Dokumentation

Implementation of a cryptographic Binding for a Client-TLS Authentication

Client TLS Authentication on Smartphones

DI Kevin Theuermann BSc, MSc – kevin.theuerman@egiz.gv.at

Summary: This document describes the requirements and details for the implementation of a Client-TLS- authentication on smartphones. The architecture and components that are necessary are provided and explained.

E-Government Innovationszentrum

Inffeldgasse 16/a, A-8010 Graz
Tel. +43 316 873 5514
Fax. +43 316 873 5520
E-Mail post@egiz.gv.at
www.egiz.gv.at

Das E-Government Innovationszentrum ist eine gemeinsame Einrichtung des Bundeskanzleramtes und der TU-Graz

Table of Contents

1 General Information

The concept of authenticating a person via secret knowledge (e.g. password, PIN), has been established and improved by different ideas for choosing safe passwords. The human memory reaches its limits regarding the various passwords and PINs that have to be remembered - which are all different in the ideal case – and furthermore the risk of a dictionary attack by means of faster computing technology arises, which causes the requirement of an even greater length of passwords/PINs. For this reason, it is necessary to use a medium of greater and more consistent capacity for the secure storage of identifiers.

1.1 X.509 Certificate

Certificates are digital proofs of specific properties of subjects, which are certified by the issuer of the certificate. Only the integrity that means the correctness of the certificate in the sense of non-modification and the authenticity (of the originator, if its identity has already been identified) can be validated. X.509 certificates whose functionality is ensured by means of a public-key infrastructure provide the de-facto standard in today's world of communication.

1.2 Certificate Authority

A certificate authority is an entity that issues digital certificates. A digital certificate certifies the ownership of a public key by the named subject of the certificate. This allows others (relying parties) to rely upon signatures or on assertions made about the private key that corresponds to the certified public key.

1.3 Certificate Signing Request

A Certificate Signing Request (CSR) or Certification Request is a digital request to create a digital certificate using a digital signature from a public key. The request is generated with the applicant's private key and consists of a public key, a distinguished name, and optional attributes.

1.4 Service Provider

Service providers provide content or technical services required for the use or operation of content and services on the Internet.

1.5 Identity Provider

An identity provider is a system entity that creates, maintains, and manages identity information for principals while providing authentication services to relying party applications within a federation or distributed network.

1.6 Client TLS Authentication

Traditionally, TLS Client Authentication has been considered the alternative to bearer tokens (passwords and cookies) for web authentication. In TLS Client Authentication, the client (browser) uses a certificate to authenticate itself during the TLS handshake. Once the TLS connection is established (and authenticated), the client and server run HTTP on top of the TLS layer.

2 Implementation Steps

For the practical exercise, a root Certificate Authority (CA), which issues certificates to intermediate CAs, which can in turn provide users with desired client certificates, is simulated. A Certificate Signing Request (CSR) has to be sent to this CA, in order to obtain a valid X.509 certificate for further authentication purposes.

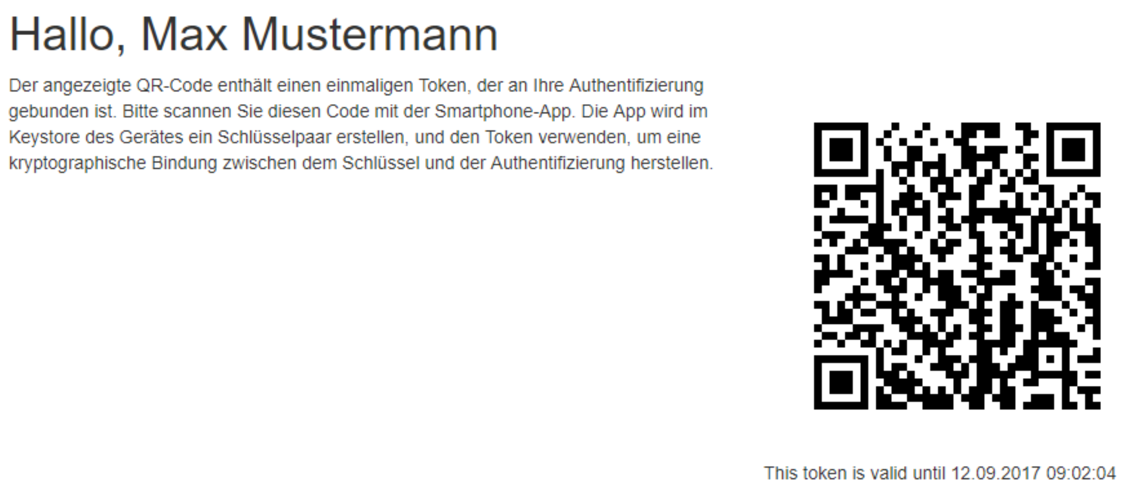
The aim of the practical exercise is to develop a smartphone app, which is able to authenticate a user via a client certificate at a sample service provider. In order to do this, a client certificate has to be obtained at a Certificate Authority and used for a client TLS authentication. The student can choose whether to program an iOS or Android smartphone app.

This chapter specifies the generic solution of a software architecture and its components for an agile, mobile authentication, which has to be implemented within the practical exercise. The following two processes have to be established:

2.1 Creation of the cryptographic binding:

When the authentication app is first used, there is no cryptographic binding, which is needed for the logon process at the underlying example-service. In order to establish a cryptographic binding, the user logs in via the application at the Token Service. For this login, the citizen card/mobile phone signature can be used. For those who do not have a citizen card/mobile phone signature a sample user (Max Mustermann) will be provided already logged in. When the login was done successfully, a QR-Code, containing a one-time session token that is needed for the cryptographical binding as well as the URL that points to the binding service, will be displayed on the desktop. (**Important**: As this as a one-time session token, you have to reload the website in order to generate a new token, if you want to start a new session/request. Otherwise, the token will be invalid. Furthermore, the token is only valid a few minutes). The token service can be accessed via the following link:

https://apps.egiz.gv.at/tokenservice/token


Hallo, Max Mustermann

Der angezeigte QR-Code enthält einen einmaligen Token, der an Ihre Authentifizierung gebunden ist. Bitte scannen Sie diesen Code mit der Smartphone-App. Die App wird im Keystore des Gerätes ein Schlüsselpaar erstellen, und den Token verwenden, um eine kryptographische Bindung zwischen dem Schlüssel und der Authentifizierung herstellen.

This token is valid until 12.09.2017 09:02:04

Figure 1: Token service for the cryptographic binding

When the one-time session token was sent to the crypto-binding URL, certain cryptographic information is returned to the client, which is necessary for the key generation. Next, the cryptographic key pair (public/private key) will be generated considering the requirements included in the cryptographic information and stored in the keystore. Afterwards, a Certificate Signing Request (CSR) including the generated public key will be transmitted to the binding service. The binding service validates the CSR and returns a valid cryptographic X.509 certificate, which can be used for further client TLS authentication purposes.

Authentication via Client TLS:

The requested cryptographic certificate should be used for a Client-TLS authentication. For this purpose, an example service will be provided. This service should be accessed via a webview within the smartphone application. On a successful authentication, the example service will be displayed.

3 Development Details

The following paragraph summarizes all details that have to be considered for developing the smartphone application.

Operating System: iOS or Android
Developing Environment: XCode, Android Studio … whatever you prefer

3.1 Details regarding the operating systems

- **iOS:** On iOS, apps can access the same key store when they are signed by the same developer. Because the system apps (e.g., browsers, etc.) have been signed by Apple, access to other apps is not possible. For iOS, the term KeyChain is used for these app-specific key memories.

- **Android:** In Android, there are two different key memories - a global one called Android KeyChain, and an app-specific one called Android KeyStore. All apps can access the Android KeyChain. The initial use of a key must be confirmed by the user. In addition, a differentiation is made for different user profiles / areas (business / private). It is not possible to access apps from one area to the KeyStore of the other area.

3.2 Development process in detail

Premise: The login at the Token Service (https://apps.egiz.gv.at/tokenservice/token) was successful and a QR code is displayed.

1.) Scan QR Code displayed on the desktop to get a one-time session token and the URL, which points to the binding service endpoint. The URL has to be extended as follows:
URL for requesting necessary cryptographic information:
a.) URL extracted from QR Code + "/binding/info"

URL to Binding Service (service for obtaining the cryptographic certificate):
b.) URL extracted from QR Code + "/binding/signV2"

2.) Send HTTP GET request including the session token and a pre-defined app-Id ("token=... appId=**firstname.surname**[1] ") to the binding service endpoint (1a), in

[1] Please use your firstname and your surname.

order to request the cryptographic information. Important: If you have a second firstname the appId is as follows: “**firstname1.firstname2.surname**” The appId is a string parameter, which is necessary for identifying an application in the binding process. The appId must be written as follows for every individual student: “**firstname.surname**”.

3.) Parse the cryptographic information, which is returned by the binding service (Note: The cryptographic information can differ between some students). You also have to save the X-Auth-Token given in the response header of the server. This token serves as a session parameter.

4.) Generate a cryptographic key-pair (public/private key) and store it into the keychain/keystore. There is a parameter listed in the cryptographic information called subject. You will need the information given by the subject for creating the CSR (CommonName, DistinguishedName, etc. …) Only set the parameters which are included in the subject parameter.

5.) Create Certificate Signing Request (CSR) using meaningful parameters.

6.) Send CSR via HTTP post request to the binding service (1b). The CSR have to be send as a body parameter “csrEncoded= …” The CSR data should be Base64 URL encoded. The appID must also be provided as a body parameter: “appId= …” In the header, you have to transmit the “X-Auth-Token” which was sent to you in the response after transmitting the QR-Code information.

7.) After successful server-validation of the CSR, the server will return the cryptographic certificate and additional information (“eidblob”) which you can ignore. Store the issued cryptographic certificate into the keychain of the smartphone for future authentication purposes. Try to find a way to store this information in a secure way.

8.) Register/Use the certificate for a client TLS authentication for an example service provider (https://apps.egiz.gv.at/sslclientcertdemo/), which will finally return a list of further service providers. If you receive this list, the client authentication was performed successfully.

(The minimum criteria ends here!)

4 Grading Criteria and Minimal Requirements

The minimal criteria in order to get a positive grade (4 "Sufficient") is defined by developing a smartphone application, that is able to perform a cryptographic binding with the given server endpoint and authenticates against an example service provider via client TLS. In detail that means that development step 1-8 have to be implemented. Starting by scanning the QR-code displayed by the Token Service to the receipt of the cryptographic X.509 certificate issued by the binding service and providing it for an authentication at a service provider.

<u>In order to get a better grade</u>, you can extend the functionality regarding the data security. If you want to use your client certificate for establishing a client TLS connection, the access to this certificate can be protected. You can require an authentication method like entering a PIN/password for enabling the use of the certificate. This can further improve the security measures implemented on the client side.

Next, you can implement a user interface similar to Figure 2.

To learn about the mobile key management you should implement a user interface that enables the entering of any desired string. Afterwards, it should be possible to sign this string with you generated private key or to encrypt it using your public key.

The result of the chosen operation should be displayed. If the string was signed, a hash value should be the result, if it was encrypted, any string should be displayed.

In the next step, it should be possible to decrypt the encrypted string again, using your private key in your KeyStore or Keychain.

If the string was signed, you should decrypt the signed value with the public key, generate the hash value of the string entered at the beginning (save the string for this reason), and compare the result of the hash-value with the decrypted signed value. They should be the same. If so, the verification is successful.

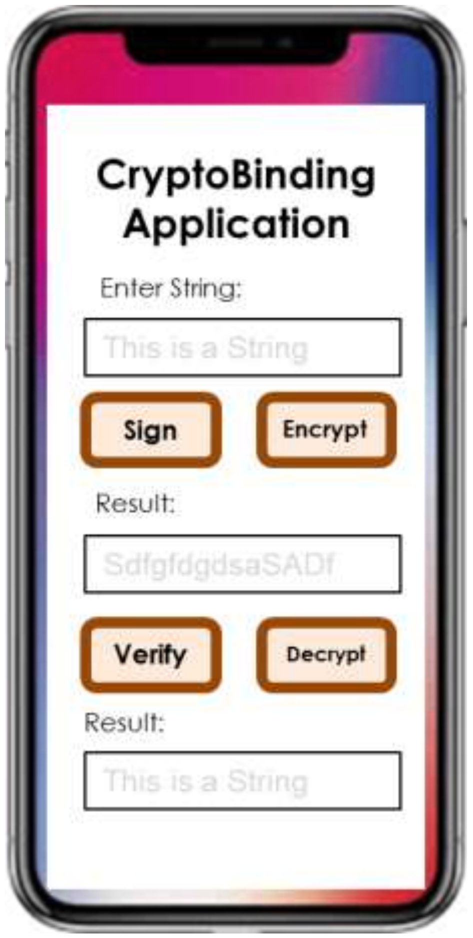


Figure 2: CryptoBinding Interface

Further important requirements/information:

- Individual work (do not use the code of another student)
- Send the application to kevin.theuermann@egiz.gv.at until the 10.01.2020
- Compulsory meetings for discussing the assignments ("Abgabegespräche") will be announced during the semester
- For important questions, contact me directly: kevin.theuermann@egiz.gv.at