A Parametric Approach for Security Testing of Internet Applications

Arun K. Singh
E-Commerce Research Lab.

Anand J. Iyer,
Infosys Software Labs. for Test Execution and Research.

Venkat Seshadri
Product Competency Center

Infosys Technologies Limited
Electronic City, Bangalore, India
{arunks, Anandj_iyer, venkatar}@infy.com

Abstract. Security is one of the prime concerns for all the Internet applications. Often it is noticed that during testing of the application, security doesn't get due focus and whatever security testing is done, it is mainly limited to security functionality testing as captured in the requirements document. Any mistake at requirement gathering stage can leave the application vulnerable to potential attacks. This paper discusses the issues and challenges in security requirement gathering and explains how it is different from normal functionality requirement gathering. It presents an approach to handle different factors that affect application security testing.

1 Introduction

As compared to traditional client-server or mainframe-based applications, Internet applications have different security requirements [1]. For any Internet application, there is a need to provide an unrestricted global access to the user community. This also brings in the threat of outside attack on the system. The hackers may try to gain unauthorized access to these systems for a variety of reasons and disturb the normal working of the application.

The security of Internet applications necessarily means prevention of two things: destruction of information and unauthorized availability of information. The broad spectrum of application security can be expressed as "PA^{3}IN" viz. **P**rivacy, **A**uthentication, (Authorization, Audit Control,) **I**ntegrity and **N**on-repudiation.

The first issue in application security is to provide data privacy to the users. This means protecting information confidentiality from the prying eyes of unauthorized internal users and external hackers. Before allowing the access permissions to the users, it is essential to ensure the legitimacy. The process of identifying users is called authentication. Establishing the users' identity is only half the battle. The other half is access control/ authorization which means attaching information to various data objects denoting who can and who cannot access the object and in what manner (read, write, delete, change access control permissions, and so forth). The third 'A' is for audit control which means maintaining a tamper proof record of all security related events. And then when the data is in transit across the network, we need to protect it against malicious modification and maintain its integrity. The non-repudiation means the user should be unable to deny the ownership of the transactions made by them. It is implemented using Digital Signature [3].

This paper presents an approach for testing security of Internet applications.

2 Issues in Testing Security of Internet Applications

The two main objectives of an application security testing are to:

1. Verify and validate that the security requirements for the application are met.
2. Identify the security vulnerabilities of the application under the given environment

For security testing of an application, the first step is to capture requirements related to each of the security issue. This could not only be a tricky and painstaking exercise because the security requirements are seldom known very clearly at the time of project initiation. The traditional Use case approaches have proven quite useful in general requirements engineering, both for eliciting requirements

and getting a better overview of the stated requirements [4]. However security requirements essentially need to concentrate on what should not happen in the system and this cannot be captured by traditional Use case approach.

Also, the security model of Internet applications has undergone many changes. The traditional model for securing an application from outside elements mainly relied on access control. This model was based on creating a hard perimeter wall around the system and providing a single access gate that can be opened only for authenticated users. This security model has worked well with most of the simple Internet based applications. The gateway here refers to a firewall that classifies all users as "trusted" or "untrusted". In this simplistic model of security, every "trusted" user who is allowed to cross the gate, gains access to every portion of one's business and no further security checks are done. As the requirements for security increases, the applications need to implement a fine-grained security. Instead of allowing the trusted user to access every portion of the business, the modern security models divide the business domain into many regions and ensure different levels of security for each region. This means creating a security perimeter for each region. The first level of security check can not be very rigorous as one would want to let in prospective customers, vendors and service providers as quickly as possible. Most of the Internet applications today have multiple security regions with different levels of security and these regions could be nested or overlapping with other regions within same single application. While developing an approach for testing security of these kinds of Internet applications, a proper strategy planning is very much essential. It is better to plan security testing at requirement gathering stage itself because there are many issues, which cannot be captured later on by usual methods of testing [2]. One such example is testing for denial of service attack.

The issues discussed above make security testing of Internet a very challenging task. In the next section, a parametric approach to security testing is presented and each step is explained with the help of a sample example.

The Parametric Approach for Security Testing

As discussed in previous section, many of the security parameters cannot be captured and tested using traditional approach. In the proposed parametric approach for security testing, before we start the requirement gathering, a template to enlist all security parameters are created. This task can be achieved in four steps as described below:

1. Create an exhaustive list of all security issues in the application
2. Identify all possible sub parameter for each of these issues
3. List all the testing activities for each sub-parameter
4. Assign weightages corresponding to the level of security and priority.

Once this template is ready, it can be used to streamline each stage of testing lifecycle. The lifecycle process of security testing is similar to the software development lifecycle process. In the proposed approach, the security testing lifecycle stages are as follows:

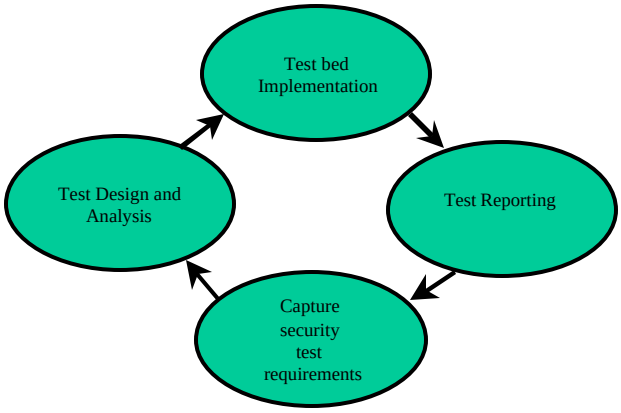

Figure 3.1: Security Testing life cycle

a) Capture security test requirements

b) Analyze and design security test scenarios

c) Test bed implementation

d) Interpreting test reports

In the following sub-section each of these stages are explained.

3.1 Parametric Approach to Capture Security Test Requirements

To define the scope of security testing, check the stated requirement against the parametric template. This will help in identifying all the missing elements or the gaps in security requirement capture. The next step would be to allocate appropriate weightages to the various security sub parameters. These weightages would typically be determined by the nature of business domain that the application caters to and the heuristic data available with the security test experts.

For example, consider an e-shopping application for an online store. The key security requirement for such an application would be to ensure that the user who provides his own payment information is indeed, who he claims to be and to make sure that the information provided is correct and is not visible to anyone else. From that perspective, authentication, confidentiality and non-repudiation are the key parameters for consideration. One possible set of weightages for these parameters could be 0.5, 0.25 and 0.25 respectively (on the scale of 0 to 1). This is in line with the impact; a security breach could have on the business.

The advantage of assigning appropriate weightages is that it helps to optimize the test scenarios for each of these parameters. On the next page, table 3.1 presents a sample parametric table. The snapshot in table 3.1 only comprises of those parameters that are relevant for testing the application requirement. The main security parameters, as listed in the table 3.1 are - Authentication, Authorization, Access control, Non-Repudiation and Audit control. This list is just a sample list and in actual scenario there can be many more parameters in the template.

These parameters are further classified according to testing subtask. For example the requirement of authentication can be implemented using userid/password, smart card, biometric and digital certificates. And each one of this implementation has different testing requirement. Based on the testing complexity, appropriate weightages are assigned. The weightages can be determined based on past experience

and the series of interactions conducted with the customer during the requirement gathering stage. Going back to the example, since biometric requires substantial investment from the client's side the digital certificates score higher than the other modes of implementing authentication.

The last column holds the priority level assigned to each sub-parameter. The business domain and the experience of the test expert determine the priority level assigned to each sub-parameter. For example, consider a case of Internet banking application. Here the authentication is slightly of higher priority than that of authorization or access control. Where as in case of a b2b application the authorization and the access control takes higher priority.

All the parametric values are recorded and this constitutes a requirement document outlining the areas and the level of testing that has to be executed.

No.	**Parameters**	**Dimension (scale of 1 - 5)**	**Dimension (scale 1 - 5)**	**Dimension (scale 1 - 5)**	**Dimension (scale 1 - 5)**	**Dimension (scale 1 - 5)**	**Priority (scale of 1 – 100)**
1	**Authentication - corroboration that an entity is who it claims to be.**	No authentication (0)	Password Based (3)	Smart card and PIN (3)	Biometric (4)	Digital certificates. (5)	10
2	**Authorization**	No Authorization (0)	Role Based (3)	User Based (3)	Using Electronic Signatures (for non-repudiation) (3)	Role and User Based (4)	10
3	**Access Control**	No ACL (0)	Context Based (3)	User Based (3)	Role Based (3)	All 3 (4)	7
4	**Non repudiation**	Nothing (0)	Digital signature (3)	Encryption and digital signature (4)			5
5	**Audit control**	Nothing (0)	Use Firewall/ VPN logs (1)	Custom made logs (3)	Employ managed security services (4)		5

Table 3.1

3.2 Parametric Approach to Security Test Analysis and Design

Once the weightages for each security parameter is finalized, the next task is to create test scenarios for testing the security requirements. The weightages assigned to each sub-parameter goes on to suggest the number of scenarios that need to be tested for a particular requirement and the complexity involved.

The parametric approach also helps in identification of the extent of data that would be needed for test execution and also the distribution of data based on various scenarios. Also, this will help in identifying the kind of tools that will be needed and determine the appropriateness of a tool for a given type of test.

Now the data that is recorded with the parametric template can be tabulated in the below given form:

No.	Parameter	Weightages (Scale of 1- 5)	Priority (Scale: 1 – 100)
1	Authentication	3	10
2	Authorization	2.75	10
3	Access Control	2.6	7
4	Non-Repudiation	2.3	5
5	Audit control	2.0	5

Table 3.2

The table 3.2 above is comprised of the security parameter with its corresponding values of Weightages and Priority levels. These weightages are the average of weightages assigned for the testing activity of each sub-parameter as illustrated in the table 3.1. With the availability of these values for each sub-parameter the test expert is now able to plan the test scenarios to the desired level of priority and depth.

3.3 Security Test bed Implementation

In case of application level testing the approach of testing the security vulnerabilities should be more logical. Here logical security deals with the use of computer processing and/or communication capabilities to improperly access information. Second, access control can be divided by type of perpetrator,

such as employee, consultant, cleaning or service personnel, as well as categories of employees. The type of test to be conducted will vary upon the condition being tested and can include:

Determination that the resources being protected are identified, and access is defined for each resource. Access can be defined for a program or individual.

Evaluation as to whether the designed security procedures have been properly implemented and function in accordance with the specifications.

Unauthorized access can be attempted in on-line systems to ensure that the system can identify and prevent access by unauthorized sources.

Most of the security test scenarios are oriented to test the system in the abnormal conditions. This throws up the challenge of simulating the real life conditions that the test scenarios require. There are various tools that can be used to test various vulnerabilities that the system may have. The choice of such a tool is determined by the extent of vulnerability, the tool is able to test, Customization of the tool should be possible and testing of the system using both the external as well as the internal views. The local users are the internal view and the external users constitute the external views. There are various programming tools that can be programmed to simulate the various access control breaches that can happen.

4 Interpreting Security Test Reports

Once tests have been executed and security gaps have been identified, the gaps need to be analyzed in order to suggest improvements. The reports need to be validated as many a times as possible . The reports may be misleading and the actual loopholes might be elsewhere or might not exist at all. The security test reports are like X-ray reports, where a security expert is essential to decipher the vulner-

abilities reported and bring out the actual problem report. There are various security reporting tools. While selecting a security-reporting tool, the following parameters have to be taken in to consideration:

1. Extent if vulnerability, the tool is able to report.
2. Customization of the tool should be possible.
3. Testing of the system using both the external as well as the internal views.

Once the analysis of the report is completed a list of vulnerabilities and the set of security features that are working is generated. The list of vulnerabilities is classified. The classification level of vulnerability is determined by the amount risk involved in the security breach and the cost of fixing the defect. Once the classification is done the vulnerability is fixed and retested.

5 Conclusions

A parametric approach to security testing not only, works very well for security requirement capture, effort estimate and task planning but also for testing different levels of security requirements by different components within same application. This approach for requirements gathering comes out to be real handy in capturing those security requirements, which cannot be captured by traditional means. This approach enables one to adopt the parametric matrix at each and every stage of security testing, and improve the values in the matrix after each execution of a project. Due to this capability of incorporating experiential data, the parametric approach can provide the most effective mechanism for security testing of Internet applications.

6 Disclaimer

The authors of this report gratefully acknowledge Infosys for their encouragement in the development of this research. The information contained in this document represents the views of the author(s) and the company is not liable to any party for any direct/indirect consequential damages.

References

1. Singh, A.K.: E-Commerce Security. Proc. of National Seminar on E-Commerce & Web Enabled Applications, Harcourt Butler Technological Institute, Kanpur, India (Mar., 2000).
2. Singh, A.K., Subraya B.M. : Modelling Security at Requirements stage in E-Commerce Projects, Proc. of Information Security Solutions Europe Conference (ISSE 2000), Barcelona, Spain (Sept., 2000).
3. Microsoft Product Security – http://www.microsoft.com/ntserver/info/securitysummary.htm
4. Eliciting Security Requirements by Misuse cases -- Guttorm Sindre, Norwegian Univ. of Sci. and Tech and Andreas L. Opdahl, Univ. of Bergen, Norway