Towards a Pattern Language for Security Risk Analysis of Web Applications

Yan Li, SINTEF ICT and University of Oslo
Ragnhild Kobro Runde, University of Oslo
Ketil Stølen, SINTEF ICT and University of Oslo

This article introduces a pattern language for security risk analysis of web applications in an example driven manner. The example patterns presented include a composite pattern and three basic patterns, namely a security requirements pattern, a web application design pattern and a risk analysis modelling pattern. The pattern language is intended to be used as a guideline to capture the security risk picture of a web application, especially in the early phase of the software development life cycle. The overall aim is to support light weighted security risk analysis for web applications.

Categories and Subject Descriptors: D.D.9 [**Software Engineering**]: Management—*Software quality assurance (SQA)*; K.6.M [**Management of Computing and Information Systems**]: Miscellaneous—*Security*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

General Terms: Pattern language

Additional Key Words and Phrases: Security, Risk analysis, Web application

ACM Reference Format:

Li, Y.,Runde, R.K. and Stølen, K., 2013.Towards a Pattern Language for Security Risk Analysis of Web Application. jn 2, 3, Article 1 (October 2013), 17 pages.

1. INTRODUCTION

For the past decades or so, the web has become the main platform for business, social exchange, and governance, etc. It offers the relatively cheaper and easier way to communicate and exchange information with fast growing usage of different web applications. A web application has the advantage that it can perform its functions regardless of the operating system (e.g. Android, iOS, Windows) and user device (e.g. desktop, laptop, mobile phone, watch, glasses). On the other hand, which is different from traditional software, the wide spread use of web applications and relatively open environment give rise to potential known and unknown vulnerabilities. Security risks could expose web users to threats and unwanted incidents caused by web-based crime. The incidents occurring due to vulnerabilities of web applications could be very costly as well.

Risk analysis is a very efficient way to identify security risks in web applications. Security risk analysis is the process of risk analysis specialized towards security [Li 2012], in which risk analysis is defined as a collective term of the following steps: establishing the context, risk identification, risk estimation, risk evaluation and risk treatment (adapted from ISO 31000 [International Standards Organization 2009a]). To deal with new security threats and

Author's address: Yan Li, Department for Networked Systems and Services, SINTEF ICT P.O. Box 124 Blindern, N-0314 Oslo, Norway; email:yan.li@sintef.no; Ragnhild Kobro Runde, Department of Informatics, University of Oslo P.O. Box 1080 Blindern, N-0316 Oslo, Norway ; email: ragnhild.runde@ifi.uio.no; Ketil Stølen, Department for Networked Systems and Services, SINTEF ICT P.O. Box 124 Blindern, N-0314 Oslo, Norway; email: ketil.stolen@sintef.no

increasing effort on quality assurance of web applications, it is wise to introduce a specialized methodology for security risk analysis that can be carried out continuously as an integrated part of the business and the development process, especially in the early phase of the software development life cycle. In addition, to ensure that the risk picture can be updated regularly, there is need for system owners to do small risk analysis without heavy involvement from other departments. All of these motivate the needs for a pattern language supporting light weighted security risk analysis for web applications.

Patterns are generally referred to as a mean to address the essence of a recurring problem and define best practices for that problem, with the possibility to derive context specific solutions [Buschmann et al. 1996][Gamma et al. 1995]. A pattern language [Buschmann et al. 1996] is a structured method for describing good design practices within a field of expertise. It may be expressed as a combination of patterns where each pattern offers a solution for an isolated problem while the combined solutions from the application of several patterns can successfully solve very large, complex problems.

The terms "pattern" and "pattern language" were initially introduced in [Alexander 1977] for the design of buildings and towns. Later, the concept of pattern was adapted to many other domains, including the software development domain. Software design patterns are used to enable software design in a more effective and efficient way [Gamma et al. 1995] and some of them address security issues. A group of seven architectural patterns for building application security was presented in [Yoder and Barcalow 1998]. Eight patterns in the form of architectural and procedural guidelines was proposed in [Romanosky 2001] . Afterwards, a group of 29 security patterns categorized as structural and procedural patterns for web application development was introduced in [Kienzle and Elder 2002]. A book [Steel et al. 2006] about core security patterns proposes various patterns to support building end-to-end security into J2EE enterprise applications. A list of security design patterns addressing the architectural level, design level and implementation level of software is introduced in [Dougherty et al. 2009]. Recently, a book [Fernandez 2013] has been published about how to incorporate security into every phase of the software life-cycle with a vast catalog of up-to-date security patterns as support.

From the above mentioned literature, we may conclude that many patterns have been proposed for web applications or software security from a development perspective. However, very little research has focused on security risk analysis patterns that address different contexts of web applications in the early phase of design. In addition, as pointed out in a security pattern survey [Yoshioka et al. 2008], there is a need for security risk patterns in the design and implementation phases. Thus, this paper presents a first step towards a pattern language and corresponding patterns for security risk analysis in the context of web applications, aiming to provide a light weighted support for IT staff who needs to perform security risk analysis as early as possible when building web applications.

An outline of our pattern language is given in Section 2. Section 3 shows the usage of the pattern language and illustrates the pattern selection. Section 4 introduces a web application example for the patterns presented later. Section 5 describes a composite pattern example that serves as the basis for selecting basic patterns. Section 6, Section 7 and Section 8 present examples of basic patterns for security requirement capture, web application design and risk analysis modelling respectively. The work is concluded in Section 9.

2. OUTLINE OF SECURITY RISK ANALYSIS PATTERN LANGUAGE FOR WEB APPLICATIONS

A security risk analysis pattern language for web applications with its corresponding patterns should formalize best practices that can be followed when analysing the security risk picture of web applications based on its context (e.g. functionality, platform for development, environment for maintenance). A pattern here defines a reusable and extendible solution for a specific problem or task occurring in security risk analysis for a web application; it can either be a basic pattern or a composite pattern. A basic pattern is an element pattern addressing a partial problem or sub task, while a composite pattern describes how to compose a set of basic patterns or composite patterns in order to solve a specific task in the risk analysis process.

As shown in Figure 1, to address the most central notions within security risk analysis, we distinguish between three kinds of basic patterns, which are basic pattern for security requirements, basic pattern for web application design and basic pattern for risk analysis modelling. Each represents its own dimension, namely:

—Requirements with respect to basic principles of information security standardized in ISO 27000 [International Standards Organization 2009b] (e.g. confidentiality, integrity, availability, authenticity and non-repudiation).

—Key features of web application (e.g. architecture design from business view, logical process view, deployment view and data view).

—Risk analysis modelling (e.g. asset modelling, threat modelling, risk modelling, treatment modelling).

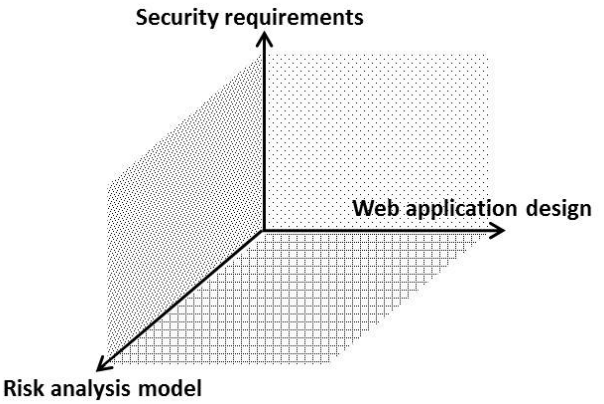


Fig. 1. Dimensions addressed by three different kinds of patterns

In security risk analysis for web applications, a composite pattern mainly addresses one particular phase in the security risk analysis process (establishing the context, risk identification, risk estimation, risk evaluation and risk treatment). A composite pattern is basically a kind of skeleton with place holders for its constituent patterns. A composite pattern specifies an instantiation order for its element patterns, so it is a kind of process pattern. Each composite pattern addresses a specific task in the risk analysis process, which is different from the composite pattern defined in [Riehle 1997] where a composite pattern mainly address a concrete recurring solution for a software design problem.

The documentation of a basic pattern follows a defined format, while composite patterns are expressed graphically, as presented in Section 4. The documentation format of a basic pattern is organized and interpreted according to Table I.

3. USAGE OF SECURITY RISK ANALYSIS PATTERN LANGUAGE FOR WEB APPLICATIONS

By defining the best practices and effective solutions addressing the essence of recurring problems in security risk analysis for web applications, the pattern language facilitates a light weighted method for conducting risk analysis based on the context of each specific web application to be analysed.

As shown in Figure 2, when using this pattern language, we first select a composite pattern from the pattern library based on the phase of the risk analysis in which we are currently involved and the context of the analysis. As already explained, a composite pattern is a kind of skeleton with place holders for its constituent patterns. Based on selected composite pattern, the suitable constituent patterns can be selected and instantiated. For a given pattern, the input parameter is defined based on the context of the web application to be analyzed, while the output parameter results from its instantiations according to the proposed solution, inspired by SaCS [Hauge and

Table I. Documentation format for basic patterns

Notion	Content
Pattern name	The name of the pattern.
Classification	The category of the pattern, i.e. security requirement.
Pattern signature	The symbol, input and output parameters of the pattern.
Motivation	The motivation for applying the pattern.
Context of use	The situation for which the pattern could apply as well as the usage of the pattern.
Problem	The problem that the pattern addresses.
Solution	The solution presented by this pattern.
Instantiation rule	The rule to instantiate the pattern.
Participants	The relevant human stakeholders to be consulted when using this pattern.
Related patterns	The relationship between this pattern and other patterns.
Known uses	The known uses of the pattern, either in preliminary or modified form.

Stølen 2011]. A composite pattern can be instantiated following the instantiation rule of the relevant constituent patterns.

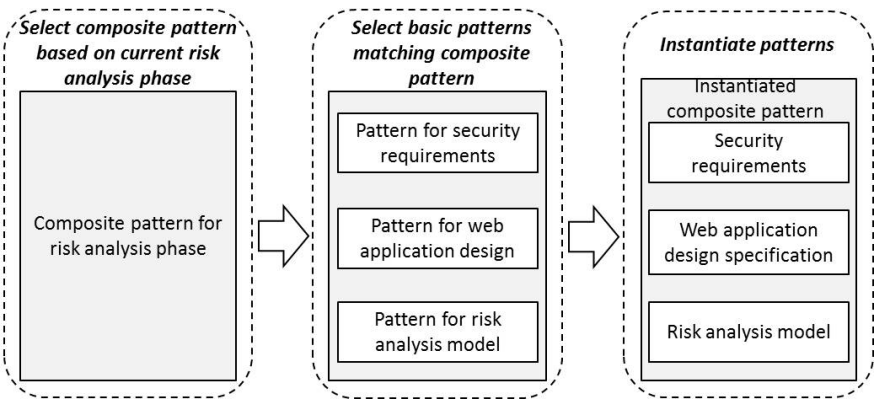


Fig. 2. Usage of pattern language

The process captured by Figure 2 may be summarized as follows:

—Composite pattern
A composite pattern mainly addresses one of the main phases or sub-phases in the security risk analysis process where the partial security risk picture is derived from combined use of security requirements patterns, web application design patterns and risk analysis modelling patterns. In each phase of the risk analysis, as shown in the leftmost part of the Figure 2, there may be one or more composite patterns that risk analysis staff can follow to select proper consitituent patterns for its instantiation. A composite pattern may be instantiated by basic patterns as well as other consitituent composite patterns. Combined usage of instantiated results from basic patterns or composite patterns enables the instantiation of a composite pattern to get relevant risk analysis results.

—Security requirement pattern
For a given web application, we address the expected security objective by selecting relevant security requirement patterns from the basic security requirement pattern library, for example, a pattern for data confidentiality in the cloud environment, a pattern for service availability in Intranet, a pattern for data integrity of mobile devices, etc. There may exist several security requirement patterns according to different security objectives, and

overlap may happen since many security features of web applications are related to each other. By instantiating the selected patterns, we get concrete security requirements for the web application being analysed.

—Web application design pattern
A proper risk analysis usually requires in-depth understanding of the web application design. We can thus use web application design patterns to describe the application to be analysed. They are usually selected and instantiated based on the concrete security requirements instantiated from the selected security requirements patterns. When using a selected web application design pattern from the library, we may consider the security risks that arise from the architectural view of the business, the underlying logic, the deployment and the data that are described in one or several basic web application design patterns. Various architecture design specifications can be obtained after instantiating the selected design patterns.

—Risk analysis modelling pattern
Risk analysis models are defined according to the general risk analysis process presented in ISO 31000. They can for example be described and visualized using the CORAS language [Lund et al. 2011]. One could choose the relevant risk analysis modelling pattern and get the right risk model to use according to the risk analysis phase of relevance. The corresponding risk analysis models for the relevant web application analysed can be obtained after instantiation of the risk analysis modelling pattern.

There will be a library of composite patterns specialized for the different phases of the risk analysis process and the various contexts of analysis. For each basic pattern type, there will be a pattern library consisting of different patterns, which capture best practices. There are libraries of security requirement patterns, web application design patterns and risk analysis modelling patterns according to different security objectives, context of different web applications (e.g. functionality, environment, and platform) and various risk analysis activities. The proper patterns should be selected based on the risk analysis phase involved, security objectives and the context of the web application.

This security risk analysis pattern language can be used for web applications that either already exist or is in the process of being built. As long as one can extract the proper features of the application required by the selected patterns, one can perform necessary security risk analysis in all stages of the software development life cycle. Since it is usually of great importance to get the security risk picture right as early as possible, the effectiveness and efficiency of using this security risk analysis pattern language is particularly needed in the early phase of the software development life cycle.

4. AN ILLUSTRATION EXAMPLE FOR THE PATTERN LANGUAGE

To help readers to better understand the idea of our pattern language, we use a web application example to illustrate some existing patterns and their usage.

As shown in Figure 3[1], we consider a health monitoring web application employed by smart glasses for collecting and analysing health conditions based on data collected by a smart watch. The collected data can be displayed to the user in smart glasses in real time and uploaded to a cloud environment through the web application. The history of health data stored in the cloud environment is analysed periodically by a computing server according to pre-defined algorithms and patterns. Once a hazard threshold is reached, the user will receive warnings and suggestions in the smart glasses through this health monitoring web application.

The composite pattern as well as related basic patterns presented in the following sections are selected based on features in this web application according to the usage of the pattern language described in Section 3.

[1] Photos are adapted from Clip Art in Microsoft Powerpoint 2010.

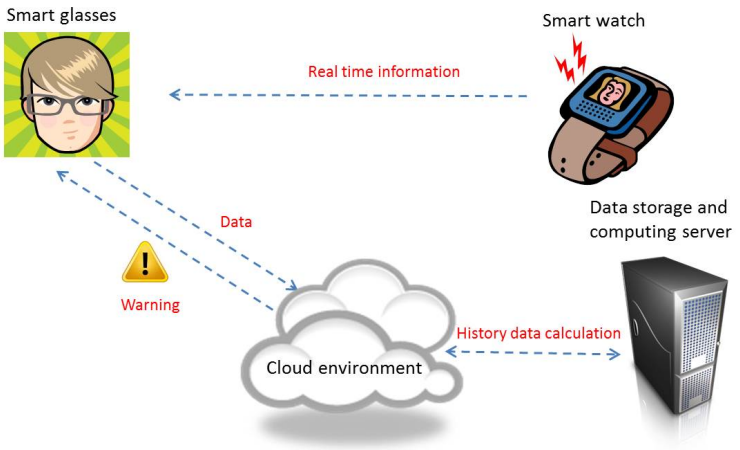


Fig. 3. A demonstration example

5. COMPOSITE PATTERN FOR RISK IDENTIFICATION OF WEB APPLICATIONS

A composite pattern is a kind of skeleton with place-holders for its constituent patterns, and it describes how the instantiation of the patterns selected for the place-holders may be composed. Composite patterns are expressed graphically and may refer to four different icons. Figure 4 presents these icons and their interpretations.

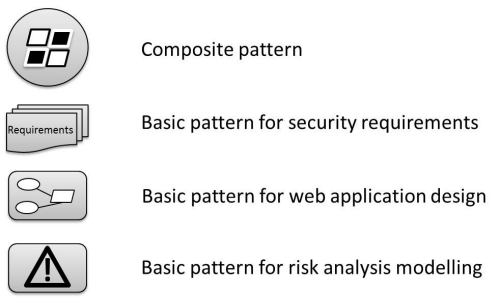


Fig. 4. Icons and their interpretations in pattern language

We have selected the composite pattern presented in Figure 5 for our example introduced in Section 4. From the figure, we can seet hat a composite pattern is divided in two by a horizontal line. The signature is specified above the horizontal line, while the body is specified below the horizontal line. The signature of the composite pattern in Figure 5 specifies "WebApp" (short for web application) and SecObj (short for security objectives) to be the external inputs and "Threat model" to be the external output. Moreoever, "Asset", "SecReq" (short for security requirement), and "ArcDes" (short for architecture design) are internal inputs and outputs.

The selection of constituent patterns and the instantiation order of this composite pattern as well as the relationships of each basic pattern are illustrated in the body below the straight line. By following the composition order represented by the solid arrow lines, we first select an appropriate security requirement pattern based on the current web application and the security objectives. Then web application design specifications can be derived with the aim to satisfy the obtained security requirements. The last pattern to be instantiated is the risk analysis

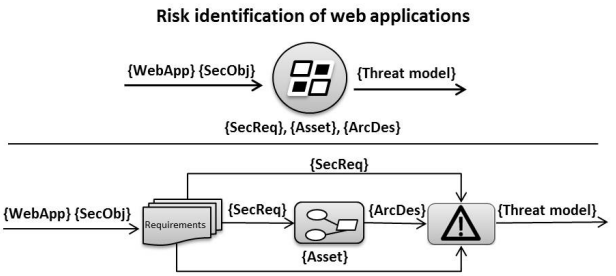


Fig. 5. Selected composite pattern

modelling pattern, which will use the outputs from both the security requirements and the web application design. The final result will then be an instantiated threat model for our specific web application.

6. SECURITY REQUIREMENT PATTERN FOR CONFIDENTIALITY OF DATA IN A CLOUD ENVIRONMENT

By considering a data-centric web application in a cloud environment, one basic security requirement pattern is confidentiality of data. It is desired that all confidential data collected, generated, transmitted to the web application and cloud environment, as well as stored on various devices should be secured and kept confidential. The confidential data should not be exposed by any means. A basic pattern for confidentiality of information in a cloud environment is given below.

Pattern name :

Confidentiality of data in a cloud environment

Classification :

Security requirement

Pattern signature :

"WebApp" (short for web application) is an input parameter representing the web application's features to be

analyzed.

"SecObj" (Short for security objectives) is an input parameter representing the security objectives need to be fullfilled in the risk analysis.

"SecReq" (short for security requirement) and "Asset" are output parameters representing security requirement for data confidentiality in a cloud environment and security assets to be protected.

Motivation :

The intent of this pattern is to address the basic security problem domains regarding data confidentiality in a cloud environment and to specify the relevant requirements as a basis for secured application engineering and risk analysis.

Context of use :

This pattern should normally be used to address the confidentiality of sensitive data affiliated to a web application in a cloud environment with functionality to collect, generate, display, communicate or use of sensitive and critical information. It is suitable to apply in the following situations:

—when it is assumed that the main functionalities of the web application require extensive and multiple ways of sensitive information usage, and the information is mostly personal or other confidential information which requires high-level of security assurance.

—when it is assumed that a web application is designed following the three-layered architecture design principle [Microsoft 2008] in a cloud environment.

—when it is assumed that one can structure the web application's requirements following the three-layered architecture design.

Problem :

How to derive and structure security requirements for confidentiality of data in a cloud environment for a web application?

Solution :

For the three-layered web application architecture [Patterns and Team 2009], the main problems relevant for establishing security requirements for a web applications are:

—Possible security threats and vulnerabilities in the presentation layer.

—Possible security threats and vulnerabilities in the business layer in the cloud.

—Possible security threats and vulnerabilities in the database layer in the cloud.

—Possible security threats and vulnerabilities that exist in the communication of information within each layer.

—Possible security threats and vulnerabilities that exist in the communication of information among different layers.

The problem frame [Jackson 2001], inspired diagram in Figure 6 illustrates the relationships between the web application, its different layers, the security objective, the security requirement and asset.

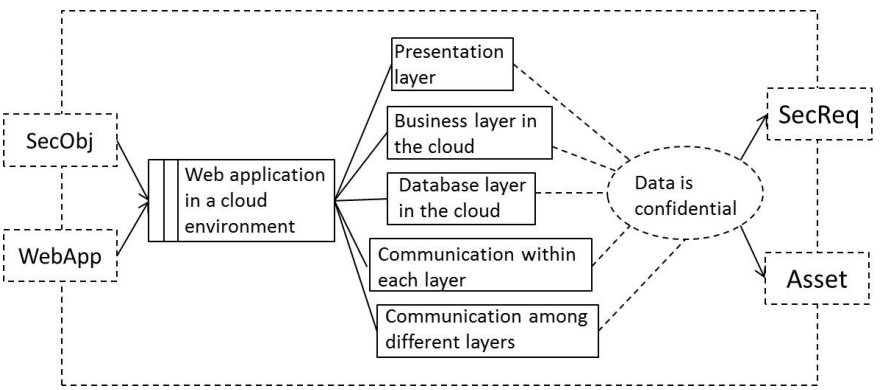


Fig. 6. Problem diagram for Confidentiality of data

The different diagram entities above are defined according to the problem frames notation [Jackson 2001] and are interpreted in Table II.

The generic requirements corresponding to the output parameter "SecReq" for confidentiality of data in a cloud environment for web application are described in Table III. The asset corresponding to the output parameter "Asset" is confidentiality of data.

Instantiation rule :

Table II. Interpretation of notions in the problem diagram in Figure 6

Notion	Interpretation
WebApp	Input parameter that represents the web application in the cloud to be analysed.
SecObj	Input parameter that represents the security objectives to be fullfilled.
SecReq	Output parameter that represents the set of security requirements that are derived by instantiating the pattern according to a specific context of a web application in the cloud.
Asset	Output parameter that represents the asset that one would like to protect.
Web application in a cloud environment	The web application that represents the target of analysis.
Presentation layer	The user interfaces and the interactions with the user of the application.
Business layer in the cloud	An abstract layer that controls the application's functionality by performing detailed processing according to internet protocols.
Database layer in the cloud	The layer that hosts the database in the cloud environment for the application.
Communication within each layer	The information flows exchanged within each layer.
Communication among different layers	The information flows exchanged among different layers.
Data is confidential	The main problem that forms the basis to derive security requirements.

Table III. Security requirements with respect to data confidentiality in a cloud environment

ID	Requirement	Description
R.1	The access to the web application should be secure on the client side, namely in the presentation layer.	Access to the application should be restricted in such a way that non-authorized people should not obtain access.
R.2	The business logic of the web application should work properly and handle confidential information in a secure way. This applies to both the business layer locally and the business layer in the cloud.	The dataflow through the web application server should be controlled in such a way that confidential data cannot be exposed to un-authorized people.
R.3	The confidential data stored in the database should be secure. This applies to both information security and physical security.	Data should be stored in a secure way and in a secure location.
R.4	The dataflow within each layer should be secure. This applies to both communication locally and communication in the cloud.	Dataflow within each layer should be encrypted and communicated in a secure way.
R.5	The dataflow between each layer should be secure. This applies to both communication locally and communication in the cloud.	Dataflow between different layers should be encrypted and communicated in a secure way.

Every requirement instantiating "SecReq" should be an instance of an abstract requirement defined in Table III. An asset instantiating "Asset" should be a data asset one would like to keep confidential.

Participants :

Decision maker

Related pattern :

A security pattern format usable within software development life cycle and specified security requirement patterns for accounting, access control, identification, and authentication are developed in [Schumacher 2006]. In [Withall 2007], we can also find some requirement patterns that are related to security including access control (registration, authentication, authorization), audit (chronicle), and some aspects of privacy (archiving,comply-with-standard).

Known uses :

There are no known uses of this pattern entirely, but the benefits of the pattern are partly presented in some pattern examples. Problem frames are used in a pattern based security engineering process for requirement analysis of a secure remote display system [Hatebur et al. 2007b] [Hatebur et al. 2007a]. Three-Tier Architecture pattern is also mentioned and used in [Aarsten et al. 1996] [Kircher and Jain 2001] [Microsoft patterns & practices developer center 2013], which offers significant advantages for application analysis and design.

7. WEB APPLICATION DESIGN PATTERN OF WEB APPLICATIONS FOR DATA COLLECTION AND ANALYSIS IN A CLOUD ENVIRONMENT

By considering a type of web application for data collection and analysis that uses a server and a database in the cloud, one basic web application architecture design pattern is the one following below.

Pattern name :

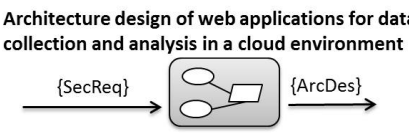
Architecture design of web applications for data collection and analysis in a cloud environment

Classification :

Web application design

Pattern signature :

"SecReq"(short for security requirement) is an input parameter representing security requirement for the web



application to be analyzed, which is usually provided as the output from a security requirement pattern.
"ArcDes"(short for architecture design) is an output parameter representing architecture design of web applications for data collection and analysis.

Motivation :

The intent of this pattern is to show the architecture design views of a web application for data collection and analysis in a cloud environment from a structural and behavioral perspective. It also serves as a basis for secured web application engineering and risk analysis.

Context of use :

The pattern should normally be used to show abstract views of the web application architecture in a cloud environment. It is suitable to be used in the following context:

—when there is a need to design abstract architecture for a web application in a cloud environment to satisfy security requirements.

—when a web application architecture view serves as a basis for security risk analysis in the early design or development phase.

—when the main functionality of a web application includes data collection and data analysis.

Problem :

What are security architecture design views of web applications for data collection and analysis in a cloud environment?

Solution :

Based on the relevant security requirements as input, we may address behavioural parts and structural parts of the web application architecture design. Considering the design from a behavioural perspective, use case diagrams and activity diagrams can be used to illustrate the web application architecture from business and logical process point of view. Considering the design from a structural perspective, deployment diagrams and class diagrams can be used to illustrate design from deployment and data point of view.

—Use case diagram from a business point of view.
The use case diagram in Figure 7 shows the basic functionality that a user would like to use, e.g. collect data, upload data to the cloud and download data from cloud through a smart device (e.g. PC, mobile, pad, glasses) that holds the web application. They are tagged by a fish symbol representing the sub-function level. The main function of such a type of web application is determined by the ways of using the results from the data

analysis, which is tagged with cloud symbol representing high level functionality. Each use case will further help with identifying proper risk elements in a certain risk analysis activity, such as threats, threat scenarios, and vulnerabilities and so on.

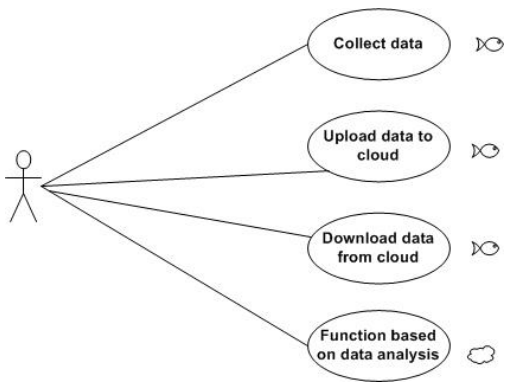


Fig. 7. Use case diagram - Business point of view

—Activity diagram from logical process point of view.
The activity diagram in Figure 8 illustrates the logical process of a user using a type of web application for data collection and analysis through a data collection device and smart device in a cloud environment to serve a specific purpose. The dashed line illustrates how an instantiation of this activity diagram should be adjusted according to the context of the specific web application to be analysed.

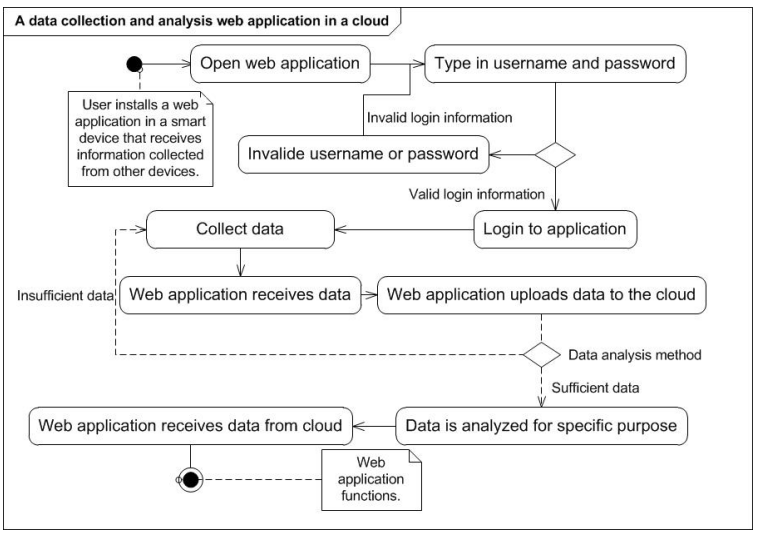


Fig. 8. Activity diagram - Logical process point of view

—Deployment diagram from deployment point of view.
The deployment diagram in Figure 9 shows how this web application in the cloud should be deployed following the three-layered web architecture design principle [Patterns and Team 2009]. The smart devices for data

collection and web application hosting will be linked together to conduct information communication through Internet or Bluetooth. The data communicated between the the smart devices should be kept secrect. Communications between the smart device and application server in the cloud use internet with a firewall link. The database in the cloud and application servers in the cloud are linked through an internet network link based on LAN where the data is required to be highly secure.

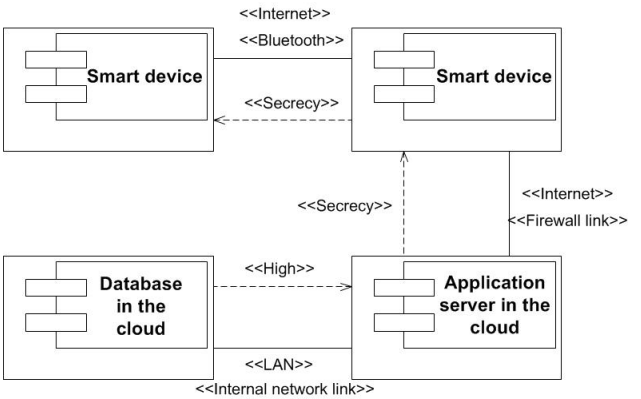


Fig. 9. Deployment diagram - Deployment point of view

—Class diagram from data point of view.

The class diagram in Figure 10 shows the data types related to the usage of this type of web application and the relationships among data and relevant components. This type of application has a set of databases in the cloud, and the databases in the cloud support the applications by storing data, transmitting data, and analysing data, etc. There can be different types of data, such as health data, personal data or other types of data with requirements for confidentiality.

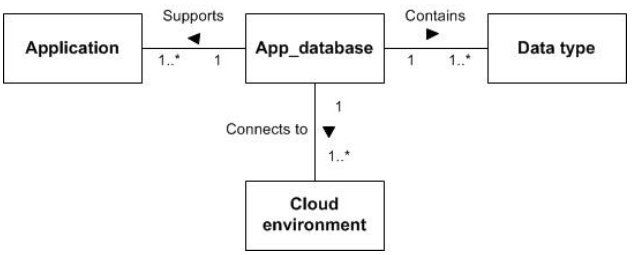


Fig. 10. Class diagram - Data point of view

Instantiation rule :

An artefact "ArcDes" is a set of architecture design views instantiating the business, the logical process, the deployment and the data point of view given in Figure 6, 7, 8 and 9.

For a web application under analysis, the sub functions and main functions are instances of the functions in the business process point of view in Figure 6. The internal logical process is an instance of the logical process point of view in Figure 7. The smart device to collect data and to use the web application are instances of smart devices in the deployment point of view in Figure 8. The data need to be protected is an instance of data type in the data

point of view in Figure 9.

Participants :

System architect

Related patterns :

There is no known architecture design patterns directly related to data collection and analysis web application as presented here. But architectural patterns in [Taylor et al. 2007], security architectures in SEPP (Security Engineering Process with Patterns) [Schmidt et al. 2011], S&D patterns in SERENITY [Gallego-Nicasio et al. 2009] and security design patterns in [Fernandez 2004] can also be considered when making architecture design with respect to security.

Known uses :

The 4+1 view model of architecture [Kruchten 1995], the viewpoints presented in [Garland and Anthony 2003] and experiences using viewpoints for information system in [Woods 2004] partly presents the usefulness and acceptance of architecture design in terms of different views in this pattern.

8. RISK ANALYSIS MODELLING PATTERN OF WEB APPLICATIONS FOR DATA COLLECTION AND ANALYSIS IN A CLOUD ENVIRONMENT

Pattern Name :

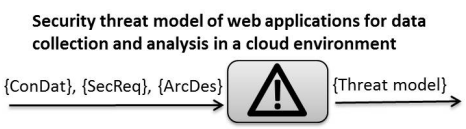
Security threat model of web applications for data collection and analysis in a cloud environment

Classification :

Risk analysis modelling

Pattern signature :

"ConDat"(short for confidentiality of data) is an input parameter representing the asset to be protected and is



therefore a special case of an "asset" parameter, which is usually provided as the output from a security requirement pattern.

"SecReq"(short for security requirement) is an input parameter representing security requirements for data confidentiality in a cloud environment, which is usually provided as the output from a security requirement pattern.

"ArcDes"(short for architecture design) is an input parameter representing an architecture design of web applications for data collection and analysis in a cloud environment, which is usually provided as the output from a web application design pattern.

"Threat model" is an output parameter for this pattern that describes the basis for modelling identified risks in terms of unwanted incidents of "ConDat", threat scenarios, threats and vulnerabilities according to "SecReq" and "ArcDes".

Motivation :

To identify, recognize and describe risks related elements that form the basis of threat modelling.

Context of use :

This pattern should normally be used to identify risk related unwanted incidents, threat scenarios, threats and vulnerabilities of targeted assets that one would like to protect. It is suitable to be used in the following situations:

—when one would like to identify security risks of web applications for data collection and analysis in a cloud environment.

—when one would like to design a secure web application starting from risk identification following a risk-based approach.

—when the results of the risk identification can serve different other purposes (e.g. eliciting security requirements [Braz et al. 2008], assisting with security testing [Li 2012]).

Problem :

How to identify risks with respect to confidentiality of data for a type of web application for data collection and analysis in a cloud environment? What are risks related elements, including unwanted incidents, threat scenarios, threats and vulnerabilities?

Solution :

As shown in Figure 11, OWASP[2] or CVE[3] is used to identify risks for a web application according to asset

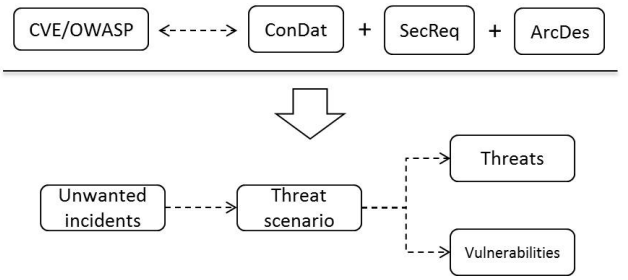


Fig. 11. Risk identification

of data confidentiality, security requirements and architecture design of a web application. By matching among CVE/OWASP and these input parameters, one can derive unwanted incidents, threat scenarios, threats, and vulnerabilities accordingly.

Various threats are presented in Table IV, including accidental human threats(AHT), deliberate human threats

Table IV. Threat codes of web application for data collection and analysis in the cloud

Threat	User	Employee of cloud provider	Malicious customer	Attacker	Hacker	Insider of cloud provider	Business management of cloud provider
Threat code	AHT1	AHT2	DHT1	DHT2	DHT3	DHT4	NHT1

Table V. Vulnerability codes of web application for data collection and analysis in the cloud

Vulnerability	Insufficient security awareness	Lack of competence	Insufficient security patch for virtualization from cloud provider	Insufficient physical protection	Work progress is not aligned with policy	Insufficient access control	Poor encryption	Insufficient virus protection	Poor multiple identity management of one employee
Vulnerability code	V1	V2	V3	V4	V5	V6	V7	V8	V9

[2]www.owasp.org

[3]cve.mitre.org

Table VI. Security vulnerability and threat scenario matrix of web application for data collection and analysis in the cloud

ASSET	Condat (Confidentiality of data)					
UWANTED INCIDENT	Confidential data is exposed					
THREAT SCENARIO	Data collection device is lost.	Smart device is lost.	Resources of different cloud customers are not well isolated logically.	Access to server and database in the cloud is compromised.	Access to server and database in the cloud is misused.	An employee's access cannot be shut off automatically, following termination of an employee.
Vulnerability	V1,V6	V1,V6	V1,V2,V5,V7	V2,V3,V4,V5,V7	V3,V4,V6,V7,V8	V1,V5,V6,V9
Threat	AHT1	AHT1	AHT2,DHT1	DHT3	DHT2,DHT4	NHT1

(DHT) and non-human threats (NHT). Each threat is assigned a threat code. Various vulnerabilities are presented in Table V, where a vulnerability code is assigned to each vulnerability. By exploiting vulnerabilities, a threat can initiate a threat scenario, and one threat scenario may also initiate another threat scenario. When a threat scenario happens, an unwanted incident may occur that harms the protected asset. Table VI presents a matrix relating unwanted incidents, threat scenarios, threat and vulnerabilities, which address confidentiality risks of web applications for data collection and analysis in a cloud environment.

One may interpret the relation corresponding to DHT3 and V3 in Table VI in this way: " The *Hacker* threat could cause the threat scenario *Access to server and database in the cloud is compromised* to occur due to the vulnerability of *Insufficient security patch for virtualization from cloud provider*. When the unwanted incident *Confidential data is exposed* happens, the asset *Confidentiality of data* is harmed.".

Instantiation rule :

An artefact instantiating *Threat model* can be structured according to Table VI.

Participants :

Risk analyst, security engineer, software architect, software developer, project manager, etc.

Related pattern :

Some risk patterns are used in [Miler and Gorski 2004] to help with risk identification, where a set of classes of risk events are defined and the different risk events are combined into risk patterns.

Known uses :

There are no known uses of this pattern in its entirety, but the essential solution presented here has been partly applied in several examples. For example, model driven risk analysis method CORAS applied in an evolving critical infrastructures in [Solhaug and Seehusen 2013], fault trees for secure system design and analysis [Brooke and Paige 2003], attack tree risk analysis method applied in power system control networks [Ten et al. 2007] and so on.

9. CONCLUSION AND FUTURE WORK

The pattern language presented in this paper aims to provide the basis for a light weighted method for security risk analysis for web applications, especially in the early phase of software development life cycle. We have illustrated the usage of the pattern language and provided examples of various kinds of patterns. More specific, we have presented a composite pattern, a basic security requirement pattern, a basic web application design pattern and a basic risk analysis modelling pattern.

Future work would be to expand the libraries of patterns according to different types of web applications and to define a formal syntax for it. In addition, the work will be evaluated in proper cases.

Acknowledgments:

This work has been conducted as a part of the DIAMONDS (201579/S10) project funded by the Research Council of Norway, the NESSoS(256980) network of excellence funded by the European Commission within the 7th

Framework Programme, as well as a part of the RASEN (316853) project funded by the European Commission within the 7th Framework Programme. The authors also acknowledge the shepherding process by Eduardo Fernandez.

REFERENCES

AARSTEN, A., BRUGALI, D., AND MENGA, G. 1996. Patterns for three-tier client/server applications. *Pattern Languages of Programs (PloP '96)*.

ALEXANDER, C. 1977. *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press.

BRAZ, F. A., FERNANDEZ, E. B., AND VANHILST, M. 2008. Eliciting security requirements through misuse activities. In *International Workshop on Database and Expert Systems Application (DEXA '08)*. 328–333.

BROOKE, P. J. AND PAIGE, R. F. 2003. Fault trees for security system design and analysis. *Computers & Security 22,* 3, 256–264.

BUSCHMANN, F., MEUNIER, R., ROHNERT, H., SOMMERLAD, P., AND STAL, M. 1996. Pattern-oriented software architecture volume 1: A system of patterns. Wiley.

DOUGHERTY, C., SAYRE, K., SEACORD, C., SVOBODA, D., AND TOGASHIO, K. 2009. Secure design patterns. Tech. Rep. CMU/SEI-2009-TR-010, Software Engineering Institute, Carnegie Mellon.

FERNANDEZ, E. 2013. *Security Patterns in Practice: Designing Secure Architectures Using Software Patterns*. Wiley.

FERNANDEZ, E. B. 2004. A methodology for secure software design. In *Software Engineering Research and Practice (SERP '04)*. 21–24.

GALLEGO-NICASIO, B., MUNOZ, A., MANA, A., AND SERRANO, D. 2009. Security patterns, towards a further level. *International Conference on Security and Cryptography (SECRYPT 2009)*, 349–356.

GAMMA, E., HELM, R., JOHNSON, R., AND VLISSIDES, J. 1995. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley.

GARLAND, J. AND ANTHONY, R. 2003. *Large Scale Software Architecture*. John Wiley.

HATEBUR, D., HEISEL, M., AND SCHMIDT, H. 2007a. A pattern system for security requirements engineering. *Seventh International Conference on Availability, Reliability and Security (ARES '07)*, 356–365.

HATEBUR, D., HEISEL, M., AND SCHMIDT, H. 2007b. A security engineering process based on patterns. In *18th International Workshop on Database and Expert Systems Applications (DEXA '07)*. 734–738.

HAUGE, A. AND STØLEN, K. 2011. SACS-A pattern language for safe adaptive control software. In *18th Conference on Pattern Languages of Programs (PLOP'11)*.

International Standards Organization 2009a. *ISO31000:2009(E), Risk management – Principles and guidelines*. International Standards Organization.

International Standards Organization 2009b. *ISO/IEC 27000:2009(E), Information technology – Security techniques – Information security management systems – Overview and vocabulary*. International Standards Organization.

JACKSON, M. 2001. *Problem Frames: Analyzing and Structuring Software Development Problems*. Addison-Wesley.

KIENZLE, D. M. AND ELDER, M. C. 2002. Final technical report: Security patterns for web application development. *DARPA, Washington DC*.

KIRCHER, M. AND JAIN, P. 2001. The three-tier architecture pattern language design fest. In *European Conference on Pattern Languages of Programs (EuroPLoP '01)*. 575–580.

KRUCHTEN, P. 1995. The 4+1 view model of architecture. *Software, IEEE 12,* 6, 42–50.

LI, Y. 2012. Conceptual framework for security testing, security risk analysis and their combinations. In *9th Workshop on Systems Testing and Validation (STV'12)*. Fraunhofer, 17–21.

LUND, M., SOLHAUG, B., AND STØLEN, K. 2011. *Model-Driven Risk Analysis: The CORAS Approach*. Springer.

MICROSOFT PATTERNS & PRACTICES DEVELOPER CENTER. 2013. Tiered distribution. `http://msdn.microsoft.com/en-us/library/ff647195.aspx`. Accessed on 02.05.2013.

MILER, J. AND GORSKI, J. 2004. Risk identification patterns for software projects. *Foundations of Computing and Decision Sciences 29,* 1-2, 115–132.

PATTERNS, M. AND TEAM, P. 2009. *Microsoft Application Architecture Guide*. Microsoft Press Series. Microsoft Press.

RIEHLE, D. 1997. Composite design patterns. *ACM SIGPLAN Notices 32,* 10, 218–228.

ROMANOSKY, S. 2001. *Security Design Patterns Part 1*. Carnegie Mellon.

SCHMIDT, H., HATEBUR, D., AND HEISEL, M. 2011. A pattern-based method to develop secure software. In *Software Engineering for Secure Systems: Industrial and Research Perspectives*. IGI Global, 32–74.

SCHUMACHER, M. 2006. *Security patterns : integrating security and systems engineering*. John Wiley & Sons.

SOLHAUG, B. AND SEEHUSEN, F. 2013. Model-driven risk analysis of evolving critical infrastructures. *Journal of Ambient Intelligence and Humanized Computing*, 1–18.

STEEL, C., NAGAPPAN, R., AND LAI, R. 2006. *Core Security Patterns: Best Practices and Strategies for J2EE,Web Services, and Identity Management*. Prentice Hall core series. Upper Saddle River, N. J. Prentice Hall PTR.

TAYLOR, R. N., MEDVIDOVIC, N., AND DASHOFY, E. M. 2007. *Software Architecture: Foundations, Theory and Practice*. Addison-Wesley.

TEN, C.-W., LIU, C.-C., AND GOVINDARASU, M. 2007. Vulnerability assessment of cybersecurity for scada systems using attack trees. In *Power Engineering Society General Meeting, 2007. IEEE*. 1–8.

WITHALL, S. 2007. *Software Requirement Patterns*. Microsoft Press.

WOODS, E. 2004. Experiences using viewpoints for information systems architecture: An industrial experience report. In *Software Architecture*. Springer, 182–193.

YODER, J. AND BARCALOW, J. 1998. Architectural patterns for enabling application security. *Urbana 51*, 61801.

YOSHIOKA, N., WASHIZAKI, H., AND MARUYAMA, K. 2008. A survey on security patterns. In *Progress in Informatics No.5*. 35–47.