Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Informatica
indirizzo Cybersecurity

A.a. 2023/2024

Sessione di Laurea Dicembre 2024

La rivoluzione Open Banking:

Storia, motivazioni e stato dell'arte del modello e caratteristiche del servizio di NameCheck

Relatori:
Danilo Bazzanella
Piergiorgio Rettaroli

Candidato:
Simone Costanzi

Ai miei genitori, ai miei zii e ai miei nonni che mi hanno sempre supportato non facendomi mai mancare nulla.

Indice

1 Introduzione

La Direttiva dei Sistemi di Pagamento (PSD2, Direttiva EU 2015/2366) è una direttiva europea che regolamenta i servizi di pagamento e i gestori dei servizi di pagamento all'interno dell'Unione europea. In particolare l'obiettivo della PSD2, che ha abrogato la precedente Direttiva dei Sistemi di Pagamento (PSD, Direttiva EU 2007/64/EC), consiste in una maggiore integrazione in Europa dei sistemi di pagamento, in modo da abilitare nuovi servizi e aumentare la sicurezza e la protezione degli utenti. Le principali novità sono sintetizzabili nelle seguenti quattro tipologie:

- Servizi di disposizione di ordini di pagamento online (Payment Initiation Services – PIS) che consentono di avviare un pagamento online tramite un prestatore di servizi di pagamento diverso da quello presso il quale si detiene il conto;
- Servizi di informazione sui conti di pagamento online (Account Information Services – AIS) in base ai quali si possono ottenere informazioni aggregate su uno o più conti online detenuti anche presso istituti diversi;
- Servizi di conferma disponibilità fondi previsti nel caso di pagamenti effettuati con carte di debito emesse da un operatore diverso rispetto a quello presso il quale si detiene il conto (cioè il CISP, Card Issuer Service Provider);
- Fornitori di servizi di emissione di strumenti di pagamento (Payment Instrument Issuing Service Provider - PIISP) i quali hanno il compito di verificare la disponibilità dei fondi sul conto di un cliente per garantire che ci siano sufficienti risorse per coprire un pagamento. Questo ruolo è simile a quello del CISP, ma può includere una gamma più ampia di strumenti di pagamento oltre alle carte, come portafogli digitali o altri metodi di pagamento elettronico;
- Parametri di obbligatorietà per la SCA (Strong Customer Authentication).

Tutti i servizi sono erogati, attraverso le API software, con il paradigma del cosiddetto Open Banking, in modo da poter scambiare dati tra i vari soggetti in modo veloce e aperto, per migliorare o creare nuovi prodotti o servizi digitali. L'entrata in vigore della direttiva europea PSD2 nel 2018 ha quindi segnato l'inizio della rivoluzione (ancora in pieno sviluppo) di Open Banking. Con l'Open banking si vuole creare una spaccatura con il passato permettendo a operatori di terze parti (Trusted-party providers, TPP) di interrompere la posizione di dominio degli istituti finanziari e bancari nelle transazioni economiche. L'obiettivo che si vuole raggiungere è di permettere a società terze di fornire servizi aggiuntivi al cliente (tendenzialmente non forniti dalle tradizionali società bancarie) permettendogli, ad esempio, di avere a portata di click la situazione patrimoniale di tutti i conti a lui intestati (magari in diversi istituti di credito), avere una maggiore comprensione delle entrate e delle uscite, grafici utili ad analizzare quanto una tipologia di spesa influisca sul proprio stile di vita, suggerimenti finanziari resi possibili mediante l'utilizzo di algoritmi AI che analizzano il profilo utente andando a creare un "profilo personalizzato" e consigliando come investire al meglio i propri risparmi in base ai piani presenti nei vari istituti bancari, assicurativi ecc ecc...
Per riassumere, mediante queste applicazioni di terze parti è possibile tenere d'occhio più facilmente tutta la propria situazione economica ed effettuare qualunque movimento economico (che sia un VRP o un movimento manuale).
Dopo questo breve paragrafo avente unicamente lo scopo di introdurre l'argomento si passerà prima a fornire gli elementi necessari per comprendere al meglio l'intera trattazione e poi ad analizzare nel dettaglio cosa sia un TPP, che cosa si intenda per Open Banking Directory, il processo di Dynamic Client Registration e il servizio di Confirmation of Payee.

2 Perchè la cifratura

I pagamenti digitali per poter essere considerati autentici e quindi evitare tentativi di frode hanno bisogno di metodi e strategie (tra i quali la cifratura) che garantiscano l'integrità, la confidenzialità e l'autenticazione della transazione. Applicare correttamente la cifratura fa sì che nessuna parte non autorizzata possa vedere i dati del pagamento e modificarli per il proprio tornaconto personale.
Prima di iniziare a parlare di firma digitale è necessario riportare alcune pillole di base di cifratura (per evitare una trattazione eccessivamente verbosa e al di fuori del tema centrale è stato deciso di riportare solamente i concetti basilari; si lascia al lettore la volontà di approfondire gli aspetti matematici e le implementazioni):

2.1 Cenni di crittografia

La crittografia è una "tecnica" o meglio l'insieme dei metodi utili a rendere un messaggio (plaintext) non comprensibile nei confronti di parti terze non autorizzate a prenderne visione. Il testo cifrato (ciphertext) gode pertanto della proprietà di confidentiality (confidenzialità o riservatezza). Semplificando al massimo il concetto esistono due principali tipi di cifratura:

- Cifratura a chiave simmetrica: in cui la chiave utilizzata per cifrare il messaggio è la stessa che viene impiegata per decifrare;
- Cifratura a chiave asimmetrica: in cui esiste una coppia di chiavi pubblica e privata. Il mittente del messaggio utilizzerà la chiave pubblica del destinatario per cifrare il messaggio, mentre il destinatario utilizzerà la propria chiave privata per decifrarlo.

L'analisi tecnica completa della cifratura e delle sue proprietà esula dallo scopo del presente documento. Pertanto, si procede ora a descrivere la storia e la necessità della firma digitale.

2.2 Origini della Firma Digitale

Le origini della firma digitale possono essere fatte risalire agli anni '70, quando la necessità di garantire la sicurezza e l'autenticità delle comunicazioni elettroniche divennero sempre più pressanti. Tuttavia, il vero sviluppo della firma digitale come la conosciamo oggi si ebbe negli anni '90, con l'avvento di tecnologie avanzate di crittografia e la diffusione di standard e protocolli per la sicurezza informatica. Uno dei primi documenti legali a riconoscere l'efficacia giuridica della firma digitale è stata la legge americana ESIGN (Electronic Signatures in Global and National Commerce Act) del 2000, seguita da normative simili in molte altre giurisdizioni in tutto il mondo, infatti troviamo:

- **La legge sulla tutela delle informazioni personali e sui documenti elettronici del Canada**, meglio conosciuta come PIPEDA, la quale stabilisce come i dati dei consumatori debbano essere trattati in materia di privacy e sicurezza. La legge ha lo scopo di promuovere l'e-commerce, aumentando la fiducia dei consumatori nei rapporti con le imprese del settore privato canadese;
- **L'Electronic Transactions Act australiano del 1999**, che fornisce un quadro normativo che facilita l'uso e assicura la validità delle transazioni elettroniche. Questo atto è stato modificato nel 2011, aumentando le tutele per consumatori e imprese;
- **La direttiva europea 199/93/EC**, che è stata la prima normativa su larga scala sulla firma elettronica a entrare in vigore nell'UE. La direttiva è simile all'E-Sign Act degli USA. Protegge imprese e consumatori che decidono di concludere affari online usando firme elettroniche e documenti digitali;
- **Electronic Communication Act del 2000 del Regno Unito**, che fornisce garanzie per la validità legale della firma elettronica per quanto riguarda i servizi di crittografia, le comunicazioni elettroniche e la memorizzazione dei dati di persone che risiedono in Inghilterra, Scozia, Galles e Irlanda del Nord;
- **L'Electronic Signatures Regulations 2002**, che è stato redatto di pari passo con l'Electronic Communications Act del 2000 in attuazione del recepimento della Direttiva UE del 1999. Sulla base dei regolamenti del 2002, una firma elettronica nel Regno Unito è definita come "dati in formato elettronico allegati o associati logicamente ad altri dati elettronici e che servono come

metodo di autenticazione". La norma si concentra sulla funzione di autenticazione informatica, indipendentemente dalla tecnologia usata;

- **L'Information Technology Act dell'India, noto anche come IT Act o ITA-2000**, che è stato introdotto per rispondere alle crescenti preoccupazioni delle imprese che effettuano transazioni on-line in India. La legge prevede il riconoscimento legale per i documenti elettronici e le firme digitali ed è stata emendata dal parlamento indiano nel 2008;
- **La legge sulle firme elettroniche e attività di certificazione** (Act on Electronic Signatures and Certification Business) che promuove l'uso di documenti elettronici e afferma che le firme elettroniche sono importanti per l'economia della nazione e per la qualità della vita dei cittadini;
- **La Legge sulla firma elettronica della Repubblica Popolare Cinese**, la quale, adottata nel 2004, mira a standardizzare il modo in cui le firme elettroniche sono create in Cina e a tutelare chi esegue transazioni on-line, fornendo il quadro necessario per garantire che le firme elettroniche siano giuridicamente vincolanti;
- **L'Electronic Transactions Act 2002** (divenuto poi effettivo nel 2003) della Nuova Zelanda, che ha riconosciuto il ruolo importante che l'e-commerce e le firme elettroniche giocheranno nell'economia futura del Paese. Per favorire ciò, la legge prevede tutele per i consumatori e le imprese e consente la comunicazione elettronica tra le imprese e il governo.

2.3 Utilizzi Attuali della Firma Digitale

Oggi, la firma digitale è ampiamente utilizzata in una vasta gamma di settori e applicazioni, offrendo numerosi vantaggi rispetto alle firme tradizionali su carta. Alcuni dei suoi utilizzi più comuni includono:

1. **Contratti e Documenti Legali:** la firma digitale è ampiamente accettata per la conclusione di contratti legali e la firma di documenti importanti, garantendo l'autenticità delle firme e la non alterazione dei contenuti;
2. **Transazioni Finanziarie:** nelle transazioni finanziarie online, la firma digitale svolge un ruolo fondamentale nella conferma dell'autenticità delle parti coinvolte e nell'assicurare l'integrità delle transazioni stesse;
3. **Documenti Amministrativi e Certificati:** settori come la pubblica amministrazione, la sanità e l'istruzione utilizzano sempre più la firma digitale per la gestione efficiente dei documenti e la riduzione della burocrazia;
4. **Firme Elettroniche nei Processi di Lavoro:** le aziende adottano la firma digitale per semplificare i processi interni, come l'approvazione di documenti, le firme sui contratti di lavoro e altro ancora, aumentando l'efficienza operativa e riducendo i tempi di elaborazione.

2.4 Il Futuro della Firma Digitale

E' possibile riscontrare un esempio di applicazione della firma digitale nella tecnologia Blockchain: la tecnologia blockchain, per semplificare la trattazione, può essere descritta come un database distribuito tra peer, immodificabile dal singolo utente o da gruppi di utenti. In questo ambito la firma digitale risulta fondamentale nel momento in cui il blocco (contenente le transazioni) viene aggiunto nel database da parte del miner. La firma digitale e le applicazioni della cifratura risultano fondamentali in questo contesto per proteggere i dati delle transazioni e garantire autenticità e integrità del lavoro svolto dal miner (oltre a poter così ottenere la "ricompensa" per il lavoro svolto).

2.5 Cos'è la Firma Digitale

Fino a questo punto sono stati discussi la storia, l'importanza nelle transazioni, gli utilizzi in documenti ufficiali e prospettive future della firma digitale, ma quindi, che cos'è la firma digitale?
Secondo la guida AgID (Agenzia per l'Italia digitale), la firma digitale è definita come: *"un particolare tipo di firma elettronica qualificata basata su un sistema di chiavi asimmetriche a coppia, una pubblica e una privata, che consente al titolare tramite la chiave privata e al destinatario tramite la chiave pubblica, rispettivamente, di rendere manifesta e di verificare la provenienza e l'integrità di un documento*

informatico o di un insieme di documenti informatici ". Apposta su un documento informatico la firma digitale garantisce le seguenti proprietà:

Autenticità: garantisce l'identità del sottoscrittore del documento;
Integrità: attesta che il documento non sia stato modificato dopo la sottoscrizione;
Non ripudiabilità: attribuisce piena validità legale al documento, pertanto non potrà essere ripudiato dal sottoscrittore.

Quindi, apporre una firma digitale su un documento implica che:

- il sottoscrittore non potrà disconoscere il documento stesso, in quanto, dopo la sottoscrizione, non potrà essere assolutamente modificato;
- il sottoscrittore è l'unico titolare del certificato di firma digitale (in quanto detiene il dispositivo e le credenziali per accedervi);
- il sottoscrittore garantisce la veridicità e la correttezza delle informazioni riportate nel certificato (dati anagrafici del titolare).

Per poter "ottenere" una firma digitale valida occorre rivolgersi a specifici enti accreditati in grado di certificare l'autenticità e l'identità del soggetto firmatario. La normativa (Codice dell'Amministrazione Digitale (CAD), emanato con il Decreto Legislativo 7 marzo 2005, n. 82) definisce così la tecnologia da utilizzare per ottenere la firma digitale:

- si deve basare su un sistema elettronico a due chiavi crittografiche simmetriche, una pubblica e una privata, attribuite in maniera univoca a un soggetto, che ne è il titolare. In tal modo le due chiavi consentono al titolare (tramite la chiave privata) e al destinatario (tramite chiave pubblica) di verificare la provenienza e l'integrità del documento informatico sottoscritto digitalmente;
- È associata a uno specifico certificato digitale emesso da un soggetto autorizzato dallo Stato;
- Viene creata tramite un dispositivo assolutamente sicuro (generalmente una smart card, ma è possibile utilizzare anche Business Key o Firma Remota).

2.6 Firma digitale e firma elettronica

Nel linguaggio comune, i termini firma elettronica e firma digitale vengono spesso utilizzati come sinonimi. In realtà, 'firma digitale' è sinonimo di un particolare tipo di firma elettronica, ovvero la Firma Elettronica Qualificata (FEQ). Oltre alla FEQ o firma digitale esistono altre firme elettroniche che non posseggono tutti i requisiti di sicurezza e autenticità della prima. Il CAD (Codice dell'Amministrazione Digitale) distingue 3 tipologie di firme elettroniche:

- **Firma Elettronica Semplice (FES):** è definita come *"l'insieme dei dati in forma elettronica, allegati oppure connessi tramite associazione logica ad altri dati elettronici, utilizzati come metodo di identificazione informatica"*. Ad esempio, un comune PIN associato ad una carta magnetica (tipo bancomat) oppure credenziali di accesso (user e password), che non possiedono i requisiti minimi di sicurezza richiesti ai fini probatori;
- **Firma Elettronica Avanzata (FEA):** questo tipo di firma rispetta alcuni **requisiti minimi di sicurezza**, in quanto consente di **identificare il firmatario** e rilevare se i dati che compongono il documento siano stati modificati successivamente all'apposizione della firma stessa (**integrità**). Ad esempio, la firma biometrica apposta su un tablet, che per alcuni documenti assume valore probatorio;
- **Firma Elettronica Qualificata (FEQ) o Firma Digitale:** è una firma elettronica avanzata **basata su un certificato qualificato** realizzato tramite un dispositivo di sicurezza tipo una smart card o un token usb. Questa è **l'unica tipologia di firma elettronica che ha piena validità giuridica e può essere utilizzata per firmare qualsiasi atto**. **La differenza sostanziale** tra le altre firme elettroniche e la firma digitale sta nella sua **efficacia giuridica**: la firma digitale equivale in tutto e per tutto ad una sottoscrizione autografa con piena efficacia probatoria e legale, ed è l'unica che in sede di giudizio inverte l'onere della prova di autenticità in capo al firmatario (il firmatario che deve dimostrare che la firma digitale non è autentica, piuttosto che l'altra parte dover dimostrare che lo è).

2.7 Differenza tra firma digitale e altre firme elettroniche

Il tema della validità legale della firma elettronica e dell'efficacia probatoria è sintetizzato nello schema che segue:

Tipo di firma	**Definizione ed esempio**	**Efficacia Probatoria ai fini giuridici**	**Atti tra privati di cui Art. 1350 c.c. (13)**	**Atti tra privati di cui Art. 1350 c.c. (1-12)**	**Documenti e atti delle PA**
Firma Elettronica Semplice (FES)	Firma debole tramite sistemi di identificazione del firmatario "non certi" Esempio: PIN	A discrezione del Giudice	NO	NO	NO
Firma Elettronica Avanzata (FEA)	Firma forte tramite sistemi e/o dispositivi personali che identificano il firmatario senza certificati di integrità Esempio: Firma grafometrica su dispositivi elettronici	SI	SI	NO	NO
Firma Elettronica Qualificata (FEQ) o Firma Digitale	Firma forte tramite software e dispositivi personali che identificano il firmatario con certificati di integrità Esempio: Firma tramite Tessera Sanitaria	SI	SI	SI	SI

2.8 Come funziona tecnicamente la firma digitale

Dal punto di vista tecnico, la firma digitale utilizza la tecnica della crittografia asimmetrica a doppia chiave (una pubblica e una privata), applicando particolari funzioni matematiche, come la funzione di hash.
Inserire una firma digitale in un documento elettronico significa:

- **Generare l'impronta digitale**: nella fase iniziale, viene generata l'impronta digitale usando la funzione di hash, la quale -per definirla brevemente- è una funzione matematica che produce in output una stringa di lunghezza fissata -a seconda dell'algoritmo impiegato, ad esempio SHA256, SH512 -, indipendentemente dalle dimensioni della stringa originale. L'impronta (o digest) per ogni documento è unica e non invertibile (per definizione di funzione di hash): questo significa che modificando anche un solo carattere del testo si otterrà un'impronta diversa;

- **Generare la firma**: viene applicata la cifratura con chiave privata dell'impronta digitale precedentemente generata. La firma sarà quindi legata, da un lato (tramite la chiave privata usata per la generazione) al soggetto sottoscrittore, e dall'altro (tramite l'impronta) al testo sottoscritto. L'impronta, e non l'intero documento, viene cifrata con la chiave privata del mittente ottenendo la generazione della firma digitale;
- **Apporre la firma**: viene aggiunta la firma del sottoscrittore in una posizione predefinita, generalmente alla fine del documento. Al destinatario del documento verranno spediti:
  - il documento firmato secondo la procedura tecnica descritta;
  - il certificato che deve essere rilasciato dall'ente di certificazione a garanzia della titolarità della chiave pubblica necessaria a decifrare la firma digitale

Quando il destinatario aprirà il documento utilizzerà un apposito software per la verifica della firma digitale che acquisirà dal certificato annesso al documento firmato la chiave pubblica del mittente. Tramite questa chiave, viene decifrata la stringa della firma digitale che produrrà come risultato l'impronta del documento. Il destinatario poi farà passare la funzione hash sul documento originario e genererà l'impronta.
A questo punto se le due impronte coincideranno il destinatario sarà sicuro dell'integrità e dell'autenticità del documento ricevuto.

2.9 Il certificato di firma digitale

Il **certificato di firma digitale** in pratica lega l'identità di una persona fisica alla chiave pubblica. È l'Autorità di Certificazione che si fa garante della veridicità di questo legame e dell'identità del titolare attraverso un incontro *de visu* tra un suo incaricato (*Registration Authority*) e l'utente.
La CA (Certification Authority) ha il dovere di permettere a chiunque l'accesso ad un **Registro dei Certificati** al fine di consentire l'identificazione del titolare della chiave pubblica e di mantenere la lista dei certificati sospesi e revocati costantemente aggiornata. Ogni certificato deve riportare i seguenti dati:

- informazioni per identificare in modo univoco il possessore di una chiave pubblica (ad esempio nome e cognome);
- il valore della chiave pubblica;
- il periodo di validità temporale del certificato;
- la firma digitale della autorità di certificazione con cui si assicura autenticità della chiave ed integrità delle informazioni contenute nel certificato.

Un soggetto può essere titolare di uno o più certificati a seconda dei ruoli che riveste come sottoscrittore di documenti informatici. Questo significa che una persona può avere anche due o più certificati se ha bisogno, ad esempio, di firmare documenti come semplice cittadino o come amministratore di una società.
Per non entrare troppo nel dettaglio di come questi certificati vengano creati e il processo di controllo di validità del certificato (che implicherebbe descrivere nel dettaglio il funzionamento dei server OCSP e le liste CRL) ci limiteremo a dire che ogni certificato ha un suo periodo di validità e in caso di dubbi sull'integrità del certificato stesso e sul possesso della chiave privata associata a quel certificato sarà possibile chiedere la revoca del certificato.

2.10 Necessità della firma digitale

Come è stato accennato in precedenza un documento firmato digitalmente gode delle seguenti proprietà:

- autenticità
- integrità
- non ripudiabilità

L'ultima proprietà è fondamentale ed è l'elemento cardine che ha permesso alla firma digitale di diventare importante e di poter essere applicata in molti contesti.
Consideriamo ora il caso in cui due utenti (Alice e Bob) decidano di scambiarsi un documento utilizzando il meccanismo di cifratura simmetrica come nella seguente immagine:

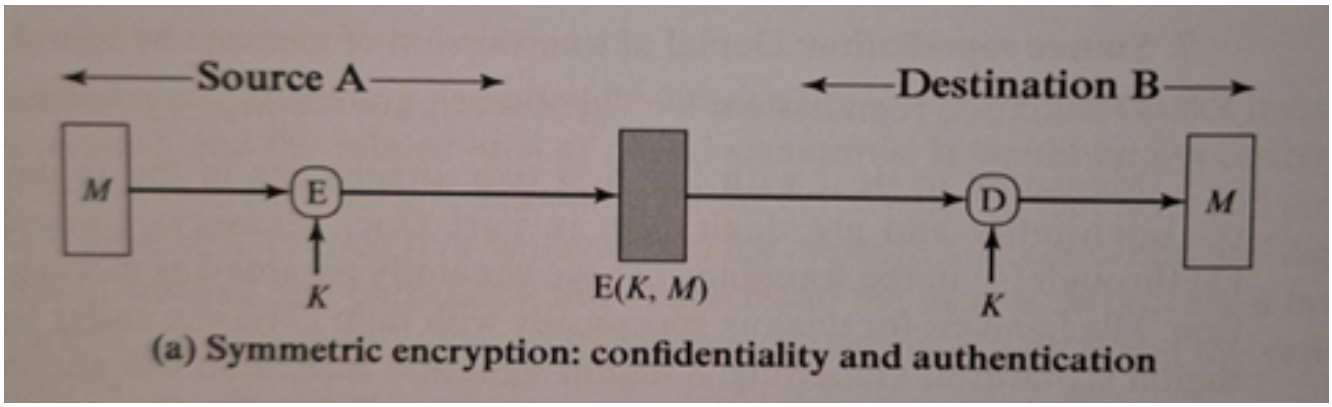


Figure 1: Esempio di cifratura simmetrica

Alice cifra il messaggio M utilizzando la chiave simmetrica K e invia il messaggio cifrato a Bob. Quest'ultimo provvederà poi a decifrarlo utilizzando la stessa chiave di Alice. Analizzando meglio questo scenario si possono evidenziare le seguenti criticità:

- Alice può creare un messaggio diverso e dichiarare di averlo ricevuto da Bob. Alice potrebbe quindi appendere un authentication code usando la chiave che condivide con Bob;
- Bob potrebbe negare di aver inviato il messaggio. Lui potrebbe affermare che sia stata Alice a creare il messaggio; infatti, non c'è nessun modo per provare che Bob sia il vero autore del messaggio.

Pertanto, la firma digitale permette di risolvere situazioni in cui mittente e destinatario non si fidino l'uno dell'altro.

2.11 Funzionamento firma digitale

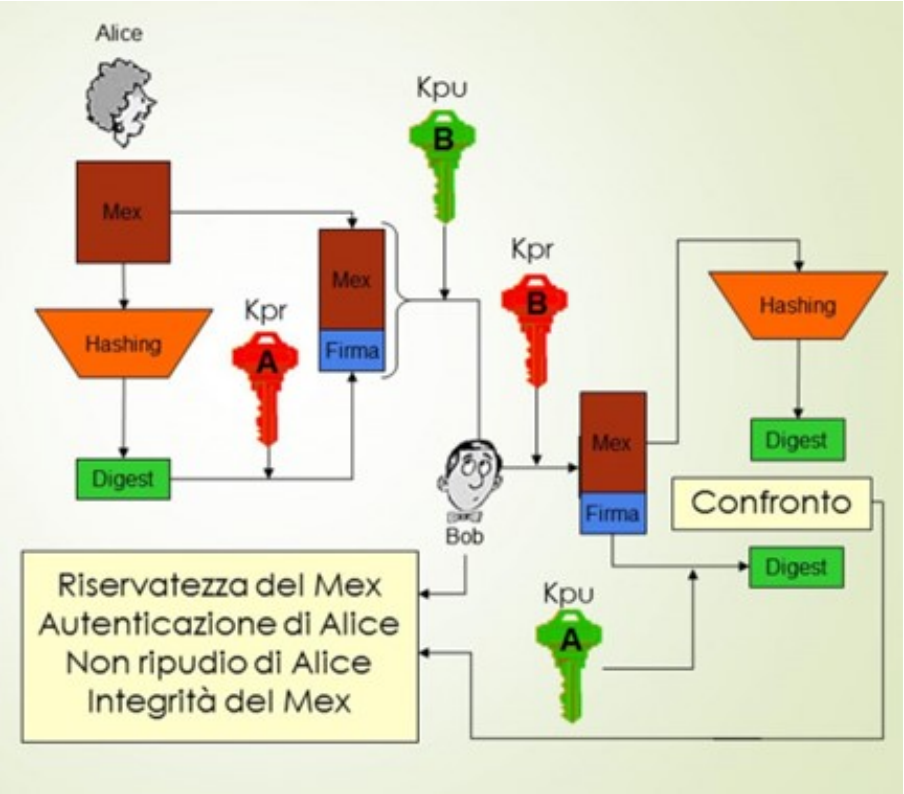


Figure 2: Esempio di Firma digitale

Per poter applicare il concetto di firma digitale è possibile seguire i seguenti step che descrivono la precedente immagine (per prima cosa ignoriamo la cifratura del messaggio concatenato al digest che Alice manda a Bob) considerando che entrambe le parti devono essersi accordate sulla funzione di hash da impiegare:
ALICE:

- Utilizza il messaggio come input alla funzione di hash (stabilita con Bob) ottenendo così un digest (d) di lunghezza fissata che dipende dalla funzione scelta
- Cifra il digest utilizzando la propria chiave privata
- Il digest cifrato (d_c) viene concatenato al messaggio: $R = M \| d_c$
- Alice invia quindi R

BOB:

- Utilizza il messaggio come input alla funzione di hash (stabilita con Alice) ottenendo in tal modo il digest (d')
- Decifra il digest d_c utilizzando la chiave pubblica di Alice ottenendo in tal modo d'
- Effettua il confronto tra d_c e d'. Se i digest sono identici allora avremo la certezza che il messaggio sia stato opera di Alice e che durante la trasmissione non sia stato alterato, altrimenti il messaggio è stato manipolato e deve essere scartato

Quindi, è facilmente intuibile come tali step garantiscano le proprietà di autenticità, integrità e non ripudiabilità.

Soffermiamoci un attimo sull'ultima proprietà: come facciamo ad essere sicuri che il messaggio provenga effettivamente da Alice?

Dal momento che Bob utilizza la chiave pubblica di Alice per decifrare il messaggio la chiave utilizzata per cifrare il documento deve necessariamente essere la chiave privata di Alice (la quale è l'unica -in un contesto safe- a poter avere accesso a tale chiave).
Però rimane un potenziale problema: tale meccanismo fa trasmettere il messaggio in chiaro (perdendo quindi la proprietà di confidenzialità). Quale meccanismo è necessario impiegare nel caso in cui il messaggio contenesse dati sensibili che non devono essere letti da soggetti non autorizzati?
E' possibile utilizzare un doppio livello di cifratura per ovviare a tale problema.
Tornando ora all'immagine precedente: quando Alice crea $R = M \| d_c$ può utilizzare la chiave pubblica di Bob per cifrare il "pacchetto". Di conseguenza Bob, non appena lo riceve, prima di procedere alle verifiche del digest, dovrà decifrare quanto ricevuto facendo uso della propria chiave privata. In tal modo abbiamo quindi aggiunto la confidenzialità al messaggio.
Applicazioni della firma digitale e alcune varianti (come l'HMAC) possono essere trovati nei JSON Web Token (JWT) nel quale il campo signature conterrà appunto la firma digitale del relativo header e payload.

Passiamo ora ad approfondire le tematiche JSON

3 Mondo JSON

3.1 Introduzione

JSON (JavaScript Object Notation) è un formato per lo scambio dati basato sul linguaggio di programmazione JavaScript. È utilizzato in programmazione web come alternativa al formato XML e preferito nel mondo mobile e sistemi IoT. Applicazioni del formato JSON possono essere individuate nelle web signature (JWS) ma anche nella generazione di web token (JWT) e nell'utilizzo dei campi cifrati (JWE). JSON può rappresentare quattro tipi primitivi (stringhe, numeri, booleani e null) e due tipi strutturati (oggetti e array).
Un oggetto è un insieme non ordinato di zero o più coppie nome/valore, dove un nome è una stringa e un valore è una stringa, un numero, un booleano, un null, un oggetto o un array. Un array è una sequenza ordinata di zero o più valori. Il testo JSON DEVE essere codificato in UTF-8, UTF-16 o UTF-32. La codifica predefinita è UTF-8.

```
{
    "Image": {
        "Width":  800,
        "Height": 600,
        "Title":  "View from 15th Floor",
        "Thumbnail": {
            "Url":    "http://www.example.com/image/481989943",
            "Height": 125,
            "Width":  100
        },
        "Animated" : false,
        "IDs": [116, 943, 234, 38793]
    }
}
```

Figure 3: Esempio di oggetto JSON

```
  [
    {
       "precision": "zip",
       "Latitude":  37.7668,
       "Longitude": -122.3959,
       "Address":   "",
       "City":      "SAN FRANCISCO",
       "State":     "CA",
       "Zip":       "94107",
       "Country":   "US"
    },
    {
       "precision": "zip",
       "Latitude":  37.371991,
       "Longitude": -122.026020,
       "Address":   "",
       "City":      "SUNNYVALE",
       "State":     "CA",
       "Zip":       "94085",
       "Country":   "US"
    }
  ]
```

Figure 4: Esempio di array JSON contenente due oggetti:

Passiamo ora ad analizzare il JWS:

3.2 JSON Web Signature (JWS)

JWS è stato sviluppato per firmare contenuti arbitrari utilizzando una rappresentazione compatta basata su JSON. La rappresentazione di un oggetto firmato JWS si basa su tre parti: header, payload e signature. In JWS c'è una serializzazione compatta: si cerca di rendere la rappresentazione il più breve possibile. Le parti sono codificate in base64url e concatenate, separate dal carattere "." come segue:

$$header\_b64u"."payload\_b64u"."signature\_b64u \quad (1)$$

- JOSE Header: oggetto JSON contenente i parametri che descrivono le operazioni e i parametri crittografici impiegati. L'intestazione JOSE (JSON Object Signing and Encryption) è composta da un insieme di parametri di intestazione;
- Payload JWS: sequenza di ottetti da proteggere, ovvero il messaggio. Il payload può contenere una sequenza arbitraria di ottetti;
- Firma JWS: firma digitale o MAC sull'intestazione protetta JWS e sul payload JWS.

In questo modo tutto è URL-safe (la codifica base64 non è URL-safe mentre lo è base64url) e compatto; ciò è stato reso possibile usando una sola signature base64 posta alla fine.

NB: La definizione di base della codifica Base64, contenuta nell'RFC-4648, consiste nel codificare 6 bit alla volta utilizzando un carattere ASCII a 8 bit. "URL-safe" è un termine che si riferisce alla codifica di un insieme di caratteri che può essere utilizzato in modo sicuro all'interno di un URL senza causare problemi o errori. La codifica base64url è definita come "URL-safe" perché tutti i suoi caratteri possono essere utilizzati in un URL senza causare problemi. Al contrario, la codifica base64 standard non è "URL-safe" perché include i caratteri '+' e '/', che hanno significati speciali in un URL. Quindi, quando si parla di "URL-safe" nel contesto della serializzazione JWS, significa che l'oggetto firmato può essere trasmesso in modo sicuro come parte di un URL. La codifica base64url sostituisce '+' con '-', '/' con '', e rimuove il padding '=', rendendo la stringa sicura per l'uso in un URL.

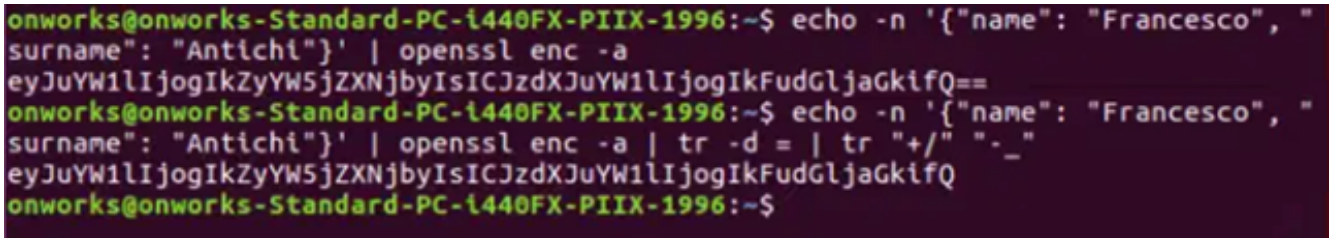


Figure 5: Esempio realizzato su una macchina Ubuntu online che evidenzia le differenze tra base64 e base64url

La firma JWS copre sia l'intestazione che il payload. Infatti, se la firma coprisse solo il contenuto, l'intestazione potrebbe essere modificata e, ad esempio, un aggressore potrebbe cambiare l'algoritmo di firma o il tipo di contenuto, sollevando problemi nella verifica della firma o nell'interpretazione del contenuto. L'input di firma è la concatenazione dell'intestazione codificata e del payload separati da un punto. Ciò consente di firmare direttamente la rappresentazione in uscita (ovvero ciò che verrà effettivamente inviato attraverso la rete).

Passiamo ora ad analizzare alcuni parametri descritti nella RFC 7515 (dedicata a JWS)

- Il parametro di intestazione "alg" (algoritmo) identifica l'algoritmo crittografico utilizzato per proteggere il JWS. Il valore della firma JWS non è valido se il valore "alg" non rappresenta un algoritmo supportato o se non esiste una chiave da utilizzare con tale algoritmo associata alla parte che ha firmato digitalmente il contenuto;
- Il parametro dell'intestazione "jku" (JWK Set URL) è uno dei parametri utilizzati nelle firme digitali JSON Web Signature (JWS) e nelle cifrature JSON Web Encryption (JWE) come parte

delle specifiche del framework JSON Web Token (JWT). Il parametro "jku" fornisce un riferimento a un URL da cui recuperare una serie di chiavi pubbliche in formato JSON Web Key (JWK). Queste chiavi pubbliche possono essere utilizzate per verificare la firma digitale o per decifrare il contenuto cifrato di un token JWT.

Quindi, quando un token JWT contiene il parametro "jku", il client può utilizzare l'URL specificato per recuperare le chiavi pubbliche necessarie per verificare la firma digitale o per decifrare il contenuto del token. Questo consente una gestione flessibile delle chiavi pubbliche, poiché le chiavi possono essere aggiornate o modificate senza dover modificare direttamente il token JWT.

- Il parametro header "jwk" (JSON Web Key) è la chiave pubblica che corrisponde alla chiave utilizzata per firmare digitalmente il JWS. Questa chiave è rappresentata come una chiave web JSON;
- Il parametro "kid" fornisce un identificatore univoco per una chiave utilizzata per firmare o cifrare il token JWT. Questo identificatore può essere utilizzato per associare una chiave specifica a un token, consentendo al destinatario di identificare quale chiave debba essere utilizzata per verificare la firma digitale o per decifrare il contenuto del token.

Quando un token JWT contiene il parametro "kid", il destinatario può utilizzare questo identificatore per cercare la corrispondente chiave pubblica o privata nella propria cache delle chiavi o in un repository di chiavi remoto, in base all'implementazione specifica.

- Il parametro di intestazione "x5u" (URL X.509) è un URI [RFC3986] che rimanda a una risorsa per il certificato a chiave pubblica X.509 o la catena di certificati [RFC5280] corrispondente alla chiave utilizzata per firmare digitalmente il JWS;
- Il parametro dell'intestazione "typ" (Type) è un parametro utilizzato nei token JWT (JSON Web Token) per specificare il tipo di token JWT che è stato creato. Questo parametro è facoltativo ma può essere utile per applicazioni che necessitano di informazioni aggiuntive sul tipo di token che stanno trattando. Il valore del parametro "typ" è una stringa che identifica il tipo di token. Comunemente, i valori utilizzati includono:
  - JWT: Indica che il token è un JSON Web Token;
  - JWS: Indica che il token è una firma digitale JSON Web Signature;
  - JWE: Indica che il token è una cifratura JSON Web Encryption.

Quando un token JWT contiene il parametro "crit", il suo valore è una lista di nomi di estensioni o caratteristiche del token che il destinatario deve trattare come critiche. Ciò significa che, se il destinatario del token non fosse in grado di comprendere o gestire correttamente una delle estensioni elencate come critiche, deve rifiutare il token e non proseguire con l'elaborazione. Supponiamo, ad esempio, che un token JWT contenga una firma digitale utilizzando un algoritmo specifico e richieda che questo algoritmo sia trattato come critico per la corretta verifica della firma. In tal caso, il parametro "crit" verrebbe utilizzato per indicare che l'algoritmo di firma è critico e che il destinatario del token deve essere in grado di comprendere e supportare quell'algoritmo per elaborare correttamente il token.
Di seguito un esempio di un JWS con alcuni campi descritti in precedenza:

Figure 6: Esempio JWS

3.3 Creazione di un JWS

Per creare un JWS, si possono eseguire i seguenti passaggi:

1. Creare il contenuto da utilizzare come payload JWS.
2. Calcolare il valore codificato del payload $BASE64URL(JWSPayload)$.
3. Creare l'oggetto o gli oggetti JSON contenenti l'insieme dei parametri di intestazione desiderati, che insieme costituiscono l'intestazione JOSE (l'intestazione JWS protetto e/o l'intestazione JWS non protetto).
4. Calcolare il valore codificato dell'intestazione $BASE64URL(UTF8(ProtectedHeaderJWS))$.
   Se l'intestazione protetta JWS non è presente (cosa che può accadere solo quando si utilizza la serializzazione JWS JSON e non è presente alcun membro "protected"), questo valore sarà una stringa vuota

5. Calcolo della firma JWS
   $ASCII(BASE64URL(UTF8(ProtectedHeaderJWS))||'.'$
   $||BASE64URL(JWSPayload))$

6. Calcolare il valore codificato della firma $BASE64URL(JWSSignature)$.

7. Se si utilizza la serializzazione JWS JSON, ripetere questo processo (passi 3-6) per ogni firma digitale o MAC.

8. Creare l'output serializzato desiderato. La serializzazione compatta JWS di questo risultato è:
   $BASE64URL(UTF8(JWSProtectedHeader))||'.'||$
   $BASE64URL(JWSPayload)||'.'||BASE64URL(FirmaJWS)$.

Altra peculiarità di JWS sono le sue due modalità:

1. **JWS Compact Serialization:** questa modalità è una rappresentazione compatta delle firme digitali, ottimizzata per ambienti con restrizioni di spazio, come gli header HTTP o i parametri delle query URI. In questa modalità, i componenti della firma (intestazione protetta, payload e firma) sono codificati in base64url e separati da un singolo punto;

2. **JWS JSON Serialization:** questa modalità rappresenta le firme digitali come oggetti JSON, consentendo maggiore flessibilità e possibilità di includere informazioni aggiuntive. Questa modalità supporta anche la possibilità di applicare più firme e/o codici di autenticazione dei messaggi (MAC) allo stesso contenuto.

L'importante è che le applicazioni che utilizzano le firme digitali JWS specificano quale modalità di serializzazione e quali caratteristiche di serializzazione vengono utilizzate per quella specifica applicazione. Ad esempio, un'applicazione potrebbe specificare che utilizza solo la JWS JSON Serialization, oppure che supporta solo una firma o un MAC singolo, oppure che supporta firme multiple.

3.4 Detached Content

Il concetto di "Detached Content" si riferisce a un metodo per proteggere l'integrità di contenuti che non sono inclusi direttamente in un JSON Web Signature (JWS). Questo può essere utile in contesti in cui si desidera proteggere il contenuto, ma non è pratico o necessario includerlo direttamente nel token JWS stesso.
Ecco come funziona il processo di "Detached Content":

1. Creazione del JWS: inizialmente viene creato un JWS utilizzando una rappresentazione del contenuto come payload del token JWS;

2. Rimozione del payload dal JWS: dopo aver creato il JWS, la rappresentazione del payload viene rimossa dal token JWS. Questo può essere fatto sostituendo il valore del campo che contiene il payload codificato in base64url con una stringa vuota nella serializzazione compatta, o eliminando completamente il campo "payload" nell'oggetto JSON nella serializzazione JSON;

3. Invio del JWS modificato: il JWS modificato, privato del payload, viene inviato al destinatario;

4. Ricostruzione del JWS da parte del destinatario: il destinatario può ricostruire il JWS originale reinserendo la rappresentazione del payload nel JWS modificato. Questo può essere fatto manualmente dal destinatario;

5. Utilizzo del JWS ricostruito: una volta ricostruito il JWS completo, il destinatario può utilizzarlo normalmente per verificare l'integrità del contenuto e per accedere al payload.

Questo metodo di "Detached Content" presuppone che il destinatario sia in grado di ricostruire esattamente il payload utilizzato nel JWS originale. Questo può essere appropriato in scenari in cui il contenuto è disponibile separatamente e il destinatario ha il mezzo per accedervi.
Il destinatario può ricostruire il payload utilizzando un meccanismo che consente di accedere al contenuto originale. Ecco alcuni modi in cui il destinatario potrebbe fare ciò:

1. **Referenza al contenuto:** il mittente e il destinatario possono concordare su un modo per accedere al contenuto originale, ad esempio tramite un URL. Il JWS modificato conterrà quindi un riferimento al contenuto anziché il contenuto stesso. Il destinatario può quindi recuperare il contenuto utilizzando questa referenza;

2. **Condivisione diretta del contenuto:** se mittente e destinatario hanno un canale di comunicazione diretta e sicuro, il mittente può inviare il contenuto direttamente al destinatario insieme al JWS modificato. In questo caso, il destinatario ha accesso diretto al contenuto senza bisogno di ulteriori operazioni;

3. **Accesso tramite archiviazione condivisa:** mittente e destinatario possono condividere il contenuto utilizzando un servizio di archiviazione condivisa, come un servizio cloud. Il mittente può caricare il contenuto su questo servizio e condividere l'accesso al destinatario. Il JWS modificato conterrà quindi un riferimento al contenuto memorizzato nell'archiviazione condivisa.

Pertanto, riassumendo gli step mediante i quali è possibile inviare e ricostruire un JWT senza payload:

Passaggio 1: Mittente crea e invia il JWS modificato

1. **Creazione del JWS:** il mittente crea un JWS come al solito, includendo un payload.

2. **Rimozione del payload:** dopo aver creato il JWS, il mittente rimuove il payload dal token JWS modificato.

3. **Invio del JWS modificato:** il mittente invia il JWS modificato al destinatario tramite un canale sicuro.

Passaggio 2: Destinatario ricostruisce il payload

1. **Accesso al contenuto originale:** il mittente e il destinatario possono aver concordato in precedenza un meccanismo per accedere al contenuto originale. Ad esempio, potrebbero avere un canale di comunicazione diretta o potrebbero utilizzare un servizio di archiviazione condivisa.

2. **Ricostruzione del payload:** il destinatario accede al contenuto originale utilizzando il meccanismo concordato.

Passaggio 3: Destinatario verifica il JWS

1. **Inserimento del payload:** il destinatario reinserisce il payload originale nel JWS modificato, ripristinandolo alla sua forma originale

2. **Verifica del JWS:** il destinatario utilizza il JWS ricostruito per verificare l'integrità del contenuto originale. Questo può includere la verifica della firma digitale e l'analisi dell'intestazione per confermare che il JWS sia stato creato correttamente.

3. **Conclusione:** se il JWS supera con successo la verifica, il destinatario può essere sicuro che il contenuto del file di testo non sia stato modificato e può utilizzarlo in base alle proprie esigenze.

Questo processo dimostra come il mittente possa proteggere il contenuto del file di testo utilizzando un JWS, senza includerlo direttamente nel token. Il destinatario, a sua volta, è in grado di ricostruire correttamente il JWS e verificare l'integrità del contenuto originale.

Per completezza andiamo ora a descrivere la Flattened JWS per poi procedere oltre:

3.5 Flattened JWS JSON Serialization

La Flattened JWS JSON Serialization è una delle due modalità di serializzazione utilizzate per rappresentare le firme digitali JSON Web Signature (JWS). La sua struttura è più compatta rispetto alla JWS JSON Serialization standard e non prevede l'utilizzo di un array di firme. Invece, ogni firma è rappresentata come un oggetto JSON separato.
Ecco una spiegazione più dettagliata della sintassi della Flattened JWS JSON Serialization:

1. **Protected Header (Intestazione Protetta):** l'intestazione protetta è un oggetto JSON che contiene le informazioni relative alla firma digitale, come l'algoritmo di firma utilizzato e altri parametri pertinenti. Questo oggetto è codificato in base64url e inserito nell'oggetto principale come il campo "protected".

2. **Unprotected Header (Intestazione Non Protetta):** l'intestazione non protetta è un oggetto JSON che contiene informazioni aggiuntive relative alla firma digitale, ma che non sono protette dalla firma stessa. Questo oggetto è incluso direttamente nell'oggetto principale come un campo separato.

3. **Payload:** il payload è la parte del messaggio che viene firmata. In una Flattened JWS JSON Serialization, il payload è rappresentato come un campo separato nell'oggetto principale.

Signature (Firma): la firma è il risultato dell'applicazione dell'algoritmo di firma al JWS Signing Input (cioè all'intestazione protetta codificata in base64url, un punto, e il payload codificato in base64url). La firma è rappresentata come un campo separato nell'oggetto principale.

Prima di andare a descrivere nel dettaglio i JSON Web Token è necessario mettere un altro mattoncino sul mondo JSON. Pertanto passiamo ora ad analizzare:

3.6 JSON Web Encryption (JWE)

JOSE supporta anche la crittografia. Il JWE permette la cifratura di contenuti arbitrari utilizzando la rappresentazione JSON.
Ci sono tre parti: header, chiave crittografata e testo cifrato.
Esistono due possibili rappresentazioni: serializzazione compatta nativa JWE e serializzazione JSON JWE.
Nota che anche la chiave è crittografata, quindi nell'header devono essere specificati due algoritmi, uno per la chiave e l'altro per il testo cifrato.

3.7 Parametri dell'header JWE

I parametri dell'header sono descritti nella seguente immagine:

Parametro	Descrizione
alg	Algoritmo di crittografia per la chiave
enc	Algoritmo di crittografia per il contenuto
jku	URL della chiave JSON Web
x5t	Impronta SHA-1 del certificato X.509
x5t#S256	Impronta SHA-256 del certificato X.509
x5c	Certificato X.509
x5u	URL del certificato X.509
kid	Identificatore chiave

Table 1: Descrizione dei parametri

3.8 Serializzazione compatta JWE

Nella serializzazione compatta JWE è possibile riscontrare alcune con JWS, infatti si ha:
$header b64u" ." encryptedCEK b64u" ." ivb64u" ." ciphertext b64u" ." authNtag b64u$
Prima c'è l'header codificato in base64url, poi la chiave crittografata, di seguito il valore di inizializzazione codificato in base64url (se è necessario un valore di inizializzazione, è il caso di CBC, CTR o altri algoritmi simmetrici come chacha20), il testo cifrato stesso e infine, se è stata utilizzata la crittografia autenticata, allora ci sarà un tag di autenticazione. Tutte le parti sono separate da un separatore "." Ecco alcune caratteristiche:

- è una rappresentazione compatta e sicura per URL;

- c'è solo un destinatario (uno dei limiti) il che significa che non è qualcosa che può essere decifrato da molti destinatari (in PKCS#7 c'era la possibilità sia per le firme di avere vari firmatari che per la crittografia di avere chiavi crittografate per diversi destinatari), perché JSON è pensato per essere scambiato tra un client e un server;
- non ci sono intestazioni non protette e non ci sono dati associati (quindi il tag è calcolato su tutto).

3.9 Serializzazione JSON JWE

In questo caso il contenuto cifrato è rappresentato come un oggetto JSON. Di seguito la descrizione dei parametri della Serializzazione JSON JWE:

Membro	Descrizione
protected	Contiene intestazioni protette dall'integrità (contiene elenco di intestazioni protette con autenticazione)
unprotected	Contiene le intestazioni che non devono essere calcolate nella parte di autenticazione
iv	Vettore di inizializzazione
aad	Dati autenticati aggiuntivi se si desidera avere altri dati calcolati nella parte di autenticazione, che non fanno parte delle intestazioni ma fanno parte del contenuto. In questo caso il JSON può avere il contenuto diviso in due parti: il vero contenuto crittografato e i dati aggiuntivi
ciphertext	Il vero contenuto cifrato (obbligatorio)
tag	Tag di autenticazione
recipients	In questo caso, con la serializzazione JSON, c'è la possibilità di avere più destinatari rappresentati come un array di oggetti JSON che identificano ciascun destinatario e forniscono la chiave di crittografia del contenuto, crittografata con la sua chiave pubblica. Per ciascuno di essi c'è l'intestazione alg, kid e la chiave crittografata

Table 2: Descrizione dei parametri

L'esempio seguente mostra l'intestazione protetta e quella non protetta (in cui è possibile identificare la chiave). Il campo "destinatari" mostra l'intestazione che contiene l'algoritmo e l'identificatore della chiave e poi la chiave cifrata con la chiave pubblica specificata nell'identificatore. Nel secondo destinatario è presente l'algoritmo "a128kw" che significa AES-128-KW (Key Wrap) in cui la chiave simmetrica viene cifrata con un'altra chiave simmetrica. In questo caso c'è un accordo precedente su una chiave identificata dal numero 7 e poi c'è la chiave criptata. Poi c'è un vettore di inizializzazione, il testo cifrato stesso e il tag di autenticazione. Si noti che l'algoritmo utilizzato non è specificato, poiché non è obbligatorio. Nell'esempio si assume che l'algoritmo di cifratura (dichiarato all'esterno) sia "A128CBC-HS256" (AES 128 CBC con HMAC SHA256 per il tag di autenticazione). Si noti che non è necessario usare la crittografia autenticata, perché si tratta di crittografia e autenticazione (l'algoritmo specificato nel campo "enc" all'esterno). Il formato, anche se denominato crittografia, in realtà supporta sia la crittografia che il MAC. Si potrebbe usare lo stesso formato specificando null per la crittografia e usando solo la parte di autenticazione (ma per questo è preferibile JWS). Questo è fatto per evitare di usare la crittografia e poi incapsulare di nuovo all'interno di una firma (se sono entrambe necessarie, lo standard è questo).

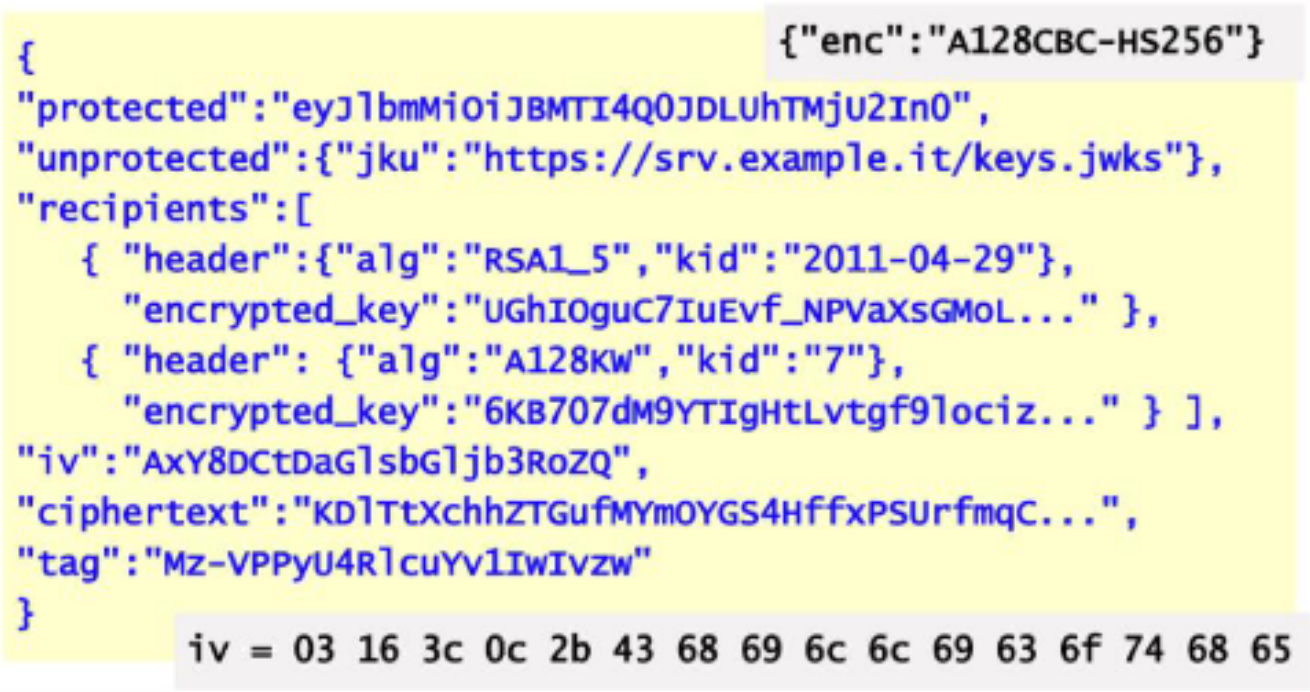


Figure 7: Esempio

NON È CONSIGLIATO riutilizzare l'intero set di chiavi (chiave di crittografia, chiave di crittografia dei contenuti, vettore di inizializzazione, ecc). Un suggerimento per evitare il riutilizzo è quello di generare sempre almeno una nuova chiave per ogni operazione di crittografia (ad esempio, una nuova chiave di crittografia del contenuto, un nuovo IV e/o un nuovo sale PBES2).
Analizzati per bene JWS e JWT soffermiamoci ora su:

3.10 JSON Web Token (JWT)

JSON web token (JWT) è uno standard aperto (RFC 7519 JSON Web Token (JWT)) che definisce un modo compatto e autonomo per trasmettere in modo sicuro informazioni tra le parti come oggetto JSON. Ecco alcuni vantaggi:

- **Grazie alle sue dimensioni relativamente ridotte**, un JWT può essere inviato tramite un URL, un parametro POST o un'intestazione HTTP e viene trasmesso rapidamente. Un JWT contiene tutte le informazioni necessarie su un'entità per evitare di interrogare un database più di una volta. L'uso dei JWT presenta dei vantaggi rispetto ai token web semplici (SWT) e ai token SAML;
- **Più compatto:** JSON è meno verboso di XML, quindi una volta codificato, un JWT è più piccolo di un token SAML. Questo fa sì che il JWT sia una buona scelta per essere passato in HTML.

Di seguito un esempio di token in JSON e con SAML

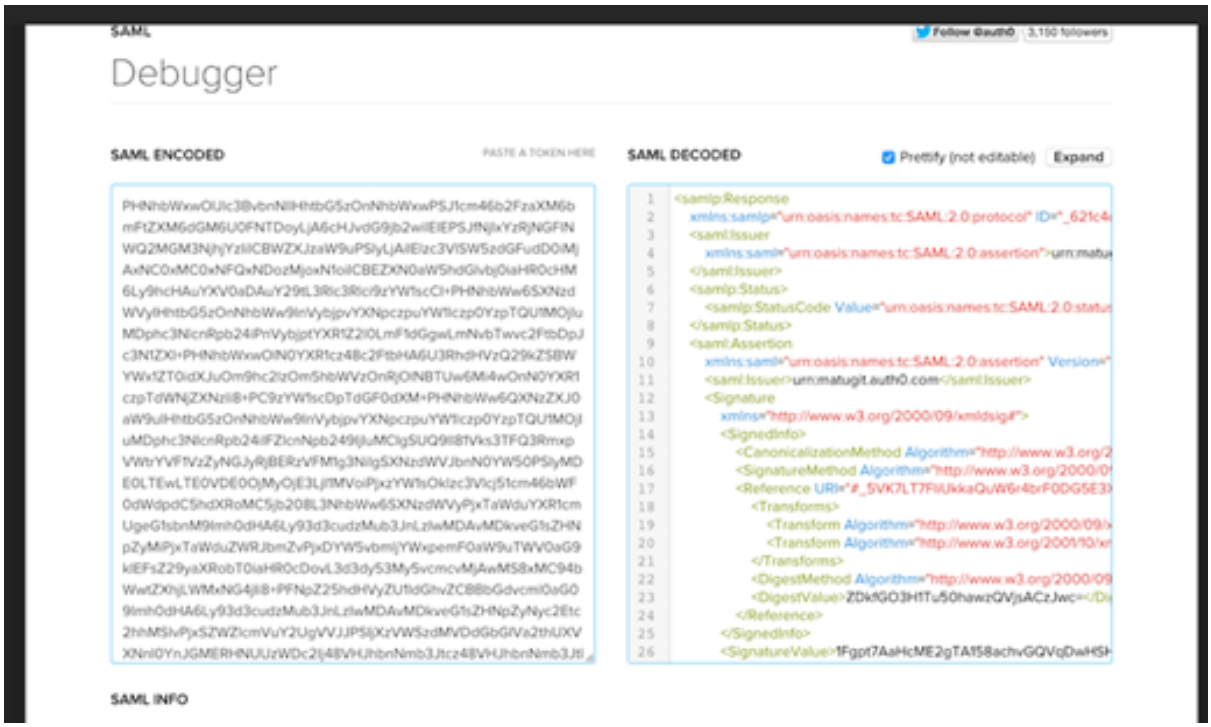


Figure 8: JWS vs SAML

- **Più sicuri:** i JWT possono utilizzare una coppia di chiavi pubbliche/private sotto forma di certificato X.509 per la firma. Un JWT può anche essere firmato simmetricamente da un segreto condiviso utilizzando l'algoritmo HMAC. Mentre i token SAML possono utilizzare coppie di chiavi pubbliche/private come i JWT, firmare XML con XML Digital Signature senza introdurre oscure falle nella sicurezza è molto difficile rispetto alla semplicità della firma di JSON.

Usi di JWT

- **Autorizzazione:** una volta che l'utente ha effettuato l'accesso, un'applicazione può richiedere di

accedere a percorsi, servizi o risorse (ad esempio, API) per conto di quell'utente. Per fare ciò, in ogni richiesta, deve passare un token di accesso, che può avere la forma di un JWT.
Il Single Sign-on (SSO) utilizza ampiamente il JWT a causa del ridotto overhead del formato e della sua capacità di essere facilmente utilizzato in diversi domini;

- **Scambio di informazioni:** i JWT sono un buon modo per trasmettere informazioni in modo sicuro tra le parti perché possono essere firmati, il che significa che si può essere certi che i mittenti siano chi dicono di essere. Inoltre, la struttura di un JWT consente di verificare che il contenuto non sia stato manomesso;

- **Autenticazione:** le informazioni contenute nell'oggetto JSON possono essere verificate ed essere considerate attendibili perché sono firmate digitalmente. Sebbene i JWT possano anche essere cifrati per garantire la segretezza tra le parti, i JWT emessi da Auth0 sono JSON Web Signatures (JWS), ovvero sono firmati piuttosto che cifrati. Per questo motivo, ci concentreremo sui token firmati, che possono verificare l'integrità delle affermazioni in essi contenute, mentre i token crittografati nascondono tali affermazioni ad altre parti.

In generale, i JWT possono essere firmati utilizzando una chiave segreta (con l'algoritmo HMAC) o una coppia di chiavi pubbliche/private con RSA o ECDSA. Quando i token sono firmati utilizzando coppie di chiavi pubbliche/private, la firma certifica anche che solo la parte in possesso della chiave privata è quella che ha firmato.
Prima di utilizzare un JWT ricevuto, è necessario convalidarlo correttamente utilizzando la sua firma. Si noti che un token convalidato correttamente significa solo che le informazioni contenute nel token non sono state modificate da nessun altro. Ciò non significa che altri non siano stati in grado di vedere il contenuto, che è memorizzato in testo normale. Per questo motivo, non si dovrebbero mai memorizzare informazioni sensibili all'interno di un JWT e si dovrebbero adottare altre misure per garantire che i JWT non vengano intercettati, ad esempio inviando i JWT solo tramite HTTPS.
Come visto in precedenza per JWS e JWE anche un JWT consiste in 3 campi codificati in base64:

- **JOSE Header:** contiene metadati sul tipo di token e sugli algoritmi crittografici utilizzati per proteggerne il contenuto;

- **Payload JWS (set of claims):** contiene dichiarazioni di sicurezza verificabili, come l'identità dell'utente e le autorizzazioni concesse;

- **Signature JWS:** utilizzata per convalidare che il token è affidabile e non è stato manomesso. Quando si utilizza un JWT, è necessario verificarne la firma prima di memorizzarlo e utilizzarlo.

3.11 JWT Claims

Il concetto di "JWT Claims" è legato alle informazioni contenute all'interno di un JSON Web Token (JWT) che rappresentano i dati condivisi o le affermazioni sull'entità (utente o oggetto) a cui il token si riferisce:

1. **JWT Claims Set:** questo rappresenta un oggetto JSON i cui membri sono le affermazioni trasmesse dal JWT. Ogni affermazione è un pezzo di informazione sul soggetto rappresentato dal token. Le affermazioni possono includere informazioni come l'identità dell'utente, le autorizzazioni, la scadenza del token e altro ancora.

2. **Unicità dei Claim Names:** i nomi delle affermazioni all'interno di un JWT devono essere univoci. Questo significa che non possono esserci due affermazioni con lo stesso nome all'interno dello stesso JWT. Se ciò accade, i parser JWT devono o rifiutare il token JWT con nomi di affermazioni duplicati o utilizzare un parser JSON che restituisca solo l'ultimo membro duplicato, come specificato nelle specifiche ECMAScript.

3. Classi di nomi delle affermazioni JWT: Esistono tre classi di nomi delle affermazioni JWT:

    - **Registered Claim Names:** questi sono nomi di affermazioni definiti dalle specifiche JWT. Alcuni esempi includono "iss" (emittente), "sub" (soggetto), "exp" (scadenza) e "aud" (destinatario);

- **Public Claim Names:** questi sono nomi di affermazioni definiti da chiunque sia libero di utilizzare, ma è buona pratica registrare i propri nomi di affermazioni in un registro globale per evitare collisioni di nomi;
- **Private Claim Names:** questi sono nomi di affermazioni definiti da un accordo tra le parti che scambiano il JWT. Sono destinati a essere utilizzati per informazioni specifiche dell'applicazione che non sono coperte dalle affermazioni registrate o pubbliche.

In sostanza, le affermazioni del JWT sono le informazioni principali contenute nel token che consentono alle applicazioni di autenticare e autorizzare gli utenti. Sono strutturate come un oggetto JSON e possono essere di varie tipologie, ognuna con un significato specifico nel contesto dell'applicazione che utilizza il token JWT.
Passiamo ora ad elencare alcuni campi del token tratti dalla RFC:

3.11.1 *"iss" (Issuer) Claim*

Il claim "iss" (issuer) identifica il committente che ha emesso il JWT. L'elaborazione di questo claim è generalmente specifica dell'applicazione. Il valore "iss" è una stringa case sensitive contenente un valore StringOrURI. L'uso di questo claim è OPZIONALE.

3.11.2 *"sub" (Subject) Claim*

Il claim "sub" (soggetto) identifica il principale che è oggetto del JWT. I claim di un JWT sono normalmente dichiarazioni sull'oggetto. Il valore subject DEVE avere uno scope che lo renda localmente unico nel contesto dell'emittente o che sia globalmente unico. L'elaborazione di questo claim è generalmente specifica dell'applicazione. Il valore "sub" è una stringa case sensitive contenente un valore StringOrURI. L'uso di questo claim è OPZIONALE.

3.11.3 *"aud" (Audience) Claim*

L'indicazione "aud" (pubblico) identifica i destinatari a cui è destinato il JWT. Ogni mandante destinato a elaborare il JWT DEVE identificarsi con un valore nel claim audience. Se il mandante che elabora la richiesta non si identifica con un valore nell'indicazione "aud" quando è presente, il JWT DEVE essere rifiutato. Nel caso generale, il valore "aud" è un array di stringhe case sensitive, ciascuna contenente un valore StringOrURI. Nel caso speciale in cui il JWT abbia un solo audience, il valore "aud" PUO' essere una singola stringa case-sensitive contenente un valore StringOrURI. L'interpretazione dei valori di audience è generalmente specifica dell'applicazione. L'uso di questa indicazione è OPZIONALE.

3.11.4 *"exp" (Expiration Time) Claim*

L'indicazione "exp" (expiration time) identifica il tempo di scadenza dopo il quale il JWT NON DEVE essere accettato per l'elaborazione. L'elaborazione dell'indicazione "exp" richiede che la data/ora corrente DEVE essere precedente alla data/ora di scadenza elencata nell'indicazione "exp". Gli implementatori POSSONO prevedere un piccolo margine, di solito non superiore a qualche minuto, per tenere conto delle oscillazioni dell'orologio. Il suo valore DEVE essere un numero contenente un valore NumericDate. L'uso di questa indicazione è OPZIONALE.

3.11.5 *"nbf" (Not Before) Claim*

L'indicazione "nbf" (not before) identifica il momento prima del quale il JWT NON DEVE essere accettato per l'elaborazione. L'elaborazione dell'indicazione "nbf" richiede che la data/ora corrente DEVE essere successiva o uguale alla data/ora "non prima" elencata nell'indicazione "nbf". Gli implementatori POSSONO prevedere un piccolo margine, di solito non superiore a qualche minuto, per tenere conto delle oscillazioni dell'orologio. Il suo valore DEVE essere un numero contenente un valore NumericDate. L'uso di questa indicazione è OPZIONALE.

3.11.6 *"iat" (Issued At) Claim*

L'indicazione "iat" (issued at) identifica l'ora in cui è stato emesso il JWT. Questo claim può essere utilizzato per determinare l'età del JWT. Il suo valore DEVE essere un numero contenente un valore NumericDate. L'uso di questo claim è OPZIONALE.

3.11.7 *"jti" (JWT ID) Claim*

Il claim "jti" (JWT ID) fornisce un identificatore univoco per il JWT. Il valore dell'identificatore DEVE essere assegnato in modo da garantire una probabilità trascurabile che lo stesso valore venga accidentalmente assegnato a un oggetto di dati diverso; se l'applicazione utilizza più emittenti, si DEVONO evitare collisioni anche tra valori prodotti da emittenti diversi. Il claim "jti" può essere utilizzato per impedire che il JWT venga riprodotto. Il valore "jti" è una stringa sensibile alle maiuscole e alle minuscole. L'uso di questo claim è OPZIONALE.
Passiamo ora ad analizzare gli step necessari per creare un JWT:

3.12 Creazione di un JWT

Per creare un JWT, si eseguono i seguenti passaggi (l'ordine non è significativo nei casi in cui non ci siano dipendenze tra gli input e gli output dei passaggi):

1. Creare un set di richieste JWT contenente le richieste desiderate. Si noti che gli spazi bianchi sono esplicitamente ammessi nella rappresentazione e non è necessaria la canonicalizzazione prima della codifica.

2. Il messaggio è costituito dagli ottetti della rappresentazione UTF-8 dell'insieme di rivendicazioni JWT Claims Set.

3. Creare un'intestazione JOSE contenente l'insieme desiderato di parametri di intestazione. Il JWT DEVE essere conforme alle specifiche [JWS] o [JWE]. Anche qui gli spazi bianchi sono esplicitamente consentiti nella rappresentazione.

4. Se verrà eseguita un'operazione di firma o crittografia annidata, lasciare che il messaggio sia il JWS o il JWE e tornare al punto 3, utilizzando un tipo di contenuto "cty di "JWT" nella nuova intestazione JOSE creata in quel passaggio.

5. Altrimenti, lasciare che il JWT risultante sia il JWS o il JWE.

Analizziamo ora la validazione di un JWT:

3.13 Validazione di un JWT

Quando si convalida un JWT, vengono eseguite le seguenti fasi. Se una qualsiasi delle fasi elencate fallisce, il JWT DEVE essere rifiutato, ossia trattato dall'applicazione come non valido:

1. Verificare che il JWT contenga almeno un punto ('.');

2. L'intestazione JOSE codificata è la porzione del JWT prima del primo punto ('.');

3. Base64url decodifica l'intestazione JOSE codificata seguendo la restrizione di non utilizzare interruzioni di riga, spazi bianchi o altri caratteri aggiuntivi;

4. Verificare che la sequenza di ottetti risultante sia una rappresentazione codificata UTF-8 di un oggetto JSON completamente valido conforme a RFC 7159 (The JavaScript Object Notation (JSON) Data Interchange Format);

5. Verificare che l'intestazione JOSE risultante includa solo parametri e valori la cui sintassi e semantica siano comprese e supportate o che siano specificate come ignorati quando non vengono compresi;

6. Determinare se il JWT è un JWS o un JWE. A seconda che il JWT sia un JWS o un JWE, ci sono due casi:

   - Se il JWT è un JWS, seguire la [JWS]. Far sì che il Message sia il risultato della codifica base64url del JWS Payload.
   - Altrimenti, se il JWT è un JWE, seguire[JWE]. Far sì che il Message sia il risultante plaintext

7. Se il JOSE Header contiene un valore JWT "cty" (content type) allora il Message è un JWT che era il subject di operazioni di firma o cifratura annidate. In questo caso tornare a 1 e usare il Message come JWT;

8. Altrimenti decodificare in base64url il messaggio senza usare spazi aggiuntivi e caratteri aggiuntivi;

9. Verificare che la sequenza di ottetti risultante sia una rappresentazione codificata UTF-8 di un oggetto JSON completamente valido conforme a RFC 7159; che il set di richieste JWT sia questo oggetto JSON.

3.14 Considerazioni di privacy

Un JWT può contenere informazioni sensibili alla privacy. In questo caso, si DEVONO adottare misure per impedire la divulgazione di queste informazioni a persone non interessate. Un modo per raggiungere questo obiettivo è utilizzare un JWT crittografato e autenticare il destinatario. Un altro modo è quello di garantire che i JWT contenenti informazioni sensibili alla privacy non cifrati siano trasmessi solo utilizzando protocolli che utilizzino la crittografia e che supportino l'autenticazione dell'endpoint, come Transport Layer Security (TLS). L'omissione di informazioni sensibili alla privacy da un JWT è il modo più semplice per ridurre al minimo i problemi di privacy.

Dopo esser entrati profondamente nel dettaglio nelle applicazioni del formato JSON è possibile ora avere più chiara la seguente tabella con valori customizzati utili come esempio

Sample JOSE Header

Figure 9: Esempio JOSE Header

Claim	Description
alg	The algorithm that will be used for signing the JWS. The list of valid algorithms is here: https://tools.ietf.org/html/rfc7518#section-3.1 This value **must** be "PS256".
typ	This is an optional claim. If it is specified, it **must** be set to the value "JOSE".
cty	This is an optional claim. If it is specified, it **must** be set to the value "json" or "application/json".
kid	This is a mandatory claim. It **must** match a value that can be used to look up the key in a key store hosted by the Trust Anchor.
iat	This **must** be a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in GMT until the date/time. This is a private header parameter name. (See RFC 7515 - Private Header Parameter Names)
iss	This **must** be a string that identifies the Participant. For example, as CoP uses the Open Banking Directory, the value **must** be: • When issued by the CoP Requester, of the form {{org-id}}/{{software-statement-id}}, • When issued by the CoP Responder of the form {{org-id}} Where: • org-id is the organization id • software-statement-id is the statement issued by open-banking
tan	This **must** be a string that consists of a domain name that is registered to and identifies the Trust Anchor that hosts the public counterpart of the key used for signing. For example, as CoP uses the Open Banking Directory, the value **must** be openbanking.org.uk
crit	This **must** be a string array consisting of the values iss tan iat This indicates that the JWS signature validator **must** understand and process the three additional claims.

Prima di entrare nel dettaglio nel mondo Open Banking è necessario spendere alcune parole nel definire meglio la direttiva PSD2, che cosa ha comportato la sua entrata in vigore e quale potrebbe essere la sua evoluzione. Solamente dopo si sarà in grado di comprendere a pieno che cosa sia l'Open Banking

4 La direttiva PSD2

La Direttiva sui servizi di pagamento 2 (Payment Service Directive 2, PSD2) è una direttiva europea sui servizi di pagamento, entrata in vigore nel 2018. Il suo obiettivo principale è quello di:

- **Aumentare la concorrenza nel settore dei servizi di pagamento**
- **Migliorare la sicurezza delle transazioni online**
- **Favorire l'innovazione tecnologica**

La PSD2 è un'evoluzione significativa della regolamentazione esistente per il settore dei pagamenti. Essa mira ad aumentare la concorrenza (obbligando le banche ad aprire le proprie API -Application Programming Interface- a terze parti autorizzate) in un settore dei pagamenti già competitivo, a includere nell'ambito di applicazione nuovi tipi di servizi di pagamento, a migliorare la protezione e la sicurezza dei consumatori e ad ampliare la portata della direttiva. La PSD2 è un passo importante verso un mercato unico digitale in Europa, che mira a rendere il mercato dell'UE adatto all'era digitale.

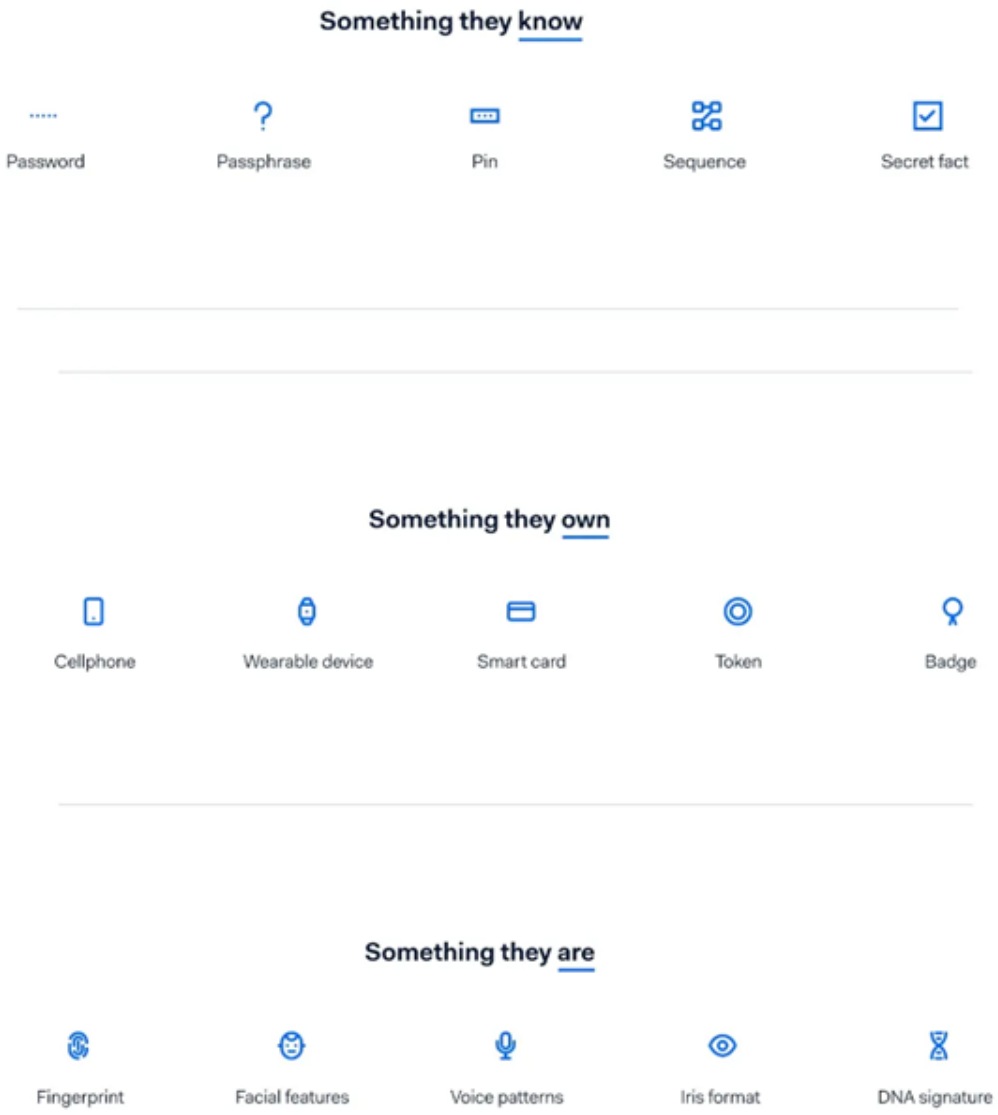
4.1 Strong Customer Authentication

Una grande novità introdotta dalla Direttiva PSD2 è la SCA, Strong Customer Authentication, ideata per identificare e autenticare in modo univoco l'utente quando vuole pagare online con una carta di credito. Gli obiettivi sono diversi, tra i quali:

- proteggere gli utenti da potenziali reati online o furti di dati
- fornire uno strumento più sicuro per i consumatori che sono riluttanti ad acquistare online

Questa autenticazione forte (SCA), in sostanza, consiste nella verifica di almeno due dei seguenti elementi necessari per accertare l'identità di un utente:

- **conoscenza:** qualcosa che solo l'utente conosce (something you know), ad esempio, PIN, password, etc.
- **inerenza:** qualcosa che è riconducibile solo a quell'utente (something you are), ad esempio, riconoscimento vocale, un'impronta digitale, etc.
- **possesso:** qualcosa che solo l'utente possiede (something you have), ad esempio, cellulare, token, etc.



Pertanto, se in precedenza era sufficiente il numero di una carta di credito per poter effettuare un pagamento online, con l'entrata in funzione della PSD2 sarà necessario utilizzare qualcosa in aggunta, ad esempio, anche il PIN o il riconoscimento biometrico.

4.2 Quando non si applica la SCA

L'autenticazione forte (SCA) può non essere applicata alle seguenti tipologie di pagamenti:

- **ricorrenti:** significa pagamenti dello stesso importo e da parte dello stesso beneficiario (ad esempio, un abbonamento ad un servizio online). In questo caso l'autenticazione forte avviene solo la prima volta;
- **di basso valore** inferiori a € 30 o se ripetute in un arco di 24 ore non devono superare complessivamente € 100 (es. se nello stesso giorno faccio 2 pagamenti di € 50 ciascuno);
- **verso beneficiari attendibili** che vengono identificati come soggetti affidabili;
- **a basso rischio** fino ad un valore massimo di € 500 (sono operazioni considerate dai prestatori di servizi di pagamento sotto certi parametri di rischio)

4.3 Il futuro della PSD2: PSD3

La nuova Direttiva PSD3, o Payment Services Directive 3, approvata nella seduta del Parlamento Europeo del 23 Aprile 2024, è l'ultima evoluzione delle normative europee sui servizi di pagamento online in Europa. Questa direttiva, adottata dall'Unione Europea, promette di migliorare la sicurezza, la trasparenza e l'efficienza del settore, coinvolgendo anche i player di natura non bancaria.
Una delle caratteristiche più interessanti della Direttiva PSD3 è la sua logica di armonizzazione a livello europeo. Questo significa che le regole si applicheranno uniformemente in tutta l'Unione Europea, contribuendo a prevenire disparità e distorsioni concorrenziali sul mercato. Ciò rappresenta un importante passo avanti nell'obiettivo di creare un ambiente equo per i fornitori di servizi di pagamento.
Inoltre, PSD3 introduce significative modifiche rispetto alla sua versione precedente, orientate a rafforzare i processi di autenticazione e l'adozione di servizi digitali. Un aspetto cruciale è l'introduzione dell'autenticazione forte nel caso di inserimento di uno strumento di pagamento all'interno di un digital wallet, come Apple Pay o Google Wallet. Questa misura mira a offrire maggiori garanzie ai titolari di carte e strumenti, eliminando le vulnerabilità dei processi di enrollment deboli.
La tutela dei consumatori è un principio fondamentale su cui la Direttiva PSD3 si concentra. Vieta la pratica del surcharging (sovrapprezzo) nei pagamenti internazionali, garantendo che i consumatori non siano soggetti a costi aggiuntivi non giustificati, inoltre si ampliano i requisiti di trasparenza in termini di costi, tassi di cambio e canoni di utilizzo, persino per quanto riguarda i servizi offerti tramite ATM

Per comprendere appieno l'impatto della direttiva PSD3, è necessario esaminare le principali differenze rispetto alla PSD2:

- **Pagamenti transfrontalieri:** PSD3 cerca di semplificare ulteriormente i pagamenti transfrontalieri, eliminando alcune delle complessità che possono sorgere quando si effettuano transazioni tra paesi dell'Unione Europea. Ciò potrebbe comportare una maggiore convenienza per le imprese e i consumatori che operano a livello internazionale;

- **Prevenzione delle frodi:** la nuova direttiva pone una maggiore enfasi sulla prevenzione delle frodi, imponendo norme più rigorose per le procedure di autenticazione e controllo delle transazioni. Questo dovrebbe contribuire a ridurre il rischio di transazioni fraudolente e migliorare la sicurezza delle operazioni finanziarie online;

- **Protezione dei dati personali:** PSD3 introduce misure più rigide per la protezione dei dati personali dei clienti. Questo è particolarmente importante alla luce delle preoccupazioni crescenti sulla privacy e la sicurezza dei dati. Le aziende dovranno essere più attente nel trattamento delle informazioni dei loro clienti;

- **Regolamentazione delle criptovalute:** mentre PSD2 aveva solo accennato alle criptovalute e alle tecnologie finanziarie emergenti, PSD3 si occupa in modo più specifico di regolamentarle. Questo riflette il crescente interesse e la rapida evoluzione del settore delle criptovalute, che richiede una regolamentazione più dettagliata per garantire la stabilità e la sicurezza finanziaria.

Terminata la breve descrizione, utile principalmente nel voler fornire un'idea delle sue potenzialità e implicazioni nel mondo economico, della direttiva PSD2 si passa ora a trattare il mondo Open Banking.

5 Che cos'è Open Banking

L'Open Banking è un modello di servizi finanziari che consente agli sviluppatori terzi di accedere ai dati finanziari dei sistemi bancari tradizionali attraverso API (Application Programming Interface).
(**NB:** si ricorda che mentre la PSD2 è una direttiva europea -ovvero un atto legislativo che stabilisce un obiettivo che tutti i paesi dell'UE devono realizzare lasciando agli Stati membri la scelta della forma e dei mezzi per raggiungere tale obiettivo-, Open Banking è un modello di business che ha potuto vedere la luce grazie alla direttiva PSD2).

Questo modello cambia completamente le modalità di accesso e condivisione per i dati finanziari. L'Open Banking può dare ai consumatori un maggiore controllo sulle loro informazioni finanziarie e rendere disponibili nuovi servizi e applicazioni. Per le aziende che non operano nei comparti finanziari, questo cambiamento significa avere la possibilità di offrire servizi finanziari personalizzati ai loro clienti, prendere decisioni maggiormente basate sui dati e adottare soluzioni innovative per i pagamenti e la gestione dei conti. Un accesso più efficace ai dati finanziari significa anche poter semplificare i processi di pagamento e generare nuovi flussi di reddito.

5.1 Un po' di storia

L'Open Banking nasce nel Regno Unito ed è stato effettivamente avviato sotto la guida della Competition and Markets Authority (CMA), che nel 2016 ha creato l'Open Banking Implementation Entity (OBIE). Questa iniziativa è stata progettata per aumentare la concorrenza e l'innovazione nel settore bancario, obbligando le nove maggiori banche del Regno Unito (RBS Group, Lloyds Banking Group, Barclays, HSBC Group, Nationwide, Santander, Danske Bank, Bank of Ireland e Allied Irish Banks Group) a condividere i dati dei clienti con terze parti autorizzate, previa autorizzazione del cliente.
Nel Regno Unito gli standard e le specifiche dell'open banking sono sviluppati da Open Banking Limited, un'organizzazione senza scopo di lucro creata nel 2016 dalla Competition and Markets Authority (CMA).

5.2 Contributo di EBA, OBIE e CMA alla diffusione dell'Open Banking

European Banking Authority (EBA): l'EBA ha avuto un ruolo cruciale nella definizione degli standard tecnici e regolamentari per l'implementazione della PSD2. Ha sviluppato le Regulatory Technical Standards (RTS) per la sicurezza e l'autenticazione forte del cliente (SCA), garantendo che le banche e i fornitori di servizi di pagamento rispettino requisiti di sicurezza elevati.
Open Banking Implementation Entity (OBIE): l'OBIE è stata istituita dalla Competition and Markets Authority (CMA) del Regno Unito per implementare l'Open Banking. Ha sviluppato gli standard tecnici e le API che permettono alle terze parti di accedere ai dati bancari in modo sicuro e standardizzato. L'OBIE ha anche lavorato per garantire che le banche aderiscano agli standard di Open Banking, promuovendo l'adozione e l'innovazione nel settore.

Competition and Markets Authority (CMA) la CMA ha ordinato alle nove maggiori banche del Regno Unito di implementare l'Open Banking come parte di un pacchetto di misure per migliorare la concorrenza nel settore bancario. La CMA ha supervisionato l'implementazione e ha garantito che le banche rispettassero gli standard stabiliti dall'OBIE.

Di seguito viene riportato un breve paragrafo riassuntivo per comprendere al meglio gli obiettivi delle varie parti coinvolte:

5.3 Analogie e Differenze tra EBA, OBIE e CMA

Analogie:

- **Obiettivo Comune:** tutti e tre gli enti mirano a migliorare la sicurezza, la trasparenza e la concorrenza nel settore dei servizi di pagamento
- **Standard Tecnici:** sia l'EBA che l'OBIE hanno sviluppato standard tecnici per garantire la sicurezza e l'interoperabilità dei servizi di pagamento.
- **Promozione dell'innovazione:** tutti e tre gli enti promuovono l'innovazione nel settore finanziario, facilitando l'accesso ai dati e migliorando i servizi per i consumatori.

Differenze:

- **Ambito di Applicazione:** l'EBA opera a livello europeo, mentre l'OBIE e la CMA operano principalmente nel Regno Unito.
- **Ruolo Regolamentare:** l'EBA è un'autorità di regolamentazione che sviluppa standard e linee guida, mentre l'OBIE è un'entità di implementazione che sviluppa standard tecnici specifici per l'Open Banking. La CMA è un'autorità di concorrenza che supervisiona l'implementazione delle misure per migliorare la concorrenza.

5.4 Come funziona l'Open Banking?

L'Open Banking consente l'interoperabilità dei servizi finanziari tramite l'uso di API. Queste API facilitano lo scambio sicuro di informazioni finanziarie tra banche e fornitori terzi autorizzati. A differenza dei servizi bancari tradizionali, che spesso operano in un ambiente chiuso, l'Open Banking decentralizza i servizi finanziari. Nel settore bancario tradizionale, i dati sono spesso isolati all'interno dei singoli istituti, rendendo difficile le interazioni dirette tra le applicazioni esterne e i conti finanziari. L'Open Banking rivoluziona questa situazione imponendo formati di dati standardizzati e protocolli di comunicazione sicuri. In questo modo si creano condizioni di parità in cui i servizi terzi possono integrarsi con più banche nel contesto di un insieme comune di norme, regolamenti e standard tecnici. Le API Open Banking sono solitamente classificate in tre tipi principali:

- **API per i dati:** forniscono accesso in sola lettura alle informazioni, ai saldi e alla cronologia delle transazioni del conto;
- **API per le transazioni:** queste API consentono di trasferire fondi, impostare addebiti diretti e avviare pagamenti;
- **API per i prodotti:** consentono a terzi di elencare prodotti finanziari, tassi e condizioni e vengono spesso usate per siti web o marketplace di confronto

Abbattendo le barriere dei dati e consentendo l'interoperabilità tra le piattaforme, l'Open Banking può accelerare l'innovazione nel settore dei servizi finanziari, fornendo alle attività informazioni più granulari sui dati e una gamma più ampia di opzioni per i servizi finanziari.

5.5 Esempi di servizi di Open Banking

L'Open Banking non è un prodotto o un servizio distinto, ma un quadro di riferimento all'interno del quale possono essere abilitati diversi servizi finanziari. Si tratta di un settore in continua evoluzione, per cui è probabile che la gamma dei servizi finanziari introdotti nel mercato sia destinata ad ampliarsi. Ecco alcuni degli attuali utilizzi dell'Open Banking:

- **Servizi di avvio del pagamento**: i venditori al dettaglio possono avviare i pagamenti direttamente dal conto bancario del cliente, senza dover ricorrere a un gateway di pagamento tradizionale. Questo metodo potrebbe portare a regolamenti dei pagamenti più rapidi e a commissioni ridotte per le transazioni;
- **Aggregazione di conti**: i consulenti finanziari e le società di gestione patrimoniale possono raccogliere i dati da più conti, ottenendo una visione più completa della situazione finanziaria di un cliente e riuscendo così a offrire consulenze più accurate e personalizzate;
- **Budget automatizzati:** le attività possono offrire ai dipendenti un sistema di gestione delle spese intelligente, in grado di categorizzare e tracciare automaticamente le spese da più conti bancari, con conseguente semplificazione della reportistica e della supervisione finanziarie;
- **Prestiti istantanei e punteggio di credito:** gli istituti finanziari possono accedere a dati in tempo reale per valutazioni più accurate del credito, accelerando i processi di approvazione dei prestiti;
- **Riconciliazione automatica delle fatture:** le attività possono utilizzare le API Open Banking per automatizzare il processo di corrispondenza tra fatture e transazioni, riducendo il lavoro amministrativo e migliorando l'accuratezza;

- **Piattaforme multibanca:** una società che opera in più mercati potrebbe consolidare i conti di diverse banche in un'unica dashboard, facilitando il monitoraggio delle operazioni globali;
- **Marketing personalizzato:** i venditori al dettaglio possono analizzare i dati delle transazioni per offrire promozioni mirate o premi fedeltà direttamente legati alle consuetudini di spesa di una persona;
- **Rilevamento delle frodi in tempo reale:** grazie all'analisi immediata dei dati delle transazioni, si possono rilevare attività insolite più rapidamente che mai, riducendo il rischio di perdite finanziarie.

Sebbene questi esempi siano solo un assaggio, danno un'idea di come l'Open Banking possa cambiare il settore dei servizi finanziari.

5.6 Utenti dell'Open Banking

L'Open Banking ridefinisce completamente il modo in cui le attività e i clienti accedono ai dati finanziari e li usano. Ecco alcuni dei gruppi più interessati da questa evoluzione:

- **Singoli clienti:** le persone utilizzano l'Open Banking per accedere a un'ampia gamma di servizi finanziari tramite applicazioni di terzi. Possono rivedere i modelli di spesa, ricevere consigli finanziari altamente personalizzati o automatizzare operazioni come il pagamento delle bollette. L'API può anche esaminare i dati delle transazioni dei consumatori per identificare i migliori prodotti e servizi finanziari per loro, come un nuovo conto di risparmio che guadagnerebbe un tasso di interesse più elevato rispetto al conto di risparmio corrente o una carta di credito diversa con un tasso di interesse più basso;
- **Istituti finanziari:** le banche tradizionali, le cooperative di credito e altri fornitori di servizi finanziari utilizzano l'Open Banking per modernizzare le loro offerte e creare esperienze migliori per i clienti. Possono anche collaborare con piccole aziende tecnologiche per introdurre servizi innovativi sul mercato;
- **Aziende Fintech:** le aziende più recenti, orientate alla tecnologia, utilizzano le funzionalità di condivisione sicura dei dati dell'Open Banking per creare servizi specializzati, che vanno da app per la definizione dei budget a soluzioni complesse di gestione finanziaria per le attività;
- **Piccole e medie imprese:** queste organizzazioni scelgono l'Open Banking per automatizzare varie attività, come la riconciliazione delle fatture con le transazioni bancarie e per avere un quadro più chiaro del loro stato finanziario;
- **Enti normativi:** le organizzazioni che stabiliscono e applicano le regole finanziarie trovano l'Open Banking utile per creare un ambiente standardizzato. Ciò contribuisce a proteggere i clienti e a garantire pratiche di gestione dei dati sicure nel settore dei servizi finanziari;
- **Aziende di e-commerce:** le attività che vendono prodotti o servizi online possono elaborare le transazioni in modo più diretto, spesso evitando i sistemi di pagamento tradizionali e riducendo i costi;
- **Piattaforme contabili:** il software finanziario può accedere ai dati delle transazioni in tempo reale, facilitando la gestione dei conti e riducendo la necessità di inserire manualmente i dati;
- **Sviluppatori di software:** con le API Open Banking gli sviluppatori di software possono creare una serie di servizi e strumenti utili sia per i singoli clienti che per le attività, aprendo nuove strade per l'innovazione;
- **Istituti di credito:** queste entità possono prendere decisioni più rapide e accurate accedendo rapidamente ai dati finanziari, ottimizzando le procedure di erogazione dei prestiti e di credit scoring.

5.7 Vantaggi dell'Open Banking

L'Open Banking offre alle attività svariate opportunità per migliorare l'operatività, rispettare più facilmente le normative e offrire servizi a valore aggiunto in grado di posizionarle in modo ottimale per il futuro.
Ecco i vantaggi per le attività:

- **Processo decisionale basato sui dati:** l'Open Banking consente alle attività di ottenere dati finanziari dettagliati che possono informare le scelte strategiche, dalla valutazione dei rischi alla pianificazione degli investimenti. Il dettaglio delle informazioni disponibili supera quello dei rendiconti finanziari tradizionali;
- **Agilità operativa:** l'Open Banking garantisce un flusso di dati accelerato, velocizzando le transazioni e consentendo una riconciliazione più rapida. Questa rapidità si traduce in una maggiore adattabilità in condizioni di mercato in rapida evoluzione;
- **Collaborazione interoperabile:** l'Open Banking crea un ambiente in cui gli istituti finanziari e le aziende tecnologiche possono innovare insieme. Questo si traduce in un'offerta di servizi più ampia ed elaborata per i clienti;
- **Processi di pagamento ottimizzati:** le API Open Banking supportano modalità di pagamento più dirette, spesso aggirando i gateway tradizionali e riducendo i costi delle transazioni;
- **Allineamento alle normative:** l'Open Banking incorpora spesso protocolli standardizzati e misure efficaci di protezione dei dati, aiutando le attività a soddisfare più facilmente i requisiti normativi;
- **Personalizzazione per i clienti:** le attività possono offrire servizi finanziari personalizzati ai propri clienti, da soluzioni di prestito specializzate a servizi di tesoreria, il tutto abilitato dall'enorme quantità di dati accessibili tramite l'Open Banking;
- **Allocazione di risorse:** la semplificazione delle operazioni finanziarie consente al personale di concentrarsi su altre aree del business. Che si tratti di automatizzare la riconciliazione o di snellire i processi di fatturazione, le API Open Banking possono rendere più efficiente l'impiego del personale;
- **Penetrazione nel mercato:** l'Open Banking può aiutare le attività a entrare in nuovi mercati, grazie a partnership con aziende fintech locali e a un più facile accesso ai dati dei clienti per una personalizzazione localizzata;
- **Catalizzatore di innovazione:** per le attività del settore tecnologico e finanziario, l'Open Banking rappresenta un modo dinamico per offrire nuovi servizi che possono essere monetizzati, aumentando i flussi di ricavi e la fidelizzazione dei clienti.

5.8 Open Banking Directory

L'Open Banking Directory è la componente architettonica chiave che consente ai Third Party Provider (TPP) di iscriversi e fornire loro le funzionalità necessarie per usufruire del servizio di Open Banking e di avviare interazioni tramite API con gli Account Servicing Payment Service Provider (ASPSP). La Directory è un servizio di gestione dell'identità e dell'accesso che fornisce informazioni sull'identità di persone fisiche, organizzazioni e classi di identità software. Gli ASPSP, a loro volta, potranno utilizzare le informazioni tecniche contenute nella Directory per convalidare l'identità dell'organizzazione e del software (attivo/inattivo), la validità del QTSP e lo stato del registro NCA dello Stato membro. Le capacità funzionali della Directory possono essere suddivise a grandi linee in tre gruppi:

1. **Gestire identità e accessi:** in cui si ha la capacità di emettere e gestire i record di identità per le organizzazioni e le persone fisiche che interagiscono con l'Open Banking Directory;
2. **Gestire certificati e chiavi:** in cui si ha la possibilità di caricare, gestire e rimuovere i certificati eIDAS (QWAC e QSeals), le chiavi di firma e le chiavi di crittografia. Inoltre è possibile emettere, gestire e revocare i certificati digitali OB;
3. **Gestire le informazioni della directory:** in cui si ha la capacità di aggiornare e trovare le informazioni mantenute nella Directory - attraverso API e/o un'interfaccia utente self-service (UI) fornita come applicazione web.

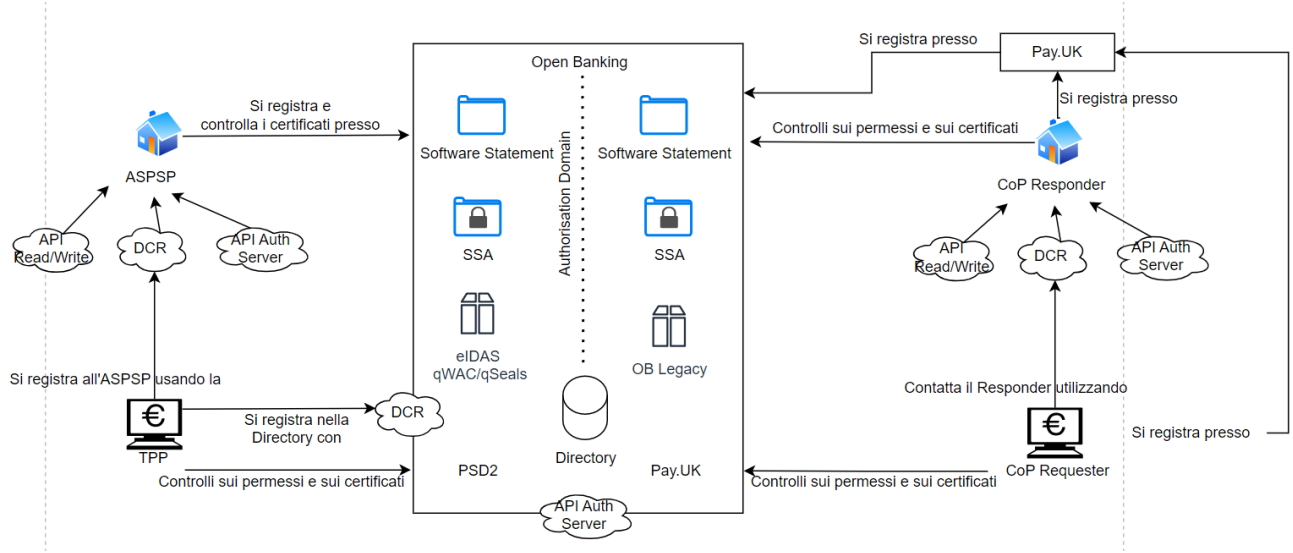


Figure 12: Esempio directory

Andando maggiormente nel dettaglio in merito ai partecipanti dell'Open Banking Directory: la Developer zone Open Banking (piattaforma dedicata agli sviluppatori e che contiene la documentazione tecnica -con le varie API e guide-, sandbox di testing e accesso alla produzione per lanciare il servizio sviluppato) afferma anche che: *"I partecipanti devono essere autorizzati o registrati presso la propria autorità competente. Il registro FCA sarà la fonte principale di partecipanti che possono essere aggiunti all'Open Banking Directory con autorizzazioni PSD2 nel Regno Unito. Anche i partecipanti che hanno un passaporto attraverso il registro della loro autorità nazionale competente per operare nel Regno Unito possono essere aggiunti all'Open Banking Directory"*, ovvero i partecipanti devono essere autorizzati o registrati presso la loro autorità competente. Il registro FCA sarà la fonte principale dei partecipanti che possono essere aggiunti all'Open Banking Directory con le autorizzazioni PSD2 nel Regno Unito. I partecipanti che hanno un passaporto attraverso il loro registro dell'autorità nazionale competente per operare nel Regno Unito possono anche essere aggiunti all'Open Banking Directory.
In sintesi, sia gli ASPSP che le TPP devono essere registrati e autorizzati dalle loro rispettive Autorità Nazionali Competenti, e questa registrazione è un prerequisito per l'aggiunta alla Directory Open Banking.
Ritornando alla struttura della Directory: può essere vista come un insieme di JSON Web Key Stores (JWKS) -collezioni di chiavi crittografiche utilizzate per la firma e la crittografia dei dati-; fornisce una serie di key store, ognuna dei quali contiene certificati e chiavi per una specifica organizzazione o dichiarazione software:
Key Store per ogni Organizzazione: questi contengono i certificati attivi e inattivi, le chiavi di firma e di crittografia per ogni organizzazione. Un certificato o una chiave "attiva" è quella attualmente in uso, mentre un certificato o una chiave "inattiva" è quella che è stata utilizzata in precedenza ma non è più in uso.
Key Store per ogni Dichiarazione Software: questi contengono i certificati attivi e inattivi, le chiavi di firma e di crittografia per ogni dichiarazione software. Una dichiarazione software è un documento che descrive le caratteristiche e le capacità del software di un Fornitore di Servizi di Terze Parti.
Inoltre, la Directory pubblica anche tutti i certificati che emette come file PEM. L'accesso ai certificati presenti nella Directory Open Banking può variare a seconda delle specifiche implementazioni e delle politiche di sicurezza. In generale, i certificati e le chiavi sono accessibili tramite l'API o l'interfaccia front-end della Directory, che sono solitamente riservate ai partecipanti registrati nell'Open Banking, come i TPP e gli ASPSP.
Ad esempio, un TPP o un ASPSP potrebbero essere in grado di accedere ai propri certificati e chiavi effettuando il login nell'interfaccia front-end della Directory navigando verso la sezione appropriata. Allo stesso modo, potrebbero essere in grado di recuperare i certificati e le chiavi tramite l'API della Directory, inviando una richiesta API appropriata.
Per motivi di sicurezza, l'accesso ai certificati e alle chiavi è solitamente strettamente controllato e limitato solo alle parti autorizzate. Pertanto, se si è un consumatore o un utente finale, potrebbe non essere possibile visualizzare direttamente i certificati nella Directory.

In seguito, verranno brevemente accennate due tecnologie ampiamente utilizzate nel mondo Open Banking per lo scambio di informazioni e per garantire l'accesso ai dati, andando nel dettaglio:

- **JWT (JSON Web Tokens)**: è un formato di token compatto e sicuro per lo scambio di informazioni tra due parti. Nell'Open Banking, i JWT vengono utilizzati per autenticare le TPP presso gli ASPSP inviando un token JWT, firmato mediante la chiave ricevuta durante la registrazione tra il TPP e la Directory, assieme a un certificato digitale utilizzato durante l'handshake TLS tra le parti
- **OAuth, o Open Authorization**: è uno standard aperto che permette a un utente di autorizzare l'accesso ai propri dati da parte di un'applicazione di terze parti, senza condividere la password del proprio account

In precedenza si accennava all'uso di TLS per poter garantire la trasmissione delle informazioni tramite un canale sicuro: la specifica Financial API Read Write individua gli algoritmi da utilizzare per TLS e per le firme digitali.
Per TLS devono essere supportati solo i seguenti cifrari:
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_DHE_RSA_WITH__256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

A scanso di equivoci, ciò si applica solo alla connessione TLS tra ASPSP e TPP mentre per le firme digitali (e la verifica) dei JWT:

- devono essere usati algoritmi PS256 o ES256
- non devono essere usati algoritmi che usano RSASSA-PKCS1-v1_5 (e.g. RS256)

5.9 OBIE (Open Banking Implementation Entity)

L'OBIE ha il compito di creare specifiche per le API (Application Programming Interfaces) che le banche e altri fornitori di servizi di pagamento devono implementare. Queste API permettono alle terze parti autorizzate di accedere in modo sicuro ai dati dei clienti, a condizione che i clienti stessi abbiano dato il loro consenso.
Open Banking è nato sotto questo principio, è una decisione (diritto) della persona decidere se condividere o no i propri dati finanziari tra banche, istituti di credito, TPP.
Ecco alcuni compiti dell'OBIE:

1. **Definizione degli standard API:** l'OBIE definisce gli standard tecnici e funzionali per le API che le banche devono implementare. Questo include dettagli su come le API dovrebbero funzionare e quali dati dovrebbero essere in grado di condividere;
2. **Gestione del framework di sicurezza:** l'OBIE stabilisce le linee guida sulla sicurezza che le banche e le terze parti devono seguire quando utilizzano le API. Questo include la definizione di protocolli di autenticazione e autorizzazione;
3. **Monitoraggio della conformità:** l'OBIE monitora se le banche e le altre organizzazioni stanno rispettando gli standard dell'Open Banking e le relative normative. Può anche intervenire in caso di problemi o dispute.

Descritta nel dettaglio la directory si passa ora a descrivere gli attori del contesto Open Banking: ASPSP e TPP.

5.10 ASPSP (Account Servicing Payment Service Providers)

Gli ASPSP (le banche, per semplicità di terminologia) per poter sfruttare la rivoluzione dell'Open Banking devono poter essere iscritte alla Directory e mettere a disposizione le proprie API (realizzate seguendo le indicazioni della OBIE); in tal modo possono utilizzare le informazioni tecniche contenute nell'elenco della Directory per validare l'identità dell'organizzazione e del software (attivo/inattivo), la validità del QTSP (Qualified Trust Service Provider) e lo stato del registro NCA (National Competent Authority) del paese membro (incluso le informazioni sul passaporto). Questo significa che le banche possono utilizzare

l'elenco presente nella Directory per verificare che i TPP siano chi affermino di essere e che abbiano il diritto di accedere ai dati che stanno cercando di utilizzare. Questo aiuta a mantenere la sicurezza e la privacy dei dati bancari dei clienti. Come verrà spiegato più avanti nel documento l'iscrizione alla Directory e l'esposizione delle API permetterà ai TPP di poter comunicare correttamente e in modo sicuro (considerando l'uso di TLS come protocollo di trasporto) permettendo di poter fornire le funzionalità promesse nel rispetto delle normative privacy vigenti (ad esempio il CPPA canadese e il GDPR europeo).

5.11 Third Party Provider (TPP)

Il Third Party Provider risulta essere l'attore emergente (e in un certo senso anche il protagonista) della rivoluzione Open Banking. Come accennato nel paragrafo introduttivo un TPP è una qualunque azienda iscritta nell'Open Banking Directory. I TPP, in base alle funzionalità offerte, sono soggetti alla seguente classificazione (anche se spesso forniscono entrambe le funzionalità):

- **PISP:** fornitore di servizi di pagamento che inizializza un pagamento per conto del cliente da un conto detenuto presso un'altra istituzione di pagamento. Se, per esempio, si stesse utilizzando un'applicazione di gestione delle finanze per effettuare un pagamento, l'applicazione stessa potrebbe agire come PISP inizializzando il pagamento dal conto bancario.
- **AISP:** fornitore di servizi che ha accesso alle informazioni del conto del cliente. Questo permette all'AISP di aggregare tutte le informazioni dei conti del cliente in un unico posto, fornendo una visione completa delle finanze del cliente

I TPP con certificati eIDAS validi potranno registrarsi dinamicamente nella Directory utilizzando l'API della Directory stessa. I certificati eIDAS sono un tipo di certificato digitale utilizzato nell'Unione Europea per garantire transazioni sicure e autenticate. Nell'ambito Open Banking consideriamo due tipi di certificati eIDAS: QWACs e QSeals:

- **QWACs:** questi certificati sono utilizzati per autenticare i siti web e garantire una connessione sicura tra il browser dell'utente e il sito web. In pratica, quando un TPP si connette a un ASPSP un certificato QWAC può essere utilizzato per stabilire una connessione sicura.
- **QSeals:** questi certificati sono utilizzati per sigillare elettronicamente i dati, garantendo l'integrità e l'autenticità dei dati. Ad esempio, quando un TPP invia una richiesta a un ASPSP, un certificato QSeal può essere utilizzato per "sigillare" la richiesta, garantendo che i dati non siano stati alterati durante il trasporto.

Questi certificati possono essere caricati e associati a dichiarazioni software tramite l'API della Directory o l'interfaccia front-end della Directory. Una dichiarazione software è un documento che descrive le caratteristiche e le capacità del software di un TPP. Associando un certificato eIDAS a una dichiarazione software, il TPP può dimostrare che il suo software è sicuro e autentico.

Passiamo ora a descrivere alcuni step necessari per ottenere le certificazioni e le dichiarazioni necessarie per potersi iscrivere nella Directory Open Banking.

5.12 Come può un Fornitore di Servizi di Terze Parti (TPP) ottenere i certificati QWAC e QSeal?

Una serie di passaggi che un TPP potrebbe seguire per poter ottenere i certificati QWAC e QSeal sono i seguenti:

- **Identificare un QTSP:** il primo passo per un TPP è identificare un QTSP che fornisca sia il certificato QWAC che quello QSeal. L'elenco dei QTSP è indicato direttamente da OBE, Open Banking Europe;
- **Richiedere i Certificati:** il TPP deve poi richiedere i certificati QWAC e QSeal dal QTSP scelto. Questo processo può variare a seconda del QTSP, ma in genere richiede che il TPP fornisca informazioni sulla propria organizzazione e sul software che intende utilizzare per l'Open Banking;
- **Installazione dei Certificati:** una volta ottenuti i certificati, il TPP deve installarli nel suo sistema. Questo garantirà che tutte le comunicazioni tra il TPP e gli ASPSP siano sicure e autenticate;

- **Utilizzo dei Certificati:** infine, quando il TPP comunica con un ASPSP, deve presentare il suo certificato QWAC o QSeal. Questo permette all'ASPSP di verificare l'identità del TPP e di garantire che la comunicazione sia sicura.

Di seguito un esempio utile a comprendere bene gli step elencati in precedenza.
Supponiamo che un TPP, chiamato "FinTechX", voglia partecipare all'Open Banking. FinTechX identifica un QTSP chiamato "TrustCert" che fornisce certificati QWAC e QSeal. FinTechX contatta TrustCert e richiede i certificati, fornendo tutte le informazioni necessarie su FinTechX e sul suo software. TrustCert verifica le informazioni e rilascia i certificati a FinTechX. FinTechX installa quindi i certificati nel suo sistema. Ora, ogni volta che FinTechX comunica con una banca nell'ambito dell'Open Banking, presenta il suo certificato QWAC o QSeal per autenticare la comunicazione.

5.13 Software Statement

Il Software Statement è un componente chiave dell'Open Banking che facilita la comunicazione sicura tra le diverse entità dell'ecosistema, come le banche, le società fintech e gli sviluppatori di terze parti. Si tratta essenzialmente di un documento firmato digitalmente che fornisce informazioni su un'applicazione o un servizio software, sulle sue capacità e sulle autorizzazioni necessarie per accedere alle API bancarie. Ecco come è formata e strutturata una Dichiarazione del software:

1. **Informazioni sull'emittente:** questa sezione contiene informazioni sull'organizzazione che rilascia la dichiarazione del software. Include il nome dell'organizzazione, le informazioni di contatto e qualsiasi identificativo pertinente;

2. **Informazioni sull'applicazione software:** questa parte descrive l'applicazione o il servizio software per il quale viene rilasciata la dichiarazione del software. Include dettagli quali il nome dell'applicazione, la versione, la descrizione ed eventuali URL pertinenti;

3. **Informazioni sulla sicurezza:** questa sezione descrive le misure di sicurezza implementate dall'applicazione software, come i protocolli di crittografia, i meccanismi di autenticazione e le misure di protezione dei dati;

4. **Permessi API:** uno degli aspetti cruciali di una Dichiarazione del software è rappresentato dalle autorizzazioni concesse all'applicazione software per l'accesso alle API bancarie. Questa sezione specifica l'ambito di accesso consentito, compresi i tipi di dati e di transazioni che l'applicazione può eseguire;

5. **Firma digitale:** infine, la Dichiarazione del software viene firmata digitalmente dall'emittente utilizzando tecniche crittografiche per garantirne l'autenticità e l'integrità. La firma può essere verificata dai destinatari per confermare che la dichiarazione del software non è stata manomessa e proviene da una fonte affidabile.

La struttura e il formato di una dichiarazione di software possono variare a seconda degli standard e dei protocolli specifici adottati dall'ecosistema Open Banking in una particolare regione o giurisdizione. Ad esempio, nel Regno Unito, l'Open Banking aderisce agli standard definiti dall'Open Banking Implementation Entity (OBIE), mentre altre regioni possono avere le proprie specifiche.

5.14 Come può un Fornitore di Servizi di Terze Parti creare e usare un Software Statement?

Prima di procedere a elencare gli step che potrebbero essere impiegati per creare una dichiarazione software (denominata anche Software Statement Assertion -SSA-) ricordiamo quanto segue: quando un TPP vuole comunicare per la prima volta con una banca (durante la fase di Dynamic Client Registration, descritta in seguito) presenta la sua dichiarazione software. L'ASPSP può quindi verificare l'autenticità e la sicurezza del TPP controllando la dichiarazione software nella directory (in particolar modo fa un check delle firme digitali e controlla che le proprietà di integrity e authentication siano state rispettate, da notare come durante la comunicazione tra le due parti dovrà essere instaurato anche un canale TLS che garantirà tutte le proprietà di sicurezza necessarie ad evitare episodi di sniffing e manipolazioni) e, se tutti i controlli dovessero procedere a buon fine, può decide di accettare il TPP nell'elenco di aziende le quali, previo esplicito consenso dell'end-user, potranno accedere ai dati inerenti al conto corrente del cliente.

Un elenco di passaggi che il Fornitore di Servizi di Terze Parti (TPP) potrebbe seguire per poter creare un Software Statement e caricarlo nell'Open Banking Directory è il seguente:

- **Creazione del Software Statement:** il TPP deve prima creare un Software Statement che descriva le caratteristiche e le capacità del suo software. Questo potrebbe includere informazioni come l'identità del TPP, le funzionalità del software, i certificati di sicurezza e altre informazioni rilevanti;
- **Generazione dei Certificati:** il TPP deve generare oppure ottenere i certificati necessari, come i certificati QWAC e QSeal. Questi certificati sono essenziali per garantire la sicurezza e l'autenticità delle comunicazioni nell'Open Banking;
- **Associazione dei Certificati al Software Statement:** una volta generati i certificati, il TPP deve associarli al Software Statement. Questo può essere fatto tramite l'API della Directory o l'interfaccia front-end della Directory;
- **Caricamento del Software Statement nell'Open Banking Directory:** infine, il TPP deve caricare il Software Statement, completo dei certificati associati, nell'Open Banking Directory. Questo può essere fatto tramite l'API della Directory o l'interfaccia front-end della Directory.

Una volta caricato il Software Statement, il TPP (sottoforma di JWT) riceve la dichiarazione software firmata digitalmente, e solo allora potrà iniziare a comunicare con gli ASPSP nell'ambito dell'Open Banking. Pertanto quando il TPP invierà una richiesta a un ASPSP (che include il suo Software Statement) quest'ultimo potrà quindi verificare l'autenticità e integrità del documento consultando l'Open Banking Directory.

5.15 Dynamic Client Registration (DCR)

La DCR, o Dynamic Client Registration, è un processo che permette a un TPP di registrarsi dinamicamente con un Fornitore di Servizi di Pagamento che Gestisce un Conto (ASPSP), per esempio con una banca. In pratica, la DCR è un protocollo che permette a un TPP di inviare una richiesta di registrazione a un ASPSP e, in caso di successo ed esplicito consenso dell'utente a fornire l'accesso a determinati dati, di garantire i servizi indicati nelle condizioni d'uso. Questa richiesta di registrazione include dettagli sul TPP e sul software che intende utilizzare per fornire servizi finanziari.
Durante la DCR il TPP (una volta essersi registrato nella Directory, ovvero di aver creato in precedenza un'asserzione di dichiarazione software, la quale è un JSON Web Token (JWT) contenente metadati sul software client del TPP, e averla ricevuta firmata mediante la JSON Web Signature (JWS)) invia una richiesta al server OAuth della banca con i metadati desiderati, come il nome dell'applicazione, il logo, l'URL di redirect, ecc (ovvero il SSA). Il server OAuth della banca, effettuati i controlli in termini di validità delle firme, integrità delle informazioni, controlli di validità sulle chain dei certificati presenti nella richiesta, ecc... risponde con un identificativo client e un segreto client con i quali l'applicazione potrà "autenticarsi" nei confronti della banca (in tal modo non serve effettuare una DCR per ogni connessione ma basta ripresentare tale segreto e la banca saprà ritrovare in un suo database le informazioni relative a permessi e operazioni che il TPP può svolgere). Nel momento in cui l'applicazione sarà registrata, potrà iniziare a fare richieste alle API della banca per accedere ai dati del conto degli utenti. Queste richieste devono essere autenticate, il che è solitamente svolto utilizzando un JWT. Il JWT è firmato digitalmente utilizzando il segreto client che l'applicazione ha ricevuto durante la registrazione. La banca può quindi verificare la firma sul JWT per autenticare la richiesta.
Ecco un esempio pratico di come potrebbe funzionare:
Supponiamo che ci sia un TPP, chiamato "FinTechX", che fornisce un servizio di gestione finanziaria. FinTechX vuole accedere ai dati bancari dei suoi clienti per fornire questo servizio; quindi, deve registrarsi mediante la DCR con le banche dei suoi clienti. Per fare questo, FinTechX invia una richiesta di registrazione a ogni banca. Questa richiesta di registrazione include dettagli su FinTechX e sul suo software, come il nome dell'organizzazione, l'URL del suo sito web, i certificati di sicurezza e altre informazioni rilevanti firmate digitalmente dalla Directory. Ogni banca riceve la richiesta di registrazione di FinTechX, verifica le firme, i certificati e tutte le altre informazioni contenute nella richiesta. Se tutto è in ordine, la banca registra FinTechX nel suo sistema.
Nell'ambito dell'Open Banking, una volta completata la DCR il processo di autorizzazione OAuth 2.0 può iniziare.
Ecco come funziona:

- **Richiesta di autorizzazione:** il fornitore di servizi di terze parti (TPP) invia una richiesta di autorizzazione all'endpoint di autorizzazione dell'Account Servicing Payment Service Provider (ASPSP). Questa richiesta include l'identificativo del client e il segreto del client che il TPP ha ricevuto durante la DCR;
- **Autenticazione dell'utente:** l'utente viene reindirizzato alla pagina di login dell'ASPSP per autenticarsi;
- **Consenso dell'utente:** dopo l'autenticazione, l'utente visualizza le informazioni che il TPP ha richiesto di accedere e fornisce il consenso per l'accesso;
- **Rilascio del token di autorizzazione:** se l'utente fornisce il consenso, l'ASPSP rilascia un token di autorizzazione al TPP;
- **Scambio del token di autorizzazione:** il TPP scambia il token di autorizzazione per un token di accesso presso l'endpoint del token dell'ASPSP;
- **Accesso alle risorse:** con il token di accesso, il TPP può ora accedere alle risorse protette (ad esempio, i dati del conto dell'utente) presso l'ASPSP.

Ora la trattazione segue la generazione di un software statement. Prima di procedere, però, si riporta di seguito la differenza tra Software Statement e Software Statement Assertion per chiarire del tutto le differenze ed evitare confusione tra i termini:

- **Software Statement:** una dichiarazione software (o "Software Statement") nell'Open Banking è una dichiarazione che descrive le caratteristiche e le capacità del software di un Fornitore di Servizi di Terze Parti. Questa dichiarazione è un elemento chiave per garantire la sicurezza e l'autenticità delle comunicazioni nell'Open Banking. Nell'Open Banking, un TPP crea una dichiarazione software che include informazioni come l'identità del TPP, le funzionalità del software, i certificati di sicurezza e altre informazioni rilevanti. Questa dichiarazione viene poi registrata in una directory centralizzata (la Open Banking Directory);
- **Software Statement Assertion (SSA):** è un token JWT (JSON Web Token) contenente metadati relativi a un Software Statement. Risulta essere firmato digitalmente dalla Directory Open Banking in modo tale che l'ASPSP che riceverà un SSA durante la Dynamic Client Registration (DCR) sarà in grado di valutare l'autenticità del token applicando la chiave pubblica della Directory.

6 Manage Directory Information

6.1 Generare Software Statement Assertion

Un TPP può generare e scaricare un software statement assertion attraverso l'interfaccia Open Banking Directory User, in alternativa un SSA può essere acceduto tramite SSA API end-point.
Recupero di una Signed Software Statement (asserzione di dichiarazione software): consente ai client autenticati di recuperare una Software Statement Assertion tramite il software_id. Per accedere a questo endpoint, il richiedente deve disporre di un token di accesso. Un'asserzione sulla dichiarazione del software può essere generata solo se la dichiarazione del software e il TPP che la possiede sono attivi. Se uno dei due è contrassegnato come inattivo, verrà restituito un errore.

6.2 Creare Software Statement

Affinché i TPP e gli ASPSP possano comunicare tra di loro devono creare delle Dichiarazioni software che identifichino in modo univoco la loro applicazione. Questa sezione illustra i passaggi per la creazione delle Dichiarazioni software.
Inizio: Fare clic sul pulsante "Dichiarazioni software". In alternativa, selezionare la scheda Software Statement nella parte superiore dello schermo se i dettagli dell'organizzazione sono già visualizzati sullo schermo.
Step per creare Software Statement
Step 1: Selezionare il tab Software Statement.
Step 2: Cliccare il pulsante Add new Software Statement.

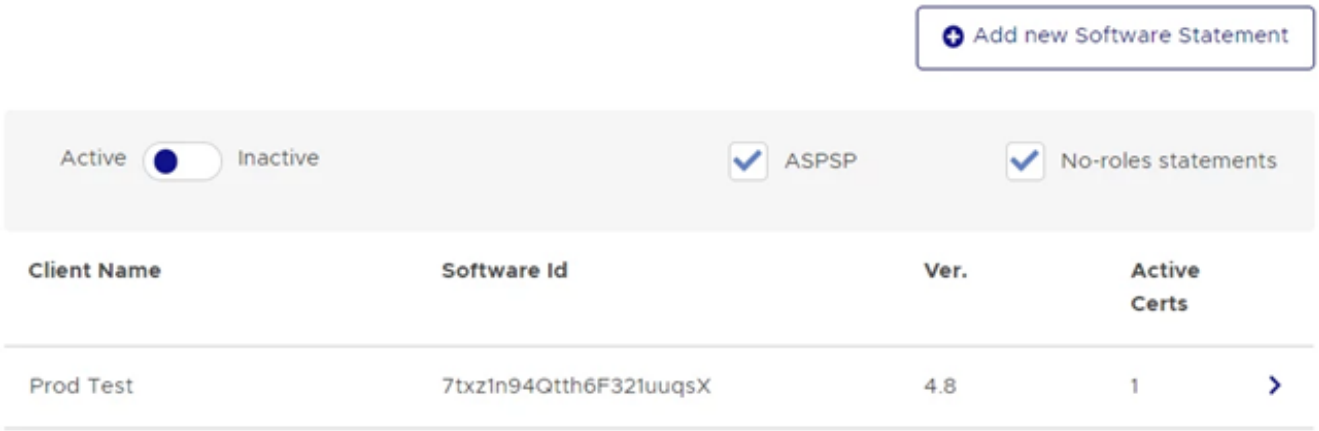


Figure 13: Step 1-2

Step 3: Completare il form

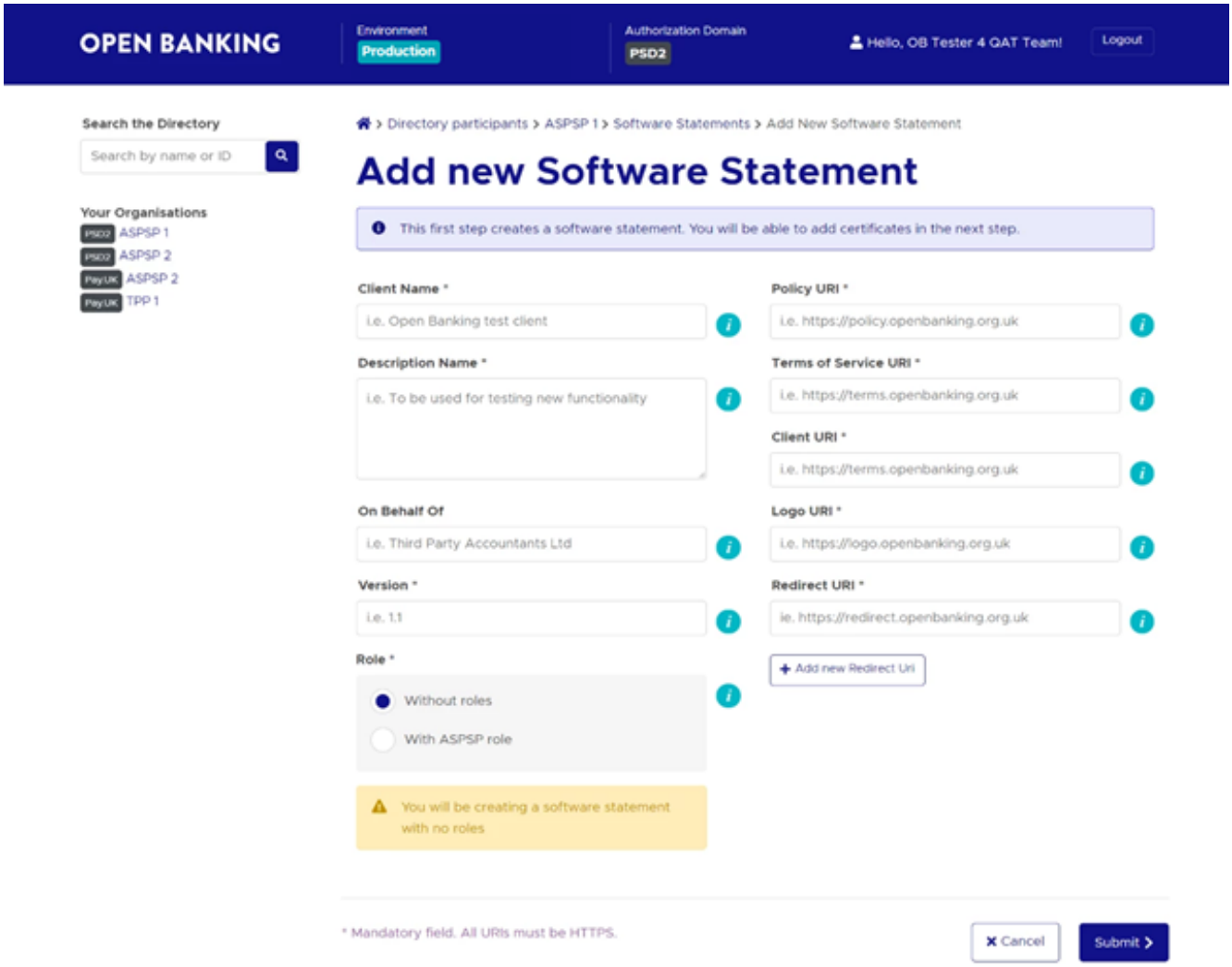


Figure 14: Step 3

Step 4: Verificare che il nuovo Software Statement sia stato creato.

Piccola nota sul redirect_uri: il campo redirect_uri nella Dynamic Client Registration è un elemento molto importante. Questo campo specifica l'URL a cui l'Authorization Server invierà la risposta dopo che l'utente ha completato l'autenticazione. In altre parole, redirect_uri è l'indirizzo a cui l'utente verrà reindirizzato dopo aver effettuato l'accesso con successo. Questo URL deve essere registrato e verificato dall'Authorization Server durante la registrazione del client per garantire che i token di accesso vengano inviati solo a indirizzi fidati.

Example SSA

The elements defined in the software statement will consist of the following values.

Note that there are inconsistent applications of booleans or "Active" strings in the current data model.

Note that there are inconsistent applications of status flags case sensitivity.

The attributes required to be displayed by ASPSPs

```
{
    "typ": "JWT",
    "alg": "ES256",
    "kid": "ABCD1234"
}
```

```
{
    "iss": "OpenBanking Ltd",
    "iat": 1492756331,
    "jti": "id12345685439487678",
    "software_environment": "production",
    "software_mode": "live",
    "software_id": "65d1f27c-4aea-4549-9c21-60e495a7a86f",
    "software_client_id": "OpenBanking TPP Client Unique ID",
    "software_client_name": "Amazon Prime Movies",
    "software_client_description": "Amazon Prime Movies is a moving streaming service",
    "software_version": "2.2",
    "software_client_uri": "https://prime.amazon.com",
    "software_redirect_uris":
    [
        "https://prime.amazon.com/cb",
        "https://prime.amazon.co.uk/cb"
    ],
    "software_roles": [
        "PISP",
        "AISP"
    ],

     "organisation_competent_authority_claims": [
       {
         "authority_id": "FMA", // Austrian Financial Market Authority
         "registration_id": "111111",
         "status": "Active",
         "authorisations":  [
                {
                    "member_state": "GBR",
                        "roles": [
                            "PISP",
                            "AISP"
                    ]
                },
                {
                    "member_state": "ROI",
                        "roles": [
                            "PISP"
                    ]
                }
            ]
       }
     ],
```

```
"software_logo_uri": "https://prime.amazon.com/logo.png",
"org_status": "Active",
"org_id": "Amazon TPPID",
"org_name": "OpenBanking TPP Registered Name",
"org_contacts": [
  {
    "name": "contact name",
    "email": "contact@contact.com",
    "phone": "+447890130558"
    "type": "business"
  },
  {
    "name": "contact name",
    "email": "contact@contact.com",
    "phone": "+447890130558",
    "type": "technical"
  }
],
```


Figure 16: Esempio di SSA_parte 2

7 Cenni di sicurezza

Quanto segue non vuole essere una trattazione esaustiva ma solamente elencare alcune pillole di sicurezza che bisogna tener presente e alcune casistiche di attacco a cui bisogna prestare attenzione.

7.1 Considerazioni varie di sicurezza

- L'endpoint di registrazione del client DEVE essere protetto dalla sicurezza del livello di trasporto (TLS 1.2 o superiore). Il livello di trasporto DEVE essere autenticato reciprocamente utilizzando certificati collegati all'autorità di certificazione di Open Banking;
- L'endpoint di registrazione ASPSP DEVE accettare messaggi HTTP POST con parametri di richiesta codificati nel corpo dell'entità utilizzando il tipo di contenuto "application/jwt";
- L'endpoint di registrazione ASPSP DEVE accettare tentativi di registrazione da qualsiasi canale autenticato reciprocamente il cui certificato sia collegato all'autorità di certificazione di Open Banking e non sia stato revocato.

Il passaggio di dati sensibili, come un token di accesso, attraverso l'URI è generalmente sconsigliato a causa delle debolezze di sicurezza associate a questo metodo.
Ecco perché:

- **Registrazione dell'URL:** gli URL vengono spesso registrati nei log dei server web. Se un token di accesso viene inviato tramite l'URI, esiste un'alta probabilità che l'URL contenente il token di accesso venga registrato. Questo potrebbe esporre il token di accesso a persone non autorizzate che hanno accesso ai log del server.
- **Condivisione dell'URL:** gli URL possono essere facilmente condivisi o potrebbero essere esposti se un utente dovesse salvare la pagina web o aggiungerla ai segnalibri. Se un URL contenesse un token di accesso, il token di accesso potrebbe essere condiviso o esposto insieme all'URL.

Per questi motivi, l'invio del token di accesso nell'intestazione della richiesta "Authorization" o nel corpo della richiesta HTTP è generalmente preferito. Tuttavia, il metodo URI può essere supportato dai server delle risorse se è impossibile trasportare il token di accesso in questi modi

7.2 Attacchi

Le tipologie di attacchi più importanti sono relative al modificare l'access token (un campo importante a cui bisogna prestare attenzione è il periodo di validità), replay attack e manipolazione di url. Un'ampia gamma di minacce può essere mitigata proteggendo il contenuto del token con una firma digitale o un Message Authentication Code (MAC). In alternativa, un bearer token può contenere un riferimento alle informazioni di autorizzazione, piuttosto che codificare direttamente le informazioni.

Utilizzare il canale di trasporto TLS -almeno la versione 1.2-, che garantisce le proprietà integrity e confidentiality, come misura di prevenzione dagli attacchi sul contenuto e divulgazione del token.

Un altro punto critico e a cui biogna prestare attenzione sono i cookie, perchè generalmente vengono trasmessi in chiaro.
Pertanto, qualsiasi informazione in essi contenute è a rischio di divulgazione. I bearer token, quindi, NON DEVONO essere memorizzati in cookie che possono essere inviati in chiaro.
Per far fronte alla cattura e alla riproduzione dei token, si forniscono le seguenti raccomandazioni:

- **La durata del token DEVE essere limitata**: un modo per ottenere questo risultato è inserire un campo "tempo di validità" nella parte protetta del token. Ad esempio, un token potrebbe avere un campo "expires_in" che indica il numero di secondi dopo i quali il token non è più valido. Si noti che l'uso di token di breve durata (un'ora o meno) riduce l'impatto della loro perdita;
- **Confidentiality negli scambi tra il client e il server di autorizzazione e tra il client e il resource server DEVE essere applicata**. Di conseguenza, nessun eavesdropper lungo il canale deve essere in grado di osservare lo scambio di token. Di conseguenza, un avversario sul percorso non può riprodurre il token;

- Quando si presenta il token a un server di risorse, **il client DEVE verificare l'identità di tale server di risorse**, come da Sezione 3.1 di "HTTP Over TLS" [RFC2818].
  Si noti che il client DEVE convalidare la catena di certificati TLS quando effettua queste richieste a risorse protette. La presentazione del token a un server di risorse non autenticato e non autorizzato, oppure non convalidare la catena di certificati consentirà agli avversari di rubare il token e di ottenere l'accesso non autorizzato alle risorse protette.

7.3 Request Validation

Prima di emettere una risposta di registrazione del client, l'ASPSP DEVE eseguire i seguenti controlli:

- **l'ASPSP DEVE verificare se il TPP che ha avviato la connessione TLS sia lo stesso TPP elencato nell'SSA**. Nel caso in cui un gateway o un'altra infrastruttura termini il canale mTLS davanti all'endpoint di registrazione, il certificato utilizzato per avviare la connessione o una parte di tale certificato (come DN e Issuer) deve essere messo a disposizione dell'ASPSP per essere convalidato rispetto a quanto indicato nell'SSA;
- **La richiesta di registrazione DEVE essere firmata con una chiave contenuta nel JWKS a cui fa riferimento l'SSA incluso nella richiest**a. Questo assicura che venga eseguita una prova di possesso del titolare della chiave che dimostri che l'applicazione TPP era il destinatario originario dell'SSA quando l'OB l'ha emesso;
- **L'SSA DEVE essere convalidato secondo [RFC7519], compresa la convalida della firma e della finestra di validità**;
- **La firma JWT deve essere convalidata**: ciò comporta il recupero del keystore jwks sia per l'OB che per il TPP. La posizione del keystore dell'OB sarà pubblicata come parte della specifica della directory, mentre quella del TPP sarà inclusa nella dichiarazione del software.

7.4 Client Registration Request

Per registrarsi come client presso un ASPSP, il TPP invia un HTTP POST all'endpoint di registrazione dell'ASPSP. La richiesta DEVE essere presentata nel formato di un JWT conforme a [RFC7519] e DEVE utilizzare il metodo HTTP POST, utilizzando il metodo application/jwt.

8 PISP vs AISP

All'inizio del presente documento è stato definito il funzionamento di un PISP e di un AISP. Si procede ora a riportare nuovamente una breve descrizione e il funzionamento di entrambi:

8.1 Payment Initiation Service Provider (PISP)

I PISP sono i servizi online che chiedono all'utente (il PSU) di autorizzare un pagamento da un conto di cui l'utente è titolare a un altro conto. Il movimento da autorizzare potrebbe essere un pagamento istantaneo o trattarsi di ordini permanenti, considerando sia i pagamenti nazionali e sia internazionali. Di seguito il funzionamento:

8.1.1 Funzionamento

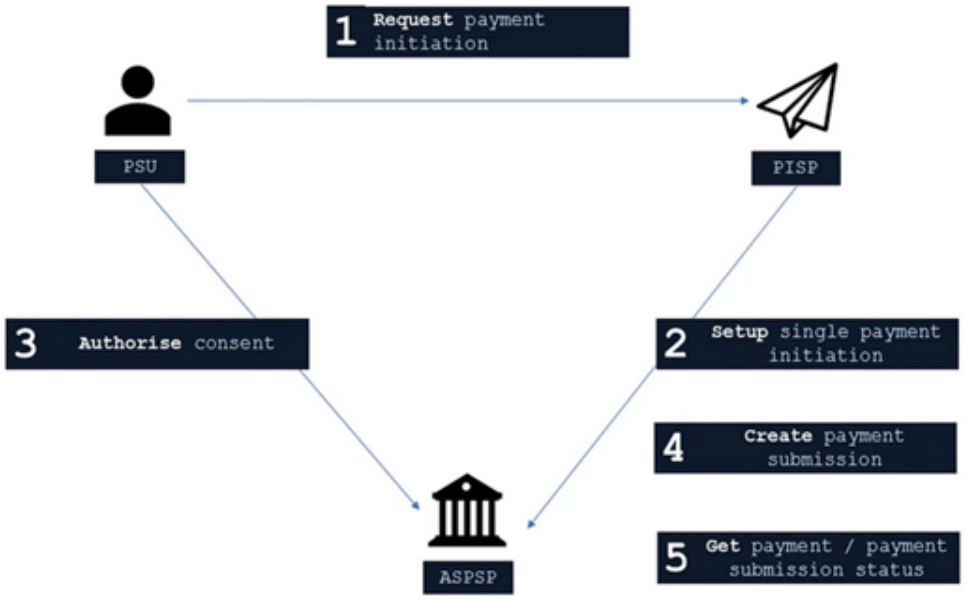


Figure 17: Flusso operazioni PISP

Fase 1: Richiesta di autorizzazione al pagamento

- Questo flusso inizia con il consenso di un PSU a effettuare un pagamento. La richiesta viene inviata tramite un PISP;
- In questa fase è possibile specificare i dettagli del conto del debitore

Fase 2: Impostazione dell'avvio del pagamento singolo

- Il PISP si collega all'ASPSP che gestisce il conto di pagamento del PSU e crea una nuova risorsa di pagamento. Informa l'ASPSP che uno dei suoi PSU intende effettuare un pagamento. L'ASPSP risponde con un identificativo per la risorsa (PaymentId - che è l'identificativo dell'intento);
- Questa fase viene eseguita effettuando una richiesta POST alla risorsa pagamenti

Fase 3: Autorizzare il consenso

- Il PISP reindirizza il PSU all'ASPSP. Il reindirizzamento include il PaymentId generato nella fase precedente. Questo permette all'ASPSP di correlare il pagamento che è stato impostato;
- L'ASPSP quindi autentica il PSU (ad esempio attraverso la procedura di Strong Customer Authentication, SCA, se l'ASPSP determina che non si applica nessuna delle esenzioni SCA) e aggiorna internamente lo stato della risorsa pagamenti per indicare che il pagamento è stato autorizzato;

- In questa fase, il PSU seleziona il conto del debitore, se non è stato specificato in precedenza nella fase 1. Il PSU viene reindirizzato alla risorsa pagamenti per indicare che il pagamento è stato autorizzato;
- Il PSU viene reindirizzato al PISP

Fase 4: Creare l'invio del pagamento

- Una volta che il PSU è stato reindirizzato al PISP, quest'ultimo crea una risorsa payment-submissions per indicare che il pagamento creato nei passaggi precedenti debba essere inviato per l'elaborazione
- Per poter inviare ed elaborare la richiesta di pagamento è necessario effettuare una richiesta POST alla risorsa payment-submissions;
- L'ASPSP restituisce al PISP il PaymentSubmissionId

Fase 5: Ottenere lo stato dell'invio del pagamento

- Se l'ASPSP fornisce un'API di stato, il PISP può verificare lo stato del pagamento (con il PaymentId) o dell'invio del pagamento (con il PaymentSubmissionId) attraverso una richiesta GET alla risorsa pagamenti o invii di pagamento

Nel seguente sequence diagram vengono elencati le varie azioni compiute dai vari attori presenti:

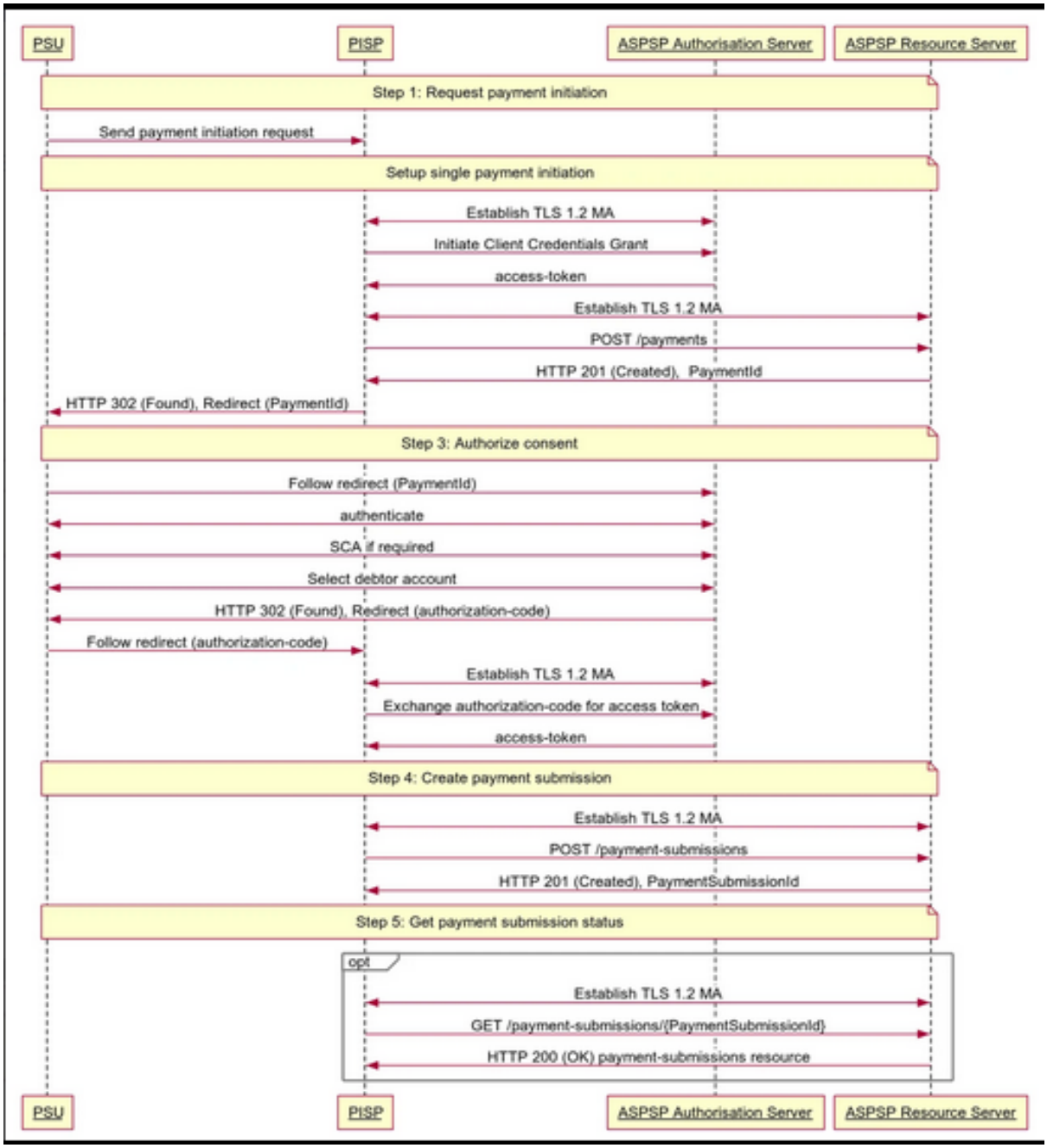


Figure 18: Sequence diagram PISP

8.2 Account Information Service Provider (AISP)

Gli AISP sono le applicazioni di contabilità e confronto, ovvero applicazioni che vogliono leggere i dati del vostro conto e fornire una sorta di valore aggiunto. Questo può andare dalla visualizzazione della cronologia delle transazioni e del saldo alla creazione di ordini permanenti e addebiti diretti. Di seguito vengono riportati i processi logici dietro al comportamento di un AISP:

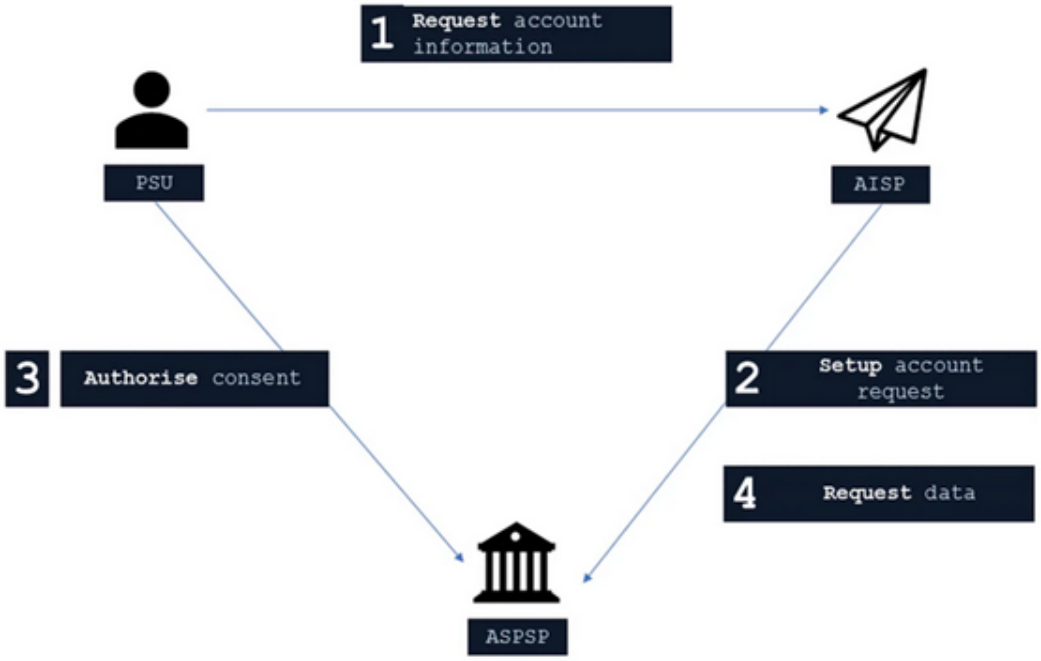


Figure 19: Flusso operazioni AISP

Fase 1: Richiesta di informazioni sul conto

- Questo flusso inizia con il consenso del PSU nel permettere a un AISP l'accesso ai dati dell'account

Fase 2: Impostazione della richiesta del conto

- L'AISP si connette all'ASPSP che gestisce gli account del PSU e crea una risorsa di richiesta informazioni dell'account. Questo step permette all'ASPSP di sapere che uno dei suoi PSU sta concedendo l'accesso alle informazioni sui conti e sulle transazioni a un AISP. L'ASPSP risponde con un identificatore per la risorsa (AccountRequestId)
- La richiesta precedente può essere eseguita effettuando una richiesta POST all'endpoint /account-requests
- Il payload di configurazione includerà questi campi, che descrivono i dati che il PSU ha acconsentito all'AISP:
  - → **Permessi:** un elenco di cluster di dati per i quali è stato dato il consenso all'accesso;
  - → **Data di scadenza:** una scadenza opzionale per quando l'AISP non avrà più accesso ai dati del PSU;
  - → **Periodo di validità delle transazioni:** l'intervallo di date From/To che specifica un periodo di cronologia delle transazioni a cui l'AISP può accedere.
- Un AISP può essere un broker di dati per altre 4 parti, e quindi è possibile che un cliente abbia più richieste per gli stessi conti, con diversi parametri di consenso/autorizzazione concordati

Fase 3: Autorizzare il consenso

- L'AISP reindirizza il PSU all'ASPSP. Il reindirizzamento include l'AccountRequestId generato nel passaggio precedente. Ciò consente all'ASPSP di correlare l'account-request che è stato impostato.
- Successivamente l'ASPSP autentica il PSU e aggiorna internamente lo stato della risorsa AccountRequest per indicare che la richiesta di account è stata autorizzata
- Il principio che abbiamo concordato è che il consenso è gestito tra il PSU e l'AISP, quindi i dettagli della richiesta di account non possono essere modificati (con l'ASPSP) in questa fase. Il PSU potrà solo autorizzare o rifiutare i dati della richiesta del conto nella sua interezza

- Durante l'autorizzazione, il PSU seleziona i conti autorizzati per la richiesta dell'AISP (nell'interfaccia bancaria dell'ASPSP)
- Il PSU viene reindirizzato all'AISP

Fase 4: richiesta di dati

- Si effettua una richiesta GET alla risorsa in questione
- Gli AccountId univoci validi per la richiesta del conto verranno restituiti con una chiamata a GET /accounts. Questa sarà sempre la prima chiamata quando un AISP avrà un token di accesso valido

Vale la pena notare che in questo flusso, tutte le comunicazioni backchannel utilizzano la mutua autenticazione Transport Layer Security (TLS) 1.2.
Nel seguente sequence diagram vengono elencati le varie azioni compiute dai vari attori presenti:

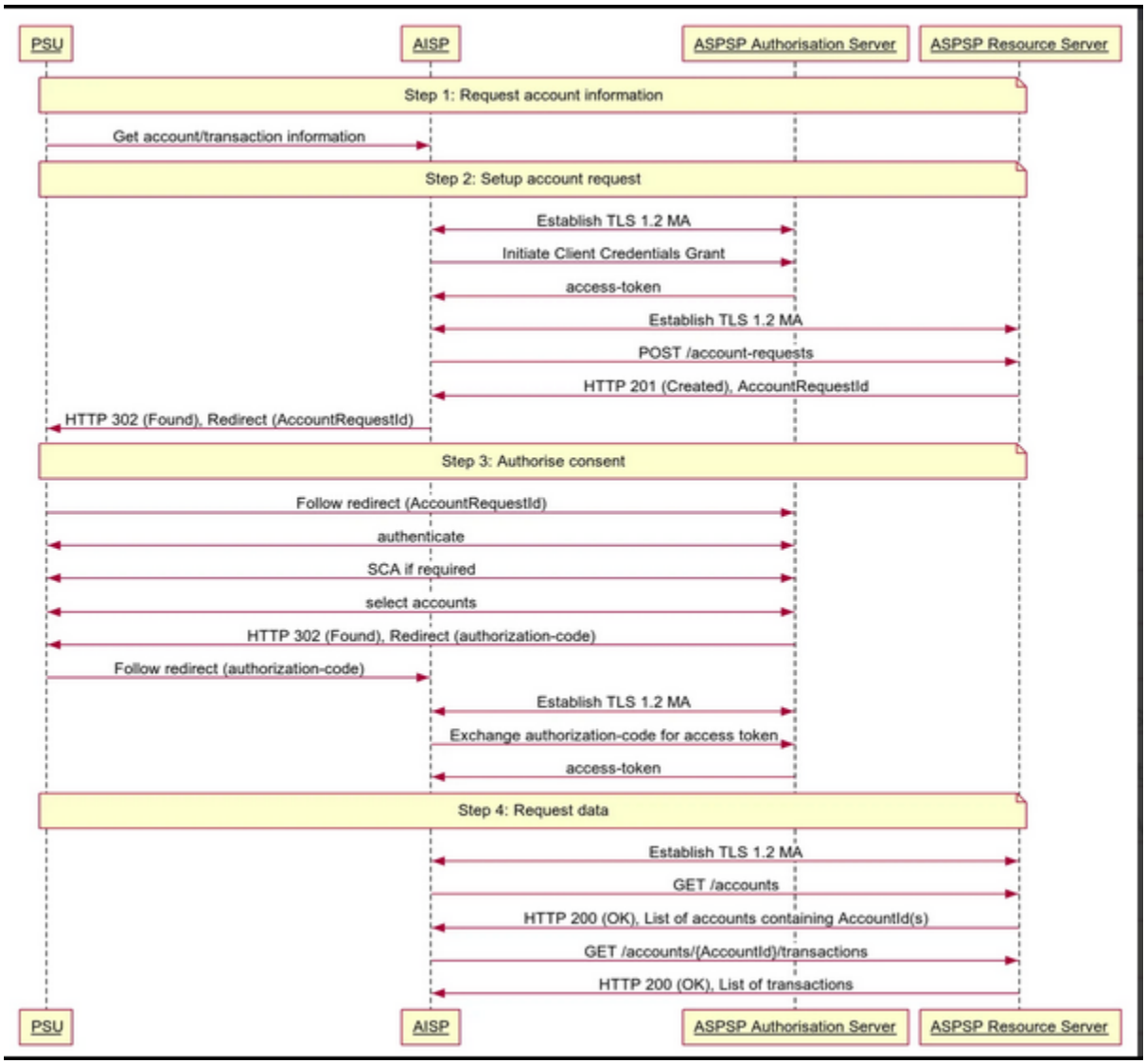


Figure 20: Sequence diagram AISP

9 Cenni sul servizio Confirmation of Payee (CoP)

Nell'ambito delle transazioni la Confirmation of Payee è un servizio che mira a prevenire alcuni tipi di truffe (frodi di pagamento push autorizzato) e pagamenti accidentalmente errati da parte di clienti che cercano di effettuare pagamenti a beneficiari. Il servizio è caratterizzato da un'architettura distribuita basata su API che consente ai partecipanti di scambiare messaggi in modo sicuro per verificare che il nome del conto sia corretto prima della creazione di un modello di beneficiario o dell'avvio e/o dell'incasso di un pagamento. Nell'impostare un nuovo pagamento per un cliente, un PSP è in grado di verificare il nome, il codice di smistamento, il numero di conto e, se applicabile, i dati di riferimento secondari come il numero di iscrizione all'albo della società e il tipo di conto della persona o dell'organizzazione del destinatario previsto, confrontandoli con i dati presenti sul conto. Il CoP è gestito dall'operatore dei sistemi di pagamento Pay.UK . Il servizio di Confirmation of Payee è la trasposizione nel mercato inglese del servizio di Name Check, già presente nelle banche aderenti al consorzio CBI (si tornerà più avanti su questo aspetto).

Pay.UK è l'organizzazione che gestisce i principali sistemi di pagamento del Regno Unito, inclusi Bacs, Faster Payments e Cheque and Credit Clearing, e il suo compito è di garantire che i pagamenti digitali tra banche, società di costruzione e fornitori di servizi di pagamento siano sicuri, affidabili e semplici. Pay.UK, inoltre, collabora strettamente con l'ecosistema di Open Banking per migliorare la sicurezza e l'efficienza dei pagamenti.
Pay.UK ha implementato le normative della PSD2 attraverso una serie di misure volte a migliorare la sicurezza e l'efficienza dei pagamenti. La PSD2, come detto in precedenza, ha introdotto requisiti di Autenticazione Forte del Cliente (SCA) e standard di comunicazione sicura che Pay.UK ha integrato nei suoi sistemi di pagamento. Questi requisiti includono:

1. **Autenticazione Forte del Cliente (SCA):** Pay.UK ha adottato misure per garantire che tutte le transazioni siano soggette a un'autenticazione forte, riducendo il rischio di frodi;

2. **Accesso ai Conti:** La PSD2 ha aperto il mercato dei pagamenti a fornitori di servizi di pagamento di terze parti (TPP), permettendo loro di accedere ai dati dei conti con il consenso del cliente. Pay.UK ha facilitato questo accesso attraverso interfacce di programmazione delle applicazioni (API) sicure.

Come accennato a inizio capitolo, il servizio di NameCheck è attualmente disponibile verso tutte le banche appartenenti al CBI S.c.p.a. (Corporate Banking Interbancario), società che gestisce il servizio di CBI, il quale consente alle imprese di poter gestire la propria tesoreria in modalità telematica. In particolare attraverso un unico collegamento l'impresa può ottimizzare tutti i rapporti di conto che intrattiene presso più banche.
Il servizio CBI:

- centralizza i rapporti di un'impresa verso l'intero sistema bancario in un unico punto;
- fornisce una vasta gamma di funzioni finanziarie, informative e commerciali;
- utilizza sempre lo stesso standard di comunicazione;
- fornisce un servizio basato sulla cooperazione delle banche, senza ridurre gli spazi competitivi

All'interno del CBI è possibile trovare, giusto per citare le più importanti, banche come Intesa Sanpaolo S.p.A, UniCredit SpA, Bper banca e Banca Monte dei Paschi di Siena S.p.A.
La grande sfida che è stata affrontata negli ultimi mesi è stata consentire a ecosistemi che utilizzano interfacce funzionali diverse di coesistere e permettere l'utilizzo del servizio NameCheck e Confirmation of Payee, rispettivamente nelle seguenti casistiche:

- **Pay.UK calling- CBI responding:** in cui una banca appartenente al circuito bancario inglese tenta di raggiungere una banca appartenente al circuito CBI;
- **CBI calling - Pay.UK responding:** in cui una banca appartenente al circuito bancario CBI tenta di raggiungere una banca appartenente al circuito inglese Pay.UK

Di seguito vengono riportati degli esempi di come potrebbero essere le interfacce di CBI e di Pay.UK. In entrambi i casi sono stati utilizzati dati e parametri di fantasia.

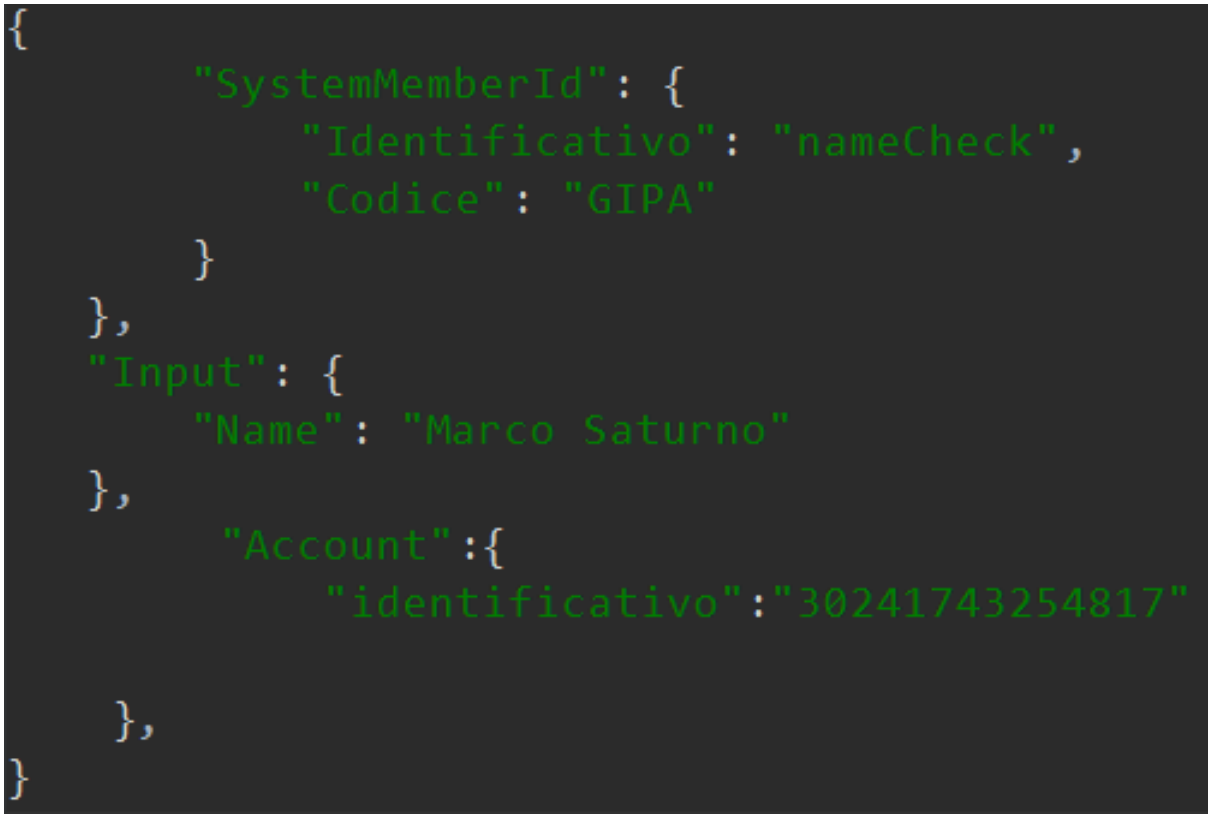


Figure 21: Esempio di possibile interfaccia CBI

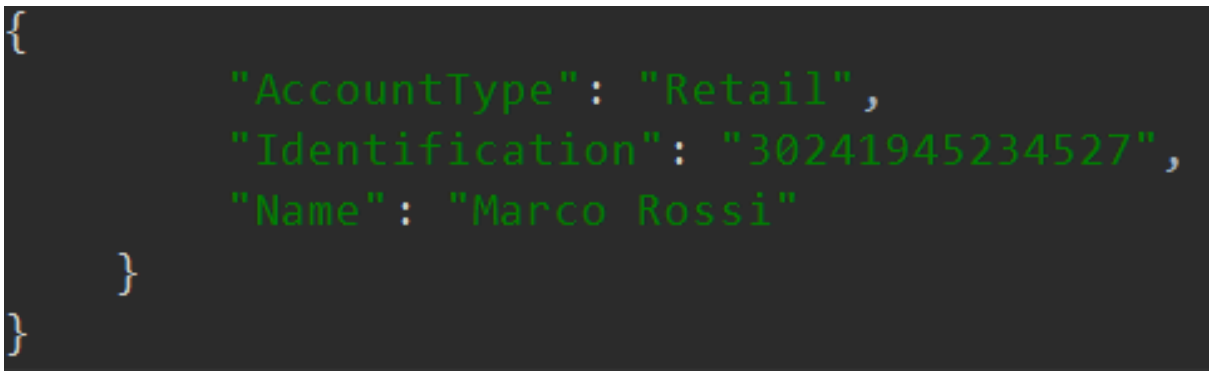


Figure 22: Esempio di possibile interfaccia Pay.UK

9.1 Attori

Nella tabella seguente vengono elencati gli attori protagonisti nel servizio di CoP:

Attore	Descrizione
Payer	Colui che trasferisce fondi in un Account
CoP Requester	CoP Participant autorizzato a inviare una CoP Request
CoP Responder	CoP Participant autorizzato a inviare una risposta a CoP Request
Payee	Destinatario della transazione

Table 3: Descrizione attori

Per completezza di esposizione verrà ora descritto un possibile Sequence Diagram delle operazioni da compiere per inviare una CoP Request:

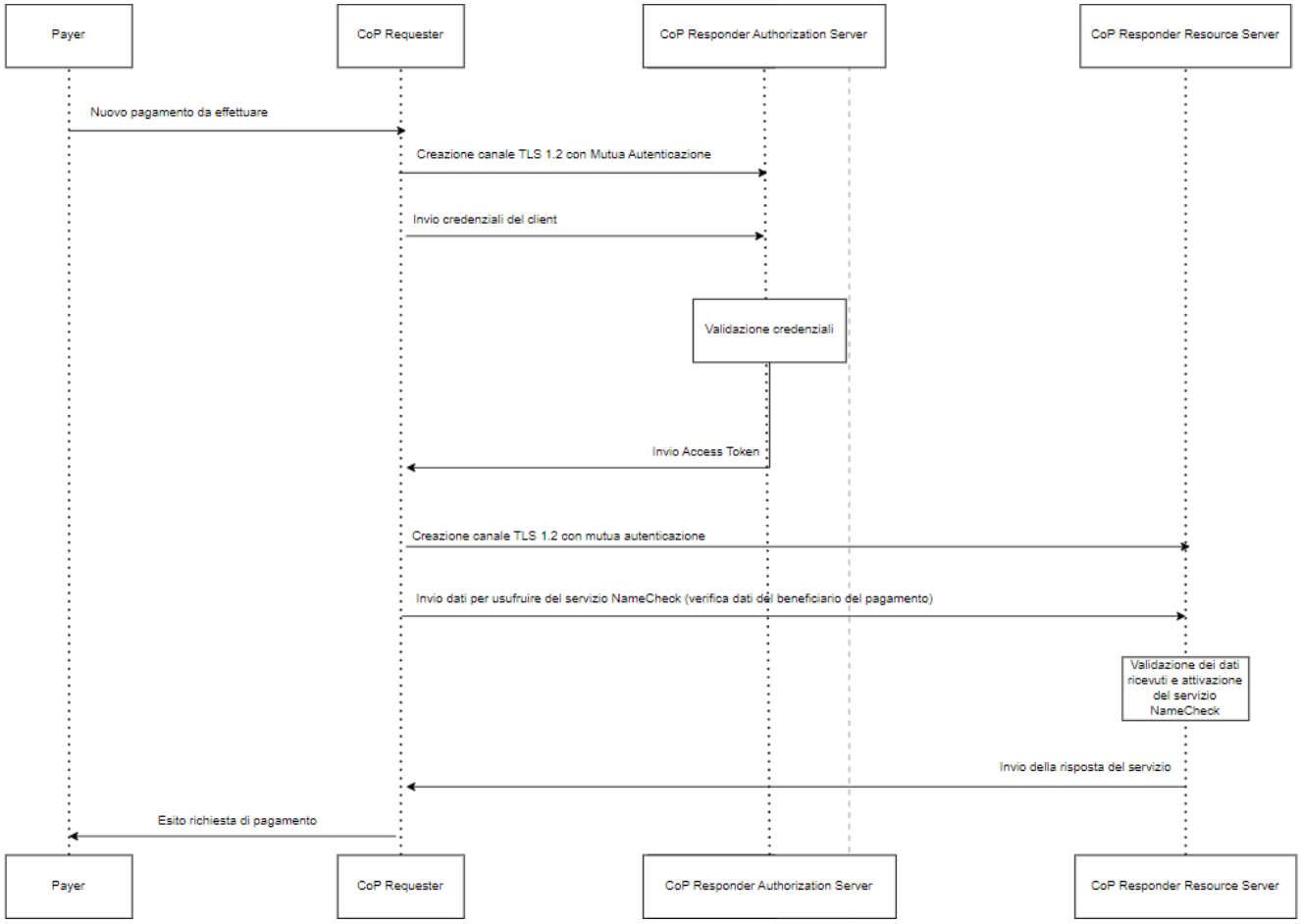


Figure 23: Sequence diagram CoP

Inseriti i dati della transazione il CoP Requester stabilisce una connessione sicura (utilizzando almeno TLS 1.2 con la proprietà di mutua autenticazione) con l'Authorization Server del CoP Responder fornendo le credenziali client. Dopo aver controllato e validato le credenziali verrà fornito al CoP Requester un access token che servirà per poter controllare l'esistenza dei dati del destinatario della transazione. Per poter usare l'access token è quindi necessario stabilire una connessione sicura (anche qui mediante l'uso di almeno TLS 1.2 con proprietà di mutua autenticazione) tra CoP Requester e CoP Responder Resource Server e inviare la richiesta di verifica del nome con il token ricevuto in precedenza. Infine il Resource Server invierà il risultato dei controlli al CoP Requester il quale provvederà a mostrarli al Payer.

9.2 Elementi chiave

Di seguito verranno descriti gli elementi chiave necessari per poter implementare il servizio Confirmation of Payee:

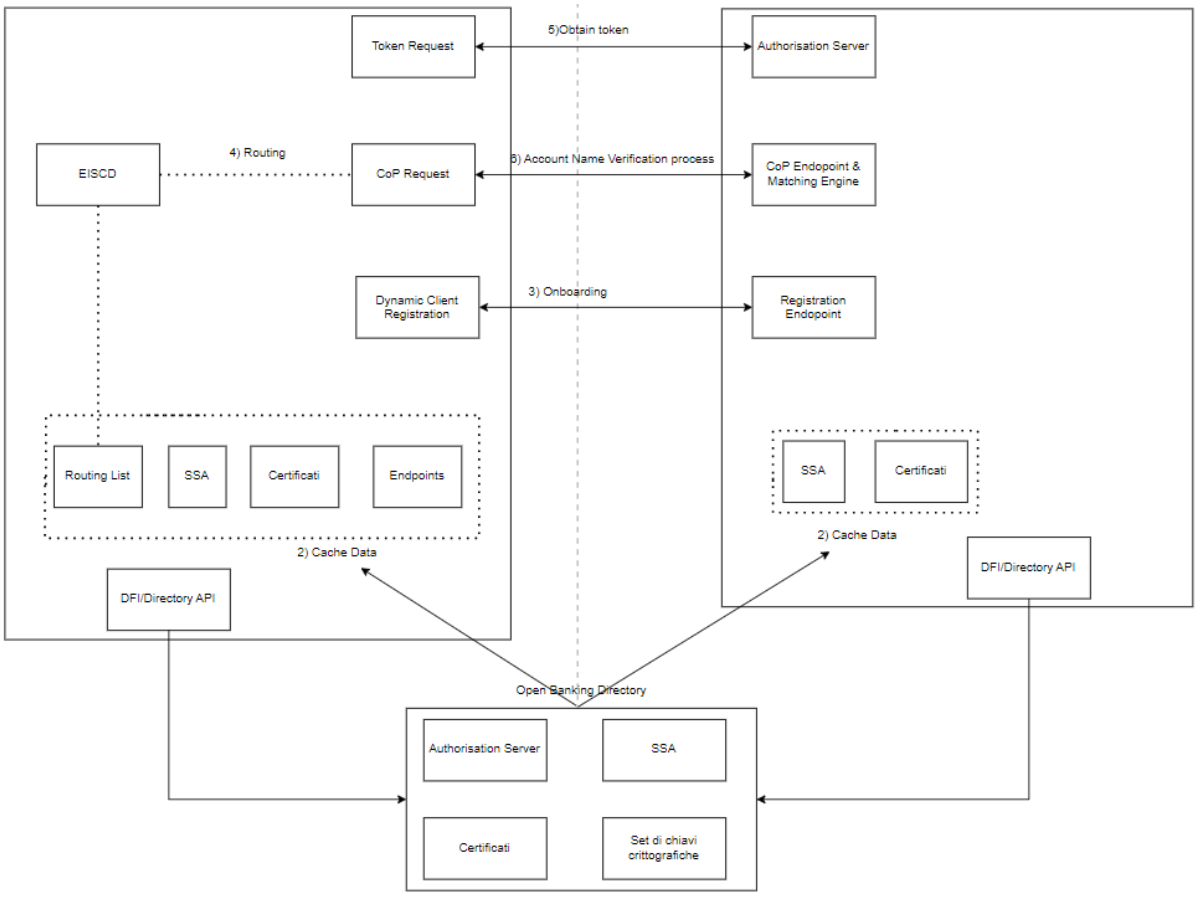


Figure 24: Elementi chiave CoP

SRD List (Secondary Reference Data): nel contesto del CoP, l'SRD viene utilizzato per identificare alcuni conti bancari.
Endpoints: gli endpoint sono punti di comunicazione in una rete o in un servizio API dove le risorse possono essere accedute. Nel contesto del CoP, gli endpoint si riferiscono ai punti di accesso per l'invio di richieste CoP e la ricezione di risposte. Ovviamente richieste e risposte contengono certificati per garantire la propria autenticità e la sicurezza delle comunicazioni (si parla di Transport certificate e Certificato digitale)
EISCD: è una versione estesa dell'Industry Sort Code Directory (ISCD). L'EISCD contiene informazioni relative agli uffici o alle filiali dei fornitori di servizi di pagamento, come il codice di ordinamento, l'identificatore dell'ufficio bancario (BIC), il titolo della filiale, le operazioni di compensazione in cui partecipa l'ufficio bancario, comprese le informazioni di regolamento, l'indirizzo postale e il numero di telefono. L'EISCD viene utilizzato per vari scopi, tra cui:

- Verificare i dettagli dell'account nei pagamenti elettronici, riducendo così il numero di pagamenti respinti e il costoso processo di correzione
- Accedere a informazioni sull'indirizzo e sul numero di telefono, ad esempio, istruzioni di addebito diretto, con dettagli sull'indirizzo
- Accedere alle informazioni BIC per gli uffici bancari del Regno Unito

Modulo DFI/Directory API: è un componente chiave dell'Open Banking Directory, fornisce funzionalità che erano precedentemente disponibili solo tramite il DFI (Directory File Interface). Queste funzionalità includono:

- Gestione dei contatti dell'organizzazione
- Gestione delle dichiarazioni software
- Generazione di asserzioni di dichiarazioni software

- Caricamento/gestione delle chiavi di firma
- Caricamento/gestione delle chiavi di crittografia
- Generazione/gestione di OB (Open Banking)

In sostanza, il modulo DFI/Directory API consente alle organizzazioni di gestire i loro profili, i certificati digitali e le dichiarazioni software necessarie per connettersi con i fornitori di conti. Questo modulo è fondamentale per garantire la sicurezza e l'efficienza delle transazioni nell'ecosistema Open Banking.

Modulo JWKS S3 Key Store. JWKS è l'acronimo di JSON Web Key Set, che è un insieme di chiavi contenenti le chiavi pubbliche utilizzate per verificare qualsiasi JSON Web Token (JWT) emesso dal server di autorizzazione. Il modulo JWKS S3 Key Store potrebbe essere utilizzato per memorizzare le chiavi pubbliche utilizzate per firmare e verificare le asserzioni del Software Statement (SSA). Queste asserzioni sono utilizzate durante la registrazione dinamica del client, un processo che consente ai Third Party Provider (TPP) di registrare i loro client OAuth con i fornitori di conti.

Pertanto il servizio di Confirmation of Payee è fondamentale per evitare qualunque tipo di frode informatica ma anche solo che un pagamento venga inviato alle persone e/o a enti sbagliati. Ciò è reso possibile grazie al controllo che viene effettuato prima di effettuare un pagamento.
La logica concettuale dietro a questo servizio può essere riassunta nei seguenti punti:

1. L'utente tramite l'interfaccia predisposta della propria banca compila i dati (tra i quali nome intestatario e codice iban del destinatario) per effettuare un pagamento;
2. La banca mittente contatta e si "connette" alla banca del destinatario;
3. La banca del mittente "trasmette" iban e nome intestatario del pagamento alla banca destinataria;
4. La banca destinataria controlla nei propri sistemi se i dati presenti nel pagamento esistano e siano corretti;
5. La banca destinataria fornisce un feedback (match, close match, no match) alla banca mittente;
6. A seguito del risultato viene gestito l'esito del pagamento

Come viene descritto nella seguente immagine: il servizio di Confirmation of Payee inizia nel momento in cui viene creata una "nuova richiesta di pagamento". In questo momento il pagante inserisce tutti i dati del beneficiario del pagamento: intestatario (account name), sort code e account number (corrispondente all'IBAN) e la tipologia di account (personal o business).
A questo punto entra veramente in funzione l'algoritmo (vengono raggiunti determinati endopoint per chiamare le API esposte dalla banca in questione, si effettua una procedura di autenticazione e infine si effettua il controllo dei dati). L'algoritmo, può ritornare uno dei seguenti risultati:

- **Yes (match):** in cui l'intestatario, l'IBAN e il tipo di account inseriti sono corretti;
- **No (close match):** il nome è un close match, ovvero potrebbe esserci stato qualche errore nella digitazione e verrà mostrata una schermata con suggerimenti su chi si aveva intenzione di inviare il bonifico. Un'altra situazione di close match si ha quando i dati (intestatario e IBAN) sono corretti ma la tipologia di account inserita è errata (ad esempio si è inserito personal ma invece era un account business);
- **No:** i dati forniti non sono corretti e non si ha nessun riscontro;
- **Unavailable:** situazione generica di errore scaturita da timeout, account inesistente, problemi di connessione, ecc...

Pertanto a seconda delle implementazioni il sistema potrebbe ad esempio dire all'utente che ha commesso un errore e mostrare un "suggerimento" di risultato basato su quanto immesso dall'utente oppure terminare solamente il pagamento.
Come detto in precedenza, lo scopo del servizio di conferma di pagamento è importante per evitare qualunque tipo di frode informatica ma anche un pagamento inviato alle persone e/o enti sbagliati.

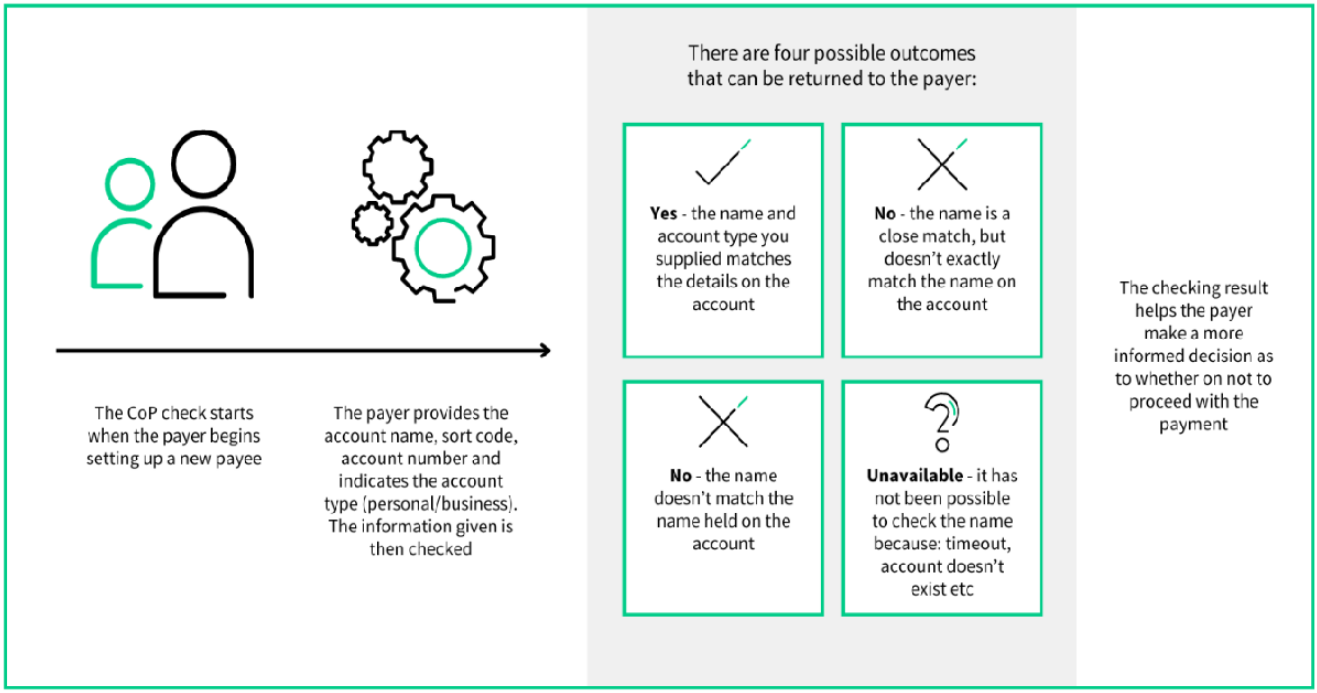


Figure 25: Esempio COP

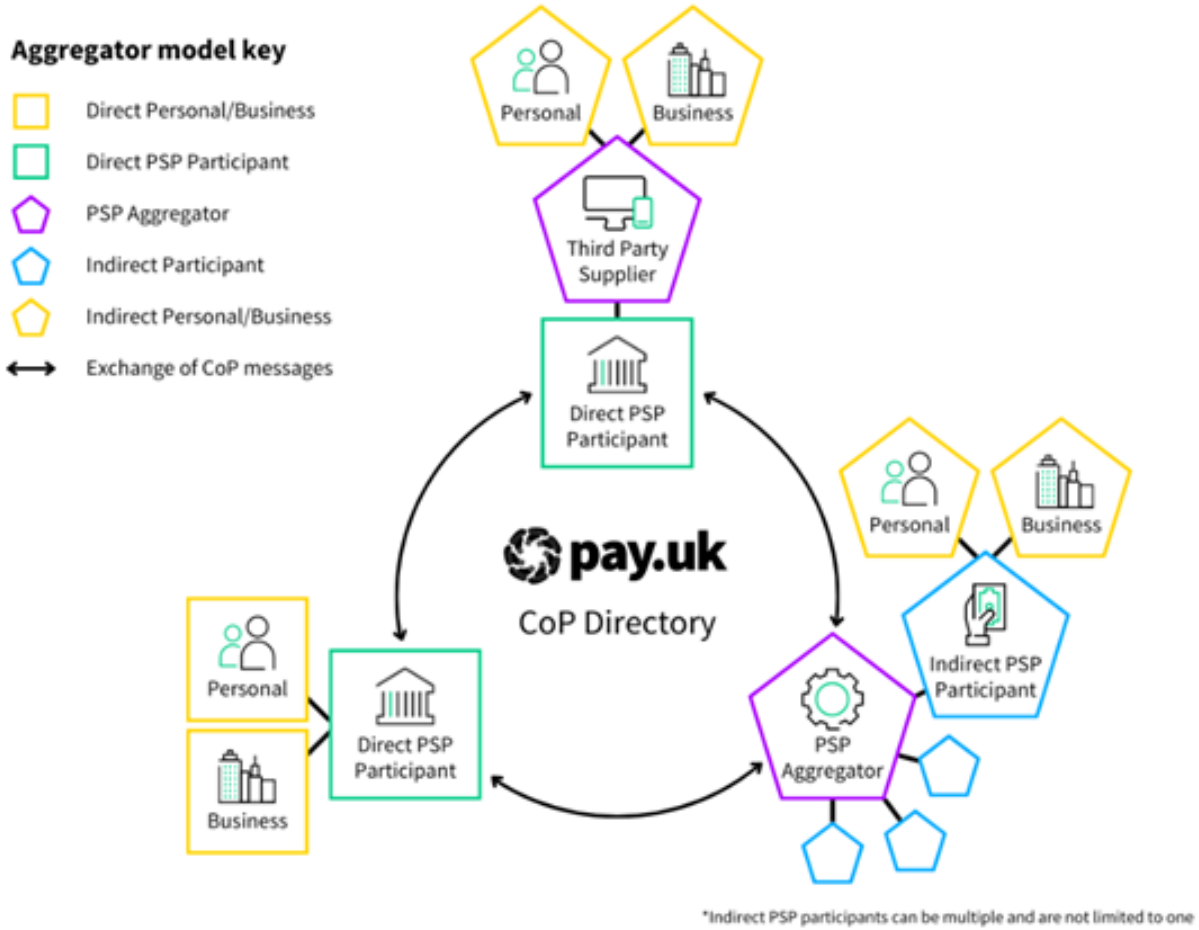


Figure 26: Esempio PayUK

Come riporta l'immagine tratta dal sito di Pay.UK ci sono due modi per poter partecipare nella CoP: il modello diretto o il modello aggregator. Di seguito viene riportata la differenza tra i due modelli. La differenza tra il modello Direct e il modello Aggregator per la Confirmation of Payee (CoP) riguarda principalmente il modo in cui le istituzioni di pagamento (PSP) si interfacciano con il servizio.

Modello Direct:

- Richiede un onboarding meticoloso, contratti e connessioni multipli, sottolineando la necessità di manutenzione e supporto continui;
- Ogni PSP deve stabilire una connessione con tutti gli altri PSP, creando e mantenendo numerose connessioni;
- I PSP possono mantenere il controllo utilizzando i propri certificati di identità per connettersi con altri PSP;
- Questo modello sta evolvendo verso un processo più snello, richiedendo solo che il PSP e Pay.UK firmino i contratti

Modello Aggregator:

- Semplifica l'ecosistema CoP consentendo ai fornitori di servizi tecnici (TSP) di aderire;
- Gli aggregator possono utilizzare i loro certificati per identificarsi con altri PSP, semplificando il processo di connessione;
- Questo modello opera esclusivamente sul nuovo directory di Pay.UK, senza prevedere l'attuale directory OBIE

In sintesi, il modello Direct offre maggiore autonomia ai PSP, ma richiede più sforzo per l'onboarding e la manutenzione. D'altra parte, il modello Aggregator offre un approccio più inclusivo e semplificato, ma opera esclusivamente sul nuovo directory di Pay.UK.

Partecipante indiretto: un "partecipante indiretto" è un'istituzione finanziaria che non ha un collegamento diretto con PayUK, ma accede al sistema tramite un partecipante diretto. Quindi, un individuo che apre un conto corrente presso una banca non è considerato un partecipante (né diretto né indiretto) nel contesto di PayUK. Si è considerati solamente come un cliente della banca, mentre la banca stessa, se è un partecipante diretto, gestisce le transazioni di pagamento per conto del cliente attraverso il sistema PayUK.

Perché una banca dovrebbe scegliere di essere un partecipante indiretto?
In sintesi, le ragioni principali sono due: costi e velocità.
In primo luogo, diventare partecipante diretto e stabilire una connessione a un CSM ("Clearing and Settlement Mechanism" (Meccanismo di Compensazione e Regolamento)) implica la realizzazione di progetti importanti, inoltre, senza un volume minimo di transazioni, non c'è ritorno sull'investimento.
I partecipanti indiretti sono per lo più piccole banche che si avvalgono dei servizi di banche più grandi per lo scambio delle loro transazioni. In questo modo possono risparmiare i costi di gestione e mantenimento di una connessione diretta a un CSM.

10 Appendice

Nella seguente appendice verrà trattata la tematica del testing. Quest'ultima non è strettamente correlata all'argomento principale del presente documento ma risulta interessante capire l'evoluzione e il perchè l'attività di testing risulta importante in ambito aziendale. Infine verrà riportata anche una breve descrizione del software JMeter, con il quale sono stati effettuati i test di funzionamento.

10.1 Testing

A partire dai primi anni 2000 con il costante incremento delle funzionalità dei software è stato reso necessario rivoluzionare il concetto di development e di testing di un prodotto per contrastare l'aumento della complessità del codice e, come conseguenza, l'aumento di errori e difetti che, se identificati tardivamente rischiano di generare un nuovo effort di sviluppo e un incremento dei costi non necessari. Tradizionalmente, le procedure di test manuale richiedono, da parte del tester, la definizione, scrittura ed esecuzione di tutti i test case ritenuti necessari per verificare il funzionamento del codice, con notevole dispendio di tempo e probabilità di commettere vari errori. Una strategia di test automation, invece, si può identificare come un insieme di metodologie, tecnologie e strumenti di automazione che ha l'obiettivo di ridurre il più possibile procedure e interventi manuali, tramite l'uso di script programmati per eseguire in automatico i test e validare il software.
L'obiettivo è completare l'esecuzione dei test in minor tempo, accelerare velocità e qualità delle operazioni di collaudo e verifica, e ottimizzare nel loro complesso i processi di software testing. Le aziende hanno iniziato quindi ad adottare pratiche Agile, con un ciclo di vita di sviluppo accelerato caratterizzato da frequenti feedback dei clienti. Ciò ha determinato l'adozione di strumenti per la continuous integration e la continuous delivery che automatizzavano i processi di build, test, configurazione e distribuzione. Tuttavia, funzioni chiave come sviluppo, test e delivery in produzione venivano eseguite da team separati che operavano all'interno dei propri silos (termine che deriva dal mondo agricolo -contenitore chiuso e sigillato- e che vuole trasmettere l'idea di isolamento di ogni dipartimento senza possibilità di comunicazione tra i vari team di lavoro).
Questa modalità di lavoro comportava inefficienze e blocchi nel ciclo di vita dello sviluppo del software dovuti anche a una comunicazione limitata e a obiettivi separati. Per superare questi problemi "strutturali" è stato necessario cambiare approccio andando verso il DevOps, ovvero l'insieme di filosofie organizzative, pratiche e strumenti in grado di abilitare i team piccoli e interfunzionali, noti anche come Squad (team composti da membri con competenze diverse come sviluppatori, tester, ecc. che lavorano insieme per raggiungere un obiettivo comune), che sono responsabili della continuous delivery e della qualità degli aggiornamenti dei prodotti, end-to-end. Inizialmente DevOps ha unificato solo le operazioni IT e di sviluppo, mentre i test continuavano a essere eseguiti da un team separato con metodi prevalentemente manuali.
Ciò ha contribuito a risolvere le problematiche legate alla distribuzione e al monitoraggio delle applicazioni cloud e ha inoltre determinato la creazione di pipeline CI/CD completamente automatizzate. Tuttavia, non offriva cicli di rilascio molto più rapidi, poiché i test erano in silos e spesso eseguiti con un processo manuale che richiedeva molto tempo.

Prima di passare a parlare dei test passiamo a definire tutti i concetti introdotti in precedenza:

10.2 DevOps (Development and Operations)

E' un metodo che ha l'obiettivo di integrare le attività (e i team) di sviluppo e di messa in produzione di un sistema software. Il development team sviluppa il software, mentre l'operations team lo integra nell'ambiente di produzione. L'operations team si occupa dell'approvvigionamento (provisioning), della gestione dell'infrastruttura (server, reti, cloud) e degli strumenti di lavoro (ad es. database). L'integrazione dei due team comporta il superamento delle barriere culturali e ciò si può ottenere mediante l'istituzione di team multifunzionali. L'integrazione deve essere efficiente e pertanto il metodo DevOps richiede il supporto della tecnologia per automatizzare dei processi che prima erano manuali. DevOps diventa quindi un principio aziendale e si inserisce in modo rilevante nel software life cycle. Questa metodologia, inoltre, integra le tecniche di Continuous Integration (CI), Continuous Delivery (CD) e Continuous Deployment:

- **Continuous Integration (CI)**: è una pratica di automatizzazione dell'integrazione delle modifiche al codice apportate da più collaboratori all'interno di un progetto software. È una best practice principale di DevOps, che consente agli sviluppatori di unire frequentemente le modifiche al codice

in un repository centrale dove poi vengono eseguiti le build e i test. Gli strumenti automatizzati vengono utilizzati per affermare la correttezza del nuovo codice prima dell'integrazione.
I test possono riguardare un componente (unit testing), un sottosistema (integration testing), il sistema (system testing) e possono comprendere analisi statiche (ad esempio per verificare la conformità con le regole aziendali di scrittura del codice). Un sistema di controllo delle versioni del codice sorgente è il punto cruciale del processo di CI ed è generalmente integrato con altri controlli come i test automatizzati della qualità del codice, gli strumenti di revisione dello stile della sintassi e molto altro. Senza CI, gli sviluppatori devono coordinare e comunicare manualmente quando stanno contribuendo con il codice al prodotto finale. Questo coordinamento va oltre i team di sviluppo, fino alle operazioni e al resto dell'organizzazione.
Infatti i benefici del CI possono essere riassunti nel seguente modo: rilevamento precoce degli errori grazie a una costante integrazione e attività di testing del codice, e migliore qualità del codice grazie alla presenza di test automatizzati.
Un esempio di funzionamento della "filosofia CI" è il seguente: gli sviluppatori effettuano il commit con le nuove modifiche al nuovo repository centrale, il sistema rileva il commit e avvia una nuova build, dopo la build vengono avviati i test automatizzati e infine gli sviluppatori ottengono un feedback in merito all'esito della build e dei test;

- **Continuous Delivery (CD):** il codice eseguibile del sistema è trasferito in uno staging environment che riproduce le caratteristiche dell'ambiente di produzione. In questa seconda parte della pipeline sono effettuati vari test (tra i quali i test di performance) per controllare la qualità del prodotto prima che venga messo in produzione. Infine si effettua il deployment nell'ambiente di produzione. L'obiettivo è automatizzare il processo di rilascio, riducendo al minimo l'intervento manuale e assicurando che il software possa essere distribuito in modo sicuro e rapido.
  I componenti principali del CD sono: Staging environment (copia dell'ambiente di produzione dove il codice viene testato. Questo ambiente riproduce le caratteristiche dell'ambiente di produzione per garantire che i test siano il più realistici possibile), Pipeline di rilascio (serie di passaggi automatizzati -build, test, deployment- che il codice deve attraversare prima di essere rilasciato in produzione) e test automatizzati.
  I test automatizzati si dividono in:
  - **test di performance**: che verificano la corretta funzionalità del sistema sotto carico, test di sicurezza per verificare assenza di vulnerabilità che potrebbero essere sfruttate come veicolo di attacco;
  - **test di regressione:** in cui le nuove funzionalità introdotte vengono verificate affinchè non introducano bug ed errori nelle funzionalità preesistenti

- La continuous deployment è la fase finale della pipeline. In questa fase, l'artefatto software viene distribuito agli utenti finali. Dopo aver superato con successo le fasi di integrazione e consegna, l'artefatto è pronto per essere distribuito automaticamente. Questo processo avviene tramite script o strumenti che trasferiscono automaticamente l'artefatto su server pubblici o attraverso altri meccanismi di distribuzione, come un app store.

10.3 Che cos'è l'automazione dei test?

L'automazione dei test indica la pratica di esaminare e convalidare automaticamente un prodotto software, ad esempio per un'applicazione Web assicurarsi che soddisfi gli standard di qualità predefiniti per lo stile del codice, la funzionalità (logica aziendale) e l'esperienza utente. L'importanza dei tool di test automation sta crescendo anche in relazione al modello DevOps, che, rispetto alle metodologie tradizionali, improntate sul classico paradigma 'waterfall', si propone di introdurre ulteriore velocizzazione e automazione nel ciclo di sviluppo e distribuzione del software.

10.3.1 Cosa automatizzare

- Casi di test che testano la funzionalità critica dell'applicazione;

- Scenari di test che richiedono l'esecuzione di test ripetuti con un set di dati di grandi dimensioni (esistono molti casi di test o flussi di applicazioni che richiedono l'esecuzione di un'azione ripetutamente);

- Test che richiedono tempo: anche i flussi di lavoro che richiedono una notevole quantità di tempo per l'esecuzione e la configurazione dovrebbero essere candidati ideali per l'automazione;
- Casi di test che devono essere eseguiti in ambiente parallelo o distribuito: alcuni casi di test richiedono l'esecuzione di operazioni simultanee sull'applicazione, ad esempio in caso di test delle prestazioni o scenari in cui è necessario controllare il comportamento dell'applicazione all'accesso simultaneo alle risorse

Prima di analizzare nel dettaglio vantaggi, pratiche e tipologie di testing è altresì importante capire quali casistiche di test non possono, o piuttosto non dovrebbero essere automatizzate:

1. Casi di test relativi all'interfaccia utente: i casi di test relativi all'interfaccia grafica dovrebbero essere lasciati al test manuale o alla convalida umana. Questo perché, anche con il minimo cambiamento nell'interfaccia utente, i casi di test fallirebbero ed è anche molto difficile creare casi di test dell'interfaccia utente affidabili su più dispositivi e risoluzioni dello schermo;
2. Casi di test relativi all'usabilità (perché viene richiesto il giudizio soggettivo);
3. Funzionalità che vengono utilizzate raramente e richiedono tempo per lo scripting: è bene automatizzare gli scenari complessi, ma investire i propri sforzi in scenari che sarebbero usati raramente non fornisce un buon ritorno sull'investimento;
4. Test esplorativi: i test esplorativi richiedono l'apprendimento immediato dell'applicazione e il test simultaneo richiedendo la necessità di adattarsi e reagire a nuove informazioni. Pertanto, non è possibile automatizzare gli scenari di test esplorativi.

10.4 Pratiche di test

Le pratiche di test prevedono in genere le seguenti fasi:

- **Test unitari:** convalidano le singole unità di codice, come una funzione, in modo che il funzionamento sia quello previsto;
- **Test di integrazione:** garantiscono che diverse parti di codice (e quindi i componenti) possano interagire senza conseguenze involontarie;
- **Test end-to-end:** verificano che l'applicazione soddisfi le aspettative dell'utente. Infatti l'obiettivo è simulare scenari reali per garantire che tutte le parti del sistema funzionino insieme come previsto;
- **Test esplorativi:** adottano un approccio non strutturato per esaminare numerose aree di un'applicazione dal punto di vista dell'utente, per far emergere problemi funzionali o visivi.

Man mano che le pratiche DevOps maturano nell'ambito delle organizzazioni, la necessità di automazione dei test durante l'intero ciclo di vita è importante per sfruttare i principali vantaggi di DevOps: la capacità di compilare, testare e rilasciare in modo più rapido e affidabile, semplificare le risposte agli imprevisti e migliorare la collaborazione e la comunicazione tra i team. L'adozione di test automatizzati offre i seguenti vantaggi DevOps:

- **Migliore collaborazione tra i team:** la responsabilità condivisa per la qualità migliora la collaborazione tra i membri del team;
- **Affidabilità:** migliora l'affidabilità dei rilasci aumentando la copertura con l'automazione dei test. Durante la produzione, i problemi dovrebbero verificarsi raramente, non essere la norma;
- **Scalabilità:** produci risultati di qualità coerenti con rischi ridotti distribuendo lo sviluppo tra più team di piccole dimensioni che operano in modo autosufficiente;
- **Sicurezza:** procedi rapidamente senza compromettere la sicurezza e la conformità sfruttando policy di conformità automatizzate, controlli granulari e tecniche di gestione della configurazione;
- **Maggiore soddisfazione dei clienti:** la maggiore affidabilità e le risposte rapide al feedback degli utenti aumentano la soddisfazione degli utenti e determinano un maggior numero di segnalazioni di prodotti.

10.5 Tipologie di test:

A seconda delle strategie adottate i test possono essere divisi nelle seguenti categorie:

- **test manuali**, in cui uno specialista esegue i casi di test, senza ricorrere a tool o script, ma definendo gli scenari, identificando i casi limite, preparando i set di dati e attivando gli input o le azioni necessarie. In questi test lo specialista ha sotto controllo la situazione e può facilmente trovare visual bug o problemi in ambito User Experience (UX). Inoltre, non ricorrendo a tool, i test potrebbero risultare più economici. Dall'altro lato questi risultano essere più lenti e soggetti a errori umani;
- **test automatizzati**, nei quali si ricorre a script e tool per preparare i dati ed eseguire i passaggi necessari per verificare lo scenario in modo automatico; rendono agevoli diversi tipi di test (anche complessi) e consentono la verifica rapida di non regressione nel caso di introduzione di nuove funzionalità o modifiche. Di contro però non sono adatti per trovare i visual bug e altri aspetti legati alla UX, particolarmente critici per app e siti web;
- **Il crowd testing**, che si basano su una community di tester qualificati, ingaggiati sulla base di specifiche competenze o del target del software da verificare e aventi quindi le caratteristiche necessarie per mettersi nei panni di un utilizzatore reale. Fra i principali vantaggi, va sicuramente segnalata la rapidità della risposta e i costi contenuti rispetto all'utilizzo di risorse dedicate o all'uso di strumenti automatici costosi.

Per concludere: la rivoluzione dell'organizzazione di un progetto in termini di testing ha permesso un cambio radicale, passando da un modello waterfall (a cascata) al V-Model. Nel modello waterfall si cercava di definire i requisiti nelle prime fasi del ciclo di vita del progetto per poi preoccuparsi della progettazione del software. Il grave problema che si verificava in questo modello era l'impatto dovuto a modifiche dei requisiti o a test realizzati esclusivamente durante le fasi finali. Pertanto, questo era un modello molto veloce ma poco flessibile.

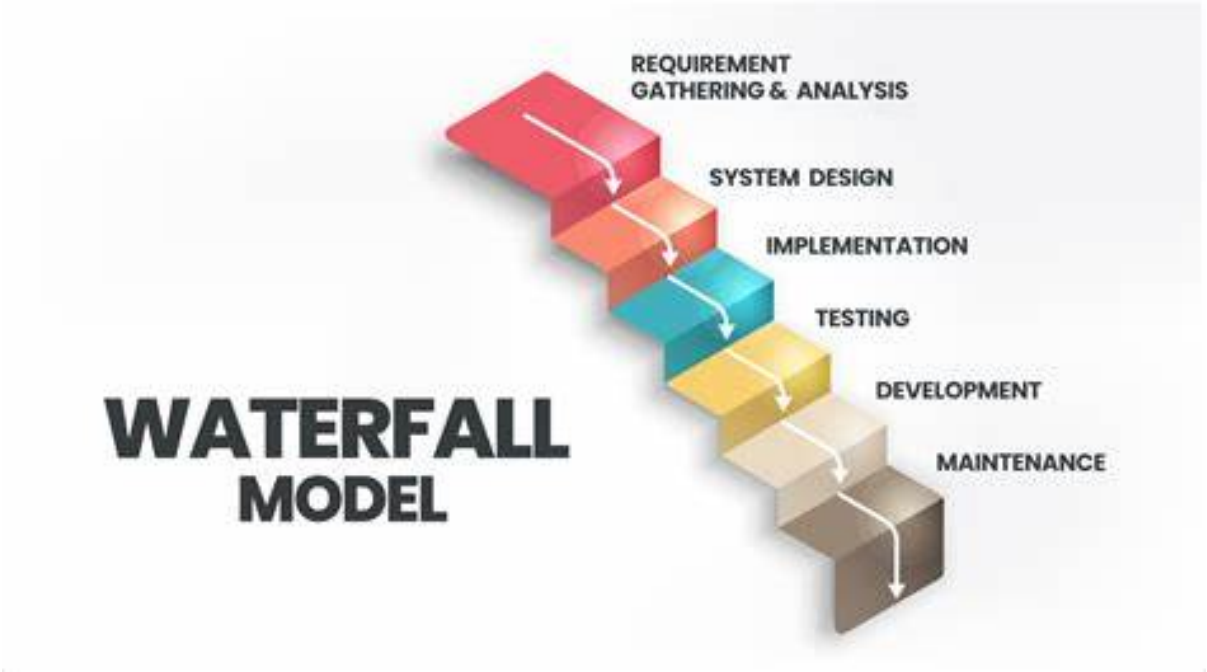


Figure 27: Modello Waterfall

Il V-Model invece prevede in ogni fase di design (Requirement, System, Architecture e Module) dei test per avere sempre sotto controllo cosa sta accadendo fornendo flessibilità in caso di modifiche strutturali o problematiche riscontrate durante la fase di development. Pertanto rispetto al modello waterfall risulta essere molto più lento ma molto più flessibile e sicuro.

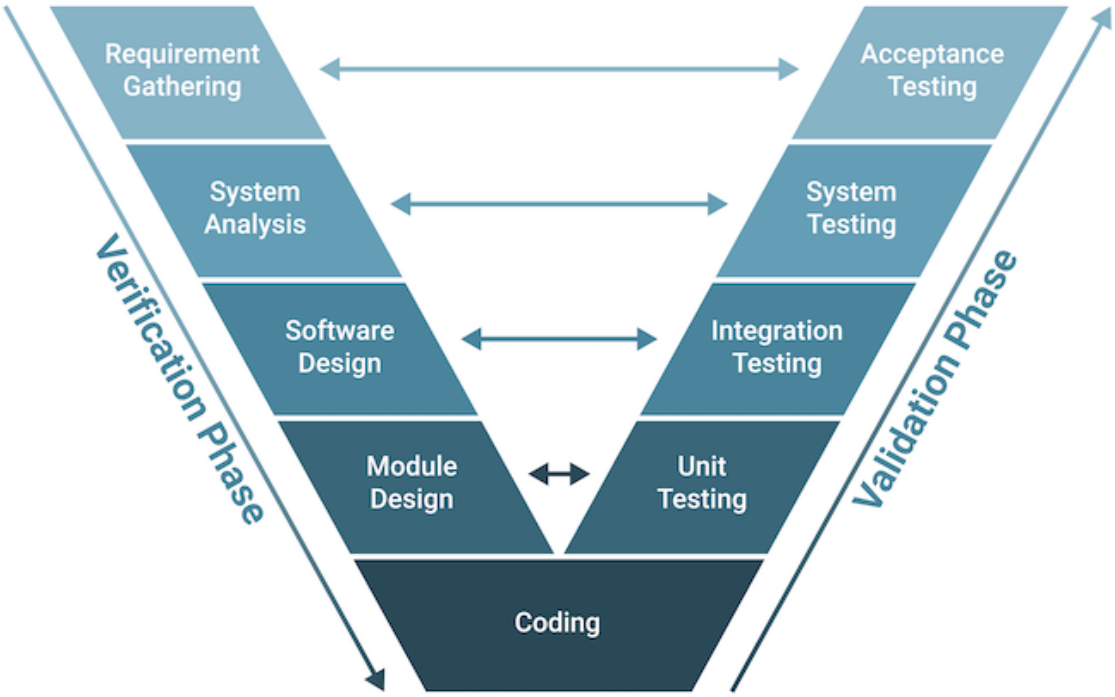


Figure 28: V-model

10.6 JMeter

Terminata la spiegazione di automation testing passiamo ora a descrivere JMeter, uno strumento di test con il quale è stato possibile verificare la correttezza della piattaforma aziendale realizzata per implementare l'Open Banking per conto di Pay.UK.
JMeter è uno strumento di test open source, sviluppato da Apache Software Foundation. Si tratta di un'applicazione Java pura, che può essere utilizzata per misurare le prestazioni di applicazioni, servizi software diversi e prodotti sia su risorse statiche che dinamiche. Inizialmente JMeter è stato progettato per testare applicazioni Web, ma in seguito si è espanso per testare altre funzioni come test funzionali, test delle prestazioni, test di regressione, stress test, server di database testati basati su varie tecnologie.
Come detto in precedenza JMeter è uno strumento di test delle prestazioni che funziona a livello di protocollo: sembra un browser ma non esegue tutte le azioni supportate dai browser, infatti non esegue il rendering di pagine HTML né esegue JavaScript in una pagina HTML. Di seguito viene riportato il flusso di operazioni che svolge il tool:

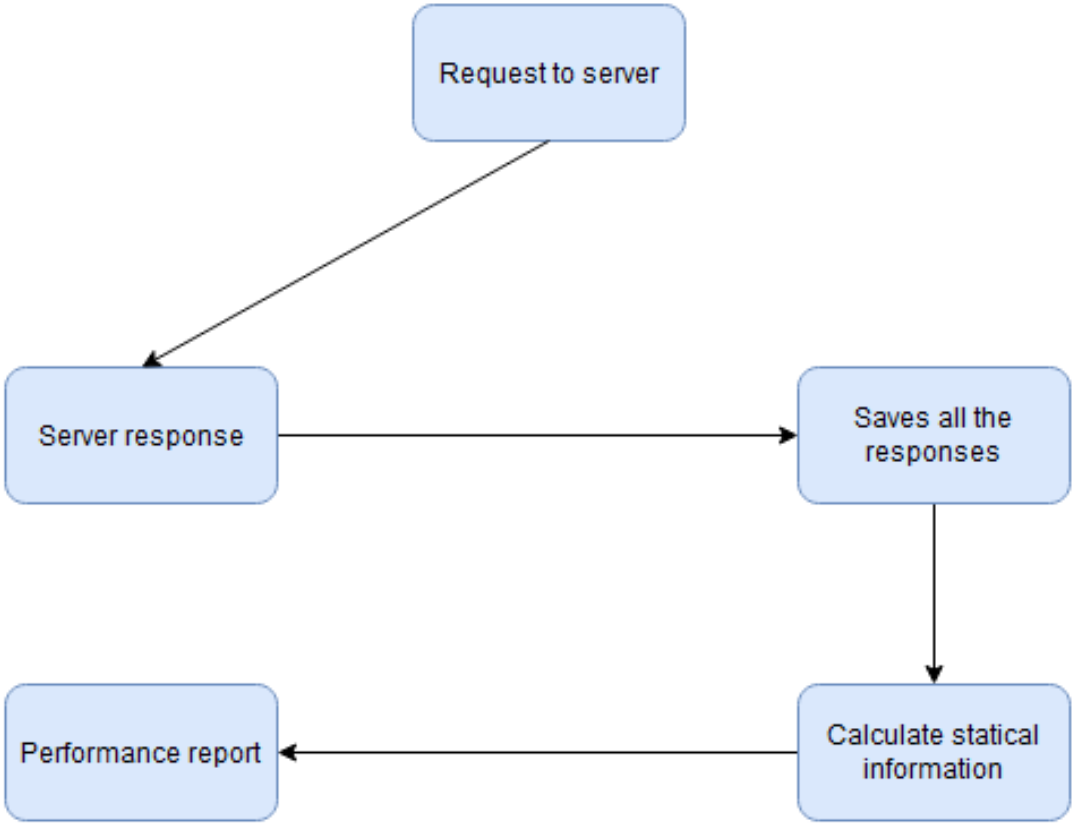


Figure 29: Flusso Jmeter

- Il gruppo di utenti invierà la richiesta al server di destinazione;
- Il server risponderà alla richiesta dell'utente;
- Successivamente, salverà tutte le risposte per la richiesta fornita;
- Ora restituisce le informazioni statistiche del server di destinazione per mostrare le prestazioni di un'applicazione;
- Nell'ultimo passaggio, verrà visualizzato il rapporto sulle prestazioni di un'applicazione

10.7 Cosa puoi fare con JMeter?

Puoi testare varie applicazioni usando JMeter. Di seguito sono riportate alcune applicazioni che possono essere testate da JMeter:

- Siti Web - HTTP e HTTPS (Java, NodeJS, PHP, ASP.NET, ecc.);
- Servizi Web - REST e SOAP;
- Server di database;
- Server FTP;
- Server LDAP;
- Server di posta - SMTP, POP3, IMAP;
- Script di shell;
- Server TCP;
- Middleware orientato ai messaggi tramite JMS

Vantaggi:

- **Licenza open source:** è uno strumento di test open source, in cui lo sviluppatore può utilizzare liberamente il codice sorgente per lo sviluppo;
- **GUI:** è molto facile da usare in quanto ha un'interfaccia grafica intuitiva e puoi familiarizzare con esso in un breve periodo di tempo;
- **Indipendente dalla piattaforma:** JMeter è un'applicazione desktop pura, quindi può essere eseguita su diversi tipi di piattaforme come Windows, Linux, ecc;
- **Multithreading Framework:** dà il permesso per il campionamento sincrono e parallelo di varie funzioni usando diversi gruppi di thread;
- **Visualizza i risultati del test:** utilizzando lo strumento JMeter, è possibile visualizzare i risultati del test in vari formati come grafico, tabella, albero e file di registro;
- **Estensibile:** JMeter consente di estendere i propri test utilizzando plug-in di visualizzazione;
- **Supporta test multipli:** JMeter supporta numerosi processi di test come test di carico, test distribuito e test funzionali;
- **Test degli script:** è possibile incorporare Bean Shell e Selenium con test automatizzati;

Svantaggi:

- **Utilizzo della memoria:** JMeter riproduce carichi pesanti e genera le informazioni del test. Questo porta all'utilizzo di molta memoria e produce una grande quantità di memoria dal carico pesante;
- **Applicazione Web:** è possibile testare le applicazioni Web utilizzando JMeter, ma non può essere utilizzata per l'applicazione desktop di prova;
- **Nessun utilizzo di JavaScript:** non è possibile utilizzare il linguaggio JavaScript nell'applicazione; perché sappiamo che JMeter è uno strumento di test, non un browser;
- **Monitoraggio dei test limitato:** rispetto ad altri strumenti di test, JMeter offre una capacità di monitoraggio dei test limitata

11 GLOSSARIO

FCA Register: è un registro pubblico tenuto dalla Financial Conduct Authority (FCA) del Regno Unito. Elenco di tutte le imprese, individui e altre entità che la FCA supervisiona o è in qualche modo collegata.
ASPSP (Account Servicing Payment Service Providers): fornitori di servizi di pagamento che gestiscono un conto, come le banche
TPP (Third Party Providers): fornitori di servizi di pagamento di terze parti, come le fintech che forniscono servizi finanziari innovativi
OBIE (Open Banking Implementation Entity): è l'organizzazione che è stata creata per definire e sviluppare gli standard necessari per implementare l'Open Banking nel Regno Unito. È stata istituita dalla Competition and Markets Authority (CMA) del Regno Unito nel 2016.
PSU (Payment Servicies User): end-user che sfrutta il servizio, ad esempio per effettuare un pagamento
PISP: Payment Initiation Service Provider.
AISP: Account Information Service Provider.
CISP: Card Issuer Service Provider.
PIISP: Payment Instrument Issuing Service Provider.
CPPA (Consumer Privacy Protection Act): Legge sulla privacy dei dati del Canada aggiornata nel 2020
GDPR (General Data Protection Regulation): Regolamento dell'Unione Europea in merito al trattamento dei dati personali e di privacy del 2016
eIDAS (electronic IDentification, Authentication and trust Services): Regolamento europeo per l'identificazione elettronica e servizi fiduciari per le transazioni elettroniche nel mercato interno, regolamento UE n. 910/2014
QWACs (Qualified Website Authentication Certificates): certificati utilizzati per autenticare i siti web e garantire una connessione sicura tra il browser dell'utente e il sito web
QSeals (Qualified Electronic Seal Certificates): certificati utilizzati per sigillare elettronicamente i dati, garantendo l'integrità e l'autenticità dei dati
QTSP (Qualified Trust Service Provider): fornitore di servizi fiduciari ai quali è stato riconosciuto uno status di ente qualificato da un organo di governo e vigilanza. Questo fornitore ha ricevuto l'autorizzazione per l'erogazione dei propri servizi in ottica eIDAS
SSA (Software Statement Assertion): una dichiarazione software (o "Software Statement") nell'Open Banking è una dichiarazione che descrive le caratteristiche e le capacità del software di un Fornitore di Servizi di Terze Parti. Questa dichiarazione è un elemento chiave per garantire la sicurezza e l'autenticità delle comunicazioni nell'Open Banking. Nell'Open Banking, un TPP crea una dichiarazione software che include informazioni come l'identità del TPP, le funzionalità del software, i certificati di sicurezza e altre informazioni rilevanti. Questa dichiarazione viene poi registrata in una directory centralizzata (la Open Banking Directory).
PTC (Primary Technical Contacts): individui, designati dal TPP, che hanno la responsabilità di gestire aspetti tecnici specifici del servizio Open Banking del TPP. Questo può includere compiti come la gestione dei certificati di sicurezza, la configurazione delle API, la risoluzione dei problemi tecnici e la comunicazione con l'Open Banking Directory e gli ASPSP. Ad esempio, un Primary Technical Contact potrebbe essere responsabile della creazione e dell'invio di una "Certificate Signing Request" (CSR) all'Open Banking Directory, come parte del processo di registrazione del software del TPP. Potrebbero anche essere responsabili dell'aggiornamento dei certificati quando necessario e della risoluzione di eventuali problemi tecnici che potrebbero sorgere durante l'utilizzo delle API Open Banking. MTLS (Mutual Transport Layer Security): TLS ma con autenticazione del client obbligatoria OAuth (Open Authorization): standard aperto che permette a un utente di autorizzare l'accesso ai propri dati da parte di un'applicazione di terze parti, senza condividere la password del proprio account. Quando un servizio Web si connette a un altro, viene utilizzato il protocollo OAuth per garantire un'interazione sicura senza richiedere agli utenti di condividere le proprie password. Vedere la RFC7591 per approfondire OAuth
NCA (National Competent Authority): le entità che desiderano fornire servizi di pagamento come fornitore di servizi di pagamento devono prima fare domanda alla NCA per ottenere l'autorizzazione
PS256: algoritmo che utilizza la crittografia RSA con una chiave di dimensioni 2048 bit e applica la firma RSASSA-PSS;
ES256: algoritmo che utilizza la crittografia a curva ellittica (ECC) con la curva P-256
RSASSA-PSS (RSA Signature Scheme with Appendix - Probabilistic Signature Scheme) :

schema di firma basato sull'assunzione RSA
SCA (Strong Customer Authentication): processo di autenticazione che richiede l'uso di almeno due dei seguenti tre requisiti: informazioni note solo al cliente (come un PIN), un oggetto posseduto solo dal cliente (ad esempio, uno smartphone), caratteristiche uniche del cliente (come le impronte digitali o il volto)
VRP (Variable Recurring Payments):tipo di pagamento ricorrente che consente alle aziende di raccogliere pagamenti direttamente dai conti bancari dei clienti, senza la necessità di condividere informazioni sensibili come i dettagli della carta di credito
FAPI-RW (Financial-grade API Read Write): standard di sicurezza per OAuth 2.0 e OpenID Connect che fornisce linee guida specifiche per la sicurezza e l'interoperabilità. FAPI-RW è utilizzato per autenticare sia gli utenti dei servizi di pagamento (PSU) che i fornitori di terze parti (TPP)
CMA, Competition and Markets Authority: è l'autorità principale del Regno Unito responsabile della regolamentazione della concorrenza e della protezione dei consumatori1. È un dipartimento governativo indipendente, non ministeriale, che ha il compito di promuovere mercati competitivi e affrontare comportamenti scorretti
EBA, European Banking Authority: agenzia regolatoria che ha scopo di regolamentazione e supervisione nel settore bancario europeo

References

[] *AISP, PISP.* URL: https://www.compliancejournal.it/servizi-di-pagamento-tpp-aisp-pisp/.

[] *AISP, PISP, CISP.* URL: https://vitolavecchia.altervista.org/definizione-e-differenza-tra-pisp-aisp-e-cisp-nella-psd2-bancaria/.

[] *Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants.* URL: https://www.rfc-editor.org/rfc/rfc7521.

[] *Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants.* URL: https://www.rfc-editor.org/rfc/rfc7523.

[] *authenticate with private key jwt.* URL: https://auth0.com/docs/get-started/authentication-and-authorization-flow/authenticate-with-private-key-jwt.

[] *Certificati psd2.* URL: https://www.infocert.it/grandi-aziende/certificati-psd2.

[] *Che cos'è JMeter? - Come Funziona - Funzionalità e carriera - Vantaggi di Jmeter (education-wiki.com).* URL: https://it.education-wiki.com/9105507-what-is-jmeter.

[] *Cos'è protocollo OAuth.* URL: https://www.proofpoint.com/it/threat-reference/oauth.

[] *Definizione VRP.* URL: https://www.volt.io/vault/payment-innovation/variable-recurring-payments-what-are-they-and-how-do-they-work/.

[] *Directory Technical.* URL: https://openbanking.atlassian.net/wiki/spaces/DZ/pages/1150124033/Directory+2.0+Technical+Overview+v1.3.

[] *Directory Technical.* URL: https://openbanking.atlassian.net/wiki/spaces/DZ/pages/1150124033/Directory+2.0+Technical+Overview+v1.7.

[] *Direttiva PSD2: cosa cambia e quando entra in vigore.* URL: https://www.lexdo.it/blog/direttiva-psd2-cosa-cambia-e-quando-entra-in-vigore/.

[] *EISCD.* URL: https://www.vocalink.com/tools/extended-industry-sort-code-directory/.

[] *EISCD PayUK.* URL: https://www.wearepay.uk/what-we-do/industry-services/extended-industry-sort-code-directory/.

[] *Examples of Protecting Content Using JSON Object Signing and Encryption (JOSE).* URL: https://www.rfc-editor.org/rfc/rfc7520.

[] *Firma digitale.* URL: https://www.datalog.it/definizioni/firma-digitale/.

[] *Firma digitale dal punto di vista tecnico.* URL: https://www.datalog.it/firma-digitale-come-funziona-dal-punto-di-vista-tecnico/.

[] *Firma digitale funzionamento*. URL: https://vitolavecchia.altervista.org/firma-digitale-caratteristiche-e-funzionamento-in-informatica/.

[] *JSON Web Algorithms (JWA)*. URL: https://www.rfc-editor.org/rfc/rfc7518.

[] *JSON Web Encryption (JWE)*. URL: https://www.rfc-editor.org/rfc/rfc7516.

[] *JSON Web Key (JWK)*. URL: https://www.rfc-editor.org/rfc/rfc7517.

[] *JSON Web Signature (JWS)*. URL: https://www.rfc-editor.org/rfc/rfc7515.

[] *JSON Web Signautre (JWS) Unencoded Payload Option*. URL: https://www.rfc-editor.org/rfc/rfc7797.

[] *JSON web token*. URL: https://auth0.com/docs/secure/tokens/json-web-tokens/json-web-key-sets.

[] *JSON web token*. URL: https://openbankinguk.github.io/dcr-docs-pub/v3.3/dynamic-client-registration.html.

[] *JSON Web Token (JWT)*. URL: https://www.rfc-editor.org/rfc/rfc7519.

[] *JSON Web Token Best Current Practices*. URL: https://www.rfc-editor.org/rfc/rfc8725.

[] *Millions of customers benefit as Open Banking reaches milestone*. URL: https://www.gov.uk/government/news/millions-of-customers-benefit-as-open-banking-reaches-milestone.

[] *Modello a V*. URL: https://builtin.com/software-engineering-perspectives/v-model.

[] *OAuth 2.0 Dynamic Client Registration Protocol*. URL: https://www.rfc-editor.org/rfc/rfc7591.

[] *OAuth2*. URL: https://goteleport.com/learn/what-is-oauth-2/.

[] *Open Banking*. URL: https://it.wikipedia.org/wiki/Open_banking.

[] *Open Banking Direcotry eIDAS Client Registration*. URL: https://openbanking.atlassian.net/wiki/spaces/DZ/pages/2659385345/Open+Banking+Directory+Usage+-+eIDAS+release+Production+-+v2.3.

[] *Open Banking Dynamic Client Registration*. URL: https://openbanking.atlassian.net/wiki/spaces/DZ/pages/36667724/OpenBanking+OpenID+Dynamic+Client+Registration+Specification+-+v1.0.0-rc2.

[] *Open Banking Security Profile*. URL: https://openbanking.atlassian.net/wiki/spaces/DZ/pages/83919096/Open+Banking+Security+Profile+-+Implementer+s+Draft+v1.1.2.

[] *Open Banking Standards in Europe*. URL: https://thebanks.eu/articles/open-banking-standards-in-Europe.

[] *OpenBanking for oauth developers*. URL: https://www.identityserver.com/articles/open-banking-for-oauth-developers.

[] *Payment Services Directive 2 (PSD2)*. URL: https://eba.europa.eu/regulation-and-policy/single-rulebook/interactive-single-rulebook/14575.

[] *PSD2 E OPEN BANKING: NUOVI MODELLI DI BUSINESS E RISCHI EMERGENTI*. URL: https://www.bancaditalia.it/compiti/vigilanza/analisi-sistema/approfondimenti-banche-int/2021-PSD2-Open-Banking.pdf?pk_campaign=EmailAlertBdi&pk_kwd=it.

[] *PSD2 e PIISP*. URL: https://docs.solarisgroup.com/guides/compliance/psd2-sca/.

[] *Psd2 Eidas*. URL: https://eadtrust.eu/en/psd2-eidas-test-certificates-qwac-and-qseal/.

[] *PSD3: What you need to know*. URL: https://www.adyen.com/knowledge-hub/psd3.

[] *QWac e QSeals*. URL: https://www.aruba.it/magazine/enterprise/qwac-e-qsealc-banche-e-soggetti-intermediari-sulla-sfida-della-sicurezza.aspx.

[] *test automation*. URL: https://www.atlassian.com/it/devops/devops-tools/test-automation.

[] *test automation*. URL: https://www.zerounoweb.it/software/test-automation-i-vantaggi-dei-nuovi-tool-per-garantire-la-qualita-del-software/.

[] *Test software.* URL: https://blog.unguess.io/it/test-del-software-orientarsi-tra-automatizzati-manuali-e-crowd-testing#:~:text=Il%20test%20automatizzato%2C%20generalmente%20implementato%20da%20un%20team,molto%20pi%C3%B9%20rapido%20del%20test%20del%20software%20manuale..

[] *The JavaScript Object Notation (JSON) Data Interchange Format.* URL: https://www.rfc-editor.org/rfc/rfc7159.

[] *The OAuth 2.0 Authorization Framework.* URL: https://www.rfc-editor.org/rfc/rfc6749.

[] *The OAuth 2.0 Authorization Framework: Bearer Token Usage.* URL: https://www.rfc-editor.org/rfc/rfc6750.

[] *The OBIE publishes Open Banking Standard version 3.1.10 and reaches Roadmap milestone.* URL: https://www.openbanking.org.uk/news/the-obie-publishes-open-banking-standard-version-3-1-10-and-reaches-roadmap-milestone/.

[] *The Payment Services Regulations 2017.* URL: https://www.legislation.gov.uk/uksi/2017/752/note/made.

[] *The revised Payment Services Directive (PSD2) and the transition to stronger payments security.* URL: https://www.ecb.europa.eu/press/intro/mip-online/2018/html/1803_revisedpsd.en.html.

[] *Tipi di atti legislativi.* URL: https://european-union.europa.eu/institutions-law-budget/law/types-legislation_it.

[] *Tpp client authentication.* URL: https://www.pingidentity.com/en/resources/blog/post/open-banking-tpp-client-authentication-with-ping-identity.html.

[] *Understanding Strong Customer Authentication PSD2.* URL: https://www.adyen.com/knowledge-hub/psd2-understanding-strong-customer-authentication.

[] *Understanding the difference between Open Banking and PSD2.* URL: https://www.bottomline.com/resources/blog/understanding-difference-between-open-banking-psd2.

[] *What Are The Differences Between Open Banking And PSD2?* URL: https://thinksmobility.com/insights/blog/tech-trends/what-are-the-differences-between-open-banking-and-psd2/.

[] *Ws02 authorization.* URL: https://ob.docs.wso2.com/en/latest/learn/consent-authorization-intro/.

[Bal24] Gabriella Balestra. *Lezioni universitarie della Balestra.* 2024.

[Bru23] Giorgio Bruno. *Lezioni universitarie di Bruno.* 2023.