31/05/19 Versione 0.1

Aggiornamento e prima versione

02/09/19 Versione 0.2

Aggiunta parte relativa alla decodifica dei file con AES256

Semplice guida/SDK per lo sviluppo di un client del Servizio Lambda.

La verifica dell'integrità e l'utilizzo dei dati contenuti in Lambda Seal, avviene attualmente tramite l'uso di Universal QReader™ una App per smartphone/tablet, liberamente scaricabile da PlayStore(Android) ed AppStore(IOS).

Di seguito un breve documento che descrive le diverse fasi che le due applicazioni svolgono per mostrare a video il documento PDF (PADES)

Fase di acquisizione e decodifica del QRCode presente nel documento PDF

Per iniziare l'operazione di verifica e recupero del documento PDF (PADES) digitale, presente in maniera codificata sul Lambda Store, è necessario partire dall'interpretazione dei dati presenti sul Lambda Seal (QRcode) . A questo scopo possono essere utilizzare diverse librerie per l'acquisizione via telecamera del device mobile dei dati QRCode stampanti. Di seguito alcuni connsigli su esempi e librerie con licenza Open Source:

iOS:

- https://github.com/YannickL/QRCodeReaderViewController (Object C)
- https://github.com/yannickl/QRCodeReader.swift (Swift)

Android:

- https://developers.google.com/vision/android/barcodes-overview
- https://medium.com/cashify-engineering/barcode-reader-using-google-mobile-vision-88b3e9f31668
- http://www.stdioh.in/blog/61/qr-code-scanner-in-android-using-google-mobile-vision-api-with-example

Tramite questi sistemi il QRCode verrà convertito in una sequenza CSV da interpretare come l'esempio seguente:

> MEBKM:TITLE:Per la verifica del contrassegno elettronico e l'accesso al documento firmato, utilizzare l'App gratuita Universal QReader;URL:https://www.gt50.org/IT/down_info_uqr/;;SQ:0306;51098876E57C06773621E8E1BACB0D17B56AB683C438BB6BBA4701256070D49B;TPz+ACursskhvxGIT/tM2t9nRHbLjvIkUljKpE4F4EY=;1554209172-SCLSGF65C15A794D-dlgAgMz.pdf;20190402145129;

Per un'analisi dei singoli elementi si consulti il documento:

https://services-aruba.gt50.org/documentation/SLSSqCode0306.pdf

Fase di verifica stato del documento e download dal gateway Lambda Service

Per prelevare il file binario codificato dal gateway è necessario effettuare due diverse chiamate GET in HTTS partendo da alcuni dei dati decodificacati dal QRCode. Nello specifico sia lo SHA256 (Es. 51098876E57C06773621E8E1BACB0D17B56AB683C438BB6BBA4701256070D49B) del documento, che la sua data di inserimento saranno (Es. 20190402145129), insime, la chiave univoca per la ricerca del file sul Lambda Store.

E' compito del client gestire tutte le chiamate verso il server in HTTPS.

La libreria di Networking per iOS ed Android è a completa scelta dello sviluppatore:

- https://github.com/Alamofire/Alamofire (iOS)
- http://square.github.io/okhttp/ (Android)

Le operazioni da svolgere saranno:

1. Richiesta HTTPS GET verso REST API "getdocumentinfo". Questa chiamata REST restituisce in formato JSON tutti i dati sullo stato del documento, sul suo eventuale firmatario, sui dati (opzionali) di inserimento nella blockchain FACTOM.
2. Qualora il documento NON sia revocato o cancellato allora sarà disponibile al download tramite la chiamata REST "downloadfile" che restituirà lo stream binario codificato presente in archivio.
    1. Qualora invece il documento sia in stato di revoca o cancellato verrà riportato il messaggio da riportare a video con la motivazione dellìoperazione. L'operazione si conclude in questo caso immediatamente dato che un'eventuale richiesta REST tramite la chiamata "downloadinfo" restituirebbe un errore di file NON presente.
3. I dati di firma inviati sono in formato BASE64 e codificati con lo SHA256 della chiave presente nel QRCode.

Per le diverse chiamate verso il server REST si veda il documento:

https://services-aruba.gt50.org/documentation/ListaChiamateRESTServerLambdav1.15.pdf

Fase di decodifica del documento e sua visualizzazione da Mobile iOS, Android

Una volta scaricato localmente il file binario sarà necessario decodificarlo e mostrarlo a video.

La sequenza di decodifica si avvale dell'apposita libreria libqid/crypto sviluppata da GT50 ed appositamente compilata per piattaforme Android ed iOS. La libreria si avvale poi di un Wrapper (Object C) e (Java JNI) per poter poi essere utilizzata all'interno del codice dell'App.

La sequenza di decodifica è la seguente:

1. DecodeBase64 della chiave presente nel QRCode (Campo Chiave)
2. DecodeAES256 dello stream binario scaricato con la chiave del passo precedente
3. Salvataggio del file per la visualizzazione

Per la visualizzazione si consiglia di utilizzare per Android un Intent verso Acrobat Reader od altro visualizzatore presente nel sistema Mobile, per iOS il file verrà presentato a video tramite il visualizzatore interno di UIKit.

Note sulla decodifica del file cifrato con AES256

I file presenti nel Lambda Store sono codificati con algoritmo AES256 in questo modo:

- generazione di una chiave AES256 random (K)
- generazione di 16 byte random (R16)

(blocco casuale aggiunto)

- vettore di inizializzazione (IV):

  16 byte posti a 0 (quindi "inefficace")

Al contenuto originale si prepende R16 e si cifra con K e IV:

`AES-256( <R16><contenuto>, K, <IV nullo> )`

Esempio esplicativo usando OpenSSL:

1 - DECODIFICARE LA CHIAVE DA BASE64 A BINARIO

(LA CHIAVE È UN CAMPO DEL CONTENUTO DEL QR CODE in Base64)

openssl enc -d -base64 -in key.b64 -out key.dat -A

2 - CODIFICARE LA CHIAVE IN CARATTERI ESADECIMALI

`<?php $kkk = bin2hex(file_get_contents("key.dat")); file_put_contents("key.hex", $kkk); ?>`

3 - OPERARE LA DECIFRAZIONE

openssl enc -aes256 -d -in f.aes256 -K $(cat key.hex) -iv 00000000000000000000000000000000 >f.orig

4 - SCARTARE I PRIMI 16 BYTE DEL RISULTATO FINALE