Zorginzage V6 hackathon spec

This document is in DRAFT

Purpose

The purpose of the hackathon is to demonstrate a simple implementation of Nuts Zorginzage in Nuts V6. The present Nuts Zorginzage 2022 spec uses Nuts V5, but has several limitations, which is why the intention is to make use of Nuts V6 in the future. This hackathon is the stepping stone towards creating V6 Zorginzage usecases in the future.

Scope

Zoals in de email van Maarten is gedeeld:

> Zorgviewer
>
> Nuts technisch team
>
> **Voorbereiding deelnemende leveranciers: (Allen)**
>
> - Nuts v6 Node met een domainnaam draaien
> - FHIR Server met een test Patient
> - FHIR Client die een request kan maken
> - Interface om zorginstellingen te beheren en identiteiten aan te maken in de Nuts-Node
> - Een plek om informatie op te halen en te tonen
>
> **Voorbereiding rol raadpleger: (HinQ/Topicus/Zorgviewer)**
>
> - FHIR Client mot een access token kunnen aanvragen
> - FHIR Client moet het acces token gebruiken in een FHIR request
> - Nuts-node moet een OrgCredential kunnen aanvragen/verkrijgen
> - Nuts node configureren voor het abonneren op de discovery-service
> - FHIR Client kan zoeken binnen list van de discovery-service
>
> **Voorbereiding rol dossierhouder: (ECare/Nedap)**
>
> - Een PEP inrichten die een token introspectie doet bij de Nuts-node

>   - Het request en introspectie resultaat naar een OPA server sturen
>   - De OPA server moet in stat zijn het opgestelde policy te enforcen en een ja/nee antwoord aan de PEP te geven
>   - De zorginstelling aanmelden bij de discovery-service
>
> **Rol Discovery Service hoster: (Topicus)**
>
> Hosten van een Nuts-node die en discovery service aanbiedt voor de use-case met de bijbehorende presentation definition
>
> **Route hier naartoe**
>
> Start zal de output van de werkgroep zijn om via publieke documentatie de scope in te vullen. Leveranciers zullen onderling asynchroon de gemene deler en specificaties kunnen opstellen/delen binnen de Nuts bolt
>
> **Datum & Locatie hackathon** Voorstel is 2 dagen met 2 technische gegadigden van de leveranciers bij Neap in Groenlo. Eventueel kunnen er overnagedacht worden.
>
> Graag uiterlijk 1 augustus de datumprikker invullen, zodat we ook een doel hebben om naar toe te werken met elkaar
>
> Vriendelijke groet, Maarten

For the overview of the roles and info per participaint, see this section

Functional design

Discovery Service (adressbook & use-case whitelist)

The Discovery Service serves the function of an adressbook for the organizations that are compliant and available for this specific usecase. Organizations can periodically publish themselves by presenting a set of Verifiable Credentials to the DS. Which credentials should be presented is part of the governance of the use-case. To automatically select these credentials from the organization wallet a query must be agreed upon and ran on each node. The syntax of this query must follow the Presentation Defenition of the PEX specification

For more info about how the DS works see the following presentation of a Nuts Tech session

Promedico will host the Discovery Service within this Hackathon.

Presentation definition

Verifiable Credential

The PDs credibility is determined by the combined assurance level of the required credentials (VC). During the hackathon, only one verifiable credential, `NutsUraCredential` will be used. This is very similar to the Nuts Organization Credential (RFC012), but the primary difference is that it is no longer issuer by the vendor, but rather issued by a trusted authority. Also it uses the URA number instead of the KVK number as primary identifier.

The contents of the new 2024 version of the `NutsUraCredential` will be discussed in this Github Issue #3324.

```
{
  "id":"did:nuts:123#demo-uracredential",
  "type": [
    "VerifiableCredential",
    "NutsUraCredential"
  ],
  "issuer":"did:tdw:cibg-issuer",
  "credentialSubject": {
    "@id":"did:nuts:123",
    "@type":"Organization",
    "legalName": "De Regenboog",
    "memberOf": {
      "@type": "ProgramMembership",
      "membershipNumber": "12345",
      "programName": "UZI Register Abonnee"
    }
  }
}
```

In the hackathon the NutsUraCredential will be sufficient, however it is likely that in a production V6 Zorginzage usecase more information is desired. For example: the public name, adress, department, or region. This information is not as important to be issued from a trusted-issuer. No authorization would be dependant on it for instance. Hence, this information could be packaged in a self-issued VC. The structure and contents of this VS (for example: NutsLocationCredetinal) would ideally be standardized by the use-case participants.

Credential issuer:

There will be one trusted party that issues these VCs to the organization wallets. The issuer needs a piece of software to issue credentials. There are a few available implementations which can issue such a credential.

Options:

- Issue through Nuts Node, optionally using Nuts Admin. Holders will have to receive the credential(s) out of band, and load it into their Nuts node using its REST API or Nuts Admin. The needed software is part of https://github.com/nuts-foundation/nuts-admin

The authority that will spin it up and issue the credentials for this hackathon will be ZorBijJou.

> TODO: ZorgBijJou will choose a platform to issue NutsUraCredential and describe how it will work

Service Definition

> TODO: Promedico to finalize the Service Definition

```
{
  "id": "hackathon_v2024.10",
  "endpoint": "https://example.com/usecase/hackathon/v2024.10",
  "presentation_max_validity": 259200,
  "presentation_definition": {
    "id": "pd_care_organization",
    "format": {
      "ldp_vc": {
        "proof_type": ["JsonWebSignature2020"]
      },
      "ldp_vp": {
        "proof_type": ["JsonWebSignature2020"]
      }
    },
    "input_descriptors": [
      {
        "id": "1",
        "constraints": {
          "fields": [
            {
              "path": ["$.type"],
              "filter": {
                "type": "string",
                "const": "NutsUraCredential"
              }
            },{
              "id": "name",
              "path": ["$.credentialSubject.legalName"],
              "filter": {
                "type": "string"
              }
            },{
              "id": "ura",
              "path": ["$.credentialSubject.memberOf.ura"],
              "filter": {
                "type": "string"
              }
            }
          ]
        }
      }
    ]
  }
}
```

PD logistical information

Alongside the VCs in the presentation definition, there following logistical information is also be presented

- Usecase ID:
- Max validity: `60 minutes`
- Server endpoint: T.B.D by Promedico
- ?

Endpoint discovery

In order to request data, the requestor needs to interact with several services. Once a source has been located, the services as defined in the DID Document of the source will be used to discover their endpoints. For more context see the documentation about endpoint-discovery. This does not differ from Nuts V5.

```
{
  "services": [
    {
      "id": "#hackathon_v2024.10",
      "type": "fhir-api",
      "serviceEndpoint": {
        "api": "https://fhir.example.com/api/",
        "auth": "https://auth.example.com/auth"
      }
    }
  ]
}
```

Localization

By localization we mean “finding out where a patient is in treatment”. That will not be in scope for this hackathon*. Instead, we will request directly to the other organizations in the discovery service without checking/finding out if there is a patient file there.

**Zorg bij jou intends to host a service (Care Plan Service) where every organization can check which other organizations have a care treatment with the patient by hosting CareTeam FHIR-resources. It's optional for other parties to make use of this during the Hackathon.*

Data-access policies

Wheter or not a requestor gets access to the data they are requesting depends on whether they pass the access-polices of the source (bronhouder). To ensure interoperability, a uniform standard for the data-acces policies on a Policy Enforcement Point (PEP) is described for each usecase. The current intentino for the hackaton is to define this in the Open Policy Agent (OPA) scripting language. Validating the access-policies is a two part process. Firstly, the requester presents all the required verificable credentials to the bronhouder. Then, the bronhouder does their own checks and based on the outcome provides an access token or not.

Access policy step 1: requester presents required VCs

1.1 Organisation information (`NutsUraCredential`)

The same fields that are present in the Service Defention for the NutsUraCredential are required to be presented to the requested party.

1.2 Practitioner identification & authentication

The practitioner that initiates the request must be identifiable so that it can be verified at a later moment, who initiated what action (in line with NEN7513). Just having the identification information present has been deemed insufficient because that lacks the proof that a human performed an action (as opposed to a machine). To add some traceability and log who performed the action, the Employee Identity Credential will be used and checked in the data-access policy. This credential-issuance is part of the default Nuts node and is a required resource.

Access policy step 2: requester checks VCs and enforces the Policy

The Policy Decision will be based on input such as the identity of the requesting organisation and practitioner, a legal basis such as a patient consent and if the requestor is part of the care team. During the hackathon we reduce complexity, and not require a patient-consent. The technical implementation

2.1 Organisation information

Check whether the URA number provided is currently present in the Discovery Service.

2.2 Practioner information

Check wheter a valid Employee ID credential has been provided. Also check if it's expired.

> TODO: decide whether RFC021 (service access token) is used v.s. user access token (OpenID4VP flow)

2.3 Authorization policy (checking legal base)

It is the responsibility of the source to check whether the requestor has a legal-base for accessing the data. A legal-base can be implied or be in the form of an explicit patient consent.

During the hackathon we will first attempt a retrieval without an authorization policy to validate the chain works. A, we can test an authorization policy based on being a participant of a CareTeam, using ZorgbijJou's Careteam service, is the intended secondary authorisation policy to use.

2.4 Requested resources

The requested party will check wheter the requested FHIR resources are in scope for the requester in this usecase.

Technical implementation access policy

In OPA, a policy is called with an "input" set and generates an output.

The input consists of the access token introspection result, the http-request and additional information such as results from FHIR queries for `Consent` and `CareTeam`.

The following policy and input can be developed using the OPA playground.

Policy (check it out in the playground:

```
package hackathon_v2024

import rego.v1

default allow := false

allow if {
	# this policy is applicable to the scope
	input.introspectionResult.scope[0] == "hackathon_v2024.10"
    # the requesting org is part of the care team
	care_team_members[requestor_ura]
    # the operation on the requested resource is alowed
	operation_allowed
}

resource_rights := {"Patient": ["read"]}

operation_allowed if {
	http_operations := {
		"GET": "read",
		"POST": "write",
		"PUT": "update",
		"DELETE": "delete",
	}
	operation := http_operations[input.request.http.method]
	some allowed_operation in resource_rights[resource_name]
	allowed_operation == operation
}

path := input.request.http.path

bsn := regex.find_all_string_submatch_n(`^.*identifier=http://fhir.nl/fhir/NamingSystem/bsn\|(\d+)$`, path, -1)[0][1]

resource_name := regex.find_all_string_submatch_n(`\/([A-Z]\w+)\??`, path, 1)[0][1]

requestor_ura := input.introspectionResult.input_descriptor_constraint_id_map[uracredential_uraNumber]
```

```
scope := input.introspectionResult.scope[0]

care_team_members contains identifier if {
    # select the care team for the patient with bsn
    care_team := [team |
        input.resources[_].resourceType == "CareTeam"
        input.resources[_].subject.identifier.value == bsn
        team := input.resources[_].participant[j]
    ]

    # select the members with an ura as identifier (all organisations)
    some member in care_team
    identifier := member.member.identifier.value
    member.member.identifier.system == "$ura"
}
```

Input:

```
{
  "introspectionResult":{
    "active": true,
    "client_id": "did:web:nuts-node-gbz.nuts.example.nl:iam:d73ca5d4-4dc8-4137-8992-578e825e3f36",
    "exp": 1706689514,
    "iat": 1706688614,
    "input_descriptor_constraint_id_map": {
        "uracredential_uraNumber": "32475534"
    },
    "iss": "did:web:nuts-node-zkh.nuts.example.nl:iam:2333ea28-a719-4896-9a12-f855b225755b",
    "scope":["hackathon_v2024.10"]
  },
  "request": {
    "http": {
      "headers": {
        ":method": "GET",
        ":path": "/fhir/Patient?identifier=http://fhir.nl/fhir/NamingSystem/bsn|111222333",
        "accept": "*/*",
        "authorization": "Bearer Y2hhcmxpZTpwYXNzdzByZA==",
        "content-length": "0",
        "user-agent": "curl/7.68.0-DEV",
        "x-ext-auth-allow": "yes",
        "x-forwarded-proto": "http",
        "x-request-id": "1455bbb0-0623-4810-a2c6-df73ffd8863a"
      },
      "host": "example-app",
      "id": "8306787481883314548",
      "method": "GET",
      "path": "/fhir/Patient?identifier=http://fhir.nl/fhir/NamingSystem/bsn|111222333",
      "protocol": "HTTP/1.1"
    }
  },
  "resources": [{
    "resoureceType": "Consent",
    "subject" : {
      "reference" : "Patient/88eb7e81-a3b6-4236-b458-451cf6e437b3"
    }
  },
  {
    "resourceType" : "CareTeam",
    "id" : "cps-careteam-01",
    "meta" : {
      "versionId" : "1",
      "profile" : ["http://santeonnl.github.io/shared-care-planning/StructureDefinition/SCPCareTeam"]
    },
    "category" : [{
      "coding" : [{
        "system" : "http://snomed.info/sct",
        "code" : "135411000146103",
        "display" : "Multidisciplinary care regime"
      }]
    }],
    "subject" : {
      "identifier" : {
        "system" : "http://fhir.nl/fhir/NamingSystem/bsn",
        "value" : "111222333"
      }
    },
    "participant" : [{
      "member" : {
        "identifier" : {
          "system" : "$bsn",
          "value" : "111222333"
        }
      },
      "period" : {
        "start" : "2024-08-27"
      }
    },
    {
      "member" : {
        "identifier" : {
          "system" : "$ura",
          "value" : "32475534"
        }
      },
      "period" : {
        "start" : "2024-08-27"
      }
    }]
  }]
}
```

- □ Design an OPA policy
- □ Decide on error-responses for authorization

Access token specification

Access tokens follow the format of the Nuts v6 tokens: Example AccessToken introspection

Resources

The aim of the hackathon is model a simple data-request and validate each step in the proces.

Therefore the set of resources is deliberately limited to a nl-core-patient.

Hackathon participant information

This table must be filled in by all the participants so it's clear what role they perform during the hackathon. The role can be a multiple of the following: (credential)Issuer, Discovery, (medical data)Requestor, (medical data)Source. The Issuer and Discovery roles can not be combined with the Requestor and Source roles. If a participant performs both roles, add more lines. The Requestor and Source lines need to provide a fictional URA as well.

Participant	Fictional Org	Role	URA	Connects with
Ecare	Ecare PUUR. Development	Source	66677777	
Formelio	HA het Piratenschip	Source	66633333	
Formelio	HA het Piratenschip	Requestor	66633333	
Nedap	VVT Grolle	Source	66644444	
Nedap	HA Grolle	Requestor	66655555	
Nuts - Steven	VVT de regenboog	Source	66611111	
Nuts - Steven	HA de Leeuw	Requestor	66622222	
Topicus	HA Topicus	Requestor	66699999	
Topicus	RSO "om de hoek"	Discovery	none	
Zorg Bij Jou	CIBG	Issuer	none	

Test Patients

The following patients and their BSNs will be used during testing:

Name	BSN
Tanja Test	99999001
Tinus Test	99999002

Localisation overview hackathon

Since localisation is out of scope for this hackathon, we use this simple table. Every organisation who contains information about this patient should add a row to the following table:

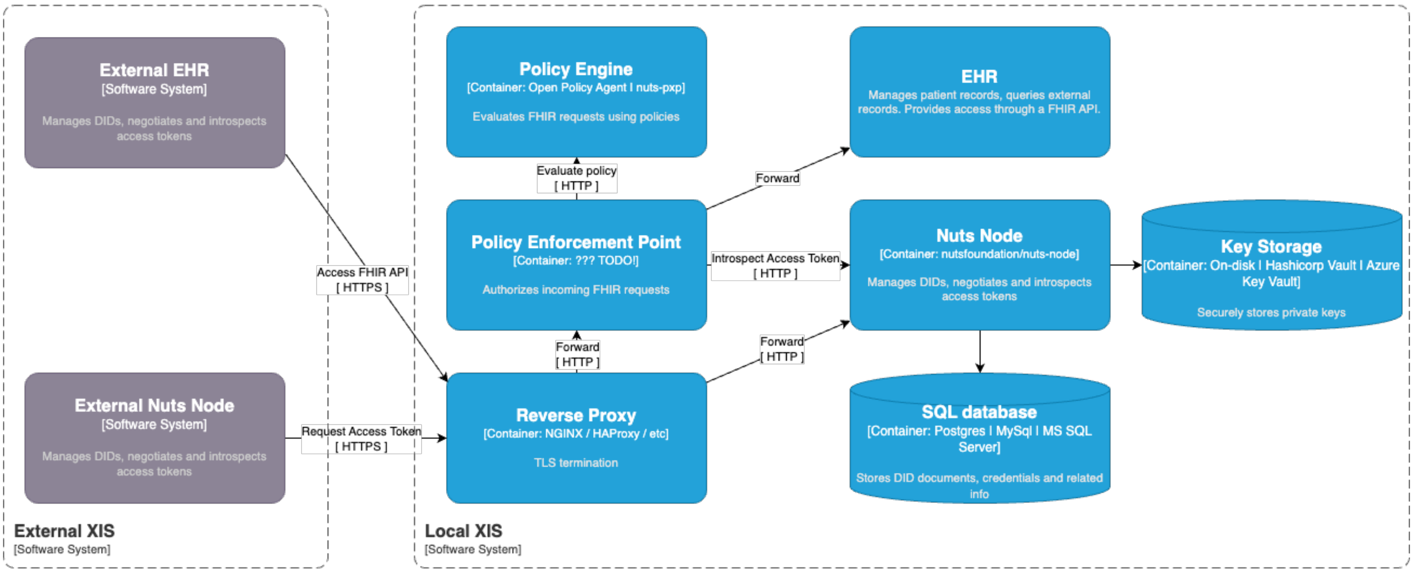
BSN	URA	Fictional org
099999001	66677777	Ecare PUUR. Development
099999002	66677777	Ecare PUUR. Development
99999001	66611111	VVT de regenboog
99999001	66622222	HA de Leeuw
99999001	66633333	HA Het Piratenschip
99999002	66611111	VVT de regenboog
99999002	66622222	HA de Leeuw

BSN	URA	Fictional org
99999002	66633333	HA Het Piratenschip
99999001	66644444	VVT Grolle
99999002	66644444	VVT Grolle

Deployment

Hackaton Zorginzage Deployment Diagram

Recommended deployment for providing Zorginzage functionality through Nuts (v6) and Policy Evaluation.


External EHR
[Software System]
Manages DIDs, negotiates and introspects access tokens
External Nuts Node
[Software System]
Manages DIDs, negotiates and introspects access tokens
External XIS
[Software System]
Access FHIR API
[HTTPS]
Request Access Token
[HTTPS]
Policy Engine
[Container: Open Policy Agent | nuts-pxp]
Evaluates FHIR requests using policies
Evaluate policy
[HTTP]
Policy Enforcement Point
[Container: ??? TODO!]
Authorizes incoming FHIR requests
Forward
[HTTP]
Reverse Proxy
[Container: NGINX / HAProxy / etc]
TLS termination
EHR
Manages patient records, queries external records. Provides access through a FHIR API.
Forward
Introspect Access Token
[HTTP]
Forward
[HTTP]
Nuts Node
[Container: nutsfoundation/nuts-node]
Manages DIDs, negotiates and introspects access tokens
Key Storage
[Container: On-disk | Hashicorp Vault | Azure Key Vault]
Securely stores private keys
SQL database
[Container: Postgres | MySql | MS SQL Server]
Stores DID documents, credentials and related info
Local XIS
[Software System]

Revision #45
Created 29 August 2024 13:18:39 by Edwin de Wit
Updated 6 December 2024 09:27:43 by Hugo van der Geest