CS Games 2017

Reverse Engineering

Participants	2
Workstations	1
Total Value	6%
Duration	3 hours

Reverse Engineering

Context

In 2013, computer genius Edouard Kropotkin left Mother Russia and started his own company in Silicon Valley. He hired accountants, clerks and baristas, but remained the only developer in his start-up.

Weeks went by, and while everyone was working hard on their ping-pong game and Mario Kart skills, Edouard was the only one getting any work done. One day, he had enough and decided to retreat in self-managed anarchist community, leaving all his startup dreams in the hands of his former colleagues.

The only problem is that, without him, nobody can make the business work! Edouard was using custom software he created himself and disappeared with the source code.

Tasks

Task 1 - File transfer

Introduction

Implementation/Protocol

Before he disappeared, Edouard was working on a file transfer system. We believe he was done with the server implementation, but we need a client in order to sell this innovative solution.

Please help us! We don't care which language you use.

Technical description

You have a local copy of the server binary. You need to implement a client that can interact with the server and:

- Authenticate with the server
- Get a list of files you can download
- Download a file from the server
- Upload a file from your drive to the server
- Each task has to end cleanly (no unexpected packet, or unexpected closed connection)

What you have:

You have the following files and directories:

- server_folder/ contains files and directories hosted on server
    - srv/
        - Is the root of the file server. It has to be in the same folder as “server”. It contains files that can be served to clients. It’s also where uploaded files are stored.
    - server
        - This is the server binary you need to reverse engineer. Understand how it works to create a client.
- client_folder/ contains files and directories for the client
    - local/
        - Contains your local files. This is where you must store downloaded files.
    - client
        - This is **empty**. It’s up to **you** to replace it by a script or an executable. What it receives from server must be written to stdout.
    - run.sh
        - A tool to evaluate your client. You can run it with ./run.sh <username> <password>. It will execute 3 commands. **Those three commands are those you must implement in your client**:
            - List the files hosted on the server
            - Download a file from the server (srv/) to your local directory (local/)
            - Upload a local file (in local/) to the server (srv/)

The commands to implement

- List the remote files
    - $./client <ip> <username> <password> list_files
- Download a remote file
    - $./client <ip> <username> <password> download <filename>
- Upload a local file
    - $./client <ip> <username> <password> upload <filename>

Practical

Phase	Tâche	Pts
List remote files	You are authentified by the server	/2
	You get the list of files on the server (contained in srv/)	/2
	You cleanly end the communication	/1
Download a remote file	The downloaded file in local/ is identical to the original one in srv/	/3
	You cleanly end the communication	/1
Upload a local file	The uploaded file in srv/ is identical to the original one in local/	/3
	You cleanly end the communication	/1

- All client-server communication must be done via network.
- You can’t modify the server. You must only implement the client.

Theoretical

A) Give the username;password accepted for authentication.

Username	Password	Pts
		/2

B) What is the transport protocol used by the server?

Answer: _____ /1

C) On which port is it listening?

Answer: _____ /1

Bonus

A) A vulnerability exists in the download function. What is it?

Answer: _____ /1

B) How would you fix it?

Answer: _____ /1

C) Provide a proof of concept to steal the /etc/passwd file.

/1

Task 2 - Clients list

Introduction

Crypto/Scripting

Edouard had a list of possible clients interested in our products. This list is invaluable, which is why he encrypted it. The problem is… There's no decryption script. Fortunately, we were able to recover the encryption script, which is a little bit hard to read. We think the key was derived from a picture, but we couldn't find it.

Technical description

You have an encrypted list of clients and the encryption script. It's up to you to implement a script to decrypt it!

What you have:

- encrypt_list.py
    - The encryption script
- o.enc
    - The list to decrypt

Practical

Task	Pts
Implement a decryption script	/10

Task 3 - Password manager

Introduction

Patching

We use Instagram a lot, but always forget our password. When we brought this up with Edouard, he created a password manager to help us out. When we give it the master-password, the software displays our Instagram password. To be sure that nobody accesses the software without authorization, the password manager is full of security measures.

The password manager can only be ran from Edouard's computer and takes multiple hours to execute.

We need to publish a new Instagram post as fast as possible - It's urgent!

Technical description

The Instagram password of the company is kept in a password manager. Inspired by bank vaults systems, you only get the password multiple hours after executing the manager. Other security mechanisms are in place to ensure this critical information does not fall into the wrong hands.

What you have:

- pwManager
    - The password manager. You can launch it from the command prompt using $./pwManager <master-password>

Practical

Task	Pts
Create a patched version of the password manager that shows the password instantly. (You get more points the fewer bytes you change)	/6

Formula used: *number_bytes / minimum * 6*

Theoretical

A) What is the Instagram account's password?

Answer: _____ /4