v5 -> v6 migration guide

This book contains various pages on what to do before/during and after migrating your node. This includes things on configuration, DID management, API usage and use case design.

Configuration update

When updating your node from V5 to V6, you can choose to disable/enable certain parts depending on your needs. The common paragraph describes the required configuration changes.

Common

Before you update, make a backup and check the diagnostics of your node. To have the migrations run smoothly, make sure `vdr.conflicted_did_documents.owned_count` equals `0`. If not, please refer to the documentation on how to fix.

As stated in the release notes, you MUST do the following configuration changes:

- add `storage.sql.connection` config value according to the storage documentation
- add `url` config value according to the manual. The `auth.publicURL` has been removed and also uses that same config value.
- when using docker, make sure user ID 18081 has access to the mounted volumes.
- change port settings in the configuration and in your reverse-proxy. The new default is that public endpoints are bound to :8080 and internal endpoints to 127.0.0.1:8081.
- change reverse-proxy configuration to allow/disallow the correct endpoints. (See also later paragraphs)
- remove legacy internal API tokens and migrate to new API tokens if applicable.

Without additional features

You can disable the `did:web` method and only use the `did:nuts` method (like in v5) by adding the `didmethods` config param:

```
didmethods: [nuts]
```

Keep using the VDR v1 API for did document updates and the auth v1 API for access token requests. The V6 Nuts node exposes more paths to the public interfaces. These are only used by new features. If your reverse-proxy is configured correctly you block any unwanted paths already. The following paths resolve to APIs in V6:

- /iam
- /oauth2
- /.well-known
- /statuslist
- /discovery

Note: although some features may be disabled, your did:nuts documents are still migrated to an SQL database and their controller is set to blank (self-controlled).

Downgrade

It's possible to downgrade an v6.0.0 node back to v5.4.x. You'll also need to reset your configuration. The SQL database you configured for V6 can be removed. DID documents are still changed after the downgrade with the controller field removed.

DID management

When you upgrade from V5 to V6, your DID documents will be migrated to a SQL database. A new VDR V2 API has also been added. It's no longer possible to set a controller on a DID document. All DID Documents are *self-controlled*. It's also no longer possible to fine tune key usage. You can add or remove a key from a DID document but can't control if it's used for changing the DID document or signing data. The only exception is encryption keys which will be added in a later version.

In V6 you register **subjects** not **DIDs**. Existing **did:nuts** identifiers will be migrated to subjects (with the same ID). Operations done on subjects through the VDR V2 API will affect all supported DID documents (currently web and nuts).

With V6, we also promote a different way of registering services: on the **discovery service**. Previously, you registered services in the DID document and might even use references between DID documents to control a set of endpoints. There's a international tendency to only register public keys in the DID document. Also with the introduction of **did:web**, DID documents are not signed. This difference is reflected by the VDR V2 API which won't add services to **did:web** documents, only to **did:nuts** documents.

Given all of the above, you might have a challenge if your adding a V6 style use case to your existing set of V5 use cases:

- Which DID methods do you enable?
- Should existing **did:nuts** subjects get a **did:web**?
- When and how to switch to the new VDR V2 API?
- When to switch from DID document services to service registration on the discovery service?

The V6 release notes state that you should not use the VDR V1 API together with the VDR V2 API. This is a general statement to prevent runtime problems. The V1 API only transforms **did:nuts** documents and not **did:web** documents. So if you're using **did:web** in use cases, you should no longer use the VDR V1 API.

The problem can be broken down in two parts:

- Create/Delete/KeyCreation operations on DID documents (subjects in V6)
- Service registration

Note: each time you restart the node, all changes to **did:nuts** will be migrated to **did:web** if they are not in sync.

Create/Delete/KeyCreation migration

The Create/Delete/KeyCreation operations are done with the following API calls:

- POST /internal/vdr/v2/subject
- DELETE /internal/vdr/v2/subject/{id}
- POST /internal/vdr/v2/subject/{id}/verificationmethod

After updating your node to V6 you may choose the moment you change these operations to the new VDR V2 API. After you've made the transition, restart the node once to make sure any **did:web** document is in sync with the **did:nuts** document for the subject. Any service registration in a **did:nuts** document will not be copied to the **did:web** document.

You might also want to change the designation of the identifier you store in your own DB, it used to be a DID, now it's a subject. After the transition you're also able to choose your own identifier.

Service registration

For service registration it's recommended to create a new separate management page for the discovery API. That way the old management page can use the VDR V1 service operations and the new management page can use the discovery API. Both can coexist, even for the same subject. It's then still recommended to switch the old management page to the new VDR V2 API.

Being able to register service on the discovery service does not solve the problem how a use case migrates from V5 style to V6 style services.

Recommendation

Given the little amount of work in migrating the VDR API and usage of **did:web**

- update your node to V6 with **did:web** disabled (at least for staging/production)
- switch all VDR V1 API calls to the VDR V2 API in a big bang
- enable **did:web**
- add a new management page for service discovery to support new use cases
- Never use DID Document service registration on use cases which support **did:web**

Use case migration

todo

differences:

- service registration (including policy definition)
- trust configuration
- endpoint search
- token requests (including policy definition)
- token introspection
- did method support
- authorizations

order:

- determine trust and security baseline
- determine policies
- determine service registration params/vcs
- all updated to v6
- all migrate away from authorizing through NutsAuthorizationCredentials
- all support new access tokens and token introspection
- all register on discovery service (did:web)
- V6 style client interaction (discovery search, token request with XIS/ECD user info)
- Remove old service registration from DID documents
- Remove old authorization logic