PROTOS

Systematic approach to eliminate software vulnerabilities

Juha Röning

Marko Laakso

Ari Takanen

ouspg@ee.oulu.fi

http://www.ee.oulu.fi/research/ouspg

Motivation

- Software vulnerabilities prevail:

  “Fragile and insecure software continues to be a major threat to a society increasingly reliant on complex software systems.”
  - Anup Ghosh [Risks Digest 21.30]

- A focal problem area is software implementation, which may introduce potential for unanticipated and undesired program behaviour
- We have made some rather strong claims:
  - (A) Secure programming errors are systematic!
  - (B) Many vulnerabilities could be eliminated with low cost!
  - (C) Dynamic black-box testing would be a decent first-aid!

PROTOS presentation outline

- Background and context - Röning
- Testing approach - Laakso
- Results and vulnerability handling - Takanen

OUSPG

- Active as an independent and academic research group in the Computer Engineering Laboratory since summer 1996.
- Our purpose:

*“To study, evaluate and develop methods of implementing and testing application and system software in order to prevent, discover and eliminate implementation level security vulnerabilities in a **pro-active** fashion.*
*Our focus is on **implementation level** security issues and software security testing.”*

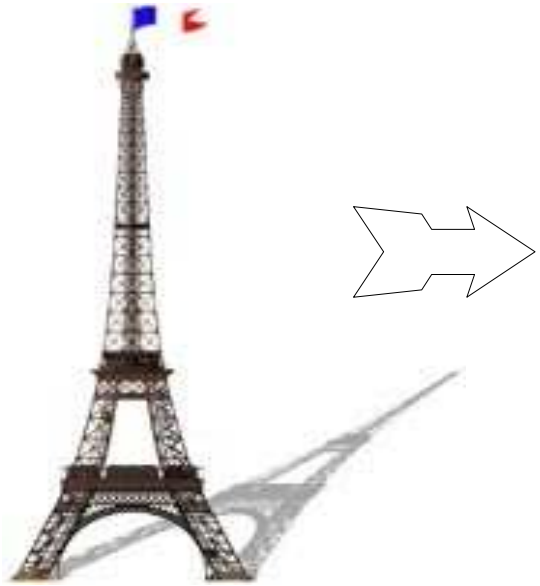
Implementation & testing

- The total security of the release is the product of the specification, design, implementation and testing performed in the software process.

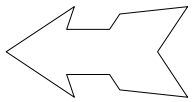
1. Specification
2. Design
3. Implementation
4. Testing
5. Maintenance/Use

Security Development

- Distribution of effort in development



Specification
Design
Implementation
Testing
Maintenance

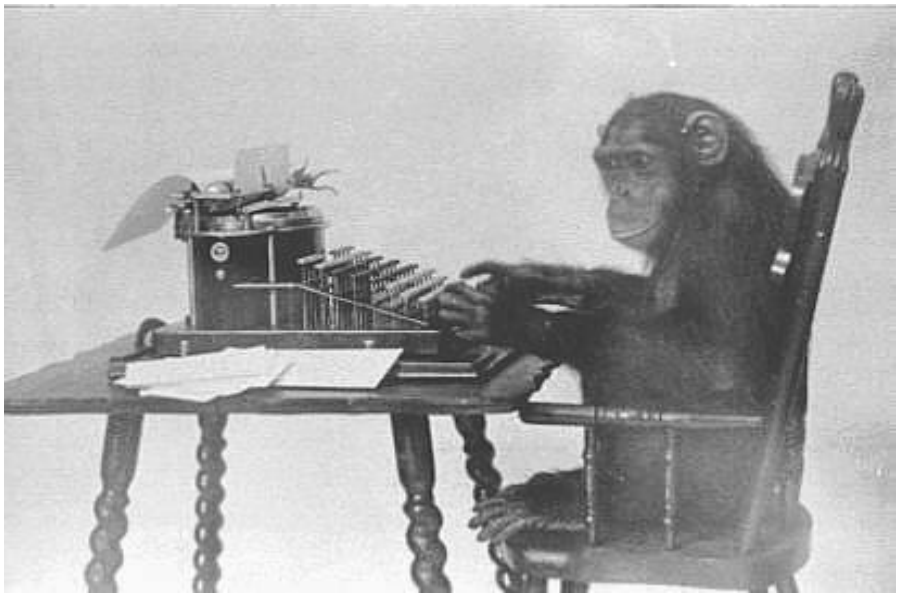


Security Endangered by Vulnerabilities

- InfoSec vulnerabilities endanger (CIA):
  - **confidentiality** of information
  - **integrity** of information
  - **availability** of information
- Security may have Safety implications
- InfoSec vulnerability could be caused by:
  - a software failure
  - a misconfiguration
  - a human or procedural error
- What threatens our InfoSec:
  - Spontaneous combustion
    - Hardware and software reliability
    - Natural disasters
  - Malicious activity (who we prepare for)
    - Pranksters, Script kiddies, Terrorists, Professionals ...

Our approach - in a nutshell

Today, thousands of gifted and patient, but uncoordinated monkeys are pounding different products in order to reveal vulnerabilities.



Visual by
http://www.PDImages.com

Think of us as rather dumb monkeys using a monkey-machine and systematic methodology to eliminate the most trivial ones.

Vulnerability Reality Check

THREAT * VULNERABILITY = RISK

- Security is not the Holy Grail:
  - Address and understand risks first.
  - Risk arithmetics [T * V = R]:
    - 0 * V = 0 (no threats equals no risks)
    - T * 0 = 0 (no vulnerabilities equals no risks)
- Risk is impossible to assess without possibility of measuring the vulnerability and threat
- Reactive or Proactive approach to the risk

Searching for the process Grail to reduce vulnerability

- Bug prevention and elimination methods in the software development process (by B. Beizer)
  - Thorough analysis
  - Prototypes
  - Analytical models
  - Formal methods
  - Inspections
- Awareness: skills in secure programming and safety engineering
- Testing is the means for discovering the bugs that persist after these

Searching for the technical Grail to reduce vulnerability

- Alternatives for educating the engineers:
  - Safer libraries
  - Better compilers and languages (e.g. Java)
  - Operating System (kernel) solutions
- Methods behind them:
  - Bounds checking / strong typing (run/compile time)
  - Non-executable stack, stack guarding techniques
  - Sandboxing and managed code
  - Code signing (You will know who to blame? ;)
- Deployment? Adaptation? Completeness?
  - There will be room for a safety net provided by testing

Software Security Testing

- Evolution of the software testing (by B. Beizer):
  - **0:** No difference between testing and debugging
  - **1:** The purpose of testing is to show that the software works.
  - **2: ... is to show that the software DOESN'T work.**
  - **3:** ... purpose of testing is to reduce the perceived risk ...
  - **4:** ... a mental discipline ... (minimum effort in test stage)

- From the practical security perspective:
  - Software vendors are at phase 1 (conformance)?
  - Vulnerability research is stuck at phase 2?

Black-box vs. White-box

- Black-box testing (no src)
  - Cheap? First-aid?
  - Can be adopted in QA?
  - Poor code-path coverage
  - Effective against casual bugtraq disclosures (trivial vulnerabilities)

    ...

    Same starting point as for any bugtraq submitter?
- Methods are complementary
  - Our approach is black-box testing
- White-box (with src)
  - Costly?
  - Complex
  - 3rd party software?

Static vs. Dynamic testing

- Dynamic testing
  - Testing at run time
  - Poor code-path coverage?
  - Coverage improved by stress-test suites?
  - Proven effective even for passive monitoring
  - Vulnerabilities detected by actual run-time context
- Methods are completary
  - Our approach is dynamic testing

- Static testing
  - Off-line testing
  - Complex?
  - Vulnerabilities detected by emulated run-time context

Software Security Testing

- From *Software Testing Techniques* by Boris Beizer (2nd Edition, p. 2):

"Thrill to the excitement of the chase!
Stalk bugs with ***care, methodology, and reason****. Build traps for them.*
....
Testers!
Break that software (as you must) and
drive it to the ultimate
- but don't enjoy the programmer's pain."

PROTOS

Testing the Security of Protocol Implementations

- **PROTOS will:**
  - Develop practical vulnerability testing methods
  - Distribute awareness
  - Develop procedures to prevent errors
  - Inform vendors of found vulnerabilities
- Results public, except for the bug reports and demonstration exploits

PROTOS - "the goal"

- Dispite existence of TTCN and others, vulnerabilities were constantly found
- Testing framework
  - *a skeletal structure designed to support or enclose something* - Webster
- Testing platform (a.k.a. scripting platform)
  - *(Mil.) (a) solid ground on which artillery pieces are mounted ... (b) a metal stand or base attached to certain type of artillery pieces* - Webster
- At least we learn the protocols ... ;)

PROTOS
- Framework & Platform