Web Application Security Assessment

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering

By

Sandeep Hodkasia (161260)

Under the supervision of

Mr. Anand Prakash
[CEO & Founder, Appsecure Inc.]

to

विद्या तत्व ज्योतिसमः

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat,

Solan-173234, Himachal Pradesh

[illegible] 28/05/2020

[illegible]

Candidate's Declaration

I hereby declare that the work presented in this report entitled **Web Application Security Assessment Scanning** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from February 2020 to May 2020 under the supervision of **Mr. Anand Prakash,** CEO & Founder Appsecure Inc.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

[illegible] 28/05/2020

Sandeep Hodkasia, 161260

This is to certify that the above statement made by the candidate is true to the best of my knowledge

Anand Prakash

Digitally signed by Anand Prakash
Date: 2020.06.05 13:06:04 +05'30'

Anand Prakash

CEO & Founder

Appsecure Inc., Bangalore

Acknowledgment

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to **Mr. Anand Prakash** for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

We would like to express our gratitude towards our parents and Jaypee University of Information Technology for their kind cooperation and encouragement which helped us in the completion of this project.

Our thanks and appreciations also go to our colleagues in developing the project and people who have willingly helped us out with their abilities.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In recent years a lot of internet companies and internet based startups have been released in the world. At the same time, cyber attacks against websites and mobile applications have also increased. In such a situation, conducting the security audit and web security assessment would be a good approach for the startups and companies to protect their product from hackers and malicious users. Appsecure provides continuous security audit and web application security assessment to the partner companies for securing their web application from hack and data breach.

Chapter-1

Introduction

1.1 Introduction - Web Security Assessment

A web security assessment identifies, distinguishes, evaluates, and executes key security controls in applications. It likewise centers around forestalling application security deformities and vulnerabilities.

Completing a security assessment permits an organization to see the application portfolio comprehensively—from an attacker's point of view. It bolsters administrators in making educated asset designation, tooling, and security control usage choices. Along these lines, directing an appraisal is an indispensable piece of an association's hazard the executives procedure. The Web security assessments includes:

1. Vulnerability Assessment
2. Penetration testing
3. Client-Side Scanning
4. Source Code Analysis
5. Reverse Engineering
6. Network Scanning
7. Identifying Cloud leak

How Do Companies Carry Security Assessment and How does it works?

Factors, for example, size, development rate, assets, and resource portfolio influence the profundity of hazard appraisal models. Associations can do summed up appraisals when encountering spending plan or time requirements. In any case, summed up evaluations don't really give the definite mappings between resources, related dangers, distinguished dangers, sway, and alleviating controls.

Whenever summed up appraisal results don't give a sufficient connection between these territories, a more top to bottom evaluation is vital.

The 4 stages of an effective security assessment model are:

1. **Recognizable proof**. Decide every single basic resource of the innovation foundation. Next, analyze touchy information that is made, put away, or transmitted by these advantages. Make a hazard profile for each.
2. **Evaluation**. Regulate a way to deal with survey the recognized security dangers for basic resources. After cautious assessment and evaluation, decide how to successfully and effectively assign time and assets towards chance relief. The evaluation approach or system must break down the connection between advantages, dangers, vulnerabilities, and relieving controls.
3. **Alleviation**. Characterize an alleviation approach and uphold security controls for each hazard.
4. **Avoidance**. Execute apparatuses and procedures to limit dangers and vulnerabilities from happening in your company's assets.

1.2 How CVSS Score is Calculated ?

CVSS Score is an important factor in measuring the vulnerabilities discovered during web security assessment. These includes following parameters:

1. **Confidentiality, Integrity, and Availability**
   Depicts the essential security targets, which are completely principal to getting security
2. **Vulnerabilities**
   Characterizes the significant classifications of vulnerabilities and examines the nearness of vulnerabilities in all product
3. **Dangers**
   Quickly presents significant risk ideas
4. **Security Controls**

Characterizes significant classes of security controls and talks about their potential weaknesses

1.3 Web Application Security Assessment Vulnerabilities

A product imperfection powerlessness is brought about by an unintended blunder in the structure or coding of programming. A model is an info approval blunder, for example, the client gave input not being appropriately assessed for pernicious character strings and excessively long qualities related to known assaults. Another model is a race condition blunder that enables the assailant to play out a particular activity with raised benefits.

A security arrangement setting is a component of a product's security that can be adjusted through the product itself. Instances of settings are a working framework offering access to control records that set the benefits that clients have for documents, and an application offering a setting to empower or incapacitate the encryption of touchy information put away by the application.

A security arrangement issue weakness includes the utilization of security setup settings that adversely influence the security of the product.

A product highlight is a practical capacity given by programming. A product highlighting abuse powerlessness is a helplessness wherein the element likewise gives a road to bargain the security of a framework. These vulnerabilities are brought about by the product originator making trust suspicions that license the product to give valuable highlights, while likewise presenting the probability of somebody disregarding the trust suppositions to bargain security.

For instance, email customer programming may contain an element that renders HTML content in email messages. An assailant could make a false email message that contains hyperlinks that, when rendered in HTML, appear to the beneficiary to be considerate however take the beneficiary to a malignant site when they are tapped on. One of the trust presumptions in the structure of the HTML content rendering highlight was that clients would not get malignant hyperlinks and snap on them.

Programming highlight abuse vulnerabilities are presented during the plan of the product or a segment of the product (e.g., a convention that the product actualizes). Trust suppositions may have been expressed - for instance, an architect monitoring a security shortcoming and discovering that a different security control would make up for it.

In any case, trust suspicions are regularly certain, for example, making an element without first assessing the dangers it would present. Dangers may likewise change over the lifetime of programming or a convention utilized in programming.

For instance, the Address Resolution Protocol (ARP) believes that an ARP answer contains the right mapping between Media Access Control (MAC) and Internet Protocol (IP) addresses. The ARP reserve utilizes that data to give valuable assistance—to empower sending information between gadgets inside a nearby system. Be that as it may, an assailant could create bogus ARP messages to harm a framework's ARP table and in this way dispatch a refusal of-administration or a man-in-the-center assault.

The ARP convention was institutionalized more than 25 years back, and dangers have changed a lot from that point forward, so the trust suspicions characteristic in its structure at that point are probably not going to at present be sensible today.

It might be difficult to separate programming highlight abuse vulnerabilities from the other two classifications. For instance, both programming blemishes and abuse vulnerabilities might be brought about by lacks in programming configuration forms. Be that as it may, programming blemishes are negative—they give no positive advantage to security or usefulness—while programming highlights abuse vulnerabilities happen because of giving extra highlights.

There may likewise be perplexity concerning abuse vulnerabilities for highlights that can be empowered or crippled—as it were, arranged—versus security design issues. The key contrast is that for abuse powerlessness, the arrangement setting empowers or impairs the whole component and doesn't explicitly adjust only its security; for a security design issue defenselessness, the setup setting modifies just the product's security.

For instance, a setting that cripples all utilization of HTML in messages significantly affects both security and usefulness, so powerlessness identified with this setting would be an abuse defenselessness. A setting that debilitates the utilization of an anti phishing highlight in an email customer significantly affects just security, so helplessness with that setting would be viewed as a security setup issue weakness.

1.4 Exploits

A Exploit is any situation with the possibility to unfavorably affect information or frameworks through unapproved gets to, obliteration, revelation, or alteration of data, as well as disapproval of administration. Dangers may include deliberate entertainers (e.g., aggressor who needs to get to data on a server) or accidental on-screen characters (e.g., the executive who neglects to impair client records of a previous worker.) Threats can be nearby, for example, a displeased representative, or remote, for example, an assailant in another topographical territory.

A danger source is a reason for risk, for example, an antagonistic digital or physical assault, a human blunder of oversight or commission, a disappointment of association controlled equipment or programming, or other disappointment outside the ability to control of the association. A risk occasion is an occasion or circumstance started or brought about by a danger source that has the potential for causing an unfriendly effect.

Numerous dangers against information and assets are conceivable as a result of mix-ups—either bugs in the working framework and applications that make exploitable vulnerabilities or blunders made by end clients and executives.

System traffic commonly goes through middle PCs, for example, switches, or is persisted unbound systems, for example, remote hotspots. Along these lines, it tends to be captured by an outsider. Dangers against organizing traffic incorporate the accompanying:

Eavesdropping: Data stays unblemished, yet its security is undermined. For instance, somebody could get familiar with your charge card number, record a delicate discussion, or block arranged data.

Tampering: Data in travel is changed or supplanted and afterward sent on to the beneficiary. For instance, somebody could modify a request for merchandise or change an individual's resume.

Impersonation: Data goes to an individual who acts like the expected beneficiary. Pantomime can take two structures:

○ Unauthorization: An individual can claim to be another person. For instance, an individual can claim to have the email address jdoe@example.net, or a PC can distinguish itself as a site called www.example.net when it isn't. This sort of pantomime is known as ridiculing.

○ Deception: An individual or association can distort itself. For instance, assume the site www.example.net professes to be a furniture store when it is extremely only a site that assumes praise card installments however never sends any merchandise.

1.5 Security Controls

Sensitive information ought to be ensured dependent on the potential effect of lost classification, uprightness, or accessibility. Insurance measures (also called security controls) will in general fall into two classes. To begin with, security shortcomings in the framework should be settled. For instance, if a framework has known helplessness that assailants could abuse, the framework ought to be fixed so the defenselessness is evacuated or relieved. Second, the framework should offer just the necessary usefulness to each approved client, with the goal that nobody can utilize capacities that are a bit much. This guideline is known as the least benefit. Constraining usefulness and settling security shortcomings have a shared objective: give aggressors as not many open doors as conceivable to rupture a framework.

There are three sorts of security controls, as pursues:

1. The board controls: The security controls that emphasis on the administration of hazard and the administration of data framework security.

2. Operational controls: The security controls that are principally actualized and executed by individuals (instead of frameworks).

3. Specialized controls: The security controls that are essentially actualized and executed by the framework through the framework's equipment, programming, or firmware.

Every one of the three kinds of controls is important for strong security. For instance, a security strategy is an administration control, yet its security prerequisites are actualized by individuals (operational controls) and frameworks (specialized controls). Consider phishing assaults. An association may have a worthy use arrangement that determines the lead of clients, including not visiting pernicious sites. Security controls to help foil phishing, other than the administrative control of the adequate use strategy itself, incorporate operational controls, for example, preparing clients not to succumb to phishing tricks, and specialized controls that screen messages and site use for indications of phishing action.

A typical issue with security controls is that they frequently make frameworks less helpful or progressively hard to utilize. At the point when ease of use is an issue, numerous clients will endeavor to bypass security controls; for instance, if passwords must be long and complex, clients may record them. Adjusting security, usefulness, and convenience is frequently a test. The objective ought to be to find some kind of harmony:

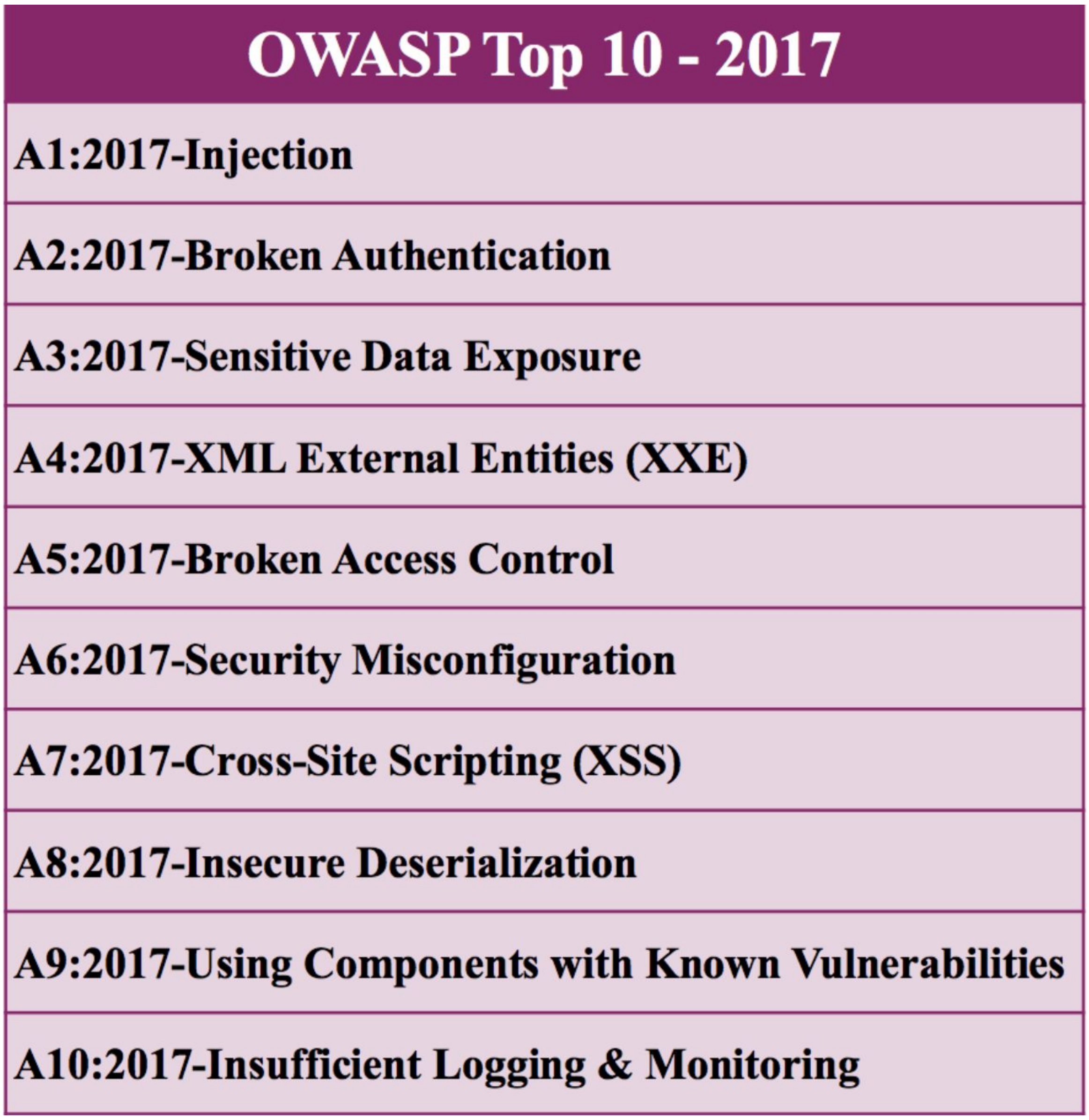
give a sensibly secure arrangement while offering the usefulness and ease of use that clients require. Another central guideline with security controls is utilizing various layers of security—safeguard inside and out. For instance, touchy information on a server might be shielded from outside assault by a few controls, including a system based firewall, a host-based firewall, and OS fixing. The inspiration for having different layers is that on the off chance that one layer comes up short or generally can't neutralize a specific risk, different layers may keep the danger from effectively breaking the framework. A mix of a system based and have based controls is commonly best at giving steady assurance.

Chapter-2

LITERATURE SURVEY

2.1 OWASP Top 10 Web

The OWASP Top 10 is an amazing mindfulness archive for web application security. It speaks to an expansive agreement about the most basic security dangers to web applications. Venture individuals incorporate an assortment of security specialists from around the globe who have shared their aptitude to deliver this rundown.



OWASP Top 10 - 2017
A1:2017-Injection
A2:2017-Broken Authentication
A3:2017-Sensitive Data Exposure
A4:2017-XML External Entities (XXE)
A5:2017-Broken Access Control
A6:2017-Security Misconfiguration
A7:2017-Cross-Site Scripting (XSS)
A8:2017-Insecure Deserialization
A9:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring

Table 2.1: OWASP top 10 Web

1. Injection

Injections are among the most established and most perilous assaults to any web application. In this assault, the assailant essentially sends malignant information, so as to make the application procedure and accomplish something it should do. Infusion imperfections are extremely pervasive, especially in inheritance code. The center explanation being: User-provided information isn't approved, sifted or purified by the application.

Some other potential reasons include:

1. Utilizing dynamic questions straightforwardly in the translator.

2. Utilizing non-parameterized calls without executing setting mindful getting away (auto-getting away in short) straightforwardly in the translator. Peruse increasingly about getting away from all client provided contributions here.

3. Antagonistic information being utilized inside article social mapping (ORM) search parameters to remove extra, delicate records.

4. Unfriendly information being utilized or connected straightforwardly, with the end goal that the SQL or order contains both structure and threatening information in unique questions, orders, or put away techniques.

Here is OWASP's case of a SQL question expending untrusted information:

String question = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "'";

This question can be abused when the assailant alters the 'id' parameter esteem in their program to send: ' or '1'='1. For instance:

http://example.com/application/accountView?id=' or '1'='1 This can make the database table return all the columns.

To make sure about your application from injection assaults, here are a portion of OWASP's specialized suggestions to pay special mind to:

1. Forestalling infusion requires keeping information separate from orders and inquiries.

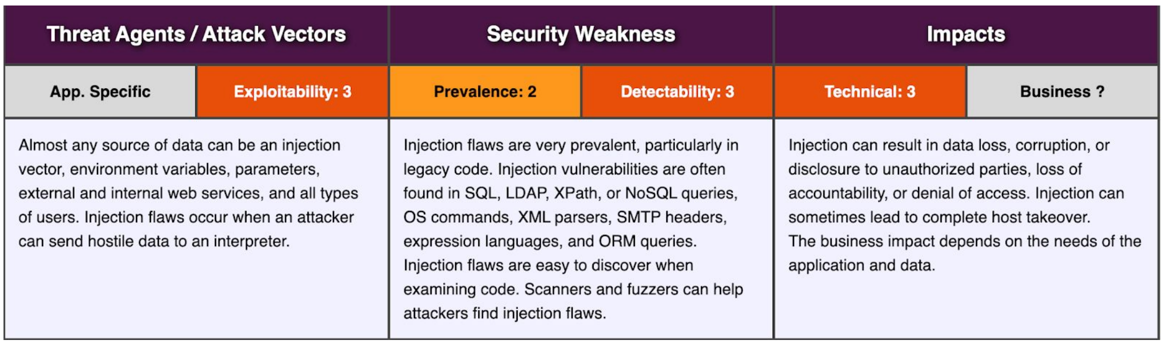
2. The favored alternative is to utilize a protected API, which stays away from the utilization of the mediator totally or gives a parameterized interface, or relocate to utilize Object Relational Mapping Tools (ORMs). Note: Even when parameterized, put away methodology can at present SQL infusion if PL/SQL or T-SQL links questions and information, or executes unfriendly information with EXECUTE IMMEDIATE or executive().

3. Utilize positive or "whitelist" server-side info approval. This is definitely not a total barrier the same number of utilizations require uncommon characters, for example, content regions or APIs for versatile applications.

4. For any leftover powerful inquiries, get away from uncommon characters utilizing the particular getaway sentence structure for that mediator. Note: SQL structure, for example, table names, segment names, etc can't be gotten away, and along these lines client provided structure names are risky. This is a typical issue in report-composing programming.

5. Use LIMIT and other SQL controls inside inquiries to forestall mass revelation of records if there should arise an occurrence of SQL infusion.

Code infusion can be lethal as it can bring about information misfortune or defilement, absence of responsibility, refusal of access or can at times lead to termination.



Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 3	Prevalence: 2	Detectability: 3	Technical: 3	Business ?
Almost any source of data can be an injection vector, environment variables, parameters, external and internal web services, and all types of users. Injection flaws occur when an attacker can send hostile data to an interpreter.		Injection flaws are very prevalent, particularly in legacy code. Injection vulnerabilities are often found in SQL, LDAP, XPath, or NoSQL queries, OS commands, XML parsers, SMTP headers, expression languages, and ORM queries. Injection flaws are easy to discover when examining code. Scanners and fuzzers can help attackers find injection flaws.		Injection can result in data loss, corruption, or disclosure to unauthorized parties, loss of accountability, or denial of access. Injection can sometimes lead to complete host takeover. The business impact depends on the needs of the application and data.	

Fig 2.1: Injection: Threat Agent, Security Weakness, Impacts

2. Broken Authentication

Making sure about client verification is a vital piece of making the web application more secure. To comprehend this weakness better, let us investigate how a run of the mill verification process resembles:

1. The client enters their login qualifications
2. The server confirms the certifications and afterward makes a meeting which is then put away on the server-side
3. A treat with the meeting ID is then put away on the client's program
4. On each ensuing solicitation made by the client, the meeting ID is checked against the worth put away on the server to process it. On the off chance that the ID differs, the validation fizzles and the solicitation isn't prepared effectively
5. This meeting is pulverized on both customer and server when the client logs out of the application

Presently, if there are any defects in this instrument, or if the confirmation procedure doesn't deal with the prescribed procedures in its execution, an aggressor can utilize manual or programmed mediums to attempt to get entrance over any record, or in the most pessimistic scenario, authority over the framework.

When is the application defenseless?

While the danger can come in numerous structures, these are some conspicuous cases which make the application defenseless against the assault:

1. Putting away passwords as plain-content. It is never a smart thought. Continuously hash the passwords with salt before putting away them, so that regardless of whether there is a database break, the certifications are not uncovered. Try not to utilize powerless hashes like SHA1 or MD5.
2. Leaving a login page for heads openly available to all guests of the site.
3. Not dealing with computerized assaults, for example, qualification stuffing, where the assailant has a rundown of legitimate usernames and passwords.
4. Not dealing with beast power or other robotized assaults.
5. Not checking the secret phrase against a rundown of most regular passwords.
6. Utilizing frail or inadequate accreditation recuperation and overlooked secret phrase forms, for example, information based answers, which can't be made safe.
7. Uncovering Session IDs in the URL. e. g. URL changing: http://example.com/deal/saleitems;jsessionid=2P0OC2JSNDLPSKHCJUN2JV?dest=Hawaii
8. Not turning meeting IDs after fruitful login.
9. Not appropriately refuting meeting IDs. Client meetings or verification tokens, especially single sign-on (SSO) tokens, aren't appropriately nullified during logout or a time of idleness.

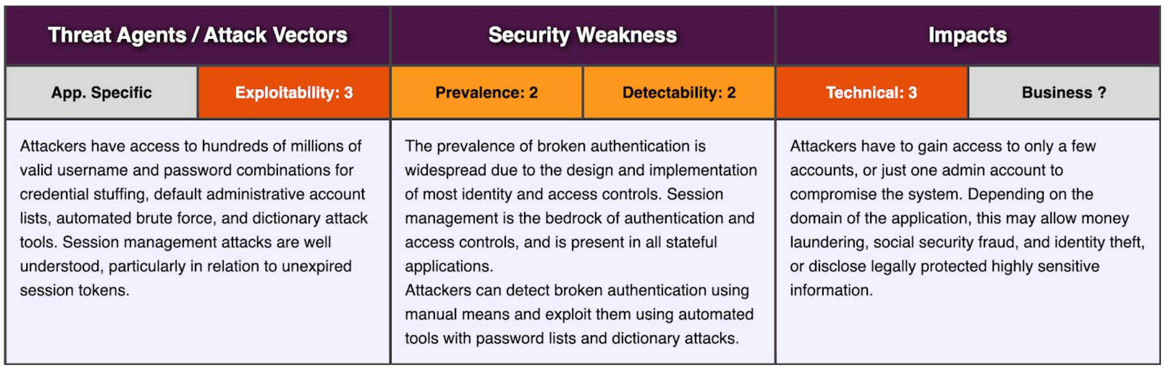
How might you forestall broken confirmation vulnerabilities?

To keep the application secure from this specific powerlessness, it is significant for the designer to know about the accepted procedures of site security. The code ought to be tried together before it is sent to creation.

Here are a portion of OWASP's specialized proposals to ensure your application is sheltered from these messed up verification vulnerabilities:

1. Utilize a server-side, secure, working meeting chief that produces another arbitrary meeting ID with high entropy after login.
2. Meeting IDs ought not be in the URL. IDs ought to likewise be safely put away and discredited after logout, inert, and supreme breaks.

3. At every possible opportunity, actualize multifaceted validation to forestall mechanized, accreditation stuffing, animal power, and taken qualification re-use assaults.
4. Try not to deliver or send with any default accreditations, especially for executive clients.
5. Execute feeble secret key checks, for example, testing new or changed passwords against a rundown of the main 10000 most exceedingly terrible passwords.
6. Adjust secret phrase length, intricacy and pivot strategies with NIST 800-63 B's rules in segment 5.1.1 for Memorized Secrets or other present day, proof based secret key approaches.
7. Guarantee enlistment, accreditation recuperation, and API pathways are solidified against account identification assaults by utilizing similar messages for all results. for example Verification disappointment reactions ought not show which part of the validation information was inaccurate. Rather than "Invalid username" or "Invalid secret key", simply use "Invalid username or potentially secret key" for both. Mistake reactions must be really indistinguishable in both showcase and source code.
8. Breaking point or progressively delay fizzled login endeavors. Log all disappointments and ready executives when accreditation stuffing, savage power, or different assaults are identified. The utilization of arrangements of realized passwords is a typical assault. In the event that an application doesn't execute robotized danger or certification stuffing assurance, the application can be utilized as a secret phrase prophet to decide whether the qualifications are substantial.



Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 3	Prevalence: 2	Detectability: 2	Technical: 3	Business ?
Attackers have access to hundreds of millions of valid username and password combinations for credential stuffing, default administrative account lists, automated brute force, and dictionary attack tools. Session management attacks are well understood, particularly in relation to unexpired session tokens.		The prevalence of broken authentication is widespread due to the design and implementation of most identity and access controls. Session management is the bedrock of authentication and access controls, and is present in all stateful applications. Attackers can detect broken authentication using manual means and exploit them using automated tools with password lists and dictionary attacks.		Attackers have to gain access to only a few accounts, or just one admin account to compromise the system. Depending on the domain of the application, this may allow money laundering, social security fraud, and identity theft, or disclose legally protected highly sensitive information.	

Fig 2.2: Broken Authentication: Threat Agents, Security Weakness, Impacts

3. Sensitive Data Exposure

As the name proposes, this security vulnerability occurs when the web application doesn't satisfactorily ensure touchy data like meeting tokens, passwords, banking data, area, wellbeing information, or whatever other comparative urgent information whose break can be basic for the client. This danger influences clients the most and can cause budgetary misfortune, access to the casualty's records, extorting which at last outcomes in diminished trust in the brand.

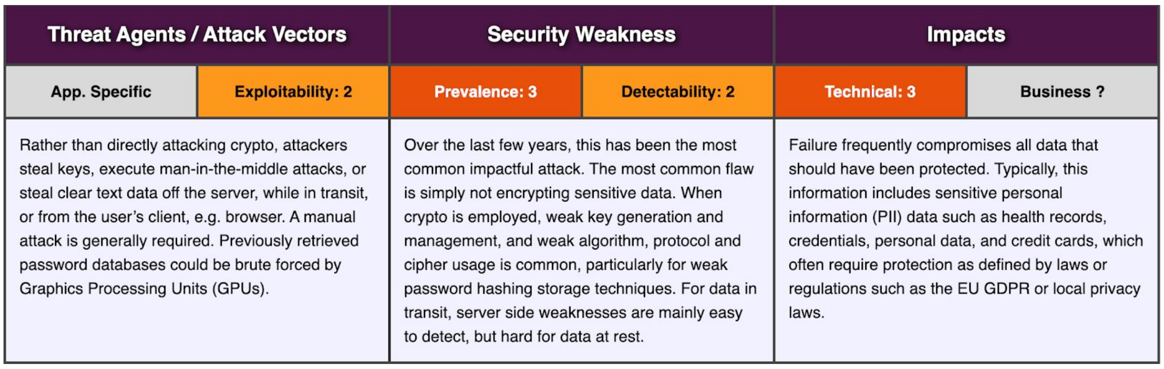
When is the application vulnerable?

1. Hardcoding information like tokens, secret_keys, passwords in the source code.
2. Logging touchy information in server logs.
3. Reserving delicate information.
4. Transmitting touchy data in plain content.
5. Utilizing old or powerless cryptographic calculations.
6. Utilizing default crypto keys, producing or re-utilizing frail crypto keys.
7. Client specialist (for example application, API) not approving got server declaration which can bring about a maverick server endeavoring to take on the appearance of a genuine server.
8. A SSL-empowered customer experiences the accompanying strides to verify a server's character:
    1. Is the present date inside the legitimacy time frame?
    2. Is the giving CA a confided in CA?
    3. Does the giving CA's open key approve the backer's advanced mark?
    4. Does the area name in the server's testament coordinate the space name of the server itself?

Prevention and Fixes :

1. It is critical to recognize what information gathered are delicate and group them. This can rely upon the sort of utilization, security laws, administrative prerequisites or business needs.

2. Apply get to controls on these information according to the order.
3. Try not to store delicate information pointlessly. Dispose of it at the earliest opportunity or use PCI DSS agreeable tokenization or even truncation. Not holding touchy information limits the hazard.
4. Try to encode every single delicate datum very still.
5. Guarantee that all encryption calculations are most recent, and solid, and that the relating conventions and keys are set up. Keys ought to be put away securely.
6. Handicap reserving for any reaction that contains touchy information.
7. Store passwords utilizing solid, versatile and salted hashing capacities with a work factor (defer factor, for example,
    - Argon2
    - scrypt
    - bcrypy
    - PBFKDF2
9. Encode all information in travel with secure conventions.

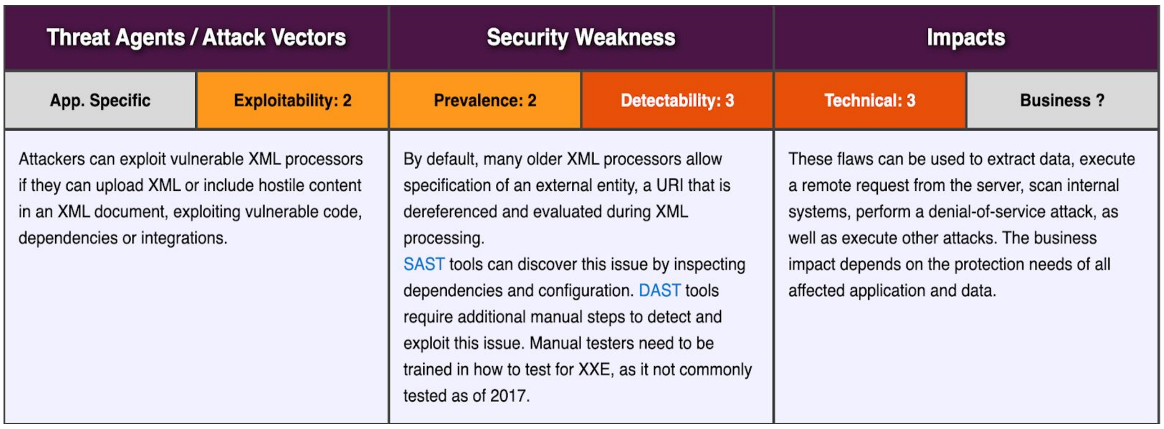


Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 2	Prevalence: 3	Detectability: 2	Technical: 3	Business ?
Rather than directly attacking crypto, attackers steal keys, execute man-in-the-middle attacks, or steal clear text data off the server, while in transit, or from the user's client, e.g. browser. A manual attack is generally required. Previously retrieved password databases could be brute forced by Graphics Processing Units (GPUs).		Over the last few years, this has been the most common impactful attack. The most common flaw is simply not encrypting sensitive data. When crypto is employed, weak key generation and management, and weak algorithm, protocol and cipher usage is common, particularly for weak password hashing storage techniques. For data in transit, server side weaknesses are mainly easy to detect, but hard for data at rest.		Failure frequently compromises all data that should have been protected. Typically, this information includes sensitive personal information (PII) data such as health records, credentials, personal data, and credit cards, which often require protection as defined by laws or regulations such as the EU GDPR or local privacy laws.	

Fig 2.3: Sensitive Data Exposure: Threat Agents, Security Weakness, Impacts

4. XML External Entities (XXE)

As indicated by Wikipedia, A XML External Entity assault is a kind of assault against an application that parses XML input. This assault happens when XML input containing a reference to an outer element is handled by a feebly designed XML parser.



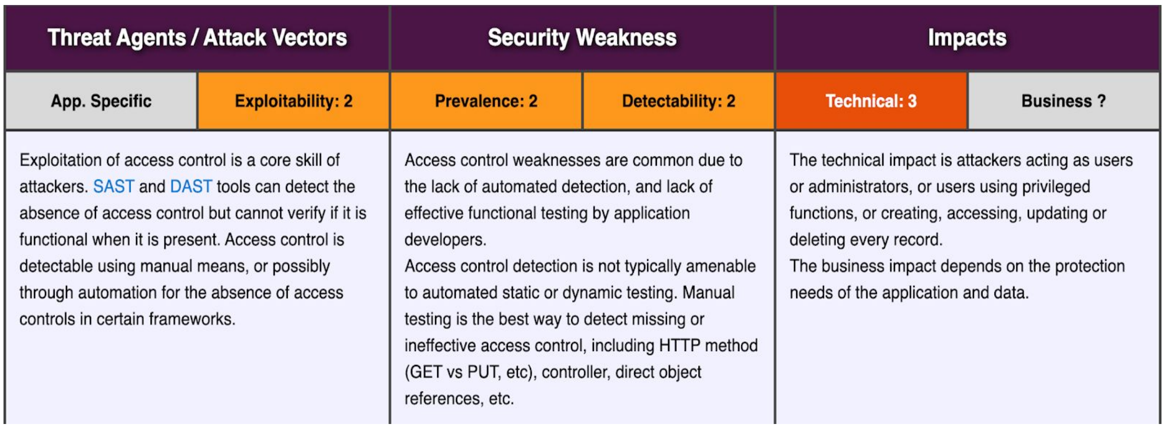
Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 2	Prevalence: 2	Detectability: 3	Technical: 3	Business ?
Attackers can exploit vulnerable XML processors if they can upload XML or include hostile content in an XML document, exploiting vulnerable code, dependencies or integrations.		By default, many older XML processors allow specification of an external entity, a URI that is dereferenced and evaluated during XML processing. SAST tools can discover this issue by inspecting dependencies and configuration. DAST tools require additional manual steps to detect and exploit this issue. Manual testers need to be trained in how to test for XXE, as it not commonly tested as of 2017.		These flaws can be used to extract data, execute a remote request from the server, scan internal systems, perform a denial-of-service attack, as well as execute other attacks. The business impact depends on the protection needs of all affected application and data.	

Fig 2.4: XXE: Threat Agents, Security Weakness, Impacts

5. Broken Access Control

In site security, get to control intends to set a cap for what areas or pages guests can reach, contingent upon their necessities.

For instance, in the event that you own a web based business store, you most likely need access to the administrator board so as to add new items or to set up an advancement for the forthcoming occasions. Be that as it may, scarcely any other individual would require it. Having the remainder of your site's guests have the option to come to your login page just opens up your web based business store to assaults. Also, that is the issue with practically all significant substance the executives frameworks (CMS) nowadays. Naturally, they give overall access to the administrator board. The vast majority of them additionally won't compel you to set up a second-factor confirmation strategy (2FA).



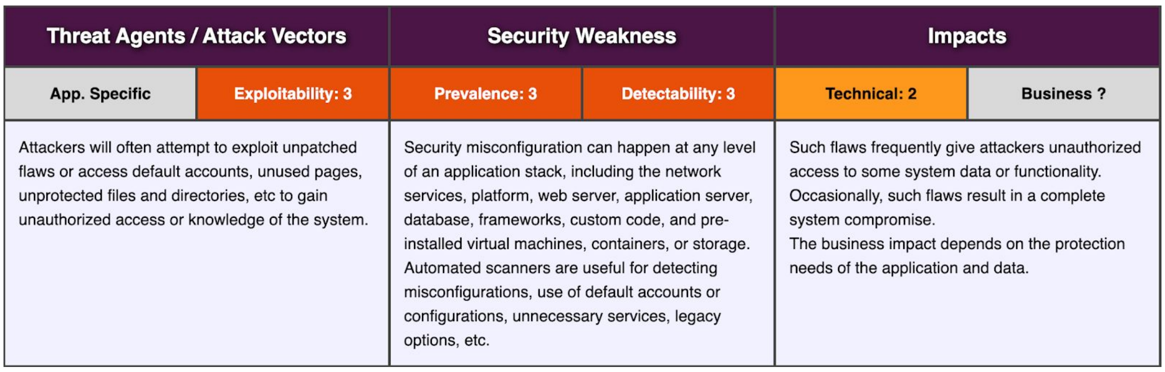
Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 2	Prevalence: 2	Detectability: 2	Technical: 3	Business ?
Exploitation of access control is a core skill of attackers. SAST and DAST tools can detect the absence of access control but cannot verify if it is functional when it is present. Access control is detectable using manual means, or possibly through automation for the absence of access controls in certain frameworks.		Access control weaknesses are common due to the lack of automated detection, and lack of effective functional testing by application developers. Access control detection is not typically amenable to automated static or dynamic testing. Manual testing is the best way to detect missing or ineffective access control, including HTTP method (GET vs PUT, etc), controller, direct object references, etc.		The technical impact is attackers acting as users or administrators, or users using privileged functions, or creating, accessing, updating or deleting every record. The business impact depends on the protection needs of the application and data.	

Fig 2.5:Broken Access Control: Threat Agents, Security Weakness, Impacts

6. Security Misconfigurations

Programmers are continually searching for approaches to infiltrate sites, and security misconfigurations can be a simple path in. Here are a few instances of things that programmers ordinarily attempt to misuse so as to increase unapproved get to:

1. unpatched defects
2. default arrangements
3. unused pages
4. unprotected documents and registries
5. superfluous administrations



Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 3	Prevalence: 3	Detectability: 3	Technical: 2	Business ?
Attackers will often attempt to exploit unpatched flaws or access default accounts, unused pages, unprotected files and directories, etc to gain unauthorized access or knowledge of the system.		Security misconfiguration can happen at any level of an application stack, including the network services, platform, web server, application server, database, frameworks, custom code, and pre-installed virtual machines, containers, or storage. Automated scanners are useful for detecting misconfigurations, use of default accounts or configurations, unnecessary services, legacy options, etc.		Such flaws frequently give attackers unauthorized access to some system data or functionality. Occasionally, such flaws result in a complete system compromise. The business impact depends on the protection needs of the application and data.	

Fig 2.6: Security Misconfiguration: Threat Agents, Security Weakness, Impacts

7. Cross-Site Scripting (XSS)

Cross Site Scripting (XSS) is a boundless defenselessness that influences many web applications. XSS assaults comprise of infusing pernicious customer side contents into a site and utilizing the site as a spread strategy. The threat behind XSS is that it permits an aggressor to infuse content into a site and change how it is shown, compelling a casualty's program to execute the code gave by the assailant while stacking the page.

Types of XSS:

1. **Reflected XSS**

Reflected XSS is the easiest type of cross-site scripting. It occurs when an application gets information in a HTTP demand and remembers that information inside the quick reaction for a risky way. Here is a basic case of a reflected XSS helplessness:

```
https://test.test?message=testing
<p>testing.</p>
```

The application doesn't play out some other handling of the information, so an aggressor can without much of a stretch develop an assault this way:

```
https://test.test/test?message=<script>/*+Blahblah+*/</script>
<p>Test <script>/* blahblah */</script></p>
```

In the event that the client visits the URL built by the assailant, at that point the aggressor's content executes in the client's program, with regards to that client's meeting with the application. By then, the content can complete any activity, and recover any information, to which the client approaches.

2. Stored XSS

Stored XSS occurs when an application gets information from an untrusted source and remembers that information inside its later HTTP response in a dangerous way.

The information being referred to may be submitted to the application by means of HTTP demands; for instance, remarks on a blog entry, client monikers in a talk room, or contact subtleties on a client request. In different cases, the information may show up from other untrusted sources; for instance, a webmail application showing messages got over SMTP, a promoting application showing online life posts, or a system observing application showing bundle information from arrange traffic.

Here is a basic case of a put away XSS helplessness. A message board application lets clients submit messages, which are shown to different clients:

```
<p>Hello SANDEEP!</p>
```

The application doesn't play out some other preparing of the information, so an aggressor can without much of a stretch communicate something specific that assaults different clients:

```
<p><script>/* XSS */</script></p>
```

3. DOM Based XSS

DOM-based XSS (otherwise called DOM XSS) emerges when an application contains some client side JavaScript that forms information from an untrusted source in a risky manner, for the most part by composing the information back to the DOM.

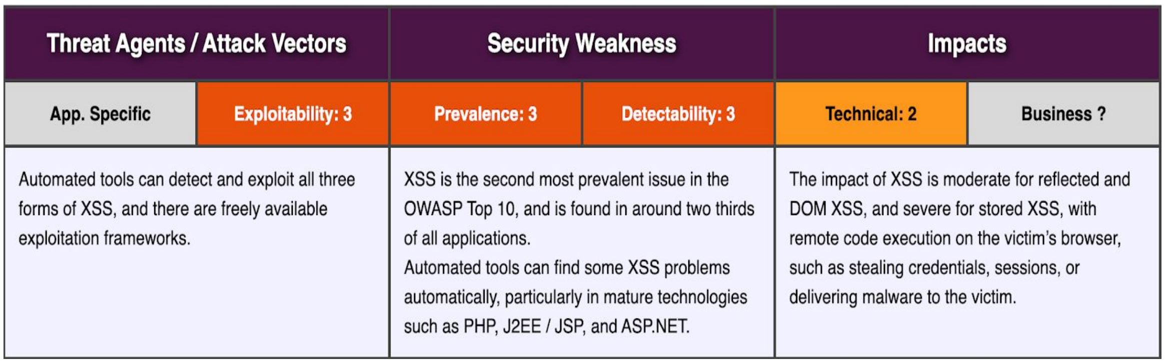
In the accompanying model, an application utilizes some JavaScript to peruse the incentive from an info field and compose that incentive to a component inside the HTML:

```
var search = document.getElementById(redirectURL).value;

var results = document.getElementById('results');

results.innerHTML = 'You scanned for: ' + search;
```

On the off chance that the aggressor can control the estimation of the information field, they can without much of a stretch build a malevolent worth that makes their own content execute:

```
You looked for: <img/src=111111111 onerror='/* XSS PAYLOAD */'>
```

In a common case, the info field would be populated from part of the HTTP demand, for example, a URL inquiry string parameter, permitting the assailant to convey an assault utilizing a malevolent URL, in a similar way as reflected XSS.

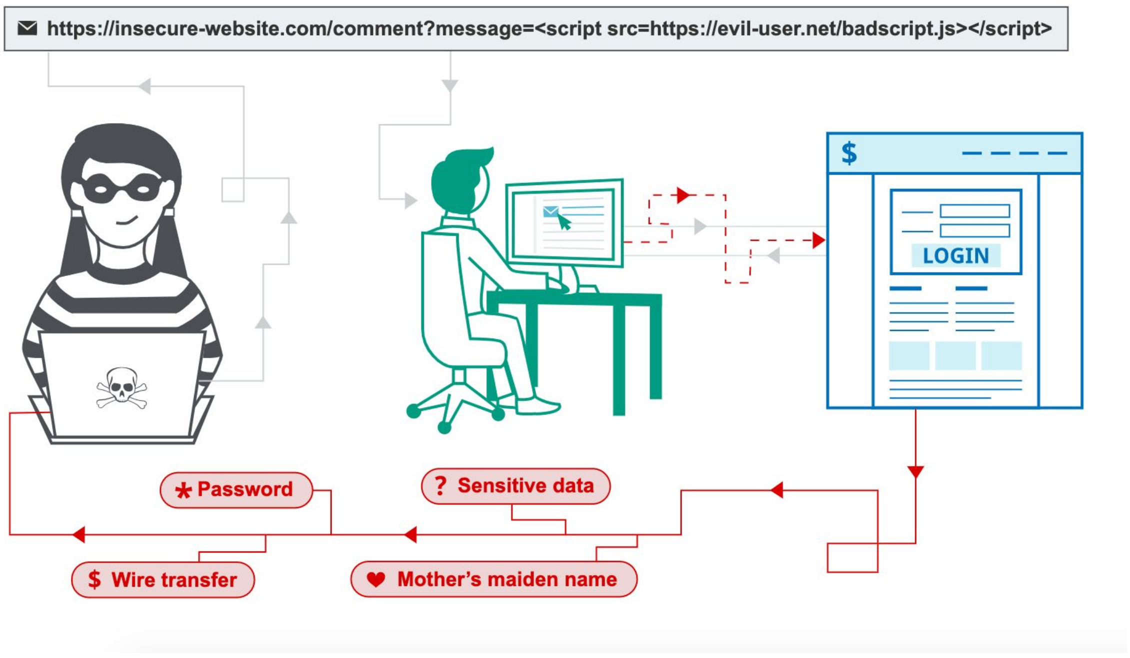


Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 3	Prevalence: 3	Detectability: 3	Technical: 2	Business ?
Automated tools can detect and exploit all three forms of XSS, and there are freely available exploitation frameworks.		XSS is the second most prevalent issue in the OWASP Top 10, and is found in around two thirds of all applications. Automated tools can find some XSS problems automatically, particularly in mature technologies such as PHP, J2EE / JSP, and ASP.NET.		The impact of XSS is moderate for reflected and DOM XSS, and severe for stored XSS, with remote code execution on the victim's browser, such as stealing credentials, sessions, or delivering malware to the victim.	

Fig 2.7: XSS: Threat Agents, Security Weakness, Impacts

How XSS Works?

When an xss payload gets executed in the victim's browser, the malicious script extract the victim's session identifiers and delivers it to the attacker.



```
https://insecure-website.com/comment?message=<script src=https://evil-user.net/badscript.js></script>
```

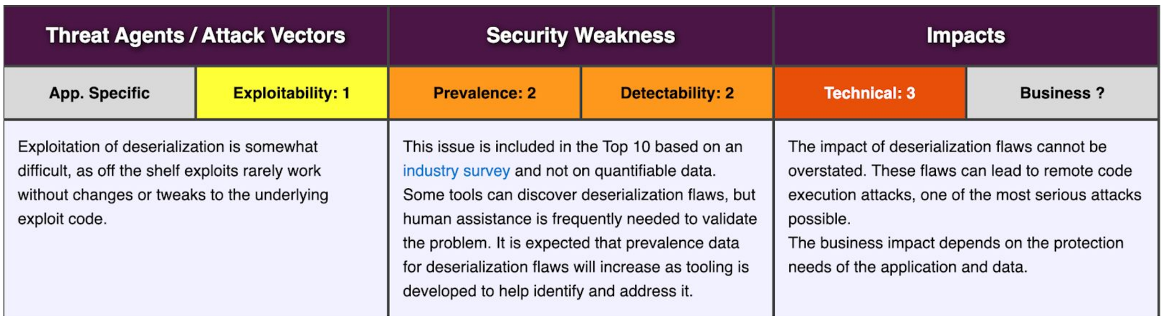
Fig 2.8: Working of XSS

8. Insecure Deserialization

Each web engineer needs to make harmony with the way that assailants/security specialists are going to attempt to play with everything that associates with their application–from the URLs to serialized objects.

In software engineering, an article is an information structure; at the end of the day, an approach to structure information. To make it more obvious two key ideas:

1. The procedure of serialization is changing over items to byte strings.
2. The procedure of deserialization is changing over byte strings to objects.

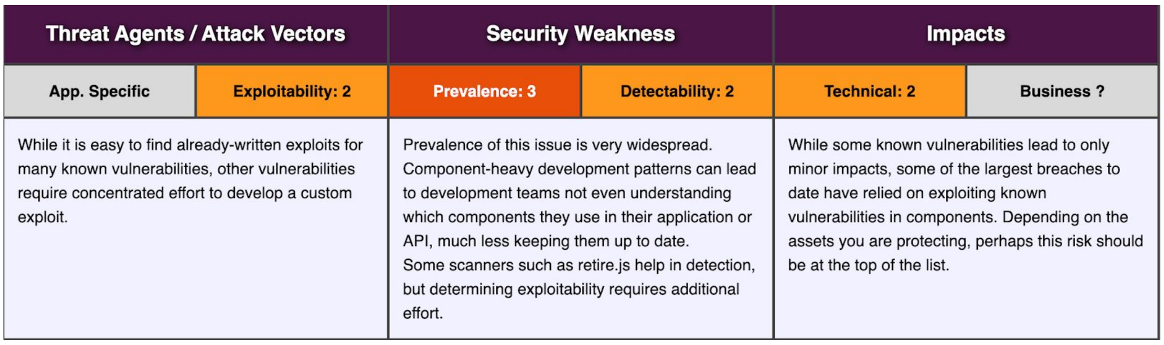


Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 1	Prevalence: 2	Detectability: 2	Technical: 3	Business ?
Exploitation of deserialization is somewhat difficult, as off the shelf exploits rarely work without changes or tweaks to the underlying exploit code.		This issue is included in the Top 10 based on an industry survey and not on quantifiable data. Some tools can discover deserialization flaws, but human assistance is frequently needed to validate the problem. It is expected that prevalence data for deserialization flaws will increase as tooling is developed to help identify and address it.		The impact of deserialization flaws cannot be overstated. These flaws can lead to remote code execution attacks, one of the most serious attacks possible. The business impact depends on the protection needs of the application and data.	

Fig 2.9: Insecure Deserialisation: Threat Agents, Security Weakness, Impacts

9. Utilizing Components with Known Vulnerabilities

Nowadays, even straightforward sites, for example, individual web journals have a great deal of conditions. The neglecting to refresh each bit of programming on the backend and frontend of a site will, indeed, present substantial security hazards in the near future.



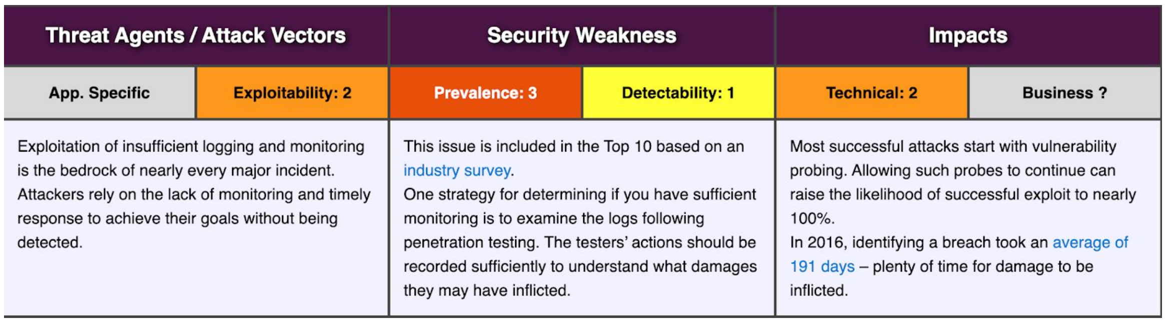
Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 2	Prevalence: 3	Detectability: 2	Technical: 2	Business ?
While it is easy to find already-written exploits for many known vulnerabilities, other vulnerabilities require concentrated effort to develop a custom exploit.		Prevalence of this issue is very widespread. Component-heavy development patterns can lead to development teams not even understanding which components they use in their application or API, much less keeping them up to date. Some scanners such as retire.js help in detection, but determining exploitability requires additional effort.		While some known vulnerabilities lead to only minor impacts, some of the largest breaches to date have relied on exploiting known vulnerabilities in components. Depending on the assets you are protecting, perhaps this risk should be at the top of the list.	

Fig 2.10: UCKV: Threat Agents, Security Weakness, Impacts

10. Inadequate Logging and Monitoring

The significance of making sure about a site can't be downplayed. While 100% security is certainly not a reasonable objective, there are approaches to keep the site observed all the time so a security designer can make prompt moves when something occurs. A more called Automation helps for this situation.

Nowadays a few WAF likewise helps organizations in making sure about their areas yet basically depending on the WAF isn't a fitting relief from the security vulnerabilities the same number of the WAFs can be circumvented without any problem.



Threat Agents / Attack Vectors		Security Weakness		Impacts	
App. Specific	Exploitability: 2	Prevalence: 3	Detectability: 1	Technical: 2	Business ?
Exploitation of insufficient logging and monitoring is the bedrock of nearly every major incident. Attackers rely on the lack of monitoring and timely response to achieve their goals without being detected.		This issue is included in the Top 10 based on an industry survey. One strategy for determining if you have sufficient monitoring is to examine the logs following penetration testing. The testers' actions should be recorded sufficiently to understand what damages they may have inflicted.		Most successful attacks start with vulnerability probing. Allowing such probes to continue can raise the likelihood of successful exploit to nearly 100%. In 2016, identifying a breach took an average of 191 days – plenty of time for damage to be inflicted.	

Fig 2.11: Inadequate Logging and Monitoring: Threat Agents, Security Weakness, Impacts

2.2 OWASP Top 10 Mobile

Open Web Application Security Project, or OWASP as it is usually known, is a network based establishment which targets spreading mindfulness about application and programming security. OWASP runs a few network run open-source ventures to improve the information on security lovers everywhere throughout the globe. The establishment arranges few driving instruction and preparing programs in the field of cybersecurity too. The a large number of individuals and various nearby sections of OWASP guarantee that security specialists and designers stay mindful of the continuous security dangers and plan for their moderation ahead of time.

Underneath, you'll locate the main 10 portable security chances as characterized by the OWASP Top 10 Project for Mobile. Understanding these dangers can assist you with setting up your application and ensure yourself, your information and your clients. Beginning one week from now, we will examine every danger in detail.

Top 10 Mobile Security Vulnerabilities as per OWASP TOP 10 Mobile

1. Improper Platform Usage
2. Insecure Data Storage
3. Insecure Communication
4. Insecure Authentication
5. Insufficient Cryptography
6. Insecure Authorization
7. Client Code Quality
8. Code Tampering
9. Reverse Engineering
10. Extraneous Functionality

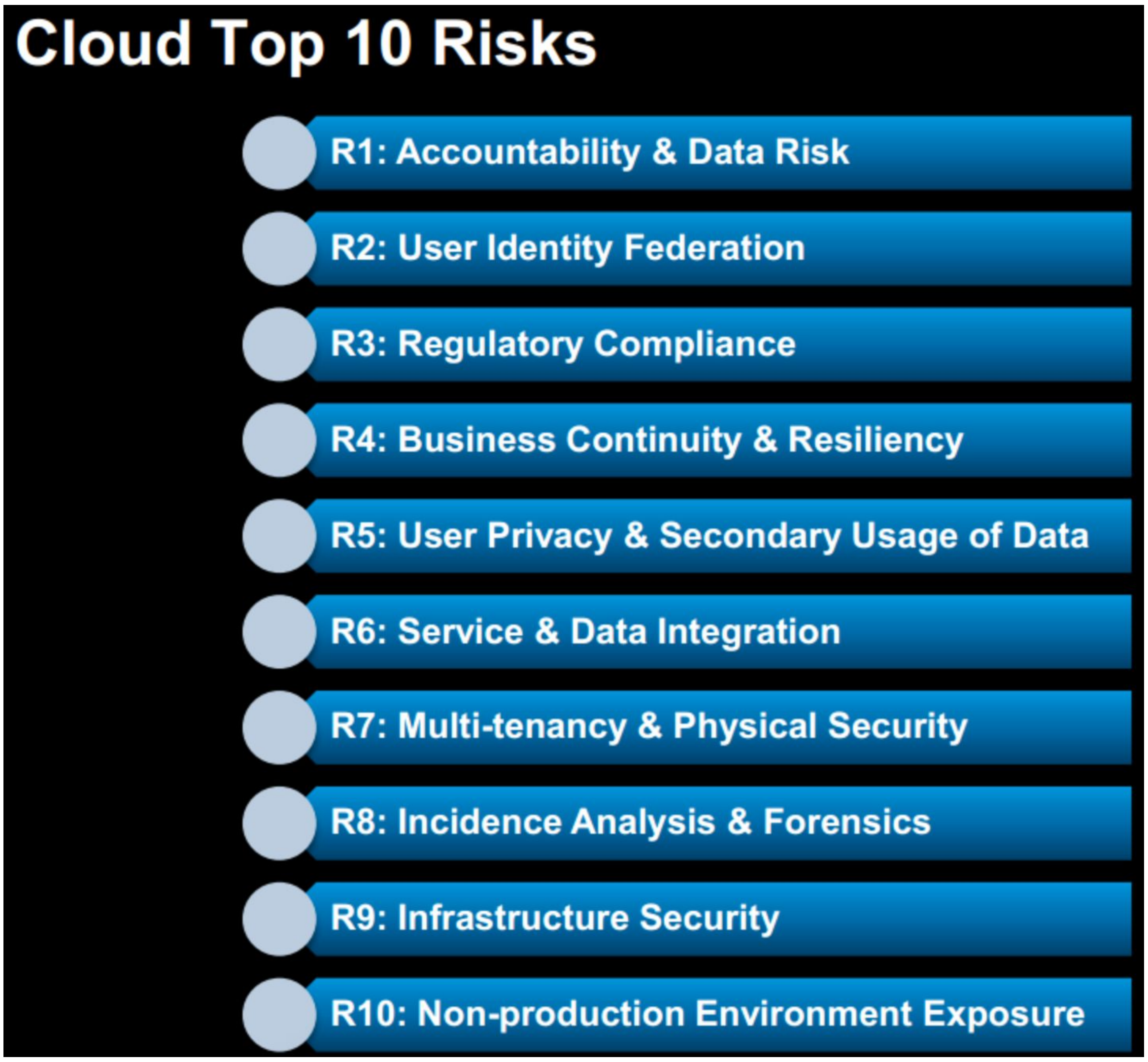
1. Improper Platform Usage

This classification covers abuse of a stage highlight or inability to utilize stage security controls. It may incorporate Android expectations, stage consents, abuse of TouchID, the Keychain, or some other security control that is a piece of the versatile working framework. The characterizing normal for dangers in this class is that the stage (iOS, Android, Windows Phone, and so on.) gives an element or a capacity that is recorded and surely known. The application neglects to utilize that capacity or utilizations it erroneously. This varies from other versatile top ten dangers in light of the fact that the plan and usage isn't carefully the application engineer's issue.

There are a few different ways that portable applications can encounter this hazard.

1. **Infringement of distributed rules**. All stages have advancement rules for security (c.f., ((Android)), ((iOS)), ((Windows Phone))). In the event that an application repudiates the accepted procedures suggested by the maker, it will be presented to this hazard. For instance, there are rules on the most proficient method to utilize the iOS Keychain or how to make sure about traded benefits on Android. Applications that don't follow these rules will encounter this hazard.
2. **Infringement of show or regular practice**. Not every best practice are systematized in producer direction. In certain cases, there are true accepted procedures that are normal in portable applications.
3. **Inadvertent Misuse**. Some applications expect to make the best decision, yet really get some piece of the execution wrong. This could be a basic bug, such as setting an inappropriate banner on an API call, or it could be a misconception of how the securities work.

Inorder to satisfy this category, mitigate these risks:



Cloud Top 10 Risks

R1: Accountability & Data Risk

R2: User Identity Federation

R3: Regulatory Compliance

R4: Business Continuity & Resiliency

R5: User Privacy & Secondary Usage of Data

R6: Service & Data Integration

R7: Multi-tenancy & Physical Security

R8: Incidence Analysis & Forensics

R9: Infrastructure Security

R10: Non-production Environment Exposure

Fig 2.12: Risks in Improper Platform Usage

2. Insecure Data Storage

Insecure Data Storage vulnerabilities happen when advancement groups accept that clients or malware won't approach a cell phone's filesystem and resulting delicate data in information stores on the gadget. Filesystems are effectively available. Associations ought to anticipate a malevolent client or malware to investigate delicate information stores. Utilization of poor encryption libraries is to be dodged. Establishing or jailbreaking a cell phone goes around any encryption securities. At the point when information isn't ensured appropriately, specific apparatuses are on the whole that is expected to see application information.

3. Insecure Communication

Versatile applications much of the time don't secure system traffic. They may utilize SSL/TLS during confirmation yet not somewhere else. This irregularity prompts the danger of uncovering information and meeting IDs to block attempt. The utilization of transport security doesn't mean the application has actualized it effectively. To recognize fundamental imperfections, watch the telephone's system traffic. Progressively unobtrusive imperfections require assessing the plan of the application and the application's design.

4. Insecure Authentication

Poor or missing authentication vulnerabilities permit an attacker or malicious user to namelessly execute usefulness inside the versatile application or backend server utilized by the portable application. More fragile validation for portable applications is genuinely common because of a cell phone's info structure factor. The structure factor profoundly supports short passwords that are frequently simply dependent on 4-digit PINs. Confirmation necessities for portable applications can be very unique in relation to customary web verification plots because of accessibility prerequisites.
In customar web applications, clients are relied upon to be on the web and verify progressively with a backend server. All through their meeting, there is a sensible desire that they will have persistent access to the Internet.

In versatile applications, clients are not expected to be online consistently during their meeting. Portable web associations are substantially less solid or unsurprising than conventional web associations. Subsequently, versatile applications may have uptime necessities that require disconnected validation. This disconnected necessity can have significant implications on things that engineers must consider while actualizing portable verification.

To identify poor verification plans, analyzers can perform twofold assaults against the portable application while it is in 'disconnected' mode. Through the assault, the analyzer will compel the application to sidestep disconnected verification and afterward execute usefulness that ought to require disconnected validation (for more data on parallel assaults, see M10). Too, analyzers should attempt to execute any backend server usefulness secretly by expelling any meeting tokens from any POST/GET demands for the portable application usefulness.

5. Insufficient Cryptography

Insecure utilization of cryptography is basic in most versatile applications that influence encryption. There are two basic ways that messed up cryptography is showed inside portable applications. To begin with, the versatile application may utilize a procedure behind the encryption/unscrambling that is essentially defective and can be abused by the foe to decode touchy information. Second, the portable application may actualize or use an encryption/decoding calculation that is frail in nature and can be straightforwardly unscrambled by the foe. The accompanying subsections investigate both of these situations in more profundity:

Dependence Upon Built-In Code Encryption Processes

Of course, iOS applications are ensured (in principle) from figuring out by means of code encryption. The iOS security model requires that applications be scrambled and marked by dependable sources so as to execute in non-jailbroken conditions. Upon fire up, the iOS application loader will unscramble the application in memory and continue to execute the code after its mark has been confirmed by iOS. This component, in principle, keeps an aggressor from leading parallel assaults against an iOS versatile application.

Utilizing unreservedly accessible apparatuses like ClutchMod or GBD, an enemy will download the encoded application onto their jailbroken gadget and take a preview of the unscrambled application once the iOS loader loads it into memory and decodes it (not long before the loader commences execution). When the enemy takes the preview and stores it on circle, the foe can utilize devices like IDA Pro or Hopper to effectively perform static/dynamic investigation of the application and lead further paired assaults.

Bypassing worked in code encryption calculations is paltry, best case scenario. Continuously accept that an enemy will have the option to sidestep any implicit code encryption offered by the basic versatile OS. For more data about extra advances you can take to give extra layers of figuring out avoidance, see M9.

Poor Key Management Processes

The best calculations don't make a difference on the off chance that you misuse your keys. Many wrongly use the right encryption calculation, however executing their own convention for utilizing it. A few instances of issues here include:

1. Remembering the keys for a similar assailant decipherable index as the encoded content;
2. Making the keys in any case accessible to the aggressor;
3. Stay away from the utilization of hardcoded keys inside your double; and
4. Keys might be caught through parallel assaults. See M10 for more data on forestalling parallel assaults.

Creation and Use of Custom Encryption Protocols

There is no simpler method to misuse encryption–portable or something else than to attempt to make and utilize your own encryption calculations or conventions.

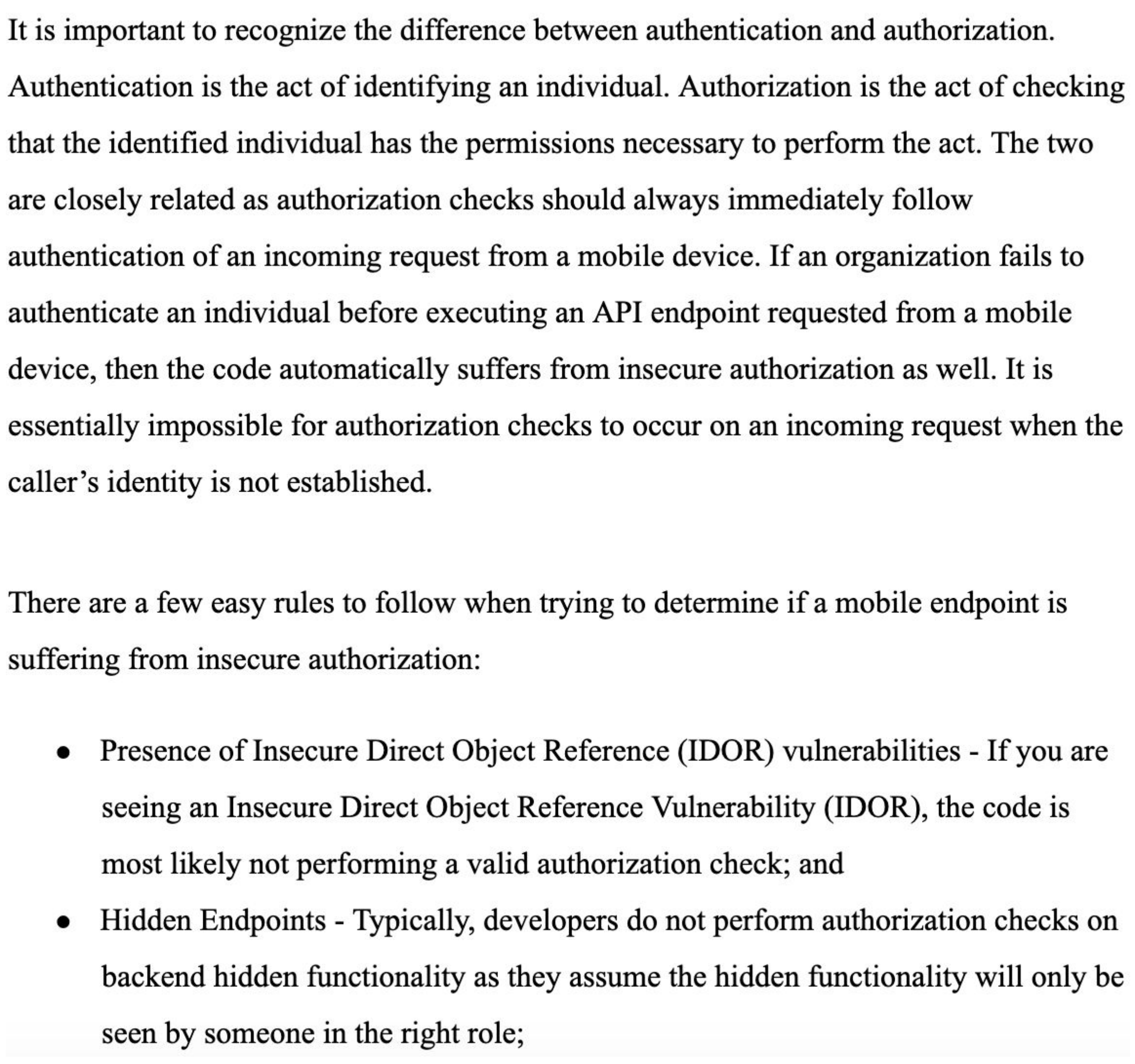
Continuously utilize current calculations that are acknowledged as solid by the security network, and at whatever point conceivable influence the cutting edge encryption APIs inside your portable stage. Paired assaults may bring about foe recognizing the normal libraries you have utilized alongside any hardcoded enters in the parallel. In instances of high security necessities around encryption, you ought to emphatically consider the utilization of whitebox cryptography. See M10 for more data on forestalling parallel assaults that could prompt the misuse of regular libraries.

Utilization of Insecure or potentially Deprecated Algorithms

Numerous cryptographic calculations and conventions ought not be utilized in light of the fact that they have appeared to have huge shortcomings or are in any case lacking for present day security necessities. These include:

1. RC2
2. MD4
3. MD5
4. SHA1

6. Insecure Authorization



It is important to recognize the difference between authentication and authorization. Authentication is the act of identifying an individual. Authorization is the act of checking that the identified individual has the permissions necessary to perform the act. The two are closely related as authorization checks should always immediately follow authentication of an incoming request from a mobile device. If an organization fails to authenticate an individual before executing an API endpoint requested from a mobile device, then the code automatically suffers from insecure authorization as well. It is essentially impossible for authorization checks to occur on an incoming request when the caller's identity is not established.

There are a few easy rules to follow when trying to determine if a mobile endpoint is suffering from insecure authorization:

- Presence of Insecure Direct Object Reference (IDOR) vulnerabilities - If you are seeing an Insecure Direct Object Reference Vulnerability (IDOR), the code is most likely not performing a valid authorization check; and
- Hidden Endpoints - Typically, developers do not perform authorization checks on backend hidden functionality as they assume the hidden functionality will only be seen by someone in the right role;

- User Role or Permission Transmissions - If the mobile app is transmitting the user’s roles or permissions to a backend system as part of a request, it is suffering from insecure authorization.

7. Client Code Quality

Code quality issues are genuinely predominant inside most portable code. Fortunately most code quality issues are genuinely kindhearted and bring about awful programming practice. It is commonly hard to distinguish these kinds of issues through manual code audit. Rather, assailants will utilize outsider devices that perform static investigation or perform fluffing. These kinds of instruments will regularly distinguish memory releases, cushion floods, and different less extreme issues that bring about awful programming practice. Programmers with outrageous low-level information and ability can viably misuse these kinds of issues. The run of the mill essential objective is to execute outside code inside the portable code's location space.

8. Code Tempering

Altered types of utilizations are shockingly more typical than you might suspect. There is a whole security industry worked around distinguishing and evacuating unapproved adaptations of versatile applications inside application stores. Contingent on the methodology taken to taking care of the issue of distinguishing code change, associations can have constrained to exceptionally fruitful methods of identifying unapproved adaptations of code in nature. This classification covers parallel fixing, nearby asset change, technique snaring, strategy swizzling, and dynamic memory alteration. When the application is conveyed to the cell phone, the code and information assets are inhabitant there. An assailant can either straightforwardly adjust the code, change the substance of memory progressively, change or supplant the framework APIs that the application utilizes, or alter the application's information and assets. This can give the aggressor an immediate strategy for undermining the proposed utilization of the product for individual or money related increase. Actually, all versatile code is defenseless against code altering. Portable code runs inside a situation that isn't heavily influenced by the association creating the code. Simultaneously, there are a lot of various methods of changing nature wherein that code runs. These progressions permit a foe to tinker with the code and alter it voluntarily.

Although portable code is intrinsically powerless, it is imperative to inquire as to whether it merits recognizing and attempting to forestall unapproved code change. Applications composed for certain business verticals (gaming for instance) are substantially more defenseless against the effects of code adjustment than others (neighborliness for instance). In that capacity, it is basic to consider the business sway before choosing whether or not to address this hazard.

Impact: The business sway from code alteration regularly brings about the accompanying:

1. Income misfortune because of robbery; or
2. Reputational harm.

9. Reverse Engineering

For most of the mobile applications, all mobile code is vulnerable to reverse engineering. Some applications are more defenseless than others. Code written in dialects/systems that take into account dynamic thoughtfulness at runtime (Java, .NET, Objective C, Swift) are especially in danger for figuring out. Recognizing defenselessness to figure out is genuinely straightforward. To begin with, unscramble the application store variant of the application (if parallel encryption is applied). At that point, utilize the instruments illustrated in the "Assault Vectors" segment of this archive against the twofold. Code will be helpless on the off chance that it is genuinely straightforward the application's controlflow way, string table, and any pseudocode/source-code created by these instruments. An attacker may abuse figuring out to accomplish any of the accompanying:

1. Uncover data about back end servers;
2. Uncover cryptographic constants and figures;
3. Take licensed innovation;
4. Perform assaults against back end frameworks; or

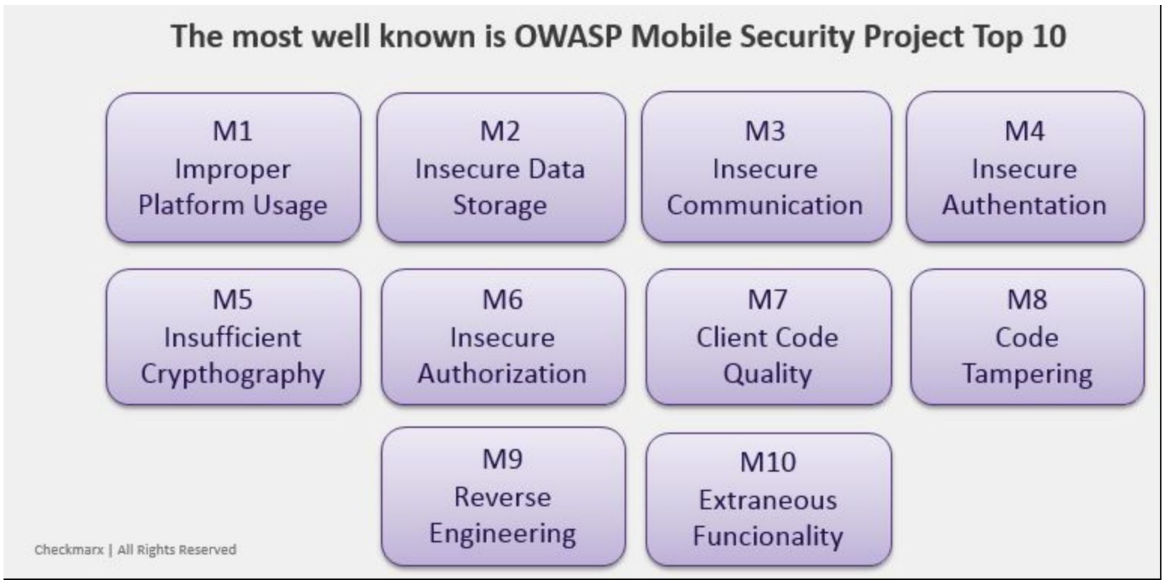
5. Increase knowledge expected to perform ensuing code alteration.

10. Extraneous Functionality

Commonly, an attacker tries to comprehend incidental usefulness inside a versatile application so as to find concealed usefulness in backend frameworks. The assailant will ordinarily abuse unessential usefulness legitimately from their own frameworks with no association by end-clients.

An assailant will download and look at the portable application inside their own neighborhood condition. They will look at log records, design documents, and maybe the twofold itself to find any concealed switches or test code that was abandoned by the engineers. They will misuse these switches and concealed usefulness in the backend framework to play out an assault.

There is a high probability that any given versatile application contains superfluous usefulness that isn't legitimately presented to the client by means of the interface. The vast majority of this extra code is amiable in nature and won't give an aggressor any extra knowledge into backend capacities. Be that as it may, some incidental usefulness can be extremely helpful to an assailant. Usefulness that opens data identified with back-end test, demo, arranging, or UAT conditions ought not be remembered for a creation fabricate. Furthermore, managerial API endpoints, or informal endpoints ought not be remembered for definite creation assemblies. Distinguishing superfluous usefulness can be precarious. Mechanized static and dynamic examination instruments can get low hanging natural products (log explanations). Be that as it may, a few secondary passages are hard to recognize in a robotized manner. Thus, it is in every case best to forestall these things utilizing a manual code survey.



The most well known is OWASP Mobile Security Project Top 10

M1
Improper
Platform Usage

M2
Insecure Data
Storage

M3
Insecure
Communication

M4
Insecure
Authentication

M5
Insufficient
Crypthography

M6
Insecure
Authorization

M7
Client Code
Quality

M8
Code
Tampering

M9
Reverse
Engineering

M10
Extraneous
Funcionality

Checkmarx | All Rights Reserved

Fig 2.13: OWASP top 10 mobile

Chapter 3

SYSTEM DEVELOPMENT

3.1. BURP SUITE PROFESSIONAL

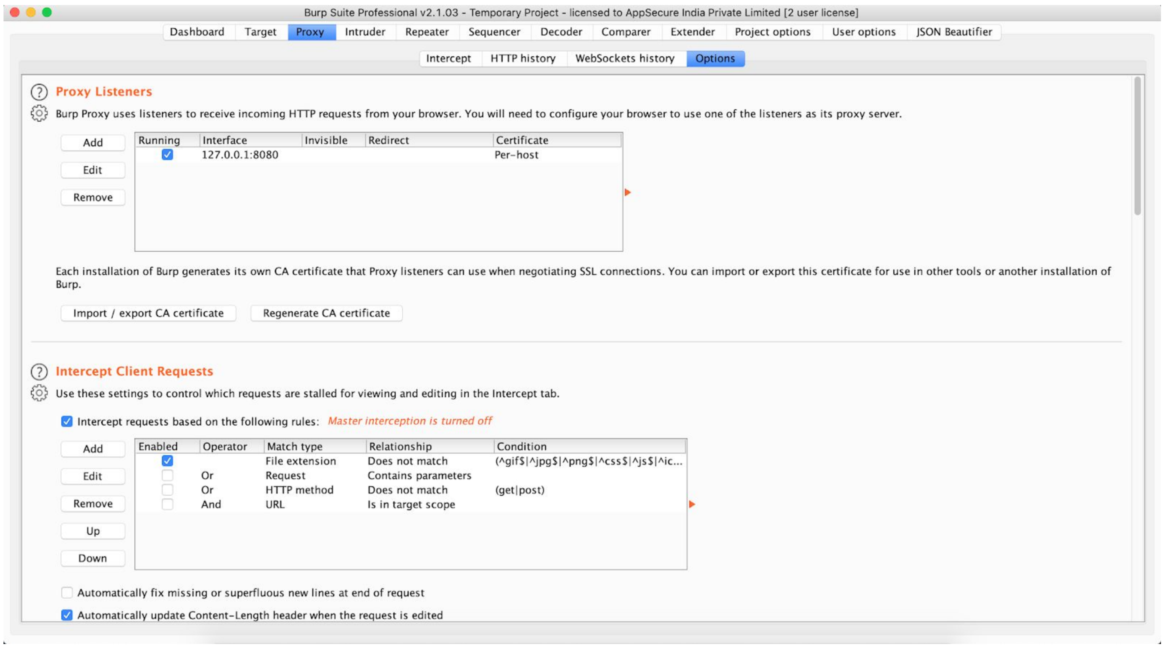
Burp Suite Professional is a propelled set of softwares for testing web applications - all inside a solitary item. From an essential catching intermediary to a bleeding edge powerlessness scanner, with Burp Suite Pro, the correct apparatus is never in excess of a tick away. The amazing robotization of burp suite pro gives security engineers a greater chance to do what they specialize in, while Burp Suite handles low-hanging organic products. Propelled manual instruments will at that point assist you with distinguishing your objective's increasingly inconspicuous vulnerable sides. Burp Suite Pro is worked by an examination drove group. This implies before we even distribute a paper, its discoveries have been remembered for our most recent update. Our pentesting devices will make your activity quicker while keeping you educated regarding the most recent assault vectors.

Burp Suite Pro can be utilized to test for the entire OWASP Top 10. Crafted by PortSwigger's reality driving exploration group guides Burp Suite's turn of events, and new capacities are included with each new update. At Burp Suite's heart lies the web api/url scanner. This is the equivalent ground-breaking scanner trusted by numerous individuals of the world's biggest associations. It's fit for both uninvolved and dynamic investigation.

Burp suite pro is a complete tool for security assessment and penetration testing. It helps in capturing both website and mobile application’s traffic and allows an attacker to play with the API. The proxy tab of the tool can be configured with mobile device and web browsers using the system proxy and CA certificate. The CA certificate is used to ensure that the traffic is captured by the trusted source.

3.2 How to Configure Burp Suite with a Web Browser?

1. Open the burp suite proxy and click on the proxy tab followed by the option button.



Burp Suite Professional v2.1.03 - Temporary Project - licensed to AppSecure India Private Limited [2 user license]

Dashboard | Target | Proxy | Intruder | Repeater | Sequencer | Decoder | Comparer | Extender | Project options | User options | JSON Beautifier

Intercept | HTTP history | WebSockets history | Options

Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.

Running	Interface	Invisible	Redirect	Certificate
✓	127.0.0.1:8080			Per-host

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating SSL connections. You can import or export this certificate for use in other tools or another installation of Burp.

Import / export CA certificate | Regenerate CA certificate

Intercept Client Requests

Use these settings to control which requests are stalled for viewing and editing in the Intercept tab.

✓ Intercept requests based on the following rules: *Master interception is turned off*

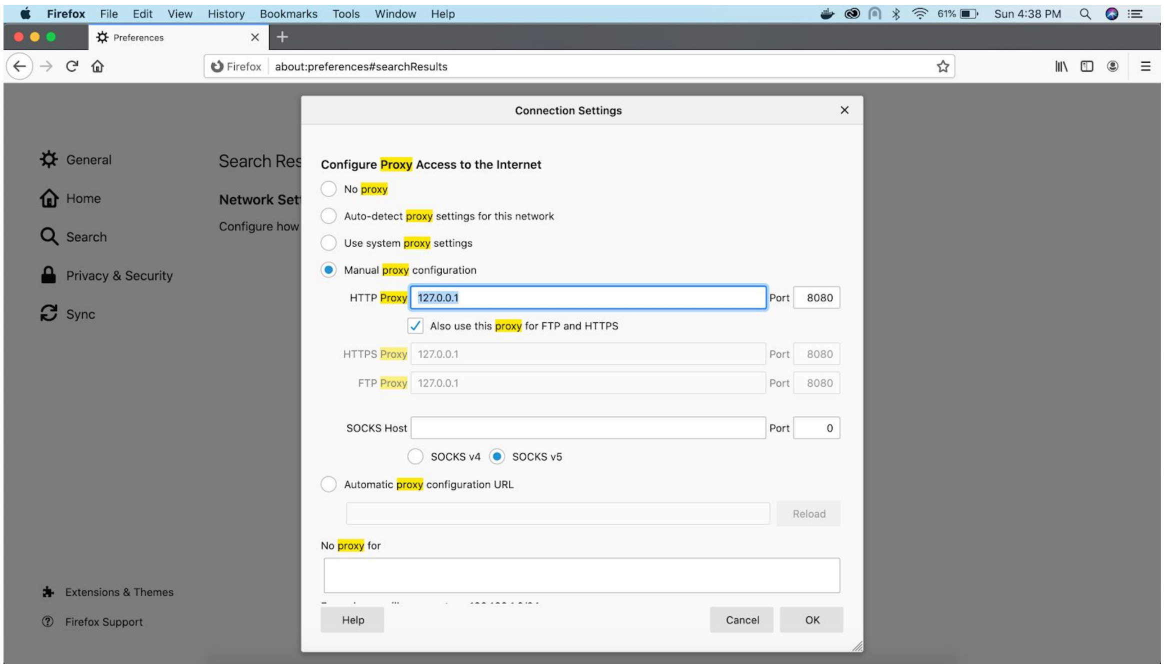
Enabled	Operator	Match type	Relationship	Condition
✓		File extension	Does not match	(^gif$\|^jpg$|^png$\|^css$|^js$|^ic...
☐	Or	Request	Contains parameters	
☐	Or	HTTP method	Does not match	(get|post)
☐	And	URL	Is in target scope	

☐ Automatically fix missing or superfluous new lines at end of request

✓ Automatically update Content-Length header when the request is edited

Fig 3.1: Setting Burp Suite Proxy Listener

2. Copy the proxy listener address and port.
3. In the Firefox web browser, open the proxy menu and add the proxy listener address and port to it.



Connection Settings

Configure Proxy Access to the Internet

No proxy
Auto-detect proxy settings for this network
Use system proxy settings
[✓ Manual proxy configuration]

HTTP Proxy 127.0.0.1 Port 8080

✓ Also use this proxy for FTP and HTTPS

HTTPS Proxy 127.0.0.1 Port 8080

FTP Proxy 127.0.0.1 Port 8080

SOCKS Host Port 0

SOCKS v4 [✓ SOCKS v5]

Automatic proxy configuration URL

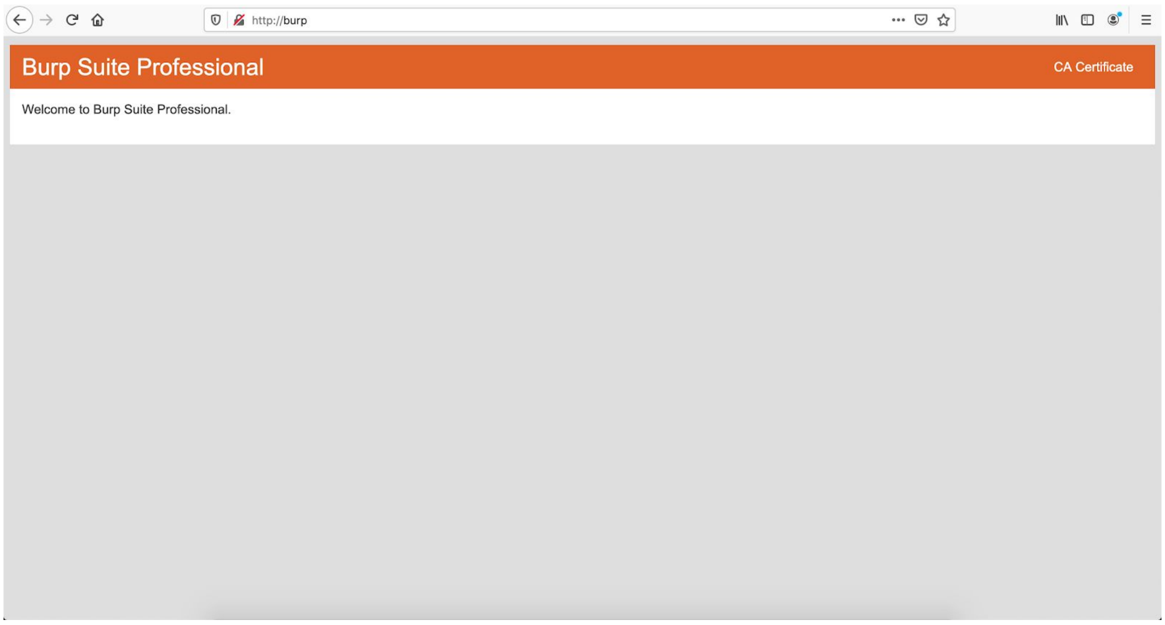
Reload

No proxy for

Help Cancel OK

Fig 3.2: Setting Proxy in FireFox Web Browser

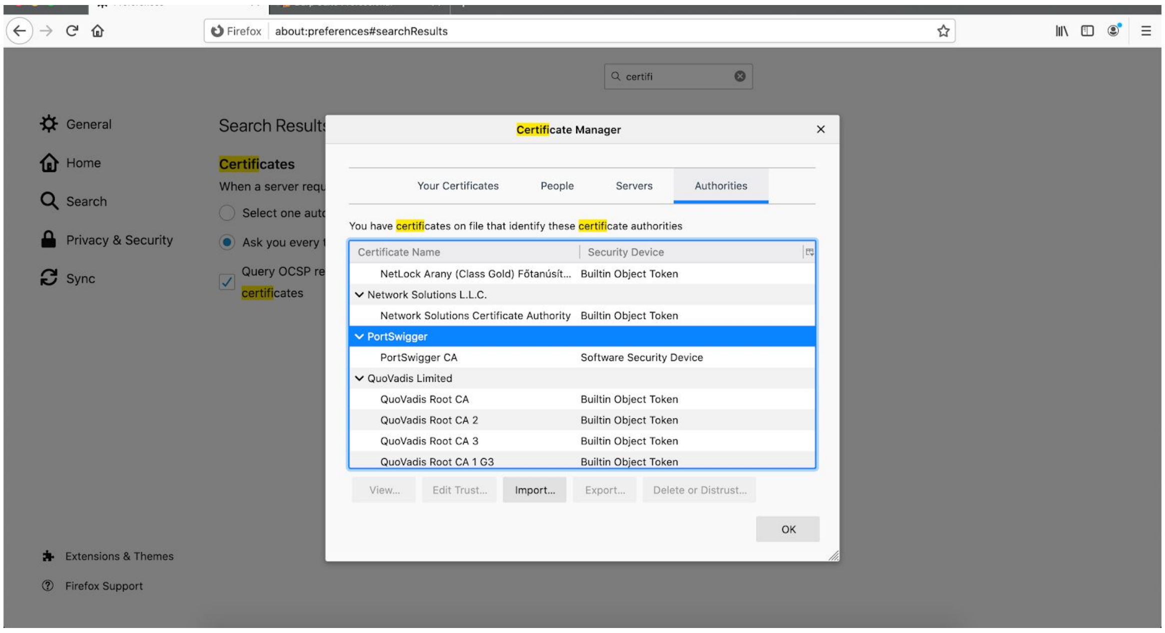
4. Navigate to the http://burp in the firfox and download the CA certificate.



Welcome to Burp Suite Professional.

Fig 3.3: Download CA certificate in FireFox Browser

5. Install the CA-certificate in the firefox browser.

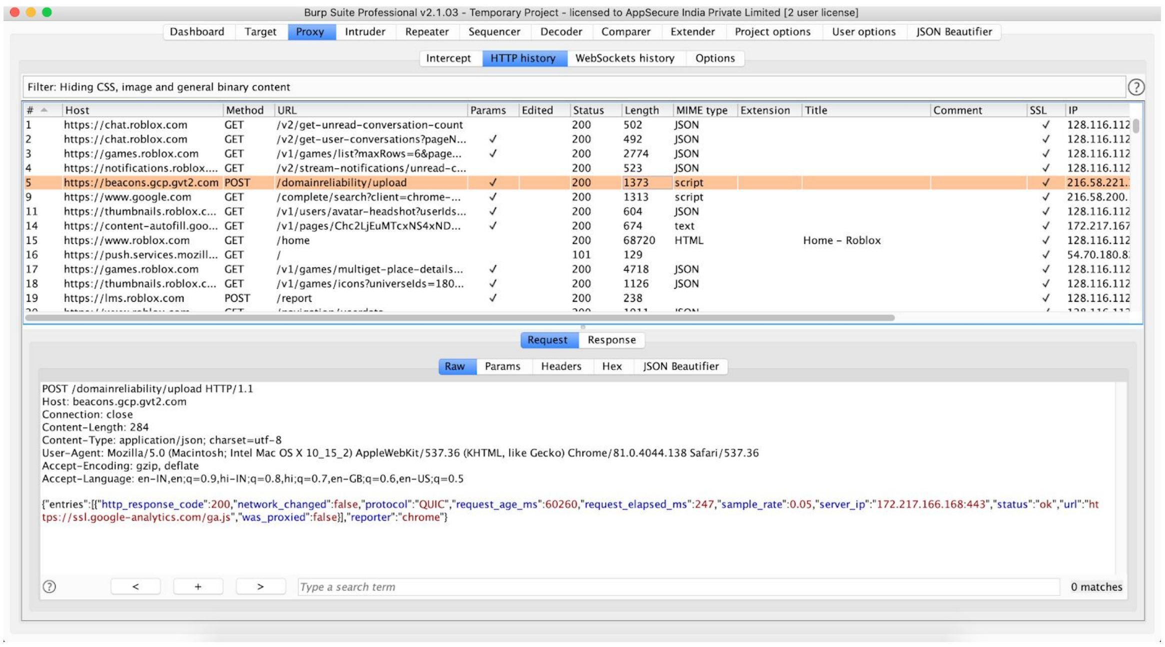


You have certificates on file that identify these certificate authorities

Certificate Name	Security Device
NetLock Arany (Class Gold) Főtanúsít...	Builtin Object Token
Network Solutions L.L.C.	
Network Solutions Certificate Authority	Builtin Object Token
PortSwigger	
PortSwigger CA	Software Security Device
QuoVadis Limited	
QuoVadis Root CA	Builtin Object Token
QuoVadis Root CA 2	Builtin Object Token
QuoVadis Root CA 3	Builtin Object Token
QuoVadis Root CA 1 G3	Builtin Object Token

Fig 3.4: Installing CA Certificate in FireFox Web Browser

6. Open any website in the web browser and intercept the network traffic in burp suite proxy.



Burp Suite Professional v2.1.03 - Temporary Project - licensed to AppSecure India Private Limited [2 user license]

Filter: Hiding CSS, image and general binary content

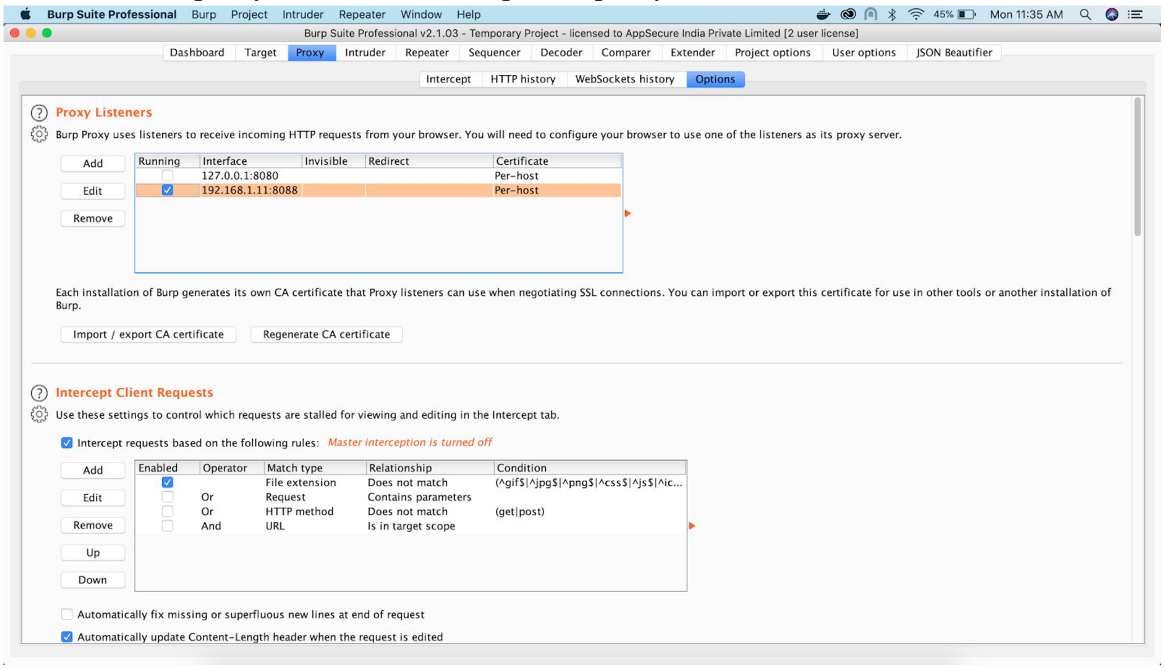
#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	SSL	IP
1	https://chat.roblox.com	GET	/v2/get-unread-conversation-count			200	502	JSON				✓	128.116.112
2	https://chat.roblox.com	GET	/v2/get-user-conversations?pageN...	✓		200	492	JSON				✓	128.116.112
3	https://games.roblox.com	GET	/v1/games/list?maxRows=6&page...	✓		200	2774	JSON				✓	128.116.112
4	https://notifications.roblox....	GET	/v2/stream-notifications/unread-c...			200	523	JSON				✓	128.116.112
5	https://beacons.gcp.gvt2.com	POST	/domainreliability/upload	✓		200	1373	script				✓	216.58.221.
9	https://www.google.com	GET	/complete/search?client=chrome-...	✓		200	1313	script				✓	216.58.200.
11	https://thumbnails.roblox.c...	GET	/v1/users/avatar-headshot?userIds...	✓		200	604	JSON				✓	128.116.112
14	https://content-autofill.goo...	GET	/v1/pages/Chc2LjEuMTcxNS4xND...	✓		200	674	text				✓	172.217.167
15	https://www.roblox.com	GET	/home			200	68720	HTML		Home – Roblox		✓	128.116.112
16	https://push.services.mozill...	GET	/			101	129					✓	54.70.180.8
17	https://games.roblox.com	GET	/v1/games/multiget-place-details...	✓		200	4718	JSON				✓	128.116.112
18	https://thumbnails.roblox.c...	GET	/v1/games/icons?universeIds=180...	✓		200	1126	JSON				✓	128.116.112
19	https://lms.roblox.com	POST	/report	✓		200	238					✓	128.116.112

```
POST /domainreliability/upload HTTP/1.1
Host: beacons.gcp.gvt2.com
Connection: close
Content-Length: 284
Content-Type: application/json; charset=utf-8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36
Accept-Encoding: gzip, deflate
Accept-Language: en-IN,en;q=0.9,hi-IN;q=0.8,hi;q=0.7,en-GB;q=0.6,en-US;q=0.5

{"entries":[{"http_response_code":200,"network_changed":false,"protocol":"QUIC","request_age_ms":60260,"request_elapsed_ms":247,"sample_rate":0.05,"server_ip":"172.217.166.168:443","status":"ok","url":"ht
tps://ssl.google-analytics.com/ga.js","was_proxied":false}],"reporter":"chrome"}
```

Fig 3.5: Capturing Web Site Traffic in Burp Suite Proxy

3.3 How To Configure Burp Suite with Android / iOS Device?

1. Connect both mobile device and computer on the same internet network.
2. Add a new proxy listener in the burp suite proxy and run it.



Burp Suite Professional v2.1.03 - Temporary Project - licensed to AppSecure India Private Limited [2 user license]

Proxy Listeners

Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.

Running	Interface	Invisible	Redirect	Certificate
☐	127.0.0.1:8080			Per-host
✓	192.168.1.11:8088			Per-host

Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating SSL connections. You can import or export this certificate for use in other tools or another installation of Burp.

Intercept Client Requests

Use these settings to control which requests are stalled for viewing and editing in the Intercept tab.

✓ Intercept requests based on the following rules: *Master interception is turned off*

Enabled	Operator	Match type	Relationship	Condition
✓		File extension	Does not match	(^gif$\|^jpg$|^png$\|^css$|^js$|^ic...
☐	Or	Request	Contains parameters	
☐	Or	HTTP method	Does not match	(get|post)
☐	And	URL	Is in target scope	

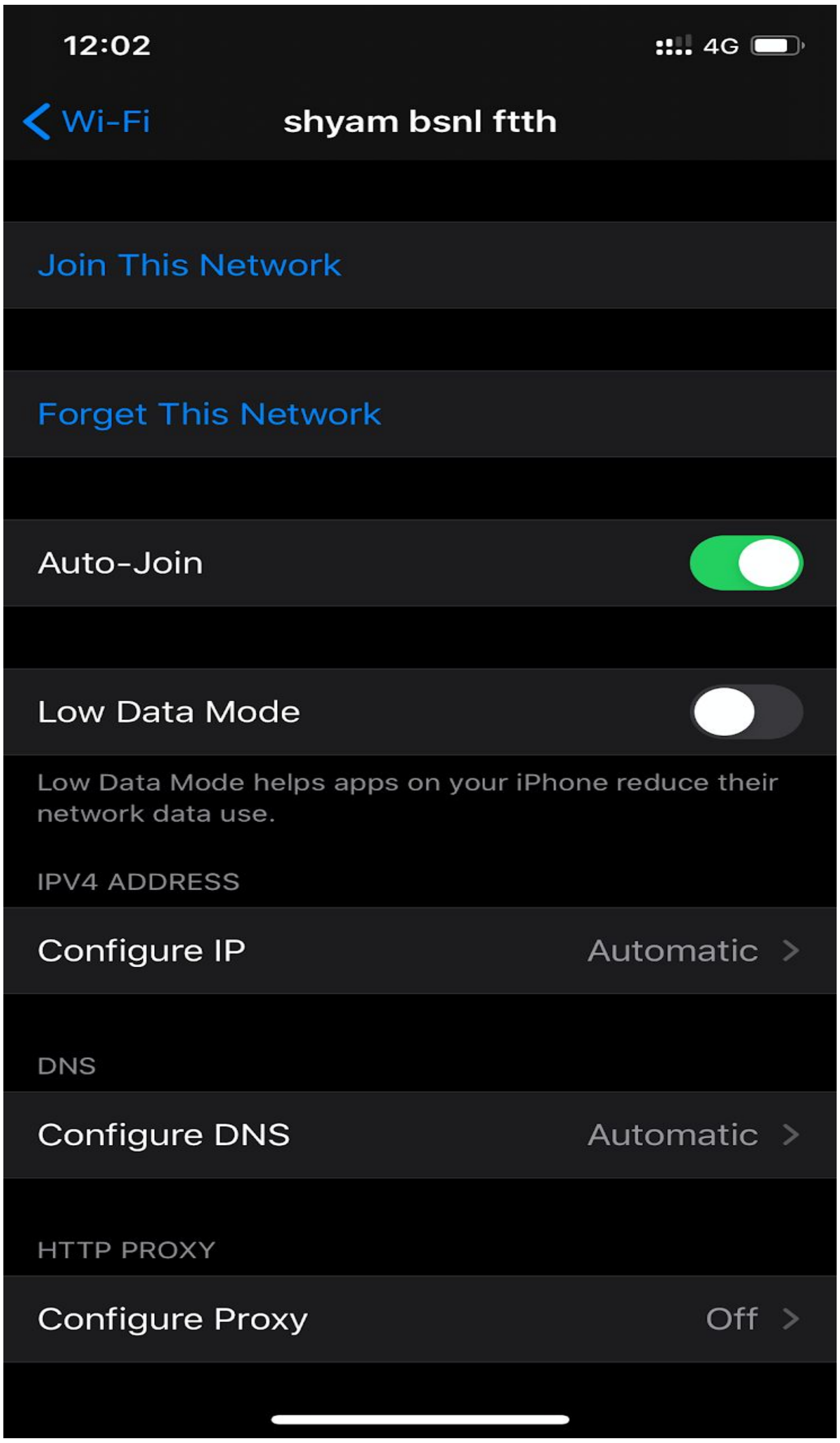
☐ Automatically fix missing or superfluous new lines at end of request

✓ Automatically update Content-Length header when the request is edited

Fig 3.6: Setting proxy listener on burp suite for mobile device

3. Copy proxy listener address and port.

4. In Mobile device, select the same wifi network in which the computer is connected or make an internet sharing network from the computer and connect it with a mobile device.
5. In wifi selection, click on the advanced option and configure proxy mode as manual.


12:02 4G

Wi-Fi shyam bsnl ftth

Join This Network

Forget This Network

Auto-Join

Low Data Mode

Low Data Mode helps apps on your iPhone reduce their network data use.

IPV4 ADDRESS

Configure IP Automatic

DNS

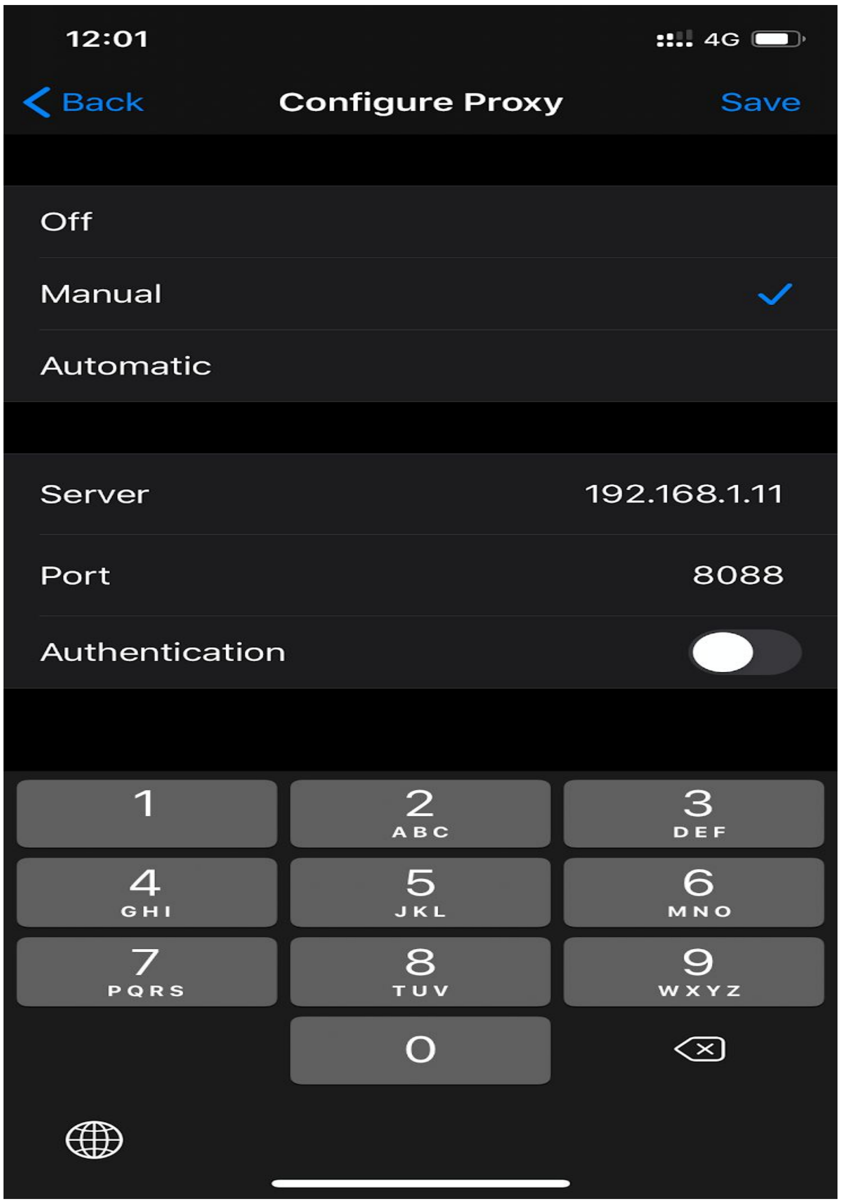
Configure DNS Automatic

HTTP PROXY

Configure Proxy Off

Fig 3.7: Advance Option wifi settings

6. Add burp suite proxy listener address and port in the manual proxy of the device.



Off

Manual ✓

Automatic

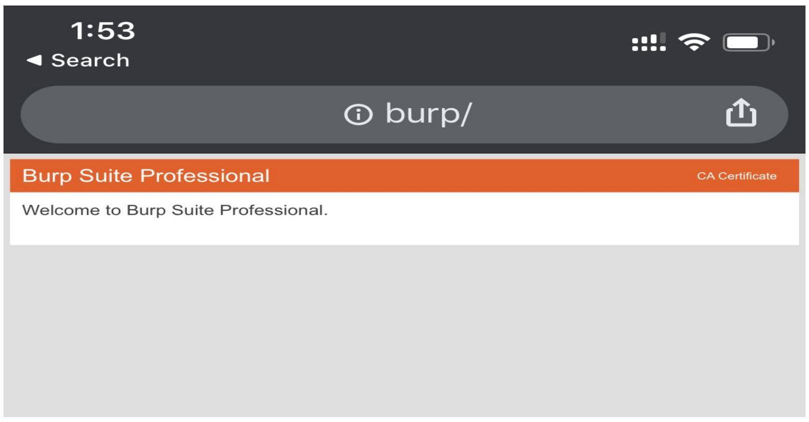
Server 192.168.1.11

Port 8088

Authentication

Fig 3.8: Adding Proxy Listener to Mobile Device

7. Open web browser in mobile and download CA certificate by navigating to http://burp



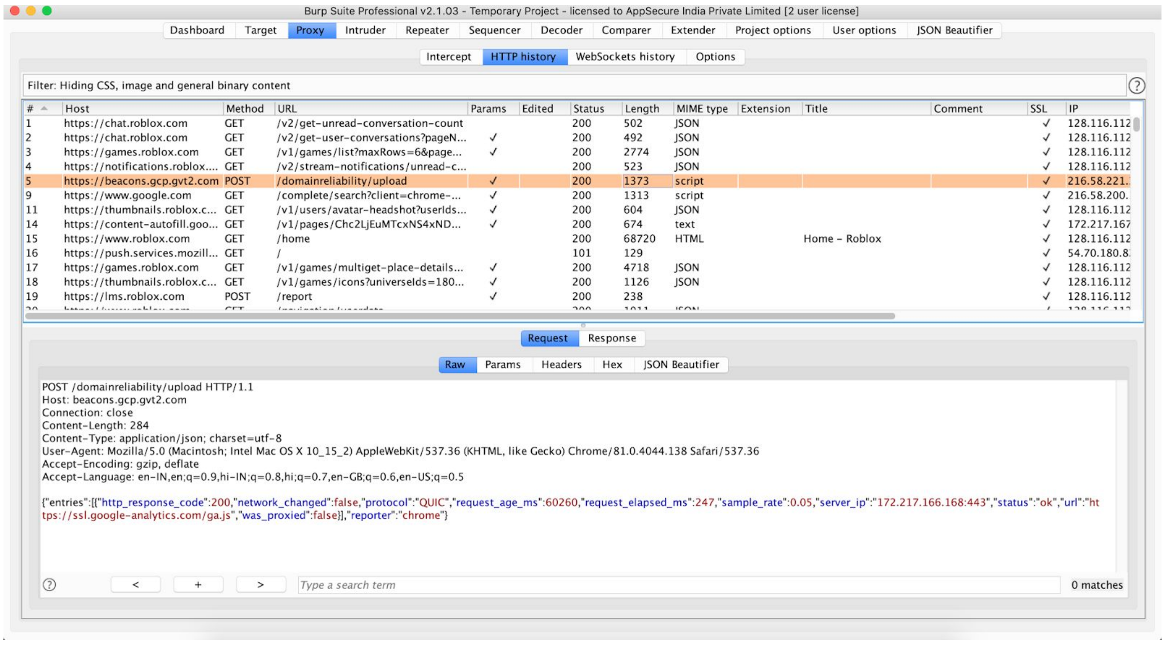
Burp Suite Professional

Welcome to Burp Suite Professional.

Fig 3.9: Download burp CA certificate to mobile device

8. Install the downloaded CA certificate in the mobile settings.

9. Capture the mobile application traffic in the burp suite proxy tab.



Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length	MIME type	Extension	Title	Comment	SSL	IP
1	https://chat.roblox.com	GET	/v2/get-unread-conversation-count			200	502	JSON				✓	128.116.112
2	https://chat.roblox.com	GET	/v2/get-user-conversations?pageN...	✓		200	492	JSON				✓	128.116.112
3	https://games.roblox.com	GET	/v1/games/list?maxRows=6&page...	✓		200	2774	JSON				✓	128.116.112
4	https://notifications.roblox....	GET	/v2/stream-notifications/unread-c...			200	523	JSON				✓	128.116.112
5	https://beacons.gcp.gvt2.com	POST	/domainreliability/upload	✓		200	1373	script				✓	216.58.221.
9	https://www.google.com	GET	/complete/search?client=chrome-...	✓		200	1313	script				✓	216.58.200.
11	https://thumbnails.roblox.c...	GET	/v1/users/avatar-headshot?userIds...	✓		200	604	JSON				✓	128.116.112
14	https://content-autofill.goo...	GET	/v1/pages/Chc2LjEuMTcxNS4xND...	✓		200	674	text				✓	172.217.167
15	https://www.roblox.com	GET	/home			200	68720	HTML		Home - Roblox		✓	128.116.112
16	https://push.services.mozill...	GET	/			101	129					✓	54.70.180.8
17	https://games.roblox.com	GET	/v1/games/multiget-place-details...	✓		200	4718	JSON				✓	128.116.112
18	https://thumbnails.roblox.c...	GET	/v1/games/icons?universeIds=180...	✓		200	1126	JSON				✓	128.116.112
19	https://lms.roblox.com	POST	/report	✓		200	238					✓	128.116.112

```
POST /domainreliability/upload HTTP/1.1
Host: beacons.gcp.gvt2.com
Connection: close
Content-Length: 284
Content-Type: application/json; charset=utf-8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_2) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.138 Safari/537.36
Accept-Encoding: gzip, deflate
Accept-Language: en-IN,en;q=0.9,hi-IN;q=0.8,hi;q=0.7,en-GB;q=0.6,en-US;q=0.5

{"entries":[{"http_response_code":200,"network_changed":false,"protocol":"QUIC","request_age_ms":60260,"request_elapsed_ms":247,"sample_rate":0.05,"server_ip":"172.217.166.168:443","status":"ok","url":"ht
tps://ssl.google-analytics.com/ga.js","was_proxied":false}],"reporter":"chrome"}
```

Fig 3.10: Capturing mobile application traffic

Chapter-4

Performance Analysis

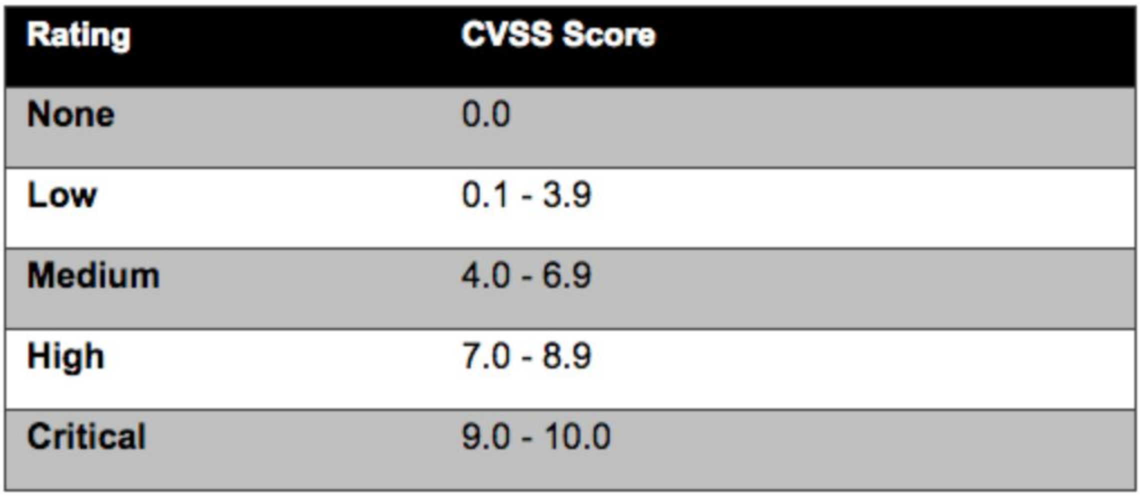
The performance analysis in the web security assessment is based on the cvss score of the vulnerabilities. CVSS score is based on the multiple factors which decided the business impact of the vulnerability on the system.

4.1 CVSS System

The Common Vulnerability Scoring System (CVSS) is an open structure for conveying the qualities and seriousness of programming vulnerabilities. CVSS comprises three measurement gatherings: Base, Temporal, and Environmental. The Base measurements produce a score running from 0 to 10, which would then be able to be changed by scoring the Temporal and Environmental measurements. A CVSS score is likewise spoken to as a vector string, a packed printed portrayal of the qualities used to infer the score. In this manner, CVSS is appropriate as a standard estimation framework for ventures, associations, and governments that need precise and steady weakness seriousness scores. Two regular employments of CVSS are figuring the seriousness of vulnerabilities found on one's frameworks and as a factor in prioritization of defenselessness remediation exercises. The National Vulnerability Database (NVD) gives CVSS scores to practically completely known vulnerabilities.

The Base score is the metric most depended upon by undertakings and manages the natural characteristics of a helplessness. The Temporal scores speak to the characteristics of the powerlessness that change after some time, and the Environmental score speaks to the characteristics of the weakness that are explicit to the influenced client's condition. As indicated by the latest rendition of the CVSS, v3.0, a score of 0.0 gets a "None" appraising; a 0.1-3.9 score gets a "Low" seriousness rating; a score of 4.0-6.9 is a "Medium" rating; score of 7.0-8.9 is a "High" evaluating; and a score of 9.0 - 10.0 is a "Critical" rating.

The CVSS permits associations to organize which vulnerabilities to fix first and measure the effect of the vulnerabilities on their frameworks. Numerous associations utilize the CVSS, and the National Vulnerability Database gives scores to most known vulnerabilities. As indicated by the NVD, a CVSS base score of 0.0-3.9 is considered "Low" seriousness; a base CVSS score of 4.0-6.9 is "Medium" seriousness; and base score of 7.0-10.0 is "High" seriousness.



Rating	CVSS Score
None	0.0
Low	0.1 - 3.9
Medium	4.0 - 6.9
High	7.0 - 8.9
Critical	9.0 - 10.0

Fig 4.1: CVSS Score Distribution

CVSS is made out of three measurement gatherings: Base, Temporal, and Environmental, each comprising a lot of measurements.

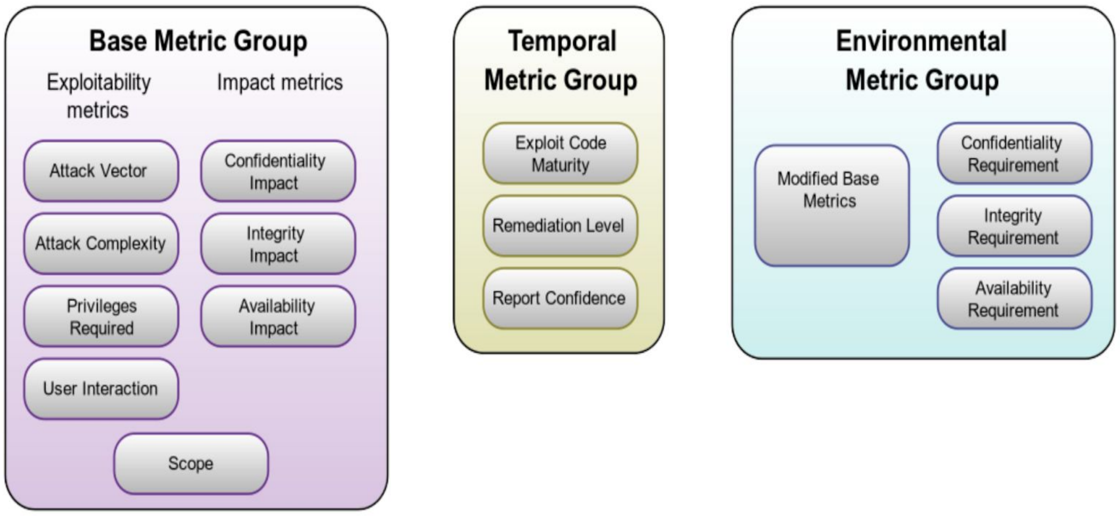
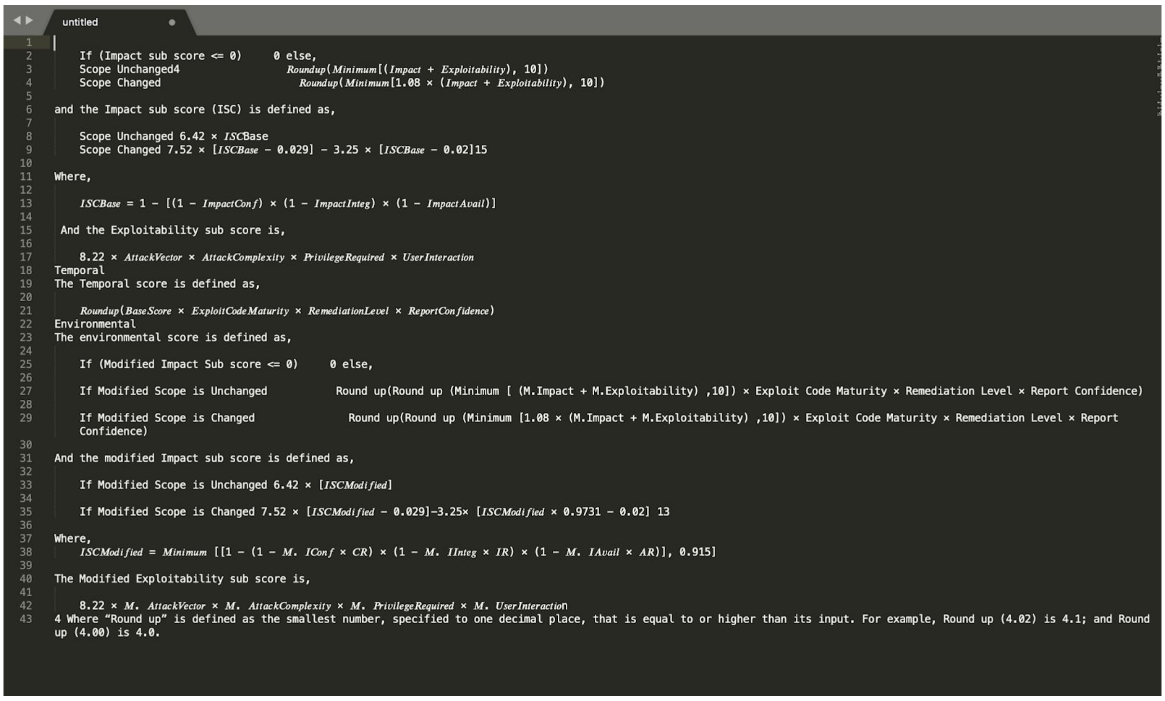


Fig 4.2: CVSS Metric Group

4.2 CVSS Calculator Equation



```

    If (Impact sub score <= 0)     0 else,
    Scope Unchanged4                 Roundup(Minimum[(Impact + Exploitability), 10])
    Scope Changed                      Roundup(Minimum[1.08 × (Impact + Exploitability), 10])

and the Impact sub score (ISC) is defined as,

    Scope Unchanged 6.42 × ISCBase
    Scope Changed 7.52 × [ISCBase − 0.029] − 3.25 × [ISCBase − 0.02]15

Where,

    ISCBase = 1 − [(1 − ImpactConf) × (1 − ImpactInteg) × (1 − ImpactAvail)]

 And the Exploitability sub score is,

    8.22 × AttackVector × AttackComplexity × PrivilegeRequired × UserInteraction
Temporal
The Temporal score is defined as,

    Roundup(BaseScore × ExploitCodeMaturity × RemediationLevel × ReportConfidence)
Environmental
The environmental score is defined as,

    If (Modified Impact Sub score <= 0)     0 else,

    If Modified Scope is Unchanged           Round up(Round up (Minimum [ (M.Impact + M.Exploitability) ,10]) × Exploit Code Maturity × Remediation Level × Report Confidence)

    If Modified Scope is Changed               Round up(Round up (Minimum [1.08 × (M.Impact + M.Exploitability) ,10]) × Exploit Code Maturity × Remediation Level × Report
    Confidence)

And the modified Impact sub score is defined as,

    If Modified Scope is Unchanged 6.42 × [ISCModified]

    If Modified Scope is Changed 7.52 × [ISCModified − 0.029]−3.25× [ISCModified × 0.9731 − 0.02] 13

Where,
    ISCModified = Minimum [[1 − (1 − M. IConf × CR) × (1 − M. IInteg × IR) × (1 − M. IAvail × AR)], 0.915]

The Modified Exploitability sub score is,

    8.22 × M. AttackVector × M. AttackComplexity × M. PrivilegeRequired × M. UserInteraction
4 Where “Round up” is defined as the smallest number, specified to one decimal place, that is equal to or higher than its input. For example, Round up (4.02) is 4.1; and Round
up (4.00) is 4.0.
```

Chapter-5

Conclusion

5.1 Conclusion

A security audit or security assessment is a procedure utilized via prepared security experts or researchers to evaluate and break down your property to find feeble focuses in your security, and afterward offer a strategy to help improve your wellbeing and security. Security reviews are perfect for business structures and edifices, office towers, townhouses, schools, and other business or private spaces needing expanded security. Not exclusively will this assistance improve the security of your property for your occupants or workers, yet it will help forestall burglary and harm to your property.

Here are five reasons why your property needs an expert security review:

1) Security Assessment

Prior to introducing a powerful security framework, it is significant that you comprehend your property. At Condor Security, our security reviews break down your property to recognize powerless focuses in your security frameworks and structure. We at that point give a report to ensure you know about powerless zones that are helpless to vandalism, robbery, and different wrongdoings. Our group of prepared security experts evaluate your property and distinguish where extra safety efforts are fundamental. Contingent upon the necessities of your property, we may suggest camcorder observation, alert frameworks, open air lighting, method changes, and acclimations to security monitor administrations.

2) Risk Assessment and Cost/Benefit Plan

Past our security evaluation administrations, we likewise offer hazard appraisal administrations. This includes a security proficient distinguishing your advantages and organizing your security speculation to guarantee your most important resources are ensured. This procedure starts with resource distinguishing proof and afterward inspects the significant dangers to these benefits. From that point, we will work with you to make a cost/advantage investigation manual for decide how much security innovation you require and make a custom security plan that accommodates your financial plan and guarantees your benefits are ensured. Along these lines, we can make a security framework custom fitted to your interesting needs, so you can get the most insurance for your speculation.

3) Upgrade Your Security System

When you have decided the high-hazard regions of your property, just as the shortcomings in your present security framework, you can start to redesign your framework. At Condor Security, we offer a wide scope of security administrations customized to the special needs of your structure or property. We offer live security video checking with custom cameras in explicit zones of your property, assisting with catching any undesirable occurrences on camera. Recordings can be recorded and seen later, assisting with guaranteeing crimes bring about captures with appropriate proof. Past video checking, we likewise offer alert frameworks and coded get to control frameworks which limit access to the structure, guaranteeing just endorsed people approach your property. We additionally will work with you to guarantee that you see how to keep up your security framework to guarantee it is continually working appropriately, forestalling issues like bogus caution calls to the police. For more data about our alert frameworks and security gear, investigate our site here.

4) Crime Prevention

One of the additional advantages of camcorders, alerts, and other safety efforts is that their insignificant nearness assists with stopping crooks from focusing on your property. Cameras can be introduced in a concealed way, however cameras can likewise be introduced so they are entirely noticeable, out in the open, and in sufficiently bright regions to forestall vandals, hoodlums, and different crooks from focusing on your property. Albeit no home security framework can ensure the wellbeing of your property, we do everything possible to guarantee your benefits and inhabitants are secured and that any criminal demonstrations are gotten on camera so there is adequate proof, should a sad occasion happen.

5) Mobile Patrols and Security Guards

For high-chance regions and properties that have a background marked by being vandalized or ransacked, we likewise offer the administrations of expert security watches and exceptionally prepared portable watch units. The entirety of our security monitors are prepared in wrongdoing discouragement, remembering an accentuation for viable correspondences (Verbal Judo), which includes strategic correspondence to alleviate hazard. At Condor, our gatekeepers are prepared to deflect crooks from entering the premises because of a solid security nearness. Should undesirable people despite everything enter the premises, our watchmen are prepared to recognize undesirable acts and react as needs be. Past private and business properties, we likewise offer security watchmen and portable units for huge modern properties, schools, and huge retail places.

Getting a far reaching security review will assist you with deciding the security needs of your property. Not exclusively would we be able to help recognize the qualities and shortcomings of your present security framework, yet we can likewise see high-chance zones that need as tended to. From that point, our prepared security experts will work with you to make a custom security framework including video observing, alert frameworks, coded section, and some other safety effort that is important to guard your inhabitants and resources. At long last, we likewise offer the administrations of prepared portable units and security monitors. Our watchmen have broad medical aid, CPR, and fire security preparing, while additionally being specialists in verbal judo and other prevention strategies.

5.2 Future Scope

With the rise and expansion of the internet companies, the attack surface is becoming larger for hackers. Security audit or security assessment, when appropriately checked, features resources and usefulness which can be mishandled by an attacker hoping to access an association. Be that as it may, inadequately checked infiltration tests don't generally offer great worth.

Regularly organizations use entrance tests not on the grounds that they truly need to test the security of their frameworks yet rather as a method of conciliating an evaluator or exhibiting consistency. On the off chance that the inspiration is essentially to meet inflexible consistency necessities, at that point the results are regularly not valuable.

Far more atrocious, maybe, a few sellers seem to offer infiltration testing yet then charge a lot of cash to perform what is basically a helplessness and fix evaluation filter utilizing business off the rack items. At that point they take the report from said item, re-identification it, and send it to a client. Unhelpfully, this could tar all entrance testing organizations, to whom such conduct is an abomination, with a similar negative brush.

While simply playing out a powerlessness evaluation helps as it can distinguish any low hanging natural product that could be a possibly simple assault surface for content kiddies or expert aggressors to concentrate on.

It is, as it may, a long way from appropriate infiltration testing which hopes to use the entrance analyzers long periods of experience and underhandedness/tricky to utilize mixed assaults to bargain the client in a fundamentally the same as approach to how genuine assaults may look.

Toward the finish of the commitment imparting the hazard is perhaps the hardest test in both entrance testing and cybersecurity as a rule: how would we make the message coherent to the beneficiary, particularly on the off chance that they don't have a digital foundation (just like the case for some leaders).

Conventional pen-testing and weakness checking can fall into this class — regularly the aftereffects of entrance tests are intricate and possibly tangled that the client doesn't get the full profit by them.

REFERENCES

[1] OWASP top 10 Web [https://storage.googleapis.com/google-code-archive-downloads/v2/code.google.com/owasptop10/OWASP%20Top%2010%20-%202013.pdf]

[2] OWASP Top 10 Mobile [https://owasp.org/www-project-mobile-top-10/]

APPENDICES

1. Client-Side attacks can be examined based on the page source response of the web site in the web browser.
2. Server-Side vulnerabilities can be detected by using two test accounts and manipulating the data within the test accounts.
3. Injection can be done based on the technologies or database used in the web application.

[illegible] 28/05/2020