

Secure By Design: Security in the Software Development Lifecycle



Twin Cities Rational User's Group
Security Briefing by Arctec Group
(www.arctecgroup.net)

ARCTEC

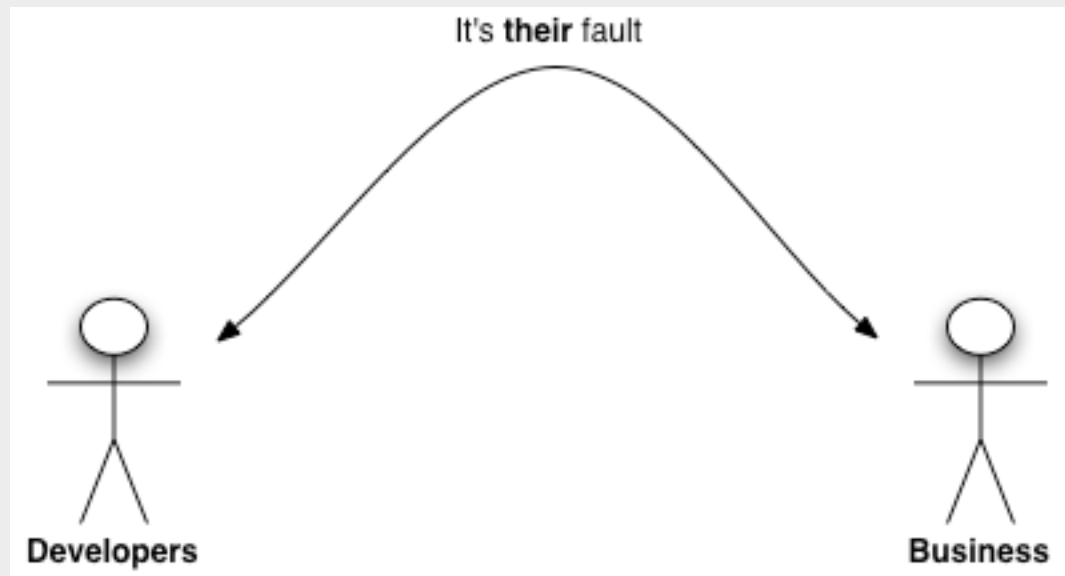
©2005 Arctec Group

Integrating Security into Software Development

- Focus Areas
 - Collaborative Approach
 - Security Goals
 - Leveraging Existing Artifacts and Processes
 - Security-centric artifacts

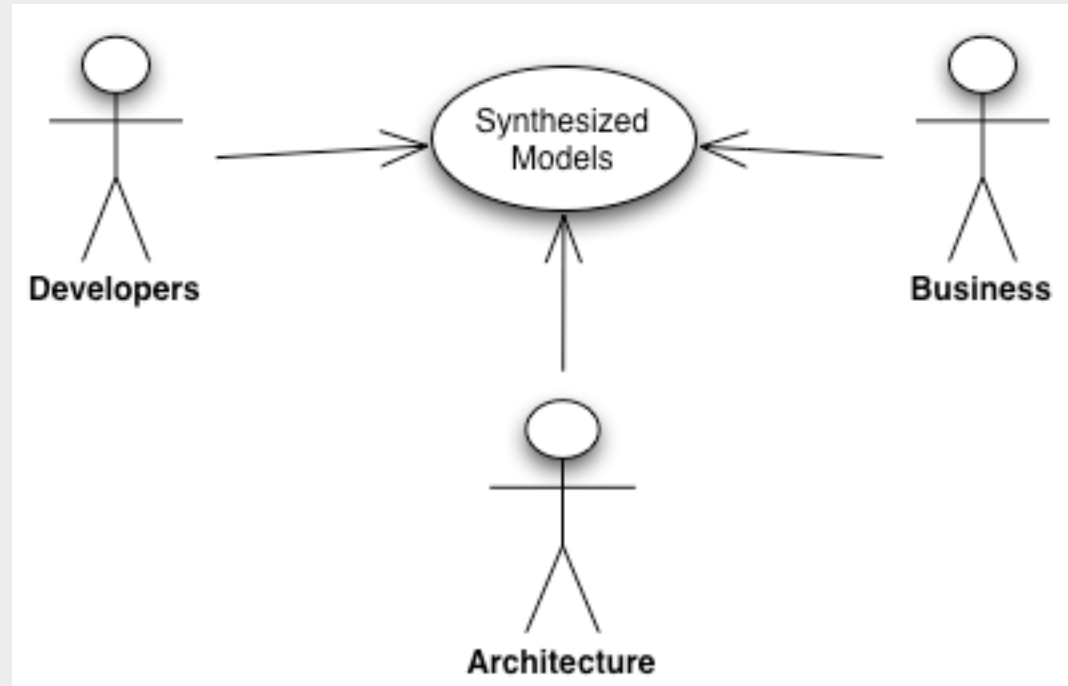
Software Development Realities

- Blame game: why is software insecure?



Software Development: The Way Forward

- Build a shared understanding of risk & risk management



Security Goals

- Confidentiality
 - “Concealment Information and resources”
- Integrity
 - “Trustworthiness of data and resources”
- Availability
 - “Availability of information or resources”
 - *definitions from *Computer Security Art and Science*, By Matt Bishop, Addison Wesley

Security & Security Mechanisms

- Protection - prevent the attack from succeeding
- Detection - detect abuse or malicious use
- Response - recover from attack
- Understand role of policy
- Understand trust assumptions

...but why do we need application security?

- Firewall/DMZ security model does not reflect current threat model or deal with threats emerging from web application, thin client, and web services application
- Current paradigm of front of front door protection does not protect against malicious insider
- 93% of vulnerabilities occur at the application level (source: ICAT)

...but why do we need application security?

	Highest Reported (\$US)	Average Losses (\$US)
Denial of Service	60,000,000	1,427,000
Theft of Proprietary Information	35,000,000	2,700,000
Insider abuse	6,000,000	135,000
Viruses and Worms	6,000,000	200,000
Financial fraud	4,000,000	329,000
Sabotage	2,000,000	215,000

* Excerpted from CSI/FBI Report 2003 - poll of 530 US based corporations, government, and educational institutions

Security in Inception

"A problem, properly stated, is a problem on its way to being solved," Buckminster Fuller

Security in Inception

- Requirements
 - Functional versus non-functional requirements
 - Usability & Security & related ilities
- Risk Analysis
 - Asset classification
- Data Classification
 - Classify entities to drive out security model for access control and related controls
- Domain Glossary of Security Terms

Data Classification Example

	Public	Private	Confdl	Site Config
Everyone	R			
Current Customers	R	C, R, U	R	
Administrators				R, U, D

Security in Inception

- Security Use Case Modeling
 - Show behavioral flows
 - Understand/analyze/vet security implications of pre and post conditions
 - Understand/analyze/vet exceptional flows
 - Useful to synthesize authorization structure

Security Use Case Modeling

- Modeling Identity
 - Identity forms the basis of many security mechanisms, including access control (authentication, authorization)
 - However identity is not an entity, but the result of a process
 - Use Use Case modeling to drive out the relationships and dependencies to establish identity in a system
 - SSO Federated Identity example: User logs on to local system in manufacturer policy domain, message is sent to supplier domain, message includes client user information & manufacturer domain security information; supplier system authenticates request

Security in Elaboration

- Elaborating Use Cases
 - Using relationships to model security
 - Includes relationships can illustrate many protection mechanisms, e.g. logon process
 - Extends relationships can illustrate many detection mechanisms, e.g. audit logging

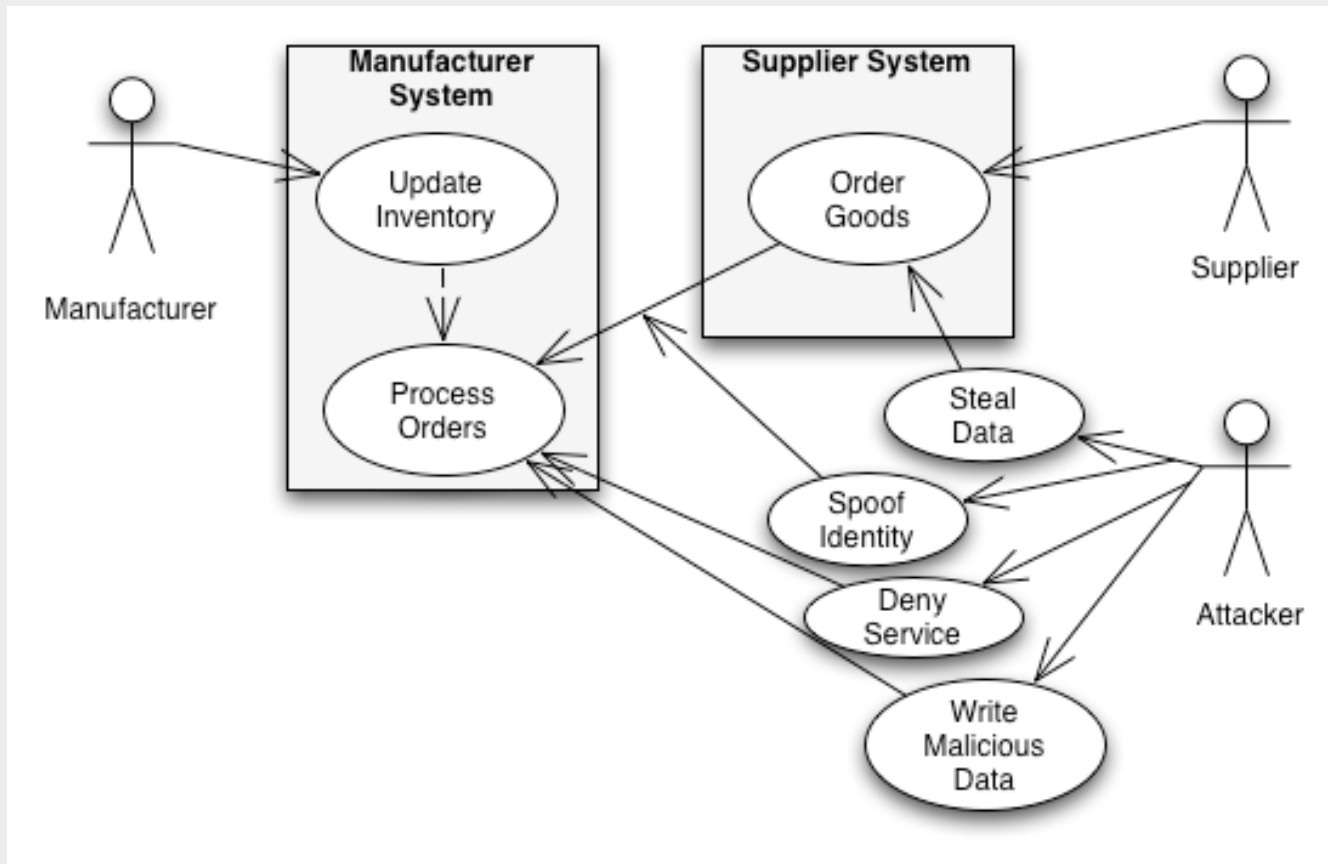
Security in Elaboration

- Abuse Cases
 - Look at the system from an attacker point of view
 - Useful to glean/verify security requirements, create threat models
 - Discussed in detail paper by Guttorm Sindre and Andreas Opdahl.
 - More information at:
www.ifi.uib.no/conf/refsq2001/papers/p25.pdf

Misuse Case Format

- “A *misuse case* is the inverse of a use case, i.e. A function that the system should not allow” -Sindre & Opdahl
- “A *mis-actor* is the inverse of an actor, i.e., an actor that one does not want the system to support, an actor who initiates misuse cases.” - Sindre & Opdahl
- Additional elements
 - Worst Case Threat: end system state if Misuse succeeds
 - Prevention and Detection Guarantees: these guarantees closely resemble a Use Case Post-condition, but encapsulate security-specific concepts of prevention and detection.
 - Stakeholders and Risks: this field gives the security team a place to address what the business risk that is generated by the application.

Misuse Case Example



Security in Elaboration

- Implement Saltzer and Schroeder's Principles*
 - Principle of Least Privilege
 - Principle of Fail Safe Defaults
 - Principle of Economy of Mechanism
 - Principle of Complete Mediation
 - Principle of Open Design
 - Principle of Separation of Privilege
 - Principle of Least Common Mechanism
 - Principle of Psychological Acceptability
- * from "The Protection of Information in Computer Systems", Saltzer and Schroeder, *Proceedings of the IEEE*, 1975

Security in Elaboration

- Threat Modeling
 - Howard and Leblanc's STRIDE and DREAD
 - Identification
 - STRIDE
 - **S**poofing, **T**ampering, **R**epudiation, **I**nformation disclosure, **D**enial of service, and **E**levation of privilege
 - Prioritization
 - DREAD
 - **D**amage potential, **R**eproducibility, **E**xploitability, **A**ffected users, and **D**iscoverability

Using Threat Models

- Threat Models consist of:
 - Entry points
 - Assets
 - Threats
 - Dependencies

Using Threat Models

- Order Books Web Service example Use Case
 - Entry point: web service is available via network, e.g. firewall does not block access
 - Assets: customer number, credit card, bookstore brand and trust
 - Threats
 - Information disclosure: attacker could gain access to Web Service
 - Denial of Service: attacker could effect accessibility of site
 - Tampering & Elevation of privilege: attacker could use vulnerability in service tamper with data or elevate privilege

Security in Construction

- Test cases
 - Use threat models and abuse cases to drive test cases - the same as use cases drive testing for functional requirements
- Logging & monitoring
 - Assess quality of logging and reporting mechanisms during development
- Code Review
 - Peer review
 - Source Code Analysis Tools
 - Scanning tools

Security in Transition

- Security baseline configuration
 - Utilize baseline configurations from vendors and best practice guides
- Secure builds - separation of privileges
- Incident response planning
- Security Metrics

The Way Forward

- Security in SDLC is still relatively new
- Aim for incremental improvement
- Aim for reuse
- Educate/mentor/co-evolve

- Manage risk:

“We have no future because our present is too volatile. We have only risk management. The spinning of the given moment's scenarios. Pattern recognition...”

-William Gibson “Pattern Recognition”