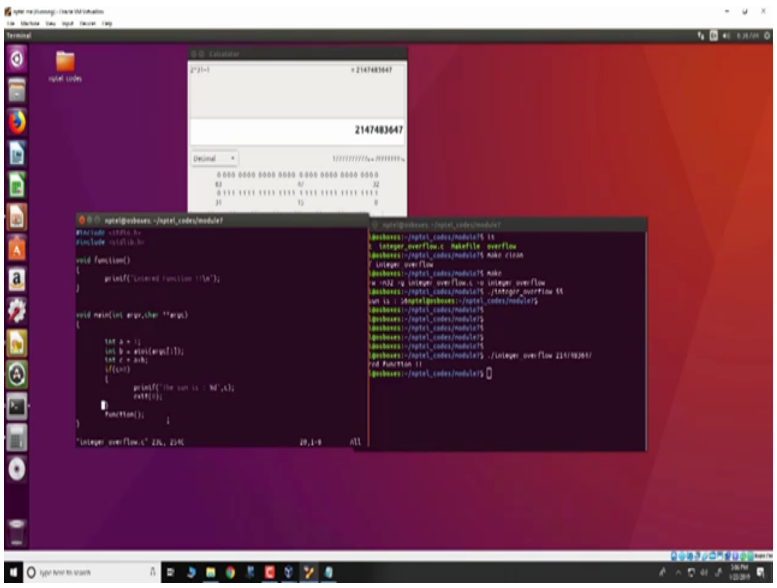
Information Security 5 Secure Systems Engineering
Professor Chester Rebeiro
Indian Institute of Technology, Madras
Lecture 25
Demo of Integer Vulnerabilites

Hello and welcome to this demonstration on integer overflow vulnerabilities. So what we will be seeing today is a very simple program which many of us actually code in this particular way but we will actually demonstrate that there is a vulnerability in this program and we will show how this vulnerability can be used to actually subvert execution or make the program behave in a very abnormal way.

(Refer Slide Time: 00:41)



```
2147483647
```

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

void function()
{
        printf("Entered function !!\n");
}


void main(int argv,char **args)
{

        int a = 1;
        int b = atoi(args[1]);
        int c = a+b;
        if(c>0)
        {
                printf("The sum is : %d",c);
                exit(0);
        }
        function();
}

"integer_overflow.c" 23L, 234C                    20,1-8        All
```

```
@osboxes:~/nptel_codes/module7$ ls
  integer_overflow.c  Makefile  overflow
@osboxes:~/nptel_codes/module7$ make clean
 integer_overflow
@osboxes:~/nptel_codes/module7$ make
 -m32 -g integer_overflow.c -o integer_overflow
@osboxes:~/nptel_codes/module7$ ./integer_overflow 55
sum is : 56nptel@osboxes:~/nptel_codes/module7$
@osboxes:~/nptel_codes/module7$
@osboxes:~/nptel_codes/module7$
@osboxes:~/nptel_codes/module7$
@osboxes:~/nptel_codes/module7$
@osboxes:~/nptel_codes/module7$
@osboxes:~/nptel_codes/module7$ ./integer_overflow 2147483647
red function !!
@osboxes:~/nptel_codes/module7$
```

The code that we are looking at is present in the VM that is shipped along with this particular course that is the secure system engineering NPTEL course and the program as such is present in this directory NPTEL course module 7. So we look into the file integer overflow dot c. Now the program we look at is very simple there are two functions a main and the function, in the main function we have three local variables a, b and c, a has a value of 1, b takes it is input from the command line arguments and c essentially does a plus b.

Now if c has a value greater than 0 that is if c is a positive number then the value of c becomes 0 and the function would terminate. So let us first see this working in the right way and so for example let us make this code as follows make team and then make and you run integer overflow and specify a number a such as 55 and what we see is that the sum is printed as 55 plus 1 is 56.

Now at just by eyeballing the code the bug in the code is not very obvious however what we will now demonstrate now is that by changing the input which is which would typically be in the user control the attacker would be able to make this program behave maliciously for example the attacker could give a positive value of b and make this particular function get invoked.

So the one vulnerability that we are actually going to exploit here is that each of these variables int has a size of 2 power 31, so typically this is a signed integer so the maximum value that can be represented in the signed integer on this 32-bit machine is 2 power 31 minus 1. So if this value is actually given as input to b then this maximum value plus 1 will cause a wrapping to occur and the value of c would become 0.

So the value of 2 power 31 minus 1 can be obtained we will just calculate that right now, so it is put in decimal mode and we take 2 power 31 minus 1 would give be this particular value 2147483647. So let us give this as input overflow and paste this value here and what we see is that the value of C indeed has become 0 and therefore this if condition has not entered but rather function has being been invoked.

So what attackers could do this way is that they could manipulate what inputs take you manipulate the automatic that has been computed so that there is an overflow and would not have thought of such overflow during the testing and with this they would be able to actually execute vulnerable functions in the code and do a lot of other malicious aspects. So some of these things we have actually discussed as part of this lecture corresponding to integer overflow which we have seen in the previous videos and many of these modern malware would use such vulnerabilities in integer overflow or signedness of integer or the width of integer to subvert execution and create payloads, thank you.