A light-weight Unix-based Contest Management System

Axel T. Schreiner & Paul T. Tymann.
Computer Science, Rochester Institute of Technology, USA
http://www.cs.rit.edu/~ats/talks/icpc/
Objectives

- *Kiss*
  - Rely on proven mechanisms
- *Kiss*
  - Support recovery from failure
- *Kiss*

*keep it simple, stupid!*
Stakeholders

- Team
- Judge
- Coach
Actions

- Team
- Judge
- Coach
Actions

Team
- submit solution
- request clarification
- receive feedback (judgement/clarification)

Judge

Coach
Actions

Team
- submit solution
- request clarification
- receive feedback (judgement/clarification)

Judge
- receive request (submission/clarification)
- send reply

Coach
Actions

Team
- submit solution
- request clarification
- receive feedback (judgement/clarification)

Judge
- receive request (submission/clarification)
- send reply

Coach
- watch standings
Actions

enqueue

depqueue

to

one to one

all to all

Scoreboard

watch
Mechanisms

enqueue

one to one

dequeue

all to all

Scoreboard

watch
Mechanisms

enqueue

decqueue

all to all

one to one

mail

Scoreboard

watch
Mechanisms

enqueue one to one
dequeue

Scoreboard

mail
nfs/cd

watch
Mechanisms
Team

- Unique user and group id, home directory
- Mail viewer running for feedback
- PATH set to team’s commands:

  - $ scoreboard & (optional)
  - $ compile problem file ...
    sh, 49 lines
  - $ submit problem file ...
    sh, 85
  - $ clarify problem question
Judge

- Unique user and common group id
- Mail editor running for clarifications
- PATH set to judge’s commands:

```
$ scoreboard &  # optional
$ status [-judged] [-working] &  # sb, 75
$ judge [-run] team problem  # sh, 185
```
Scoreboard

- **sb.Model**  
  - “chatroom” server: single input port, multiple output ports (can be frozen individually), TCP

- **sb.View**  
  - client: displays pushed HTML

- **sb.Controller**  
  - client: sends command line to input port; easier to script than `telnet`, can apply fixes
Links

- **Documentation and download**
  

- **Sources for inspection**
  
  http://www.cs.rit.edu/~ats/projects/acm/contest/

- **Team guide**
  
  http://www.cs.rit.edu/~icpc/environment.html
Submission

- setgid-ed shell script (85 lines)
- relies on common file system

create timestamp
create PENDING file to prevent resubmission
copy files to team/problem-specific directory
tell scoreboard, mail timestamp to all judges
Judging

- shell script, requires group permission (185 lines)
- plugin interface for automated judging

```bash
cd to team/problem-specific directory
link WORKING file to prevent race among judges
compile as per compile command
execute with each test.n input file
(by default) show aside answer.n output file
select standard feedback via menu
tell scoreboard, mail feedback to team
remove lock files
```
Recovery

- Mail (mainly submission timestamps) archived in inbox of (abstract) user id of contest sources
- Submissions archived with time stamps; can replay any submission
- Scoreboard command language for input, stored in log file; can be replayed or used for manual fixes
Security
Security

- Teams: separate users (and groups).
- Judges: mutually excluded, separate directories.
- Submissions: via setgid into judges’ group, i.e., test data and archives are separate from teams.
Security

- Teams: separate users (and groups).
- Judges: mutually excluded, separate directories.
- Submissions: via setgid into judges’ group, i.e., test data and archives are separate from teams.

- Code is open and documented.
- Trojans could retrieve judging data, e.g., using sockets, mutate archives, etc.
- `telnet/mail` allow to subvert scoreboard and status.
Security

- Teams: separate users (and groups).
- Judges: mutually excluded, separate directories.
- Submissions: via setgid into judges’ group, i.e., test data and archives are separate from teams.
- Code is open and documented.
- Trojans could retrieve judging data, e.g., using sockets, mutate archives, etc.
- `telnet/mail` allow to subvert scoreboard and status.
- Could add encryption, random data, and replay from archive.
Portability

- Deployed on Solaris.
- Scoreboard also on JVMs on Windows and Mac.

- Scripting based on Bourne Shell.
- Locking based on Unix file operations.
- Security based on Unix permissions.

- Teams on Windows or multi-site contests might be possible with tunneling over ssh.