# Federal Data and REST Standardization

The session will be recorded

April , 26th 2021

# Content

- **G-Cloud Federal Service Platform** (Bob Lannoy – SMALS) (10’)
- **Why REST ?** (Peter Van Den Bosch – KSZ) (10’)
- **Functional standards** (federal, regional, interfederal – vocabularies) (Marc Bruyland – BOSA) (20’)
- **Technical standards** (REST styleguide, datatypes) (Peter Van Den Bosch – KSZ) (35’)
- **Secure REST** (authentication, authorization, signing, encryption) (Cihan Kucukkececi – BOSA) (15’)
- **Practical examples** where we could improve (Peter Van Den Bosch – KSZ) (15’)
- **Q&A** (15’)


# G-Cloud Federal Service Platform

# G-Cloud Portfolio

## SOFTWARE - STANDAARD COMPONENTEN EN TOEPASSINGEN

- BABELFED
- UCC SHAREPOINT BASIC O365
- REGISTER
- IT SERVICE MANAGEMENT
- BECONNECTED
- SHAREPOINT COMMUNITY PLATFORM
- CSAM AANMELDEN, BTB, SSM
- IWF INTELLIGENT WEB FORMS
- WEB CONTENT MANAGEMENT

## PLATFORM - ONTWIKKELING EN GESPECIALISEERDE TECHNISCHE TOOLS

- GREENSHIFT OPEN SOURCE PAAS
- YELLOWSHIFT MICROSOFT PAAS
- BLUE STACK DB IBM DB
- SERVICE PLATFORM

## INFRASTRUCTUUR - SERVICES ("ZACHTE" INFRA)

- UCC VOICE/IM
- UCC EXCHANGE ON PREMISE
- SHAD SHARED DIRECTORY
- ARCHIVING
- UCC CONTACT CENTER
- UCC EXCHANGE O365
- INTERNET ACCESS PROTECTION SECURITY AS A SERVICE
- BACKUP
- UCC MOBILE DEVICE MANAGEMENT

## INFRASTRUCTUUR - FUNDAMENTEN ("HARDE" INFRA)

- COMPUTE VIRTUAL MACHINE
- HOUSING
- FEDWAN
- STORAGE
- COMPUTE HYPERVISOR
- FEDMAN

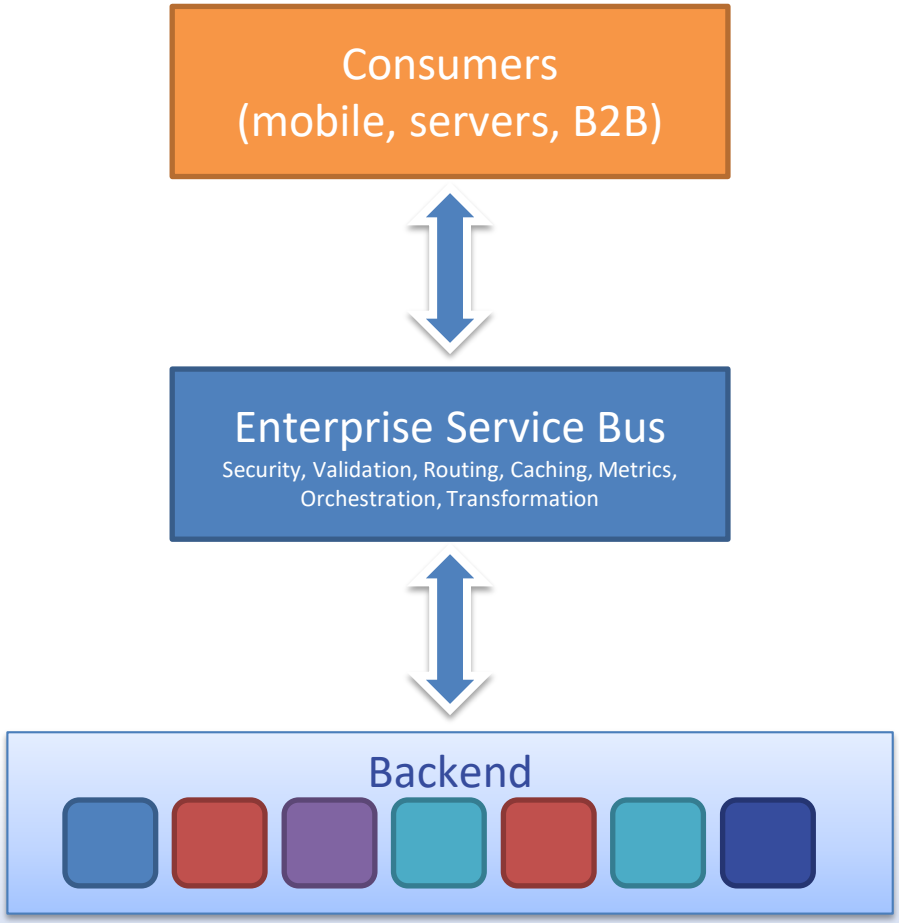
# Some history ...

- 2016/2017
  - Widespread use of "Enterprise Service Bus" technology at BOSA, eHealth, RSZ, Minfin, KSZ, ...
  - Different vendors: Oracle, IBM, ...
  - ≠ use cases (service integrator, isolation layer internal/external world, authentic source)
  - ≠ standards, protocol sets
  - Transaction volume ↑ ( -> licence cost ↑)

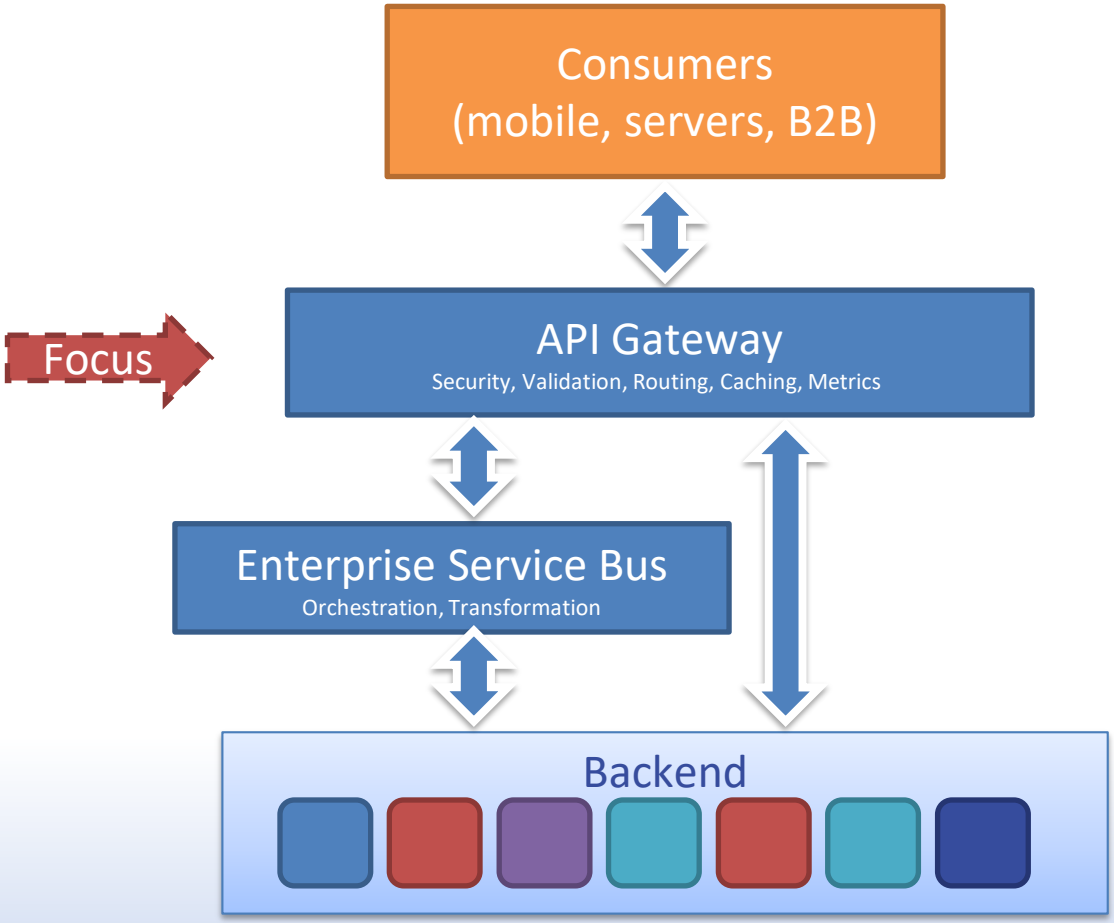
➔RFP awarded to Sopra Steria

  - **Axway API Gateway**
  - RedHat Fuse (ESB)

# Technical landscape – pre-2018



# Technical landscape – post-2018



# Technological shift

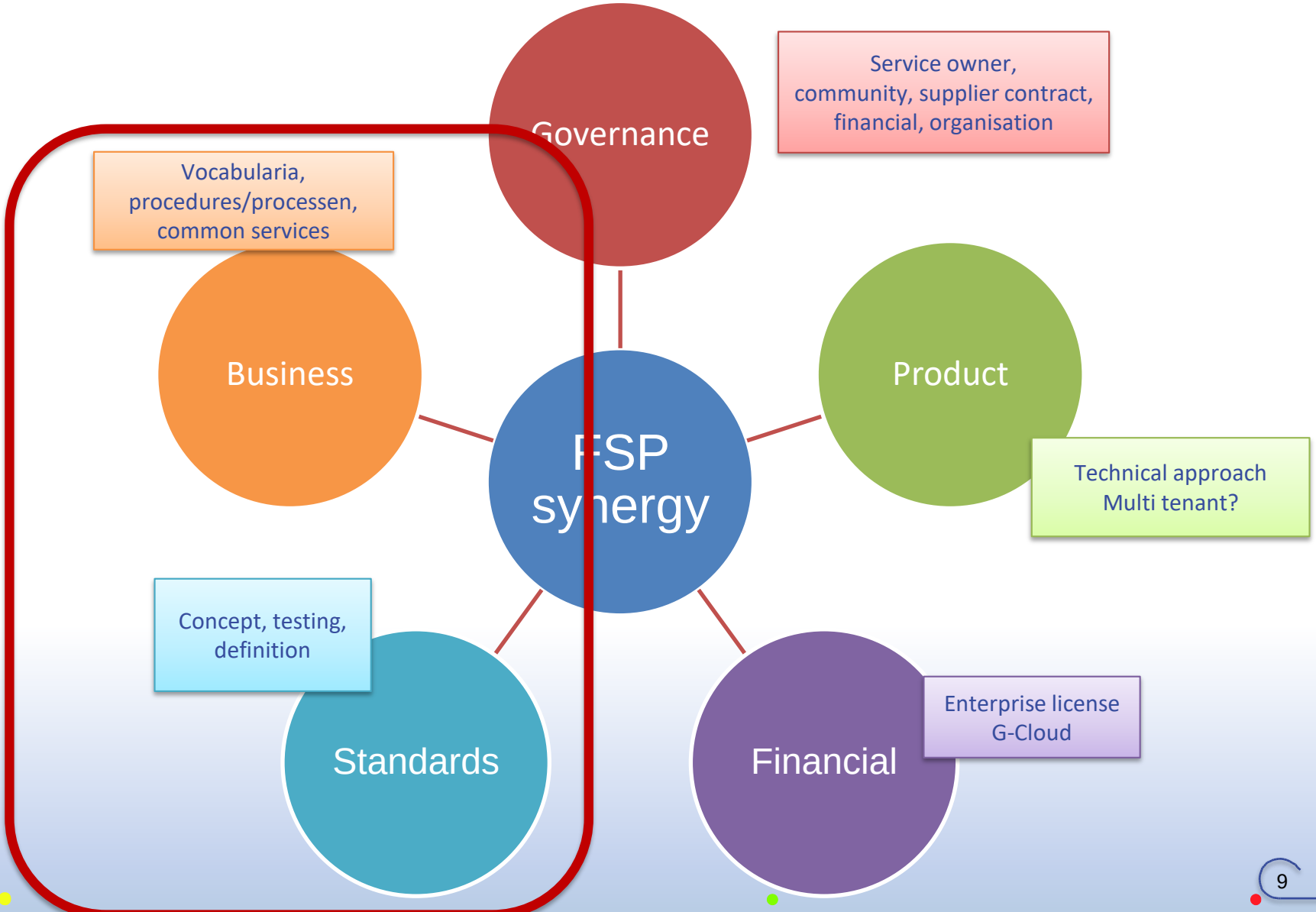
- SOAP+XML -> REST+JSON
- REST

*Protocol* *Payload*

  - De facto industry standard (both supplier and customer side)
  - Focused on modern architectures (mobile)
- API Gateway solution supports both standards (migration path)

- Opportunity to work together on product implementation & standardisation efforts

# Synergy on several levels



# Participants

- BOSA
- eHealth
- KSZ
- RSZ / Smals
- Minfin

+ Justice

+ SFPD

# Working groups

- REST Design / Error handling
  - Standard approach towards REST services
  - Making choices
- Functional aspects: vocabularies
- REST Security aspects

- Main goals
  - Interoperability ↑
  - Time to market ↑


# Why REST ?

# “REST API” or “RESTful webservices”

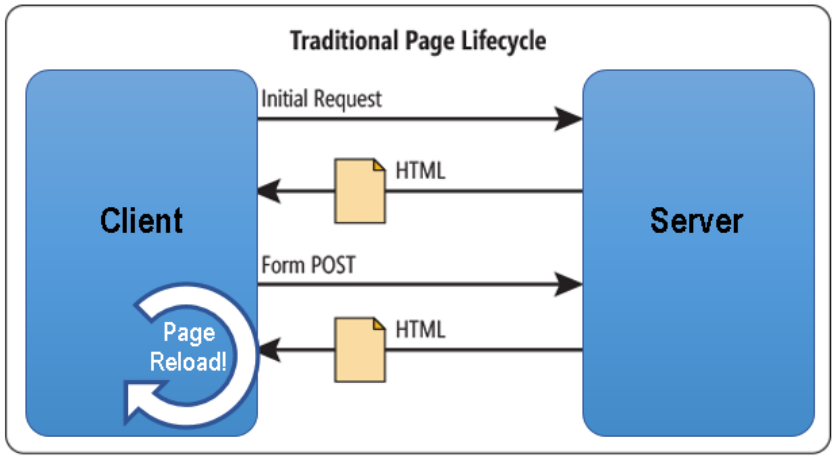
- REST (Representational State Transfer)
  - Type of service architecture: resource oriented
    - like SOA, microservices (not mutually exclusive)
  - Typically using HTTP and JSON
- API = Application Programming Interface
  => an interface/service that can be called from a program

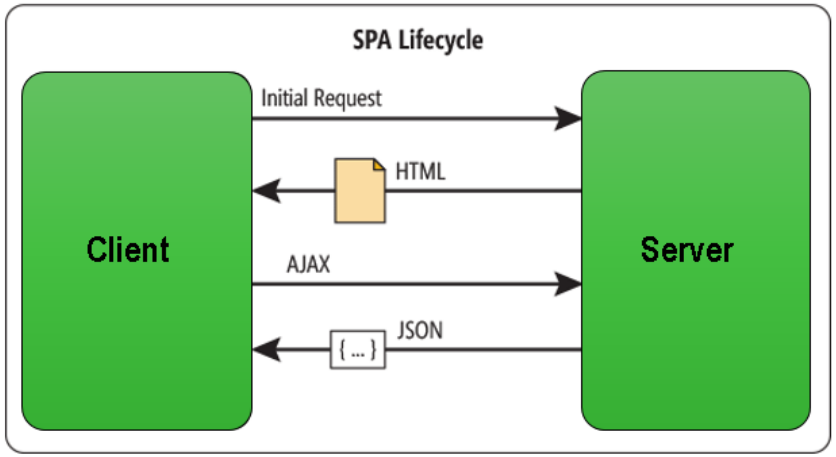
| | REST API | SOAP webservice |
|---|---|---|
| Exchange protocol | https | SOAP (over https) |
| Format of messsages | JSON | XML |
| Service/message structure definition | OpenAPI (Swagger) | WSDL and XSDs |
| Security protocols | server TLS and OAuth2 | mTLS and WS-Security |

# Why REST?

- REST is simpler, less strict and easier to implement
  - thin layer over HTTP, fully uses HTTP features
  - can be used when HTTP and JSON parser is available
    vs WS-* specifcations (e.g. MTOM/SwA) requiring SOAP libraries
- REST APIs can be used from ‘thin clients’ (JavaScript, browser) and mobile applications (Android, iOS)
- Industry moved to REST, SOAP became 2nd class
- ! REST-related standards are still evolving (OpenAPI)

# Single Page web Applications




- Server is (mostly) stateless
- View layer only in browser
- Model layer in REST APIs

Access control at REST API level: easier to enforce

# Functional standards

- **European standards – Core vocabularies**
  - https://ec.europa.eu/isa2/solutions/core-vocabularies_en
- **Interfederal standards at ICEG**
  - https://github.com/belgif/review
  - https://github.com/belgif/thematic
- **Federal functional standard**
  - https://github.com/belgif/fedvoc
- **Regional standards**
  - https://data.vlaanderen.be/standaarden/

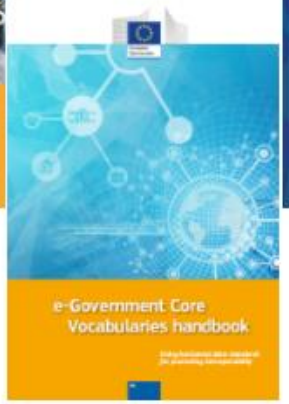
# European Core vocabularies

https://ec.europa.eu/isa2/solutions/core-vocabularies_en

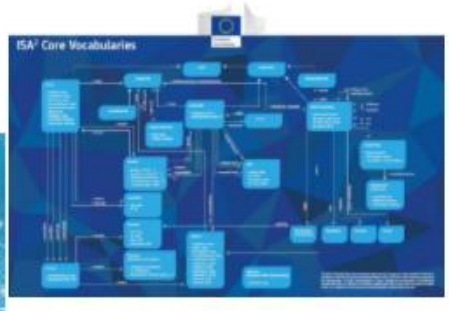
**Our solution at a glance**

Core Vocabularies leaflet

Core Vocabularies handbook




Core Vocabularies poster




**Get started**

Download the latest version of the Core Vocabularies via Joinup:

- Core Person Vocabulary
- Core Business Vocabulary
- Core Location Vocabulary
- Core Criterion and Core Evidence Vocabulary
- Core Public Organisation Vocabulary

# Interfederal standards at ICEG

https://github.com/belgif/review , https://github.com/belgif/thematic

**ICEG** = collaboration agreement between federal, regional and community levels
Scope = **e-government**

## ICEG Review Group

Mission statement and roles

The review group 'open standards' has a permanent character and is responsible for the central coordination and follow-up of the work related to the standardisation of information.

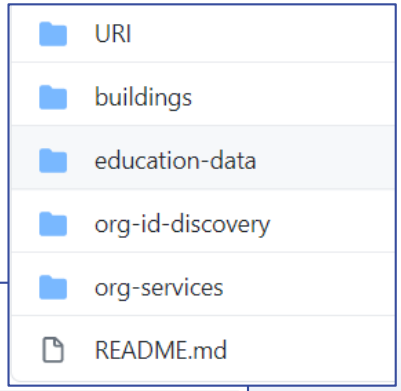
Mission aligned to the existing collaboration agreement between the federal, regional and community authorities (dd. 2013-08-26).

The work is part of the standardisation of: meaning of the information (semantic), syntax (grammar) and technical standards for the exchange of the information metadata for discoverability ('data on data').

In addition, the working group monitors mutual consistency of standards, international standards that impact governments in Belgium generic development and the change process.

The working group on data standards gathers on a regular basis.

Organizer of the review meeting (round-robin): BOSA => Flanders => CFWB

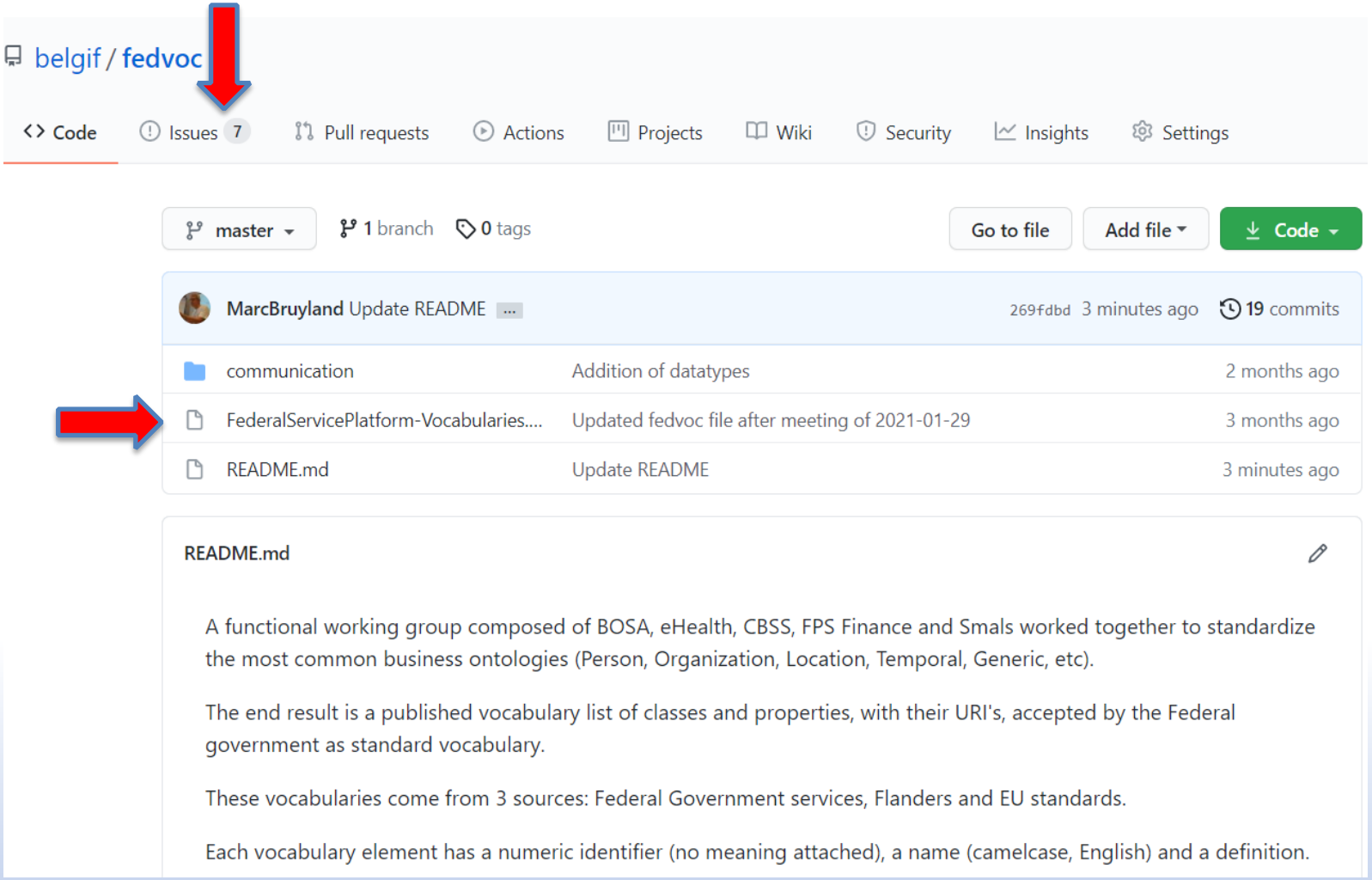



## Thematic Working Groups

This group of experts with knowledge of existing data models and implementations is responsible for the development of the domain model.

# Federal functional standard

https://github.com/belgif/fedvoc



# Federal functional standard

https://github.com/belgif/fedvoc ⇨ Excel

- Intro
- Standard
- Draft
- Government institutions
- Vocabulary
- VocabularyNL
- VocabularyFR
- VocabularyTranslations
- VocabularyMapping
- VocabularyAdoption
- Datamodels
- Prefix
- List
- PrepPublished

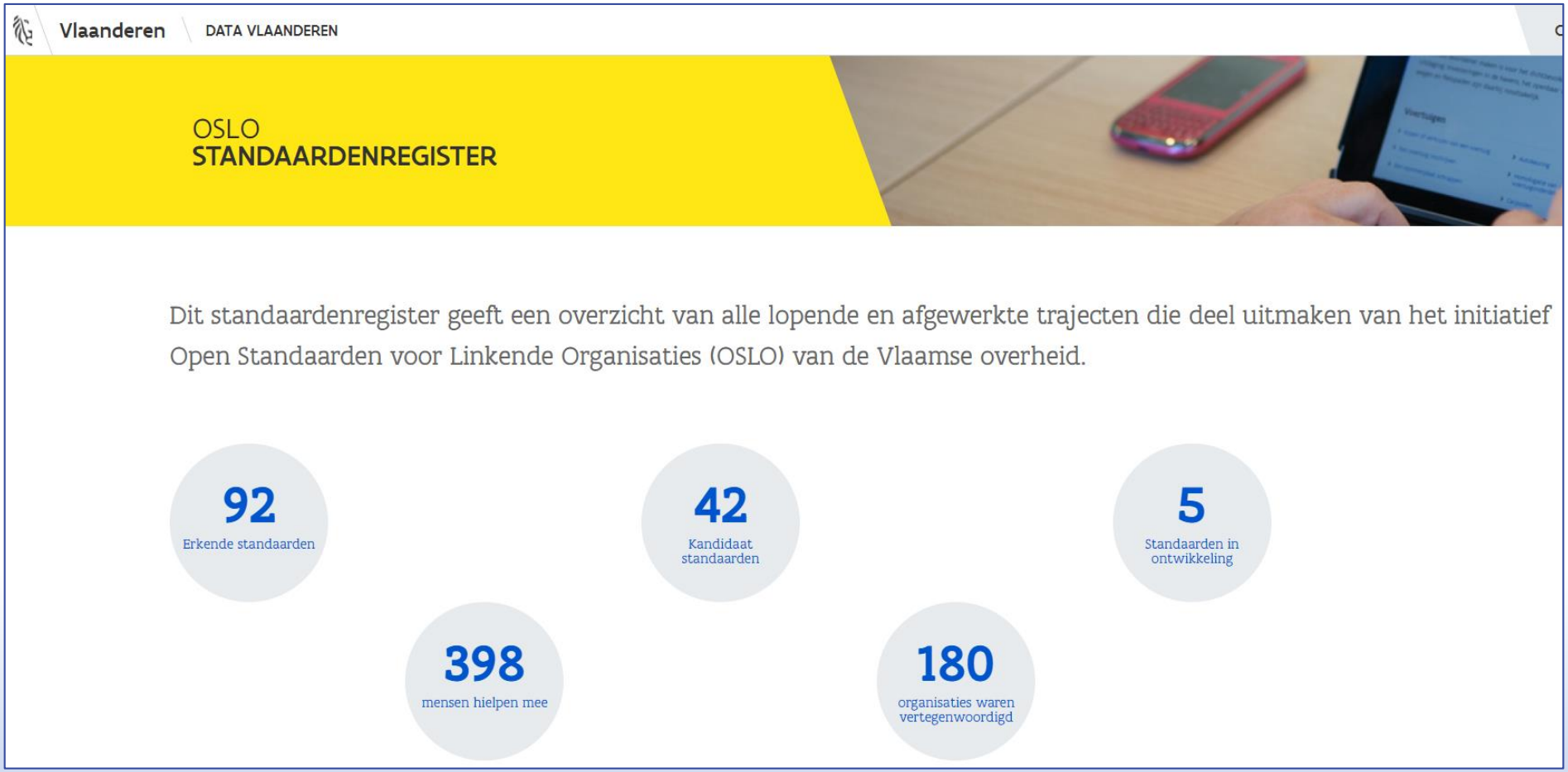
Ontologies:

- Generic
- Location
- Organization
- Other
- Person
- Temporal

| | |
|---|---|
| **Id** | 662 |
| **Ontology** | Location |
| **Type** | Class |
| **URI** | <http://inspire.ec.europa.eu/ont/ad#ThoroughfareName> |
| **Name** | StreetName |
| **LabelNL** | Straatnaam |
| **LabelFR** | Nom de la rue |
| **Definition** | An address component that represents the name of a passage or way through from one location to another. A thoroughfare is not necessarily a road, it might be a waterway or some other feature. |
| **DefinitionNL** | Een adrescomponent die de naam is van een passage of een weg door de ene locatie naar de andere. Een verkeersweg is niet noodzakelijkerwijs een weg, het kan een waterweg zijn of een andere functie. |
| **DefinitionFR** | Un composant d'adresse qui est le nom d'un passage ou d'une route passant d'un endroit à un autre. Une route n'est pas nécessairement une route, elle peut être une voie navigable ou une fonction différente. |
| **Comment** | BEST: Address component with the name officially assigned to a street (runway, passage, square) or to a hamlet and to which addresses can be linked.<br>BEST = Belgian standard for adresses.<br>(also see <locn:Thoroughfare>) |
| **CommentNL** | BEST: Adrescomponent met de naam die officieel werd toegekend aan een straat (baan, doorgang, plein) of aan een gehucht en waaraan adressen kunnen zijn gekoppeld.<br>BEST = Belgische standaard voor adressen.<br>(zie ook <locn:Thoroughfare>) |
| **CommentFR** | BEST: Composant d'adresse avec le nom attribué officiellement à une rue (piste, passage, carré) ou à un hameau et auquel des adresses peuvent être liées.<br>BEST = standard belge pour les adresses<br>(voir également <locn:Thoroughfare>) |
| **inSwagger** | no |

# Regional standards

https://data.vlaanderen.be/standaarden/

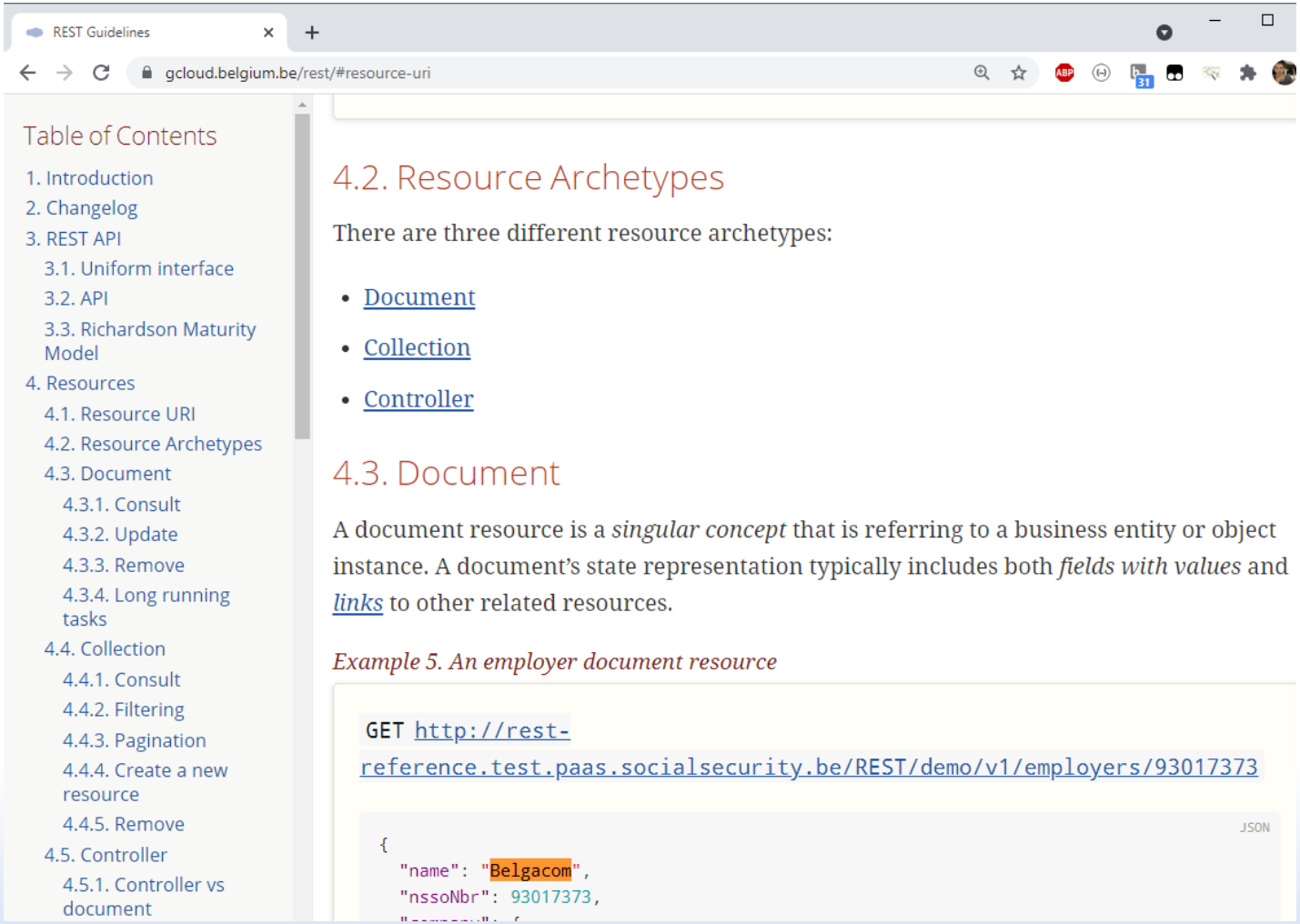



# Technical standards

# Technical standards

- REST guide and datatypes
- REST design WG active since September 2017
  - Meets each 2 months
- Based on existing REST guides, industry best practices, own experiences
- Open Source (Apache License 2.0)

# REST guide




https://www.gcloud.belgium.be/rest/

sources: https://github.com/belgif/rest-guide

# Archetypes

- Document
- Collection
- Controller

- CRUD operations (use of HTTP GET/POST/PUT/DELETE)
- Searching on collections: pagination, filtering
- Long running tasks (longer than HTTP transaction)
- Links between resources

# URI structure

GET https://demo.socialsecurity.be/employerRegister/v1/employers/93017373

/employerRegister/v1 = first version of the personnelFile REST API

/employers/93017373 = the employer with id 93017373


```
{
  "name": "Belgacom",
  "employerId": 93017373,
  "enterprise": {
    "enterpriseNumber": "0202239951",
    "href": "https://demo.kbo-bce.fgov.be/v1/registeredOrganizations/0202239951"
   },
   "self": "https://demo.socialsecurity.be/personnelFile/v1/employers/93017373"
}
```


Don't include full representation of other resources, only their URL

=> more reuse

# Errors

- HTTP status codes
- Problem Details for HTTP (RFC7807) for additional details

# REST guide: other topics

- Embedding other resources in response
- Caching (Cache-Control – ETag)
- Versioning of REST APIs
- Internationalization (I18N)
- Tracing (message id in HTTP header)
  - for troubleshooting and privacy logs
- Events: CloudEvents standard
- API health check for monitoring

# Data types

- Overview: https://github.com/belgif/rest-guide/blob/master/README.md#reusable-openapi-schemas
- Technical data types:
  - common: Uuid, Href, health check, pagination, …
  - problem
  - time (Period, YearQuarter, …)
- Business Data types:
  - based on output of functional WG
  - domains: person, person-identifier, location, organization-identifier, employment-identifier, money
  - e.g. Ssin, EnterpriseNumber, CountryIsoCode…

# Data types

- Released as zip files on GitHub and on Maven Central repository
- Both OpenAPI 2.0 and 3.0 variants

```yaml
# location-v1.yaml
components:
  schemas:
    BelgianRegionCode:
      description: Belgian Region represented by an ISO 3166-2:BE code
      type: string
      enum:
        - BE-BRU
        - BE-WAL
        - BE-VLG
```

```yaml
# example reference
  region:
    $ref: "./belgif/location/v1/location-v1.yaml#/components/schemas/BelgianRegionCode"
```

# Data types

! Not all functional WG vocabulary entries have OpenAPI data types:

Example “Person” :

- does it include address, contact info, …?
- max length of a name?
- address structure varies in each country

=> representation depends on context

Standardized OpenAPI Data types only for frequently used contexts (CBSS/National Registry, BeSt address, …)

When no data type is defined: fedvoc for naming and semantics


# Secure REST

# Secure REST

- Security Aspects

  - **Authentication:** Verifying the identity of the user.
  - **Authorization (Access Control):** Verifying what an authenticated user is allowed to do.
  - **Integrity** - Ensuring the message contents haven't changed in transit. A digital signature is used to validate the signature and provides non-repudiation.
  - **Non-repudiation** - Ensuring that the sender cannot deny having sent the message
  - **Confidentiality (Privacy):** Keeping information secret. Confidentiality and privacy can be achieved by encrypting the content of a message.

# Secure REST
# Authentication & Authorization



- OAuth 2.0 is a *delegation protocol* that is useful for conveying *authorization decisions* across a network of web-enabled applications and APIs. **OAuth 2.0 is not an authentication protocol.**
- OpenID Connect 1.0 is a simple identity layer on top of the OAuth 2.0 protocol. It allows clients to verify the identity of the End-User based on the authentication performed by an Authorization Server.
- OpenID Connect can be used along with OAuth to create an authentication and identity protocol on top of this delegation and authorization protocol.

# Secure REST
# JOSE Framework

JSON Object Signing and Encryption is a framework intended to provide a method to securely transfer data between parties. The JOSE framework provides a collection of specifications to serve this purpose.

JOSE consists of several standards:

- JSON Web Token (JWT) -> RFC7519
- JSON Web Signature (JWS) -> RFC7515 → Integrity, Non-repudiation
- JSON Web Encryption (JWE) -> RFC7516 → Confidentiality
- JSON Web Key (JWK) -> RFC7517

# Secure REST
# JOSE Framework

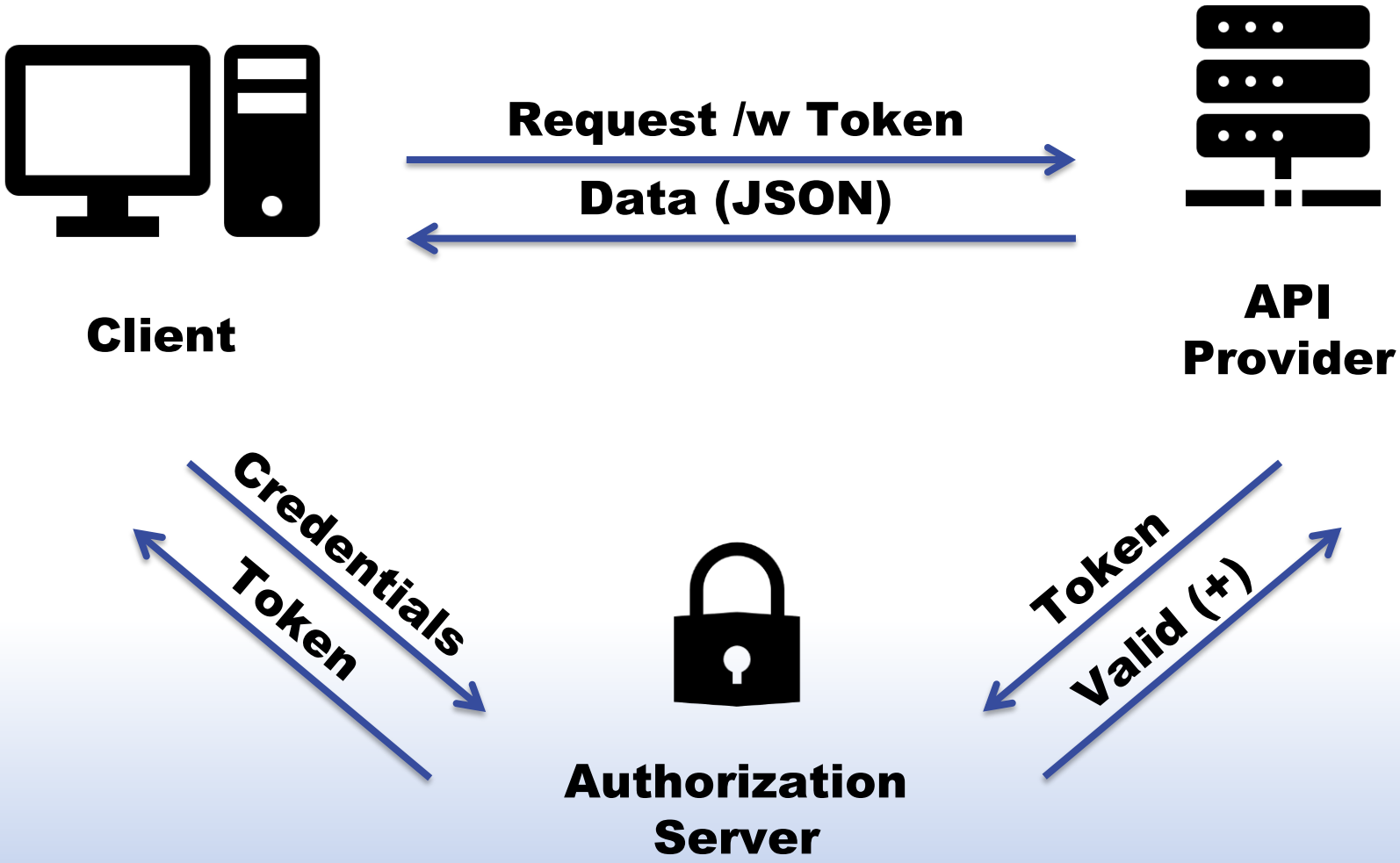
## Multiple Security Levels

- Policy0 - OAuth2 Client Credentials Flow
- Policy1 - OAuth2 JWT Flow
- Policy2 - OAuth2 JWT Flow + JWS (Sign the content)
- Policy3 - OAuth2 JWT Flow + JWE (Encrypt the content)

| Policy Level | Authorization | Integrity | Non-repudiation | Confidentiality |
|---|---|---|---|---|
| Policy0 | Yes (Via Secret) | No | No | No |
| Policy1 | Yes (Via Certificate) | No | No | No |
| Policy2 | Yes | Yes | Yes | No |
| Policy3 | Yes | Yes | Yes | Yes |

# Secure REST

- Policy0



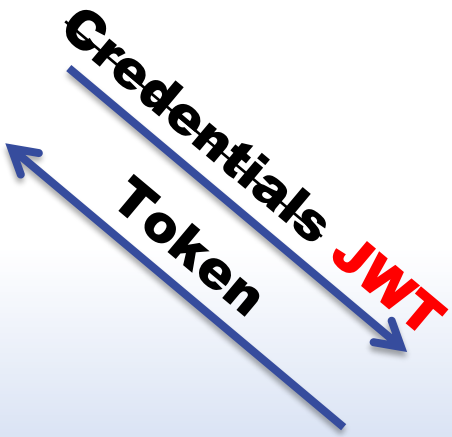
# Secure REST

- Policy1

**Client**

A **JSON Web Token (JWT)** contains information (a set of claims) that can be verified and trusted with a digital certificate.

HEADER . PAYLOAD . SIGNATURE



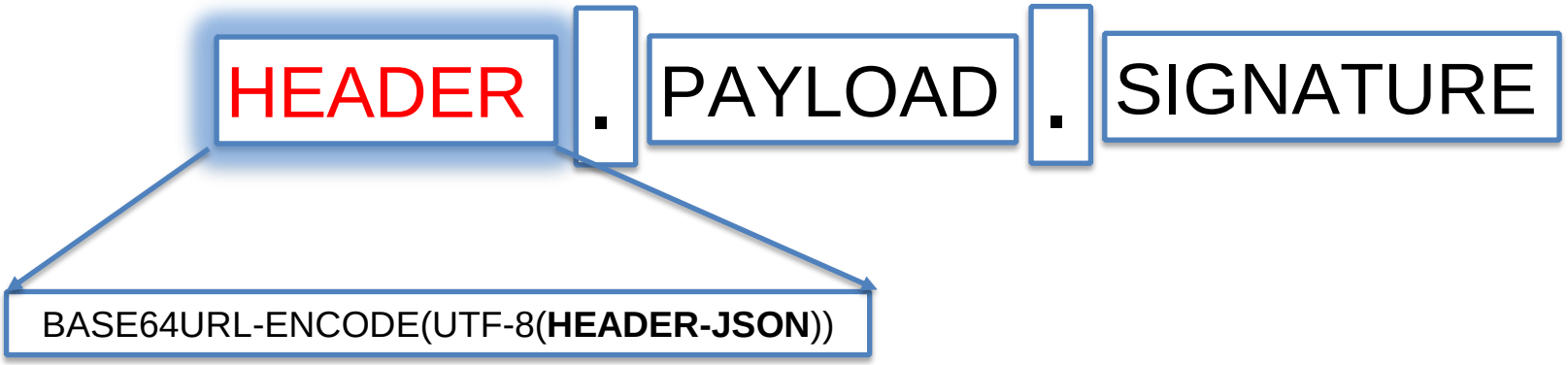

**Authorization Server**

# Secure REST

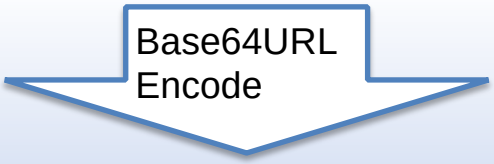
- JWT

# Secure REST

- JWT



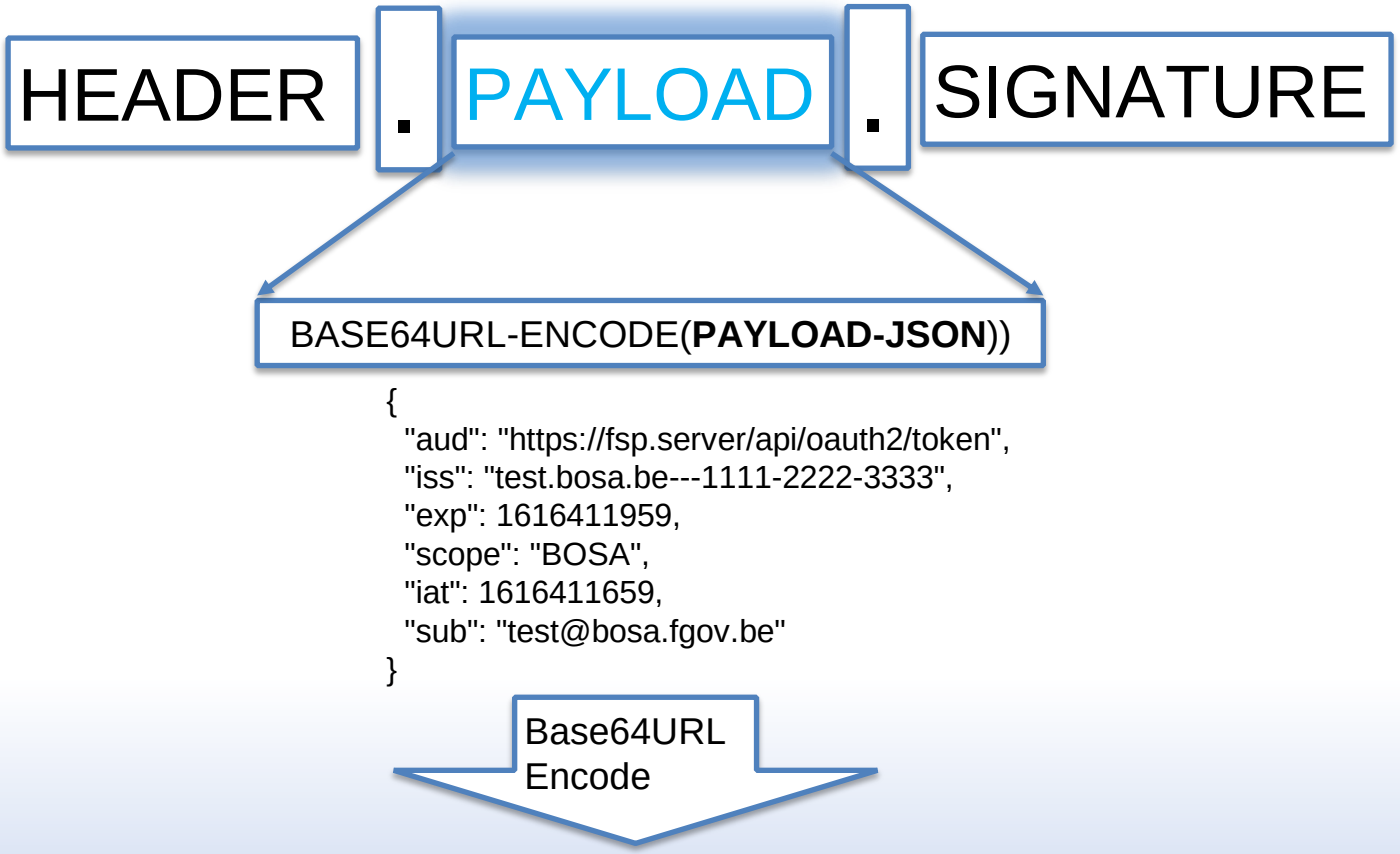

```
{
  "alg": "RS256",
  "typ": "JWT"
}
```




eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9

# Secure REST

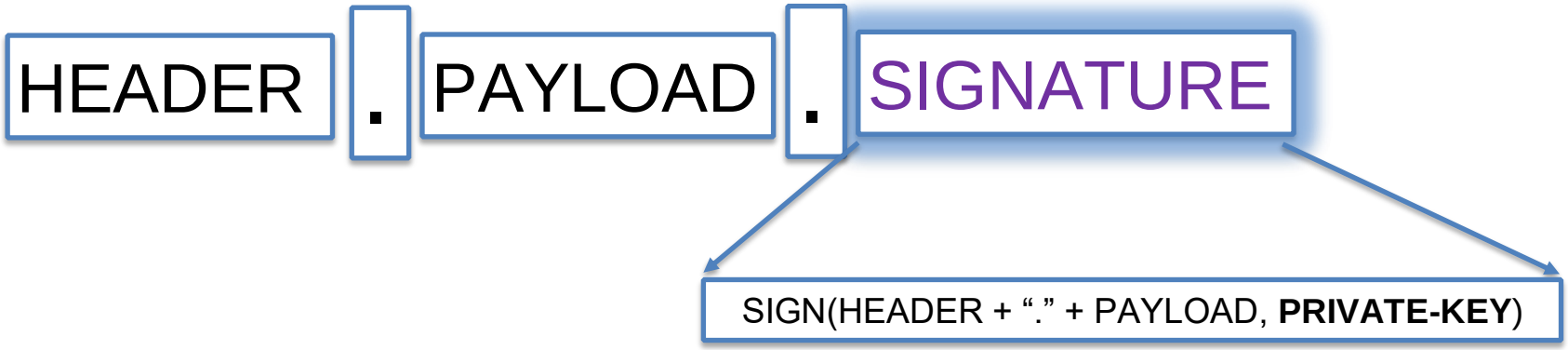
- JWT




eyJhdWQiOiJodHRwczovL2ZzcC5zZXJ2ZXIvYXBpL29hdXRoMi90b2tlbiIsImlzcyI6InRlc3QuYm9z
YS5iZS0tLTExMTEtMjIyMi0zMzMzIiwiZXhwIjoxNjE2NDExOTU5LCJzY29wZSI6IkJPU0EiLCJpYX
QiOjE2MTY0MTE2NTksInN1YiI6InRlc3RAYm9zYS5mZ292LmJlIn0

# Secure REST

- JWT




eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJodHRwczovL2ZzcC5zZXJ2ZXIvYXBpL29hdXRo
Mi90b2tlbiIsImlzcyI6InRlc3QuYm9zYS5iZS0tLTExMTEtMjIyMi0zMzMzIiwiZXhwIjoxNjE2NDExOTU5LCJzY
29wZSI6IkJPU0EiLCJpYXQiOjE2MTY0MTE2NTksInN1YiI6InRlc3RAYm9zYS5mZ292LmJlIn0

AIsYb7jjZ5az_Irg7Mri-_T38xYP-X_syp9B_iWIluVPaIvwxbOTzVBeQ6o2LzvHv0Zl2O0SioX-
hX7Hd0DS7LdJfhzMMkv7L2DH7owanynHU9u78sNRMi9hRpEX67g0LFiltlBVhOrGzVjWmQ8Z-
1xvAnBjBfQiadVd2Om6vR7nsigoDtHRCAIJUp31lMvC9HC0LmxCeAIvCL4LbhjthjtjXik35R7mKc5E0z29JU
HLjjbejPrywG_rYajOz1Ia7kCBT0nm6Lf3b-
VpBas59SJDwRYS1J6KRBap8hAfgjMnnFLXxG2D6yBarwzyjZdxxKyGXDYYJI3woBLkRidocw

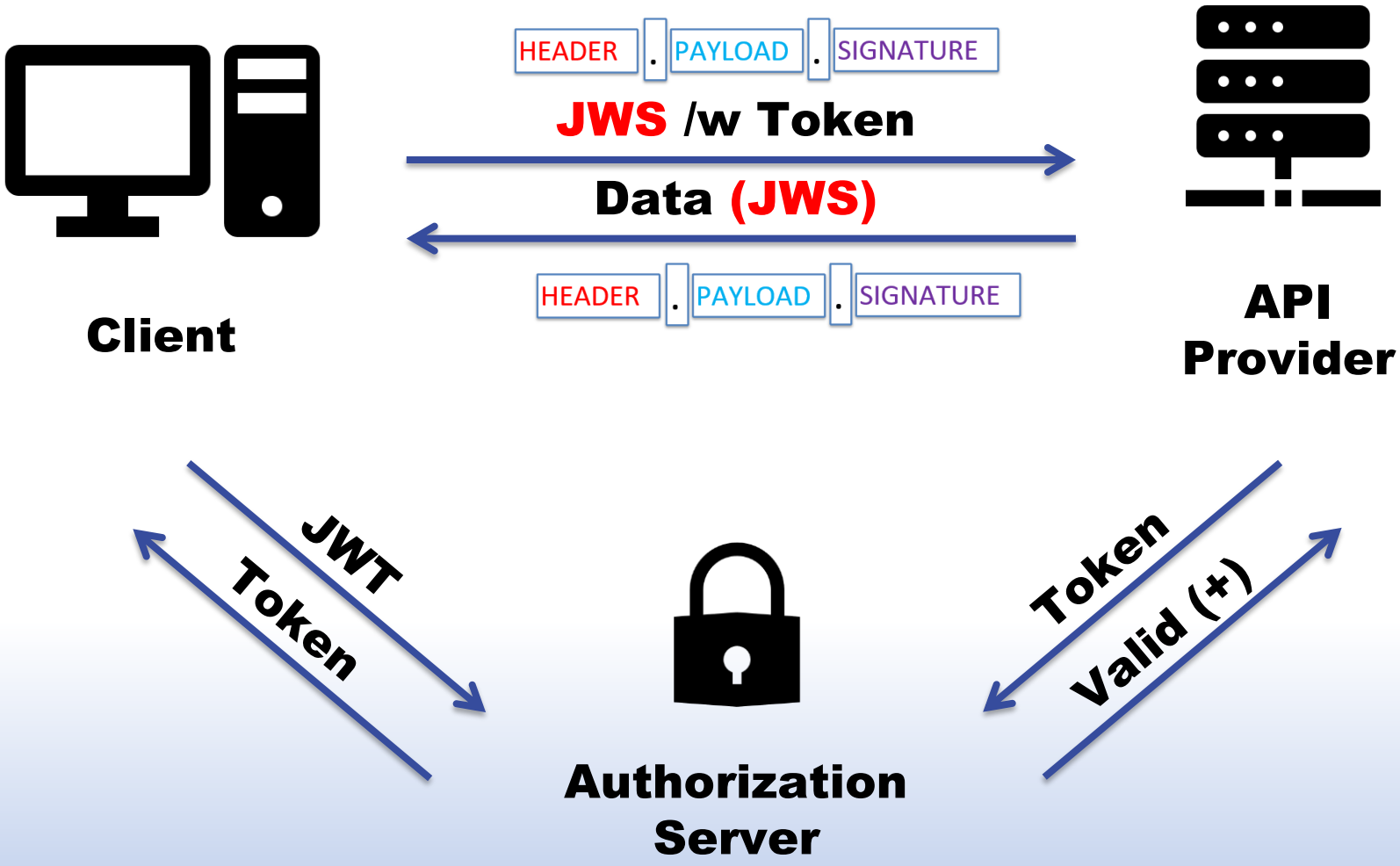
# Secure REST

- JWT


eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJodHRwczovL2ZzcC5zZXJ2ZXIvYXBpL29hdXRoMi90b2tlbiIsImlzcyI6InRlc3QuYm9zYS5iZS0tLTExMTEtMjIyMi0zMzMzIiwiZXhwIjoxNjE2NDExOTU5LCJzY29wZSI6IkJPU0EiLCJpYXQiOjE2MTY0MTE2NTksInN1YiI6InRlc3RAYm9zYS5mZ292LmJlIn0.AIsYb7jjZ5az_Irg7Mri-_T38xYP-X_syp9B_iWIluVPaIvwxbOTzVBeQ6o2LzvHv0Zl2O0SioX-hX7Hd0DS7LdJfhzMMkv7L2DH7owanynHU9u78sNRMi9hRpEX67g0LFiltlBVhOrGzVjWmQ8Z-1xvAnBjBfQiadVd2Om6vR7nsigoDtHRCAIJUp31lMvC9HC0LmxCeAIvCL4LbhjthjtjXik35R7mKc5E0z29JUHLjjbejPrywG_rYajOz1Ia7kCBT0nm6Lf3b-VpBas59SJDwRYS1J6KRBap8hAfgjMnnFLXxG2D6yBarwzyjZdxxKyGXDYYJI3woBLkRidocw

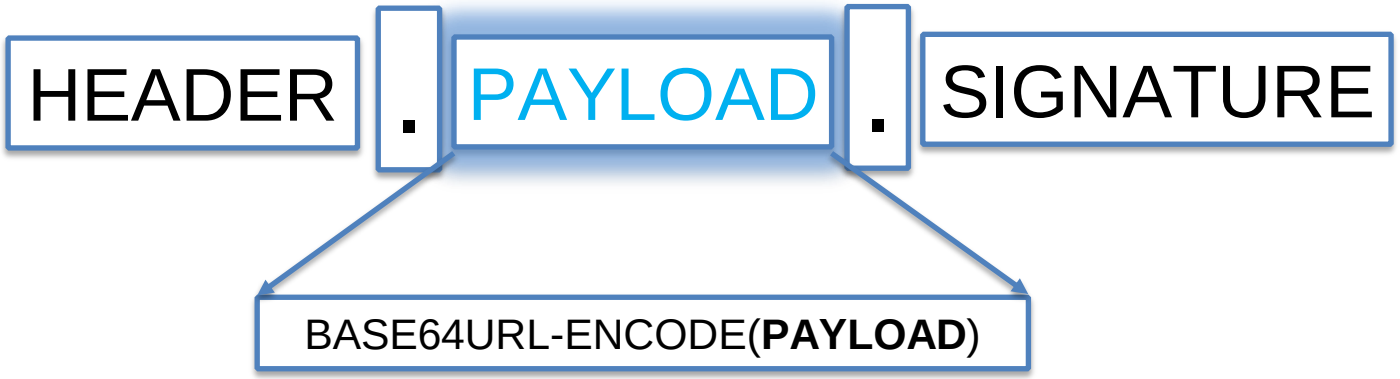
# Secure REST

- Policy2



# Secure REST

- JWS



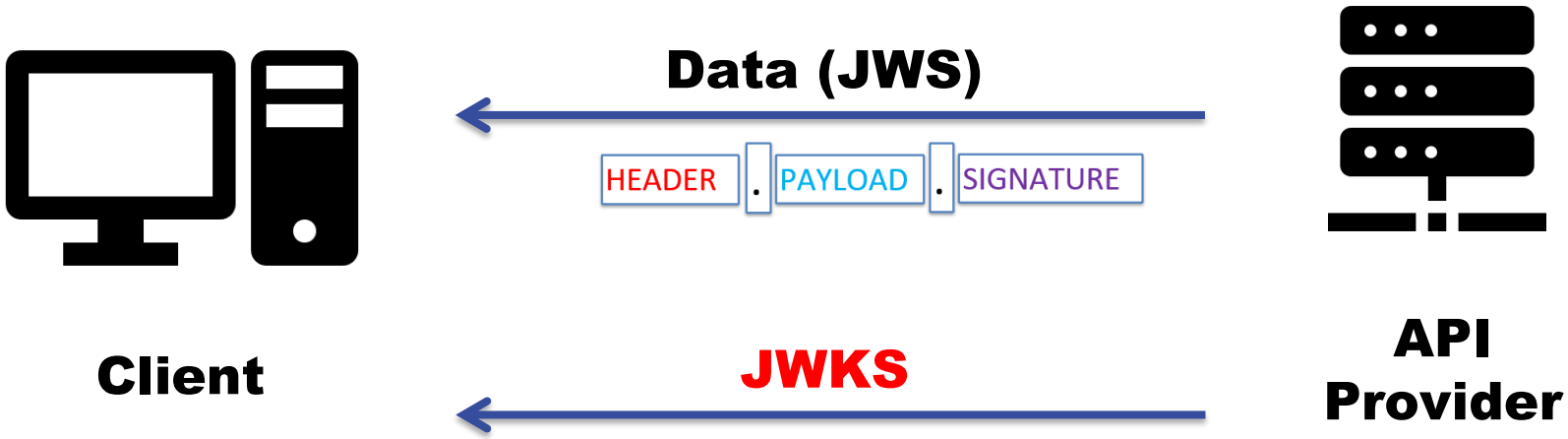

Cihan
Leuven

```
<data>
  <name>Cihan</name>
  <address>Leuven</address>
</data>
```

```
{
  “name": “Cihan”,
  “address": “Leuven“
}
```

# Secure REST

- Verification via JWKS

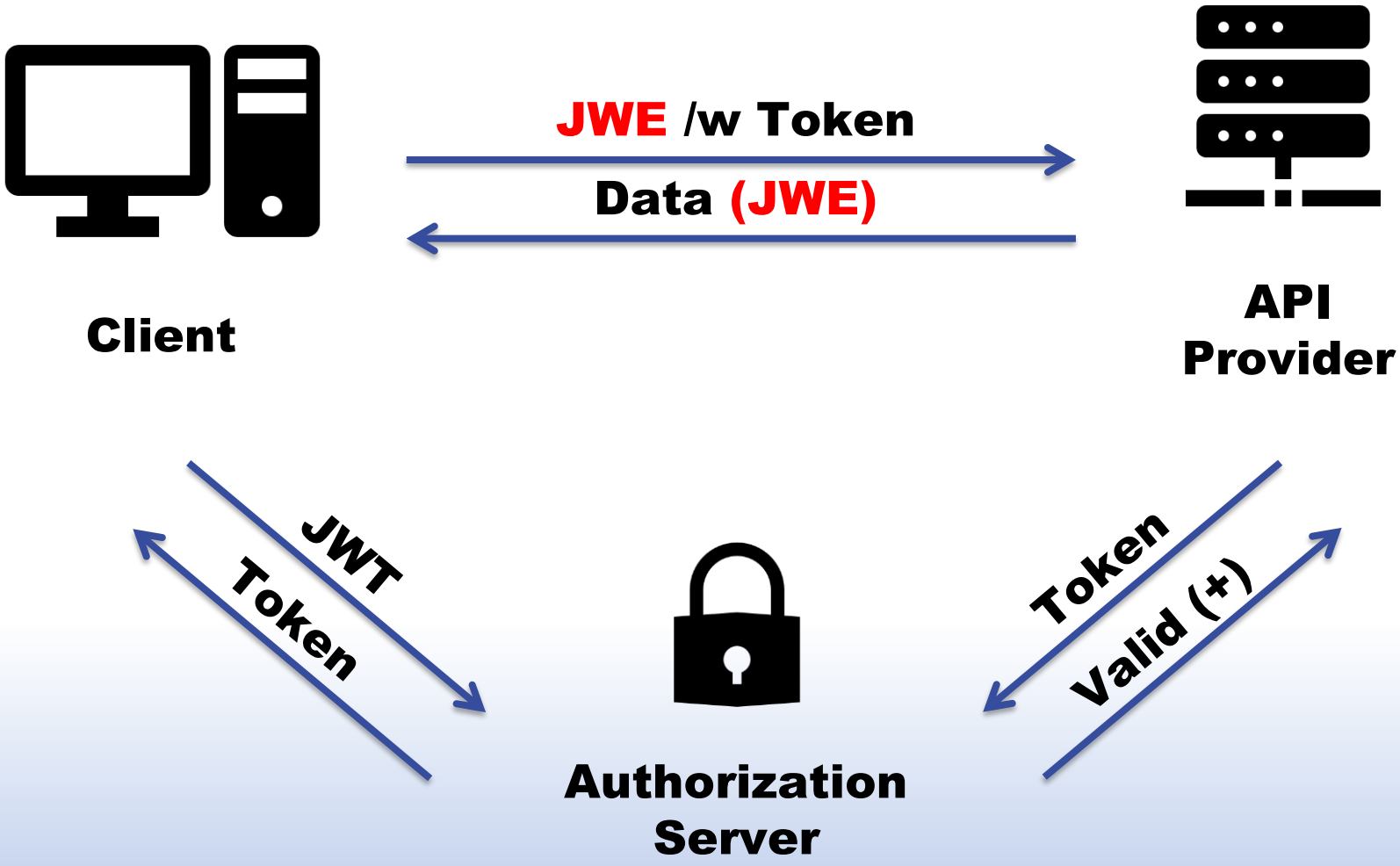



**JSON Web Key Set:** Set of public keys in JSON to ma ke them available to the external parties


```
{
    "keys": [
        {
            "kty": "RSA",
            "x5t#S256": "l0DWpFXS36hLzl8TOXjsJgyamH8jo5JglFI-pCYix2o",
            "e": "AQAB",
            "use": "sig",
            "x5t": "6a4QWdmRzM-JxdCoracInO3izEw",
            "kid": "l0DWpFXS36hLzl8TOXjsJgyamH8jo5JglFI-pCYix2o",
            "x5c": [
                "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCklJSUh6akNDQnJhZ0F3SUJBZ0lRQXg2K0dRY3VwT0dUeitUKz
            ],
            "alg": "RS256",
            "n": "ult4uVMzPBlwJ0ZuOTr_AENyMSSquN0p9xeIqleTpbw6yP88EZxEyIrQ9YNQmV8KncXqeKlJXqZvrAxWwZ-AX
        }
    ]
}
```

# Secure REST

- Policy3



# Secure REST

- JWE


- The **JWE Header** contains information about the encryption algorithms used.
- The encrypted **Content Encryption Key (CEK)** contains the key needed to decrypt the payload.
- The **Initialization Vector** is used to introduce randomness in the encryption process.
- The **Cipher text** is the encrypted payload itself.
- The **Authentication Tag** is used for integrity checks.

# JWS vs. JWE

The aims of JWS and JWE are different. A JWS is used to sign claims, a JWE is used to transmit sensitive data.

*Difference*:
JWE content is encrypted and **cannot** be seen by others.
JWS content is Base64 encoded which can be easily decoded.

*Similarity*:
The content **cannot** be modified during a transit since the signature verification would fail.

# Secure REST

- Summary

JSON Object Signing and Encryption (JOSE) Framework

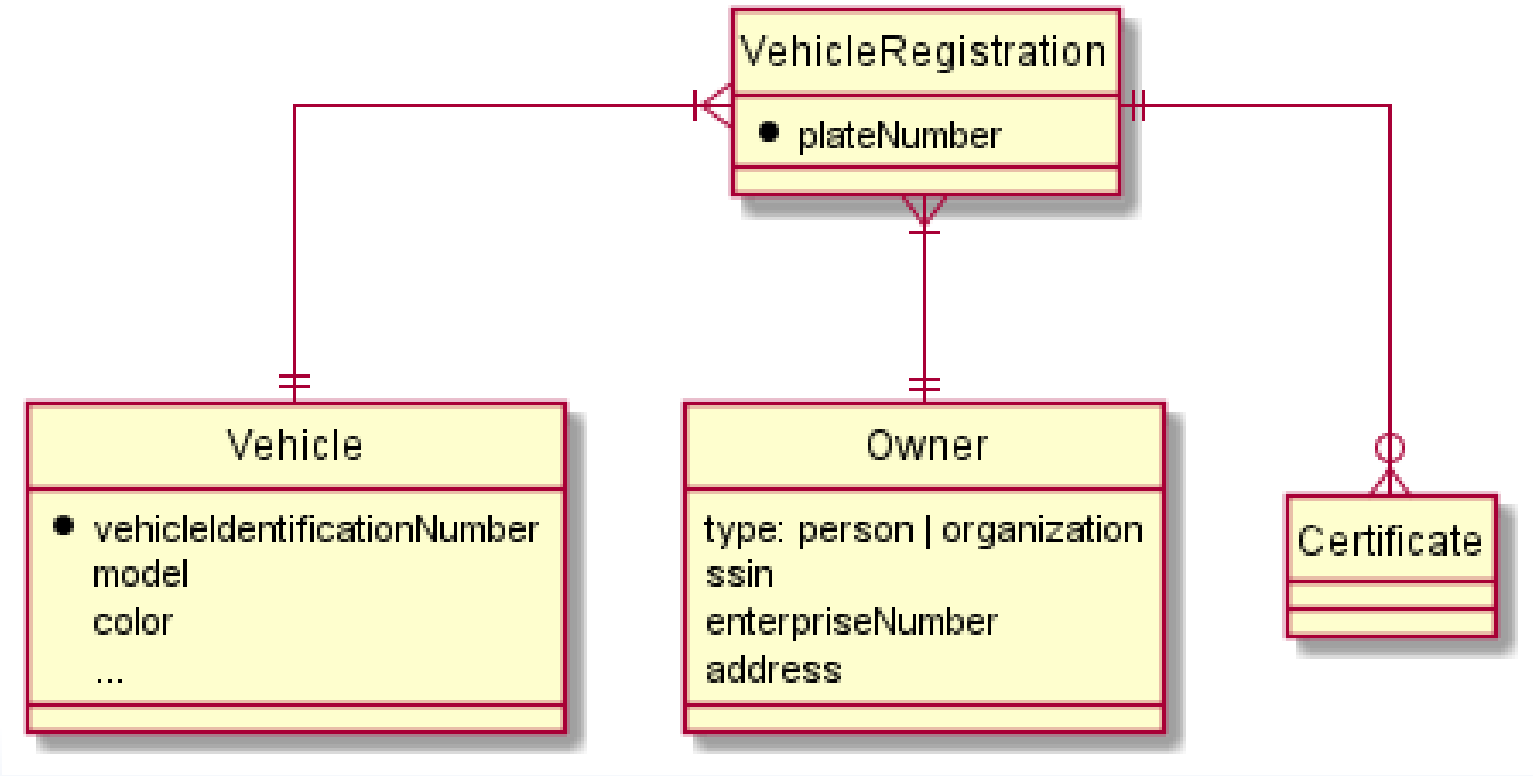
- JSON Web Token (JWT) – Exchanging claims between parties
- JSON Web Signature (JWS) – Signing the contents (+Non repudiation +Integrity)
- JSON Web Encryption (JWE) – Encryption of the contents (+Confidentiality)
- JSON Web Key (JWK) – Sharing public keys


# Practical examples

# Example: mobilityRegistrations

- Start: SOAP-like API
- GET /registrations
  Returns:
  - vehicle registrations
  - vehicle information
  - owner information
  - certificates

# Data model


VehicleRegistration
plateNumber
Vehicle
vehicleIdentificationNumber
model
color
...
Owner
type: person | organization
ssin
enterpriseNumber
address
Certificate

# Map to operations

- `GET /vehicleRegistrations?<query params>`
- `GET /vehicleRegistrations/{plateNumber}`
- `GET /vehicleRegistrations/{plateNumber}/certificates`
- `GET /certificates/{certificateId}`
- `GET /vehicles/{vehicleId}`
- ~~`GET /owners/{ownerId}`~~
  => link to enterprise (CBE) or person (CBSS) API

# Message layout

- Use standard technical types:
  - Problem types
  - Pagination
  - Health check
- Use standard data types:
  e.g. Ssin as string or as integer
  - Conversion errors when omitting leading zero
    - Number comparison: 123456789 == 0123456789
    - String comparison: “0123456789” != “123456789” ==> Error

# Other

- Add codelists under /refData/ with caching headers
- Vocabularies:
  - nationalNr -> nrn or ssin
  - companyId => enterpriseNumber
  - firstName => givenName
- BelGov-Trace-Id header

# REST API governance

- Recommended to put an organization-wide governance in place to verify compliance of REST APIs against standards
  - G-Cloud REST guide, REST functional standards, organization-specific standards, ...
  - Improves user experience
  - Encourages reuse
  - Avoid interoperability issues
- Tooling for automatic validation can assist
- Standards are not set in stone and evolve
  - REST standardization work groups can be contacted

# Call for action

- Do you want to help?
  - Give feedback on standards
  - Help shape the standards
  - Communicate internally/externally on standards
  - …

fsp@gcloud.belgium.be / Github

# Summary of links

- Federal Service Platform: NL , FR
- Functional standards: https://github.com/belgif/fedvoc (*)
- Technical standards:
  - REST styleguide https://www.gcloud.belgium.be/rest/
  - REST styleguide sources: https://github.com/belgif/rest-guide (*)
  - Datatypes https://github.com/belgif/rest-guide/blob/master/README.md#reusable-openapi-schemas
- Get in touch – leave an issue (= comment, suggestion, question) at the GitHub links (*)

- Other links
  - European standards – Core vocabularies
    - https://ec.europa.eu/isa2/solutions/core-vocabularies_en
  - Interfederal standards at ICEG
    - https://github.com/belgif/review , https://github.com/belgif/thematic
  - Regional standards https://data.vlaanderen.be/standaarden/


# Q & A