Metamodel for Security and Privacy Knowledge in Cloud Service Development[1]

Tian Xia[1], Hironori Washizaki[2,*], Yoshiaki Fukazawa[3], Takehisa Kato[4], Haruhiko Kaiya[5], Shinpei Ogata[6], Eduardo B. Fernandez[7], Hideyuki Kanuka[8], Masayuki Yoshino[9], Dan Yamamoto[10], Takao Okubo[11], Nobukazu Yoshioka[12] and Atsuo Hazeyama[13]

[1] Waseda University, lmtc668800@moegi.waseda.jp
[2] Waseda University, washizaki@waseda.jp
[3] Waseda University, fukazawa@waseda.jp
[4] Toshiba Digital Solutions Corporation, takehisa.kato@toshiba.co.jp
[5] Kanagawa University, kaiya@kanagawa-u.ac.jp
[6] Shinshu University, ogata@cs.shinshu-u.ac.jp
[7] Florida Atlantic University, fernande@fau.edu
[8] Hitachi, Ltd., hideyuki.kanuka.dv@hitachi.com
[9] Hitachi, Ltd., masayuki.yoshino.aa@hitachi.com
[10] Hitachi, Ltd., dan.yamamoto.vx@hitachi.com
[11] Institute of Information Security, okubo@iisec.ac.jp
[12] National Institute of Informatics, nobukazu@nii.ac.jp
[13] Tokyo Gakugei University, hazeyama@u-gakugei.ac.jp
* Correspondence: washizaki@waseda.jp; Tel.: +81-3-5286-3272

Received: date; Accepted: date; Published: date

Abstract: Requirements for cloud services include security and privacy. Although many security patterns, privacy patterns, and non-pattern-based knowledge have been reported, knowing which pattern or combination of patterns to use in a specific scenario is challenging due to the sheer volume of options and the layered cloud stack. To deal with security and privacy in cloud services, this study proposes the Cloud Security and Privacy Metamodel (CSPM). CSPM uses a consistent approach to classify and support existing security and privacy patterns. In addition, CSPM is used to develop a security and privacy awareness process to develop cloud systems. The effectiveness and practicality of CSPM is demonstrated via several case studies.

Keywords: cloud computing; security patterns; privacy patterns; software and system architecture

1. Introduction

Security and privacy (S&P) in cloud systems are critical. Cloud services not only tend to be accessed remotely but are also connected to other services. Because software engineers are not necessarily experts in S&P, resolving S&P concerns throughout the software lifecycle is challenging.

In software, patterns are abstractions from recurring concrete problems and corresponding solutions that appear in non-arbitrary contexts. Besides the numerous cloud computing and services S&P patterns reported to date [1–4], non-pattern-based knowledge (e.g., practice and principles) is used to address S&P issues in cloud services.

The sheer volume of S&P patterns and documentation makes selecting the appropriate pattern or combination of patterns challenging. Although this issue is relevant to security patterns in general, it is more critical in cloud services. First, cloud services and their underlying mechanisms are integrated over multiple layers in a layered cloud stack. Second, a cloud computing system links

[1] This paper is an extended version of our previous conference paper presented at [15].

numerous devices, and each device has its own deployment model and service. This can create a highly complex system.

The above issues can be overcome via metamodels or reference architectures that capture the essential concepts related to S&P in layered cloud stacks. Previously, we reported the background and an earlier version of the metamodel [10]. This study proposes an extension called the "Cloud Security and Privacy Metamodel (CSPM)" to address S&P in cloud services. CSPM integrates and extends existing cloud security metamodels with newly added concepts.

CSPM can be used in cloud service development and maintenance (Fig. 1). CSPM describes S&P-related knowledge over multiple layers. Besides selecting and combining the appropriate patterns to address S&P issues, CSPM can be used to design high-level architectures of cloud service systems efficiently and effectively .

As an extension to our previous research, we conducted experiments and a case study to address the following questions:

RQ1. Can CSPM resolve S&P problems and help application of the corresponding patterns?
RQ2. Can CSPM improve the system by efficiently providing S&P solutions?
RQ3. Can CSPM and processes using CSPM be deployed in practical real-world applications?

RQs 1 and 2 evaluate CSPM from two viewpoints. RQ3 demonstrates the usability of our approach for the metamodel itself and the process we propose. The case study, which involves an application similar to a commercial one using a conventional cloud platform, suggests that CSPM has practical applications in industrial development. Tools such as this metamodel should contribute to the ubiquity of patterns to develop secure systems.

The rest of this paper is organized as follows. Section 2 contains related work and problems addressed in this research. Section 3 proposes our metamodel. Section 4 overviews our process for S&P development. Section 5 discusses our case studies and answers our RQs, and section 6 concludes this paper.

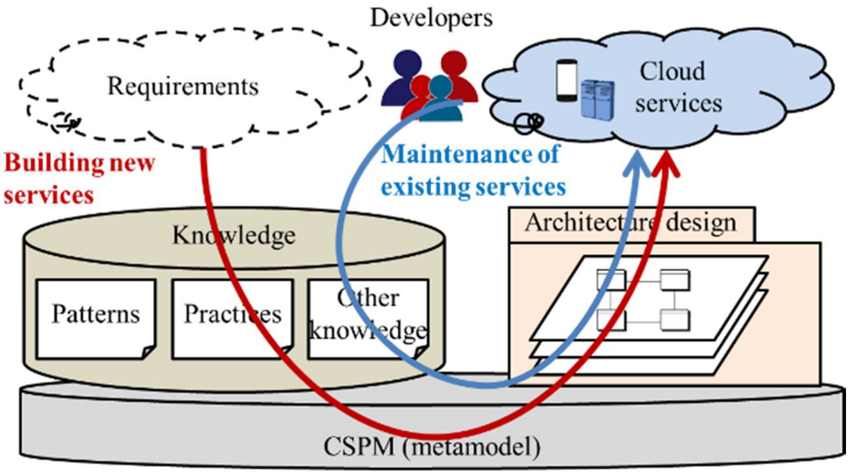


Figure 1: Overview of cloud services and our metamodel

2. Related Work and Motivating Example

2.1. Related Work

Cloud security is considered in several metamodels and abstract reference architectures [5-7]. However, cloud privacy along with security has yet to be considered. Due to their intertwined relationship, they should be addressed simultaneously.

One study reported an abstract security reference architecture model to develop secure cloud services and systems [5]. This study provided a basis to model multiple layers of the cloud in terms of the security at each layer. However, privacy was not addressed. Another study used a metamodel to model cloud services and resources [6], but neither security nor privacy were considered directly. A different study surveyed software security knowledge and proposed a metamodel to model such knowledge [7]. Unlike that study, which did not include computing, our study incorporates such knowledge into our metamodel.

Some studies have focused on privacy engineering. One did a systematic literature mapping on privacy patterns research [16]. However, this study did not consider a metamodel or security patterns. Another study proposed a metamodel for privacy engineering based on SEMDM, which is a metamodel for software and systems development methodologies [17]. This study did not consider privacy patterns, security patterns, or cloud computing. A different study proposed a privacy engineering metamodel by extending SEMDM [18]. Although it included privacy design strategies, privacy threats, and privacy design patterns as well as listed elements similar to our metamodel, relationships were not considered.

Several metamodels and conceptual models have addressed both S&P [8, 9]. However, they are difficult to apply directly to cloud services.

2.2. Motivating Example

Often a developer who is inexperienced and not an expert in S&P is tasked to build a cloud application. As the developer is aware of her shortcomings, she searches for such documents on S&P. However, this leads to several problems:

- Numerous S&P patterns and documents: Patterns are reusable solutions to recurring problems. Because many S&P patterns (and other documentation) have been proposed, the search results are overwhelming. Selecting the appropriate pattern(s) when many are not applicable to cloud services [11] is difficult, especially for a novice developer.
- Complex relationships between a cloud service and its mechanism: A cloud is composed of three main layers: infrastructure, platform, and software. Although each service is provided from one layer from the users' viewpoint, a service may control data related to other layers [12]. Consequently, selecting and utilizing the appropriate pattern(s) are challenging tasks.
- Practical metamodels for cloud development do not exist: Existing metamodels [8] consist of essential concepts when dealing with S&P issues. However, they cannot deal with real-world S&P issues in cloud development.

3. **Cloud Security and Privacy Metamodel (CSPM)**

3.1. Requirements and high-level architecture

There are three requirements for the metamodel:

- Consistency over multiple layers: Because cloud services tend to be integrated over multiple layers, the metamodel must be able to handle S&P-related knowledge over multiple layers (e.g., software application, platform, and infrastructure layers). SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service) are services that correspond to the software, platform, and infrastructure layers, respectively.
- Compatibility with existing cloud services: The metamodel must be compatible with existing cloud security metamodels and reference architectures to utilize assets.
- Convenience: The metamodel must conveniently access both cloud-specific and cloud-independent knowledge.

3.2. Design of the metamodel

Figure 2 shows the overview of Cloud Security and Privacy Metamodel (CSPM). It can be represented as a UML class diagram (Fig. 3). Table 1 overviews the purpose and major concepts of these packages. The metamodel satisfies the aforementioned requirement by having the following features.

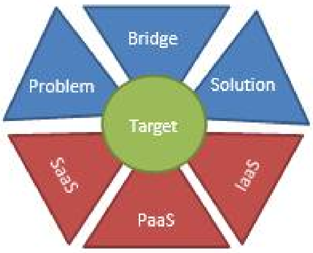


Figure 2: Overview of CSPM for S&P in cloud services

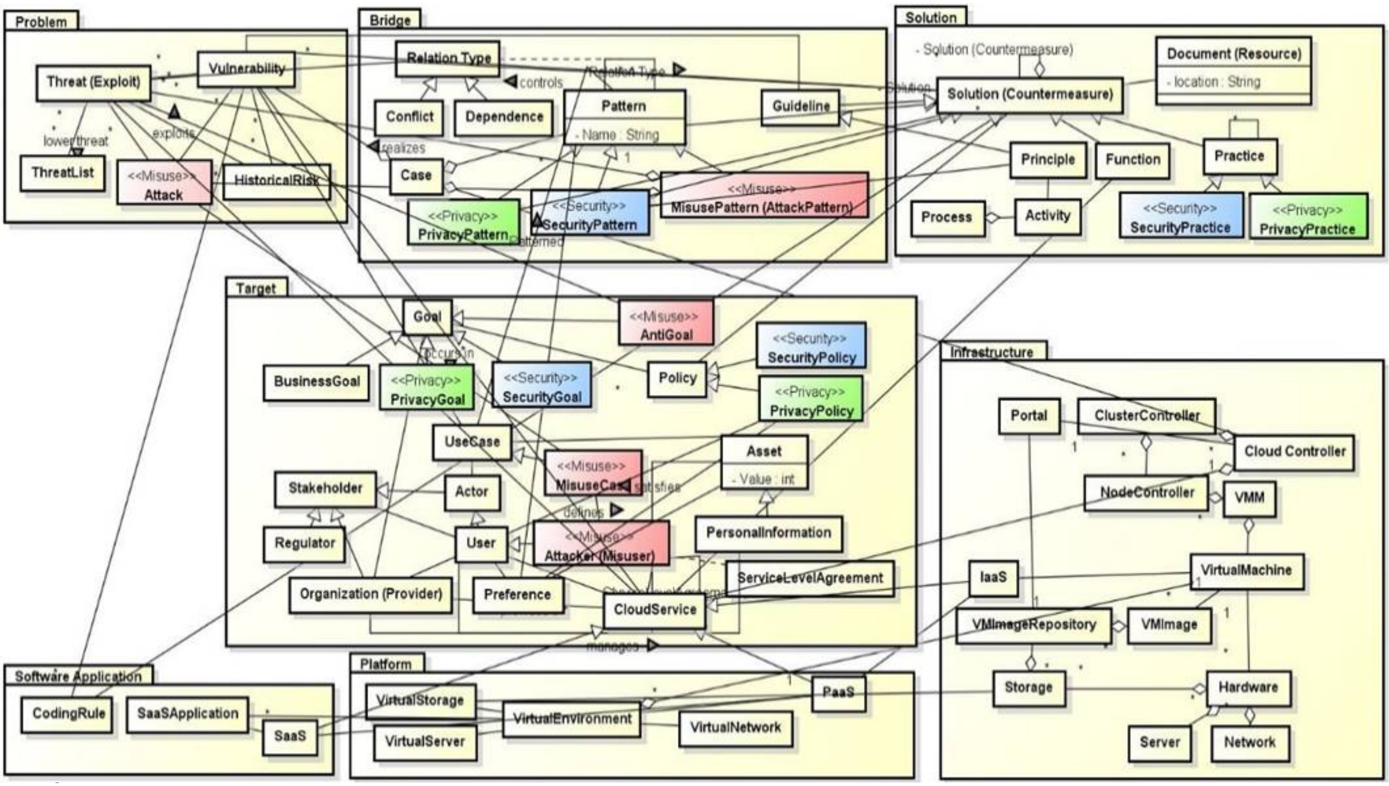


Figure 3: Overview of CSPM

- Consistency over multiple layers: The problem, bridge, and solution packages are fundamental in all layers. Not only do they provide concepts common between layers, they organize their relationships. Consequently, they uniformly handle S&P-related knowledge over different layers.

- Compatibility with existing cloud services: In addition to consistency, the packages include concepts according to the relationships defined in existing metamodels [5][7]. Hence, the proposed metamodel can work with existing metamodels.

- Convenience: Separating general concepts from specific ones (e.g., layers, cloud-specific knowledge, and cloud-independent knowledge) into packages makes the metamodel easy to access.

Table 1: Packages in the metamodel

Package	Outline	Major concepts
Problem	Common concepts for problems	Threat, vulnerability, attack
Bridge	Concepts on the relationships between problems and corresponding solutions	Pattern, case, guideline
Solution	Common concepts for solutions	Solution (countermeasure), security function, practice
Software Application	Concepts specific to the software application layer	Application, coding rule
Platform	Concepts specific to the platform layer	Virtual environment, virtual storage
Infrastructure	Concepts specific to the infrastructure layer	Virtual machine, hardware
Target	Concepts specific to the target application	Goal, policy, asset, cloud service

4. CSPM

CSPM is large scale because it covers almost all aspects of a cloud system. Hence, real-world applications may be difficult. However, one reason that metamodels seem impractical is the absence of detailed descriptions. Herein a process for S&P development with CSPM is proposed, and its usage is evaluated from different viewpoints with an emphasis on the metamodel components.

4.1. S&P Development Process

S&P development consists of four steps: analysis, design, implementation, and test (Fig. 4). Each step is described below:

(1) S&P Requirement Analysis: After analyzing the system requirements, the threats in the current system model are determined using a threat model such as STRIDE [14] or Vulnerability View that is a simplified CSPM emphasizing components related to vulnerabilities.

(2) S&P Design: Solutions are determined. S&P patterns and other documents can be used because they suggest a solution to the system. Pattern View, which is a simplified CSPM emphasizing components related to S&P patterns, can help select appropriate patterns from the knowledge base.

(3) S&P Implementation and (4) Test: Subsequently, the system is implemented and tested. If problems arise during the test, return to (1). If several patterns are combined to reach an original solution, the new patterns should be added to the knowledge base.

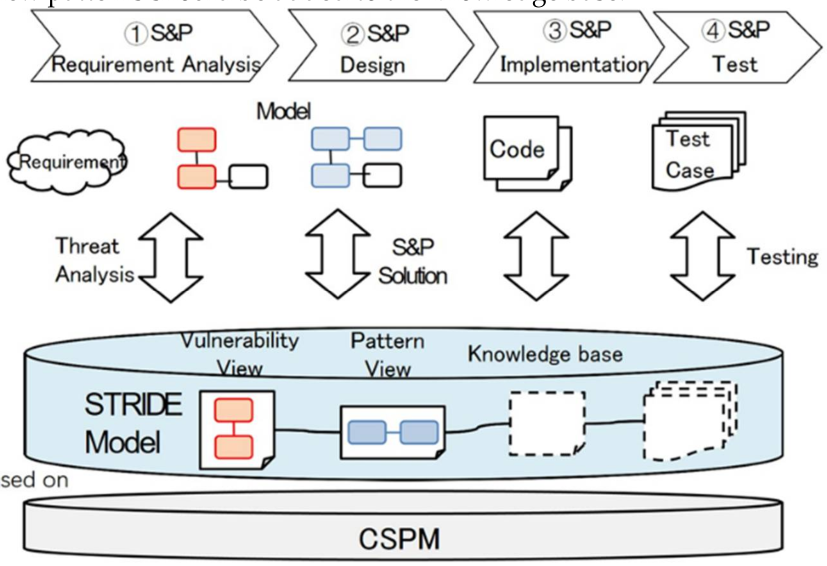


Figure 4: Overview of the S&P Development Process

4.2. Vulnerability View

The Vulnerability View is a simplification for the metamodel (Fig. 5). That is, it is an abstraction of the metamodel to allow users to focus on vulnerability and related concepts. It can model vulnerabilities from databases such as the Common Vulnerabilities and Exposures (CVE).

For example, a vulnerability Cross-site Scripting (XSS) can be modeled in Figure 6. In the figure, elements related to the vulnerability are modeled with stereotypes specifying corresponding concepts in the Vulnerability View. To identify problems and implement countermeasures easily, the model in the figure helps visualization of vulnerable elements.

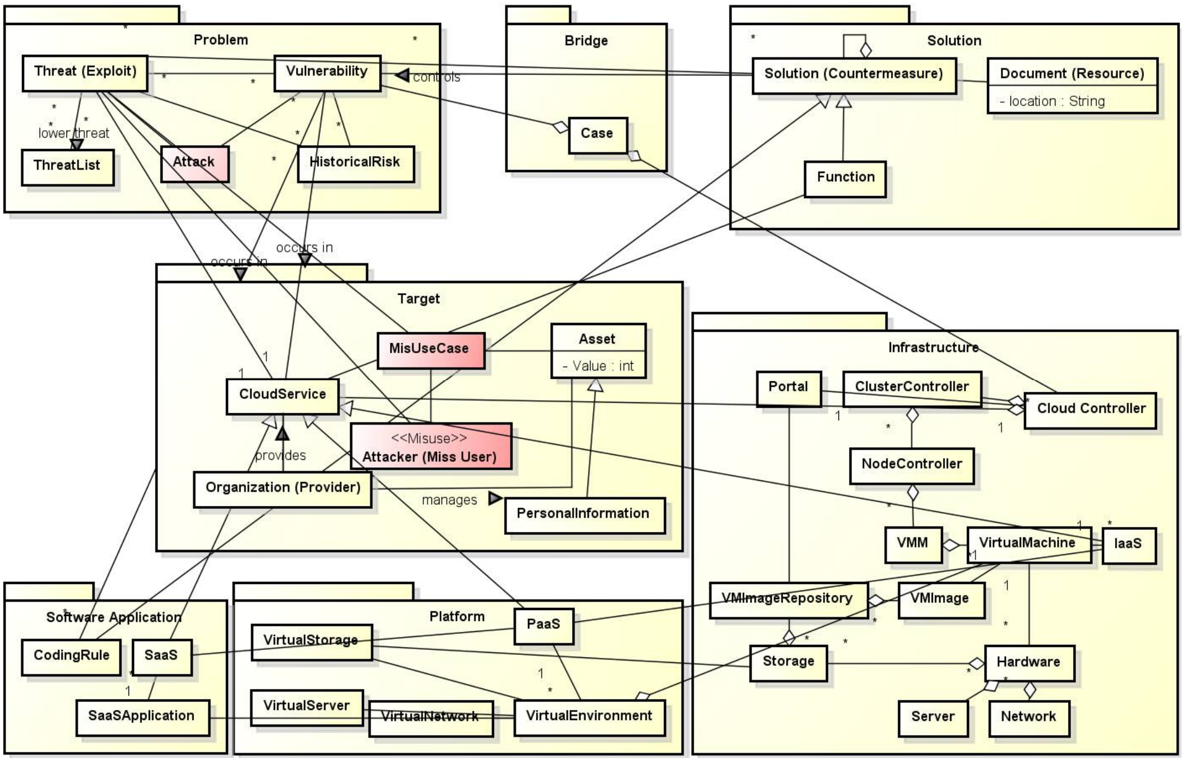


Figure 5: Overview of the Vulnerability View model

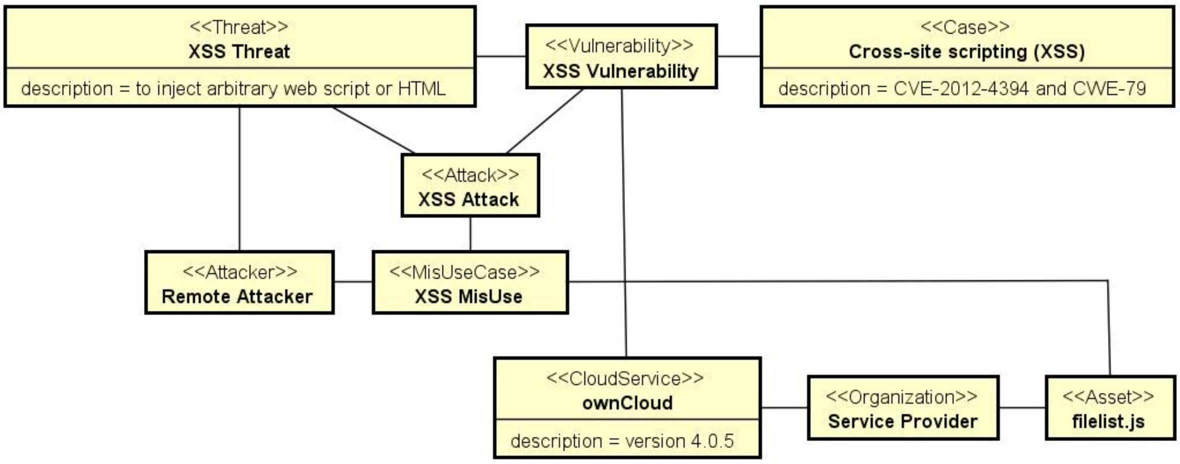


Figure 6: Model a Cross-site scripting (XSS) vulnerability

4.3. Pattern View

Pattern View is also a simplified metamodel that emphasizes elements related to S&P patterns such as goals, threats, and solutions (Fig. 7). Because it can analyze the requirements and threats to a system, applicable S&P patterns can be determined. Section 5 shows an example as a case study.

Pattern View can depict the pattern problem and solution (Fig. 8). In the figure, elements related to the pattern are modeled with stereotypes specifying corresponding concepts in the Pattern View.

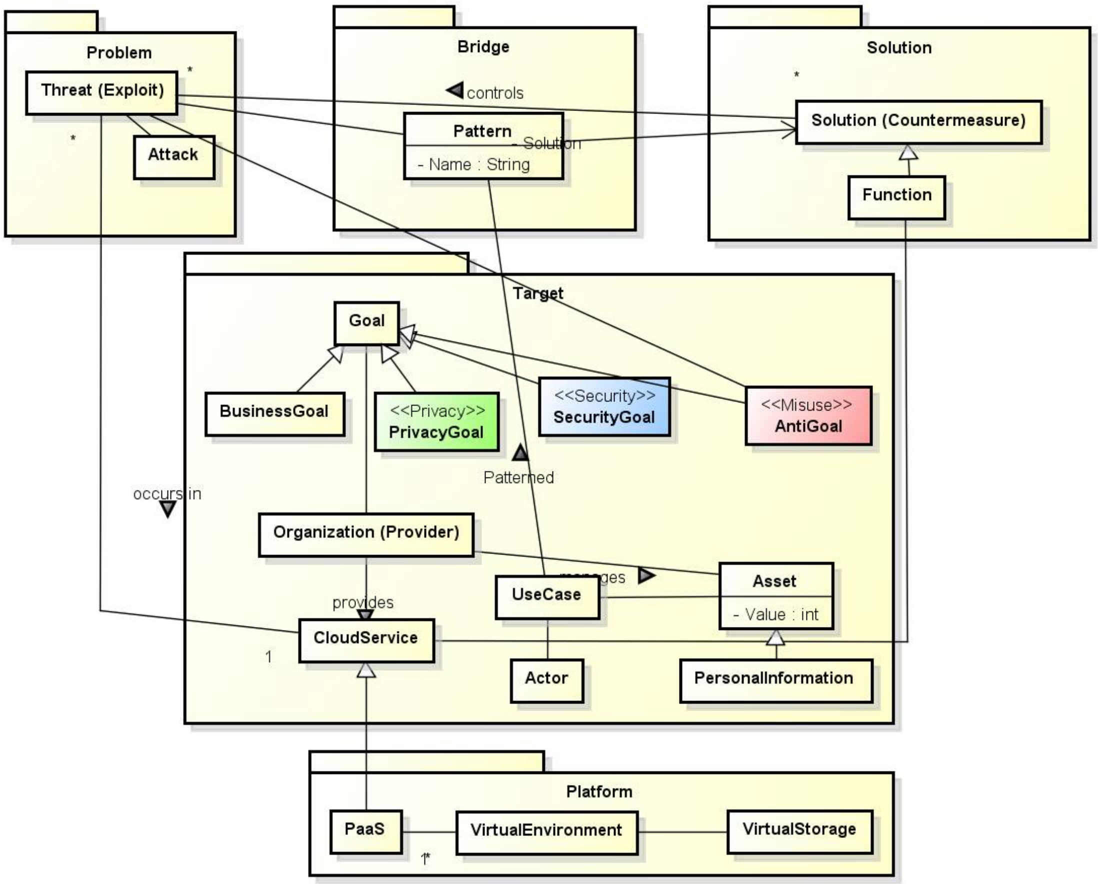


Figure 7: Overview of the Pattern View model

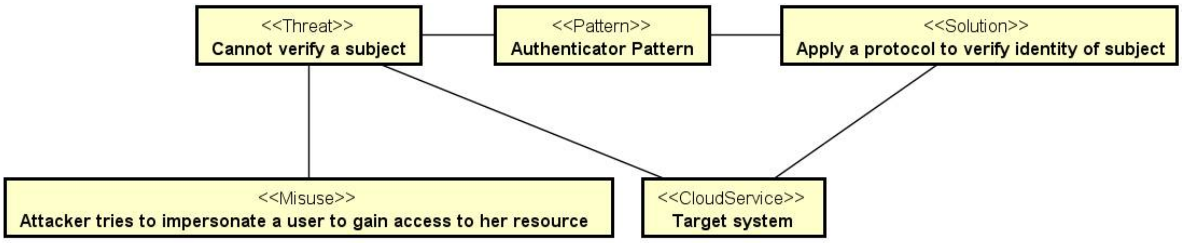


Figure 8: Example of the Pattern View structure using the Authenticator Pattern

5. Experiment and Case Studyr

5.1. Experiment

A contrast experiment evaluated the impact of CSPM and investigated the RQs.

5.1.1. Experiment Setting

The experiment was designed to evaluate the impact of CSPM. The experiment involved two groups of college students, ranging from fourth year undergraduate to second year master’s students. The groups were labeled as the experiment group (EG) and the control group (CG).

Regardless of the group, participants were asked to read the class diagram and use case explanation to determine the S&P issues in the system model. The system model was simplified from student work and contained several security threats. The participants were asked to resolve S&P issues on the model level. As a reference, we prepared some S&P patterns, but not all were applicable to this system. After the experiment, participants completed a questionnaire.

EG received additional support. They were given CSPM and guidelines that explained CSPM (section 4.1). For example, the guidelines showed how to determine the appropriate pattern for a given threat, how to apply a pattern, etc. In addition, the S&P patterns for the experiment group contained the Pattern View structure (Fig. 8) to confirm its contribution.

5.1.2. Experiment Results

The results for CG are shown in Table 2, while those of EG are shown in Table 3. The distribution for each group is shown as a box plot in Fig. 9. Four variables (total time to complete the assignment, number of problems found in the system, number of problems solved by revising the model, and number of patterns used to solve problem) were measured.

Some of the participants (C1 and C2) read all the reference patterns. C1 spent a long time on the assignment and used the patterns. However, C2 was confused about pattern use and did not use the reference patterns to complete the task. On the other hand, other participants (C3–C5) did not review the reference patterns. Due to previous development experience, C3 did not need the reference patterns to be successful. C4 and C5 finished quickly. Although they addressed the main S&P issues, they did not address minor problems.

The results of the EG group were similar. They solved a minimum number of principle problems with a greater emphasis on S&P patterns and revised the model correctly. Most completed the experiment in about an hour. Some issues not related to the reference patterns (e.g., DDoS attack) were not solved.

Although the difference between EG and CG to solve problems was not significantly different, EG was more proficient. Three or more main S&P issues were resolved by the EG participants, whereas the number of issues addressed fluctuated widely within the CG group. This difference is attributed to the S&P patterns.

Although we speculated that the EG group would complete the tasks faster than the CG group, the completion time between the two groups were statistically insignificant. This may be attributed to the time that the EG group spent reading the metamodel and guidelines. Comparing C1, who used patterns for assignment, to the EG group indicates that applying our approach is less time consuming because C1 spent a lot of time reading the reference material.

All participants in the EG group provided similar responses to the questionnaire. All indicated that the Pattern View of the metamodel itself (Fig. 7) is easy to understand, but it has low utility. On the other hand, the explanation and example in the guidelines are very helpful, especially for applying patterns. Participants responded that the Pattern View structure of the S&P pattern (Fig. 8) is helpful, but it is preferable to use this in conjunction with a detailed description of the patterns.

Table 2: Results for the control group (CG)

Participant	Time [min]	Problems found	Problems solved	Pattern used
C1	100	5	3	3
C2	180	2	1	\
C3	60	5	5	\
C4	60	3	2	\
C5	60	3	1	1
Average	92	3.6	2.5	0.8

Table 3: Results for the experiment group (EG)

Participant	Time [min]	Problems found	Problems solved	Pattern used
E1	90	3	3	3
E2	70	3	3	2
E3	60	4	3	3
E4	60	4	3	3
E5	50	5	5	3
Average	66	3.8	3.4	2.8

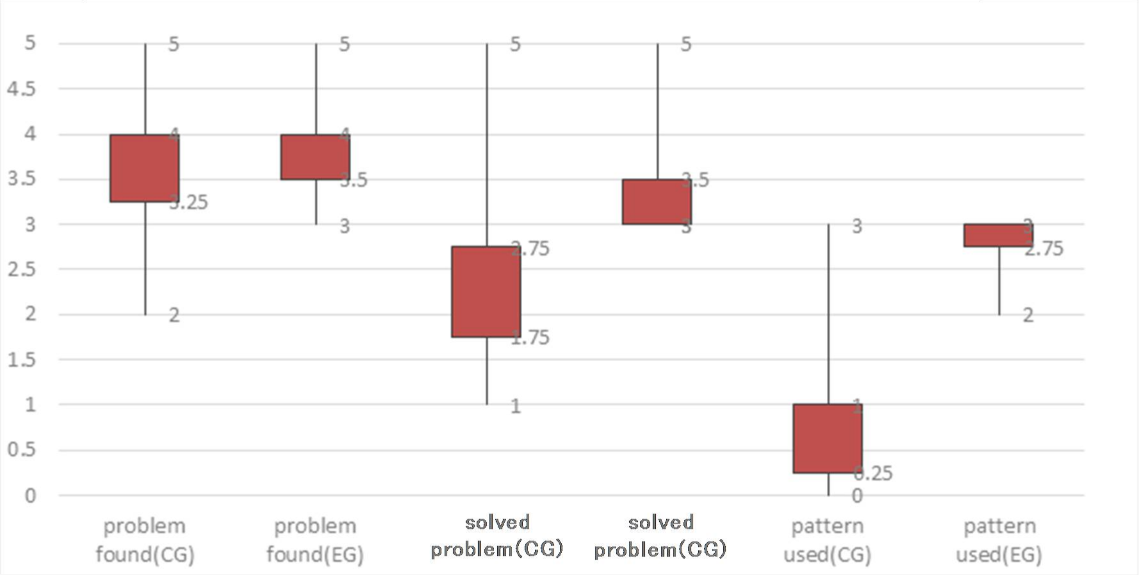


Figure 9: Box plot of data in Tables 2 and 3

5.2. Case study: "Treasure-Hunting Game"

A case study based on an Android game application that stores data in a cloud evaluated the effectiveness of the solutions analyzed by CSPM. Specifically, the original version and that enhanced by our metamodel were used to verify the contribution.

5.2.1. Preparation

There were five steps:

1. Design the target software on the model level
2. Correlate the model to our metamodel to determine the S&P goals (step 1 in our proposed process)
3. Analyze threats in the system and determine the S&P pattern for each threat to revise the target software model (step 2 in our proposed process)
4. Implement the software using both the original model and the revised model (step 3 in our proposed process)
5. Test and compare the results (step 4 in our proposed process)

Typically, S&P analysis is conducted prior to designing the model. To confirm the contribution of CSPM, a student work (the "Treasure Hunting Game") with a known model was used. Similar to popular commercial games (e.g., Pokémon Go and Ingress), this game is an AR application where streets contain multiple spots, and one spot has the hidden treasure. The initial structure and interface are shown in Figs. 10 and 11, respectively. To begin, players input their names in order to manually save their data like hints and coins into the cloud and to check target player's data (Fig. 12). In this case study, cloud functions were implemented on Amazon Web Service (AWS).

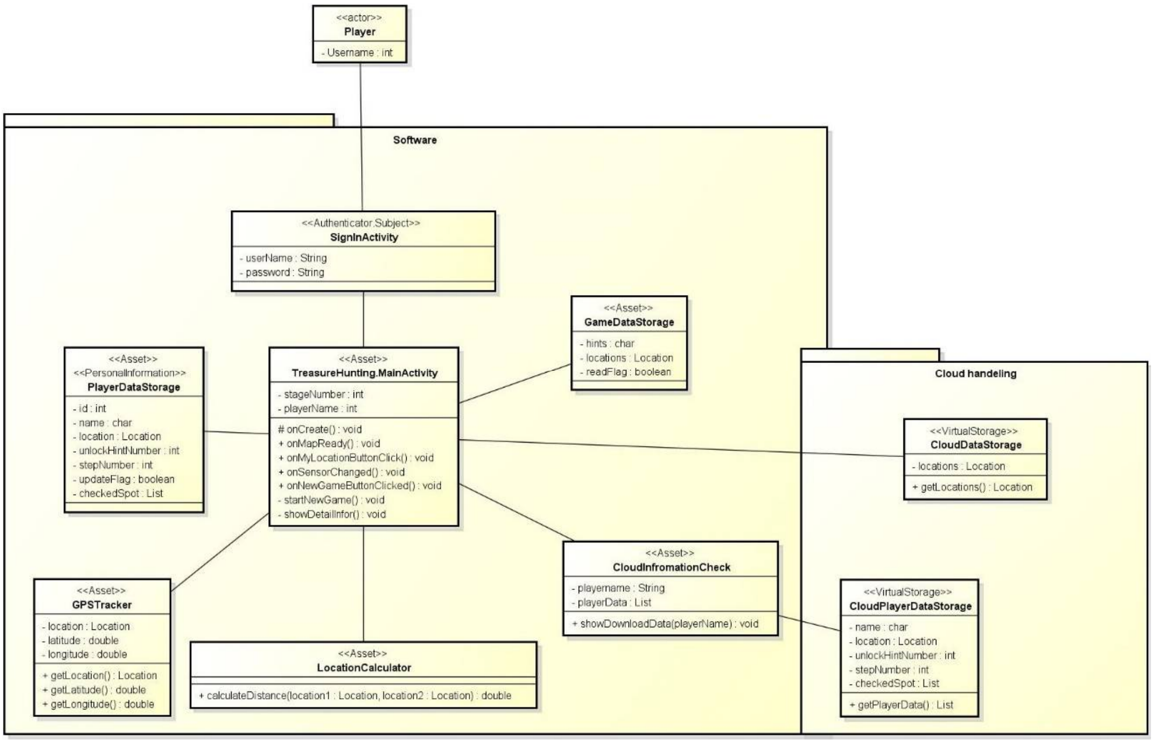


Figure 10: Initial structure of the “Treasure-Hunting Game” in UML class diagram

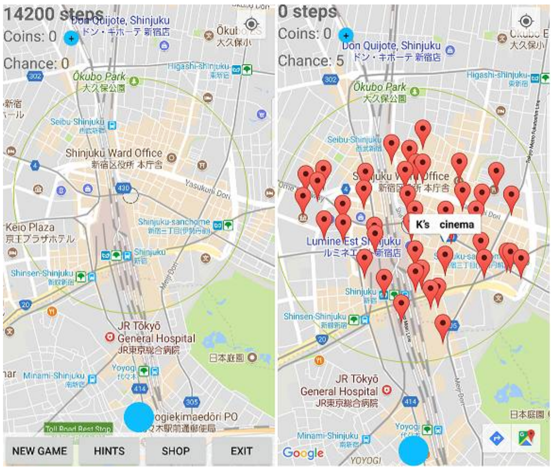


Figure 11: Interface of the “Treasure-Hunting Game”

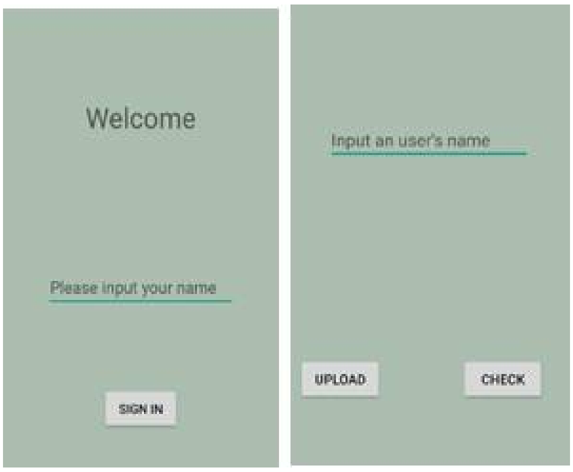


Figure 12: Interface for cloud user data storage

5.2.2. Results

The STRIDE model (Microsoft, 2002) was used in the S&P requirement analysis (Table 4). Because Android API and AWS API addressed the threats due to tampering with local data or listening to transmissions, the case study was concerned with the authentication problem and access right problem, as described below:

- **Authentication Problem**: Because anyone can use this game, the identity spoofing risk is high, which may lead to data tampering in cloud storage.
  Pattern and Solution: The *Authenticator Pattern* requires a user to sign up and sign in before accessing the system. The proposed method added an authenticator. However, other patterns like *Password Design and Use* may also provide support.
- **Access Right Problem**: The original game only requires a user name to display user data on the screen. This feature may be designed so that friends' data can be checked, but anyone can check a user's information.
  Pattern and Solution: The proposed method added the *Authorization Pattern* or *Role-based Access Control Pattern* to limit access rights.

Figure 13 shows the results of our analysis of Goals, Problems, Patterns, and Solutions for the Pattern View metamodel originally shown in Fig. 7. The modified model of the software system is shown in Fig. 14. In these figures, elements related to the security patterns are modeled with stereotypes specifying corresponding concepts in the Pattern View.

Both models were tested to validate the effectiveness of the proposed method. The results confirm that the authentication problem is resolved (Fig. 15) and the access controller works as intended (Fig. 16).

295 Table 4. Result for the S&P requirement analysis based on the STRIDE model

Goal	Anti-Goal	Security Problem	Specific example	Security pattern	Solution
Tamper proof data	Gain ability to tamper with data	Unauthorized actors tampering with local data	User accesses local data on their phone, changing their score	Encryption pattern	Provided by the Android phone itself
		Unauthorized actors tampering with cloud data	User logs in as another user once user name is known. Cloud user data might be modified.	Authentication pattern	Require a password for each user
Confidentiality	Gain access to confidential information	Unauthorized actors listening to the transmissions to and from the server	Man in the middle attack	Transmission pattern	API automatically uses SSL and can be set to use a VPN
		Information disclosure	User accesses other players' data without permission	Authorization pattern, Role-Based Control pattern (RBAC)	Control access rights for each player
		Elevation of privilege	User pretends to be an administrator and granted unlimited access to all game data	Authentication pattern, Transmission pattern	Player can only access the database through the software, which is limited by the permission levels of a third-party server.
Reliability	Gain ability to access other player's data	Identity spoofing	No user authentication in the system and anyone can access the game	Authenticator pattern	Require sign up and sign in
Availability	Bring down the servers	Denial of service	Server becomes flooded by non-legitimate messages causing packets by legitimate users to be dropped	Firewall, DDoS patterns	Unrealistic issue due to the game's small scale

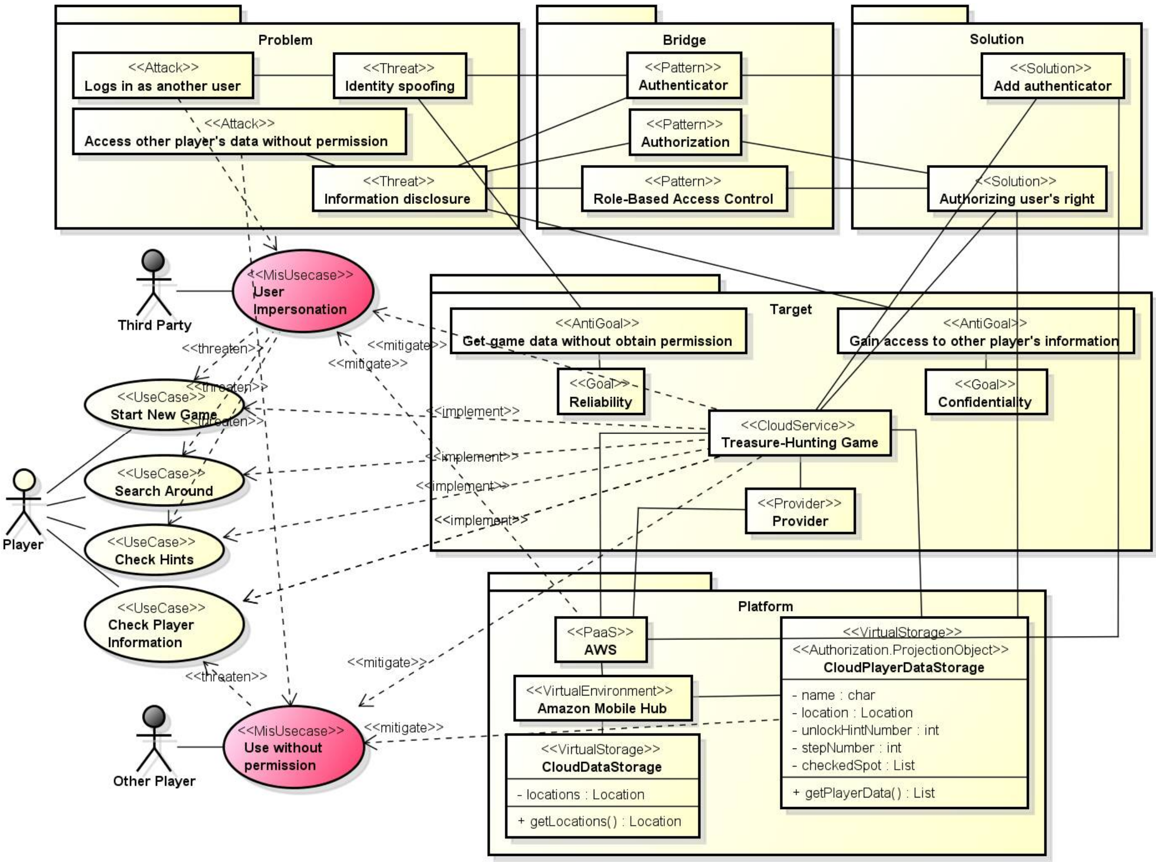

Problem
<<Attack>> Logs in as another user
<<Threat>> Identity spoofing
<<Attack>> Access other player's data without permission
<<Threat>> Information disclosure
Bridge
<<Pattern>> Authenticator
<<Pattern>> Authorization
<<Pattern>> Role-Based Access Control
Solution
<<Solution>> Add authenticator
<<Solution>> Authorizing user's right
Third Party
<<MisUsecase>> User Impersonation
<<threaten>>
<<mitigate>>
Target
<<AntiGoal>> Get game data without obtain permission
<<AntiGoal>> Gain access to other player's information
<<Goal>> Reliability
<<Goal>> Confidentiality
<<CloudService>> Treasure-Hunting Game
<<Provider>> Provider
<<implement>>
<<UseCase>> Start New Game
<<UseCase>> Search Around
<<UseCase>> Check Hints
<<UseCase>> Check Player Information
Player
Other Player
<<MisUsecase>> Use without permission
Platform
<<PaaS>> AWS
<<VirtualEnvironment>> Amazon Mobile Hub
<<VirtualStorage>> CloudDataStorage
- locations : Location
+ getLocations() : Location
<<VirtualStorage>> <<Authorization.ProjectionObject>> CloudPlayerDataStorage
- name : char
- location : Location
- unlockHintNumber : int
- stepNumber : int
- checkedSpot : List
+ getPlayerData() : List

Figure 13: Relationship between Problems, Patterns, Solutions, and cloud system based in CSPM

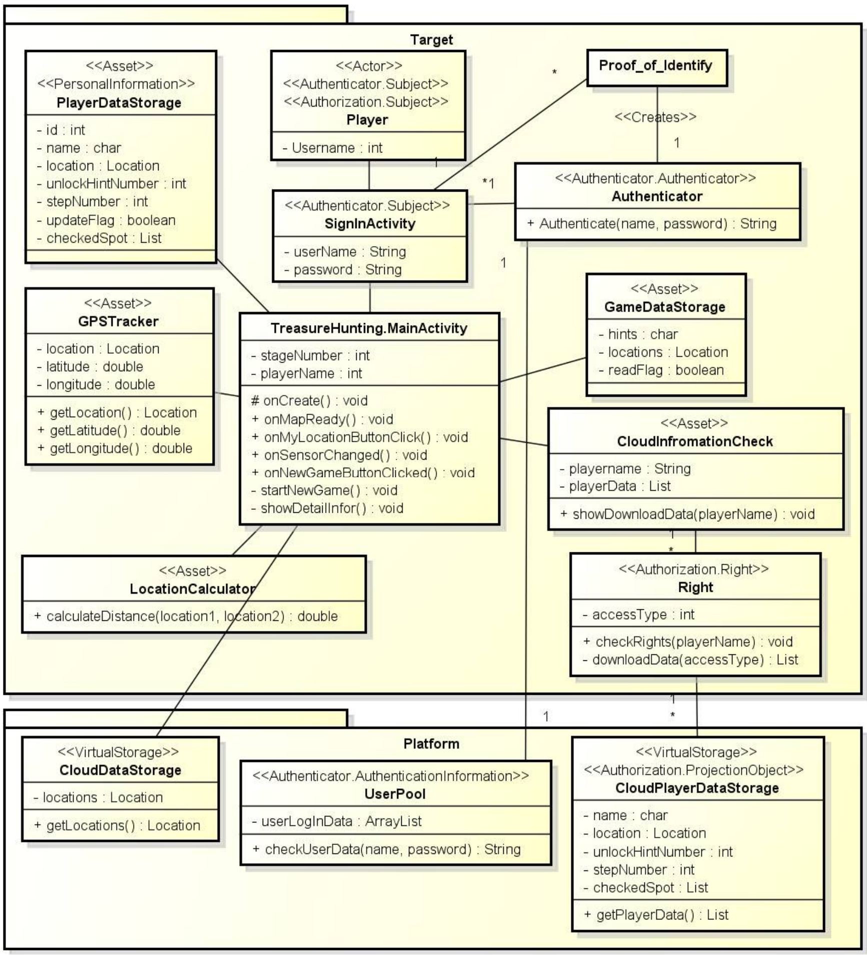
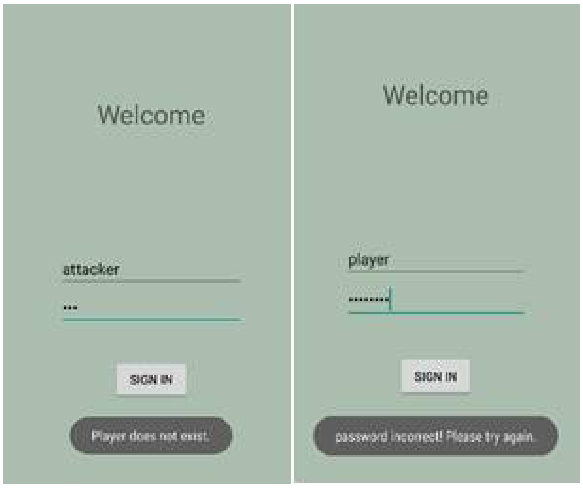

Target
<<Asset>> <<PersonalInformation>> PlayerDataStorage
- id : int
- name : char
- location : Location
- unlockHintNumber : int
- stepNumber : int
- updateFlag : boolean
- checkedSpot : List
<<Actor>> <<Authenticator.Subject>> <<Authorization.Subject>> Player
- Username : int
Proof_of_Identify
<<Creates>>
*
1
*1
<<Authenticator.Subject>> SignInActivity
- userName : String
- password : String
<<Authenticator.Authenticator>> Authenticator
+ Authenticate(name, password) : String
<<Asset>> GPSTracker
- location : Location
- latitude : double
- longitude : double
+ getLocation() : Location
+ getLatitude() : double
+ getLongitude() : double
TreasureHunting.MainActivity
- stageNumber : int
- playerName : int
# onCreate() : void
+ onMapReady() : void
+ onMyLocationButtonClick() : void
+ onSensorChanged() : void
+ onNewGameButtonClicked() : void
- startNewGame() : void
- showDetailInfor() : void
<<Asset>> GameDataStorage
- hints : char
- locations : Location
- readFlag : boolean
<<Asset>> CloudInfromationCheck
- playername : String
- playerData : List
+ showDownloadData(playerName) : void
<<Asset>> LocationCalculator
+ calculateDistance(location1, location2) : double
<<Authorization.Right>> Right
- accessType : int
+ checkRights(playerName) : void
- downloadData(accessType) : List
Platform
<<VirtualStorage>> CloudDataStorage
- locations : Location
+ getLocations() : Location
<<Authenticator.AuthenticationInformation>> UserPool
- userLogInData : ArrayList
+ checkUserData(name, password) : String
<<VirtualStorage>> <<Authorization.ProjectionObject>> CloudPlayerDataStorage
- name : char
- location : Location
- unlockHintNumber : int
- stepNumber : int
- checkedSpot : List
+ getPlayerData() : List

Figure 14: Structure of the “Treasure-Hunting Game” revised by CSPM



Welcome

attacker

•••

Player does not exist.

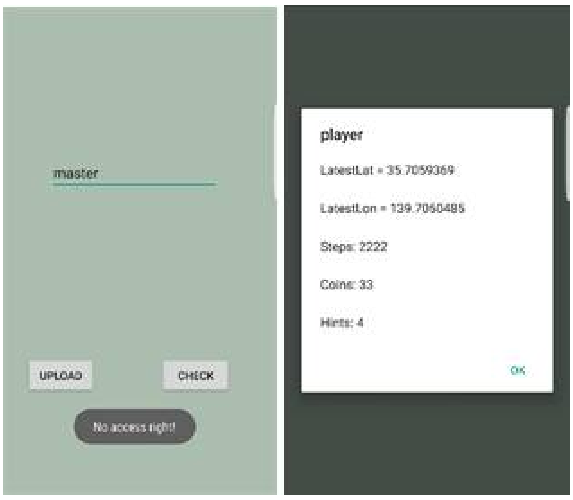
Welcome

player

••••••••

password incorrect! Please try again.

Figure 15: Prevention of Misuse of Unregistered user (left) and Identity Thief (right)



master

No access right!

player

LatestLat = 35.7059369

LatestLon = 139.7050485

Steps: 2222

Coins: 33

Hints: 4

Figure 16: Player data check with an access controller

5.3. Discussion

RQ1. Can CSPM resolve S&P problems and help application of the corresponding patterns?

In the experiment, EG solved more problems than CG in the same or less time. EG participants selected and applied the appropriate pattern to revise the model due to the support of our approach. Although the knowledge base in this study is small, the proposed method should provide improved results when dealing with more S&P patterns. Our approach, especially the Pattern View structure of the S&P pattern, can identify necessary patterns and improve pattern comprehension.

Unlike previous research, which used metamodels for security issues, this study used CSPM to combine security (i.e., authentication) and privacy (i.e., access right control). This study only considered simple combinations of S&P patterns, which were indicated previously (e.g., *Authenticator Pattern* with *Single Access Point Pattern*), due to the small scale of the target system. In the future, more complex systems should be evaluated.

In this case study, the Pattern View deals mainly with the S&P pattern. Because problems are found before using patterns, the ability of Pattern View to find problems has yet to be confirmed. In the future, the ability of Vulnerability View to find problems should be investigated.

CSPM can select and use S&P patterns to solve problems through the Pattern View. In the future, the ability to find solutions for more complex systems or to combine several S&P patterns should be confirmed.

RQ2. Can CSPM improve the system by efficiently providing S&P solutions?

The ability of Pattern View of CSPM to revise the model was investigated via a case study. The problems in the original target system are addressed in the revision. CSPM is effective, at least for a simplified system. Not all the components of the cloud system were considered by CSPM in the case study. As a system becomes more complex, other issues may arise. Hence, the entire metamodel should be further evaluated in the future.

RQ3. Can CSPM and processes using CSPM be deployed in practical real-world applications?

The proposed process was used in the experiment and case study, demonstrating that CSPM is applicable to S&P analysis and during cloud system development, respectively. Both indicate that CSPM is practical in some situations. However, the participants in the experiment provided negative feedback about the metamodel's usefulness. They felt that the current guidelines are more useful than the metamodel. Revising the guidelines to provide more examples of CSPM usage should improve the practicality of our approach.

6. Conclusion and Future Work

CSPM, which deals with S&P in cloud services, can be used in software development. Its effectiveness and usability are confirmed via a case study and an experiment. The case study, which involves an application similar to a commercial one using a conventional cloud platform, suggests that CSPM has practical applications in industrial development.

There are several future directions. The first is to implement more complex case studies such as a cloud system with several layers to evaluate the effectiveness of CSPM. The second is to apply Vulnerability View and the Pattern View semi-automatically to detect specific threats. The third is to develop a detailed framework to broaden the usage of CSPM.

Studies have demonstrated the effectiveness of tools such as patterns, reference architectures, and metamodels to handle the complexity of cloud-based systems and other new technologies such as IoT. However, such tools are difficult to apply in real-world situations due to their elaborate methodologies. This study should contribute to the practical application of such tools.

References

1. D. Riehle and H. Zullighoven, 1996, "Understanding and Using Pat- terns in Software Development," Theory and Practice of Object Systems, Vol.2, No.1, pp.3-13.
2. K. Hashizume, N. Yoshioka and E.B. Fernandez, 2011, "Misuse Patterns for Cloud Computing," 2nd Asian Conference on Pattern Languages of Programs (AsianPLoP'11).
3. K. Hashizume, N. Yoshioka and E.B. Fernandez, 2012, "Three Misuse Patterns for Cloud Computing," in "Security Engineering for Cloud Computing: Approaches and Tools," edited by David Garcia Rosado and Eduardo Fernandez-Medina, IGI Global.
4. T. Reimer, P. Abraham and Q. Tan, 2013, "Federated Identity Access Broker Pattern for Cloud Computing", 16th International Conference on Network-Based Information Systems (NBiS).
5. E.B. Fernandez, Raul Monge, and Keiko Hashizume, 2016, "Building a security reference architecture for cloud systems", Requirements Engineering, Doi: 10.1007/s00766-014-0218-7, Vol. 21, No. 2, pp. 225-249
6. K. Chatziprimou, K. Lano, and S. Zschaler, 2013, "Towards a Meta-model of the Cloud Computing Resource Landscape." MODELSWARD.
7. A. Hazeyama, 2012, "Survey on Body of Knowledge Regarding Software Security", 13th ACIS International Conference on Software Engineering, Artificial Intelligence
8. C. Kalloniatis, E. Kavakli and S. Gritzalis,2008, "Addressing privacy requirements in system design: the PriS method," Requirements Engineering, Vol.13
9. R. Tesoriero, 2011, "Model-Driven Privacy and Security in Multimodal Social Media UIs," International Workshops MSM 2011.
10. H. Washizaki, et al., 2016, "A Metamodel for Security and Privacy Knowledge in Cloud Services," Proc. 12th IEEE World Congress on Services (SERVICES 2016)
11. E.B. Fernandez, N. Yoshioka, H. Washizaki, J. Jurjens, M.VanHilst, and G. Pernul, 2010, "Using security patterns to develop secure systems," in "Software Engineering for Secure Systems," edited by H. Mouratidis, IGI Global, pp. 16-31

12. S. Subashini and V. Kavitha, 2011, “A survey on security issues in service delivery models of cloud computing,” Journal of Network and Computer Applications, Vol.34, No.1, pp.1–11, 2011.
13. A.A. Almutairi, et al., 2012, “A Distributed Access Control Architecture for Cloud Computing,” IEEE Software, Vol. 29, No. 2, pp.36-44
14. Microsoft, 2002, "The STRIDE Threat Model", https://msdn.microsoft.com/enus/library/ee823878(v=cs.20).aspx
15. T. Xia, H. Washizaki, T. Kato, H. Kaiya, S. Ogata, E.B. Fernandez, H. Kanuka, M. Yoshino, D. Yamamoto, T. Okubo, N. Yoshioka and A. Hazeyama, 2018, “Cloud Security and Privacy Metamodel: Metamodel for Security and Privacy Knowledge in Cloud Services,” 6th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2018)
16. J. Lenhard, L. Fritsch, and S. Herold, “A literature study on privacy patterns research,” Proceedings of the 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pp. 194-201, IEEE, 2017.
17. Y. S. Martín, and J. M. del Álamo, “A metamodel for privacy engineering methods,” Proceedings of the CEUR Workshop, 2017.
18. J. M. Alamo, Y. S. Martín, and J. C. Caiza, “Towards Organizing the Growing Knowledge on Privacy Engineering,” Proceedings of the IFIP International Summer School on Privacy and Identity Management, pp. 15-24, Springer, 2017.