Ecole polytechnique fédérale de Lausanne

Microservice Architecture for Urban Digital Twin platform and its security implications

Prof. Edouard Bugnion, Thesis Supervisor
Florent Martin, External Expert

Master Thesis

Elvric Trombert

March 6, 2023

problem: lack of common requirements,
architecture framework and security best practices

Abstract

Smart city initiatives are on the rise and with them interests in technologies aiming to improve urban planning, optimize asset management and cohesion between city actors. This has led to a growing interest in the creation of an Urban digital twin of cities. Although multiple implementations already exists, there is a lack of common requirements, architecture framework and security best practice when it comes to their implementation. To address them, this paper derived urban digital twin requirements from existing literature reviews. From these requirements a micro service architecture is proposed. Based on this architecture a threat model analysis using the STRIDE was conducted and high risk threats regarding authentication and authorization were identified. Mitigation points addressing these two threats are proposed focusing on certificate based authentication and user-managed access control standards (UMA). Finally a solution architecture is presented integrating the identified mitigation points.

goal: propose micro service architecture
(that aims at solving previously
mentioned problems)

Contents

1 Introduction

Interest in Smart Cities (SC) has been growing for the passed decade, with a steady increase in publication numbers [67]. With the advancement of Information and Communication Technology cities capabilities to gather data from different sectors has never been as high [50, 42]. Yet SC is still in its infancy with a lack of a shared definition. As a result the objectives and approaches to make use of that data have yet to be defined. However, the principle of digitisation: pushing for a transformation of physical assets into a set of interconnected digital system has gained traction in the context of SC [57]. One of the driver of this digitisation effort is the concept of Digital Twin (DT) driven by the manufacturing sector, that takes the shape of the Urban Digital Twin (UDT) in the context of the smart city.

DT first appeared in the 2000s as a mean to control product life cycle management. A DT can be defined as "a virtual representation of a physical system (and its associated environment and processes) that is updated through the exchange of information between the physical and virtual systems" [63]. A similar vision is shared in [39] which proposes three distinct level of integration representing the evolution from models to DTs, presented in Figure 1. The digital model is the representation of an existing or planned physical object that does not include any automated data flow. The digital shadow includes an automated data flow from the physical object to its digital object but not the other way around. Finally the digital twin includes an automated data flow in both directions. Thus what truly differentiate a DT is its bi-directional connection with its Physical Twin (PT).

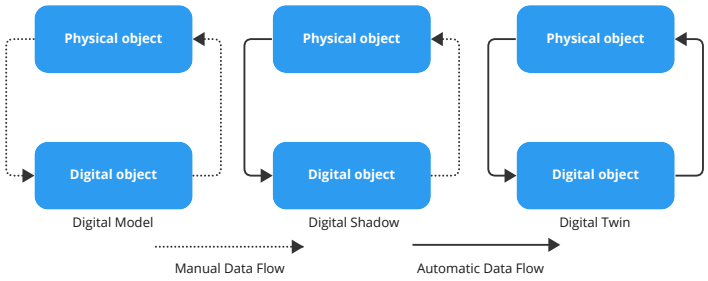

Figure 1: Digital Model, Shadow and Twin communication flows

DT itself can be broken down in 5 components presented in Figure 2.

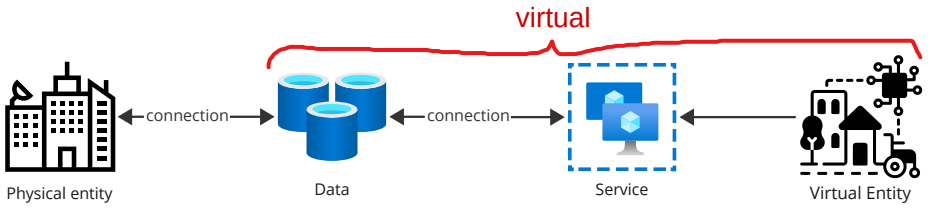

Figure 2: 5 components of DT derived from [61]

Data component, represents the attribute and contextual information of the physical entity. Service component, represents the services derived from the data provided by the physical entity, together representing the virtual entity. Connection component, represents the bi-directional connection that exists between the physical and virtual entity.

In order to constitute an accurate representation of its PT, the DT needs access to real-time reliable data, that it then uses to derive the state of its PT. Such data is provided via Information and Communication Technologies and IoTs which are the main drivers in the production of real-time data [21]. Thus DT emerges from the combination of hardware (IoT) providing real-time data streams, relayed via technologies such as 5G or Lora [54, 29] to a permanent storage system where the data can be treated [21]. That data is then used to perform simulations, visualization or machine learning computations [24]. These empower decision making processes that can impact the PT in real time. Despite its potential to lead the digital transformations of many different sectors, one of the main obstacle to the adoption of DT is the lack of reference frameworks. Th DT landscape today is heterogeneous with a lack standardized architecture guidelines [35].

Although there has been attempts to address this factor in the context of DT used in the manufacturing sector [59]. The relatively recent interest for DT in the urban context has led to the creation of dissimilar UDT implementations as UDT characteristics is heavily influenced by their purpose of use [24]. Also unlike other sectors UDT involves different city domains represented by multiple actors [42, 24]. As these actors have themselves independently went through the process of digitalization, UDT can take advantage of the data already gathered from them [57, 42]. The ability for that data to be shared between actors, and be used by the UDT services is at the center of its framework and represents one of its biggest challenge. Each actor comes with its own data management processes, internal standards and unique data. From this heterogeneous base the need for standardization framework specifically adapted for UDT is a necessary step to enable the integration of data gathered by the different actors into a common platform.

As DT and by extension UDT, have access to real time information used to represent the state of their PT. Cyber attacks targeting the DT, can lead to significant information disclosure over the state and type of physical infrastructure represented by the DT [8]. Given the presence of the bi-directional data flow from DT to its PT. These cyber attacks can also have a direct impact on the physical system increasing its attack surface [31]. With the evolving nature of DT frameworks, the security standards and risks assessment have not sufficiently been considered. The security aspect of DT is fundamental as DT can become part of critical systems [8].

In UDT, dealing with multiple actors increases the attack surface and security risks further, which in the context of a city, can have devastating effects on services and infrastructure relying on the UDT.

The main objective of this paper was to propose a UDT architecture and identify its security risks. goal To do so UDT requirements were derived from existing literature reviews, based on these requirements a micro service architecture was proposed, a threat model analysis was then conducted on the architecture. Using this analysis high risks threats were identified and mitigation methods were propose to address them. These included authentication method via client certificates and access control method via the use of User manage access control (UMA) standard. Finally a proof of concept (POC) proposal is presented making use the identified mitigation methods.

2 The 6 layers of the Urban Digital Twin

In order to define the scope of the Urban Digtal Twin we will break down DT general architecture into layers. A divide by layers is proposed in [8] and used as a level of abstraction to present operational requirements for DT. Another divide by layers model following the RAMI 4.0 IT layers standard [51] to divide DT requirements is presented in [59]. Base on these two approaches the following 6 UDT layers presented in Figure 3 were identified:

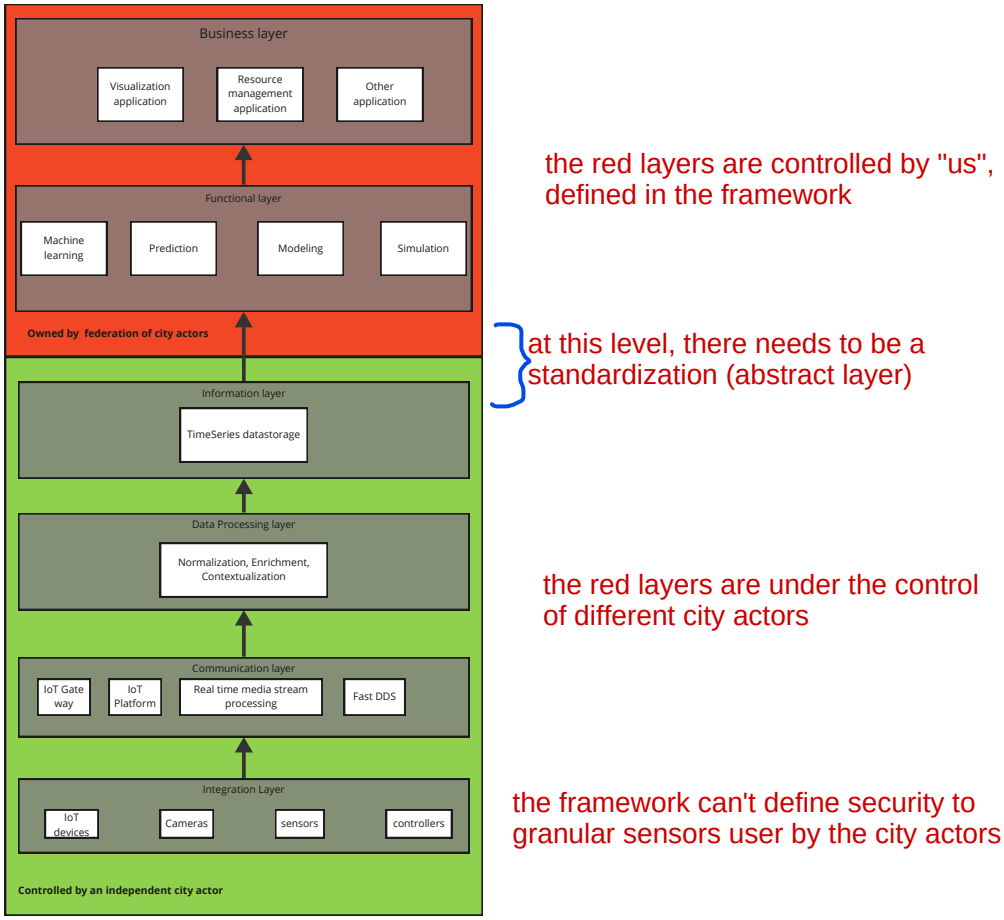

Figure 3: 6 Layer based Urban Digital Twin

2.1 Defining the layers in scope

As presented in Figure 3 some of the layers forming the UDT are not directly under the control of the organization federating the city actors. The layers controlled by the city actors have already evolved independently of each other forming a heterogeneous system. In order to establish the standards and architecture that connects the heterogeneous city actors data pipelines, this paper focuses on the identification of requirement covering the business, functional and informational layers, used to then propose an architecture framework aiming to make each city actor independent data pipelines interoperable.

2.2 Integration layer: data creation

This layer contains the technology use to obtain information about the physical environment a physical twin is subjected to. In most cases this technology is provided by the use of IoT devices such as sensors [24]. This layer forms the base of the UDT, and is controlled and owned by private or public city actors. These devices rely heavily on proprietary components with vendor which has lead to security vulnerabilities and lack of common standards across IoT devices [36].

2.3 Communication Layer: data capture and acquisition

This layer captures data generated by sensors and other physical devices. Some data pre processing is performed so it can be transferred to the information layer. To transfer data from the integration layer to this layer multiple communication protocols can be used such as MQTT, CoAP AMQP [22]. Review over the performance and security implications of these protocols can be found in [53]. In order to support the transfer of information from the local area network to the internet protocols such as LoRaWAN [29] and 5G are used [54].

2.4 Data Processing Layer

This layer extract, normalize and load the data into the information layer. For large quantity of time series data , the two following aspects must be considered.

1. The need for contextual metadata: the context in which the data was obtained such as the time, sensor location and overall environment can influence the nature of the data collected. This contextual metadata can directly come from the IoT device itself, or be inferred from multi sensor data fusion techniques [26]. In DT this is an important step, as the DT state should represent the state of its PT, thus the accuracy and context of the data gathered to derive the virtual state cannot be overlooked.

2. The need for data processing: DT requires the use of accurate and reliable data, raw data comes with inaccuracies and its own formatting that must be transposed to a format that can be stored and used by the layers above.

These two aspects put a heavy strains on the computing capabilities of the overall platform. The computing architecture may be split into three levels, edge, fog and cloud computing [64].

- Edge computing: a set isolated computing platforms close to the network edge, e.g a computing device within the same network as a set of sensors that perform a set of computation on sensor data.

performs computations on raw data from sensors

- Fog computing: a set of distributed but interconnected intermediate computing nodes that act as the intermediate layer between edge devices and the cloud. This level has increase processing power and storage capacity that can enable data aggregation from edge devices. e.g an edge server that receives data inputs from multiple sensors on different networks located in close geographical locations, performing a set of computation and storing the data before transmitting it to a data store on the cloud in a distant geographical location.

between edge devices, and the cloud, receives data from multiple sensors on different networks in close geographical location

- Cloud: centralize high capacity computing, networking and storage.

centralized and high capacity computing , networking and storage

This split is usually required when dealing with large amount of time series information from IoT devices to ensure rapid data processing along the data pipeline.

2.5 Information Layer: data management and synchronization

Stores the data gathered from lower layers to be consumed by the functional layer.

2.6 Functional layer: data modeling and additional services

Contains the set of processes that simulate, model and analyse the state of the UDT using the processed data.

2.7 Business Layer: data visualization and accessibility

Contains application logic that using the different services of the functional layers allow for end user interaction with the UDT.

3 Existing framework and standards in UDT

3.1 Building Information Modeling (BIM) and Geographic Information System (GIS)

3.1.1 BIM framework

The BIM framework was first proposed in [62], as a mean to improve collaboration between the different actors involved in the life cycle of a building e.g architect, engineers and construction workers. BIM uses an object oriented approach with semantic data models to construct various virtual models of a building divided by domains e.g (architecture, technical equipment) [30].

These models can be shared between domains via specified encoding such as Information Foundation class (IFC) which can be used to transfer building components, structure and attributive properties between domains, providing a data schema and file format structure [25] addressing the data heterogeneity problem that exists between the different silos. Further enhancement such as IFCowl are being proposed making use of Semantic Web technology such as RDF [3, 12] providing better interoperability.

Research pushing for further integration between domains and moving from BIM to a building DT are still at a nascent stage. In [60] it is argued that BIM integration with IoT data will allow it to gain access to time series data streams enabling the development of dynamic models. Different approaches to connect the BIM model with heterogeneous sensor data are proposed. The semantic approach presented as "one of the most promising methods to facilitate IoT integration" suggests the use of Semantic Web technology to address the need of storing and sharing heterogeneous datasets. Concrete suggestions include the use of RDF format to link data silos across different domains with SPARQL as the query language. Thus their is a movement suggesting the use Web semantic like solution to enable data exchanges between BIM domains and data exchanges from sensors to BIM domains.

Requirements and current limitations to move from BIM to a Building Digital Twin are described in [12]. The author argues that the use of Semantic Web in BIM is a necessary step for it to integrate with new domains included by not limited to IoT, web resources, Geographic Information Software (GIS), moving from a static model to a dynamic model. The author also presents the 3-Tier evolution of digital twin, arguing that implementation of DT first goes through the creation of a monitoring platform with limited analysis capabilities. Then to an intelligent semantic platform creating a knowledge base from which simulation and prediction can run. To finally, an agent driven socio-technical platform where the DT is able to adapt in real time and respond in real time, engaging with end users and adapt to social requirements.

3.1.2 BIM and GIS integration

GIS is a framework use to capture, store and use geographical spacial data [14]. CityGML [16] is on of the standards used to encode the 3D model of a city using GIS. BIM and GIS integration can be used to generate a 3D model of cities integrating geographical and building data together. However BIM already relies on geographical data especially in the early stage of the construction model, using it to plan and design the building based on the terrain specific characteristics.

Multiple approaches have been propose to integrate these two models. In [30], it is argued that the greatest limiting factor is the lack of interoperability between the two models and the loss of information when converting from one to the other. Multiple solutions are proposed including linked models making use of Semantic Web RDF technology to define the data schema and context to solve the interoperability problem. The same approach is also proposed in [33]. An architecture framework is proposed and evaluated in [32]. A translation workflow from CityGML to RDF and from IFC to RDF is define. The translated data is then put onto a Neo4j RDF store and can be queried using SPARQL, a language designed to query RDF graphs. The evaluation then showed a good data retrieval performance.

4 Findings and Requirements

Building on the approach used in [59] to extract requirements for DT. This section presents literature reviews used to derive requirements for the UDT. These requirements are presented in Table 1, 2, 3, 4.

A basic structure of a UDT is presented in [24]. Existing urban model implementation were reviewed, from there resulting requirements were identified and later used to propose a UDT model. Identified aspects of what makes a UDT include its multi actor component (RN-5,RN-10,RN-8) which in turns derive the need to assign responsibilities regarding the maintenance of the UDT services, data flow and access control (RI-5,RF-5,RI-8,RI-14). To do so a governance model must be provided to coordinate the different actors with a defined business model (RN-4,RN-9) [50, 43] Another component is the presence of a wide range of different domains that make up the UDT as compare to the DT, with their integration being one of the major challenge. The UDT platform must have the capacity to be scalable, modular, and process heterogeneous data sources.

In [43, 57], cities needs are analyzed, to define what aspects of the DT could be used to cover those needs. BIM and GIS framework forming the corner stone of the UDT (RI-9,RI-10). The authors also highlight the dynamic nature of a city. Which should be reflected in its DT if it is to be used for real time decision making and assist in planification decisions. The importance of real time data shared over

the entire city implying the importance of its quality (RI-11), its availability (RN-6), its security (RN-7) and its privacy (RI-8).

The cross domain nature of UDT comes with the concept of Big Data management. As multiple actors and domains share their data, managing and treating it becomes challenging [7]. The bidirectional connection present in a DT in addition to the changing nature of a city pushes for their adoption of IoT technology such as sensors. These sensors provide real time data on the evolving city environment passed on to its UDT. In order to do so a seamless connection among the different city components must be provided with up to date data (RI-16). The contextualization of the provided data is also at the core of UDT. As the UDT models a virtual image of the city, understanding where the data comes from, how it was collected and how trustworthy form an integral part in the UDT model (RI-14,RI-11,27) [9]. Given the scale of a city, the involvement of private actors, having already deployed this technology in their domain lowers the barrier of entry to create the UDT. To foster their involvement, control over who accesses and uses their data must not be lost (RI-5,RI-12,RI-13). Trust in UDT security must be fostered (RN-7,RI-8), with clear guideline on who can access it (RI-7).

From this data, models and services that combine create the UDT are derived [42]. In a similar fashion to data interoperability, service interoperability must be considered in order for the overall UDT to function (RF-4,RF-5,RF-6).

4.1 Identified Requirements

The requirements are sorted by layers presented in section 2 with the additional Non-Functional layer covering the layers in scope as defined in subsection 2.1.

4.1.1 Non-functional requirements

Table 1 shows the identified non functional requirements that do not fall into the 6 UDT layers following the approach proposed in [59]. In contrast with DT, in UDT multiple actors are involved their is a need for an institutional framework through which the UDT will be built (RN-4, RN-7). In a similar fashion, this also drives the need for a multi-party access control framework (RN-5, RN-10). Finally in order to foster the engagement of actors on the UDT a common guideline established within the institutional framework must be defined (RN-8, RN-9).

ID	Requirements	Origin	Covered [1]
RN-1	The DT and its services should be able to be hosted on the cloud as well as on-premises for data ownership and performance reasons.	[7, 12, 9, 21, 59]	Yes
RN-2	Services of a DT should be loosely coupled to add or remove new services without influencing each other.	[42, 57, 24, 21, 59]	Yes
RN-3	Services of a DT should be maintainable by different development teams (third party integration).	[42, 21, 24, 59]	Yes
RN-4	UDT should follow a federated governance model at city level	[7, 43, 50, 42, 12]	No
RN-5	Actors using the UDT platform should be uniquely identifiable	[24, 7]	Yes
RN-6	UDT should be accessible by private and public entities	[42, 43, 57]	Yes
RN-7	UDT platform should define and enforce security standards	[43, 21]	No
RN-8	UDT platform should support the city value proposition	[21, 12, 42, 24]	No
RN-9	UDT platform should have a defined business model that fosters actors' involvement	[7, 43]	No
RN-10	UDT platform should have the ability to manage multi party access	[43, 42, 24]	Yes
RN-11	UDT should have the ability to actuate on its physical twin	[24, 21]	No

Table 1: Non Functional Requirements

4.1.2 Information layer Requirements

Table 2 shows the identified requirements present in the information layer where the information exchanged between services is present. Compare to DT with the involvement of multi actors data ownership

[1] covered by the derived architecture presented in Figure 4

and trust principles need to be established (RI-5, RI-6, RI-7, R-11, RI-14, R-13). In a similar fashion private actors need to maintain control over who gets to use and access the data they own (RI-12). As data is moving from private sector onto a shared platform regulation regarding its privacy must be taken into account (RI-8). Unlike the DT in the industrial sector already established standards such as BIM and GIS exists and are widely used. The ability to integrate information from these standards is needed to create a UDT (RI-9, RI-10).

ID	Requirements	Origin	Covered
RI-1	The DT should be able to process heterogeneous data from different sources.	[24, 12, 66, 21, 9, 59]	Yes
RI-2	The DT should be able to interlink time series data with context information, to make it interpretable for other services.	[7, 21, 59]	Yes
RI-3	The DT should have a service which provides access to the information provided by all services of the DT.	[57, 42, 59]	Yes
RI-4	Services of the DT should exchange information in a semantically meaningful way.	[24, 12, 7, 59]	Yes
RI-5	All data shared on the platform should be associated with a data owner.	[7, 24]	Yes
RI-6	UDT data objects need to be uniquely identifiable.	[43, 21]	Yes
RI-7	UDT platform should define and enforce which actors can share data on the platform	[12, 7, 50]	Yes
RI-8	UDT platform should comply with privacy regulation regarding data sharing	[43]	No
RI-9	UDT platform should have the capability of integrating BIM data.	[43, 12, 21]	Yes
RI-10	UDT platform should have the capability of integrating GIS data.	[21, 57, 43]	Yes
RI-11	UDT platform should define and enforce data quality standards.	[9, 43]	No
RI-12	Actors involved in the UDT platform should have control over data sharing policies on their data.	[7]	Yes
RI-13	UDT platform should enforce end to end data traceability.	[7]	No
RI-14	UDT platform should enforce the principle of non-repudiation on the data shared.	[7, 12, 50]	No
RI-15	UDT should have access to historical data.	[21, 43]	Yes
RI-16	UDT should have access to time series data	[24, 43, 21]	Yes

Table 2: Information layer requirements

4.1.3 Functional layer

Table 3 shows the requirements present on the functional layer. The holds the functions and services. Again compare to DT, in UDT multiple actors will be involved, services ownership and standards need to be established to foster interoperability (RF-5, RF-6).

ID	Requirements	Origin	Covered
RF-1	Services shall be able to access a continuous stream of data in real-time.	[24, 7, 21, 57, 43, 9, 59]	Yes
RF-2	Data streams should be accessible by multiple services simultaneously.	[57, 7, 59]	Yes
RF-3	Services of the DT should be able to receive data streams from multiple sources at the same time.	[57, 7, 59]	Yes
RF-4	Actors involved in the UDT platform should have control over access policies to the services they own.	[24, 7]	No
RF-5	UDT services need to be uniquely identifiable.	[24, 7]	No
RF-6	UDT platform should enforce services standards to foster horizontal and vertical interoperability.	[42, 57, 7]	No
RF-7	UDT service should have the computing power necessary to process data in real time	[7, 9]	No

Table 3: Functional layer requirements

4.1.4 Business layer

Table 4 shows the requirements present on the Business layer. There are no major difference at this level between DT and UDT as in both cases services present at the Functional layer should have the ability to be integrated into Business processes

ID	Requirements	Origin	Covered
RB-1	The functional services of the DT should be able to be integrated into the business processes at enterprise level to support the value-added chain.	[59]	No

Table 4: Business layer requirements

5 Proposed framework

As seen from the adapted requirements one of the biggest variation from DT to UDT is its multi actor aspects from different domains which raises the concern of data ownership and governance framework. Prior to building an architecture framework covering the set of technical requirements, a common high level framework should be provided to reason about the role and responsibilities of the city actors and the federation creating the UDT.

5.1 Data mesh paradigm

Data mesh is a concept created by Dehghani [18] in 2018, as a response to the misalignment between the organizational needs and modern data management architectures. In a similar fashion to what micro service architecture, moving from monolithic application to distributed application. Data mesh proposes a decentralize data architecture permitting the extraction of large scale analytical data [45].

Data mesh is broken down in into 4 pillars: domain ownership, data as a product, self-serve data platform and federated computer governance. The concepts and relation to the UDT requirements are presented bellow.

5.1.1 Domain ownership

The postulate taken here is that data is that teams that work closely with the data can better understand, how to make it usable for their specific domain. A domain can be represented as an area of operation in a company e.g Sales, human resource etc... Domain ownership establishes the notion that domains are responsible to store and serve the data they create. [19]

In the context of UDT a domain can be represented by the city actors providing a set of the UDT data. Each city actor being responsible to store and serve the data they provide to the UDT. This pushes the responsibility of data quality and ownership on the participating actors. As from a governance stand

by looking at figure 3 --> the red layers, are controlled by private actors, and this governance cant be controlled by the "governance"

point the shared UDT architecture cannot directly influence the internal architecture of its partners. By considering each city actor as a domain, the responsibility of data quality (RI-11) and provision is pushed onto the individual actors. The responsibility integrates with the notion of association of data shared on the platform with an owner (RI-5), with the right of control over the data shared a given actor should have over the data it owns (RI-12), and the decoupling of services running over the data (RI-2, RI-3).

5.1.2 Data as a product

This pillar aims to address the issues linked to data accessibility and quality. As the data is now managed and stored in a distributed fashion, it is important for that data to remain usable by other domains, by following a product thinking approach to the way data is provided to its consumers. Data must be:

- Discoverable: provided by registry of available data products with their meta information such as their owner and location. This service accessible to data consumers in our case UDT services allow for data of interest to be found.
- Addressable: in order to be found and queried a data product should have a unique standardized address for data users to access it. (RI-3)
- Trustful: owner of a data product are responsible for ensuring quality objective mandated by the platform. For metadata regarding data provenance, time series and context information (RI-13) should be provided (RI-3).
- Defined semantics: consumers should have the ability to discover, understand and consumer data products with minimum friction. Set data schema and standards simplifies data consumers use and access to the data (RI-4, RI-1)
- Interoperable: data consumers should have the ability to aggregate and join the data from different domains. Standards and harmonization rules defined globally are the key enabler for distributed system interoparability. (RI-1)
- Secure: access control to which data consumer can access which data can be finer grained being applied at the domain level. (RI-14, RI-12)

some of the requirements mentioned in stage 1

5.1.3 Self serve data platform

Represents the platform that must be made available for data product owners to manage and serve their data enabling domain autonomy. It has the following capabilities:

- scalable: should scale to handle the addition of multiple data products and large flow of data across domains. (RN-2)
- Encryption for data in motion and at rest (RN-7)
- Data product versioning: in UDT refers to the evolution of the data overtime which must be available on the platform (RI-16)

access to time series data

- Data product schema: enforces the type of schema used to represent data products
- Unified access control: allow domain owners to define on the platform who can access their data products (RI-12, RI-7)
- Data product lineage: enable users to trace the origin of a data product to its producer (RI-13, RI-14)
- Federated Identity management: manage identities of participating entities (RN-4, RN-10, RN-5)

In the UDT context this platform takes the shape of the micor service architecture, that enables the integration of data owned by independent city actors into a federated platform from which UDT services can get access to data products.

independent city actors create data, store it in a federated platform --> but are able to decide which other city actors have access to given data. This accesses can be revoked/modified at any given time

(1) we want these rules to be easy to create
(2) and to be very granular !

---> stage 1 tackles this problem

5.1.4 Federated computing governance

Data mesh follows a distributed approach to data management and ownership. That degree of local independence with independent teams managing the internal data pipline of their domain (RN-3) requires regulation to foster the interoperability of the data products created. This is role of the federated governance, that jointly defines the standards and rules that all data products have to abide by.

Due to the plurality of actors in UDT in contrast to DT. The need for federated governance is present in the requirements, enabling the definition of global standard (RN-4, RN-9) and global security policies (RN-7, RI-11, RF-6).

5.2 Micro service architecture

Following the data mesh approach, the architecture propose aims at covering the need for a self serve data platform in the context of the UDT. Participating actors acting as domain owner within their respective silos. The Figure 4 represent a propose architecture framework in the form of a data flow graph following the schema standards proposed in [15].

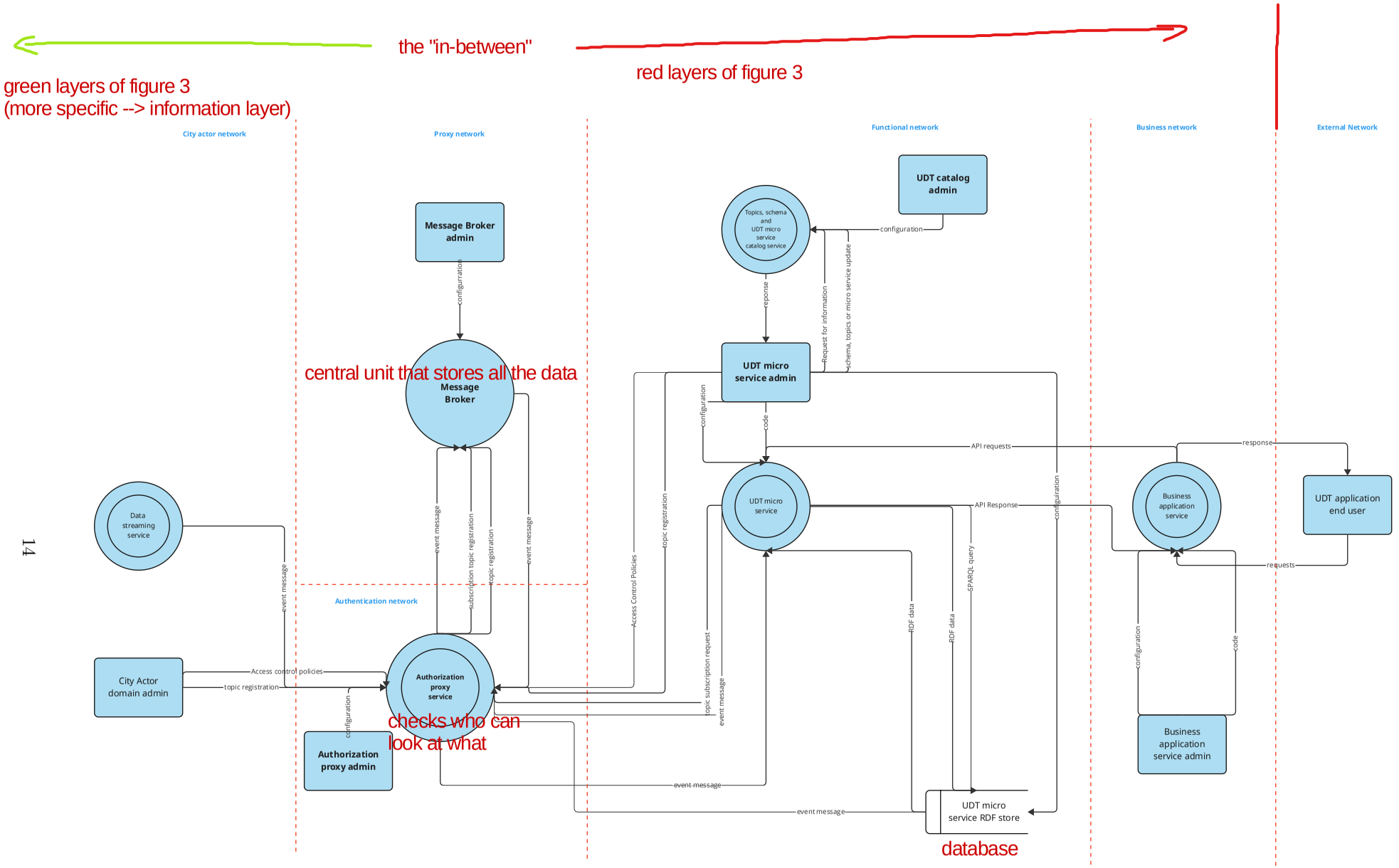

Figure 4: Data flow graph of proposed micro service architecture

5.3 Coverage of Non-Functional requirements

The use of a micro service decoupled architecture covers requirements (RN-1, RN-2, RN-3). Interdependence between micro services are limited to the type of inter connectivity they have over the data they share. Thus micro services can be deployed and managed independently of each other to some degree (RN-2, RN-3). The micro service flexibility is also exported at the level of deployment with the message broker. As long as a communication pipeline is established between them and the message broker, all micro services can communicate with each other and retrieve data from different pipelines. This allows the deployment of such services on the cloud, on premise or even on the edge of the network (RN-1). The authorization service managing all the connection between the micro service themselves but also between micro services and data streams, covers the multi party access requirements (RN-10). The isolation between private functional micro services in contrast to public business level micro services allows for some part of the UDT to be public while keeping the internal UDT system private (RN-6)

In order to interact with the platform each micro service, data stream provider and individual actors on the platform need to be authenticated. The authentication system is not directly covered by the architecture but the architecture requires and identity management service to be defined to function (RN-5).

The actuation of the UDT back to its physical twin (RN-11) is not covered, as the physical technology, digital technologies and standards needed for such interaction are still in their early development stage with most existing implementation making use of indirect actuators [24].

Requirements focusing on governance standards and legal constrains that require further evaluation to establish how these constrains can be reflected in the architecture and standards (RN-4, RN-7, RN-8, RN-9).

5.4 Coverage of the Information layer requirements

In our propose architecture, city actors push their data into the message broker via their own internal data streaming services. These data streams are then consumed by the UDT micro services and directly used to perform computation. Thus no city actor data is directly being stored on the platform, with the platform only acting as the proxy between the data sources and their consumers. The UDT micro services can store the data generated from their computation in a personal RDF store. These RDF stores can also act as data streams to be consumed by other UDT micro services via the same process as external data streams. All connections to the message broker must pass through the Authorization proxy service that define who can consume or push data via the message broker (RI-7). With all the data schema, data streams list and micro service list are indexed via the catalog service (RI-3).

Requirements regarding interoperability (RI-1, RI-2 and RI-4) are covered via the use of Semantic Web Technology RDF data model. As all data exchanged within the UDT platform must follow an RDF format.

In RDF data resources are represented as nodes, and relationships between resources are represented as directed edges between nodes. Each node is identified by a URI, which serves as a unique identifier for that resource in the graph.

One subset of URI is the Uniform Resource Locator (URL), which in addition to uniquely identifying a resource also includes its location. Following the notion of domains in URL, the same notion is applied to URI in the context of our architecture, the link between a resource and its owner is established by the URI (RI-6) with the first level of the URI path being set to the owner's domain (RI-5).

In RDF, data is stored in triplet following the Resource Description Framework (RDF) [48]. Resources are stored as subject predicate object. Subject represents a resource with a relation defined by the predicate to the object. Thus RDF does not only describe the data but also maintains the semantic of the data via the predicate so that information can be shared across services without loss of semantic [65].

The Web Ontology Framework (OWL) provides the semantics (RI-4) that defines agreed upon schema that the data must abide by. Other ontology's created specifically in the context of the UDT can also be developed and added. This allow for the creation of a knowledge graph. A knowledge graph allows for queries to be created based on the implicit knowledge coming from the accepted shared ontology [23].

RDF data can be queried using SPARQL. SPARQL queries consist of a set of patterns, which are expressed as triples that match the structure of the RDF data being queried. In our UDT archiecture, UDT micro services, make use of the SPARQL syntax onto data streams to retrieve the relevant information they need for their computations.

UDT requires access to temporal data in order to track the state evolution of its physical counter part (RI-15, RI-16). In [65] a review of ongoing temporal RDF models is provided. Two solutions provide

an approach to link historical and time series data with RDF. One proposes an extension of the RDF semantic with quadruplets, while the other makes use of existing RDF predicate to link data objects to streams and time stamps.

The C-SPARQL model (Continuous SPARQL) is proposed in [10]. In this implementation the RDF triple is extended to an RDF quadruplet where the new element is a timestamp. They present the notion of RDF streams, which are sequences of continuous non-decreasing time stamped RDF triples. Streams are created and identified via a unique stream URI. The stream is accessed via streaming windows that define the time range from which we want to receive the streaming data. Extended C-SPARQL queries are registered by the SPARQL endpoint as an event listener. As data is pushed onto the stream it is evaluated against the C-SPARQL queries and the updates are sent to their requester.

Another approached proposed in [56] makes use of default RDF semantics by including timestamps as predicates *dsv:hasTimeStamp* known as Time-Annotated RDF, TA-RDF. Data streams are composed of frames, which represents the state of a set of time stamped RDF objects associated at a given time. The membership of a resource to a frame is defined by the predicate *dsv:belongsTo*.

To query graphs following that semantic the TA-SPARQL extension is defined. This graph unlike C-SPARQL can directly be queried using standard SPARQL. The other propose a prototype that makes use of Semantic Data Stream Manager (SDSM), in front of the RDF store, that manages requests and continuously fetches update on streams to be pushed to the data consumers.

Requirements focusing on governance standards and legal constrains that require further evaluation to establish how these constrains can be reflected in the architecture and standards (RI-8). Data traceability and non-repudiation constrains is not directly addressed by the architecture (RI-13,RI-14). These will need to be considered from a data management stand point at a lower level. The same is true for RI-12, that will need to be covered via the technical implementation of the authorization proxy service.

Lastly another technical implementation that will need to be considered is the support of time series RDF vis C-SPARQL or TA-SPARQL. Although covered from a theoretical stand point, there is a gap between their theoretical implementation and commercial implementation that will need to be reviewed when implementing a proof of concept.

5.5 Coverage of the Functional layer requirements

The message broker and data stream service cover RF-1, RF-2 and RF-3.

In order to interact with the platform each micro service, data stream provider and individual actors on the platform need to be authenticated. The authentication system is not directly covered by the architecture but the architecture requires and identity management service to be defined to function (RF-5). In a similar fashion to RI-12, RF-4 will need to be covered via the technical implementation of the authorization proxy service.

For RF-6 more research will need to be performed to determine the standards to be respected by all UDT micro services and how they will reflect on the UDT architecture.

5.6 Coverage of Business layer requirements

The UDT platform propose, includes a direct link from UDT micro services to Business level services to account for (RB-1). However the exact details regarding the structure of these links requires further evaluation to determine the type of interface and connection that will be required to address the business use cases.

6 Threat and risk evaluation

This section presents the security assessments performed on the proposed architecture in Figure 3. The process was adapted from the OWASP threat modeling processes. [15]

6.1 STRIDE methodology

Originally developed by Microsoft it appears on the threat model methodology proposed by OWASP [15] and is commonly used in enterprise systems. As described in [37] STRIDE is an acronym standing for

- Spoofing X successfully identifies as Y
- Tampering unauthorized modification of a system

- Repudiation users cant deny past actions
- Information Disclosure information leakage
- Denial of Service render service inaccessible
- Elevation of privilege gain unauthorized privileged access into a system

Each of these applicable threat is checked against assets [2] divided by the categories presented in Table 5.

Element	Spoofing	Tampering	Repudiation	Information Disclosure	Denial of Service	Elevation of Privilege
Data Flows	Not Applicable	Applicable	Not Applicable	Applicable	Applicable	Not Applicable
Data Stores	Not Applicable	Applicable	Not Applicable	Applicable	Applicable	Not Applicable
Processes	Applicable	Applicable	Applicable	Applicable	Applicable	Applicable
Actors	Applicable	Not Applicable	Applicable	Not Applicable	Not Applicable	Not Applicable

Table 5: Threats applicable to asset categories adapted from [37]

6.2 Risk evaluation

For each identified threat the risk is then calculated based on two factors, the likelihood of the threat occurring and the impact that such threat can have over the system in scope [3]. Likelihood is further divided into two sub-parts, likelihood of the threat event occurring and likelihood of the threat event to cause an adverse impact. The overall likelihood calculation is described in Table 6. The risk score calculated as a product of the impact and likelihood is presented in Table 7.

The scores are represented by qualitative values. This approach was chosen as quantitative risk assessment requires existing data on the system analysed. Given that UDT architecture standards have yet to be defined, access to such data is limited. Furthermore the architecture presented does not include the exact technologies used for each service which makes our interpretation of the services running and overall structure of the system more subjective.

Likelihood of event initiation	Likelihood of adversarial impact				
	Very low	Low	Moderate	High	Very High
Very High	Low	Moderate	High	Very High	Very High
High	Low	Moderate	Moderate	High	Very High
Moderate	Low	Low	Moderate	Moderate	High
Low	Very Low	Low	Low	Moderate	Moderate
Very Low	Very Low	Very Low	Low	Low	Low

Table 6: Overall Likelihood calculation derived from [34]

[2] The assets are listed in appendix table **??**
[3] See **??** and **??** in the appendix for more details

Overall Likelihood	Level of Impact				
	Very low	Low	Moderate	High	Very High
Very High	Very Low	Low	Moderate	High	Very High
High	Very Low	Low	Moderate	High	Very High
Moderate	Very Low	Low	Moderate	Moderate	High
Low	Very Low	Low	Low	Low	Moderate
Very Low	Very Low	Very Low	Very Low	Low	Low

Table 7: Risk calculation derived from [34]

6.3 Security Policies and standards

Here we are presenting the security principles used to guide the creation of security policies:

- Economy of mechanism: keep the security mechanism in place a simple as possible
- Confidentiality: prevention of unauthorized access to information
- Integrity: prevention of unauthorized modification to information
- Availability: prevention of denial of service
- Accountability: entities performing actions on the system are answerable to the actions they perform
- Traceability: track down origin of an action or change on the system
- Resiliency: protect the integrity of systems against adversarial threats
- Separation of privilege: segmentation of privileges required to perform specific actions on the system
- Least privilege: minimum number of privilege required by an entity to perform its expected tasks
- Multi factor authentication: an authentication method that requires a user to presence 2 or more verification factor to authenticate

These security principles assist in the creation of the security policies by guiding and clearly defining the security principle a security policy is aiming to achieve. Table 8 list the security policies that must be respected by the architecture framework derived from known security threats in DT [8] and from the requirements presented in subsection 4.1 linked to security.

ID	Statement	Objective	Security concept
P-1	Services and actors should not have the capability to write to data sources they do not own	Only the owner of data sources must have the capability to add, modify or delete the data it contains.	Integrity
P-2	Services and actors should not have the capability to access data without permission from its owner	Entities consuming data must be granted permission to consume that data by the data owner	Confidentiality
P-3	Services should not have the capability to directly interact with the data sources they do not own	Centralize the location where data flows to facilitate access control enforcement	Economy mechanism, Resiliency, Availability
P-4	Integrity of data passing through the UDT platform should be preserved	Ensure that the data transiting through the platform cannot be altered	Integrity
P-5	Only owner of a data source should define access control policies associated with that data source	Access control policies can only be managed by the resource owner what S1 focused on	Confidentiality
P-6	Data transiting on the platform should be traceable back to its data owner	Ensure data traceability	Traceability, Accountability
P-7	UDT micro services should remain operational and have some degree of resiliency against DDoS	Ensure UDT model remains accurate at the business layer	Availability
P-8	Entities accessing the platform must always be authenticated	Ensure that only users and services with valid clearance can access the platform	Confidentiality

Table 8: Security Policies

6.4 Threat model

6.4.1 Default security measures

In order to maximize the relevance of security threats identified by the threat model, subsubsection 6.4.1 provide the list of security measures assumed to be present by default on our UDT platform. These security measures are commonly applied by most data platforms and cloud providers.

6.5 Threats

The full threat model analysis can be found in **??**. The aim of the threat analysis was to identify high risk components and propose technical mitigation solutions addressing the high level threat. These threats are presented in Table 10, overall three high threats were identified, centered around the tempering or unauthorized access to the data flowing through the UDT platform.

The message broker is single point of failure in the UDT architecture, thus any compromise of this service will ripple down into all other services using the UDT platform. The authorization proxy service also acts as a single point of failure as it manages the access control of micro services to data streams. If the access control can be by passed or is compromise this could lead to large information disclosure.

ID	Description	Technical Implementation	Security Principle	Implication if not present
SM-1	Encryption of data at rest	Enable encryption of data on every database	Confidentiality	Entity with direct access to the machine be it physically or digitally) storing the data can compromise its confidentiality
SM-2	Encryption and integrity of data in transit	Apply the TLS protocol for all communications	Integrity, confidentiality	Any entity able to sniff the traffic will be able to break data confidentiality and integrity.
SM-3	2 factor authentication enabled for all admin users	Use 2 factor authentication technique based on: Something the user owns Something the user knows Something the user is	multi factor authentication	Increase the effectiveness of phishing attacks on accounts and thus the likelihood of compromise. As certain admin accounts can grant significant privileges on the platform. Spoofing these accounts can highly impact the platform.
SM-4	Logging and monitoring	Use logging service such as elastic search agents deployed on running services using service mesh	Traceability, non-repudiation	Lost of traceability over changes performed on the UDT platform. Hinders the identification of the origin of compromise in the event of an attack
SM-5	Zero trust model, where all services running on the platform have to be authenticated by default to communicate with other services	IDP or certificate authority, authentify all users and services present on the platform	Authentication	Without authentication it is not possible to manage access control policies to services and data sources
SM-6	Network isolation	Deployment of firewalls on the edge of each network to filter outgoing and incoming traffic	Least privilege	With no network isolation the compromise of a single service on the UDT platform acts as a single point of entry into the entire UDT system.

Table 9: Default security measures

ID	Threat	Description	Likelihood of event	Likelihood of adversarial impact	Overall Likelihood	Impact	Risk Score
PT-2	Tampering with message broker configuration and data	All data flow from data stores to micro services go through the message broker. If data can be tampered at this service that will impact all micro services present in the UDT	Low	Very High	Moderate	Very High	High
PT-13	A malicious actors manages to remove, modifies or add access control policy to resources it does not own	Access control policies and enforcement are at the core of the UDT architecture. If the access control policies are tampered with the trust in the entire UDT system is compromise.	Low	High	Moderate	Very high	High
PT-17	Bypass of authorization service allowing unauthenticated access to micro services and data stores directed	Access control policies and enforcement are at the core of the UDT architecture. If the access control policies are tampered with the trust in the entire UDT system is compromise.	Low	Very High	Moderate	Very high	High

Table 10: Identified high risk threats

7 Identity management for access control

To address the threat linked to access control an authentication system must be put in place.

7.1 Security requirements of the authentication system

Based on the security requirements identified in subsection 4.1 this section described the authentication requirements that should be covered by the authentication system used by the UDT platform.

Following the economy of mechanism principle the authentication system should permit the authentication of services running on the platform along with admin and other users interacting with the platform. The authentication system should support the zero trust model presented as a default security measure, [4]. Given the multi actor components of the UDT platform each city actor should have the capability to manage its users and services credentials independently. Depending on the level of transparency of the overall UDT platform, the authentication system may need to provide to the platform stakeholders proof that no malicious credentials have been issued using their respective identities.

[4] See SM-5 in subsubsection 6.4.1

7.2 Certificate based authentication

Public key certificate is the standard used today to secure client to server connections over TLS [55]. The certificate based authentication is a common method use in zero trust architecture as it allows for two independent services to perform mutual TLS authentication. Their common use in decentralized systems and in recent service mesh architecture [13] make it a good candidate for the UDT platform authentication system.

Certificate-based authentication can serve two purposes:

1. Authenticate the entity holding the private key associated with the public key present in the certificate
2. Allow for identifiable attributes to be included directly in the certificate.

7.2.1 Certificate attributes

In this section, we will go over the certificate structure and attribute as presented in the RFC 5280 [11] that can be used for a UDT authentication system. Here is a list of certificate default attribute names:

- Serial Number: a unique certificate identifier
- Issuer: section that contains information identifying the certificate the signed the current certificate containing the same attribute than subject
- Subject: section that contains information identifying the certificate
  - CN: common name uniquely identifying the entity holding the certifcate
  - O: organization creating the certificate
  - email: email address of the user associated with the certificate (if it is a user certificate)
- Validity: define the time from where the certificate can be considered valid and the time at which it becomes invalid
- Subject key info: holds the public key of the certificate

Certificates also have an extension section that enables the addition of extra attributes adapted for specific certificate usage. Each attribute can be marked as critical meaning that the systems using the certificate must reject the certificate if it cannot validate it or does not recognize it. Or they can be marked as non-critical meaning that systems using the certificate must process the attribute if recognize otherwise it can be ignored.

As a mean to standardize certificate attribute names, all attribute names must be represented by an Object Identifier (OIDs) [49] it is that identifier that is used to encode the attribute names in the certificates. IOD code is provided upon the registration of an attribute by the International Standards Organization (ISO) or International Communication Union (ITU). Thus attribute cannot independently be added to certificate extensions but rather requires to go through this larger process prior to be used in a general setting.

Thus the creation of custom extension attribute for the UDT platform although feasible for internal use will not be supported by default certificate verification technologies and may later come in conflict with newly issued IOD. In order to avoid these complications, only already registered extension attribute were considered:

- key usage: define the usage that a certificate can make of its private and public key
  - Signature: public key can be used to verify digital signature used on anything but a certificate
  - keyCertSign: allow public key to be used to verify signature on other certificate
  - cRLSign: allow public key to be used to verify signature on certificate revocation lists
- Basic constrains: if set allows the certificate to issue other certificate
- Name constraints: indicate the name spaces in which subsequent certificate issued must remain
  - DNSname: constrain over domain associated with the certificate must be checked against subject CN and alternative name extension attribute

  - rfc822Name: restriction over subject email attribute
  - directoryName: applied to the subject attribute of the certificate
- Subject Alternative Name: allows to extend the set of identities in addition to the one in the CN subject field.
  - DNSname: domain associated with the certificate
- Extended Key usage: defined extra key usage in addition to the usage defined in the key usage extension
  - TLS client AUTH: allows certificate to be used for client side authentication
  - TLS server AUTH: allows certificate to be used for server side authentication

7.3 Certificate based authentication for UDT platform

Figure 5 represents the Public Key Infrastructure (PKI) proposed for the UDT platform manage by the federation.

The federation creates a root certificate, following the security best practices for certificate management, the root certificate controlled by the federation signs a primary intermediate certificate. That intermediate CA is now use two create two types of intermediate CA.

7.3.1 City actor intermediate CA

This intermediate certificate can be issued by the federation to each individual city actors. The Name Constraints extension attribute is used to restrict the certificate domains that can be issued by this intermediate CA to only the ones owned by the city actor. This applies to DNSnames and email addresses used to identify admin users.

Using that intermediate CA, the city actors can now create certificates that identifies data streams, micro services and admin users it controls

7.3.2 Federation secondary intermediate CA

This intermediate CA is issued by the federation to the federation following the logic than for the city actor intermediate CA.

7.3.3 Authentication requirements covered by the CA architecture

This approach proposes a federated authentication system, delegating the identity management to the city actors while retaining some centralized controlled over the chain of trust should a city actor becomes malicious or get compromise. The CA also integrates by default the least privilege principal via the Name Constraints that ensures that city actors and even the federation cannot issue certificates for domains and actors that are not under their control.

Certificate based authentication standardizes the authentication approach to be used by both services and actors. The use of such authentication method can also act as a form of two factor authentication for actors when combined with the more common login password approach.

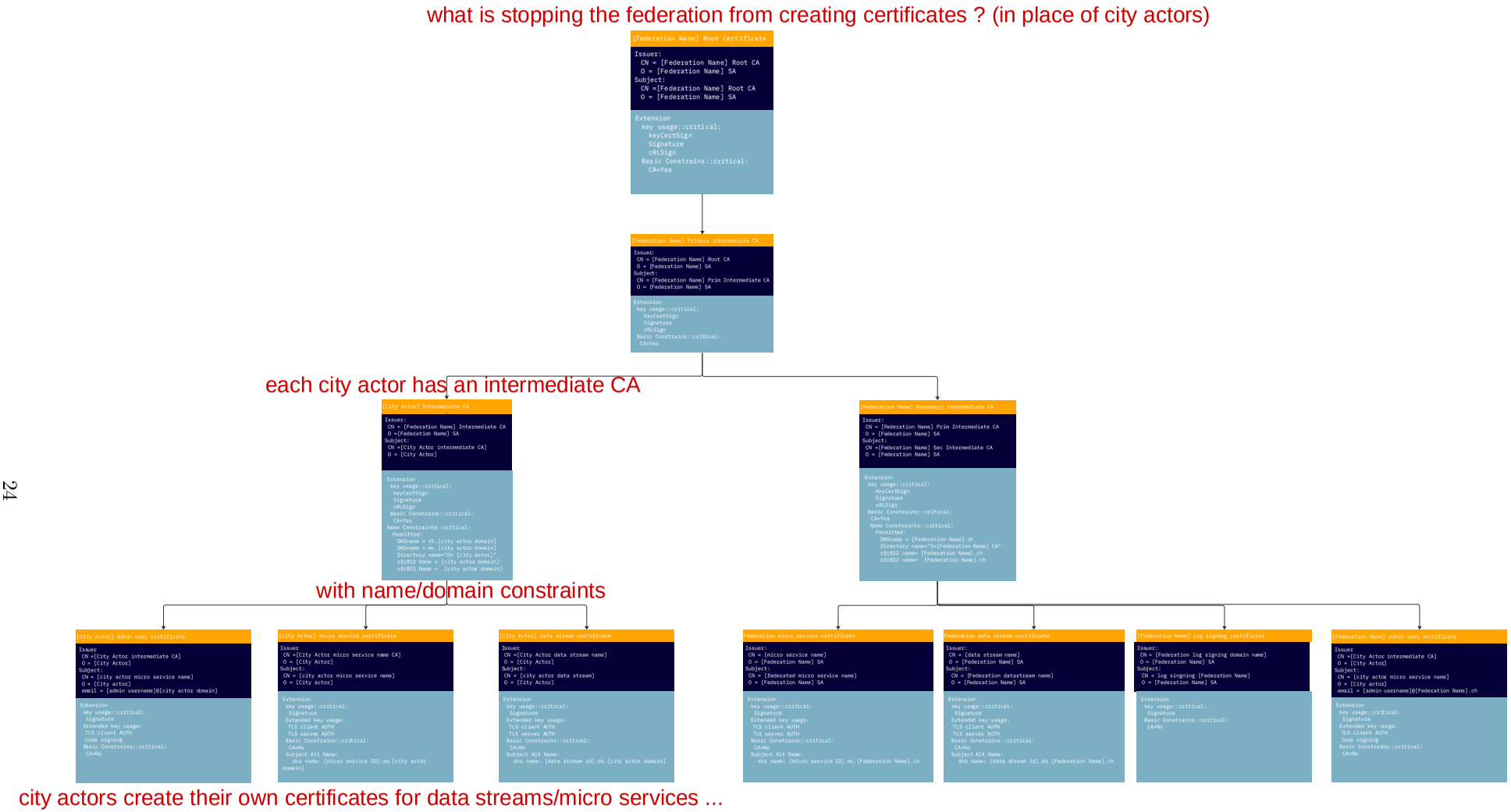

Figure 5: Proposed public key infrastructure for UDT

if CA not in logs --> auth denied

this way city actors track all CA, and check CA in their domains are authentic

7.4 Certificate Transparency (CT)

One of the unique trait of the UDT platform is its multi actor component. The decentralization of identity management via the certificate authority architecture poses a challenge for the federation and other city actors to track the issuance of certificates that can be used on the platform.

A first version of certificate transparency protocol is described in RFC 6962 [40] is an initiative to index and publicly disclose the list of all issued certificates be it for private domains or the entire internet.

The objective is to have logging server that store append only logs using a Merkle tree structure. The tree head of the Merkle tree must be signed by a valid certificate owned and controlled by the log server this is defined as a signed tree head (STH). append only logs --> blockchain like security

For a certificate to be considered valid in a system implementing certificate transparency, a certificate holder must prove that its certificate exists in the logs or will appear in the logs by a defined upper bounded time. This proof is presented as a Signed Certificate Timestamp (SCT) issued by the logging server to the certificate authority upon registering their certificate. These SCT signed by the logging server promises the certificate inclusion in the logs with a maximum merge delay. This ensures that all valid certificates are indexed and can be tracked, as certificates not present in the logs are considered invalid by the system. It also ensures that entities can discover the existence malicious certificates issued under their name via the logs within a time frame bounded by the maximum SCT duration

deal with "lag" in system

In order to ensure that that all entities querying the logs have the same vision of the logs at a given time frame and to keep the logging server accountable. Independent monitoring log services make copies of the logs and exchange STH. --> since everything is logged, if federation starts acting up (malicious way)

Certificate transparency protocol can be used in the context of UDT to keep the federation in check. As the federation acts as the root of trust for the system, deploying the CT mitigates the risk that a malicious actor within the federation or within a city actor organization manages to generate certificates for client actors or the federation itself without being noticed. Figure 6 describes how the CT could be implemented within the UDT platform.

logs will tell the truth

never mentions difficulty of keeping the CA logs !!!

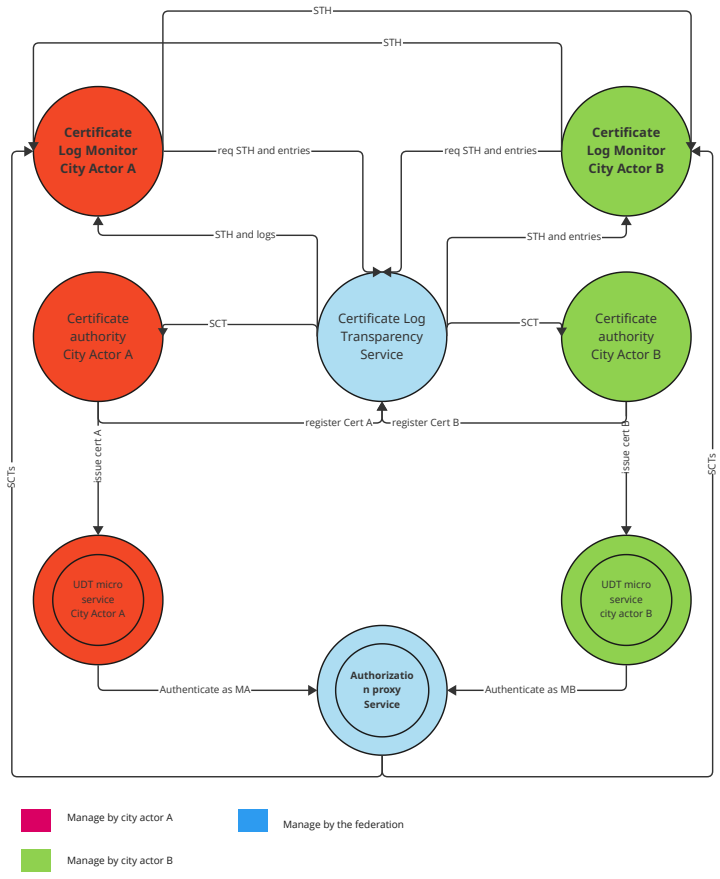

Figure 6: certificate transparency implementation for udt platform

is the federation above everyone ?
can federation revoke any CA ??

7.5 Certificate revocation

Certificate issued may be revoked by the CA that issued it or any CA above it. To do so different methods can be used. The most common one was to use a certificate revocation list (CRL) maintain by the certificate authority. To revoke a certificate issuing certificate authority, the CA adds an entry containing the certificate sequence number, signature of the issuing certificate and reason for revocation.

Clients or server wanting to check the revocation status of a certifiicate would download that list and look for the presence of the certificate sequence number they wanted to check against. As this process requires each service to download the full list CRL periodically, such system require a significant storage space and computational time as the CRL list grew.

To palliate to these limitations, the Online Certificate Status Protocol (OCSP) defined in RFC 6960 [58] was created. OCSP requests are sent by the services wanting the check the status of a certificate to an OCSP responder. The server sends a signed OCSP response that indicate whether the certificate has been revoked or not.

external point of failure

Version to of CT currently drafted in RFC 9162 [41] propose the addition of extensions to the protocol. An extension that could be included would be the addition include a Certificate Revocation Status (CRS) within the logs. Such extension would enable the CT logging server to act as an OCSP responder, by including the certificate revocation status within the logs. This way certificate and their revocation status are ensured to be transparent. Figure 7 presents how such system can be implemented on the UDT platform.

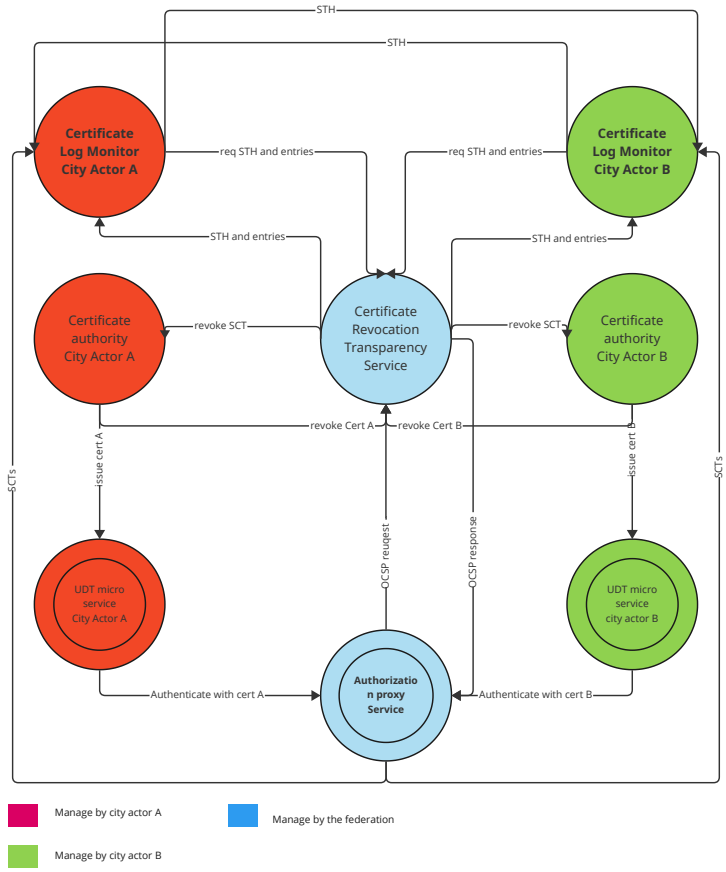

Figure 7: Certificate transparency implementation for UDT platform

7.6 PKI considerations for implementations

The theoretical architecture proposed above covers the authentication of entities using the UDT platform. When moving to a technical implementation the following points will need to be considered.

The name constraints in the certificate extended attribute is not enforced upon the creation of new certificates. The enforcement of the constrains is applied by the TLS client checking the certificates. However the following checks are not always performed correctly [1]:

- DNSName must be checked against all existing CN field in the certificate in addition to the ones in the subject alternate name extension.
- DNSName must be checked against all CN fields and reject the certificate if one of them is not in the whitelist
- Constrains present in all upstream parent certificates must be checked against all their direct descendant certificates.

As these are not always apply the BetterTLS initiative [4] provide a tool that tests if TLS clients implementation process the name constraints correctly.

The certificate transparency implementation library can be found in [2]. This is the one currently being developed by the certificate transparency initiative. At this time, this implementation cover RFC 6962 [40]. However the OCSP protocol is not yet supported by it. Currently the library is only compatible with a CRL list.

section 7 --> deals with authentication of entities (city actors)
section 8 --> deals with creation of topics by CA, and the rules "who can read my topic"

8 Authorization method for resource access control

The first step to identify the type of access control mechanism required by the UDT architecture is to define the type of resources that need to be protected and which entities will access them.

In the case of UDT, the platform provide proxy layer allowing real time data streams coming from city actors to be read by UDT micro services. City actors retain full control over the data that they choose to send over a specific stream. The stream acts as a layer of abstraction between the data being consumed and the method through which it is consumed, thus city actors must have the capability to control which entity can access their data streams to keep control over which entity can access their data.

8.1 Message brokers data stream management via topics

In our UDT platform, data streams are flowing through the message broker that relays it to its consumers. In event driven architecture, topics are used as a mean to make these data streams addressable. Data is packed into a message event that is then sent to the message broker via a topic. For a message event to be published into a topic, the topic must be registered in the message broker and the message event source registers as a publisher for the given topic. To consume data present in a topic, consumers must registered as a subscriber to that topic.

Topic are identified via a hierarchical scheme $a/b/c/d$, which allows for the classification of information. Topic subscription can be performed via the topic absolute name or relative name with wild cards.

Thus in order to keep control over which entity gets to access their data, city actors must be able to define access control policies over which entity can access the topics they publish to, making the topic the resource to be protected on the platform [17].

8.2 User Manage Access Control (UMA) standards

After defining the type of resource that needs to be protected by the platform we can now define the type of access control method that would address the requirements identified in subsection 4.1. The multi actor component of the UDT, and control that these actors must maintain over their resource requiring a federated access control.

The UMA model proposed by the Kantara initiative [46, 47] extends the traditional OATH 2.0 authorization framework [28] in order to let independent entities control the access to resources they own over private or public networks. Figure 8 describes the overall architecture and data flow for UMA.

8.2.1 Resource Server (RS)

The RS is responsible to manage resources that need to be protected, working as a Policy Enforcement Endpoint (PEP). The RS authenticates to the Authorization server by holding a protection API access token (PAT). This grants it access to the following endpoints on the Authorization server:

- resource registration: registers a resource to be protected by the authorization server with defined scopes.

- create a resource description
- read resource description
- update resource description
- delete resource description
- list resource description

- permission: allow the resource server to request access permissions on behalf of a requesting party over some resources.
- token introspection: allow the resource server to retrieve the content of Requesting Party Token (RPT), this allows it to gain permission scope and the resource name the RPT grants access to. Based on these the resource server can either grant or deny access to the RPT holder.

8.3 Authorization server

The Authorization server acts as a Policy Decision Point (PDP), having the following responsibilities and structure:

- Verify and identity and performs access policy evaluations against access requests from requesting parties
- Create and index RPT tokens to be used by requesting parties to access a resource
- Revoke access
- Enable resource owners to set up their access control policies on the resources they own
- has an authorization endpoint through which requesting parties with an authorization API token can request access to specific resources with a specific scope.

The resource owner uses the resource server to register its resource and the authorization server to define the access control policies linked to the resources it owns.

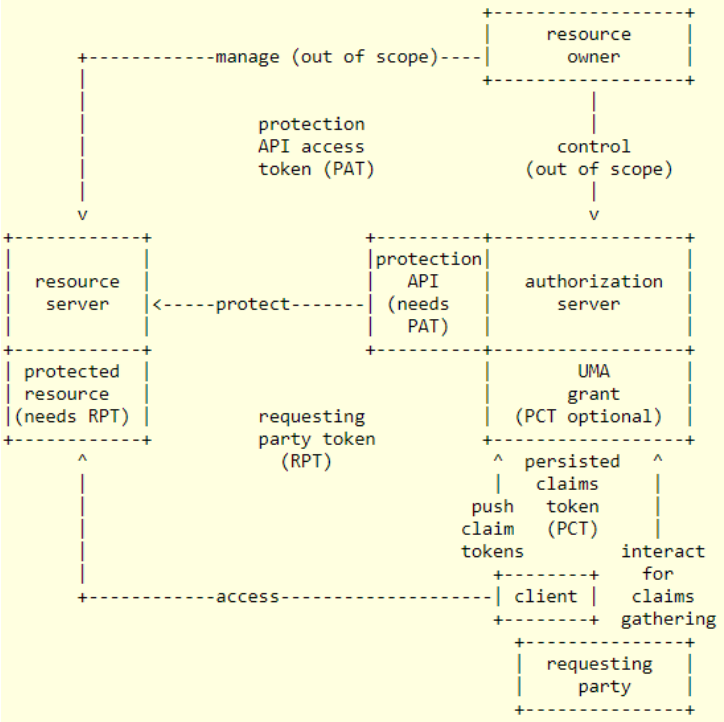

Figure 8: UMA architecture derived from [46]

8.3.1 Out of Scope elements of UMA

"resource" authentication

UMA standards does not define the authentication method through which a resource owner or requesting party can authenticate to the resource or authorization servers. In our case this is covered by the client certificate authentication. The UMA also does not enforce any special policy language used to define the access control policies, these would need to be defined. The method through which a resource owner can register a resource and through which a requesting party can request access to a resource is also not part of the standard. Finally resource discovery is not directly integrated in the UMA standard.

8.4 UDT platform UMA implementation on topics

In [17], an access control architecture adapted for IoT streams over an MGTT message broker with UMA is proposed. Building on their approach the following UDT access control model is proposed.

8.4.1 Topic creation

who owns a topic

As topics are registered under a hierarchical paths, this paths can be used in correlation with admin domains path to establish ownership. Topics that are subsets of the admin domain are considered to be owned by the admin user following the economy of mechanism principle. Topics can thus be broken down by city actor organization on the first level and then into departments and so on e.g transport_company_domain/train_subdommain/train_type_subdomain/data_type_subdomain

To create a topic on the message broker, the city actor admin or UDT micro service admin first sends a topic creation request to the resource server. That requests includes, the topic name, a description of the data to be streamed through that topic along with its schema. The admin user signs the request with its client certificate. The resource server verifies that, the client certificate is still valid, that the signature on the request came from the same client certificate and that the certificate domain is a super set of the topic name path. Upon validation the resource server, sends a topic creation request to the message broker and sends a resource registration request to the Authentication Server for that topic.

8.4.2 Access control management over topics

Resource owner connects to the Authorization server with their client certificate. Based on the domain associated with the client certificate, the Authorization server retrieves all subset topics of that domain. The Resource owner can then modify or create access control policies on these topics. The resource owner can also see request access made by requesting parties awaiting its approval.

8.4.3 Publishing to a topic

make sure it comes from a reliable CA

In order to publish to a topic, each message sent to a topic is signed with a data stream certificate. The resource server intercepts these messages, verifies the signature and certificate validity. If no RPT token is present, the resource server forwards the request to the Authorization server following the UMA standards and sends back an authorization API token to the requesting party that uses it to request access from the Authorization server

loggs

If the RPT is present, the resource server sends it to token introspection endpoint of the Authorization server and verifies that its scope and resource matches the resource and scope required for the requesting party to publish to the topic.

8.4.4 Subscribing to a topic

This step follows the exact same steps than the one defined in the UMA standard for resource access requests.

8.5 Authorization policy language

As mentioned in subsubsection 8.3.1, UMA does not enforce any specific authorization policy language. The policy language aim is to be able to determine from entities claims and context whether that given entity is granted access to a resource or not. In the case of the UDT the identification of the desired properties of such language will require further analysis over the use case and resulting access control capacities needed by the UDT.

However a similar access control adapted for federated micro service system is proposed in [52]. This approach proposes the use of UMA in combination with (Open Policy Agent) OPA policy language [6] to implement access control over a federate architecture. Another common policy language also mentioned is eXtensible Access ControlMarkup Language (XACML), however this policy language is does not easily scale [20] and is very verbose as compare to OPA. S1 looks at this

A proof of concept is also presented making use a Keycloack [5] that is an open source identity and access management tool. Keycloack uses its local javascript policy standard to evaluate access control policies. Thus a javascript policy was created making a proxy call to the OPA endpoint where the access control policies are defined and evaluated. In the context of the UDT OPA policies could be defined to enforce the topic vs domain name path constraints.

9 Message broker data management

single point of failure

As the message broker manages the entire flow of data on the UDT platform, any entity with access to the message broker will gain a high privilege role on the entire UDT system. These can include, the modification of message events, topic creations, and sniffing on topic contents.

9.1 Data traceability in UDT

done by the CA --> federation cant enforce it themselves

In the context of UDT, data streams will be produce by many different city actors, that will have to adhere to a set of common guideline when it comes it its reliability and quality assurance. The set of standards will need to be defined by the federation. However one important aspect of data quality insurance is its traceability through the UDT system. This traceability knowledge only give more contextual information over the data, it also allows for error sources to be quickly identified.

In our proposed architecture all data exchange follows the RDF format which includes the use of URI to uniquely identify a data object. URI can thus also represent a path in a similar fashion to topics and URL (which is subset of URI). Hence one way to enforce traceability can be to use the URI paths to determine data object origin e.g http://transportdomain/bus/bustype/busID. Here the transportdomain identifies a unique city actor, and thus by looking at the data objects URI, ownership and thus traceability can be established.

The main limitation being the method through which this standard can be enforced on the platform.

9.2 Data integrity in UDT

As data is flowing through the platform, its integrity must be maintained. Data integrity in transit is ensured via the use of TLS communication. However the message broker, aggregates all the data flows. Thus modification to the data there could occur should a malicious actor gain control over it.

To mitigate this risk one method would be to include a message authentication code (MAC), along with each message event transmitted. The presence of the MAC along with the message event ensures two things.

1. That modification to message content can be detected
2. Non-repudiation can be established the MAC acting as a form of digital signature

This could be done using the client certificate private key, that could sign the hash of the data exchanged, hash and signature that are then verified by the consumer after receiving it from the message broker.

Limitations to this system includes the extra overhead that will be necessary to hash and sign as well as verify the hash and signature on the receiver side.

computational cost !!

9.3 Data confidentiality

As all data flows converge through the message broker, access to it could leak all the data contents flowing through the platform. One common mitigation strategy is to enable encryption of data at rest. However as the service storing and transmitting the messages is the same, the encryption and decryption will be managed by the same service. Thus access to the service could eventually lead to a compromise of the encryption key.

~not really a solution

if we use end to end encryption, we basically lose the ability to broadcast a message

--> we would have to set up a private key between a creator and each reader of a given data stream
this would be costly, and render the whole system complex (and we would lose some advantages)

One method would be to enable end to end encryption on the messages data via a Diffie-hellman key exchange prior to the creation and consumption of messages in a topic. However this would drastically reduce their functionality, as one of their key aspects is their ability to be consumed by multiple entities.

To address this limitation an encryption scheme involving multiple parties would be required. However these encryption schemes such as Distributed Diffie-Hellman or Tree-based Group Diffie-Hellman [38] or J-PAKE [27] all involve a partial key exchange between the participants, which must be done over a secured channel. As in our architecture all communication between micro services passes through the message broker, this would either require the creation of a second communication system not going through the message broker, or the addition of an extra communication framework through the message broker, that allows end to end encryption used to exchange partial keys.

Additional consideration involve the addition or removal of participating entities which require further architecture changes representing an additional layer of complexity for message exchanges.

10 Proposed proof of concept (POC)

This section aims to present a POC derived from the data flow architecture presented in Figure 4 and taking into account the mitigation addressing its high threats. This proposal also aims to show the current limitations and work that would need to be done to address the theoretical gap between existing technical solutions and the theoretical assumption made to create the data flow architecture. The POC is presented in Figure 9. The sequence diagram representing the topic registration and topic subscription are presented in Figure 10 and Figure 11 respectively.

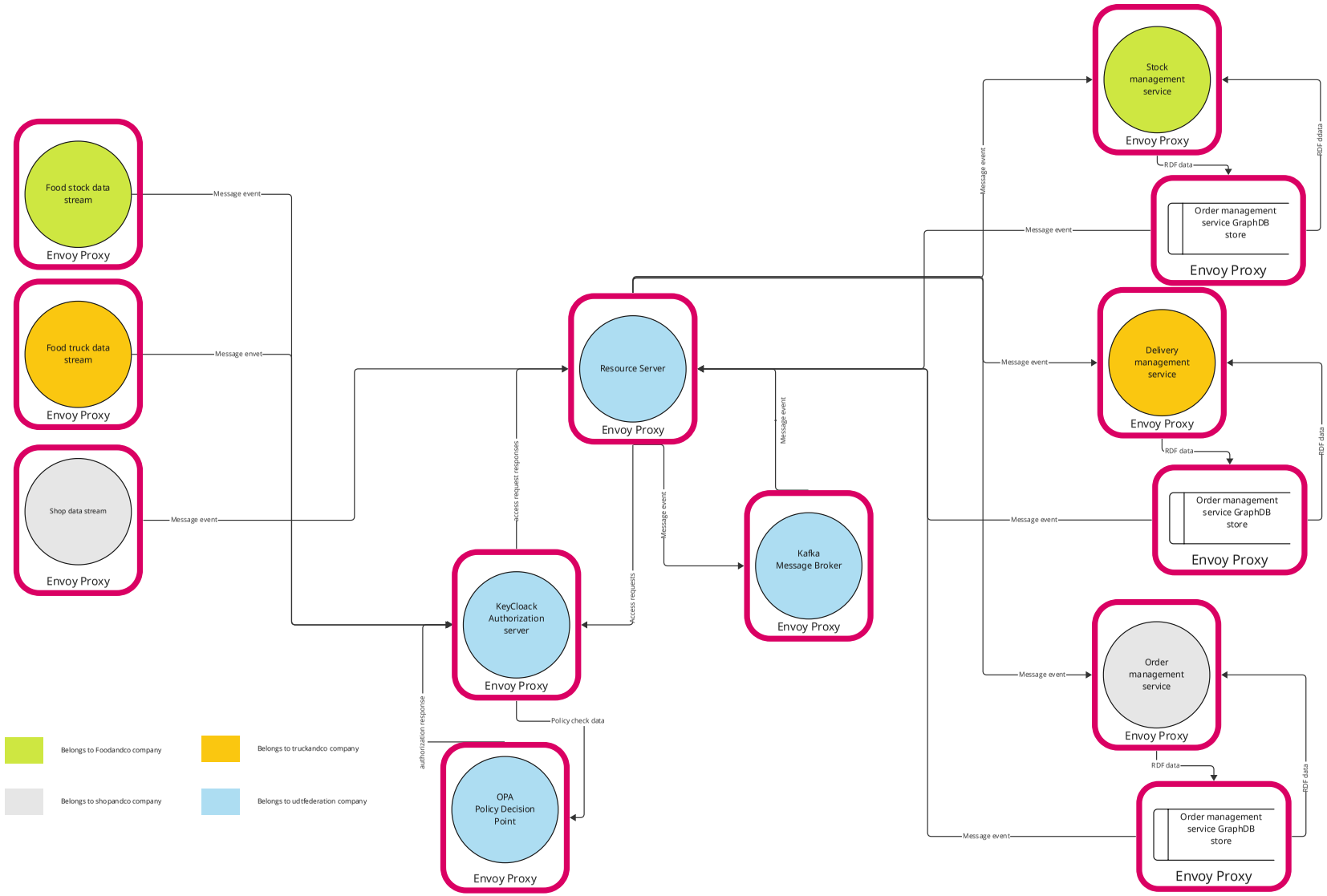

Figure 9: Proposed POC architecture

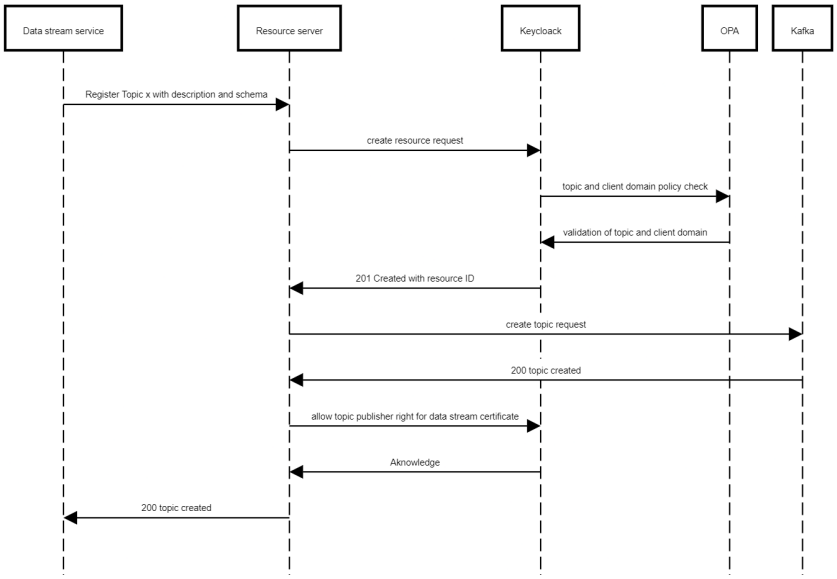

Figure 10: Topic creation sequence diagram

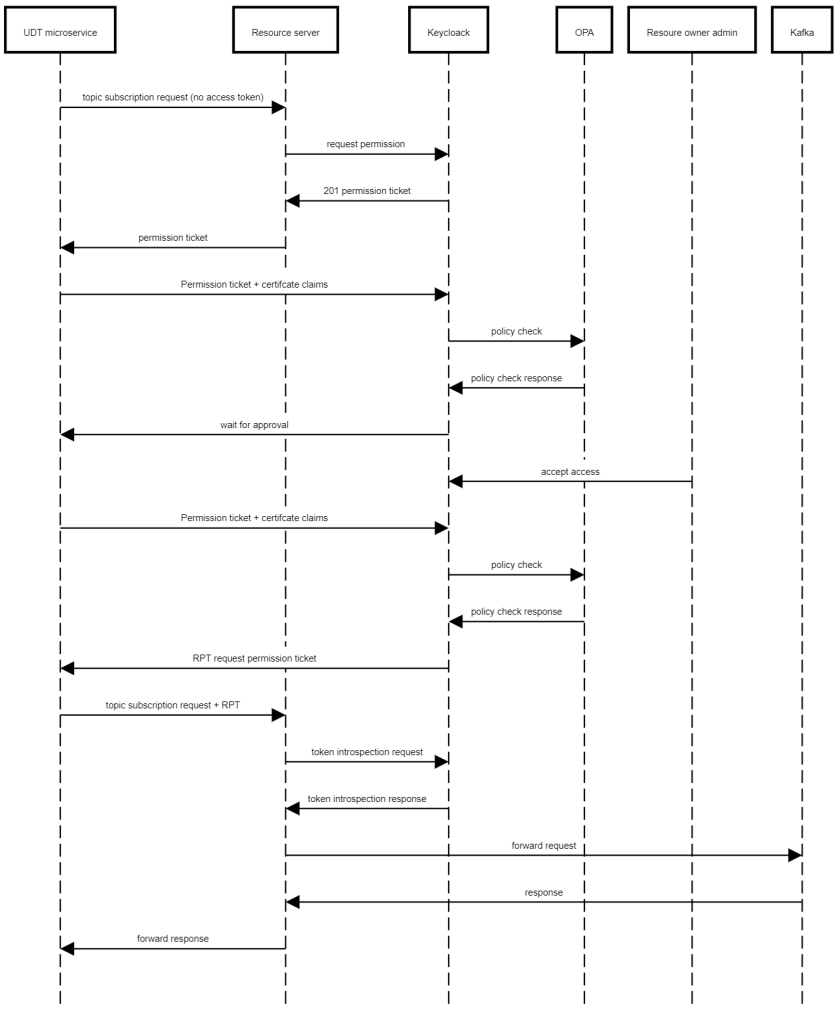

Figure 11: Topic subscription sequence diagram

10.1 Description of the use case and actors

The goal of this POC is to present a use case centered arround the flow of goods within a city.

Shopandco company provides a data stream to its order management service that controls the level of stock of food items e.g number of carrots available at one store location. The order management service uses this flow to define when a new order needs to be placed. That order is sent to the stock management service owned by Foodandco company.

The Foodandco company can be seen as a farmer cooperative, that provides a data stream to its stock management service, keeping track of food items produce on a daily basis by their partners. When receiving on order via the order management service, the service maps the order to its internal stock and sends a transportation request to the delivery management service managed by Truckandco company.

The delivery management service receives a data stream from its truck fleet, keeping track of the trucks current location and utility. The delivery management service upon receiving an transport request can allocate a truck in its fleet and send a transport approval back to the stock management service, that itself sends a order approval back to the order management service.

10.2 Implementation coverage

This POC proposal does not cover all aspects of UDT requirements. Rather its aim is to show how data can be exchanged between different city actors through the use of client certificates for authentication and UMA for authorization. The following aspects are not covered:

- Data traceability
- Data integrity
- Data product and services discoverability
- Certificate management system
- Integration with business applications
- Logging

However it could still be used as the base through which these aspects can be added.

10.3 Technical implementation

10.3.1 Envoy proxy

The envoy proxy main goal in this architecture is to off load the responsibility of micro services to authenticate client certificates. Envoy proxy, is the proxy used by Istio, a service mesh technology [44]. The main benefits of using envoy are:

- Has a growing community
- Can be deployed using Istio
- Can be dynamically configured
- Handles the certificate name constraints verification by default [1]
- Can be used to log and monitor network traffic

These proxies would hence be configured to perform client certificate verification and mutual TLS authentication between services. The big advantage is that a generic configurations can be used and deployed on all the micro services of the system, and then slightly adapted to suite the specifics of each micro service.

10.3.2 Kafka role of message broker

The kafka message broker will be the one managing the flow of messages from their source to their destination. The topic creation, modification and management is controled by the resource server, acting as a layer of abstraction between kafka and the rest of the services.

10.3.3 Resource server

The resource server is a service created for the UDT platform, responsible for managing the creation, deletion and modification of topics via kafka along with playing the role of the Resource server described in the UMA protocol. It is the only service directly communicating with Kafka.

10.3.4 Keycloack

Keycloack supports the UMA standard by default, acting as the authorization server. The parts that will require further development is the topic to resource owner association. By default keycloack allows anyone to be the owner of a resource, but in our UDT POC, owner of topics should have a domain that is a superset of the topic name path, this can be enforced via OPA.

10.3.5 OPA

Keycloack can be connected to the OPA service via its default javascript policies interface. The role of OPA could include the enforment of ownership over topics with a superset domain association from the resource owner. In addition OPA policies could be pushed by resource owners to automate resource access validation for certain services parties belonging to a specific set of domains or having other specific attributes.

10.3.6 GraphDB RDF store

This implementation of RDF store has a minimal learning curve with an intuitive design as compare to other solutions such as Apache Fuseki or Virtuoso. It also provides a plugin API in the form of Java classes allowing for customization that can be used to trigger message events to be submitted to topics.

11 Conclusion and future work

This paper covers to some degree some of the requirements identified to create a UDT platform along with potential leads to address the high security threats identified from the threat model. However the following methods will

References

[1] BetterTLS, . URL https://bettertls.com/.

[2] GitHub - google/certificate-transparency-go: Auditing for TLS certificates (Go code), . URL https://github.com/google/certificate-transparency-go.

[3] ifcOWL, . URL https://technical.buildingsmart.org/standards/ifc/ifc-formats/ifcowl/.

[4] BetterTLS, January 2023. URL https://github.com/Netflix/bettertls. original-date: 2017-01-03T19:18:04Z.

[5] Keycloak, March 2023. URL https://github.com/keycloak/keycloak. original-date: 2013-07-02T13:38:51Z.

[6] Open Policy Agent, March 2023. URL https://github.com/open-policy-agent/opa. original-date: 2015-12-28T22:08:25Z.

[7] Ejaz Ahmed, Ibrar Yaqoob, Ibrahim Abaker Targio Hashem, Imran Khan, Abdelmuttlib Ibrahim Abdalla Ahmed, Muhammad Imran, and Athanasios V. Vasilakos. The role of big data analytics in Internet of Things. *Computer Networks*, 129:459–471, December 2017. ISSN 1389-1286. doi: 10.1016/j.comnet.2017.06.013. URL https://www.sciencedirect.com/science/article/pii/S1389128617302591.

[8] Cristina Alcaraz and Javier Lopez. Digital Twin: A Comprehensive Survey of Security Threats. *IEEE Communications Surveys & Tutorials*, 24(3):1475–1503, 2022. ISSN 1553-877X. doi: 10.1109/COMST.2022.3171465. Conference Name: IEEE Communications Surveys & Tutorials.

[9] Muhammad Babar and Fahim Arif. Smart urban planning using Big Data analytics to contend with the interoperability in Internet of Things. *Future Generation Computer Systems*, 77:65–76, December 2017. ISSN 0167-739X. doi: 10.1016/j.future.2017.07.029. URL https://www.sciencedirect.com/science/article/pii/S0167739X17308993.

[10] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, Emanuele Della Valle, and Michael Grossniklaus. C-sparql: a continuous query language for rdf data streams. *International Journal of Semantic Computing*, 04(01):3–25, March 2010. ISSN 1793-351X. doi: 10.1142/S1793351X10000936. URL https://www.worldscientific.com/doi/10.1142/S1793351X10000936. Publisher: World Scientific Publishing Co.

[11] Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, Stephen Farrell, and David Cooper. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Request for Comments RFC 5280, Internet Engineering Task Force, May 2008. URL https://datatracker.ietf.org/doc/rfc5280. Num Pages: 151.

[12] Calin Boje, Annie Guerriero, Sylvain Kubicki, and Yacine Rezgui. Towards a semantic Construction Digital Twin: Directions for future research. *Automation in Construction*, 114:103179, June 2020. ISSN 0926-5805. doi: 10.1016/j.autcon.2020.103179. URL https://www.sciencedirect.com/science/article/pii/S0926580519314785.

[13] Ramaswamy Chandramouli and Zack Butcher. Building secure microservices-based applications using service-mesh architecture. Technical Report NIST SP 800-204A, National Institute of Standards and Technology, Gaithersburg, MD, May 2020. URL https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-204A.pdf.

[14] Kang-Tsung Chang. Geographic Information System. In *International Encyclopedia of Geography*, pages 1–10. John Wiley & Sons, Ltd, 2019. ISBN 978-1-118-78635-2. doi: 10.1002/9781118786352.wbieg0152.pub2. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118786352.wbieg0152.pub2. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118786352.wbieg0152.pub2.

[15] Larry Conklin, Victoria Drake, and Sven Strittmatter. Threat Modeling Process | OWASP Foundation. URL https://owasp.org/www-community/Threat_Modeling_Process.

[16] Open Geospation Consortium. CityGML | OGC. URL https://www.ogc.org/standards/citygml.

[17] Luis Cruz-Piris, Diego Rivera, Ivan Marsá-Maestre, Enrique De la Hoz, and Juan Velasco. Access Control Mechanism for IoT Environments Based on Modelling Communication Procedures as Resources. *Sensors (Basel, Switzerland)*, 18, March 2018. doi: 10.3390/s18030917.

[18] Zhamak Dehghani. Data Mesh Principles and Logical Architecture, . URL https://martinfowler.com/articles/data-mesh-principles.html.

[19] Zhamak Dehghani. How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh, . URL https://martinfowler.com/articles/data-monolith-to-mesh.html.

[20] Fan Deng, Zhenhua Yu, Wenjing Liu, Xiaoqing Luo, Yu Fu, Ben Qiang, Chaoyang Xu, and Zhiwu Li. An efficient policy evaluation engine for XACML policy management. *Information Sciences*, 547:1105–1121, February 2021. ISSN 0020-0255. doi: 10.1016/j.ins.2020.08.044. URL https://www.sciencedirect.com/science/article/pii/S0020025520308148.

[21] Tianhu Deng, Keren Zhang, and Zuo-Jun (Max) Shen. A systematic review of a digital twin city: A new pattern of urban governance toward smart cities. *Journal of Management Science and Engineering*, 6(2):125–134, June 2021. ISSN 2096-2320. doi: 10.1016/j.jmse.2021.03.003. URL https://www.sciencedirect.com/science/article/pii/S2096232021000238.

[22] Jasenka Dizdarević, Francisco Carpio, Admela Jukan, and Xavi Masip-Bruin. A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration. *ACM Computing Surveys*, 51(6):116:1–116:29, January 2019. ISSN 0360-0300. doi: 10.1145/3292674. URL https://doi.org/10.1145/3292674.

[23] Lisa Ehrlinger and Wolfram Wöß. Towards a Definition of Knowledge Graphs.

[24] Jaume Ferré-Bigorra, Miquel Casals, and Marta Gangolells. The adoption of urban digital twins. *Cities*, 131:103905, December 2022. ISSN 0264-2751. doi: 10.1016/j.cities.2022.103905. URL https://www.sciencedirect.com/science/article/pii/S0264275122003444.

[25] International Organization for Standardization (ISO. ISO 16739-1:2018. URL https://www.iso.org/standard/70303.html.

[26] Shilpa Gite and Himanshu Agrawal. On Context Awareness for Multisensor Data Fusion in IoT. In Suresh Chandra Satapathy, K. Srujan Raju, Jyotsna Kumar Mandal, and Vikrant Bhateja, editors, *Proceedings of the Second International Conference on Computer and Communication Technologies*, Advances in Intelligent Systems and Computing, pages 85–93, New Delhi, 2016. Springer India. ISBN 978-81-322-2526-3. doi: 10.1007/978-81-322-2526-3_10.

[27] Feng Hao. J-PAKE: Password-Authenticated Key Exchange by Juggling. Request for Comments RFC 8236, Internet Engineering Task Force, September 2017. URL https://datatracker.ietf.org/doc/rfc8236. Num Pages: 15.

[28] Dick Hardt. The OAuth 2.0 Authorization Framework. Request for Comments RFC 6749, Internet Engineering Task Force, October 2012. URL https://datatracker.ietf.org/doc/rfc6749. Num Pages: 76.

[29] Jetmir Haxhibeqiri, Eli De Poorter, Ingrid Moerman, and Jeroen Hoebeke. A Survey of LoRaWAN for IoT: From Technology to Application. *Sensors*, 18(11):3995, November 2018. ISSN 1424-8220. doi: 10.3390/s18113995. URL https://www.mdpi.com/1424-8220/18/11/3995. Number: 11 Publisher: Multidisciplinary Digital Publishing Institute.

[30] Stefan Herle, Ralf Becker, Raymond Wollenberg, and Jörg Blankenbach. GIM and BIM: How to Obtain Interoperability Between Geospatial and Building Information Modelling? *PFG – Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 88(1):33–42, February 2020. ISSN 2512-2789, 2512-2819. doi: 10.1007/s41064-020-00090-4. URL http://link.springer.com/10.1007/s41064-020-00090-4.

[31] David Holmes, Maria Papathanasaki, Leandros Maglaras, Mohamed Amine Ferrag, Surya Nepal, and Helge Janicke. Digital Twins and Cyber Security – solution or challenge? In *2021 6th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*, pages 1–8, September 2021. doi: 10.1109/SEEDA-CECNSM53056.2021.9566277.

[32] A.-H. Hor and G. Sohn. Design and evaluation of a BIM-GIS integrated information model using RDF graph database. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VIII-4/W2-2021:175–182, October 2021. ISSN 2194-9050. doi: 10.5194/isprs-annals-VIII-4-W2-2021-175-2021. URL https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/VIII-4-W2-2021/175/2021/.

[33] A-H. Hor, A. Jadidi, and G. Sohn. BIM-GIS integrated geospatial information model using semantic web and RDF graphs. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, III-4:73–79, June 2016. ISSN 2194-9050. doi: 10.5194/isprsannals-III-4-73-2016. URL http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/III-4/73/2016/isprs-annals-III-4-73-2016.pdf.

[34] Joint Task Force Transformation Initiative. Guide for conducting risk assessments. Technical Report NIST SP 800-30r1, National Institute of Standards and Technology, Gaithersburg, MD, 2012. URL https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf. Edition: 0.

[35] Klementina Josifovska, Enes Yigitbas, and Gregor Engels. Reference Framework for Digital Twins within Cyber-Physical Systems. In *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*, pages 25–31, May 2019. doi: 10.1109/SEsCPS.2019.00012.

[36] Hakan Kayan, Matthew Nunes, Omer Rana, Pete Burnap, and Charith Perera. Cybersecurity of Industrial Cyber-Physical Systems: A Review, January 2021. URL http://arxiv.org/abs/2101.03564. arXiv:2101.03564 [cs].

[37] kexugit. Uncover Security Design Flaws Using The STRIDE Approach, October 2019. URL https://learn.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach.

[38] Yongdae Kim, Adrian Perrig, and Gene Tsudik. Tree-based group key agreement. *ACM Transactions on Information and System Security*, 7(1):60–96, February 2004. ISSN 1094-9224. doi: 10.1145/984334.984337. URL https://doi.org/10.1145/984334.984337.

[39] Werner Kritzinger, Matthias Karner, Georg Traar, Jan Henjes, and Wilfried Sihn. Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, January 2018. ISSN 2405-8963. doi: 10.1016/j.ifacol.2018.08.474. URL https://www.sciencedirect.com/science/article/pii/S2405896318316021.

[40] Ben Laurie, Adam Langley, and Emilia Kasper. Certificate Transparency. Request for Comments RFC 6962, Internet Engineering Task Force, June 2013. URL https://datatracker.ietf.org/doc/rfc6962. Num Pages: 27.

[41] Ben Laurie, Adam Langley, Emilia Kasper, Eran Messeri, and Rob Stradling. Certificate Transparency Version 2.0. Request for Comments RFC 9162, Internet Engineering Task Force, December 2021. URL https://datatracker.ietf.org/doc/rfc9162. Num Pages: 53.

[42] Jung Hoon Lee, Marguerite Gong Hancock, and Mei-Chih Hu. Towards an effective framework for building smart cities: Lessons from Seoul and San Francisco. *Technological Forecasting and Social Change*, 89:80–99, November 2014. ISSN 0040-1625. doi: 10.1016/j.techfore.2013.08.033. URL https://www.sciencedirect.com/science/article/pii/S0040162513002187.

[43] Ville V. Lehtola, Mila Koeva, Sander Oude Elberink, Paulo Raposo, Juho-Pekka Virtanen, Faridaddin Vahdatikhaki, and Simone Borsci. Digital twin of a city: Review of technology serving city needs. *International Journal of Applied Earth Observation and Geoinformation*, 114:102915, November 2022. ISSN 1569-8432. doi: 10.1016/j.jag.2022.102915. URL https://www.sciencedirect.com/science/article/pii/S1569843222001169.

[44] Wubin Li, Yves Lemieux, Jing Gao, Zhuofeng Zhao, and Yanbo Han. Service Mesh: Challenges, State of the Art, and Future Research Opportunities. In *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)*, pages 122–1225, April 2019. doi: 10.1109/SOSE.2019.00026. ISSN: 2642-6587.

[45] Inês Araújo Machado, Carlos Costa, and Maribel Yasmina Santos. Data Mesh: Concepts and Principles of a Paradigm Shift in Data Architectures. *Procedia Computer Science*, 196:263–271, January 2022. ISSN 1877-0509. doi: 10.1016/j.procs.2021.12.013. URL https://www.sciencedirect.com/science/article/pii/S1877050921022365.

[46] E. Maler, M. Machulak, and J. Richer. Federated Authorization for User-Managed Access (UMA) 2.0, January 2018. URL https://docs.kantarainitiative.org/uma/wg/rec-oauth-uma-federated-authz-2.0.html.

[47] E. Maler, M. Machulak, and J. Richer. User-Managed Access (UMA) 2.0 Grant for OAuth 2.0 Authorization, January 2018. URL https://docs.kantarainitiative.org/uma/wg/rec-oauth-uma-grant-2.0.html.

[48] Frank Manola, Eric Miller, and Brian McBride. RDF 1.1 Primer, June 2014. URL https://www.w3.org/TR/rdf11-primer/.

[49] Michael H. Mealling. A URN Namespace of Object Identifiers. Request for Comments RFC 3061, Internet Engineering Task Force, February 2001. URL https://datatracker.ietf.org/doc/rfc3061. Num Pages: 6.

[50] Albert Meijer and Manuel Pedro Rodríguez Bolívar. Governing the smart city: a review of the literature on smart urban governance. *International Review of Administrative Sciences*, 82, April 2015. doi: 10.1177/0020852314564308.

[51] Bettina Melzer. Reference Architectural Model Industrie 4.0 (RAMI 4.0), 2019. URL https://ec.europa.eu/futurium/en/system/files/ged/a2-schweichhart-reference_architectural_model_industrie_4.0_rami_4.0.pdf.

[52] Davy Preuveneers and Wouter Joosen. Towards Multi-party Policy-based Access Control in Federations of Cloud and Edge Microservices. In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 29–38, June 2019. doi: 10.1109/EuroSPW.2019.00010.

[53] Stefan Profanter, Ayhun Tekat, Kirill Dorofeev, Markus Rickert, and Alois Knoll. OPC UA versus ROS, DDS, and MQTT: Performance Evaluation of Industry 4.0 Protocols. In *2019 IEEE International Conference on Industrial Technology (ICIT)*, pages 955–962, February 2019. doi: 10.1109/ICIT.2019.8755050. ISSN: 2643-2978.

[54] Adil Rasheed, Omer San, and Trond Kvamsdal. Digital Twin: Values, Challenges and Enablers From a Modeling Perspective. *IEEE Access*, 8:21980–22012, 2020. ISSN 2169-3536. doi: 10.1109/ACCESS.2020.2970143. Conference Name: IEEE Access.

[55] Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. Request for Comments RFC 8446, Internet Engineering Task Force, August 2018. URL https://datatracker.ietf.org/doc/rfc8446. Num Pages: 160.

[56] Alejandro Rodriguez, Robert Mcgrath, Yong Liu, and James Myers. Semantic Management of Streaming Data. 522, January 2009.

[57] Zaid O. Saeed, Francesco Mancini, Tanja Glusac, and Parisa Izadpanahi. Future City, Digital Twinning and the Urban Realm: A Systematic Literature Review. *Buildings*, 12(5):685, May 2022. ISSN 2075-5309. doi: 10.3390/buildings12050685. URL https://www.mdpi.com/2075-5309/12/5/685. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute.

[58] Stefan Santesson, Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Carlisle Adams. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. Request for Comments RFC 6960, Internet Engineering Task Force, June 2013. URL https://datatracker.ietf.org/doc/rfc6960. Num Pages: 41.

[59] Gernot Steindl and Wolfgang Kastner. Semantic Microservice Framework for Digital Twins. *Applied Sciences*, 11(12):5633, January 2021. ISSN 2076-3417. doi: 10.3390/app11125633. URL https://www.mdpi.com/2076-3417/11/12/5633. Number: 12 Publisher: Multidisciplinary Digital Publishing Institute.

[60] Shu Tang, Dennis R. Shelden, Charles M. Eastman, Pardis Pishdad-Bozorgi, and Xinghua Gao. A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. *Automation in Construction*, 101:127–139, May 2019. ISSN 0926-5805. doi: 10.1016/j.autcon.2019.01.020. URL https://www.sciencedirect.com/science/article/pii/S0926580518305764.

[61] Fei Tao, Meng Zhang, and Andrew Nee. Five-Dimension Digital Twin Modeling and Its Key Technologies. pages 63–81. January 2019. ISBN 978-0-12-817630-6. doi: 10.1016/B978-0-12-817630-6.00003-5.

[62] G. A. van Nederveen and F. P. Tolman. Modelling multiple views on buildings. *Automation in Construction*, 1(3):215–224, December 1992. ISSN 0926-5805. doi: 10.1016/0926-5805(92)90014-B. URL https://www.sciencedirect.com/science/article/pii/092658059290014B.

[63] Eric VanDerHorn and Sankaran Mahadevan. Digital Twin: Generalization, characterization and implementation. *Decision Support Systems*, 145:113524, June 2021. ISSN 0167-9236. doi: 10.1016/j.dss.2021.113524. URL https://www.sciencedirect.com/science/article/pii/S0167923621000348.

[64] Ashkan Yousefpour, Caleb Fung, Tam Nguyen, Krishna Kadiyala, Fatemeh Jalali, Amirreza Niakanlahiji, Jian Kong, and Jason P. Jue. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, 98:289–330, September 2019. ISSN 1383-7621. doi: 10.1016/j.sysarc.2019.02.009. URL https://www.sciencedirect.com/science/article/pii/S1383762118306349.

[65] Fu Zhang, Zhiyin Li, Dunhong Peng, and Jingwei Cheng. RDF for temporal data management – a survey. *Earth Science Informatics*, 14(2):563–599, June 2021. ISSN 1865-0481. doi: 10.1007/s12145-021-00574-w. URL https://doi.org/10.1007/s12145-021-00574-w.

[66] Chuanjun Zheng, Jingfeng Yuan, Lei Zhu, Yajing Zhang, and Qiuhu Shao. From digital to sustainable: A scientometric review of smart city literature between 1990 and 2019. *Journal of Cleaner Production*, 258:120689, June 2020. ISSN 0959-6526. doi: 10.1016/j.jclepro.2020.120689. URL https://www.sciencedirect.com/science/article/pii/S0959652620307368.

[67] Yu Zheng, Sen Yang, and Huanchong Cheng. An application framework of digital twin and its case study. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1141–1153, March 2019. ISSN 1868-5145. doi: 10.1007/s12652-018-0911-3. URL https://doi.org/10.1007/s12652-018-0911-3.