SPLOIT Exploits PoC

In this report, we present the five proof-of-concept exploits of Sploit program.

Stack Overflow (2)

1. `sockfd` and `cmdBuffer` are global variables on the stack, using `fgets()` to receive `cmdBuffer` can overwrite the value stored in `sockfd`, which can be used to exploit.

```
int sockfd;
char cmdBuffer[1024];
```

2. Configuration File: Based on the specification, client side has the freedom to put `any` files to server. This implies client can override the configuration file. In the server parsing code, we used a wrong `size` in the `fgets()` function. Thus malicious client can crash server by uploading a pre-defined configuration file.

Format String (1)

When displaying the `server_ip` specified by the user, we use the method:

```
printf(server_ip);
```

which is a `string` supplied by the user, without any checks/validation.

This is a format string vulnerability since the malicious user can pass a crafted string to reveal the data stored on the stack (e.g., passing `%x` to print the data stored on the stack or passing `%s%s%s%s%s%s%s%s%s%s%s%s%s` to crash the program.

Command Injection (1)

We design the client side to be a dummy agent which *only* honestly sends whatever it receives from the user, whoever is benign or malicious. And we also intentionally implement an non-exhaustive check on the server side.

With these two aspects combined, user can type `ctrl+c` to end her own client, however, this will also be honestly send to the server side and cause the server to receive same signal, thus shutting down as well.

This directly paralyzes the server as the command `ctrl+c` is directly executed, sending a `SIGINT` to the server.

Arbitrary Vulnerability (1)

DoS attack on the server

By design, a server should spawn a `thread` to handle the `put` requests from authenticated clients.

We implement such mechanism using `pthread` while totally ignoring an upper limit of the number of spawned threads.

This can be used for DoS attack, since a malicious party can control the bots to simultaneously launch such requests.

As a result, the server will spawn these many threads and potentially hang itself due to high loads.

Bonus

We have implemented the bonus part by making `put` and `get` to be executed in parallel.