# Requirements Engineering for SaaS Application Security in Cloud Using SQUARE Methodology

**E. Pragnavi**
Assistant Professor,
Dept. of CSE, UCE,
Osmania University

J. Sandeep **Kumar**
Product Technical Lead,
Infosys, Hyderabad

***Abstract*--The advent of the digital business and the digital workplace is increasing the speed at which mobility and security are disconnecting. Cloud computing is becoming an increasingly established as cost efficient and needs oriented information system at the same time the use of cloud computing systems also involves a number of security risks. As a result the use of cloud computing systems is still restricted. Security is the top criteria along with availability and cost which must be satisfied. This paper applies a detailed assessment of the potential security threats in one of the service model identified by NIST[2] i.e., Software-as-a service (SaaS) applications compiled within all phases of the software development life cycle using SQUARE methodology[1].**

***Index Terms*-- Cloud Computing, Software-as-a-Service, SQUARE methodology.**

## I. INTRODUCTION

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and release with a minimal management effort or service provider interaction.

Cloud computing generally, and SaaS in particular is a rapidly growing method in delivering technology, which is also visible to cloud's end users and services are deployed by end users. The SaaS applications are mainly accessed on various clients devices through a thin client interface such as web browsers. SaaS in turn is built upon the underlying IaaS and PaaS stacks and provides a self-contained operating environment that is used to deliver the entire user experience, including the content, its presentations, and applications and management capabilities.

Figure 1 : Visibility of SaaS


SaaS provides the most integrated functionality built directly into the offering, with least consumer extensibility, and relatively a high level of integrated security. The increase interest in the advancement of SaaS is due to the SaaS service provider installs runs and maintains all the essential software and its associated hardware.

The consumers of SaaS can be organizations that provide their members with access to software applications, end users who directly use software applications, or software application administrators who configure applications for end users. SaaS consumers access and use applications on demand, and can be billed on the number of consumers or the amount of consumed services. Cloud services in various application categories which are available today under “Software as a Service” are as follows:

- Scalable Websites
- Data Management and Distribution
- Software Development and Testing
- Interactive, Mobile applications


ISSN: 2278 – 1323 


- Scientific Computing.

Figure 2 : Software-as-a-Service

SaaS are becoming the gatekeeper for sensitive information, whether it is in the form of personal data or corporate data. Security breaches can therefore be potentially devastating for both users and SaaS providers. A SaaS provider will need to check if the development team has implemented secure engineering practices in the design and code to make sure the SaaS application is secure.

Security must be enhanced and integrated into all phases of software development life cycle and should be overlooked frequently for the underlying requirements catering what software should do and what not. Without requirements security architecture cannot be built. Organizations should ensure that the best practices of application security, identity management, data management, privacy are integral throughout the life cycle of the application. Requirements definition is conducted at the start of the start of the life cycle to establish what the software shall and shall not to do. Adding security to software after the project is delivered drives up maintenance cost and effects internal or external customers data and corporate reputation. Establishing patterns for security requirements will enable software development teams to utilize best practices in security requirements.

The Secure Quality Requirements Engineering (SQUARE) methodology defines a process for a software development team to effectively elicit security requirements from different stakeholders and prioritize security requirements based on the security goals of the system. In step 1 (Agree on Definitions), the various stake holders and development team agree on definitions that will be used throughout the elicitation process. In step 2 (Identify security goals), the stakeholders and the requirements engineer team identify assets and security goals for the system. In step 3 ( Develop Artifacts), the requirements engineer team uses the security goals to develop artifacts for the security requirements definition. In Step 4 (Perform Risk Assessment), the requirements engineer, risk expert, and stakeholders use the artifacts developed in Step 3 to conduct a structured risk assessment to define security risks. In Step 5 (Elicitation Technique), the requirements engineer selects requirements elicitation technique(s). In Step 6(Elicit Security Requirements), the stakeholders and requirements engineer elicit security requirements using the risk assessment and artifacts and produce an initial requirements list. In Step 7(Categorize Requirements), the requirements engineer categorizes the security requirements. In Step 8(Prioritize Requirements), stakeholders and the requirements engineer prioritize the requirements using the risk assessment and the proposed requirements list. In Step 9(Requirements Inspection), the inspection team inspects and validates the proposed security requirements [1].

## II. SQUARE METHODOLOGY

SQUARE can be decomposed into nine discrete steps. Each step identifies the necessary inputs, participants, suggested techniques, and final output. The following are the nine steps of SQUARE methodology.

### *A. Agree on Definitions*

Especially in highly technical domains of knowledge, agreeing on definitions is critical for effective communication. Software development requirements represent a contract of complex terms between the clients and the software development organizations, so using agreed upon definitions that are common to both parties is vital.

The security controls and their scope are negotiated into the contracts for service; service levels, privacy and compliance are all issues to be dealt legally in contracts. This may include the following:


ISSN: 2278 – 1323 

*Application Security Architecture* – Consideration must be given to the reality that most applications have dependencies on various other systems. With Cloud Computing, application dependencies can be highly dynamic, even to the point where each dependency represents a discrete third party service provider. Cloudcharacteristics make configuration management and ongoing provisioning significantly more complex than with traditional application deployment. The environment drives the need for architectural modifications to assure application security.

*Software Development Life Cycle (SDLC)* – Cloud computing affects all aspects of SDLC, spanning application architecture, design, development, quality assurance, documentation, deployment, management, maintenance, and decommissioning.

*Compliance* – Compliance clearly affects data, but it also influences applications (for example, regulating how a program implements a particular cryptographic function), platforms (perhaps by prescribing operating system controls and settings) and processes (such as reporting requirements for security incidents).

*Tools and Services* – Cloud computing introduces a number of new challenges around the tools and services required to build and maintain running applications. These include development and test tools, application management utilities, the coupling to external services, and dependencies on libraries and operating system services, which may originate from cloud providers. Understanding the ramifications of who provides, owns, operates, and assumes responsibility for each of these is fundamental.

*Vulnerabilities* – These include not only the well-documented—and continuously evolving—vulnerabilities associated with web applications.

*B. Identify Security goals*

As this papers discusses about SaaS application security, some of the most significant threats faced now-a-days are attackers defacing websites, stealing credit cards numbers and bombarding websites with denial of service attacks and also the malware present in the websites. Other problems faced by the compromised service provider, internal employees and other end users who mistakenly stumbles across sensitive data and pose significant risk. The solution is ongoing process involving the stake holders and practices. To examine the security threats to an application, the application security goals must be understood in various categories as

- Business goals- To provide an application that supports services for an organization.
- Individual goals- to provide an application that supports services for end users (public cloud)

The SQUARE process places security goals near the beginning of the process.

The security goals are of the following:

- Authentication: It addresses the question, Who are You? It uniquely identifies the end users of an application and services.
- Authorization: It addresses the question, What can you do? It is the process that governs the resources and operations to the authenticated clients and permitted to access.
- Auditing: Effective auditing and logging leads to non-repudiation. It guarantees that user cannot deny performing an operation or initiating a transaction.
- Confidentiality: Also referred as privacy, is the process of making sure that data remains private and confidential.
- Integrity: it is a guarantee that data is protected from modifications, particularly data passed across networks.
- Availability: The application is available for legitimate users.

*C. Develop Artifacts:*

Architectural diagrams, use cases, misuse cases, security use cases, attack trees and services and also essential assets will be documented in this step. The various stakeholder with requirements engineering team can generate a comprehensive set of security requirements, the team must collect a complete set of artifacts of the system. In order to develop artifacts this paper proposes a modeling technique with the combination of misuse case and security use case[8].

While Misuse case are used to explore and document security threats, the security use cases are used to specify and analyze security requirements.


ISSN: 2278 – 1323 

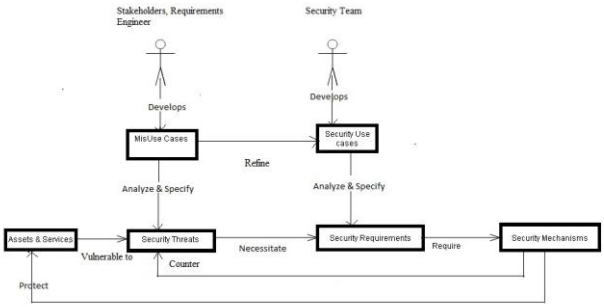


Figure 3 : *MisUse case & SecurityUsecase*

For instance an attack scenario using security use case example on Authentication is documented in the following way:

*Use Case Path:* Attempted spoofing using valid user identity.

*Security Threat:* The system authenticates and authorizes the misuser as if the misuser were a valid user.

*Preconditions:*

1) The misuser has a valid means of user identification.
2) The misuser has an invalid means of user authentication.

*Interactions:*

1) System Interaction: the system shall request the misuser's means of identification and authentication.

2) Misuser's Interaction: provides a valid use of user identity but an invalid means of user authentication.

3) System Action:

1) The system shall misidentify the misuser as a valid user.

2)System shall fail to authenticate the misuser's.

4) Misuser's Interaction: The system shall reject the misuser by canceling the transaction.

*Postconditions:*

1)The system shall not have allowed the misuser to steal the user's means of authentication

2) The system shall not have authenticated the misuser as a valid user.

3) The system shall not have authorized the misuser to perform any transaction that requires authentication.

4) The system shall have recorded the access control failure.

*D. Perform Risk Assessment:*

In SaaS model the control over the data is delegated to SaaS provider and the interactions happen on the web. To measure the security of a SaaS application first step is to identify or assess the application vulnerability categories which represent the areas where security mistakes are most frequently made and are used as a framework when evaluating the security of a SaaS application.

The SaaS application vulnerabilities categories as given by CSA[4] are as follows:

Data Breach

Data Loss

Account hijacking

Insecure application interfaces

Denial of Service

Malicious Insiders

abuse and nefarious use

Insufficient due diligence

shared technology issues

The purpose of this step in the SQUARE process is to identify the vulnerabilities and threats that face the system, the likelihood that the threats will materialize as real attacks, and any potential consequences of an attack. The risk assessment also serves to prioritize the security requirements at a later stage in the process.

The best way to avoid such disasters is to establish an ongoing security risk management process that begins with quantifying the value of SaaS applications, as well as the data they manage, through a complete security risk assessment. There are several risk assessment techniques to get rid of different types of risks. For example, the various risk management techniques that were field tested were selected after a review was completed. This review examined the usefulness and applicability of various risk assessment techniques:

1) The Government Accountability Office's (GAO) model


ISSN: 2278 – 1323 

2) National Institute of Standards and Technology (NIST) model

3) NSA's INFOSEC Assessment Methodology

4) Butler's Security Attribute Evaluation Method (SAEM)

5) CMU's "V-RATE" method

6) Yacov Haimes's RFRM model

7) CMU's Survivable Systems Analysis method

8) Martin Feather's DDP model

*E. Elicitation Technique*

The requirements engineering team must select an elicitation technique that is suitable for the system.

The various techniques are as follows [1]:

1) IBIS (Issue Based Information Systems)

2) JAD (Joint application development)

3) ARM (Accelerated Requirements Methodology).

*F. Elicit Security Requirements*

This step is the heart of the SQUARE process: the elicitation of security requirements. The process of discovering the requirements for a system by communication with customers, system users and others who have a stake in the system development.

In this technique, a correct focus question is presented to an expert who identifies security requirements, and the requirements are collected, categorized and refined.

*G. Categorize Requirements*

Categorization is the process which allows the requirements engineer and stakeholders to classify the requirements as High, Medium or Low. Here in this paper another type of dimension is added i.e., counter measure which is an action, process, device or system that can prevent or mitigate the effects of threats to a SaaS application. Different types of counter measures for authentication, authorization, session management, input validation, logging, cryptography, error handling will be taken into consideration.

*H. Prioritize Requirements*

Thus, the purpose of this step in the SQUARE process is to prioritize the security requirements so that the stakeholders can choose which security requirements to implement and in what order. Security requirements are often implemented in stages, and prioritization can help to determine which ones should be implemented first. A number of prioritization methods have been found to be useful in traditional requirements engineering and could potentially be used for security requirements. Some of the techniques are AHP (Analytical hierarchy process), Triage, Theory-W, Binary Search Tree.

*I. Requirements Inspection*

The last step of the SQUARE process, requirements inspection, is one of the most important elements in creating a set of accurate and verifiable security requirements.

Based on the security recommendations from regulatory bodies like CSA, NIST, ENISA, Common Criteria. The Framework can be conceptualized for SaaS security evaluation. A consolidate report should be published to the SaaS provider and security recommendations should be provided .

## *III. CONCLUSION*

Most of the Cloud services are rendered through its SaaS model and is accompanied by a permanent revolution in the services launched on the market by vendors. In this context particular attention is driven by the security functions which are offered by cloud SaaS applications. The cloud security taxonomy provides a clearly structured framework of the security areas that should be considered when using cloud services. Security technologies are not yet adequate to achieve the protection goals in Cloud SaaS applications. By using Security Quality Requirements Engineering (SQUARE) process the quality and appropriateness of the security requirements should rise and the system should be more likely to be designed from the beginning to properly support security requirements in addition to the functional and non functional requirements and security requirements must take into account at each phase of development cycle. There are continuous


ISSN: 2278 – 1323 

work to improve the methodology in support of SaaS applications in steps used for eliciting and prioritizing the security requirements in cloud.

## REFERENCES

[1] N. R. Mead, E. Hough, and T. Stehney, Security Quality Requirements Engineering (SQUARE) Methodology, Software Engineering Institute, Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU/SEI-2005-TR-009,2005. http://www.sei.cmu.edu/publications/documents/05.reports/05tr009.html

[2] "NIST Cloud Computing Standards Roadmap – Version 1.0"- Michael Hogan, Fang Liu, Annie Sokol, Jin Tong

[3] OWASP "open web application security project" http://www.owasp.org/index.php title=Special:Search&search=security+ requirements&go=Go.

[4] Cloud Security Alliance document, "Security Guidance for Critical Areas of Focus in Cloud Computing,"

[5] Mead, N. R.; Chen, P.; Dean, M.; Ojoko-Adams, D.; Osman, H.; Lopez, L.; & Xie, N. *System Quality Requirements Engineering (SQUARE) Methodology: Case Study on Asset Management System* (CMU/SEI-2004-SR-015, ADA431068). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004. http://www.sei.cmu.edu/publications/documents/04.reports /04sr015.html.

[6] "Cloud,Grid, High performance Computing: Emerging Applications"- Emmanuel Udoh

[7] OPROF "Open Process Framework"

[8] "Misuse/Use cases and Security use Cases in Eliciting Security Requirements"- Qian Gao

E.PRAGNAVI, received B. Tech in Electronics &Computer science Engineering from JNTU, Hyderabad, India, holds M.Tech in Computer Science from JNTU, Hyderabad, India, She has 5 years of academic experience in Computer Science and Engineering.

J.SANDEEP KUMAR,, received B. Tech in Computer Science & Engineering from JNTU, Hyderabad, India, hold M.S in Software Engineering from BITS Pilani University, Rajasthan, India. He has 10 years of experience in IT.



ISSN: 2278 – 1323 