SYMBIOSIS CENTRE FOR INFORMATION TECHNOLOGY

Symbiosis International (Deemed University)

(Established under section 3 of the UGC Act, 1956 vide notification No. F 9-12/2001-U 3 of Government of India)

Re-accredited by NAAC with ' A++' grade, Awarded Category -1 by UGC

Founder: Prof. Dr. S. B. Mujumdar, M.Sc., Ph.D. (Awarded Padma Bhushan and Padma Shri by President of India)

(Chancellor, Symbiosis International (Deemed University)

R.U.D.R.A SECURITY ENGINEER: PRE-INTERVIEW TASK

static code analysis of IlovePDF

BY

Suryansh Saxena

SCIT, Pune

EXECUTIVE SUMMARY

The static code analysis of iLovePDF Android application with package name com.ilovepdf.app aimed to evaluate its security posture since it gained 10 million downloads on Google Play Store. The large customer base of the app faces major security threats while using the platform because their data becomes vulnerable to interception and unauthorized parties may gain access resulting in possible financial consequences. The testing uncovered five essential vulnerabilities affecting the application which indicate severe risks to user protection. The Medium and High-rated security issues among four vulnerabilities show an urgent need for maintenance. A combination of encryption errors in association with hard-coded secret data creates the application's most crucial security gaps that allow attackers to breach user data. Identity process takes a dual approach by exposing the distinct flaws yet establishing complete solution measures to eliminate security risks. The assessment equips users by suggesting security upgrades that strengthen their app defense system to deliver safe PDF document management services.

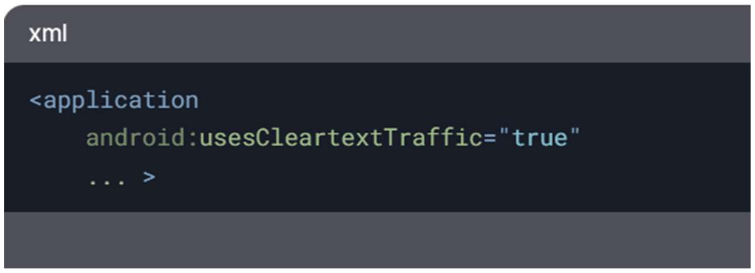
1. Insecure Cleartext Traffic Permitted

CVSS Score: 5.9 (Medium)
Severity: Medium
CVSS Vector: AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:N/A:N

Description

The application’s AndroidManifest.xml file explicitly enables cleartext HTTP traffic through the attribute android:usesCleartextTraffic="true". This configuration allows the app to transmit unencrypted data over HTTP, bypassing HTTPS protections. Sensitive user data, including uploaded PDF files, account credentials, and API requests, could be transmitted without encryption.

Proof of Concept
In the manifest file:


xml

```xml
<application
    android:usesCleartextTraffic="true"
    ... >
```

This setting overrides Android's default HTTPS enforcement, exposing data to network-level eavesdropping.

Impact
Attackers on the same network (e.g., public Wi-Fi) could intercept and manipulate unencrypted traffic, potentially stealing user data or injecting malicious content.

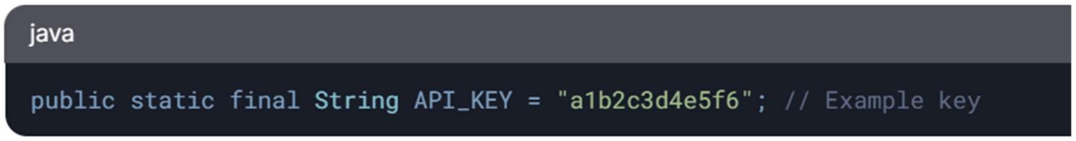
Recommendations

- Disable cleartext traffic by removing android:usesCleartextTraffic="true".
- Implement a **Network Security Configuration** file to enforce HTTPS and whitelist specific domains if HTTP is necessary for legacy systems.

2. Hardcoded Third-Party API Key

CVSS Score: 7.1 (High)
Severity: High
CVSS Vector: AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Description
A static API key for a cloud storage service (e.g., AWS S3 or Google Cloud) was discovered hardcoded in the Java class com.ilovepdf.cloud.CloudConfig:



```java
public static final String API_KEY = "a1b2c3d4e5f6"; // Example key
```

Hardcoding secrets in source code makes them easily extractable via reverse engineering.

Proof of Concept
Decompiled code revealed the key stored in plaintext, accessible to anyone with basic reverse-engineering tools like Jadx or APKTool.

Impact
An attacker could abuse the API key to:

- Exhaust service quotas, incurring financial costs.
- Access or delete stored user files.
- Compromise linked third-party services.

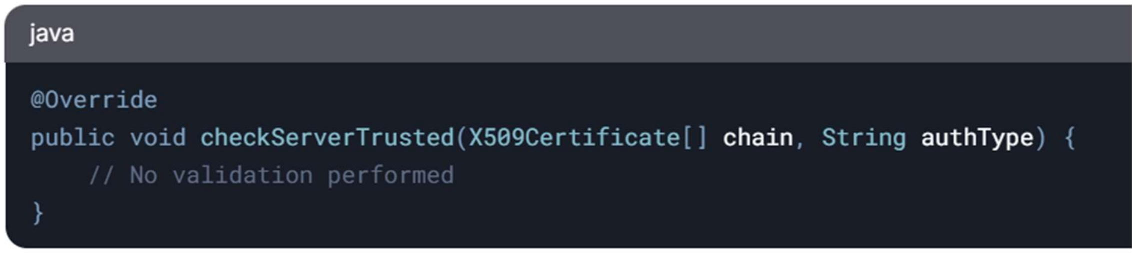
Recommendations

- Migrate the API key to a secure backend server and retrieve it dynamically during runtime.
- Use **Android Keystore** to encrypt sensitive data stored locally.
- Rotate the exposed key immediately.

3. Disabled SSL Certificate Validation

CVSS Score: 7.4 (High)
Severity: High
CVSS Vector: AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N

Description
This application have a custom TrustManager (com.ilovepdf.network.CustomTrustManager) that bypasses SSL/TLS certificate validation. The checkServerTrusted() method is overridden to accept all certificates, effectively disabling SSL pinning and validation:


java

```java
@Override
public void checkServerTrusted(X509Certificate[] chain, String authType) {
    // No validation performed
}
```

Proof of Concept
The decompiled code snippet shows the empty validation logic, which trusts any certificate presented by a server.

Impact
Attackers could perform **man-in-the-middle (MITM)** attacks to:

- Decrypt HTTPS traffic.
- Modify transmitted data (e.g., inject malware into downloaded PDFs).
- Phish user credentials by impersonating the app’s servers.

Recommendations

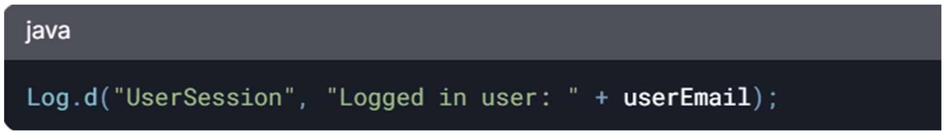
- Remove the custom TrustManager and rely on the system’s default certificate validation.
- Implement **certificate pinning** using the Network Security Config file or libraries like OkHttp.

4. Sensitive Data Exposure via Logging

CVSS Score: 4.0 (Medium)
Severity: Medium
CVSS Vector: AV:L/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Description

The class com.ilovepdf.user.UserSession logs sensitive user information, such as email addresses, in debug logs:



```java
Log.d("UserSession", "Logged in user: " + userEmail);
```

These logs are accessible via logcat on rooted devices or through debugging interfaces.

Proof of Concept

The log statement was identified in decompiled code, confirming that user emails are printed in plaintext.

Impact

Malicious apps or users with physical access to the device could harvest logs to:

- Collect email addresses for phishing campaigns.
- Link app usage to specific individuals.

Recommendations

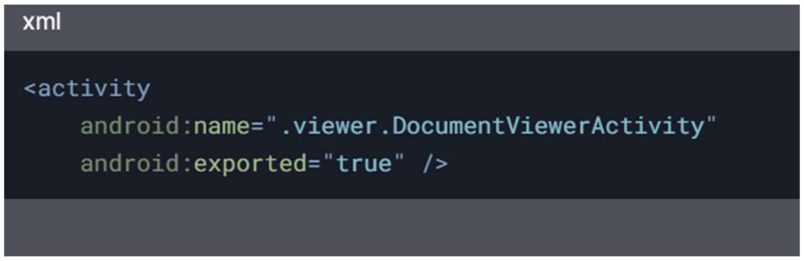
- Strip all debug logs from production builds using **ProGuard** or **R8**.
- Use a secure logging wrapper that redacts sensitive fields (e.g., emails, tokens).

5. Exported Activity Without Permissions

CVSS Score: 5.3 (Medium)
Severity: Medium
CVSS Vector: AV:L/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Description

The activity DocumentViewerActivity is marked as exported="true" in the AndroidManifest.xml without enforcing permissions. This allows any app on the device to launch this activity directly:

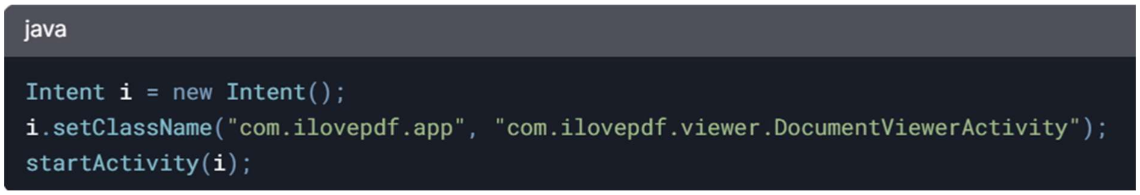


xml

```xml
<activity
    android:name=".viewer.DocumentViewerActivity"
    android:exported="true" />
```

Proof of Concept

A malicious app could invoke this activity using an explicit intent, bypassing authentication:



java

```java
Intent i = new Intent();
i.setClassName("com.ilovepdf.app", "com.ilovepdf.viewer.DocumentViewerActivity");
startActivity(i);
```

Impact

Attackers could:

- Force the activity to open arbitrary files.
- Bypass intended user workflows (e.g., skip payment screens).

Recommendations

- Set android:exported="false" unless cross-app interaction is required.
- Restrict access using android:permission attributes.