ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

An Overview of Security Techniques for Web Applications

Fatima Mulla [1], Varshapriya J N [2]

P.G. Student, Department of Computer Engineering, V.J.T.I, Mumbai, Maharashtra, India[1]

Assistant Professor, Department of Computer Engineering, V.J.T.I, Mumbai, Maharashtra, India[2]

ABSTRACT: Security is the most crucial aspect of information and communication technology. As applications deployed over the internet are prone to attacks from all over the world, securing web applications has become a major concern of developers. This paper presents an overview of different security techniques and mechanisms available for securing web applications. These techniques are classified with respect to the security trait they incorporate. This paper also includes an organized approach which can be adapted along with the development lifecycle of a web application to incorporate security mechanisms into the system.

KEYWORDS: Security, web applications, cryptography, security concepts

I. INTRODUCTION

Internet is the most efficient way of distributing products and services globally. Thus, businesses are moving over the internet and web applications are being developed for providing the client-side interface. But the global accessibility of the internet, makes web applications prone to attackers from all over the world. Even if the server of an application is confined within the secure boundary of organizations, the web application can be manipulated to penetrate into the server and ultimately the entire system can be compromised. As assets of an organization decide its business value, protection of these assets is crucial. The goal of attackers is to get hold of these assets or hamper their use when required. Thus, a thorough analysis and proper selection of security techniques is necessarily required in order to protect the assets of the organization.

An effective security model is designed under the assumption that attackers are completely aware of the physical and logical structure of a system. They have complete knowledge of all the cryptographic algorithms and are aware of all the vulnerabilities. The idea behind this assumption is: if an attacker with complete system knowledge cannot get into the system, an attacker without knowledge cannot. The goal of a security model should be to protect the system against such attackers.

KEY CONCEPTS OF INFORMATION SECURITY:

i. **Confidentiality:** Confidentiality is defined as the property of a system which prevents unauthorized users and processes from accessing data stored in the system.

ii. **Integrity:** Integrity is the property of a security system which protects the systems against unauthorized modification. Data present in the system should be equivalent to its exact version received at the time of storage. This property restricts users and processes from modifying data they are not granted permission to modify.

iii. **Availability:** It is the property of a system which makes it always available to its users. This property ensures that all the services provided by the system are responsive whenever a user requests for it. Security techniques implemented in systems should be optimized so that other functionalities of the system do not suffer decay in their performance.

iv. **Authentication:** Authentication is the mechanism deployed in a system which deals with the verification of the identity of users. It is composed of a set of techniques which verifies that the person sitting on the client side is the one who he/she is claiming to be.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

v. **Authorization:** Authorization defines permissions. It manages what a user/process is allowed to access, modify or execute.

vi. **Access Control:** Access control is the mechanism which restricts the flow which users adapt while browsing around the system. It defines fixed paths which must be traveled for reaching a critical item.

vii. **Non-repudiation:**Non-repudiation is defined as the mechanism built into a system which keeps track of accesses and modifications made into the system; this ensures that no action performed in the system can be denied.

II. LITERATURE SURVEY

A. **Confidentiality:** Confidentiality in web applications needs to be applied both on the stored data and source code of the application. To incorporate confidentiality in a system, different methodologies can be adapted. Some of them are listed as follows:

1. **Encryption:** Encryption is the key component for ensuring confidentiality in any system. Encryption algorithms are primarily classified as stream ciphers and block ciphers, where symmetric ciphers perform encryption on single bits and block ciphers works on blocks of defined sizes. RC4, Salsa20 [1], and ChaCha[2] are examples of stream ciphers. RC4 was the most widely used stream cipher because of its speed and simplicity, but its vulnerabilities [3] lead to its depreciation by RFC 7465, in February 2015. Based on the keys used, the block ciphers are classified as symmetric key and asymmetric key ciphers [4]. DES, 3DES, AES, Blowfish [5], Twofish [6], IDEA [7] are examples of strong symmetric key encryption algorithms. RSA, Elgamal [8] are examples of asymmetric key algorithms, whereas Diffie Hellman is a key exchange algorithm which takes advantage of asymmetric cryptography to generate a shared secret key between two communicating parties. Block ciphers are further classified based on their mode of operations as ECB, CBC, PCBC, CFB, OFB, CTR [9], GCM [10].

   In practice a combination of different encryption algorithms is used in order to ensure confidentiality. For example, we can generate a pair of public and private key using RSA algorithm, these generated keys can then be used of encrypting and decrypting data using 128-bit AES in CBC mode.

2. **Identifier Encoding:** Another approach towards ensuring confidentiality is to store data with encrypted identifiers [9]. In some systems the associations are more confidential than the values themselves, then the relational identifiers can be encrypted rather than the entire data. For example, in a database, we can encrypt the foreign keys which identify the association between relations, while rest of the attributes can be stored as plaintext. A similar approach can be applied for storing multimedia where, instead of encrypting a media file, we can encrypt its name and store it in the database.

3. **Negative Database:** Negative database is defined as the category of database which stores a huge amount of counterfeit data along with the actual data. Different approaches can be used to implement negative database. Basic approach is to encrypt the real data and distribute it inside the attributes of a large set of randomly generated negative data values. As per the security requirements of a system, data storing and retrieving mechanisms can be designed. For example, to provide fast retrieval of frequently used data, a data caching mechanism can be provided [11]. Data can also be segregated into sensitive and non-sensitive data, where non-sensitive data can be directly stored, and sensitive data can be distributed in multiple attributes with negative data [12].

B. **Integrity:** Integrity is the most important property of a system as it provides assurance that data stored in the system has not been tampered. For implementing integrity checks, different techniques are available, which include the following:

1. **Hashing:** Hashing is the technique of using mathematical algorithms to map an input of arbitrary length into an output of fixed length. This output produced is referred as hash and it has the property of being deterministic, i.e. the same input will always produce the same hash output whereas a changing a single input bit will change the output. Cryptographic hash functions are the subgroup of hash functions which have the property of being preimage resistant and collision-free. Examples of cryptographic hash functions include

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

MD4, MD5, SHA, MAC, HMAC, BLAKE2[13]. The security of hash functions is amplified by pairing it with Salt. Some non-cryptographic hash functions for integrity corrections are also used like CRC, Checksum, etc.

C. **Authentication:** Authentication deals with the identification of users before they are allowed to enter a system. In order to prevent masqueraders from entering a system, different authentication control techniques can be used. Few of them are specified as follows:

1. **Static Authentication:** In static authentication techniques, users are identified based on some predefined credentials like username, password, passphrase, biometrics, ID cards, PIN number, etc. The authentication process can be implemented to be single factored, two factored or multi-factored. Example of single-factor authentication is digital credentials where an association of username and password is used for logging a user into a website; whereas an example of two-factor authentication is bill payment through debit card where in addition to physical swiping of the card, a digital pin needs to be authenticated. It is common practice to hide the characters of a user's password while it is being typed to avoid shoulder surfing. Also, the passwords should be encrypted at client side first, in order to avoid eavesdropping on the network. Another approach is to generate graphical password selection scheme to avoid shoulder surfing and spyware attacks [14].
2. **Dynamic Authentication:** Dynamic authentications use time variant credentials for identifying a user. An example of dynamic authentication is One Time Password (OTP). It is used as two-factor authentication, where in addition to digitally verifying oneself, the user needs to verify the session by correctly submitting a time-limited pseudorandom token sent to a trusted device or email address. This OTP can also be generated both at client and server side eliminating the need for a server to send it exclusively for each session start [15].
3. **Automated Guessing Discouragement:** Captchas can be provided at the time of user authentication to discourage brute-force attack for password guessing. Also, after a fixed number of unsuccessful authentication attempts, user accounts should be suspended for some time, and a notification needs to be sent to corresponding user's trusted device or email account informing a possible security breach.

D. **Access Control:** Access control constraints what a user can do in a system in order to prevent activities which can lead to security breach [16]. It works in close association with authorization, where authorization controls the permissions provided to users. Access control can be deployed over different aspects of a system, some of those are enlisted as follows:

1. **Access Control Models:** Traditionally, access control models are classified as: Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role-based Access Control (RBAC) [17]. DAC policies allow the owner an object to decide who has access over it. This model is commonly used in social networking websites. MAC policies provide centralized access control, where subjects and objects are classified based on predefined security levels. This model is not commonly deployed in web applications as it lacks flexibility. In RBAC model, policies are applied to roles rather than to users; where, roles define the set of rights and responsibilities of a user. It is the most commonly used access control model in web applications because of its high flexibility. Example of roles in an examination system are students, teachers, administrators, etc. In such a system if a teacher is to be promoted administrator, his rights can be very easily changed by simply changing role assigned to it; making it very dynamic and flexible to use.
2. **Cryptographic Access Control:** Cryptographic access control, access control over data items is provided by distribution of keys. A user having read access over a data item, must hold the decryption key of that data item, whereas a user having right access over an data item, must hold the encryption key of that data item [18]. The encryption mechanism adapted can either be symmetric or asymmetric. RSA can be used for key generation in cryptographic access control [19]. The key generation process can also be applied over hierarchical cryptosystems [20].
3. **Flow Control:** Flow of a user's browsing path should also be controlled while designing web applications. For example, in a shopping website, a should not be allowed to go to payment page, without selecting any product. Security codes should be written inside the header section; before loading of actual page; to redirect unsafe requests. Similarly access to sensitive resources; like database access; should be restricted inside the domain of current LAN.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

E. **Non-repudiation:** Non-repudiation is the assurance that a user will always be accountable for the actions performed by him. The most commonly used mechanism to ensure non-repudiation is digital signature. Where, the sender of a message or modifier of a data value generates a digest of data and encrypts it using his private key and sends it along with the data so as to be verified later by the receiver by recalculation with the sender's public key. Different combination of algorithms can be used for calculating the message digest as well as for its encryption process. Some efficient implementations include the use of RSA [21], elliptical curve cryptography [22], etc.

F. **Security Requirement Engineering Process**: Security requirement specification and validation is an important phase for software quality assurance. This phase is crucial, as it is the base on which a system's security model is built. Security requirements are dependent on many parameters such as the nature of the application, purpose of the application, stakeholders involved, etc. These parameters decide the importance of different security measures and thus will help to draw a trade-off while implementing these security measures. For example, in banking applications, integrity and confidentiality are very crucial, they must be taken care of, even if it comes at the cost of availability. Integration of security requirements in a pre-designed system leads to serious designing challenges as well as the introduction of vulnerabilities. The security requirements engineering process; MOSRE-WebApp Process [23]; presents a spiral model for security requirement engineering. In this model, security requirements are defined alongside the software development lifecycle. The steps involved in the process include the following:

1. Inception of Security Requirements: The process of security requirement engineering starts with the system's software specification, where the nature of the application, stakeholders involved and assets available are analysed to form an inception of the security system to be implemented.
2. Identification of Security Goals / Objectives: The process then moves towards elicitation of requirements for the application. Different elicitation techniques can be used which can draw out goals the application is expected to perform. After identification of application's primary goals; security goals/ objectives of the application are identified with respect to assets, business goals, organizational policies, etc. Criticality of each security goal is also measured based on the elicitation conducted and is associated with the goals defined. The top-level security goals generally are confidentiality, integrity, and availability. These goals can have a further hierarchy of sub-goals/objectives.
3. Identification of Threats and Vulnerabilities: By analysis of security goals and assets, threats to the application can be identified. The design of the system should be studied to find possible vulnerabilities. An approach to finding out threats and vulnerabilities in the system can to be to test the design of the system against the known threats and eliminating the possibilities which are handled by the system.
4. Risk Assessment: This step assesses the risk when threats and vulnerabilities occur. After analysis of each threat and vulnerability, corresponding risk is determined.
5. Prioritization and Mitigation: Threats and vulnerabilities are categorized and prioritized based on security goals. Based on this classification, high priority threats and vulnerabilities should be mitigated first with.
6. Identification of Security Requirements: Security requirements are the defences built into a web application to protect it against attacks. They are specified as functional requirements of the system. These requirements, implement the mechanisms to which meet the security goals.
7. Development of UML Diagrams: UML diagrams can be built to provide a detailed view of security requirements identified. These diagrams can be used to test the design for different test cases. These diagrams also provide a blueprint for code generation.
8. Negotiation & Validation: In this step, the security requirements defined are categorized as essential and non-essential according to the security goals and preference of stakeholders. Validation of security requirements is done with the association of security experts and stakeholders involved. Then the efforts required to implement these requirements are measured in terms of time and cost, and a trade-off is decided.
9. Specification: It is the final phase of the security requirement engineering process. This presents the final model of security requirements to be implemented in the web application.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

III. CONCLUSION

Security modelling of web applications is a very important aspect of web application development. Security is a dynamic research domain, where new techniques are introduced whereas some older ones are declared deprecated. A working understanding of the basic security traits and the new techniques introduced to implement them leads to the designing of an effective security model. Also, the objective of the application should be analysed to find an acceptable balance between the security techniques implemented and their effect on the performance of the system.

REFERENCES

1. Bernstein, Daniel J. "The Salsa20 family of stream ciphers." New stream cipher designs. Springer, Berlin, Heidelberg, 2008. 84-97.
2. Bernstein, Daniel J. "The ChaCha family of stream ciphers." DJ Bernstein's Webpage.
3. Klein, Andreas. "Attacks on the RC4 stream cipher." Designs, Codes and Cryptography 48.3 (2008): 269-286.
4. Stallings, William. Cryptography and network security: principles and practices. Pearson Education India, 2006.
5. Mousa, Allam. "Data encryption performance based on Blowfish." ELMAR, 2005. 47th International Symposium. IEEE, 2005.
6. Schneier, Bruce, et al. "Twofish: A 128-bit block cipher." NIST AES Proposal 15 (1998): 23.
7. Basu, Sandipan. "International Data Encryption Algorithm (Idea)–A Typical Illustration." International Journal of Global Research in Computer Science (UGC Approved Journal) 2.7 (2011): 116-118.
8. ElGamal, Taher. "A public key cryptosystem and a signature scheme based on discrete logarithms." IEEE transactions on information theory 31.4 (1985): 469-472.
9. McGrew, David, and John Viega. "The Galois/counter mode of operation (GCM)." submission to NIST Modes of Operation Process 20 (2004).
10. Damiani, Ernesto, et al. "Balancing confidentiality and efficiency in untrusted relational DBMSs." Proceedings of the 10th ACM conference on Computer and communications security. ACM, 2003.
11. Patel, Anup, Niveeta Sharma, and MagdaliniEirinaki. "Negative database for data security." Computing, Engineering and Information, 2009. ICC'09. International Conference on. IEEE, 2009.
12. Dubey, Gaurav, Vikram Khurana, and Shelly Sachdeva. "Implementing security technique on generic database." Contemporary Computing (IC3), 2015 Eighth International Conference on. IEEE, 2015.
13. Aumasson, Jean-Philippe, et al. "BLAKE2: simpler, smaller, fast as MD5." International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, 2013.
14. Mulwani, Kunal, et al. "3LAS (three level authentication scheme)." International Journal of Emerging Technology and Advanced Engineering 3 (2013): 103-107.
15. Eldefrawy, Mohamed Hamdy, Khaled Alghathbar, and Muhammad Khurram Khan. "OTP-based two-factor authentication using mobile phones." Information Technology: New Generations (ITNG), 2011 Eighth International Conference on. IEEE, 2011.
16. Sandhu, Ravi S., and PierangelaSamarati. "Access control: principle and practice." IEEE communications magazine 32.9 (1994): 40-48.
17. Joshi, James BD, et al. "Security models for web-based applications." Communications of the ACM 44.2 (2001): 38-44.
18. Harrington, Anthony, and Christian Jensen. "Cryptographic access control in a distributed file system." Proceedings of the eighth ACM symposium on Access control models and technologies. ACM, 2003.
19. Chang, Chin-Chen, and Chao-Wen Chan. "A database record encryption scheme using the RSA public key cryptosystem and its master keys." Computer Networks and Mobile Computing, 2003. ICCNMC 2003. 2003 International Conference on. IEEE, 2003.
20. Akl, Selim G., and Peter D. Taylor. "Cryptographic solution to a problem of access control in a hierarchy." ACM Transactions on Computer Systems (TOCS) 1.3 (1983): 239-248.
21. Cao, Ying-yu, and Chong Fu. "An efficient implementation of RSA digital signature algorithm." Intelligent Computation Technology and Automation (ICICTA), 2008 International Conference on. Vol. 2. IEEE, 2008.
22. Johnson, Don, Alfred Menezes, and Scott Vanstone. "The elliptic curve digital signature algorithm (ECDSA)." International journal of information security 1.1 (2001): 36-63.
23. Salini, P., and S. Kanmani. "Security requirements engineering process for web applications." Procedia Engineering 38 (2012): 2799-2807.