Use case migration

Together with V6 comes a new understanding of how data exchange should be secured across organization boundaries. Some of the concepts, promoted by Nuts in the past, are no longer viable or don't make any sense anymore. V6 supports alternatives or this wiki describes how an alternative should be implemented.

Nuts defines the NutsAuthorizationCredential Verifiable Credential type. VCs of this type were used to authorize party A by party B. Then B had to check the contents of the VC when used. But since B issues these credentials based on some identity of A and knowledge of its own, it's actually enough to check the identity of A and this knowledge runtime as part of the authorization step. No credentials needed...

The Nuts network was used to transfer OrganizationCredential type VCs to every participant. These were used, in combination with the DID Document, to do service discovery. This mechanism contradicts the usage of VCs. VCs are to be held in a wallet by the holder. The service discovery mechanism replaces this mechanism.

User authentication was added to the access token request by signing some text with a means that supports signing. This links a real user to the organization that is doing the request. Development shows that signing is not yet a first class citizen and doing it properly conflicts with usability. A proper authentication procedure lets the user sign a challenge given by the authenticating party, this is usually the resource owner in healthcare data exchange. A user probably has to fetch data from multiple sources per patient. With a proper procedure this would result in a lot of authentication challenges. A more future proof model is when the user authenticates with its IdP. Multiple IdPs could form a federation where they trust each other's process. Additionally the ID token could be inspected by each relying party to obtain VCs presented at login.

Authorization

The biggest difference between a *V6 style* and *V5 style* use case is the way authorizations are handled. Authorization has always been the responsibility of the implementer. The Nuts node only provides assertions connected to an access token. With the NutsAuthorizationCredential these assertions could be quite "rich". They listed resources and access rights. When doing a *V6 style* use case you need to get this information from somewhere else. If all is well, you already have this information in a DB, since the NutsAuthorizationCredential is only a representation of the authorization state. Use identifying information from the token introspection (organization name/city/id) to query your own database for allowed resources.

Service discovery

Service discovery in a *V5 style* use case depends on publication of NutsOrganizationCredentials and services in DID Documents. This is replaced for *V6 style* use cases to a registration with a

central service (per use case). This has some pluses and minuses. The hardcoded requirement for a NutsOrganizationCredential and dynamic requirement for service types in a DID Document has been replaced by a flexible configuration. Per use case a service definition is published. This definition contains a *Presentation Definition* which specifies the required Verifiable Credentials, required endpoints and trust anchors. The trust API is no longer needed for *V6 style* use cases. From a client app point of view, you no longer register endpoints on a DID Document to enable a use case, but you call the Discovery registration API to enable a use case for a subject (and thus a DID). The registration API allows arbitrary key/value pairs which can be used to register endpoints. This also allows for more fine grained control over your registration, for example only register the receiver role for a use case. Once registered, the Nuts node will continue to refresh the registration periodically. The main downside of this mechanism is that a use case needs to select a single party to host the discovery server. The Nuts node acts as a server so each use case participant can fulfill this role.

Revision #8
Created 2024-11-05 08:16:50 UTC by Wout Slakhorst
Updated 2026-10-28 15:11:59 UTC by Rein Krul