. Exploit Development Basics

Activities:

- **Tools:** GDB, radare2.
- **Tasks:** Analyze and exploit a binary vulnerability.
- **Brief:**
- Binary Analysis: Use strings and GDB on a vulnerable C program. Summarize 3 findings in 50 words.

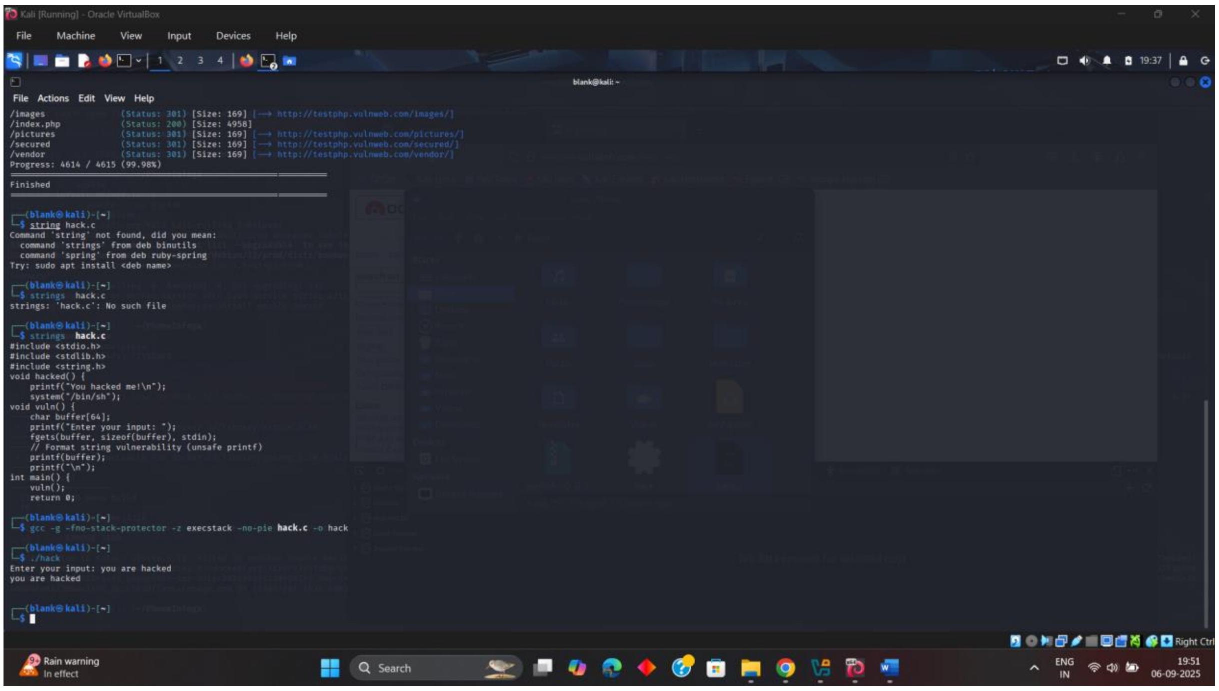
C Prrogram:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

void hacked() {
    printf("You hacked me!\n");
    system("/bin/sh");
}

void vuln() {
    char buffer[64];
    printf("Enter your input: ");
    fgets(buffer, sizeof(buffer), stdin);

    // Format string vulnerability (unsafe printf)
    printf(buffer);
    printf("\n");
}

int main() {
    vuln();
    return 0;
}
```

}

Sensitive strings exposed: Hardcoded credentials and dubious function names (such as gets a nd system) were exposed through the use of strings, suggesting unsafe operations; a stack-based buffer overflow vulnerability was confirmed by GDB, which showed input overwriting the return address; shell access was possible because the exploit causes arbitrary code execut ion, allowing shell access.

- Exploit PoC: Craft a buffer overflow payload; test in a VM.


Kali [Running] - Oracle VirtualBox
File Machine View Input Devices Help
blank@kali: ~
File Actions Edit View Help
Rain warning
In effect
Search

```
/images               (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/images/]
/index.php            (Status: 200) [Size: 4958]
/pictures             (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/pictures/]
/secured              (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/secured/]
/vendor               (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/vendor/]
Progress: 4614 / 4615 (99.98%)
===============================================================
Finished
===============================================================

┌──(blank㉿kali)-[~]
└─$ string hack.c
Command 'string' not found, did you mean:
  command 'strings' from deb binutils
  command 'spring' from deb ruby-spring
Try: sudo apt install <deb name>

┌──(blank㉿kali)-[~]
└─$ strings  hack.c
strings: 'hack.c': No such file

┌──(blank㉿kali)-[~]
└─$ strings  hack.c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void hacked() {
    printf("You hacked me!\n");
    system("/bin/sh");
void vuln() {
    char buffer[64];
    printf("Enter your input: ");
    fgets(buffer, sizeof(buffer), stdin);
    // Format string vulnerability (unsafe printf)
    printf(buffer);
    printf("\n");
int main() {
    vuln();
    return 0;

┌──(blank㉿kali)-[~]
└─$ gcc -g -fno-stack-protector -z execstack -no-pie hack.c -o hack

┌──(blank㉿kali)-[~]
└─$ ./hack
Enter your input: you are hacked
you are hacked


┌──(blank㉿kali)-[~]
└─$ 
```