**Windows Kernel permission promotion vulnerability**

**(CVE-2023-35359)**

**Deep analysis report**

**Catalogue**

[1. Basic information 1](#_Toc148018158)

[2. Threat assessment 2](#_Toc148018159)

[3. The security status of the affected assets 3](#_Toc148018160)

[4. Disposal advice 3](#_Toc148018161)

[5. The full utilization process 4](#_Toc148018162)

[6. Use of monitoring and protection 8](#_Toc148018163)

1. Basic information

|  |  |  |  |
| --- | --- | --- | --- |
| **Vulnerability name** | Windows Kernel permission promotion vulnerability | | |
| **Open time** | 2023-08-08 | **refresh time** | 2023-10-12 |
| **CVE number** | CVE-2023-35359 |
| **Threat type** | Permission promotion | **Technology type** | Improper permissions, privileges, and access controls |
| **Manufacturer name** | Microsoft | **Product name** | Windows |
| **Real threat state** | | | |
| **POC state** | **EXP state** | **Exploited in the wild** | **Technical detail status** |
| **publiced** | **publiced** | None | **publiced** |
| **Vulnerability description** | When an unprocessed exception occurs to Windows, the program will attempt to wake the Windows Error Reporting (WER) service for logging and analysis. When the up fails, the fault program creates a WerFault. The exe child process to collect program-specific information. When the fault program is to simulate the privileged process of the current user, the forged DOS device mapping can be used to create the hijacking process and execute arbitrary code with high integrity, finally achieving the authority promotion. | | |
| **Influence version** | Windows Server 2012 R2 (Server Core installation)  Windows Server 2012 R2  Windows Server 2012 (Server Core installation)  Windows Server 2012  Windows Server 2008 R2 for x64-based Systems Service Pack 1 (Server Core installation)  Windows Server 2008 R2 for x64-based Systems Service Pack 1  Windows Server 2008 for x64-based Systems Service Pack 2 (Server Core installation)  Windows Server 2008 for x64-based Systems Service Pack 2  Windows Server 2008 for 32-bit Systems Service Pack 2 (Server Core installation)  Windows Server 2008 for 32-bit Systems Service Pack 2  Windows Server 2016 (Server Core installation)  Windows Server 2016  Windows 10 Version 1607 for x64-based Systems  Windows 10 Version 1607 for 32-bit Systems  Windows 10 for x64-based Systems  Windows 10 for 32-bit Systems  Windows 10 Version 22H2 for 32-bit Systems  Windows 10 Version 22H2 for ARM64-based Systems  Windows 10 Version 22H2 for x64-based Systems  Windows 11 Version 22H2 for x64-based Systems  Windows 11 Version 22H2 for ARM64-based Systems  Windows 10 Version 21H2 for x64-based Systems  Windows 10 Version 21H2 for ARM64-based Systems  Windows 10 Version 21H2 for 32-bit Systems  Windows 11 version 21H2 for ARM64-based Systems  Windows 11 version 21H2 for x64-based Systems  Windows Server 2022 (Server Core installation)  Windows Server 2022  Windows Server 2019 (Server Core installation)  Windows Server 2019  Windows 10 Version 1809 for ARM64-based Systems  Windows 10 Version 1809 for x64-based Systems  Windows 10 Version 1809 for 32-bit Systems | | |
| **Other affected components** | None | | |

2. Threat assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CVSS 3.1 Rating** | **high risk** | **The CVSS 3.1 Score** | | 7.8 |
| **CVSS vector** | **Access route (AV)** | | **Attack complexity (AC)** | |
| this locality | | low | |
| **Required permission (PR)** | | **User Interaction (UI)** | |
| non-essential | | non-essential | |
| **incidence (S)** | | **Confidentiality Impact (C)** | |
| no change | | high | |
| **Integrity Impact (I)** | | **Availability Effects (A)** | |
| high | | high | |
| **Hazard description** | The authenticated local attacker uses this vulnerability to increase access to SYSTEM. | | | |

3. The security status of the affected assets

Not available

4. Disposal advice

|  |  |  |
| --- | --- | --- |
| **Self-examination and testing scheme** | 1. Pass version detection (recommended to update to the latest version)   2, by PoC | |
| **Automated vulnerability scanning method** | Not available | |
| **Repair mitigation measures** | Please keep automatic updates enabled, or refer to the following link to upgrade to the secure version: https: / / msrc.microsoft.com/update-guide/vulnerability/CVE-2023-35359 | |
| **Repair solution (including vulnerability patches)** | Microsoft has now released a security update, and users can obtain it through the Windows update. If the update cannot be updated automatically, please refer to the following link to download and install the corresponding version of the security patch:  https://msrc.microsoft.com/update-guide/vulnerability/CVE-2023-35359 | |
| **The effects of the repair caused** | Need to restart the operating system | yes |
| Need to restart the application system | deny |
| other |  |

5. The full utilization process

|  |  |
| --- | --- |
| **EXP/POC** | See the attachment for the PoC. |
| **Use details to describe** | Windows Permission promotion vulnerability in the kernel, due to Windows, during simulation call LoadLibrary can load any DLL, or create process (CreateProcessW), with low permissions attacker can the system32 folder redirected to any position, which can execute arbitrary code to simulate high permission, eventually the low permissions attacker rights to SYSTEM. Low permission users can be triggered by multiple paths to simulate a high permission call LoadLibrary to load the DLL.  Including, but not limited to, the following three categories:  1. Windows Error Reporting (WER) Services  When an unprocessed exception occurs on the Windows, the program involuntarily attempts to wake up the Windows Error Reporting (WER) service for recording and analysis.  In the case that a wake call fails, such as when the service is explicitly marked as Disabled, the wrong program creates an WerFault. The exe subroutine to collect program-specific statistics. When the error, program is, simulate high permission process and meet: WER service marked as Disabled, privilege process P analog medium IL user, simulated untreated exception in P, will be able to use deceptive DOS device mapping to hijack process creation, and execute arbitrary code with high integrity.  Windows Unhandled exception backback to the default exception handler KernelBase! UnhandledExceptionFilter, Pass the execution to the Kernel32! BasepReportFault To report a crash, and finally call the Kernel32! WerpReportFaultInternal,  Function calls ntdll! RtlWerpReportException, It is responsible for waking up the WER service and sending messages to it.  If the call is successful, the service program WerSvc! CWerService:: The SvcReportCrash will be called to organize the crash statistics.  Finally, a thread is created to export the Faultrep with a secondary DLL before the fault process ends! CreateCrashVerticalProcess Start the WerFault. The exe acts as a child process and eventually calls the CreateProcessAsUserW function. In cases where the service is explicitly disabled, the ntdll! RtlWerpReportException Cannot wake up the service. Kernel 32! WerpReportFaultInternal Will call their own CreateCrash, namely Kernel32! StartCrashVertical. Kernel32! StartCrashVertical Call CreateProcessW, and CreateProcessW uses an analog token when trying to find the process image. With this high permission simulation token, you can eventually raise the permission to system.  2. B ITS Services  CVE-2017-0213: The BITS service operates on the SYSTEM IntegrityLevel (IntegrityLevel). If it is called  IBackgroundCopyJob::SetNotifyInterface (IUnknown \*pNotifyInterface )  And upload a carefully constructed COM interface, causing type confusion, you can exploit the vulnerability to load a TypeLibrary. Later, I use TypeLibrary to load tapi3.dll, and link tapi3.dll to the tapi3.dll defined by the attacker through the symbol link, so as to achieve the permission promotion.  The patch of CVE-2017-0213 fixes type confusion and disables simulation when the type library is loaded, taking no action on the simulation problem, so you can pass an object that supports a specific interface (such as comuid. Interface ITrkEvntObjControl in dll), which causes the COM runtime to create the agent instance from the selected DLL with the BITS SetNotifyInterface analog caller. The loading of the DLL causes the embedded manifest to be loaded, without looking for simulation operations, so CSRSS will read the manifest modified by the vulnerability exploit, including it into the dependency of a custom path which includes a new custom manifest. This custom manifest then loads the modified malicious DLL,  The malicious DLL loads into the system service and gets the promoted permission.     1. Windows File History service.   The file history service can be started by the normal user, and when the service starts, the core file fhsvc is loaded when the service starts. The dll, and then hit the vulnerable function CManagerThread:: QueueBackupForLoggedOnUser. When performing this function it will simulate the currently logged in user and load fhcfg.dll 。    When loading the fhcfg. When dll, the thread will run as the current ordinary user.fhcfg. The resource for the dll contains the manifest properties. Once the DLL loading is complete, the csrss. The exe will create the default activation context based on the dependencies in the manifest file to automatically load the desired assembly because the thread for the file history service is in analog state so csrss. The exe will also simulate the identity of ordinary users to access the manifest files.  When the normal user changes and adds a C: symbol link to a false directory (for example), the csrss. Exe, will look for C: as a list file. After setting up the symbol link, the csrss. The exe will look for it based on the content list of the manifest file. The file name of the manifest file will vary depending on the operating system environment, such as:  csrss.exeC:\Users\Public\test\Windows\WinSxS\Manifests\amd64\_microsoft.windows.common-controls\_6595b64144ccf1df\_6.0.22621.1635\_none\_270f70857386168efhcfg.dll.manifest    If you write the false DLL path directly in the false manifest, then still cannot exploit because the loaded dll, will come from C: \ Windows \ WinSxS \ Manifestsm, which is not a controllable directory, so need a second manifest and change the file properties first to archive the list name=...\..\..\..\..\..\test\test 。  When the file attribute of the false list content is a file name=...\..\..\..\..\..\test\test ，csrss. The exe will continue to search for the second command list, C: \ Users \ Public \ test \ test \ test.manifest 。  Found that the msasn1.dll is loaded when the File History Service is about to exit (the File History Service runs for 30 seconds by default).  Build a false list test.manifest And add a DLL named msasn1.dll dependency, then msasn1.dll tries to open C:\test\msasn1.dll when it is generated.  Since the file history service does not have a permission SeIncreaseQuotaPrivilege, it cannot directly pop up the cmd window that can be displayed, but the service has a permission SeImpersonatePrivilege. You can add a scheduled task and specify an exe start, then it will start with the default permissions of the system account. |
| **Use the description results and screenshots** | POC 1：  Windows 10 22H2 and Windows 11 22H2 installed under the May 2023 patch  The directory structure of the poc is is as follows:  C:\fakeroot\PoC\_BITS.exe  C:\fakeroot\sspicli.dll  C:\fakeroot\fakeroot\EXPLOIT.MANIFEST  C:\fakeroot\fakeroot\sspicli.dll  C:\fakeroot\Windows\system32\comuid.dll  C:\fakeroot\Windows\WinSxS\amd64\_microsoft.windows.common-controls\_6595b64144ccf1df\_6.0.19041.1110\_none\_60b5254171f9507e  C:\fakeroot\Windows\WinSxS\amd64\_microsoft.windows.common-controls\_6595b64144ccf1df\_6.0.22621.1635\_none\_270f70857386168e  C:\fakeroot\Windows\WinSxS\Manifests\amd64\_microsoft.windows.common-controls\_6595b64144ccf1df\_6.0.19041.1110\_none\_60b5254171f9507e.manifest  C :\fakeroot\Windows\WinSxS\Manifests\amd64\_microsoft.windows.common-controls\_6595b64144ccf1df\_6.0.22621.1635\_none\_270f70857386168e.manifest    POC2：  The file needs to be placed in the same directory |

Vi. Use of monitoring and protection

**6.1 Threat hunting ideas and methods**

Not available

**6.2 Detection and alarm rules and protection strategies on the safety equipment side**

Not available