POR-FESR EMILIA ROMAGNA 2014-2020

Asse 1 - Ricerca e innovazione

Azione 1.2.2 - Supporto alla realizzazione di progetti complessi di attività di ricerca e sviluppo su poche aree tematiche di rilievo e all'applicazione di soluzioni tecnologiche funzionali alla realizzazione della strategia di S3

Bando 2018

Progetti di ricerca industriale strategica rivolti agli ambiti prioritari della Strategia di Specializzazione Intelligente

Sistemi interoperabili ed efficienti per la gestione sicura di filiere industriali

Deliverable D4.3:
SmartChain: Analisi, validazione e lezioni apprese

Data di consegna prevista:	31 Gennaio 2022
Autori:	MECHLAV, CIRI ICT, CRIS, CROSSTEC, INFN-TTLab
Versione:	1

Indice

1. Introduzione

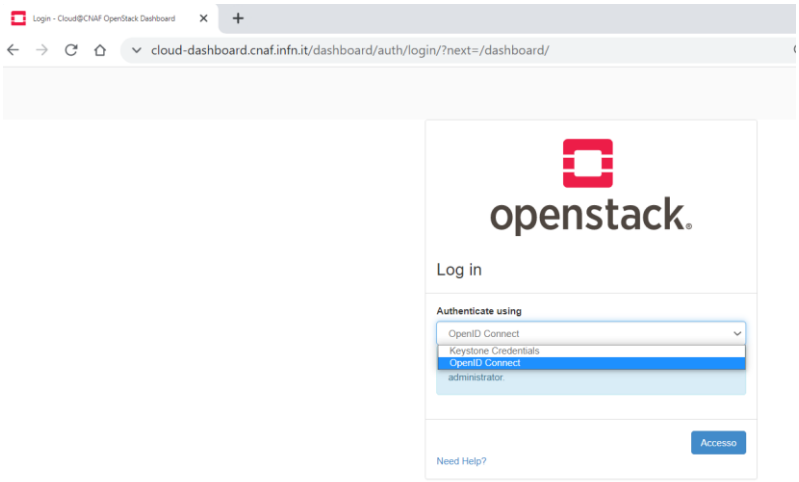
In questo documento si descrivono i test, con relativo calcolo delle prestazioni e considerazioni sul sistema, svolti nell'ambito della Fase 4 "Sperimentazione e verifiche dei risultati". In particolare, l'attuale allegato tecnico prende in considerazione le informazioni evidenziate nel documento che descrive l'implementazione della soluzione (Deliverable D4.2) e, in aggiunta, risulta strettamente correlato al documento D3.2 "Progetto di dettaglio di SmartChain per filiere", da cui attinge le principali funzionalità necessarie per effettuare i test e trarre le dovute considerazioni. L'obiettivo che si vuole raggiungere consiste nel descrivere i risultati ottenuti mediante valutazione sperimentale, verificando e testando l'implementazione dei sistemi descritti in Fase 3 e di confrontarli rispetto ai requisiti individuati nella Fase 1 "Analisi delle diverse filiere industriali".

Il documento assume la seguente struttura: nella Sezione 2 vengono presentati i criteri di test impiegati nelle attività svolte per il caso d'uso Bianco Accessori; nella Sezione 3 si descrivono i risultati dell'attività di test e considerazioni per il caso d'uso Bianco Accessori; nella Sezione 4 vengono presentati i criteri di test impiegati nelle attività svolte per il caso d'uso Carpigiani; nella Sezione 5 si presentano le considerazioni e i risultati ottenuti dell'attività di test per il caso d'uso Carpigiani.

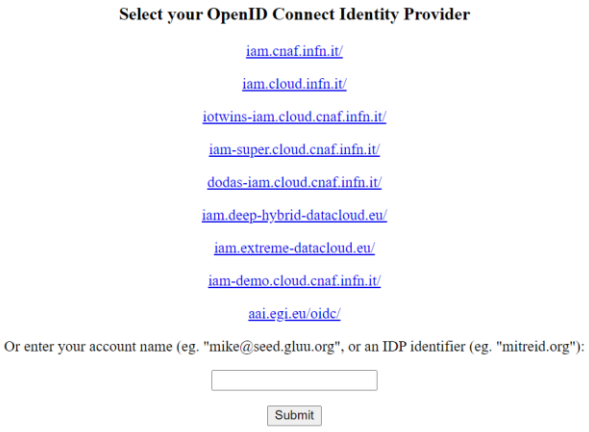
2. Ambiente di test per il caso d'uso basato su Ethereum

2.1. Infrastruttura di test

L'ambiente di test è ospitato sulla piattaforma INFN Cloud che rende disponibili risorse di calcolo e storage distribuite su tutto il territorio nazionale. Il testbed SmartChain è stato creato nella forma di tenant sulla IaaS INFN Cloud, seguendo le istruzioni presenti nella documentazione online[1], che riassumeremo brevemente in questo documento. L'accesso all'ambiente di test avviene via dashboard OpenStack selezionando l'opzione di autenticazione "OpenID Connect".

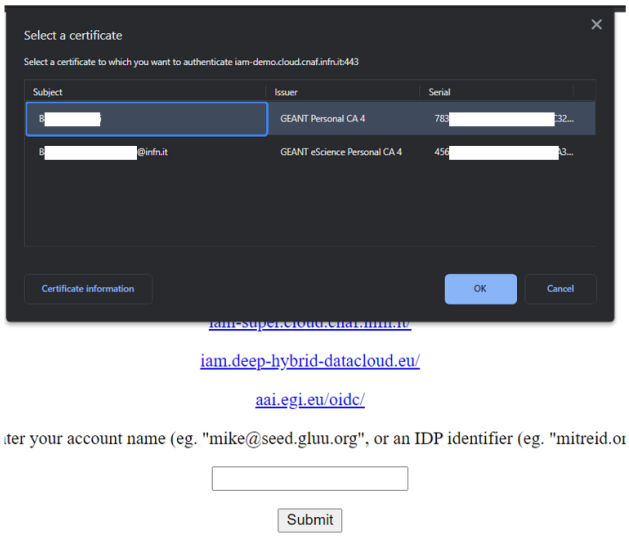


Viene presentata una schermata con una serie di opzioni di Identity Provider, selezionare "iam-demo.cloud.cnaf.infn.it"

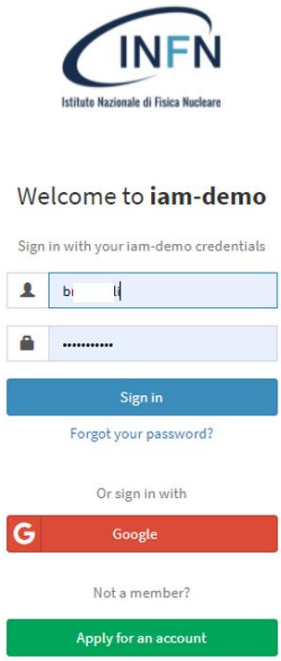


[1] https://guides.cloud.infn.it/docs/users-guides/en/latest/users_guides/getting_started.html

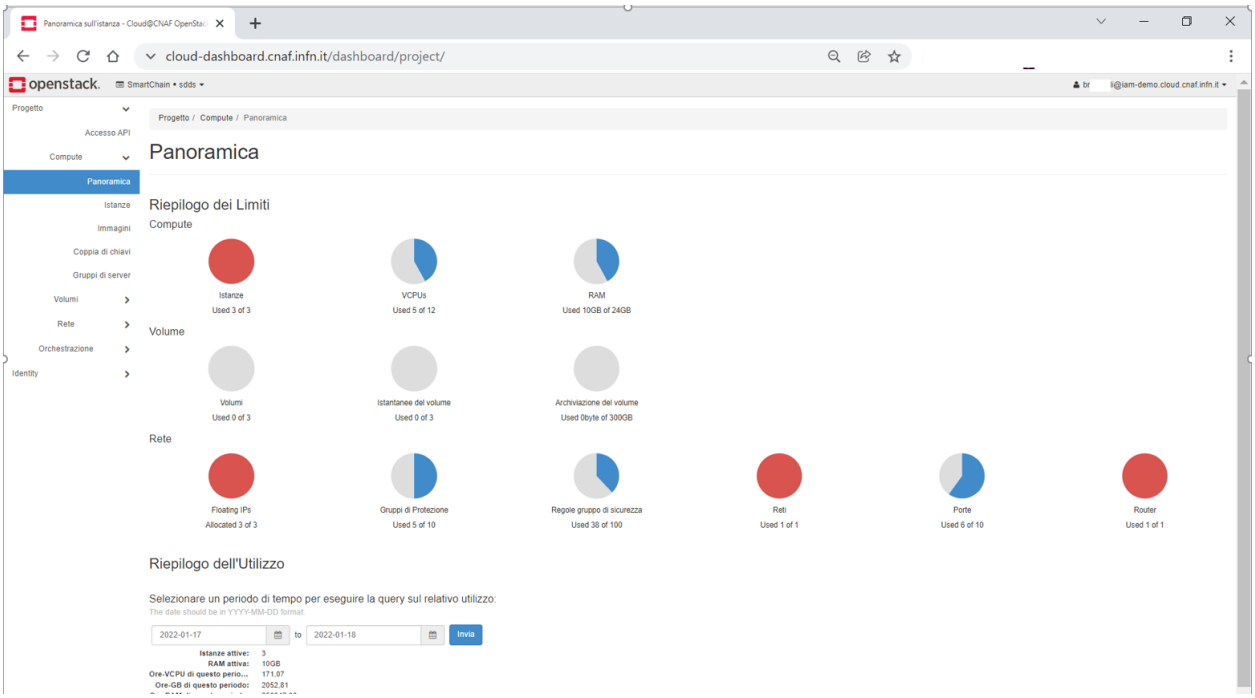
L’autenticazione puo’ avvenire attraverso certificati X.509, credenziali username/password, oppure utilizzando identity provider esterni compatibili con OpenID (ad esempio Google).



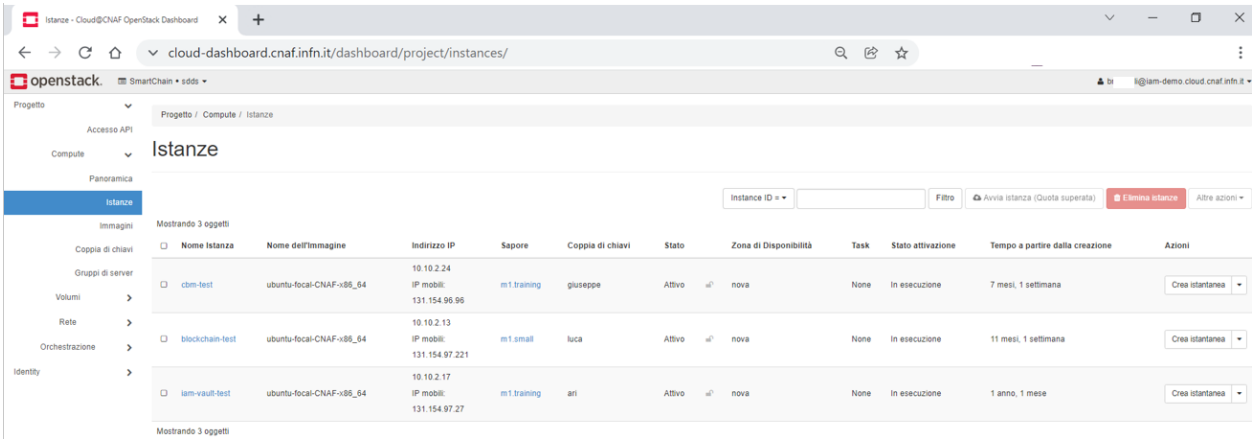
In questo esempio, per l'autenticazione viene utilizzato un certificato X.509 memorizzato nel browser dell’utente (alcuni dati personali sono stati oscurati).



In questo esempio l’utente si autentica con username/password (alcuni dati personali sono stati oscurati).



Una volta autenticati, si accede alla dashboard per la visualizzazione e la gestione delle risorse assegnate al tenant. In particolare, è possibile creare macchine virtuali (VM) a partire da diverse immagini di sistema operativo, assegnare alle VM risorse CPU, RAM, Volumi storage, IP pubblici (floating IP), sottoreti, router virtuali e gruppi di sicurezza (configurazioni del firewall locale).



Per il testbed smartchain sono state create tre VM (cbm-test, blockchain-test, iam-vault-test), accessibli da remoto via SSH, con una coppia di chiavi asimmetriche per ogni utente.

2.2. Gestione delle chiavi crittografiche e compliance

Come evidenziato nel deliverable D1 al paragrafo 4.1.4, privacy e sicurezza costituiscono requisiti non funzionali di fondamentale importanza per l'azienda. L'elemento architetturale adibito alla gestione delle chiavi crittografiche della piattaforma (Key Management System

KMS) gioca un un ruolo centrale nel garantire la sicurezza, in quanto custode di tutte le chiavi utilizzate per cifrare e decifrare le informazioni successivamente memorizzate sulla block chain.

Al fine di verificare la funzionalita' di questo elemento si è fatto riferimento ai requisiti identificati nello standard ISO/IEC 27001, uno dei piu' diffusi e riconosciuti standard di sicurezza delle informazioni, si è proceduto a verificare che il componente KMS da noi scelto fosse in linea con questi standard.

Tra le funzionalità che rispondono ai controlli principali ISO/IEC 27001 elenchiamo:

- possibilità di garantire *accountability* del sistema, attraverso la generazione di audit log che tracciano l'origine e l'utente che ha effettuato ogni singola transazione;
- confidentialità dei dati memorizzati su blockchain: ogni dato che lascia il KMS viene automaticamente cifrato con algoritmo 256-bit Advanced Encryption Standard (AES) in modalità Galois Counter Mode (GCM) con *nonce* di 96-bit. Il nonce viene generato in modo casuale per ogni oggetto cifrato;
- robustezza rispetto ad attacchi di tampering: il KMS è in grado di rilevare i tentativi di modifica dei dati da esso gestiti ed in tal caso interrompe l'elaborazione della transazione;
- possibilità di forzare l'applicazione di policy di accesso ai segreti crittografici durante tutto il ciclo di vita del segreto (generazione, memorizzazione, archiviazione, recupero, distribuzione, invalidazione e distruzione);
- protezione delle chiavi crittografiche da modifiche non autorizzate e perdita accidentale.

2.3. Interazione con il Confidential Blockchain Manager e Scalabilità

La piattaforma smartchain è stata progettata e sviluppata analizzando le specifiche esigenze del settore tessile in stretta collaborazione con le aziende. Ciò ci ha consentito di dare voce ad esigenze reali del settore e ha permesso di sviluppare così un progetto che va a colmare vuoti tecnologici importanti relativi al concetto di notarizzazione delle commesse lungo tutta la filiera. Il risultato è stata un'architettura software che risulta essere immediatamante utilizzabile in contesti reali poichè costruita ad hoc.

La piattaforma smartchain utilizza e astrae quattro concetti del mondo aziendale ossia: impresa, fornitore registrato, fornitore non registrato e certificato. Le azioni che possono essere fatte su queste quattro risorse vengono diversificate dal relativo metodo http: GET, POST, PUT, e DELETE e variando la parte finale dell'URI.

Le azioni eseguibili sono tipicamente 5 per ciascuna risorsa ovvero la registrazione di una risorsa, la visualizzazione dei dettagli, la visualizzazione di tutte le risorse di un certo tipo, la modifica dei dati inerenti alla risorsa e la cancellazione della risorsa. Ciò permette di ricreare un

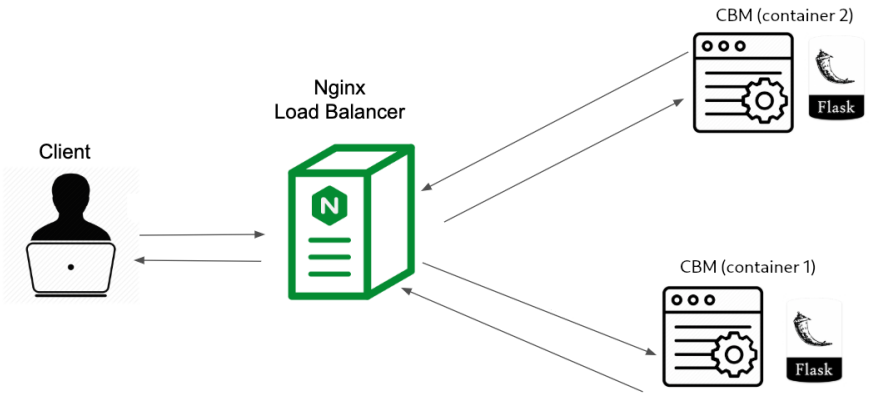
modello virtuale della filiera su cui possono essere eseguite le principali interazioni tra imprese e fornitori ed essere tracciate in maniera irripudiabile grazie all'utilizzo della blockchain.

Un altro punto di forza della piattaforma smartchain è la facilita di deployment grazie alla sua struttura a container (come descritto nel dettaglio nel deliverable D4.2). Per ogni componente vengono infatti generati attaverso la build uno o più distinti container Docker.

In particolare, il backend è costituito da tre container, il primo si occupa dell'autorizzazione e della criptazione ed è basato su Hashicorp Vault, il secondo su cui gira la blockchain Ethereum ed il relativo smart contract, il terzo su cui gira il componente CBM che intercetta tutte le richieste provenienti dal frontend.

I container sono diventati negli ultimi anni lo standard nel packaging e nel deploy di codice applicativo. L'architettura a container abilita a migliorare le applicazioni in termini di modularità e scalabilità ed inoltre incrementa la riusabilità.

Il container CBM in caso di un numero elevato di richieste, ad esempio, può scalare in maniera semplice e veloce lanciando ulteriori copie del container stesso. Questa permette di migliorare notevolmente le prestazioni in caso appunto di carico elevato. Un semplice esempio di architettura è mostrato nella figura sottostante.



Essa introduce l'utilizzo di un load balancer che intercetta tutte le chiamate verso il CBM e redirige le richieste seguendo una sequenza definita da uno specifico algoritmo come, ad esempio, il round-robin che a rotazione smista le richieste sui diversi container.

3. Risultati dell'attività di test e considerazioni per il caso d'uso Bianco Accessori

Il caso d'uso Bianco Accessori si è mostrato un esempio molto interessante della dualità che contraddistingue in maniera intrinseca la tracciabilità e la trasparenza della supply chain in contesto tessile. Se da una parte, l'utilizzo di tecnologie decentralizzate volte a fornire una mappa della filiera del tessile, per mantenere un alto livello di sicurezza e di privacy per tutte le informazioni critiche delle aziende, impone un'elevata complessità di struttura; dall'altra, l'esigenza di rendere la soluzione quanto più fruibile dalle aziende che compongono l'intero distretto di San Mauro Pascoli, richiede una attenzione particolare alla facilità di utilizzo. La sintesi che nasce dal confronto di queste due nature deve quindi tradursi in un compromesso, che riguarda necessariamente anche lo sviluppo delle interfacce delle soluzioni adottate.

La soluzione web quindi si prende carico di gestire informazioni delle utenze e di gestirne le diverse tipologie, in maniera da supportare gli utenti stessi nell'utilizzo dei diversi componenti e funzionalità forniti dagli strati sottostanti dell'architettura software predisposta dal progetto, cercando di venire incontro alla naturale diffidenza delle imprese a fornire i dettagli sui propri fornitori.

L'attenzione particolare a non divulgare queste informazioni da parte delle aziende non risponde alla volontà di nascondere il proprio operato, bensì a logiche di mercato molto stringenti. Venire incontro alle imprese in questa direzione si è manifestato di fondamentale importanza: le potenzialità e l'autorevolezza del caso d'uso nel settore moda/accessori (per il quale è stata coinvolta l'impresa Biancoaccessori, parter industriale del progetto che ha fornito supporto nell'identificare requisiti e vincoli), dipendono in maniera diretta dalla percentuale di imprese del distretto partecipanti.

Tanto maggiore è la partecipazione, tanto più fedele e accurata è la fotografia della composizione della filiera e delle aziende della supply chain; questo rafforza e rende ancora più convincente una dimostrazione di produzione etica e sostenibile non solo all'interno della propria azienda, ma anche nell'intero processo di produzione della filiera.

Per quanto riguarda la gestione delle certificazioni aziendali, le esigenze di mercato delle aziende richiedono un approccio più elastico per consentire la possibilità a imprese e clienti di accedere, visionare e verificare i documenti e le informazioni. Data la natura eterogenea delle possibili certificazioni (ad esempio, DURC, SA8000), la flessibilità della soluzione nella gestione e nella notarizzazione su blockchain dei tipi di certificati in possesso alle imprese ha il vantaggio di adeguarsi facilmente all'evoluzione della supply chain e di facilitare la partecipazione di tutti gli attori della filiera. In questo contesto, la soluzione implementata raggiunge pienamente gli obiettivi predisposti nel Deliverable D1, fornendo un esempio di supporto affidabile e trasparente alla gestione delle certificazioni rilevanti alla filiera. Attraverso l'interfaccia web, un'impresa può agilmente pubblicare, modificare e cancellare i propri certificati

interfacciandosi facilmente verso la blockchain, che notarizza comunque tutte le informazioni in maniera pubblica e dando garanzia delle operazioni effettuate.

Tutta la complessità di connessione verso i componenti di backend del software (come il CBM ad esempio) viene nascosta e il suo utilizzo rimane trasparente.

I relativi controlli da parte degli organismi preposti vanno comunque effettuati sul materiale e sulla documentazione cartacea di cui l'hash viene caricato sulla blockchain; tuttavia, l'applicativo web fornisce uno strumento per poter esporre una certificazione sulla qualità della filiera in termini di prestigio e di marketing, consentendo di dare una forte caratterizzazione dell'identità dell'intero distretto basata, ad esempio, sulla registrazione dei relativi DURC. In ogni caso la pubblicazione su blockchain pubblica consente anche ad altri di accedere e verificare le stesse informazioni.

La piattaforma web è stata disegnata e progettata seguendo la metodologia privacy by design in modo che rispetti tutte le normative in materia di privacy e le linee guida del GDPR e consentire di aumentare la competitività delle imprese del distretto fornendo uno strumento trasparente, affidabile e sicuro.

Gli scenari futuri sono basati su un accrescimento delle funzionalità del software in merito alla gestione delle relazioni di fornitura delle imprese facenti parte della filiera, tema che rimane molto delicato per la criticità dei dati che devono essere maneggiati (e che si vuole spesso proteggere da una diffusione incontrollata). Su questi aspetti si sono comunque fornite alcune funzionalità di base e creati i presupposti per uno studio, uno sviluppo e un consolidamento sempre più affidabile di questo tipo di servizi:

- da una parte, per implementare una gestione più modulare e flessibile delle informazioni di indicazione di fornitura, in modo da venire incontro alle specifiche esigenze o necessità di una determinata impresa anche in relazione alle modifiche di queste con il passare del tempo;
- dall'altra, per fornire alle utenze un servizio che consenta una gestione differenziata dei permessi di accesso alle proprie informazioni critiche da parte di altri attori del distretto registrati o di specifici grossi partner esterni, in un'ottica di trasparenza volta ad ottenere nuovi contratti e commissioni.

Ulteriori considerazioni potranno inoltre essere condotte sulla natura e sulla gestione della blockchain in un contesto di produzione reale e testare quale soluzione più si adatta alle necessità concrete.

Un contributo fondamentale nei futuri sviluppi del software deriverà dalle osservazioni condotte dall'utilizzo della piazzaforma nel contesto produttivo del distretto di San Mauro Pascoli, che forniranno indicazioni fondamentali per futuri perfezionamenti del progetto.

4. Criteri di test per il caso d'uso Carpigiani

La fase di test inerente al caso d'uso Carpigiani è stata guidata da diversi criteri basati su determinate specifiche appositamente scelte dall'azienda. Difatti il sistema implementato risulta essere in linea con quanto richiesto, ovvero:

- modulare, potenzialmente modificabile in alcune sue parti o estendibile con ulteriori componenti;
- scalabile;
- fault-tolerant.

Nella scelta dei possibili test da effettuare, si è deciso di privilegiare quelli che mettono in evidenza le prestazioni dal punto di vista delle **tempistiche** di approvazione e salvataggio dei refill nella blockchain, della **scalabilità** della rete all'aumentare del numero di transazioni al secondo, dell'**adattamento** della rete alla mancanza di nodi che partecipano alla fase di endorsement-ordering e del **comportamento** dei singoli nodi che non posseggono il ledger sincronizzato per via dell'assenza (mancata connessione o crash) dello stesso.

Al fine di favorire una più semplice gestione e configurazione dell'infrastruttura per la fase di test, si è deciso di procedere con un'integrazione semplificata dei sistemi blockchain all'interno dell'architettura di prova (si veda allegato tecnico D4.2), facendo uso di apposite macchine virtuali finalizzate all'esecuzione dei software per mezzo dei più comuni servizi cloud presenti sul mercato.

Invece, per quanto concerne l'accesso alle informazioni inerenti ai test, si è deciso di sviluppare dei client appositi, in modo da riuscire ad avere degli strumenti che possano essere configurati e adattati correttamente allo specifico caso di test. Oltre a questi software, sono stati sviluppati degli script bash configurabili in base al tipo di test da eseguire. Questi ultimi sono stati fondamentali per permettere di ottenere informazioni, sia generali che di dettaglio, sul comportamento della rete e sull'intero processo di approvazione della transazione del refill, fino alla fase di salvataggio dei blocchi sul ledger.

5. Risultati dell'attività di test e considerazioni per il caso d'uso Carpigiani

In questo capitolo si discuterà dei test svolti e dei relativi risultati ottenuti, creando una rete secondo i passaggi descritti nel documento D4.2. La configurazione della rete utilizzata per la fase di testing, tuttavia, si distacca da quella mostrata nel documento D4.2 al fine di configurare i nodi in modo tale che risultino in linea con le specifiche da soddisfare individuate nella Fase 1 "Analisi delle diverse filiere industriali". A riguardo, sono state configurate le seguenti tre organizzazioni:

- Carpigiani;

- Client1;
- ProducerA.

Per ogni organizzazione sono previsti due peer, con lo scopo di testare sia la scalabilità che la versatilità della rete nel caso in cui un nodo endorser e/o committer dovesse risultare improvvisamente non più disponibile.

In seguito, è stato creato un canale di sistema in cui vengono configurati cinque nodi orderer che costituiscono il servizio di ordering. La creazione di questi permette di effettuare test sulla stabilità e versatilità della rete, verificando così anche il grado di adattabilità di quest'ultima in mancanza di determinati nodi (fault-tolerance).

In generale, riguardo la scelta del numero di peer e/o di orderer, è possibile gestire e scalare il numero di tali nodi liberamente in base alle proprie necessità.

5.1. Test di fault-tolerance

Di seguito vengono illustrati i test effettuati sui componenti della rete per verificare quanto descritto nel paragrafo precedente riguardo la fault-tolerance della rete:

- **Test 1.** In questo test è stato verificato il comportamento della rete in caso di rimozione di un nodo orderer che non sia stato eletto leader in fase di creazione del servizio di ordering. Ne è conseguito che il servizio ha continuato a operare normalmente proseguendo l'attività di creazione dei blocchi a ogni richiesta di transazione legittima effettuata.
- **Test 2.** In questo test è stato osservato il comportamento della rete in caso di rimozione di un nodo orderer, eletto leader in fase di creazione del servizio di ordering. In questo caso la rete ha interrotto l'esecuzione del servizio poiché non risultava più disponibile il leader incaricato della creazione del blocco per ogni nuova legittima richiesta di transazione. La causa di ciò è dovuta alla rete, la quale impiega un breve periodo di tempo per eleggere un nuovo nodo orderer come leader tramite lo scambio dei pacchetti di voto. Una volta terminata l'elezione, e quindi ottenuta la maggioranza assoluta, il nuovo leader avrà il compito di creare i nuovi blocchi e distribuirli per tutta la rete in broadcast.
- **Test 3.** In questo test è stata creata la condizione necessaria, tramite rimozione di più orderer contemporaneamente dalla rete, affinché non possa essere eletto un nuovo leader a causa del non raggiungimento della maggioranza assoluta ((n+1)/2). Nel nostro caso, sono stati arrestati tre nodi orderer su cinque, in modo da non riuscire a ottenere la maggioranza assoluta ((5+1)/2). Non raggiungendo il quorum, per ogni richiesta di transazione effettuata, questa veniva rifiutata a causa proprio del servizio di ordering interpretato come "non disponibile". Per ritornare a una condizione di

normale funzionamento della rete, è stato necessario ripristinare almeno un nodo orderer e raggiungere così il numero minimo di nodi attivi necessario a eleggere il nuovo leader durante la fase di elezione.

- **Test 4.** Il quarto test si prefigge di verificare il comportamento della rete nel momento in cui un nodo committer si arresta. In questo caso non viene manomesso il servizio di ordering, bensì uno dei nodi dell'organizzazione che ha il compito di detenere il ledger aggiornato. Il test consiste nello spegnere il nodo per simulare un crash o l'assenza di comunicazione di rete da parte del suddetto nodo. Quest'ultimo non avrà modo di aggiornare il suo ledger ogni qualvolta vengono invocate nuove richieste di transazione e creati nuovi blocchi. Per porre rimedio a questo problema, Fabric sfrutta un protocollo chiamato **Gossip**. Questo consiste in un servizio che garantisce il processo di riconciliazione dello stato, ovvero permette la sincronizzazione dello **stato globale** della blockchain tra i peer su ciascun canale. Ogni peer estrae continuamente blocchi da altri peer sul canale, al fine di aggiornare il proprio stato se vengono identificate discrepanze. Inoltre, poiché non è necessario disporre di una connettività stabile per permettere la diffusione dei dati basata su **Gossip**, il processo fornisce, alla rete, in modo affidabile la consistenza e l'integrità dei dati del registro condiviso, inclusa la tolleranza per le assenze del nodo. Come risultato del test, infatti, si è notato un incremento degli scambi di messaggi all'interno della rete dovuto al fatto che il peer committer, una volta riconnesso, ha effettuato richiesta di tutti i blocchi che erano stati creati e che non aveva potuto registrare durante il periodo di assenza.
- **Test 5.** Il quinto test consiste nella verifica dell'adattabilità della rete alla rimozione del nodo endorser appartenente a una delle tre organizzazioni e di una successiva richiesta di transazione sul canale. Dall'esecuzione del test si è notato che questo non ha comportato effetti sul funzionamento generale della rete poiché ogni organizzazione possiede due nodi endorser; pertanto, è stato possibile sfruttare il secondo nodo endorser presente sul canale. Successivamente, si è testato il caso in cui una organizzazione non avesse più nodi endorser spegnendo anche il secondo. In questo scenario, non è stato possibile approvare le transazioni richieste poiché non risulta possibile soddisfare la politica di tipo AND implementata. Qualora fosse stata implementata una politica differente, la richiesta sarebbe stata approvata dalle altre due organizzazioni.

5.2. Test prestazionali

Per determinare le prestazioni della rete realizzata inerenti al momento in cui si attiva la creazione di nuove transazioni dal nodo endorser, è stato monitorato il tempo necessario per eseguire le transazioni e l'utilizzo della CPU aumentando la frequenza delle transazioni.

Si noti che, in fase di test, sono state sfruttate delle VM con prestazioni simili a quelle che poi sono state effettivamente create in cloud per la rete di Carpigiani. Le suddette VM dispongono di 1GB di RAM per consentire il corretto funzionamento dei peer o degli orderer. A riguardo, è stato comunque testato il funzionamento di una VM con 512 MB di RAM, osservando che riesce a supportare un nodo. D'altro canto, poiché il peer crea un altro container (concetto introdotto nel Deliverable D4.2) per la gestione del chaincode, si è deciso di aumentare la capacità di memoria RAM per avere un piccolo margine tra il quantitativo usato e la disponibilità massima ed essere così sicuri del fatto che sia sufficiente a mantenere in esecuzione due container contemporaneamente. Per quanto concerne le capacità di elaborazione, sono state virtualizzate quattro CPU da 1.4Ghz. Questa configurazione risulta sufficiente a gestire un discreto traffico del canale senza che vada in saturazione. Come sistema operativo si è scelto Linux, più precisamente Ubuntu 18.04 LTS. Le VM sono state successivamente distribuite su diverse reti per simulare uno scenario di produzione, i cui nodi sono dislocati in luoghi differenti. La versione di Fabric utilizzata è la 2.2.

Per svolgere il test si è partiti dall'eseguire una transazione ogni 20.0s fino a una ogni 2.5s (ogni valore rappresenta la media di dieci test) e il servizio di ordering è stato configurato per creare un nuovo blocco per ogni transazione (la dimensione di ogni blocco è di 6.2 KB). Si noti che in un ambiente di produzione vi sono diverse macchine che richiedono contemporaneamente la generazione di nuove transazioni. In tale scenario ogni blocco contiene tipicamente più transazioni, limitando così l'overhead di ciascuna di esse. Tuttavia, limitando a uno il numero di transazioni per blocco, abbiamo potuto testare le prestazioni della soluzione proposta considerando un caso impegnativo in cui vengono creati nuovi blocchi non appena il servizio di ordering riceve una nuova transazione.

In caso di una transazione ogni 20.0s, il tempo totale di esecuzione del Fabric client, ovvero da quando viene avviata una nuova transazione a quando viene aggiornato il registro, è di circa 4.3s. Si noti che le funzioni di configurazione consistono principalmente nell'identificazione e nella lettura dei certificati TLS dei nodi partecipanti, nella creazione di istanze dell'attore della rete Fabric (canali, peer e orderer) e nell'ottenimento/verifica delle credenziali dell'utente che crea la transazione. Il tempo per l'approvazione, l'ordine, la convalida e il commit è dell'ordine di decine di ms, mostrando che i processi di endorsement-ordering-validation sono quasi immediati.

Analizzando la quantità di risorse computazionali utilizzate nel nodo, si è notato che questa è limitata, ovvero una media del 9% della CPU. All'aumentare della frequenza delle transazioni, la media del carico della CPU aumenta, ma il tempo totale dall'inizio della richiesta di transazione al commit non aumenta, anche portando il sistema a gestire una transazione ogni 2.5s.

Eseguendo un ulteriore test e portando il tempo di ogni transazione a 1.0s, il carico computazionale sul nodo satura la capacità computazionale offerta dalla CPU, causando un forte aumento del tempo totale.

Infine, si noti che i risultati delle prestazioni presentati mostrano e sottolineano la bontà della soluzione proposta. Infatti, la frequenza tipica con cui una macchina da gelato esegue le ricariche è inferiore a quelle testate; pertanto, i risultati raggiunti confermano che l'architettura implementata, sfruttando l'hardware discusso nel punto 3.3.2, è adatta per il caso d'uso che richiede tempistiche ben definite.

5.3. Considerazioni conclusive

Il gruppo di ricerca dell'Università di Ferrara ha deciso di esporre alcune considerazioni osservate sia in fase di progettazione, che durante lo sviluppo della piattaforma discussa nell'allegato D3.2. Inizialmente vengono illustrate le difficoltà che sono state riscontrate in fase di progettazione e, soprattutto, in fase di implementazione. Queste, tuttavia, variano in base anche alla conoscenza di base che l'utente possiede per quanto riguarda Fabric e, in generale, Hyperledger.

Si noti che per implementare l'architettura presentata sono state utilizzate delle Virtual Machine (VM) configurate in modo da avere prestazioni simili ai server usati per la rete Carpigiani nell'infrastruttura IaaS. Inoltre, in questa sezione vengono esposte le scelte adottate in fase di implementazione, considerazioni sull'usabilità del software e come potrebbe essere utilizzato in contesti reali. A riguardo, per avere un feedback da parte dell'azienda relativa al caso d'uso implementato, sono state svolte interviste al personale aziendale che si è occupato di utilizzare tale piattaforma, in modo da avere un loro parere su prestazioni, intuitività, gestione dell'infrastruttura ed eventuale presenza di bug o arresti improvvisi del sistema.

5.3.1. Difficoltà riscontrate: conoscenze di base, progettuale e implementativo

Durante la fase implementativa, di cui si è discusso nel documento D3.2, sono state riscontrate alcune difficoltà che hanno portato a dover effettuare determinate modifiche a livello di implementazione rispetto a ciò che era stato ideato in fase di progettazione.

Per quanto riguarda il nodo inizialmente progettato all'interno della macchina da gelato, è stato deciso di spostarlo in cloud sfruttando le tecnologie rese disponibili da Amazon Web Services (AWS). Tale scelta ha permesso di ridurre il carico di lavoro dei server Carpigiani e la gestione dell'infrastruttura. Nello specifico, i motivi che hanno portato a prendere questa decisione, sono i seguenti:

- **Piano tariffario carta SIM.** La carta SIM installata sulle macchine da gelato ha un piano tariffario molto limitato. Il consumo massimo di dati mensili è di 10 MB. Con le sole

operazioni svolte dai nodi nella fase di endorsement-ordering-validation, il consumo viene gestito in maniera opportuna e rimane entro i limiti stabiliti; tuttavia, in Fabric viene implementato di default il protocollo Gossip che, come anticipato, genera molto traffico. D'altra parte, questo è indispensabile per verificare quali nodi sono online e quali assenti tramite lo scambio di pacchetti keep-alive tra i nodi della rete. Come soluzione, si sarebbe potuto modificare il comportamento di questo protocollo rallentando notevolmente lo scambio dei messaggi per riuscire a restare entro i limiti del piano tariffario, ma Fabric non sarebbe stato "automatizzato", specialmente nella fase di scelta dei nodi endorser. Nello specifico, durante la fase di richiesta di transazione, poiché il client non avrebbe modo di verificare quale nodo sia attivo, risulterebbe necessario impostare manualmente l'indirizzo del nodo endorser al quale richiedere l'approvazione. Ciò, però, avrebbe causato problemi a livello di endorsement una volta che il nodo prescelto si sarebbe assentato e, quindi, non avrebbe svolto la fase di approvazione bloccando di conseguenza l'intero canale. In sintesi, non vi sarebbe stata alcuna forma di resilienza. Mantenendo l'uso del protocollo Gossip configurato di default, invece, il client può riconoscere quali nodi sono attivi e quindi può scegliere il nodo endorser in maniera automatizzata.

- **Costo schede.** Le schede necessarie per eseguire il software per i nodi Fabric devono possedere un'architettura del processore a 64bit e supportare il sistema operativo Linux. Pertanto, il requisito spinge all'uso di microprocessori sufficientemente potenti da supportare un elevato carico. Inoltre la quantità di RAM disponibile non deve essere inferiore a 1 GB. In conclusione il costo calcolato risulta essere eccessivo per questa tipologia di configurazione.
- **Gestione certificati**. Tutti i certificati per le comunicazioni in sicurezza risiedono nel file system del nodo blockchain e, per poterli aggiornare, bisognerebbe connettersi alle schede. Essendo queste ultime collocate sulle macchine e potenzialmente distribuite in qualsiasi paese, la connessione remota porterebbe a un consumo elevato a livello di traffico di rete, rischiando di eccedere il limite massimo imposto dal piano tariffario discusso nel primo punto.
- **Aggiornamento software**. La soluzione che colloca il nodo all'interno delle macchine da gelato potrebbe creare problemi dovuti alla difficile gestione degli aggiornamenti OTA dello stesso. Non solo si potrebbe presentare la necessità di aggiornare il software Carpigiani, ma anche Fabric a una versione più recente e potenzialmente più sicura e prestante.

In aggiunta, sempre per quanto riguarda la gestione dei certificati, questa potrebbe divenire ostica nel tempo poiché è necessario tenere traccia della loro scadenza e, di conseguenza, provvedere alla loro sostituzione. Per ovviare a questo problema, Fabric fornisce una

Certification Authority che, in modo automatico, rilascia certificati ogni qualvolta un membro dell'organizzazione abilitato ne faccia richiesta.

5.3.2. Scelte implementative

In questa sezione si discutono le scelte implementative effettuate in fase di creazione della rete per Carpigiani, usata anche per i test prestazionali e di fault-tolerance.

Canale

Per Carpigiani, risulta semplice poter aggiungere organizzazioni alla rete blockchain: è sufficiente creare un nuovo canale in cui inserire la nuova organizzazione che farà parte della terna di attori. Quando viene creato il canale, per comodità viene automaticamente inserita Carpigiani dato che sarà presente in ognuno di essi e questo ne semplifica la gestione da parte del personale amministratore. Successivamente è possibile creare una nuova organizzazione da inserire nel suddetto canale. Per raggiungere la terna si potrebbe inserire Client1 già creato in precedenza o inserire una organizzazione completamente nuova che faccia riferimento a un cliente differente.

Fabric client

Per quanto concerne il Fabric client, è stato sviluppato in modo tale da richiedere l'approvazione della proposta di transazione per un massimo di cinque volte con un intervallo di tempo via via crescente in modo esponenziale. Se questo non è riuscito a contattare tutti i peer di interesse entro il numero di tentativi stabiliti, si occuperà di avvisare la macchina da gelato dell'impossibilità di effettuare la richiesta.

Chaincode

Considerato il fatto che Fabric e i componenti sono scritti in Go, si è deciso di implementare lo smart contract nello stesso linguaggio di programmazione con lo scopo di ottenere migliori prestazioni. Per quanto riguarda l'installazione dello smart contract nella rete, è un passaggio che, per motivi di più semplice gestione, andrebbe svolto successivamente alla creazione del canale e all'inserimento delle organizzazioni. È possibile, tuttavia, installare il chaincode prima che venga aggiunta un'organizzazione. In questo caso il chaincode va installato separatamente nei nuovi peer e, successivamente, approvato a livello di canale. Questo passaggio è fondamentale per sincronizzare tutti i peer sulla versione del chaincode che si sta usando a livello di canale.

Peer

Le organizzazioni inserite nel canale inizialmente non hanno peer. Difatti, questi ultimi vanno creati e inseriti nel canale attraverso l'operazione di Join. Carpigiani ha la possibilità di inserirli uno alla volta o effettuare una Join multipla di tutti i nodi dell'azienda. Per decidere quale nodo

sarà effettivamente un endorser, basta installare il chaincode su di esso. Gli altri che non posseggono il chaincode sono dei semplici nodi committer. Una volta creato il canale, si è scelto di effettuare una multiple Join da parte dei peer di Carpigiani. Questo consente di avere maggiore resilienza in tutti i canali e gestire meglio il carico di lavoro sulla rete in generale.

5.3.3. Applicabilità in casi reali e intervista al personale aziendale

Nel corso del lavoro di progetto, è stato intervistato il personale aziendale che si occupa della gestione della piattaforma riguardo l'usabilità della stessa. Da queste interviste è risultato che l'applicazione implementata svolge adeguatamente i compiti richiesti senza particolari richieste espresse dal personale. Inoltre, l'azienda ha anche effettuato un test inerente alla condizione in cui il peer endorser del client risulti spento per un paio di settimane e, una volta riattivato, tutte le precedenti transazioni arretrate vengono sincronizzate attraverso il protocollo *Gossip* spiegato precedentemente. Da questo test è risultato che la comunicazione tra macchina e infrastruttura Fabric è sempre avvenuta con successo senza alcuna perdita dei messaggi scambiati. Una volta riattivato il peer, infatti, il client di Fabric ha ricevuto i messaggi pendenti e ha potuto effettuare nuovamente la proposta di transazione che poi è stata approvata e salvata correttamente nella blockchain. A riguardo, per poter identificare un eventuale problema relativo a un peer, Carpigiani ha espresso la necessità di un allarme o una notifica email che avvisi immediatamente lo stato di assenza del nodo.

Per quanto concerne l'usabilità dell'applicazione, risulta essere molto performante anche in casi in cui vengono richieste molte approvazioni di Refill. Tuttavia, possono esserci rallentamenti nell'ottenere le informazioni a causa della congestione di rete o per il carico eccessivo del peer. Questo potrebbe essere risolto aggiungendo più nodi peer, permettendo un migliore bilanciamento del carico delle operazioni. La gestione della rete blockchain non ha mai dato problemi, consentendo agli operatori di poter effettuare le operazioni volute correttamente.

La richiesta di altre informazioni, come visualizzare le macchine presenti nel canale, vedere il proprio profilo utente e osservare le informazioni inerenti alle macchine da gelato, risulta essere veloce. Il layout dell'applicazione e la sua organizzazione risulta essere molto intuitiva, semplice da usare e di facile interazione. In aggiunta, un modo per semplificare ulteriormente la visualizzazione di tutte le macchine appartenenti a un client, sarebbe quella di sviluppare una funzionalità di ricerca dedicata, per poi prendere visione di quella specifica macchina da gelato, similmente a come avviene nella schermata dei Refill (si veda documento D3.2).

Infine, per quanto riguarda la gestione dei certificati, deve ancora essere trovata una soluzione per il loro aggiornamento in modo semplice. Un modo potrebbe consistere in un comando impartito dal client Fabric che effettui in modo automatizzato la richiesta di un nuovo certificato al CA attivo su un altro server.