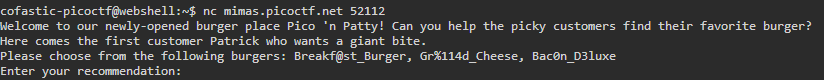
FORMAT STRING 0

After doing some research on format strings and revisiting the source code, I discovered that the option "Gr%114d\_Cheese" is the only one that fulfills the specific conditions in the code. Let's explore why:



After doing some research on [**format strings**](https://book.hacktricks.xyz/binary-exploitation/format-strings) and revisiting the source code, I discovered that the option "Gr%114d\_Cheese" is the only one that **fulfills the specific conditions in the code.**

Why? well, let’s take a look into the code again:

Int count = printf(choice1);  
if (count > 2 \* BUFSIZE) {  
 serve\_bob();  
} else {  
 printf("%s\n%s\n",  
 "Patrick is still hungry!",  
 "Try to serve him something of larger size!");  
 fflush(stdout);  
}

* The printf(choice1); line prints the user's choice.
* int count = printf(choice1); captures the number of characters printed.

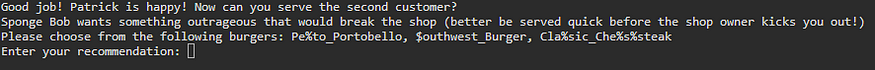
if (count > 2 \* BUFSIZE) {  
 serve\_bob();  
}

The key condition is **if (count > 2 \* BUFSIZE)**, which checks if the printed characters exceed 64 bytes (since BUFSIZE is defined as 32 bytes).

**Why Only “Gr%114d\_Cheese” Works:**

* **String Lengths**: "Breakf@st\_Burger" has 16 characters, "Gr%114d\_Cheese" has 13 characters, and "Bac0n\_D3luxe" also has 13 characters.
* **The Trick with %114d**: %114d is a format specifier in "Gr%114d\_Cheese". When printf encounters %114d, it expects to **print a number with a width of 114 characters**. Since no number is provided, it can print some garbage data,**inflating the count to 114 characters.**

This format specifier **increases the output length significantly**, making the **count** large enough to satisfy the condition **if (count > 2 \* BUFSIZE)**. This is why selecting "Gr%114d\_Cheese" moves the program to the next stage where you serve Bob.



**Moving to Bob’s Order:**

Next, we encounter another menu, and this time, the third option "Cla%sic\_Che%s%steak" reveals the flag. But why?

**Why Only “Cla%sic\_Che%s%steak” Works:**

* **Format Specifiers**: The string "Cla%sic\_Che%s%steak" contains several % symbols, typically used by printf to insert specific data types.
* %s expects a string argument.
* %steak is not valid, but %s still tries to process it.
* **Undefined Behavior**: When printf processes %s without a corresponding string argument, it can access arbitrary memory or cause **the program to crash**. This might be what**fulfills the condition for SpongeBob's "outrageous" order.**

The other choices, "Pe%to\_Portobello" and "$outhwest\_Burger", **don't trigger the same vulnerability**, making "Cla%sic\_Che%s%steak" **the only one that reveals the flag.**

