Information Security Aspects of Public Software

Henrique Soares\textsuperscript{1}, Raphael Machado\textsuperscript{1}, Bruno Salgado\textsuperscript{1}, Rafael Soares\textsuperscript{1}, Jarbas Cardoso Júnior\textsuperscript{2} (presenter), Luis Felipe Costa\textsuperscript{3}

\textsuperscript{1}Clavis Segurança da Informação
\textsuperscript{2}CTI - Centro de Tecnologia da Informação
\textsuperscript{3}Ministério do Planejamento, Orçamento e Gestão
Public Software

• Software as a public good
  – Strategic importance of software availability
  – Information systems in public services
  – Government as a software provider/endorser

• Public Software initiatives around the world
  – Europe (ISA)
  – United States (SPI)
  – Brazil (Portal do SW Público)
Information Security

• Availability, confidentiality, integrity
• Computer systems security
• Software vulnerabilities
• Cybercrime, cyberspionage, cyberwar
Importance of validation of Public Software

• “Meaning” of the “Public Software Seal”
  – Endorsement (public agents, government)
  – Appropriate for sensitive/critical applications

• Typical Applications of Public Software
  – Email servers
  – Network management
  – Distance learning
  – Content management and publishing
  – Electronic Elections
  – Information Systems for Municipalities
  – etc. etc. etc…
Validation Process for Public Software

- Purpose of software / public interest
- Functional testing
- Performance evaluation
- Security architecture
- Code analysis
- Security testing
Security Architecture

• Modules and architecture
  – What are the modules of the software and how are they organized?

• Security requirements
  – What should be protected? What kind of protection (privacy, integrity, identification, etc.)?

• Protocols
  – How do the modules interact? What are the communication media? Are they protected?
Code analysis

• Focus on the source code
  – Always available (open source)

• Static analysis
  – Control flow graph, call graph, ICFG
  – Describe the “structure” of the software

• Dynamic analysis
  – Slicing and tainting
  – Traces the flow of data
Control Flow Graph (high level)

```c
int modexp(int y, int x[], int w, int n) {
    int R, L;
    int k = 0;
    int s = 1;
    while (k < w) {
        if (x[k] == 1)
            R = (s*y) % n;
        else
            R = s;
        s = R*R % n;
        L = R;
        k++;
    }
    return L;
}
```
Control Flow (low level)
Call graphs (high/low levels)

Binary code

Source code
Security testing

• Penetration Testing
  – Web-based applications and other architectures

• Fuzzing
  – Test the software against several kinds of input
  – Usually allows to identify buffer overflow and code injection vulnerabilities, among others

• Debugging
  – Step-by-step execution of software

• Emulation / Virtual Machines
  – Execution of software in a controlled environment
Validation results and report

- Possible results of a validation process
- Report
  - Scope
  - Vulnerabilities
  - Recommendations
- Responsibilities
  - Public agent competencies
  - The role of third parties
Conclusion (1)

- Software is an issue of public interest
- Government must guarantee that citizens are provided with adequate software
- Government can work so as to provide such software
- A nice model is where a public agent “validates” third parties’ software
- Such “endorsement” stimulates the use of such software
A careful validation process should be performed in a software before it can be declared as “Public Software”

- The responsibility of the public agent is huge

Such validation process should be strongly oriented to Information Security aspects

- The risks of insecure software can be considered a national security issue