Validating the Security and Stability of the Grader for a Programming Contest System

Tocho TOCHEV, Tsvetan BOGDANOV
Sofia University, Bulgaria

Why?

- Pressure to change the grading system
  - New security methods and fixes
  - New grading methods / task types
  - Always in preparation for the next big event
  - New grading systems
- Different software and hardware specifications
- Ensuring ”Fair game”

How Could It Be Verified?

- Code review
  - Peer
  - External audit
- Dry run contests
- Suite of submissions

What Can Be Tested?

- Grader
  - Stability
  - Security
- Other components
  - Contestant interface
  - Network
  - …
- Related articles

Stability

- Correct solution (for each programming language)
- Wrong answer
- Time-limit exceeded (including deadlocks)
- Memory limit exceeded
- Runtime error
- Available libraries (for each programming language)

Security

Attacks during:

- Compilation
- Sandboxing
- Result checking

Attacks During Compilation

- Excessive submit size
- Referencing forbidden files
- Compile time/memory exploits

Attacks During Sandboxing

- Read/write files/directories *
- Accessing (network) sockets
- Multiple threads
- Multiple processes *
- Raise privileges / break out of sandbox
- Exploit the judge's module (in reactive tasks)
- Denial of service (wasting resources)

* Banning policy should be flexible

Exploits During Checking

- Sandboxing checkers
- Expecting abnormal checker execution

Further Work

- Our experience
- Central checklist / Sample submissions
- Similar checklists for other parts of the system
  - Contestant interface
  - Network
  - Printing
  - Backup
  - Contestant machine setup