International Institute of Information Technology Hyderabad

System and Network Security (CS5470)

Lab Assignment 4: A Practical Ethical hacking: Buffer-overflow attack

Hard Deadline: **February 27, 2021 (23:55 P.M.)**

Total Marks: 100

Note1:- *Number beside the question denotes the number allocated to that question. Full marks is 100. You may either get 0 or full marks for each part of the question.No partial marks are there.Good luck*

Note2:- *It is strongly recommended that no student is allowed to copy programs from others. Hence, if there is any duplicate in the assignment, simply both the parties will be given zero marks without any compromise. Rest of assignments will not be evaluated further and assignment marks will not be considered towards final grading in the course. No assignment will be taken after deadline. Please upload in code along with a README file in the course moodle portal through a ZIP file (Groupnumber-Lab4.zip).*

Note3:- *Take your group number and take mod 3 of it.Use Ubuntu -32 bit - ISO file 14.04/16.04/18.04 accordingly . E.g. group no%3 == 0 will download 14.04 and group no % 3 == 1 will download 16.04 and so on for actual assignment. Use VM / Live boot for safety. For each question make one badfile with the name badfile_questionNumber.*

Stack overflow

```c
/* stack.c */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int bof(char *str)
{
    char buffer[12];

    /* The following statement has a buffer overflow problem */
    strcpy(buffer, str);

    return 1;
}

```

```c
int main(int argc, char **argv)
{
    char str[517];
    FILE *badfile;

    badfile = fopen("badfile", "r");
    fread(str, sizeof(char), 517, badfile);
    bof(str);

    printf("Returned Properly\n");
    return 1;
}
```

- What is Stack guard? What is ASLR protection? 5
- Perform a stack overflow attack on the stack.c and launch shell as root under when Stack is executable stack and ASLR is turned off. 10
- Perform a stack overflow attack on the stack.c and launch shell as root and perform seteuid() under when Stack is executable stack and ASLR is turned off. 10
- Perform a stack overflow attack on the stack.c and kill all processes when Stack is executable stack and ASLR is turned off. It is a kind of Denial of Service attack. 10
- Perform a stack overflow attack on the stack.c and reboot the system when Stack is executable stack and ASLR is turned off. 10
- Now turn on ASLR and perform all the tasks from 2 to 5. 15
- Turn on a non-executable stack . Perform any ret2libc attack. 10

Heap overflow

```c
/* heap1.c */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

struct data {
  char name[64];
};

struct fp {
  int (*fp)();
};

void excuteShell()
{
        char *name[2];

    name[0] = "/bin/sh";
    name[1] = NULL;
    execve(name[0], name, NULL);

```

```
}

void Failed()
{
  printf("You failed to exploit the heap\n");
}

int main(int argc, char **argv)
{
  struct data *d;
  struct fp *f;

  d = malloc(sizeof(struct data));
  f = malloc(sizeof(struct fp));
  f->fp = Failed;

  strcpy(d->name, argv[1]);

  f->fp();

}
```

- Exploit the heap and try to execute executeShell function to launch a shell. 10

```
/* heap2.c */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

void excuteShell()
{
        char *name[2];

    name[0] = "/bin/sh";
    name[1] = NULL;
    execve(name[0], name, NULL);
      }

int main(int argc, char **argv)
{
  char *a, *b, *c;
  a = malloc(32);
  b = malloc(32);
  c = malloc(32);
  strcpy(a, argv[1]);
  strcpy(b, argv[2]);
  strcpy(c, argv[3]);
  free(c);
  free(b);
  free(a);
  printf("You failed\n");
}
```

- Exploit the heap and try to execute executeShell function to launch a shell. 20

All the best!