From: **Kaliya Identity Woman**
Date: Sun, Nov 22, 2020 at 10:52 AM
Subject: VCs & OCap - please talk
To: Manu Sporny, Samuel Smith

HI Sam and Manu,
I'm opening up a thread because I have heard via Drummond that Sam is thinking about a new VC to do access management.
I know this is an area where ZCaps are being used and I think it is really critical to have discussions across the community amongst the deep experts before 'just starting new things'.
I hope you two can talk sooner rather than later.
Warm Regards,
- Kaliya

From: **ProSapien Sam Smith**
Date: Sun, Nov 22, 2020 at 11:34 AM
Subject: Re: VCs & OCap - please talk
To: Kaliya Identity Woman Cc: Manu Sporny, Drummond Reed, Hardman Daniel

Kaliya,

The proposed task force is to look at VC for authorizations in general, not merely access management. Access management (aka ZCaps) is an important use case of authorization but a generic broad definition of authorization includes much more than access management. Object capabilities are a very useful model for access management and this is not an anti-zcap effort but a broader approach where an object capability semantic may be one of many. These include but are not limited to custodial relationships, guardianship relationships, business process management relationships, supply chain relationships etc. The important feature is that the semantics are embedded into existing VCs, i.e. VC Native, and not part of a separate conveyance as is the stated objective of https://w3c-ccg.github.io/zcap-ld/

Indeed the primary semantic we are looking at is not an authorization semantic per se but a chaining semantic for establishing chained VCs. A VC chaining semantic supports more than authorization/delegation it also supports provenance of data transformations, data custody source to sink, trust provenance (reputation), audibility, fine grained issuance semantics etc. In this regard authorization may be viewed as a sub-semantic of a chaining super semantic. This is especially useful in open loop systems (not closed loop like access control).

In summary, access control (aka z-caps) may best be implemented via a distinct process from VC chaining and hence benefit from a separate home and conveyance. In this regard, pure access control via Z-caps is not impinged upon by adding a general chaining semantic to VCs of which trust provenance and/or authorization are sub-classes.

See this:

https://github.com/hyperledger/aries-rfcs/blob/master/concepts/0104-chained-credentials/README.md

https://github.com/evernym/sgl

Sam

From: **ProSapien Sam Smith**
Date: Sun, Nov 22, 2020 at 1:33 PM
Subject: Re: VCs & OCap - please talk
To: Kaliya Identity Woman
Cc: Manu Sporny Drummond Reed, Hardman Daniel

Kaliya,

To give you a little more background.

There are two primary approaches to privacy protection in online exchanges of information.

1) Manage the degree of disclosure
2) Manage the degree of exploitation given disclosure

The first approach is inherently leaky. Over time any disclosure becomes more and more correlatable. It is best employed to provide temporary ephemeral privacy protection

The second approach imposes liability or counter incentive to the un-permissioned exploitation of correlated data. It is a persistent and enforceable check on exploitation that removes the incentive to correlate. This may be accomplished with contracts or with contracts in combination with regulations such as GDPR. A really good legal analysis of privacy protection using contracts is given here called "chain link confidentiality". https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2045818

Consent is a type of authorization. It is usually open-loop and persistent. It benefits from a chaining semantic. It is hard to shoe horn consent semantics into access control semantics. They are both types of authorizations in the broad sense of the word, but they do not work the same. Chain link confidentiality (or the like) authorization semantics could be implemented with self-contained VCs that employ a native chaining authorization semantic.

When sharing data via VCs it makes the most sense that any associated authorizations/consents/restrictions etc. be conveyed in-band, self-contained, attached, embedded, or entrained with that data as it is shared. Not conveyed in a separate out-of-band mechanism.

I am sure you are well acquainted with the Kantara Initiative's Consent Receipt spec. https://kantarainitiative.org/download/7902/

In addition to the links in my prior email, some other relevant links are as follows:

https://www.iso27001security.com/html/27560.html

https://wiki.idesg.org/wiki/index.php/Consent_to_Create_Binding

https://www.rfc-editor.org/rfc/rfc2693.txt

These all benefit from an embedded VC native chaining semantic for which a separate out of band Z-cap conveyance is not well suited.

A further example of a use for a chaining semantic. The IETF and TCG (trusted computing group) are promulgating several standards based on verifiable remote attestations of the configuration of trusted executions environments. https://datatracker.ietf.org/wg/rats/about/ https://trustedcomputinggroup.org/wp-content/uploads/TCG-NetEq-Attestation-Workflow-Outline_v1r9b_pubrev.pdf
The semantics of these verifiable attestations could be embedded in and conveyed by VCs with appropriate chaining and provenancing semantics

The goal here is to make VCs the universal locus of interoperability so that implementers that want Verifiable Containers of Data need use only one infrastructure. The VC spec with VC schema is sufficiently expressive syntactically to encompass all these semantics we just have to standardize the syntax for the semantics.

I hope this helps clarify the technical motivations and distinctions.

Regards

Sam

From: **Daniel Hardman**
Date: Wed, Dec 2, 2020 at 8:54 PM
Subject: Re: VCs & OCap - please talk
To: Kaliya Identity Woman
Cc: ProSapien Sam Smith, Dmitri Zagidulin, Tobias Looker , Manu Sporny, Drummond Reed

Kaliya, I think it's noble to look for a way to bring separate communities together, but I feel like there are some misunderstandings about the history and nature of our divergence. They may not doom such efforts to failure, but they do get in the way. What follows is my attempt to describe

the problem from my perspective. I don't claim it to be objective truth -- only an accurate capture of the way I see the situation.

VCs can be verified by anyone *without* going back to the issuer. I consider this one of their most important characteristics. It changes power dynamics, technical architecture, regulatory compliance, trust, and many human factors upon which SSI depends.

OCaps are generally validated/redeemed/invoked in a way that circles back to their issuer. The OS gives a file handle; an application that wants to work with the file makes an OS call and presents that handle as proof that they have the right to use it. I believe this direct or indirect dependency on the issuer is what Sam meant when he described OCaps as a "closed loop" system.

Because the use cases for VCs and OCaps sound a bit different when described like I have above, it is not crazy to implement them differently. But it is also not necessary. You can use a VC like a bearer token that should be presented to and verified by its issuer. You can add to such a VC various capabilities like attenuation chains, composite delegation, etc -- *with only schema choices, not changes to the VC spec*. In other words, VCs can prove delegated privilege just like OCaps can. (On the other hand, you can't address the breadth of VC use cases with the narrower featureset of OCaps.) Note how this claim I'm making is similar to ones I have heard Dave Longley and Manu make about other possible innovations proposed in the VC space: "We don't need `serviceEndpoint`; just emit a VC that makes your endpoint into verifiable data. We don't need `type` on a DID doc; use a VC to describe attributes of a DID subject."

When I have advocated for ideas like `serviceEndpoint` and `type`, Dave and Manu have sought to put the burden of proof on me: *Please demonstrate why you can't do what you want with what we already have*. Why have we not applied that same logic to the zCaps work item that CCG has undertaken? What can we not do with VCs that forces us to invent an entirely new spec for a kind of signed, verifiable, revocable data? (Note that I'm NOT asking why we want OCap-like behavior in a security context; I'm asking why there has to be a separate spec/impl for such a mechanism, when VCs -- the very work product this group is known for -- are capable of expressing all the same semantics.)

OCaps were emphasized at an RWOT that many of us attended (Boston, I think). Some OCap experts were there, and I loved what they had to say. OCaps are useful. However, when Manu and Chris W wanted to explore at that same conference an OCap-centric vision of how the authorization section of a DID Doc ought to work, I felt a lot of dissonance. So did others. We eventually agreed that we wouldn't bog down other dimensions of progress with a tough battle to consensus on the authorization topic. zCaps (originally OCAP-LD, IIRC) were born soon thereafter as a method that DB wanted to use for authorization in the DB ecosystem. I was fine with that, as a feature of their DID method. I read an early draft of the spec with some care to stay informed. Meanwhile, I began imagining how OCaps could be done with a VC and any DID method, giving them selective disclosure, strong privacy, powerful revocation, governance, regulatory compliance etc -- without inventing a new mechanism for signing and verification that would need DID method support and its own story about all of these supporting topics. The outgrowth of my work was first a webinar about delegated credentials, and later Aries RFC 0104

about chained credentials. I did not make an attempt to standardize my work, because it only depends on the VC standard which already exists. Everything else is just picking schemas cleverly. And my approach worked from day 1 with all variants of VCs -- JSON-LD, JWT, Indy's flawed impl, etc.

When DB proposed zCaps as a CCG work item, I did not object. I figured that any DID method that wanted to use zCaps should be able to. However, I had no interest in collaborating on what I felt was an unnecessary new mechanism that was missing important features VCs already had. I also felt that chained credentials addressed another use case that zCaps can't handle and that the general VC ecosystem sorely needed, which is general data provenance. So I was getting 2 for 1. (Provenance of authorization is a subset of a larger problem. Think of how academics and journalists cite sources, and imagine that data provenance mechanism allowing someone to attribute a `legalName` field in an employer credential to the government ID it came from. But I digress...)

Anyway, I have been consistent about my disinterest in zCaps for several years now. I spoke about my reasons (and about my alternate approach) at IIW (unfortunately not attended by zCaps proponents). I have pushed back on the way that `capabilityInvocation` enthrones an OCap worldview in the DID core. I've linked to the chained credential RFC in several CCG emails. When the CCG sponsored a special learning session about zCaps, I wrote the group to note that an alternate solution was possible, where OCaps were built from VCs. Joe engaged in a public debate with me about it on the mailing list. His arguments were the only substantive engagement I got. He invited me to come present to the group about it, but when the date didn't work, and then COVID happened, we never circled back.

My point in all this is not recrimination -- it's simply to acknowledge that Sam's task force is not a new, casual, or ill-considered divergence. It is an old one that stems from philosophical, architectural, and use case differences. These differences ARE NOT related to our other sore spot of divergence around ZKPs; they are entirely independent. The approach Sam is imagining is not a new effort to ignore a nascent zCaps standard; it is as old as OCAP-LD itself, and just as well documented. And if anything, it is Sam's approach that is standard; zCaps requires that a new standard be developed, instead of using the one we already matured and have worked so hard to implement.

None of this needs to prevent us from converging -- but the precondition to convergence would be an agreement on the requirements we're trying to address. I don't think we have that. I have modest interest in seeing if we could find common ground, and would be willing to attend a meeting to discuss. I would also be willing to revisit the offer from CCG to explain this alternate approach, if there is interest. But achieving a convergence here doesn't feel like a rational top priority -- only a nice-to-have. So I'm waiting for someone else to take the bull by the horns. If no one does, I'll continue to support Sam's task force because it is more closely aligned with my priorities and my view of which standard I want to focus on (the VC standard). I don't view divergence like this as a terrible tragedy; often it's better to standardize after the market has weighed in on what it wants. I feel like many in the CCG want to standardize too early.

--Daniel

From: **ProSapien Sam Smith** <sam@prosapien.com>
Date: Thu, Dec 3, 2020 at 5:55 PM
Subject: Re: VCs & OCap - please talk
To: Tobias Looker
Cc: Hardman Daniel, Kaliya Identity Woman, Dmitri Zagidulin, Manu Sporny, Drummond Reed

Correcting typos and better wording.

Should read:

Not authorization statements in general. But even in the realm of authorization statements, they do not even begin to encompass all the types of authorization statements indigenous to real world of data processing. Access control is only one thin slice of all authorizations.

From: **Tobias Looker**
Date: Thu, Dec 3, 2020 at 6:37 PM
Subject: Re: VCs & OCap - please talk
To: ProSapien Sam Smith
Cc: Hardman Daniel, Kaliya Identity Woman, Dmitri Zagidulin , Manu Sporny, Drummond Reed

Sam,

> I appreciate the ongoing discussion. It is helpful to see how various viewpoints and definitional understanding differ. I want to comment on one statement because I believe it is one of those that are pare of a core divergences in understanding.

Appreciate the feedback and discussion too. I think I see the source of our disagreement quite well. Essentially you view the verifiable credentials spec through a more abstract lense than I (which I totally understand), whereas I view verifiable credentials as a cryptographically secure identity claim assertion format, not one for describing authorization, because I see them having divergent requirements meaning they should not be conflated and technologies like OAuth and OpenID Connect are examples where they have maintained this separation (id_token vs access_token).

> The verifiable in verifiable credential means cryptographically verifying signatures. A more generic way of expressing this is that a verifiable credential is verifying the authenticity of a statement.
Really nothing more and nothing less. The meaning of that verified statement is open, its free. The schema and semantics of a VC have nothing to do with its primary purpose, that of verifying authenticity of a statement. The "identity" in this case is the holder of the private key from the

public private key pair. Nothing more nothing less. It is identity at its simplest expression, that is a cryptographic identifier.

Again I see it differently, what I believe you are *indirectly* talking about is the concept of transferability of a capability, for instance bearer tokens in OAuth (access_tokens) have virtually no protection against transfer, he who possesses the access_token can exercise the authority it permits. Employing a cryptographic binding layer to tokens of authorization, imposes the requirement that the invoker of the capability (token) prove possession of the cryptographic key the token is bound to. However this is NOT about the identity of the invoker, its about ensuring the invocation of that capability is sound in the eyes of the authority who issued it (or chain that delegated it). An example is a resource server in an OAuth architecture can readily validate requests from authorized parties by simply checking the validity of the access_token, they do not need to know (nor should they in many instances) the concrete identity of the authorized party.

Thanks,
Tobias Looker

From: **ProSapien Sam Smith**
Date: Thu, Dec 3, 2020 at 7:23 PM
Subject: Re: VCs & OCap - please talk
To: Tobias Looker
Cc: Hardman Daniel, Kaliya Identity Woman, Dmitri Zagidulin Manu Sporny, Drummond Reed

So to close the loop.

Given a generic authentic data container. One need merely specify precise semantics for a given syntactical schema to unambiguously perform an associated task. There would be no confusion as to the task because the syntactical schema and semantics are precisely defined. A container can hold multiple sets of syntactical blocks each with associated semantics. Again completely unambiguously. One mechanism establishes authenticity of all the contents. A chained authentic data container would allow granular provenance of each set. Moreover a chained container may be a node in multiple disjoint trees which are completely unrelated semantically. Each chained block contains references or proofs of the external provenance of contents the chained block in the container.

Now it may be a bad idea to chain different types of syntactical blocks with different semantics. But that would be an easily detected invalid use of chaining. Chains can easily require coherent semantics. But the container itself is merely establishing the authenticity of its contents to the source of the container. A single source may make multiple statements that all share the same source but have no other relationship to each other..

For example, suppose that there are three disjoint blocks in a container, namely A, B, and C. By virtue of those blocks being in the container, the authentic data container specification allows a verifier to ascertain the origin of the container, i.e is authenticity relative to a key-pair or a set of key-pairs and by implication to the holder of those key pairs whomever that may be. The verification happens only once. And all three blocks are now verified independent of the semantics and syntax each block.

Let suppose that block A has a sub block A.1 that comes from some other source. I.E another authentic data container Z. By including a reference to that other container Z in A.1, a verifier is able to verify the provenance of A.1 as authentically sourced from Z. And so on up a chain of provenance.

Likewise independently blocks B and C in the container, merely by virtue of being in the container are simultaneously verified as being authentic to the source of the container independent of their semantics. They are not in anyway necessarily dependent on each other. The container conveys authentic information from a source. The authentic data container is merely multiplexing one or more authentic statements, that are, A, B, and C. And each of A,B, and C may belong to completely mutually disjoint provenance chains.

Suppose that block A is an authorization statement. Its semantics do not in any impede the semantics of Block B or C. Nor do the semantics of block B or C impede or confuse A. They are not semantically related. They just all happen to come from the same authentic source.

The hard part of security is establishing authenticity of statements. Having one generic method for establishing authenticity of multiplexed statements via one conveyance is way more secure in general than having separate methods of establishing authenticity one for each type of statement each via multiple non-multiplexed conveyances.

Other than narrow corner cases, it defies best practice security principles to not multiplex.

To put a fine point on it. Suppose I want to access a web site. Should I establish a secure communication channel via TLS and then tunnel all my requests for any resource on that website via that secure channel or should I create multiple simultaneous channels one for each mime type of resource or even each resource. Well if there are some highly unique characteristics of different types of resources like asynchronous video streams versus database queries then maybe yes. But the authenticity of the website is the same in all cases. So it I have a secure way of establishing authenticity for any type of resource, it makes sense to reuse that one secure way whenever its practical. An to not use a different authenticity mechanism resource. The authentic source is the same so establish authenticity the same way. In general one just establishes one multiplexed authentic channel and lets the mime type of each multiplexed resource determine the semantic behavior.

If we build a single standard for authentic data containers with extensible and separable semantics for the data conveyed inside that container then we have a very high degree of reuse of tooling. This fosters much higher interoperability and adoption. If instead every time we have a

different type of semantic for data conveyed in a container we build a new non interoperable type of authentic data container then we will have a hot mess. We are just recreating at the VC layer, the DID method proliferation mess that happened at the DID layer.

From: **ProSapien Sam Smith**
Date: Thu, Dec 3, 2020 at 7:33 PM
Subject: Re: VCs & OCap - please talk
To: Tobias Looker
Cc: Hardman Daniel, Kaliya Identity Woman , Dmitri Zagidulin , Manu Sporny, Drummond Reed

Finally,

It may never be that one wants to put all of A, B, and C in the same container. That is not the point. The point is that the verification of the container as authentic does not care about A, B, or C. If it does then we have designed a bad container. We do not have separation of concerns. We have confusion of concerns. Arguing that VC somehow are meant for identity purposes is fundamentally confusing what a VC truly is. The term Verifiable Credential may be part of the problem. The definition of Credential is a right or privilege, i.e. it is synonymous with authorization. It is indeed ironic to be in a discussion where one side is arguing that verifiable authorizations (credentials) must not be authorizations.

From: **Drummond Reed** <drummond.reed@evernym.com>
Date: Thu, Dec 3, 2020 at 7:39 PM
Subject: Re: VCs & OCap - please talk
To: ProSapien Sam Smith <sam@prosapien.com>
Cc: Tobias Looker <tobias.looker@mattr.global>, Hardman Daniel <daniel.hardman@evernym.com>, Kaliya Identity Woman <kaliya@identitywoman.net>, Dmitri Zagidulin <dzagidulin@gmail.com>, Manu Sporny <msporny@digitalbazaar.com>

I am not a real software architect like the rest of you. But I must admit that Sam's argument that a verifiable container is required in all these scenarios and that having one consistent way of handling verifiable containers independent of the semantics in the container seems to be a serious advantage, especially considering all the infrastructure we are talking about building on top of this foundation.

What am I missing?

=Drummond

From: **Tobias Looker**
Date: Thu, Dec 3, 2020 at 7:46 PM
Subject: Re: VCs & OCap - please talk
To: =Drummond Reed
Cc: ProSapien Sam Smith , Hardman Daniel, Kaliya Identity Woman, Dmitri Zagidulin, Manu Sporny

> What am I missing?

I understand, in essence I think Verifiable Credentials are more opinionated than the generalized verifiable data container vision that Sam is putting out. For instance if you use the credentialSubject property in the VC you issue, you as the issuer are aiming to describe a subject in some capacity, which is fundamentally different than describing the authority to do something (a capability), it is that commingling that is problematic in my eyes and leads to a well document set of problems that often plague identity systems.
Tobias Looker

From: **Daniel Hardman**
Date: Thu, Dec 3, 2020 at 8:41 PM
Subject: Re: VCs & OCap - please talk
To: Tobias Looker
Cc: ProSapien Sam Smith, Kaliya Identity Woman, Dmitri Zagidulin Manu Sporny, Drummond Reed

> in essence I think Verifiable Credentials are more opinionated than the generalized verifiable data container vision that Sam is putting out.

Okay, excellent. I believe we've zeroed in on one root of our divergence, which is *whether VCs are a general mechanism for verifiable statements, or are specially dedicated to establishing identity*. This is progress. Commentary on that subtopic is indented below.

It is true that identity use cases predominate in the minds of VCs proponents today. However, I don't believe it's correct to consider them inherently related to identity. Here are several quotes from the VC spec that might be relevant:

- "Verifiable credentials represent statements made by an issuer in a tamper-evident and privacy-respecting manner." (section 1.3; note that identity isn't mentioned)
- Issuers can issue verifiable credentials about any subject." (section 1.3)
- "credential: A set of one or more claims made by an issuer. A verifiable credential is a tamper-evident credential that has authorship that can be cryptographically verified." (official definition from section 2)

- "It is possible to have a credential that does not contain any claims about the entity to which the credential was issued." (note in section 3.2)
- "When expressing statements about a specific thing, such as a person, product, or organization, it is often useful to use some kind of identifier so that others can express statements about the same thing. This specification defines the optional id property for such identifiers... If the id property is present..." (section 4.2)
- "A holder might transfer one or more of its verifiable credentials to another holder." (section 5.1)
- "This section describes possible relationships between a subject and a holder and how the Verifiable Credentials Data Model expresses these relationships... [Diagram: Subject present? no->Bearer Credential; yes->Subject = Holder? no->Credential Uniquely Identifies Subject? no->Subject Passes VC to Holder? no->Issuer Independently Authorises Holder?...]" (Appendix C)

Key points: VCs are defined by their composition from verifiable assertions, which can be about anything and for any purpose; identity is only one of many uses. VCs can be bearer tokens. VCs are explicitly contemplated for authorization use cases, including ones where the credential (authorization instrument) does not uniquely identify the subject. VCs can be transferable. This is all explicitly in the spec. Do you agree?

>For instance if you use the credentialSubject property in the VC you issue, you as the issuer are aiming to describe a subject in some capacity, which is fundamentally different than describing the authority to do something (a capability)

This is a second point of divergence, also important. *Is it really true that a capability describes the authority to do something, that this doesn't describe a subject in some capacity, and that this makes a capability a different animal from a VC?*

Here is the definition of a capability from wikipedia: "A capability... is a communicable, unforgeable token of authority. It refers to a value that references an object along with an associated set of access rights."

Note the part in red. A capability is not a capability unless it asserts that privileges attach to an object; it never describes privileges in the abstract. The distinction with VCs collapses by this definition, does it not?

Suppose a robotics student asserts that her course of study at Cambridge is very rigorous. I think we can all agree that this sort of assertion is not identity-centric. (It does reference a subject just like a capability references its resource or an RDF triple references its subject, but the student isn't asserting (and her listeners don't evaluate her statement) to establish the identity of Cambridge, or of herself. Right?)

It seems indisputable to me that this assertion is easily represented as a verifiable credential, without any distortion or abuse of the standard:

```
{
  "@context": [ "https://www.w3.org/2018/credentials/v1",
"https://studentu.org/campuslife" ],
  "type": ["VerifiableCredential", "AcademicExperienceReport"],
  "issuer": "https://facebook.com/AmyLovesRobots",
  "issuanceDate": "2021-01-01T19:73:24Z",
  "credentialSubject": { "id": "https://www.cam.ac.uk/",
"opinionAboutRigor": "Super challenging."},
  "proof": {...}
 }
```

So, now let's imagine that Amy wants to let a few of her friends play with the robot prototype she's been developing as her capstone project. She makes another verifiable statement framed in a way that satisfies the VC spec:

```
{
  "@context": [ "https://www.w3.org/2018/credentials/v1",
"https://roboticlab.cam.ac.uk/grants" ],
  "type": ["VerifiableCredential", "RobotPrivilege"],
  "issuer": "https://facebook.com/Amy",
  "issuanceDate": "2021-01-01T19:73:24Z",
  "credentialSubject": { "id": "https://roboticlab.cam.ac.uk/
~amyt/capstone-project", "allow": "operate"},
  "proof": {...}
 }
```

This VC also happens to be an OCap, in that it contains an unforgeable object reference combined inseparably with a statement that authorizes or confers privileges on its holder. It's an assertion that the bearer is authorized.

>it is that commingling that is problematic in my eyes and leads to a well documented set of problems that often plague identity systems.

Can you give some specific examples -- not examples of identity system problems solved by OCaps (which I think we all agree with), but of problems that would be introduced if we implemented OCaps with VCs instead of writing a new spec for them?

I suggest that we explore whether the separation into identity tokens vs authorization tokens is advisable or not. That might be an interesting question. However:

1. I suggest we define the purpose of VCs by the language in the standard, not by tribal wisdom -- not assuming that VC != OCap.
2. If we decide that two different tokens are desirable, it does not free us from the responsibility of explaining why we need two different specs for them.

From: **Manu Sporny**
Date: Fri, Dec 4, 2020 at 7:28 AM
Subject: Re: VCs & OCap - please talk
To: Tobias Looker, =Drummond Reed
Cc: ProSapien Sam Smith Hardman Daniel, Kaliya Identity Woman, Dmitri Zagidulin

On 12/3/20 10:46 PM, Tobias Looker wrote:
> I understand, in essence I think Verifiable Credentials are more
> opinionated than the generalized verifiable data container vision
> that Sam is putting out. For instance if you use the
> credentialSubject property in the VC you issue, you as the issuer are
> aiming to describe a subject in some capacity, which is fundamentally
> different than describing the authority to do something (a
> capability), it is that commingling that is problematic in my eyes
> and leads to a well document set of problems that often plague
> identity systems.

This is the crux of the philosophical divergence.

Fundamentally, Verifiable Credentials were meant to be used to do authorization. You *can* use them to do authorization in the same way that you can use a hammer to drive in a screw.

The abstraction that Sam is applying to VCs is really achieved one layer lower with Linked Data Proofs/Signatures, which allows any information to be digitally signed and verified as authentic.

Just because we *can* do authorization using VCs doesn't mean we should. One reason we shouldn't is because a number of things that can be capabilities (such as a DID Document on a ledger) are definitely not VCs (square peg, round hole problem). Another reason is because we don't want people to confuse what a VC is used for and what a ZCAP is used for (confused developers problem).

I certainly don't disagree with a number of the points that Daniel and Sam have made, but I don't think they're enough to get us over the two concerns I have above. That said, seeing a concrete proposal of the use cases Sam outlined might help analyse the benefits and drawbacks of the approach.

I also want to point out that this si a very useful discussion and it's
a shame that it's happening in a private channel. The entire community
would benefit from the discussion. Can we move it to the CCG mailing list?

-- manu

--
Manu Sporny - https://www.linkedin.com/in/manusporny/
Founder/CEO - Digital Bazaar, Inc.
blog: Veres One Decentralized Identifier Blockchain Launches
https://tinyurl.com/veres-one-launches

From: **Daniel Hardman**
Date: Fri, Dec 4, 2020 at 8:02 AM
Subject: Re: VCs & OCap - please talk
To: Manu Sporny
Cc: Tobias Looker, =Drummond Reed, ProSapien Sam Smith , Kaliya Identity Woman, Dmitri Zagidulin

I don't mind moving the conversation to the CCG channel. How should we provide the background context? Would it be good to spend some conversation time in an interactive meeting doing that, and connecting the email thread to that interactive discussion either before or after?

>Fundamentally, Verifiable Credentials were meant to be used to do authorization.

Just checking. You meant to say "weren't" there, right?

Regarding this statement from Manu:

>Fundamentally, Verifiable Credentials were meant to be used to do
authorization. You *can* use them to do authorization in the same way
that you can use a hammer to drive in a screw.

I strongly disagree with two aspects of this statement.

First, Manu can absolutely speak with authority about his own intentions here -- and as an editor of the spec, he can even speak about his experience getting the spec matured, where that perception was reinforced. But Manu is not the only editor of the spec, and I know his characterization has never been true about a significant subset of the community who contributed.

Second, I have yet to see ANY justification for the hammer/screw analogy. I gave an example of using a VC to express an ocap. It was simple and elegant, and 100% compatible with the VC

spec. To assert that this is using a hammer to drive a screw implies a much stronger misfit than that. So if we're going to make an assertion like that, I want concrete, specific examples to justify it. I repeat the invitation I gave to Tobias:

Can you give some specific examples -- not examples of identity system problems solved by OCaps (which I think we all agree with), but of problems that would be introduced if we implemented OCaps with VCs instead of writing a new spec for them?