Exercise 3

Capture-the-Flag Challenge Design: "Crack the Code"

1. **Type of Challenge**: Crackme (Binary Patching)
2. **Difficulty**: easy/beginner
3. **Technical Details**:
    - **Platform**: Linux
    - **Binary**: A simple command-line application written in C
    - **Objective**: Bypass the authentication mechanism to reveal the flag.
    - **Protection**: The binary will check for a hardcoded password. The player's goal is to patch the binary to bypass the check. Basic encryption was added for not storing password as a string.

4. **Write-up**:
    - Use a disassembler/debugger like `Ghidra` or `GDB` to reverse engineer the binary. Identify the function or block responsible for password checking.
    - Modify the binary, patching the condition checks, effectively allowing any input to reveal the flag.
    - Run the patched binary and capture the flag.

5. **Hints for Players**:
    - "Everything you need is within the main function."
    - "Can you identify any string comparing instructions in main?"
    - "NOP the JNE so that it doesn’t lead to the “access denied” part of the code."
    - "Remember the JNE instruction occupies 2 bytes, so you need 2 NOPs, otherwise you will get a segmentation fault.”

6. **Resources**:
    - **Docker Container**: Provided a pre-built Docker container with the binary and necessary tools.
    - **Network Access**: Not required for this challenge.

Setup instructions

1. Loading the Docker Image from the .tar File:

Before you proceed, ensure Docker is installed and running on your system.

To load the Docker image from the `.tar` file, use the `docker load` command:

“docker load -i path_to_the_tar_file.tar”

Replace `path_to_the_tar_file.tar` with the actual path to the `.tar` file.

2. Listing Docker Images:

After loading the image, you can verify that the image is loaded correctly using:

“docker images”

You should see `crackme_assignment_image` in the list of images.

3. Running a Container from the Image:

To start a container based on the image:

```

**docker run -it crackme_assignment_image /bin/bash**

```

This will drop you into a bash shell inside the container.

4. Solving the CrackMe Challenge:

Inside the container, navigate to the directory where the `crackme` binary is located (if not already there) and run the binary to attempt the challenge.

You can also explore the file system within the container, view files, run any installed software, etc., just as you would in any Linux environment. The `crackme` binary is the challenge you need to solve.

5. Exiting the Container:

Once you're done, you can exit the container by typing `exit` or pressing `CTRL+D`.

6. Removing the Image:

If you want to remove the image after working with it

```

**docker rmi crackme_assignment_image**

```

7. Additional Notes:

Remember that changes made inside the container are not persistent. If you exit the container and start a new one, any changes you made in the previous container will be gone.

SOLUTION

I was using radare2 for this, but the thought process is the same no matter the tools that were used.

Step 1:

Make a backup so that you have the original crackme binary

Step 2:

Run radare2 with write mode on the copied crackme file using this command “r2 -w crackmeNEWPATCH” where crackmeNEWPATCH is the copy of original crackme.

Step 3:

Analyse the binar using the command “aaa”

Step 4:

Go to the ‘main’ function using this command “s main”, you should see the address to the left side of your input field change.

Step 5:

Use this command to see the main function “pdf”

Step 6:

Identify the address for the JNE or JNZ instruction, which is under the test instruction, which is also under the int strcmp(const char *s1, const char *s2).

Step 7:

Go to that address you identified in step 6 using this command

“s 0xThe_Address_I_Identified_in_Step 6”

Step 8:

NOP (Do nothing instruction) the instruction in step 7 by using this command.

“ "wa nop;nop" ” , including the inner apostrophes, and not the outer.

Step 9:

Check if you have two nop instructions at that address by using the command “pdr”, if so then save the file by using the command “ :w ”.