Network Enumeration and Port Scanning

در این فصل، چگونگی استفاده از PowerShell را به عنوان بخشی از تست نفوذ برای انجام اسکن پورت TCP/UDP بررسی خواهیم کرد. از طریق مثال‌های عملی، چگونگی استفاده از cmdletهای خاص را بررسی خواهیم کرد و اینکه PowerShell چگونه می‌تواند از توابع در چارچوب‌های دیگر مانند .NET استفاده کند. در نهایت، برخی از ابزارهای متن باز را که می‌توان از آن‌ها برای انجام اسکن پورت استفاده کرد، بررسی خواهیم کرد. اکنون این فصل را با بررسی چگونگی استفاده از cmdlet های مختلف برای انجام Enumeration شبکه آغاز می‌کنیم.

موضوعات زیر مهمترین موضوعاتی است که در این فصل به آنها خواهیم پرداخت:

- Network enumeration using PowerShell
- TCP port scanning using PowerShell
- UDP port scanning using PowerShell
- Using PowerShell tools for port scanning

Network enumeration using PowerShell

Network Enumeration یک مرحله مهم در تست نفوذ است که در آن متخصصان امنیتی سعی دارند اطلاعات بیشتری را در خصوص شبکه هدف بدست آورند. PowerShell، یک زبان برنامه نویسی و فریمورک قدرتمند در محیط های ویندوز، می تواند ابزاری ارزشمند برای انجام Network Enumeration باشد. در اینجا، نحوه استفاده از PowerShell برای این منظور در تست نفوذ را بررسی خواهیم کرد:

Discovery of network assets: PowerShell به تست نفوذگران اجازه می‌دهد تا دارایی‌های شبکه مانند Host ها، سرورها و دستگاه‌ها را کشف کنند. دستوراتی مانند Test-Connection را می‌توان برای Ping کردن Host ها و بررسی در دسترس بودن آن‌ها به کار برد. Resolve-DnsName می‌تواند به شناسایی نام هاست کمک کند، در حالی که Test-NetConnection می‌تواند پورت‌ها و سرویس‌های باز را ارزیابی کند.

Active Directory enumeration: با PowerShell، تست نفوذگران می‌توانند اطلاعات ارزشمندی در مورد محیط Active Directory (AD) هدف جمع‌آوری کنند. ابزارهایی مانند Get-ADUser، Get-ADComputer و Get-ADGroupMember می‌توانند برای بازیابی اطلاعات کاربر، رایانه و گروه استفاده شوند. این داده‌ها به درک ساختار شبکه و نقاط ورودی بالقوه کمک می‌کند.

Network shares and permissions: اسکریپت‌های PowerShell را می‌توان برای شناسایی Shareهای شبکه و مجوزهای آن‌ها به کار برد. دستوراتی مانند Get-SmbShare و Get-Acl را می‌توان برای شناسایی Share های باز و حقوق دسترسی مرتبط با آن‌ها به کار برده و بینش‌هایی را در مورد مناطق بالقوه بهره برداری ارائه بدست آورد.

Enumeration of network shares and user information: PowerShell می‌تواند شناسایی Share های شبکه و اطلاعات کاربران را با پرس و جو از سرویس‌های LDAP و SMB تسهیل کند. ابزارهایی مانند NetView و NetUser می‌توانند اطلاعات بیشتری را در مورد ساختار شبکه و آسیب‌پذیری‌های احتمالی آشکار کنند.

Enumeration of running services: PowerShell به تست نفوذگران اجازه می‌دهد تا سرویس‌های در حال اجرا را روی میزبان‌های هدف شناسایی کنند. با استفاده از Get-Service یا پرس و جو از مخزن Windows Management Instrumentation (WMI) با Get-WmiObject، تست نفوذگران می‌توانند اطلاعاتی در مورد سرویس‌ها، تنظیمات آن‌ها و آسیب‌پذیری‌های احتمالی جمع آوری کنند. این نوع سرویس‌ها شامل پایگاه‌های داده می‌شود.

Vulnerability scanning: از PowerShell می‌توان برای شروع اسکن آسیب‌پذیری در سیستم‌های هدف، با استفاده از اسکریپت‌هایی که آسیب‌پذیری‌های شناخته شده یا نسخه‌های نرم افزار قدیمی را بررسی می‌کند، استفاده کرد. این به اولویت بندی بردارهای حمله بالقوه کمک می‌کند.

Port scanning and banner grabbing: PowerShell's Test-NetConnection و InvokeWebRequest را می‌توان برای انجام اسکن پورت و گرفتن بنر استفاده کرد. این اطلاعات به تست نفوذگر کمک می‌کند پورت‌های باز، سرویس‌ها و نسخه‌های نرم‌افزاری بالقوه قدیمی را که ممکن است مستعد سوءاستفاده‌های شناخته‌شده باشند، شناسایی کند.

از این رو، PowerShell یک ابزار ضروری برای Network Enumeration در طول تست نفوذ در محیط‌های ویندوز است. تطبیق‌پذیری، قابلیت‌های اسکریپت نویسی و دسترسی به منابع سیستم، تست نفوذگر را قادر می‌سازد تا اطلاعات حیاتی در مورد شبکه هدف را جمع آوری کند و به وی کمک می‌کند تا نقاط ضعف بالقوه و نقاط ورودی را برای بهره برداری بیشتر شناسایی نماید. با این حال، انجام این فعالیت‌ها به صورت اخلاقی و با مجوز مناسب برای اطمینان از امنیت شبکه ارزیابی شده ضروری است.

TCP port scanning using PowerShell

اسکن پورت، عملی برای بررسی سیستماتیک پورت‌های باز، بسته یا فیلتر شده در یک سیستم هدف است. پورت‌های باز نشان دهنده نقاط ورود بالقوه برای مهاجمان هستند، در حالی که پورت‌های بسته یا فیلتر شده ممکن است نشان دهنده اقدامات امنیتی در هدف باشند. با انجام اسکن پورت، تست نفوذگران می‌توانند اطلاعات مهمی در مورد وضعیت امنیتی شبکه یا سیستم جمع آوری کنند.

Test-NetConnection یک cmdlet همه کاره موجود در Windows PowerShell (نسخه ۴.۰ و جدیدتر) است که در درجه اول برای تشخیص اتصال به شبکه استفاده می‌شود. با این حال، می‌توان آن را برای انجام اسکن پورت در زمینه تست نفوذ تغییر کاربری داد.

Single port scanning with Test-NetConnection

برای انجام یک اسکن پورت ساده روی میزبان هدف با استفاده از Test-NetConnection، این مثال را دنبال کنید:

```
Test-NetConnection -ComputerName 192.168.1.100 -Port 80
ComputerName            : 192.168.1.100
RemoteAddress           : 192.168.1.100
RemotePort              : 80
InterfaceAlias          : Ethernet
SourceAddress           : 192.168.1.101
TcpTestSucceeded        : True
```

در این مثال، Test-NetConnection تأیید می‌کند پورت ۸۰ در سیستم هدف باز است، که نشان دهنده وجود یک وب سرور در آن می‌باشد.

Multiple port scanning with Test-NetConnection

یک تست نفوذگر اغلب نیاز به اسکن چندین پورت در یک سیستم هدف دارد. Test-NetConnection را می‌توان در یک حلقه برای اسکن طیفی از پورت‌ها یا لیستی از پورت‌های خاص استفاده کرد:

```
$RemoteHost = "192.168.1.100"
```

```
$Ports = 80, 443
foreach ($Port in $Ports) {
   Test-NetConnection -ComputerName $RemoteHost -Port $Port }
ComputerName            : 192.168.1.100
RemoteAddress           : 192.168.1.100
RemotePort              : 80
InterfaceAlias          : Ethernet
SourceAddress           : 192.168.1.101
TcpTestSucceeded        : True
ComputerName            : 192.168.1.100
RemoteAddress           : 192.168.1.100
RemotePort              : 443
InterfaceAlias          : Ethernet
SourceAddress           : 192.168.1.101
TcpTestSucceeded        : False
```

در این حالت Test-NetConnection پورت‌های ۸۰ و ۴۴۳ را روی سیستم مورد نظر اسکن می‌کند. نتایج نشان می‌دهد که آیا هر پورت باز است یا خیر.

Enumerating open ports with Test-NetConnection

یکی از اهداف اولیه تست نفوذ، Enumeration پورت‌های باز است. شما می‌توانید از PowerShell برای فیلتر کردن و نمایش پورت‌های باز استفاده کنید:

```
$RemoteHost = "192.168.1.100"
$Ports = 1..65535
$OpenPorts = foreach ($Port in $Ports) {
    $Result = Test-NetConnection -ComputerName $RemoteHost -Port $Port
    if ($Result.TcpTestSucceeded) {
        $Port
    }
}
```

در این مثال، Test-NetConnection تمام پورت‌های ممکن را اسکن می‌کند و پورت‌های باز را شناسایی می‌نماید. Test-NetConnection، در حالی که در درجه اول برای تشخیص اتصال شبکه طراحی شده است، می‌تواند ابزار ارزشمندی برای تست نفوذگران برای انجام اسکن پورت باشد. با استفاده

از قابلیت‌های آن برای بررسی وضعیت پورت‌های خاص در یک سیستم هدف، تست نفوذگران می‌توانند اطلاعات مهمی در مورد آسیب‌پذیری‌های احتمالی و ضعف‌های امنیتی جمع آوری کنند. با این حال، توجه به این نکته مهم است که تست نفوذ باید همیشه با مجوز مناسب و به شیوه‌ای مسئولانه و اخلاقی انجام شود تا از هرگونه مشکل قانونی یا اخلاقی جلوگیری شود. Test-NetConnection، زمانی که در این دستورالعمل‌ها استفاده می‌شود، می‌تواند یک دارایی در جعبه ابزار یک تست نفوذگر باشد.

Single port scanning with .NET

PowerShell می‌تواند یکNET Socket Objects. ایجاد کند تا با یک پورت در یک میزبان هدف ارتباط برقرار کند. به مثال زیر توجه نمایید:

```
$RHost = "192.168.1.100"
$Port = 80
$TcpClient = New-Object System.Net.Sockets.TcpClient
try {
    $TcpClient.Connect($RHost, $Port)
    Write-Host "Port $Port on $RHost is open."
}
catch {
    Write-Host "Port $Port on $RHost is closed or filtered."
}
finally {
    $TcpClient.Close()}
```

در این مثال، PowerShell یک شی کلاینت TCP ایجاد می‌کند و سعی می‌کند به پورت مشخص شده در میزبان هدف متصل شود. سپس باز یا بسته بودن پورت را گزارش می‌دهد.

Multiple port scanning with .NET

یک تست نفوذ معمولی شامل اسکن چندین پورت در یک سیستم هدف است. PowerShell می‌تواند لیستی از پورت‌ها را اسکن نموده و وضعیت آن‌ها را بررسی کند:

```
$RHost = "192.168.1.100"
$Ports = 80, 443, 22
foreach ($Port in $Ports) {
    $TcpClient = New-Object System.Net.Sockets.TcpClient
    try {
        $TcpClient.Connect($RHost, $Port)
        Write-Host "Port $Port on $RHost is open."
    }
    catch {
        Write-Host "Port $Port on $RHost is closed or filtered."
    }
    finally {
        $TcpClient.Close()}}
```

این اسکریپت PowerShell لیستی از پورت‌ها را در میزبان هدف اسکن می‌کند و وضعیت آن‌ها را گزارش می‌کند.

Enumerating all open ports with .NET

در برخی موارد، شناسایی تمام پورت‌های باز در یک سیستم هدف ضروری است. از PowerShell می‌توان برای اسکن سیستماتیک طیفی از پورت‌ها استفاده کرد:

```
$RHost = "192.168.1.100"
$StartPort = 1
$EndPort = 65535
for ($Port = $StartPort; $Port -le $EndPort; $Port++) {
    $TcpClient = New-Object System.Net.Sockets.TcpClient
    try {
        $TcpClient.Connect($RHost, $Port)
        Write-Host "Port $Port on $RHost is open."
    }
    catch {# Port is closed or filtered.}
```

```
finally {
    $TcpClient.Close()}}
```

PowerShell به طور سیستماتیک تمام پورت‌های ممکن را در میزبان هدف اسکن می‌کند و پورت‌های باز را گزارش می‌دهد.

استفاده از اشیاء سوکت .NET در PowerShell به تست نفوذگران، یک رویکرد همه کاره و قابل برنامه ریزی برای انجام اسکن پورت، به عنوان بخشی حیاتی از ارزیابی‌های هک اخلاقی، را ارائه می‌دهد. اسکن پورت نقشی اساسی در شناسایی آسیب‌پذیری‌های بالقوه در یک سیستم هدف بازی می‌کند، و .NET Socket Objects به آزمایش‌کنندگان این امکان را می‌دهد تا این فرآیند را به طور موثر، خودکار و سفارشی کنند.

با این حال انجام تست‌های نفوذ مسئولانه، رعایت دستورالعمل‌های قانونی و اخلاقی و کسب مجوز مناسب ضروری است. هنگامی که به طور مسئولانه استفاده میشود، .NET Socket Objects در PowerShell به عنوان ابزار قدرتمندی برای تست نفوذگران جهت ارزیابی وضعیت امنیتی سیستم‌ها و شبکه‌ها عمل می‌کند و در نهایت امنیت را افزایش می‌دهد.

UDP port scanning using PowerShell

انجام اسکن پورت UDP در PowerShell شامل ارسال بسته‌های UDP به پورت‌های خاصی در میزبان هدف برای تعیین باز یا بسته بودن آن پورت‌ها است. برخلاف TCP، UDP بدون اتصال است، که اسکن پورت UDP را کمی چالش‌برانگیزتر می‌کند، زیرا هیچ Handshake ای برای تایید وضعیت پورت وجود ندارد. در ادامه یک روش ساده با استفاده از PowerShell آورده شده است:

```
$RHost = "192.168.1.100"
$Ports = 53, 67, 123
foreach ($Port in $Ports) {
    $UdpClient = New-Object System.Net.Sockets.UdpClient
    try {
        $UdpClient.Connect($RHost, $Port)
        $UdpClient.Send([byte[]](0), 0, 0)
        Write-Host "UDP Port $Port open - $RHost"
```

```
}
catch {
    Write-Host "UDP Port $Port closed - $Host."
}
finally {
    $UdpClient.Close()}}
```

این اسکریپت از طریق پورت‌های UDP مشخص شده تکرار می‌شود، یک شی کلاینت UDP ایجاد می‌کند و یک بسته UDP خالی به هر پورت ارسال می‌کند. اگر یک استثنا در طول فرآیند ثبت شود، به این معنی است که پورت بسته یا فیلتر شده است. اگر هیچ استثنایی مطرح نشود، پورت باز در نظر گرفته می‌شود.

توجه داشته باشید که اسکن UDP می‌تواند کمتر از اسکن TCP قابل اعتماد باشد، زیرا UDP مکانیسم‌های تایید و پاسخ مشابه TCP را ارائه نمی‌دهد. برخی از پورت‌های UDP باز ممکن است به بسته‌های خالی UDP پاسخ ندهند، که می‌تواند منجر به False Positive شود. علاوه بر این، فایروال‌ها و فیلتر شبکه می‌توانند بر دقت اسکن پورت UDP تأثیر بگذارند. همیشه از داشتن مجوز مناسب اطمینان حاصل کنید و از ابزارهای تخصصی‌تری برای کارهای جامع تست نفوذ استفاده کنید.

Using PowerShell tools for port scanning

چندین ابزار منبع باز PowerShell وجود دارد که از اسکن پورت TCP/UDP پشتیبانی می‌کند. لینک زیر نمونه‌ای از ابزار اسکن PowerShell است:

https://github.com/BornToBeRoot/ PowerShell_IPv4PortScanner

IPv4PortScan یک ابزار اسکن TCP ناهمزمان است که به کاربر اجازه می‌دهد تا محدوده پورت مورد اسکن را تعریف کند. خط فرمان ابزار به صورت زیر است:

```
.\IPv4PortScan.ps1 [-ComputerName] <String> [[-StartPort]
<Int32>] [[-EndPort] <Int32>] [[-Threads] <Int32>] [[-Force]]
[<CommonParameters>]
```

در ادامه از این ابزار برای اسکن ۵۰۰ پورت اول در سیستم مورد نظر استفاده خواهیم کرد:

```
PS> .\IPv4PortScan.ps1 -ComputerName www.snowcapcyber.com -EndPort 500
Port Protocol ServiceName  ServiceDescription   Status
---- -------- -----------  ------------------   ------
  53  tcp     domain       Domain Name Server   open
  80 tcp      http         World Wide Web HTTP  open
```

ابزار PS2 یک ابزار اسکنر پورت TCP است که برخی از عملکردهای Nmap را تقلید می‌کند. این ابزار را می‌توانید در https://github.com/nccgroup/PS2 دانلود کنید.

این ابزار از اسکن TCP و UDP و همچنین نقشه برداری از یک شبکه با استفاده از یک تابع traceroute پشتیبانی می‌کند. در ادامه، از این ابزار برای اسکن ۱۰۰۰ پورت TCP رایج استفاده خواهیم کرد:

```
PS C:\>ps2.ps1 -sT -i 192.168.1.1
```

در مثال زیر، از ابزاری برای اسکن تمام پورت‌های TCP استفاده می‌کنیم:

```
PS C:\>ps2.ps1 -sT -p (1..65535) -i 192.168.1.1
```

در نهایت، ما از این ابزار برای اسکن ۱۰۰۰ پورت رایج UDP استفاده خواهیم کرد:

```
PS C:\>ps2.ps1 -sU -i 192.168.1.1
```

به طور خلاصه، PowerShell توانایی ایجاد ابزارهای ساده و قدرتمند برای اسکن پورت TCP/UDP را در اختیار ما قرار می‌دهد. ما نشان داده‌ایم که چگونه می‌توان از cmdletها استفاده نموده و یک اسکریپت ایجاد کرد. همچنین چگونه PowerShell می‌تواند از توابع .NET. استفاده کند. این ابزارها ما را قادر می‌سازند تا در یک تست نفوذ در زمین زندگی کنیم (منظور استفاده از تکنیک‌های Living off the Land و عدم استفاده از ابزارهای خارجی است) و تعداد ابزارهایی را که باید در یک تست امنیتی پشتیبانی کنیم، به حداقل برسانیم.

Summary

این فصل یک کاوش جامع در مورد استفاده از PowerShell برای شناسایی و ارزیابی امنیتی شبکه قوی بود. این فصل چگونگی استفاده از PowerShell را برای Enumeration دستگاه‌ها و سرویس‌ها به طور کارآمد نشان می‌دهد.

این فصل بیشتر به حوزه اسکن پورت پرداخته و بر نقش حیاتی آن در ارزیابی آسیب‌پذیری تأکید می‌کند. این فصل با نشان دادن قدرت PowerShell برای اسکن پورت، استراتژی‌هایی را برای شناسایی

پورت‌های باز، سرویس‌ها و آسیب‌پذیری‌های بالقوه معرفی کرد. نمونه‌های عملی و سناریوهای دنیای واقعی شما را با تجربه عملی تجهیز می‌کنند و به شما قدرت می‌دهند تا امنیت شبکه را تقویت کنید. چه برای مقاصد دفاعی و چه برای هک اخلاقی، این فصل به عنوان یک منبع ارزشمند عمل می‌کند و درک دقیقی از Enumeration شبکه و اسکن پورت با استفاده از قابلیت‌های همه کاره PowerShell ارائه می‌کند.

در فصل بعد، نحوه استفاده از PowerShell در هنگام انجام تست نفوذ روی REST و SOAP APIها را یاد خواهیم گرفت.