Introduction

- National Institutes of Informatics
  National Research Center for Information Science
  - 80 Research staffs and about 100 PhD students

- Research Interests
  - Software Engineering on Pervasive Computing
    - Middleware, Design, Modeling
  - Security Patterns, Method using Security Patterns
A Survey of Security Patterns and Vulnerability Analysis using Attack Patterns

Nobukazu Yoshioka†, Hironori Washizaki†,††, Katsuhisa Maruyama‡

†National Institute of Informatics, ††The Graduate University for Advanced Studies, ‡Ritsumeikan University
Backgrounds

- Importance of Security
- Difficulty of Development of Secure Systems
  - We should consider many kinds of concerns and situations
- Many Security Patterns have been proposed
  - It is still difficult to use …
  - Classification them and propose a new pattern
Classification of Pattern

- Development Process
  - Security Requirements and Analysis
  - Security Design
  - Security Implementations

Discuss Efficiency of Patterns from the security concern view point

Propose a new pattern
Security Concerns


- **Asset**: Information or resources that have value to an organization or person.
- **Stakeholder**: An organization or person who places a particular value on assets.
- **Security objective**: A statement of intent to counter threats and satisfy identified security needs.
- **Threat**: A potential for a security breach of an asset.
- **Attack**: An action that violates the security an asset.
- **Attacker**: The entity which carries out attacks.
- **Vulnerability**: A flaw or weakness that could be exploited to breach the security of an asset.
- **Countermeasure**: An action taken in order to protect an asset against threats and attacks.
- **Risk**: The probability that a successful attack occurs.

Discuss Efficiency of Patterns based on these concerns
Classification of Pattern

- Development Process
  - Security Requirements and Analysis
  - Security Design
  - Security Implementations

→ Discuss Efficiency of Patterns from the security concern view point

→ Propose a new pattern
Patterns for Security Requirements and Analysis

There are Analysis Process Patterns.

- **Determine WHAT**, assets, we need to protect.
  - Security Needs Identification for Enterprise Assets[SecPat06]

- **Determine security needs** and **HOW FAR** protect assets for the requirements?
  - **Security needs**:
    - Security Types: Confidentiality, Integrity, Availability, Accountability
      - Security Needs Identification for Enterprise Assets[SecPat06]
  - **HOW FAR** protect assets?: We need take priority.
    - Asset Valuation Pattern[SecPat06]
    - Threat Assessment Pattern[SecPat06]
    - Vulnerability Assessment Pattern[SecPat06]
Example of Patterns: Security Needs Identification

Security Needs Identification for Enterprise Assets [SecPat06]

- Identify Enterprise assets
- Identify Business drivers
- Determine business drivers-asset relation
- Identify Security properties

Security needs solution sequence

Common information asset categories

<table>
<thead>
<tr>
<th>Asset Type</th>
<th>Security Needs</th>
<th>Business Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer and business partner data</td>
<td>Confidentiality, Integrity, Accountability</td>
<td>- Competitive issues</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Service issues if a public company</td>
</tr>
</tbody>
</table>
## Efficiency of Security Patterns on Requirements phase

<table>
<thead>
<tr>
<th>Concept</th>
<th>Requirements and Analysis Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Countermeasure</td>
<td>Identified</td>
</tr>
<tr>
<td>Risk</td>
<td>Identified</td>
</tr>
<tr>
<td>Threat</td>
<td>Identified</td>
</tr>
<tr>
<td>Attack</td>
<td>Identified ++</td>
</tr>
<tr>
<td>Attacker</td>
<td>Identified ++</td>
</tr>
<tr>
<td>Vulnerability</td>
<td>Identified</td>
</tr>
<tr>
<td>Asset</td>
<td>Defined</td>
</tr>
<tr>
<td>Stakeholder</td>
<td>Defined</td>
</tr>
<tr>
<td>Security objective</td>
<td>Defined</td>
</tr>
</tbody>
</table>

Almost all concerns are mentioned.
Classification of Pattern

- Development Process
  - Security Requirements and Analysis
  - Security Design
  - Security Implementations

Discuss Efficiency of Patterns from the security concern view point

Propose a new pattern
Design Patterns for Security Functions

- Determine HOW TO protect assets
  - Security type
    - Confidentiality
    - Integrity
    - Availability
    - Accountability
  - Many Design Patterns Access control, Confidentiality
    - Role based Access control Pattern
    - Single Access Point, Check point patterns
  - Patterns for Availability
    - Firewall Pattern: IP level, Transportation Level, Service Level
  - Security Patterns on OS Level for Web applications
Security Design Pattern Examples

Role based Access Control Pattern [Fernandez01]

Secure Logger Pattern with Secure Log Store Strategy [CoreSecurityPatterns05]
## Efficiency of Security Patterns on Design phase

<table>
<thead>
<tr>
<th>Phase</th>
<th>Design Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td></td>
</tr>
<tr>
<td>Countermeasure</td>
<td>Feasibility</td>
</tr>
<tr>
<td>Risk</td>
<td>Estimated</td>
</tr>
<tr>
<td>Threat</td>
<td>Feasibility</td>
</tr>
<tr>
<td>Attack</td>
<td>Feasibility</td>
</tr>
<tr>
<td>Attacker</td>
<td>Feasibility</td>
</tr>
<tr>
<td>Vulnerability</td>
<td>Feasibility</td>
</tr>
<tr>
<td>Asset</td>
<td>Designed with security</td>
</tr>
<tr>
<td>Stakeholder</td>
<td>Reviews</td>
</tr>
<tr>
<td>Security objective</td>
<td>Reviewed</td>
</tr>
</tbody>
</table>

Mainly focused on Security Countermeasure pattern
Classification of Pattern

- Development Process
  - Security Requirements and Analysis
  - Security Design
  - Security Implementations

Discuss Efficiency of Patterns from the security concern viewpoint

Propose a new pattern
Patterns for Security Implementation

【Implementation Phase】

■ Secure Programming
  ■ Guidelines for secure program to avoid security flaws
    ➔ Input validation, buffer overflow, etc.

■ Secure Refactoring
  ➔ Remove security vulnerability
    ◆ Change public field to private one
    ◆ Remove setting method and declare final
    ◆ Hiding classes which do not need to be publicly visible

■ Attack Patterns
  ■ How to break software: ⇒ useful for improving the implementation
    ➔ Attack Patterns for Web application, MediaPlayer, Web Browser, etc.
# Efficiency of Security Patterns on Implementation phase

<table>
<thead>
<tr>
<th>Concept</th>
<th>Implementation Phase</th>
</tr>
</thead>
</table>
| Countermeasure       | Implemented                           ++
| Risk                 | Measured                              |
| Threat               | Realized                              +
| Attack               | Tested                                ++
| Attacker             | Tested                                |
| Vulnerability        | Realize                               ++
| Asset                | Implemented with security             +
| Stakeholder          | Tests                                 |
| Security objective   | Reviewed                              |

Mainly focused on attack and vulnerability concerns
Classification of Pattern

- Development Process
  - Security Requirements and Analysis
  - Security Design
  - Security Implementations

- Discuss Efficiency of Patterns from the security concern view point

- Propose a new pattern
## Discussion: Efficiency of Security Patterns

<table>
<thead>
<tr>
<th>Concept</th>
<th>Phase</th>
<th>Requirements Phase</th>
<th>Design Phase</th>
<th>Implementation Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Countermeasure</td>
<td>Feasibility</td>
<td>+ Feasibility</td>
<td>++ Implemented</td>
<td>++</td>
</tr>
<tr>
<td>Risk</td>
<td>Estimated</td>
<td>+ Estimated</td>
<td>Measured</td>
<td></td>
</tr>
<tr>
<td>Threat</td>
<td>Feasibility</td>
<td>+ Feasibility</td>
<td>Realized</td>
<td>+</td>
</tr>
<tr>
<td>Attack</td>
<td>Feasibility</td>
<td>++ Feasibility</td>
<td>Tested</td>
<td>++</td>
</tr>
<tr>
<td>Attacker</td>
<td>Feasibility</td>
<td>++ Feasibility</td>
<td>Tested</td>
<td></td>
</tr>
<tr>
<td>Vulnerability</td>
<td>Feasibility</td>
<td>+ Feasibility</td>
<td>Realize</td>
<td>++</td>
</tr>
<tr>
<td>Asset</td>
<td>Designed with security</td>
<td>+ Designed with security</td>
<td>Implemented with security</td>
<td>+</td>
</tr>
<tr>
<td>Stakeholder</td>
<td>Reviews</td>
<td>+ Reviews</td>
<td>Tests</td>
<td></td>
</tr>
<tr>
<td>Security objective</td>
<td>Reviewed</td>
<td>+ Reviewed</td>
<td>Reviewed</td>
<td></td>
</tr>
</tbody>
</table>

**Specification**: Overleaped Area
## Discussion: Efficiency of Security Patterns

<table>
<thead>
<tr>
<th>Concept</th>
<th>Phase</th>
<th>Requirements Phase</th>
<th>Design Phase</th>
<th>Implementation Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Countermeasure</td>
<td>Design Phase</td>
<td>Feasibility</td>
<td>Feasibility</td>
<td>Implemented</td>
</tr>
<tr>
<td>Risk</td>
<td>Design Phase</td>
<td>Estimated</td>
<td>Estimated</td>
<td>Measured</td>
</tr>
<tr>
<td>Threat</td>
<td>Design Phase</td>
<td>Feasibility</td>
<td>Feasibility</td>
<td>Realized</td>
</tr>
<tr>
<td>Attack</td>
<td>Design Phase</td>
<td>Feasibility</td>
<td>Tested</td>
<td>++</td>
</tr>
<tr>
<td>Attacker</td>
<td>Design Phase</td>
<td>Feasibility</td>
<td>Tested</td>
<td>++</td>
</tr>
<tr>
<td>Vulnerability</td>
<td>Design Phase</td>
<td>Feasibility</td>
<td>Realize</td>
<td>++</td>
</tr>
<tr>
<td>Asset</td>
<td>Design Phase</td>
<td>Designed with security</td>
<td>Designed with security</td>
<td>Implemented with security</td>
</tr>
<tr>
<td>Stakeholder</td>
<td>Design Phase</td>
<td>Reviews</td>
<td>Reviews</td>
<td>Tests</td>
</tr>
<tr>
<td>Security objective</td>
<td>Design Phase</td>
<td>Reviewed</td>
<td>Reviewed</td>
<td>Reviewed</td>
</tr>
</tbody>
</table>

**Lacked Relation**

**Lacked Area**
Classification of Pattern

- Development Process
  - Security Requirements and Analysis
  - Security Design
  - Security Implementations

- Discuss Efficiency of Patterns from the security concern view point

- Propose a new pattern
Security Requirements and Design

【Security Requirements】
- WHAT we protect?
- Security Needs
  - Confidentiality
  - Integrity
  - Availability
  - Accountability
- HOW FAR protect assets?:
  We need take priority.
  - Threat Analysis
  - Vulnerability Analysis

【Security Design】
- WHERE we need to protect?
  - Which Object? Classes?
  - Messages? Protocols?
- HOW TO protect?
  - Access Control
  - Authentication
  - Encryption
  - Signature
  - Logging, etc
Gap between Security Requirements and Design

【Security Requirements】
- WHAT we protect?
- Security Needs
  - Confidentiality
  - Integrity
  - Availability
  - Accountability
- HOW FAR protect assets?: We need take priority.
  - Threat Analysis
  - Vulnerability Analysis

【Security Design】
- WHERE we need to protect?
  - Which Object? Classes?
  - Messages? Protocols?
- HOW TO protect?
  - Access Control
  - Authentication
  - Encryption
  - Signature
  - Logging, etc

- Which Security Level is proper?
- • What is detailed threat?
  - • Which parts are vulnerability?
A New Development Method

【Security Requirement】

WHAT we protect?

Security Needs
- Confidentiality
- Integrity
- Availability
- Accountability
- HOW FAR protect assets?:
  - We need take priority.

【Security Design】

WHERE we need to protect?
- Which Object? Classes?
- Messages? Protocols?

HOW TO protect?
- Access Control
- Authentication
- Encryption
- Signature
- Logging, etc

Attack Design

HOw TO attacked?

WHERE is vulnerability?

Threat Analysis
Vulnerability Analysis
Overview of Our Development Method

Refine security concerns step by step and design security functions


Assets, Confidentiality → Security Requirements

Misuse case → Attack Requirements

Security Design → Vulnerability, Assumptions

Security function → Design

Process of Attacks
Security Requirements

• use case including
  • Assets definition with <<asset>>
  • Confidentiality definition with <<permit>>

Suppose Role based Access Control
Attacker’s Requirements

Specify attacks against assets with **misuse case**

- **Attacker**
  - C-II member
  - Others

- **View individual cost**
- **View Product Information**
- **Change the total cost**

Scenario of attacks from the asset view points
Security Design

- Reification of assets
- Specify Vulnerability with <<insecure>>

Vulnerable Part: this communication might be attacked potentially
⇒ Need Security!

How to be harmed?
Design of an attack against vulnerable parts

- confirmation and find of vulnerability

An attack on an asset using insecure network and host

Host A: machine
Permit=Set{sec1}

Host C: machine
Permit=Set{C-II}

Host C: machine
Insecure

Malicious Requester

Host A: machine
Requester

Host C: machine
Sim

Requester

DB

Sim

Malicious Requester

- query

Product info

<<asset>>

dividual costs

<<insecure>>

<<asset>>

dividual costs

<<insecure>>

<<insecure>>
Security Modeling based on Attack Patterns

- Difficulty of Covering all vulnerability
  - We need to check combinations of insecure parts in data flow and deployments

- Difficulty of Consistency Check between vulnerability, attacks and counter-measures

  **Attack Patterns abstracting attacks**
  - We can cover all by applying patterns
  - Clearly relation between vulnerability, attacks and counter-measures
Definition of Attack Patterns

- **Application Context**
  - Specify environments and context to be attacked
  - Asset and vulnerability are specified

- **Method of Attack: Problem**
  - Procedure of attacks

- **Solution**
  - Counter-measures with security functions
An Example: Wiretapping Pattern (1/2)

Application Context: Specifies environments and context to be attacked

- **Client**: Requester
- **ClientM**: Malicious Requester
- **Server**: Service

**Static Aspects with Deployment Diagrams**

---

**Dynamic Aspects with Sequence Diagrams**

Definition of insecure: members who don’t have a permission to access the asset can access to client host

Precondition: a malicious requester can be created in an insecure host
An Example: Wiretapping Pattern (2/2)

Method of Attack specifies attack sequence against systems

Solution includes security functionalities

Diagram:
- Requester → Service → Requester
- Malicious Requester
- Security functionalities include encryption.
Modeling Support with Attack Patterns

Attack Patterns bridge the gap between models

- Application Context
- Method of Attack
- Solution

Design  Security Design  Attack Design

Security function

Vulnerability, Assumptions

Process of Attacks
Application of Pattern (1/3)

Check of insecure parts

Check definition of insecure in patterns
Application of Pattern (2/3)

Check structure of application context using deployment diagrams

Deployment Diagram of a target system

Application Context in Pattern

Check correspondence of insecure and components between system model and pattern
Application of Pattern (3/3)

Check data flows in application context using sequence diagrams

Sequence Diagram of a target system

Application Context in Pattern

Check correspondence of asset between models
Conclusion

- Categorize Security Patterns from the development process view point
  - Discussion: Efficiency of Security Patterns
  - Need Attack Design and Relation between requirements and Implementation
- Propose: A New Security Design Method
  - Stepwise development including Attack Design
  - Attack Patterns support Security Modeling
    - Support vulnerability analysis
    - Relation between vulnerability, attacks and counter-measures
Future works

- Definition of Models: Syntax and Semantics
- Evaluation based on Example
- Provide Many Patterns
- Methodologies using Security Patterns
- Tool Support
  - Auto-detection of
    - vulnerability, insecure parts
    - Application Context
  - Semi-auto instantiation of attacks and counter-measures