Towards a Pattern Language for Security Risk Analysis of Web Applications

Yan Li, SINTEF ICT, University of Oslo
Ragnhild Kobro Runde, University of Oslo
Ketil Stølen, SINTEF ICT, 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 model pattern. The pattern language is intended to be used as a guideline to capture the security risk picture of a web application in a lightweight way with extended user groups, especially in the early phase of the software development life cycle.


General Terms: Pattern language

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

ACM Reference Format:

1. INTRODUCTION

For the past decades or so, the web has become the main platform for the purposes of 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 refers to any program that is accessed over a network connection using HTTP (Hyper Text Transport Protocol). It can be either web browser-based or client-based, where a small part of the program is downloaded to a user’s desktop, but processing is done over the Internet on an external server (Adapted from [Janssen 2013]).

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). More and more individuals and commercial companies are using web applications to serve different purposes (e.g. governance, banking, shopping). On the one hand, web applications offer many opportunities and much flexibility, and our life benefits much from various web-based services. 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 threats. Therefore, it is necessary to analyze security risk to ensure the safety and normal operation of web applications.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission. A preliminary version of this paper was presented in a writers’ workshop at the 20th Conference on Pattern Languages of Programs (PLoP). PLoP’13, October 23-26, Monticello, Illinois, USA. Copyright 2013 is held by the author(s). ACM 978-1-4503-0107-7
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 [International Standards Organization 2009]). To deal with new security threats and 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 business and 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 1997]. Eight patterns as architectural and procedural guideline was proposed in [Romanosky 2001]. Afterwards, a group of 29 security patterns categorized as structural and procedural pattern for web application development was introduced in [Kienzle and Elder 2003]. 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 architectural level, design level and implementation level of software is introduced later in [Dougherty et al. 2009]. Recently a book [Fernandez 2013] has been published about how to incorporate security into every phase of the software lifecycle 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 context 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 based on context of web applications, aiming to provide a light weighted way of doing 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 and pattern selection of the pattern language. Section 4 describes a composite pattern example that serves the basis for selecting basic patterns. Section 5, Section 6 and Section 7 present examples of basic pattern for security requirement, web application design and risk analysis model respectively. Section 8 presents the usage of a composite pattern together with its relevant basic patterns in a web application example. The work is concluded and discussed 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 analyzing the security risk picture of web applications based on its con-
text (e.g. functionality, platform for development, environment for maintenance). A pattern here defines a reusable and extendable solution for a specific problem occurring in security risk analysis for a web application; it can either be a basic pattern or a composite pattern that is composed of several basic patterns or composite patterns. A basic pattern is an element pattern that can be used to compose a composite pattern by addressing a partial problem or sub-process, while a composite pattern is composed of a set of basic patterns / composite patterns in order to solve a specific task in the risk analysis process.

As shown in Figure 1, 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 model. Each represents its own dimension, namely:

— Basic principles of information security standardized in ISO / IEC 27000 [ISO 27000 2009] (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 model (e.g. Asset model, Threat model, Risk model, Treatment model).

Fig. 1. Dimensions of different patterns

In security risk analysis for web applications, a composite pattern mainly refers to one of the main phases or sub-phases in the security risk analysis process where the security risk picture is derived from combined use of security requirements patterns, web application design patterns and risk analysis model patterns. A composite pattern requires an instantiation order for composed patterns, so it is a type of process pattern. Each composite pattern addresses a specific aspect of the risk analysis process, based on which we select relevant basic patterns that fit with the structure of the composite pattern and the features of web application analyzed. The composite pattern here mainly address recurring solutions that solve security risk analysis problems, which is different from the composite pattern defined in [Riehle 1997] where a composite pattern mainly address a concrete recurring solution for a design problem. Outputs with various risk analysis elements (e.g. Threat, Vulnerability, Threat scenario, Unwanted incidents, Assets, Risk, Treatment) can be obtained as a basis for risk analysis conducted for a specific type of web application based on composite patterns and related basic patterns.

For a given pattern, the input parameter is defined based on the context of the web application, the output parameter can be results of instantiating the pattern, inspired by SaCS [Hauge and Stølen 2011]. A composite pattern can be instantiated following the instantiation rule of the relevant basic patterns. The documentation of a basic pattern follows a defined format, while the documentation of composite pattern usually includes a
composite pattern signature and description, as presented in Section 4. The documentation format of a basic pattern is organized and interpreted in Table I:

<table>
<thead>
<tr>
<th>Notion</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pattern name</td>
<td>The name of the pattern.</td>
</tr>
<tr>
<td>Classification</td>
<td>The category of the pattern, i.e. security requirement.</td>
</tr>
<tr>
<td>Motivation</td>
<td>The motivation for applying the pattern.</td>
</tr>
<tr>
<td>Context of use</td>
<td>In which situation the pattern applies as well as the usage of the pattern.</td>
</tr>
<tr>
<td>Problem</td>
<td>The problem that the pattern addresses.</td>
</tr>
<tr>
<td>Solution</td>
<td>The solution presented by this pattern, including the expected input parameter and output parameter.</td>
</tr>
<tr>
<td>Instantiation rule</td>
<td>The rules to instantiate the pattern.</td>
</tr>
<tr>
<td>Participants</td>
<td>The relevant human participants to be consulted when using this pattern.</td>
</tr>
<tr>
<td>Related patterns</td>
<td>The relationship between this pattern and other patterns.</td>
</tr>
<tr>
<td>Known uses</td>
<td>The known uses of the pattern, either in preliminary or modified form.</td>
</tr>
</tbody>
</table>

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

By defining the best practices and effective solutions to address 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 analyzed. In addition, by using the pattern language as a guideline, both expert and non-expert groups of risk analysis staff should be able to carry out risk analysis for the web application at either business level or technical level.

As shown in Figure 2, when using this pattern language, one may first select a composite pattern to assist with basic pattern selection according to the general risk analysis process described in ISO 31000 [International Standards Organization 2009]. A composite pattern defines the basic/composite pattern types for composition and the instantiation order of the patterns. Based on selected composite pattern, the suitable patterns can be selected and combined to form a composite pattern. By instantiating the selected patterns following the instantiation order defined in composite pattern, the composite pattern is instantiated. One can thus obtain the concrete security requirements, web application design specification, relevant risk analysis models as well as the specific risk analysis related results. The usage of composite pattern as well as each basic pattern is described below:

---

Towards a Pattern Language for Security Risk Analysis of Web Applications — Page 4
—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 model patterns. In each phase of the risk analysis, as shown in the left most part of the Figure 2, there may be one or more composite pattern that risk analysis staff can follow to select proper patterns for composition before further usage. A composite pattern may be composed of basic patterns as well as other 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 could address the expected security objective by selecting relevant security requirement pattern 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 device, etc. There may exist several security requirement patterns according to different security objectives, and overlap may happen as well since many security features of web applications are related to each other. By instantiating the selected pattern, we get concrete security requirements for the web application analyzed.

—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 pattern to describe the application to be analyzed. It is usually selected and instantiated based on the concrete security requirements instantiated from 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 process, 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 model pattern
Risk analysis models here are mostly defined according to general risk analysis process presented in ISO 31000, which for example can be described and visualized using the CORAS language [Lund et al. 2011]. One could choose the relevant risk analysis model pattern and get the right risk model to use according to the risk analysis phase they are involved in. The corresponding risk analysis models for a web application analyzed can be obtained after instantiation of the risk analysis model pattern.

In the general security risk analysis process consisting of establishing the context, risk identification, risk estimation, risk evaluation and risk treatment, there will be composite patterns specialized for different web applications in each step. For each basic pattern type, there will be a pattern library consisting of different patterns. There are libraries of security requirement patterns, web application design patterns and risk analysis model 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 web application’s context.

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 software development life cycle. Since it is usually of great importance to get security risk picture as early as possible, the effectiveness and efficiency of using this security risk analysis pattern language is particularly needed in the early phase of software development life cycle.
4. COMPOSITE PATTERN FOR RISK IDENTIFICATION OF WEB APPLICATIONS FOR DATA COLLECTION AND ANALYSIS IN A CLOUD ENVIRONMENT

A security risk analysis composite pattern for web applications is composed of patterns that are combined and organized by rules. A well-formed risk analysis composite pattern is composed of a set patterns addressing the principal notions within three basic pattern types, namely security requirements, web application design and risk analysis model. As shown in the composite pattern signature below, three basic patterns described later in Section 5, Section 6 and Section 7 work in the illustrated process to identify risks for web applications of such type (Web application for data collection and analysis in a cloud environment).

The external input, output as well as internal inputs and outputs are illustrated above the straight line in the signature. "WebApp" (short for web application) is the input and "Threat model" is the output for the composite pattern externally, while "Asset", "SecReq" (short for security requirement), "ArcDes" (short for architecture design) are internal inputs and outputs.

The pattern selection and instantiation order of this composite pattern as well as the relationships of each basic pattern is illustrated below the straight line. The proper security requirement pattern, web application design pattern and risk model pattern should be selected based on the security objective, features of web application and instantiation order described in composite pattern signature. By following the composition order, we can firstly get security requirements and assets to be analyzed based on security objectives. Then a web application design specification can be derived with aim to satisfy security requirements obtained. Both security requirement and web application architecture design pattern provide the inputs for risk identification phase. The threat models of such applications could be identified by following the risk identification process in the pattern signature and relevant risk analysis model patterns.

5. 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 could be confidentiality of data. It is desired that all confidential data collected, generated, transmitted to 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 for malicious purpose by any means. A pattern for confidentiality of information in a cloud environment is given below as an example.

Pattern name: Confidentiality of data in a cloud environment
Classification: Security requirement
**Pattern signature:**

"WebApp" (short for web application) is an input parameter representing web application to be analyzed.

"SecReq" (short for security requirement) and "Asset" are output parameters representing security requirement for data confidentiality in a cloud environment and security asset 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 that 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 a web application require extensive and multiple ways usage of information, mostly personal and other confidential information, and all the information processed from web application is sensitive and it 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 web application's requirement following the three-layered architecture design if there is no specific design rules to follow.

**Problem:**

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

**Solution:**

For input parameter of "WebApp", one may consider web application's functionality, its environment for maintenance, platform for development, etc. to describe web application analyzed, depending on application's features addressed in the selected patterns based on a specific composite pattern. The web application analyzed can either already exist or it is being built, depending on the needs of security risk analysis throughout the development life cycle.

Confidentiality of data is considered as an output of "Asset". Concrete security requirements are considered as an output of "SecReq".

When deriving the output of "SecReq", one may consider the three-layered web application architecture [Patterns and Team 2009], from which the main problems relevant for establishing security requirements for a web applications could be:

— Possible security threats and vulnerabilities in presentation layer.

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

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

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

— Possible security threats and vulnerabilities that exist in communication of information among different layers.
A problem diagram can be used to illustrate the relationships among web application, different layers in web application, security objective, security requirement and asset, as shown in Figure 4. The different diagram entities above are defined according to the problem frames notation [Jackson 2001] and are interpreted in Table II. According to the problem diagram, the generic requirements for confidentiality of data in a cloud environment for web application are described in Table III, which captures the main challenges when approaching the problems presented in the problem diagram.

**Instantiation rule:**
A general security requirement set is given above according to problem diagram for information confidentiality, a concrete security requirement set can be derived by instantiating this requirement pattern.

An output artefact $\text{SecReq}$ instantiates the pattern if $\text{SecReq}$ is a set of security requirements. Every requirement in $\text{SecReq}$ is an instance of abstract requirement defined in the solution table according to problem diagram defined.

An output artefact $\text{Asset}$ instantiates the pattern if $\text{Asset}$ is the asset need to be protected in the target of analysis, each $\text{Asset}$ is an instance of data asset one would like to keep confidential.

![Fig. 4. Problem diagram for Confidentiality of data](image)

### Table II. Interpretation of notions in the problem diagram for confidentiality of data

<table>
<thead>
<tr>
<th>Notion</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>WebApp</td>
<td>The input parameter to the problem frame diagram which represents the web application in the cloud to be analyzed.</td>
</tr>
<tr>
<td>SecReq</td>
<td>The output parameter to the problem frame diagram which 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.</td>
</tr>
<tr>
<td>Asset</td>
<td>The output parameter to the problem frame diagram which represents the asset that one would like to protect.</td>
</tr>
<tr>
<td>Web application in the cloud</td>
<td>The application for the pattern to be applied that represents the main problem.</td>
</tr>
<tr>
<td>Presentation layer</td>
<td>The user interfaces and interacts with the user of the application.</td>
</tr>
<tr>
<td>Business layer in the cloud</td>
<td>An abstract layer that controls the application’s functionality by performing detailed processing according to internet protocols.</td>
</tr>
<tr>
<td>Database layer in the cloud</td>
<td>The layer hosts the database in the cloud environment for the application.</td>
</tr>
<tr>
<td>Communication within each layer</td>
<td>The information flows exchanged within each layer.</td>
</tr>
<tr>
<td>Communication among different layers</td>
<td>All forms of information flows exchange among different layers.</td>
</tr>
<tr>
<td>Data is confidential</td>
<td>The main objective that forms the basis to derive security requirements.</td>
</tr>
</tbody>
</table>
Table III. Security requirements of data confidentiality in a cloud environment

<table>
<thead>
<tr>
<th>ID</th>
<th>Requirement</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R.1</td>
<td>The access to the web application should be secure in the client side, namely presentation layer.</td>
<td>Access to the application should be restricted that non-authorized people should not obtain access.</td>
</tr>
<tr>
<td>R.2</td>
<td>The business logic of the web application should work properly and handle confidential information in a secure way. This applies to both business layer locally and business layer in the cloud.</td>
<td>The data flow through the web application server should be controlled in a secure way that confidential data cannot be exposed to un-authorized people.</td>
</tr>
<tr>
<td>R.3</td>
<td>The confidential data stored in database should be secure. This applies to both information security and physical security.</td>
<td>Data should be stored in a secure way and in a secure location.</td>
</tr>
<tr>
<td>R.4</td>
<td>The data flow communicated during each layer should be secure. This applies to both communication locally and communication in the cloud.</td>
<td>Data flow communicated during each layer should be encrypted and communicated in a secure way.</td>
</tr>
<tr>
<td>R.5</td>
<td>The data flow communication among each layer should be secure. This applies to both communication locally and communication in the cloud.</td>
<td>Data flow communicated among different layers should be encrypted and communicated in a secure way.</td>
</tr>
</tbody>
</table>

For example, according to abstract requirement R.1 in the table and web application analyzed, one may consider access control, access related human interaction design, and access related functionality to derive detailed security requirements.

**Participants:**
Decision maker

**Related pattern:**
This pattern can be used as inputs to security engineering for web applications, in which design is determined by security requirements. It can also support security risk analysis pattern in different risk analysis phases and provide inputs for composite patterns.

**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 and Brugali 1996] [Kircher and Jain 2001] [patterns and practices developer center ], which offers significant advantages for application analysis and design.

6. 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 use server and database in the cloud, one basic web application architecture design pattern example could be architecture design of web application for data collection and analysis in a cloud environment, the details of this pattern are given 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 a web ap-
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:
To show architecture design views of a web application type for data collection and analysis in a cloud environment from structure and behavior 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 web application architecture in a cloud environment with possibility to be more specified based on detailed context. 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 for satisfying security requirements.
— when a web application architecture views serve 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:
By following the relevant security requirements as inputs, we can in the outputs address behavior parts and structure parts of the web application architecture design. Considering the design from behavior perspective, use case diagram and activity diagram can be used to illustrate the web application architecture design from business and logical process point of view. Considering the design from structure perspective, deployment diagram and class diagram can be used to illustrate design from deployment and data point of view.

— Use case diagram from business point of view.
  The use case diagram in 5 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. The main function of such a type of web application is determined by the ways of using the results from data analysis. 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.

— Activity diagram from logical process point of view.
  The activity diagram 6 illustrates the logical process of 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. As illustrated by dashed line, the logic view represented by activity diagram should be adjusted according to the context of a specific web application to be analyzed.

— Deployment diagram from deployment point of view.
  The deployment diagram 7 shows how this web application in the cloud should be deployed following the three-layered web architecture design principle [Patterns and Team 2009]. The data collection device and smart device will be linked together to conduct information communication through Internet or Bluetooth. The data communicated between data collection device and smart device should be kept secrecy. Communications between smart device and application server in the cloud use Internet with firewall link. The database in the cloud and application in the cloud is linked through Internet network link based on LAN where the data is required to be high secrecy.
The class diagram 8 shows the data types related to the usage of this type of web application and the relationship 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 analyzing data, etc. There can be different types of data, such as health data, personal data or other types of data with high requirement for confidentiality.

**Instantiation rule:**
The architecture design of web applications for data collection and analysis proposed in this pattern is a general solution. The use of this pattern should be adjusted based on the context of a specific web application of this type.

An artefact ArcDes instantiates the design pattern if ArcDes is a set of architecture design views of the business, the logical process, the deployment and the data for a web application analyzed.
The sub functions and main functions of web application analyzed are instances of the functions in the business process point of view in Figure 6. The internal logical process of a web application is an instance of the logical process point of view in Figure 7. The device to collect data and smart device to use the web application are instances of data collection device and smart device 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:**
This pattern can be used as initial fulfillment of security requirement in architecture design. Together with other web application design pattern, a more detailed architecture design can be derived according to specific usage and context of a web application. 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. Patterns presented in A security risk picture can be drafted by its combined usage with security requirements pattern and risk analysis model pattern.

**Known uses:**
There are no know uses of this pattern in its entirety, but the benefits of this pattern is partly presented in similar patterns or methods for making architecture design of web application. The 4+1 view model of architecture [Kruchten 1995], the viewpoints presented in [J. and R. 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.
7. RISK ANALYSIS MODEL 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 model

**Pattern signature:**
"ConDat" (short for confidentiality of data) is an input parameter representing the asset to be protected, which is usually defined in a security requirement pattern. So this is a special instance of the Asset parameter in Figure 4.

"SecReq" (short for security requirement) is an input parameter representing security requirement for data confidentiality in a cloud environment, which is usually defined in a security requirement pattern.

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

"Threat model" is an output parameter for this pattern that describes the basis for modeling 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 forms the basis of threat model by conducting risk identification activity according to a given focus and scope.

**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.
—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 9, OWASP\(^1\) or CVE\(^2\) is used to identify risks for a web application according to asset of data confidentiality, security requirements and architecture design of web application. By matching among

---

\(^1\) [www.owasp.org](http://www.owasp.org)
\(^2\) [cve.mitre.org](http://cve.mitre.org)
CVE/OWASP and these input parameters, one can derive unwanted incidents, threat scenarios, threat, and vulnerabilities accordingly.

Various threats are presented in the threat Table IV, including human threat accidental (THA), human threat deliberate (THD), and non-human threat (TNH). Each threat is assigned a threat code. Various vulnerabilities are presented in vulnerability Table V, where a vulnerability code is assigned for each vulnerability. By exploiting vulnerabilities, a threat can initiate a threat scenario, and one threat scenario may also initiate another threat scenario. Due to a threat scenario happening, an unwanted incident may occur that harms the protected asset. A table describing matrix among unwanted incidents, threat scenarios, threat, and vulnerabilities is given in Table VI, which addresses confidentiality risks of web applications for data collection and analysis in a cloud environment.

One may interpret one of the risk pictures consisting THD3 and V3 from the table in this way: "A threat of Hacker could launch the threat scenario of Access to server and database in the cloud is compromised due to the vulnerability of Insufficient security patch for virtualization from cloud provider. When unwanted incident of Confidential data is exposed happens, the asset of Confidentiality of data is harmed."

**Instantiation rule:**
A general threat model for a data collection and analysis web application in the cloud is constructed by instantiating the various elements in Table VI using the relevant risk analysis modeling language (e.g. CORAS [Lund et al. 2011], Attack trees [Schneier 1999], Fault tree analysis (FTA) [International Electrotechnical Commission 2006]),
Table VI. Security vulnerability and threat scenario matrix of web application for data collection and analysis in the cloud

<table>
<thead>
<tr>
<th>ASSET</th>
<th>Confidentiality of data</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNWANTED INCIDENT</td>
<td>Confidential data is exposed</td>
</tr>
<tr>
<td>THREAT SCENARIO</td>
<td>Data collection device is lost.</td>
</tr>
<tr>
<td></td>
<td>Smart device is lost.</td>
</tr>
<tr>
<td></td>
<td>Resources of different cloud customers are not well isolated logically.</td>
</tr>
<tr>
<td></td>
<td>Access to server and database in the cloud is compromised.</td>
</tr>
<tr>
<td></td>
<td>Access to server and database in the cloud is mis-used.</td>
</tr>
<tr>
<td></td>
<td>An employee’s access cannot be shut off automatically, following termination of an employee.</td>
</tr>
<tr>
<td>Vulnerability</td>
<td>V1,V6</td>
</tr>
<tr>
<td>Threat</td>
<td>THA1</td>
</tr>
</tbody>
</table>

The usage of the pattern language for security risk analysis of web applications will be explained by instantiating the composite pattern described in Section 4 and corresponding basic patterns described in Section 5, Section 6 and Section 7, based on an application example described here.

As shown in Figure 10, we consider a health monitoring web application employed by smart glasses for collecting and analyzing health condition 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. All the history health data stored in a cloud environment is analyzed 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.

Assuming that the goal is to establish the context and identify confidentiality risks for the example application, we may instantiate the Composite Pattern named “Risk identification of web applications for data collection and analysis in a cloud environment” by selecting three concrete basic patterns, namely “Confidentiality of data in a...
cloud environment", "Architecture design of web applications for data collection and analysis in a cloud environment" and "Security threat model of web applications for data collection and analysis in a cloud environment".

The **Security Requirement Pattern** named "Confidentiality of data in a cloud environment" may be instantiated starting from the problem frame. We may associate the problem domain specified by Figure 4 such that the Web application in the cloud represents the health monitoring web application in the cloud. The Presentation layer represents the interface and the interactions of the smart glasses and the smart watch. The Business layer in the cloud represents confidential data processing protocols and control mechanism. The Communication within each layer and the Communication among layers represent the actual communication of confidential data (e.g. health data, personal data) when using this web application. We can then derive relevant requirements based on this context and detail the requirements table III further in Section 5, where the Asset is defined as confidentiality of data.

The **Web Application Design Pattern** named "Architecture design of web application for data collection and analysis in a cloud environment" may be instantiated by mapping the context of the web application example to proposed architecture design views of the business, the logical process, the deployment and the data. The Sub functions in Figure 5 may represent collect data from smart watch, upload collected data to cloud and smart glass download data from cloud. The Main function may represent monitor health condition based on data analysis. The Condition 1 in Figure 6 may represents normal health condition from data analysis and the Condition 2 may represents abnormal health condition from data analysis. The Data collection device and the Smart device in Figure 7 may represent the smart watch and the smart glasses respectively. The Data type in Figure 8 may represent health data and personal data used in the application example.

The **Risk Analysis Model Pattern** named "Security threat model of web applications for data collection and analysis in a cloud environment" may be instantiated by mapping the Threat model elements to usage scenario and maintenance scenario of the example application. In the Table VI, the Asset may represent "confidentiality of health data and personal data". The Unwanted incident may represent "health data and personal data is exposed". The instantiated Threat scenario may be "Smart watch is lost.", "Smart glasses is lost.", "Resources of different cloud customers are not well isolated logically.", "Access to server and database is compromised by the cloud provider of health monitor application.", "Access to server and database in the cloud is misused by the employee of cloud provider and An employee’s access cannot be shut off automatically during termination of contract.", etc. The Vulnerability and Threat in the table represents the actual vulnerabilities and threats for the application.
After mapping of these threat model elements, we may use relevant risk analysis modeling language (e.g. CORAS [Lund et al. 2011]) to visualize this threat model for the health monitoring application.

9. CONCLUSION AND FUTURE WORK

The pattern language presented in this paper aims to provide a light weighted method of doing security risk analysis for web applications, especially in the early phase of software development life cycle. It is intended to be used by both expert and non-expert groups of people for conducting risk analysis with possibility to have the effort adjusted and selected. We introduce the usage of this pattern language as well as the way of selecting suitable patterns based on the context of the web application analyzed. An example of composite pattern and three examples of basic security requirement pattern, basic web application design pattern and basic risk analysis model pattern are presented to show how these patterns can work together to establish context and conduct risk identification for web applications. In addition, we use a health monitoring web application as an example to illustrate the usage of this pattern language following the usage of example patterns proposed in this paper.

Future work of this security risk analysis pattern language 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 through case studies to conclude and improve this pattern language.

REFERENCES


PATTERNS, M. AND PRACTICES DEVELOPER CENTER. Tiered distribution.


