New York University
Courant Institute of Mathematical Sciences
CSCI-GA 3033: Practical Computer Security

Command Injection Lab

Due: Feb 17 2021, 11:59 PM (EST)

1 Lab Description

Command injection is an attack in which the goal is to execute arbitrary commands on the host operating system via a vulnerable application. Command injection attacks are possible when an application passes unsafe user-supplied data (forms, cookies, HTTP headers, etc.) to a system shell.

The objective of this lab is to 1. get familiar with the Git-based CTF lab structure; 2. construct a user input string to attack the provided vulnerable echo server.

We list some common questions on Piazza.

Question 1: Please provide the names and NetIDs of your collaborator (up to 3). If you finished the lab alone, write None.

2 Lab Setup

This lab requires quite a bit of effort to set up. Please start as early as possible so that you have time to set up, debug, and ask for help if necessary.

Unzip the assignment zip file, and you should see two folders `answer_template` and `keys` and two files `requirements.txt` and `config.json`.

For Windows users, we recommend using `WSL` (Windows Subsystem for Linux) to run all the commands in this lab. Please refer to the documentation for how to install it.

2.1 Python and Python Packages

We assume you already have `Python 3` installed on your computer. If not, please visit Python.org and follow the instructions to download and install it.

Open the assignment folder and use the following command to install the required packages.

```
pip3 install -r requirements.txt
```

2.2 Git and GitHub

We assume you already have `Git` installed on your computer. If not, please visit Git and follow the instructions to download and install it.

We assume you already have a `GitHub` account and have set up the `SSH` connection on your computer. If not, please visit GitHub to create an account and Connecting to GitHub with SSH to set up the `SSH` connection.

To verify that you have set up `GitHub` connection correctly, enter the following command in your terminal:

```
ssh -T git@github.com
```

And you should see something like this:

```
> Hi [username]! You've successfully authenticated, but GitHub does not
> provide shell access.
```

2.2.1 Personal Access Token

This lab also requires a `GitHub` personal access token in order to submit your solution. Please visit Personal Access Tokens to generate one. Make sure you select `repo` as the scope so that the token can work properly. You will only see the token once, so make sure you save it somewhere safe. We will reuse the token for the next couple of labs.

DO NOT include your personal access token in the submission.

2.3 Docker

This lab runs on `Docker`. Please visit Get Started with Docker to download and install `Docker`.

To verify `Docker` works correctly on your machine, run the following command. You should see the welcome message.

```
docker run hello-world
```

2.4 GPG and Key Pair

Part of this lab will be submitted to a git repo as an issue so that the auto grader can run your code on our server. A `GitHub` issue is public to everyone. To protect your submission, this lab also requires `GPG` (GNU Privacy Guard) and a key pair to encrypt your submission so that only the professor and TAs can access your code.

For windows user, the `gpg` command should already be installed in `Git Bash` or `WSL`.

For macOS user, please use GPGTools.

For Linux user, the `gpg` command should already be installed in your system.

2.4.1 Generate Key Pair

Now, follow the instructions from `GitHub` to Generate a New GPG Key. You don't need to add the `GPG` key to your `GitHub`.

DO NOT include your private key in the submission.

2.4.2 Import Our Public Keys

Please use the following command to import our public keys:

```
gpg --import keys/*
```

2.5 Lab File

Now we have everything installed and set up, let's get the lab server.

Open the assignment folder and use the following commands to clone the `scripts` folder and the lab server folder from GitHub:

```
git clone git@github.com:nyupcs/GitCTF-scripts.git scripts
git clone git@github.com:nyupcs/pcs-sp21-lab1-server.git
```

Go to the folder `pcs-sp21-lab1-server` and use the following command to build the server container:

```
docker build . --file Dockerfile --tag pcs-sp21-lab1-server:latest
```

2.5.1 config.json

Open `config.json`. Replace `[GitHub Username]` with your `GitHub` username. Replace `[NetID]` with your NYU NetID. Replace `[Public Key ID]` with the key ID you generated in section 2.4.

3 Lab Tasks

3.1 Play With the Echo Server

Use the following command to start the server container:

```
docker run -d -p 4000:4000 --name echo-server pcs-sp21-lab1-server
```

Now use the `netcat` command (`nc`) to start a TCP connection to the server:

```
nc 127.0.0.1 4000
```

Since this is an echo server, type something in the terminal and hit enter to send the string, and you should see the same line come back.

When you are done playing with it, press `Ctrl + C` to quit `netcat` and use the following command to stop and remove the docker container.

```
docker rm -f echo-server
```

3.2 Analyze the Vulnerable Program

```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<string.h>
4  #include<unistd.h>
5
6  int main()
7  {
8      char buf[256];
9      char *cmd_prefix = "echo␣";
10     while (1)
11     {
12       strcpy(buf, cmd_prefix);
13       gets(buf + strlen(cmd_prefix));
14       system(buf);
15       fflush(stdout);
16     }
17     return 0;
18 }
```

Question 2: There is a vulnerability at line 14. What is it?

Question 3: There is also a buffer overflow issue in the code. Where is it?

3.3 Perform the Attack Locally

Follow the instructions in section 3.1 to start and connect to the docker container.

Question 4: We placed a secret message in the file `/var/ctf/flag`. How would you construct a string that utilizes the vulnerability in question 1 and get the secret message?

Now enter the string in the terminal and hit enter. Can you get the flag?

When you finished, remember to follow the instructions in section 3.1 to stop and remove the docker container.

3.4 Prepare the File to Submit

Go to the `answer_template` folder. We have provided some files to start.

The `Dockerfile` contains instructions to build a client to perform the attack. Unless you choose to ignore `exploit.py` and write your solution, you don't have to worry about this file.

The `exploit.py` contains some basic code to start. It already set up the TCP connection for you. All you need to do is send the string you constructed in section 3.3 and print the flag to the `stdout`.

When you are ready to test your code, follow the instructions in section 3.1 to start the docker container, and use the following command to run your code:

```
python3 exploit.py 127.0.0.1 4000
```

If everything is correct, you should be able to see the flag printed out.

Question 5: Please copy and paste your code of `exploit.py` to the report.

When you finished, remember to follow the instructions in section 3.1 to stop and remove the docker container.

Attention: Before submitting your code, make sure you have stopped and removed the docker container. Otherwise, the submission script will not be able to verify your code because of port conflict.

4 Submission

4.1 Submit Your Code

Rename the folder `answer_template` to `answer` and the file `exploit.py` to `exploit`, and use the following command to submit your code:

```
python3 scripts/gitctf.py submit --exploit answer
                                 --service-dir pcs-sp21-lab1-server
                                 --target lab1 --branch main
```

The script will verify your solution locally. If the code passes the test, it will be uploaded to the lab repo as an issue.

During the upload stage, the script will ask you for your GitHub password, enter the personal access token that you generated in section 2.2.1.

Question 6: What is the issue URL?

4.2 Submit Your Lab Report

You need to submit a lab report to answer all the questions above with screenshots to describe what you have done and what you have observed; you also need to provide brief explanations to the observations that are interesting or surprising.

Late submissions are accepted with 50% grading penalty within 24 hours of the due. Submissions that are late for more than 24 hours will NOT be accepted.

Please submit your solution in PDF or image on `https://www.gradescope.com/courses/225188`, using Entry Code: **86EDWX**. And kindly choose the right page for your answer to every question.

Collaboration Policy

You can optionally form study groups consisting of up to 3 people per group (including yourself).

Everybody should write individually by themselves and submit the lab reports separately. DO NOT copy each others lab reports, or show your lab reports to anyone other than the course staff, or read other students lab reports.

5 Acknowledgements

This lab is adopted from the Git-based CTF project. The original project is here and the paper is here.

This PDF was created on 2021-02-09 22:09.