Work in progress

This is an early draft of a chapter of a book on the foundations of cryptographic authentication being coauthored by Francisco Corella, Sukhi Chuhan and Veronica Wojnas. Please send comments to the authors.

15. Verifiable credentials and self-sovereign identity

15.1 Origin, scope, and "verifiability" of verifiable credentials

"Verifiable credentials" are cryptographic credentials being standardized by the World-Wide-Web consortium (W3C)[1], with a different origin and a much wider scope than the traditional credentials of Chapter 3.

The origin of verifiable credentials can be traced back to the early nineteen seventies, when the amount of data stored in computers was growing and methods were needed for organizing the contents of databases in ways that would be easier to comprehend by humans. This resulted in several "data models", the most successful of which was the relational model[2], better known today as the Structured Query Language (SQL) model, where data is arranged in tables. There was also an "entity-relationship model"[3], where data was arranged in triples, each consisting of two entities connected by a named relationship.

After Tim Berners-Lee invented the World-Wide Web, he envisioned using it as a universal distributed database, with nodes communicating using HTTP. He referred to this vision as the "Semantic Web"[4], and he proposed a data model based on triples to organize the data, which he called the "Resource Description Framework (RDF)"[5]. Like the entity-relationship model, the RDF is a conceptual model that can be depicted on paper as a graph.

Verifiable credentials started out as collections of "verifiable claims", which are subject-property-value triples that can be connected into information graphs, in accordance with the RDF framework. Examples of such claims can be found in Figures 3 and 4 of the Verifiable Credentials (VC) Data Model v1.1[1], and examples of information graphs in Figures 4, 6, and 8. The data represented in an information graph is called Linked Data, and an information graph can be "serialized" into a textual description written in a language such as JSON-LD, where LD stands for Linked Data.

By contrast with the traditional cryptographic credentials of Chapter 3, the primary purpose of a verifiable credential is not to authenticate a party at one end of a connection to a party at the other end of the connection, but rather to certify a state of affairs in the physical world. Hence, they have a much wider scope than traditional credentials. Verifiable credentials are being used, have been proposed, or could be used for the following purposes:

- Certifying the kinds of plastics that make up a particular plastic recyclate[6].

- Certifying the supply-chain components that have been used in a product being shipped across an international border[7].
- Certifying the emission reduction methodology, verification status, and ownership of a carbon credit[8].
- Certifying that two people are married or legal domestic partners.
- Certifying that an adult is the guardian of a child.

Certifying that a widowed person is the legal representative of a deceased person.
Verifiable credentials make it possible to provide the above certifications in the form of a cryptographic signature applied to a state of affairs described by a serialization of an information graph. They are "verifiable" in the sense that they can be verified cryptographically, by contrast with the various non-cryptographic methods that are traditionally used to verify such certifications.

The use cases where the above certifications would be performed do not require authentication of the subjects of the credentials. In fact, in use cases where the subject is a person, the subject may not have to be online when the certification is used, and in use cases where the subject is an object, that object may not have to be connected to the internet and may not even have a means of performing computations.

The topic of this book, however, is cryptographic authentication, so this chapter will focus on uses cases where a verifiable credential is used to authenticate its subject.

15.2 Using verifiable credentials for authentication

As indicated by its name, the VC Data Model is only a data model, and as such it does not specify an authentication protocol. This is made clear in the preamble of Section A:

> "While this specification does not provide conformance criteria for the process of the validation of verifiable credentials or verifiable presentations, readers might be curious about how the information in this data model is expected to be utilized by verifiers during the process of validation. This section captures a selection of conversations held by the Working Group related to the expected usage of the data fields in this specification by verifiers."

Furthermore, the VC Data Model does not normatively specify how a credential refers to its subject. A verifiable credential must have a "credentialSubject*"* property, which may have an "id" property. But, according to section A.1:

> "The id property is optional. Verifiers could use other properties in a verifiable credential to uniquely identify a subject."

However, one possible method of using a verifiable credential for authentication can be gleaned from the "Concrete Lifecycle Example" of Section 3.4., where "Pat receives an alumni verifiable credential from a university, and Pat stores the verifiable credential in a digital wallet." Pat's verifiable credential is a JSON object shown in EXAMPLE 1, which appears to be a serialization in JSON-LD syntax of an information graph similar to the one in

Figure 6 (with slightly different data). The credential object comprises a "credentialSubject" property, which itself comprises an "id" property and an "alumniOf" property. The value of the "id" property is a decentralized identifier (DID), like those discussed in Chapter 14. The value of the "alumniOf" property is a claim that the credential asserts about the subject. Its value is an object with properties specifying a DID denoting the university and the name of the university in two languages.

The credential object also comprises a "proof" property having a "jws" property that contains the signature of the issuer on the credential. The name of the "jws" property suggests that its value is a JSON Web Signature (JWS)[9]. However, a JWS is a JSON Web Token (JWT)[10] comprising a header, a payload, and a signature, separated by periods. Including the credential as the payload in the "jws" property would be redundant, and the value of the "jws" property in the example is clearly too short to include the credential. The presence of two consecutive periods in the value of the property suggests that the payload has been omitted.

Continuing the lifecycle example, Pat attempts to use the credential to redeem an alumni discount on season tickets to sports events. To that purpose, using a mobile device, "Pat starts the process of purchasing a season ticket. A step in this process requests an alumni verifiable credential, and this request is routed to Pat's digital wallet. The digital wallet asks Pat if they would like to provide a previously issued verifiable credential. Pat selects the alumni verifiable credential, which is then composed into a verifiable presentation. The verifiable presentation is sent to the verifier and verified."

The verifiable presentation used by Pat is the JSON object shown in EXAMPLE 2. It comprises Pat's verifiable credential, and a "proof" property comprising a "digital signature by Pat on the presentation", in the form of a "jws" property with a presumably omitted payload. The signature can be verified using the verification method "did:example:ebfeb1f712ebc6f1c276e12ec21#keys-1", which is a public key contained in the document of the DID "did:example:ebfeb1f712ebc6f1c276e12ec21". As seen in the verifiable credential included in the verifiable presentation, this DID is the decentralized identifier of the credential subject. Verification methods are discussed in Chapter 14, Section 14.2.1.1.

The "proof" property of the verifiable presentation also comprises "challenge" and "domain" properties. In traditional cryptographic authentication, a credential presentation is a protocol executed by a prover and a verifier over a communication channel. But since verifiable credentials are specified by a data model, a "verifiable presentation" is just data, it is not a protocol. However, a challenge is sent by a verifier, and the "challenge" property could be used to implement the following challenge-response authentication protocol between the credential subject (Pat in the example) and a verifier, conducted over a secure connection:

1. The verifier sends a challenge string.
2. The subject (Pat), or more precisely the subject's agent (Pat's wallet), composes the verifiable presentation as follows:
    a. It assigns the credential as the value of "verifiableCredential" property.

   b. It assigns the challenge string as the value of the "challenge" property of the presentation proof.
   c. It resolves the DID of the credential subject and assigns an authentication method found in the DID document as the value of the "verificationMethod" property of the presentation proof.
   d. It constructs a JWS object with a payload comprising a portion of the presentation being constructed, not including the jws property.
   e. It constructs a serialization of the JWS object, omitting the payload.
   f. It assigns the serialization to the "jws" property of the proof.
3. The subject sends the verifiable presentation to the verifier.
4. The verifier obtains the credential being presented as the value of the "verifiableCredential" property of the presentation.
5. The verifier verifies that the challenge that it sent to the party at the other end of the secure connection is the value of the "challenge" property of the presentation proof.
6. The verifier reconstructs the JWS, adding the omitted payload.
7. The verifier resolves the DID that is the value of the "id" property of the "credentialSubject" property of the credential included in the presentation, obtaining the DID document.
8. The verifier finds the public key referenced by the "verificationMethod" property of the presentation proof in the DID document and verifies that it is included as an authentication method in the document.
9. The verifier uses the public key to verify the signature in the JWS.

The above protocol demonstrates to the verifier that the party at the other end of the connection is the subject of the credential included in the presentation, based on the following two pieces of evidence:

1. The payload of the JWS includes the challenge string.
2. The signature in the JWS can be verified with a public key that is included as an authentication method in the document of the DID of the subject of the credential.

The comment in EXAMPLE 2 refers to the presentation proof as a "digital signature by Pat on the presentation". Since the presentation includes the credential, this implies that the signature covers the credential, and suggests that the omitted payload in the "jws" property includes an encoding of the credential. It should be noted, however, that the fact that the presentation signature covers the credential is not part of the evidence that authenticates the subject. Therefore the presentation could be shortened by omitting the credential, which includes a long signature, from the jws payload.

The "domain" property is discussed in the next section.

15.3 Protection against man-in-the-middle attacks

Neither the "domain" property of the presentation proof shown in EXAMPLE 2, nor its value "4jt78h47fh47", are defined or discussed anywhere in the VC data model. However, it seems plausible to assume that they refer to the intended verifier of the presentation. If so, the

"domain" property could be used to implement a countermeasure against man-in-the-middle phishing attacks.

The challenge-response protocol of the previous section authenticates the prover to the verifier as the subject of the credential, but only if the prover and the verifier exchange messages over a connection that is *secure* as defined in Chapter 4. If the messages are relayed by a man-in-the-middle (MITM) attacker, the verifier will authenticate the party at the other end of the connection as the subject, but that party will be the attacker.

In Chapter 4 we defined a MITM *phishing* attack against user authentication as an attack where the user agent is tricked into sending its messages to an address controlled by the attacker rather than to the relying party (RP). The domain property of the presentation proof provides the following countermeasure against such an attack. The user agent (e.g. Pat's wallet in the example) always uses the address where it is going to send its messages as the value of the domain property. If it has been tricked into sending its messages to the attacker, the value of the domain property is the attacker's address. If the attacker relays the presentation as it receives it, the RP detects the attack because the value of the domain property is not its own address. If the attacker replaces the value of the domain property with the address of the RP, the RP detects the attack because the verification of the presentation signature fails.

Section 8.4 of the VC Data Model recognizes that authentication with a verifiable presentation is vulnerable to a MITM attack, and proposes to use token binding as a countermeasure:

> "A verifier might need to ensure it is the intended recipient of a verifiable presentation and not the target of a man-in-the-middle attack. Approaches such as token binding [RFC8471], which ties the request for a verifiable presentation to the response, can secure the protocol."

But as explained in Section 1 of RFC 8471[11], the token binding protocol is designed to provide protection against the exfiltration of a bearer token such as a cookie by binding the token to a TLS connection over which it is sent. It is not designed to provide protection against a MITM attack, and there does not seem to be any way to leverage token binding for protection against a MITM attack. In a successful MITM phishing attack against user authentication to a web site, all messages are relayed by the attacker, so there is no direct TLS connection between the user agent and the web site to which a token could be bound.

15.4 Self-sovereign identity

15.4.1 Definition

In the physical world, a self-sovereign identity is a name that you have freely chosen yourself, rather than one that has been chosen for you. Having a self-sovereign identity is a human right that has not always been recognized but is now being asserted. In Canada, indigenous people have recently been able to replace the western-style names given to them by missionaries with their indigenous names in official documents[12].

A name is a persistent label with which diverse information about the person can be associated. Therefore, in the digital world, and more specifically in the context of cryptographic authentication, a self-sovereign identity (SSI) can be defined as a persistent identifier created by your user agent rather than a centralized registration authority, with which multiple cryptographic credentials can associate claims or attributes.

In the SSI literature, a self-sovereign identity is a W3C decentralized identifier, best known by the DID acronym, and the credentials that associate claims with it are W3C verifiable credentials. But these are not the only choices. It is possible to meet the above definition of self-sovereign identity with traditional identifiers and credentials rather than DIDs and VCs.

15.4.2 SSI with existing technology

In first approximation, a verifiable credential that asserts claims about a subject is functionally equivalent to a public key certificate (such as for example an X.509 certificate[13]) that asserts attributes about the same subject, claims and attributes being different names for the same concept. The only difference is that the VC binds its claims to a DID, while the public key certificate binds its attributes to a public key. And this difference seems minimal when considering that a DID constructed by the "did:key" method, is essentially a public key.

The public key in a public key certificate is a component of a key pair that is generated at random by the user agent. Does it qualify as a self-sovereign identity? No, because it is not persistent. It is generated specifically for that certificate and not meant to be used in any other certificate.

But there is actually no cryptographic reason why a public key cannot be used in multiple certificates. It has not been done before because there has been little motivation to do so. Prior to the advent of the semantic web, the resource description framework and linked data there were few use cases where different issuers would issue credentials to the same subject on the web at large. There were use cases in the enterprise, but a different solution was found to handle those cases: instead of issuing multiple public key certificates to an employee, a single certificate would bind a public key to the employee number, then multiple attribute certificates[14] would bind sets of attributes with different expiration dates, to the employee number.

A public key used in multiple public key certificates qualifies as a self-sovereign identity.

There is another reason why a public key has not been used before as an identifier in multiple public key certificates. A public key is a very large number randomly generated as part of a key pair, which does not look like an identifier. As we saw in Section 1 of Chapter 14, identifiers of human subjects have traditionally been human-meaningful and memorizable. This changed when Bitcoin used public keys, then hashes of public keys, as payment addresses. DIDs are not memorizable, and a did:key DID contains an encoding of a public in the method-specific portion of the identifier.

Using a public key in multiple public key certificates is a case of reverse technology transfer from a newer technology (VCs) to an older one (public key certificates). Another one is using a hash of the public key instead of the public key itself in a public key certificate, or in multiple public key certificates. (To authenticate, the subject supplies the public key in addition to the signature on the challenge.) This has remarkable benefits. Besides shortening the certificate, it protects the public key against post quantum attack. (A hash of the pubic key is not used in did:key, because the public key is needed to derive the DID document from the DID; but it is used in did:peer and KERI.)

15.4.3 Benefits of SSI

As we saw in Chapter 1, there are two kinds of authentication: two-party authentication as a returning visitor, and third-party authentication as a new visitor. Different technologies are used in these two kinds of authentication, and SSI is primarily concerned with the latter.

Today, most third-party authentication is performed using federated identity[15] as discussed in Chapter 5, using OpenID Connect or proprietary protocols. Federated identity has serious downsides for the user. The identity provider is involved in each authentication transaction and can thus observe the user's activities without having to do any tracking; it must be online when authentication takes place and its users may not be able function on the internet when it is not online; it could censor its users' activities; and it controls the identities by which its users are known on the internet. Avoiding these downsides has been a major motivating force towards self-sovereign identity.

Involvement of an identity provider at authentication time can be avoided by using a cryptographic credential for authentication, and all the above downsides can be eliminated by binding the claims or attributes in the credential to a decentralized identifier. A verifiable credential binds its claims to a DID, which is a persistent decentralized identifier, and hence qualifies as a self-sovereign identity. A public key certificate binds its attributes to a public key, which is also a decentralized identifier but may or may not be persistent, as discussed above in Section 15.4.2. Thus, both a verifiable credential and a public key certificate can be used to eliminate the federated identity downsides, and eliminating the downsides is a benefit of SSI; but those downsides can also be eliminated using an ordinary public key certificate, where the public key is not persistent and hence does not qualify as a self-sovereign identity.

A question that comes to mind is whether SSI provides some additional, exclusive, benefit. The answer is yes. Using a persistent identifier, it is possible to bind the claims or attributes of two different credentials to the same subject identifier, and thus provide evidence that the same subject has been issued both credentials. This might be useful, for example, if issuance of a new credential would be dependent on the subject having both existing credentials.

The evidence if not ironclad, however, since two people could cheat and share their persistent identifier by sharing its private key. Credential consistency (i.e. prevention of credential sharing) could be enforced by using a biometric as a second factor for authentication of a person as the subject of an identifier.

We saw above in Section 15.4.2 that substantial benefits can be obtained by replacing the public key in a public key certificate with a hash of the public key. The hash of the public key is also a persistent decentralized identifier that qualifies as a self-sovereign identity and can provide the same benefits as a DID or a persistent public key.

15.4.4 Privacy implications of SSI

The use of the same subject identifier in multiple credentials provides an additional tracking mechanism besides any tracking that might be available based on claims or attributes in the credential. For that reason, SSI should not be used in some kinds of credentials. For example it should not be used in employee credentials if that would allow linking credential use in the enterprise and outside of it.

---

[1] Verifiable Credentials Data Model v1.1. W3C Recommendation 03 March 2022. https://www.w3.org/TR/vc-data-model/.

[2] Codd, E.F (1970). "A Relational Model of Data for Large Shared Data Banks". Communications of the ACM. Classics. 13 (6): 377–87.

[3] Chen, Peter (March 1976). "The Entity-Relationship Model - Toward a Unified View of Data". ACM Transactions on Database Systems. 1 (1): 9–36.

[4]Tim Berners-Lee. The Semantic Web. 2000 August 15. https://www.w3.org/2000/Talks/0906-xmlweb-tbl/text.htm

[5] RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation 25 February 2014. https://www.w3.org/TR/rdf11-concepts/

[6] Berg, Holger et al. Overcoming information asymmetry in the plastics value chain with digital product passports: How decentralised identifiers and verifiable credentials can enable a circular economy for plastics. Wuppertal Papers, No. 197. https://www.econstor.eu/bitstream/10419/251914/1/1798057344.pdf.

[7] The United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT). eDATA Verifiable Credentials for Cross Border Trade. https://unece.org/sites/default/files/2023-08/WhitePaper_VerifiableCredentials-CrossBorderTrade_September2022.pdf

[8] Shiv Aggarwal and Priya Guliani. Carbon Credit Issuance, Verification, Tracking, and Standardisation Through Various Means. https://t20ind.org/research/carbon-credit-issuance-verification-tracking-and-standardisation-through-various-means/

[9] M. Jones et al. JSON Web Signature (JWS). IETF RFC 7515. May 2015. https://datatracker.ietf.org/doc/html/rfc7515.

[10] M. Jones et al. JSON Web Token (JWT). IETF RFC 7519. May 2015. https://datatracker.ietf.org/doc/html/rfc7519.

[11] A Popov et al. The Token Binding Protocol Version 1.0. IETF RFC 8471. October 2018. https://datatracker.ietf.org/doc/html/rfc8471

[12] Government of Canada. Reclaiming Indigenous names on Immigration, Refugees and Citizenship Canada identity documents. https://www.canada.ca/en/immigration-refugees-citizenship/news/2021/06/reclaiming-indigenous-names-on-immigration-refugees-and-citizenship-canada-identity-documents.html

[13] D. Cooper et al. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. IETF RFC 5280. May 2008. https://datatracker.ietf.org/doc/html/rfc5280

[14] S. Farrell et al. An Internet Attribute Certificate Profile for Authorization. IETF RFC 5755. https://datatracker.ietf.org/doc/html/rfc5755.

[15] National Institute of Standards and Technology. Digital Identity Guidelines: Federation and Assertions. NIST SP 800-63C. June 2017 (includes updates as of 03-02-2020). https://csrc.nist.gov/pubs/sp/800/63/c/upd2/final