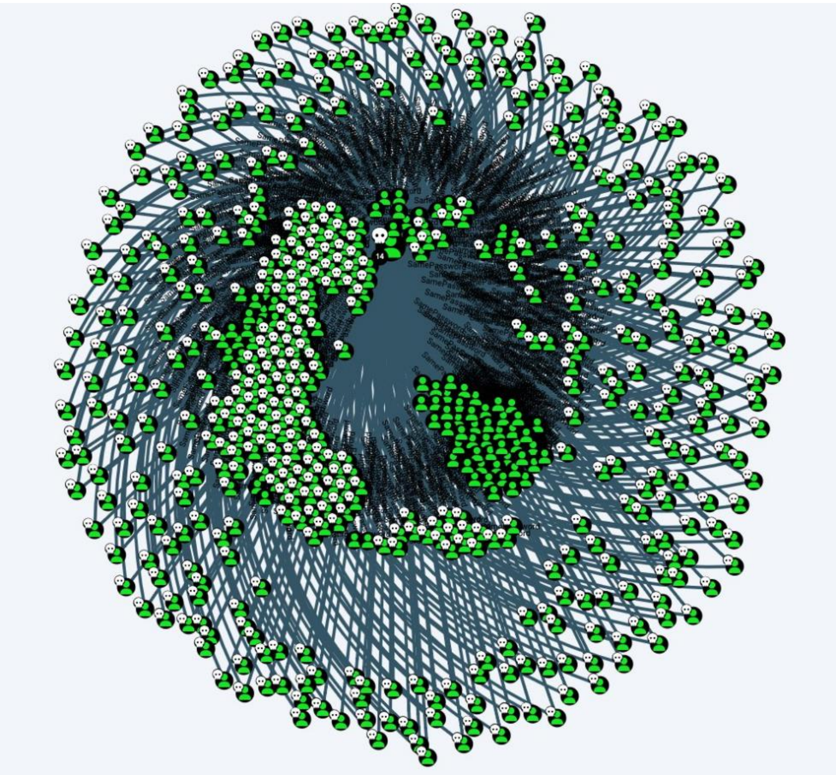
Offensive Tooling: Port Scanning/Local Network Enumeration

- Let's be real most TAs are going to be loud af with this
- Detect high amounts of connections / connection attempts to all ports from a **single source IP**
- For more sophisticated actors, similar logic but specific to popular Windows ports or what is primary in your environment

- Unfortunately not foolproof, but what is?
- Most tooling, including built-in C2 capability, has throttle options

- Depending on your logging (how much and how long), alerts can be tuned to span longer periods of time, defeating stealth scanning

Offensive Tooling: LDAP Enumeration



- Similar to Port Scanning
- OH LAWD HE COMIN
- NOT JUST BLOODHOUND

- Massive amounts of LDAP, SMB, RPC, etc. traffic
- Again, from a **single source IP**
  - Well, sometimes
  - We're trying to catch the 90%
  - LDAP tooling/queries can be split amongst beacons

- "DCOnly" is a thing, in Bloodhound at least
- How do I know which DCs will be interacted with???
- Asset group DCs together in Splunk, detect abnormal amounts of traffic to the group

- "ComputerOnly" is a thing too. Something to keep in mind.

How have I benefited as a Red Teamer?

- Being face to face with security solutions day in and day out has worked wonders for my skills as a red teamer
  - Pushing yourself to beat a specific security control, building custom tools, etc.
- A lot of red teaming is plannin', and a little (depending) doin'. A lot of purple teaming is a little plannin' (there's so much to do, it's basically planned for you!) and a LOTTA doin', if you plan properly and stick to it!
- Collaboration is a great thing. It's looked good on me, especially given the history of red vs. blue, and it's made me an even better red teamer by understanding and helping with the blue side
- Chances are no one is doing this at your org, you get a chance to step up and be a leader
- Contacts within top security vendors is very nice, for various support and lab goodies
- Maybe…the real treasure is the friends we make along the way?!?

Q/A, wakey wakey!