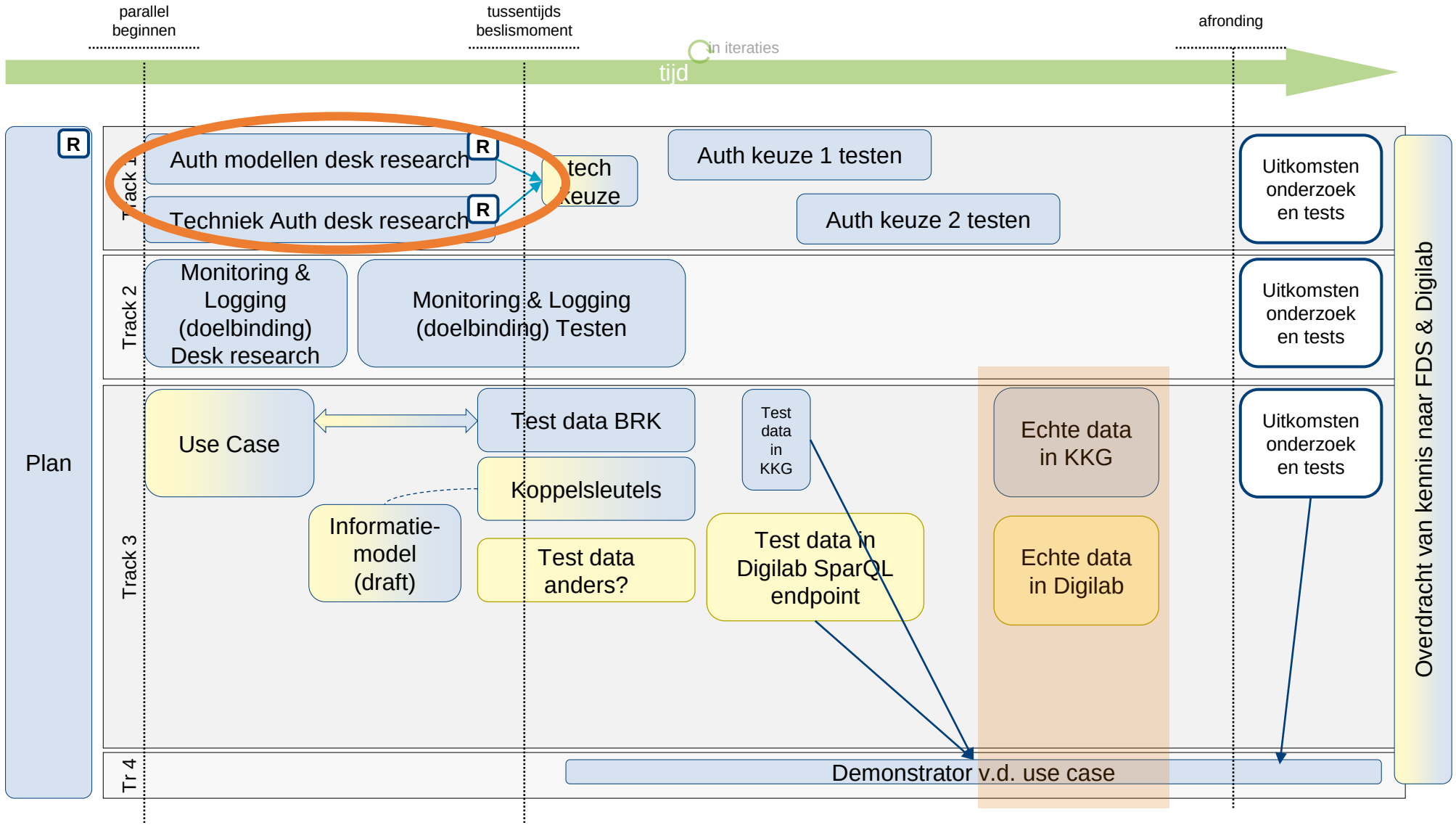
Desk Research Autorisatie

Versie 1.0 – 27 oktober 2023

Inhoud document

- Inleiding autorisatie
- (Verdiepende) Requirements voor Lock-Unlock
- Technologische oplossingen
- Voorstel beproevingen Lock-Unlock
- Achtergrond/bijlagen

Context – project Lock - Unlock



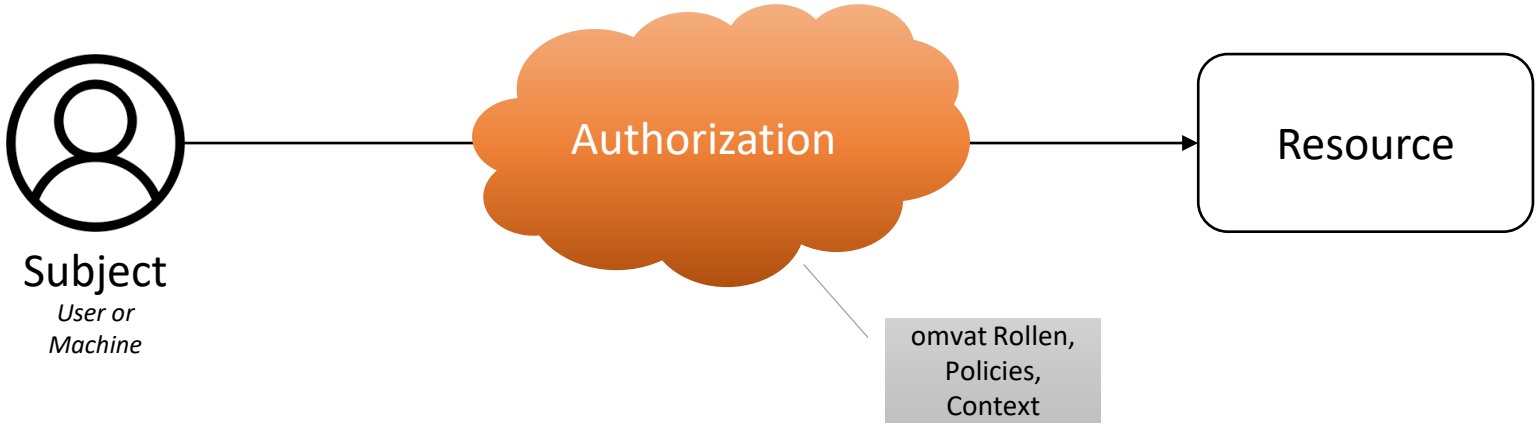
Doel document

- Door middel van desk research verschillende mogelijkheden beschrijven voor autorisatiebeheer, met daarbij de focus op Linked Data.
- Op basis van deze verkenning een voorstel doen voor welke techniek of technieken het beste beproefd kan/kunnen worden binnen het Lock – Unlock project.

Wat hebben we gedaan?

- Basisconcepten rondom autorisatie tot ons genomen.
- Diverse technieken voor autorisatie bekeken, incl. een aantal implementaties van deze technieken.
- Bovenstaande hebben we geabstraheerd tot requirements en autorisatieconcepten/-modellen.
- De volgende slides zijn een samenvatting van wat we hebben gezien en de belangrijkste concepten lichten we toe.
- In de “Achtergrond” slides is meer / extra informatie te vinden.

Autorisatie, hoe moeilijk kan het zijn?



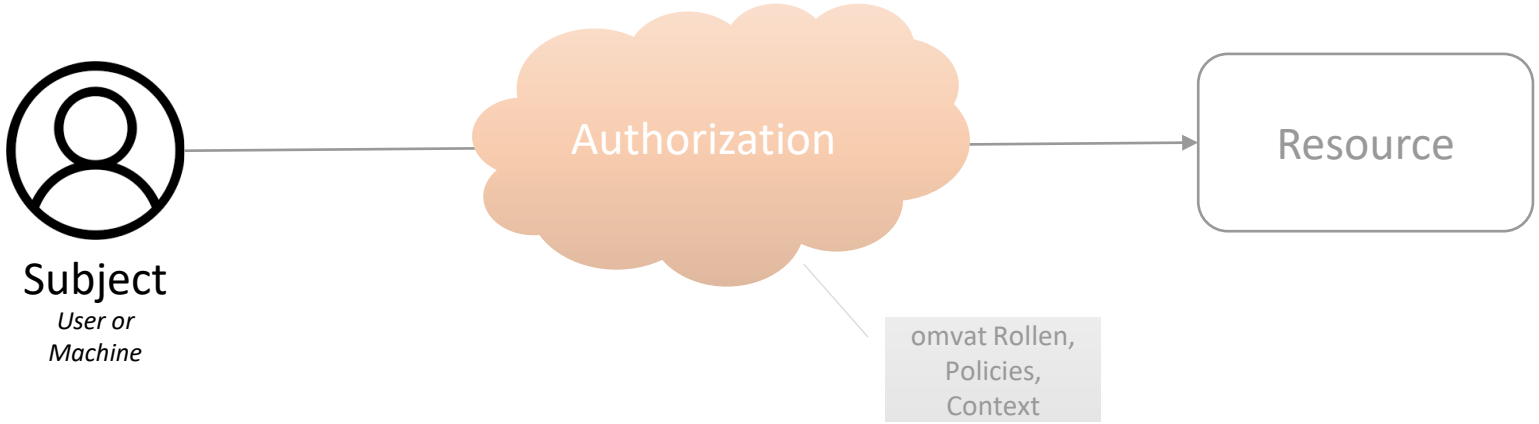
In beginsel is autorisatie eenvoudig: iemand wil bij iets en de vraag is of dat mag. Was het maar zo gemakkelijk.

‘iemand’ betekent dat bekend moet zijn wie diegene is. Dit vraagt om identificatie, identificerende gegevens, identiteiten. Het proces om dit te verifiëren heet authenticatie en dat is een onderwerp op zich.

Het ‘iets’ betekent ‘een resource’. Dat kan een hele dataset zijn, bijv. een basisregistratie als BAG, BGT of BRK. Het kan ook een specifieke selectie zijn uit een dataset of een selectie uit de *combinatie* van meerdere datasets. Dat ‘iets’ is al snel complex.

Veel gehanteerde termen zijn ‘subject’ voor ‘de iemand die de vraag stelt’. ‘Resource’ wordt gebruikt voor het ‘iets dat opgevraagd wordt’. De autorisatie wordt bepaald door ‘policies’, de voorwaarden, waaronder het subject toegang krijgt of niet. Deze gaan uit van een ‘context’ waarin de ‘rollen’ van het ‘subject’ en extra informatie op basis van het ‘request’, de vraag, kunnen worden meegenomen in de afweging. Deze terminologie is formeel vastgelegd in XACML (zie achtergronden).

Autorisatie: Subject



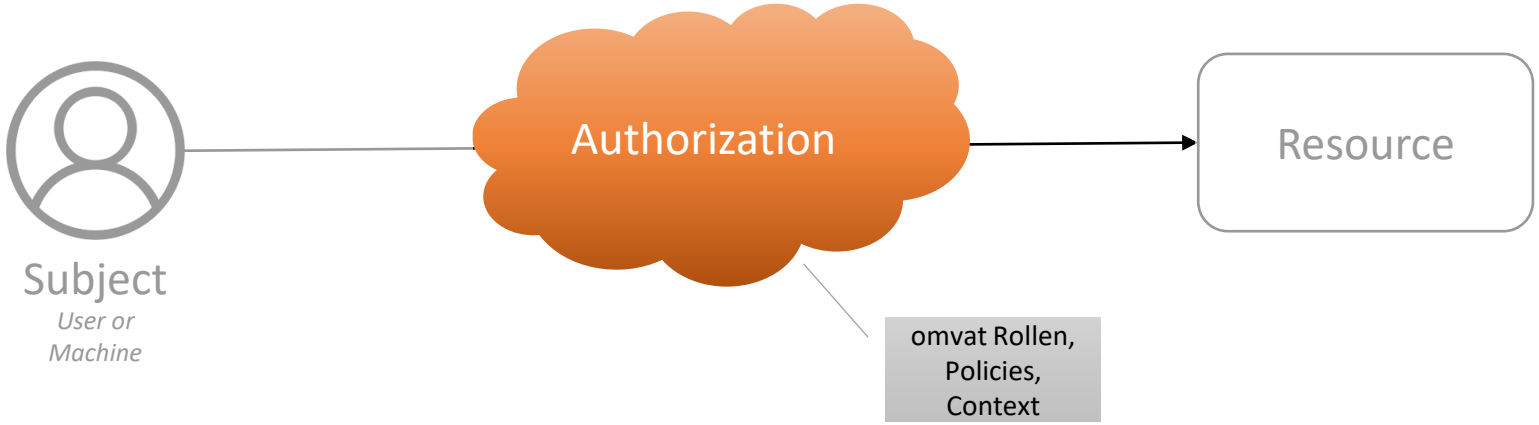
Een **subject** kan een gebruiker zijn en ook een machine (computer). Afhankelijk van het type bestaan er verschillende soorten van identiteiten.

Gebruikers worden meestal beheerd in een ‘User Management System’, welke weer gevoed kan worden door een HR systeem. Daarin worden meestal ook functies en rollen bijgehouden en uitgegeven door de beheerorganisatie over wat ‘iemand mag’. Een gebruiker heeft dan een ‘username’ (en wachtwoord) en een set aan rollen. Een geautoriseerd persoon geeft toestemming en legt verantwoording af over de rechten (rollen) van een gebruiker.

Machines worden vaak dmv certificaten of (API) ‘keys’ (sleutels) geïdentificeerd. Deze zijn uitgegeven door een autoriteit en zijn geautomatiseerd te controleren.

Zowel gebruikersaccounts als sleutelbeheer zijn complexe beheersprocessen waarin vele lagen van (verborgen) complexiteit zitten, zoals het kunnen intrekken, delegeren, verschillende niveaus van vertrouwen passend bij de verschillende niveaus van betrouwbaarheid die nodig is, etc, etc.

Autorisatie: Doelbinding

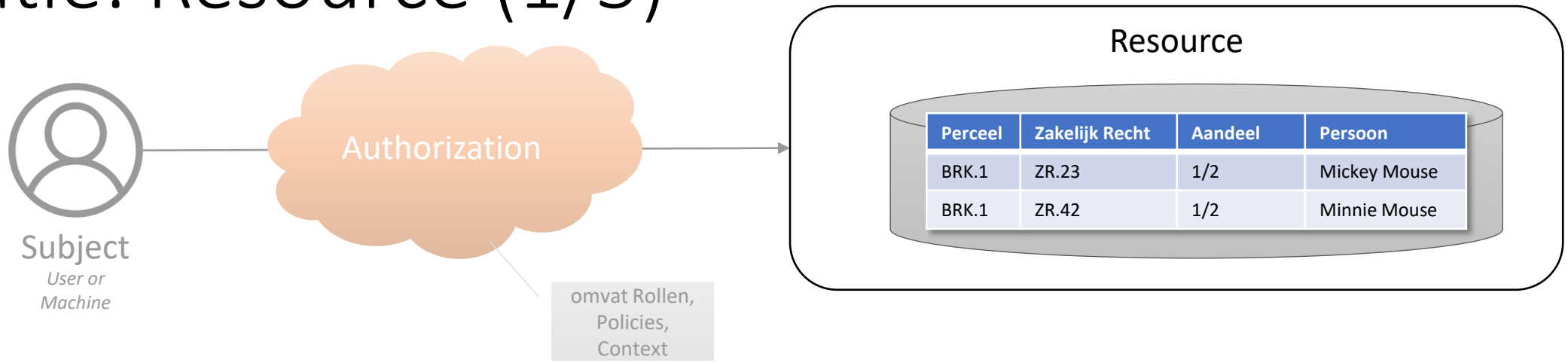


Een subject krijgt vaak niet zomaar toegang tot een resource. Bij het uitdelen van autorisaties wordt o.a. gekeken naar interne en externe beleidstukken en wet- en regelgeving. Een belangrijk aspect hierbij is de doelbinding, zeker als het om persoonsgegevens gaat: gegevens mogen alleen worden verwerkt en verzameld voor een specifiek en gerechtvaardigd doel.

Dit is niet gemakkelijk te controleren. Het punt is namelijk dat die doelbinding per casus gebonden zou moeten worden, maar informatie vaak over organisatiegrenzen heen gedeeld en gebruikt wordt. Het is voor de ene organisatie niet mogelijk om te beoordelen of de specifieke casus waar de betreffende gebruiker van de andere organisatie mee bezig is, passend is voor het doel dat beoogd is. Vooraf specifieke doelbinding controleren is daarom niet (volledig) mogelijk. Daar komt bij dat doelbinding op dit moment niet “machine readable” is en de relatie met het datamodel is niet formeel is vastgelegd. Hierdoor is het (ook) niet mogelijk om een koppeling te leggen tussen de doelbinding en de bijbehorende attributen en queries.

In geval van toegang verlenen, oftewel autorisatie, wordt wel vaak een afgeleide gemaakt van de doelbinding op een hoger niveau dan specifieke casussen. Een hele organisatie heeft toegang tot de gehele dataset van een andere organisatie. Door gestandaardiseerd te loggen wat precies wordt opgevraagd door wie, is wel specifieke controle achteraf mogelijk. De GEMMA Verwerkingenlogging is hier de standaard (in ontwikkeling) voor.

Autorisatie: Resource (1/5)

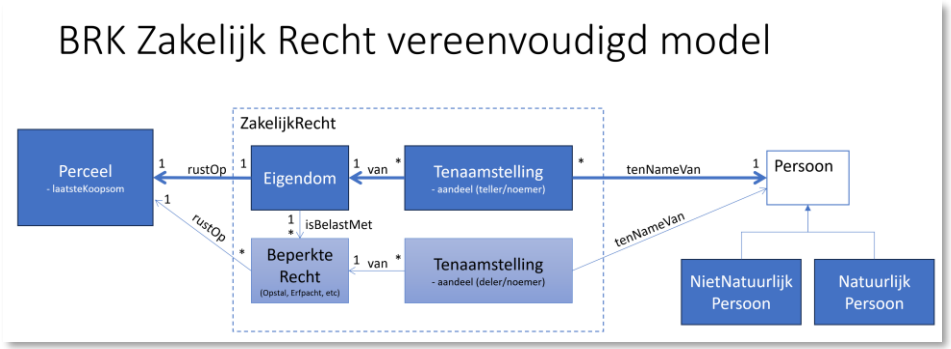


Bij een '**resource**' denken we in eerst instantie al snel aan een '**tabel**'. Een set aan gegevens van rijen en kolommen.

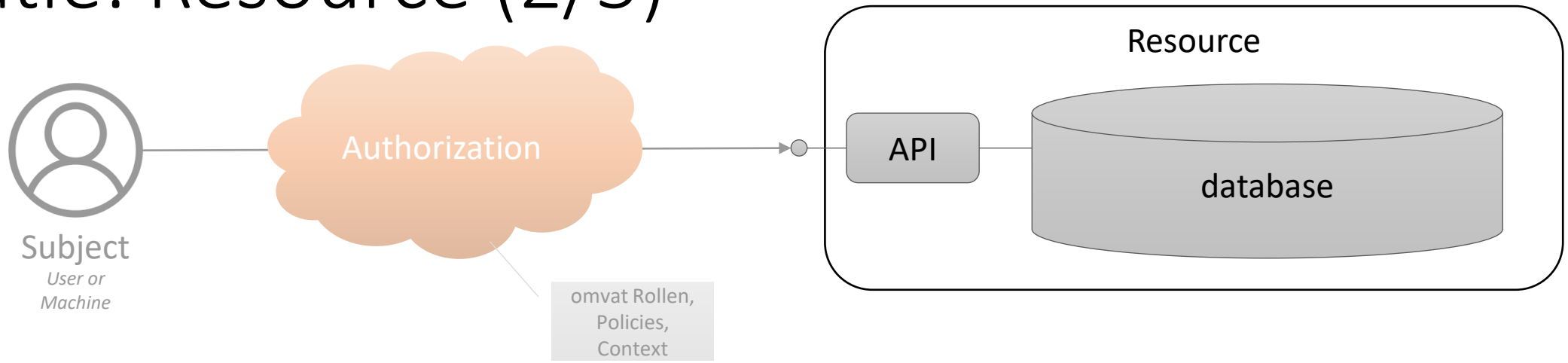
Was het maar zo eenvoudig... Meestal is een 'set of data', een verzameling tabellen. In veel gevallen is er een volledige informatiemodel (IM) of datamodel ontworpen om alle relaties tussen deze tabellen nauwkeurig en volledig weer te geven. De verschillende objecten in het model hebben elk hun eigen tabel en worden in de tabel impliciet gedefinieerd door sleutelkolommen. Zo'n hele verzameling gerelateerde objecten en een set gegevens wordt een **dataset** genoemd.

Voorbeelden: de basisregistraties zoals BRK, BAG, BGT.

In deze context is een resource een database die een of meer tabellen bevat.



Autorisatie: Resource (2/5)



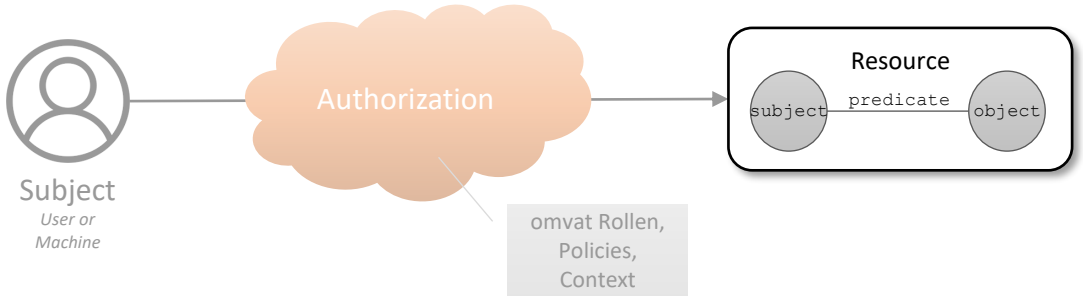
Een tabel of een dataset zit meestal in een database. Goed gebruik is om deze niet direct toegankelijk te maken voor gebruikers, maar te voorzien van een **Application Programming Interface (API).** Een API is een technisch koppelvlak die mogelijkheden biedt voor het opvragen (en muteren) van data en daar ook controles en beperkingen aan kan stellen. Hoewel API een generiek concept is, wordt in de huidige staat van de technologie meestal een '**REST API**' bedoeld.

Een API kan bijvoorbeeld een beperkt deel van de data(base) beschikbaar maken. Hierin is een verschil tussen horizontale en verticale scheiding (segmentatie). Horizontale scheiding betekent dat niet alle rijen op te vragen zijn. In het kader van autorisatie zou dat bijvoorbeeld beperkt kunnen zijn tot een bepaalde regio, bijvoorbeeld gemeentelijke grenzen.

Een verticale scheiding betekent dat niet alle kolommen op te vragen zijn. Een voorbeeld hiervan is dat perceelinformatie als open data beschikbaar is, maar de persoonsgegevens (uiteraard) niet. Deze zijn alleen op te vragen met de juiste rechten, of eigenlijk: de juiste grondslag.

In deze context is de resource de combinatie van database en (REST) API.

Autorisatie: Resource (3/5)



Een resource kan een ingewikkeld informatiemodel hebben en het wordt nog ingewikkelder als meerdere datasets gecombineerd moeten worden. Dat betekent dat er relaties tussen meerdere informatiemodellen moeten worden gelegd.

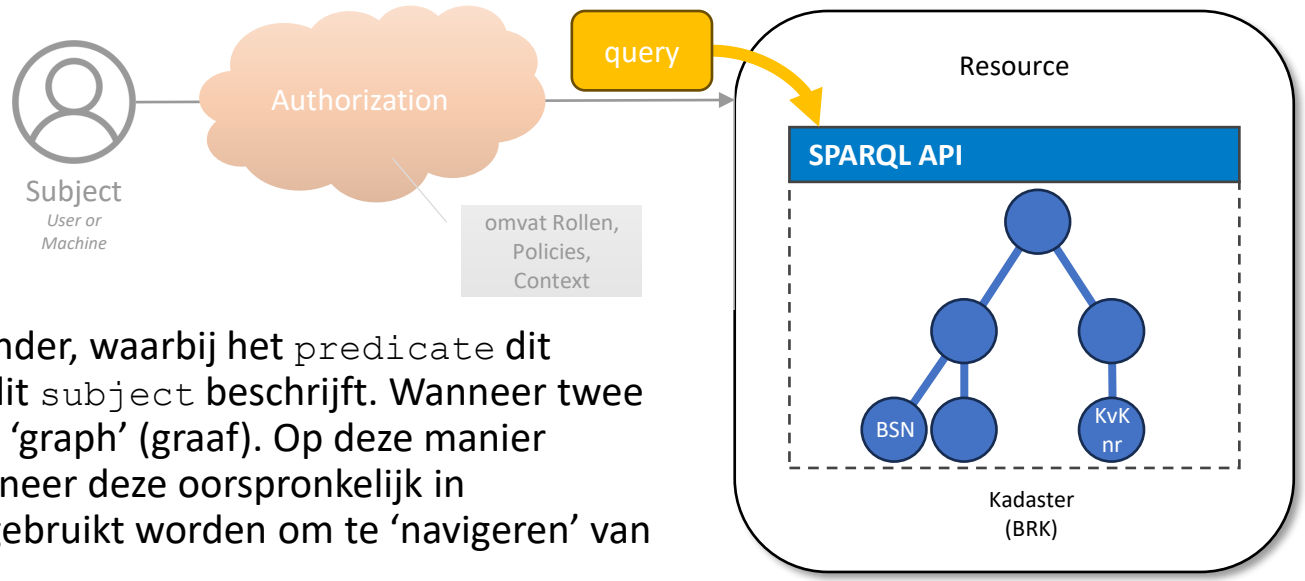
Linked Data is een concept en technologie die hier veel flexibiliteit en expliciete ondersteuning voor biedt. In plaats van tabellen wordt hierin de data 'uit elkaar gehaald' tot zogenaamde '**triples**'. Elke data instantie is een '`subject`' dat een relatie heeft ('`predicate`') tot een '`object`'. En dit kan oneindig! Zo kunnen relaties leiden tot een object die attributen beschrijft zoals in een meer traditioneel informatiemodel en een tabel in een database.

In deze context wordt een triple als een resource beschouwd.

Perceel	Zakelijk Recht	Aandeel	Persoon
BRK.1	ZR.23	1/2	Mickey Mouse
BRK.1	ZR.42	1/2	Minnie Mouse

Let op: een resource wordt in de terminologie van Linked Data anders gedefinieerd dan in de context van dit rapport. Zie achtergrond voor meer informatie. In de context van dit rapport wordt *niet* de definitie van resource zoals gedefinieerd in Linked Data gebruikt.

Autorisatie: Resource (4/5)



Het `object` in de ene triple kan het `subject` worden in een ander, waarbij het `predicate` dit `subject` aan een ander `object` koppelt of een attribuut van dit `subject` beschrijft. Wanneer twee of meer triples met elkaar verbonden zijn, resulteert dit in een 'graph' (graaf). Op deze manier kunnen triples informatie/data met elkaar verbinden, ook wanneer deze oorspronkelijk in verschillende tabellen stonden. De graph die zo ontstaat, kan gebruikt worden om te 'navigeren' van `subject` naar `object` naar `subject` naar `object`.

Een graph wordt opgeslagen in een specifieke database, namelijk een 'triple store'. Ook deze wordt niet direct ontsloten voor gebruikers, maar beveiligd en afgeschermd door een API. Voor Linked Data is een 'SPARQL API' of 'SPARQL endpoint' een veelgebruikte API. SPARQL is een 'query language', een vraagtaal voor Linked Data obv internet technologie (oa HTTP).

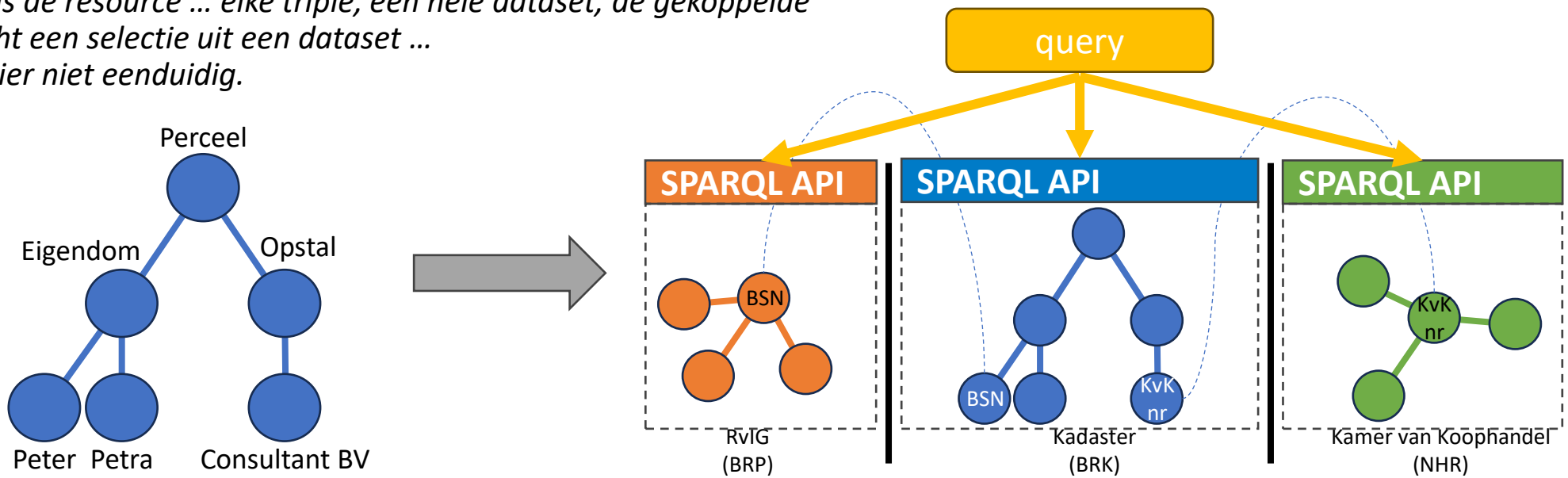
In deze context is de resource de graph en het bijbehorende SPARQL API / endpoint.

Autorisatie: Resource (4/5)

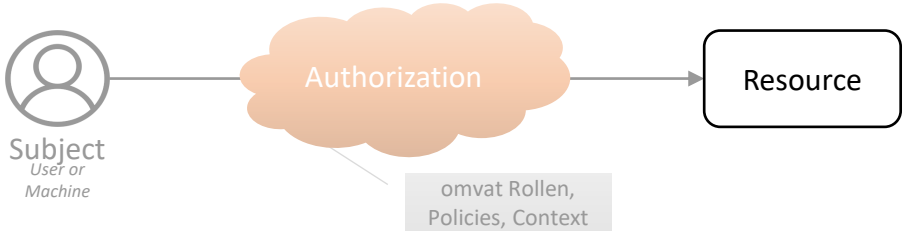
Een bijzondere toegevoegde waarde van het publiceren van data als Linked Data is het federatief kunnen bevragen van data. Datasets zijn vrijwel altijd opgeslagen in silo's. Mbv Linked Data kunnen datasets gemakkelijk aan elkaar gerelateerd worden en op een federatieve manier in één keer bevraagd worden. De Kadaster Knowledge Graph is hier een voorbeeld van met de BRK, BGT, BRT en BAG gekoppelde datasets.

De koppeling van elke graph (of resource) vereist de opname van een gedeeld identificerend sleutelveld of attribuut (verder uitgelegd in de achtergrond). In onderstaande figuur zijn deze attributen of sleutelvelden gedefinieerd als een BSN- of KvK-nummer waarmee drie verschillende graphs met elkaar in verband kunnen worden gebracht. Met behulp van deze velden kan één enkele query worden geschreven, verwijzend naar elke SPARQL API en het relevante sleutelveld, om zo de benodigde informatie uit drie bronnen op te halen.

In deze context is de resource … elke triple, een hele dataset, de gekoppelde datasets, wellicht een selectie uit een dataset …
De resource is hier niet eenduidig.



Autorisatie: Resource (5/5)

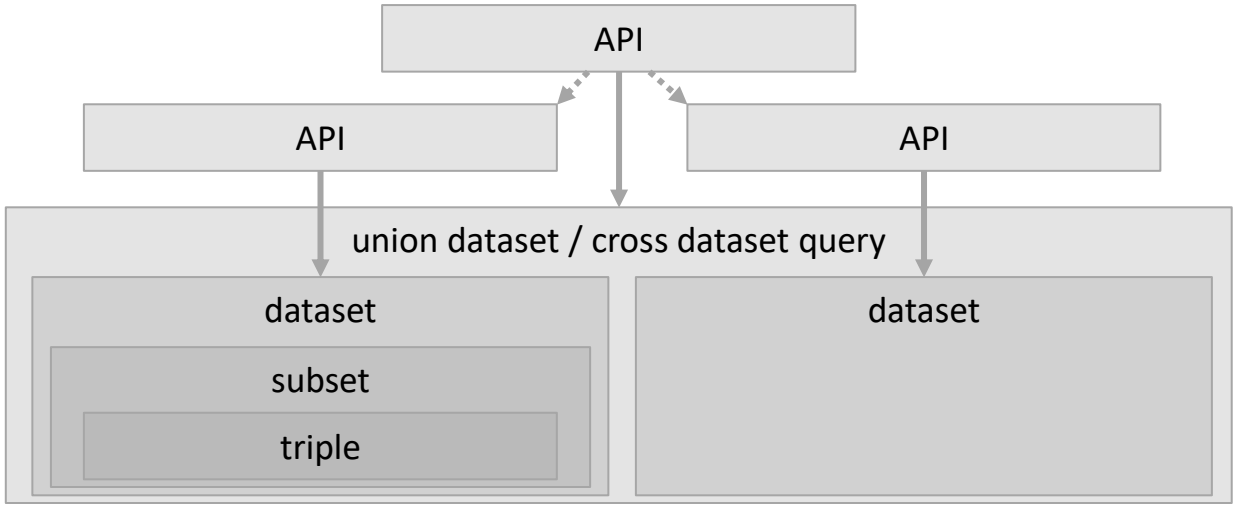


Samenvattend: een **resource** is dus nog niet zo eenvoudig. Er is hier ook een gelaagdheid in te beschrijven.

Het meest elementaire onderdeel is een triple. Gerelateerde triples vormen een graph voor een subset van gerelateerde gegevens, traditioneel vaak een object of tabel genoemd. Alle triples en graphs binnen één context vormen een dataset. Zo'n dataset is ontsloten met een bijbehorende API, wat deze in een silo plaatst. Wanneer een dataset relaties naar andere datasets bevat, kunnen refereerde datasets via hun eigen API bevraagd worden. Een selectie uit één dataset wordt een subset genoemd. Gerelateerde selecties over meerdere datasets heen wordt ook subset genoemd en soms superset.

In het kader van het afschermen van gegevens en het autoriseren van de juiste mensen (en machines) met de juiste rechten om deze afgeschermde gegevens juist wél op te vragen, is deze gelaagdheid van belang. Afscherming en autorisatie in de technologie van nu, voornamelijk REST API's, is binair: je hebt toegang of je hebt het niet. En dat is dan voor de gehele API. Om toch onderscheid en variaties te kunnen doen, worden meerdere API's gepubliceerd voor specifieke doeleinden of doelgroepen.

Met SPARQL API's is het mogelijk om vrije queries (vragen) te stellen per dataset/-silo of over datasets heen. Autorisaties en variaties in doelgroepen is hierin echter veel ingewikkelder. Dit is precies het onderwerp van dit project.



Verdiepende requirements

De voorgaande slides tonen de complexiteit van autorisatie en geven (suggesties voor) requirements. Daar bovenop zijn nog verdiepende requirements te benoemen die te maken hebben met flexibiliteit en complexiteit in data in het kader van afscherming en autorisatie.

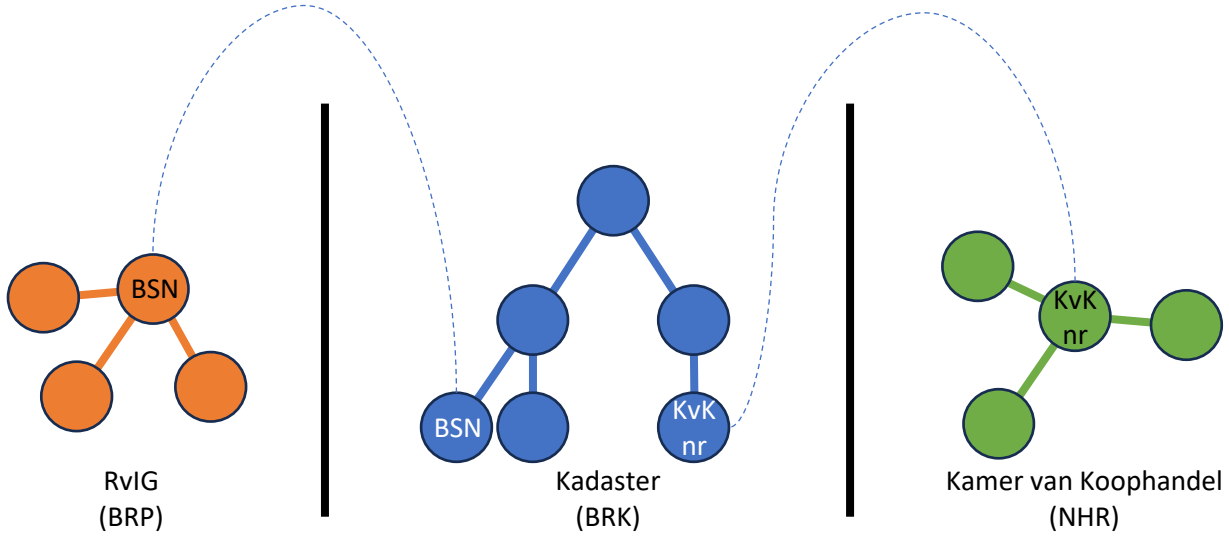
In de context van een federatief datastelsel zijn verschillende silo-datasets aan elkaar gekoppeld op basis van sleutelvelden. Een voorbeeld van sleutelvelden zijn het BSN-nummer en het KvK-nummer in de verbinding van de BRK naar de BRP en het NHR.

Omdat deze databronnen gevoelige informatie bevatten, moet autorisatie op deze gegevenssets worden toegepast. Wie toegang heeft tot informatie (en tot hoeveel informatie) hangt af van de context. Gegevens kunnen bijvoorbeeld niet openbaar beschikbaar zijn op basis van beleid en wetten die verband houden met de bescherming van persoonsgegevens of om commerciële redenen, zoals gevoelige informatie die verband houdt met bedrijfsprocessen. Zelfs binnen deze contexten kunnen bepaalde rollen meer toegang krijgen tot informatie dan andere rollen en alleen voor bepaalde doeleinden.

Voor de flexibiliteit van het gebruik en hergebruik van data is per gebruik vrij kunnen bevragen (query'n) of navigeren door het datastelsel zeer gewenst / noodzakelijk.

Verdiepende requirements zijn dan ook:

- Vrije query mogelijkheden
- Toegang tot een subset van gegevens
- Toegang tot data in een bepaalde richting en niet andersom



Toegang tot een Subset

‘Data partioning’ is een concept dat bij databasebeheer wordt gebruikt om grote datasets in kleinere, beter beheersbare partities of subsets te verdelen. Dit kan zowel **verticaal** als **horizontaal** gedaan worden. Op deze kleinere subsets kunnen toegangscontrole mechanismen toegepast worden.

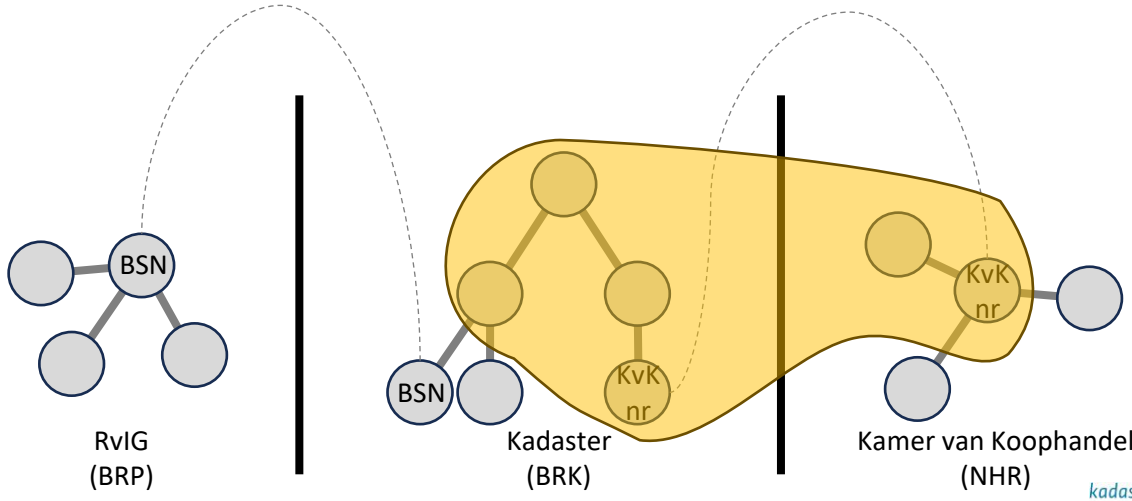
In Linked Data kan een subset ook sub-graph genoemd worden. Bovenstaande concept is dan ook voor Linked Data van toepassing. Het is mogelijk om horizontale of verticale sub-graphs te maken. Een horizontale sub-graph bevat dan niet alle triples maar een selectie van bijvoorbeeld een specifieke regio. Een verticale sub-graph bevat alleen de triples die een specifiek attribuut betreffen van een `subject` en niet alle triples van dat `subject`.

verticale subset

Perceel	eigenaar	koopsom	aandeel
Perceel 1	X	**€ 354.000**	X
Perceel 2	X	**€ 500.000**	X
Perceel 3	X	**€ 878.878**	X
Perceel 4	X	**€ 456.000**	X

horizontale subset

Perceel	Eigenaar	koopsom	aandeel
~~Perceel 1~~	~~X~~	~~X~~	~~X~~
Perceel 2	**Jan Janssen**	**€ 500.000**	**1/1**
~~Perceel 3~~	~~X~~	~~X~~	~~X~~
Perceel 4	X	X	X

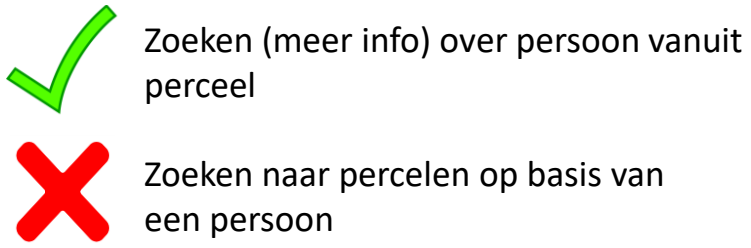


Toegang tot Data in een Bepaalde Richting

Een van de intrinsieke kwaliteiten van graphs is om op verkennende wijze door de graph te navigeren. Binnen de context van een federatief datastelsel bestaan er echter een aantal gebruiksscenario's waarin deze vrije navigatie niet gewenst is en (dus) beperkt moet worden.

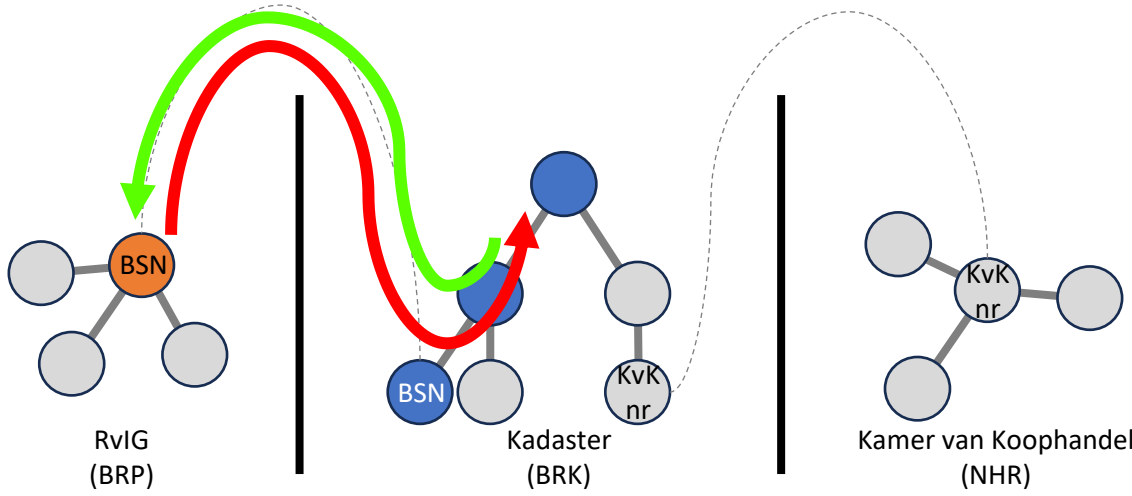
Een van de noodzakelijke beperkingen in deze context is de mogelijkheid om inverse queries uit te voeren. Met andere woorden, het is noodzakelijk om toegangscontroles op gegevens te implementeren waarbij het opvragen van informatie in een bepaalde richting van de graph mogelijk is, maar het opvragen van de omgekeerde richting niet mogelijk is.

Bijvoorbeeld: het zou mogelijk moeten zijn om te zoeken naar de eigenaar van een specifiek perceel op basis van het perceelnummer maar niet alle perceelnummers die iemand in eigendom heeft op basis van naam/BSN.



Zoeken (meer info) over persoon vanuit perceel

Zoeken naar percelen op basis van een persoon



Technologische oplossingen voor autorisatie

In de vorige sectie zijn de verschillende requirements belicht die aan autorisatie gesteld kunnen worden. Verschillende technologische oplossingen bieden verschillende kansen en uitdagingen om de doelen van Lock-Unlock te behalen: Lock de data, Unlock het potentieel. De technologische oplossingen zijn eindeloos en toch beperkt in het kader van een federatief datastelsel en Linked Data. In ons desk research zijn we diverse standaarden, voorbeelden en implementaties tegengekomen (zie achtergronden). Deze kunnen we grofweg verdelen en bundelen in de volgende selectie van technologische oplossing(srichtingen):

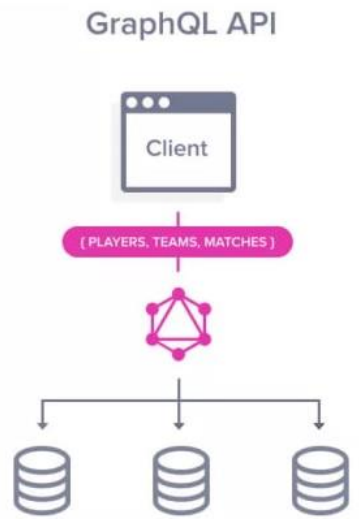
- Query Auditing
- Autorisatie in REST API
- Autorisatie in GraphQL API
- Autorisatie in Linked Data
  - Predefined Queries
  - Predefined Sub-Graphs
- Autorisatie als Linked Data

API’s: REST vs GraphQL vs SPARQL

Voordat we de inhoud van elk scenario gedetailleerd beschrijven, wordt hieronder een kort overzicht gegeven van de verschillende typen API’s die in dit rapport aan bod komen. Zie de achtergrond voor nog meer informatie.

- **REST API** (Representational State Transfer API); Bij het woord ‘API’ is de algemene lading: een REST API, een API waarin objecten worden bevraagd over HTTP en de data in JSON formaat (JavaScript Object Notation) wordt geretourneerd. REST API’s/RESTful-webservices zijn opgezet om geïsoleerd en enkelvoudige informatie beschikbaar te stellen. Elke API heeft één specifieke input en output. Relaties tussen objecten zijn in de (output) JSON als URL opgenomen welke direct gebruikt kan worden om dan het gerelateerde object op te vragen.
- **GraphQL API**; GraphQL is een nieuwe techniek die lijkt op een REST API wat betreft het resultaat, namelijk data in JSON formaat. Echter waar REST een reeks endpoints gebruikt per object, gebruikt GraphQL een enkel endpoint/gateway. Wat er opgevraagd kan worden en wat er geretourneerd wordt, is volgens een GraphQL schema gepubliceerd. Daarbij is het mogelijk om in het bevragen een selectie te maken van welke data je precies nodig hebt. Daarin is het mogelijk om meerdere objecten te combineren, zolang dat volgens het schema beschikbaar is. GraphQL is daarmee flexibeler dan REST API’s en maakt het mogelijk om integraal informatie beschikbaar te stellen.
- **SPARQL** (**endpoint**) **API**; Een SPARQL API is gebaseerd op HTTP net zoals REST API’s en GraphQL. SPARQL ondersteunt een zeer krachtige query taal waarmee één of meerdere endpoints / API’s in samenhang te bevragen zijn. SPARQL is volledig gebaseerd op Linked Data en de mogelijkheden daarvan.

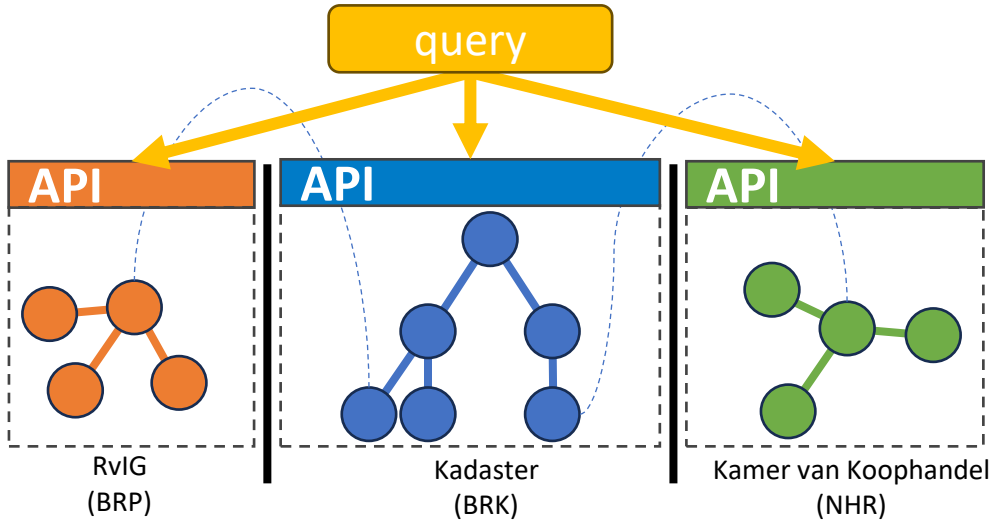
- Vergelijking: GraphQL is een mix tussen REST API’s en SPARQL. Het biedt vergelijkbare mogelijkheden als REST API’s maar met meer mogelijkheden. Waar SPARQL volledig is gebaseerd op Linked Data, voldoet GraphQL slechts ten dele aan (de) Linked Data (principes). Bijvoorbeeld kunnen in een SPARQL endpoint / API meerdere schema’s gecombineerd worden, terwijl bij GraphQL slecht één expliciet schema van toepassing is.



Query Auditing

Beschrijving
Deze technologische oplossing is geen oplossing voor autorisatie. Bij alleen query auditing kan de gebruiker namelijk alle data bevragen en worden slechts de queries met bijbehorende context (denk bijv. aan de gebruiker en meegegeven referentie naar doelbinding) opgeslagen. Vervolgens kan er *achteraf* handmatig of wellicht automatisch bepaald worden of er onrechtmatige vragen/queries zijn gesteld en of dat er onterecht toegang is verkregen tot gesloten gegevens.

Deze technologische oplossing wordt verder verkend in track 2 van het Lock-Unlock project. Hierbij willen we onder andere kijken naar de GEMMA verwerkingenlogging.



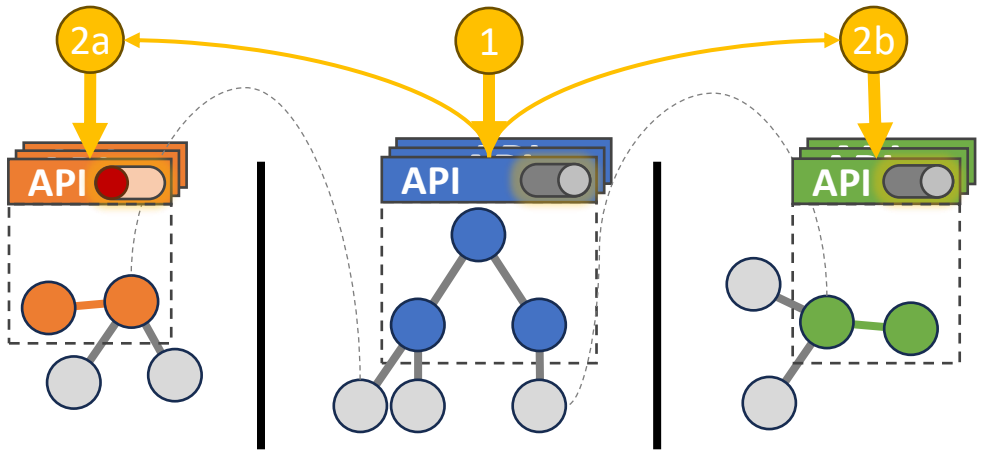
Kansen voor Lock-Unlock
- Om te kunnen verantwoorden dat (met name persoons)gegevens op een juiste manier verwerkt worden, is het nodig om de verwerking te bewaren. Dat gebeurt door de queries te loggen.
- Het loggen van queries en context biedt altijd de mogelijkheid om later onregelmatigheden op te sporen.

Uitdagingen voor Lock-Unlock
- Query Auditing is geen oplossing voor autorisatiecontroles.
- Uitdagingen rondom het kunnen toepassen van de GEMMA verwerkingenlogging gaan we in track 2 beproeven.
  - De standaard wordt doorontwikkeld wat onzekerheid geeft rondom de stabiliteit van de standaard.
  - Query logging is niet gestandaardiseerd en mogelijkheden in producten zijn nog onduidelijk.

Autorisatie in REST API

Beschrijving
Bij een REST API wordt data in JSON formaat (JavaScript Object Notation) geretourneerd over HTTP. Dit laatste onderdeel, namelijk HTTP, geeft direct de mogelijkheden van deze oplossingsrichting. Alles wat met HTTP kan, kan met REST API's. Dit is gestandaardiseerd en is op dit moment breed beschikbaar in vele platformen op een uniforme wijze. De standaard die hier voornamelijk een rol speelt, is OAuth2. Hierin is het mogelijk om met username/password, een token als een 'API Key' of een certificaat authenticatie te doen en rollen mee te geven om autorisatie af te dwingen.

REST API's zijn voor specifieke ontsluiting van data. Een REST API kan een gehele dataset betreffen. Of juist een subset, waarbij een verticale subset gemakkelijker is dan een horizontale subset. Een REST API is altijd specifiek en daarom wordt voor elke toepassing of ontsluiting een nieuwe API ontwikkeld. Een samengestelde dataset of meerdere datasets betekent een nieuwe API. Als een andere subset dan in de beschikbare API's gewenst is, betekent dat de ontwikkeling van een nieuwe API.

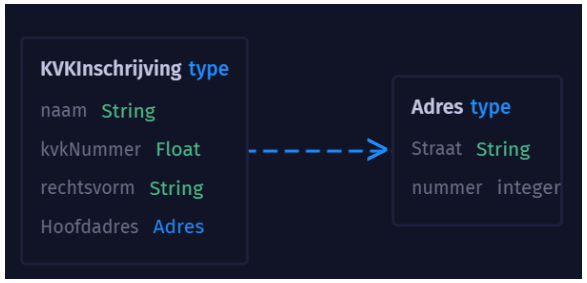


Kansen voor Lock-Unlock
Veel data is al dmv een API beschikbaar, waarbij dit al vaak REST API's betreft. Soms nog voorlopers als SOAP. De techniek is gestandaardiseerd met HTTP (oa OAuth2) en in vele technologie stacks beschikbaar.

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	✗

Uitdagingen voor Lock-Unlock
Afscherming van data is op het niveau van de REST API. Autorisaties geven toegang of niet. Het is aan of uit, ja of nee. Er zijn geen gradaties van autorisaties mogelijk. Er zijn geen mogelijkheden om andere datasets of subsets op te vragen dan dat er in de API's (collectie) wordt aangeboden. Er is orkestratie noodzakelijk om meerdere datasets over meerdere API's te bevragen. Navigeren en vrij bevragen is niet mogelijk.

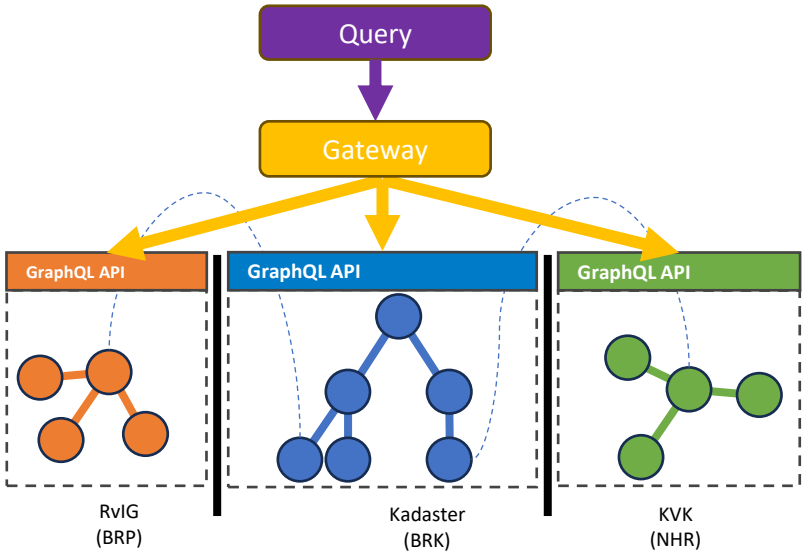
Autorisatie in GraphQL



Beschrijving
GraphQL is gebaseerd op een voor-gedefinieerd schema (zie voorbeeld). Dit schema is een object georiënteerde benadering waarin objecten en de relaties beschreven zijn. Welke objecten beschikbaar zijn, welke relaties en in welke richting die relaties mogelijk zijn, staat gedefinieerd in het schema. Dit geeft mogelijkheden voor het afschermen van data. Er kunnen filters toegepast worden, zowel als gebruiker en in de API. Bijvoorbeeld: *geef alle KVKInschrijvingen met rechtsvorm "BV"*. Of in het voorbeeld hiernaast: *vanuit KVKInschrijving kan wel het adres opgevraagd worden maar niet andersom*.

Met een GraphQL Gateway is het mogelijk om meerdere samenhangende datasets tegelijkertijd te bevragen. De Gateway distribueert de query en stelt de resultaten samen tot een samenhangend geheel (schema stitching).

Er zijn verschillende manieren om de objecten in het schema te beveiligen. Net zoals met REST API's kan wel of geen toegang verleent worden tot de hele API. Met GraphQL is het ook mogelijk om in het schema te configureren welke rollen toegang krijgen tot objecten of attributen daarvan. Waar in REST API's meerdere API's beschikbaar gemaakt zouden worden, kan dat met GraphQL in één GraphQL API/Gateway, uiteraard beperkt tot het betreffende schema.



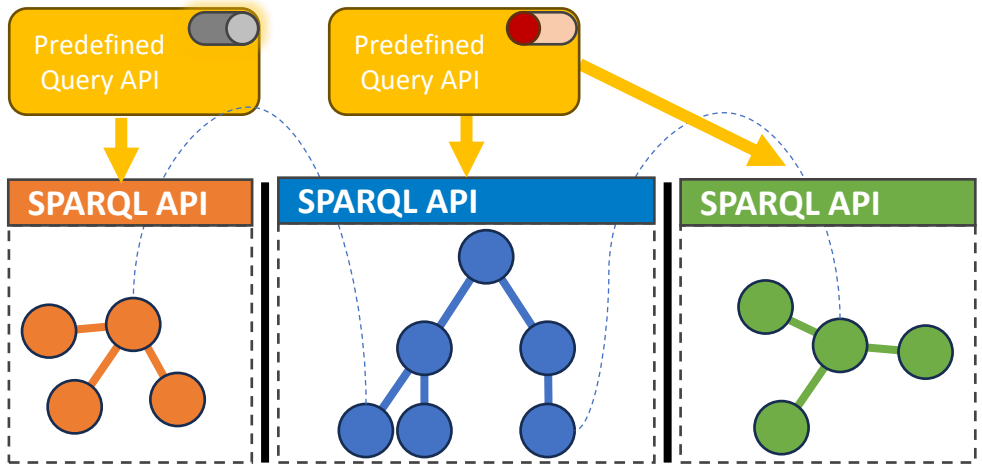
Kansen voor Lock-Unlock
Verfijnde autorisatie op schema niveau mogelijk. Binnen het schema zijn (vrije) queries mogelijk. Gateways ondersteunen federatieve queries.

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	~

Uitdagingen voor Lock-Unlock
GraphQL kan aangeboden worden op basis van een Linked Data architectuur, maar volgt niet volledig de Linked Data principes.

Autorisatie in Linked Data | Predefined Queries

Beschrijving
Het is mogelijk om in een SPARQL endpoint vooraf queries te definiëren. Hoewel de basis vrije bevraging is, worden alleen van tevoren gecontroleerde queries toegestaan. De vooraf gedefinieerde queries kunnen natuurlijk variëren per rol en wellicht kunnen die achter een gewone REST API geïmplementeerd worden. De uitwerking hiervan is dat er wél Linked Data technieken gebruikt worden, maar dat de oplossing dichter bij het REST API's scenario komt mbt gesloten data. Voor open data is een open (dus apart) SPARQL endpoint beschikbaar.



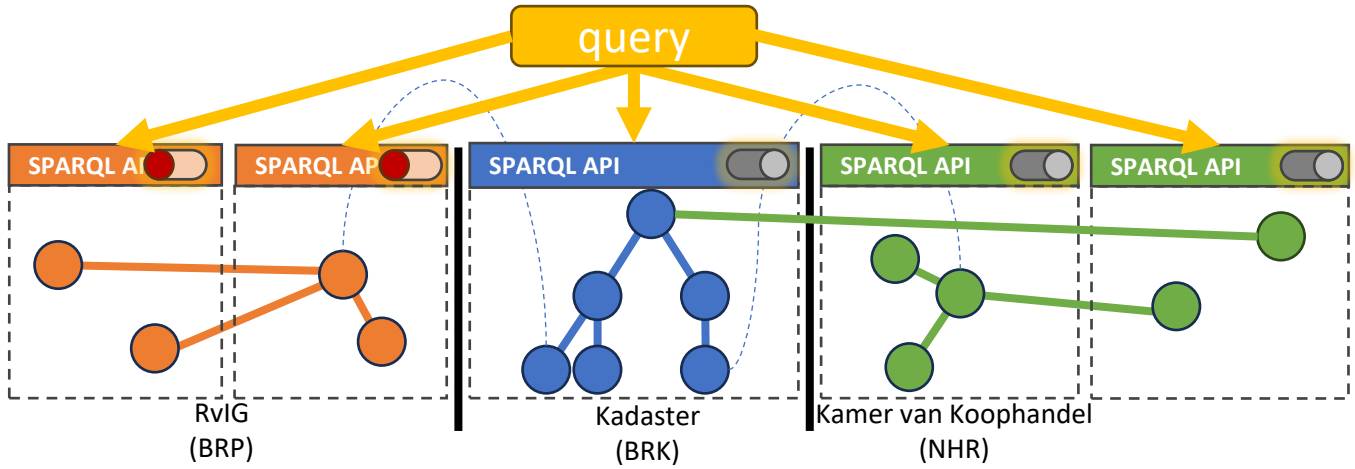
Kansen voor Lock-Unlock
Deze oplossing maakt het mogelijk om Linked Data technieken te gebruiken en data af te schermen. De Predefined Queries zijn een eenvoudige manier om afscherming mogelijk te maken á la REST API's.

Uitdagingen voor Lock-Unlock
Er is geen standaard voor het vastleggen Predefined Queries. Dit is daarom afhankelijk van de technische oplossing of dit beschikbaar is of niet. Deze oplossing levert nauwelijks meer dan de mogelijkheden bij REST API's. Het is namelijk voor de gebruiker niet meer mogelijk om zelf queries op te stellen en te navigeren door de knowledge graph voor gesloten data.

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	✗

Autorisatie in Linked Data | Predefined Sub-Graphs

Beschrijving
Zoals er met REST API's meerdere API's beschikbaar worden gesteld voor de verschillende doelgroepen en autorisaties, is het in Linked Data mogelijk om meerdere sub-graphs te definiëren / publiceren. Dit geeft vergelijkbare mogelijkheden wat betreft afscherming en toegang en biedt in tegenstelling tot Predefined Queries wél de mogelijkheid tot vrij query'n.



Kansen voor Lock-Unlock
Door het werken met Predefined Sub-Graphs is het mogelijk om toegang op subset niveau te regelen, zowel horizontaal (gebruiker X mag alleen bij die specifieke instanties) als verticaal (gebruiker Y mag wel de koopsom zien maar niet de eigenaar). Op die sub-graphs kan de gebruiker vervolgens vrij query'n en ook federatieve queries doen.

Horizontale subset	✓
Verticale subset	✓
Richting beperken	~
Vrije query	~

Uitdagingen voor Lock-Unlock
Aangezien bij deze oplossing niet naar de querykant zelf wordt gekeken, is het slechts beperkt mogelijk om de richting van de vraag te beperken. Daarnaast zijn de mogelijkheden beperkt tot de beschikbare Predefined Sub-Graphs. Dat betekent voorbereiding en beperking in afscherming. De sub-graphs zijn afgeleiden van de originele totale dataset / graph. Het mechanisme voor het beschikbaar maken van de sub-graphs bepaalt actualiteit, beschikbaarheid, etc.

Andere benadering

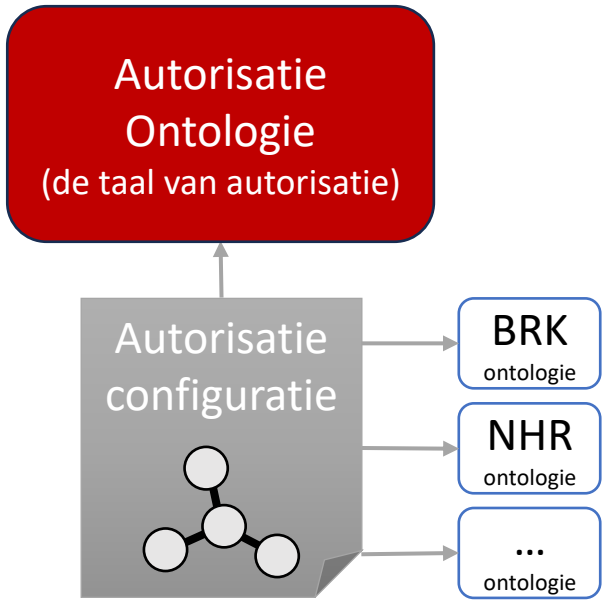
Autorisatie als Linked Data

Beschrijving

De autorisatie als Linked Data is een geheel andere benadering dan voorgaande technologische oplossingsmogelijkheden. Het houdt in dat de autorisatie zelf als Linked Data wordt uitgedrukt. Daarvoor zijn twee onderdelen nodig:

1. De **autorisatie ontologie** is een "woordenboek" welke de autorisatie terminologie bevat. Denk aan *Gebruiker*, *Rol*, *Toegestaan, Autorisatie hiërarchie,* etc. Zo'n vocabulaire zou gestandaardiseerd moeten zijn. Deze bestaat op dit moment nog niet, maar sluit goed aan bij de wereld van Linked Data.
2. De **autorisatie configuratie** bevat de specifieke autorisatie regels voor een resource. In deze configuratie wordt autorisatie metadata toegevoegd (verrijkt) aan bestaande ontologie van de resource. Bijvoorbeeld *Binnen de BRK is het anonieme Gebruikers niet toegestaan het LaatsteKoopsom Predicate op te vragen.*

Deze twee delen samen vormen een autorisatiebeleid dat door machines te interpreteren en gebruiken is. Door dit autorisatiebeleid uit te drukken in Linked Data kunnen dezelfde technieken gebruikt worden als voor het bevragen van de Linked Data zelf.



Kansen voor Lock-Unlock

- Autorisaties kunnen gerelateerd worden aan de reden van toegang (of niet) naar data. Dit is in bestaande technologische oplossing niet of nauwelijks mogelijk. Hier liggen enorme kansen voor vooraf en achteraf controle en uitleg.
- Autorisatie wordt in dezelfde "technische taal" uitgedrukt als de data.
- Uitgebreide evaluatie van vrije queries, data partionering, etc, etc wordt hierdoor mogelijk.
- Door de standaardisatie is meervoudig gebruik in meerdere implementaties en oplossingen mogelijk van autorisatiebeleid.

Uitdagingen voor Lock-Unlock

- Er bestaat nog geen gestandaardiseerde ontologie voor autorisatie. Hiervoor zullen wij zelf de eerste aanzet moeten doen (er zijn wel enkele papers beschikbaar met suggesties en voorbereidend werk).
- Er zijn (dus) ook nog geen standaard implementaties voor deze oplossingsrichting.

Andere benadering

Autorisatie als Linked Data | Implementaties

- **Autorisatie Configuratie geeft Predefined Queries**
  In deze implementatie wordt de Autorisatie Configuratie vertaald naar Predefined Queries. Dit zou een standaardisatie zijn van de nu specifieke invulling die technische oplossingen bieden voor het definiëren van Predefined Queries.

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	✗

- **Autorisatie Configuratie geeft Predefined Sub-Graphs**
  In deze implementatie wordt de Autorisatie Configuratie vertaald naar Predefined Sub-Graphs. Dit geeft mogelijkheden in het beheer en complexiteit van de Predefined Sub-Graphs oplossingsrichting. Het zou ook voor deze richting standaardisatie bieden.

Horizontale subset	✓
Verticale subset	✓
Richting beperken	~
Vrije query	~

- **On the fly query evaluatie**
  In deze implementatie wordt elke query ‘on the fly’ volledig ontleed en ieder onderdeel hiervan wordt afzonderlijk geautoriseerd in de context van de query. Gebruikers kunnen zelf hun query opstellen en herstructureren. Dit is het ultieme scenario. Echter bestaan er nog geen implementaties van en zal onderzoek en beproeving moeten uitwijzen tot welk niveau dit haalbaar is (= op welke termijn, tegen welke kosten).

Horizontale subset	?
Verticale subset	?
Richting beperken	?
Vrije query	?

Overzicht technologische oplossingen

Query Auditing

Bij Query Auditing worden de queries en context gelogd zodat er achteraf bepaald kan worden of de queries (on)rechtmatig zijn.

GEMMA verwerkingenlogging

geen autorisatie oplossing

Horizontale subset	✗
Verticale subset	✗
Richting beperken	✗
Vrije query	✓

Autorisatie in REST API

Hierbij is de afscherming van data op het niveau van de REST API. Het is aan of uit. Het vrij bevragen van de data is niet mogelijk en voor elke subset en/of richting moet een aparte API worden aangeboden.

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	✗

Autorisatie in GraphQL API

Met GraphQL is het mogelijk om meerdere datasets tegelijkertijd te bevragen op basis van een voor-gedefinieerd schema (incl. richting). Het is mogelijk om per dataset wel/geen toegang te verlenen, maar dit kan ook op object en attribuut niveau.

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	~

Autorisatie in Linked Data | Predefined Queries

Door vooraf queries te definiëren en deze te relateren aan de typen gebruikers is het mogelijk om fijnmaziger toegang te verlenen. Hierdoor wordt wel ingeleverd op het vrij bevragen en lijkt het op het REST API scenario.

NUTS | iShare in Topsector Logistiek

Horizontale subset	~
Verticale subset	✓
Richting beperken	✓
Vrije query	✗

Autorisatie in Linked Data | Predefined Sub-Graphs

Hierbij worden verschillende sub-graphs gegeneerd uit één of meerdere datasets. Afhankelijk van de context wordt bepaald welke gebruiker bij welke sub-graph(s) mag. Deze zijn vervolgens vrij te bevragen.

NUTS | iShare in Topsector Logistiek | SOLID | Personal Health Train

Horizontale subset	✓
Verticale subset	✓
Richting beperken	~
Vrije query	~

Autorisatie als Linked Data

Dit is een geheel **andere benadering** waarbij de autorisatie zelf als Linked Data wordt uitgedrukt. Dit autorisatiebeleid is door machines te interpreteren en kan resulteren in verschillende implementaties waarbij de autorisatie mogelijk zelfs “on the fly” kan worden toegekend.

Horizontale subset	?✓
Verticale subset	?✓
Richting beperken	?✓
Vrije query	?✓

Voorstel beproevingen Lock-Unlock

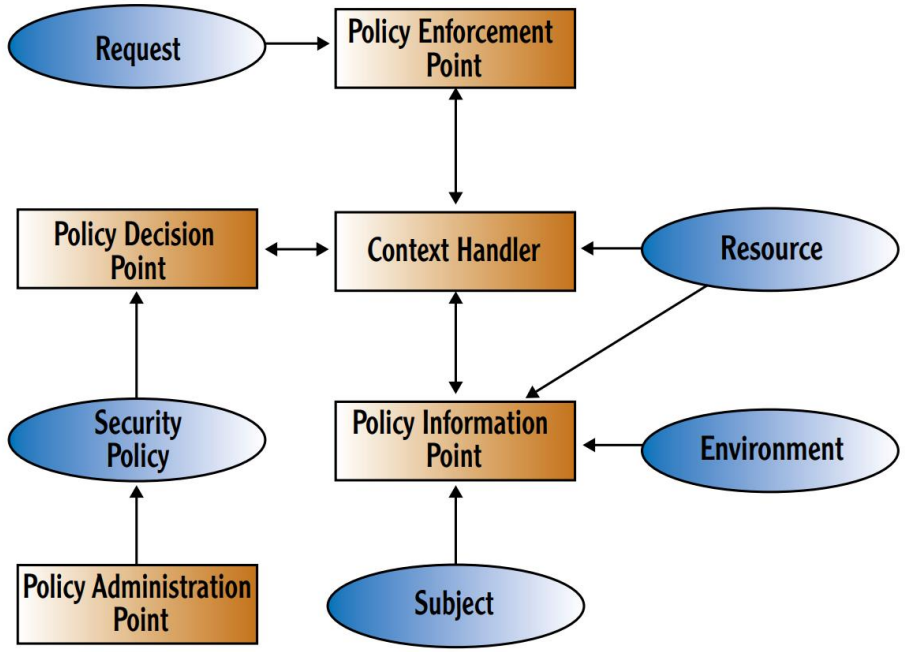
In de vorige slides zijn verschillende technologische oplossingen geschetst inclusief de bijbehorende kansen en uitdagingen voor Lock-Unlock. Op basis van ons desk research en de vergelijking van de technologische opties stellen we het volgende vervolg voor van track 1 binnen Lock-Unlock:

- In een of meerdere demonstrators (data stories) onderzoeken we de kansen en uitdagingen van zowel **Autorisatie in REST API** als **Autorisatie als Linked Data**.
- De verwachting is dat autorisatie in REST API's relatief snel geïmplementeerd en getest kan worden. Dit laat dan gelijk de kansen maar ook beperkingen zien: het zal lastig worden om op een fijnmazige manier autorisaties uit te delen en navigeren door de graph is niet mogelijk. Aangezien REST API's vaak worden gezien als de huidige oplossing (en soms de heilige graal) is het goed om de mogelijkheden voor autorisatie hierin te beproeven.
- De autorisaties als Linked Data is een meer uitdagende oplossingsrichting. Ook al is dit een richting die nog weinig verkend en geïmplementeerd is, verwachten we in een relatief korte tijd een aantal autorisatie opties te kunnen demonstreren. Hierbij willen we kijken naar de implementaties zoals geschetst in de mogelijke implementaties van deze richting. Het voordeel van deze oplossing is dat de kracht van Linked Data behouden blijft.

Wat betreft GraphQL: GraphQL is een stap terug tov Linked Data en zien we als te beperkend voor het vrij kunnen bevragen van en navigeren door de data. We willen daarom eerst het Linked Data pad verkennen. Wel zou GraphQL een goed alternatief zijn als we met het Linked Data pad onvoldoende voortgang boeken.

Achtergrond: XACML

- OASIS Committee XACML (link), the eXtensible Access Control Markup Language; een taal voor het uitdrukken van regels voor toegangsverlening. Deze regels zijn technologie onafhankelijk en gescheiden van de gegevens waarop ze betrekking hebben. De taal ondersteunt Attribute Based Access Control (ABAC).
- Artikel Java Magazine mei 2011 (link)



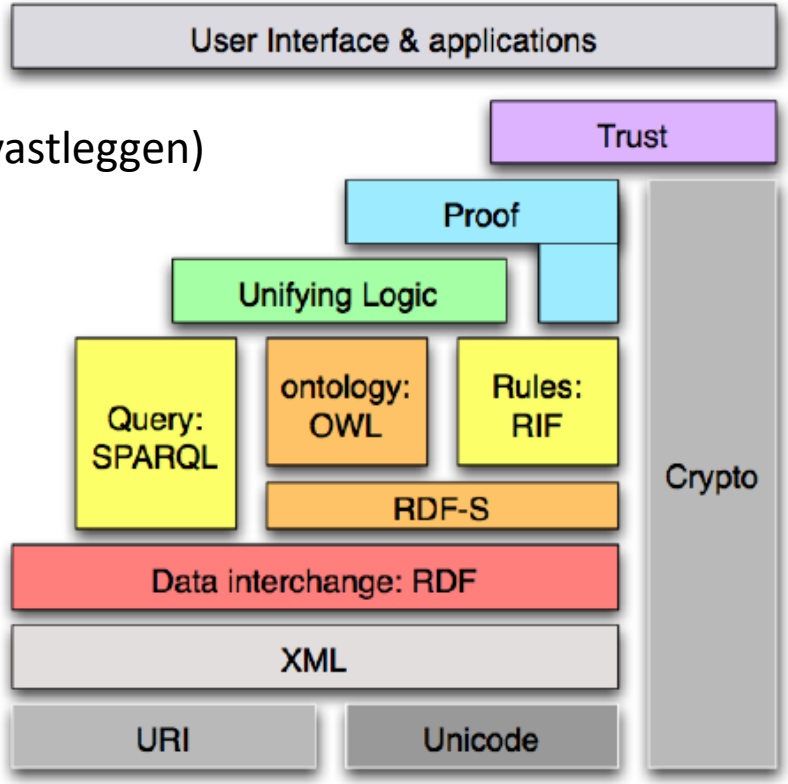
Figuur 1: Architectuur van een XACML-systeem.

Achtergrond: methodes voor access control

- Role Based Access Control (RBAC): zoals de naam al aangeeft wordt hierbij met verschillende rollen gewerkt. Dit maakt het mogelijk om voor één bepaalde groep gebruikers gelijk alle autorisatierechten te verlenen. Je hoeft dus geen rechten toe te kennen aan individuele gebruikers.
- Attribute Based Access Control (ABAC): deze methode is nog iets fijnmaziger en hierbij wordt autorisatie wel/niet verleend op basis van de attributen. Per attribuut kan worden aangeven wanneer iemand hier wel/niet bij kan; dit is afhankelijk van de gebruiker, kenmerken van het attribuut en andere context.
- Bovenstaande methodes zijn de meest bruikbare en bekende methodes voor access control. Andere methodes zijn bijv.: Mandatory Access Control, Discretionary Access Control, Rule Based Access Control & History Based Access Control.

Achtergrond: Linked Data

- Internationale set van Linked Data standaarden van het W3C (https://www.w3.org/)
- Van een web van gelinkte documenten naar een web van (gelinkte) data
- Een Resource is een URI waarover meer informatie vast gelegd kan worden
- Basis principes
  - URI's voor globale webgebaseerde ID's
  - RDF taal om URI's /Resources te beschrijven (meer informatie vastleggen)
  - Resources kunnen verwijzen naar andere resources (linked)
  - SPARQL voor bevragingen
  - Formele logica om informatie af te leiden
  - Nieuwe basis voor data-gedreven applicaties
- Meer informatie via
  - https://www.w3.org/wiki/LinkedData
  - https://www.w3.org/TR/rdf11-concepts/
  - https://www.w3.org/TR/sparql11-query/



Achtergrond: Hoe de silo’s koppelen?

Linked Data

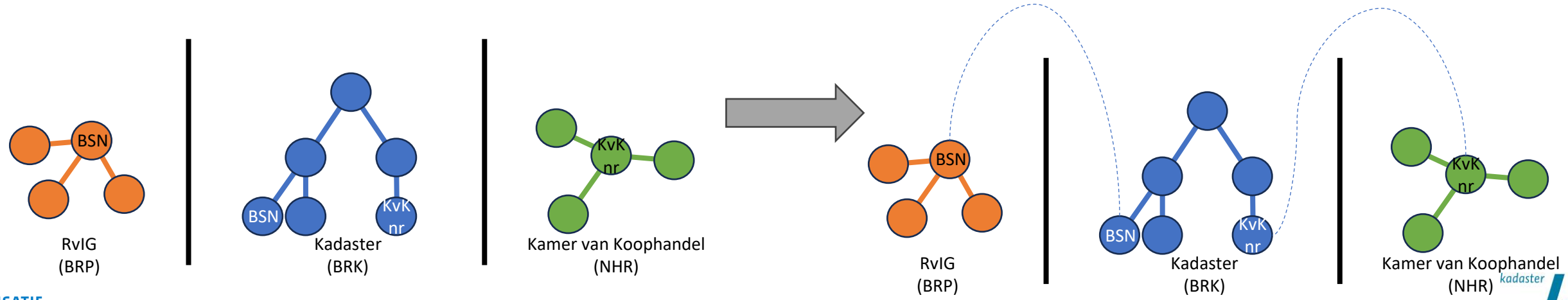
Herbruik van uri's van andere silo's levert een directe koppeling op. Dit wil zeggen dat registers en andere datasets dezelfde uri's gebruiken voor dezelfde 'dingen'. Bijvoorbeeld KvK gebruikt Kadaster's BAG uri's om te verwijzen naar gebouwen, Kadaster gebruikt BRP uri's om te verwijzen naar personen, etc.

Top-level ontologie

Silo's maken gebruik van een top-level-ontologie waarin koppelvelden gedefinieerd zijn. Denk hierbij aan één uri voor het kenmerk BSN nummer of BAG ID. Het idee is dat registers en andere datasets deze uri's gebruiken voor koppelvelden.

Linksets/koppeltabellen

Een ander mechanisme is een koppeltabel waarbij één waarde direct gekoppeld is aan een andere waarde. Denk hierbij bijvoorbeeld aan een intern persoons ID en een BSN nummer. Hiermee ontstaan links tussen datasets.



Achtergrond: REST vs GraphQL vs SPARQL

Hieronder wordt een kort overzicht gegeven van de verschillende voor- en nadelen van de typen API's die in dit rapport aan bod komen.

REST API

Geïsoleerd en enkelvoudig informatie beschikbaar te stellen

Pros	Cons
• Zeer gangbaar • Binaire authorisatie is gestandaardiseerd	• Silo gedachte • API orkestratie nodig voor samenhang • Slechte schema definities ontsluiting • Geen vrije (schema loze) query mogelijkheden

GraphQL API

Informatie beschikbaar stellen op basis van gefixeerde schema's met silo overkoepelende mogelijkheden

Pros	Cons
• Minder silo gedachte • Mogelijkheden tot samenhang • Verfijnde authorisatie mogelijk • Gangbaarder / bekender dan SPARQL	• Matige schema definities ontsluiting • Geen vrije (schema loze) query mogelijkheden • Minder gangbaar dan REST

SPARQL Endpoint / API

Integraal & uitbreidbaar informatie beschikbaar stellen op basis van schemaloze queries (alles kan bevraagd worden)

Pros	Cons
• Geen silo gedachtes • Hoge samenhang • Zeer goede schema definities ontsluiting • Schemaloze query mogelijkheden	• Verfijnde autorisatie mogelijkheden moeten nog ontwikkeld worden • Minder gangbare oplossing onder developers

meer 'vraagkracht' en meer 'metadatagedreven'

(meer vrijheid in bevragen, ook cross dataset)

(meer ondersteuning vanuit de data in begrip en mogelijkheden)

Achtergrond: Bestaande implementaties

Een van de onderdelen van het desk research was de beoordeling van bestaande oplossingen voor toegangscontrole voor Linked Data bronnen. De lijst van implementaties is niet uitputtend, maar geeft wel een beeld van welke implementaties we zijn tegengekomen.

In de volgende slides zullen deze implementaties worden besproken en wordt ingegaan op de kansen en uitdagingen voor dit project.

Hierbij komen de volgende implementaties aan bod:

- PoC Topsector Logistiek & iSHARE
- Personal Health Train
- Nuts Bolt voor KIK-V
- Solid Project
- GEMMA verwerkingenlogging

Implementatie | PoC Topsector Logistiek & iSHARE

Probleem context
In de logistieke sector bestaat er een keten van stakeholders die allemaal toegang nodig hebben tot informatie over een bepaald project. Deze informatie is vaak gevoelig, vooral in het kader van concurrentie. Daarnaast heeft de overheid verantwoordelijkheden waarvoor ook data nodig is, maar dat betekent niet toegang tot de volledige dataset(s). Het is dus belangrijk om ervoor te zorgen dat verschillende belanghebbenden op verschillende tijdstippen toegang hebben tot verschillende subsets van informatie. Door de verschillende belangen zijn verschillende oplossingsrichtingen mogelijk en als Proof-of-Concept uitgewerkt.

Oplossing
Voor de PoC zijn twee aanpakken voor autorisatie getest. Eén aanpak werkt obv het creëren van *Predefined Sub-Graphs*. Op basis van de context, die opgebouwd is uit query + autorisatie + doelbinding, wordt *periodiek* een dataset opgebouwd op waarvoor toegang ingeregeld is. Op deze subset kunnen vervolgens vrije queries uitgevoerd worden. Deze subset is feitelijk een kopie van de originele data die via een tweede eindpunt beschikbaar wordt gesteld. Deze oplossing past bij de informatievoorziening benodigd voor de overheidsverantwoordelijkheid / stakeholder.

De andere aanpak werkt met *Predefined Queries* die op de gehele dataset uitgevoerd kunnen worden. Aangezien voor het logistieke proces de informatiebehoefte van tevoren duidelijk is (uitgewerkt), is het mogelijk om de benodigde queries van tevoren op te stellen en te valideren vanuit het perspectief van toegang. Dit past bij de concurrerende stakeholders in de logistieke keten.

Inzichten voor Lock-Unlock
- Deze PoCs tonen aan dat er verschillende autorisatiemogelijkheden zijn voor Linked Data, die verder gaan dan de REST API oplossingen (het open-/dichtzetten) van een SPARQL endpoint.
- Volledige query mogelijkheden blijven behouden door het creëren van een nieuwe subset (Predefined Sub-Graphs) met een eigen endpoint.
- Ook de richting wordt impliciet bepaald door het selectief genereren van een subset.
- Deze implementatie laat zien dat *Predefined Queries* en *Predefined Sub-Graphs* mogelijk zijn … en ook dat dit behoorlijke impact heeft en voorbereidingen vraagt. Vrij bevragen direct op de data is nog niet zomaar mogelijk.

Zie ook
- Github reposity van de PoC

Implementatie | Personal Health Train (PHT)

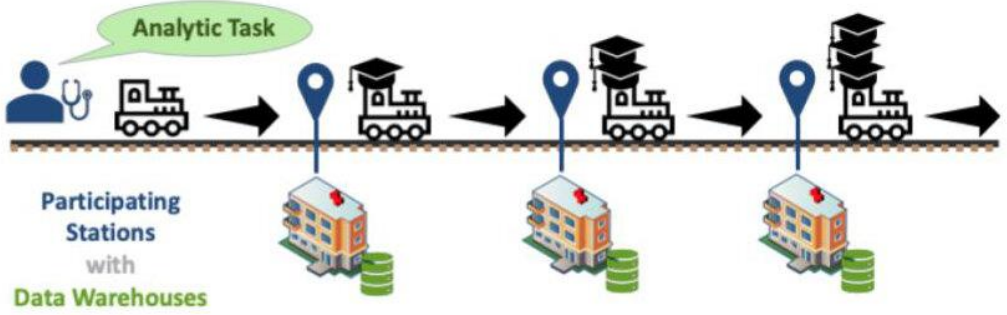
Probleem context
In het zorgdomein was een oplossing nodig voor het ontsluiten en uitvoeren van onderzoek naar gevoelige zorginformatie. Hier moesten de gegevens bij de bron blijven (bijvoorbeeld gegevens van een bepaald ziekenhuis) en mocht er geen persoonlijk identificeerbare informatie buiten deze bron worden gedeeld.

Oplossing
In deze context wordt de analogie van een trein, een station en een spoor gebruikt, waarbij de trein een bepaalde vraag of onderzoeksvraag vertegenwoordigt, het station een bepaald ziekenhuis of een bepaalde gegevensbron en het spoor de infrastructuur waarop de vraag tussen stations wordt overgedragen. De query die naar elk station wordt verzonden, vraagt om dezelfde informatie uit elke gegevensbron.

De PHT is een federatieve oplossing waarbij een ‘technical bubble’ infrastructuur wordt gecreëerd van verschillende 'stations' met een bepaalde samenwerkingsovereenkomst en onderzoekers in deze bubbel informatie van het station kunnen opvragen. In sommige gevallen kunnen stations ook worden gemaakt om zeer specifieke vragen te behandelen en daarom een subset van de informatie bevatten die beschikbaar is in de oorspronkelijke gegevensbron en/of voorbereid zijn om alleen gecertificeerde (vooraf gedefinieerde) vragen te behandelen. De methode ondersteunt ook dat de onderzoeker geen inzicht krijgt in de data op persoonsniveau maar wel de gewenste analyse kan uitvoeren. Hiervoor wordt federated learning toegepast (zie ook PET’s).

Inzichten voor Lock-Unlock
- De Personal Health Train houdt zich vrijwel uitsluitend bezig met de kwestie van horizontale partitie. Vanwege grote geografische scheiding is er vaak geen overlappende informatie tussen stations omdat patiënten in het ene ziekenhuis zeer zelden patiënten in een ander ziekenhuis zijn.
- De stations bevatten FAIR (Findable, Accessible, Interoperable and Reusable) data en per element is vastgelegd wat een trein ermee mag doen (autorisatie).
- De implementatie biedt concrete voorbeelden van hoe samenwerking tussen partijen kan worden beheerd en hoe vooraf gedefinieerde zoekopdrachten kunnen worden gebruikt om informatie naar subsets van gegevens te beheren.
- Verticale partities worden in deze implementatie niet goed ondersteund.

Zie ook
- Open source tech stack: vantage6
- Personal Health Train: https://www.dtls.nl/fair-data/personal-health-train/
- Horizontale en vertical partities: (zie Azure Docs)



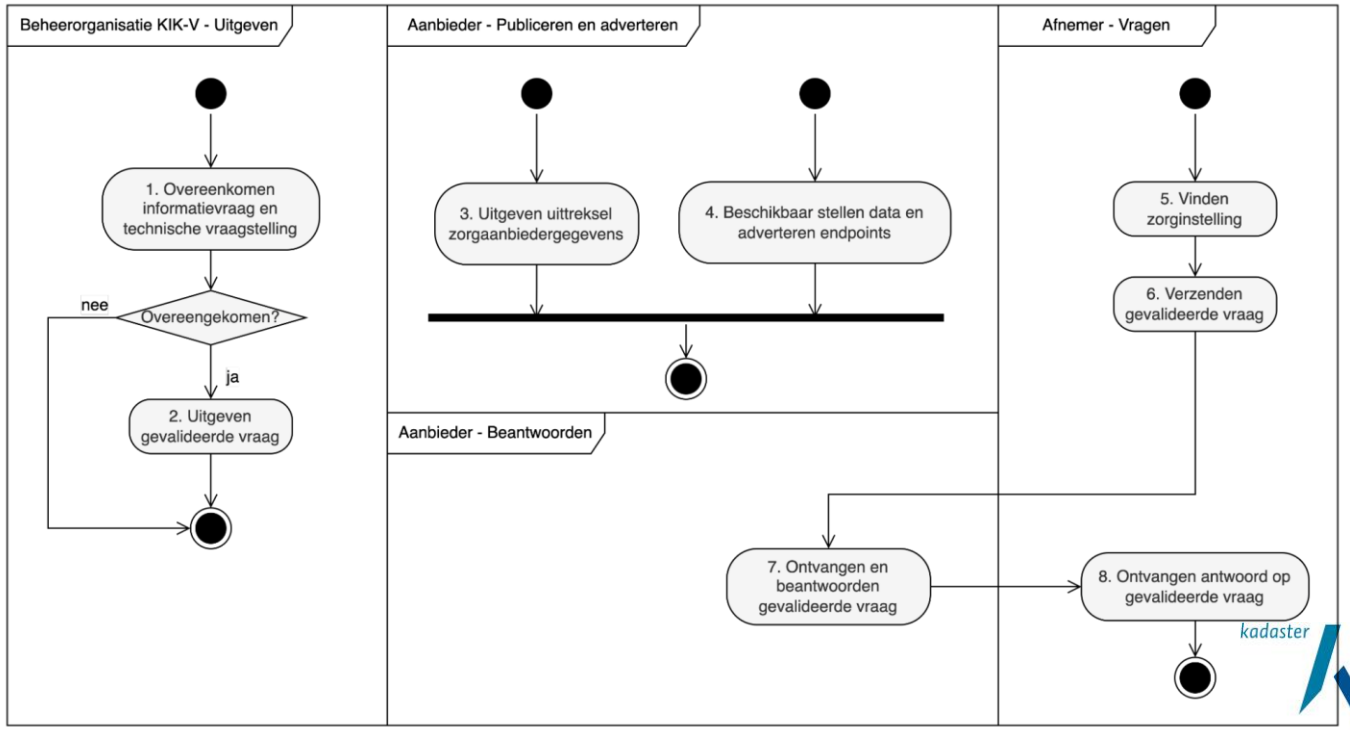
Implementatie | Nuts Bolt voor KIK-V

Probleem context
KIK-V, Keteninformatie Kwaliteit Verpleeghuiszorg, is een initiatief waarin instellingen voor verpleeghuiszorg (als aanbieder) en andere partijen met een publieke taak (als afnemer) samenwerken om de gegevensuitwisseling over kwaliteit en bedrijfsvoering te verbeteren. Het doel is het stroomlijnen van de uitwisseling van informatie, het beter afstemmen van nieuwe uitvragen en het meer hergebruiken van informatie.

Oplossing
Nuts is een samenwerkingsverband van partijen in de zorg om tot een breed gedragen, open, decentrale infrastructuur te komen ten behoeve van de uitwisseling van gegevens in de zorg en het medische domein. “Bolt” is de Nuts-term voor een concrete toepassing van de Nuts-standaarden om een tastbare use-case in de zorg mogelijk te maken. De Nuts standaarden focussen zich vooral op authenticatie en daarbij wordt o.a. gewerkt met Authorization Credentials die aangeven welke data een gebruiker mag inzien. Dit zorgt ervoor dat gebruikers dus niet alle data kunnen bevragen. Daarnaast is in de implemenatie voor KIK-V ook gewerkt met gevalideerde vragen (predefined queries). Indien de gebruiker een credential heeft voor die gevalideerde vraag dan kan die de query uitvoeren.

Inzichten voor Lock-Unlock
- In de Nuts Bolt implementatie voor KIK-V wordt zowel gewerkt met autorisaties op data- als query niveau door middel van predefined sub-graphs en predefined queries. Dit biedt dus weinig flexibiliteit voor de gebruikers.
- Hiervoor werken ze met “credentials”. In dit geval is KIK-V de “issuer” van de credentials en de gebruikers zijn de “holders”. De credentials worden zowel gebruikt voor het uitgeven van de queries als toegang tot de data.

Zie ook
- Implementatie van Nuts Bolt voor KIK-V
- Nuts Specificatie
- Nuts Documentatie

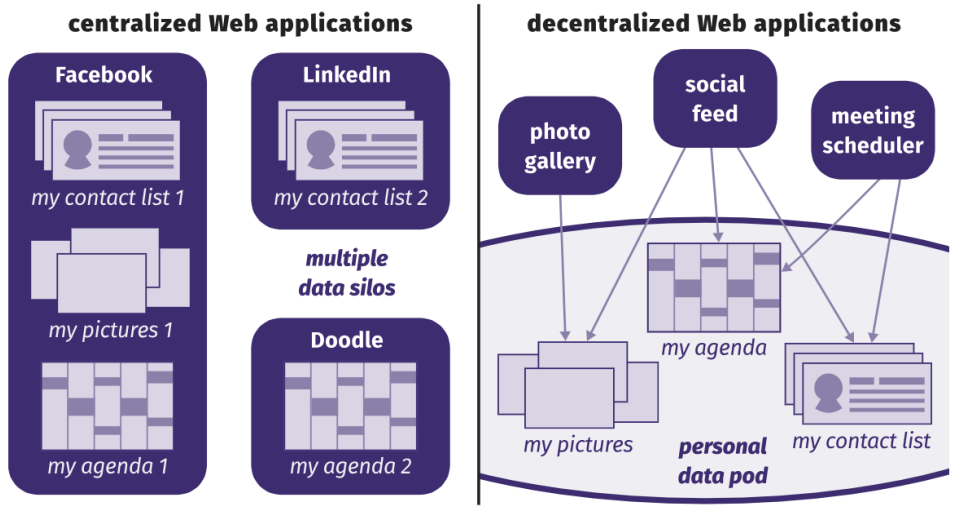


Implementatie | Solid Project

Probleem context
Uitgangspunt van de (social) platformen van tegenwoordig (denk aan Facebook, Google etc.), is dat je een account voor elk platform nodig hebt. De data die je produceert bevindt zich ook *in* dat platform … en is daarbuiten niet toegankelijk. Gebruikers hebben beperkte of vooral geen controle over hun eigen data en de data zit ook vast in de silo's bij techgiganten. Sterker nog, dat is de manier waardoor deze techgiganten zo groot geworden zijn!

Oplossing
Het Solid Project is bedacht door Sir Tim Berners Lee, uitvinder van het web. Het Solid Project introduceert het concept van een Personal Online Datastore (POD), een online opslagruimte voor *mij* persoonlijk. Platformen zijn nog steeds belangrijk en kunnen nog steeds werken, door gebruik te maken van en de data op te slaan op in de PODs van gebruikers. De data wordt zo decentraal voor de platformen en juist centraal voor de gebruikers opgeslagen.
Het Solid Porject is een reeks (Linked Data) specificaties voor standaarden die ervoor zorgt dat het concept van PODs kan gaan werken. De data wordt gescheiden van de applicaties. Het is daarom mogelijk om met verschillende applicaties gebruik te maken van dezelfde data … mits toestemming (autorisatie) gegeven door de gebruiker. Gebruikers krijgen zo regie over hun eigen data en wie (of wat) toegang heeft tot hun data.

Inzichten voor Lock-Unlock
- Solid Project is gebaseerd op Linked Data
- Autorisatie is heel specifiek per dataset per gebruiker vanuit het concept van PODs.
  Het verlenen van toegang tot je eigen data is heel fijnmazig en kan door de eigenaar zelf worden ingesteld en gewijzigd. Dit maakt dat er veel autorisatiemogelijkheden zijn, maar het beheer daarvan is al snel erg complex en onbegrijpelijk voor de 'normale gebruiker'
- Door toepassing van Linked Data is 'cross dataset' bevragen, 'queriën' over meerdere endpoints zeer krachtig en goed mogelijk … en tegelijk een uitdaging op zich als het 19 mln burgers betreft. Het is wel in de basis zeer federatief opgezet.
- De specificatie is nog volop in ontwikkeling en er zijn nog weinig volwassen implementaties



Zie ook
- Solid Project

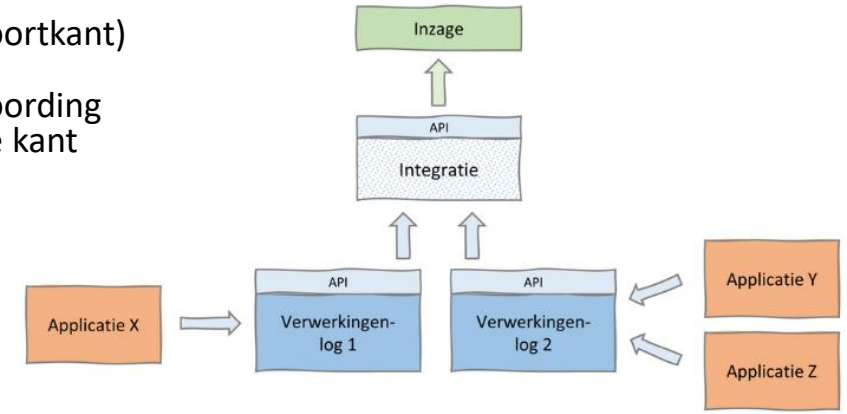
Implementatie | GEMMA verwerkingenlogging

Probleem context
Gemeenten en andere overheidsorganisaties gebruiken bij de uitvoering van hun taken veel (persoons)gegevens. Vanuit de AVG zijn er duidelijke regels over het verwerken van persoonsgegevens. (Overheids)organisaties zijn bijvoorbeeld verplicht om dit gebruik vast te leggen en inzichtelijk te maken voor burgers. Ook moeten zij kunnen aantonen dat verwerking van deze gegevens voldoet aan de belangrijkste beginselen van verwerking zoals rechtmatigheid en doelbinding. Zolang dat *binnen* één (overheids)organisatie plaatsvindt, kunnen autorisaties en doelbinding (tot zekere hoogte) goed gecontroleerd worden. Zodra dat over organisatiegrenzen heen gaat en er API koppelingen tussen systemen van verschillende organisaties zijn, wordt dat moeilijk en complex.

Oplossing
De Verwerkinggenlogging is een API-standaard die is ontwikkeld door VNG Realisatie en deze standaard is opgenomen in de GEMMA referentiearchitectuur. Deze API-standaard biedt leveranciers van informatiesystemen gestandaardiseerde API-specificaties voor het vastleggen en ontsluiten van metagegevens behorend bij vastgelegde (logging) verwerkingen.

De API standaard is ontwikkeld voor gemeenten en gemeentelijke samenwerkingsverbanden. Er zitten echter geen gemeente-specifieke zaken in de standaard dus andere organisaties kunnen de standaard ook toepassen. Door ontwikkeling vindt dan ook op dit moment (’23/’24) plaats door MinBZK.

Inzichten voor Lock-Unlock
- Bij de controle voor toegang (autorisatie) is het voor een data-leverende partij niet mogelijk om te controleren of rechtmatigheid en doelbinding valide zijn. Dat komt omdat deze bij het specifieke gebruik ligt, die binnen de organisatiegrenzen van de vragende partij ligt. Daarom is de verantwoording op rechtmatigheid en doelbinding voor beide partijen: de vragende partij dient de specifieke gebruiker en specifieke casus vast te leggen en daarvan de referentie door te geven aan de data-leverende partij, welke verplicht is om deze referentie vast te leggen en globale autorisatiecontrole toe te passen voor de vragende organisatie.
- Verwerkingenlogging biedt (daarom) controle achteraf en niet vooraf. Autorisatie (aan de voortkant) is daarom niet te baseren op de verwerkingenlogging of controle op doelbinding.
- In het kader van autorisatie is de verwerkingenlogging (dus) niet relevant. Voor de verantwoording voor het verwerken van persoonsgegevens is dit juist wél relevant, zowel aan data-vragende kant als data-leverende kant.



Zie ook
- https://vng-realisatie.github.io/gemma-verwerkingenlogging/

Extra achtergrond: Privacy enhancing technologies (PETs)

- PET is een verzamelnaam voor verschillende technieken die de afscherming van (persoons-)gegevens ondersteunen
- PETs faciliteren dat analyses kunnen worden uitgevoerd zonder de onderliggende data in te zien, zo worden toepassingen mogelijk die anders onmogelijk zouden zijn.
- PETs zien wij dan ook niet als technieken die autorisaties inregelen voor het bevragen van specifieke data maar faciliteren voornamelijk analyses waarbij toegang tot de onderliggende data niet toegestaan/gewenst is.

Zie ook:

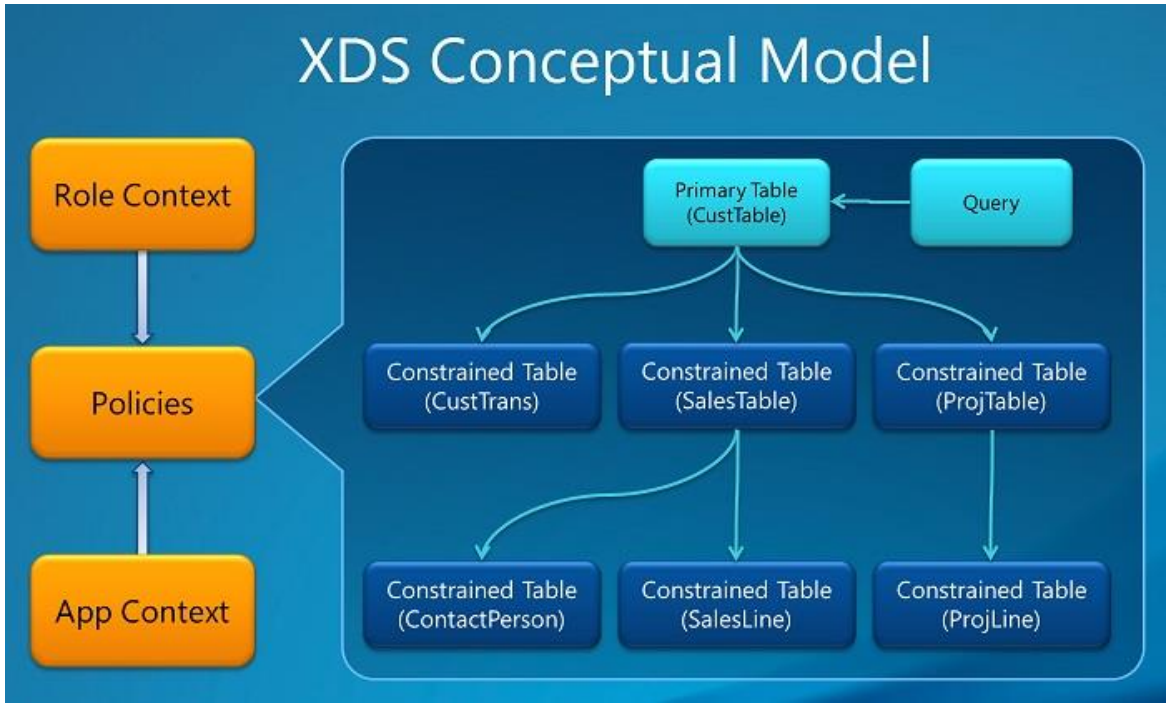
- Wegwijzer PETs van IBDS
- PETs in de praktijk – VKA
- TNO over PETs

Voorbeelden van PETs:

- Zero knowledge proofs
- Bloom filters
- Differential Privacy
- Synthetische data
- Federated Learning
- Multi Party Computation
- Homomorphic Encryption

Extra achtergrond: Microsoft XDS

Microsoft Extensible Data Security (XDS) Framework (link) maakt het mogelijk om role-based access control te implementeren op basis van policies/beleid waarbij de toegang tot de data (tabel) beperkt kan worden.



Extra achtergrond: Open Policy Agent

Open Policy Agent (OPA) is een open-source beleidsmotor voor toegangscontrole en beleidsautomatisering in softwaretoepassingen. Het maakt gebruik van de beleidstaal genaamd Rego om beleidsregels te definiëren en uit te voeren. Rego is ontworpen om leesbare en krachtige beleidsregels te maken. OPA integreert met verschillende services en applicaties en kan complexe beveiligings- en autorisatievereisten implementeren door Rego-beleidsregels te evalueren op basis van inputgegevens, waardoor wordt beslist of een actie al dan niet wordt toegestaan. Dit wordt vaak gebruikt in moderne IT-omgevingen voor flexibele en geautomatiseerde beleidsafdwinging.

- Documentatie (link)
- Policies in Rego lang (link)