Università degli Studi di Salerno

Dipartimento di Informatica

Corso di Laurea Magistrale in Sicurezza Informatica

Tesi di Laurea Magistrale in Sicurezza Informatica

Progettazione di un'infrastruttura di sicurezza Multi-Banca per utenti non-consumer

Relatore
Prof. **Christiancarmine Esposito**
Correlatore
Emanuele Natola

Candidato
Luigi Vollono
Matricola
05225/01163

Anno Accademico 2023-2024

Università degli Studi di Salerno

Dipartimento di Informatica

Via Giovanni Paolo II, 132, 84084 Fisciano SA

"Se non si crede spesso in quello che possiamo dare, è importante tenere strette quelle persone che ci dimostrano che invece ci sbagliamo."

Ringraziamenti

Desidero esprimere la mia più profonda gratitudine a tutti coloro che hanno reso possibile la realizzazione di questa tesi. Un sentito ringraziamento va innanzitutto al mio relatore per la sua guida sapiente, il sostegno costante e la pazienza dimostrata nel corso di questi anni. La sua competenza e il suo impegno professionale sono stati per me fonte di ispirazione e crescita accademica.
Non posso dimenticare l'importante contributo dei miei genitori che con il loro amore incondizionato, il loro incoraggiamento e il loro sacrificio, mi hanno permesso di perseguire i miei sogni. A loro devo la mia tenacia e la mia determinazione.
Infine, ma non per importanza, desidero ringraziare i miei amici più cari per avermi sempre incoraggiato e per avermi offerto una valvola di sfogo nei momenti di stress. La loro presenza è stata una costante fonte di allegria e rilassamento.

Abstruct

La sicurezza informatica è un tema cruciale per le attività economiche e professionali che si basano su servizi digitali, come quelli offerti dalle banche. In particolare, gli utenti non-consumer, come notai e commercialisti, hanno bisogno di accedere a più istituti bancari con un'unica identità digitale, garantendo al tempo stesso la protezione dei dati sensibili e la conformità alle normative vigenti.

Questa tesi rappresenta la conclusione dei miei studi universitari ed ha lo scopo di descrivere il lavoro svolto durante i sei mesi di tirocinio presso l'azienda "Technology Reply Roma". Tutto ciò è avvenuto sotto la supervisione del mio docente relatore, il Prof. Christiancarmine Esposito, e del Dott. Emanuele Natola correlatore, nonché tutor aziendale.

L'obiettivo della tesi è di progettare e implementare una infrastruttura di sicurezza multi-banca per utenti non-consumer, come notai e commercialisti, che necessitano di accedere a servizi bancari online in modo sicuro e conveniente. La sfida principale è di garantire un livello elevato di protezione dei dati sensibili degli utenti e delle transazioni finanziarie, senza compromettere l'usabilità e l'efficienza del sistema. Per raggiungere questo scopo, si propone di utilizzare una soluzione basata sulla federazione tra la componente Axway API Gateway e Oracle Access Manager, che sfrutta il protocollo OAuth 2.0 per l'autorizzazione dei client.
Axway API Gateway è una piattaforma per la gestione, la consegna e la sicurezza delle API web, che offre funzionalità di integrazione, accelerazione, governance e protezione per i sistemi basati su API e SOA. Oracle Access Manager è un prodotto per la gestione dell'identità e dell'accesso, che fornisce funzionalità di autenticazione, autorizzazione, single sign-on, federazione e auditing per le applicazioni web. OAuth 2.0 è uno standard aperto per il protocollo di autorizzazione, che consente ad un client di ottenere un token di accesso da un server di autorizzazione, per accedere a risorse protette ospitate da un server di risorse, in nome di un utente.

La soluzione proposta prevede che ogni banca partecipante esponga le proprie API bancarie tramite Axway API Gateway, che funge da punto di ingresso unico per i client. Le API bancarie sono protette da token OAuth 2.0, che vengono rilasciati da Oracle Access Manager, che funge da server di autorizzazione federato per tutte le banche. Gli utenti non-consumer si autenticano presso Oracle Access Manager, che verifica la loro identità tramite un dispositivo di sicurezza software, e ottengono un token di accesso per le API bancarie a cui sono abilitati. I client utilizzano il token di accesso per invocare le API bancarie tramite Axway API Gateway, che verifica la validità

del token e applica le regole di business e di sicurezza definite dalle banche. In questo modo, si realizza un meccanismo di delega dell'autorizzazione, che consente agli utenti non-consumer di accedere a servizi bancari multi-banca in modo sicuro e trasparente, senza dover condividere le proprie credenziali con i client o con le banche.
La tesi descrive il processo di progettazione e implementazione della soluzione proposta, illustrando le scelte architetturali, le tecnologie utilizzate, le funzionalità offerte e le sfide affrontate. Inoltre, la tesi presenta i risultati di una valutazione sperimentale della soluzione, condotta su un prototipo realizzato con le componenti Axway API Gateway e Oracle Access Manager, e basata su scenari di uso realistici. L'analisi ha riguardato aspetti quali la sicurezza, la performance, la scalabilità e l'usabilità del sistema. I risultati hanno dimostrato che la soluzione proposta è in grado di soddisfare i requisiti di sicurezza e di qualità del servizio richiesti dagli utenti non-consumer e dalle banche, e di offrire un'esperienza utente semplice e intuitiva.
La tesi contribuisce al campo della sicurezza informatica applicata al settore bancario, proponendo una soluzione innovativa e basata su standard aperti per la federazione di servizi bancari online per utenti non-consumer. La soluzione può essere estesa e adattata ad altri contesti applicativi, in cui sia richiesta una gestione sicura e federata dell'accesso a risorse protette tramite API web.
La tesi esplora anche nuove prospettive per sviluppi futuri, confrontando il protocollo OIDC con il nuovo protocollo OID4VC. Questo confronto evidenzia come OID4VC possa superare le limitazioni e criticità di OIDC, offrendo soluzioni innovative e miglioramenti significativi per l'autenticazione e la gestione delle identità digitali.

Indice

Elenco delle figure

Capitolo 1

Introduzione

In questo capitolo viene fornita una panoramica generale della tesi.
La prima sezione esamina l'obiettivo da raggiungere, considerando il problema da risolvere.
La seconda sezione illustra le motivazioni che hanno portato alla realizzazione di questo lavoro di tesi.
Infine, la terza sezione illustra la struttura della tesi, delineando i capitoli e i loro contenuti.

1.1 Obiettivo

L'obiettivo principale di questa tesi è progettare e implementare un'infrastruttura di sicurezza multi-banca per utenti non-consumer, come notai e commercialisti, che necessitano di accedere a servizi bancari online in modo sicuro e conveniente. Per raggiungere questo obiettivo, la tesi si propone di:

- Implementare misure di sicurezza avanzate per proteggere i dati sensibili degli utenti e le transazioni finanziarie, utilizzando tecnologie e protocolli di sicurezza all'avanguardia.
- Assicurare che il sistema sia facile da usare e non comprometta l'efficienza operativa degli utenti, mantenendo un equilibrio tra sicurezza e usabilità.
- Sfruttare la federazione tra componenti di sicurezza per creare un sistema sicuro e centralizzato.
- Condurre una valutazione della soluzione implementata, analizzando aspetti quali sicurezza, performance, scalabilità e usabilità.
- Esplorare nuove prospettive per sviluppi futuri, confrontando diversi protocolli di sicurezza per evidenziare soluzioni innovative e miglioramenti significativi.

Il sistema proposto è stato sviluppato per un cliente dell'azienda dove lavoro e, attualmente, le infrastrutture di sicurezza esistenti si dividono per contesti di business:

- Infrastruttura di Federazione:
  Dedicata agli accessi ad applicativi interni da parte di operatori di back-office delle Banche.
- Infrastruttura di accesso Multi-istituto:
  Dedicata agli accessi all'Home Banking per i clienti delle banche.
- Infrastruttura di accesso singolo istituto:
  Dedicata alla messa in sicurezza delle API REST/SOAP esposte dai web service di una determinata banca.

Di conseguenza, è necessario sviluppare una nuova infrastruttura specificamente dedicata al contesto descritto all'inizio.

1.2 Motivazioni

Le motivazioni dietro questo progetto sono molteplici. In primo luogo, la sicurezza informatica è diventata una priorità assoluta per le attività economiche e professionali che si basano su servizi digitali, come quelli offerti dalle banche. In particolare, gli utenti non-consumer, come notai e commercialisti, necessitano di accedere a più istituti bancari con un'unica identità digitale, garantendo al tempo stesso la protezione dei dati sensibili e la conformità alle normative vigenti.
In secondo luogo, questa esigenza nasce dalla crescente complessità e interconnessione dei servizi bancari, che richiedono soluzioni innovative per gestire in modo sicuro e conveniente l'accesso alle risorse finanziarie.
In terzo luogo, la crescente incidenza di attacchi informatici e violazioni dei dati ha reso evidente la necessità di rafforzare le misure di sicurezza per proteggere le informazioni sensibili degli utenti. Le conseguenze di tali violazioni possono essere devastanti, sia in termini di perdita finanziaria che di danno alla reputazione delle istituzioni coinvolte. Pertanto, sviluppare un'infrastruttura di sicurezza robusta è essenziale per prevenire tali rischi e garantire la fiducia degli utenti nei servizi bancari online.
Inoltre, si ritiene che l'adozione del protocollo OID4VC possa offrire ulteriori miglioramenti rispetto al protocollo OIDC. Pertanto, un confronto tra i due protocolli sarà effettuato per valutare come OID4VC possa affrontare le limitazioni e criticità di OIDC, offrendo soluzioni innovative e miglioramenti significativi per l'autenticazione e la gestione delle identità digitali.

Il lavoro di tesi è stato intrapreso per rispondere a queste esigenze, con l'obiettivo di progettare e implementare un'infrastruttura di sicurezza multi-banca che possa soddisfare le esigenze specifiche degli utenti non-consumer.

1.3 Struttura della tesi

La tesi è strutturata come segue:

- **Capitolo 2**: Fornisce una panoramica sui principali concetti e protocolli utilizzati, come l'Identity and Access Management e lo standard di autenticazione OpenID Connect (OIDC), per poi introdurre il protocollo OpenID Connect for Verifiable Credentials (OID4VC), dove è stata formalizzata anche una research question.

- **Capitolo 3**: Illustra le principali tecnologie utilizzate, come le componenti del pacchetto Oracle Identity and Access Management Suite Plus e l'API Gateway di Axway, escludendo quelle secondarie.

- **Capitolo 4**: Descrive il sistema proposto e il suo funzionamento con un livello di dettaglio tecnico maggiore, includendo lo scenario di utilizzo, l'architettura del sistema, i dettagli del flusso di autenticazione, un'analisi dettagliata tra il modello Federated Identity (FI) e Self-Sovereign Identity (SSI) con tutte le loro componenti e protocolli, infine, descrive la fase di sviluppo dei plugin OAM e configurazione delle componenti dell'API Gateway di Axway, non trattandoli nel dettaglio a causa di Accordi di Non Divulgazione.

- **Capitolo 5**: Presenta in dettaglio le valutazioni effettuate sul sistema proposto, con particolare attenzione agli stress test eseguiti. Vengono analizzati i risultati ottenuti per dimostrare la robustezza e l'affidabilità del sistema sotto condizioni di carico estremo.

- **Capitolo 6**: Riassume i risultati ottenuti dal sistema realizzato e i potenziali sviluppi futuri, delineando anche i dettagli dell'analisi condotta per la parte di ricerca.

Capitolo 2

Background

In questo capitolo viene fornita una panoramica sui principali concetti e protocolli trattati nel lavoro di tesi.
La prima sezione offre una descrizione generale dell'Identity and Access Management, con un focus particolare sull'Access Management, uno dei temi centrali del lavoro.
Nella seconda, terza e quarta sezione vengono illustrate le tre tipologie di servizi in cui si suddivide l'IAM.
La quinta sezione introduce lo standard di autenticazione su cui si basa la soluzione IAM implementata, ovvero OpenID Connect (OIDC).
La sesta sezione presenta la parte di ricerca, ovvero lo standard di autenticazione OpenID Connect for Verifiable Credentials.
Infine, la settima sezione illustra la research question che guida il lavoro svolto.

2.1 Identity and Access Management (IAM)

Uno dei problemi che le organizzazioni di grandi dimensioni si trovano ad affrontare è quello di gestire gli accessi alle informazioni e alle applicazioni, sia su reti interne che su reti esterne. Con l'aumentare del personale, interno e esterno all'organizzazione, e dei servizi IT messi a disposizione, il numero degli accessi e delle identità da controllare e proteggere diventa sempre più un punto critico.

L'"Identity and Access Management" gestisce il ciclo di vita end-to-end delle identità e dei diritti degli utenti per tutte le risorse aziendali. È un controllo fondamentale della sicurezza in quanto autentica gli utenti e regola l'accesso a sistemi, reti e dati.
Il gestore delle identità concede agli utenti l'accesso e le autorizzazioni su una varietà di applicazioni e servizi, prevedendo la verifica delle identità degli utenti e dei diritti di accesso associati a un sistema specifico. Pertanto, le soluzioni IAM forniscono strumenti per gestire le identità digitali degli utenti e garantire un accesso adeguato alle risorse aziendali. [1]

Figura 2.1: Gestione dell'identità e degli accessi

L'"Identity and Access Management" può essere suddiviso in tre tipologie di servizi, come mostrato dall'immagine 2.1 [2]:

- Identity Management Services;
- Access Management Services;

- Directory Services;

Lo scopo dell'Identity and Access Management non è solo quello di ottimizzare e di garantire la sicurezza di un sistema, ma bensì deve essere capace di tener traccia di tutte le attività, considerate a rischio, effettuate dagli utenti. Questo viene fatto attraverso l'Auditing.
L'Auditing è il processo che si occupa di raccogliere tutte le informazioni riguardo la sicurezza nei sistemi IT, in modo da poter identificare "chi ha fatto cosa". Questo viene fatto per diversi scopi: investigativi, per individuare tentativi di accesso o comportamenti fraudolenti, per rispettare gli standard di legge, e di controllo, per verificare se il sistema risponde come dovrebbe.

2.2 Identity Management Services

Spesso le grandi organizzazioni sono costituite da più reparti, distribuite in varie regioni o nazioni, ed ognuna di queste sezioni può doversi trovare ad utilizzare sistemi IT differenti, trovandosi in grandi difficoltà in termini di gestione, costi e sicurezza.
L'Identity Management si occupa di risolvere queste problematiche, gestendo non solo la creazione, la modifica e l'eliminazione delle identità, ma occupandosi anche della distribuzione delle informazioni sui vari sistemi, chiamati "Sistemi Target".

Un'identità digitale è costituita da un identificatore univoco e da attributi che descrivono una persona, gruppo, dispositivo o servizio [2]. Pertanto, è sbagliato pensare alle identità digitali come astrazione di persone fisiche, in quanto queste possono essere anche oggetti o servizi, come ad esempio una stampate all'interno di un ufficio.
Il concetto di identità può essere mappato in un modello a tre livelli come mostrato in figura 2.2, anche se non comunemente utilizzato [3].

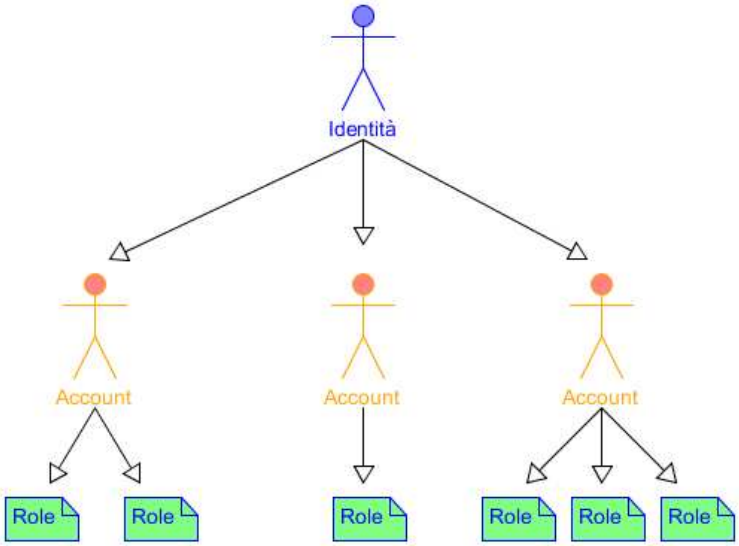

Figura 2.2: Modello a tre livelli

Il primo livello del modello rappresenta l'Identità, che è definita come una costante all'interno di un dominio.
Il livello successivo, costituito dagli Account, può essere visto come un'astrazione dell'identità in una determinata situazione; in altre parole, un Account

è la rappresentazione di una identità all'interno di una risorsa specifica. Come è facile pensare ogni Identità può avere accesso a più risorse e questo implica che ad ogni Identità si possono associare uno o più Account.
Il terzo ed ultimo livello sono i Roles, proprietà specifiche associate ad un Account. Ad ogni Account possono essere associati uno o più Roles. Il modello a tre livelli può essere visto come una descrizione object oriented di una identità.
In sintesi, ad un'Identità possono essere associati uno o più Account, e ad ogni Account possono essere associati uno o più Roles, e per la proprietà transitiva le Identità ereditano i Roles associati ai propri Account.

2.3 Access Management Services

L'Access Management è il processo di riconoscimento e controllo degli accessi alle risorse protette. Lo scopo è quello di garantire la sicurezza di un sistema impedendo accessi fraudolenti da utenti non autorizzati. Per fare ciò bisogna prima identificare l'utente che richiede una risorsa e poi verificare se questo ha i privilegi necessari.
Come mostrato in figura 2.3, gli Access Management Services si occupano di fare questo e possono essere suddivisi in due servizi principali:

- Authentication Services, si occupa di dimostrare "chi richiede cosa";
- Authorization Services, si occupa di determinare "chi può accedere a cosa";

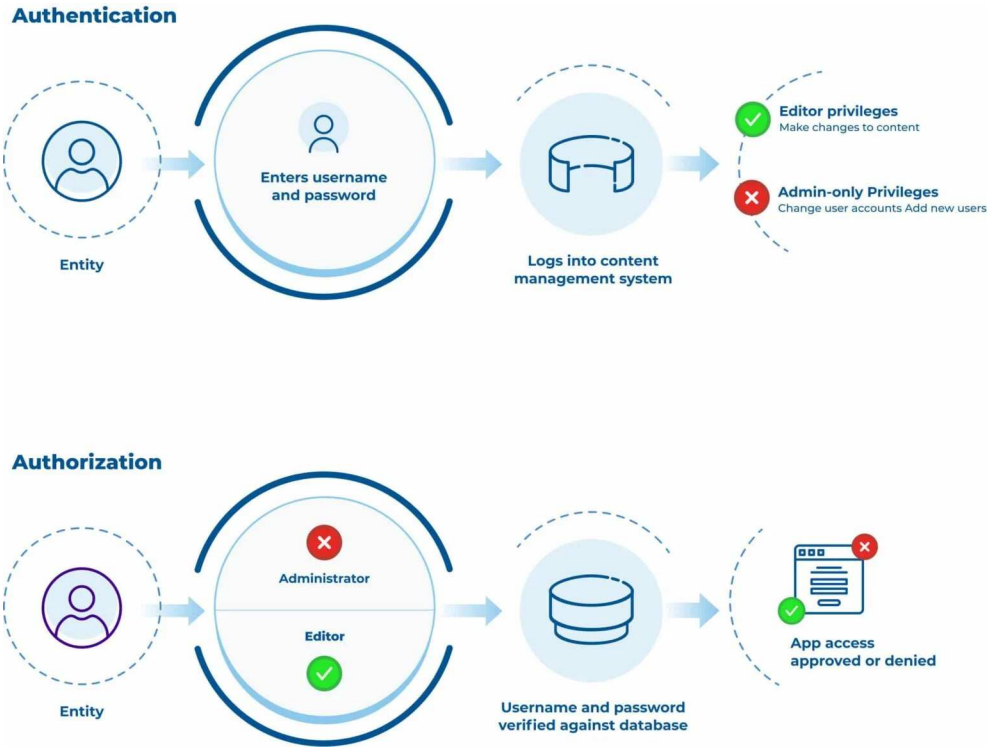

Figura 2.3: Authentication e Authorization Serivices

2.3.1 Authentication Services

L'autenticazione è il processo utilizzato per verificare l'identità di un utente. Questa si divide in due aree specifiche, l'identificazione e la gestione delle

sessioni. L'identificazione permette agli utenti di fornire le credenziali per poter ottenere un accesso iniziale al sistema o ad una particolare risorsa. Una volta che l'utente viene identificato viene creata una sessione e gli viene assegnata per tutta la durata dell'interazione con la risorsa, finché non effettua il log-out o la sessione termina (es timeout).

2.3.2 Single Sign-On (SSO)

Per sfruttare al meglio le sessioni, il modulo di autenticazione utilizzato può fornire il servizio di Single Sign-on (SSO), così da permettere all'utente di non dover effettuare ulteriori log-in per poter accedere ad altre risorse gestite dallo stesso ambiente di Identity and Access Management.

Il metodo più comune di autenticazione utilizzato sono l'username e la password. Per far sì che questo metodo sia affidabile la password deve essere lunga e composta da caratteri alfanumerici. In sistemi che richiedono un livello di sicurezza più elevato, ci si può trovare nella situazione di dover cambiare periodicamente questa password. Questo fa sì che con l'aumentare dei servizi, a cui doversi autenticare, ricordare le diverse credenziali di accesso con il tempo diventi sempre più complicato. Il Single Sign-on ha l'obiettivo di risolvere questo problema.

Il concetto base del SSO è quello di permettere ad un utente di autenticarsi un'unica volta, dopodiché la sessione creata autentica automaticamente l'utente su ogni altra risorsa presente come evidenziato in figura 2.4, a patto che abbia i privilegi di accesso appropriati [4].

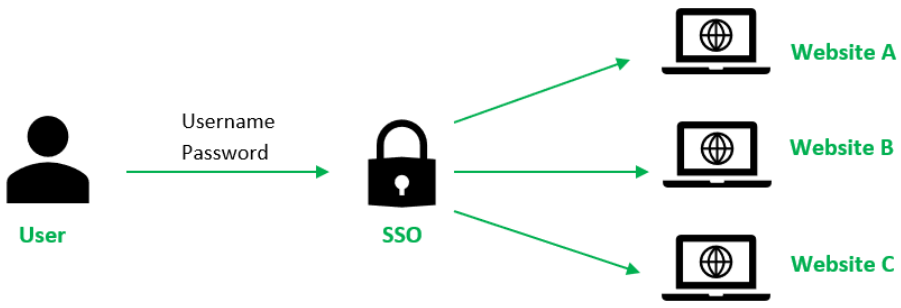

Figura 2.4: Sessione SSO

Solitamente il Single Sign-on è implementato tramite l'utilizzo di un SSO Agent, un servizio che riceve le richieste di autenticazione dell'utente ed esegue le opportune verifiche su un Directory Server.
Una volta eseguita la prima autenticazione l'agente mantiene i dati di accesso e li passa a tutte le applicazioni di cui l'utente ha bisogno.
Queste applicazioni a loro volta controllano, sui propri sistemi per la gestione degli accesi, le informazioni ricevute dal SSO Agent per convalidare le richieste.

2.3.3 Multi-Factor Authentication (MFA)

Definita anche 2FA o MFA (Multi-Factor Authentication), rappresenta un'ulteriore sicurezza ed è oggi uno dei sistemi di protezione più sicuro che si ha a disposizione per proteggere gli account.
L'autenticazione nell'Identity and Access Management avviene attraverso dei moduli che possono implementare uno o più dei classici metodi di autenticazione, come riportato anche in figura 2.5:

- Conoscenza: "Una cosa che sai", per esempio una password o il PIN.
- Possesso: "Una cosa che hai", come uno smartphone o un token di sicurezza.
- Inerenza: "Una cosa che sei", come l'impronta digitale, il timbro vocale, il viso, l'iride, o qualunque altro dato biometrico.

Figura 2.5: Autenticazione Multifattore

Dopo aver inserito la password (primo fattore) del proprio account, sarà richiesto di digitare un secondo fattore, che nella maggior parte dei casi è un

codice numerico. Questo secondo fattore in genere viene ottenuto attraverso lo smartphone (sotto forma di sms o tramite un'apposita applicazione) o tramite un token fisico.
A differenza della password, il secondo codice è di fatto inattaccabile, perché generato in maniera pseudocasuale secondo uno specifico algoritmo ed ha una durata molto limitata nel tempo (solitamente 30 secondi). Per questo motivo, lo si definisce anche OTP: "one time password".

Il secondo fattore può essere, in alternativa, di tipo biometrico ("una cosa che sei"). Ne abbiamo un esempio nelle applicazioni per smartphone che ci forniscono le banche: per aprire l'app ed anche per eseguire operazioni dispositive (es. fare un bonifico), ci viene richiesta la seconda autenticazione con l'impronta digitale o con il riconoscimento facciale [5].

2.3.4 Authorization Services

Dopo la fase di Authentication, il secondo step per l'accesso ad una risorsa è quello dell'Authorization. Questo step si occupa di determinare se un utente ha i privilegi necessari per accedere ad una risorsa o per effettuare una determinata azione.

Una volta che l'utente viene autenticato, l'Authorization permette di controllare se le richieste di accesso alle risorse, tipicamente sotto forma di URL nei sistemi web-based, rispettino le "Authorization Policy" (regole ben definite che descrivono "chi può accedere a cosa") e gli "Authorization Module" (formalismi utilizzati per descrivere le politiche di accesso) definiti nel motore IAM.
L'Authorization sfrutta modelli più o meno complessi che si possono basare su: ruoli utente / gruppi, le azioni intraprese, i canali di accesso, il tempo, le risorse richieste, dati esterni e le regole di business.
I modelli di controllo degli accessi possono essere suddivisi in tre classi principali [6]:

- Discretionary Access Control (DAC), si basa sull'identità e/o sull'appartenenza ai gruppi. Il concetto base è quello di dividere le risorse in "oggetti", ognuno di questi oggetti ha un "proprietario", che può concedere i diritti di accesso ad altri utenti per il proprio oggetto.

- Mandatory Access Control (MAC), permette agli amministratori di sistema, di assegnare ad ogni risorsa un "livello" di sicurezza. In tal

modo solo gli utenti con sufficienti privilegi possono accedere alle risorse protette.

- Role Based Access Control (RBAC), si basa sui ruoli e responsabilità assegnati agli utenti all'interno dell'organizzazione. I permessi di accesso sono associati ai ruoli, ad ogni utente sono associati uno o più ruoli specifici in modo da concedere i privilegi a seconda dei ruoli attivati. L'individuazione e l'assegnazione dei ruoli è uno dei problemi chiave specialmente per le grandi organizzazioni.

2.4 Directory Services

I Directory Services sono una componente fondamentale per l'Identity and Access Management, in quanto forniscono un repository centralizzato e distribuito per le identità e le risorse, che può essere utilizzato sia nelle reti intranet che extranet.
Un Directory Server permette di raccogliere quasi ogni tipo di dati, dai profili utenti, ai privilegi di accesso, alle informazioni su applicazioni e risorse di rete come stampanti, dispositivi di rete e molto altro. Le informazioni memorizzate in un Directory Server sono utilizzate per l'autenticazione e l'autorizzazione negli ambienti IAM.

2.4.1 Protocollo Lightweight Directory Access Protocol (LDAP)

La maggior parte dei Directory Services sono basati sul Lightweight Directory Access Protocol (LDAP), un protocollo che fornisce un linguaggio standard che sia le applicazioni client che quelle server possono utilizzare per comunicare tra loro [7].
Le applicazioni basate su LDAP sono in grado di cercare, aggiungere, eliminare e modificare le voci in modo semplice e veloce. Questo protocollo è una versione "leggera" del Directory Access Protocol (DAP) definito nello standard ISO/ITU-T X.500. Il protocollo DAP fornisce un accesso alla directory tramite un framework estensibile e robusto, ma ha un costo computazionale elevato, in quanto non utilizza il protocollo standard di Internet TCP/IP come presentato in figura 2.6.

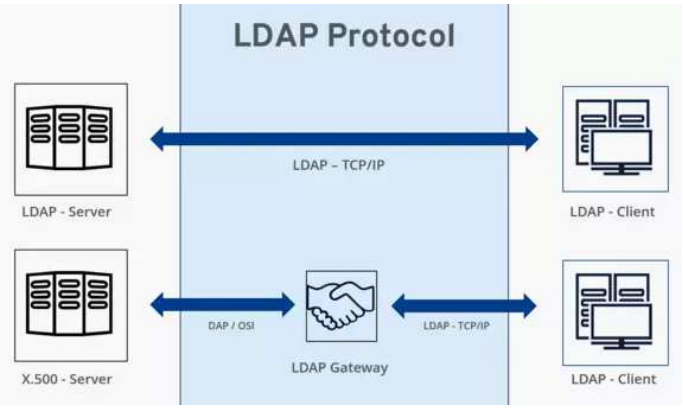

Figura 2.6: Panoramica protocollo LDAP

Il protocollo LDAP, quindi, conserva le caratteristiche di robustezza del protocollo DAP, riducendone i costi computazionali in modo da poter gestire milioni di voci con un investimento hardware e di rete relativamente basso.

Come mostrato nell'immagine 2.7, i Directory Server basati su LDAP hanno una struttura gerarchica ad albero, in cui l'accesso ai dati è gestito tramite le entry, una collezione di attributi che identifica univocamente un oggetto.

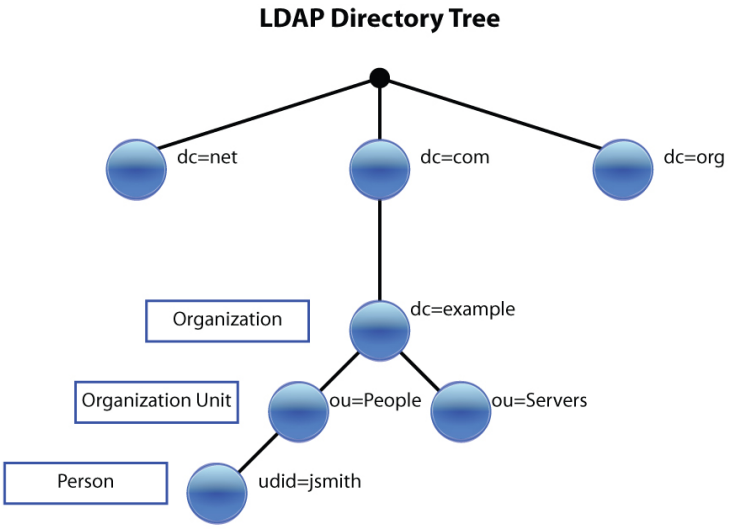

Figura 2.7: Alberatura LDAP

Per rappresentare una entry si utilizza il Distinguished name (DN), che è la concatenazione del Relative Distinguished Name (RDN), il nome di un attributo, seguito dai DN dei suoi predecessori, fino ad arrivare alla radice dell'albero. Solitamente la radice della struttura ad albero rappresenta il Paese, poi lo Stato, le organizzazioni e continuando così fino ad arrivare a rappresentare persone, documenti ed altro ancora.
In termini di Directory Service la sfida principale per l'Identity and Access Management è quella di riuscire ad ottenere un'organizzazione chiara di tutte le identità e di tutte le risorse, in modo da poter gestire gli accessi in modo rapido, sicuro ed affidabile.

2.5 Protocollo OpenID Connect

OpenID Connect è un protocollo di autenticazione che si basa su framework OAuth 2.0. OIDC permette alle applicazioni di terze parti di verificare l'identità degli utenti finali e di ottenere informazioni di profilazione degli utenti. Trattandosi di un protocollo standard, OpenID Connect consente ai client di tutti i tipi, compresi quelli basati su Web, dispositivi mobili e JavaScript, di richiedere e ricevere informazioni sulle sessioni autenticate e sugli utenti finali.
Il mondo dell'identità federata rappresenta, infatti, un evoluzione del concetto del single sign-on (SSO) centralizzato dell'azienda. L'utente ha un'unica identità digitale per accedere alle risorse dell'intero ecosistema digitale e, nel caso di risorse differenti ma in trust tra loro, effettuando un'unica autenticazione, e gli sarà consentito l'accesso al pool di queste risorse senza autenticarsi nuovamente.
Il modello astratto di riconoscimento dell'identità è sintetizzato nei seguenti otto passi, come anche mostrato in figura 2.8:

1. Il Client prepara una richiesta di autenticazione contenente i parametri di richiesta desiderati tramite lo User Agent.
2. Il Client invia la richiesta all'Authorization Server.
3. L'Authorization Server autentica l'End-User.
4. L'Authorization Server ottiene il consenso/autorizzazione dell'utente.
5. L'Authorization Server ritorna al Client un Access Token e, se richiesto, un ID Token.
6. Il Client richiede una risposta utilizzando l'Access Token al Token Endpoint.
7. L'Authorization Server valida l'Access Token e restituisce l'ID e l'Access Token.
8. Il Client convalida l'ID Token e recupera il Subject Identifier dell'utente.

OpenID Connect gestisce il processo di autenticazione per far connettere l'utente o per determinare se l'utente è già collegato. OIDC restituisce al Client il risultato dell'autenticazione, eseguita sull'Authorization Server in modalità sicura, cosicché il Client possa fare affidamento sul server per questa funzionalità [13].

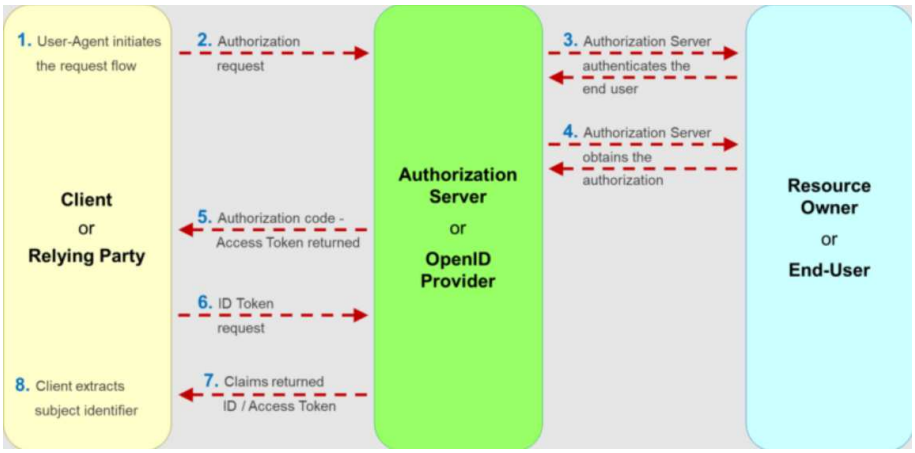

Figura 2.8: Flusso autenticazione OIDC

2.5.1 OAuth 2.0 (protocollo di autorizzazione alla base di OIDC)

l framework di autenticazione OAuth 2.0 è uno standard aperto che consente ad un utente di concedere ad un sito Web o ad un'applicazione di terze parti l'accesso alle proprie risorse protette, senza necessariamente rivelare a terzi le proprie credenziali o addirittura l'identità.
Tutto ciò è possibile perché OAuth 2.0 è un protocollo di autenticazione basato sui token (token-based authentication). Il token è una stringa, firmata da un server per le verifiche di integrità, che può contenere molte informazioni sull'utente, quali autorizzazioni, gruppi di appartenenza ed indicatori di tempo. L'utente mantiene l'accesso finché il token rimane valido.
Vengono eseguite una serie ben precisa di passaggi per accedere al server delle risorse dal proprio browser:

1. **Accesso**: utilizziamo il nome utente ed una password (credenziali) per dimostrare la nostra identità.

2. **Verifica**: il server autentica le credenziali ricevute ed emette il token d'accesso.

3. **Immagazzinamento**: il token viene inviato al browser per essere conservato localmente.

4. **Comunicazione**: ogni volta che si accede a qualcosa di nuovo sul lato server, il token viene sempre verificato nuovamente.

5. **Cancellazione**: al termine della sessione, il token viene eliminato.

Il flusso generale di funzionamento del protocollo di autorizzazione è sintetizzato dai seguenti sei passi, come anche mostrato in figura 2.9 [14]:

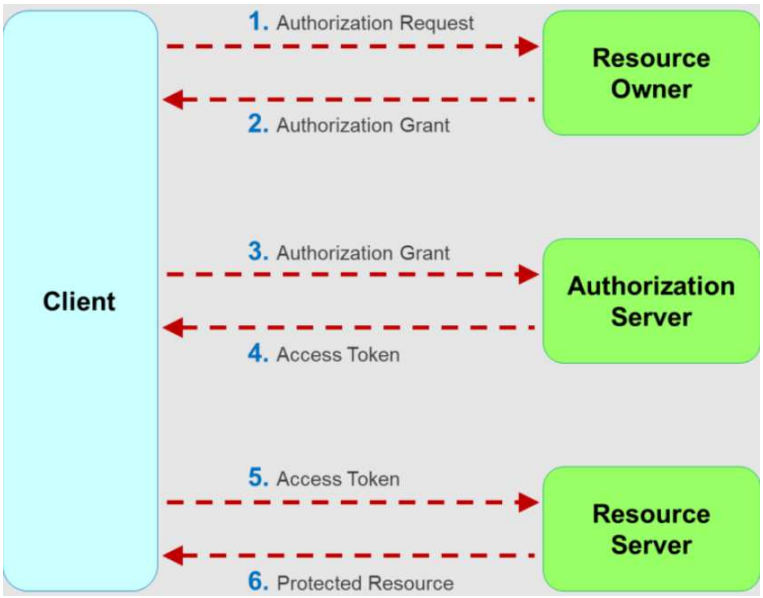

Figura 2.9: Flusso OAuth 2.0

1. Il Client invia una richiesta di autorizzazione al Resource Owner. Può avvenire direttamente o per tramite dell'Authorization Server che agisce come intermediario.

2. Il Client riceve una concessione di autorizzazione (authorization grant), che rappresenta una credenziale con l'autorizzazione del Resource Owner.

3. Il Client richiede un token di accesso autenticandosi sull'Authorization Server, presentando la concessione di autorizzazione.

4. L'Authorization Server autentica il Client, convalida la concessione di autorizzazione e, se valido, emette un token di accesso.

5. Il Client richiede la risorsa protetta dal server delle risorse, presentando il token di accesso.

6. Il Resource Server convalida il token di accesso e, se valido, soddisfa la richiesta.

2.6 Protocollo OpenID Connect for Verifiable Credentials (OID4VC)

In questa sezione viene introdotta la parte di ricerca legato al lavoro di tesi, ovvero "**OpenID Connect for Verifiable Credentials**".
OpenID Connect for Verifiable Credentials (OID4VC) è un insieme di specifiche sviluppate dal Gruppo di lavoro OpenID Connect, che mira a disaccoppiare l'emissione e la presentazione delle credenziali, consentendo agli utenti di presentare direttamente le informazioni di identità ai Verifiers.
Questo modello decentralizzato si basa sul concetto di Verifiable Credentials che è utilizzato per verificare l'identità degli utenti e che può rappresentare delle informazioni dell'utente finale, come identità del soggetto, autorità che ha convalidato le informazioni, ecc...
L'aggiunta di tecnologie, come le firme digitali, rende le credenziali verificabili più evidenti di manipolazioni e più affidabili rispetto ai loro equivalenti fisici.
Per far funzionare tutto l'ecosistema delle VC è necessario chiarire tre ruoli fondamentali:

- **Issuer**: chi rilascia le credenziali
- **Holder**: chi possiede le credenziali
- **Verifier**: chi verifica le credenziali

Attraverso l'uso di un Verifiable Data Registry (come rappresentato nell'immagine 2.10) è possibile effettuare operazioni di controllo e gestione delle chiavi.

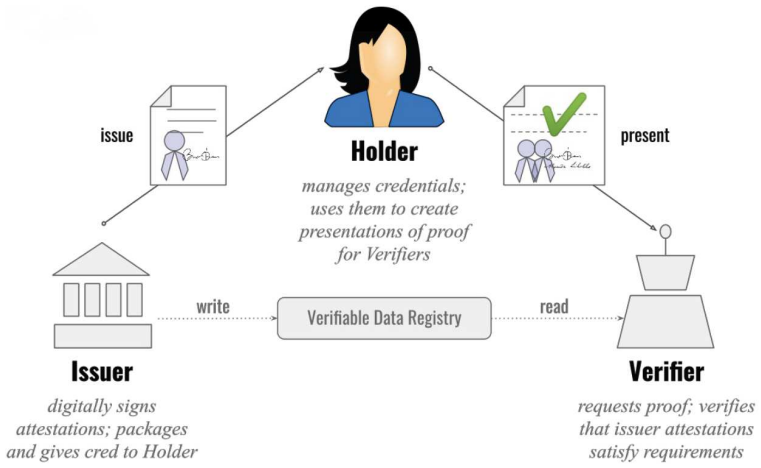

Figura 2.10: Verifiable Data Registry

Dopo il rilascio delle credenziali, l'utente è in grado di utilizzare tali credenziali per autenticarsi sui servizi che ne permettono l'utilizzo. In questo caso, le credenziali saranno inviate attraverso una Verifiable Presentation (come riportato nell'immagine 2.11), che è necessaria per verificare la proprietà sulla credenziale.

1. Il Controller (Verifier) invia una richiesta di verifiable presentation, che contiene gli attributo che devono essere rilasciati dall'Owner (Holder) per la verifica dell'identità;

2. L'Holder controlla la richiesta, andando a vedere se gli attributi richiesti sono in linea con le sue aspettative, e un framework cercherà all'interno del wallet l'attributo richiesto.

3. Se l'Holder possiede l'attributo e vuole condividerlo produce una Verifiable Presentation contenente la VC in questione e la sua firma in modo che il Controller ne possa verificare la proprietà.

4. Il Verifier consulta il verify registry per assicurarsi che le firme sono vere, e che le credenziali non siano state revocate.

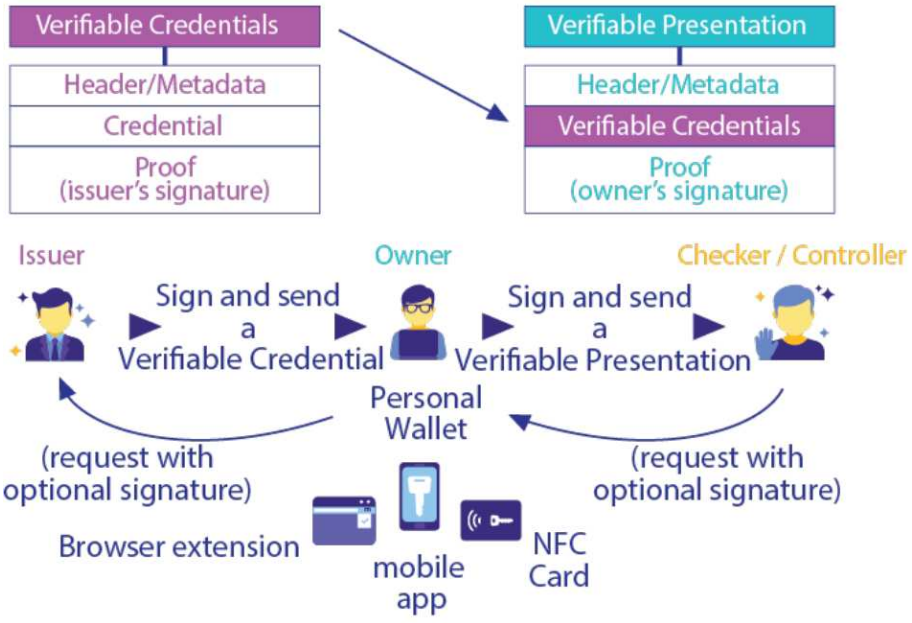

Figura 2.11: Verifiable Presentation

Ecco una panoramica delle tre specifiche che compongono OID4VC:

1. **OpenID for Verifiable Credential Issuance (OID4VCI)**: Questo protocollo definisce un'API denominata "Credential Endpoint" utilizzata per l'emissione di credenziali verificabili (Verifiable Credentials) che possono essere in vari formati, e i relativi meccanismi di autorizzazione basati su OAuth 2.0.
   In altre parole, stabilisce come vengono create e rilasciate le credenziali verificabili.

2. **OpenID for Verifiable Presentations (OID4VP)**: Questo protocollo estende OAuth 2.0 per consentire la presentazione di richieste sotto forma di credenziali verificabili durante il flusso di comunicazione.
   In altre parole, permette di mostrare le credenziali a chi ne ha bisogno, come parte del processo di autenticazione

3. **Self-Issued OpenID Provider v2 (SIOPv2)**: Questo protocollo consente agli utenti finali di utilizzare provider OpenID (OP) che controllano direttamente.
   In altre parole, gli utenti possono autenticarsi utilizzando Identity Provider (IdP, servizio che fornisce autenticazione e identificazione, esempio il sistema SPID) che gestiscono personalmente.

OID4VC è stato adottato in diversi contesti e casi d'uso, come il progetto EBSI [1], dove 18 portafogli nel progetto EBSI supportano le specifiche OID4VCI e OID4VP.

[1](European Blockchain Services Infrastructure)

2.7 Research Question

Considerando gli obiettivi e le motivazioni delineati nel Capitolo 1, insieme allo studio bibliografico appena condotto, la research question che guida questo lavoro di tesi è la seguente:

"Come può una federazione tra un gateway che espone servizi ed una soluzione che permette di gestire identità e accessi utente garantire un elevato livello di protezione dei dati sensibili e delle transazioni finanziarie per utenti non-consumer, come notai e commercialisti, senza compromettere l'usabilità e l'efficienza del sistema? Inoltre, in che modo l'adozione del protocollo OID4VC potrebbe migliorare ulteriormente la sicurezza e l'efficienza rispetto al protocollo OIDC?"

Questa domanda di ricerca permette di esplorare sia gli aspetti tecnici della sicurezza che quelli legati all'usabilità e all'efficienza, mantenendo il focus sulla soluzione proposta e considerando potenziali miglioramenti futuri.

Capitolo 3

Tecnologie utilizzate

In questo capitolo vengono descritti i principali applicativi utilizzati, escludendo quelli di contorno.
La prima sezione elenca le componenti Oracle utilizzate, appartenenti al pacchetto Oracle Identity and Access Management Suite Plus.
La seconda sezione descrive l'altra componente utilizzata, non appartenente a Oracle, ovvero Axway API Gateway.

3.1 Oracle Identity and Access Management Suite

La suite dei prodotti utilizzati è quasi interamente sviluppata da Oracle, e fa parte del pacchetto Oracle Identity and Access Management Suite Plus. L'utilizzo di questi prodotti permette di creare un'architettura scalabile ed in alta affidabilità per garantire massima sicurezza alle piattaforme di Business. Questi applicativi si integrano facilmente con qualsiasi altro sistema preesistente dal momento che fanno parte di una suite costruita ad hoc per l'Identity and Access Management, e sono quasi tutti interamente scritti in java [1].

3.1.1 Oracle Access Manager

Oracle Access Manager (OAM) è una componente della Oracle Access Management Suite, progettata per offrire una soluzione user-friendly, centralizzata, altamente scalabile ed ottimizzata rivolta alle aziende, con l'obiettivo di soddisfare la più ampia gamma di funzionalità rivolte alla gestione degli accessi. La componente, sviluppata con l'obiettivo di controllare l'accesso alle risorse, si compone essenzialmente di due fasi:

- Autenticazione;
- Autorizzazione.

L'autenticazione è un processo che ha lo scopo di dimostrare che un utente è chi sostiene di essere. Oracle Access Manager permette di autenticare gli utenti eseguendo un set predefinito di operazioni con lo scopo di verificare l'identità digitale dell'utente. Oltre a supportare le tipologie base di autenticazione, come username e password, OAM fornisce anche tipologie avanzate di autenticazione.
La fase di autenticazione, tramite le componenti OAM Access Server, Policy Manager e di plug-in nativi per web server, chiamati WebGates, permette di intercettare le richieste di accesso alle risorse, verificare una precedente autenticazione, convalidare le credenziali ed autenticare gli utenti.

Il flusso di identificazione di un utente viene gestito mediante l'Authentication Policy, un insieme di regole utilizzate per proteggere le risorse, a sua volta, si basa su uno o più Authentication Schema, che identificano la reale tipologia di autenticazione, ad esempio mediante l'utilizzo di un AD. Durante questo flusso di autenticazione è possibile svolgere diversi controlli sugli

utenti, mediante il Multi-Step Authentication Framework. Gli steps sono identificati da dei plug-in custom, ed ognuno di questi restituisce un valore di ritorno che a sua volta attiva o meno un altro step, fino ad arrivare alla fase di autorizzazione.

La fase di autorizzazione, invece, indica se un utente è autorizzato o meno ad accedere ad una risorsa, ed è gestita mediante l'Autorization Policy. Ogni politica di autorizzazione ha una o più condizioni che devono essere soddisfatte al fine di fornire all'utente l'accesso alla risorsa richiesta. Ad esempio due utenti con ruoli e privilegi di accesso differenti, possono richiedere la stessa risorsa, sarà poi compito di OAM servire differenti funzionalità tramite un'interfaccia web specifica, basata sugli attributi di ruolo.

Una versione base dell'architettura che utilizza OAM per gestire il flusso per l'accesso ad una risorsa protetta è mostrato nella figura 3.1

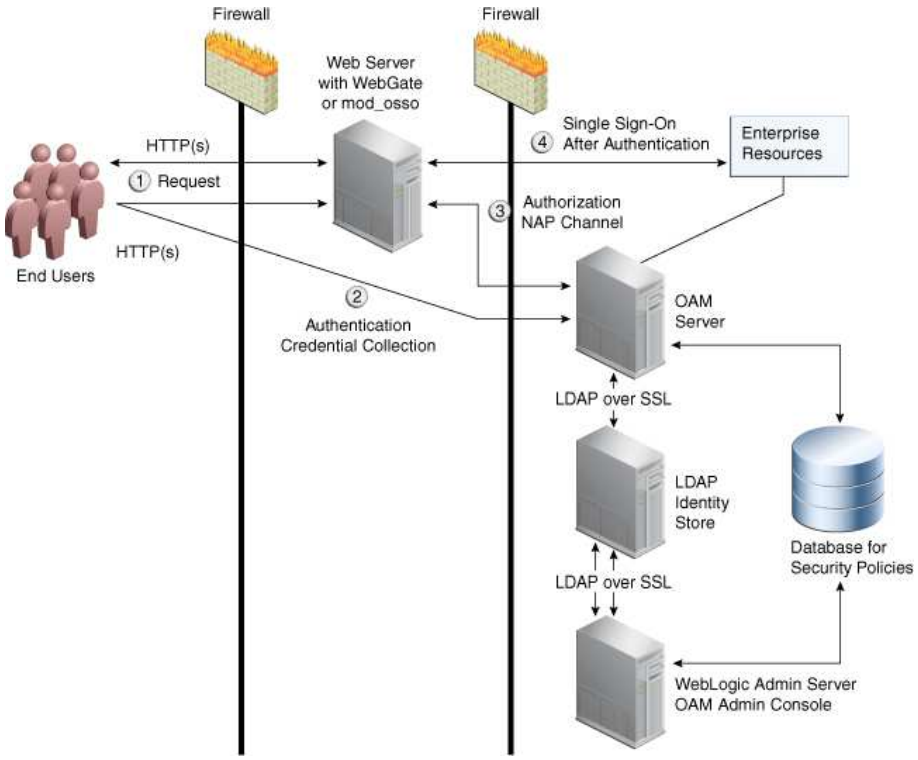

Figura 3.1: Architettura OAM Base

Quando un utente richiede una risorsa protetta il webgate ridirige la richiesta al server OAM che esegue tutti i controlli necessari. Nel caso in cui un utente

ha i diritti per accedere alla risorsa richiesta, OAM rimanda la richiesta al webgate che reindirizza l'utente alla risorsa.
La figura mostra un'architettura base, di seguito verranno mostrate architetture più complesse integrate con altri prodotti, come ad esempio Oracle Adaptive Access Manager, e verrà dettagliato, in modo più approfondito, il flusso delle richieste [9].

3.1.2 Oracle Http Server

Oracle Http Server è un Web Server che effettua operazioni di reverse proxy verso la componente Oracle Access Manager e su cui sono distribuite le risorse statiche dell'Identity Server.
Oracle HTTP Server sfrutta WebLogic Management Framework per fornire un ambiente semplice, coerente e distribuito per l'amministrazione di Oracle HTTP Server. Funziona come front-end HTTP che ospita le risorse statiche all'interno e sfruttando i plug-in proxy Oracle WebLogic Server integrati per instradare le richieste delle risorse dinamiche alle istanze del server gestito.
Oracle HTTP Server dispone dei seguenti componenti per gestire le richieste dei client:

- Listener HTTP, per gestire le richieste in entrata e instradarle all'utilità di elaborazione appropriata.
- Moduli (mod), per implementare ed estendere le funzionalità di base di Oracle HTTP Server. Molti dei moduli standard di Apache HTTP Server sono inclusi con Oracle HTTP Server.
- Interprete Perl, che consente di impostare Oracle HTTP Server come proxy inverso tramite il protocollo fcgi in un ambiente runtime Perl persistente.
- Plug-in proxy Oracle WebLogic Server, che consente a Oracle HTTP Server di eseguire il front-end dei server WebLogic.

Oracle HTTP Server può anche essere un server proxy, sia diretto che inverso. Un proxy inverso consente ai contenuti serviti da server diversi di apparire come se provenissero da un server [8].

3.1.3 Oracle Unified Directory

Oracle Unified Directory (OUD) è un servizio di Directory sviluppato appositamente per l'integrazione con i prodotti di Identity and Access Management

di Oracle. È basato sullo standard LDAP, e vanta una semplicità di gestione e configurazione, e alta affidabilità.
A differenza dei più comuni Directory Server è stato progettato per massimizzare anche le operazioni di scrittura, e non solo quelle di ricerca e di lettura.
L'alta affidabilità di questo prodotto, utilizzato anche da organizzazioni finanziarie dove la percentuale di affidabilità deve essere alta, è data anche dalla sua scalabilità. Per garantire questo livello di prestazioni non basta utilizzare delle semplici repliche ridondanti su un unico server, ma bensì bisogna sviluppare un'architettura, in cluster, distribuita su macchine geo localizzate in posizioni diverse, in modo da creare istanze multiple su data-store indipendenti. Una volta configurato sarà lo stesso OUD, in modo autonomo, a gestire l'indicizzazione dei dati e i backup, in modo da evitare perdite di dati [10].

3.2 Axway API Gateway

L'Axway API Gateway è una soluzione software che permette di gestire e mettere in sicurezza le API di qualunque organizzazione, in modo da abilitare rapidamente l'accesso a servizi Cloud, Mobile e SOA, come evidenziato dalla figura 3.2. Si integra con facilità in architetture molto complesse, con lo scopo di rendere disponibili servizi, verso dispositivi mobili senza richiedere modifiche alla logica dei servizi stessi. Questo si frappone fra i client e il dominio da proteggere, all'interno della DMZ[1] in modo da rendere sicuri Web Services e altri servizi [11].

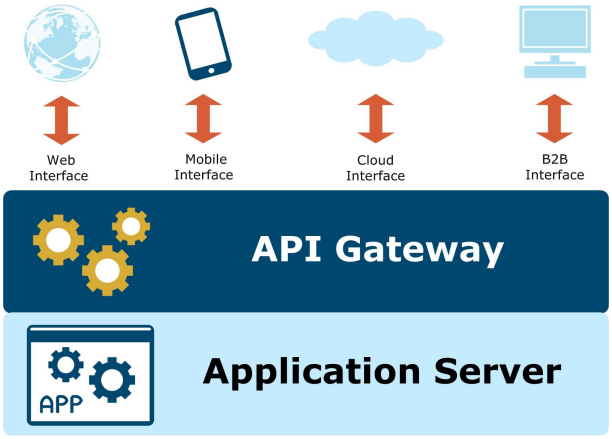

Figura 3.2: Ruolo Axway API Gateway

I principali servizi supportati da API Gateway sono:

- Sicurezza, fornisce autenticazione e autorizzazione;
- Reverse proxy, collegamento con servizi esterni appartenenti ai backend;
- Data transformation, modeling dei messaggi SOAP/REST in entrata o in uscita ai servizi esposti.

Per gestire le funzionalità di Axway API Gateway viene utilizzato Policy Studio, che è uno strumento grafico che consente di virtualizzare le API e sviluppare policy (ad esempio, per applicare requisiti operativi, di sicurezza e di conformità).

[1] Delimitarized Zone

3.2.1 Cassandra: NO-SQL Database

Cassandra è un NO-SQL Database che gestisce le configurazioni dell'Axway API Gateway.
Apache Cassandra utilizza il sistema NoSQL al posto del tradizionale DBMS relazionale perché quest'ultimo non è adatto a gestire grandi volumi di dati non strutturati. Cassandra fornisce archivi altamente scalabili e affidabili di enormi set di dati, offrendo un'elevata disponibilità senza un punto di errore singolo ed ha la capacità di sopravvivere a intere interruzioni del data center [12].

Capitolo 4

Sistema proposto

In questo capitolo verrà esaminato il sistema proposto e il suo funzionamento con un livello di dettaglio tecnico approfondito.
La prima sezione introduce lo scenario di utilizzo, delineando l'obiettivo finale del sistema.
La seconda sezione illustra l'architettura del sistema, descrivendo ogni componente e la sua topologia.
La terza sezione dettaglia i flussi di autenticazione che possono essere attivati dal sistema.
La quarta sezione descrive l'installazione e la configurazione delle componenti OAM e Axway API Gateway.
Infine, la quinta, sesta e settima sezione offrono un'analisi dettagliata dei modelli Federated Identity (FI) e Self-Sovereign Identity (SSI), con tutte le loro componenti e protocolli, per poi effettuare un confronto tra i due modelli.

4.1 Scenario

Il sistema proposto è stato sviluppato per un cliente dell'azienda dove lavoro e, attualmente, le infrastrutture di sicurezza esistenti si dividono per contesti di business:

- Infrastruttura di Federazione:
  Dedicata agli accessi ad applicativi interni da parte di operatori di back-office delle Banche.

- Infrastruttura di accesso Multi-istituto:
  Dedicata agli accessi all'Home Banking per i clienti delle banche.

- Infrastruttura di accesso singolo istituto:
  Dedicata alla messa in sicurezza delle API REST/SOAP esposte dai web service di una determinata banca.

Si rende necessaria la realizzazione di una nuova infrastruttura di autenticazione e controllo degli accessi dedicata ad un nuovo portale 'Multi-Banca', rivolto a diverse tipologie di utenti non-consumer, come Notai e Commercialisti.
Per garantire la presenza di una nuova infrastruttura dedicata a nuovi contesti di Business la soluzione preferenziale è quella di realizzare una nuova infrastruttura di sicurezza, basata sulle componenti Oracle Access Manager e Axway API Gateway. Tale infrastruttura si baserà nativamente sullo standard di autenticazione OpenID Connect ed il flusso è il seguente, come viene evidenziato in figura 4.1:

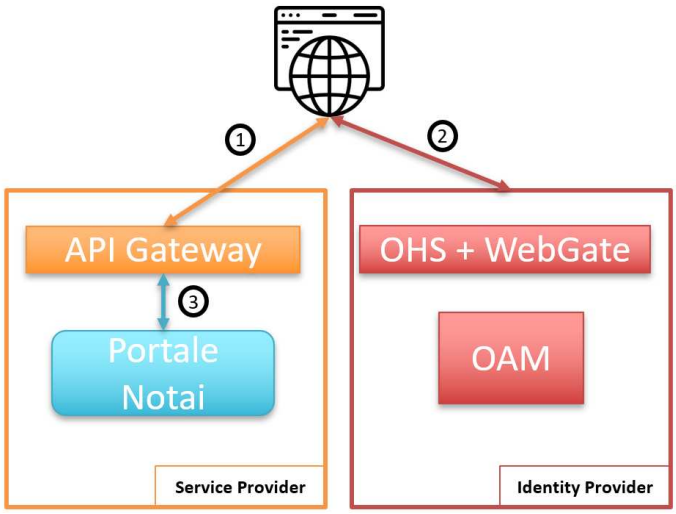

Figura 4.1: Flusso di autenticazione Federato(OIDC)

1. L'utente prova ad accedere all'applicativo esposto dall'API GW ed innesca il flusso di federazione, che lo rimanda sull'Identity Provider per l'autenticazione.

2. L'infrastruttura OAM valida le credenziali di primo e secondo fattore e rimanda l'utente sull'API GW.

3. L'API Gateway costruisce la chiamata verso l'applicativo inserendo le informazioni di contesto utente all'interno della request (es. jwt utente, header specifici, ecc..).

4.2 Architettura del sistema

4.2.1 Componenti architetturali

Nella tabella seguente è riportato l'elenco e la descrizione dei componenti interessati dall'infrastruttura:

Componente	**Descrizione**
Enduser	Utente che effettua il login per accedere nell'area privata del Portale. Nel flusso OpenID Connect assume il ruolo di Resource Owner
Portale Utenti Esterni API	Layer di servizi di backend esposti dall'applicazione Portale Utenti Esterni
Identity Server API	Layer di API che espone i servizi necessari per l'invio e la convalida dell'OTP e per la gestione del ciclo di vita utente (sign up, recupero password)
Portale Utenti Esterni UI	Applicativo web che si interfaccia con le API dell'applicazione Portale Utenti Esterni. Nel flusso OpenID Connect assume il ruolo di Relying Party.
Identity Server UI	Applicativo web per la gestione dei processi di login e gestione del ciclo di vita utente (sign-up, recupero password)
OHS	(Oracle HTTP Server) Web Server che effettua operazioni di reverse proxy verso la componente Oracle Access Manager e su cui sono distribuite le risorse statiche dell'Identity Server.
APIGW	(Axway API Gateway) Questa componente si occupa di esporre le risorse del Portale Utenti Esterni e della gestione della sessione utente. Nel flusso OpenID Connect assume il ruolo di Resource Server.
OAM	(Oracle Access Manager) Componente che gestisce l'autenticazione di primo e secondo livello. Nel flusso OpenID Connect assume il ruolo di Authorization Server.
OUD	(Oracle Unified Directory) Directory server che gestisce le utenze del Portale Utenti Esterni insieme alle password policy definite per la credenziale di primo fattore
Cassandra	NO-SQL Database che gestisce le configurazioni dell'APIGW

4.2.2 Topologia

Nella figura 4.2 viene riportata la topologia dell'infrastruttura.

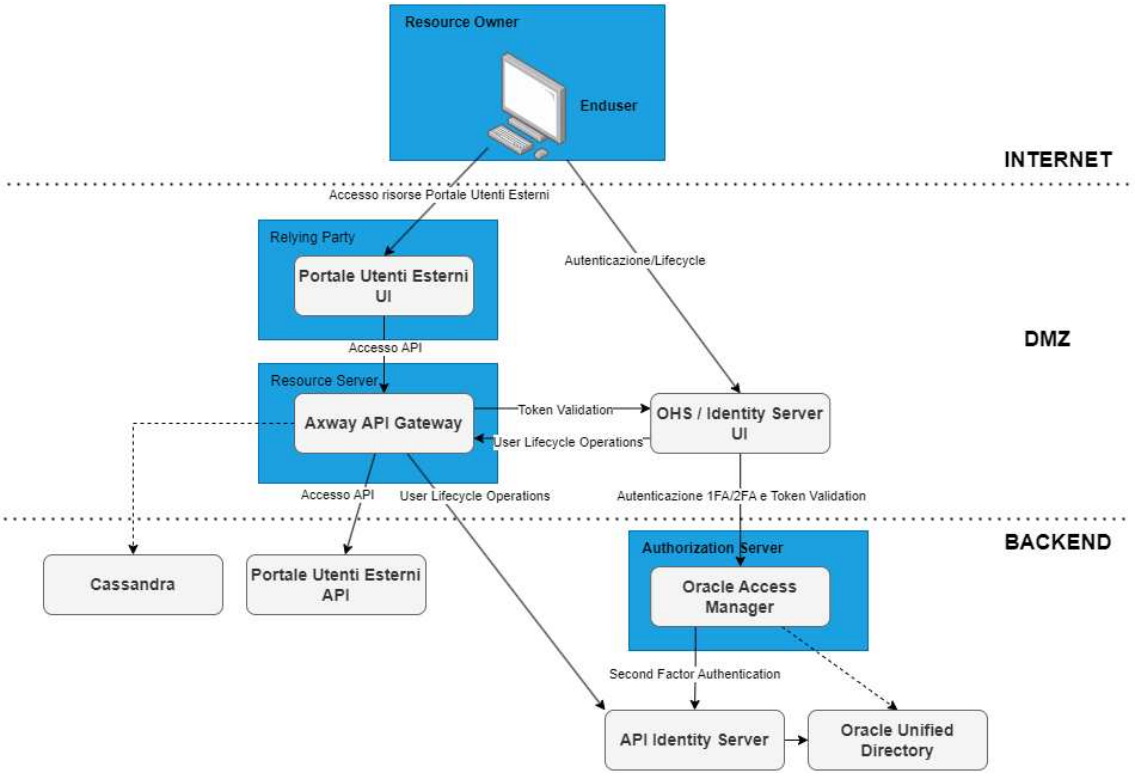

Figura 4.2: Topologia Sistema PUE

4.3 Processo di autenticazione

Il processo di autenticazione si basa sul protocollo OpenID Connect. Nel dettaglio, viene realizzata una federazione OIDC tra la componente Axway API Gateway, con la funzione di Resource Server, e Oracle Access Manager come Authorization Server.

4.3.1 Attori

Sono elencati di seguito i principali attori del processo di autenticazione:

Attore	**Descrizione**
Oracle HTTP Server	Web Service che effettuano reverse proxy alle componenti Oracle Access Manager
Portale Utenti Esterni API	Backend API del Portale Utenti Esterni
Portale Utenti Esterni UI	Componente che espone le pagine web per l'accesso alle API del Portale Utenti Esterni
Axway API Gateway	Resource Server OpenID Connect che espone le risorse del Portale Utenti Esterni e verifica gli accessi alle risorse protette
Cassandra	Database No-SQL adottato da Axway API Gateway per conservare gli authentication token e le configuration property
Oracle Access Manager	Authorization Server OpenIDConnect per la convalida delle credenziali utente di primo e secondo fattore
Oracle Unified Directory	LDAP Server contenente le utenze di Portale Utenti Esterni
Identity Server API	Componente che espone le funzionalità di multi factor authentication e di gestione del ciclo di vita utente
Identity Server UI	Componente che espone le pagine web per la login e la gestione del ciclo di vita utente
Browser	Client Web tramite cui l'utente effettua la login per accedere nell'area privata del Portale Utenti Esterni

4.3.2 Flusso di autenticazione

Di seguito è illustrata la modalità del processo di autenticazione web che termina con successo. I token di autenticazione vengono rilasciati in seguito al processo di autenticazione e si dividono in:

- **Identity Token**, ha lo scopo di provare all'applicazione client l'avvenuta autenticazione dell'utente sull'Authorization Server. Questa tipologia di token può contenere informazioni base del profilo utente e possono essere usate dal client per migliorare la User Experience.
- **Access Token**, viene utilizzato dall'applicativo client per accedere alle API protette esposte dal Resource Server.
- **Refresh Token**, viene utilizzato per generare un nuovo Access Token.

Use case diagram In figura 4.3 viene mostrato lo use case diagram che illustra le interazioni tra un utente finale (enduser) e vari componenti del sistema, tra cui OHS, OUD, OAM e API Gateway.

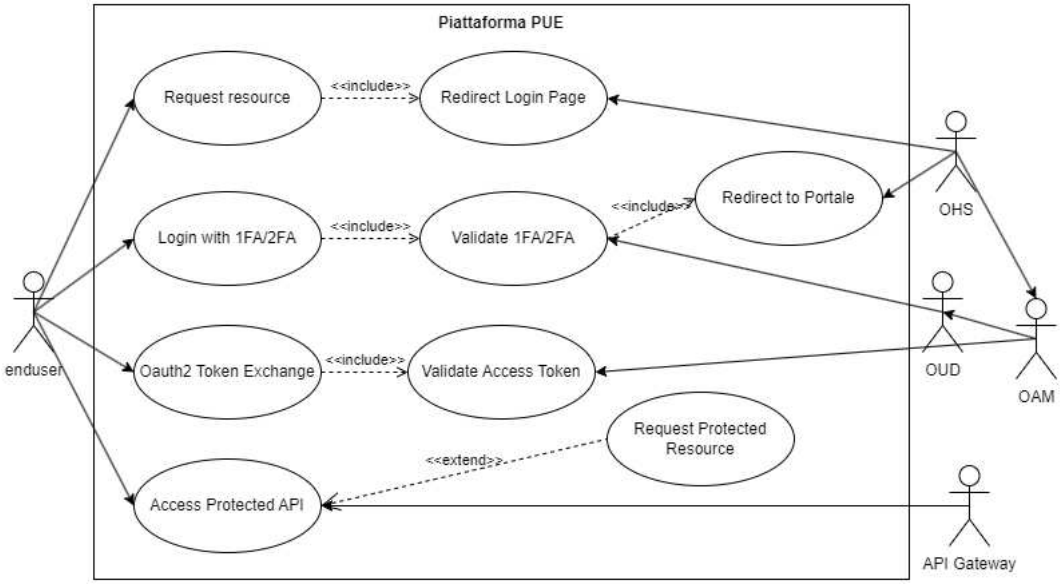

Figura 4.3: Use Case flusso di Login

Sequence diagram In figura 4.4 viene mostrato il sequence diagram del flusso di autenticazione web:

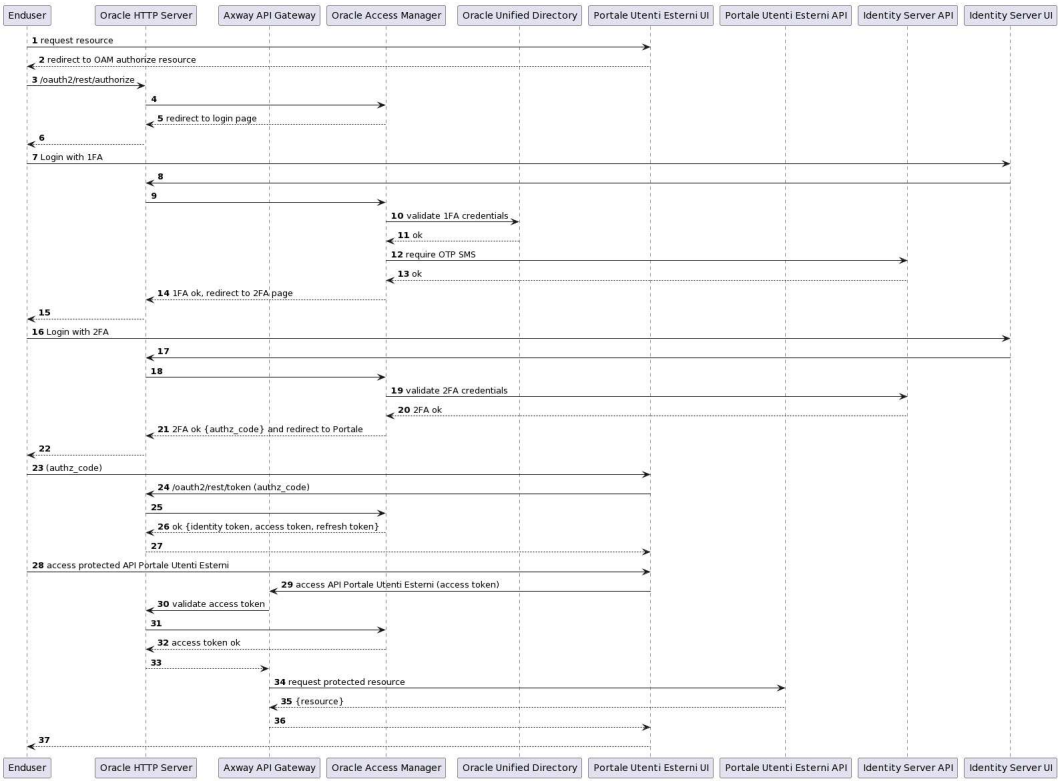

Figura 4.4: Flusso di autenticazione Web

Il flusso indicato funziona più precisamente nel seguente modo:
1. Il flusso inizia quando l'utente, tramite Portale Utenti Esterni UI, richiede una risorsa protetta.
2-6. In questo modo, viene invocato l'Authorization Server facendo ridirigere l'utente verso la pagina di login presente sull'Identity Server UI.
7-9. L'utente inserisce le credenziali di primo fattore che verranno convalidate da OAM nel caso in cui le credenziali sono presenti su OUD.
10-13. Una volta eseguita l'autenticazione di primo livello (username/password), si richiede un secondo fattore di autenticazione, viene quindi invocato l'IdP API per la generazione e l'invio di un OTP per l'utente tramite SMS.
14-18. Successivamente, OAM genera un redirect verso un'altra pagina di login, dove l'utente inserisce le credenziali di secondo fattore (OTP).
19-23. Viene convalidato l'OTP invocando l'IdP API e risponde con una redirect verso il Portale Utenti Esterni e con l'authorization code.

24-29. Terminata l'autenticazione con primo e secondo fattore, Oracle Access Manager rilascia all'applicativo Portale Utenti Esterni i token di autenticazione (Identity Token, Access Token e Refresh Token) per la profilazione dell'utente e per gli accessi verso le API protette di Portale Utenti Esterni, passando come Authorization Header l'access token.
30-37. Questi ultimi accessi sono intermediati da Axway API Gateway che si occupa di verificare la validità/scadenza del token e intercettare le richieste verso il Portale Utenti Esterni API restituendo il contenuto della risposta al chiamante.

4.3.3 Refresh Token

Il seguente flusso descrive il processo di generazione di un nuovo Access Token utilizzando il Refresh Token emesso durante l'autenticazione da parte di Oracle Access Manager.

Sequence diagram In figura 4.5 viene mostrato il sequence diagram del flusso di generazione di un nuovo Access Token:
Le fasi del processo sono le seguenti:

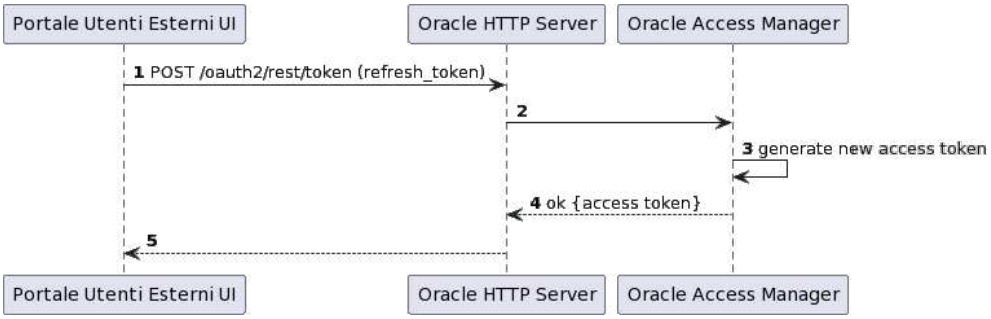

Figura 4.5: Flusso Refresh Token

1-2. Il Portale Utenti Esterni UI invoca la risorsa /oauth2/rest/token passando nel body della richiesta il refresh token.
3. L'OAM verifica la validità del refresh token e genera un nuovo access token.
4-5. OAM restituisce al chiamante il nuovo access token.

4.3.4 Revoke Token

Il seguente flusso descrive il caso in cui viene richiesta la revoca dell'access token e del refresh token precedentemente generati da Oracle Access Manager.

Sequence diagram In figura 4.6 viene mostrato il sequence diagram del flusso di revoca dell'Access Token:
Le fasi del processo sono le seguenti:

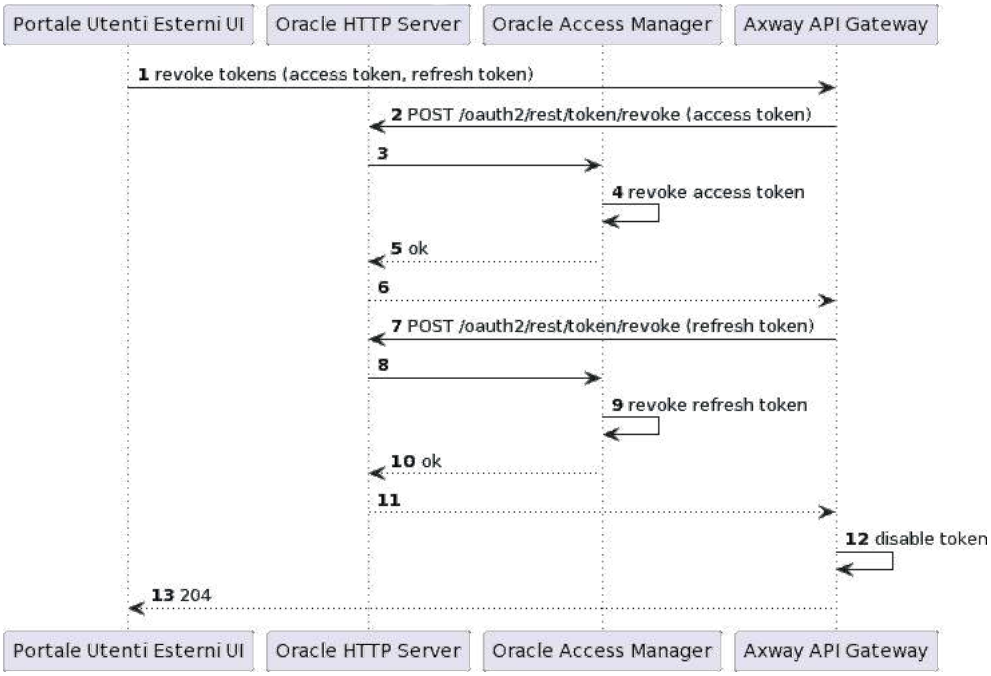

Figura 4.6: Flusso Revoke Token

1. Il Portale Utenti Esterni UI invoca la risorsa /gw-restpue/oauth2/rest/token/revoke di Axway API Gateway passando nel body della richiesta l'access token e il refresh token.
2-6. L'API Gateway richiede la revoca dell'access token ad OAM.
7-11. L'API Gateway richiede la revoca del refresh token ad OAM.
12-13. L'API Gateway disabilita la validità dell'access token per l'invocazione delle risorse protette e risponde con un codice 204 al chiamante.

4.4 Installazione e configurazione tecnologie

Per l'installazione delle componenti, è stata principalmente utilizzata la guida utente di Oracle.
Non si entrerà nei dettagli, verranno descritte le configurazioni delle componenti utilizzando esempi semplificati, anziché le configurazioni reali, per motivi di privacy.

4.4.1 Componente Oracle HTTP Server (OHS)

L'OHS è la prima componente da configurare, siccome, essa consente il reverse proxy, ovvero inoltra le richieste dei client (come browser) ai server di destinazione (come servizi API) e restituisce le risposte ai destinatari.
Quando si configura un reverse proxy per OAM, è necessario definire un Virtualhost sull'OHS, ovvero una configurazione nel server proxy che associa un nome di dominio/indirizzo IP a un'applicazione/servizio specifico.

Per configurare un Virtualhost bisogna andare sul file system della macchina su cui è installato l'OHS e navigare nella cartella contenente i moduli di configurazione (di solito contenuti nella cartella moduleconf).
Di seguito viene riportato un esempio di configurazione e spiegato cosa permette di fare:

ProxyPass /foo/ http://app1.domain.com/
ProxyPassReverse /foo/ http://app1.domain.com/

La direttiva 'ProxyPass' istruisce il server OHS a inoltrare tutte le richieste che iniziano con '/foo/' a un'altra destinazione, ovvero 'http://app1.domain.com/'.
Quindi, quando il server OHS riceve una richiesta per un URL che inizia con '/foo/', inoltrerà la richiesta a quell'URL di destinazione.
La direttiva 'ProxyPassReverse' è correlata alla direttiva 'ProxyPass'. Quando il server di destinazione restituisce una risposta, OHS modifica gli header della risposta in modo che corrispondano all'URL originale richiesto dal client. Questo è importante per garantire che le risposte siano correttamente indirizzate al client.

Una volta fatto ciò, bisogna creare un Agent e distribuire gli artifacts su webgate.
L'Agent è un componente software che si trova sul server Web, ad esempio su Oracle HTTP Server, e comunica con il server OAM per gestire l'autenticazione e l'autorizzazione degli utenti.

Come visualizzato in figura 4.7, per creare un Agent è necessario accedere alla Console OAM, andare nella sezione 'Agents' e cercare la configurazione di interesse. Modificare i valori esistenti e fare clic su Applica, questo rigenererà l'agente. Fatto questo, scaricare la configurazione appena modificata.

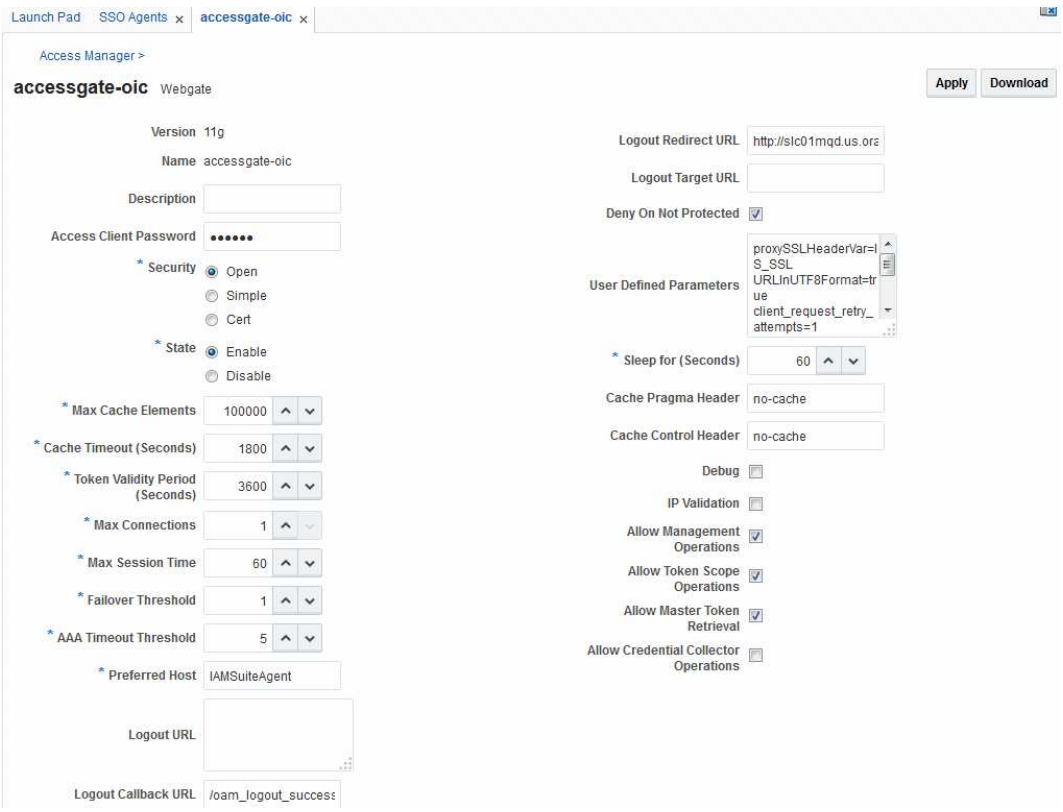

Figura 4.7: Configurazione Agent OAM

Dopo aver scaricato la configurazione dell'agente, è necessario copiare gli artefatti su ciascun server Web in cui è installato il WebGate. Fatto ciò, riavviare l'Oracle HTTP Server per applicare le modifiche.

4.4.2 Componente Oracle Access Manager (AOM)

Per quanto riguarda la componente OAM, come in figura 4.8, bisogna configurare l'IDS Store Profile (LDAPS), permettendo di gestire gli identity store degli utenti.
Lo User Identity Store (IDS) è un repository LDAP centralizzato in cui ven-

gono archiviate e mantenute in modo organizzato le informazioni sugli amministratori e gli utenti.

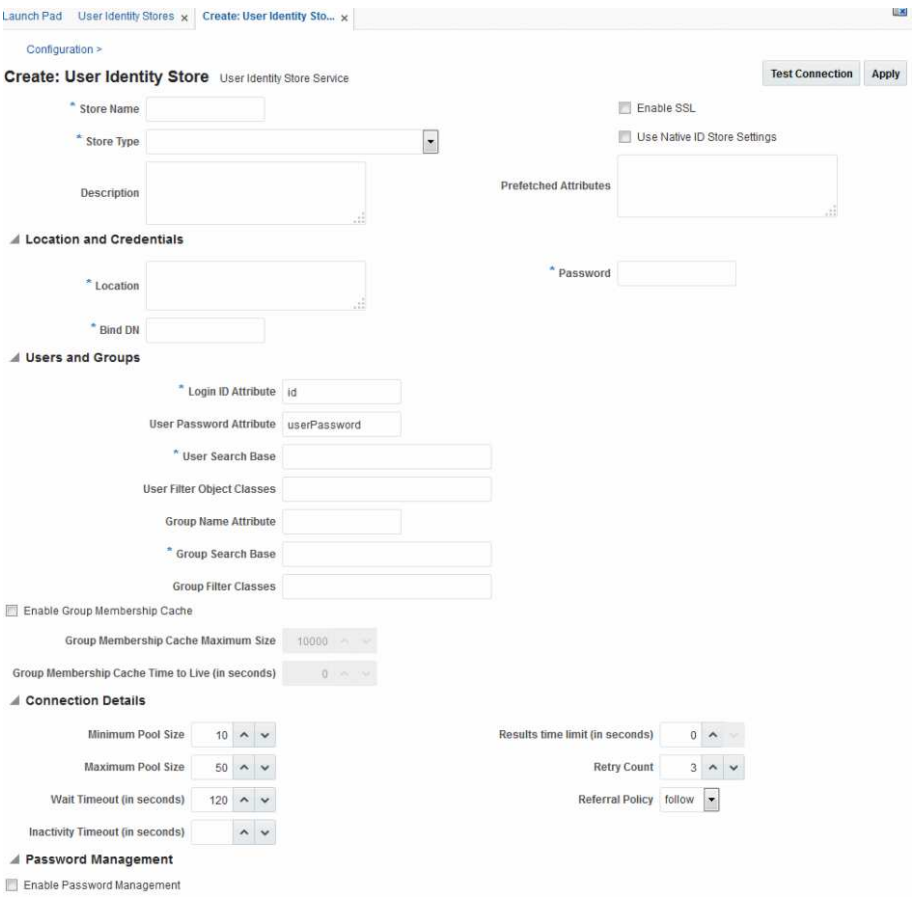

Figura 4.8: Configurazione IDS Store Profile su Console OAM

Una volta impostato lo Store Profile, bisogna inserire la CA[1] del ciente nel certstore di OAM, in modo da abilitare la comunicazione SSL tra Oracle Access Manager e i directory server che contengono le informazioni di profilazione.
Una volta ottenuta la CA cert, andare sulla macchina contenente la componente OAM e caricare il certificato nella cartella adeguata (di solito chiamata 'security'), successivamente, utilizzare lo strumento di gestione delle chiavi e dei certificati chiamato *keytool* presente sul file system, lanciando il comando:

[1]Certification Authority

./keytool -import -trustcacerts -keystore cacerts -storepass changeit -noprompt -alias "CA" -file /home/oracle/cert/ca.cer

In poche parole, questo comando importa il certificato specificato e indica che deve essere considerato attendibile[2]. Questo consentirà al certificato di essere utilizzato per autenticare le connessioni SSL che utilizzano questo keystore.

Infine, bisogna configurare OAM in modo da supportare il protocollo di autenticazione OpenID Connect (OIDC). È necessario abilitare il servizio OpenIDConnect da OAMConsole, come evidenziato in figura 4.9.

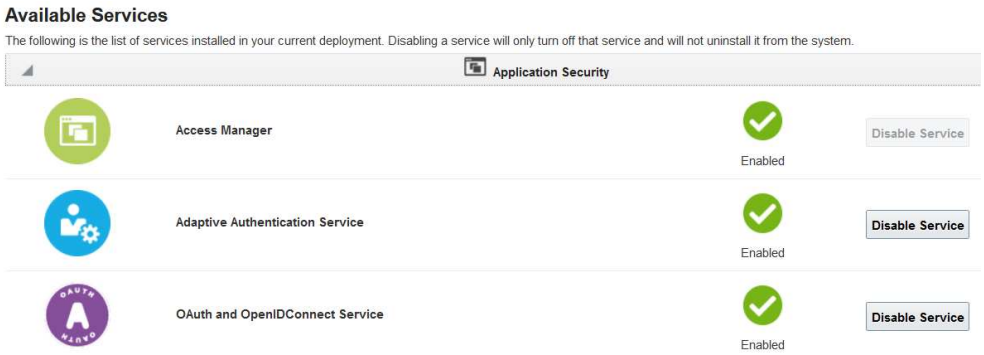

Figura 4.9: Abilitazione OIDC da OAM Console

Successivamente, creare un Authentication Module Custom inserendo l'ordine dei plugin che devono essere richiamati in base agli steps, come mostrato in figura 4.16.

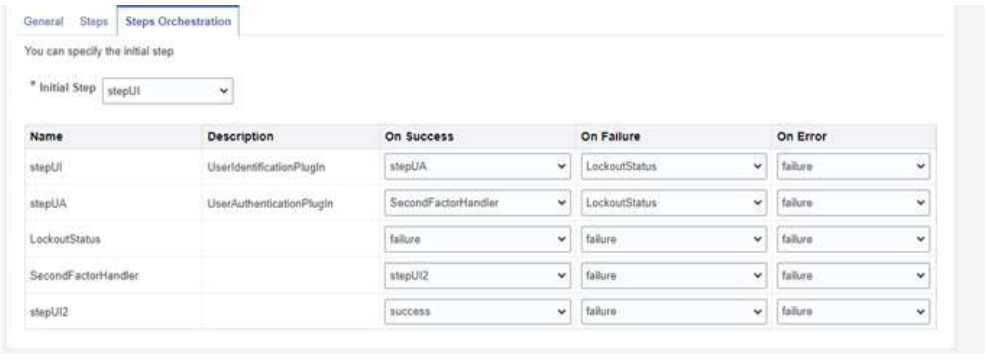

Figura 4.10: Steps Orchestration plugin

[2]fidato

In questo modo, in base alla fase in cui si trova il flusso di autenticazione, verranno attivati i plugin specificati.

4.4.3 Componente Axway

L'ultima componente da configurare è l'Axway API Gateway, che permette di esporre API di backend e microservizi, e permette di proteggere, applicare sicurezza e garantire scalabilità e alta disponibilità.
Per impostare Axway API Gateway bisogna configurare le sue componenti tramite Axway API Manager e Policy studio, che sono strumenti essenziali per la gestione del ciclo di vita delle API web, permettono di impostare e personalizzare le istanze del Gateway API e i listener associati, e permettono lo sviluppo di policy che regolano il comportamento del Gateway.

In figura 4.11 viene mostrata la prima policy che permette di verificare la validità del token passato.
Se il token è valido, la richiesta viene inoltrata alle risorse protette. In caso contrario, viene restituito un errore o la richiesta viene bloccata.

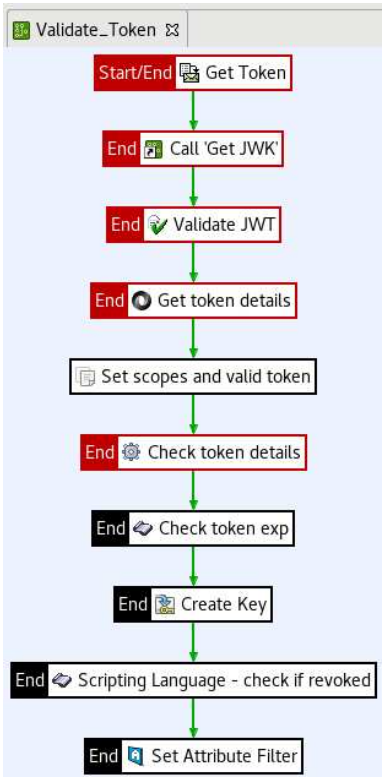

Figura 4.11: Policy che verifica la validità del token passato

La policy illustrata, estrae il token dall'HTTP Header e valida il JWT[3]. Per verificare la validità del JWT, viene decodificato il token per ottenere l'header e il payload, verifica la firma utilizzando la chiave segreta o la chiave pubblica corrispondente ed infine controlla le affermazioni nel payload per assicurarsi che siano valide (ad esempio, data di scadenza, emittente, ecc.). Successivamente, crea una chiave per il token che verrà conservata nella cache del gateway. I JWT vengono spesso utilizzati per l'autenticazione in flussi OAuth 2.0 e OpenID Connect, come in questo caso.

In figura 4.12 viene mostrata la seconda policy che permette di avviare la procedura di revoca del token passato. Durante il flusso di Revoke Token è l'Axway API Gateway che deve richiedere la revoca dell'access token e del refresh token ad OAM. In questo modo, disabilita la validità dell'access token per l'invocazione delle risorse protette e risponde con un codice 204 al chiamante.

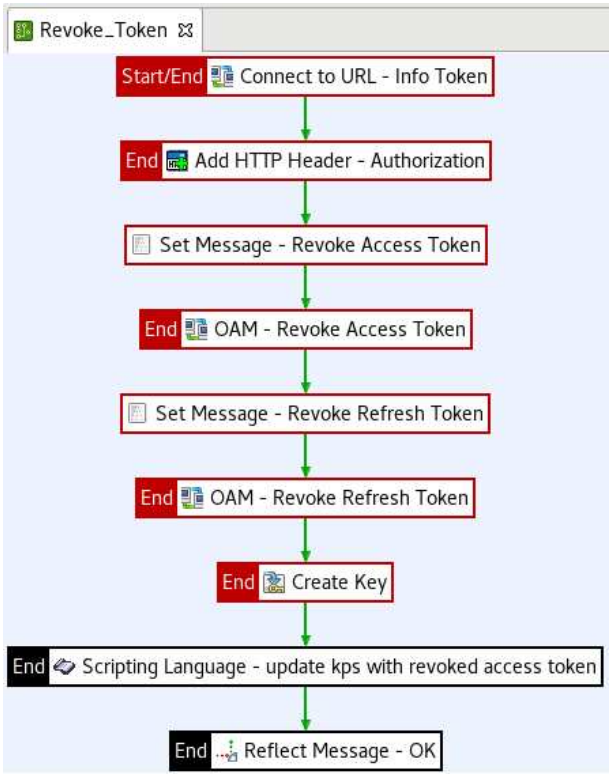

Figura 4.12: Policy che avvia la revoca del token passato

La policy illustrata, per prima cosa, prende le informazioni necessarie del

[3]JSON Web Token

token richiedendole ad OAM (attraverso il filtro Connect To). Una volta recuperato il tutto, effettua altre due richieste ad OAM richiamando la risorsa che permette di invalidare i token in questione, invalidando Access e Refresh Token. Infine, ricrea una chiave per il token invalidato e aggiorna lo stato della cache del gateway, in questo modo, le informazioni sui token rimangono sincronizzate tra le due componenti, OAM e API Gateway.
Infine, bisogna configurare il frontend API, che gestisce l'interazione con l'utente, ed il backend API, che è la parte che elabora i dati generati dal frontend, in questo modo, una volta che il flusso di autenticazione ha avuto successo il Gateway inoltra le richieste alle risorse protette.
Su Console API Manager di Axway è possibile configurare Backend e Frontend API.
Per il Backend API bisogna impostare il server al quale verranno inoltrate le richieste per accedere ai servizi sottostanti e ad altre risorse, e restituirà i dati richiesti, come evidenziato nell'immagine 4.13.

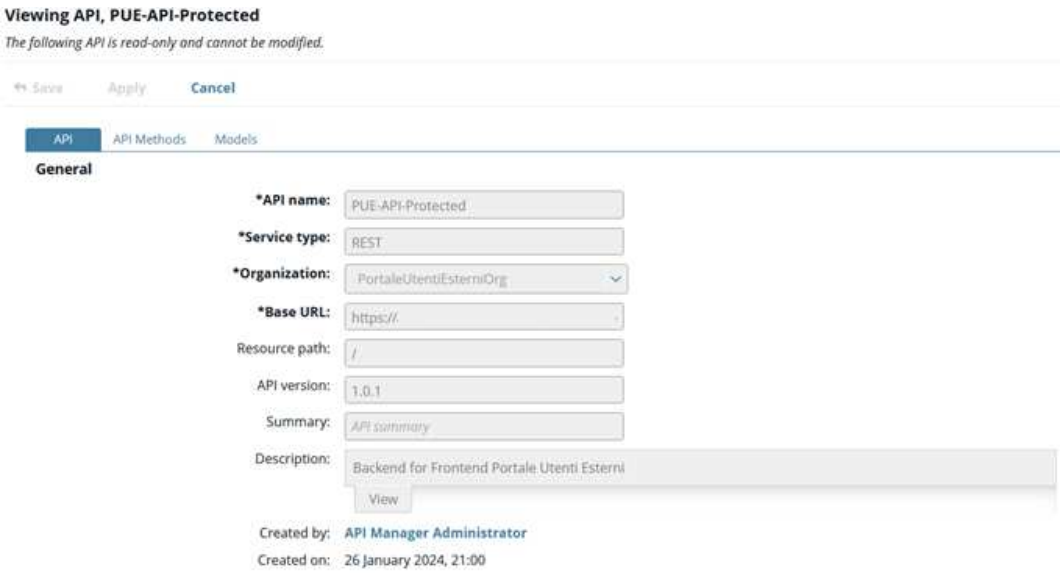

Figura 4.13: Configurazione Backend API su Console Axway

Per il Fontend API, invece, bisogna configurare la risorsa che si occupa dell'interfaccia utente e della comunicazione con l'utente, come in figura 4.14.

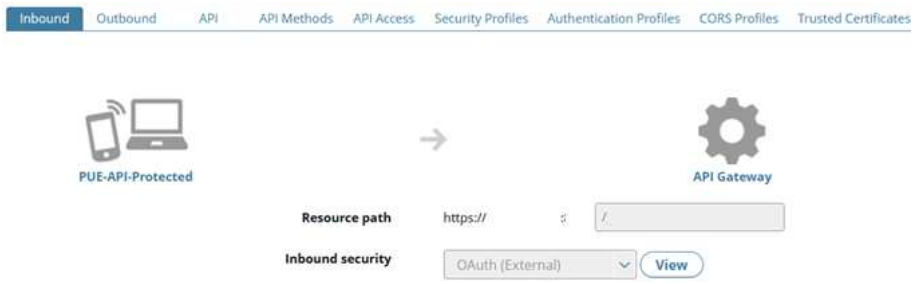

Figura 4.14: Configurazione Frontend API su Console Axway

Successivamente, deve essere anche configurato che utilizzerà il protocollo OAuth, ed è possibile farlo sempre da Console API Manager, come in figura 4.15.

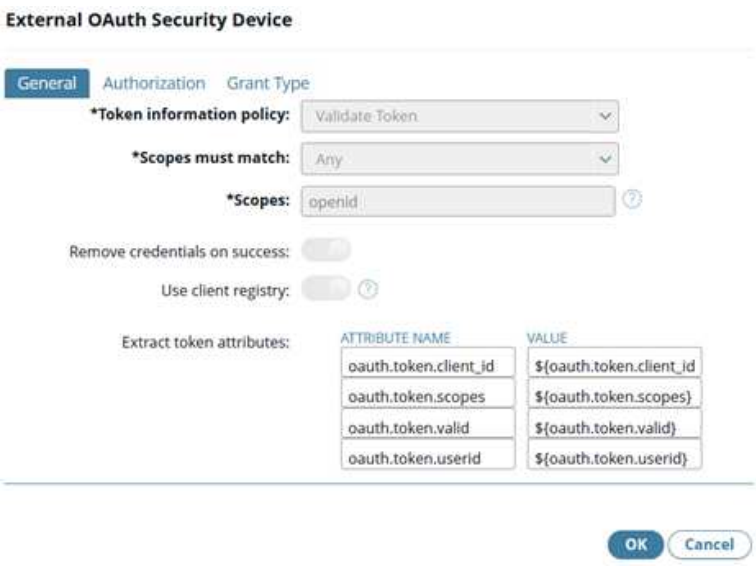

Figura 4.15: Configurazione OAuth su Console Axway

4.4.4 Sviluppo plugin

Sono stati sviluppati anche plugin custom per Oracle Access Manager, destinati alla gestione del blocco utente e all'orchestrazione del secondo fattore di autenticazione (OTP). Tuttavia, a causa di Accordi di Non Divulgazione (NDA), questi ultimi non verranno trattati nel dettaglio, ma saranno oggetto di astrazione, senza mostrare il codice sorgente.

Come detto in precedenza, su OAM Console è possibile configurare l'ordine dei plugin che devono essere richiamati in base agli steps, come evidenziato dall'immagine 4.16.

Figura 4.16: Steps Orchestration plugin

Il primo step coinvolge lo stepUI (User Identification), che esegue la fase di identificazione dell'utente ed è un plugin 'out of the box'[4].
Analogamente, lo step successivo, lo stepUA (User Authentication), gestisce la fase di autenticazione dell'utente.
Mentre, i plugin successivi sono stati sviluppati ad hoc.
Il primo plugin custom è il **SecondFactorHandler** il quale:

1. Recupera lo schema del secondo fattore dalla request e verifica che sia di tipo 'SMS'.

2. Una volta recuperata la modalità di scambio del secondo fattore, si procede a inviare una richiesta al wrapper di autenticazione[5] per inviare l'OTP via SMS all'utente e reindirizzarlo alla pagina di inserimento del secondo fattore.

[4]Soluzione pronta all'uso, già presente nel sistema

[5]componente che gestisce l'autenticazione e l'autorizzazione per altre parti di un'applicazione o sistema

3. Se tutto va a buon fine, il sistema procede con la validazione dell'OTP restituendo un codice 200 OK, altrimenti viene restituito un errore in base a determinate casistiche, ad esempio quando l'utente non ha il permesso di accedere, oppure quando l'OTP inserito è errato, o semplicemente un caso di errore generico.

L'altro plugin custom, invece, è **LockoutStatus** che entra in gioco in caso di failure di stepUI e stepUA, ovvero quando l'identificazione e/o l'autorizzazione fallisce. Esso verifica se l'utente ha un tempo di blocco o se l'account è stato bloccato permanentemente, verificando anche il numero di tentativi errati di inserimento password.

4.5 Confronto tra Federated Identity e Self-Sovereign Identity

I modelli Federated Identity (FI) e Self-Sovereign Identity (SSI) sono due approcci fondamentali per la gestione dell'identità digitale, ma differiscono significativamente nei loro principi, protocolli e implementazioni. In questa tesi, esploreremo i concetti principali, concentrandoci sulle differenze chiave.

4.5.1 Modello Federated Identity (FI)

La Federated Identity è un modello in cui gli utenti possono accedere a servizi online utilizzando le stesse credenziali di accesso. Questo sistema si basa su un Identity Provider (IdP), che autentica l'utente e rilascia un token di accesso ai servizi [16].
La Federated Identity spesso utilizza il protocollo **OpenID Connect (OIDC)**. OIDC è un framework di autenticazione e autorizzazione basato su OAuth 2.0, che consente l'autenticazione federata e l'autorizzazione basata su token.
Ecco alcuni punti salienti:

- **Funzionamento**: Quando un utente accede a un servizio, l'IdP verifica le credenziali e rilascia un token di accesso. Il servizio utilizza questo token per verificare l'identità dell'utente senza dover gestire direttamente le credenziali.
- **Autorità di Federazione**: Le Federazioni OIDC hanno un'autorità di federazione (come AgID per SPID) che certifica i Metadata. Questi contengono le chiavi pubbliche per le operazioni di firma digitale e criptazione e le definizioni necessarie all'interscambio delle informazioni. I Metadata sono certificati da un parte fidata che all'interno della Federazione SPID è AgID.
- **Controllo limitato**: Questo principio si riferisce alla necessità di limitare l'accesso agli utenti solo alle risorse e ai servizi per i quali hanno l'autorizzazione. Ciò aiuta a ridurre il rischio di accesso non autorizzato e a proteggere i dati sensibili.
- **Centralizzazione**: Anche se il Federated Identity coinvolge più entità, c'è una certa centralizzazione nel gestire le credenziali degli utenti. Questo aiuta a mantenere un controllo unificato sulla sicurezza e l'autenticazione.

Il modello Federated Identity che utilizza il protocollo OIDC, l'Identity Provider ha il controllo sull'autenticazione dell'utente, esso è un modello centralizzato, ma permette di semplifica l'accesso ai servizi digitali con le federazioni OIDC.

4.5.2 Vantaggi del protocollo OIDC

OpenID Connect è uno standard di autenticazione che estende il protocollo di autorizzazione OAuth 2.0. Questo protocollo è caratterizzato da alti livelli di flessibilità e sicurezza, semplicità di implementazione ed efficacia nell'interoperabilità.
I principali vantaggi di OpenID Connect sono principalmente:

1. **Facilità di integrazione**: OpenID Connect permette di integrare applicazioni su diverse piattaforme, come single-page app, web, backend, mobile e IoT.

2. **Integrazione di componenti di terze parti**: In modo sicuro, interoperabile e scalabile, è possibile integrare componenti di terze parti.

3. **Risolve problematiche di sicurezza**: OpenID Connect affronta alcune delle problematiche di sicurezza riscontrate in OAuth 2.0, garantendo una maggiore resilienza informatica.

4. **Autenticazione dell'utente finale**: Consente ai client di verificare l'identità dell'utente finale e di ottenere informazioni di base sul suo profilo in modo interoperabile.

5. **Controllo dell'utente sugli accessi**: L'utente può revocare ogni autenticazione, mantenendo il pieno controllo sugli accessi effettuati.

Questi vantaggi rendono OpenID Connect uno strumento di riferimento per l'autenticazione e l'identificazione digitale, utilizzato anche da grandi aziende come Google, Facebook e Microsoft. Inoltre, in ambito italiano, è stato adottato per il Sistema Pubblico di Identità Digitale (SPID) e la Carta d'Identità Elettronica (CIE).

4.5.3 Modello Self-Sovereign Identity (SSI)

L'SSI è un modello di identità digitale decentralizzato implementato tramite tecnologia Blockchain. Si basa sulla restituzione all'utente del controllo sulle proprie informazioni personali e risolve i limiti dei sistemi di identificazione digitale centralizzati [17].
L'SSI utilizza il protocollo **OpenID for Verifiable Credentials (OID4VC)**. Questo protocollo estende OIDC per supportare le credenziali verificabili.

Figura 4.17: FI vs SSI

Ecco alcuni principi fondamentali, evidenziati anche dall'immagine 4.17:

- **Controllo**: L'utente genera un identificativo che può dimostrare di controllare, ha il controllo totale delle proprie credenziali e può decidere come e quando condividerle, utilizzando meccanismi crittografici simili a quelli di Bitcoin o Ethereum.
- **Esistenza**: L'identità digitale non può essere separata dall'entità fisica dell'utente.
- **Sicurezza**: Grazie all'uso di tecnologie come la blockchain, le informazioni personali sono protette da accessi non autorizzati.
- **Privacy**: L'utente può scegliere quali informazioni rivelare e a chi, mantenendo la privacy delle altre.

- **Decentralizzazione**: L'utente non delega la custodia delle informazioni personali a terzi.
- **Disponibilità**: Le credenziali sono accessibili in qualsiasi momento e luogo, senza dipendenza da terze parti.
- **Accessibilità selettiva**: Solo dopo aver ricevuto l'autorizzazione esplicita dall'utente, le informazioni possono essere condivise con altri.
- **Dinamica e Scalabile**: L'SSI offre una fiducia dinamica e scalabile. Le entità possono essere abilitate o disabilitate nella Federazione in modo flessibile.

Il modello Self-Sovereign Identity, l'utente ha il controllo completo sulla propria identità, questo perché esso è un modello decentralizzato, basato su Blockchain, offrendo maggiore privacy agli utenti.

4.5.4 Vantaggi del protocollo OID4VC

OID4VC offre diversi vantaggi rispetto ai protocolli tradizionali per la gestione delle credenziali e l'autenticazione, tra cui:

1. **Verificabilità e Integrità**: Le credenziali verificabili emesse tramite OID4VC sono basate su standard aperti, come le specifiche del Consorzio World Wide Web (W3C). Ciò garantisce che le informazioni siano verificabili e integre, riducendo il rischio di frodi o manipolazioni.
2. **Interoperabilità**: OID4VC è progettato per funzionare con qualsiasi formato di credenziali, inclusi quelli del W3C. Questo livello di flessibilità consente l'interoperabilità tra diverse implementazioni e sistemi.
3. **Privacy e Controllo Utente**: OID4VC mette l'utente al centro del processo di gestione delle credenziali. Gli utenti possono controllare quali informazioni vengono condivise e con chi. Questo approccio rispetta la privacy e dà agli utenti maggiore fiducia nel sistema.
4. **Flusso di Lavoro Semplificato**: OID4VC semplifica il flusso di lavoro per l'emissione e la presentazione delle credenziali. L'uso di OAuth 2.0 per l'autorizzazione semplifica l'integrazione con altre applicazioni e servizi.
5. **Supporto per Casi d'Uso Specifici**: OID4VC è stato adottato in vari contesti, come l'identità digitale europea, il progetto EBSI e altri. Questo dimostra la sua idoneità per casi d'uso reali.

OID4VC offre una soluzione moderna, flessibile e sicura per la gestione delle credenziali, migliorando l'esperienza dell'utente e la sicurezza complessiva.

4.6 Come funziona il protocollo OID4VC

Il protocollo OID4VC è composto da tre specifiche principali [**?**]:

1. **OpenID for Verifiable Credential Issuance**: Definisce un'API e meccanismi di autorizzazione basati su OAuth per l'emissione di credenziali verificabili.

2. **OpenID for Verifiable Presentations**: Definisce un meccanismo su top di OAuth 2.0 per permettere la presentazione di affermazioni sotto forma di credenziali verificabili all'interno del flusso del protocollo.

3. **Self-Issued OpenID Provider v2**: Consente agli utenti di utilizzare OpenID Provider (OPs) che controllano.

4.6.1 Emissione di credenziali verificabili

Per quanto riguarda l'emissione di credenziali verificabili (Verifiable Credential Issuance) si compone di quattro azioni chiave, come mostrato anche in figura 4.18:

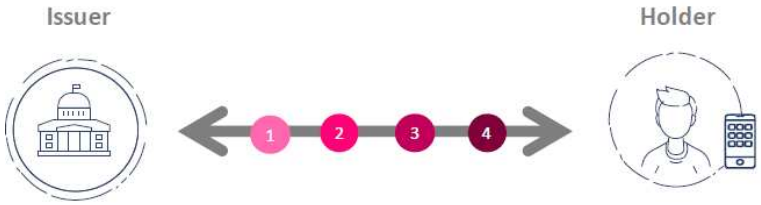

Figura 4.18: Verifiable Credential Issuance

1. **Richiesta delle Credenziali Verificabili**: Un utente richiede una VC a un emittente. L'emittente include dati sull'utente nella VC digitale, inclusa l'identificazione dell'utente.

2. **Autorizzazione**: L'emittente utilizza un OAuth protetto per ottenere l'autorizzazione per ricevere VC. Questo passaggio garantisce la sicurezza e la conformità con gli standard esistenti.

3. **Emissione delle Credenziali Verificabili**: L'emittente firma la VC con la propria chiave privata, garantendo l'integrità e la verificabilità della VC.

4. **Collezionare le Credenziali Verificabili**: L'utente riceve la VC e la conserva in un portafoglio digitale. Questo passaggio permette all'utente di presentare le VC quando necessario.

Un elemento importante è il **DID**, ovvero un identificatore composto da caratteri alfanumerici, è unico e attraverso la risoluzione è possibile consultare il DIDDocument, che è un documento JSON contenente varie informazioni per interagire con il proprietario del DID. Un DID Document include chiavi pubbliche crittografiche, service endpoint, parametri di autenticazione, timestamp e altri metadati, come evidenziato anche in figura 4.19.
Di seguito una immagine che mostra gli elementi standard di un DID document:

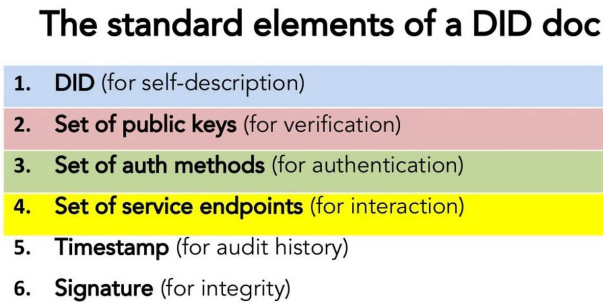

Figura 4.19: Standard DID document

Nella fase 2 di autenticazione, gli emittenti possono associare le credenziali verificabili al DID dopo che il titolare ha dimostrato di controllare le chiavi private corrispondenti.

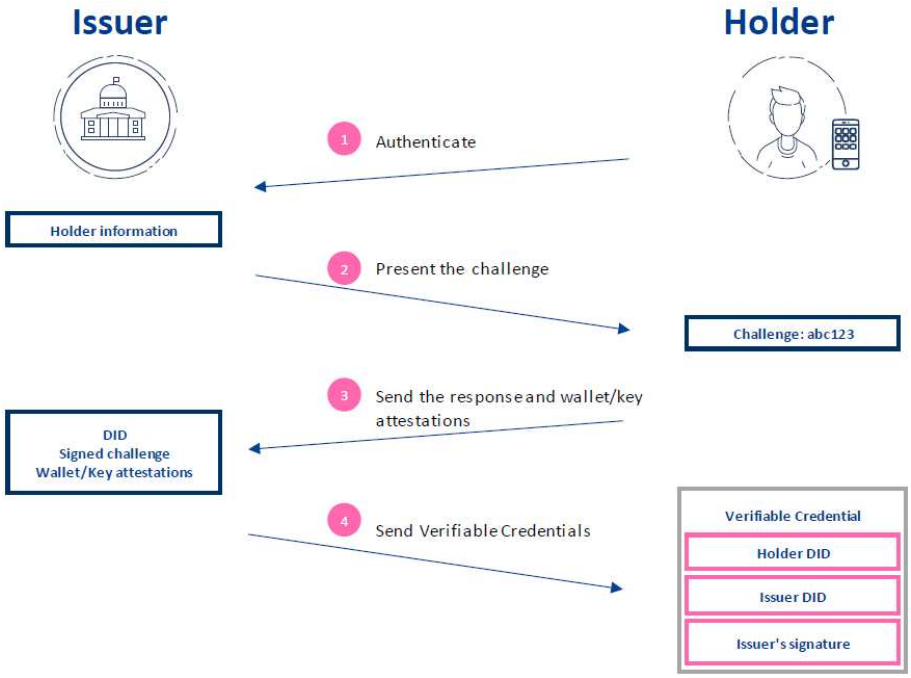

Figura 4.20: Sfida VC DID

Come mostrato in figura 4.20, l'emittente crea una sfida per il titolare per dimostrare di controllare il suo DID. Successivamente, verifica le attestazioni della chiave e/o del portafoglio per garantire che le chiavi DID siano archiviate e gestite in un portafoglio che soddisfi i requisiti di sicurezza dell'emittente. Una volta che si è sicuri che il titolare che richiede la/e VC controlla il DID, vengono emesse le VC per quel DID.

4.6.2 Presentazione di credenziali verificabili

Per quanto riguarda la presentazione di credenziali verificabili (Verifiable Presentations) si compone anche questo di quattro azioni chiave, come mostrato in figura 4.21:

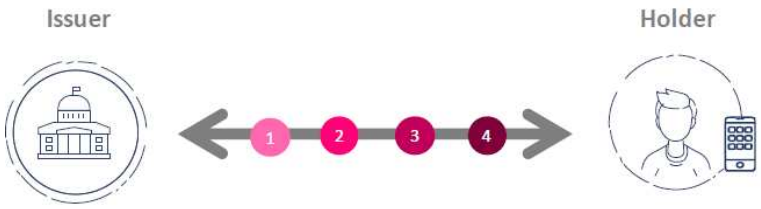

Figura 4.21: Verifiable Credential Presentations

1. **Richiesta delle Credenziali Verificabili**: Il verificatore richiede una o più VC all'utente.

2. **Autorizzazione**: Il titolare si autentica presso il verificatore tramite il metodo di autenticazione fornito.

3. **Presentazione delle Credenziali Verificabili**: Il wallet dell'utente elabora la richiesta di condivisione VC che compila e presenta al verificatore.

4. **Verifica delle Credenziali Verificabili**: Il verificatore verifica le credenziali verificabili per assicurarsi della loro autenticità e validità.

Un elemento, invece, importante in questa fase è il **digital wallet**.
Tipicamente le credenziali rilasciate vengono conservate in wallet hardware che permettono di avere un elevato livello di sicurezza e di resistere ai possibili attacchi (mobile app, NFC Card). Attualmente, diverse soluzioni commerciali sono già esistenti, e quasi tutte si basano sull'utilizzo di una blockchain. Il wallet sa quali VC condividere in base al tipo di credenziali o criteri richiesti dal verificatore. I metadati della richiesta di autenticazione contengono le richieste di autenticazione OAuth2 standard in modo che il wallet possa apprendere tutto sul verificatore (endpoint, formati supportati, firme, ecc.) e per proteggere il flusso di presentazione.
Il processo di verifica include tipicamente la verifica dell'autenticità delle credenziali, la conferma che sono state emesse da un'autorità affidabile e che sono state firmate digitalmente.

L'immagine 4.22 rappresenta un esempio di come è formata la richiesta di presentazione di credenziali verificabili:

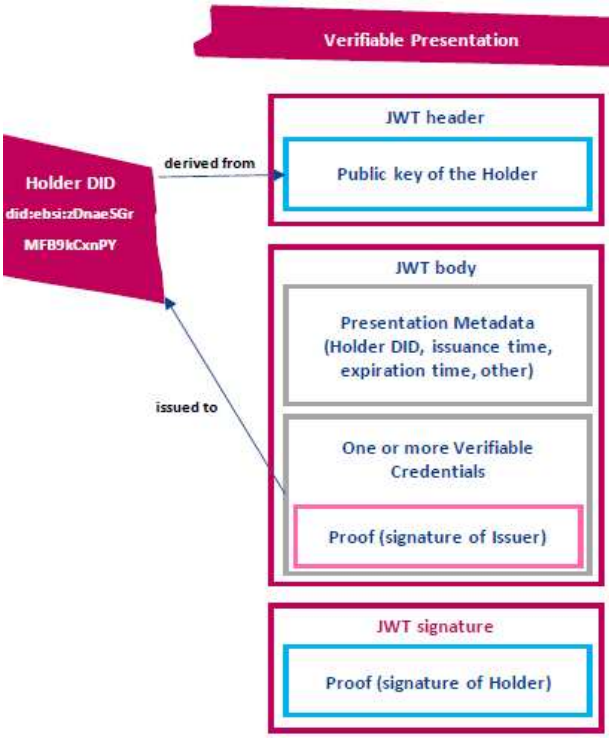

Figura 4.22: Presentazione VC

- **Header**: La chiave pubblica nell'intestazione è la chiave pubblica DID del titolare che deve verificare la firma della richiesta.
- **Body**: I metadati di presentazione contengono informazioni sul DID del titolare che devono essere derivate dalla chiave pubblica condivisa nell'intestazione e da altre attestazioni della richiesta standard.
  Una o più credenziali verificabili sono incorporate nella richiesta. Le credenziali verificabili devono essere emesse allo stesso DID indicato nei metadati della presentazione. Se le richieste vengono rilasciate a più DID, il titolare deve presentare più presentazioni verificabili.
- **Signature**: La prova (Proof) è la firma del titolare della Presentazione Verificabile. La chiave pubblica nell'intestazione deve verificare la firma.

4.7 OIDC vs OID4VC

OpenID Connect (OIDC) è un protocollo di autenticazione e autorizzazione basato su OAuth 2.0. Tuttavia, presenta alcune limitazioni che possono essere affrontate attraverso OpenID Connect for Verifiable Credentials (OID4VC). Alcuni punti deboli di OIDC e come OID4VC può contribuire a risolverli sono:

1. **Centralizzazione dell'autenticazione**:
   OIDC richiede un Identity Provider (IDP) centralizzato per l'autenticazione.
   OID4VC consente agli utenti di utilizzare Verifiable Credentials senza la necessità di un IDP centrale. Questo decentralizza l'emissione e la presentazione delle credenziali.

2. **Privacy e controllo dell'utente**:
   OIDC richiede che l'utente si autentichi tramite l'IDP, che potrebbe raccogliere informazioni personali.
   OID4VC permette agli utenti di gestire direttamente le proprie credenziali verificabili senza intermediari, garantendo maggiore controllo sulla privacy.

3. **Complessità per gli sviluppatori**:
   OIDC richiede l'implementazione di flussi di autenticazione complessi.
   OID4VC semplifica il processo di emissione e presentazione delle credenziali, riducendo la complessità per gli sviluppatori.

4. **Riproduzione di token**:
   OIDC utilizza token di accesso che possono essere soggetti a riproduzione.
   OID4VC introduce meccanismi per evitare la riproduzione delle credenziali, migliorando la sicurezza.

5. **Flessibilità delle credenziali**:
   OIDC è principalmente focalizzato su token di accesso e affermazioni standard.
   OID4VC supporta una vasta gamma di formati di credenziali verificabili, inclusi SD-JWT VC, mDL e VCDM.

La centralizzazione dell'autenticazione è il punto più critico, dato che, la maggior parte dei protocolli consente solo ai titolari di autorizzare i verificatori che sono in relazione con l'emittente (Isser) ad accedere alle informazioni utente. In qualità di titolare, non posso condividere le informazioni con il

verificatore che desidero.
La figura 4.23 schematizza l'idea.

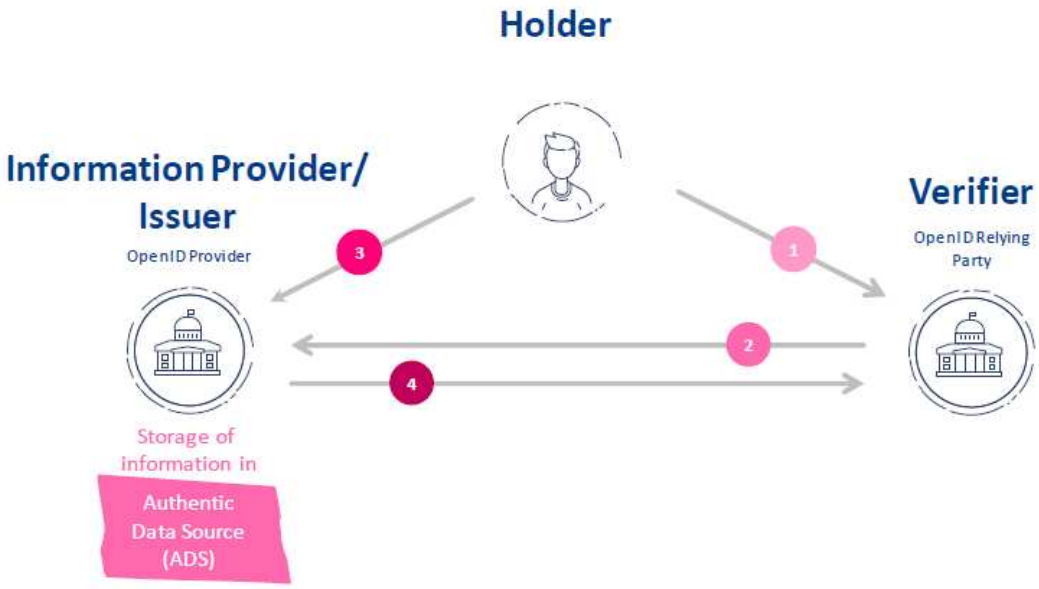

Figura 4.23: Modalità di scambio informazioni del protocollo OIDC

1. Richiesta di Accesso alla risorsa protetta;
2. Richiesta delle informazioni sul titolare;
3. Autenticazione e autorizzazione del verificatore a ottenere le tue informazioni richieste;
4. Condivisione delle informazioni sull'Holder.

Per quanto riguarda, invece, il protocollo OID4VC, lo scambio di informazioni avviene solo in tre fasi consentendo ai titolari di condividere le proprie informazioni con chiunque desiderino.

Di seguito uno schema rappresentato dall'immagine 4.24:

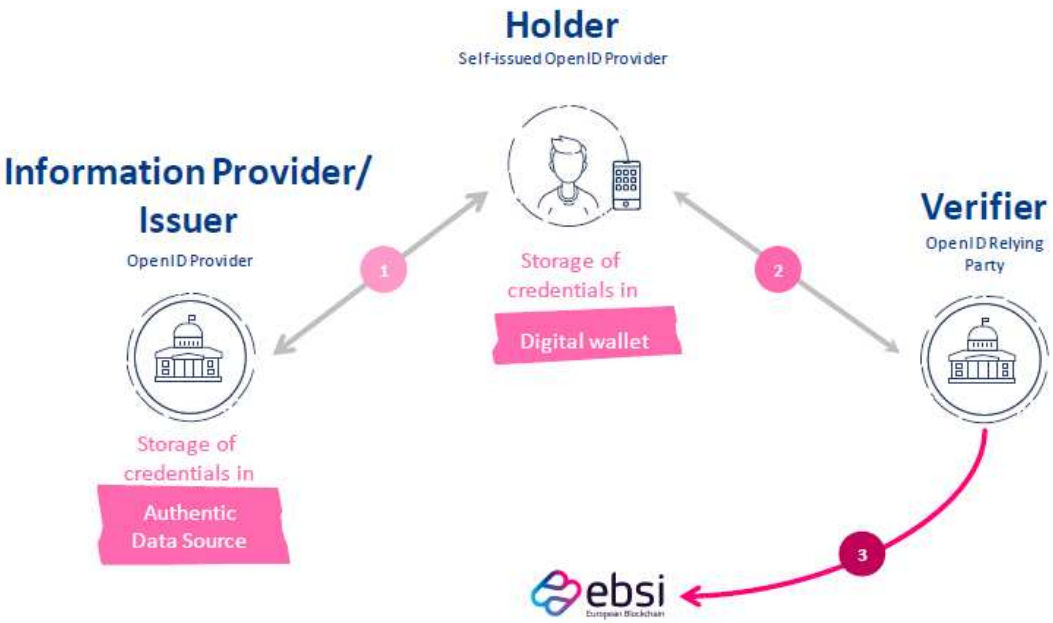

Figura 4.24: Modalità di scambio informazioni del protocollo OID4VC

1. L'Holder richiede le credenziali;
2. Il Verificatore richiede le credenziali all'Holder;
3. Autenticazione e autorizzazione del verificatore a ottenere le informazioni richieste;

OID4VC affronta le limitazioni di OIDC decentralizzando l'emissione e la presentazione delle credenziali, garantendo maggiore controllo dell'utente e semplificando lo sviluppo di applicazioni che richiedono autenticazione e autorizzazione.
Tuttavia, OID4VC presenta alcune sfide e limiti che è importante considerare:

1. **Complessità Tecnica**: L'implementazione di OID4VC richiede una comprensione approfondita dei protocolli coinvolti, come OAuth 2.0 e le specifiche del W3C. Questa complessità può rendere difficile per gli sviluppatori adottare OID4VC senza una curva di apprendimento significativa.

2. **Standardizzazione in Evoluzione**: Le specifiche OID4VC sono ancora in fase di sviluppo e potrebbero subire modifiche. Questo può comportare sfide per l'interoperabilità tra diverse implementazioni e versioni.

3. **Gestione delle Chiavi e della Privacy**: OID4VC richiede la gestione delle chiavi crittografiche per firmare e verificare le credenziali. La sicurezza delle chiavi e la protezione della privacy degli utenti sono aspetti critici che richiedono attenzione.

4. **Adozione e Consapevolezza**: Molti sviluppatori e organizzazioni potrebbero non essere a conoscenza di OID4VC o potrebbero preferire utilizzare protocolli tradizionali con cui sono più familiari. L'adozione di OID4VC richiede una maggiore consapevolezza e promozione.

5. **Scalabilità**: La gestione di grandi quantità di credenziali verificabili può comportare sfide di scalabilità. Le implementazioni devono essere in grado di gestire carichi di lavoro crescenti senza compromettere le prestazioni.

6. **Compatibilità con Sistemi Esistenti**: L'integrazione di OID4VC con sistemi legacy o altre soluzioni di autenticazione può essere complessa. La compatibilità con le infrastrutture esistenti è un aspetto da considerare.

7. **Rischi di Abuso**: Come per qualsiasi sistema di autenticazione, OID4VC potrebbe essere soggetto ad abusi o attacchi. La gestione delle autorizzazioni e la prevenzione delle frodi sono sfide importanti.

Sebbene OID4VC offra vantaggi significativi, è essenziale considerare attentamente queste sfide e pianificare di conseguenza durante l'implementazione. In sintesi, OIDC offre una solida base per l'autenticazione ed è uno standard consolidato, mentre OID4VC richiede ulteriori approfondimenti per comprenderne appieno l'utilizzo e le potenzialità dato che è una nuova tecnologia ancora in fase di sviluppo.

Capitolo 5

Valutazione del sistema

In questo capitolo, il sistema verrà sottoposto a una serie di test per valutarne le prestazioni in termini di velocità, efficienza e affidabilità.
A causa di Accordi di Non Divulgazione (NDA), non verranno forniti dettagli sulle specifiche tecniche dei server.
La prima sezione descrive la progettazione degli stress test, spiegando la metodologia adottata e gli strumenti utilizzati per la loro esecuzione.
Infine, la seconda sezione presenta l'esecuzione degli stress test e la valutazione del sistema.

5.1 Progettazione Stress Test

Nell'ottica di un continuo miglioramento dei servizi, è stata presa la decisione di condurre una serie di stress test sul layer di sicurezza, con l'obiettivo di identificare eventuali anomalie prestazionali, garantendo così la migliore esperienza possibile agli utenti finali.
Uno stress test valuta la resilienza[1] di un sistema in condizioni avverse, identificando punti deboli e capacità di recupero attraverso scenari estremi. È utilizzato per valutare il corretto funzionamento dei sistemi anche in situazioni di crisi.
I test sono stati condotti simulando il comportamento di utenti reali che interagiscono con il sistema. Sono stati utilizzati diversi scenari di carico per valutare le prestazioni del sistema in condizioni normali e di stress.

5.1.1 Utilizzo di JMeter

Per eseguire gli stress test è stato utilizzato Apache JMeter, al fine di misurare la velocità, l'efficienza e l'affidabilità sotto vari scenari di carico.
JMeter permette di simulare il comportamento di utenti reali per testare le prestazioni di un sito web o di un'applicazione. È particolarmente utile per verificare come un sistema risponde sotto carico intenso, con molti utenti simultanei che usufruiscono del sistema.

5.1.2 Architettura di test

Considerando il flusso di autenticazione descritto nel capitolo 4, è stato predisposto un Client di Test JMeter e 10.000 utenze fittizie. Il flusso attraverserà tutti i componenti coinvolti che includono i seguenti elementi:

- Oracle HTTP Server (OHS)
- Oracle Access Manager (OAM)
- Oracle Unified Directory (OUD)
- Axway API Gateway (APIGW)
- Identity Provider (IdP)
- API del Portale (PUE API)

[1]Capacità di un sistema informatico di adattarsi e continuare a funzionare correttamente nonostante eventi sfavorevoli, come attacchi informatici, guasti o picchi di carico

Sequence diagram In figura 5.1 è illustrato il Sequenze Diagram del flusso di login di test derivante dal processo di autenticazione.

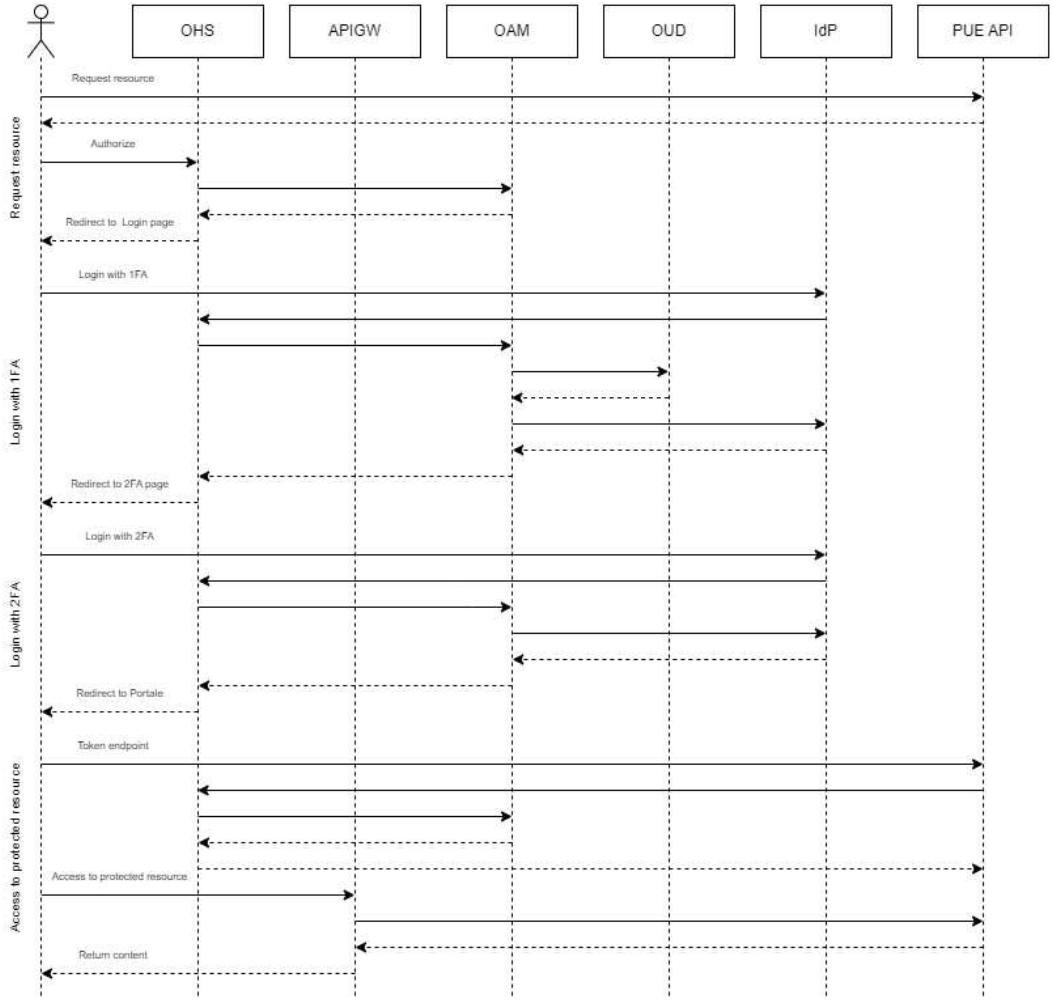

Figura 5.1: Sequence Diagram Stress Test

Nel dettaglio, verrà simulato un processo di autenticazione da parte di un utente fittizio (creato con JMeter) che richiede l'accesso a una risorsa protetta tramite PUE API. In questo modo, l'utente verrà quindi reindirizzato alla pagina di login.
L'IdP viene invocato e l'utente viene reindirizzato alla pagina di inserimento del primo fattore. Una volta inserito, questo verrà convalidato da OAM, a condizione che sia presente su OUD.
Dopo l'autenticazione di primo livello, verrà richiesto un secondo fattore. Successivamente, OAM reindirizzerà l'utente a una nuova pagina di login per

inserire l'OTP, che verrà convalidato invocando l'IdP.
Completata l'autenticazione a due fattori, l'utente richiederà la risorsa protetta e l'APIGW intercetterà la richiesta verso il Portale, restituendo il contenuto richiesto al chiamante.

5.1.3 Stress Test con Distribuzione Gaussiana

Per gli ultimi 2 test condotti è stata utilizzata la distribuzione gaussiana. Questi test sono progettati per simulare scenari realistici, come i picchi di carico che un sistema può sperimentare in determinati orari.
La distribuzione gaussiana, caratterizzata dalla sua forma a campana, permette di modellare le variabili di interesse in modo che la maggior parte dei valori si concentri attorno alla media, con variazioni che seguono un pattern naturale.
Questo approccio è particolarmente utile per analizzare come il sistema reagisce a variazioni che riflettono situazioni reali, come un aumento significativo del traffico durante le ore di punta. Utilizzando questi scenari, è possibile valutare la resilienza del sistema e la sua capacità di mantenere le prestazioni anche sotto condizioni di stress.

5.1.4 Metriche di valutazione

Per garantire una valutazione accurata e definitiva dell'efficacia del sistema realizzato, è essenziale identificare e comprendere i parametri chiave.
Le prestazioni del sistema sono state analizzate considerando principalmente i seguenti parametri:

- **Tasso di errore**: Questo parametro misura la frequenza con cui si verificano errori durante il processo di login. Un tasso di errore basso indica un'elevata affidabilità del sistema.
- **Tempo di risposta**: Rappresenta il tempo medio impiegato dal sistema per rispondere ad una richiesta. Un tempo di risposta ridotto è indicativo di un sistema efficiente e reattivo.
- **Throughput**: Questo parametro valuta la quantità di operazioni o transazioni che il sistema può gestire in un determinato intervallo di tempo. Un throughput elevato suggerisce una buona capacità del sistema di gestire carichi di lavoro intensi.

Questi parametri sono stati scelti per fornire una visione completa delle prestazioni del sistema, permettendo di identificare eventuali aree di miglioramento e di garantire che il sistema soddisfi i requisiti prestazionali previsti.

5.2 Esecuzione Stress Test

Considerando con quanto riportato sopra, sono stati definiti i seguenti scenari di test al fine di valutare la robustezza e l'efficienza del sistema nel gestire vari livelli di carico:

- **Run 1**: 5 login al secondo per 30 minuti
- **Run 2**: 15 login al secondo per 30 minuti
- **Run 3**: 20 login al secondo per 30 minuti
- **Run 4**: 30 login al secondo per 30 minuti
- **Run 5**: 15 login al secondo per 2 ore con distribuzione Gaussiana
- **Run 6**: 15 login al secondo per 7 ore con distribuzione Gaussiana

5.2.1 Run 1

Nel primo test, sono stati simulati 5 processi di login al secondo per un periodo di 30 minuti.
Il grafico in figura 5.2 illustra la distribuzione del numero di transazioni effettuate nell'arco di 30 minuti. Questa rappresentazione visiva permette di analizzare in dettaglio il comportamento del sistema durante il periodo di osservazione.

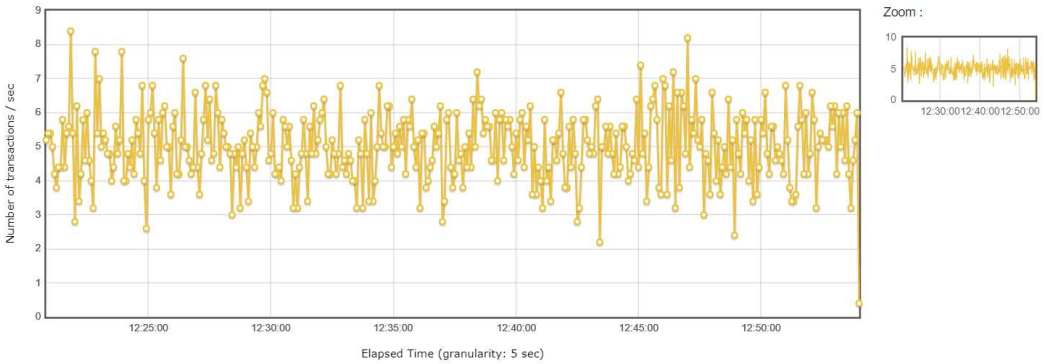

Figura 5.2: Run 1: 5 login al secondo

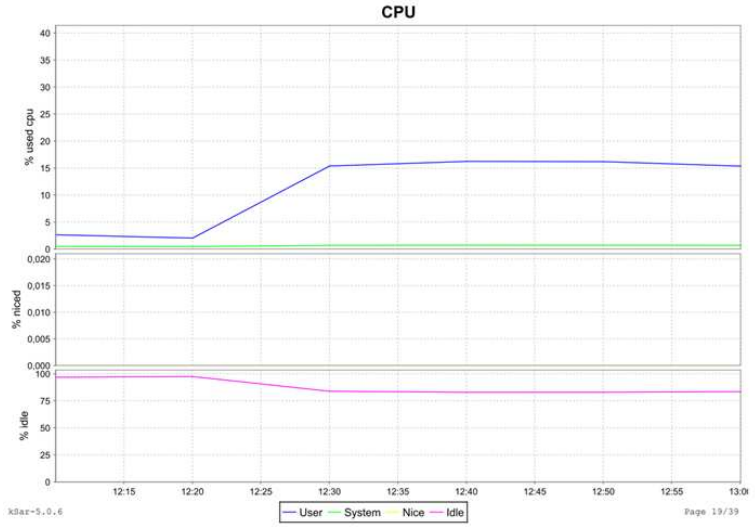

Figura 5.3: Run 1: Consumo CPU

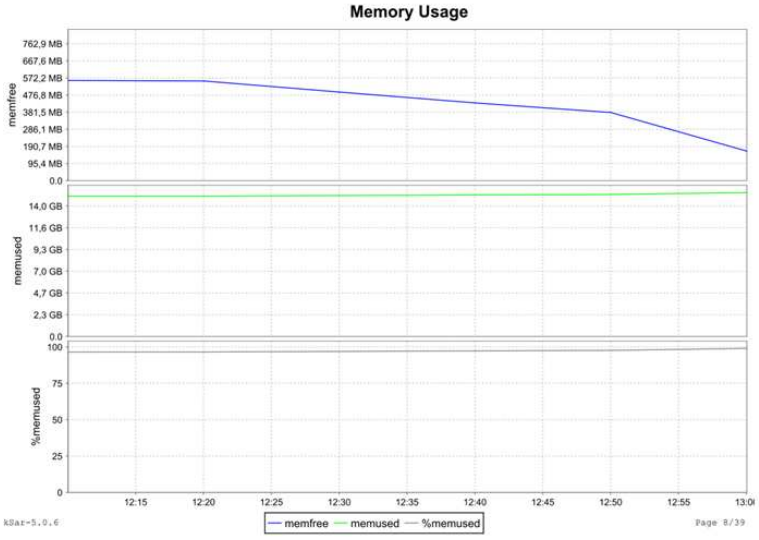

Figura 5.4: Run 1: Consumo Memoria

ordine	Requests	Executions			Response Times (ms)			Throughput
		#Samples	KO	Error %	Average	Min	Max	Transactions/s
1	Request resource	10000	0	0.00%	4.35	1	3016	5.03
2	Authorize	10000	0	0.00%	19.93	12	2881	5.03
3	Login with 1FA	10000	0	0.00%	141.97	14	506	5.03
4	Login with 2FA	10000	0	0.00%	89.38	40	3117	5.03
5	Token endpoint	10000	0	0.00%	122.67	94	3106	5.03
6	Access to protected resource	10000	0	0.00%	103.87	91	1243	5.03

Figura 5.5: Run 1: Tabella riepilogativa

Analisi Run 1: Dall'analisi della tabella riepilogativa 5.5, si può osservare che il primo test, che ha simulato 5 processi di login al secondo per un periodo di 30 minuti, ha prodotto i risultati migliori. Infatti, tutti i 10.000 processi di login sono stati completati con successo. Inoltre, come si evince dalle figure 5.3 e 5.4 il consumo della CPU e della memoria è rimasto nei limiti previsti.
Questo test è stato considerato come punto di partenza poiché il carico da gestire era relativamente basso. Di conseguenza, non ci si aspettavano criticità significative, e i risultati ottenuti hanno confermato questa previsione, dimostrando la capacità del sistema di gestire efficacemente il carico iniziale senza problemi.

5.2.2 Run 2

Nel secondo test, sono stati simulati 15 processi di login al secondo per un periodo di 30 minuti.
Il grafico in figura 5.6 illustra la distribuzione del numero di transazioni effettuate nell'arco di 30 minuti.

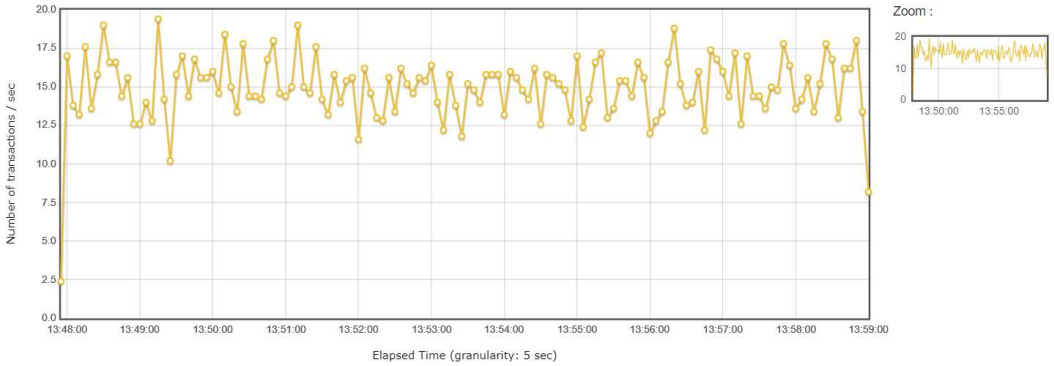

Figura 5.6: Run 2: 15 login al secondo

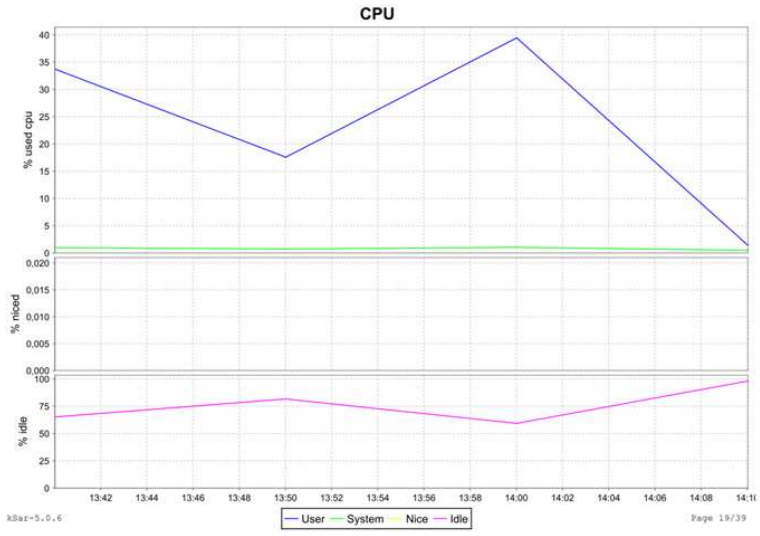

Figura 5.7: Run 2: Consumo CPU

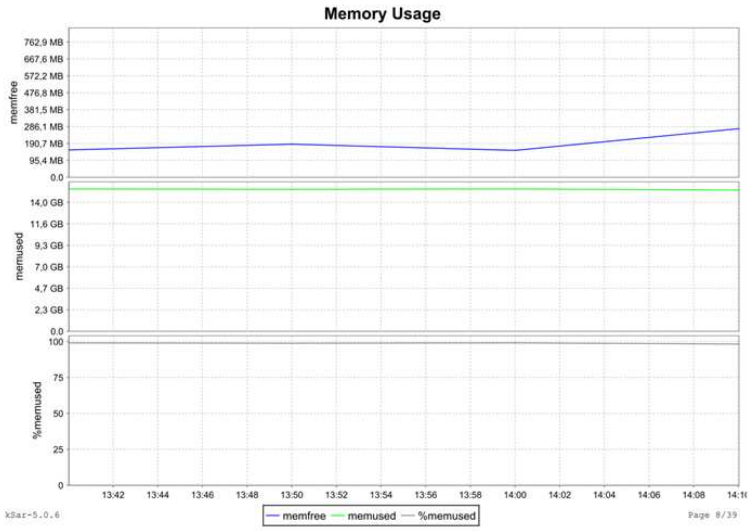

Figura 5.8: Run 2: Consumo Memoria

ordine	Requests	Executions			Response Times (ms)			Throughput
		#Samples	KO	Error %	Average	Min	Max	Transaction s/s
1	Request resource	10000	0	0.00%	3.89	1	3181	15.06
2	Authorize	10000	0	0.00%	20.03	11	1236	15.06
3	Login with 1FA	10000	0	0.00%	104.25	13	2566	15.08
4	Login with 2FA	10000	0	0.00%	80.43	32	3372	15.09
5	Token endpoint	10000	0	0.00%	165.20	95	1124	15.10
6	Access to protected resource	10000	0	0.00%	135.31	90	1108	15.10

Figura 5.9: Run 2: Tabella riepilogativa

Analisi Run 2: Dall'analisi della tabella riepilogativa 5.9, si può osservare che il secondo test, che ha simulato 15 processi di login al secondo per un periodo di 30 minuti, ha prodotto anche qui i risultati migliori. Infatti, tutti i 10.000 processi di login sono stati completati con successo e, come si evince dalle figure 5.7 e 5.8, il consumo della CPU e della memoria è rimasto sempre nei limiti previsti.
Questo test ha dimostrato la capacità del sistema di gestire un carico leggermente superiore rispetto al test precedente.

5.2.3 Run 3

Nel terzo test, sono stati simulati 20 processi di login al secondo per un periodo di 30 minuti.
Il grafico in figura 5.10 illustra la distribuzione del numero di transazioni effettuate nell'arco di 30 minuti.

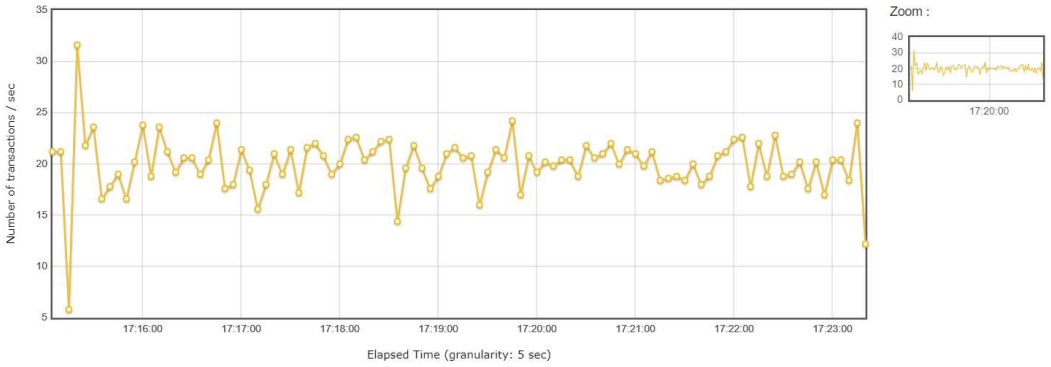

Figura 5.10: Run 3: 20 login al secondo

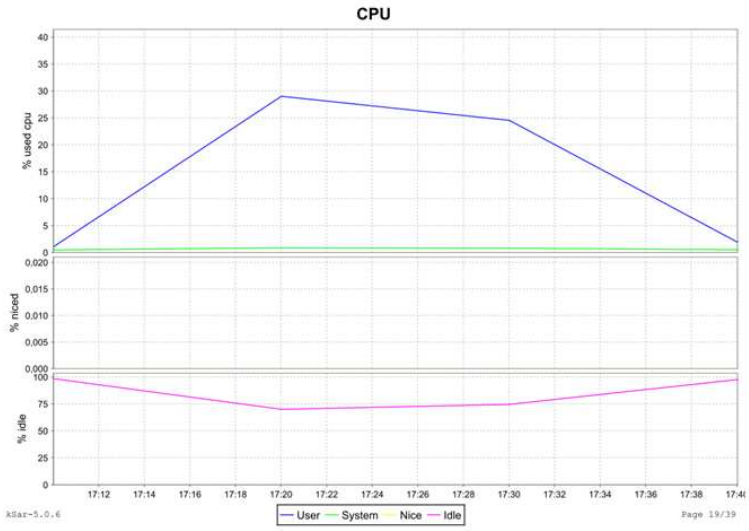

Figura 5.11: Run 3: Consumo CPU

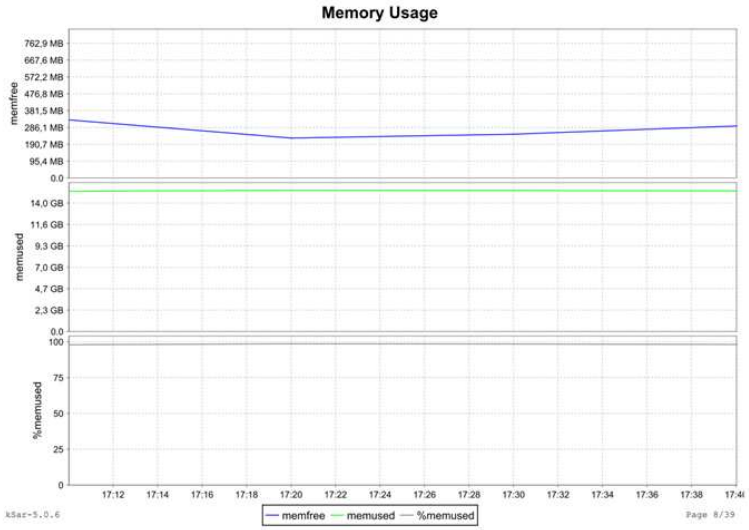

Figura 5.12: Run 3: Consumo Memoria

ordine	Requests	Executions			Response Times (ms)			Throughput
		#Samples	KO	Error %	Average	Min	Max	Transactions/s
1	Request resource	10000	0	0.00%	12.01	1	3761	20.09
2	Authorize	10000	0	0.00%	28.57	12	3848	20.09
3	Login with 1FA	10000	0	0.00%	114.73	13	4329	20.09
4	Login with 2FA	10000	0	0.00%	133.97	49	6070	20.11
5	Token endpoint	10000	0	0.00%	270.70	94	5088	20.12
6	Access to protected resource	10000	0	0.00%	242.14	90	4551	20.13

Figura 5.13: Run 3: Tabella riepilogativa

Analisi Run 3: Dall'analisi della tabella riepilogativa 5.13, si può osservare che il terzo test, che ha simulato 20 processi di login al secondo per un periodo di 30 minuti, ha ottenuto nuovamente i risultati migliori. Tutti i 10.000 processi di login sono stati completati con successo e, dalle figure 5.11 e 5.12, l'utilizzo della CPU e della memoria è rimasto costantemente entro i limiti stabiliti.
Questo test ha evidenziato la capacità del sistema di gestire un carico maggiore rispetto al test precedente.

5.2.4 Run 4

Nel quarto test, sono stati simulati 30 processi di login al secondo per un periodo di 30 minuti.
Il grafico in figura 5.14 illustra la distribuzione del numero di transazioni effettuate nell'arco di 30 minuti.

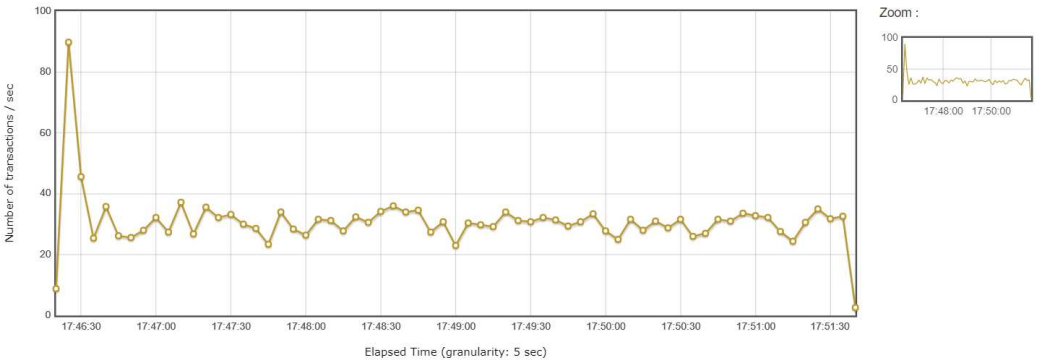

Figura 5.14: Run 4: 30 login al secondo

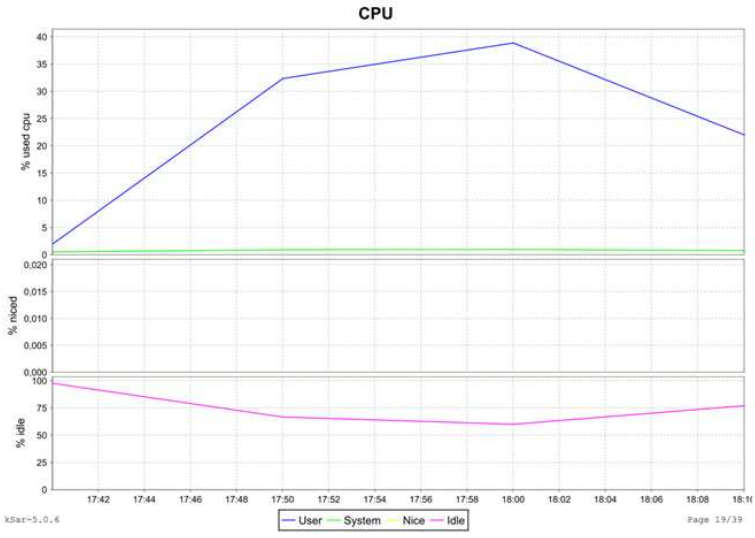

Figura 5.15: Run 4: Consumo CPU

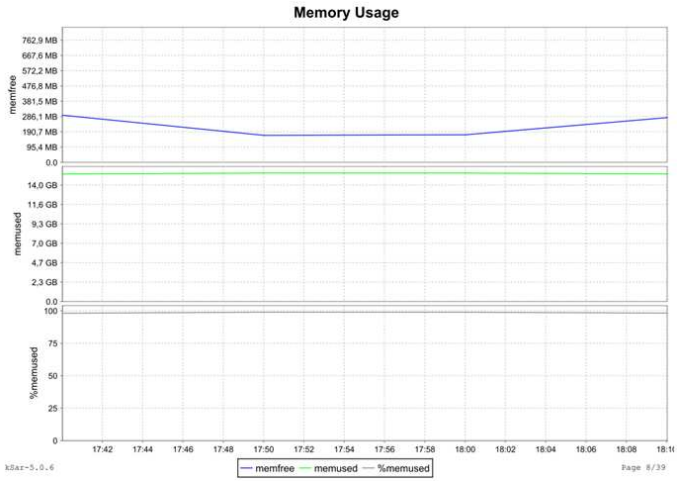

Figura 5.16: Run 4: Consumo Memoria

ordine	Requests	Executions			Response Times (ms)			Throughput
		#Samples	KO	Error %	Average	Min	Max	Transactions/s
1	Request resource	9997	0	0.00%	1642.39	1	193504	30.72
2	Authorize	9997	0	0.00%	323.27	11	9923	31.95
3	Login with 1FA	9997	0	0.00%	312.14	13	8106	32.16
4	Login with 2FA	9997	31	0.31%	2781.78	51	173125	32.41
5	Token endpoint	9966	134	1.34%	2396.67	95	164657	32.47
6	Access to protected resource	9832	39	0.40%	1630.11	91	192699	32.13

Figura 5.17: Run 4: Tabella riepilogativa

Analisi Run 4: Dall'analisi della tabella riepilogativa 5.17, si può osservare che il quarto test, che ha simulato 30 processi di login al secondo per un periodo di 30 minuti, in questo caso non ha ottenuto i risultati migliori. Non tutti i 10.000 processi di login sono stati completati con successo e, dalle figure 5.15 e 5.16, si osserva che l'utilizzo delle risorse è stato abbastanza alto.
In realtà, questo test ha dimostrato la capacità del sistema di mantenere buone prestazioni, nonostante non abbia completato tutti i processi, evidenziando una certa resilienza. Tale risultato suggerisce che il sistema è in grado di gestire situazioni di carico elevato senza compromettere significativamente le sue funzionalità.

5.2.5 Run 5

Nel quinto test, sono stati simulati 15 processi di login al secondo per un periodo di 2 ore, utilizzando la distribuzione Gaussiana.
Il grafico in figura 5.18 illustra la distribuzione del numero di transazioni effettuate nell'arco di 2 ore.
La distribuzione Gaussiana utilizzata nel test prevede un offset di ritardo costante di 1 ora e 30 minuti, con una deviazione standard di 15 minuti, permettendo di simulare realisticamente situazioni di carico elevato per un breve periodo di tempo.

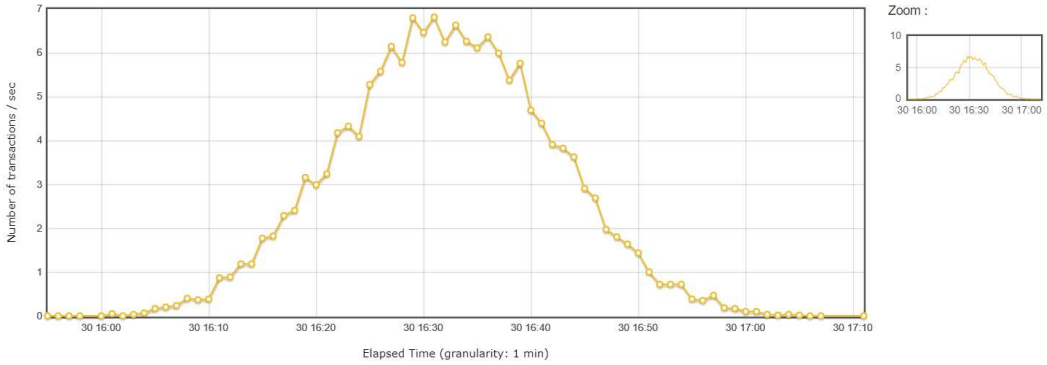

Figura 5.18: Run 5: 15 login al secondo per 2h con distribuzione Gaussiana

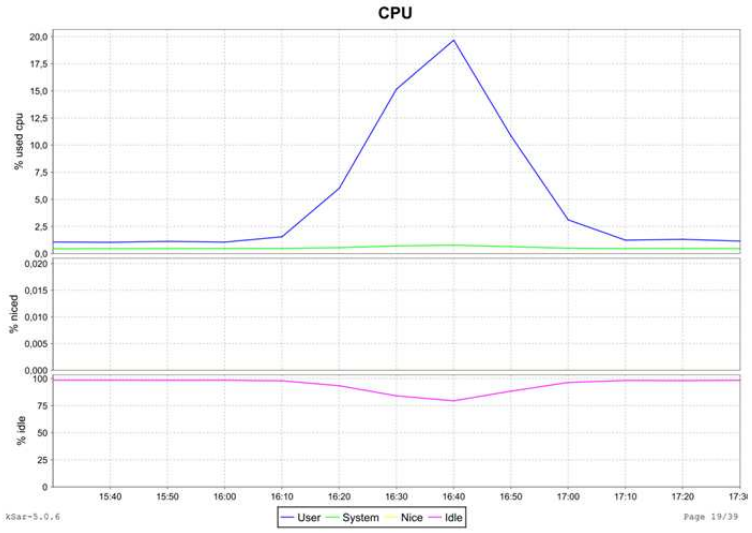

Figura 5.19: Run 5: Consumo CPU

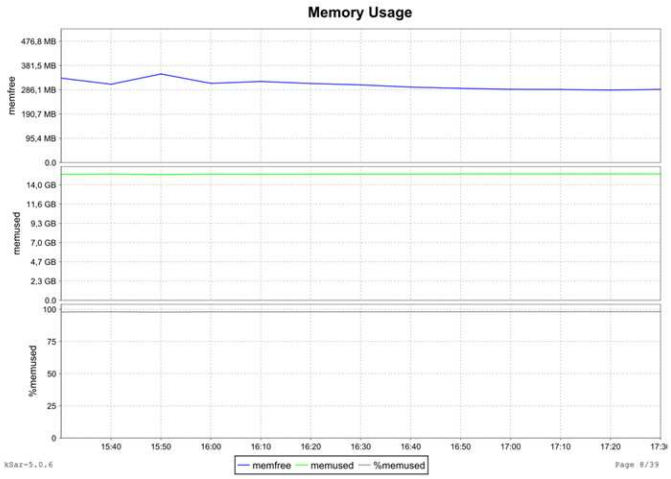

Figura 5.20: Run 5: Consumo Memoria

ordine	Requests	Executions			Response Times (ms)			Throughput
	Label	#Samples	KO	Error %	Average	Min	Max	Transactions/s
1	Request resource	10000	0	0.00%	7.97	1	1475	2.21
2	Authorize	10000	0	0.00%	21.05	11	3029	2.21
3	Login with 1FA	10000	0	0.00%	163.42	14	14429	2.21
4	Login with 2FA	10000	0	0.00%	60.78	31	5628	2.21
5	Token endpoint	10000	0	0.00%	126.88	94	25703	2.21
6	Access to protected resource	10000	0	0.00%	106.47	90	1322	2.21

Figura 5.21: Run 5: Tabella riepilogativa

Analisi Run 5: Dall'analisi della tabella riepilogativa 5.21, si osserva che il quinto test, che ha simulato 15 processi di login al secondo per un breve periodo di 2 ore utilizzando la distribuzione Gaussiana, ha raggiunto gli esiti migliori. Infatti, tutti i 10.000 processi di login sono stati completati con successo e, dalle figure 5.19 e 5.20, si nota che l'utilizzo delle risorse è stato elevato solo durante i periodi di picco, rimanendo comunque entro i limiti previsti.
Questo test ha dimostrato la capacità del sistema di mantenere prestazioni ottimali anche in presenza di carichi elevati durante le ore di punta, completando con successo tutti i processi nonostante il momento di intenso carico, evidenziando una resilienza significativa. Tale risultato dimostra che il sistema, in situazioni reali simulate attraverso la distribuzione Gaussiana, è in grado di gestire improvvisi picchi di carico per un breve periodo di tempo.

5.2.6 Run 6

Nel sesto test, sono stati simulati 15 processi di login al secondo per un periodo di 7 ore, utilizzando la distribuzione Gaussiana.
Il grafico in figura 5.22 illustra la distribuzione del numero di transazioni effettuate nell'arco di 7 ore.
La distribuzione Gaussiana utilizzata nel test prevede un offset di ritardo costante di 3 ore, con una deviazione standard di 1 ora, permettendo di simulare realisticamente situazioni di carico elevato per un lungo periodo di tempo.

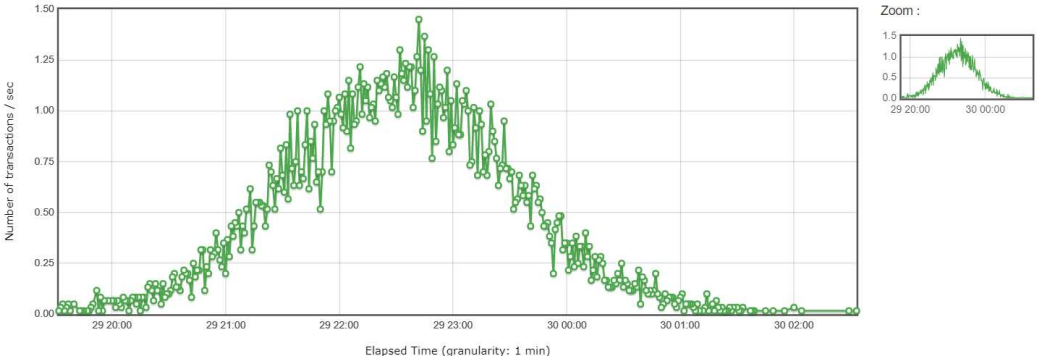

Figura 5.22: Run 6: 15 login al secondo per 7h con distribuzione Gaussiana

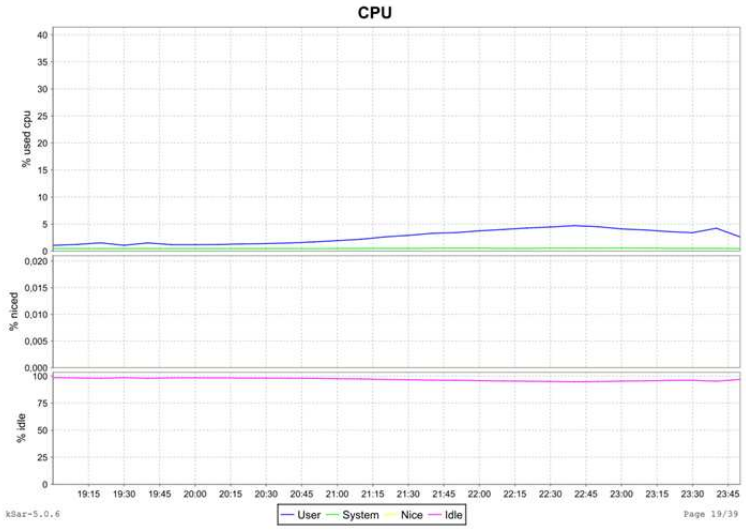

Figura 5.23: Run 6: Consumo CPU

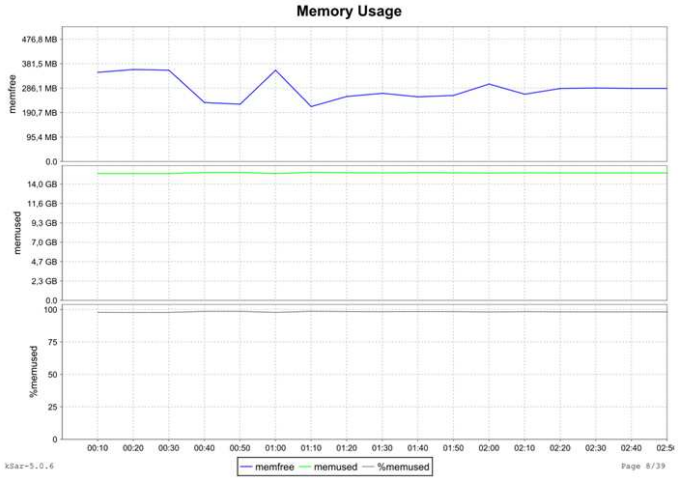

Figura 5.24: Run 6: Consumo Memoria

ordine	Requests	Executions			Response Times (ms)			Throughput
	Label	#Samples	KO	Error %	Average	Min	Max	Transactions/s
1	Request resource	10000	0	0.00%	6.80	1	1897	0.40
2	Authorize	10000	0	0.00%	19.15	12	1172	0.40
3	Login with 1FA	10000	0	0.00%	230.88	15	23647	0.40
4	Login with 2FA	10000	0	0.00%	90.63	39	23908	0.40
5	Token endpoint	10000	0	0.00%	120.65	94	12134	0.40
6	Access to protected resource	10000	0	0.00%	101.76	91	8830	0.40

Figura 5.25: Run 6: Tabella riepilogativa

Analisi Run 6: Dall'analisi della tabella riepilogativa 5.25, si osserva che il sesto ed ultimo test, che ha simulato 15 processi di login al secondo per un lungo periodo di 7 ore utilizzando la distribuzione Gaussiana, ha raggiunto risultati ottimali. Anche qui, tutti i 10.000 processi di login sono stati portati a termine con successo e, dalle figure 5.19 e 5.20, si osserva che l'impiego delle risorse è stato elevato solo durante i periodi di picco, rimanendo entro i limiti stabiliti.
Analogamente al test precedente, il sistema ha mantenuto prestazioni elevate e ha completato con successo tutti i processi anche per un lungo periodo, dimostrando una resilienza ancora maggiore. Questo risultato suggerisce che il sistema è capace di gestire carichi elevati non solo per brevi periodi, ma anche per periodi prolungati, garantendo affidabilità e stabilità nel tempo.

Capitolo 6

Conclusione e sviluppi futuri

In questo capitolo verranno riassunti i risultati ottenuti dal sistema realizzato e i potenziali sviluppi futuri.
La prima sezione sintetizza i risultati raggiunti.
La seconda sezione evidenzia il contributo offerto alla ricerca.
Infine, la terza sezione discute gli sviluppi futuri.

6.1 Conclusioni

Con il presente lavoro di tesi, abbiamo implementato un'infrastruttura di autenticazione centralizzata per la gestione delle utenze e dei diritti di accesso. Questa infrastruttura è stata progettata per soddisfare le esigenze di utenti non-consumer, come notai e commercialisti, utilizzando il protocollo OpenID Connect (OIDC) e un processo di autenticazione basato su token.
L'infrastruttura proposta si basa su principi di sicurezza moderni e consolidati, come l'autenticazione multifattoriale (MFA), il protocollo di autorizzazione OAuth 2.0 e il framework di sicurezza OpenID Connect. Questi strumenti insieme formano un'architettura che non solo protegge le risorse aziendali ma garantisce anche una user experience fluida e sicura per gli utenti finali.
Abbiamo dimostrato come la federazione tra Axway API Gateway e Oracle Access Manager possa fornire un sistema di autenticazione robusto e sicuro. La procedura inizia con un'autenticazione di primo livello tramite username e password, seguita dalla richiesta di un secondo fattore di autenticazione. Oracle Access Manager, agendo come tramite e orchestratore, gestisce questo processo sfruttando le funzionalità di Multi Factor Authentication (MFA) esposte dal Register API del Portale Utenti Esterni. Una volta completata l'autenticazione con entrambi i fattori, Oracle Access Manager rilascia i token di autenticazione necessari per la profilazione dell'utente e per l'accesso alle API del Portale Utenti Esterni. Questi accessi sono intermediati da Axway API Gateway, che verifica anche la validità e la scadenza dei token utilizzati.
Insieme a Technology Reply Roma, abbiamo implementato e testato la soluzione ad-hoc appena descritta sulla base di use-case forniti dal cliente, evidenziando i seguenti punti chiave:

- **Sicurezza**: L'uso combinato di OIDC e MFA garantisce un alto livello di sicurezza, proteggendo le informazioni sensibili degli utenti e prevenendo accessi non autorizzati.
- **Centralizzazione**: Una gestione centralizzata delle credenziali e dei diritti di accesso semplifica l'amministrazione del sistema e riduce il rischio di errori e inconsistenze.
- **Flessibilità**: L'infrastruttura è progettata per essere flessibile e adattabile a diversi ambienti professionali, garantendo così una soluzione scalabile e personalizzabile.
- **Conformità**: Il rispetto delle normative sulla privacy e sulla protezione dei dati.

In conclusione, possiamo affermare che la soluzione proposta in questo lavoro ha confermato che un'infrastruttura di autenticazione centralizzata che utilizza OIDC e MFA, integrata con Axway API Gateway e Oracle Access Manager, offre una soluzione efficace per la gestione delle utenze e dei diritti di accesso in contesti professionali per utenti non-consumer. Con OIDC e il processo di autenticazione basato su token, abbiamo delineato un percorso che coniuga sicurezza, efficienza e conformità normativa, aprendo la strada a un futuro digitale più sicuro e accessibile per diverse tipologie di utenti.

6.2 Contributi della ricerca

Arrivati a questo punto, è fondamentale evidenziare il contributo significativo che questo progetto può apportare allo sviluppo di soluzioni basate su infrastrutture di autenticazione utilizzando il protocollo OpenID for Verifiable Credentials (OID4VC). Tuttavia, è importante sottolineare che il protocollo OID4VC è attualmente in fase di sviluppo attivo e non ha ancora raggiunto una completa maturità.
L'adozione del protocollo OID4VC potrebbe consentire al sistema di sfruttare appieno le potenzialità delle Verifiable Credentials, offrendo agli utenti un metodo per convalidare la propria identità senza rivelare informazioni personali superflue. Questo approccio, orientato alla privacy, si allinea perfettamente con i principi del "minimo indispensabile" e del "consenso informato", garantendo che solo le informazioni strettamente necessarie siano condivise durante il processo di autenticazione.
Guardando al futuro, l'implementazione di OID4VC apre nuove prospettive nel campo dell'identità digitale, permettendo di affrontare in modo proattivo le sfide legate alla sicurezza e alla privacy. Con l'evoluzione delle tecnologie, OID4VC si presenta come una soluzione promettente per migliorare la gestione dell'identità digitale, offrendo un sistema più sicuro e rispettoso della privacy degli utenti.

6.3 Sviluppi futuri

In questa sezione, esploreremo le possibili strade che potrebbero essere intraprese al fine di apportare delle migliorie al lavoro svolto.

6.3.1 Penetration Testing

In futuro potrebbe essere utile effettuare una fase di Penetration Testing in modo da trovare e prevenire diverse vulnerabilità che potrebbero essere sfruttate a fini fraudolenti, come accessi non consentiti da persone non autorizzate, in modo da valutare anche l'efficienza del meccanismo di autenticazione implementato.

6.3.2 Monitoraggio e ottimizzazione risorse

In relazione alle performance, si potrebbero monitorare le risorse necessarie e utilizzate attraverso un software dedicato, per garantire prestazioni ottimali. Una piattaforma software adatta a questo contesto è 'Dynatrace', che permette di tracciare in tempo reale l'uso delle risorse, rilevando così anomalie e prevenendo disservizi.

6.3.3 Analisi log applicativi

Un altro strumento essenziale per prevenire disservizi è 'Kibana', che permette il recupero e l'analisi dei log applicativi. Infatti, attraverso le sue dashboard interattivi, è possibile facilmente filtrare, cercare e visualizzare i dati in tempo reale che passano sull'infrastruttura, trasformando i log grezzi in informazioni comprensibili. Questo permette di identificare rapidamente anomalie sui servizi esposti e diagnosticare diversi problemi dell'applicazione.

6.3.4 Integrazione con altri portali

Il lavoro di tesi ha portato alla realizzazione di un portale dedicato a diverse tipologie di utenti non-consumer, ma attualmente accessibile ai soli notai. In futuro, sarà necessario dare accesso anche ad altre figure professionali, come ad esempio i commercialisti. Pertanto, ci saranno diversi portali ma la stessa infrastruttura di single sign-on.

Bibliografia

[1] Che cos'è l'Identity and Access Management (IAM) https://www.oracle.com/it/security/identity-management/what-is-iam/

[2] Mowers D., Baladi M., Verwolf P., Steven A., Microsoft Identity and Access Management Series - "Fundamental Concepts". Microsoft Solutions for Security and Compliance (MSSC), 2006.

[3] Kearns. D., Defining identity, persona, role. Network World, 2003.

[4] Introduction of Single Sign On (SSO) https://www.geeksforgeeks.org/introduction-of-single-sign-on-sso/

[5] Autenticazione a due fattori https://www.giorgiosbaraglia.it/autenticazione-a-due-fattori-cose-come-e-perch

[6] Roychowdhuri S., Oracle Fusion Middleware – "Installation Guide for Oracle Unified Directory 11g Release 2 (11.1.2)". Oracle Corporation, March 2014.

[7] Sun Java(TM) System Directory Server 5.2 Technical Overview – "Introduction to Directory Services and Directory Server". Sun Microsystems, 2005.

[8] Cos'è Oracle HTTP Server? https://docs.oracle.com/en/middleware/fusion-middleware/web-tier/12.2.1.4/administer-ohs/

[9] Guida a Oracle Access Manager https://docs.oracle.com/cd/E17904_01/admin.1111/e15478/keytool.htm#AIAAG2090

[10] Wilcox M., Remillon E., Oracle White Paper - "Oracle Unified Directory". Oracle Corporation, December 2013.

[11] Introduction to API Gateway
https://docs.axway.com/bundle/axway-open-docs/page/docs/api_mgmt_overview/api_mgmt_components/apigateway/index.html

[12] Cos'è Cassandra
https://www.geekandjob.com/wiki/cassandra

[13] OpenID Connect: cos'è, a cosa serve e perché è importante https://www.cybersecurity360.it/soluzioni-aziendali/openid-connect-cose-a-cosa-serve-e-perche-e-importante/

[14] OAuth 2.0: cos'è e come funziona https://www.cybersecurity360.it/soluzioni-aziendali/oauth-2-0-cose-e-come-funziona-lo-standard-aperto-per-lautenticazione-sicura-

[15] cos'è OpenID Connect for Verifiable Credentials https://openid.net/sg/openid4vc/

[16] Cos'è l'identità federata? https://www.fortinet.com/it/resources/cyberglossary/federated-identity

[17] Introduzione alla Self-Sovereign Identity https://www.selfsovereignidentity.it/introduzione-alla-self-sovereign-identity/