Integration Guide

Introduction

This guide provides detailed, self-contained and actionable technical instructions for integrating and federating a marketplace in DOME.
It is meant to be the reference for what is *currently* technically available and ready to be used. When additional functionality, components and/or instruction get available, this guide will be adapted accordingly.

This guide limits on the technical details and therefore is not providing any details about contractual and business aspects of the marketplace integration and federation.

Who is this guide for?

This guide aims for future participants willing to technically integrate and federate a marketplace in DOME. Descriptions and instructions are given as a comprehensive, step-by-step guide in such detail, so that IT and engineering teams can successfully and efficiently perform the necessary steps without profound knowledge of the DOME implementation details.

Nevertheless, this guide focusses on the actual integration of a marketplace in DOME, providing only the information needed to perform the actual setup for integration and federation. More details and in-depth knowledge about DOME and its involved components can be found in the linked resources.

The suggested deployment target infrastructure is Kubernetes. Therefore a sufficient knowledge of Kubernetes and Helm is expected when following this guide. Knowledge about ArgoCD is also helpful when using it as GitOps continuous delivery tool.
This guide requires access to a domain and it's DNS settings, as well as configuring proper certificates. Therefore also knowledge about cert-manager and external-dns is recommended.
Authentication requires usage of decentralized identifiers and verifiable credentials, which requires an appropriate configuration of the related components. Therefore, the reader of this guide should be familiar with DIDs and VC/VP standards.

What is DOME?

Cloud computing is identified as a central piece of Europe’s digital future, giving European businesses and public organisations the data processing technology required to support their digital transformation. The European Commission thereby stepped up its efforts to support cloud uptake in Europe as part of its strategy, notably with the pledge to facilitate "the set-up of a cloud services marketplace for EU users from the private and public sector". DOME is materialising the envisioned online marketplace, providing the means for accessing trusted services, notably cloud and edge services, building blocks deployed under the Common Services Platform and more generally any software and data processing services developed under EU programmes such as the Digital Europe Programme, Horizon 2020 or Horizon Europe.

Relying on Gaia-X concepts and open standards, DOME is providing the finishing touch to the technical building that the Digital Europe Program is creating for boosting the development and adoption of trusted Cloud and Edge services in Europe. It will provide the single point for enabling customers and service providers to meet each other in a trustful manner. DOME is taking the form of a federated collection of marketplaces connected to a shared digital catalogue of cloud and edge services. Each of the federated marketplaces will be independent or connected to the offering of a given cloud/edge infrastructure service provider which, in turn, can be classified as cloud IaaS providers or cloud platform service providers (each of which provide a platform targeted to solve the

integration of vertical data/application services from a given vertical domain, like smart cities or smart farming, or the integration of certain type of data/application services, e.g., AI services). DOME is relying on the adoption of common open standards for the description and the management of the lifecycle of cloud and edge service products and offerings around those products, as well as their access or match-making services through a shared catalogue.

What will I accomplish following this guide?

After following this guide, you will have a complete and functioning integration of a marketplace in a federated DOME environment.

The following sections provide details about the necessary steps:

- Onboarding: How to perform onboarding as DOME participant (Note, that this topic is currently under discussion and details will be added later).
- Distributed components: How to configure, deploy and operate the different required components. This involves the access node and components required for the IAM.
- Authentication: How to authenticate at services and how to implement the authentication.
- Integration API: How to integrate with the TMForum APIs, especially when using an own marketplace implementation instead of the BAE. Some samples and tutorials are given for core federation scenarios.
- Policies: How access policies are defined, created and enforced (Note, that this topic is currently under discussion and details will be added later).

Pre-requisites

There are some prerequisites that need to be fulfilled before following this guide:

- Domain: You need control over an own domain and have access to the DNS settings.
- Public access: Certain components need to be publicly accessible under your own domain. Therefore, a suitable infrastructure (like a cloud-hosted Kubernetes cluster) is required for running the components.

Additional prerequisites are provided in the different sections.

Onboarding as a DOME Service Provider with did:web

What will I accomplish following this section?

We describe here the onboarding process using Verifiable Credentials with the `did:web` DID method. In production (June 2024) the process will use `did:elsi` and eIDAS certificates, but this process will be finished in the coming weeks, so for the moment the users will be able to test onboarding and creation of Product Offerings and replication to other marketplaces, even if the actions do not have yet the legal coverage that `did:elsi` provides.

The user will be able to login with her Wallet in the DOME BAE Marketplace instance (the one operated at this moment by the DOME project) as a Service Provider with a unique identity determined by the domain (e.g., www.in2.es) used when generating the Verifiable Credential used to login to the BAE Marketplace.

Once logged-in, the user will be able to test the creation of a Product Offering using the screens of the DOME BAE Marketplace instance, and publish the Product Offering in the marketplace. The Product Offering will be replicated to all other federated marketplaces connected to the main DOME Marketplace instance.

All Product Offerings and related entities created under this identity will be separated from the other entities created by other identities.

Pre-requisites

This scenario uses the `did:web` DID Method, so you need control over a domain and have access to the DNS settings.

Before continuing, you have to create your DID, following the instructions in the section Create (Register) of the `did:web` specification.

Install and configure Keycloack as a Verifiable Credential Issuer

You need to be able to issue a Verifiable Credential to a Wallet, using the DOME format and with the `did:web` method. There are different possibilities, but the easiest one which ensures compatibility with the current status of the OID4SSI implementation in DOME is to use the Wallet and Issuer provided by DOME.

Wallet: in order to use the Wallet provided by DOME, you do not have to install anything. Just visit with your mobile the URL: https://demo-wallet.fiware.dev/

Issuer: the Issuer is a little bit more involved. Before June, DOME will provide an Issuer acting As-a-Service (for those willing to use it), but for the moment you have to install and operate an instance the Issuer yourself.

The instructions to install and configure the Issuer (which is based on Keycloack) are here: Keycloack VC-Issuer. For a simple installation, the repo includes a containerised deployment so you only have to configure the Issuer with your specific information.

The Wallet should be able to access the relevant endpoints exposed by the Issuer, as described in the instrucions mentioned above.

Configure the Credential type and the Claims in the Issuer

Follow the instructions in Configure claims for Credential-Types to configure your Issuer for issuance of the Credential required for DOME.

Issue and receive the Verifiable Credential in your Wallet

Follow the instructions in section Demo to make Keycloack issue a QR code that can be scanned by your Wallet (remember that your Wallet is at https://demo-wallet.fiware.dev/).

Once you scan the QR code and complete the issuance process, you will have in your Wallet the required credentials to login in the DOME BAE Marketplace.

Login to the DOME BAE Marketplace instance

At this moment, you have in your mobile the credentials required to login to the DOME BAE Marketplace instance with a unique identity associated to your unique domain. Even though this credential does not have the level of legal certainty required for production use, it will allow you to test the features that the DOME BAE Marketplace instance provides to Service Providers.

TODO: add instructions to login to the DOME BAR Marketplace instance.

Create a Product Offering in the DOME BAE Marketplace instance

Once logged in, you are logged as a Service Provider with a unique identity associated to your unique domain. You can start creating Product Offerings and publishing them. The action of publishing the Product Offernings will make them visible to potential customers in the DOME BAE Marketplace instance and all other federated marketplaces which are connected to the DOME main instance.

TODO: add instructions to create and publish Product Offerings.

Logoff from the DOME BAE Marketplace instance

Once you have finished interacting with the DOME BAE Marketplace instance, you can logoff from it. In case of inactivity, the BAE Marketplace will log you off automatically.

Using your Wallet, you can login at any moment and continue working with the DOME BAE Marketplace. The Wallet has your credentials stored in your device and you can use them at any moment.

Distributed components

Components that need to be operated by a federated participant.

Distributed components

Access Node

The DOME Access-Node is a set of services, that can be used to access the DOME Marketplace. A registered participant can use it to act as a federated marketplace in DOME.

The Access-Nodes consists of 3 logical building blocks:

Building Blocks Image not found or type unknown

The TM-Forum-API Service is a service providing a growing subset of the TMForum API's while using an NGSI-LD context broker as persistence backend and change notificator.

Image not found or type unknown
image.png

TODO: Add description of blockchain connector

Overview and sub-components

The TM-Forum-API service is a cluster of individual services providing one specific API each, enabling the participant to only run the necessary subset for its use-case. Apart from offering CRUD operations on the managed entities, the service also enables the subscription to notifications based on given queries.

The services are stateless and support horizontal scaling, but require an external cache to avoid having inconsistent caches. Inconsistent caches can result from either changes due to calls to the API, or due to notifications for changes reported by the underlying persistence. If run in a single instance mode, a local cache is acceptable but for larger setups a Redis installation is recommended.

For reasons of convenience, the TM-Forum-API service can be deploying with an Envoy API proxy which provides the individual APIs via a single service, routed based on the path. Another convenient feature is a RapiDoc container, that can be deployed with the TM-Forum-API service that provides a Openapi based API documentation for the deployed services, with the functionality of querying the API too.

The requirement for the persistence is to be compliant to the NGSI-LD API v1.6 enabling the use of different available context brokers. The currently recommended Context-Broker for the access node is Scorpio, mainly due to good cloud integration and overall support. The Scorpio context-broker allows a variety of adjustments to cover the operator's specific needs (e.g. horizontal scaling utilizing Kafka) and uses Postgresql as it's persistence layer. The Postgresql is extended with Postgis for supporting geospatial data.

Infrastructure requirements

The base memory consumption per deployed pod is listed below but is will increase with the amount of traffic, therefor should only be used as a rough estimate.

Service	Memory (Mi)

TM-Forum-API Pod	250
Scorpio	400
Postgresql/Postgis	150
Redis	10

Apart from the database service, no other service will maintain a own persistence, therefor only for this service a persistent volume claim has to be dimensioned.

How to deploy

The recommended and endorsed way of deployment is via the provided helm charts (optionally wrapped in ArgoCD Applications).

To deploy a setup, the umbrella chart of the access-node can be used as followed:

- create a configuration values file according to the own environment, as described here.
- add helm chart repository to helm installation

  helm repo add dome-access-node https://dome-marketplace.github.io/access-node
  helm repo update

> ? All releases of the Access-Node reside in the helm-repository https://dome-marketplace.github.io/access-node. In addition to that, all Pre-Release versions(build from the Pull Requests) are provided in the pre-repo https://dome-marketplace.github.io/access-node/pre. The pre-repo will be cleaned-up from time to time, in order to keep the index manageable.

- install the components using the prepared configuration

  helm install <RELEASE_NAME> dome-access-node/access-node --namespace <NAME_SPACE> --version <CHART_VERSION> -f values.yaml

How to configure

The chart is released with a set of default values which act as a good starting point for an adoption. These values are also documented, enhancing the understanding. Additionally, the respective charts of the components should be consulted.

Component	Chart

TM-Forum-API	https://github.com/FIWARE/helm-charts/tree/main/charts/tm-forum-api
blockchain-connector	https://github.com/DOME-Marketplace/access-node/tree/main/charts/blockchain-connector
broker-adapter	https://github.com/DOME-Marketplace/access-node/tree/main/charts/broker-adapter
dlt-adapter	https://github.com/DOME-Marketplace/access-node/tree/main/charts/dlt-adapter
kafka	https://github.com/bitnami/charts/tree/main/bitnami/kafka
postgresql	https://github.com/bitnami/charts/tree/main/bitnami/postgresql
scorpio-broker-aaio	https://github.com/FIWARE/helm-charts/tree/main/charts/scorpio-broker-aaio
scorpio-broker	https://github.com/FIWARE/helm-charts/tree/main/charts/scorpio-broker

TODO: Replace with charts in once they are used.

To have a starting point, the this minimal config reduces the configuration to items that are likely changed by integrators. TODO: include config for the blockchain components

How to validate a deployment

All components are configured with health and readiness checks to validate their own status, therefor being the base for a validation. These checks are utilized in the kubernetes checks as defined in the helm charts. TODO: Include RapiDoc Container for validation and add explanation here

How to operate

> - Management/admin APIs.
> - Instrumentation, metrics, logs, alerts

The underlying database service holds the persisted data and therefor requires a backup&recovery mechanism when operated in a production environment. The use of managed database is strongly encouraged for safety and convenience.

The TM-Forum-API service used a json based log output by default, which can be parsed easily by log aggregators but can also be replaced if needed. The verbosity is controlled via environment variables and can be fine tuned to the operators needs.

TODO: Prometheus Metrics TODO: Grafana Dashboard

How to update

Upgrade to both a different chart version and new configuration can be accomplished with the following command

```
helm upgrade <RELEASE_NAME> dome-access-node/access-node --namespace <NAME_SPACE> --version <CHART_VERSION> -f values.yaml
```

Release process

Versioning of the main access-node helm chart is handled based on the labels used in the pull requests used to introduce changes and is enforced in the build pipeline. The requester and reviewers must set the label according to the SemVer 2.0.0 versioning scheme.

Versioning of the components and sub-charts is recommended to use the same scheme.

> Versioning, release notes, stability considerations

Troubleshooting

> To be filled once feedback from integrators comes in

Timeouts occur while querying TM-Forum-API

When encountering timeouts in calls to the TM-Forum-API service it is possible to mitigate the imminent issue by increasing the timeout of the client (called "ngsi") calling the NGSI-LD broker. The necessary client and server configuration can be handed in via additional environment variables.

IAM components

The DOME IAM-Framework is a set of microservices, that enables users in the DOME ecosystem to authenticate into the DOME Marketplace. The authentication process itself is described further below in the Authentication section.

Overview and subcomponents

The DOME IAM-Framework consists of multiple open-source components. The components are not required to be used, as long as alternatives providing the same interfaces are used.

The IAM-Framework consists of following components:

Image not found or type unknown
IAM-components

- The Trusted Issuers List service provides an EBSI Trusted Issuers Registry implementation to act as the Trusted-List-Service in the DSBA Trust and IAM Framework. In addition, a Trusted-Issuers-List API is provided to manage the issuers.
- VCVerifier provides the necessary endpoints to offer SIOP-2/OIDC4VP compliant authentication flows. It exchanges VerifiableCredentials for JWT, that can be used for authorization and authentication in down-stream components.
- Credentials Config Service manages and provides information about services and the credentials they are using. It returns the scope to be requested from the wallet per service. Furthermore, it specifies the credentials required and the issuers list endpoints to validate against, when checking access for a certain service.
- The Keycloak-VC-Issuer is plugin for Keycloak to support SIOP-2/OIDC4VP clients and issue VerifiableCredentials through the OIDC4VCI-Protocol to compliant wallets.
- PDP is an implementation of a Policy-Decision Point, evaluating Json-Web-Tokens containing VerifiableCredentials in a DSBA-compliant way. It also supports the evaluation in the context of i4Trust.
- Keyrock is the FIWARE component responsible for Identity Management. Within DOME IAM-Framework, currently Keyrock is being used as the iSHARE-compliant Authorization Registry (see for details: https://dev.ishare.eu/delegation/endpoint.html), where attribute-based access policies are stored and used during the authorization process. Note, that this will be replaced by an ODRL-compliant policy registry. A description of the policies is given in the policies section.
- Kong Plugins allow to extend the API Gateway Kong by further functionalities. Kong Gateway is a lightweight, fast, and flexible cloud-native API gateway. One of the plugins is the PEP plugin, which is especially required within the IAM-components as PEP component and interacts with the PDP mentioned above.
- Waltid manages Keys, DIDs and VCs. It is used by VC Issuer and VCVerifier.

How to deploy

The recommended way of deployment is via the provided Helm charts.

To deploy a setup, the umbrella chart of the iam-components can be used as followed:

- create a configuration values file according to the own environment, as described here.
- add helm chart repository to helm installation

  helm repo add dome-iam https://dome-marketplace.github.io/iam-components
  helm repo update

> ? All releases of the IAM-components reside in the helm-repository https://dome-marketplace.github.io/iam-components. In addition to that, all Pre-Release versions(build from the Pull Requests) are provided in the pre-repo https://dome-marketplace.github.io/iam-components/pre. The pre-repo will be cleaned-up from time to time, in order to keep the index manageable.

- install the components using the prepared configuration

  helm install <RELEASE_NAME> dome-iam/iam-components --namespace <NAME_SPACE> --version <CHART_VERSION> -f values.yaml

How to configure

The chart is released with a set of documented default values. The parameters listed below are important to set and should be updated at least:

- `rbac` and `serviceAccount`: Depending on your requirements, you might need to adapt settings for RBAC and service account
- `did`s of participants: Replace/add the DIDs of the issuer and other participants
- In the case of did:key provide correct key in keyfile.json for your issuer
- `keycloak.frontendUrl`: Externally accessible address of the keycloak (should be the same as defined in ingress/route)
- `keycloak.realm`: Adapt clients, users and roles according to your needs
- `<tir.com>`: replace everywhere with actual TIR URL
- `<dome-marketplace.org>`: replace with your own domain
- `keyrock.initData.scriptData`: Adapt the roles as in keycloak realm
- `kong.configMap`: Adapt the kong services and their routes

However, it is suggested to consult the respective charts listed below and check their documentation and configuration.

Component	Chart
postgresql	https://github.com/bitnami/charts/tree/main/bitnami/postgresql
mysql	https://github.com/bitnami/charts/tree/main/bitnami/mysql
vcwaltid	https://github.com/i4Trust/helm-charts/tree/main/charts/vcwaltid
keycloak	https://github.com/bitnami/charts/tree/main/bitnami/keycloak

Component	Chart
credentials-config-service	https://github.com/FIWARE/helm-charts/tree/main/charts/credentials-config-service
trusted-issuers-list	https://github.com/FIWARE/helm-charts/tree/main/charts/trusted-issuers-list
vcverifier	https://github.com/i4Trust/helm-charts/tree/main/charts/vcverifier
keyrock	https://github.com/FIWARE/helm-charts/tree/main/charts/keyrock
dsba-pdp	https://github.com/FIWARE/helm-charts/tree/main/charts/dsba-pdp
kong	https://github.com/Kong/charts/tree/main/charts/kong

How to validate a deployment

All components are configured with health and readiness checks to validate their own status, therefore being the base for a validation. These checks are utilized in the Kubernetes checks as defined in the helm charts.

How to operate

The underlying database service holds the persisted data and therefore requires a backup&recovery mechanism when operated in a production environment. The use of managed database is strongly encouraged for safety and convenience.

How to update

Upgrade to both a different chart version and new configuration can be accomplished with the following command

helm upgrade <RELEASE_NAME> dome-iam/iam-components --namespace <NAME_SPACE> --version <CHART_VERSION> -f values.yaml

Release process

Versioning of the main iam-components helm chart is handled based on the labels used in the pull requests used to introduce changes and is enforced in the build pipeline. The requester and reviewers must set the label according to the SemVer 2.0.0 versioning scheme.

Versioning of the components and sub-charts is recommended to use the same scheme.

Troubleshooting

> To be filled once feedback from integrators comes in

Authentication

Link to IAM Guide

Follow this link to go to the specific IAM Guide
https://dome-marketplace.github.io/iam-guide/

DOME Verifiable Credentials (LEAR)

How to implement

> using wallets, OIDC4VC&VP. verification etc.

Integration API (TMForum)

> Samples & tutorials for implementing core federation scenarios using the TMForum APIs (with focus on semantics and workflow, beyond the API reference)
>
> Some content also required for the Knowledgebase, which should be added here as well:
>
> - How to: autenticate the marketplace on the shared data layer
> - How to: retrieve DOME ecosystem notifications
> - How to: retrieve a list of products from the shared catalog
> - How to: retrieve the product description from the shared catalog
> - How to: retrieve the product price model from the shared catalog
> - How to: push a product on the shared catalog
> - How to: retrieve an order from other marketplaces
> - How to: push a provisioning status on the shared data layer
> - How to: push metering information on the shared data layer
> - How to: push billing information on the shared data layer
> - How to: retrieve billing information on the shared data layer

Policies

Defining policies

> - How can a Service Provider create policies that concern the Products that they offer ?
> - How can a Marketplace Operator create policites that concern the Products that they host ?

Local policies

> That only the federated marketplace needs to enforce locally.

Distributed policies

> That everyone in the federation needs to enforce.

Enforcing policies

In front of TMForum API

> i.e. marketplace -to- Access Node PEP, PDP - when are they needed and when not ?

In front of Context Broker API

> i.e. Access Node -to- Access Node