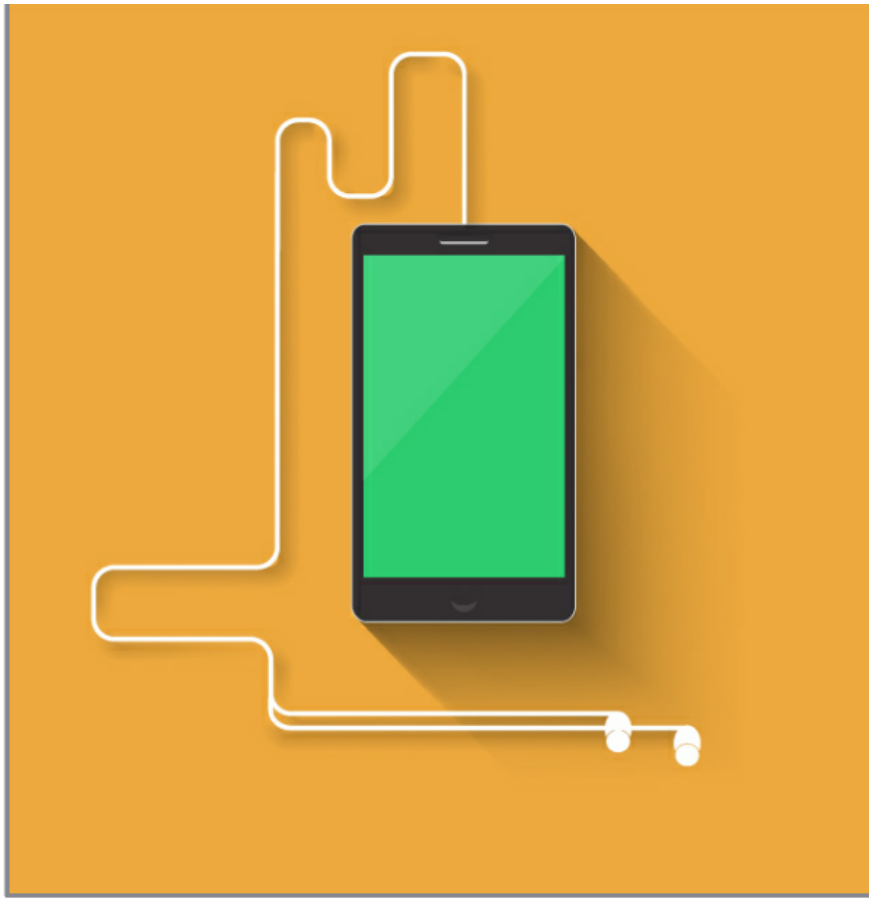


MOBILE APPLICATION

SECURITY ASSESSMENT REPORT

Mobile App: API-APP-SAMPLE.apk (Android App)
Scan ID: 387
12/12/2016

ENTERSOFT

CONFIDENTIALITY & PROPRIETARY

This report is solely for Marketing Purposes. This should not be used, circulated, quoted or otherwise referred to for any other purpose, nor included or referred to in whole or in part in any document without our prior written consent.

The specific sample report of the APK was prepared with a consent from the customer. Our subsequent test work, study of issues in detail and developing action plans are directed towards the issues identified. Consequently this report may not necessarily comment on all the weaknesses perceived as important by CLIENT.

REPORT VALIDITY

The identification of the issues in the report is mainly based on the tests carried out during the limited time for conducting such an exercise. As the basis of selecting the most appropriate weaknesses is purely judgmental in view of the time available, the outcome of the analysis may not be exhaustive and representing all possibilities, though we have taken reasonable care to cover the major eventualities.

The vulnerabilities reported in this report are valid as of DECEMBER 12, 2016. Any vulnerability, which may have been discovered after this or any exploit been made available after DECEMBER 1, 2016, does not come under the purview of this report.

Any configuration changes or software/hardware updates made on hosts/machines on the application covered in this test after the date mentioned herein may impact the security posture either positively or negatively and hence invalidates the claims & observations in this report. Whenever there is an update on the application, we recommend that you conduct penetration test to ensure that your security posture is compliant with your security policies.

RISK ASSESSMENT METHODOLOGY

The severity assigned to each vulnerability was calculated using the NIST 800-30 standard. This standard determines the risk posed by application based on the likelihood an attacker exploits the vulnerability and the impact that it would have on the business. Entersoft also covers Mobile OWASP top 10 to ensure latest security threats are assessed.

LIKELIHOOD	IMPACT
The difficulty of exploiting the described security vulnerability includes required skill level and the amount of access necessary to visit the element susceptible to the vulnerability.	The impact the vulnerability would have on the organization if it were successfully exploited.
CRITICAL	
An attacker is almost certain to initiate the threat event.	The issue causes multiple severe or catastrophic effects on organizational operations, organizational assets or other organizations.
HIGH	
An untrained user could exploit the vulnerability or the vulnerability is very obvious and easily accessible.	Exploitation produces severe degradation in mission capability to the point that the organization is not able to perform primary functions or results in damage to organizational assets.
MEDIUM	
The vulnerability requires some hacking knowledge or access is restricted in some way.	Threat events trigger degradation in mission capability to an extent the application is able to perform its primary functions, but their effectiveness is reduced and there may be damage to organizational assets.
LOW	
Exploiting the vulnerability requires application access, significant time, resource or a specialized skillset.	Successful exploitation has limited degradation in mission capability; the organization is able to perform its primary functions, but their effectiveness is noticeably reduced and may result in minor damage to organizational assets.
MINIMAL	
Adversaries are highly unlikely to leverage the vulnerability.	The threat could have a negligible adverse effect on organizational operations or organizational assets.

SEVERITY

The vulnerability severity is determined using the likelihood and impact weights as shown in the following table:

LIKELIHOOD \ IMPACT	MINIMAL	LOW	MEDIUM	HIGH	CRITICAL
CRITICAL	MINIMAL	LOW	MEDIUM	HIGH	CRITICAL
HIGH	MINIMAL	LOW	MEDIUM	HIGH	CRITICAL
MEDIUM	MINIMAL	LOW	MEDIUM	MEDIUM	HIGH
LOW	MINIMAL	LOW	LOW	LOW	MEDIUM
MINIMAL	MINIMAL	MINIMAL	MINIMAL	LOW	LOW

TABLE OF CONTENTS

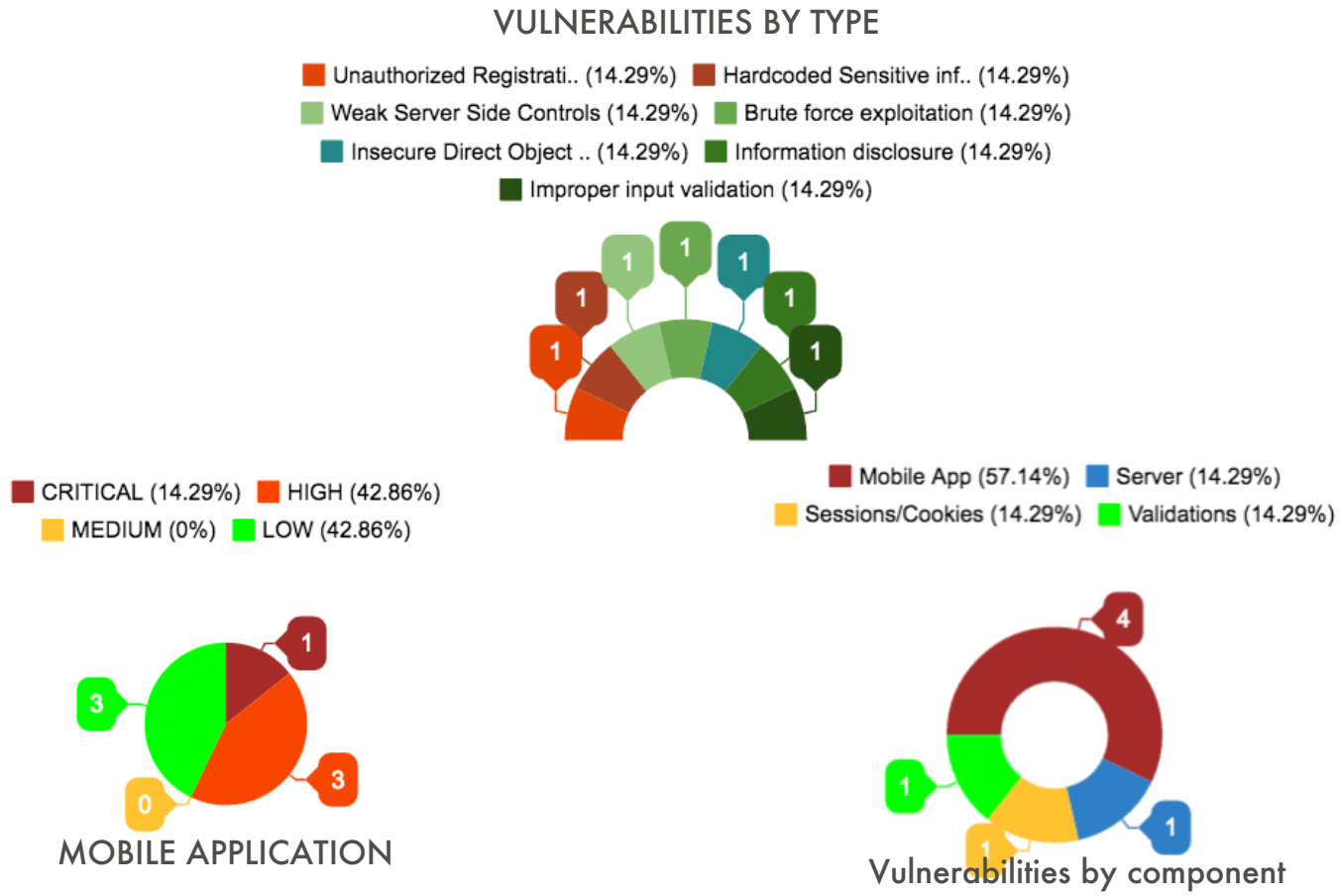
1. EXECUTIVE SUMMARY

1.1 BACKGROUND

On 23 October 2016, ABC Corp Pty Ltd has engaged ENTERSOFT as an application security provider. ENTERSOFT shall continuously evaluate the coding practices at ABC Corp Pty Ltd and ensure the application built for its customers is secure. As a part of the engagement, 1st round of security assessment was performed from 24th November 2016 to 11th December 2016 on their Android Mobile Application (API-APP-SAMPLE.apk - herein after referred as mobile application) in an effort to ensure the security of their customer's personal information and data is secure, which is processed and stored by the mobile application. ENTERSOFT has also assessed the INFRASTRUCTURE associated with the mobile application.

1.2 RISKS

During the course of this engagement we observed several areas of concern that we believe could pose risk to the security of the application, and should be addressed in a timely manner. Exploiting these vulnerabilities an attacker can cause severe losses to the CLIENT as well as end users. Application is not robust to authorisation attacks and attackers can retrieve unauthorised data. Attackers can also cause financial losses to the end users. Risks are distributed as below



2. RECOMMENDATIONS

2.1 OBSERVATIONS

While performing the assessment of ABC Corp Pty Ltd, ENTERSOFT has identified that security controls were not effective in resisting common attacks like:

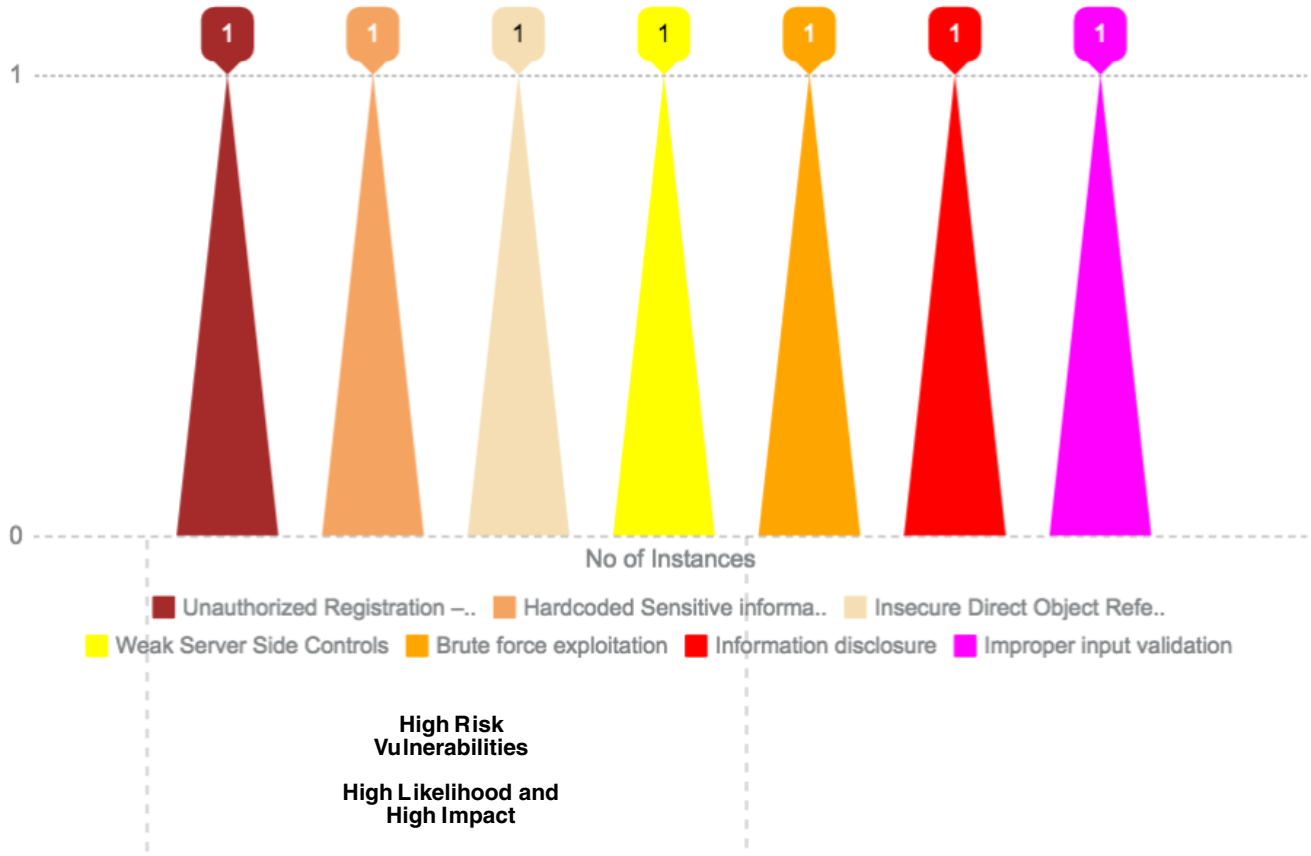
- Authorization Attacks
- Brute force attacks
- Sensitive Information Leakage

2.2 IMMEDIATE ACTIONABLE RECOMMENDATIONS

- Remove hardcoded sensitive information
- Obfuscate the source code- Validate all user inputs based on a whitelisting approach.
- Validate inputs at both Server level and at application level.
- Implement Authorization properly based on a whitelisting approach.
- Mask sensitive information in Server errors.

3. FINDINGS

3.1 SUMMARY OF FINDINGS



High Risk Vulnerabilities

High Likelihood and High Impact

4. DETAILS

4.1 CRITICAL SEVERITY BUGS

Identified bugs are considered CRITICAL based on the below table. These CRITICAL severity bugs must be fixed immediately.

ENTERSOFT		Impact				
		Minimal	Low	Medium	High	Critical
Likelihood	Critical	Minimal	Low	Medium	High	Critical
	High	Minimal	Low	Medium	High	Critical
	Medium	Minimal	Low	Medium	Medium	High
	Low	Minimal	Low	Low	Low	Medium
	Minimal	Minimal	Minimal	Minimal	Low	Low

Critical functional bug has been identified at the registration screen. An user can be locked out of the app from registration using fuzzing

4.1.1 Unauthorized Registration – Account Lockout (Business Logic Flaw) #ABC-1

DESCRIPTION

An attacker can run a simple script to disable registration process for new users(Any Mobile number). New users will panic as they will be already registered. This serious vulnerability can directly affect the brand and new registrations.

An attacker can intercept the REGISTRATION service request and can fuzz the MOBILENO parameter with any mobile number. Since the fuzzing can be automated, any mobile number can be used to register a new account. More than 1000 fuzzed registration requests were sent to the server in a minute and the server has accepted all the requested. 1000 users have been locked out of registration process. After this attack the genuine user will be directly asked for password instead of registration. A genuine user with an intention to register the app will already be registered with a different password and can never register to the app.

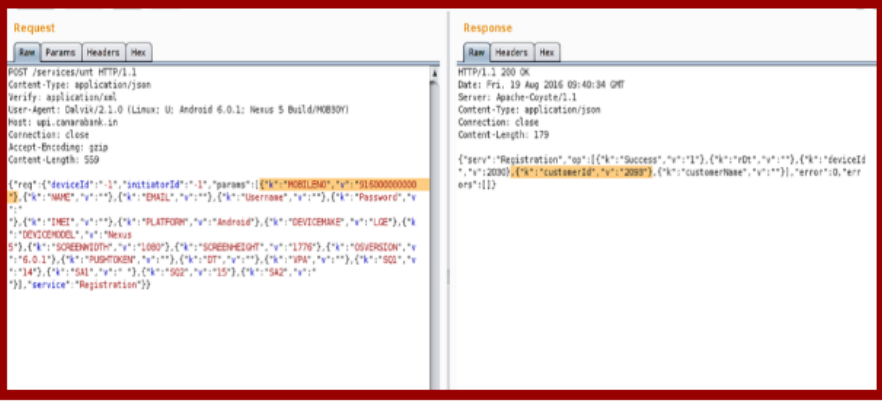
INSTANCES

Registration Service

Vulnerable Parameter: MOBILENO

EVIDENCE

Refer Video POCVIDEO-Account-Lockout.mp4



STEPS TO REPRODUCE

Intercept traffic using BURP Suite. A custom script has been implemented by Entersoft's white hat hackers to fuzz the MOBILENO Parameter. A sample fuzzer can be found here https://github.com/bunqcom/fuzzer

Watch POC Video to reproduce the bug

REMEDIATION

The following validations can be implemented to validate the mobile number

- Bind the device hardware signature like IMEI number with a mobile number
- Generate OTP to the mobile number to validate the genuineness of the mobile number

4.1 HIGH SEVERITY BUGS

Identified bugs are considered HIGH based on the below table. These HIGH severity bugs must be fixed immediately.

ENTERSOFT		Impact				
		Minimal	Low	Medium	High	Critical
Likelihood	Critical	Minimal	Low	Medium	High	Critical
	High	Minimal	Low	Medium	High	Critical
	Medium	Minimal	Low	Medium	Medium	High
	Low	Minimal	Low	Low	Low	Medium
	Minimal	Minimal	Minimal	Minimal	Low	Low

High sensitive information like API Keys are identified after reverse engineering the mobile app. Mobile app is vulnerable to business logic flaw which can be used to exploit to get extra cash in the user account

4.2.1 Hardcoded Sensitive information #ABC-2

DESCRIPTION

Application can be reversed and attacker can access Hard coded sensitive information. Information such as passwords, server IP addresses, and encryption keys can expose valuable information to the attackers. Attacker could reverse engineer the application and can access the class files. They can be decompiled and attacker can discover the sensitive information.

INSTANCES

Entire Application

Vulnerable Parameter/Component: APK

EVIDENCE

Refer Video POCVIDEO-SensitiveInformation.mp4

```
public class MainActivity extends AppCompatActivity
  implements DrawerCallbacks, AdapterView.OnItemClickListener,
OnWiFiTaskCompleteListner, LocationInterface, AdmofiViewCallback,
AdmofiOffersCallback
{
  public static String AD_APPID;
  private static final String LOG_TAG = "Map View";
  private static final String OUT_JSON = "/json";
  private static final String PLACES_API_BASE =
"https://maps.googleapis.com/maps/api/place";
  private static final String SERVER_API_KEY =
"AIzaSyDL4Rn79LorjHInQJNvKfpFM0VGkxvqJy4";
  private static final String TYPE_AUTOCOMPLETE =
"/autocomplete";
  public static int height;
  public static AdmofiHelper mAdmofiHelper = null;
  public static int width;
  FrameLayout adLayout;
  LinearLayout advertisementlayout;
  CommunicatorInterface communicatorInterface;
  DBHelper dbHelper;
  Display display;
  ImageView guideimg;
  TextView locationnametv;
  TextView locationtv;
  MyApplication mApp;
  private Nav_DrawerFragment mNavigationNavDrawerFragment;
  private FragmentTabHost mTabHost;
```

STEPS TO REPRODUCE

Refer POC viedo. Application has been reversed using Entersoft's inhouse Application reversing tool.

REMEDIATION

Encrypt or Obfuscate the code. Sensitive information should be accessed dynamically from a server DB. The ideal solution is to retrieve the server IP address and any other sensitive information from the database, and the database in turn must be encrypted.

4.2.2 Weak Server Side Controls (Business Logic Flaw) #ABC-3

DESCRIPTION

Lack of server side validation allows an attacker to exploit business logic to get extra cash. Fake bills can be uploaded multiple times in a user account with a valid bill ID. Bill validation does not happen at the server side.

Attack Scenario: This attack can be performed if attacker know a valid bill ID of previously uploaded bill. Attacker can upload a same bill and manipulate the bill ID during the transit using packet manipulators. We have successfully exploited the vulnerability and bill has been approved and application has rewarded the attacker with cash.

INSTANCES

Bug Location:
Bill Upload Screen
https://mobile-api.abc.com/api/services/billupload

EVIDENCE

Refer

POCVIDEO-UnlimitedCash.mp4

STEPS TO REPRODUCE

Refer POC Video.

REMEDIATION

It's a good practice to have both client side and server side validations. Inputs can be tampered in transit to exploit vulnerabilities if just client side validation is implemented. The file signature must be verified against the bills already uploaded to prevent the duplicates and send a prompt to customer that the bill is already uploaded.

4.2.3 Insecure Direct Object Reference #ABC-4

DESCRIPTION

Applications frequently use the actual name or key of an object when generating web pages. Applications don't always verify the user is authorized for the target object. This results in an insecure direct object reference flaw. Attackers can easily manipulate parameter values to detect such flaws.

The CUSTOMERID field that is passed in the Json request for the service API_GetAccounts can be fuzzed with a range of values to deduce the other customer's sensitive information.

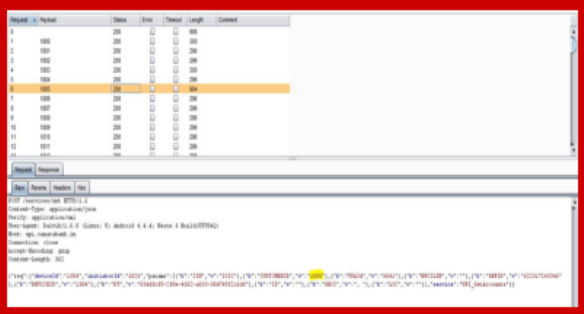
INSTANCES

Bug Location:
API_GetAccounts

EVIDENCE

Refer

POCVIDEOInsecureDOreference.mp4



STEPS TO REPRODUCE

Refer POC Video.

REMEDIATION

Preventing insecure direct object references requires selecting an approach for protecting each user accessible object (e.g., object number, filename):

1. Use per user or session indirect object references. This prevents attackers from directly targeting unauthorized resources. For example, instead of using the resource's database key, a drop-down list of six resources authorized for the current user could use the numbers 1 to 6 to indicate which value the user selected. The application must map the per-user indirect reference back to the actual database key on the server. OWASP's ESAPI includes both sequential and random access reference maps that developers can use to eliminate direct object references.

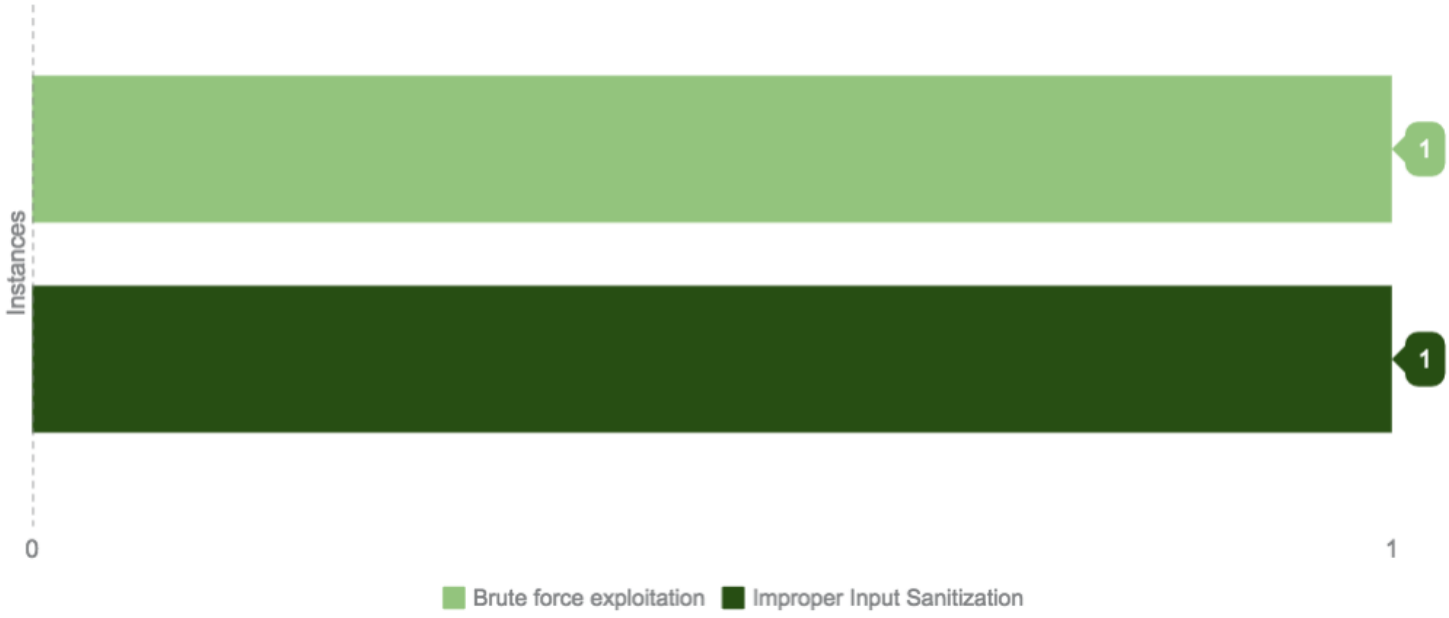
2. Check access. Each use of a direct object reference from an untrusted source must include an access control check to ensure the user is authorized for the requested object.

4.2 LOW SEVERITY BUGS

Identified bugs are considered LOW based on the below table. These bugs have to fixed at earliest convinience.

ENTERSOFT		Impact				
		Minimal	Low	Medium	High	Critical
Likelihood	Critical	Minimal	Low	Medium	High	Critical
	High	Minimal	Low	Medium	High	Critical
	Medium	Minimal	Low	Medium	Medium	High
	Low	Minimal	Low	Low	Low	Medium
	Minimal	Minimal	Minimal	Minimal	Low	Low

The following instances of LOW severity bugs are idenfied in the web application & its Server.



4.3.1 Lack of server side validation to enforce password policy: Brute force exploitation #ABC-5

DESCRIPTION

Password policy must be implemented properly on the server side. An attacker can bypass password policy restrictions by intercepting an ongoing request.

Our white hat hackers has manipulated an intercepted password and sent a weak password to the server. Server has accepted the password without validating the password policy. Password policy is implemented at client level only. A brute force attack has been executed on weak password and exploited.

INSTANCES

Bug URL:
https://api.zaggle.in/api/v1/profile/password/reset
Vulnerable Parameter: Password

EVIDENCE

Refer BruteforceExploit.mp4

REMEDIATION

Implement strong password policy at server level as well to avoid brute force attacks. Passwords should consist a minimum of 8 characters in length along with special characters.

4.3.2 Web Server Information Disclosure #ABC-6

DESCRIPTION

Improper handling of errors can introduce a variety of security problems for aN APPLICATION. The most common problem is when detailed internal error messages such as internal paths, and error codes are displayed to the user (hacker). These messages reveal implementation details that should never be revealed. Such details can provide hackers important clues on potential flaws in the site and such messages are also disturbing to normal users.

During the information gathering or reconnaissance phase the attacker equips himself with sensitive information like web server information. Any leakage of this information can give the attacker a better chance of exploiting it with more targeted exploits that are available to that web server. An attacker can easily obtain web server details, the webserver used for the mobile app is Apache Tomcat 6.0.13. This information can be used as a combination to

INSTANCES

Bug Location: GetMasterData

EVIDENCE

Evidence images below

REMEDIATION

Mask the error messages. Generate 404 Error pages without disclosing versions of the software used in the Server.

4.3.3 Improper Input Sanitization #ABC-7

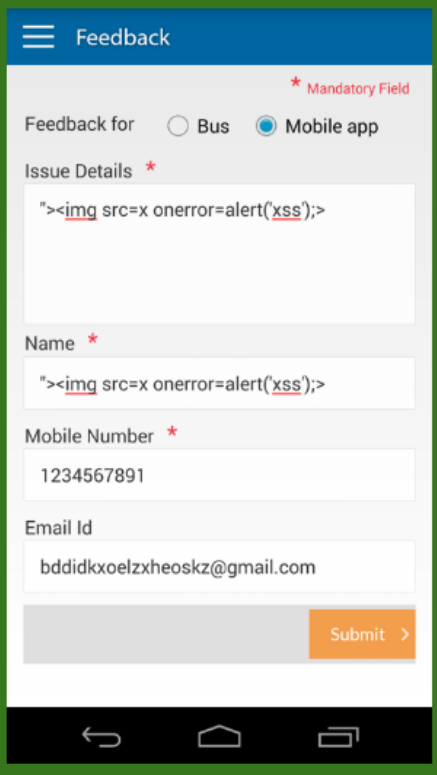
DESCRIPTION

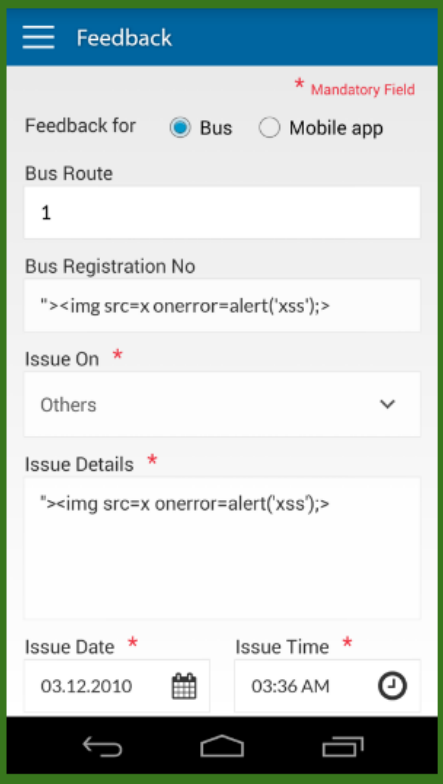
The application has a feedback mechanism where users can submit their feedback. The input fields in the feedback screen are not properly sanitized both at client and server level. This vulnerability is not exploitable but an attacker may use this bug to exploit other bugs

INSTANCES

Bug Location: Feedback Screen

EVIDENCE





REMEDIATION

Whitelist the allowable characters for each field and avoid accepting any characters out of the whitelisted characters.

5. ABOUT ENTERSOFT

Entersoft is an award winning application security provider trusted by over 300 global brands. Through our bespoke products and services we help build robust, secure applications.

Our approach is a combination of offensive assessment, proactive monitoring and pragmatic managed security which provides highly cost effective and reliable solutions to some of the most pressing problems in Application Security.

We work on real problems with real methods, and seek to understand the foundations of those methods. We stay ahead of the curve by working on a variety of cutting edge technologies with rifle focus on quality.

We aim to reduce the overall risk of your apps.

Led by ex-intelligence, Entersoft brims with interesting twists on traditional ways of operating. Our team of certified White Hat Hackers with diverse backgrounds and from various parts of the world, carry an overall experience of 40 years in breaking applications apart.

What makes us undeniably good is our core philosophy of getting the best minds in the business, measured by various standards conceptual creativity, speed, problem solving ability and brute force implementation.

Our certifications speak about our expertise. Entersoft aims at providing highly skilled and offensive white hat hackers who are fast, reliable and trustworthy.

Our team holds the world class certifications like OSCP, OSWE etc.