SSH-MITM Plugins

Simon Böhm

Feb 11, 2022

CONTENTS

SSH-MITM Plugins Documentation

CHAPTER

ONE

START

With version 0.4.0 the ssh-mitm projects locks the features shipping with the core functionality of the program. It is now preferred that any additions to the feature-set is made through the modular capabilities that the ssh-mitm project is built upon. Using entrypoints in combination with modules anyone can make their own ssh-mitm plugins.

Here you will find detailed feature-oriented documentation of the creators additions to the ssh-mitm project.

CHAPTER

TWO

SSH INTERFACES

2.1 injectorshell

Included in the original ssh-mitm suit this is a detailed documentation by its creator.

The injectorshell ssh interface allows the operator of the ssh-mitm server to serve out shell access over the network that correspond to a hijacked ssh session. Within these injected shells one is able to execute commands on the remote host using the ssh session created by the original client. Contrary to the mirrorshell there can be multiple injected shells per ssh session. All these shells - including the client itself - share their environment but are served answers individually.

Using the `--ssh-injector-enable-mirror` option injected shells can print the input of the user to their screen. This differs from the mirrorshell which always displays output on the injected as well as the clients shell. The injectorshell tries its best to not leak any unwanted output to the users session so that they can operate normally.

By default injector shell access is limited to the local maschine `localhost` but can be opened up to any network using the `--ssh-injector-net NET/IF` parameter. Due to the fact that access to the injector shells is not authenticated doing this should be thoroughly thought through.

For ease of use a private key can be used for a more consistent integrity check. It can be set with the `--ssh-injector-key ID` parameter. If this is not done a new one will be generated each time the server is spun up.

Note: It should also be noted that shell environment can be affected by any injector shell and is not accounted for when considering stealth. This means environment variables or the working directory for example can be changed by any injector shell and will alert the original shells owner of faulty operation.

Important: It is also important to mention that when multiple injector shells are inserting commands into the same hijacked ssh session at the same time discrepancies are not accounted for. Keystrokes are collectively merged at the server and the responses are served accordingly. This is also true for the clients interactive ssh session. A advanced edition of the injectorshell - the *stealthshell* - fixes both these problems.

2.2 stealthshell

As an upgrade to the *injectorshell* (implementation in ssh-mitm done by me) the stealthshell provides a way to workaround the problem of interfering with the clients interactive session. It only executes injected commands when the shell of the user wont be affected. As long as the interactive shell of the client is not typing or executing a command input from the injector shells is halted and put in a waiting queue.

Using the `--ssh-injector-super-stealth` option the injector shells will only send whole commands instead of every keystroke. This further eliminates unwanted behavior. Unfinished commands from the injector shells are not seen by the server and the user of the interactive shell will never be surprised by input they never typed. This, however, will limit the terminal functionality of the injector shell. Because the server only responds to the whole command, terminal features like command auto-completion when pressing tab or command history with the up and down keys will not work correctly.

Note: Environment considerations of the *injectorshell* are still uphold by the stealthshell. Discrepancy problems described by the *injectorshell* are solved by this newer edition (client cannot be interrupted by injected keystrokes BUT unfinished injected strokes will be seen by the server). Only with the `--ssh-injector-super-stealth` option will the discrepancy between the user and all injector shells not occur. It is recommended that the `--ssh-injector-super-stealth` option is used in combination with the `--ssh-injector-enable-mirror` option to see more clearly when commands can be executed.

For a more detailed look at the plugins operation refer to the *injectorshell* documentation.

2.3 scriptedshell

When working through a security audit gathering information is one of the most important steps.

The scriptedshell ssh interface is first and foremost an information gathering tool but due to its functionality it can also be used for different use cases. This plugin will execute a shell script when a new ssh session is opened by a client. The output of the script will be stored locally on the ssh-mitm machine under their respective session name.

Note: Stored script output is taken from the server as-is with some ANSI control characters removed.

The `--ssh-script SCRIPT` parameter declares the location of the script.

The `--ssh-out-dir DIR` parameter indicates where the output of each session script execution should be stored.