HOMEWORK 2

Shell Script

huyliu, hmhung

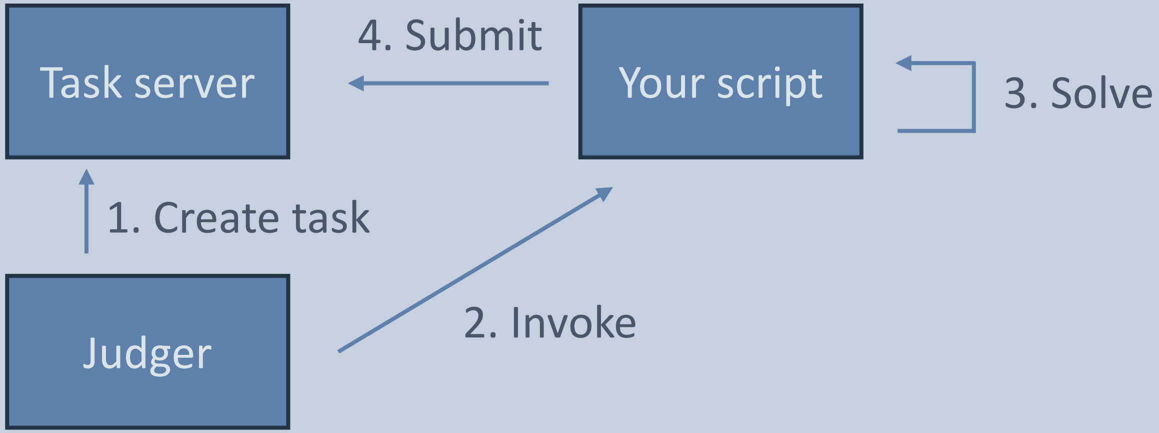
國立陽明交通大學資工系資訊中心
Information Technology Center, Department of Computer Science, NYCU

Requirements

- In homework 2, you need to write a shell script to complete the task from the task server. (/bin/sh and open source CLI tool only, others is not allowed. e.g. bash, python, go, rust…)
- Judger will log in to your server via ssh and invoke the script. If you send the wrong answer to the task server (or timeout), you will not get the point of the checkpoint.
- Hint:
  - CLI tools: curl, jq, awk, grep…

Task Server

- Create tasks and validate the problem answer
- Endpoint: http://10.113.0.253
- Flow:



Task Server API (1/4)

- POST /tasks
  - Request
    - Body (application/json)

```
{
  “type”: “MATH_SOLVER”
}
```

  - Response
    - Body (application/json)

```
{
  “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
  “type”: “MATH_SOLVER”,
  “status”: “PENDING”,
  “problem”: “1 + 1 = ?”
}
```

Task Server API (2/4)

- POST /tasks/:id/submit
  - Request
    - Body (application/json)

```
{
   “answer”: “2”
}
```

  - Response
    - Body (application/json)

```
{
   “message”: “Accept”
}
```

```
{
   “message”: “Wrong answer”
}
```

Task Server API (3/4)

- GET /tasks/:id
  - Response

```
{
   “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
   “type”: “MATH_SOLVER”,
   “status”: “PENDING”,
   “problem”: “1 + 1 = ?”
}
```

```
{
   “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
   “type”: “MATH_SOLVER”,
   “status”: “ACCEPT”,
   “problem”: “1 + 1 = ?”
}
```

Task Server API (4/4)

- GET /tasks/:id
  - Response

```
{
   “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
   “type”: “MATH_SOLVER”,
   “status”: “WRONG_ANSWER”,
   “problem”: “1 + 1 = ?”
}
```

```
{
   “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
   “type”: “MATH_SOLVER”,
   “status”: “TIMEOUT”,
   “problem”: “1 + 1 = ?”
}
```

Script Spec (1/3)

- Provide an executable shell script placed at /home/judge/hw2.sh with following available options:
- -p
  - Task id
- -t
  - Task type
  - “JOIN_NYCU_CSIT” | “MATH_SOLVER” | “CRACK_PASSWORD”
- -h
  - Optional
  - If the option is specified, just print the usage to stderr(see the next page)

Script Spec (2/3)

- If the given task type is not equal to the task server’s record, you should print “Task type not match” to stderr.
- If the given task type is not valid, you should print “Invalid task type” to stderr.
- Extra: collect the problem field in “JOIN_NYCU_CSIT”, take a screenshot of the page then upload to E3. You will get one point for each screenshot at most 5 points if you collected more than 5 screenshots.

- Hint: Judger will call POST `/tasks` for you and use appropriate options to invoke the script. If you need to test your script in your environment, you can write another script to do it.

Script Spec (3/3)

- Invalid arguments should be rejected with a non-zero status code, with the exact help message outputted to stderr

```
$ hw2.sh -a
hw2.sh -p TASK_ID -t TASK_TYPE [-h]

Available Options:

-p: Task id
-t JOIN_NYCU_CSIT|MATH_SOLVER|CRACK_PASSWORD: Task type
-h: Show the script usage
```

Problem Spec – Join NYCU CSIT

- Problem = any string
- Just send “I Love NYCU CSIT” to the task server

- Hint: problem contains some “interesting” information

Problem Example – Join NYCU CSIT

- Problem

```
{
   “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
   “type”: “JOIN_NYCU_CSIT”,
   “status”: “PENDING”,
   “problem”: “https://i.imgur.com/wP3ST2x.jpeg”
}
```

- Answer

```
{
   “answer”: “I Love NYCU CSIT”
}
```

Problem Spec – Math Solver

- Problem = a (+/-) b = c
  - -10000 <= a <= 10000
  - 0 <= b <= 10000
  - -20000 <= c <= 20000
- Add and subtract only, you don’t need to consider other conditions. If you get a problem not obey the above definition, just send “Invalid problem” to the task server.

Problem Example – Math Solver

- Problem

```
{
   “id”: “4a9c99f4-0241-4f3d-a003-7b2f6bb455db”,
   “type”: “MATH_SOLVER”,
   “status”: “PENDING”,
   “problem”: “1 + 1 = ?”
}
```

- Answer

```
{
   “answer”: “2”
}
```

Problem Example – Math Solver

- Problem

```
{
   “id”: “ccc4ecdb-c8ac-4f8a-a80c-92858e20d390”,
   “type”: “MATH_SOLVER”,
   “status”: “PENDING”,
   “problem”: “12 - 5 = ?”
}
```

- Answer

```
{
   “answer”: “7”
}
```

Problem Example – Math Solver

- Problem

```
{
   “id”: “29c19918-222c-4bd7-800a-24f1556cbe7e”,
   “type”: “MATH_SOLVER”,
   “status”: “PENDING”,
   “problem”: “46 * 2 = ?”
}
```

- Answer

```
{
   “answer”: “Invalid problem”
}
```

Problem Spec – Crack Password

- The problem would be a Caesar cipher, you need to decrypt it and send it back.
- Shift x: 1 <= x <= 13
- Plaintext regex: NYCUNASA\{[A-Za-z]{16}\}
- If you get a problem not obey the above definition, just send “Invalid problem” to the task server.
- Hint: “{“ and “}” can be ignored in the rotate process

Problem Example – Crack Password

- Problem

```
{
   “id”: “9d2cde78-837c-4e72-bc9d-4f7204b5520e ”,
   “type”: “CRACK_PASSWORD”,
   “status”: “PENDING”,
   “problem”: “ALPHANFN{QfLWrLExdqgSufIi}”
}
```

- Answer

```
{
   “answer”: “NYCUNASA{DsYJeYRkqdtFhsVv}”
}
```

Shift 13

Problem Example – Crack Password

- Problem

```
{
   “id”: “99706c79-b93d-4251-a3cd-ce818d14fb04”,
   “type”: “CRACK_PASSWORD”,
   “status”: “PENDING”,
   “problem”: “QBFXQDVD{QcMXBVxFxlXAbJyK}”
}
```

Shift 3

- Answer

```
{
   “answer”: “NYCUNASA{NzJUYSuCuiUXyGvH}”
}
```

Problem Example – Crack Password

- Problem

```
{
   “id”: “e9fdf146-aa5d-4a04-964b-9992077c33ac”,
   “type”: “CRACK_PASSWORD”,
   “status”: “PENDING”,
   “problem”: “QBFXADVD{QcMXBVxFxlXXbJyK}”
}
```

- Answer

```
{
   “answer”: “Invalid problem”
}
```

Shift ???

Shellcheck

- ShellCheck is a static analysis tool for shell scripts. It helps catch errors and suggests improvements to make your scripts better.
- shellcheck -s sh -a [your_script] (should return zero)
- Version : 0.10.0

SH(1): getopts

- getopts optstring var

  The POSIX getopts command. The getopts command deprecates the older getopt(1) command. The first argument should be a series of letters, each possibly followed by a colon which indicates that the option takes an argument. The specified variable is set to the parsed option. The index of the next argument is placed into the shell variable OPTIND. If an option takes an argument, it is placed into the shell variable OPTARG.

Restrictions

- Must not use any other interpreters, compilers or programming languages(such as Python, Ruby, Node.js, Golang, Rust, Perl, GCC, Clang…)
- Must not call any other self-written scripts, binaries or executables.
- Only one shell, sh, is allowed.
- Common tools (e.g. date, openssl, jq, etc.) are allowed.
- If you are not sure whether a tool is allowed, please ask TA on Google Groups.

Grading

Automated grading (Online Judge), 105 pts

- Usage
  - Invalid options
    - Exit Code (5%)
    - Help Message (5%)
  - Invalid type (5%)
  - Task type validation (5%)
- Arbitrary argument position (15%)
- Shellcheck (10%)
- Tasks
  - Join NYCU CSIT (15%)
  - Math Solver (20%)
    - Invalid problem check is included
  - Crack Password (20%)
    - Invalid problem check is included
- Bonus: Easter egg collection (5%)

Attention

- Your work will be scored by Online Judge system
  - Only the **LAST** submission will be scored
  - Late submission will **NOT** be accepted
- **ALWAYS BACKUP** your system before submission, as we may do malicious actions
- Set the IP address 10.113.$ID.11
- You are restricted to use only sh to complete your work
  - If you’re not sure what’s allowed, contact TAs.
  - TAs reserve the right of final explanations. Specs and the points of each sub-judges are subject to change in any time.
- Make sure everything works after reboot

Rules

- TAs reserve the rights of final explanations
- **Open from 9/26 (Thu) 19:00**
- **Deadline: 10/17 (Thu) 23:59**
- Late submissions will **NOT** be accepted

Help

- Join NCTUNASA google group
  - If you have any question, you can post your problem in this group, TAs and Students will help you.
  - https://groups.google.com/g/nctunasa
- UNIX 常見指令教學
  - https://it.cs.nycu.edu.tw/unix-basic-commands
- How To Ask Questions The Smart Way
  - https://github.com/ryanhanwu/How-To-Ask-Questions-The-Smart-Way