

Configure url rewrite for OAuth

Other parties look up your node's OAuth2 metadata at a `.well-known` URL, which always starts at the root of the domain ([RFC 8615](#)). If your node runs at the root of its own (sub)domain, e.g. `https://nuts.example.com`, you don't need to do anything.

If the node runs on a subpath, e.g. `https://example.com/nuts`, your reverse proxy must forward the OAuth2 metadata URL to the node. Example for a node on `/nuts`:

Other parties request	Forward to the node as
<code>https://example.com/.well-known/oauth-authorization-server/nuts/oauth2/<subject ID></code>	<code>/.well-known/oauth-authorization-server/oauth2/<subject ID></code>

In short: remove the subpath (`/nuts`) after the `.well-known/...` part.

The node serves all its other endpoints at the root too (`/iam`, `/oauth2`, `/statuslist`, ...). So your proxy must also remove `/nuts` from those, e.g. forward `https://example.com/nuts/oauth2/...` to `/oauth2/...`.

NGINX

```
location /.well-known/oauth-authorization-server/nuts/ {  
    proxy_pass http://nuts-node:8080/.well-known/oauth-authorization-server/;  
}
```

Azure Application Gateway

```
path: '/.well-known/oauth-authorization-server/nuts'  
rewriteCondition: '/.well-known/oauth-authorization-server/nuts(.*)'  
rewriteRule: '/.well-known/oauth-authorization-server{var_uri_path_1}'
```

Check that it works

From outside your network, open:

```
https://example.com/.well-known/oauth-authorization-server/nuts/oauth2/<subject ID>
```

It should return JSON with your node's OAuth2 metadata, e.g. an `issuer` of `https://example.com/nuts/oauth2/<subject ID>`.

Revision #4

Created 2025-09-22 10:25:20 UTC by Jorrit Spee

Updated 2026-10-27 15:26:40 UTC by Rein Krul