Foreign Source Software

Systems & Software Technology Conference
April 2009

Presented by:
Jennifer Guild
Reality

- Almost all software, proprietary as well as open source, has some form of foreign involvement.
- Foreign influences in modern software negatively effects the DoD’s security.
• In the past, proprietary software was typically developed using in-house resources
• Now, most companies have at least part of their development off-shore
• Systems use components developed by external sources
Proprietary

- Proprietary software is developed and owned by a commercial entity, and sold or licensed.
- Proprietary software may not be redistributed without permission of the owner of the software.
Open Software

• Any application developed as a public collaboration or whose source code is made available to be freely shared, used, modified, improved, or redistributed

• Open source software may be distributed at no cost or may be sold as a “commercial” product
  • No cost – CentOS
  • Commercial – Red Hat
COTS

- Commercial Off-The-Shelf (COTS) products may be either proprietary or open source
  - Ex: Red Hat Enterprise Linux v5 (RHEL5) is open source but distributed by Red Hat
- Software which was previously produced in proprietary form can become available as open source
  - Ex: Solaris went from proprietary source in Solaris 8 to open source in Solaris 10
Software Development

- Open source software development
  - Anyone can make changes
  - Changes to the source tree generally occur via a community vetting process
- Commercial, proprietary software normally has no such “community” vetting process
Vulnerabilities

• Usually defined:
  • Errors in the software requirements or implementation
  • A weakness in process, administration, or technology that can be exploited
• Probably exists in all source code regardless of its nature
Malicious Software

• Embedded, modified source code, or application
• Intentionally malicious
• Subverts intended operation, of the system, possibly covertly
• Maliciously embedded source code is equally likely in open source software and proprietary software
“Needle in the Haystack”

- Most software applications are at least 1 million lines of code in size
  - XP has 40 million lines
  - RHEL5 has 30 million lines
- US Naval Postgraduate School project to subvert a kernel in as few lines as possible
  - A student inserted 8 lines total (5 lines in one location, 3 in another) into the Linux kernel
  - Successfully subverted several million line kernel
• Not feasible to evaluate the assurance of source code to reveal any malicious intent
• As number of lines of source code increase:
  • Complexity increases
  • Increase misuse of principle of least privilege
  • Percentage of source code that can be reviewed per year decreases
Risk Mitigations

For proprietary and open source software:

• Undergo verification and validation testing during evaluation or certification process
  • Should reveal known vulnerabilities
  • Mitigations can be instituted prior to implementation
• Follow good configuration management processes and software engineering practices
• Open source approaches do allow the potential for discovering how data is being processed, the protocols being utilized, and communication channels.

• Mitigations may be put in place prior to the software’s implementation.
• Identify configuration based vulnerabilities
  • Responsible for roughly half of the vulnerabilities detected in systems
• Proactively managing systems by deploying, base-lining, and monitoring effective standardized, security configurations allows for the potential of identifying vulnerabilities
  • Integrity checking software to determine if key files have been modified from baseline configuration
CM potential for Malware

- Potential for preventing the *subversion of the supply chain* of software components
  - Libraries
- Good CM processes for both software and systems
  - Can decrease the potential of the insertion of malware
Assumptions

• Requirement for stronger software assurance
  • Information sharing links more information infrastructure together
  • Majority of DoD systems are low assurance
• No one size fits all solution
• Each situation must be assessed to determine if it requires high, medium, or low assurance
Technical Assurance Level

- Low assurance requirements
  - Validation and verification evaluation
- Medium assurance requirements
  - Review configuration management
  - Enforce good software engineering practices
  - Detect review
  - Random source review
High Assurance

• Requirement for software to perform only what is specified without fail
• Requires formal methods, or mathematical proofs of security properties of the software
  • Normally conducted on the security relevant aspects of software
  • Do not necessarily mitigate foreign involvement or incorrect/invalid assumptions
• Proofs are evaluation and certification artifacts that can be reviewed by the community
Summary Considerations

• What is the extent of foreign involvement in software used by the DoD?
• What are the associated Information Assurance (IA) concerns?
• Are there possible mitigations for those concerns?
Questions?

- Jennifer Guild
- jennifer.guild@navy.mil
- 843-218-4879