Requirements Based Web Application Security Testing – A Preemptive Approach!

Bhushan B. Gupta

Principal, Gupta Consulting, LLC.

bhushan@bgupta.com https://www.bgupta.com/

Abstract

Most web application security testing efforts are concentrated around penetration testing which is an art based on hacker's psyche, thought process, and determination to exploit software vulnerabilities. But, does it yield a high level of confidence and sense of security in a developer's mind? The answer is a "may be" especially when the bad guy is obsessed with figuring out new exploits to hack your application. The web application developers have begun to think about intrinsic security that is, building security throughout the SDLC. We build applications based upon well-formed customer requirements. Why should we not, then, build our applications based upon the fundamental principles of security and then harden security from the hacker's perspective?

This paper discusses an approach that aligns the web application security testing with the three basic principles of security namely, Confidentiality, Integrity, and Availability. The approach first establishes the requirements dictated by each element of CIA especially Confidentiality as it places the most stringent requirements on an application. Using the STRIDE model, the paper illustrate the most vulnerable processes in an application thus highlighting the test-intensive areas. It then deduces acceptance criteria and illustrates thought process to develop a test plan which spans over both static and dynamic (traditional testing) code analysis. The paper continues to demonstrate how to apply the DREAD model to prioritize the vulnerabilities found during testing to facilitate the removal of the most critical vulnerabilities first.

Biography

Bhushan Gupta is passionate about development methods and tools that yield more secure web applications especially in the agile software development environment. As a researcher he has keen interest in understanding and applying fundamental principles and known methodologies to develop dependable solutions. His interests extend to Social Engineering and Attack Surface Analysis. Bhushan worked at Hewlett-Packard for 13 years in various roles including software quality lead, engineer, software process architect, and software productivity manager. He then developed a strong interest in web application security while working as a quality engineer for Nike Inc. Bhushan has been studying various facets of web application security and promoting how to apply common sense approach to build secure solutions. He is a certified Six Sigma Black Belt (HP and ASQ) and an adjunct faculty member at the Oregon Institute of Technology in Software Engineering.

1 Introduction

The “penetration testing” (mostly referred to as Pen Testing) of a web application is an approach in which a test engineer takes on the characteristics of a hacker and tries to exploit the application vulnerabilities. Most organizations include OWASP Top 10 vulnerabilities [OWASP, 2013] in their testing scope. While that can check a system for known vulnerabilities, there are no assurances that the application will not be compromised for any other shortcomings. To really Pen Test an application a test engineer should pursue the approach that hackers take and test the application for all potential hacks. This makes the test metrics a huge task and given the urgency to release the organizations simply do not have the resources, both personnel and time, to shake out the application.

What if we pursue an approach where we first establish the security requirements and then derive an acceptance criteria that can help us establish a robust test plan? It is the same approach we have learnt to appreciate for testing the software quality. We can and should integrate security as an attribute of quality so that it becomes an intrinsic aspect of the application. Such an approach will provide us a known confidence level into security and will be preemptive as oppose to reactive. We will devote this article to understanding of the security requirements and how they relate to known OWASP 10, how to develop an acceptance criteria for each requirement, develop a test plan, and carrying out the test activities to gain a high level of security confidence in our application.

2 Hacking and Pen Testing

From the first impressions, hacking sounds like a disorganized activity without any structure. The fact is that hacking is a step by step activity where every proceeding step is based upon the results of the step just completed. It is entirely possible to take path that will lead to a failure. Kim [Kim, 2015] has explained hacking using the analogy of the American football. Kim has described each activity in football terms, before the snap, the drive, the throw, the lateral pass, the onside kick etc. etc. Each phase has a set of activities towards the goal. Before the Snap is the discovery phase in which the bad guy assesses his target. This phase includes target scanning to gather pertinent information. In the Drive phase the bad guy takes all the information gathered in Before the Snap phase and uses these exploits to get his foot in the door. The web application vulnerabilities are then manually targeted in the Throw phase. In the lateral pass the bad guy moves through the network exploiting backdoor. In the screen phase the focus shifts to client attack using social engineering techniques and then the onside kick where the bad guy looks for the physical access. This is a bird’s eye view of hacking.

The Pen testing follows the same approach. As a matter of fact the Pen testing is carried out from the hacker’s perspective to provide some assurance that the potential vulnerabilities have been examined and any potential exploits have been plugged in. The success of Pen testing depends upon the reconnaissance, the tools used, understanding attacker profile, and the scope – the types of vulnerabilities included the scope of testing. Each of these factors requires a significant knowledge to execute and gain a comfortable level of confidence. To make things worst new tool kits are developed for hacking and the time is an advantage for hackers since they are not on a schedule and the application release is. It only makes a business sense to deploy alternative approaches to security testing as we do for the other attributes of quality to achieve a level of confidence that we desire.

3 A Look into the Past

It will be educational to review a few significant security breaches to understand the root cause behind the exploitation. Armerding [Armerding, 2017], CSO, has provided a list of 15 worst data security breaches in the 21st century based upon how security practitioners weigh them. While this discussion will not be devoted to each breaches, it will highlight a few with the potential root cause. This section will also discuss a few malware that have significantly impacted he Web Application security Yahoo (2013-14)

In September 2016, Yahoo reported a security breaches that impacted 1.5 Billion user accounts involving two incidents. Yahoo was at a critical negotiation stage of selling itself to Verizon. The root cause for these breach was cookie forging by the hackers to falsify login credentials that led to getting into accounts that did not have password setup. Yahoo had the cookies encrypted with bcrypt.

3.1 eBay (May 2014)

In May 2014 eBay reported that 145 Million user records were compromised phishing, a social engineering technique. The hackers compromised three corporate employees accounts and got into the company network

3.2 Heartland Payment Systems (March 2008)

In this breach 134 million customer cards were exposed and the breach was caused using SQL Injection.

3.3 Target Stores (Dec. 2013)

Debit/credit card information of up to 110 million customers was stolen from the Target store. The hackers got access to Target POS (Point of Sale System) through a third party HVAC vendor. The hacker managed to load BlackPOS malware on the Target POS system and collected the customer data as the credit card purchases were made. The network credentials were stolen from a third party HVAC vendor.

The discussion below looks at a few malwares that have had had a significant impact on the Web Application security.

3.4 Heartbleed (CVE-2014-0160)

The transport layer security uses OpenSSLCryptography library to provide a secure communication between the client and the server. The library had a buffer overflow vulnerability caused due to the lack of bounds checking. As a result, a user was able to obtain the memory contents from the server that included the login credentials of the client. The bug was secretly reported on April 1, 2014 and patched on April 7, 2014. It is important to note that the vulnerability was caused by a programming error that could have been prevented by a code review.

To summarize this section, it is evident that the web application breaches are caused by either software bugs (shortcomings) or social engineering, coercion by hackers to exploit a vulnerabilities. In the rest of the paper we will discuss how the development group take a preemptive approach to eliminate potential for vulnerabilities.

4 Security Principles and Requirements

Security is bounded by three variables, **C**onfidentiality, **I**ntegrity, and **A**vailability (CIA) and balancing these variable yield a secure and a usable web application [Harris, 2013]. Overemphasizing one will lead to constraining the other two. For example, keeping an article of common interest in a vault will provide extreme confidentiality but zero availability. In the software engineering realm, if the core a web application or a software is highly secure, it will be very difficult to grow that application. A very secure operating system is very constrained had will have very limited expansion opportunities. The following discussion highlights the requirements for each element:

4.1 Confidentiality

Confidentiality is the assurance that information is not disclosed to unauthorized individuals, programs, or processes [Harris, 2013]. This definition leads to a well-designed access controls for any unauthorized activity and thus entails elements Identification, authentication, and authorization and audit. Each of these elements places unique requirements discussed below:

4.1.1 Identification

Identification is the process of uniquely recognizing an entity before letting it use the system. In most cases it is a unique string of characters such as a name, an email address, or an account number. In the simplest scenario the security involves verifying the string with the access control list. The methods to assure that the string is not programmatically generated are now emerging, enter a system displayed text, where the entity has to perform an additional task to prove the physical existence.

The authentication as the following requirements:

- Each user has a unique ID
- ID should follow a standard convention if needed
- A user ID is not shared with the other users
- ID value is not reflective of position or role
- Two factor authentication works adequately (if supported)
- CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) if needed

4.1.2 Authentication

Authentication process assures that a user has successfully proven to be a legitimate entity to utilize the system. Authentication is based on three aspects – something specific you know, something specific you have, and a characteristic unique to you. Something specific you know is normally a password the user has setup. The password is sometimes reinforced with a set of questions and answers, which also is a form of something you know.

Something you have can be a string of characters randomly generated by a hardware device such as a pin or a physical object, a card that you swipe to get access. A physical characteristic unique to you is biometric aspect such as your figure print or cornea. A strong authentication should include two factors.

The requirements for authentication will be driven by the type of mechanism the application is using. For a password (string of characters), a typical set of requirements will be:

- Complexity/Crackability–
    - Difficult to guess (minimal length, required character categories, prohibitive elements –last name, date of birth)
    - Should not require extra efforts to remember to avoid noting it down
- Failure/Recovery Process
    - Number of attempts before Time Out or Locking Out
    - Use of security questions for first login attempt from a new device
    - Recovery Mechanism – controlled such as email mechanism or on the fly (change it while on the site, not being sent in an email or only as a temporary password)
    - No email distribution

4.1.3 Authorization

Not every user needs access to all system resources, programs, processes, or data. The access to these resources should be controlled by an access criteria based upon the security policy. These criteria may vary from “No Access” as a default to “Need to Know”. A sample set of requirements for authorization will be:

- Policy to control access to objects – database servers
- Processes are also treated as subjects

4.1.4 Audit

A sound security system must include an audit log of both, failed and successful access attempts. Repeated unsuccessful access attempts are of particular interest to avoid a potential security breach. The successful attempts are of critical importance if a breach has occurred to perform a detective analysis.

- Maintaining read only audit logs
- Compliance with the legal requirements
- Deploying audit trail analysis tools to review logs

4.2 Integrity

Integrity is upheld when the assurance of accuracy and reliability of information and system is provided and any unauthorized modification is prevented. [Harris, 2013]. Simply stated, Integrity is maintain data accuracy at all times. A small set of requirements will have the following:

- Integrity maintained while data is at rest or in transit
- Role based access control
- Any malicious attempts logged with adequate tracking

This minimal set can be enhanced based upon the as needed based upon the application use case scenarios.

4.3 Availability

Availability protection ensures reliability and timely access to data and resources to authorized individuals. Listed below is a small set of requirements for availability:

- Available as needed (24x7x365 or as per other criteria)
- Redundancy to reinforce availability

In a well-defined and practiced SDLC, the requirements must be comprehensive, agreed upon by all stakeholders, and signed off by security/product owner. A set of rigorous requirements will lead to a reliable foundation of application security.

5 Deriving an Acceptance Criteria

This section highlights an excerpt of the acceptance criteria for “Identification”. Each requirement generates a set of criteria.

6 Creating a Test Plan

6.1 Agile Development Iterations

6.2 Vulnerability Risk Assessment Using DREAD Model

7 Integration into SDLC

8 Conclusion

References

1. OWASP (Open Web Application Security Project), Top 10 https://www.owasp.org/index.php/Top_10_2017-Top_10
2. Kim, Peter, The Hacker Playbook 2, Secure Planet LLC., July 2015
3. Armerding, Taylor, The 15 worst security breaches of the 21st Centuary,
4. http://www.csoonline.com/article/2130877/data-protection/data-protection-the-15-worst-data-security-breaches-of-the-21st-century.html
5. Harris, Shon, All in One CISSP, 6th Edition, McGraw Hill, 2013