Marie Sklodowska Curie, Research and Innovation Staff Exchange (RISE)

IdeNtity verifiCatiOn with privacy-preservinG credeNtIals for anonymous access To Online services

WP3 – Qualified Anonymity
D3.1 “Federated anonymous credentials”

Editor(s):	CUT
Author(s):	Vaios Bolgouras (UPRC - secondee), Nikos Passas (UPRC - outside secondment), Christos Xenakis (UPRC - outside secondment), Markos Charalambous (CUT - secondee), Kostantions Papadamou (CUT - outside secondment), Theodoros Christophides (CUT - secondee), Ioana Stroinea (CSGN - secondee), Scurtu Petru (CSGN - outside secondment), Orfeas Theofanis (LST - secondee), Nikolaos Salamanos (CUT - outside secondment)
Dissemination Level:	Public
Nature:	Report
Version:	2.0

INCOGNITO is funded by the European Commission’s Horizon 2020 Research and Innovation Framework program under the Marie Skłodowska-Curie Research and Innovation Staff Exchanges Action, Grant Agreement no 824015. The content of this deliverable reflects only the views of the project owner. The European Agency / Commision is not responsible for any use that may be made of the information it contains.

Project Profile

Contract Number	824015
Acronym	INCOGNITO
Title	IdeNtity verifiCatiOn with privacy-preservinG credeNtIals for anonymous access To Online services
Start Date	Jan 1st, 2019
Duration	48 Months

Partners

Τεχνολογικό Πανεπιστήμιο Κύπρου	TECHNOLOGIKO PANEPISTIMIO KYPROU (BEN2)	CUT	Cyprus
University of Piraeus	UNIVERSITY OF PIRAEUS RESEARCH CENTRE (BEN1)	UPRC	Greece
certSIGN	CERTSIGN SA (BEN3)	CSGN	Romania
Telefónica Telefónica Investigación y Desarrollo	TELEFONICA INVESTIGACION Y DESARROLLO SA (BEN6)	TID	Spain
LS TECH	LSTECH ESPANA SL (BEN4)	LST	Spain
FOGUS INNOVATIONS & SERVICES	FOGUS INNOVATIONS & SERVICES P.C (BEN7)	FOGUS	Greece

Document History

Version	Date	Author	Remarks
0.1	16/10/2020	Markos Charalambous, Kostantinos Papadamou, Vaios Bolgouras	Executive Summary and Table of Contents
0.2	16/10/2020	Kostantinos Papadamou (CUT), Ioana Stroinea (CSGN), Vaios Bolgouras (UPRC)	Introduction
0.3	01/11/2020	Ioana Stroinea (CSGN), Surtu Petru (CSGN), Vaios Bolgouras (UPRC), Orfeas Theofanis (LST), Theodoros Christophides (CUT)	Qualified Anonymity in INCOGNITO Architecture
0.4	13/11/2020	Surtu Petru (CSGN), Vaios Bolgouras (UPRC), Orfeas Theofanis (LST), Theodoros Christophides (CUT)	Revision of Qualified Anonymity Architecture
0.5	18/11/2020	Ioana Stroinea (CSGN), Kostantinos Papadamou (CUT), Nikoloas Salamanos (CUT)	Federated Authentication Architecture
0.6	28/11/2020	Ioana Stroinea (CUT), Kostantinos Papadamou (CUT), Markos Charalambous (CUT)	Revision of Federated Authentication Architecture
0.7	01/12/2020	Scurtu Petru (CSGN), Vaios Bolgouras (UPRC), Angeliki Panou (UPRC)	Component Implementation for Qualified Anonymity Support
0.8	15/12/2020	Markos Charalambous (CUT), Theodoros Christophides (CUT), Nikolaos Salamanos (CUT), Orfeas Theofanis (LST)	Qualified Anonymity Application Scenarios
0.9	23/12/2020	Kostantinos Papadamou (CUT), Ioana Stroinea, Vaios Bolgouras (UPRC), Michael Sirivianos (CUT), Christos Xenakis (UPRC)	Document Review
1.0	27/12/2020	Nikos Passas (UPRC)	Document Submission
2.0	22/01/2021	Nikos Passas (UPRC)	Edited based on comments from the EU

Deliverable dates

Delivery	**Date**
Contract delivery due date	31/12/2020 (M24)
Actual delivery date of version n.1	31/12/2020 (M24)
Actual delivery date of version n.1	22/01/2021 (M25)

Fellow ID	**Name/Surname**	**Researcher category**	**Declaration No.**	**PM**
4	Theodoros Christophides	ER	2	4
6	Markos Charalambous	ESR	3	2
7	Orfeas Theofanis	ESR	5	5
16	Vaios Bolgouras	ESR	13	4
11	Ioana Stroinea	ESR	9, 18	2

Executive Summary

The current deliverable documents the integration of anonymous credentials with federated identities in the INCOGNITO platform and is part of the Work Package 3. The development of the project is taking place under a partnership that involves both academic and industrial expertise. Considering the interdisciplinary knowledge exchange, the consortium involved in the INCOGNITO body aims to create a platform that leverages the state-of-the-art technologies so that users may securely, privately and effortlessly access online services, as well as to add features that guide users through the registration and authentication processes without disclosing their identity such as an AI-based assistant.

D3.1 tackles the objectives of Task 3.1 via the scientific research activities reported in this deliverable. More specifically, the purpose of this report is to specify techniques and solutions that provide support for the use of qualified anonymous credentials for authentication and authorization of the users through the use of pseudonyms and techniques to ensure unlinkability and untraceability of the activities of the users. On the other hand, it will develop the appropriate components devised to provide support of qualified anonymous credentials on the online services side by means of the definition and verification of adequate ABAC policies. In addition, this we will take care of integrating the proposed anonymous credentials with existing device-centric authentication solutions, such as FIDO.

Please note that hereinafter, mentions of “INCOGNITO” refer to the platform, unless stated otherwise (for example “INCOGNITO body/consortium/users etc.”).

Table of Contents

Table of Figures

Table of Abbreviations

ABAC	Attribute Based Access Control
API	Application Programming Interface
BLE	Bluetooth Low Energy
CTAP	Client to Authenticator Protocol
ID	Identity
IDC	Identity Consolidator
Idemix	Identity Mixer
IdP	Identity Provider
FIDO	Fast IDentity Online
NFC	Near-Field Communication
MITM	Man in the Middle
LoA	Level of Assurance
OAuth	Open Authentication
OIDC	OpenID Connect
OS	Operating System
OSN	Online Social Network
PABAC	Privacy-Preserving Attribute Based Access Control
PIN	Personal Identification Number
QR	Quick Response
REST	Representational State Transfer
SP	Service Provider
SSO	Single-Sign On
TEE	Trusted Execution Environment
TLS	Transport Layer Security
TPM	Trusted Platform Module
UD	User Device
UI	User Interface
UMA	User-managed Access
UX	User Experience
W3C	World-wide Web Consortium
WebAuthn	Web Autehntication

1 Introduction

This document describes the design and implementation of the corresponding INCOGNITO components and their respective modules that will enable the integration of qualified anonymous credentials with federated identities and protocols. In order to achieve the fine-grained treatment of identity attributes, combined with privacy-preserving traits, cryptographic credential implementations such as Idemix along with Keycloak's User-Managed Access implementation and the FIDO2 protocol are utilized and have a vital role in successfully adopting **Qualified Anonymity** concept at the INCOGNITO platform.

First, the Idemix cryptographic credentials protocol is leveraged and deployed on the Identity Consolidator or any other trusted Identity Provider within the INCOGNITO architecture and when integrated together with federated authentication protocols we are able to provide qualified anonymity support for Service Providers. Next, leveraging the User-managed Access (UMA) protocol we are able to provide fine-grained control over the user's identity attributes and cryptographic credentials by the means of the definition and verification of adequate Attribute-based Access Control (ABAC) policies. Last, by leveraging the FIDO2 protocol we are able to provide a secure framework for device-centric authentication that also enhances the security of the cryptographic credentials stored on the mobile device of the user.

These technologies are integrated across the involved parties, i.e., User device, Identity Consolidator, and Service Providers, according to the corresponding needs that arise for such an implementation and all the involved components in our architecture are integrated and able to communicate together over the OpenID Connect federated authentication protocol. Next, a detailed description of the aforementioned technologies, as well as a description of the implementation and the integration between each other is provided.

1.1 Technologies

1.1.1 Idemix

Idemix provides an anonymous credentials infrastructure that is employed to achieve both strong authentication and privacy. Credentials do not refer just to information regarding e.g., username and password to access an account, but also tokens and certificates that makes it possible for users to prove attributes about themselves to a third party. This is a common procedure followed especially when performing access control request procedures. For example, an identification document can be used as a credential to prove that one is over 18 years old, an identity attribute that is required to buy alcohol. The problem is that by providing this identification document, not just the information related to the customer’s age is revealed, but also a plethora of data that is not necessary for the purpose of this transaction.

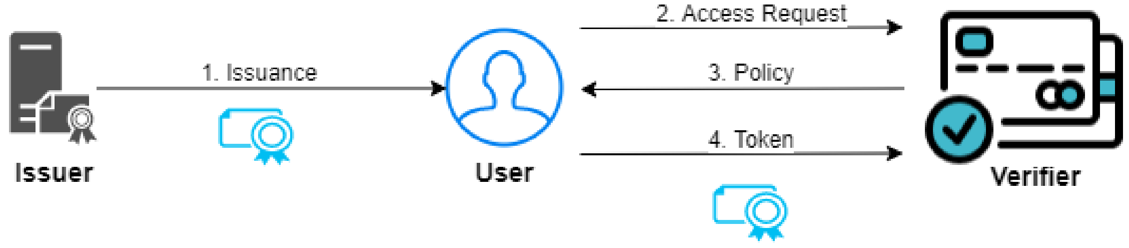


Figure 1: Idemix operation

The use of anonymous credentials tackles exactly this problem, as it allows the users to choose the attributes they need to disclose from a credential (i.e. the identification document in the aforementioned example), and not reveal all the included information. This essential function for ensuring user privacy can be achieved by utilizing Idemix. In Figure *1* we can see an overview of the steps followed by using this technology in order to ensure users’ privacy. Idemix provides an Issuer that creates such credentials for the users to communicate with the Service Providers whenever needed, along with the Verification endpoint of course. These credentials may refer only to the information that a user wishes to disclose, a process that can be repeated numerous times and maintain unlinkability between the issued credentials.

1.1.2 FIDO2

One of the latest technological advancements, FIDO2, makes it possible for the users to seamlessly authenticate and connect to inline services, regardless of the platform they are using. For each website and service that the users are utilizing, FIDO2 uses unique cryptographic login credentials that reside on the users’ device, not a central server, thus avoiding a single point of failure and eliminating the need to trust a third party. By employing such practices, the security level is dramatically increased as threats related to phising emails, password theft and credentials replay attacks are mitigated. The credentials are easily accessible by users through the utilization of built-in functionalities on their devices, such as fingerprint readers or face recognition, as well as with the use of specific FIDO security keys. Another advantage that the keys’ uniqueness offers by being used only in a specific website each, is that the user’s privacy is ensured as a specific credential cannot be used to track their movements, while at the same time the biometric data is safely stored on the device and not shared with anyone else. From the Service Providers’ perspective, the adoption of the FIDO2 technology is simple, as only a a JavaScript API call supported by all browsers and devices is needed.

1.1.3 UMA

UMA 2.0 is a new federated authorization standard protocol that is built on top of OAuth 2.0, which is the industry-standard protocol for authorization. The utilization of UMA 2.0 provides the user with the ability to manage their personal data that is or will potentially be used by online services, regardless of their location. A fine-grained control of their identity attributes is possible through the adoption of the UMA components by the participating entities. More specifically, the roles defined at the UMA implementations are as follows:

- Resource Owner: Owns resources that are protected, and grants access if needed to other parties that require it.

- Requesting Party: An entity that requires access to the protected resources of the aforementioned Resource Owner. The requesting party utilizes a client for this purpose.
- Client: Request to access the protected resources are made through an application. It is used by the requesting party and has the owner’s authorization.
- Resource Server: A repository where the protected resources reside.
- Authorization Server: Protects the resources that reside on the Resource Server.

1.2 Qualified Anonymity Components

Qualified anonymity is primarily based on the use of anonymous credentials provided by Idemix. For the realization of this solution, three pillar components are taken into account: The Issuer of the anonymous credentials, the corresponding Verifier and the User who controls the process from the corresponding device.

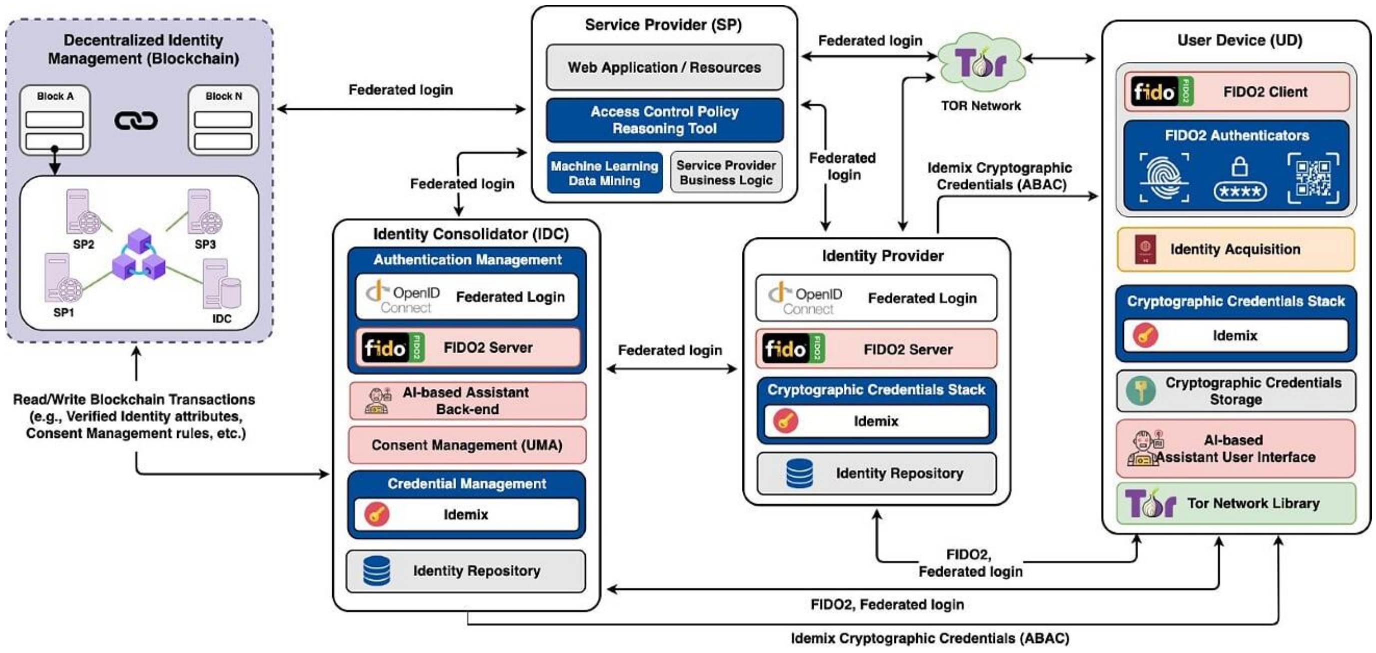
In INCOGNITO, users will use qualified anonymity in order to authenticate in an anonymous manner to Service Providers. The **Issuer** component, upon user request, provides the corresponding identity attributes in a form of a digital certificate. This component can be integrated in the Identity Consolidator component or be externalized to an external component in the INCOGNITO architecture. For enhanced security, an integrated approach is considered. The end user will authenticate to the Identity Consolidator using FIDO2 authenticators and the server will forward the issuing request to the Issuer component.

Each issuer is in possession of a private key, which is kept secret and is used only when issuing credentials in the form of a certificate. A corresponding public key is accessible to the Verifiers which can attest the originality of the issued credentials. the verifiers can check the validity of this proof of knowledge (issued certificate/anonymous credential) using the issuer's public key that corresponds with the private key with which the issuer signed the attributes. Following the verification process, the Verifier can attest that the user has received a credential corresponding to the disclosed attributes and can trust the authenticity of the attributes. It is worth mentioning, that the disclosure process does not include a full copy of the signature over the attributes. Thus, it is not possible for the verifier to use the received attributes and proof of knowledge to disclose these attributes.

The user performs requests to the issuer in order to be provided with credentials according to the requirements set by the online service providers. Upon receipt of the credentials, they are safely stored on the **User device**. When authenticating, the user will initialize a disclosure session for the requested credentials. The disclosed attributes will be used by an Identity Provider to generate access tokens for the Service Provider.

1.3 Qualified Anonymity in INCOGNITO Architecture

The INCOGNITO platform incorporates the aforementioned technologies taking advantage of the existing participating entities. The IDC, IDP, SP and UD have explicit roles in the effort made to preserve user privacy. In Figure 2 we observe the architecture diagram with the building blocks required for qualified anonymity to be achieved, and how they interact with each other.


Decentralized Identity Management (Blockchain)
Block A
Block N
SP2
SP3
SP1
IDC
Read/Write Blockchain Transactions (e.g., Verified Identity attributes, Consent Management rules, etc.)
Federated login
Service Provider (SP)
Web Application / Resources
Access Control Policy Reasoning Tool
Machine Learning Data Mining
Service Provider Business Logic
Federated login
TOR Network
Federated login
Idemix Cryptographic Credentials (ABAC)
Identity Consolidator (IDC)
Authentication Management
OpenID Connect
Federated Login
FIDO2 Server
AI-based Assistant Back-end
Consent Management (UMA)
Credential Management
Idemix
Identity Repository
Federated login
Identity Provider
OpenID Connect
Federated Login
FIDO2 Server
Cryptographic Credentials Stack
Idemix
Identity Repository
User Device (UD)
FIDO2 Client
FIDO2 Authenticators
Identity Acquisition
Cryptographic Credentials Stack
Idemix
Cryptographic Credentials Storage
AI-based Assistant User Interface
Tor Network Library
FIDO2, Federated login
FIDO2, Federated login
Idemix Cryptographic Credentials (ABAC)
Figure 2: INCOGNITO architecture

More details about the role each of these entities have at the INCOGNITO platform, are presented below.

1.3.1 User Device

The main gateway to the architecture and the component that initializes anonymous credentials issuance is the user device. It has the following responsibilities:

- Receives anonymous credentials from the Issuer.
- Stores anonymous credentials in a secure manner.
- Makes disclose sessions for anonymous credentials.

The anonymous credentials generated by the Issuer must be stored in a secure manner. Once the credentials are created, they cannot be revoked and are never to be disclosed. A disclosure session will imply only a proof by the owner of the credentials, not disclosing the actual credentials.

The anonymous credentials will be protected by using cryptographic mechanisms. They will be stored in a secure fashion and will be encrypted by using private keys under the sole control of the owner of the device. Access to the private keys will be granted by means of a secure PIN or biometric authenticators.

1.3.2 Identity Consolidator

The Identity Consolidator is the central component of the INCOGNITO architecture. The end user, through the means of a mobile device, will authenticate to the Consolidator and will be able to manage his online identity. The attributes chosen to be disclosed will be centralized on the

Consolidator and with the help of the UX Assistant the user will be able to view and manage these attributes. The user will authenticate to the Identity Consolidator using FIDO2 authenticators. After successful authentication, it can request Idemix credentials generation. The Consolidator will forward the authenticated request to the Verifier, which will generate the requested credentials.

1.3.3 Service Provider

The Service Provider is the entity responsible for granting access to the resources or services desired by the end user. Access is granted based on a per resource policy, which requests certain attributes to be validated. The Service Provider will request the needed attributes from the end user. If the end user wishes to be authenticated in an anonymous manner, it will contact the Idemix Issuer which will issue anonymous credentials. These credentials will be later used by the Identity Provider which will grant the user with an access token for the requested resources. The Service Provider trusts the Identity Provider and is responsible for validating the access tokens.

1.3.4 Identity Provider

The Identity Provider is a trust anchor in the architecture. The main goals of this component are:

- Issue access tokens for the end user. The issued tokens will grant access to a resource protected by the Service Provider.
- Validate the Idemix anonymous credentials. The issued credentials will be validated with the help of a Verifier.

1.3.5 Privacy-Preserving data flow

In this section we present how the entities involved in the INCOGNITO solution interact with each other, along with the steps followed during the procedure of a user being granted access to online resources.

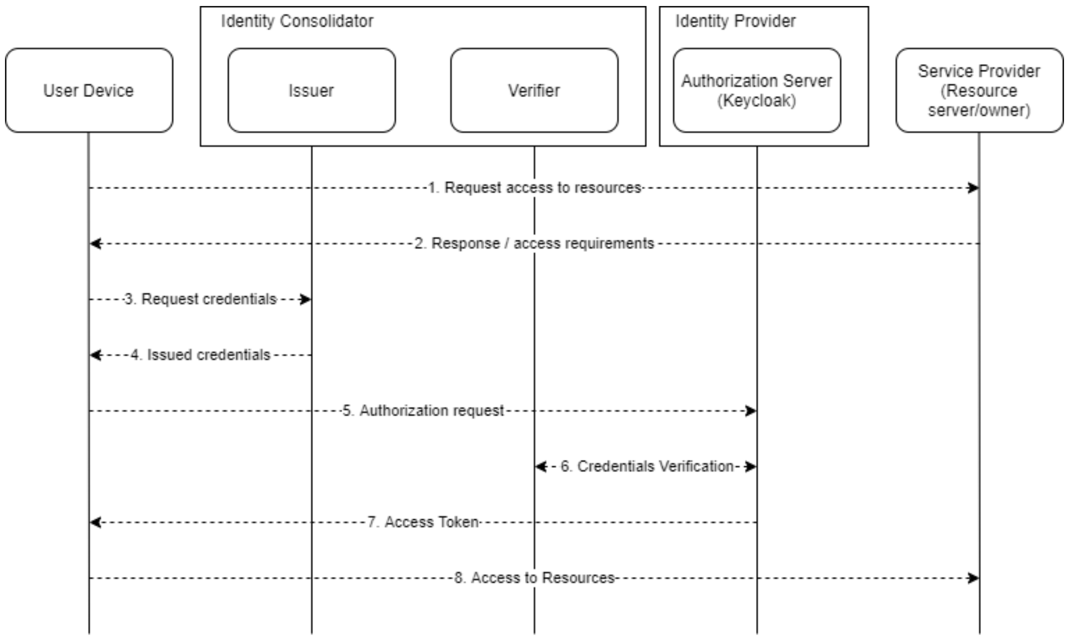


Figure 3: Steps towards QA

In Figure 3 we can observe step by step the process executed so the user can access anonymously and without revealing unnecessary identity information online resources that have restrictive permissions. More specifically:

1. The user device will send a request to a Service Provider or a Resource Owner asking for access to a particular resource.
2. The Service Provider or Resource Owner will answer with a set of access requirements which the user must fulfill in order to gain access to the resource.
3. If not already present in the user’s device, the device will ask the Idemix Issuer for the required credentials based on the attributes requested by the service provider
4. The Idemix Issuer will generate credentials based on the requested attributes
5. Using the generated Idemix credentials the user device will ask for access to a particular resource. The Service Provider will redirect the request to the Identity Provider for authorization.
6. The Identity Provider will verify the Idemix credentials by interrogating the Idemix Verifier.
7. If the Idemix credentials are valid, the Identity Provider will issue an access token for the requested resources.
8. The client device will access the desired resources using the access token.

2 Federated Authentication Architecture

One of the core concepts where all the building blocks of the INCOGNITO platform are built upon is federated authentication. Federated authentication (a.k.a. Single Sign On (SSO)) enables users to authenticate using the same identity to a trusted Identity provider and access services online. At the same time, Service Providers are relieved from the burden of having to collect, verify and maintain the personal identity information of their users.

In INCOGNITO, we aim at integrating federated authentication protocols, such as OpenID Connect (OIDC) [1], with FIDO2 [2] and cryptographic credential protocols (i.e., Idemix [3]) in order to preserve the privacy of the user while offering a user-friendly device-centric and attribute-based authentication architecture that address all the problem of traditional Web Authentication schemes. The FIDO2 specification is a password-less solution that enables IdPs to authenticate end-users using strong authenticators (e.g., fingerprint) for user-to-device authentication and cryptographic protocols (e.g., RSA) for device-to-service authentication. By combining the concepts of strong authentication and cryptographic protocols for privacy preserving authentication along-side with the delegation of authentication to IdPs we allow for a more user-friendly and secure solution for end-users.

Furthermore, to use FIDO2 in conjunction with the OpenID Connect protocol, we need to modify the FIDO2 and the OIDC implementations so that the authentication between the end-user and the OIDC Provider takes places using the FIDO2 protocol. In this case, the end-user will have to generate a public/private key pair on his/her mobile device and register the public key on the FIDO2 server running on the OIDC Provider and every time the OIDC Provider needs to authenticate the user will involve the FIDO2 protocol to do so. Figure *4* depicts the proposed FIDO2-enhanced federated authentication process. Initially, when the user tries to authenticate with an SP (step 1), the SP delegates the authentication to the OIDC Provider along with a list of identity attributes that the SP requires (step 2). Then, the OIDC Provider requests from the user to authenticate using FIDO2 on his mobile device (e.g., using fingerprint). The OIDC Provider also requires from the user authorization to reveal to the SP the requested identity attributes (step 3). As soon as FIDO authentication is successful and the user has authorized the revelation of the requested attributes (step 4), the OIDC Provider informs the SP about the result of the authentication while also providing the requested attribute values (step 5). At the end, the SP grants to the user access to its resources (step 6).

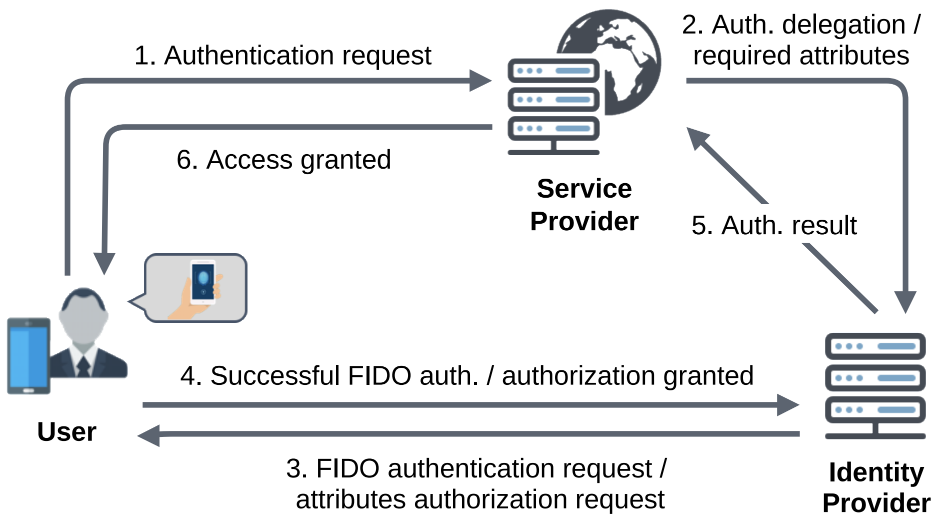

1. Authentication request
2. Auth. delegation / required attributes
6. Access granted
Service Provider
5. Auth. result
User
4. Successful FIDO auth. / authorization granted
3. FIDO authentication request / attributes authorization request
Identity Provider

Figure 4. FIDO2-enanced federated authentication process

2.1 Federated Authentication (OpenID Connect)

In INCOGNITO federated authentication is realized by utilizing the OpenID Connect (OIDC) protocol. OIDC is a simple identity layer built on top of the OAuth 2.0 protocol [4], which facilitates federated authentication. OIDC enables SPs to delegate the authentication of end-users to trusted OIDC Providers, as well as to obtain profile information about an end-user from the OIDC Providers in an interoperable REST-like manner.

In the context of the INCOGNITO platform, the OIDC Provider is the Identity Consolidator since it also holds and consolidates all the identity information of the end-users. For the OIDC Provider, we utilize Keycloak [5], which is an open-source implementation an Identity and Access Management solution. Among others, Keycloak implements the OpenID Connect protocol specification and can be used as an OIDC Provider. By deploying Keycloak on the Identity Consolidator we are able to convert easily implement an OIDC Provider while also utilizing protocols implemented by Keycloak such as UMA [6].

2.2 FIDO 2 Protocol Overview

FIDO2 Protocol Specification [2] is the newest specification released by the FIDO Alliance and it leverages common devices to enable users to authenticate to online services both from mobile and desktop environments. FIDO2 is comprised of two main specifications: Web Authentication

(WebAuthen) which was developed by the World Wide Web Consortium (W3C) and the FIDO Alliance’s new Client-to-Authenticator Protocol (CTAP) specification [7].

FIDO2 works on the principle of public key cryptography. Unlike password-based authentication where the client and server share a username and a password, in FIDO2 user devices generate a pair of public/private keys. The private key never leaves the user’s device which increases the security because the private key cannot be intercepted while being transmitted to the server or stolen from the server itself. During registration process, the user shares the public key with the server. From now on, whenever the user wants to authenticate with the server, the latter send a challenge that will be signed by the user’s device with the private key and sent back to the server. Using the public key, the server can validate the signature. Another layer of security is the requirement of a user’s presence before the private key is used to sign the challenge, such as a fingerprint or PIN. With FIDO2, a larger option of devices is available to be used for authentication. As compared to FIDO UAF where the user was using a platform authenticator or to FIDO U2F where the user used a roaming authenticator, FIDO2 enables users to use both platform and roaming authenticators such as mobile devices, USB tokens etc. [8].

- **WebAuthn Specification**

Web Authentication is a specification that was developed by the FIDO Alliance in collaboration with the World Wide Web Consortium and was declared an official web standard in March 2019. This specification describes and API that enables web applications to authenticate user through the means of public key cryptography.

There are three main components of this standard:
a. Web Authentication API
b. Relying Party
c. WebAuthn Authenticator Model

2.2.1 Web Authentication API

Web Authentication API is an API that allows the creation and utilization of public key credentials. It resides between the client side (also referred to as “script” in the official documentation) of the Relying Party and the browser. This API allows the script to request the browser to create new credentials for future use by the Relying Party. The request goes to the authenticator which in fact is the one to create the credentials and send them back to the Relying Party as objects. Since the information is encapsulated in objects, a layer of security is added because the script cannot access the information from the authenticator. During this server-authenticator communication, the information exchanges always contain the Relying Party ID to assure that the information is always sent and received from the correct Relying Party to which the user has their credentials registered.

The Web Authentication API implements two main interfaces that enable the creation and transfer of the credentials:

- ***PublicKeyCredential:*** Interface which inherits from *Credential* interface and contains the attributes that must be returned to the caller during registration and authentication to a Relying Party.
- ***AuthenticatorResponse:*** Interface which is used by the authenticator to respond to a Relying Party request.

2.2.2 Relying Party

The Relying Party contains a FIDO2 server that is in charge of registering and authenticating users. It stores the public key of the user and implements the logic of creating requests and processing the information received from the user device.

2.2.3 WebAuthn Authenticator Model

WebAuthn Authenticator Model describes an abstract model of the functionality that an authenticator should implement in order to be able to interact with the Web Authentication API. This model is not a ready-made implementation, but it is rather a client platform specific implementation that should result in the same result when used with the Web Authentication API. WebAuthn Authenticator Model describes data encoding and logical operations exposed to the client and the Relying Party.

2.2.4 Client-to-Authenticator Protocol

FIDO2 introduced a new version of the CTAP protocol defined in the FIDO U2F specification. Client to Authenticator Protocol 2 (CTAP2) is an application layer protocol meant to create a communication between a roaming authenticator and a client platform. There are three main components of CTAP2 Protocol:

- **Authenticator API** which is an abstract API that describes the way a client platform interacts with the authenticator.
- **Message Encoding** which is the encoding format used by a client platform before calling an API method. The authenticator will respond with a message encoding in the same format. The reason behind using CTAP2 canonical CBOR is represented by its light-weight form of encoding which is necessary to reduce the complexity of the messages since transport mediums can be bandwidth-constrained such as BLE. The JSON serialization used in the previous FIDO Specification is too heavy weight for the new way of transporting messages to and from roaming authenticators.
- **Transport-specific Binding** which is the format of communication specific to a certain transport protocol such as USB, NFC, BLE used to convey information between the roaming authenticator and the client platform.

2.3 FIDO 2 with Federated Authentication

FIDO2 Authentication protocol and Federated Authentication protocols complement each other while focusing on different things. FIDO2 protocol evolves in the direction of authentication, decoupling server-side authentication from the credential storage on the user device or from the transfer of attributes. Defining what are the attributes of a user is outside the scope of this protocol which acts to prove that a user is who them pretends to be without revealing their whole identity. On the other side, the federation protocols focus on securely transferring user attributes from one entity to another.

FIDO2 protocol is designed to protect user privacy and uses public key cryptography. During registration, the client devices generate a pair of keys which will be shared between the server and the client device. The public key is sent to the FIDO2 Server which stores it, while the private key resides on the user’s device. During authentication, the user only needs to prove that he has the private key by signing a challenge that is sent to the server and check using the public key. In order to unlock the private key, the user needs to prove their identity to the client device through actions such as swiping a finger, through facial recognition etc. These pieces of information about a user’s identity (fingerprint, voice etc.) will not leave the user device. Therefore, FIDO2 provides authentication without revealing user’s identity [9].

Federated authentication enables service providers to delegate authentication to an identity provider. The authentication is made by the identity provider and in exchange the service provider receives user’s attributes that are of interest for it. These authenticated attributes are sent to the relying party as an id_token which is a signed data structures encoded as a JWT (JSON Web Token). This token identifies the user, as well as the relying party to which the information is sent and the authorization server that issues the token [10].
INCOGNITO leverages both authentication protocols in order to enhance the security of the users, as well as offering a more versatile solution. It authenticates users to the Identity Provider without disclosing their biometrics and using OpenID Connect, the platform authenticates users to other relying parties just by disclosing the minimum number of necessary attributes. The interweaving of these authentication protocols offers the advantages of each solution and authenticates the user in a secure and private manner.

2.4 FIDO 2 Extensions for Federated Authentication

In order to achieve password-less authentication, INCOGNITO has to delegate this process to an external FIDO2 Server. Therefore, this chapter is dedicated to describing the extensions that were necessary to be developed at the Identity Provider in order to override the initial authentication process. INCOGNITO uses Red Hat’s Identity and Access Management solution as an OIDC provider. This implementation – namely Keycloak – has multiple authentication mechanisms but it is missing password-less solutions. As a matter of fact, after some research of the possible means of integration between FIDO2 Server and this Identity Provider, the INCOGNITO consortium decided upon a solution that complies with the security requirements of the project.

The federated authentication flow is one composed from the authentication to the FIDO2 Server, as well as from the OpenID Connect authentication of the user at the Identity Provider. When a user tries to get an access token in order to access other Service Providers’ services, the user will be prompted to first authenticate using FIDO2 Client. The flows have ten steps as seen in the picture below:

1. Keycloak Authentication Request
2. Authentication Request
3. Set lastFidoAuthenticationId
4. Successful Authentication Response
5. Delegate authentication
6. Get lastFidoAuthenticationId
7. Return lastFidoauthenticationId
8. Verify the validity of lastfidoAuthenticationId
9. Successful authentication
10. Return Access token

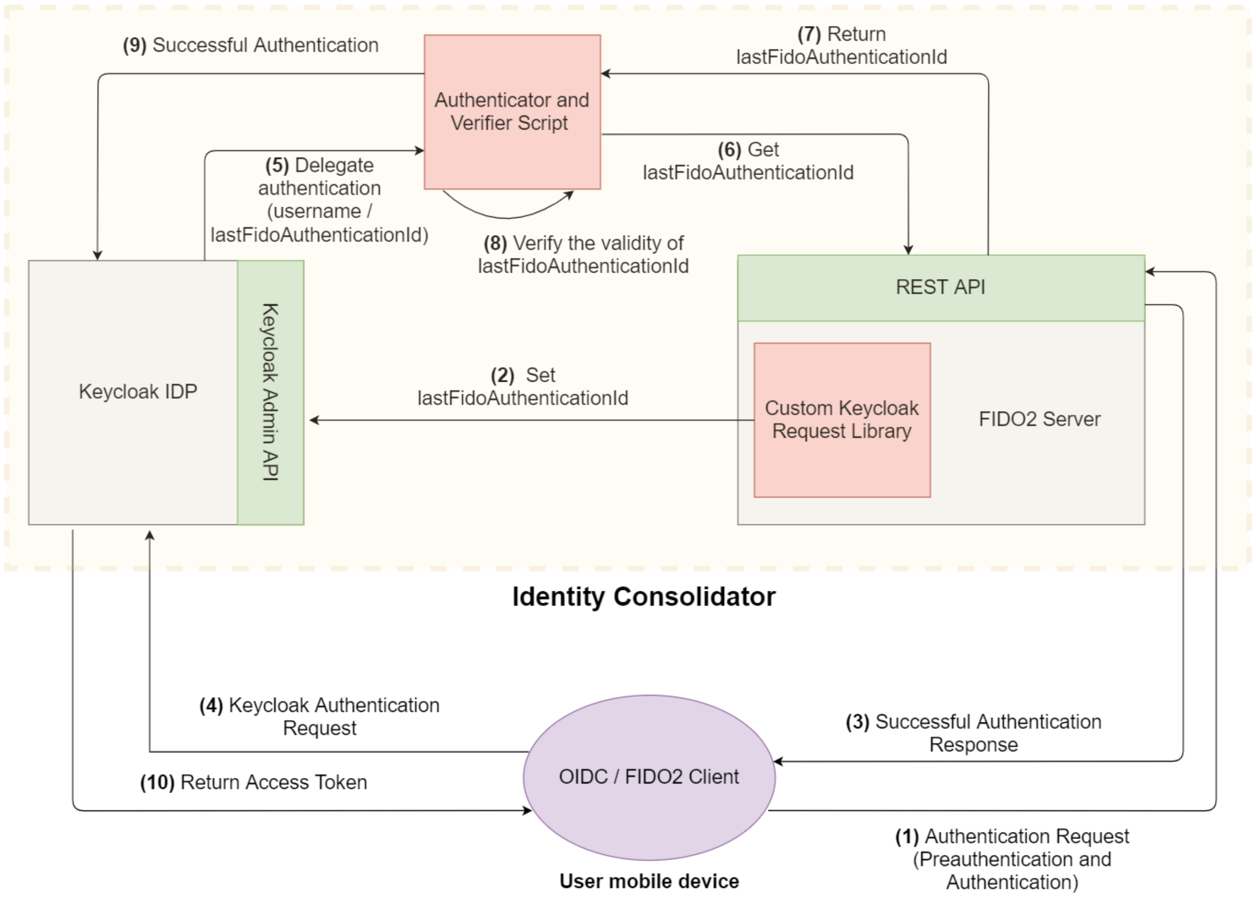

(9) Successful Authentication
Authenticator and Verifier Script
(7) Return lastFidoAuthenticationId
(6) Get lastFidoAuthenticationId
(5) Delegate authentication (username / lastFidoAuthenticationId)
(8) Verify the validity of lastFidoAuthenticationId
REST API
Keycloak IDP
Keycloak Admin API
(2) Set lastFidoAuthenticationId
Custom Keycloak Request Library
FIDO2 Server
Identity Consolidator
(4) Keycloak Authentication Request
(3) Successful Authentication Response
OIDC / FIDO2 Client
(10) Return Access Token
(1) Authentication Request (Preauthentication and Authentication)
User mobile device

Figure 5. FIDO2 and Keycloak extensions for federated authentication using FIDO2

Keycloak Authentication Response: The INCOGNITO application on the user’s device incorporates an OpenID Connect Client that is used to perform an OIDC authentication at the Identity Provider in order to get an access token for the access of the Service Providers that trust the IDP. Whenever a user wants to access some services, it has to make this request.

Authentication Request: When a user makes the request for the OIDC access token, the INCOGNITO application is immediately triggering a FIDO2 authentication by calling the FIDO2

Client running on the user’s device. This client will send an authentication request to the FIDO2 Server.

Set *lastFidoAuthenticationId*: This is a step triggered by the FIDO2 Server. When a user is successfully authenticated at the FIDO2 Server, before sending a success response to the user, the server sets a variable in the Identity Provider which represents the last authentication id of the user. This is a unique value which changes with each authentication.

Successful Authentication Response: This is a status response for the authentication made by the FIDO2 Server. This response is returned to the FIDO2 Client running on the user’s device.

Delegate authentication: When the user sends an OIDC request to the Identity Provider for an access token, the IDP delegates the authentication to the FIDO2 Server. An authentication and validation script brings the last authentication id of the user at the FIDO2 Server as well as taking the value stored in Keycloak and validates they have the same value. If the response is positive, it means that the last authentication id is still valid and therefore announces Keycloak that it can proceed further with the OIDC authentication process.

Get *lastFidoauthenticationId*: This is the call the authenticator and verifier script make to a FIDO2 Server endpoint that exposes the authentication id of the user in order to check its value against the one stored at the IDP.

Return *lastFidoauthenticationId*: The FIDO2 Serve returns to the authenticator and verifier script the value of the last authentication of the user that makes the OIDC request.

Verify the validity of *lastfidoAuthenticationId*: The authentication and verifier script, since now has access to both values - the one at the IDP and the one at the FIDO2 Server – it validates the equality and sends a response to Keycloak.

Successful Authentication: A response that represents a successful authentication response is sent to the IDP if the user is authenticated at the FIDO2 Server.

Return Access Token: Since the FIDO2 authentication has been validated in Keycloak, now the Identity Provider returns a successful OIDC authentication response that contains an access token.

In the figure of the authentication flow, there are four main modules that work together to provide FIDO2 Authentication: Mobile (OIDC / FIDO2) Client, FIDO2 Server, Identity Provider and Authenticator and Verifier Script.

The Mobile Client resides on the user’s device and is part of the INCOGNITO application. There are two modules of the Mobile Client that are involved in the authentication process. The OIDC Client is the entity that makes requests to the Identity Provider - namely Keycloak – in order to get an access token that can be used to access resources from the Service Providers that communicate with the IDP through OpenID Connect. Since the Service Providers trust the Identity Provider, each access to a Service Provider is made using an access token issued by an Authorization Server that resides at the IDP. The FIDO2 Client makes request to the FIDO2 Server in order to authenticate a user. The authentication is made by the FIDO2 Server which will propagate the result of the authentication to the Identity Provider.

The FIDO2 Server is the entity that processes the information about a user that comes from the client device in order to authenticate the user. It receives requests from the client whose format will be described in the FIDO2 Server section. The server exposes a REST API to communicate with the client and the authentication and verifier script.

The Identity Provider is the OpenID Connect implementation that assesses the identity of the user who wants to access the resources of a Service Provider. It leverages the FIDO2 Server to authenticate a user in order to be able to respond to a user’s access token request.

The Custom Keycloak Request Library module is integrated with the FIDO2 Server and it announces the Identity Provider that a user is authenticated in FIDO2 Server. When a user authenticates at the FIDO2 Server, before a successful authentication message is sent to the client device, the FIDO2 Server leverages this library to set the lastFidoAuthenticationId of the user at the Identity Provider.

When a user makes an access token request to the Identity Provider, the Authentication and Verifier script assesses the validity of the authentication session of the user at the FIDO2 Server. It checks the value of the lastFidoAuthenticationId from the IDP against the value that is stored at the FIDO2 Server. If the two values are the same, the authentication at the IDP is made and the user will get an access token.

2.5 FIDO 2 Server

The FIDO2 Server is the entity that validates the authenticity of a user. When a public / private key pair is generated on the user device, the public key is sent and stored at the FIDO 2 Server which will use it to validate each authentication of a user. The FIDO 2 Server implements the two necessary operations for password-less authentication: namely registration and authentication which are exposed to the client through a REST API.

2.5.1 Registration

When a user wants to register using the INCOGNITO application, the FIDO 2 Server initiates the registration request by sending to the FIDO 2 Client a message which encodes a Credential Creation Options data structure which will specify the information the server needs from the client in order to securely register a user.

Registration Credential Creation Options Format for the required fields:

```
dictionary PublicKeyCredentialCreationOptions {
   required PublicKeyCredentialRpEntity         rp;
   required PublicKeyCredentialUserEntity       user;
   required BufferSource                        challenge;
   required sequence<PublicKeyCredentialParameters>  pubKeyCredParams;
};
```

The PublicKeyCredentialRpEntity contains data about the relying party responsible for the request. The PublicKeyCredentialUserEntity contains information about the user account for which it is requesting attestation. The challenge sent to the user device has the role to create an attestation object that the server will use to validate that the user is in possession of the respective private key. And the last member – PublicKeyCredentialParameters – contains information about the desired properties of the credential to be created on the server side. It includes the type of the credential, as well as the cryptographic signature algorithm with which the credential and the asymmetric key pair will be used.

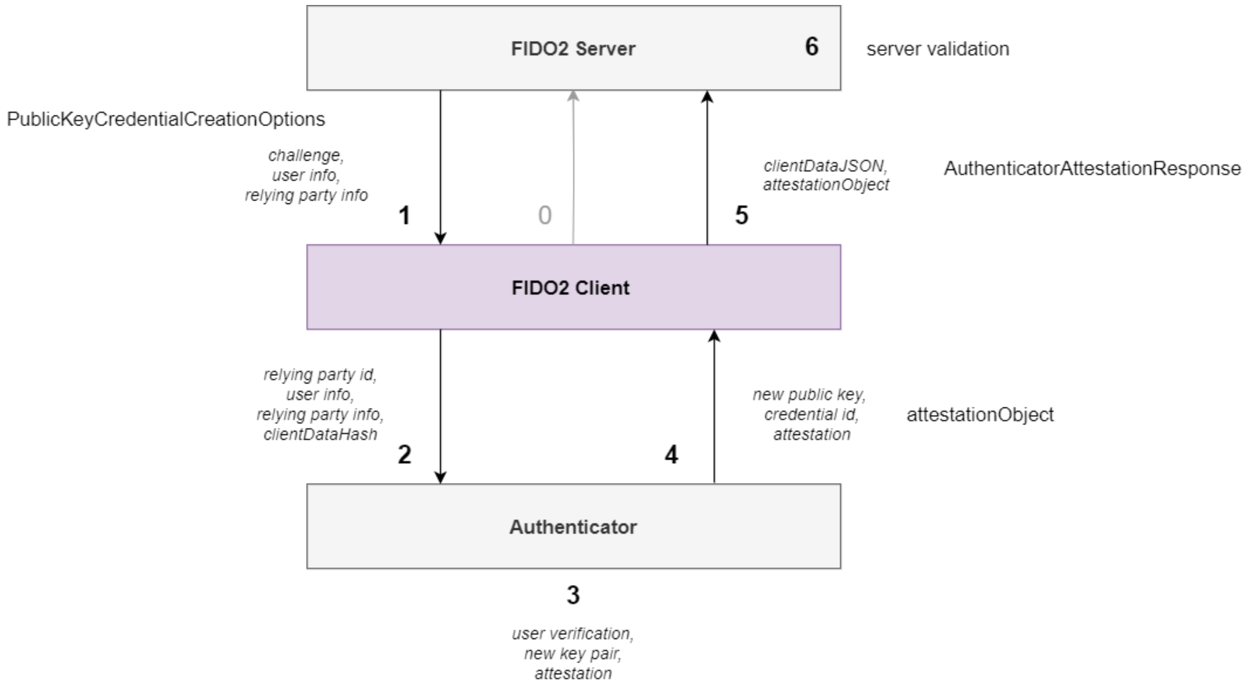

FIDO2 Server
6
server validation
PublicKeyCredentialCreationOptions
challenge, user info, relying party info
1
0
clientDataJSON, attestationObject
AuthenticatorAttestationResponse
5
FIDO2 Client
relying party id, user info, relying party info, clientDataHash
2
4
new public key, credential id, attestation
attestationObject
Authenticator
3
user verification, new key pair, attestation

Figure 6. FIDO2 Registration flow diagram

2.5.2 Authentication

After the registration with a FIDO2 Server, the user has to authenticate every time they want to access a service. Now the public key resides at the server side, therefore it is only necessary to prove to the FIDO 2 Server that they are in possession of a private key. This process is made through sending an AuthenticatorAssertionResponse structure from the client to the server. But just as in the registration process, the server is the one who initiates the communication by sending a PublicKeyCredentialRequestOption structure to the client in order to specify the options it needs to know for the completion of the authentication process.

```
Authentication PublicKeyCredentialRequestOption Format for the required fields:
dictionary PublicKeyCredentialRequestOptions {
   required BufferSource              challenge;
};
```

The authentication process only needs a challenge from the client side. This challenge is signed using the private key on the user’s device in order to prove that the user is in possession of the respective private key.

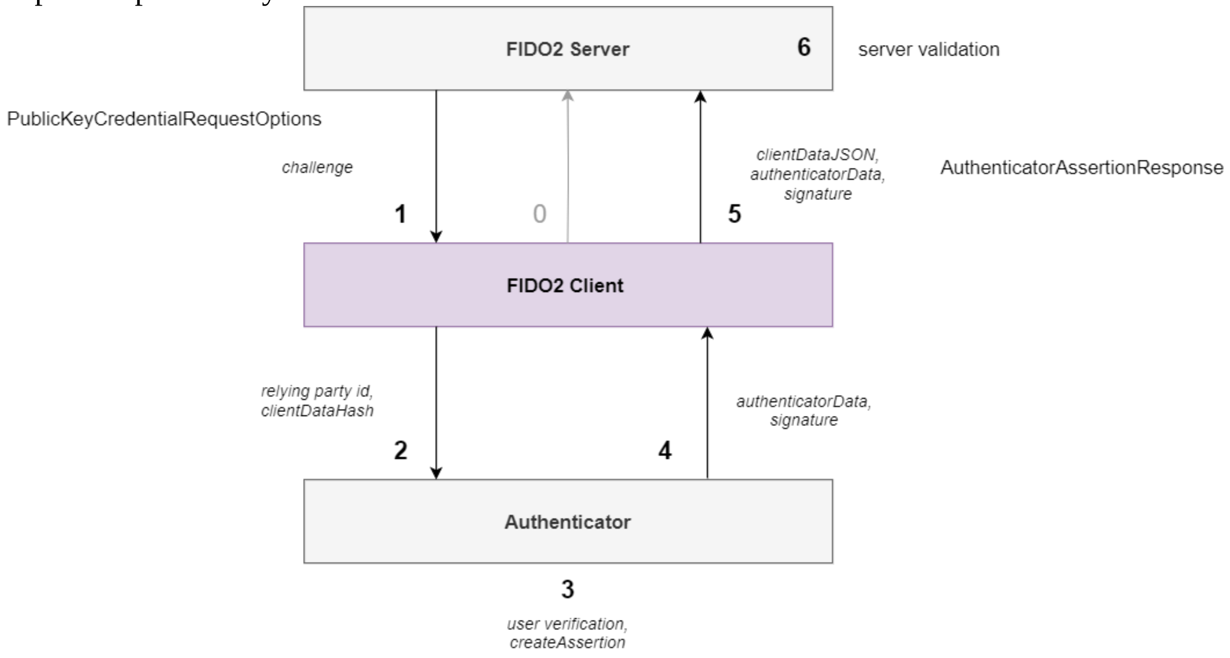

FIDO2 Server
6
server validation
PublicKeyCredentialRequestOptions
challenge
1
0
clientDataJSON, authenticatorData, signature
5
AuthenticatorAssertionResponse
FIDO2 Client
relying party id, clientDataHash
2
4
authenticatorData, signature
Authenticator
3
user verification, createAssertion

Figure 7. FIDO2 Authentication flow diagram

2.6 FIDO 2 Client

The FIDO2 Client implementation depends on the platform it is used for. It can be developed as a client embedded in browser or as a native android application. Both ways respect the same principles described in the WebAuthn Specification.

The INCOGNITO mobile application implements a client for android that uses a library from Google which is able to connect to the server. This library – namely FIDO2 API – implements a WebAuthn Client – which supports the use of roaming authenticators (BLE, NFC, USB), as well as the use of platform authenticators. The client is a mediator between the server and the authenticator and is in charge of converting the information between the two entities. The communication between the client and the server is made through JSON-serialized data. Since the communication between the client and the authenticator can sometimes take place through low-bandwidth or low-latency medium (BLE, NFC), the authenticator cannot support the transmission of long messages as JSON encoded ones. Therefore, the client receives the information from the server, it deserializes it and hashes it before sending to the authenticator. Since the authenticator is not particularly interested in the data itself, it receives the client data in a smaller hashed format that it will sign and send back to the client which will encode this data into a JSON format that can be received by the server.

The communication between the client and the server has the following JSON format:

1. **Registration Protocol**

Method: POST
Body: application/json
Response format:

```
{
	"id": "LFdoCFJTyB82ZzSJUHc c72yraRc_1mPvGX8ToE8su39xX26Jcqd31LUk",
	"rawId": "LFdoCFJTyB82ZzSJUHc-c72yraRc_1mPvGX8ToE8su39xX26Jcqd31LUkK",
	"response": {
"clientDataJSON":"eyJjaGFsbGVuZ2UiOiJOeHlab3B3VktiRmw3RW5uTWFl",
	"attestationObject":"o2NmbXRoZmlkby11MmZnYXR0U3RtdKJjc2lnWEcwR"
	},
	"type": "public-key"
}
```

2. **Authentication Protocol**

Method: POST
Body: application/json
Response format:

```
{
	"id":"LFdoCFJTyB82ZzSJUHcc72yraRc_1mPvGX8ToE8su39xX26Jcqd31LUkKOS36F
	",
	"rawId":"LFdoCFJTyB82ZzSJUHc-
c72yraRc_1mPvGX8ToE8su39xX26Jcqd31LUkKO",
	"response":{
		"authenticatorData":"SZYN5YgOjGh0NBcPZHZgW4_krrmihjLHmVzzuo",
	"signature":"MEYCIQCv7EqsBRtf2E4o_BjzZfBwNpP8fLjd5y6TUOLWt5l9D",
		"userHandle":"",
	"clientDataJSON":"eyJjaGFsbGVuZ2UiOiJ4ZGowQ0JmWDY5MnFzQVRwe"
	  },
	"type":"public-key"
}
```

3 INCOGNITO Component Implementation for Qualified Anonymity Support

3.1 Federated Anonymous Credentials Management

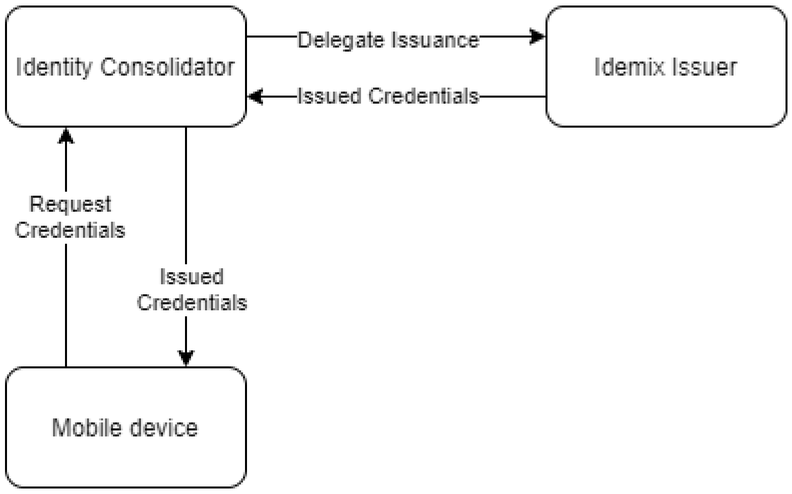


Figure 8. Idemix Issuance Flow

The main gateway to the architecture and the component that initializes anonymous credentials issuance is the user device. The main functionalities of the mobile device are to: receive anonymous credentials from the Issuer, to store anonymous credentials in a secure manner and to make disclose sessions for anonymous credentials.

Anonymous credentials are used by user who do not wish to disclose their identity when accessing a Service Provider. The user has the possibility to opt for disclosing only a subset of necessary attributes. These attributes are encoded into anonymous credentials which are used to generate proof for the required information.

To be able to authenticate in an anonymous manner, the end user will leverage a mobile application which will aide in credential management. Before generating anonymous credentials, the user needs to disclose certain attributes. The attributes will be stored and centralized on the Identity Consolidator. The UX Assistant will cooperate with the Consolidator and will help the user manage the attributes he chooses to disclose.

All the issued credentials will be stored on the user device and will never leave the secure storage. The credential will be protected with encryption mechanism and access to the encryption key will be granted by leveraging OS functionalities: PIN password, fingerprint senson, FaceID etc.

As the credential will be issued with a short-time validity (e.g., a few days or even less), there is no need to implement credential revocation and they will expire naturally. This has the added advantage that it mitigates the impact of device loss or theft.

3.2 Idemix Implementation

INCOGNITO makes use of IRMA, a collection of open-source software applications implementing all of the aspects of the Idemix attribute-based credential system.

3.2.1 Idemix security properties

IRMA implements the Idemix attribute-based credential scheme, allowing users to safely and securely authenticate themselves as privacy-preserving as the situation permits. Consequently, it guarantees essential security properties such as:

- **Unlinkability:** when a verifier receives an identical set of attributes in two distinct sessions, it cannot tell whether these sessions were started by the same user or by two distinct users
- **Unforgeability:** only the holder of the Idemix private key can issue credentials verifiable by the corresponding Idemix public key
- **Immunity to replay attacks:** eavesdroppers cannot replay disclosure sessions because the verifier first sends a number of random bits called the nonce to the mobile app, whose reply containing the disclosed attributes and the proofs of knowledge has to precisely fit this nonce; likewise, the verifiers themselves cannot disclose received attributes to other verifiers, because the mobile app never sends a complete copy of the credential’s signature to the verifier, parts of it remaining hidden using proofs of knowledge.

3.2.2 IRMA attributes

The attributes that IRMA works with are statements or properties regarding a person, such as being over 18 years old. They are contained in a credential, with possibilities of organizing them with conjunctions and disjunctions, so that, for example, users can be presented with more than one option regarding the attributes that they want to disclose.

IRMA credentials have multiple attributes ranging from the credential type to issuer metadata. The first and most important attribute of any IRMA credentials is the user private key, a random 256-bit integer value generated at installation on the mobile device. Whenever a new credential is generated, the mobile application will ensure that the first attribute associated will be the private key. As opposed to metadata regarding issuers, the private key is never disclosed and is kept hidden even during the credential issuance process.

When disclosing attributes that are proven by multiple credentials, a proof a knowledge will be generated by the mobile application. This proof will prove to the verifier that:

- The mobile application has a valid signature over each individual credential from which the attributes are disclosed
- The first attribute of all credentials coincides proving that they are held by the same user. This is also a guarding mechanism against users pooling their credentials.

Every credential must be an instance of a credential type, which what attributes it contains and what issuer instance issues them. A very important special attribute that each credential must have is the metadata attribute, which is always disclosed along with any other attributes disclosed from this credential. It specifies which credential type the current credential is an instance of, the date at which it was issued and the date when it expires. Another attribute present in every credential, but never disclosed, not even to the issuers during issuance, is the user’s secret key. The mobile app generates it the first time it is run and then ensures that all the user’s credentials have it as the first attribute. This way, the user can disclose attributes coming from multiple credentials, with

IRMA’s proof of knowledge demonstrating that the application knows a valid issuer signature over these credentials and that their first attribute is the same.

All details regarding the credential types in use, attribute names and Idemix public keys have to be included in a signed scheme, maintained by a scheme manager and distributed both to the server and to the mobile app. The scheme is also hosted online, so that when the mobile app encounters an issuer, credential type or public key that it is not aware of, it can update its scheme by downloading it from the website.

3.2.3 IRMA sessions

IRMA supports multiple sessions types:

- Disclosure session: when presented with a list of required attributes, the mobile application will display the information to the user and will request disclosure. If accepted, the application will generate a disclosure session which implies generating a proof of knowledge over the corresponding credentials
- Issuance session: the user requests new credentials to be issued for the corresponding attributes. The Issuer will generate the requested credentials which will then be stored securely locally.
- Attribute-based signature sessions: similar to disclosure sessions, they imply attaching a digital signature based on the attributes to a message. The signature can be validated at a later time proving that the message was not tampered with and that the credentials used were valid at signing time.

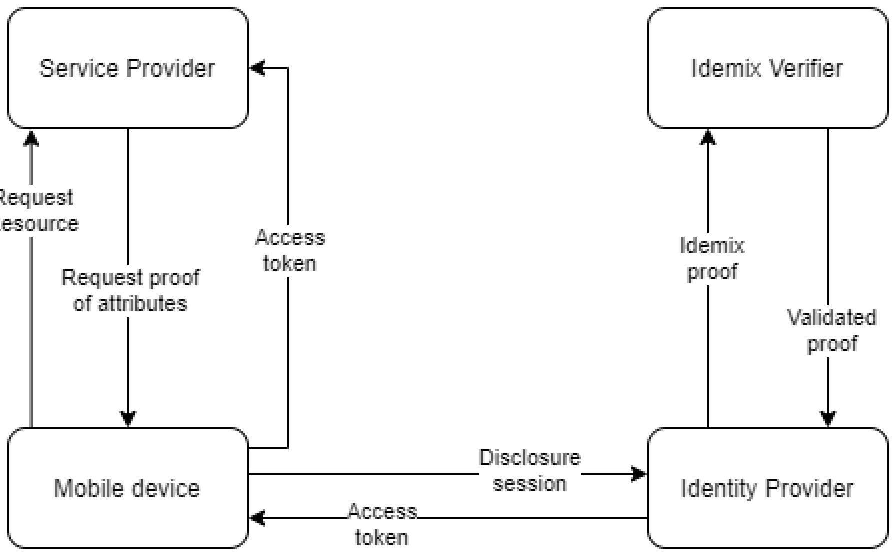


Figure 9. Disclosure session

The INCOGNITO architecture uses disclosure and issuance sessions. The end user will request new credentials to be issued after authenticating to the Consolidator. The request will be forwarded to the Issuer component and the resulting credentials will be stored in secure fashion on the mobile device.

If the user wishes to authenticate in an anonymous manner a disclosure session will be used. The credential stored in the mobile device will be used to generate a proof of knowledge which will be forwarded to the Identity Provider. The Identity Provider will validate the proof be means of the Verifier Component and will issue access tokens for valid requests.

3.3 FIDO 2 Extensions for Qualified Anonymity and Anonymous Credentials

In order to achieve password-less authentication the INCOGNITO architecture leverages FIDO2 authenticators. The federated authentication flow involves the FIDO2 Server, the Identity Consolidator or an Identity Provider. When a user tries to get an access token in order to access other Service Providers’ services, the user will be prompted to first authenticate using FIDO2 Client.

Similarly, when new credentials, anonymous credentials are required the end user must be authenticated and when issued credentials need to be validated the validation process will involve Identity Providers.

Generally, the IRMA implementation of Idemix relies on creating a session between the mobile application and the server by leveraging an identifier encoded in a QR code scanned by the mobile device. In order to support FIDO2 based authentication, extensions flows were defined at the Identity Consolidator and Identity Providers.

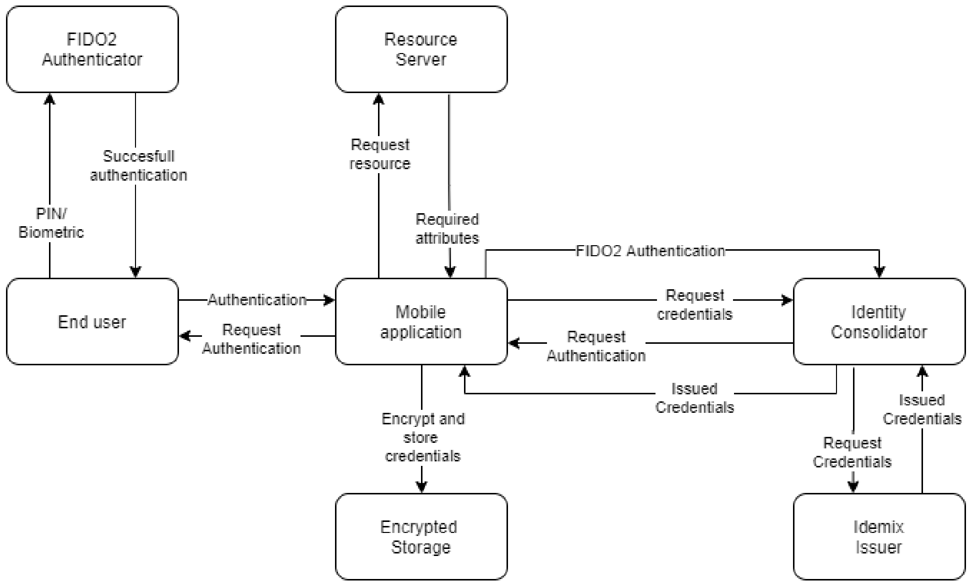


Figure 10. FIDO2 extension for Idemix credentials issuance

The anonymous credentials issuance flow is comprised of the following steps:

1. The Resource Server demands a series of attributes that need to be disclosed in order to grant access to resources.
2. The end user mobile device application scans the current available credentials and determines that new credentials need to be issued.

3. The mobile application uses FIDO2 authenticators to authenticate to the Identity Consolidator.
4. The mobile application contacts the Consolidator and requests new credentials.
5. The Consolidator delegated the issuance to the Idemix Issuer component.
6. The Issuer trusts the requests coming from the Consolidator, since all users must authenticate beforehand and issues the requested credentials.
7. The mobile application receives the requested credentials and stores them in an encrypted fashion using a private key accessible only to the legitimate user.

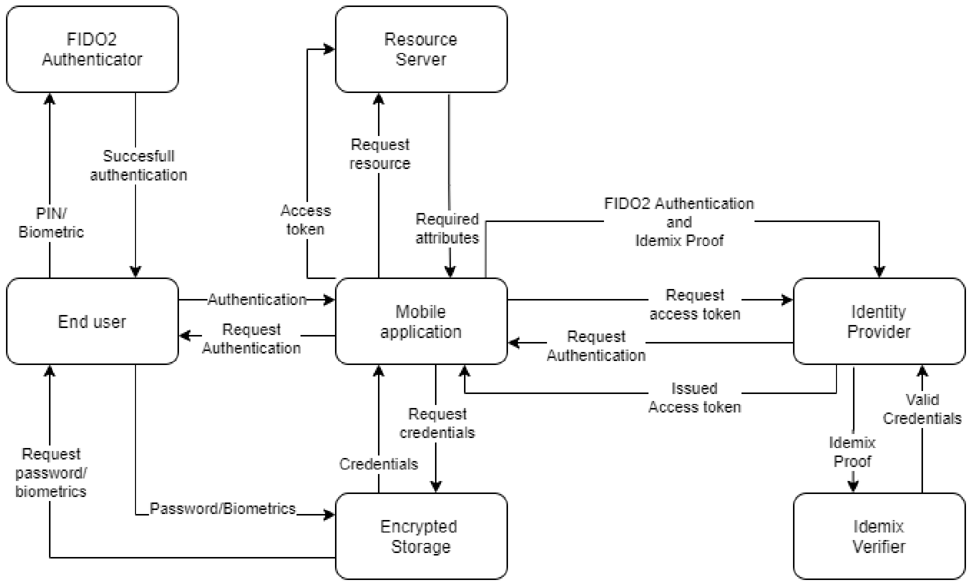


Figure 11. FIDO2 extension for Idemix credentials disclosure

The anonymous credentials disclosure process consists of the following steps:

1. The Resource Server demands a series of attributes that need to be disclosed in order to grant access to resources.
2. The end user mobile device application scans the current available credentials and determines that one of them is appropriate.
3. The mobile application request access to the secret key and decrypts the necessary credential.
4. Using the credential, a proof of knowledge is generated and disclosed to the Identity Provider. Contacting the Identity Provider is dependent on prior authentication which is obtained by using FIDO2 authenticators.
5. The Identity Provider contacts the Idemix Verifier in order to verify the proof of knowledge on the credentials.
6. The Idemix Verifier will validate the proof by checking the signature which proves that the end user has a credential for the attributes issued by a legitime issuer and that the credentials are still valid.
7. The Identity Provider will issue a corresponding access token.
8. Using the issued access token, the mobile application will request access to the resources.

4 Federated Qualified Anonymity Application Scenarios

4.1 Bot-or-not/Fake News Dissemination

Online Social Networks (OSNs) play an essential role in how people communicate and consume information. Mainly because OSNs provide an ideal environment for communication and information acquisition, as users have access to a staggering amount of posts and articles that can share with others in real-time. Unfortunately, OSNs have also become the mechanism for massive campaigns to spread false information [11] [12].

The extensive dissemination of false information in OSNs can pose a significant problem, affecting society in extremely worrying ways. For example, false information can hurt a candidate's image, potentially altering the outcome of an election. During crises (e.g., terrorist attacks, earthquakes, etc.), false information can cause result in widespread panic and general chaos.

This section presents the user stories for the use case where users want to prove attributes of their identity when posting in online forums and social media to prove that they are humans and increase the credibility of their post.

4.1.1 Before Qualified Anonymity

Users often read a variety of posts from online social media and forums without knowing if they are real or not. A general overview of the process that is currently followed is presented in *Figure 12*. For example, an End User X (malicious user) writes an article about an earthquake that hit Zaragoza, Spain (step 1). The article describes major damages in buildings and roads and a number of injuries leaving also open the possibility that people have died. The article is published in a news portal that is not using any mechanism to verify the content or to check the identity of the author. End User A and End User B who is very active on social media, reads this article and is panicked (step 2 and step 3).

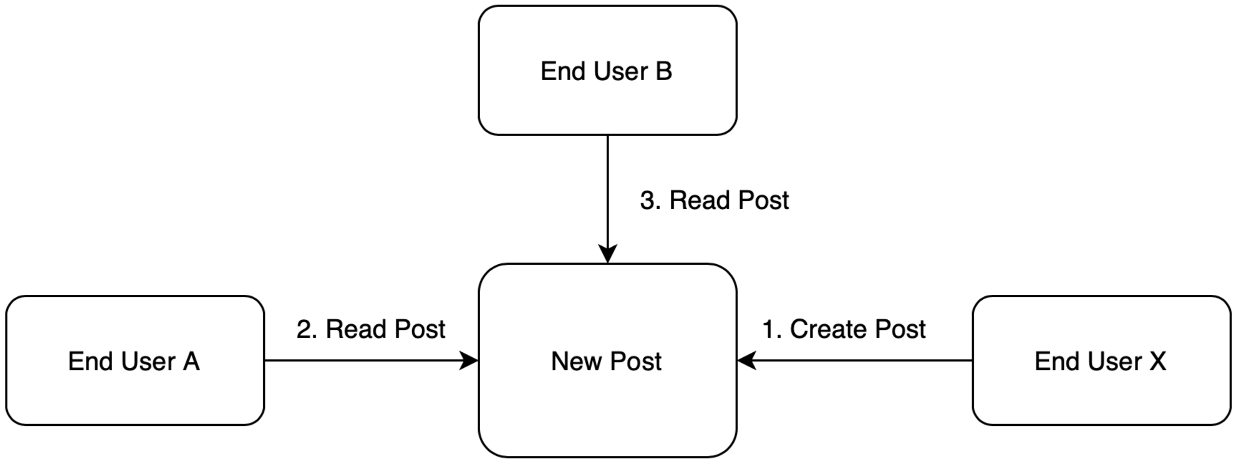


Figure 12: Before Qualified Anonymity

4.1.2 After Qualified Anonymity

This section's main scope is to use the INCOGNITO platform in conjunction with popular online social networks or news aggregators. As an example, Reddit, where users can prove that they are humans accessing online services and also that they do have credibility (sharing their profession, for example) or having acquired a good reputation (as an attribute) from previous posts. The goal is to effectively solve the information credibility problem on the Web providing qualified anonymity, proper identity acquisition and management, user-based consent management (whenever necessary), and advanced UI/UX AI-based assistant upon the process of posting and re-sharing a post.

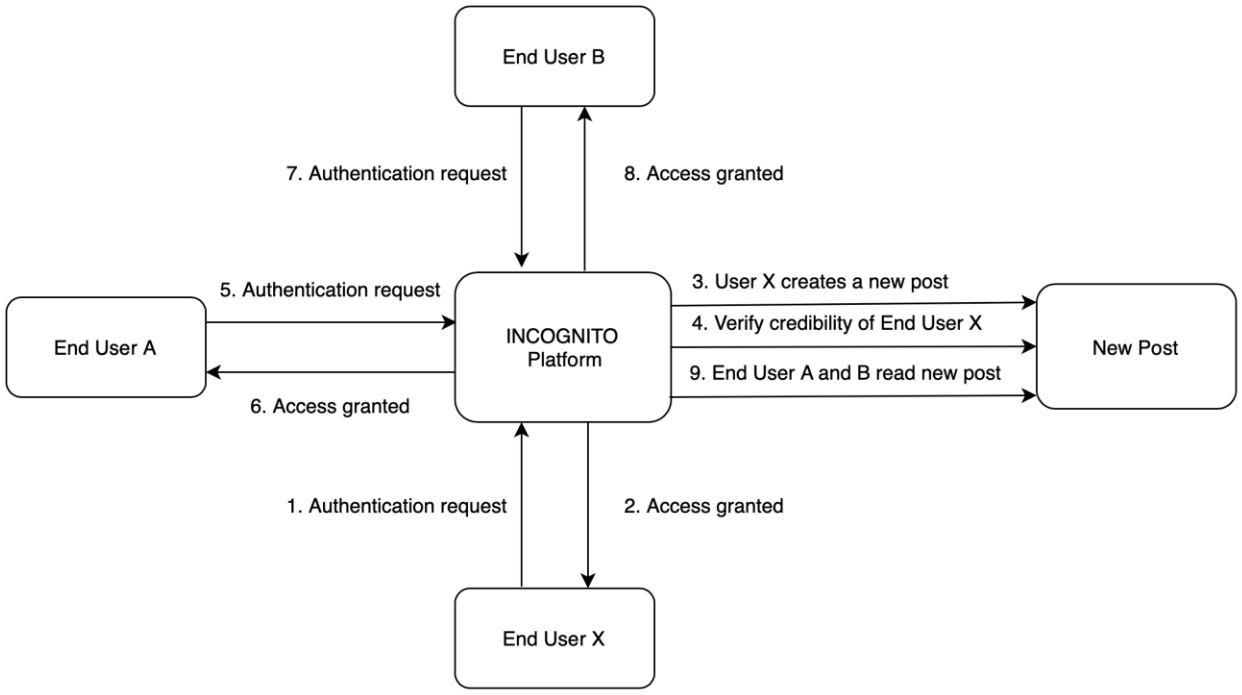

End User B
7. Authentication request
8. Access granted
5. Authentication request
End User A
6. Access granted
INCOGNITO Platform
3. User X creates a new post
4. Verify credibility of End User X
9. End User A and B read new post
New Post
1. Authentication request
2. Access granted
End User X
Figure 13: After Qualified Anonymity

Figure 13 presents a general overview of the process that is followed in order for the End User A and End User B to know if the new post/article is real or not. The contribution of the INCOGNITO platform regarding privacy preservation and credibility is being made clear by the comparison of *Figure 12* and *Figure 13*. For example, an End User X requests an authentication from the INCOGNITO platform (step 1) that require certain identity attributes to have specific values. The INCONGITO platform will then grant access to the End User X if the requirements are met (step 2) Then (step 3) the End User X creates/writes a new post/article about something (e.g., for an earthquake). The INCOGNITO platform will verify the credibility of the End User X (step 4). End User A and

User B will authenticate with the INCOGNITO platform (step 5, 6, 7 and step 8). Then they will read the new post of the End User X and can check his credibility from the INCOGNITO platform (step 9).

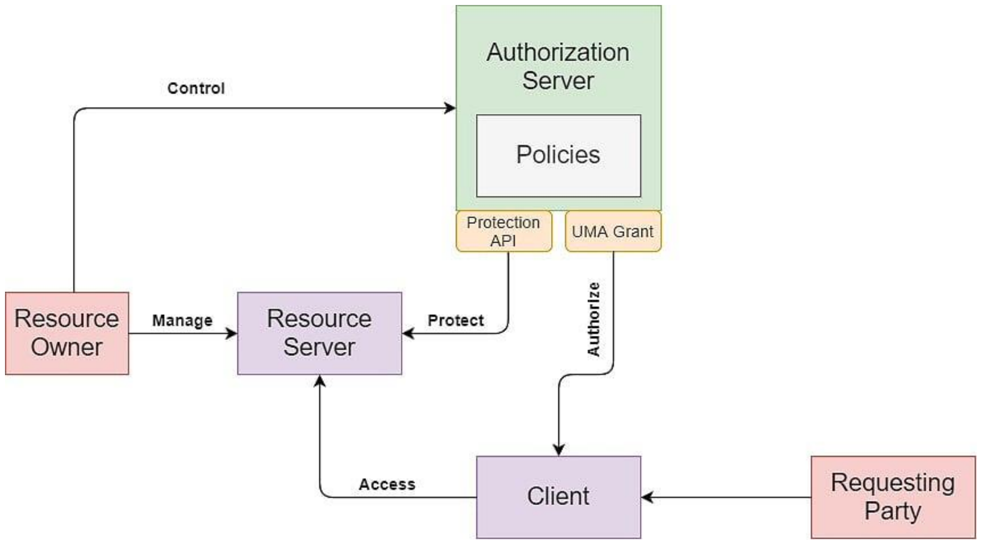


Figure 14: Bot-or-not/Fake News Dissemination

A more detailed description about the involved entities and their role regarding that is performed to parties that require information that resides on the resource server, is presented in *Figure 14.*

First, a user can register to an online multimedia service provider only if he has an online identity in order to access the content shared on the website. The user’s identity will be proven to the Service Provider (SP) through an authentication process using the Identity Provider (IdP). If the user does not have an online identity, he should not be able to register to the SP. The IdP should only assure the SP that the user that the user who wants to register has a valid identity, without revealing other information about his identity.

Second, in order to prevent spreading disinformation a user can increase his credibility by sharing specific attributes. The user should be able to provide his attributes (e.g., profession) to the SP through the Identity Consolidator (IDC). Furthermore, the user should authenticate with the IDC to verify his identity in order to be able to prove his attributes to the services. Then the user can post important news and announcements in the news portal/discussion forum with increased credibility. The SP (forum/social media) should be able to retrieve the user’s reputation through the IDC. The user should be able to view their reputation score that the IDC has assigned.

Third, the user will be able to prove his identity and his attributes in a qualified way so that other users can trust the posted articles and ensure that he can attract a reputable audience in his service avoiding malicious actors. The SP retrieves the user information from the IDC regularly in time to ensure that the user’s reputation is always high. In addition, the IDC has a mechanism to rank the various professions. In addition, the user can revoke the access of the service provider to his identity through the INCOGNITO platform. Therefore, the SP has to remove all the information it processes regarding the use of matter. Furthermore, the user can cancel his subscription from a news portal by authenticating himself in order to prove his identity so that he blocks the SP to have access to some of his attributes.

4.2 Online Multimedia Content Sharing

In the online multimedia content sharing the laws and regulations are very strict, depending on the content. Users may have to prove several attributes of their identity in order to be able to access certain content. Users have to present proofs of their identity and economic status, probably their age and various other attributes have to be also checked. In this section we will showcase how qualified anonymity affects the way online resources are accessed, in a privacy-preserving manner.

4.2.1 Before Qualified Anonymity

Users often need to access online resources and/or services, where a minimum set of requirements is expected to be met by them. Traditionally, in order to prove that they fulfill said requirements, users are often obliged to provide identification and/or certificate documents, revealing much more information than it is actually needed. A general overview of the process that is currently followed is presented in Figure *15*. For example, a student has to upload a full scan of her Academic ID card to prove her capacity as such at a bookstore in order to take advantage of a discount she is entitled to. By doing so, she reveals her name, address, identification numbers, etc., identity attributes that are not required by the bookstore to acknowledge her capacity as a student. Another example is a client who wants to buy alcohol from an online store, where he will be asked to upload a full copy of his identity card. These real-world use cases prove that while users are able to prove themselves to the service providers, the privacy aspect of the process has not been taken into consideration adequately and is neglected. This results in the unnecessary disclosure of personal data, which goes against the general directives for “Data Minimization” that countries in many regions across the world are adopting.

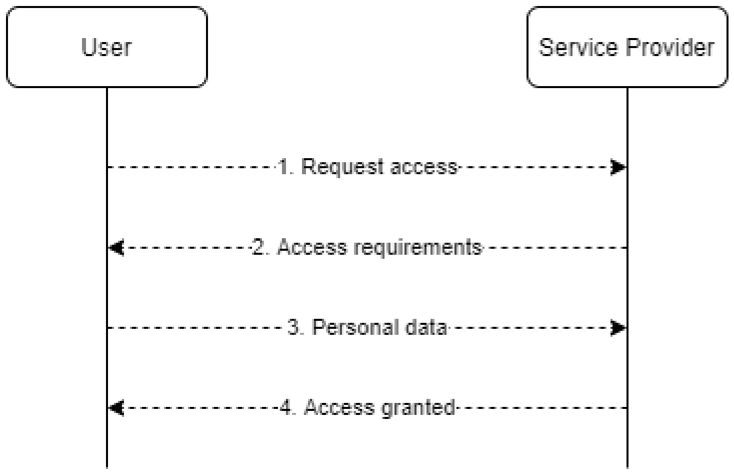


Figure 15: Multimedia Content Sharing - Before QA

4.2.2 After Qualified Anonymity

In this scenario, the goal of INCOGNITO is to safeguard the users’ personal data when trying to access online multimedia content that requires certain identity attributes to have specific values. To achieve this goal, INCOGNITO employs QA through the utilization of anonymous credentials provided by Idemix, which lies on the IDC.

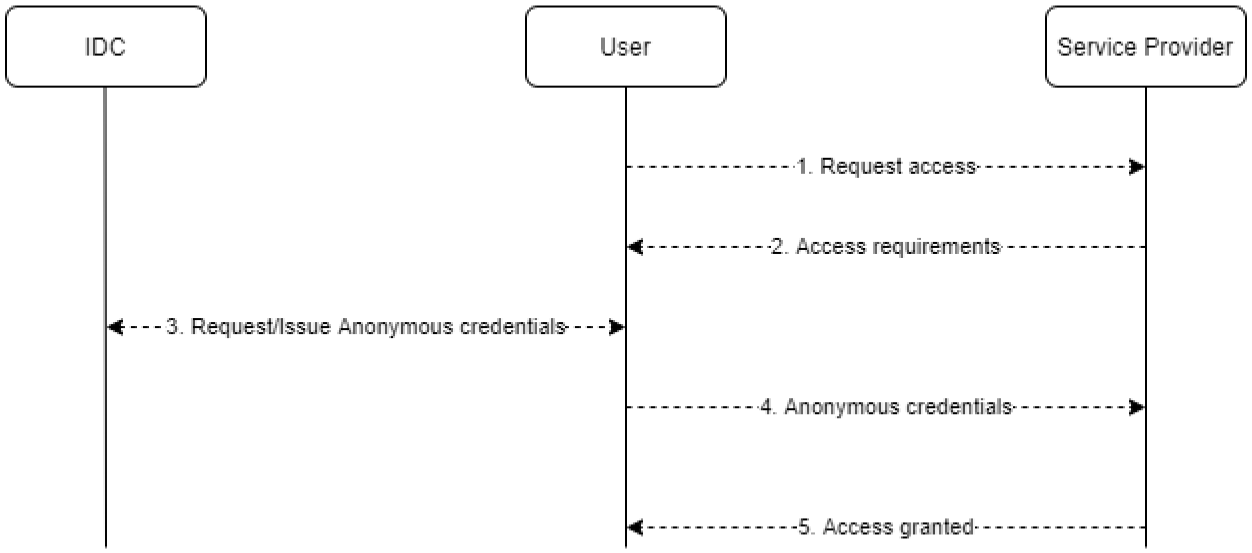


Figure 16: Multimedia Content Sharing - After QA

Figure *16* presents a general overview of the process that is followed in order for the users to prove to the SP that they possess the required identity attributes. The contribution of the INCOGNITO platform regarding privacy preservation is being made clear by the comparison of Figure *15* and Figure *16*. When a user wants to access the content of a service provider, instead of relaying personal information, anonymous credentials that prove the identity attributes that are needed are provided, without including any excess data. After their successful verification, the user is granted access to the desirable multimedia content.

A more detailed description about the involved entities and their role regarding the process that is performed to grant access to parties that require information that resides on the resource server, is presented in Figure *17*. More specifically, the UMA protocol will be leveraged to protect the online resources (multimedia content) of a Service Provider and Idemix to protect a mobile user’s privacy who wants to access these resources while remaining anonymous to the Service Provider. In this case, the Service Provider is a Resource Server (the Resource Owner can be the Service Provider itself if it has some general policies for the protection of the online multimedia platform, or it can be a content creator registered at the online multimedia platform). Keycloak is the Authorization Server, while the Client is part of the INCOGNITO application implementation and the Requesting Party will be a user of this application.

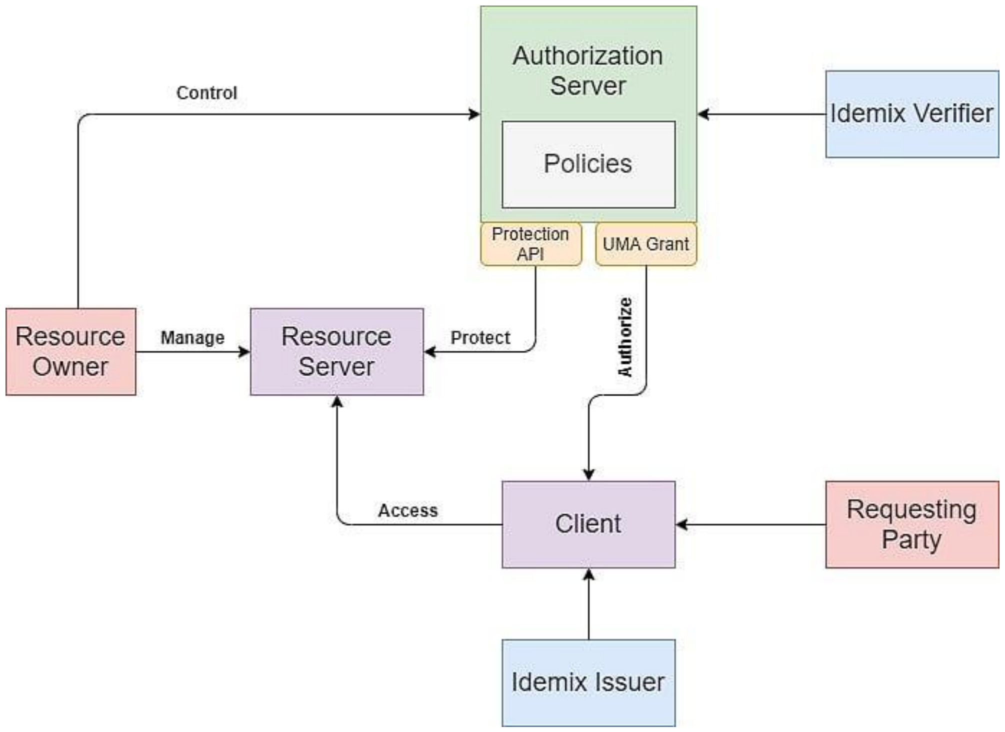


Figure 17: Multimedia Content Sharing

To better understand the interaction and functionality of these components, we present an example of this scenario. A user of the INCOGNITO application wishes to register on VideoTube and access videos available on the platform. However, VideoTube does not allow users under 18 to access its videos and protect its resources by registering them at the Authorization Server. Therefore, VideoTube acts as a Resource Server while Keycloak acts as an Authorization Server. In this example, the user wants to access some protected videos on VideoTube but first she has to prove that she is over 18 years old. However, since the user doesn’t want to reveal her identity to VideoTube, she uses the Idemix Issuer running on the Identity Consolidator to issue a cryptographic Idemix credential, which can be used to prove to the Authorization Server that she is over 18 years old and that she is eligible to access VideoTube videos. Then, the user sends the issued credential to the Authorization Server, which will assess some policies based on these Idemix credential to authorize the user to get access to these videos. In order to understand whether the user is who she claims to be or not, the Authorization Server leverages an Idemix Verifier running again on the IDC to check the credentials received from her. In case of a positive response from the Idemix Verifier, the Authorization Server will authorize the user to access videos through the means of a UMA-compliant client that is part of the INCOGNITO application on the user’s mobile. Now the user is authorized to request access to VideoTube videos using this UMA-client and the authorization access token issued by the Authorization Server.

5 Privacy and Security Considerations

This section provides the threat model, the limitations and the mitigation techniques that we employ in order to strengthen the security of the implemented solutions described in the previous sections. It is important to ensure that no malicious users can tamper, steal or expose the cryptographic credentials, or FIDO2 credentials of the INCOGNITO users.

5.1 User to Device Authentication (Biometric Authentication)

Despite the fact that biometric authentication techniques like the one provided by the FIDO2 protocol is far more secure than other traditional authentication schemes (e.g., passwords or tokens), there are certain privacy and security considerations that also need to be addressed in the context of the INCOGITO project.

First, as similar to a password or a token, it is also possible that the biometric data of a user can be compromised. If a password or token is stolen, we can change or replace it, however, this is not the case for the fingerprint or irises of a user since they cannot change so once they are compromised, they are compromised forever. The problem is much bigger when the actual biometric templates are stored in the cloud or in a central authentication server, since if such a system is bridged thousands or even millions of templates will be stolen. For this reason, INCOGNITO will not store any biometric data of the user on the cloud.

In addition, in INCOGNITO, fingerprints used for user-to-device authentication are only part of the native Android TrustZone authentications, therefore most of the security issues are addressed directly by the Android Operating System (OS). More precisely, INCOGINTO only uses the fingerprint authentication functionality of commodity mobile devices to create and access whenever this is needed to user's FIDO private key stored on the mobile device of the user. In addition, INCOGNITO platform, as in ReCRED, will be using a device-centric topology, which means that all the processing and storing of biometric data happens inside the user device. Even if the actual features are compromised (e.g., during extraction), at least this will affect only the specific user and won’t constitute a major data breach. Fingerprints have an added advantage, since they are not totally irrevocable, in the sense that even if a fingerprint is compromised some other finger can be used.

Another major issue is that certain biometrics can and will be forged. However, different biometric characteristics require different levels of effort and expertise in order to be spoofed. While it could be quite easy to fool a face detection system by using just a photograph of the “victim”, fooling a fingerprint or a retinal scanner would require much more effort and expertise (e.g., use of advanced methods in order to replicate the finger or the retina). For that reason, INCOGNITO will be using fingerprints as a verification characteristic, instead of other features with high circumvention. Moreover, as the biometric scanners evolve, they are becoming more and more capable of detecting spoofing attempts. At this point we should mention that, as a design principle, we are not collecting or processing any biometric data on the INCOGNITO platform.

5.2 FIDO2 and Federated Authentication

The integration of FIDO2 protocol with federated authentication protocols like OpenID Connect in INCOGNITO provide a far more secure way of user authentication that is based on public key cryptography compared to the tradition password authentication paradigm that OpenID Connect is based on. However, in order to achieve a sound security level with FIDO authentication, the following recommendation regarding the algorithm and key lengths have been considered when implementing and deploying the FIDO2 server on the INCOGNITO Identity Providers: *ServerChallenge* is defined by a byte array that contains different random characters each time. This data is generated by the FIDO2 server running on the Identity Consolidator or any other Identity Provider and is sent to the User Device and further included in a response message to protect from Replay Attacks. The length of such an array is defined to be between 8-64 bytes. 8 bytes should ensure a minimum-security level and 64 bytes should ensure a SHA-512 compatibility. Furthermore, authenticators are recommended to use more sources of random seeds to increase the quality of random numbers generated (RFC4086).

Moreover, the FIDO2 protocol relies on the Transport Layer Security (TLS) protocol to ensure messages policy. Protocol messages require a channel binding structure that binds security elements form the communication channel into protocol messages. The *ChannelBinding* structure is added by the client and verified by the server to protect from Man-in-the-Middle (MITM) attacks. The following bindings are supported (RFC5929): a) TLS channel ID; b) ServerEndPoint; c) TLS server certificate; and d) TLS Unique. In addition, we make use of the TLS protocol version v1.2 and only if it is not available then v1.1 should be used and we follow the rule that insecure algorithms in TLS protocol (such as MD5, RC4, and SHA1), should be avoided.

5.3 Qualified Anonymous Credentials

It is important to ensure that malicious users cannot tamper, steal or expose the cryptographic credentials of a user. To defend against this threat, in INCOGNITO we only store the cryptographic credentials of the user on his/her mobile device, which can only be accessed using a user-to-device authentication protocol, such as FIDO2. In addition, all the cryptographic credentials of the user are stored in an encrypted format in the Trusted Execution Environment (TEE) so that they may not be tampered with, even if the device of the user is stolen and/or compromised.

6 Conclusion

Deliverable D3.1 "Federated Anonymous Credentials" is the first deliverable of Work Package 3. This deliverable is a first attempt to integrate cryptographic credentials and user-to-device authentication technologies, with federated authentication protocols like OpenID Connect, with the aim to provide a secure and privacy preserving authentication framework within the INCOGNITO platform. Using the aforementioned protocols, we are able to provide support for the use of qualified anonymous credentials for the authentication of the users through the use of pseudonyms and techniques, thus ensuring unlinkability and untraceability of all the activities performed by the INCOGNITO users.

In addition to the above protocols, leveraging the User-managed Access protocol we are able to support user authorization for access to their distinct identity information through the definition and verification of adequate ABAC policies. At the same time, Service Providers in INCOGITO are able to provide FIDO authentication and qualified anonymous credentials support to their users without the need to change their infrastructure or deploy any FIDO-related or qualified anonymous credential stacks. Last, we note that the main purpose of this deliverable was to specify the techniques and solutions that enable INCOGNITO to provide support for secure device-centric authentication, as well as the technologies and techniques that enable the support of qualified anonymous credentials for authentication and authorization of users through the use of pseudonyms and techniques to ensure unlinkability and untraceability of the activities of the users.

The objectives of Task 3.1 have been fully achieved as we provide a software stack that is able to issue and verify cryptographic credentials when requested by Online Services using Idemix, combined with the aforementioned technologies. During our scientific research activities we came to the conclusion that Idemix is the better option compared to U-Prove for the integration of an anonymous credential protocol to the INCOGNITO's QA framework, thus we focused our efforts towards that end. There is still work to be done with the remaining tasks of WP3, which focus on the enhancement of the security aspects and include the following objectives. Task 3.2 aims in designing and implementing a Trusted Computing software stack for enhanced security of anonymous credential protocols, i.e. Idemix, via the execution of sensitive, from a security point of view, processes in a TEE using Intel SGX and ARM TrustZone technologies. The results from this scientific research will be presented in the context of the upcoming deliverable of this work package, deliverable D3.2. Furthermore, Task 3.3 focuses on the integration of the developed qualified anonymity infrastructure with anonymization network such as the TOR network. Being part of the work be done in the context of the last deliverable of this work package, deliverable D3.3, this work will further decrease the chances of user de-anonymization and defend against de-anonymization attacks on the network layer.

To summarize, we laid the foundations for the INCOGNITO platform to provide user anonymity, along with secure and privacy preserving ways to authenticate and authorize entities on the platform. On top of these key achievements, significant outcomes of the research activity reported in this deliverable refer to the FIDO 2.0 and OpenID Connect technologies. Our dissemination activities depict results of the research conducted in the scope of this deliverable and the corresponding task:

<u>Publication</u>

- Killing the Password and Preserving Privacy With Device-Centric and Attribute-Based Authentication[1]

<u>Talks from the Coordinator</u>

- 6th Meeting of the European Security and Defence College, Brussels, 22, Nov. 2019
- Mastering Enterprise Management and INCOGNITO, Piraeus, 24 Nov. 2019
- Critical Infrastructure Security and Resilience (CISaR) Workshop, Norway, January 30-31, 2020
- Secure Internet Day 2020, IoT: Challenges & Risks, Cyprus, 11 Feb. 2020
- 7th Information Security Conference Athens, 19 Feb. 2020
- European Association of Biometrics Conference 2020 (EAB-RPC 2020), Virtual Event, 16 Sept. 2020

For more information about the project's outcomes and activities, please visit our website[2].

[1] https://ieeexplore.ieee.org/abstract/document/8931622
[2] https://incognito.socialcomputing.eu/

7 References

[1] OpenID, "OpenID Connect Specification," 2020. [Online]. Available: https://openid.net/connect/.

[2] F. Alliance, "FIDO2: WebAuthn & CTAP," 2020. [Online]. Available: https://fidoalliance.org/fido2/.

[3] I. Mixer, "Identity Mixer: Anonyumous credentials for strong accountability and privacy," 2020. [Online]. Available: https://idemix.wordpress.com/.

[4] OAuth2.0, "OAuth 2.0," 2020. [Online]. Available: https://oauth.net/2/.

[5] R. Hat, "Keycloak: Open Source Identity and Access Management," 2020. [Online]. Available: https://www.keycloak.org/.

[6] K. Intiative, "User-Managed Access," 2020. [Online]. Available: https://kantarainitiative.org/confluence/display/uma/Home.

[7] F. Alliance, "Client to Authenticator Protocol (CTAP)," 2020. [Online]. Available: https://fidoalliance.org/specs/fido-v2.0-id-20180227/fido-client-to-authenticator-protocol-v2.0-id-20180227.html.

[8] O. C. Tradining, "What is FIDO2?," 2020. [Online]. Available: https://oxfordcomputertraining.com/glossary/what-is-fido2/.

[9] F. Alliance, "How FIDO works?," 2020. [Online]. Available: https://fidoalliance.org/how-fido-works/.

[10] G. D. Solutions, "OIDC Federated Authentication," 2020. [Online]. Available: https://www.ge.com/digital/documentation/predix-services/IZjYwZTYzYTEtZjllYS00ZTA3LTliZWItZGQ5MDMyMmY1Yzk1.html.

[11] K. Collins, "People shared nearly as much fake news as real news on Twitter during the election," 2017. [Online]. Available: https://qz.com/1090903/people-shared-nearly-as-much-fake-news-as-real-news-on-twitter-during-the-election/..

[12] T. Guardian, "Facebook’s failure: did fake news and polarized politics get Trump elected?," 2016. [Online]. Available: https://www.theguardian.com/technology/2016/nov/10/facebook-fake-news-election-conspiracy-theories.

[13] 2020. [Online]. Available: https://fidoalliance.org/specifications/overview/.