# Exam

Advanced Programming in the UNIX Environment

Chun-Ying Huang <chuang@cs.nctu.edu.tw>

# Outline

Problem style

Tools

Sample problems

Midterm scope

Q&A

# Problem Style

All problems are console-based problems

Two styles

Interaction

- Interact with a problem server
- Follow the instructions, and solve it interactively or automatically (by implementing scripts)
- Let the server call the showflag() function

Code submission

- A problem statement is given
- Implement your solution as a function or a standalone problem
- Copy-and-paste your C/C++ code to the problem server (end with **//EOF**)

# Tools

netcat

- `nc aup.zoolab.org [port-number]`

Python3 and pwntools

- https://github.com/Gallopsled/pwntools

Internet is not available - some sites are white-listed

Online documents will be available

- man pages: http://man7.org/linux/man-pages/
- Python3 reference: https://docs.python.org/3/reference/
- Pwntools reference: http://docs.pwntools.com/en/stable/
- C/C++ reference: https://en.cppreference.com/w/cpp

You can also bring your own documents (books or papers)

# Sample Problems

Interaction: The **echo** server

Code submission: The **helloworld** problem

# The **echo** Server

`nc aup.zoolab.org 20001`

Terminal — 73×13

```
[~]$ nc aup.zoolab.org 20001
Please follow my instructions shown below.
Note that there are total 1000 instructions to follow. Good luck!

1. Please type 'g' (without quotes) and press enter: g

2. Please type 'qv' (without quotes) and press enter: qv

3. Please type 'iuzok' (without quotes) and press enter: xxx

No no no ...
[~]$
```

# The **echo** Server – Source Code

The source of the sever program is provided

Two fundamental functions

- unbuffered() – disable buffering for standard input, output, and error
- showflag() – print out the flag (answer) of this problem

Terminal — 92×36

```
 1 #include "tools.c"
 2
 3 #define N       1000
 4
 5 int readword(int n) {
 6         int i, len;
 7         char buf[256], word[16];
 8         len = 1 + rand() % 8;
 9         for(i = 0; i < len; i++) word[i] = 'a' + rand() % 26;
10         word[i] = '\0';
11         printf("\n%d. Please type '%s' (without quotes) and press enter: ", n, word);
12         if((len = read(0, buf, sizeof(buf))) < 0) return -1;
13         if(buf[len-1] == '\n' && strncmp(word, buf, len-1) == 0) {
14                 return 0;
15         }
16         return -1;
17 }
18
19 int main() {
20         int i;
21         srand(time(0) ^ getpid());
22         unbuffered();
23         printf( "Please follow my instructions shown below.\n"
24                 "Note that there are total %d instructions to follow. Good luck!\n", N);
25         for(i = 0; i < N; i++) {
26                 if(readword(i+1) != 0) {
27                         printf("\nNo no no ...\n");
28                         return -1;
29                 }
30         }
31         printf("Good job!\n");
32         showflag();
33         return 0;
34 }
~
:set number                                                    2,0-1         All
```

# Solution

Implement a script to interact with the server program

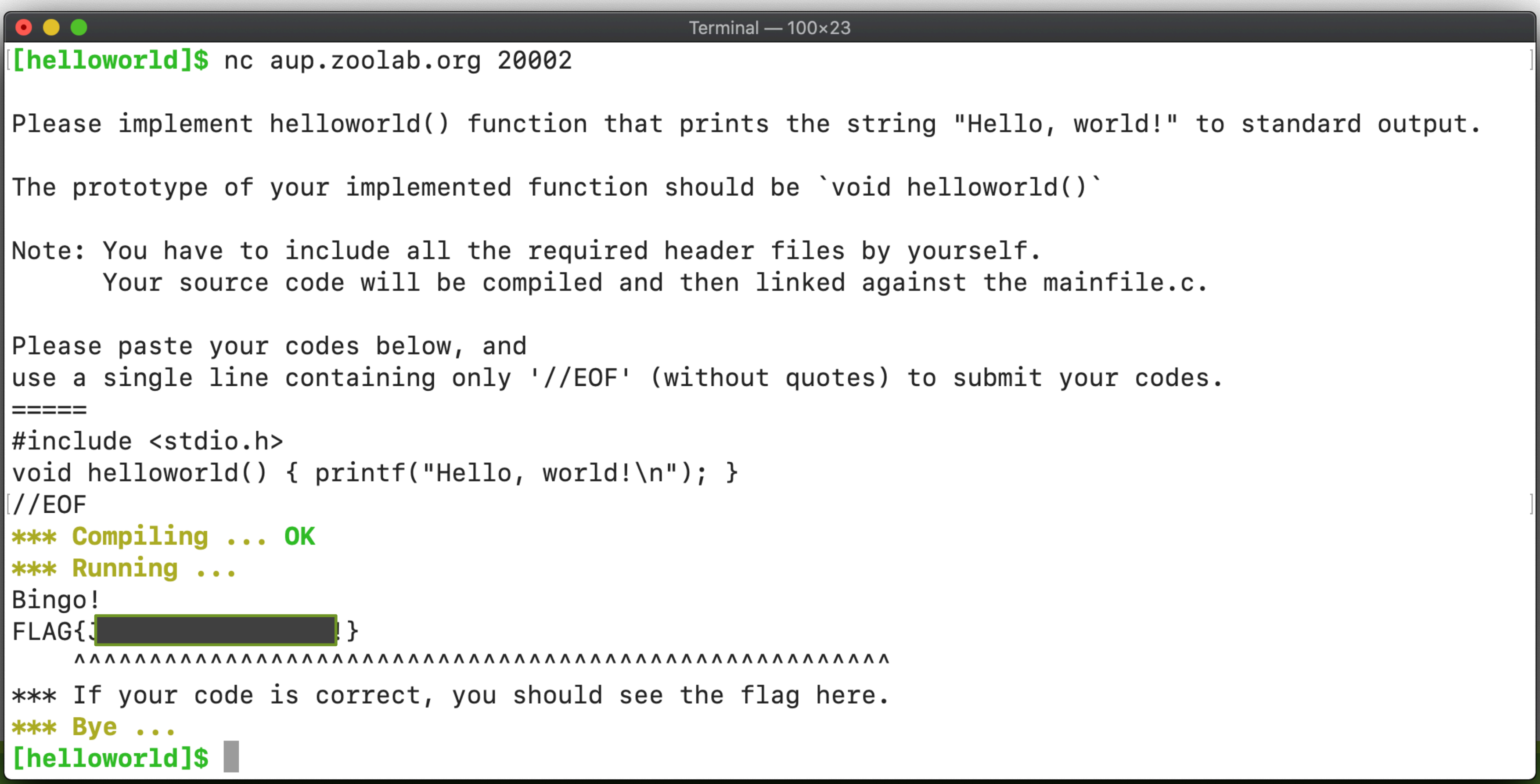
- Connect to the problem server's IP and port
- Read server response
- Parse and send the corresponding outputs

Once solved, the `FLAG` of the problem will be displayed

```
Terminal — 66×9
(pwntools) [echo]$ ./solve.py x
[+] Opening connection to aup.zoolab.org on port 20001: Done
[*] Switching to interactive mode
Good job!
FLAG{A_[redacted]ns!}
[*] Got EOF while reading in interactive
$
```

# The **helloworld** Problem

`nc aup.zoolab.org 20002`



```
[helloworld]$ nc aup.zoolab.org 20002

Please implement helloworld() function that prints the string "Hello, world!" to standard output.

The prototype of your implemented function should be `void helloworld()`

Note: You have to include all the required header files by yourself.
      Your source code will be compiled and then linked against the mainfile.c.

Please paste your codes below, and
use a single line containing only '//EOF' (without quotes) to submit your codes.
=====
#include <stdio.h>
void helloworld() { printf("Hello, world!\n"); }
//EOF
*** Compiling ... OK
*** Running ...
Bingo!
FLAG{[redacted]}
    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
*** If your code is correct, you should see the flag here.
*** Bye ...
[helloworld]$
```

# The **helloworld** Problem – How Your Program is Executed?

Your submitted codes are stored in a single file

The main function is usually implemented in a **mainfile.c** file

- Only limited syscalls are allowd in your program
- The main function is responsible to call your function

The two files are compiled independently, and then linked togather

# The helloworld Problem – The Sample mainfile

See the next page or see the course demonstration.

Terminal — 110×61

```
10 extern void helloworld();
11
12 #define HELLOWORLD      "Hello, world!"
13
14 #define xerror(x)       { perror(x); exit(-1); }
15
16 static void
17 setup_filter() {
18         scmp_filter_ctx ctx;
19         /* only the following syscalls are allowed: write exit exit_group */
20         if((ctx = seccomp_init(SCMP_ACT_KILL)) == NULL) xerror("seccomp_init");
21         if(seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(write), 0) < 0)       xerror("seccomp_rule");
22         if(seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(exit), 0) < 0)        xerror("seccomp_rule");
23         if(seccomp_rule_add(ctx, SCMP_ACT_ALLOW, SCMP_SYS(exit_group), 0) < 0)  xerror("seccomp_rule");
24         if(seccomp_load(ctx) < 0) xerror("seccomp_load");
25 }
26
27 static pid_t child = 0;
28
29 void sigchld(int s) {
30         if(child > 0 && waitpid(child, &s, 0) > 0) {
31                 if(WIFSIGNALED(s)) printf("child terminated with signal = %d.\n", WTERMSIG(s));
32                 child = 0;
33         }
34 }
35 void killchild()        { if(child > 0) kill(child, SIGKILL); }
36 void sigterm(int s)     { killchild(); }
37
38 int
39 main() {
40         int fd[2];
41         char buf[64];
42         atexit(killchild);
43         setvbuf(stdout, NULL, _IONBF, 0);
44         setvbuf(stderr, NULL, _IONBF, 0);
45         signal(SIGCHLD, sigchld);
46         signal(SIGTERM, sigterm);
47         signal(SIGINT, sigterm);
48         if(pipe(fd) < 0)                xerror("pipe");
49         if((child = fork()) < 0)        xerror("fork");
50         if(child == 0) {
51                 signal(SIGCHLD, SIG_DFL);
52                 dup2(fd[1], 1);
53                 alarm(10);
54                 setup_filter();
55                 helloworld();
56         } else {
57                 int s;
58                 if((s = read(fd[0], buf, sizeof(buf))) > 0) {
59                         if(strncmp(buf, HELLOWORLD, s-1) == 0) {
60                                 printf("Bingo!\n");
61                         } else {
62                                 printf("No no no ...\n");
63                                 return -1;
64                         }
65                 }
66         }
67         return 0;
68 }
69
                                                        69,0-1        Bot
```

# Midterm Scope

From the beginning of this semester to now

Currently we have 8 problems (exclude the two demo problems)

- Include several important concepts we introduced in the class

To be more specific …

- Basic programming practice (C/C++ and python)
- File I/O and standard I/O
- Directory operations
- Process environment and process control
- Memory layout and management

# Q & A