



IDENTITY



Zero Trust opschaalplan

“Een centrale manier voor het inrichten en beheren van policies voor de applicaties in het Common Ground platform.”

In opdracht van: de koplopers van Common-Ground, December 2022

Managementsamenvatting

De aanleiding tot dit project zijn de verschillende ontwikkelingen in de markt, zoals ZeroTrust, Commonground, Referentie architectuur VNG Werken met API's en Microservices ontwikkeling waardoor het sturen op toegangsverlening door middel van Policy Based Access Control (PBAC) een gewenste situatie lijkt. Sonicbee heeft in samenwerking met de Gemeente Den Haag een Proof of Architecture (PoA) uitgevoerd waar uit bleek dat deze techniek toepasbaar is. De PoA van voorjaar 2022 concludeerde dat PBAC een realistisch concept is, maar wekte discussie op omtrent het ontwikkelen en beheeren van de policies. Een vereiste voor succes voor PBAC is dat de toepassing beheersbaar en schaalbaar is. Hiervoor is het Zero Trust opschaalplan project gestart waarvan het eindrapport voor u ligt. Het doel van het zero trust opschaalplan is van een 'hard coded policy' voor een usecase, naar een methode om policies voor honderden usecases te kunnen modeleren.

Dit vervolgproject, uitgevoerd van augustus tot december 2022 bestond uit de verschillende onderdelen:

Deelproject 1: procesmatige analyse van use cases en policies binnen gemeenten.

In samenwerking met Ritense, de leverancier op het gebied van digitaliseren van bedrijfsprocessen bij de gemeente Den Haag, is gekeken naar administratie, beheersing en modellering van bedrijfsprocessen. Deze zijn geanalyseerd en integraal meegenomen in het project.

Deelproject 2 : technische analyse van Inrichtingen tools






Er is gekeken naar de technische haalbaarheid van verschillende oplossingen van API gateway en policy engines. Bij voorkeur een open-source optie gezien er vele stakeholders betrokken zijn en de oplossing laagdrempelig te implementeren moet zijn. Daarnaast was ook de eis voor input-filtering een overweging die werd meegenomen in de analyse. Tot slot bestond de POA van voorjaar 2022 uit een policy met 'hard coded' parameters, in dit vervolgtraject is de policy dus danig herschreven dat gebruik kan worden gemaakt van herbruikbare en externe parameters.

Deel project 3: Schaalbare methode en ontwikkelstraat omschrijven

Er is door Sonicbee een methode, de ontwikkelstraat, ontworpen, waarmee het aantal policies kan worden beperkt en die is gericht op herbruikbaarheid van de policies t.b.v. PBAC. Dit gebeurt door o.a. te inventariseren welke attributen, API's, en regels van belang zijn voor de use cases. Bovendien wordt er een policy architectuur gemaakt voor de PBAC Zero Trust aanpak voor de Gemeente Den Haag en zo indirect ook Common Ground op de langere termijn

De bevindingen: Ten aanzien van deelproject 1 en 3: het is mogelijk om op een schaalbare wijze access policies te ontwikkelen. Voordeel van de aanpak van de ontwikkelstraat is dat die naadloos past in de bestaande praktijk van procesbeschrijvingen (BPMN), zowel bij grote als kleine gemeenten, waardoor de impact van procesaanpassingen is beperkt, maar in combinatie met PBAC, zal wel een grote bijdrage worden geleverd aan governance binnen de gemeente. Uit deelproject 2, bleek dat de open-source variant van policy engine toerijkend is waardoor verdere benchmark niet nodig was. Voor API gateways zijn de eisen getoetst op verschillende oplossingen, zowel open-source als in commerciële producten. We zien voorsnog geen knelpunten voor selectie van API-gateways, mits de API-gateway overweg kan met het concept van input-filtering zoals beschreven in deze methode. Inzet van PBAC vergt wel technische expertise, hetgeen voor kleinere gemeenten wellicht andere ondersteuning vergt.

Inhoud rapport

	Introductie aan de hand van projectcontext rondom Zero Trust opschaalplan	4
	Zero trust opschaalplan – deelproject 1 en 3	13
	Technische implementatie en analyse tooling - deelproject 2	39
	Conclusie en aanbevelingen	52
	Appendices	54

● **Project context**

Aanleiding tot dit project zijn de verschillende ontwikkelingen in de markt, zoals ZeroTrust, Commonground, de Referentie architectuur van de VNG over werken met API's, en de beweging naar microservices, waardoor het sturen op toegangsverlening door middel van Policy Based Access Control (PBAC) een gewenste situatie lijkt.

De komende 4 slides bieden informatie over deze ontwikkelingen en schetsen de aanloop tot het Zero Trust opschaalplan, de scope van het project en de output

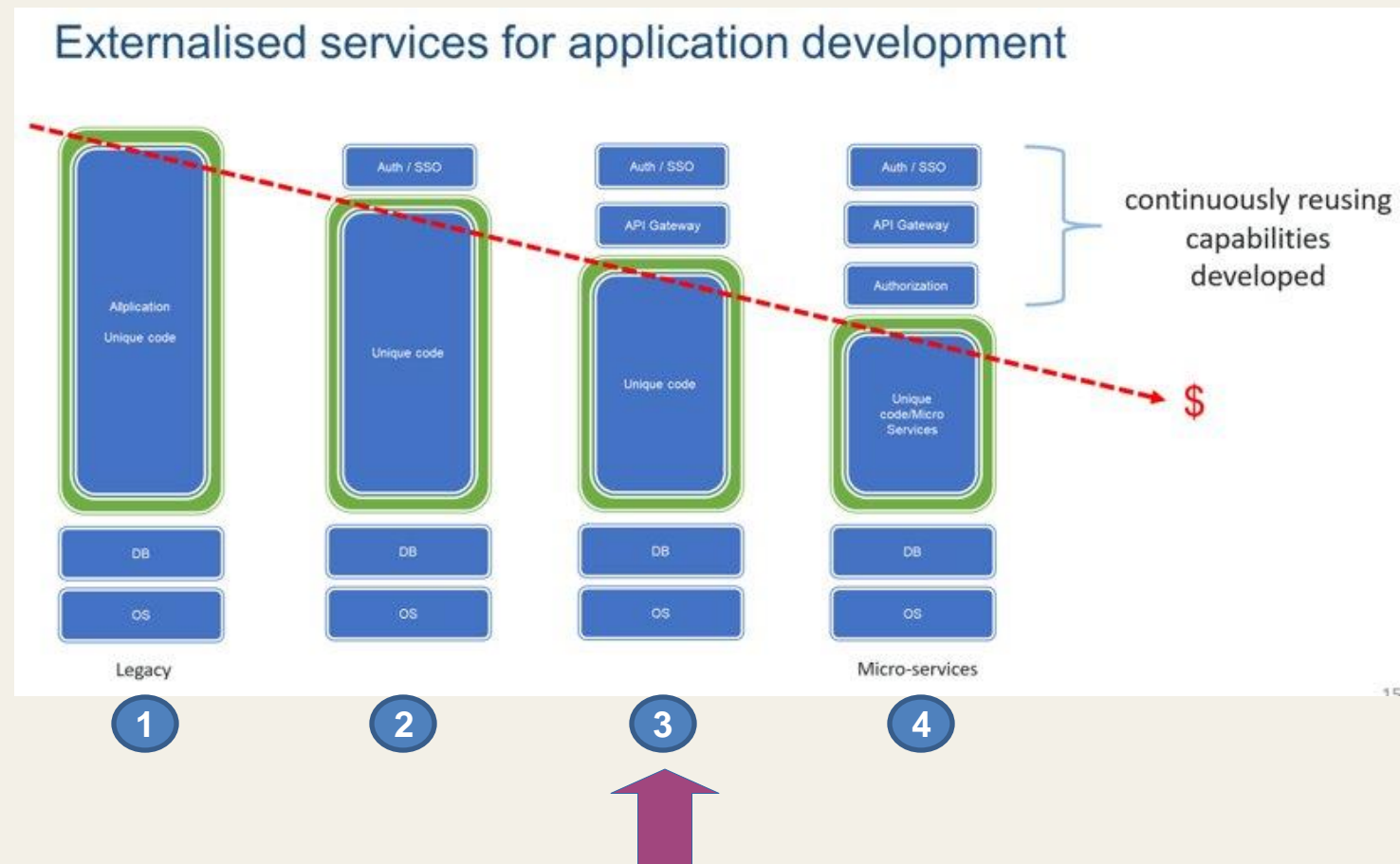
Marktontwikkeling in de tijd

Migratie van monolitische systemen naar microservices is een actuele ontwikkeling.

Van links naar rechts zijn de volgende ontwikkelingen te zien:

- 1 -> 2 : Van legacy naar SSO op legacy apps
- 2 -> 3 : Containerization, API-based access, scalability
- 3 -> 4 : Microservices en API's, voor data access: autorisaties weg uit de services

Daarnaast, zijn Zero-trust en Common Ground de aanleidingen voor de PBAC ontwikkeling.



Huidige API oplossingen voor RBAC applicaties (o.a. OpenZaak voor ontsluiting zaaksystemen)

(Afbeelding Gebaseerd op model van Axiomatics, zoals beschreven in VNG RAWA project)

Huidige situatie

Het huidige systeem voor gegevensuitwisseling binnen gemeenten maakt het moeilijk om

- 1) snel flexibel te innoveren
- 2) aan de privacywetgeving te voldoen
- 3) efficiënt én veilig met gegevens om te gaan

Het project Referentie Architectuur Werken met API's (RAWA) van de VNG beschrijft de toekomstige wijze van integratie, waarbij concepten als Commonground, API access en Zero Trust leidend zijn. De in het voorjaar van 2022, voor de koplopers van common ground uitgevoerde Proof of Architecture toonde de kracht van de mogelijkheden van moderne toegangstechnieken aan, maar liet ook zien dat governance een uitdaging is.

Dit vraagt om input-filtering: Input-filtering is het concept van het beperken van de output resultaten van een API, op basis van injectie van relevante en specifieke parameters in de access request.

Knelpunt huidige situatie: input-filtering

Momenteel worden door de koplopers van Common Ground diverse API's toegepast. De belangrijkste is wel de door de VNG ontwikkelde OpenZaak API, waarmee zaakmanagementsystemen kunnen worden ontsloten.

Geconstateerd is dat binnen de API autorisatiemanagement aanwezig is, maar dat die alleen grofmazige autorisaties bieden (op basis van traditionele role based access control - RBAC). Het resultaat is dat de API's te veel output genereren, inclusief output die niet relevant is voor de onderhanden taken. Dat betekent dat niet voldaan wordt aan 'need-to-know' en 'need-to-have' principes vanuit onder meer de privacy-wetgeving en BIO.

Om dit knelpunt op te lossen is in POA 1 onderzocht of het mogelijk is om autorisatie (deels) buiten de API's te houden door het concept van input-filtering toe te passen: de OPA policy engine injecteert aanvullende attributen in de access request om de API te dwingen meer specifiek gegevens op te leveren.

Het resultaat van POA 1 was positief, hetgeen leidde tot het vervolgproject.

Aanleiding tot Zero trust opschaalplan

- In een eerder uitgevoerde Proof of Architecture (POA) is aangetoond dat een innovatieve werkwijze voor toegangsbeheersing, waarbij gebruikt is gemaakt van inputfiltering, werkte in de casus van de OpenZaak API.
- Deze POA liet in detail zien dat fijnmazige toegang tot 'legacy' API's mogelijk is. Dit heeft de volgende voordelen:
 - Veel lagere kans op AVG-gevoelige bijvangst
 - Schaalbare architectuur
- De POA liet ook zien dat dit veel maatwerk, waaronder een hard gecodeerde policy, met zich meebracht. Dat is niet schaalbaar voor honderden use cases. Dus de policies moeten eenvoudiger om schaalbaar te zijn.
- Daarom is in het vervolg een werkwijze ontwikkeld voor het opschalen van deze aanpak, met daarin specifiek de vraag voor
 - Herbruikbaarheid van policy-items
 - Methodisch ontwikkelen van policies

Doel van het project

Wat is het doel van het project 'Zero Trust opschaalplan' en waarom willen de koplopers Common Ground dat?

Het ontwerpen van een 'Zero trust opschaal plan' op basis van het SonicBee gedachtegoed zoals aangetoond in de vorige POA, waarmee gemeentes worden geholpen access policies te decomponenten en opnieuw te componeren. Zo kunnen zij met de innovatieve Zero Trust omgeving meer interoperabiliteit bereiken. Door van ' één hard coded policy' voor een usecase, naar een methode om policies voor honderden usecases te kunnen modeleren.

Doel van de koplopers van Common Ground na inzichten Zero Trust POA:

“Een beheersbare toekomstige situatie te creëren waarbij de innovatieve richting die veel kansen biedt kan worden ingeslagen. Grip op access via policies en het inzichtelijk houden van de verschillende policies zijn hierbij van belang.”

Het Zero trust opschaalplan

De inhoud van het 'Zero Trust opschaalplan'

Hypothese:

- Er bestaan herbruikbare componenten / delen van policies
- Het project richt zich op het ontwikkelen van een methode om policies eenvoudig af te leiden van de use case, gebruikmakend van hergebruikpotentieel (deel project 1)
- Onderzoeken technische mogelijkheden voor schaalbaar (deelproject 2)
- De methode ('ontwikkelstraat' genaamd) moet bruikbaar zijn, onafhankelijk van de aard en de omvang van de gebruikende organisatie (deelproject 3)

Output

De verschillende outputs die voortvloeien uit de onderliggende deelprojecten

- Het Zero Trust opschaal plan: een presentatie waarin de projectresultaten staan, met werkinstructies, architecturen, ontwerp ontwikkelstraat (niet geautomatiseerd), methoden, aanbeveling mogelijke tools en beheer (globaal). Ook worden er, rekening houdend met de dynamiek van toepassing van een Policy Information Point met relationele attributendatabase, twee bruikbare OPA-polices als code opgeleverd met een omschrijving wanneer die toepasbaar zijn.

Scope

Deze pagina beschrijft wat in scope is gesteld voor dit project

Er wordt uitgegaan van de architectuur voor de omgeving van de gemeente Den Haag en gebruik van Open Source oplossingen in de referentie architectuur.

- Vaststellen geschiktheid van meerdere soorten API gateway voor ondersteunen POA OpenZaak
- Toetsen van de oplossing aan de use case Financiële Trainingen:
 - Ontsluiten API via input-filtering
- Toetsen van de oplossing aan de use case Ooievaarspas
 - Ontsluiten API via input-filtering
 - Ontsluiten van de HaalCentraal API
 - Beveiligen, inperken van toegang tot deze API alleen indien en voorzover daar ruimte voor is.
- Ontwikkelen van de methode van policy's samen te stellen

Scope-inperking

Deze pagina beschrijft wat buiten scope is gesteld voor dit project

Expliciet buiten scope is governance en het eigenaarschap rondom overheidshandelen.

Conform de bestaande werkwijze betekent dit dat besluiten door machines kunnen worden genomen, zonder daar een persoon verantwoordelijk voor te stellen. Het inrichten van het concept eindverwoordelijke voor beslissingen van de gemeente is een organisatiebrede discussie die buiten dit project om dient te worden gevoerd.

Zero Trust ontwikkelstraat

De Zero Trust ontwikkelstraat is de output van deelproject 1 en 3.

Het hoofdstuk is opgedeeld in de 4 onderdelen; de uitgangspunten, achtergronden, aanpak om tot Policies for PBAC te komen en een sjabloon voor toepassing binnen PBAC.

Door toepassing van deze methode is het mogelijk om binnen de bestaande werkwijze van procesmodellering de concepten van access control toe te voegen zonder veel extra handelingen te moeten uitvoeren.

Policy ontwikkelstraat – uitgangspunten

Dit hoofdstuk beschijft de methode om policies te ontwikkelen die binnen de huidige PBAC-architectuur toegepast kunnen worden

Uitgangspunten governance

Van elke beslissing ligt vast hoe die tot stand is gekomen. Per taak ligt vast

- Wie: medewerker, rol, zaaktype
- Wat: beslissing, notitie, controle
- Op grond van: procesbeschrijving
- Wie is aansprakelijk (lijn-/proces-/afdelings-manager), wie was de uitvoerende (actor/behandelaar)
- Voorstel beschrijven verantwoordelijkheden: RACI of AO/IC regels

Daarnaast moet binnen een proces bekend zijn hoe functiescheiding / 4-ogencontrole is ingericht.

Van elke activiteit ligt vast wie de betreffende taken, die leiden tot de uitkomst, heeft uitgevoerd (dit maakt borging via de audittrail mogelijk)

Uitgangspunten automatisering

Voor elke taak in het zaakmanagementsysteem dient te worden vastgelegd:

- Betrokken actor (subject, natuurlijk persoon)
- Actortype / van toepassing zijnde rol. Bijvoorbeeld: behandelaar, toezichthouder, werkverdelers
- Taaktype
- Taaktype, context
- Object: burger/aanvrager (BSN), bedrijf (KVK-nummer), kadastraal nummer
- Taak / beslissing / notitie
- Timestamp
- Functiescheiding / voorkomen belangenverstrengeling

Bedrijfsproces

De gemeentes van de koplopers van Common Ground, gebruiken voor bedrijfsproces beschrijving Business Process Modelling (BPM), in dit geval is gemodelleerd door Ritense, de leverancier van de GZAC case management oplossing die door de gemeente Den Haag wordt ingezet.

Binnen de processen (de 'swimlanes') zijn verschillende taken terug te vinden. Wanneer een taak het ophalen of wijzigen van informatie beschrijft, dient in PBAC de vraag gesteld worden: mag de actor dat? En op grond waarvan?

In dit voorbeeld word gebruik gemaakt van de swimlanes uitgewerkt door Ritense

Informatiebehoefte per proces

De antwoorden op de onderstaande vragen leveren de gewenste input voor ontwikkelen van de access policy:

- Het betreffende bedrijfsproces (in dit voorbeeld: ooievaarspas)
- Welke subjecten (actoren, deelnemers) en objecten zijn betrokken
- Welke informatie (met name persoonsinformatie) mag aan de actor worden getoond
- Welke taak-autorisaties (lezen, schrijven, bijwerken) zijn aan de actoren toegekend

Policy ontwikkelstraat - achtergronden

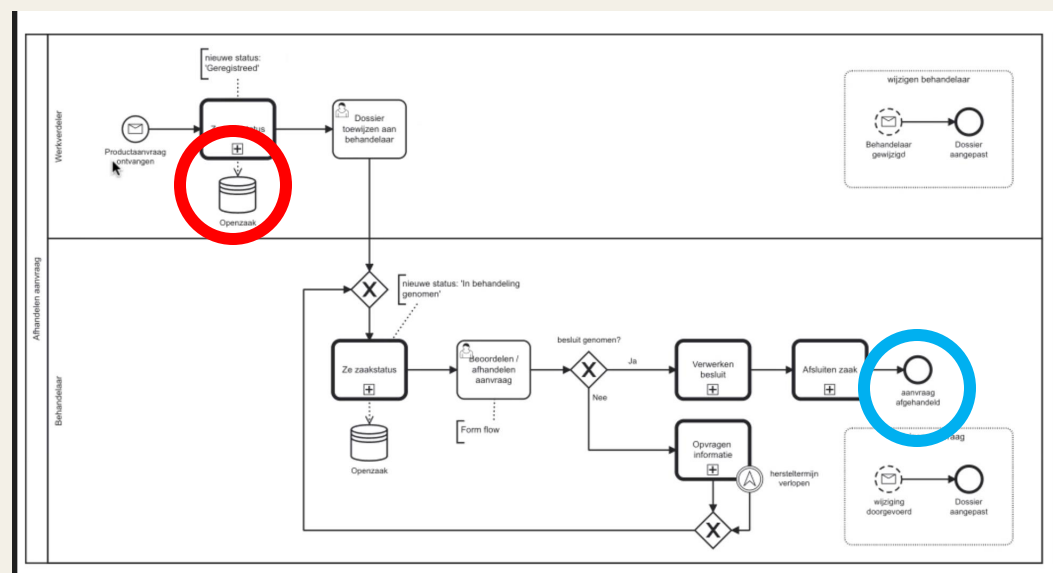
Dit hoofdstuk beschijft de methode om policies te ontwikkelen die binnen de huidige PBAC-architectuur toegepast kunnen worden

Processen: visualisatie

De onderstaande afbeelding toont de 'swimlane' van het proces 'Afhandelen aanvraag' van de casus Ooievaarspas. Binnen dit proces zijn twee hoofdlagen te zien: boven de streep de procesmatige taak van de werkverdeler, die de afzonderlijke taken toedeelt aan behandelaars.

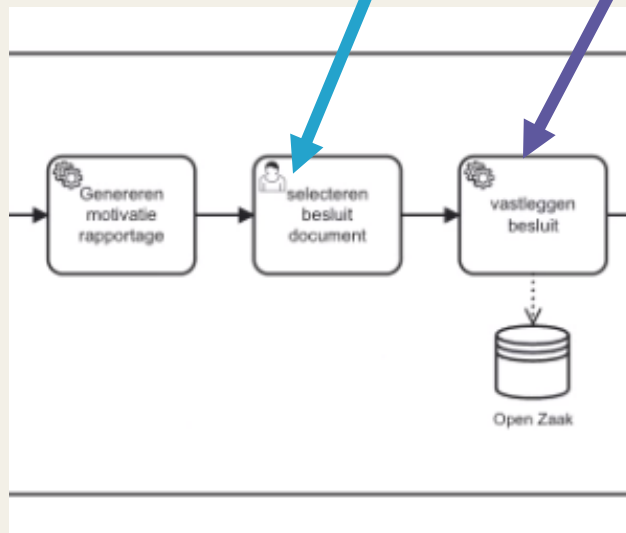
De weergave onder de horizontale lijn betreft de activiteiten die door of namens de behandelaar worden uitgevoerd.

Binnen processen worden taken uitgevoerd, vormgegeven als rechthoeken. Vanuit een taak kan zowel een systeem worden aangeroepen (zie de 'rode cirkel', in het kader van de Ooivaarspas is dat de OpenZaak API waarmee GZAC wordt benaderd) of een subprocess worden aangeroepen (zie de 'blauwe cirkel')



Handmatige en geautomatiseerde taken

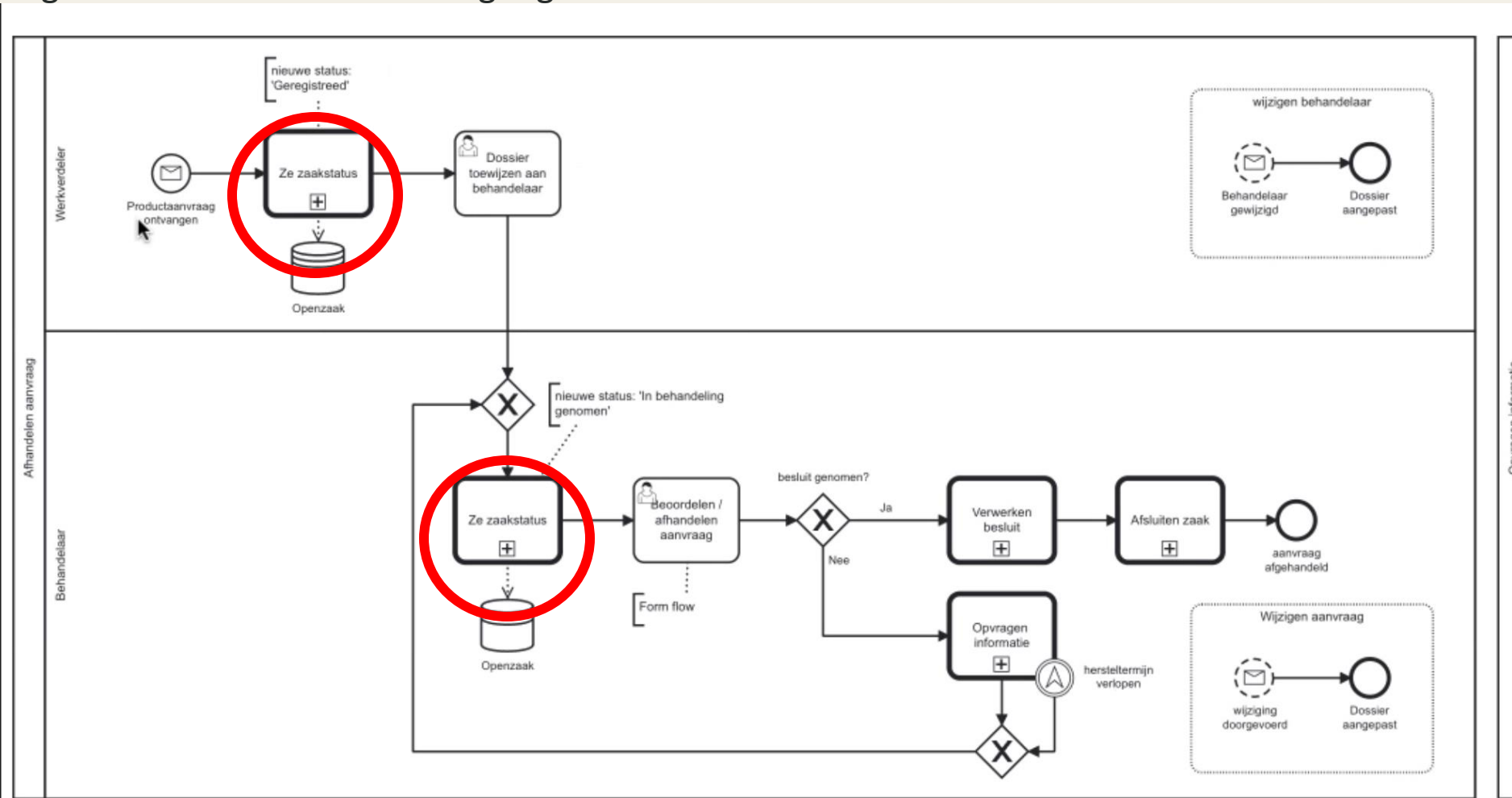
In onderstaande afbeelding is een deel van een proces met daarin zijn verschillende taken/acties weergegeven. Deze taken kunnen handmatig of geautomatiseerd, of tijdgestuurd plaatsvinden. Taken die gegevens benaderen of een API aanspreken zijn in deze visualisatie als geautomatiseerd gekenmerkt.



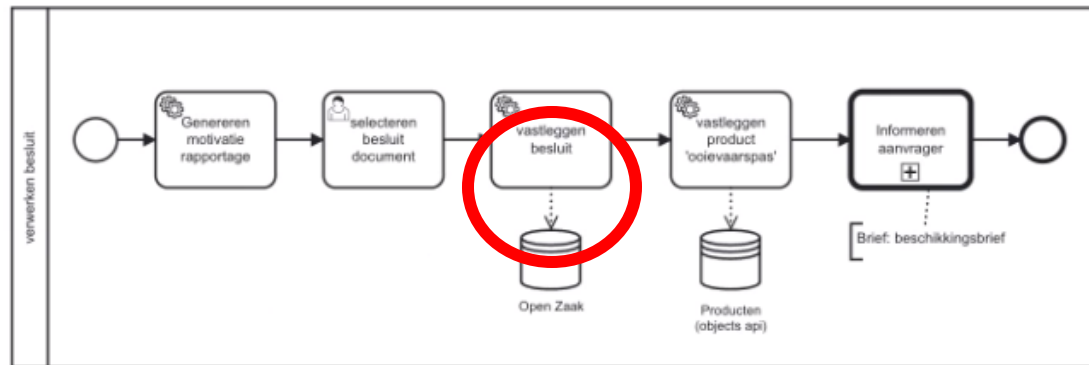
Als in een geautomatiseerde taak het systeem als verantwoordelijke wordt aangemerkt, is in het systeem in essentie de verantwoordelijke uitvoerende. Deze werkwijze is in conflict met de eerder vermelde uitgangspunten, waardoor het noodzakelijk is om een menselijke verantwoordelijke te identificeren. Dit wordt in de verder slides verder toegelicht en uitgewerkt.

Processen en API access

Als in een taak een applicatie wordt aangeroepen, moet bepaald worden of de juiste autorisatie bestaat. In onderstaande afbeelding zijn taken omcirkeld met een rode cirkel als een API wordt aansproken. Hier dient dus nagedacht te worden over toegang.



Bedrijfsproces en informatiebehoefte access control



Basis is de procesflow zoals die door Ritense in de swimlane is gevisualiseerd.

In de omcirkelde taak/activiteit wordt de OpenZaak API benaderd. Dat betekent dat op dat punt de OPA policy engine zal moeten ingrijpen. Ofwel voor toegang, of voor vóórfiltering.

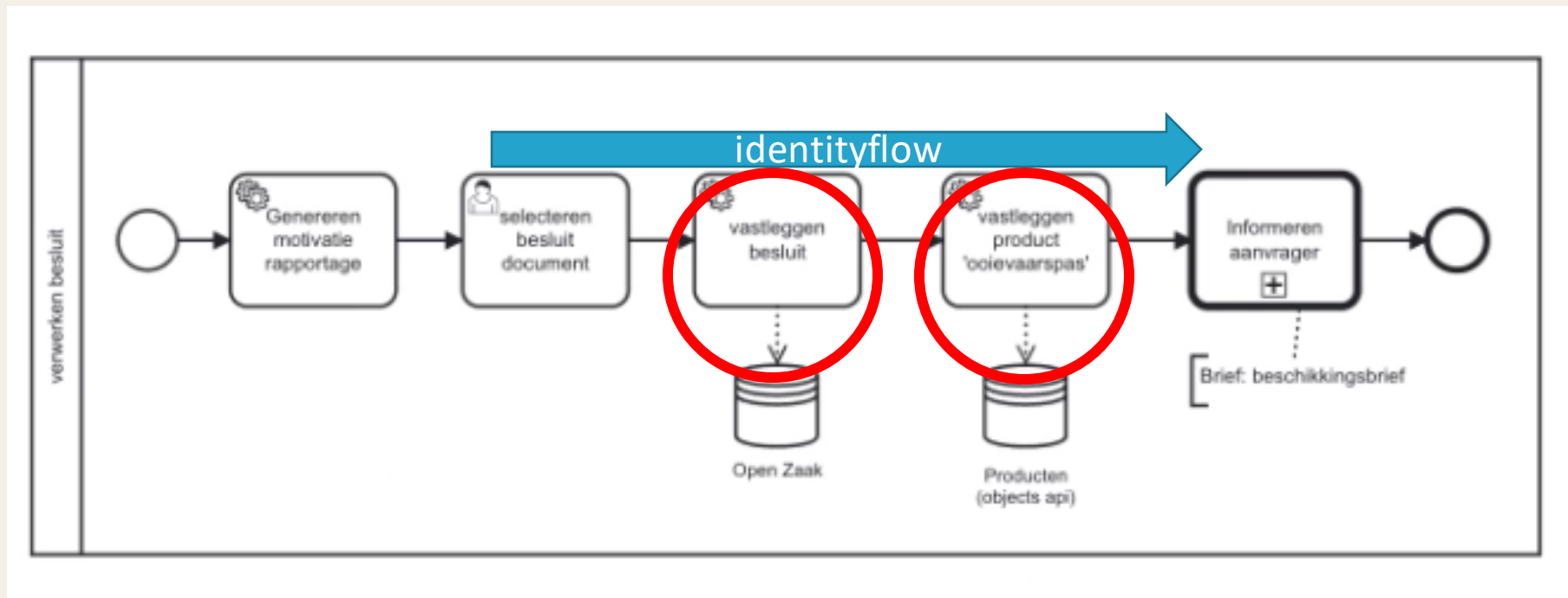
In dit geval is sprake van een geautomatiseerde actie. Dus er is in die specifieke taak geen menselijke verantwoordelijke uitvoerende. Die verantwoordelijke zal moeten worden gevonden in de voorgaande processtappen.

Informatiebehoefte voor een processtap/taak/activiteit ten aanzien van toegang:

- Wat is de aard van de taak?
- Kan sprake zijn van Functiescheidingsproblematiek/4-ogen (al zal een dergelijke beheersmaatregel via een separate swimlane worden ingericht)?
- Welke betrokkenen en/of deelnemers zijn in de taak te identificeren? - dat kan dus de verantwoordelijke zijn die de voorgaande taak binnen de procesbeschrijving uitvoert?
- Welke informatie mag aan de medewerker worden getoond?
- Welke autorisaties zijn aan de medewerker toegekend?

Actoren binnen processen

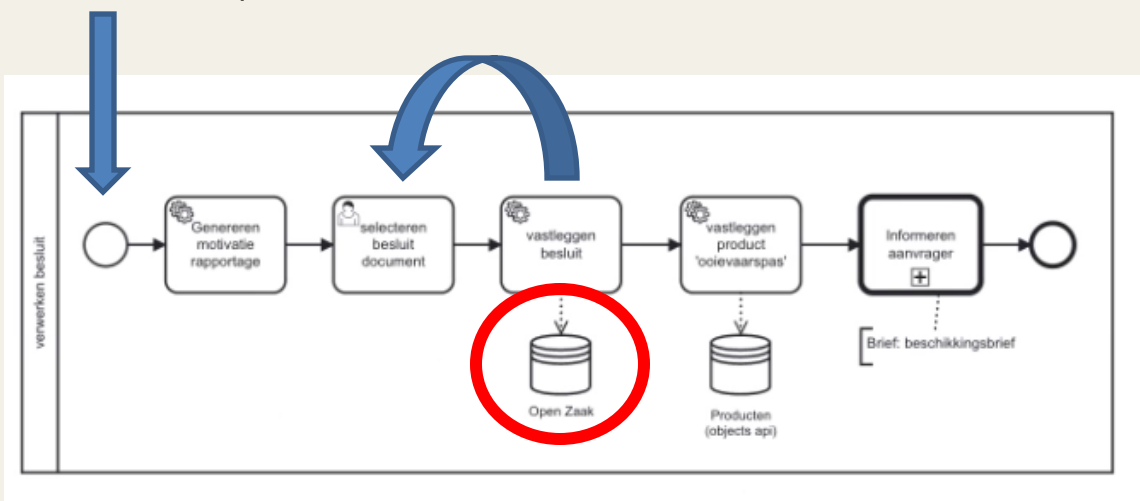
In onderstaande afbeelding is een proces van het verwerken van een besluit weergegeven volgens de BPM methode, daarin zijn verschillende taken/acties te ondervinden, deze kunnen ofwel handmatig ofwel geautomatiseerd plaats vinden. Op grond van de uitgangspunten moet de actor bepaald worden. De verantwoordelijke uitvoerende moet een mens zijn.



Rode cirkels: spreken, zoals vorige dia besproken, API aan.

Actorenanalyse geautomatiseerde taak

Deze swimminglane beschrijft geen gedefinieerde actor. Deze swimming lane is een subprocess van het bovenliggende process. Dit betekent dat de actor van het bovenliggende proces de initiële actor is in de nieuwe swimminglane, indien geen andere actor bepaald is.

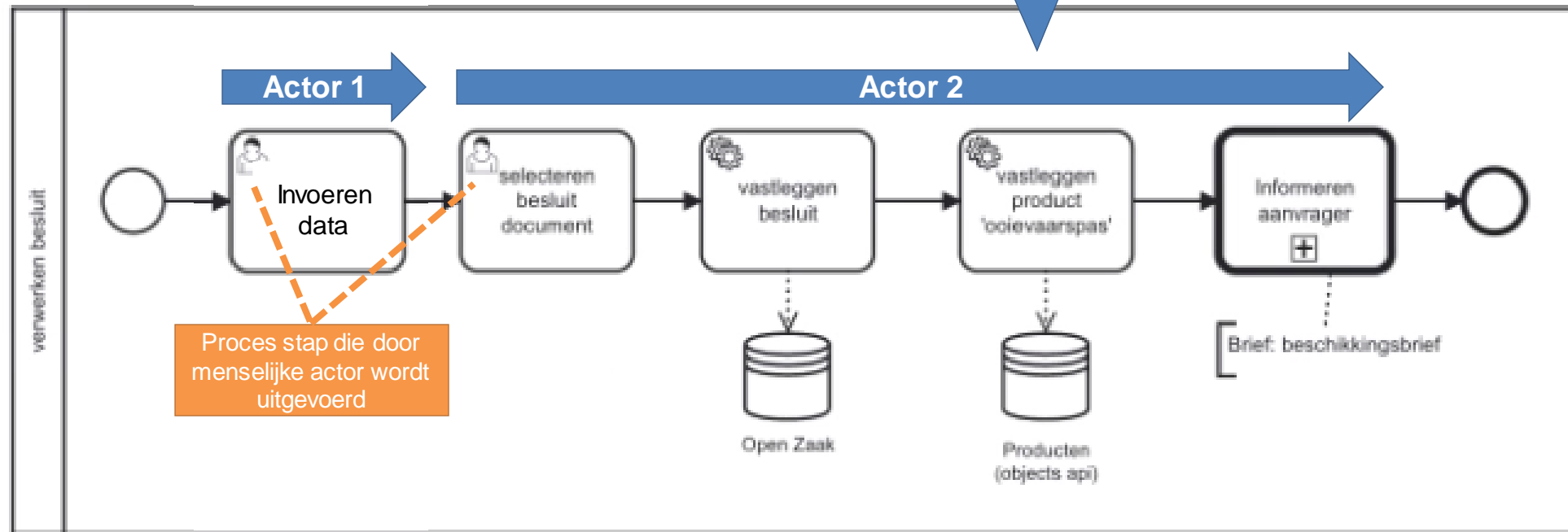


In de omcirkelde taak/activiteit wordt de OpenZaak API benaderd. Dat betekent dat op dat punt de OPA policy engine zal ingrijpen. Ofwel ten behoeve van toegang, of ten behoeve van vóórfiltering.

In dit geval is sprake van een geautomatiseerde actie. Dus er is in die specifieke taak geen menselijke actor. Die menselijke actor zal moeten worden gevonden in de voorgaande processtappen. In dit geval is de API-aanroep het gevolg van een ambtelijk besluit. De medewerker dit besluit neemt (de keuze maakt uit de mogelijke opties) is de persoon die als actor moet worden aangemerkt. Deze actor zal door OPA als actor worden getoetst (dus is de persoon geautoriseerd op grond van persoon/rol/zaaktype).

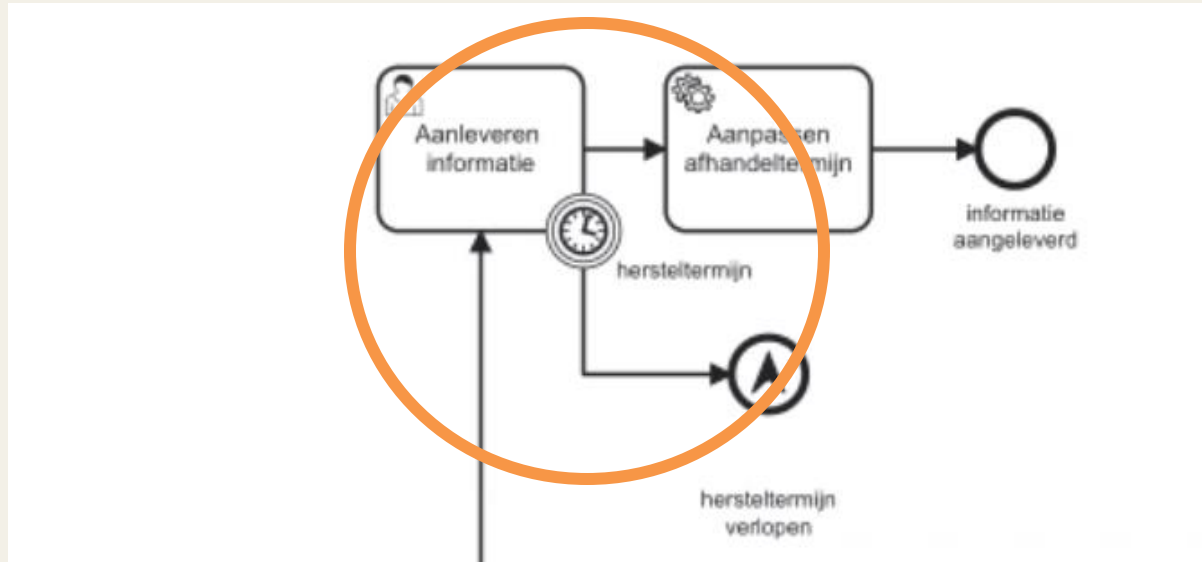
Identityflow

Actor 2 is de verantwoordelijke behandelaar voor stappen 2 en volgende, totdat in de laatste stap een andere actor kan worden geïdentificeerd



De Processtappen 'Invoeren data' en 'Selecteren besluit document' (de eerste twee stappen in de swimminglane) worden beide door menselijke actoren uitgevoerd. In de regel zal per swimminglane slechts 1 menselijke actor betrokken zijn. Toch kan het voorkomen, wegens overdracht, dat de volgende menselijke taak door een ander persoon word uitgevoerd. In dit voorbeeld geldt actor 2 als de verantwoordelijke behandelaar voor de processtappen 2 t/m 5, totdat later in bus een andere verantwoordelijke medewerker kan worden geïdentificeerd.

Actorenanalyse Timeractie



In diverse swimlanes is naast de geautomatiseerde en de handmatige taak, sprake van een tijd-gestuurde taak.

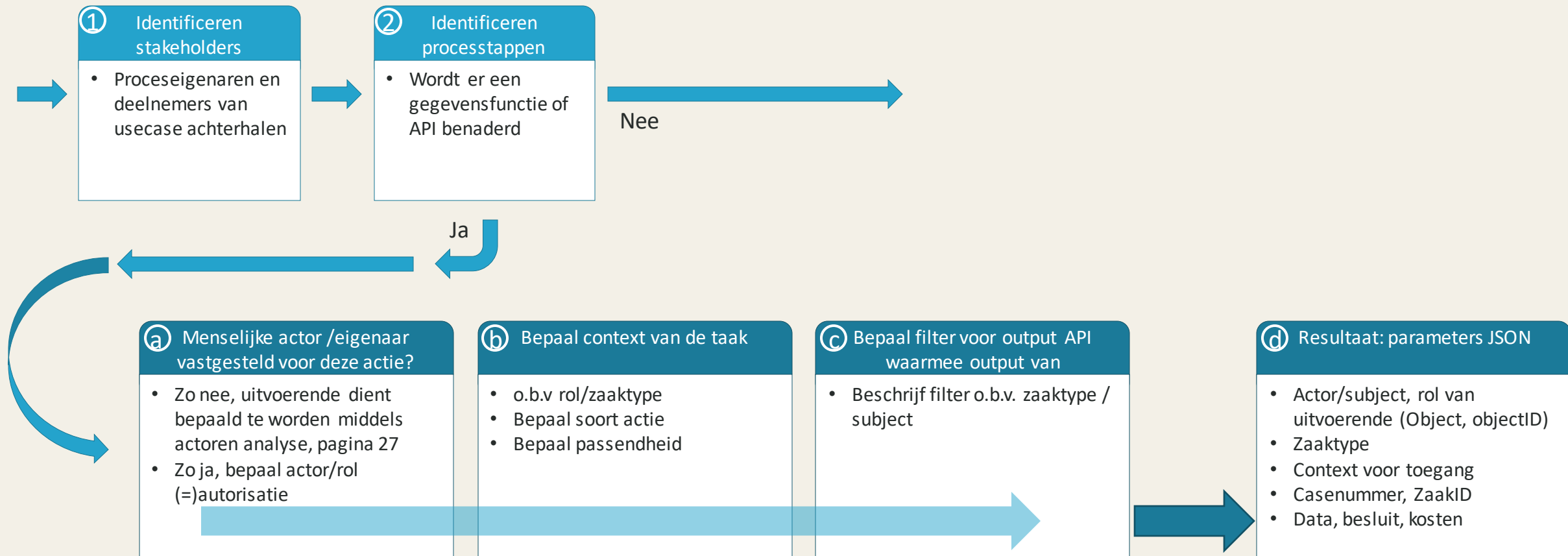
In dit voorbeeld is er voor aanlevering van informatie een antwoordtermijn gedefinieerd. Als de aanlevering niet plaatsvindt voor het einde van de interval, dan zal automatisch een voor gedefinieerde taak worden uitgevoerd, zoals het notificeren van de betrokken aanvrager, gevolgd door het sluiten van de zaak.

In dit geval is sprake van een geautomatiseerde taak waarvan de eigenaar van het proces de verantwoordelijke actor is

Policy ontwikkelstraat - aanpak

Dit hoofdstuk beschijft de methode om policies te ontwikkelen die binnen de huidige PBAC-architectuur toegepast kunnen worden

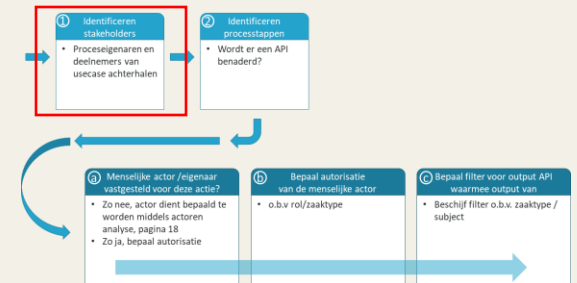
Stappen opschalen PBAC



De volgende slides detaileren elke stap in dit overzicht

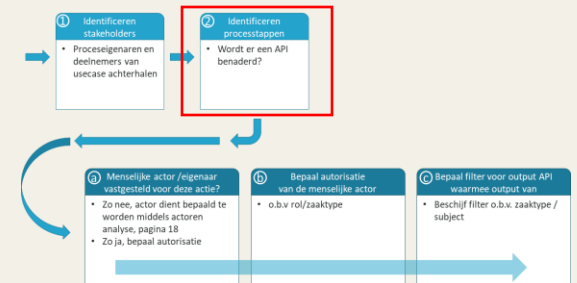
1: Identificeren actoren binnen het proces

- Wie is eigenaar van het proces?
 - Domein, manager. Iemand die aansprakelijk is voor het betreffende (bovenliggende) bedrijfsproces
- Wie zijn de deelnemers binnen het proces
 - Actoren / subjecten zijn in eerste instantie bijvoorbeeld de 'werkverdelers' en 'behandelaars', dat zijn de 'rollen' binnen zo'n swimlane. In essentie moeten deze rollen per taak worden benoemd.
 - Objecten zijn de betrokken aanvragers en personen die namens iemand anders een aanvraag doen, dus burgers en bedrijven.
 - Derden kunnen managers, QA-medewerkers, toezichthouders en auditors zijn, die toezien op de afhandeling van een taak.



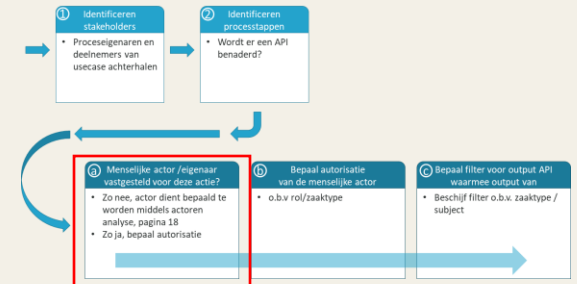
2: Identificeren processtappen van actor

- Per processtap moeten de volgende attributen geïdentificeerd worden;
 - Wat zijn de activiteiten en taken binnen de processen
 - Welke acties benaderen de API ?
 - Als in de processtap een API of gegevensfunctie wordt benaderd, is een verdere analyse nodig. Als er geen API wordt benaderd, is verdere analyse niet nodig.



2.a: Vaststellen actor en autorisatie

- Wanneer niet menselijk actor, zoek de voorgaande menselijke actor
 - Zie bovenstaande actoranalyse
- Bepaal de rol van de actor
- Voor sommige processen en taken kan ook de aanvrager (een burger of een ondernemer) toegang (moeten) krijgen. De actor is dan identificeerbaar op grond van een BSN en/of een KVK-nummer.



2.b: Vaststellen contextregels

- Zaaktype (voorgaande informatiebehoefte) bepaalt type taak
- Vaststellen vereiste autorisatie per taak
- Stel vast of contextbeperking (filter) op basis van type taak of op basis van subject van kracht is

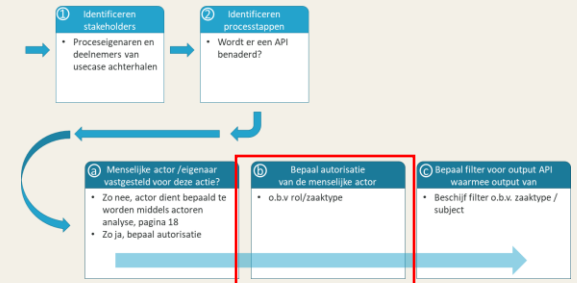
- Aard van de taak

Op basis van AO/IC

- Beschikken
- Bewaren
- Registreren
- Uitvoeren
- Controleren

Op basis van RACI/RASCI

- Responsible
- Accountable
- Consulted
- Supported
- Informed

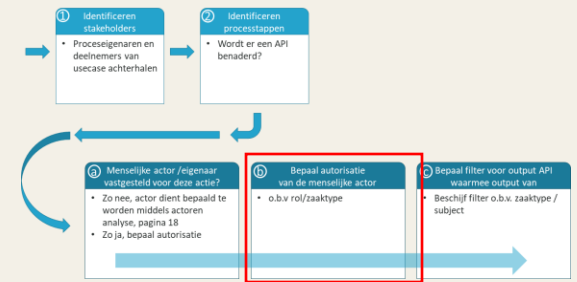


2b -> Autorisatieanalyse

- Zijn er samenvallende actoren/taken combinaties

Actoren/taken combinaties worden vermeld in een actor-taak tabel

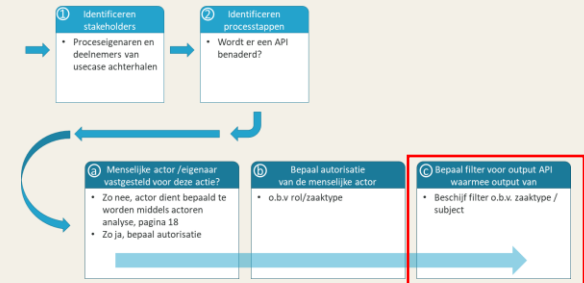
- Bestaan die combinaties ook binnen andere use cases
- Wordt algemene of specifieke informatie opgevraagd / getoond
 - Algemeen
 - Is AVG een aandachtspunt?
 - identificeer mogelijkheid van voorfilteren
 - Specifiek: beperken tot toegestane actoren



2.c: Filterbepaling

- Parameters tbv aanroep API
- Zaaktype
- Taak
- Actor
- Rol actor
- Zaak ID
- Subject
- Gegevensveld passend bij Taak binnen Zaak (vb: beslissing, kosten, tijd)

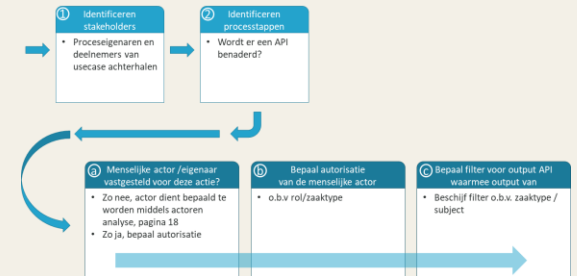
Definiëren Filter
Op basis van taak binnen zaak
Wanneer nodig dan Filteren op
Subject en/of Context



2.d: Opvraagparameters

Resultaat: parameters JSON

- Actor/subject, rol van actor (Object, objectID)
- Zaaktype
- Context voor toegang
- Casenummer, ZaakID
- Data, besluit, kosten



Deze relatie tussen zaaktype, taak en actor kan in een relationele database of tabel worden verwerkt voor raadpleging door een autorisatiemanagementfunctie:

	A	B	C	D	E	F	G	H
1	Profiel	Rol	zaakniveau	UUID Zaak type	GZAC dossier type	Entitlements per rol detailniveau	documenttypeniveau	Functies (GZAC)
2								
3	mdw frontoffice	inzage	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	n/a	n/a	n/a
4	mdw frontoffice senior	inzage+	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	alle zaaktypen	alle documenttypen behorende bij zaak	n/a
5	mdw bezwaar en beroep	inzage+	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	alle zaaktypen	alle documenttypen behorende bij zaak	n/a
6	mdw SZW	inzage+	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	alle zaaktypen	alle documenttypen behorende bij zaak	n/a
7	mdw armoederegelingen: behandelaar	inzage+	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	alle zaaktypen	alle documenttypen behorende bij zaak	n/a
8		Ooievaarspas muteren	zaaktype ooievaarspas	9517e5c0-bc2	aanvraag-ooievaarspas	zaaktype ooievaarspas	alle documenttypen behorende bij zaak	Zaakbehandelaar toewijzen, verwijderen
9	mdw armoederegelingen: werkverdelers	inzage+	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	alle zaaktypen	alle documenttypen behorende bij zaak	n/a
10		Ooievaarspas muteren	zaaktype ooievaarspas	9517e5c0-bc2	aanvraag-ooievaarspas	zaaktype ooievaarspas	alle documenttypen behorende bij zaak	Zaakbehandelaar toewijzen, verwijderen
11	mdw armoederegelingen: hoofd	inzage+	alle zaaktypen	9517e5c0-bc2	aanvraag-ooievaarspas	alle zaaktypen	alle documenttypen behorende bij zaak	n/a
12		Ooievaarspas muteren	zaaktype ooievaarspas	9517e5c0-bc2	aanvraag-ooievaarspas	zaaktype ooievaarspas	alle documenttypen behorende bij zaak	Uitvoeren taken
13	mdw armoederegelingen: manager	inzage+	alle zaaktypen	9517e5c0-bc2e-404d-9b12-16ac59f		alle zaaktypen	alle documenttypen behorende bij zaak	n/a

Standaard Policy ontwerp

Uit de POA komt naar voren dat de volgende onderdelen in de policy moeten worden gebruikt:

- Default deny
- Verifieer geldigheid access requester
 - Herkomst van access requester
 - Check Usertype, rol van de access requester
 - Check combinatie usertype – type taak
- Bij 'Geldig': allow
- Indien voorfilter nodig is:
 - Bepalen inperkingen
 - Injectie gefilterd resultaat in de request/header

Technische analyse

- De technische analyse beschrijft de Resultaten van deelproject 2.

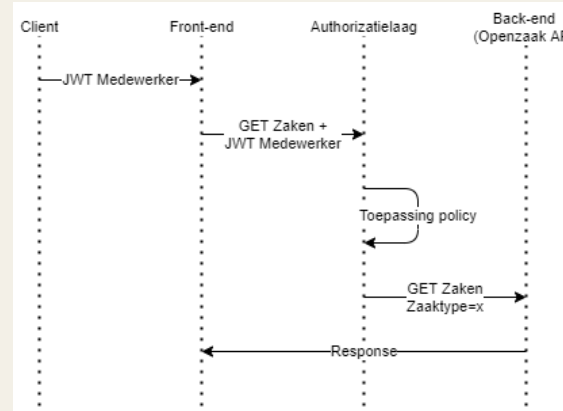
Het eerste doel was de herbruikbaarheid van de policy elementen vaststellen

Het deel twee was het toetsen van de verschillende open-source componenten op toepasbaarheid binnen
dit concept

Conclusies Hackathon Q1 2022

De slide beschrijft kort de Resultaten va POA 1, wat tot deze technische analyse heeft geleid.

- Praktische toepassing zoals opgenomen in project Referentie Architectuur Werken met Api's (RAWA) van VNG.
- Gestelde doel hackathon: Haalbaarheidsonderzoek input-filtering.



```
package example

default allow = false

rollen := {"SZW.MEDEWERKER":{"https://openzaak-cg.test.denhaag.
rollen_flat := { role | rollen[role] }
ys := split(input.request.http.headers.authorization," ")
token := io.jwt.decode(ys[1])
v := { role | role := token[1].roles[_] }
m := rollen_flat & v
#m := "OCW.MEDEWERKER"
#t := rollen[m]

t := concat(",", rollen[m[_]])

heeftRol {
    count(m) > 0
}

allowDetailed = response {
    heeftRol
    response := {
        "allow": true,
        "headers": {
            "header-from-opa": t,
        },
    }
}

allowDetailed = response {
    not heeftRol
    response := {
        "allow": false,
        "status": 418
    }
}
```

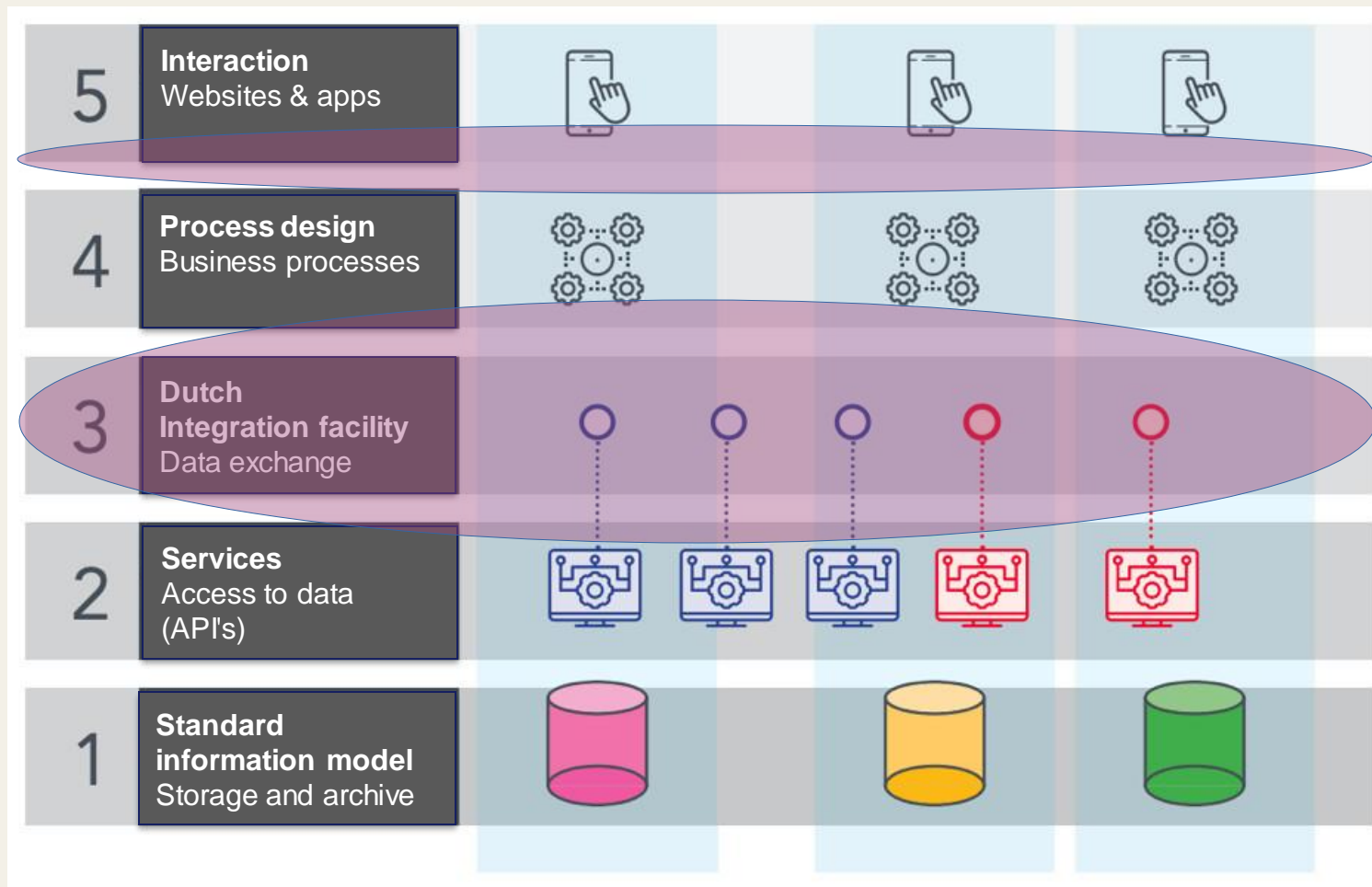
Conclusies hackathon:

- Open-source Kong Gateway – OPA plugin niet toereikend.
- Kong Konnect met eigen OPA plugin wel toereikend.
- Input-filtering haalbaar.
- Uitdaging beheersbaar houden en het ontwikkelen van policies.

Plaatsing van PBAC binnen de architectuur structuur van Common Ground

Deze plaat beschrijft de architectuur van Common ground. Dit beschrijft hoe gemeentes de informatievoorziening dienen te structureren.

Wij voorzien policy management tussen laag 4 en 5, in de servicemesh (denk aan ISTIO) en in laag 3 via de API Gateway.

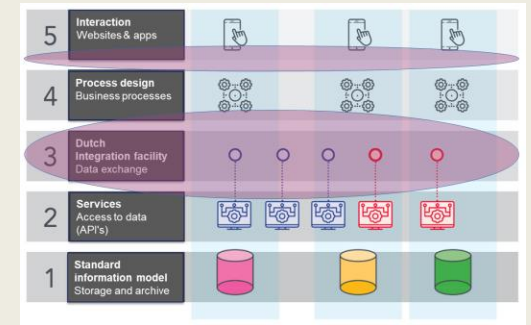


CommonGround en Zero trust

Policy management vindt plaats binnen laag 3, waar de data exchange tussen applicatiesfuncties en gegevens plaatsvindt.

Hierin moet veelal rekening gehouden worden met de NL API Strategie.

<https://docs.geostandaarden.nl/api/API-Strategie/>



Voor de communicatie tussen de laag 4 en 5 hebben we te maken API's. Hierbij dient rekening gehouden te worden met de referentie architectuur zoals op gesteld door de VNG.

<https://github.com/VNG-Realisatie/RAWA>

Doelstellingen Technische analyse

1. Haalbaarheid open source oplossing onderzoeken.
2. Alternatieve componentkeuze onderzoeken.
3. Beheersbaarheid van policies onderzoeken.
4. Datavoorziening (Policy Information Point) mogelijkheden onderzoeken.



1. Haalbaarheid toepassing open source

- Gekozen voor het onderzoeken naar een referentie implementatie, ter behoeve van laagdrempelige functionele oplossing.
- Kong Gateway en OPA zijn hier als invulling gekozen, gezien de eerdere ervaring.
- Open-source plugin aangepast en voorziet nu in terugkoppeling in zowel querystring als header.
- Conclusie: het is haalbaar

OPA plugin:

<https://github.com/EdgaratSB/kong-plugin-opa>



```
if res.result and res.result.headers then
  for key,value in pairs(res.result.headers) do
    kong.service.request.add_header(key, value)
  end
end

if res.result and res.result.headers and res.result.headers.qparams then
  kong.service.request.set_raw_query(res.result.headers.qparams)
end
```


2. Alternatieve componentkeuze

Er zijn drie alternatieven onderzocht voor de gateway component:

Layer7:

- API gateway momenteel in gebruik. Voorziet in de eisen m.b.t. input filtering. Schaaft alleen niet even goed op in vergelijking tot nieuwere gateways.



Istio:

- Service mesh, voorziet niet in de mogelijkheid om requests te transformeren. Hiermee is policy-based input filtering uitgesloten.



Kong Kuma

- Service mesh, voorziet wel in de mogelijkheid om requests te transformeren. Voldoet aan eisen.



Conclusie: request transformatieve harde eis en alternatieve componentkeuze mogelijk. Wel uitdagingen met service meshes.

Input filtering	Schaalt op
+	-
-	+
+	+

3. Beheersbaarheid policies

- Gezien de keuze voor OPA, volgde onderzoek naar Styra DAS. De premium beheersmodule ontwikkeld door Styra, tevens oorspronkelijke ontwikkelaars van OPA.
- Van Styra DAS zijn de mogelijkheden voor opschalen en werkbaarheid onderzocht, en goed bevonden.
 - In memory datavoorziening beschikbaar.
 - Policy ontwikkeling en evaluatie gemakkelijk middels decision replay functie.
 - Meerdere systemen in real time beheren, met logging en monitoring capabilities.

The screenshot displays the Styra DAS web interface. On the left, a sidebar shows the workspace structure with 'sonicbee' as the selected system. The main area is divided into two panes. The top pane shows a Rego policy rule named 'rules.rego' with the following content:

```
package rules
import data.dataset
default allow = false
allow {
  true
}
profielen_data := { profiel | dataset.profielen[profiel] }
profielen_token := {profiel | profiel := token(input.token)[_]}
matching_profielen := profielen_data & profielen_token
rollen := dataset.profielen[matching_profielen[_]]
actieve_rollen := { authz | authz := dataset.rollen[rollen[_]] }
```

The bottom pane shows the 'Replaying logged event...' section. It includes a table with 'Mock', 'Input', 'Output', 'Stacks excluded', and 'Print' columns. The 'Input' column shows a JSON object with 'input' and 'token' fields. The 'Output' column shows the resulting JSON object after evaluation, including 'actieve_rollen', 'allow', 'allowDetailed', 'authz', 'matching_profielen', and 'profielen_data'.

4. Herbruikbaarheid van policies

- Tot slot diende de herbruikbaarheid van policies aangetoond te worden.
- Niet alle gemeentes beschikken over ontwikkelcapaciteiten, vooral gezien *rego* een *niche* programmeertaal is.
- De policy van de hackathon is herschreven met herbruikbare componenten. (zie rode lijnen rechter afbeelding)
- Deze herbruikbare componenten kunnen in een library in Styra DAS, of in GitHub gedeeld worden met anderen.

Conclusie: herbruikbaarheid is mogelijk

```
package example

default allow = false

rollen := {"SZW.MEDEWERKER":{"https://openzaak-cg.test.denhaag.
rollen_flat := { role | rollen[role] }
ys := split(input.request.http.headers.authorization, " ")
token := io.jwt.decode(ys[1])
v := { role | role := token[1].roles[_] }
m := rollen_flat & v
#m := "OCW.MEDEWERKER"
#t := rollen[m]

t := concat(" ", rollen[m[_]])

heeftRol {
  count(m) > 0
}

allowDetailed = response {
  heeftRol
  response := {
    "allow": true,
    "headers": {
      "header-from-opa": t,
    },
  }
}

allowDetailed = response {
  not heeftRol
  response := {
    "allow": false,
    "status": 418
  }
}
```

```
package example

allow = result {
  hasRole
  result := {
    "header_role_string": header_role_string
  }
}

#----- Helper rules

roles_data := { role | data.roles[role] }

roles_token := roles_in_token(token(input.request.http.headers.authorization))

matching_roles := roles_data & roles_token

header_role_string := concat(" ", data.roles[matching_roles[_]])

hasRole {
  count(matching_roles) > 0
}

#----- Library

allowDetailed = response {
  result := allow
  response := {
    "allow": true,
    "headers": { "header-from-opa": result.header_role_string },
  }
}

allowDetailed = response {
  not allow
  response := {
    "allow": false,
    "status": 418,
  }
}

roles_in_token(token) = roles {
  roles := { role | role := token.roles[_] }
}

token(header) = t {
  ys := split(header, " ")
  t := io.jwt.decode(ys[1])[1]
}
```

Herschreven
herbruikbare
componenten

Conclusie

	Doel	Resultaat	
1	Haalbaarheid open source oplossing onderzoeken	Open source referentieimplementatie haalbaar, ook met input filtering mogelijkheid	+
2	Alternatieve componentkeuze onderzoeken	Alternatieve componentkeuze mogelijk. Generieke eis voor de gateway is request transformatie	+
3	Beheersbaarheid van policies onderzoeken	Styra DAS bevordert het beheersen en het hergebruiken van policies	+
4	Datavoorziening (Policy Information Point) mogelijkheden onderzoeken	Vanuit OPA zijn externe databronnen (API's, databases, etc..) beschikbaar. Styra DAS biedt in-memory dataservice aan voor OPA, waarvan de performance veel hoger ligt	+

Vervolgonderzoek

De openstaande onderzoeksvraag die uit de technische analyse naar voren is gekomen en een mogelijk vervolg onderzoek vraagt:

- In welke mate is het wenselijk policy-based input filtering toe te passen op service-mesh niveau?



Ervaringen vanuit use-cases

'Ooievaarspas' en 'Financiële trainingen'

Use-cases policy

- De policy die ontwikkeld is ontwikkeld met een focus op herbruikbaarheid.
- Kan dynamisch JSON data consumeren en op basis hiervan autorizaties bepalen en input filters terugkoppelen.
- Deze policy is onafhankelijk van de use case in te zetten, en hiermee dus maximaal herbruikbaar.

```
package opa.test

#Importeren JSON data.
import data.opa.test.dataset

#Standaardwaarden, niet relevant voor de policy. Hier kan eventueel een toegangsconditie in opgenomen worden.
default allow = false

allow {
  true
}

#Ophalen van de token data, door het invoeren van de meegestuurd JWT token in de "token" functie.
token_data := token(input.token)

#Ophalen van alle rollen uit de token
authorizaties_token := {role | role := token_data.realm_access.roles[_]}

#Ophalen van alle rollen uit de authorizatiematrix. (JSON Dataset)S
authorizaties := {authz | dataset[_][_][_][authz]}

#Formuleren van een lijst van rollen die meegestuurd zijn met de token, waarvoor er autorizaties beschikbaar zijn in de authorizatiematrix.
authorizaties_match := authorizaties & authorizaties_token

#Van de overeenkomstige rollen worden de autorizaties opgehaald, in onderstaand geval de autorizaties voor GZAC.
caseNames := {auhtz | auhtz := dataset[_][_][_][authorizaties_match[_]]["GZAC"][_]}

#De autorizaties worden doorgegeven aan GZAC voor input filtering in de response.caseNames variabele.
document_definition_search_filter = response {
  allow
  response := {
    "caseNames": caseNames
  }
}

#In het geval dat er geen toegang verleend wordt, wordt een lege lijst doorgegeven aan GZAC met de response.caseNames variabele.
document_definition_search_filter = response {
  not allow
  response := {
    "caseNames": []
  }
}

#Functie voor het decoden van de JWT token
token(header) = t {
  t := io.jwt.decode(header)[1]
}
```

Use-case data

- De policy maakt gebruik van een externe databron.
- In deze databron is JSON data opgenomen op basis van een autorizatiematrix.
- Het json schema word bepaald als een standaard waarmee policies generiek herbruikbaar zijn voor alle usecases. Het is dus van belang dat dit zorgvuldig gebeurt. Deze standaard moet goed worden bepaald en vast gelegd omdat de basis van het json schema de policy aanpast.
- Momenteel is dit nog in ontwikkeling en zal dit wanneer mogelijk worden opgeleverd en toegevoegd aan het rapport.

```
{
  "roles": [
    {
      "name": "inzage",
      "cases": [
        {
          "name": "aanvraag-ooievaarspas",
          "openZaakId": "9517e5c0-bc2e-404d-9b12-16ac59f63b8a",
          "accessLevels": [
            "READ"
          ]
        }
      ]
    },
    {
      "name": "ooievaarspas_muteren",
      "cases": [
        {
          "name": "aanvraag-ooievaarspas",
          "openZaakId": "9517e5c0-bc2e-404d-9b12-16ac59f63b8a",
          "accessLevels": [
            "READ",
            "READ_PLUS",
            "WRITE"
          ]
        }
      ]
    },
    {
      "name": "trainingscreatie_muteren",
      "cases": [
        {
          "name": "trainingscreatie_muteren",
          "openZaakId": "e470b637-44b5-46cc-8043-43ddb45126c6",
          "accessLevels": [
            "READ",
            "READ_PLUS",
            "WRITE"
          ]
        }
      ]
    }
  ]
}
```


JSON data genereren

- Er is vastgesteld dat niet alle gemeenten in staat zijn om autorizatiematrizen om te zetten naar JSON bestanden. Dit heeft veelal te maken met onbrekende resources binnen IT. Echter is het aanleveren van een JSON bestand noodzakelijk in OPA. Dit valt op een eenvoudige manier op te lossen met behulp van een API.
- De API kan data in JSON format genereren
- Tegen deze API kan aan geprogrammeerd worden, en met low code een front-end voor ontwikkeld worden. Waarmee gebruikersgemak geoptimaliseerd kan worden.

roles		
GET	/roles/	Show All
POST	/roles/	Create
GET	/roles/{id}	Show
PUT	/roles/{id}	Update
DELETE	/roles/{id}	Delete
cases		
GET	/cases/	Show All
POST	/cases/	Create
GET	/cases/{id}	Show
PUT	/cases/{id}	Update
DELETE	/cases/{id}	Delete
access_levels		
GET	/access_levels/	Show All
POST	/access_levels/	Create
GET	/access_levels/{id}	Show
PUT	/access_levels/{id}	Update

Output van de API

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8000/roles/' \
  -H 'accept: application/json'
```

Request URL

```
http://localhost:8000/roles/
```

Server response

Code

Details

200

Response body

```
[
  {
    "roles_name": "inzage",
    "cases_name": [
      {
        "cases_name": "openzaak",
        "access_levels": [
          {
            "access_levels_name": "READ"
          },
          {
            "access_levels_name": "WRITE"
          }
        ]
      },
      {
        "cases_name": "gzac",
        "access_levels": [],
        "cases_idenfier": "aanvraag-ooievaarspas"
      }
    ],
    "cases_idenfier": "9517e5c0-bc2e-404d-9b12-16ac59f63b8a"
  }
]
```



Download

Grote versus kleine gemeenten

Processen

- Ontwerp en implementatie van bedrijfsprocessen is per gemeente anders ingericht.
- De beschreven methode om te komen tot vastgestelde access policies is generiek genoeg om bij zowel grote als kleine organisaties toegepast te worden.
 - Wel zal snel duidelijk worden waar governance-vragen onbeantwoord blijven. Dat zijn echter geen technische vragen.
 - Bovendien leert de ervaring dat deze beantwoording bij zowel grote als kleine organisaties een uitdaging is. Maar die governance-uitdaging bestaat dan feitelijk nu ook al.
- Daar waar procesmodellering is uitbesteed aan een externe partij, zal de aanpassing op deze uitbestede dienst zeer beperkt zijn.
- De beschreven methode is toepasbaar, ongeacht de omvang van de gemeente.

Technologie

- Implementatie van het concept van PBAC impliceert dat ook aanvullende technische componenten moeten worden ingezet. Het betreft moderne componenten waarvan niet elke organisatie beschikt over de technische know-how en ervaring, waardoor zelf-doen nauwelijks een optie is.
 - Componenten zijn de API-gateways en de 'policy engine'.
 - Deze moderne componenten vergen ook toegang tot een API op basis van federatie, dat vergt andere expertise dan regulier role based access control. (technische) kennis van OpenID Connect en OAuth is essentieel.
 - Opbouwen van dergelijke kennis is hoe dan ook een noodzaak, los van de vraag of PBAC ingezet wordt.
- Ondersteunende ICT-kennis is wellicht niet voorhanden in de eigen organisatie. Dit impliceert dat er zal moeten worden samengewerkt met grotere organisaties en leveranciers.
 - De ervaring leert dat de reguliere leveranciers nog niet ver zijn met ontwikkeling en ondersteuning van dit concept.

Conclusies en aanbevelingen

Conclusies en aanbevelingen

- Binnen de bestaande werkwijze van procesmodellering van gemeente Den Haag zijn de concepten van policy based access control betrekkelijk eenvoudig toe te passen.
- Met de bestaande open source-oplossingen is PBAC met input-filtering mogelijk. Maar voor beheer van policies is een (aanvullende) commerciële component (in casus Styra DAS) noodzakelijk.
- Knelpunt is dat verantwoordelijkheden in geval van geautomatiseerde verwerking expliciet binnen de organisatie belegd moeten worden, hetgeen wellicht een wijziging op het gebied van governance betekent. Dit zou verder onderzocht moeten worden, maar dit valt buiten scope voor het concept van PBAC.
- De gepresenteerde methode is bruikbaar voor zowel grote als kleine gemeenten. Technische implementatie is afhankelijk van de beschikbare technische expertise waarover een gemeente kan beschikken.

Advies ten aanzien van APIs

- De bestaande API's zijn te globaal gedefinieerd om fijnmazige toegang mogelijk te maken. Toepassing van Policy Based Access Control met input-filtering kan de lancune oplossen. Maar dat is een compenserende maatregel, feitelijk zouden API's moeten worden ontworpen vanuit het VNG RAWA-concept
- Applicaties moeten zodanig worden ontworpen dat de toegangsregels (inclusief concepten als Role Based Access Control) zich daarbuiten bevinden. Applicaties en services moeten niet worden gebouwd met een security-focus.
- APIs moeten zodanig worden ontwikkeld dat er fijnmazige toegangsbeheersingsparameters kunnen worden toegevoegd.
 - Een beter alternatief is dat endpoints zonder access control (dus zonder identiteit en autorisatie, bijvoorbeeld 'rollen') beschikbaar komen.

Het standaard policysjabloon

- Tijdens het project is aangetoond dat Policy Based Access Control met, indien noodzakelijk, input-filtering, kan werken. Hiervoor is een conceptsjabloon opgesteld dat voor de policies als basis gebruikt zou kunnen worden.
- Zero Trust impliceert dat zonder te voldoen aan de toegangspolicy toegang wordt geweigerd, dit impliceert dat een 'default deny' regel in de policy wordt beschreven. Elke policy zal deze regel als eerste toepassen: 'Geen toegang, tenzij'.

- [Policy entry]
 - Default deny
- [Validate IdP / authoritative source]
- [Point to applicable standard policies / policy lines]
- [Validate access]
- [Indien input-filtering nodig]
 - Construct filter

Appendices

Vragenlijst 1: Bepalen API

- Welke API wordt benaderd?
- Welke parameters kunnen aan de API worden aangeleverd?
- Wat zijn naam en URI van de API?

Vragenlijst 2: Bepalen actor

- Wordt de processtap/taak uitgevoerd door een geautomatiseerd proces?
 - Zo, ja: bepaal wie laatst voorgaande menselijke actor is binnen de processflow

Vragenlijst 3: Functiescheiding

- Wat is de aard van de taak:
 - Beschikken
 - Bewaren
 - Registreren
 - Uitvoeren
 - Controleren
- Is uitvoering van de taak door de actor strijdig met het principe van functiescheiding gegeven de uitvoering door de actor van de vorige taak?
 - Zo ja: afwijzing met omschrijving Doorbreking functiescheiding

Vragenlijst 4: Attributen bepalen

- Wat is de rol van de actor?
- Wat is het zaaktype van de taak binnen de usecase?
- Is er een rol/zaaktype bekend?
- Is er een input-filtering voor deze zaaktype bekend?

Vragenlijst 5: Bepalen noodzaak inputfiltering

- Welke stakeholders vragen output van de API?
- Wat is de output van de api?
- Kunnen we op basis van het subject de verwachte output bepalen? Oftewel: is inputfiltering mogelijk?