Check Point ZoneAlarm Extreme Security Arbitrary File Move Elevation of Privilege

Summary

This vulnerability allows local attackers to escalate privileges on hosts where affected installation of Check Point ZoneAlarm Extreme Security is running. An attacker must first obtain the ability to execute low privileged code on target host to exploit this vulnerability.

This specific flaw exists due to weak privileges in C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates directory and self-protection driver bypass which allowed creation of junction directory which was abused to perform arbitrary file move as NT AUTHORITY\SYSTEM account.

Description

When ZoneAlarm Extreme Security is installed on Windows host, it creates several file/directories with weak permissions in C:\ProgramData\ directory.

```
PS C:\Users\lab-user> icacls C:\ProgramData\CheckPoint\ZoneAlarm\Data
C:\ProgramData\CheckPoint\ZoneAlarm\Data NT AUTHORITY\SYSTEM:(OI)(CI)(F)
                                         BUILTIN\Administrators:(OI)(CI)(F)
                                         BUILTIN\Administrators:(F)
                                         CREATOR OWNER:(OI)(CI)(IO)(F)
                                         BUILTIN\Users:(OI)(CI)(RX)
                                         BUILTIN\Users:(OI)(CI)(WD,AD,WEA,WA)
                                         NT AUTHORITY\Authenticated Users:(F)
                                         NT AUTHORITY\Authenticated Users:(OI)(CI)(IO)(F)
                                         NT AUTHORITY\SYSTEM:(I)(OI)(CI)(F)
                                         BUILTIN\Administrators:(I)(OI)(CI)(F)
                                         CREATOR OWNER:(I)(OI)(CI)(IO)(F)
                                         BUILTIN\Users:(I)(OI)(CI)(RX)
                                         BUILTIN\Users:(I)(CI)(WD,AD,WEA,WA)

Successfully processed 1 files; Failed processing 0 files
PS C:\Users\lab-user>
```

Even with weak permissions low privilege users are not allowed to write files/directories due to self-protection driver.

Self-protection driver will allow low privilege user to open handle on files/directories if FILE_WRITE_ATTRIBUTES access is requested which will allow creation of junction directories.

In this scenario junction directory will be abused to redirect file operation during the update process.

When update process is started vsmon.exe process is looking for LocalCatalog.xml file in C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates directory (this directory is created after first update process, which should happen automatically each hour by default).

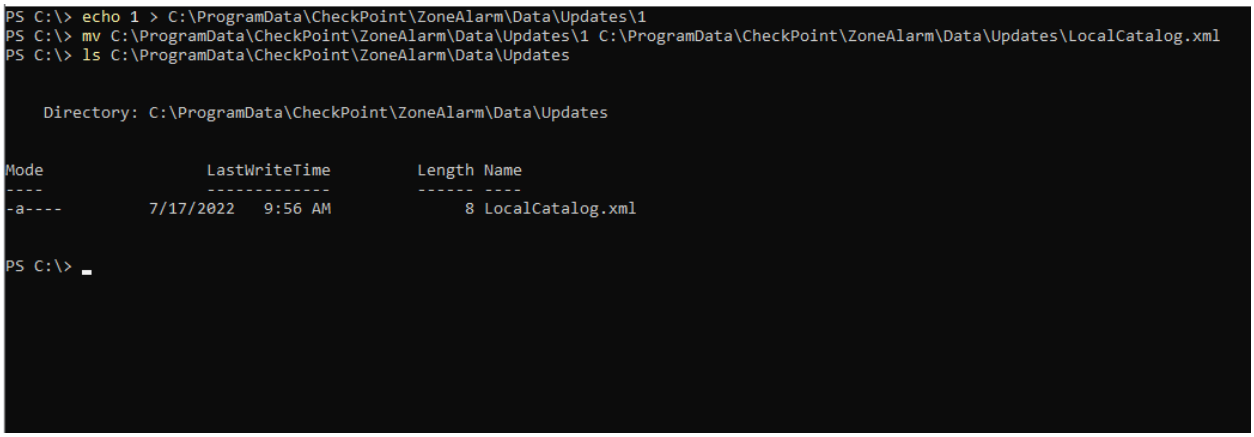
2:34:5...	vsmon.exe	14212	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	NO SUCH FILE	FileInformationClass: FileBothDirectoryInformation, Filter: LocalCatalog.xml	NT AUTHORITY\SYSTEM
2:34:5...	vsmon.exe	14212	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	NO SUCH FILE	FileInformationClass: FileBothDirectoryInformation, Filter: LocalCatalog.xml	NT AUTHORITY\SYSTEM

If file LocalCatalog.xml is not found vsmon.exe will create a file with following name Downloading-<randomnumbers>.xml which is deleted immediately after creation (this behavior can also be abused for arbitrary file delete).

If file LocalCatalog.xml is found and is not valid xml file vsmon.exe will rename LocalCatalog.xml file to LocalCatalog.xml.bad as shown below:

Time ...	Process Name	PID	Operation	Path	Result	Detail	User
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: Updates, 2: Updates	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	Desired Access: Read Data/List Directory, Synchronize, Disposition: Open, Options: Direct...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: LocalCatalog.xml, 2: LocalCatalog....	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS		NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: Synchronous IO Non-Alert, Non...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	ReadFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	Offset: 0, Length: 8, Priority: Normal	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	ReadFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	END OF FILE	Offset: 8, Length: 12,288	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS		NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml.bad	NAME NOT FOUND	Desired Access: Read Attributes, Delete, Disposition: Open, Options: Non-Directory File, Op...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: Updates, 2: Updates	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: , Attributes: n/a, ShareMode: R...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml.bad	NO SUCH FILE	FileInformationClass: FileBothDirectoryInformation, Filter: LocalCatalog.xml.bad	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS		NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	Desired Access: Read Attributes, Delete, Synchronize, Disposition: Open, Options: Synchro...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: Updates, 2: Updates	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: , Attributes: n/a, ShareMode: R...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: LocalCatalog.xml, 2: LocalCatalog....	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS		NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryBasicInfor...	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	CreationTime: 7/17/2022 9:56:38 AM, LastAccessTime: 7/17/2022 9:59:27 AM, LastWrite...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	Desired Access: Write Data/Add File, Synchronize, Disposition: Open, Options: Open For B...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryStandardI...	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	AllocationSize: 8, EndOfFile: 8, NumberOfLinks: 1, DeletePending: False, Directory: False	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml.bad	NAME NOT FOUND	Desired Access: Synchronize, Disposition: Open, Options: , Attributes: N, ShareMode: Read...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	SetRenameInfo...	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	ReplaceIfExists: False, FileName: C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: Updates, 2: Updates	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CreateFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: , Attributes: n/a, ShareMode: R...	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	QueryDirectory	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml	SUCCESS	FileInformationClass: FileBothDirectoryInformation, Filter: LocalCatalog.xml, 2: LocalCatalog....	NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS		NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates	SUCCESS		NT AUTHORITY\SYSTEM
9:59:2...	vsmon.exe	12112	CloseFile	C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates\LocalCatalog.xml.bad	SUCCESS		NT AUTHORITY\SYSTEM

Normal users “can’t” create file LocalCatalog.xml in Updates directory cause self-protection driver will block operation, but it can be bypassed by creating file with different name and renaming it to LocalCatalog.xml



This can also be a valid exploitation path if its possible to craft valid config file but that seemed like a lot of work.

Exploitation

Ability to create junction directory and file rename operation performed by vsmon.exe process create an arbitrary file move primitive which can be abused to load dll's in privileged process.

To abuse this primitive PoC code I created does the following:

It first creates two directories

C:\expl

C:\expl2

In C:\expl directory two files are created

LocalCatalog.xml

LocalCatalog.xml.bad

In C:\expl2 directory a dll, that will be written to system32 directory, is copied.

Junction directory is created from C:\ProgramData\CheckPoint\ZoneAlarm\Data\Updates to C:\expl.

Opportunistic Lock is set on LocalCatalog.xml.bad file. Opportunistic lock will prevent other processes to open LocalCatalog.xml.bad file until we close handle on it, this will give us time to remove files from c:\expl directory.

When update process is started vsmon.exe will try to open LocalCatalog.xml.bad file in c:\expl directory, this when our oplock is triggered and we remove all files from c:\expl directory.

Once files are removed new junction directory is created that points from c:\expl to \RPC Control (Object Manager).

We then create two object manager symbolic links:

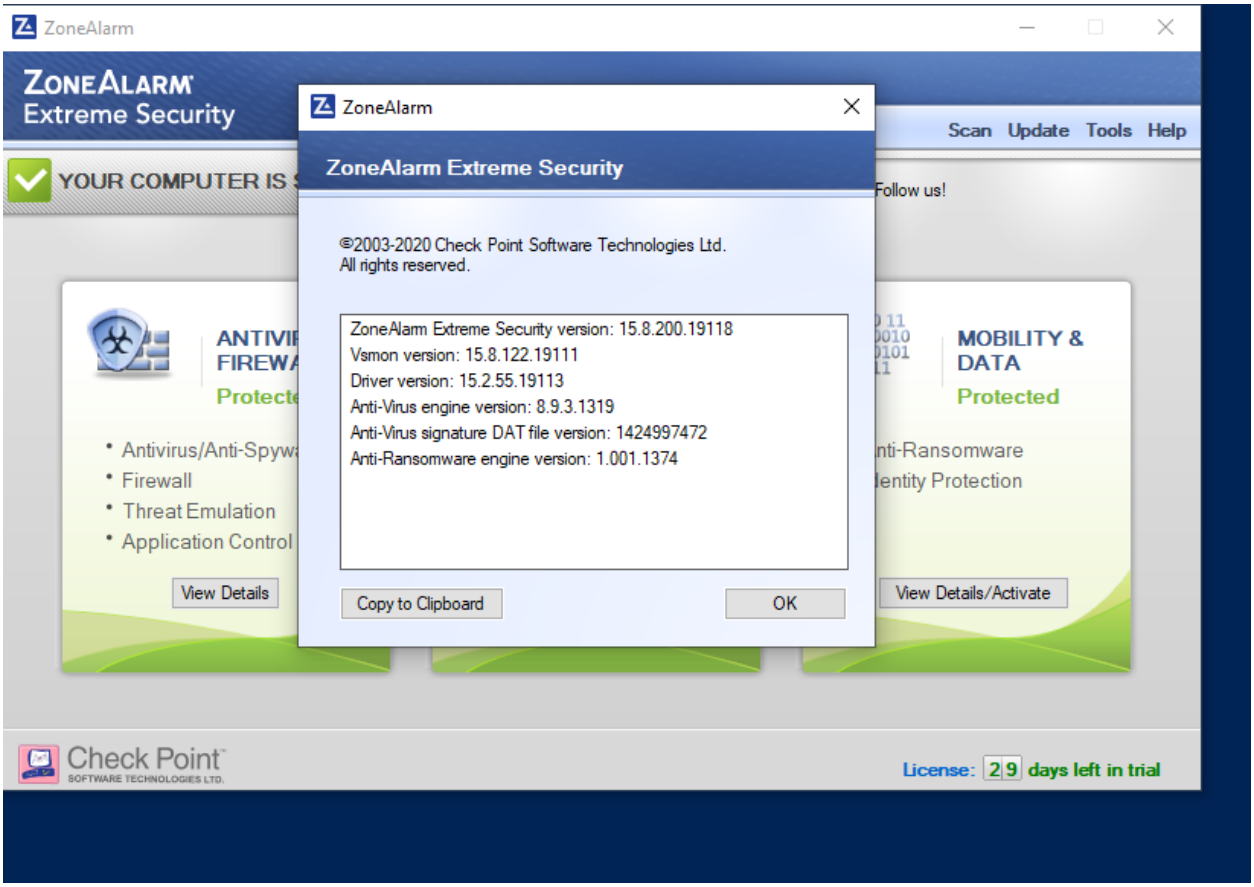
\RPC Control\LocalCatalog.xml – points to dll we want to write in system32

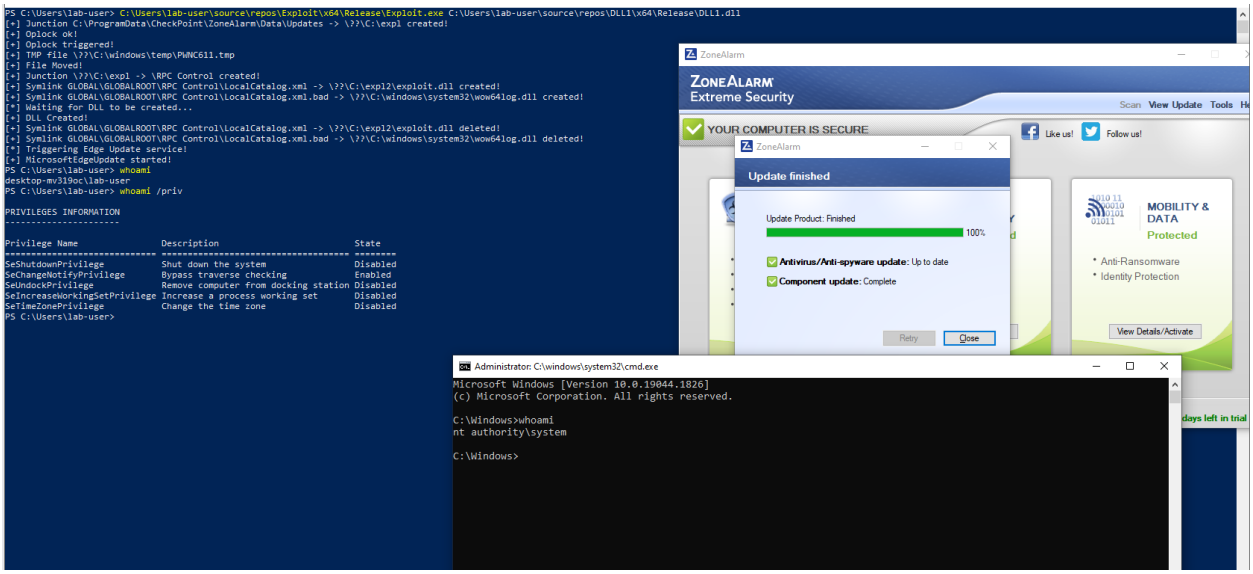
\RPC Control\LocalCatalog.xml.bad – points to file in system32 directory

Once symbolic links are created, we release oplock and vsmon.exe will copy our dll to system32 directory.

For this PoC I choose to write a wow64log.dll in system32 directory, this dll is loaded by Microsoft Edge Update service which runs in NT AUTHORITY\SYSTEM context and comes installed by default on Windows 10 (ofc other dll's can be used I simply choose this one) and can be started by creating instance of COM object.

In images below you can see PoC in action on latest version of Extreme Security:





Proof of Concept

As part of this vulnerability report I am sending a PoC in form of Visual Studio solutions and compiled binaries.

Use next steps to compile source code for Exploit.sln

1. Unzip Exploit.zip file
2. Open the solution with Visual Studio 2022 (you can also use 2019 but will need to change platform toolset from v143 to v142 if v143 is not installed).
3. Select Release + x64.
4. Compile Exploit project which will generate Exploit.exe binary.

Use next steps to compile source code for Dll1.sln

1. Unzip Dll1.zip file
2. Open the solution with Visual Studio 2022 (you can also use 2019 but will need to change platform toolset from v143 to v142 if v143 is not installed).
3. Select Release + x64.
4. Compile Dll1 project which will generate Dll1.dll binary.

Once both projects are compiled run PoC as shown below:

Exploit.exe <path to Dll1.dll>

Once exploit it started start update process by clicking **Update** in Extreme Security GUI.

If exploitation is successful a cmd.exe will appear on desktop with SYSTEM privileges.