Spécification QRCode signés

Historique

Version	Auteur(s)	Commentaire	Date
0.1	bdauvergne	cadrage	10/06/2022
0.2	mates	relecture	10/06/2022

Aperçu

Cette spécification définit un encodage simplet et sécurisé d'un certificat électronique dans un QRCode. Elle est fortement inspirée du schéma européen pour les certificats sanitaires[1]. Les points particuliers :

- un contenu encodé en utilisant CBOR[2] (Consise Binary Object Representation) contenant :
  - un numéro de série (entier, séquence)
  - un uuid
  - une date de création ("Issued At")
  - un émetteur
  - des données libres (= JSON)
- le contenu est ensuite signé électronique via la bibliothèque NaCL[3] et l'algorithme sur courbe elliptique Ed25529 donnant des clés publiques et privées courtes de 32 octets,
- le contenu signé est encodé en base45[4] pour des raisons d'économie d'espace et de compatibilité avec l'alphabet basic des QRCodes (voir la RFC draft pour les raisons techniques),
- un préfixe EO0: est ajouté pour identifier le format.

[1] https://github.com/ehn-dcc-development/eu-dcc-hcert-spec/blob/main/hcert_spec.md

[2] https://www.rfc-editor.org/rfc/rfc8949.html

[3] https://en.wikipedia.org/wiki/NaCl_%28software%29

[4] https://datatracker.ietf.org/doc/draft-faltstrom-base45/

Sécurité

Le contenu des QRCodes est signé avec un algorithme de signature à clé publique basé sur les courbes elliptiques, utilisant la courbe Ed25519 et l'algorithme de la bibliothèque NaCl[2] (état de l'art), protégeant l'intégrité du certificat.

Schéma des données

- un tableau a 5 éléments :
  - numéro de série : un entier sur 64 bits
  - identifiant unique: une chaîne d'octets de longueur 16 (uuid encodé en big-endian)
  - date/heure d'émission: un timestamp UNIX sur 64 bits
  - émetteur: une chaîne UTF-8
  - données libre : un dictionnaire CBOR

Exemple

En pseudo JSON :

```
[1, "99c6875c467e402b884ce3918ef482a7", "2022-06-10T10:00:34.34343Z", "AMP", {"immat": "AZ1234ZH",
 "deb": "2022-06-10", "fin": "2023-06-10"}]
```

Résultat après sérialisation :

```
print(cbor2.dumps([1, uuid.UUID('99c6875c467e402b884ce3918ef482a7').bytes, datetime.datetime.
utcnow(), 'AMP', {'immat': 'AZ1234ZH', 'deb': datetime.date(2022, 6, 10), 'fin': datetime.date(
```

```
2023, 6, 10)}], timezone=datetime.timezone.utc, datetime_as_timestamp=True, date_as_datetime=True
))
b'\x85\x01P\x99\xc6\x87\\F~@+\x88L\xe3\x91\x8e\xf4\x82\xa7\xc1\xfbA\xd8\xa8\xcb_\x8a\xc5\xfccAMP
\xa3eimmathAZ1234ZHcdeb\xc1\x1ab\xa2\x89\x80cfin\xc1\x1ad\x83\xbd\x00'
```

Contenu:

- numéro de série : 1
- identifiant unique : 99c6875c467e402b884ce3918ef482a7
- date/heure d'émission : le 10 juin 202 à 10h et 34 secondes
- émetteur : AMP
- données libres :
  - immatriculation : AZ1234ZH
  - date de prise d'effet de la dérogation : le 10 juin 2022
  - date de fin de la dérogation : le 10 juin 2023

Signature électronique

Se référer à une implémentation quelconque du schéma de signature à clé publique de NaCl, comme libsodium.

Initialement, la paire (une secrète, une publique) de clés de l'émetteur est générée via crypto_sign_keypair() et stockée. La clé publique est diffusée et configurée dans le lecteur de QRCode.

L'encodage CBOR est ensuite signé avec la fonction crypto_sign() par la clé privée de l'émetteur.

Pour la lecture, le contenu signé est vérifié via crypto_sign_open() et la clé publique de l'émetteur.

Encodage en Base45

L'encodage Base45 convertit une chaîne d'octets libre en une chaîne limitée à l'alphabet 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ·$%*+-./: (donc plus longue). C'est un encodage analogue à Base64 mais adapté au fonctionnement des QRCodes.

Encodage dans le QRCode

Inspiré du paragraphe équivalent dans la spécification hcert[5].

```
In order to better handle legacy equipment designed to operate on ASCII payloads, the compressed C
WT is encoded as ASCII using Base45 before being encoded into a 2D barcode.

The QR format as defined in (ISO/IEC 18004:2015) SHALL be used for 2D barcode generation. An error
 correction rate of ‘Q’ (around 25%) is RECOMMENDED. The Alphanumeric (Mode 2/QR Code symbols 0010
) MUST be used in conjunction with Base45.

In order for Verifiers to be able to detect the type of data encoded and to select the proper deco
ding and processing scheme, the base45 encoded data (as per this specification) SHALL be prefixed
by the Context Identifier string "HC1:". Future versions of this specification that impact backwar
ds-compatibilty SHALL define a new Context Identifier, whereas the character following "HC" SHALL
be taken from the character set [1-9A-Z]. The order of increments is defined to be in that order,
i.e., first [1-9] and then [A-Z].
```

Ici l'encodage sera Base45, préfixé de la chaîne EO0:@.

[5] https://github.com/ehn-dcc-development/eu-dcc-hcert-spec/blob/main/hcert_spec.md#422-qr-2d-barcode