GETTING STARTED
Buffer Overflow

Exploit Research

Based on the information found during the reconnaissance phase or in the post-exploitation phase, a public exploit might not exist for compromising the machine. In such cases, the attacker will have to analyze the files or output and write their own exploit code. In this section, the students will be taught how to debug a process and how to write code to exploit a buffer overflow vulnerability in a program.

What will you learn?

- Debugging a process with GDB
- Understanding the structure of the stack
- Understanding the registers $esp, $ebp, $eip and NOP sled
- Writing a custom script to exploit the vulnerability.
- Analyzing source code and identifying vulnerability

References:

1. Buffer Overflow (https://en.wikipedia.org/wiki/Buffer_overflow)
2. Buffer Overflow (https://owasp.org/www-community/vulnerabilities/Buffer_Overflow)
3. Buffer Overflow Attack (https://owasp.org/www-community/attacks/Buffer_overflow_attack)
4. Reverse Engineering Linux 32-bit Applications (https://www.pentesteracademy.com/course?id=40)

Labs:

Buffer Overflow:

- Command Line Argument
  - Objective: No bound check is performed on the input passed in the command. Exploit the vulnerability to escalate to root.
- Unsafe Input Functions
  - Objective: No bound check is performed on the input passed to the gets function. Exploit the vulnerability to escalate to root.
- Environment Variables
  - Objective: No bound check is performed on the input passed as an environment variable. Exploit the vulnerability to escalate to root.
- File Read
  - Objective: No bound check is performed on the input passed read from the file. Exploit the vulnerability to escalate to root.
- Pipes
  - Objective: No bound check is performed on the input read from the pipe. Exploit the vulnerability to escalate to root.
- Message Queue
  - Objective: No bound check is performed on the input read from the message queue. Exploit the vulnerability to escalate to root.
- Remote Service
  - Objective: No bound check is performed on the input received by the remote service. Exploit the vulnerability to escalate to root.