Počítačové viry a bezpečnost počítačových systémů
Protokol z předmětu (2b)

Tématická oblast: Windows API, registry, oprávnění

Přednášející: prof. Ing. Ivan Zelinka, Ph.D.; Ing. Jan Plucar, Ph.D.
Cvičící: Ing. Jan Plucar, Ph.D.

Jméno a číslo studenta: Daniel Trnka, TRN0038
Datum vypracování: 2. 3. 2019

Zadání:

1) Seznamte se s Windows API z pohledu programátora.
2) Seznamte se s Windows registry a jejich použitím přes Windows API.
3) Zjistěte, jak je možné automaticky spouštět aplikace po startu Windows (především pomocí registrů).
4) Rozšiřte Váš keylogger z minulého cvičení - keylogger při startu zjistí, zda v registrech existuje záznam, který spouští tento program:
    a. Neexistuje-li záznam, který by spustil program z aktuálního umístění, vytvořte jej.
    b. Existuje-li záznam, který spouští Vámi vytvořený program z umístění, které ovšem již neexistuje, nahraďte jej cestou k aktuálnímu souboru.
    c. Existuje-li záznam s hodnotou odkazující na existující umístění Vašeho keyloggeru, neprovádějte žádnou akci.
5) Dále rozšiřte keylogger o tuto funkci: Vyžádejte oprávnění administrátora a přes shell vypněte firewall.

Závěr:
Do tohoto protokolu nemusíte vkládat screenshoty. Společně s protokolem ovšem odevzdejte zdrojové kódy Vašeho keyloggeru. Binární verze programů neodevzdávejte.

Diskutujte následující témata:
1) Jmenujte další způsoby, jak je možné spouštět malware při/po startu systému.
2) Co je UAC? Jak funguje a k čemu slouží?
3) Diskutujte metody, jakými může malware získat administrátorská oprávnění na Windows systémech (je-li aplikace spuštěna pod běžným uživatelem).

Program z předchozího cvičení byl rozšířen o automatické spuštění po startu systému, resp. po přihlášení daného uživatele. Existuje několik možností jak zajistit spouštění malware po startu systému.

První z možnosti, jak spustit program po přihlášení, je vytvořit v registrech nový klíč s prefixem

```
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
```

obsahující cestu k programu. Stejná funkcionalita existuje i v `HKEY_LOCAL_MACHINE`, ale zápis vyžaduje administrátorské oprávnění, protože se aplikuje na všechny uživatele.

Další možnosti je zkopírovat soubor do startup adresáře, který je ve výchozím nastavení:

```
%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup
```

Využití služby není možné, protože lze nainstalovat hook pouze u procesů, které byly spuštěné "interaktivním" způsobem.

AppInit_DLL umožňuje namapovat vlastní DLL knihovnu do všech interaktivních aplikací[1]. V adresáři `c` je zdrojový kód knihovny zachytávající klávesy, které jsou následně uloženy do souboru. Procesy s `user32.dll` si danou knihovnu namapují a tím se zavolá funkce `DllMain`. Funkce spustí nové vlákno a zaregistruje hook `WH_KEYBOARD_LL` ve kterém zachytnuté klávesy ukládá do souboru.

Aby se knihovna načítala v každém procesu tak je nutné v registrech upravit dva klíče s prefixem

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT \CurrentVersion\Windows
```

Prvním klíčem je `AppInit_DLLs`, který obsahuje cestu k DLL knihovně a druhý klič `LoadAppInit_DLLs` musí být nastaven na hodnotu 1, aby byla tato funkcionalita povolena. Změnu v registrech je nutné provést jako administrátor a systém musí nabootovat bez zapnutého SecureBoot.

Dalším úkolem bylo vypnout firewall. V C# keyloggeru se spustí nový `powershell.exe` proces se skrytým oknem a v argumentech se předá skript z výpisu 1. Skript prvně vyhledá zapnuté firewall profily. Pokud nějaký zapnutý firewall existuje tak se spustí nový `powershell.exe` s administrátorským oprávněním a spustí se vypnutí firewallu. Uživatel je tak pomocí UAC dotázán pouze v případě zapnutého firewallu.

Aby se ztížila analýza programu, tak je daný powershell skript uložen v šifrované podobě, kde klíč se získá postupným zkoušením všech cest souborů

[1] https://docs.microsoft.com/en-us/windows/desktop/dlls/secure-boot-and-appinit-dlls

```
$disabled = Get-NetFirewallProfile -All `
 | Where-Object -Property Enabled -EQ true `
 | measure;

if ($disabled.Count -gt 0) {
 echo "will disable firewall";
 Start-Process powershell `
  -WindowStyle Hidden `
  -Verb runAs `
  'Get-netFirewallProfile -All `
    | where Enabled -eq True `
    | foreach { Set-NetFirewallProfile -Profile $_.Name -Enabled false }';
}
```

Výpis 1: Powershell skript pro vypnutí firewallu dle potřeby

na disku. Ve zdrojových kódech se jedná o klíč `C:\Windows\explorer.exe`. Touto metodou lze ale cílit na konkrétní skupinu počítačů na kterých se daný malware aktivuje a při běhu v sandboxu neposkytne žádná data.

Díky **User Account Control (UAC)** lze spouštět aplikace s nižším oprávněním a až dle potřeby se zobrazí potvrzovací dialogové okno pro udělení administrátorských práv. Cílem je, aby aplikace neběžely s plnými právy a uživatel měl možnost případně odmítnout. Ne vždy je ale dostatek informací k rozhodnutí zda povolit či ne - keylogger spouštěl nový proces `powershell.exe` s administrátorským oprávněním, ale v UAC dialogovém okně (obrázek 1) bylo pouze zobrazena informace o powershell od Microsoft, což může působit důvěryhodně a navíc může jakákoliv legitimní aplikace pustit powershell.

Aplikace běžící pod normálním uživatelem může naivně získat administrátorská oprávnění vytvořením nového procesu s `verb = runas`. Ve výchozím nastavení však musí uživatel potvrdit UAC dialogové okno, neznalý uživatel však může automaticky vše potvrzovat. V jazyce C# může spuštění procesu s administrátorským oprávněním vypadat následovně:

```
var proc = new ProcessStartInfo();
proc.FileName = "powershell.exe";
proc.Verb = "runas";
Process.Start(proc);
```

Další možnosti by mohlo být využití programu `runas.exe`, ale vypadá to, že vyžaduje zadat heslo administrátora ručně. Dále se může aplikace rovnou

Obrázek 1: UAC potvrzovací okno bez informacemi o předkovi, který daný proces vytvořil

při spuštění zažádat o administrátorské oprávnění. Lze to docílit přidáním manifestu z výpisu 2 do výsledného spustitelného souboru.

Dále lze například vyzkoušet zachytnutá hesla, zda nejsou použitá i pro administrátorský účet. Využít špatně nakonfigurovaný systém - příliš volná přístupová práva do systémových adresářů. Nebo využít nějakou chybu v systému.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
 <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
  <security>
   <requestedPrivileges>
        <requestedExecutionLevel
          level="requireAdministrator"
          uiAccess="false"/>
   </requestedPrivileges>
  </security>
 </trustInfo>
</assembly>
```

Výpis 2: Manifest pro zažádání administrátorského oprávnění hned při startu aplikace