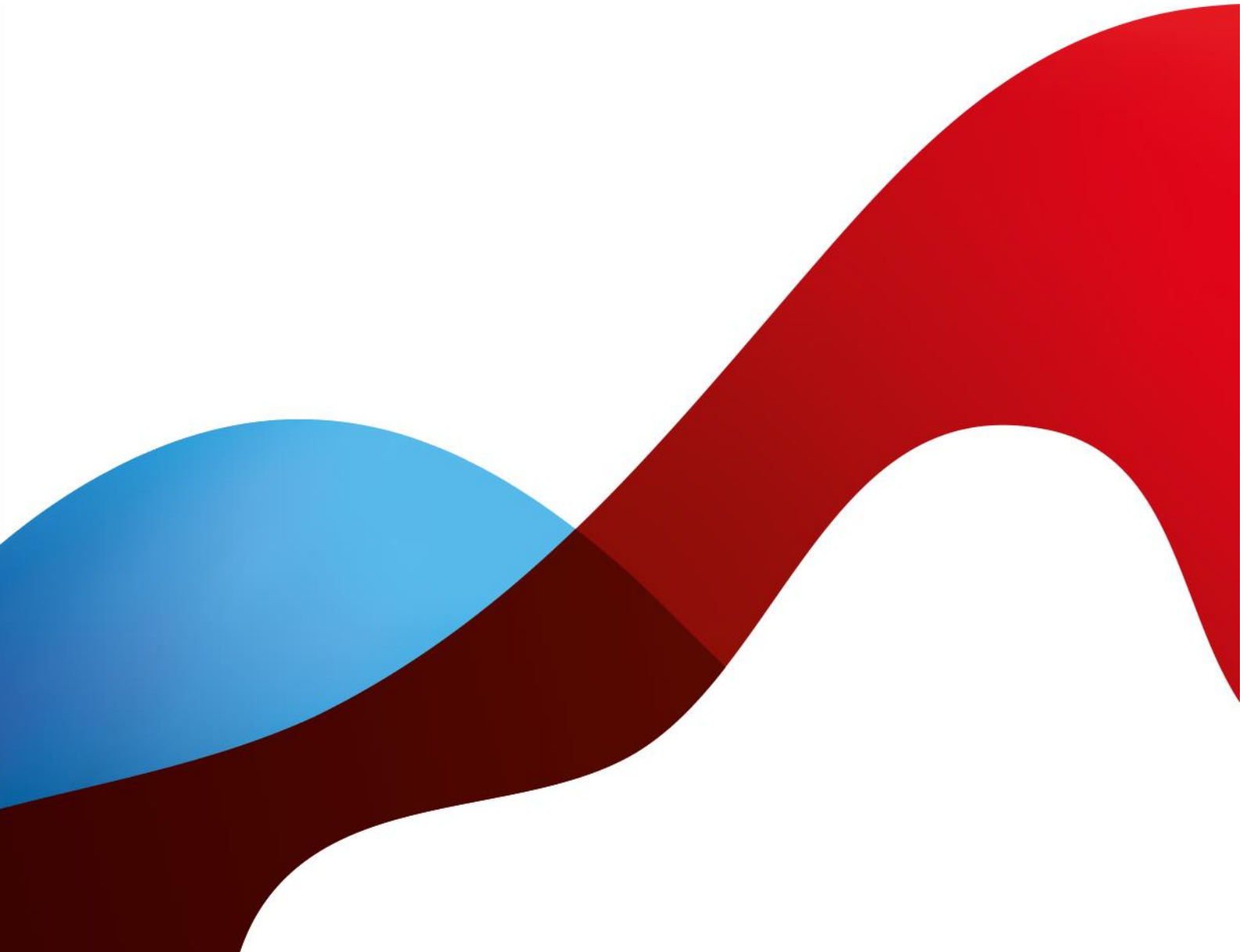
# Mobile ID Technical Guide

## New Swisscom Certificate Authority



# Table of contents




Mobile ID is a brand of Swisscom.
Swisscom (Switzerland) Ltd
Alte Tiefenaustrasse 6
CH-3050 Bern

# 1 Mobile ID API Change

## 1.1 Introduction

Within the next 12 months, it is mandatory that you take the following change into account: The signature response returned by Mobile ID contains a certificate chain that confirms the authenticity of the message. Various so-called X.509 certificates are used (currently `Swisscom Root CA 2` and `Swisscom Rubin CA 3`).

These certificates must be replaced regularly to achieve the required security level. Therefore, Swisscom must use new certificates.

Please ensure that your technical Mobile ID operation responsible persons receive this information and have *added* the new certificates to the platform or application (trust store) by **30th June 2022**.

The following chapters provide a technical description of the necessary changes and available test options.

## 1.2 Overview

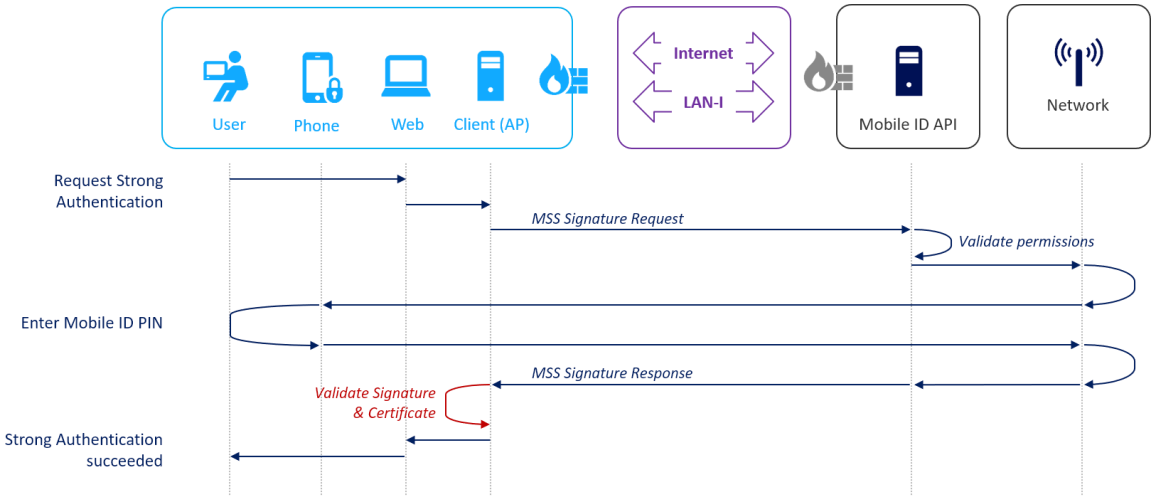
Swisscom attaches great importance to security standards. For that reason, Swisscom will go live with a new certificate chain of the Mobile ID User certificates. The new certificate authorities meet the latest security requirements and will improve the long-term security of the Mobile ID service.

From **30 June 2022** onwards, Mobile ID users who invoke a new Mobile ID Activation will use a new certificate chain that is built on certificates named `Swisscom Root CA 4` (Root Certificate) and `Swisscom Rubin CA 4` (Intermediate Certificate).

Please ensure that your Mobile ID client's TrustStore contains both the old Root Certificate `Swisscom Root CA 2` as well as the new Root Certificate `Swisscom Root CA 4` by the aforementioned date at the latest.

It is important that you still keep the old certificate(s) in the TrustStore because all Mobile ID users that are already active will keep using the old certificates until it is about to expire in late 2024 (or until they invoke a new activation).

Note, this change only affects the Mobile ID Signature[1] flow. In particular the validation of the MSS Signature Response (as depicted in the figure below, in red).




Any successful Signature Response contains a base64 encoded **CMS signature** and the **mobile user's X509 certificate**, including the corresponding intermediate- and root-certificates.

The mobile user's X509 certificate contains the public key of the mobile user, which is required if an Application Provider wants to validate the CMS signature (in red). However, the validation of the signature is actually an optional step and in the sole responsibility of the Mobile ID Customer (Application Provider).

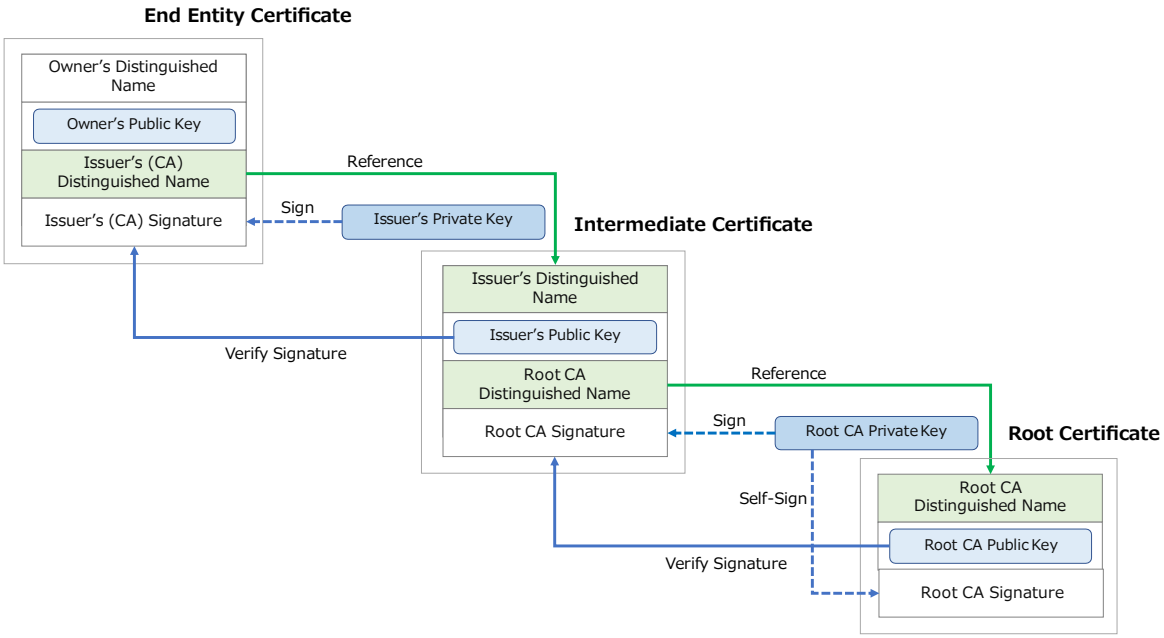
The change does not affect the certificates used for the mutual TLS/SSL authentication between the Mobile ID Customer (Application Provider) and the Mobile ID Server (API).

[1] MSS Signature is documented in the Mobile ID Client Reference Guide

## 1.3 End user's X509 certificate chain

The figure below depicts the Mobile ID Certificate Chain. The User Certificate (End Entity Certificate) is issued by the Intermediate Certificate. The Intermediate Certificate is issues by the Root Certificate.

Usually, a client's TrustStore contains the Root Certificate only, the so-called trust anchor. Having the root certificate in the client's TrustStore, the whole certificate chain can be validated, including the Mobile ID user's end entity certificate.



### 1.3.1 Before the Swisscom API change

Today, the mobile user's X509 certificate chain is built as follows:

| Certificate | Field | Value |
|---|---|---|
| **Root** | Subject | **CN = Swisscom Root CA 2**, OU = Digital Certificate Services, O = Swisscom, C = ch |
| | Issuer | CN = Swisscom Root CA 2, OU = Digital Certificate Services, O = Swisscom, C = ch |
| | Serial number | 1e9e28e848f2e5efc37c4a1e5a1867b6 |
| | Signature Alg. | sha256RSA |
| | Valid Until | 25. June 2031 08:38:14 |
| | Public Key Type | RSA 4096 Bits |
| **Intermediate** | Subject | C = ch, O = Swisscom, OU = Digital Certificate Services, **CN = Swisscom Rubin CA 3** |
| | Issuer | CN = Swisscom Root CA 2, OU = Digital Certificate Services, O = Swisscom, C = ch |
| | Serial number | 40a768fcfd9da89f4370711242e8b941 |
| | Signature Alg. | sha256RSA |
| | Valid Until | 22. September 2024 10:39:14 |
| | Public Key Type | RSA 2048 Bits |
| **End Entity** | Subject | CN = MIDCHE<*unique token*>:PN, SERIALNUMBER = MIDCHE<*unique token*> |
| | Issuer | C = ch, O = Swisscom, OU = Digital Certificate Services, CN = Swisscom Rubin CA 3 |
| | Serial number | *( unique for each mobile user certificate )* |
| | Signature Alg. | sha256RSA |
| | Valid Until | *( 3 years after creation )* |
| | Public Key Type | RSA 2048 Bits *or* ECC 256 Bits |

### 1.3.2 After the Swisscom API change

From 30 June 2022 onwards, a mobile user's X509 certificate chain is built as follows:

| Certificate | Field | Value |
|---|---|---|
| **Root** | Subject | C = CH, O = Swisscom, 2.5.4.97 = VATCH-CHE-101.654.423, OU = Digital Certificate Services, **CN = Swisscom Root CA 4** |
| | Issuer | C = CH, O = Swisscom, 2.5.4.97 = VATCH-CHE-101.654.423, OU = Digital Certificate Services, CN = Swisscom Root CA 4 |
| | Serial number | 149e3a785f82f16b8afb21aad5aff747 |
| | Signature Alg. | RSASSA-PSS |
| | Valid Until | 24. November 2038 11:21:53 |
| | Public Key Type | RSA 8192 Bits |
| **Intermediate** | Subject | C = ch, O = Swisscom (Schweiz) AG, 2.5.4.97 = VATCH-CHE-101.654.423, OU = Digital Certificate Services, **CN = Swisscom Rubin CA 4** |
| | Issuer | C = CH, O = Swisscom, 2.5.4.97 = VATCH-CHE-101.654.423, OU = Digital Certificate Services, CN = Swisscom Root CA 4 |
| | Serial number | 110f53b876f5431f0342bfd4f704467a |
| | Signature Alg. | RSASSA-PSS |
| | Valid Until | 28. January 2029 13:23:38 |
| | Public Key Type | RSA 4096 Bits |
| **End Entity** | Subject | CN = MIDCHE<*unique token*>:PN, SERIALNUMBER = MIDCHE<*unique token*> |
| | Issuer | C = ch, O = Swisscom (Schweiz) AG, 2.5.4.97 = VATCH-CHE-101.654.423, OU = Digital Certificate Services, CN = Swisscom Rubin CA 4 |
| | Serial number | *( unique for each mobile user certificate )* |
| | Signature Alg. | RSASSA-PSS |
| | Valid Until | *( 3 years after creation )* |
| | Public Key Type | RSA 2048 Bits *or* ECC 256 Bits |

#### 1.3.2.1 Root CA certificate "Swisscom Root CA 4"

Download cert at http://aia.swissdigicert.ch/sdcs-root4.crt

Cert Fingerprint: `b9 82 1b 0c 87 7d 30 24 dd 6a 8f 6e 44 3e f5 38 8e 53 16 1b`

#### 1.3.2.2 Intermediate CA certificate "Swisscom Rubin CA 4"

Download cert at http://aia.swissdigicert.ch/sdcs-rubin4.crt

Cert Fingerprint: `42 a9 87 df 7b 25 a9 3f 2d a9 a0 8b f9 e0 24 cc 4f 98 31 5a`

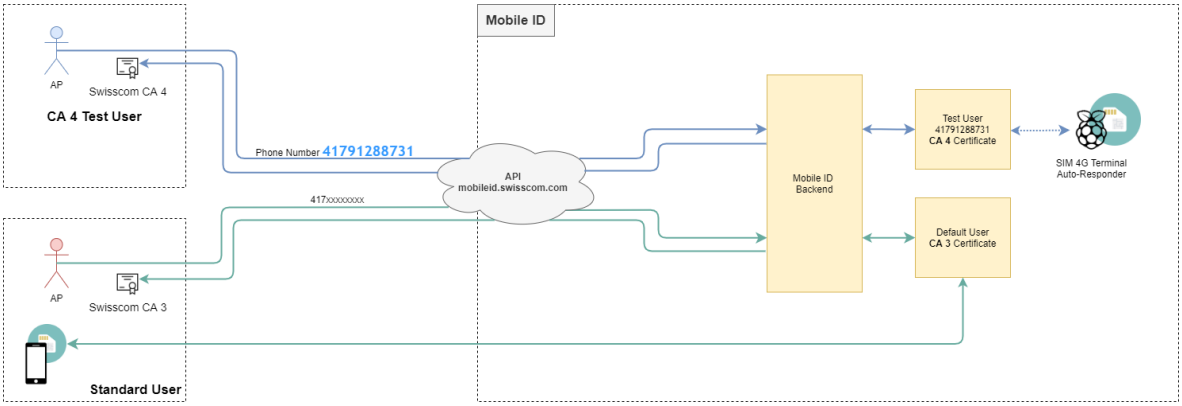
## 1.4 What the Mobile ID customer needs to do

In most cases the only change on the Mobile ID customer side is to ensure that the new `Swisscom Root CA 4` certificate has been added to the Mobile ID Client TrustStore by **30 June 2022** at the latest.

The old certificate `Swisscom Root CA 2` must still be kept in the TrustStore.

### 1.4.1 Test Option

Mobile ID customers can already test the new certificate today. Here's how testing works.




A Mobile ID customer can send an MSS Signature Request using a specific test user with MSISDN +41791288731. In this case, the Mobile ID API will always respond with a successful MSS Signature Response, that contains the new certificate chain.

Any Mobile ID customer should verify if their client will properly support the new certificate by doing such a test call. This test option is already live and Mobile ID customers can already start to test.

Be aware that this test option invokes real end-to-end Mobile ID authentications using a real Mobile ID SIM card. In the unlikely event of two Mobile ID clients trying to perform a test at the same time, one of the Mobile ID clients may get a `406 / PB_SIGNATURE_PROCESS` fault response. In this case, the client should simply try again a few Minutes later.

## 1.5 Known Issues

One of the relevant changes between the old and new certificates is the signature algorithm. The SHA256RSA signature algorithm has been replaced by the RSASSA-PSS signature algorithm. There is a known issue with old Java runtimes that do not support RSASSA-PSS.

Support for RSASSA-PSS signature algorithm has been added with **JDK11b15**, **JDK8u251** and **OpenJDK8u252**.

Please refer to https://bugs.java.com/bugdatabase/view_bug.do?bug_id=JDK-8146293