RADICALLY OPEN SECURITY

Penetration Test Report

Tracking the Trackers

V 1.0
Amsterdam, September 28th, 2022
Confidential

Document Properties

Client	Tracking the Trackers
Title	Penetration Test Report
Targets	Deployment scripts F-droid client
Version	1.0
Pentester	Abhinav Mishra
Authors	Abhinav Mishra, Marcus Bointon
Reviewed by	Marcus Bointon
Approved by	Melanie Rieback

Version control

Version	Date	Author	Description
0.1	September 19th, 2022	Abhinav Mishra	Initial draft
0.2	September 28th, 2022	Marcus Bointon	Review
1.0	September 28th, 2022	Marcus Bointon	1.0

Contact

For more information about this document and its contents please contact Radically Open Security B.V.

Name	Melanie Rieback
Address	Science Park 608 1098 XH Amsterdam The Netherlands
Phone	+31 (0)20 2621 255
Email	info@radicallyopensecurity.com

Table of Contents

1 Executive Summary

1.1 Introduction

Between September 5, 2022 and September 19, 2022, Radically Open Security B.V. carried out a penetration test for Tracking the Trackers

This report contains our findings as well as detailed explanations of exactly how ROS performed the penetration test.

1.2 Scope of work

The scope of the penetration test was limited to the following targets:

- Deployment scripts
- F-droid client

The scoped services are broken down as follows:

- Review of deployment scripts: 1 days
- F-droid client review: 3 days
- Reporting: 1 days
- **Total effort: 5 days**

1.3 Project objectives

ROS will perform a penetration test of the F-droid client and review the deployment scripts with Tracking the Trackers in order to assess their security. To do so ROS will access the F-droid client app and the deployment scripts, and guide Tracking the Trackers in attempting to find vulnerabilities, exploiting any such found to try and gain further access and elevated privileges.

1.4 Timeline

The Security Audit took place between September 5, 2022 and September 19, 2022.

1.5 Results In A Nutshell

During this crystal-box penetration test we found 2 Elevated and 3 Low-severity issues.

The review of the deployment scripts did not uncover any security issues. However, the F-droid client application had some misconfiguration. The elevated severity issues are about insecure configuration of TLS, where the application backend accepts TLS version 1.0 and 1.1. The other elevated severity issue is about a possible issue with the XML parser in the application.

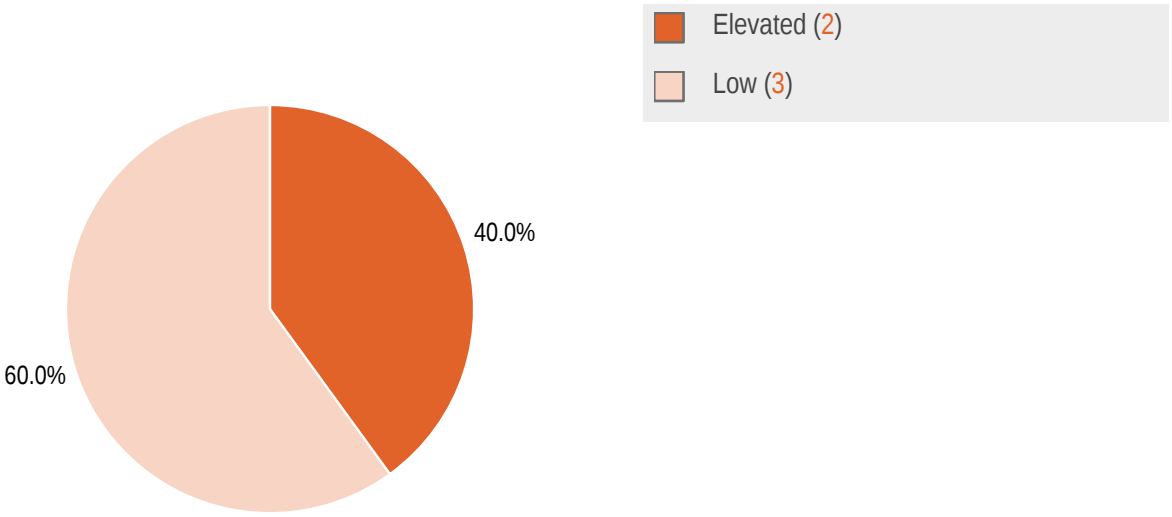
Three low-severity issues were also discovered related to the Android app leaking URLs in the Android log, an insecure base network config, and the cryptography implementation in the app.

By exploiting these issues, an attacker might be able to capture the traffic between the Android client and the server, leak information from the device, perform XXE attacks etc.

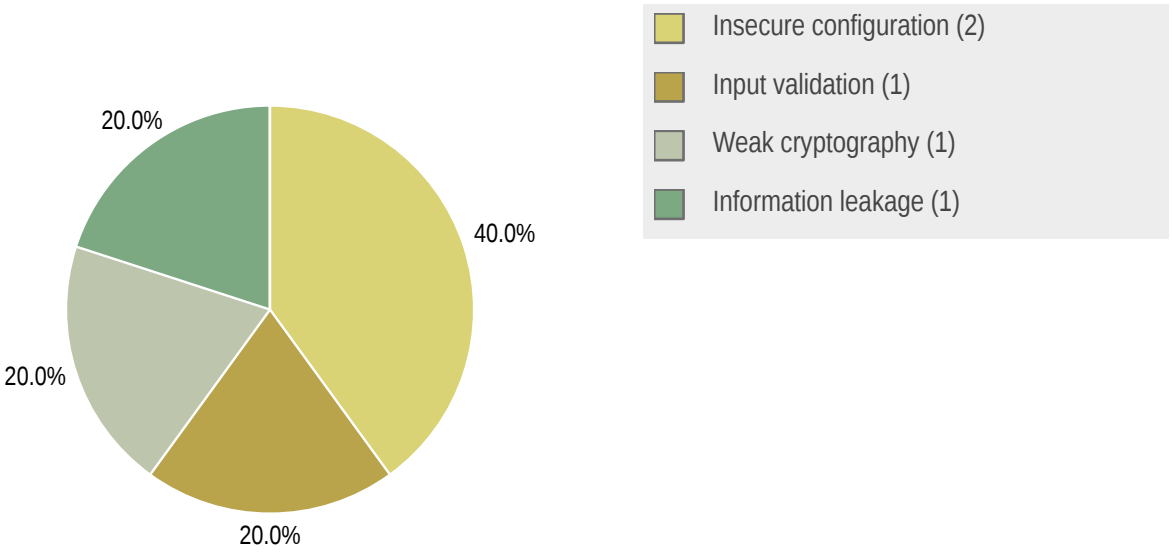
1.6 Summary of Findings

ID	Type	Description	Threat level
CLN-002	Input validation	The application's XML parser implementation might be vulnerable to XML External entity (XXE) attacks.	Elevated
CLN-005	Insecure configuration	The backend domains used by the application accept obsolete TLS 1.0 and TLS 1.1 protocols.	Elevated
CLN-001	Weak Cryptography	The encryption algorithms used in the app use an insecure mode and padding scheme.	Low
CLN-003	Insecure configuration	The base network config of the application allows clear-text traffic.	Low
CLN-004	Information leakage	The Android app (org.fdroid.fdroid.debug ver. 1.14-alpha3-505-gc8514adb9-debug) logs URLs.	Low

1.6.1 Findings by Threat Level



1.6.2 Findings by Type



1.7 Summary of Recommendations

ID	Type	Recommendation
CLN-002	Input validation	• Limit resolution of external entities when XML input is parsed.

CLN-005	Insecure configuration	• Unless support for legacy browsers/devices is needed, disable TLS 1.0 and TLS 1.1. • If TLS 1.0 support is required, disable TLS 1.0 compression to avoid CRIME attacks.
CLN-001	Weak Cryptography	• Replace all uses of `RSA/ECB/PKCS1Padding` and `SHA1` with more secure alternatives.
CLN-003	Insecure configuration	• Unless it is very explicitly needed for the app to work, do not allow or use unencrypted, clear-text traffic. • If it is needed, only allow it to explicitly permitted, trusted domains.
CLN-004	Information leakage	• Avoid logging any sensitive information like usernames, passwords, URLs, tokens etc in the Android log.

2 Methodology

2.1 Planning

Our general approach during penetration tests is as follows:

1. **Reconnaissance**
   We attempt to gather as much information as possible about the target. Reconnaissance can take two forms: active and passive. A passive attack is always the best starting point as this would normally defeat intrusion detection systems and other forms of protection afforded to the app or network. This usually involves trying to discover publicly available information by visiting websites, newsgroups, etc. An active form would be more intrusive, could possibly show up in audit logs and might take the form of a social engineering type of attack.
2. **Enumeration**
   We use various fingerprinting tools to determine what hosts are visible on the target network and, more importantly, try to ascertain what services and operating systems they are running. Visible services are researched further to tailor subsequent tests to match.
3. **Scanning**
   Vulnerability scanners are used to scan all discovered hosts for known vulnerabilities or weaknesses. The results are analyzed to determine if there are any vulnerabilities that could be exploited to gain access or enhance privileges to target hosts.
4. **Obtaining Access**
   We use the results of the scans to assist in attempting to obtain access to target systems and services, or to escalate privileges where access has been obtained (either legitimately though provided credentials, or via vulnerabilities). This may be done surreptitiously (for example to try to evade intrusion detection systems or rate limits) or by more aggressive brute-force methods. This step also consist of manually testing the application against the latest (2017) list of OWASP Top 10 risks. The discovered vulnerabilities from scanning and manual testing are moreover used to further elevate access on the application.

2.2 Risk Classification

Throughout the report, vulnerabilities or risks are labeled and categorized according to the Penetration Testing Execution Standard (PTES). For more information, see: http://www.pentest-standard.org/index.php/Reporting

These categories are:

- **Extreme**
  Extreme risk of security controls being compromised with the possibility of catastrophic financial/reputational losses occurring as a result.

- **High**

  High risk of security controls being compromised with the potential for significant financial/reputational losses occurring as a result.
- **Elevated**

  Elevated risk of security controls being compromised with the potential for material financial/reputational losses occurring as a result.
- **Moderate**

  Moderate risk of security controls being compromised with the potential for limited financial/reputational losses occurring as a result.
- **Low**

  Low risk of security controls being compromised with measurable negative impacts as a result.

3 Reconnaissance and Fingerprinting

We were able to gain information about the software and infrastructure through the following automated scans. Any relevant scan output will be referred to in the findings.

- Mobexler – https://mobexler.com/
- Frida – https://github.com/frida/frida
- Nuclei – https://github.com/projectdiscovery/nuclei
- Sonarqube – https://www.sonarqube.org/

4 Findings

We have identified the following issues:

4.1 CLN-002 — XML parsers might be vulnerable to XXE attacks

Vulnerability ID: CLN-002

Vulnerability type: Input validation

Threat level: Elevated

Description:

The application's XML parser implementation might be vulnerable to XML External entity (XXE) attacks.

Technical description:

The XML standard allows the use of external and internal entities. These are declared in the DOCTYPE of the document. When parsing an XML file or input, the content of the external entities is retrieved from an external storage such as the file system or network.

The following parts of the application implement an XML parser, which might be susceptible to XXE attacks.

Affected Instances:

- src/test/java/org/fdroid/fdroid/data/RepoXMLHandlerTest.java
- src/main/java/org/fdroid/fdroid/IndexUpdater.java
- src/testShared/java/org/fdroid/fdroid/mock/RepoDetails.java

Code snippet:

```
SAXParserFactory factory = SAXParserFactory.newInstance();
```

Impact:

If the XML parser has no restrictions for external and internal entities, then this might lead to arbitrary file disclosures or server-side request forgery (SSRF) vulnerabilities when XML input is parsed.

Recommendation:

Limit resolution of external entities when XML input is parsed. This can be done by:

- If it is not needed, disable external entity support altogether.
- If `DOCTYPE` handling is not necessary, disable support for all `DOCTYPE` declarations
- If external entities are required, then use only required protocols (eg: https) and use an entity resolver to resolve only trusted entities.

In this case, we can use `setFeature` for the SAX parser to disable the DOCTYPE declaration completely:

```
try {
      factory.setFeature("http://apache.org/xml/features/disallow-doctype-decl", true);

      SAXParser saxParser = factory.newSAXParser();

      PrintAllHandlerSax handler = new PrintAllHandlerSax();

      saxParser.parse(FILENAME, handler);

  } catch (ParserConfigurationException | SAXException | IOException e) {
      e.printStackTrace();
  }
```

In this code, if the SAX parser parses any external entities, it will throw an error.

4.2 CLN-005 — Vulnerable TLS versions accepted

Vulnerability ID: CLN-005

Vulnerability type: Insecure configuration

Threat level: Elevated

Description:

The backend domains used by the application accept obsolete TLS 1.0 and TLS 1.1 protocols.

Technical description:

The Android app communicates with multiple domains. The TLS implementations on these domains accepts weaker TLS versions, specifically 1.0 and 1.1.

Affected Instances

- f-droid.org (TLS 1.0, 1.1, 1.2 and 1.3 enabled)

- fdroid.tetaneutral.net (TLS 1.0, 1.1 and 1.2 enabled)

These obsolete protocols may be affected by well-known vulnerabilities such as FREAK, POODLE, BEAST, and CRIME.

As TLS 1.2 (on fdroid.tetaneutral.net) and 1.3 (on fdroid.tetaneutral.net) is also accepted, any communication using these later versions does not present a risk.

Android has provided support for TLS 1.2 since platform version 16, released in 2012 in Android 4.1 "Jelly Bean", so the term "legacy" would apply to devices older than that.

Impact:

Use of TLS 1.0 and 1.1 make the communication susceptible to downgrade attacks, as they work on SHA-1 hash for the integrity of exchanged messages. The handshake authentication is also done on SHA-1, which makes it easier for an attacker to impersonate a server for MITM attacks. The PCI DSS (Payment Card Industry Data Security Standard) specifies that TLS 1.0 may no longer be used as of June 30, 2018, and also strongly recommends disabling 1.1, so this may impact compliance with regulations.

Recommendation:

- Unless support for legacy browsers/devices is needed, disable TLS 1.0 and TLS 1.1.
- If TLS 1.0 support is required, disable TLS 1.0 compression to avoid CRIME attacks.

4.3 CLN-001 — Encryption algorithms using insecure mode and padding scheme

Vulnerability ID: CLN-001

Vulnerability type: Weak Cryptography

Threat level: Low

Description:

The encryption algorithms used in the app use an insecure mode and padding scheme.

Technical description:

The following sections of the application code contain cryptographically weak implementations:

Affected Instances:

- src/full/java/kellinwood/security/zipsigner/ZipSignature.java
- src/full/java/kellinwood/security/zipsigner/optional/PasswordObfuscator.java

Affected code snippet:

```
public ZipSignature() throws IOException, GeneralSecurityException {
md = MessageDigest.getInstance("SHA1");
cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
```

```
getLogger().debug("Sig File SHA1: \n" + HexDumpEncoder.encode(sfDigest));
getLogger().debug("Signature: \n" + HexDumpEncoder.encode(signatureBytes));
Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
```

```
getLogger().debug("Sig File SHA1: \n" + HexDumpEncoder.encode(sfDigest));
getLogger().debug("Signature: \n" + HexDumpEncoder.encode(signatureBytes));
Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
```

It is important to analyse the whole code base to see whether any other code is affected, and implement fixes there too, if needed.

Encryption operations should use a secure mode and padding scheme so that confidentiality and integrity can be guaranteed. In this case, check the code to find all instances where `RSA/ECB/PKCS1Padding` and `SHA1` are being used. If sensitive data is being encrypted, then consider improving the encryption mode and padding.

Impact:

If sensitive data is being encrypted using an insecure mode and padding, it might lead to data being stolen or recovered from the encrypted data.

Recommendation:

- Replace all uses of `RSA/ECB/PKCS1Padding` and `SHA1` with more secure alternatives.

4.4 CLN-003 — Clear text traffic is enabled in the application

Vulnerability ID: CLN-003

Vulnerability type: Insecure configuration

Threat level: Low

Description:

The base network config of the application allows clear-text traffic.

Technical description:

The Network Security Configuration allows apps to customize their network security settings. These settings can be configured for specific domains and for a specific app, for example; customize which certificate authorities (CA) are trusted for an app's secure connections, protect apps from accidental usage of cleartext traffic etc.

The Android application (`org.fdroid.fdroid.debug ver.1.14-alpha3-505-gc8514adb9-debug`) contains `network_security_config.xml` with the following configuration:

```
<base-config cleartextTrafficPermitted="true"/>
```

This configuration allows the app to make connections to unencrypted resources.

Impact:

Allowing clear-text traffic impacts confidentiality, authenticity, and protection against tampering. An attacker performing a machine-in-the-middle attack can eavesdrop on transmitted data and modify it without being detected.

Recommendation:

- Unless it is very explicitly needed for the app to work, do not allow or use unencrypted, clear-text traffic.
- If it is needed, only allow it to explicitly permitted, trusted domains.

4.5 CLN-004 — HTTP Request URLs are logged

Vulnerability ID: CLN-004

Vulnerability type: Information leakage

Threat level: Low

Description:

The Android app (org.fdroid.fdroid.debug ver. 1.14-alpha3-505-gc8514adb9-debug) logs URLs.

Technical description:

We noticed that the application logs URLs in the Android log.

```
2022-09-14 17:58:28.793 11840-11840/org.fdroid.fdroid.debug D/DownloaderService: Received Intent for downloading: Intent { act=org.fdroid.fdroid.net.DownloaderService.action
 .QUEUE dat=https://f-droid.org/... cmp=org.fdroid.fdroid.debug/org.fdroid.fdroid.net.DownloaderService (has extras) } (with a startId of 1)
2022-09-14 17:58:28.793 11840-11840/org.fdroid.fdroid.debug D/DownloaderService: Queued download of -2055468903/https://f-droid.org/repo/taco.scoop_34.apk using https://f-droid
 .org/repo/taco.scoop_34.apk
2022-09-14 17:58:28.793 11840-14262/org.fdroid.fdroid.debug D/ServiceHandler: >>>>> Dispatching to Handler (org.fdroid.fdroid.net.DownloaderService$ServiceHandler) {5384e6a}
 null: -2055468903
2022-09-14 17:58:28.793 11840-14262/org.fdroid.fdroid.debug D/ServiceHandler: Handling download message with ID of -2055468903
2022-09-14 17:58:28.797 11840-14262/org.fdroid.fdroid.debug D/DownloaderFactory: Using suffix /taco.scoop_34.apk with mirrors [Mirror(baseUrl=https://f-droid.org/repo,
 location=null, isIpfsGateway=false), Mirror(baseUrl=https://fdroid.tetaneutral.net/fdroid/repo, location=null, isIpfsGateway=false), Mirror(baseUrl=https://ftp.fau
 .de/fdroid/repo, location=null, isIpfsGateway=false), Mirror(baseUrl=https://ftp.lysator.liu.se/pub/fdroid/repo, location=null, isIpfsGateway=false), Mirror
 (baseUrl=https://mirror.cyberbits.eu/fdroid/repo, location=null, isIpfsGateway=false), Mirror(baseUrl=https://plug-mirror.rcac.purdue.edu/fdroid/repo, location=null,
 isIpfsGateway=false)]
2022-09-14 17:58:28.804 11840-11840/org.fdroid.fdroid.debug D/AppUpdateStatusManager: Update APK /taco.scoop_34.apk state to Downloading
2022-09-14 17:58:28.808 11840-14262/org.fdroid.fdroid.debug I/o*.f*.d*.HttpManager: HEAD https://ftp.fau.de/fdroid/repo/taco.scoop_34.apk
2022-09-14 17:58:29.193 11840-11840/org.fdroid.fdroid.debug D/ViewRootImpl[AppDetailsActivity]: changeCanvasOpacity: opaque=false
2022-09-14 17:58:29.528 11840-14262/org.fdroid.fdroid.debug I/o*.f*.d*.HttpDownloader: downloading /taco.scoop_34.apk (is resumable: false)
2022-09-14 17:58:29.532 11840-14262/org.fdroid.fdroid.debug I/o*.f*.d*.HttpManager: GET https://ftp.fau.de/fdroid/repo/taco.scoop_34.apk
2022-09-14 17:58:30.975 11840-11852/org.fdroid.fdroid.debug I/zygote: Background concurrent copying GC freed 50290(2MB) AllocSpace objects, 5(196KB) LOS objects, 50% free,
```

Note: This also applies to the production application version (org.fdroid.fdroid ver. 1.15.2).

Impact:

Logging sensitive information in the Android log is not a recommended practice as this information can (in some scenarios) be accessed by other applications on the same device. URLs can contain tokens or other sensitive data which might be logged, leading to the disclosure of that data to other apps.

Recommendation:

- Avoid logging any sensitive information like usernames, passwords, URLs, tokens etc in the Android log.

5 Non-Findings

In this section we list some of the things that were tried but turned out to be dead ends.

5.1 NF-007 — Review of Nearby Swap Feature

During the pentest, we analysed the Nearby Swap feature of the application closely. This feature allows people to share installed apps with others, even without internet access (but both the parties must have the F-droid app). The F-droid app enables user to start a local web server, and host a page with links providing direct download of selected apps.

This feature was tested for multiple cases like exploiting the web server to access local device files, tricking the users into sharing additional apps, the download process, etc. No security issues were discovered during the test.

5.2 NF-006 — Review of Deployment Scripts

The deployment scripts at `https://gitlab.com/fdroid/fdroid-http-fronters/` were reviewed for security issues; none were discovered.

6 Future Work

- **Retest of findings**
  When mitigations for the vulnerabilities described in this report have been deployed, a repeat test should be performed to ensure that they are effective and have not introduced other security problems.
- **Regular security assessments**
  Security is an ongoing process and not a product, so we advise undertaking regular security assessments and penetration tests, ideally prior to every major release or every quarter.

7 Conclusion

We discovered 2 Elevated and 3 Low-severity issues during this penetration test.

The main purpose of this pentest was to determine the security issues in deployment scripts and the F-droid Android client app. No major security issues were discovered during the deployment script review, in the time available. However, the F-droid client app has some weaknesses that, when resolved, would give the application better defences against attacks, and would correspondingly improve protection of the app's users.

We recommend fixing all of the issues found and then performing a retest in order to ensure that mitigations are effective and that no new vulnerabilities have been introduced.

Finally, we want to emphasize that security is a process – this penetration test is just a one-time snapshot. Security posture must be continuously evaluated and improved. Regular audits and ongoing improvements are essential in order to maintain control of your corporate information security. We hope that this pentest report (and the detailed explanations of our findings) will contribute meaningfully towards that end.

Please don't hesitate to let us know if you have any further questions, or need further clarification on anything in this report.

Appendix 1 Testing team

Abhinav Mishra	Abhinav has 10+ years of experience in the penetration testing of web, mobile and infrastructure. He has received numerous accolades from multiple organisations for responsible disclosure of vulnerabilities. He is also known for providing trainings on web, mobile and infrastructure security.
Melanie Rieback	Melanie Rieback is a former Asst. Prof. of Computer Science from the VU, who is also the co-founder/CEO of Radically Open Security.