Metamodel and Patterns for Cloud Security and Privacy

Hironori Washizaki
Professor at Waseda University, Tokyo, Japan

washizaki@waseda.jp
Prof. Dr. Hironori Washizaki

• Professor and the Associate Dean of the Research Promotion Division at Waseda University in Tokyo
• Visiting Professor at the National Institute of Informatics
• Outside Directors of SYSTEM INFORMATION and eXmotion
• Research and education projects
  • Leading a large-scale grant at MEXT enPiT-Pro Smart SE
  • Leading framework team of JST MIRAI eAI project
• Professional contributions
  • IARIA Fellow
  • IEEE Computer Society Vice President for Professional and Educational Activities
  • Editorial Board Member of MDPI Education Sciences
  • Steering Committee Member of the IEEE Conference on Software Engineering Education and Training (CSEE&T)
  • Associate Editor of IEEE Transactions on Emerging Topics in Computing
  • Advisory Committee Member of the IEEE-CS COMPSAC
  • Steering Committee Member of Asia-Pacific Software Engineering Conference (APSEC)
  • Convener of ISO/IEC/JTC1 SC7/WG20
Metamodel and Patterns for Cloud Security and Privacy

Hironori Washizaki
Professor at Waseda University, Tokyo, Japan

washizaki@waseda.jp  http://www.washi.cs.waseda.ac.jp/

Agenda

• Paradigm shifts in new software engineering
• Pattern language
• Security patterns
• Metamodel and Patterns for Cloud Security and Privacy
What is software engineering?

• “Application of systematic, disciplined, quantifiable approach to development, operation, and maintenance of software” – SWEBOK 2014

• Guide to the Software Engineering Body of Knowledge (SWEBOK)

- Software Requirements
- Software Design
- Software Construction
- Software Testing

- Software Maintenance
- Software Configuration Management
- Software Engineering Management
- Software Engineering Process

- Software Engineering Tools and Methods
- Software Quality
- Software Engineering Professional Practice
- Software Engineering Economics

- Computing Foundations
- Mathematical Foundations
- Engineering Foundations
Vision of SWEBOK 2022 (subject to change)

(Evolution lead: Hironori Washizaki, since 2018-)

https://www.computer.org/volunteering/boards-and-committees/professional-educational-activities/software-engineering-committee/swebok-evolution

- Expansion of SE
  - AI/Machine Learning Engineering
  - Restructuring foundation areas incl. Internet of Things (IoT)
- Value in SE
  - Value proposition
- Dependable SE
  - Architecture
  - Security
- Modern SE
  - Agile
  - DevOps

Software Engineering

Engineering Foundation incl. IoT

Value proposition

Expansion

Value

AI/ML

Modern

Agile & DevOps

Dependable

Architecture
Paradigm shifts in “new” software engineering

<table>
<thead>
<tr>
<th></th>
<th>Current</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope and perspective</td>
<td>Software systems</td>
<td>Software systems, <strong>business,</strong> society and related disciplines</td>
</tr>
<tr>
<td>Process</td>
<td>Planned, static, common, and closed</td>
<td><strong>Adaptive,</strong> dynamic, diverse, and open</td>
</tr>
<tr>
<td>Focus</td>
<td>Specification</td>
<td><strong>Value,</strong> data, and speed</td>
</tr>
<tr>
<td>Thinking</td>
<td>Cognitive (logical) or affective (design)</td>
<td>Cognitive (logical), affective (design), and <strong>conative</strong> (conceptual)</td>
</tr>
<tr>
<td>Inference</td>
<td>Deduction and analogy</td>
<td>Deduction, analogy, <strong>induction,</strong> and abduction</td>
</tr>
</tbody>
</table>

Problem and goal

- Cloud computing is one of the key enablers of digital transformations.
- Security must be a critical cross-cutting concern in cloud and any other software.
- We are conducting systematic literature reviews to reveal landscapes of security patterns and cloud security & privacy.
Agenda

• Paradigm shifts in new software engineering
• Pattern language
• Security patterns
• Metamodel and Patterns for Cloud Security and Privacy
**Problem:** Needs to have a place where people can sit lazily, legitimately, be on view, and watch the world go by…

**Solution:** Encourage local cafes to spring up in each neighborhood. Make them intimate places, with several rooms, open to a busy path …


[Image](https://unsplash.com/photos/8IKf54pc3qk)  [Image](https://unsplash.com/photos/zACLEreWKXE)
Towards a pattern language

... OK, so, to attract many people to our city, **Small Public Square** should be located in the center. At the **Small Public Square**, make **Street Cafes** be **Opening to the Street** ...
New SE needs pattern (language)!

- **Bridge** between abstract paradigms and concrete cases/tools
  - Verbalizing and documenting Know-Why (context), What (problem) and How (solution)
  - Reusing solutions and problems
  - Getting consistent architecture

- **Common language** among stakeholders
  - Security engineers, software engineers, hardware engineers, network engineers, domain experts, data analystist, …
Agenda

• Paradigm shifts in new software engineering
• Pattern language
• Security patterns
• Metamodel and Patterns for Cloud Security and Privacy
Security concerns must be addressed at any phase

- Patterns are **recurrent problems and solutions** under specific **contexts** from requirements to maintenance.
Example of security pattern

- **Name:** *Role-based access control (RBAC)*
- **Problem:** How do we assign rights to people based on their functions or tasks?
- **Solution:** Assign users to roles and give rights to these roles so they can perform their tasks.
- **Related patterns:** *Authorization, . . .*

![Diagram of security pattern]
Systematic Literature Review of Security Pattern Research

• We categorize and analyze 240 papers to clarify state-of-the-art and future directions of security pattern research in terms of 13 facets including topics and security characteristics.

• E.g., breakdown of research topics

Conclusion and future work

Current

• Targeting authentication and authorization
• Many researches using UML, but independent
• Often simple case studies
• Targeting existing patterns only
• Limited education for secure development methods in IoT era

Future

• Address various security patterns
• Integration based on common metamodel
• Complex case studies with measurements
• New vulnerabilities and patterns
• Cloud, IoT and security education program
Model-driven security pattern application

[PLoP’10]

① Selecting a pattern
② Setting parameters
Input model
③ Transformation by ATL

Parameter

Transformation rules

UML models

Helper def : SubjectName : String = ‘hoge’

rule SubjectClass {
  from s : UML!Class(s.isSbj())
  to t : UML!Class(
    namespace <- s.namespace,
    isAbstract <- false,
    ...
  )
}

Example: application of “Authorization”

Helper def: ProtObName: String = ‘Patient’

Authorization

<<Authenticator.Subject>>
<<Authorization.Subject>>
TESEM: Test Driven Secure Modeling Tool

[ARES’13][ARES’13][IJSSE’14][ICST’15][Information’16]

Security Design Pattern

Context
Problem
Solution

Test design as requirement

Test Script

! create Actor
! create UI :
! create Subject..

Test case


[ARES’14] Verification of Implementing Security Design Patterns Using a Test Template, Conf. Availability, Reliability and Security


[Information’16] Implementation Support of Security Design Patterns Using Test Templates, Information 7(2)
Test-driven secure design

- Security Properties are in testcases

- Add test cases
- Confirm tests fail
- Fix model
- Eval. of mitigation
- Confirm tests pass
- Find vulnerability
Add test cases

Verify whether model with RBAC satisfies security design requirements

<table>
<thead>
<tr>
<th>Conditions</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rights are given in “Role” which an “User” belongs</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>consider that “Actor” have access permission.</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>consider that “Actor” does not have access permission.</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>execute “delete” function</td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>can not execute “delete” function</td>
<td>✗</td>
<td></td>
</tr>
</tbody>
</table>

context subject_controller
inv access_control:
if self.RBAC.Right->exists(p |
  p.right = true and
  p.role_id = p.Role.id and
then
  self.DeleteUI.Actor.right = true and self.subject_function = true
else
  self.DeleteUI.Actor.right = false and self.subject_function = false
endif

Verify whether model satisfies security design requirement
Model does not satisfy **security design requirements.**

TESEM detected incorrect applications of design patterns.
Fix model and confirm tests pass

Fix design model until the tests successfully pass.

Incorrect design  ➔  Refactoring  ➔  Correct design
Agenda

• Paradigm shifts in new software engineering
• Pattern language
• Security patterns
• Metamodel and Patterns for Cloud Security and Privacy
Challenges in cloud security and privacy (S&P)

- How to consistently utilize diverse S&P knowledge?
  ⇒ Metamodel

- How to consider S&P over different layers?
  ⇒ Layered metamodel

Patterns

Guidelines

Practices

Cloud services

Ex.) User Authentication

Software Application

User Authorization

Platform

Secure Config.

OS Hardening

Infrastructure

Electronic Access Control system
Modeling vulnerability and security pattern

**Common Vulnerabilities and Exposures: CVE-2012-4394** Cross-site scripting (XSS) vulnerability in apps/files/js/filelist.js in own Cloud before 4.0.5 allows remote attackers to inject arbitrary web script or HTML via the file parameter.

Validator for data-injection vulnerability such as XSS
Security and privacy development process
Security requirements analysis

- Threats and vulnerability analysis based on STRIDE
- Consider corresponding security patterns (e.g., Authentication and Authorization)

<table>
<thead>
<tr>
<th>Goal</th>
<th>Anti-goal</th>
<th>Problem</th>
<th>Example</th>
<th>Pattern</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tamper proof data</td>
<td>Gain ability to tamper with data</td>
<td>unauthorized actors tampering with local data</td>
<td>users accessing local data on their phone, changing their score</td>
<td>Encryption pattern</td>
<td>Provided by the android phone itself: it encrypts stored data</td>
</tr>
<tr>
<td>confidentiality</td>
<td>Gain access to confidential info</td>
<td>unauthorized actors tampering with cloud data</td>
<td>hostile accessing the cloud server to change the goal location to current location</td>
<td>Encryption pattern</td>
<td>Handled by Amazon: their security measures are quite extensive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unauthorized actors listening to the transmissions to and from the server</td>
<td>man in the middle attack</td>
<td>Transmission pattern</td>
<td>API automatically uses SSL and can be set to use a VPN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>information disclosure</td>
<td>hostile user releases a list of goal locations</td>
<td>Encryption pattern, Authentication and (architectural solutions: firewall, server layout)</td>
<td>similar to tamper proof data: same solution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>elevation of privilege</td>
<td>a user pretends to be an administrator which gives him unlimited access to all game data</td>
<td>Authentication pattern, (limitation of access), transmission pattern</td>
<td>Player can only get access to the database through software, which is tightly controlled by API</td>
</tr>
</tbody>
</table>
Case study and result

• This table shows the result of a simple case study by assigning a vulnerable system model.
• The experiment group with CSPM perform better by solving more security problems.
Summary

• There are paradigm shifts in “new” software engineering.
  – Cloud computing is one of the key enablers of digital transformations.
  – Security must be a critical cross-cutting concern in cloud and any other software.

• New software engineering needs patterns and pattern languages.
  – Bridge between abstract paradigms and concrete cases/tools
  – Common language among stakeholders

• Security patterns
  – Systematic Literature Review of Security Pattern Research
  – Model-driven security pattern application
  – Test Driven Secure Modeling Tool

• Metamodel and Patterns for Cloud Security and Privacy
  – Cloud Security and Privacy Metamodel (CSPM)
  – Modeling vulnerability and security pattern
  – Security and privacy development process