Security Assurance and Agile Development – an Industry Perspective

John Heimann
Vice President – Security Program Management
Global Product Security
Program Agenda

1. Oracle and Agile
2. What is Software Security Assurance?
3. Oracle Software Security Assurance
4. Security and Agile
5. Conclusion
1 Oracle and Agile

2 What is Software Security Assurance?

3 Oracle Software Security Assurance

4 Security and Agile

5 Conclusion
Oracle and Agile

• Oracle’s original development methodology was loosely waterfall
  – Founded in 1977, many early government customers
  – Functional and design specifications of varying levels of detail

• Actual design and implementation was hybrid of waterfall and what is now called agile
  – Based on technical vision/innovation
  – Driven by highly technical senior staff (individual contributors and management) working with dedicated development teams

• Oracle Software Security Assurance (OSSA) evolved in this context – only loosely tied to waterfall

• Acquisitions introduced diversity in development culture and further diluted waterfall approach
Oracle and Agile

• Many acquisitions in recent years with formal agile development methodologies
• Most product development now involves some form of agile
  – Shorter release cycles due to rate of technology change, competitive pressure
  – Shift of business to cloud – “DevOps”
• OSSA uptake by agile has been mostly straightforward
• This briefing focuses on security practices that have been successful in both agile and waterfall
• Some agile limitations are noted
Oracle and Agile

What is Software Security Assurance?

Oracle Software Security Assurance

Security and Agile

Conclusion
What is software security assurance?

Definition

*The process of ensuring that software is designed to operate at a level of security that is consistent with the potential harm that could result from the loss, inaccuracy, alteration, unavailability, or misuse of the data and resources that it uses, controls, and protects.*

Security assurance implications for vendors and developers

• Software must have been designed securely
  – Security must be “built in, not bolted on”
  – Software must provide adequate security controls (e.g. reflecting the data it will store, the threat environment in which it will operate, etc.)

• Software must have been securely developed
  – Often associated with process and documentation and considered to be in conflict with agile – but doesn’t have to be.
  – Security must be a core element of the development process
  – Secure design and coding principles must have been followed
  – Vulnerability discovery tools must be used
  – Software must have been developed in a secure environment
Cultural implications for vendors and developers

- Security must be embedded in the organization’s DNA
- Organization must recognize that there is no one-time solution but that security is an ongoing commitment (continuous need to “raise the bar”)
1. Oracle and Agile
2. What is Software Security Assurance?
3. Oracle Software Security Assurance
4. Security and Agile
5. Conclusion
Oracle Software Security Assurance

Oracle Software Security Assurance encompasses all the continuously improving processes, procedures, and technologies implemented by Oracle to ensure that Oracle’s products are meeting our customers’ security requirements, while providing for the most cost-effective ownership experience.
OSSA highlights

• Maintaining the security posture of ALL Oracle customers is one of the greatest priorities of Oracle
• Applies to ALL Oracle software products throughout their lifecycle, including software components of hardware products (e.g., firmware) and cloud solutions
• Evolves constantly to adapt to new technologies, threats, and product use cases
OSSA programs

- Oracle security programs affect the entire product lifecycle

- Key programs include:
  - Secure Coding Standards, Approved technologies
  - Third party code formal approval process
  - Developer Training, Security Organizational Development
  - Architectural Risk Modeling
  - Secure Configuration Initiative
  - Critical Patch Update & Security Alert
  - Vetting and use of security tools
  - Formal release signoff
  - Internal and external security assessments

- For more information, see http://www.oracle.com/us/support/assurance
1. Oracle and Agile
2. What is Software Security Assurance?
3. Oracle Software Security Assurance
4. Security and Agile
5. Conclusion
Security and Agile

- Most OSSA programs fit into both waterfall and agile
  - Not specifically tied to waterfall milestones
  - Can be implemented within context of agile

- Recommendations for secure development
  - Organizational readiness
  - Requirements definition
  - Implementation
  - Definition of Done

- Challenges with Agile
Organizational Readiness

- Developing a culture of security is as important as specific assurance activities

- Senior management must recognize and communicate that security is a core value
  - Requires time and resources
  - May impact features and schedule

- Must establish corporate security standards and organization
  - For small organization, a single security team and process may work
  - Large and complex organizations, like Oracle, typically require more complex model
    - Central security organization to define, coordinate overall program
    - Security staff within each business unit to implement security program tailored to specific development approach, business needs
Organizational Readiness at Oracle

• Key personnel identified
  – Security Leads
  – Security Points of Contact (SPOCs)

• Security infrastructure in place
  – Standards
  – Training
  – Tools
  – Processes
Key Personnel Identified – Security Leads

• Security Leads
  – Security leaders/strategists within a business unit (lead by Sr. or Exec. VP) representing 100s -1000s of developers
  – Sr. individual contributor or mid-senior manager

• Set standards for security in a business unit
  – Tailor OSSA requirements into organization development and release practices
  – Identify security training requirements for organization
  – Select automated security tools
  – Customize release checklists
  – Primary liaison with central security team (Global Product Security)

• Good security leads are essential for a good security assurance program
Key Personnel Identified – SPOCs

- Hands-on experts within each product or product component
- Typically one for every 10-100 developers (average approx. 1:40)
- Familiar with actual product code, expert in security
- Have specialized QA SPOCs who focus on security testing
- Manage security at tactical level
  - Triage bugs
  - Review code
  - Provide guidance to developers
  - Complete security release checklists
- Ideally, agile development teams each have an active SPOC member
Standards

• OSSA Standards
  – Define each activity to be followed
  – Tasks are not tied to specific development methodologies

• Secure Coding Standards
  – Approved technologies for security critical functions (crypto, authentication, authorization)
  – Coding patterns to use vs. avoid, with examples of what can go wrong
  – Developed by Oracle based on 30+ years of coding (and code bugs)
  – Published secure coding standards are now available (e.g., CERT) and are incorporated by reference
Standards, cont.

• All developers are expected to have been exposed to OSSA and Secure Coding Standards
• SPOCs are expected to have familiarity with OSSA and SCS
• Security Leads must be expert in OSSA and SCS
Training

• Knowledgeable developers are critical for effective development

• Mandatory OSSA training
  – Required for all development staff
  – Importance of security, overview of security process and resources
  – Thinking like a hacker
  – Introduction to Secure Coding Standards

• Additional training
  – Technology-specific training (Java, C, web, PL/SQL, etc.)
  – Program-specific training (tools, architectural analysis, etc.)
  – Organizationally-specific training (SPOCs, QA, etc.)
Security Requirements

• Can be stated as features, stories, or as part of definition of done
  – Security requirements should be called out otherwise they won’t just happen
  – Critical requirements should be called out explicitly
  – Other requirements can be called out implicitly, e.g., by reference to standards

• Positive features/stories
  – E.g., users must be authenticated before they can access sensitive resources
  – Different from (but can support) security assurance
  – Are often atomic and easily verified
Security Requirements, cont.

• Avoidance of negative
  – E.g., hackers must not be able to input long strings of data and result in a buffer overflow
  – Standard lists, e.g., OWASP Top Ten Web Vulnerabilities, may be useful
  – Effective teams identify security problems which are likely to affect code, or have affected it in the past, and call them out specifically as things to be avoided

• Avoiding a negative is challenging because it is not atomic – may span many (or all) other features being developed by a team
  – May need to be kept in mind by developers when implementing other features
  – May require separate activity (e.g., sprint) to find and eliminate negatives before done
Implementation

• Individual developer security knowledge is particularly important in agile
  – Individual developers’ knowledge of secure coding patterns, and ability to think like a hacker, is the first line of defense
  – Having security experts (e.g., SPOCs) embedded in each development team is ideal but not always practical

• Automated tools are “force multipliers” for security and should be included in development process, not just at the end
  – Static tools can be run incrementally (e.g., daily code check-ins)
  – Dynamic tools (black box scanners, fuzzers) should be run before code is considered done
Implementation, cont.

• Security-only development phases/sprints may be required, either in parallel or sequential to development phases/sprints
  – To verify that negative features/stories have been avoided
  – Run tools, review code, analyze design

• Secure development environment can reduce risk of malicious code introduction
  – Only authenticated, authorized users should modify/update code
  – Code must be protected against unauthorized modification
  – Code reviews can catch at least some problems (accidental or deliberate)
Implementation – Third Party Code

• Most development projects include some third party code

• Third party code should be reviewed and approved before it is included
  – Supported through life of product?
  – Any known vulnerabilities in the version to be used?

• If code is open source then source should be reviewed just as in-house code would be
  – Code review
  – Static security analysis

• If open or closed source, human testing and dynamic security tools should be run as if code were developed in-house
Automated Security Tools

• Tools can automate many functions performed by security experts
  – Static analysis automates code review
  – Dynamic analysis automates penetration testing

• Security tools don’t replace security experts
  – Can often find coding errors
  – May detect design flaws (if manifested by observable behavior)
  – Usually don’t detect complex architectural problems
  – Can’t judge developer intent – may not find deliberately introduced bugs

• Tools should integrated into agile development process
  – As part of each sprint – e.g., static tool runs on nightly check-ins
  – As final step before done – e.g., static and dynamic runs on overall code
Automated Tools, cont.

• Selection, customization, and proper use of tools is required
  – Not all tools work equally well in all languages and environments
  – Particularly for static tools, customization is essential to avoid false positives
  – Training users in running tools and interpreting results is critical

• Oracle requires use of a general static analysis tool and at least one dynamic tool
  – Fortify and Parfait are standard static tools
  – A number of dynamic tools are recommended
  – More than 80 tools have been approved and are recommended for use
Security Criteria and Definition of Done

• At Oracle, SPOCs in development teams must complete a Release Security Checklist for every product (and major product component)

• Standard OSSA release criteria must be met, e.g.,
  – Automated tools use
  – Hardened configuration install defined
  – User security guide complete

• Product-specific technical criteria must be met
  – Customized by security lead based on knowledge of the product, customer expectations, past experience with security bugs, etc.

• Reviewed and approved by Security Lead before release
Challenges with Agile – Security Expertise

- Agile development security relies on having security expertise within the development team.
- Security expertise is not widespread among developers:
  - Most academic software programs still do not address vulnerabilities, how they occur, and how they can be avoided.
  - Thinking like a hacker does not come naturally to many people.
- Including security experts on agile development teams can be challenging:
  - Developer training becomes more important.
  - Recruiting and maintaining security experts is also important.
Challenges with Agile – Architectural Analysis

• Oracle has introduced Architectural Risk Analysis (ARA) as a structured methodology for analysis of potential vulnerability and risk based on
  – System architecture and design of product
  – Expected operational environment of product in deployment

• ARA works best when a well-developed architecture and design has been developed, and the operational environment is understood

• Fitting ARA into a development sprint has so far been difficult unless the sprint is enhancing an existing product with well-understood architecture and design.
  – ARA is still in the early stages of deployment within Oracle
  – Agile experience with ARA in Oracle is still limited
1 Oracle and Agile
2 What is Software Security Assurance?
3 Oracle Software Security Assurance
4 Integrating Security into Agile
5 Conclusion
Conclusion

• Secure development lifecycle activities can be effective in both waterfall and agile development methodologies

• A culture of security is important
  – Security must be a priority
  – Organizations must be prepared to develop securely through standards, training, tools, processes
  – Security leaders who understand and can adapt security assurance activities to the business are more important than mechanical implementation of security activities
Hardware and Software
Engineered to Work Together