

HANDBUCH DEMONSTRATOR ZUR BEREITSTELLUNG VON SMARTPHONE-FEATURES AM DESKTOP

Version 1.0 vom 10.10.2020
Bernd Prünster – bernd.pruenster@a-sit.at

Inhaltsverzeichnis

Inhaltsverzeichnis	1
1. Einleitung	1
2. Systemvoraussetzungen	1
3. Installation	1
3.1. Kompilieren von Source Code	2
3.2. Bereitgestelltes Paket	2
3.2.1. Android-Applikation	2
3.2.2. Bibliothek	2
4. Inbetriebnahme	2
4.1. Konfiguration	3
4.2. Starten	3
4.3. Testen	3
5. Handhabung	3
6. Lizenz	6

1. Einleitung

Der vorliegende Demonstrator veranschaulicht, wie moderne Android-Smartphones am Desktop als Trusted-Computing-Basis eingebunden werden können, um so trusted IO, trusted Remote Procedure Calls, sowie biometrische Authentifizierung am Desktop umzusetzen. Entsprechend besteht das bereitgestellte Paket aus einer Desktop- und einer Android-Komponente, wobei es sich bei der Android-Komponente um eine lauffähige Applikation handelt, die Desktop-Komponente jedoch vordergründig als Bibliothek (welche auch die Basis der Android-Applikation bildet) umgesetzt wurde. Diese wurde zu Demonstrationszwecken um eine rudimentäre Main-Funktion erweitert.

2. Systemvoraussetzungen

Die Implementierung wurde in *Kotlin* umgesetzt und verwendet *Apache Maven* als Buildsystem für die Bibliothek/Desktop-Applikation, sowie *Gradle* als Buildsystem der Android-App. Konkret werden Apache Maven 3.5, Java 8 und Android Version 9 (Pie) vorausgesetzt. Als Entwicklungsumgebung für die Android-Applikation (zur Durchsicht des Codes, sowie für Änderungen, etc...) wird mindestens *Android Studio 4.2 Canary 13* vorausgesetzt.

3. Installation

Da es sich beim bereitgestellten Desktop-Paket um eine Programmbibliothek handelt, welche auch

als Basis der Smartphone-App dient, ist eine Installation ins lokale Maven-Repository per `mvn install` nötig.

Die Installation der Smartphone-Applikation, welche als apk bereitgestellt wird, kann per `adb install` erfolgen.

3.1. Kompilieren von Source Code

Die Kompilation der Bibliothek erfolgt über Apache Maven durch Ausführen von `mvn compile`. Im Falle der Smartphone-Applikation per `./gradlew assemble`.

3.2. Bereitgestelltes Paket

Das bereitgestellte Paket ist in Kotlin umgesetzt und dient zu Demonstrationszwecken. Die Android-Applikation liegt im Quellcode, sowie als vorkompiliertes apk im Ordner *android* vor. Die Bibliothek liegt im Ordner *desktop* vor.

3.2.1. Android-Applikation

Die Android-Applikation besteht lediglich aus drei relevanten Quellcode-Dateien:

- `at.tugraz.iaik.asit.p2p.sp.companion.KeyStoreGlue`, welche den Zugriff auf den Android-Keystore kapselt
- `at.tugraz.iaik.asit.p2p.sp.companion.Core`, welche den Zustand der Applikation speichert, sowie den Zugriff auf relevante Konfigurationsparameter kapselt
- `at.tugraz.iaik.asit.p2p.sp.companion.MainActivity`, welche das Benutzerinterface, sowie die Businesslogik beinhaltet.

Logging wurde am Smartphone auf Grund der Inkompatibilität des (von einer für die Funktionalität essentiellen Bibliothek verwendeten) Logging-Frameworks mittels `println`-Aufrufen umgesetzt. Abgesehen von der Einbindung von Biometrie und der Anbindung des hardwarebasierten Keystore in der Smartphone-Applikation ist die Businesslogik des Demonstrators in der Bibliothek gekapselt.

3.2.2. Bibliothek

Die Bibliothek basiert auf `libp2p` und erweitert diese um drei Protokolle um: Remote-Attestation, Biometrische-Authentifizierung und Trusted-RPC. Diese sind entsprechend im Package `at.tugraz.iaik.asit.smartphonefeatures.proto` organisiert und an die Beispielimplementierungen von Protokollen des `jvm-libp2p`-Pakets angelehnt.

Der Kern der Implementierung findet sich im Package `at.tugraz.iaik.asit.smartphonefeatures.node`. Die Klasse `Cephalopod` dient dabei als Basis für die Implementierung von Desktop-, sowie Smartphone-Instanz, wobei am Smartphone lediglich Bindung an den Hardware-Keystore und entsprechende Signaturoperationen umgesetzt wurden. Wesentlich mehr Businesslogik findet sich in der Klasse `PCNode`, welche Attestation-Resultate entgegennimmt, biometrische Authentifizierung anfordern kann, sowie Trusted-RPC anstoßen kann.

Eine rudimentäre Main-Funktion ist in der Datei `at/tugraz/iaik/asit/smartphonefeatures/Main.kt` umgesetzt.

4. Inbetriebnahme

Da die Desktop-Applikation lediglich aus einer rudimentären Main-Funktion im Rahmen der Bibliothek besteht, ist eine alleinige Inbetriebnahme nicht angedacht. Stattdessen ist das Modul darauf ausgelegt, in eine IDE importiert zu werden, um Programmflüsse nachzuvollziehen. Die Geräte auf denen Desktop-Instanz und Smartphone-Instanz ausgeführt werden, müssen Teil einer gemeinsamen Kollisionsdomäne sein, da lediglich auf lokalem Multicast-DNS basierte Kommunikation implementiert wurde.

4.1. Konfiguration

Am Desktop muss die Datei `att.cfg` existieren, welche Package, der Android-Applikation, sowie deren Signatur und einen Schwellwert für ein Android-Patch-Level für den Remote-Attestation-Prozess festlegt. Eine Beispielkonfiguration liegt vor. Die Smartphone-Applikation sieht keinerlei Konfigurationsmöglichkeiten vor.

4.2. Starten

Die Smartphone-App erfordert zusätzlich die Applikation *Barcode Scanner*¹. Darüber hinaus muss über die Android-Sicherheitseinstellungen ein Fingerabdruck am Smartphone registriert sein. Am Desktop ist die Main-Funktion auszuführen (siehe Abschnitt 3.2.2)

4.3. Testen

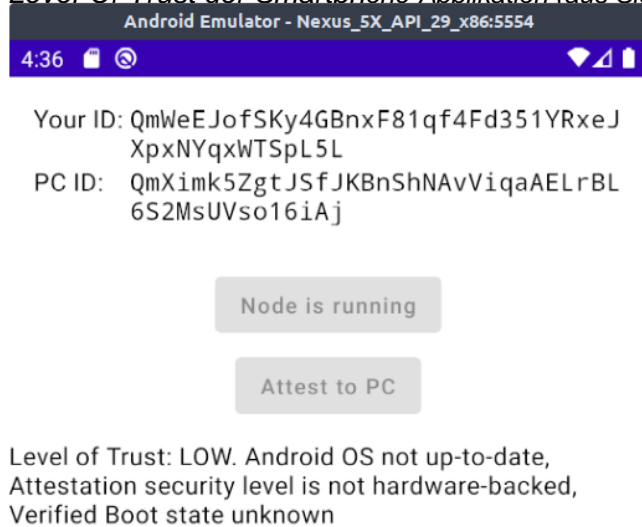
Es wurden im Rahmen des Demonstrators keine Tests umgesetzt.

5. Handhabung

Sind Desktop-Instanz und Smartphone-Applikation gestartet, wird am Desktop ein QR-Code angezeigt. Über „Bind to PC“ kann dieser in der Smartphone-App abgescannt werden. Daraufhin wird ein hardwaregebundenes Schlüsselpaar generiert, welches die Basis für den libp2p-Identifikator am Smartphone bildet. Nach erfolgreichem Scan kann mit „Start Node“ ein libp2p-Knoten am Smartphone gestartet werden. Anschließend erfolgt vollautomatisch eine Verbindung zwischen PC und Smartphone, was durch die Deaktivierung des „Start Node“-Buttons und die Änderung des Labels in „Node is running“ signalisiert wird. Anschließend kann über „Attest to PC“ die Attestierung gestartet werden.

¹<https://play.google.com/store/apps/details?id=com.google.zxing.client.android>

Nach erfolgreicher Attestierung wird der Level-Of-Trust der Smartphone-Applikation (aus Sicht der



Desktop-Instanz) angezeigt (siehe

Abbildung 1).

Die rudimentäre Main-Funktion protokolliert diese Abläufe auf der Kommandozeile. Eine grafische Benutzeroberfläche wurde nicht umgesetzt. Ein Zeilenumbruch (auf der Kommandozeile) startet den biometrischen Authentifizierungsprozess.

Ein weiterer Zeilenumbruch startet einen trusted RPC-Aufruf.

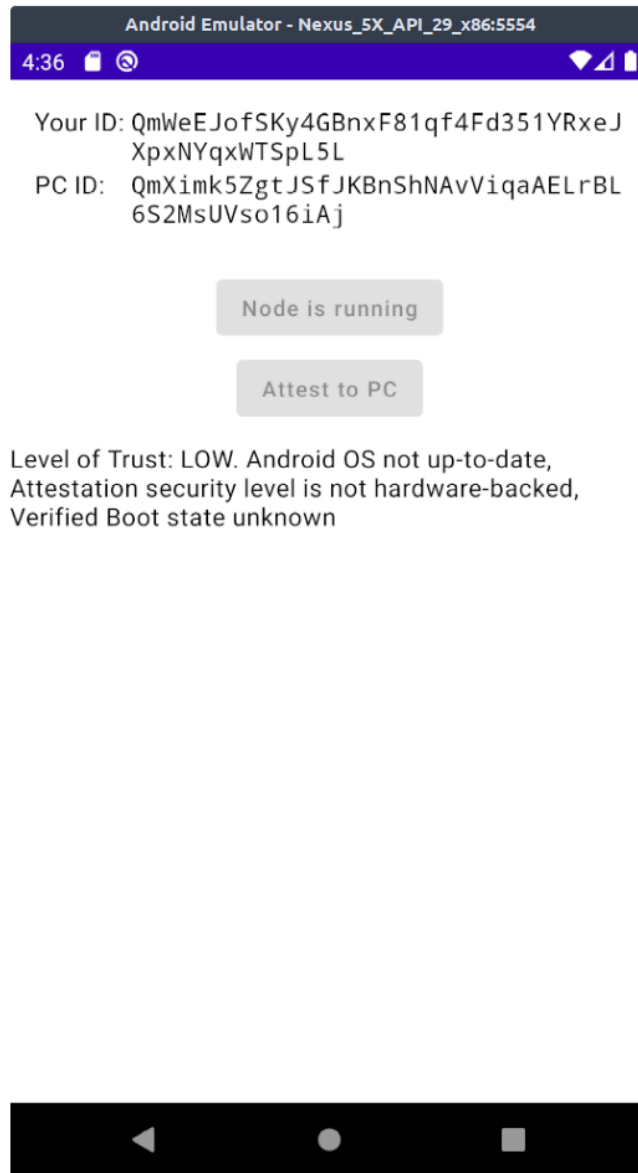


Abbildung 1: Statusbildschirm der Android-Applikation nach erfolgreicher Bindung und Attestierung

6. Lizenz

Das bereitgestellte Projekt wird unter den Bedingungen der Open-Source-Lizenz für die Europäische Union (EUPL) V1.1 bereitgestellt (siehe Lizenz.pdf).