Format String Bugs
printf(char *format,..)
sprintf(char *dest, char *format,..)
snprintf(char *dest, int n, char *format,..)

%\n
Int x;
printf(“Hello %n, OK.”, &x);
printf(“Hello%ss%n”, name, &x);

printf internals / var args functions
printf(format, a, b)     // (format is just a pointer to a string)

Example Code: (Here compiler cannot check and match number of %’s with stack pushed variables)
void loguser(char *user)
{
    char buffer[512];
    snprintf(buffer, sizeof(buffer), user);
    ...

Assumption: For every %d or %s etc, there is a variable pushed onto the stack.

Diagram:

Print Activation Record

Print Local Variable

\( %d %d \)

(local/var) Counts total number of printed char
Here, suppose the user string entered is “%x%x%x%x%x%x....”. In this case compiler cannot check or generate error. This will actually output the contents of stack above user.

It does not crash the program, thus allows knowing the address of the handle_user_login+.... And run any code without any problem. Because even if ASLR is on, it won’t change the code location until the program is completed or crashed.

Using %n to overwrite arbitrary memory

Mem[a] ← V

Trick is to use %n
user = “<address to be written> %n

for large numbers:

0X12345678

But ARGP will increment to full 4 bytes.

user = <A1><A2><A3><A4> －－－－ %n －－－－ %n……

count = ******78
count = ******56

sprintf does not stop when buffer is full. It still processes all %s, %n etc.

To insert shell code:

user = <A1><A2><A3><A4><shellcode>%n<shellcode>%n……………….