WebService and IoT

E.Masi L.Genghini M.Sertori

elisabetta.masi@studio.unibo.it
genghini.luca@studio.unibo.it
matteo.sertori@studio.unibo.it

Corso Sistemi Distribuiti
LM Ingegneria e Scienze informatiche
Alma Mater Studiorum Universita di Bologna sede di Cesena

a.a. 2017/2018

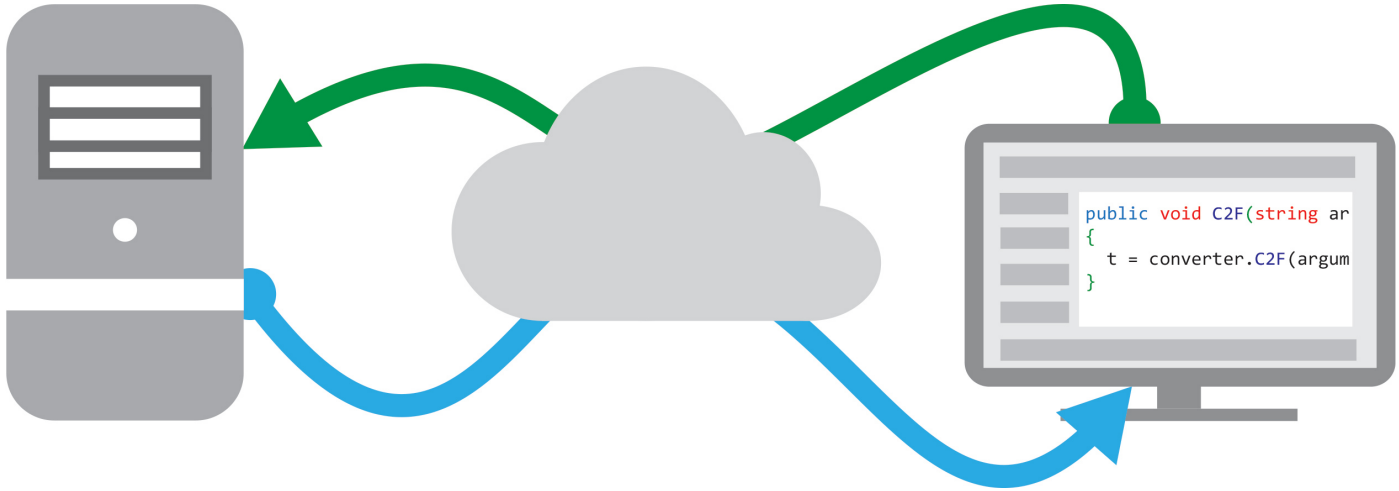
Indice

Indice

Web Service

Theorem

Un Web Service e' un componente applicativo definibile come un sistema software in grado di mettersi al servizio di un applicazione comunicando su di una medesima rete tramite un protocollo prestabilito. Un Web service consente quindi alle applicazioni che vi si collegano di usufruire delle funzioni che mette a disposizione.



Perché utilizzare i Web Service (1)

- I WS permettono linteroperabilitá tra diverse applicazioni software e su diverse piattaforme hardware/software.

Perché utilizzare i Web Service (1)

- I WS permettono linteroperabilitá tra diverse applicazioni software e su diverse piattaforme hardware/software.
- Utilizzano un formato dei dati di tipo testuale, quindi piú comprensibile e piú facile da utilizzare per gli sviluppatori (esclusi ovviamente i trasferimenti di dati di tipo binario).

Perché utilizzare i Web Service (1)

- I WS permettono linteroperabilitá tra diverse applicazioni software e su diverse piattaforme hardware/software.
- Utilizzano un formato dei dati di tipo testuale, quindi piú comprensibile e piú facile da utilizzare per gli sviluppatori (esclusi ovviamente i trasferimenti di dati di tipo binario).
- Essendo basati sul protocollo HTTP, non richiedono modifiche alle regole di sicurezza utilizzate come filtro dai firewall.

Perché utilizzare i Web Service (1)

- I WS permettono linteroperabilitá tra diverse applicazioni software e su diverse piattaforme hardware/software.
- Utilizzano un formato dei dati di tipo testuale, quindi piú comprensibile e piú facile da utilizzare per gli sviluppatori (esclusi ovviamente i trasferimenti di dati di tipo binario).
- Essendo basati sul protocollo HTTP, non richiedono modifiche alle regole di sicurezza utilizzate come filtro dai firewall.
- Sono semplici da utilizzare e possono essere combinati luno con laltro per formare servizi integrati e complessi.

Perché utilizzare i Web Service (2)

- Permettono di riutilizzare applicazioni giá sviluppate.

Perché utilizzare i Web Service (2)

- Permettono di riutilizzare applicazioni giá sviluppate.
- Fintanto che linterfaccia rimane costante, le modifiche effettuate ai servizi rimangono trasparenti.

Perché utilizzare i Web Service (2)

- Permettono di riutilizzare applicazioni giá sviluppate.
- Fintanto che linterfaccia rimane costante, le modifiche effettuate ai servizi rimangono trasparenti.
- I servizi web sono in grado di pubblicare le loro funzioni e di scambiare dati con il resto del mondo.

Perché utilizzare i Web Service (2)

- Permettono di riutilizzare applicazioni giá sviluppate.
- Fintanto che linterfaccia rimane costante, le modifiche effettuate ai servizi rimangono trasparenti.
- I servizi web sono in grado di pubblicare le loro funzioni e di scambiare dati con il resto del mondo.
- Tutte le informazioni vengono scambiate attraverso protocolli aperti.

Svantaggi e problematiche dell' utilizzo dei WS (1)

- **Le performance:** I WS presentano performance drasticamente inferiori rispetto ad altri metodi di comunicazione utilizzabili in rete. Essendo basati su XML ogni trasferimento di dati richiede linserimento di un notevole numero di dati supplementari (i tag XML) indispensabili per la descrizione dell operazione. Inoltre tutti i dati inviati richiedono di essere prima codificati e poi decodificati ai capi della connessione. Queste due caratteristiche rendono i WS poco adatti a flussi di dati intensi o dove la velocitá dell applicazione rappresenti un fattore critico.

Svantaggi e problematiche del'l'utilizzo dei WS (2)

- **Il protocollo base, HTTP:** Quando si sviluppa un WS e' necessario tener conto del protocollo di base. E' quindi indispensabile disporre di unapplicazione terza che gestisca le richieste HTTP oppure e' necessario includerla direttamente nel codice del nostro programma qualora si desideri la sua totale indipendenza. Eseguendo il codice del Web service attraverso un server web la gestione di HTTP e' immediatamente assicurata.

Web Service e protocolli

Come dettro in precedenza, il protocollo di base per i Web Service e' HTTP. OItre ad esso per, vengono utilizzati altri standard web, tutti basati su XML.
XML puo' essere utilizzato correttamente tra piattaforme differenti (Linux, Windows, Mac) e differenti linguaggi di programmazione, garantendo di esprimere messaggi e funzioni complessi e che tutti i dati scambiati possano essere utilizzati ad entrambi i capi della connessione.
Si puo' dire quindi che i Web Service sono basati su XML e HTTP e possono essere utilizzati su ogni piattaforma e con ogni tipo di software.

Web Service e protocolli

Standard web basati su XML:

Web Service e protocolli

Standard web basati su XML:

- **XML Schema**

Web Service e protocolli

Standard web basati su XML:

- **XML Schema**
- **UDDI** (Universal Description, Discovery and Integration)

Web Service e protocolli

Standard web basati su XML:

- **XML Schema**
- **UDDI** (Universal Description, Discovery and Integration)
- **WSDL** (Web Service Description Language)

Web Service e protocolli

Standard web basati su XML:

- **XML Schema**
- **UDDI** (Universal Description, Discovery and Integration)
- **WSDL** (Web Service Description Language)
- **SOAP** (Simple Object Acces Protocol)

XML Schema

Conferisce ai Web Service la caratteristica di essere auto-descrittivi. Le principali funzioni di questo protocollo sono:

- Definire gli elementi (tag) che possono apparire in un documento.
- Definire gli attributi che possono apparire in un elemento.
- Definire quali e quanti elementi devono essere inseriti in altri elementi (tali elementi vengono chiamati "child").
- Definire il tipo per ogni elemento e per gli attributi.
- Definire i valori di default o fissi per elementi ed attributi.

WSDL

Questo linguaggio serve a specificare dove si trovano i servizi e le operazioni esposte dal servizio web. All'interno di un documento WSDL esistono quattro elementi principali:

- **types:** utilizzato per definire il tipo degli elementi che utilizzeremo all'interno del documento.
- **message:** utilizzato dall'applicazione che vuole usufruire dei servizi di un Web Service per comunicare con esso, e viceversa.
- **portType:** ha il compito di definire un WS, le operazioni che puo' effettuare ed i messaggi che sono coinvolti in questo processo. Ogni operazione definibile al suo interno puo' essere:
  - **One-way:** puo' ricevere un messaggio ma non ritorna alcuna risposta.
  - **Request-response:** puo' ricevere una richiesta e ritorna una risposta.
  - **Solicit-response:** puo' inviare una richiesta ma non attende per la risposta.
  - **Notification:** puo' inviare un messaggio ma non attende la risposta.
- **binding:** utilizzato per creare il collegamento con il proptocollo Soap.

Il protocollo Soap

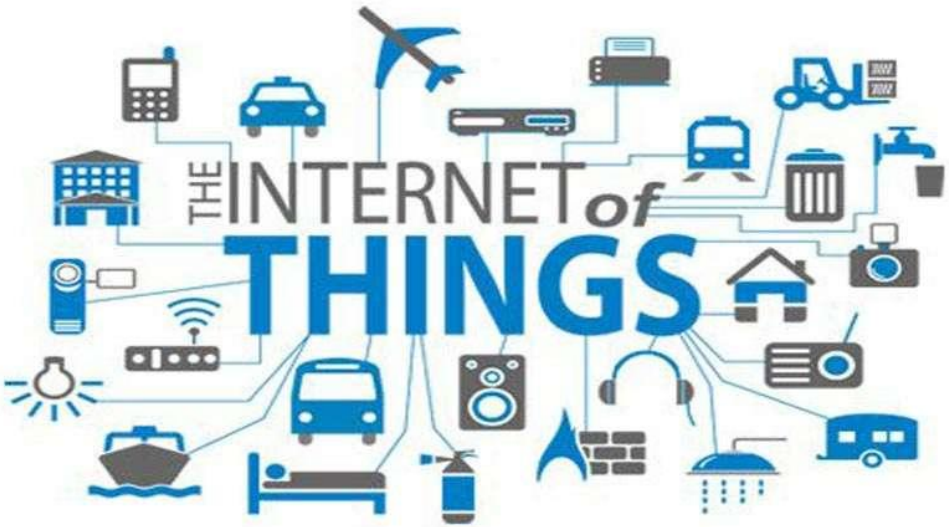
Questo protocollo fornisce una via per comunicare tra applicazioni eseguite su sistemi operativi diversi, con diverse tecnologie e linguaggi di programmazione, tramite HTTP ed XML. Un messaggio SOAP un documento XML che contiene i seguenti elementi:

- **Envelope,** identifica il documento come un messaggio SOAP.
- Un elemento **Header** opzionale, contenete informazioni specifiche per lapplicazione.
- **Body** contiene le informazioni scambiate dalle richieste/risposte.
- **Fault** fornisce informazioni riguardo ad eventuali errori manifestati durante la lettura del messaggio.

Indice

Theorem

Con Internet of Things (IoT), si intende l'insieme degli oggetti connessi tra di loro e alla rete, e che grazie questa connessione sono in grado di scambiarsi informazioni e interagire con l'ambiente circostante.



Internet of Things (1)

- "Quando tutto viene connesso, una volta che i dati vengono raccolti in tempo reale, ovunque, sempre, possiamo **misurare, analizzare, anticipare** o addirittura **prevedere eventi e "veritá"** che prima erano nascosti sotto la superficie" - *Gerd Leonhard* -

Internet of Things (1)

- "Quando tutto viene connesso, una volta che i dati vengono raccolti in tempo reale, ovunque, sempre, possiamo **misurare, analizzare, anticipare** o addirittura **prevedere eventi e "veritá"** che prima erano nascosti sotto la superficie" - *Gerd Leonhard* -
- "Connettendo tutto e tutti e diffondendo le macchine, cioé l'intelligenza artificiale e le analisi predittive, molti dei grandi fornitori globali di IoT sperano di **raggiungere una sorta di meta-intelligenza** attraverso un'abilitá esponenzialmente migliore di leggere, comprendere e applicare i dati" - *Gerd Leonhard* -

Vantaggi dell'IoT

- Il modo piú immediato per sfruttare i dati é quello di migliorare i processi interni delle aziende.

Vantaggi dell'IoT

- Il modo piú immediato per sfruttare i dati é quello di migliorare i processi interni delle aziende.
- Vantaggi:
  - Prevenzione dei guasti,
  - Garantire interventi piú tempestivi,
  - Piú precisione nel monitoraggio e di conseguenza migliore diagnosi dei problemi aziendali.

Vantaggi dell'IoT

- Il modo piú immediato per sfruttare i dati é quello di migliorare i processi interni delle aziende.
- Vantaggi:
  - Prevenzione dei guasti,
  - Garantire interventi piú tempestivi,
  - Piú precisione nel monitoraggio e di conseguenza migliore diagnosi dei problemi aziendali.
- In secondo luogo, le informazioni possono essere sfruttate per la produzione di nuove generazioni di prodotti, quindi di versioni migliorative sviluppate proprio a partire dall'analisi di quanto raccolto.

Campi applicativi dell'IoT

- Oltre allo studio e alla processazione dei dati in azienda, l'IoT ricopre la maggior parte degli ambiti informatici:
  - Casa, smart home, domotica
  - Smart health, sanitá, mondo biomedicale
  - Smart city, smart mobility
  - Nuove forme di digital payment tramite oggetti

Campi applicativi dell'IoT

- Oltre allo studio e alla processazione dei dati in azienda, l'IoT ricopre la maggior parte degli ambiti informatici:
  - Casa, smart home, domotica
  - Smart health, sanitá, mondo biomedicale
  - Smart city, smart mobility
  - Nuove forme di digital payment tramite oggetti
- Con lo sviluppo dilagante di queste nuove tecnologie, aziende del calibro di Amazon e Google hanno virato anche loro verso l'ambito dell'IoT.

Campi applicativi dell'IoT

- Oltre allo studio e alla processazione dei dati in azienda, l'IoT ricopre la maggior parte degli ambiti informatici:
  - Casa, smart home, domotica
  - Smart health, sanitá, mondo biomedicale
  - Smart city, smart mobility
  - Nuove forme di digital payment tramite oggetti
- Con lo sviluppo dilagante di queste nuove tecnologie, aziende del calibro di Amazon e Google hanno virato anche loro verso l'ambito dell'IoT.
- Sono nate cosí nuove teconlogie tra cui AWS IoT (Amazon) e Google Cloud IoT.

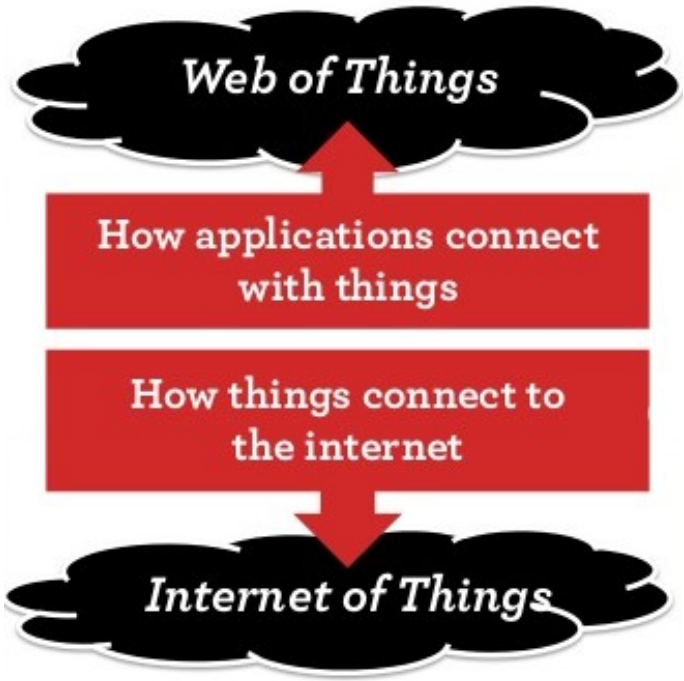
Indice

Web Service per IoT

Il continuo progredire di *Internet of Things* ha fatto crescere la necessitá di iniziare a pensare ai Web Services non solo come software utilizzabili da altri applicativi software, ma bensí come software a disposizione in qualunque momento ai vari dispositivi connessi alla rete e all'ambiente circostante che necessitino delle funzioni di quel Web Service per adempiere alla loro.

Da Internet of Things a Web of Things

Nell'ambito del *Web of Things* (WoT) i dispositivi non solo dispongono di un indirizzo IP e sono interconnessi con la rete, ma diventano anche in grado di parlare lo stesso linguaggio, diventando cosí in grado di comunicare ed interoperare liberamente sul web.



RESTful Web Services (1)

Le due classi principali di Web Service sono:

- **Representational State Transfer (REST)** Web Services: le risorse sono manipolate utilizzando un insieme uniforme di operazioni "stateless".

RESTful Web Services (1)

Le due classi principali di Web Service sono:

- **Representational State Transfer (REST)** Web Services: le risorse sono manipolate utilizzando un insieme uniforme di operazioni "stateless".
- **Web Services arbitrari:** utilizzano un insieme di operazioni arbitrarie ed utilizzano ad esempio messaggi Soap.

RESTful Web Services (2)

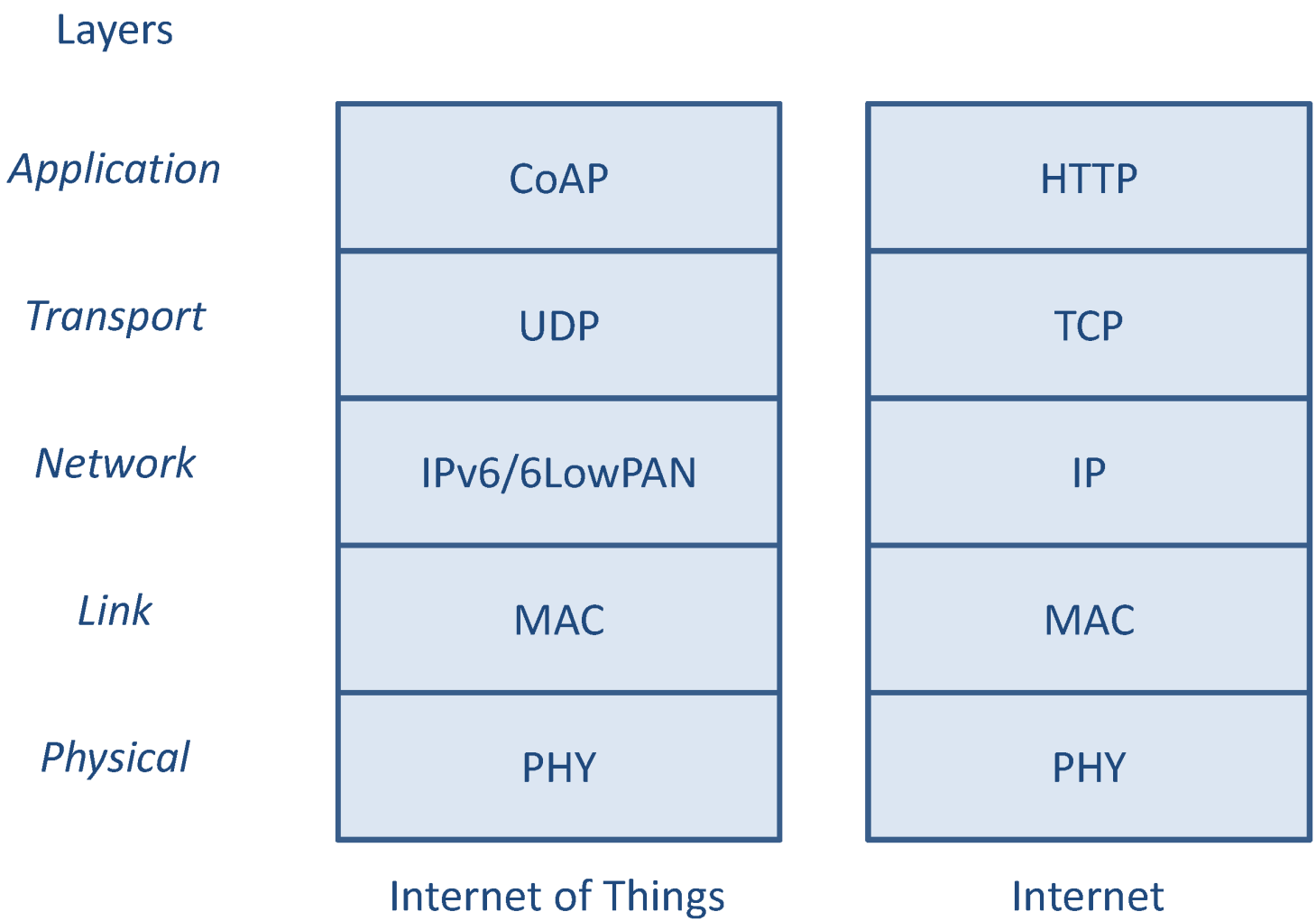
Nell'ambito dell'IoT, i RESTful Web Service sono piú vantaggiosi rispetto a quelli arbitrari, essendo meno dispendiosi e avendo una minore complessitá di analisi. Oltre a ció, le applicazioni che supportano i RESTful Web Services riescono ad essere performanti anche su dispositivi con risorse limitate, essendo piú semplici da interpretare.

Constrained Application Protocol

Nel 2010, fu pubblicata la prima bozza del nuovo protocollo di trasferimento per RESTful WS chiamato **Constrained Application Protocol (CoAP)**. Questo protocollo include molte delle funzionalitá del protocollo HTTP, rivisionate ed adattate alla minor potenza di elaborazione e ai vincoli energetici dei piccoli dispositivi tipici dell'IoT. Altre funzioni di CoAP sono:

- L'individuazione di risorse integrate.
- Il supporto del multicast IP.
- Lo scambio di messaggi asincroni.

CoAP Protocol Stack



Physical and Data Link layer

Le reti wirless sono fondamentali per WoT, in quanto risulterebbe troppo dispendioso collegare tutti i dispositivi tra di loro. L'**IEEE 802.15.4 MAC**, lo standard utilizzato dal protocollo CoAP, é di fatto focalizzato su:

- Basso consumo energetico.
- Basso costo.
- Bassa velocitá di comunicazione per il trasferimento dati tra dispositivi in collegamento.

Network and Transport layer

- **Network:** lo standard *IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN)* ha portato l'Internet Protocol (IP) per i piccoli dispositivi embedded, come ad esempio i sensori, tipici del WoT, fornendo i mezzi per l'integrazione di questi dispositivi in rete.

Network and Transport layer

- **Network:** lo standard *IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN)* ha portato l'Internet Protocol (IP) per i piccoli dispositivi embedded, come ad esempio i sensori, tipici del WoT, fornendo i mezzi per l'integrazione di questi dispositivi in rete.
- **Transport:** A differenza dell'HTTP, che utilizza solitamente TCP, CoAP utilizza UDP come protocollo per il livello di trasporto, molto piú adatto per le reti WoT, in quanto supporta il multicast ed é molto meno dispendioso per le brevi transizioni.

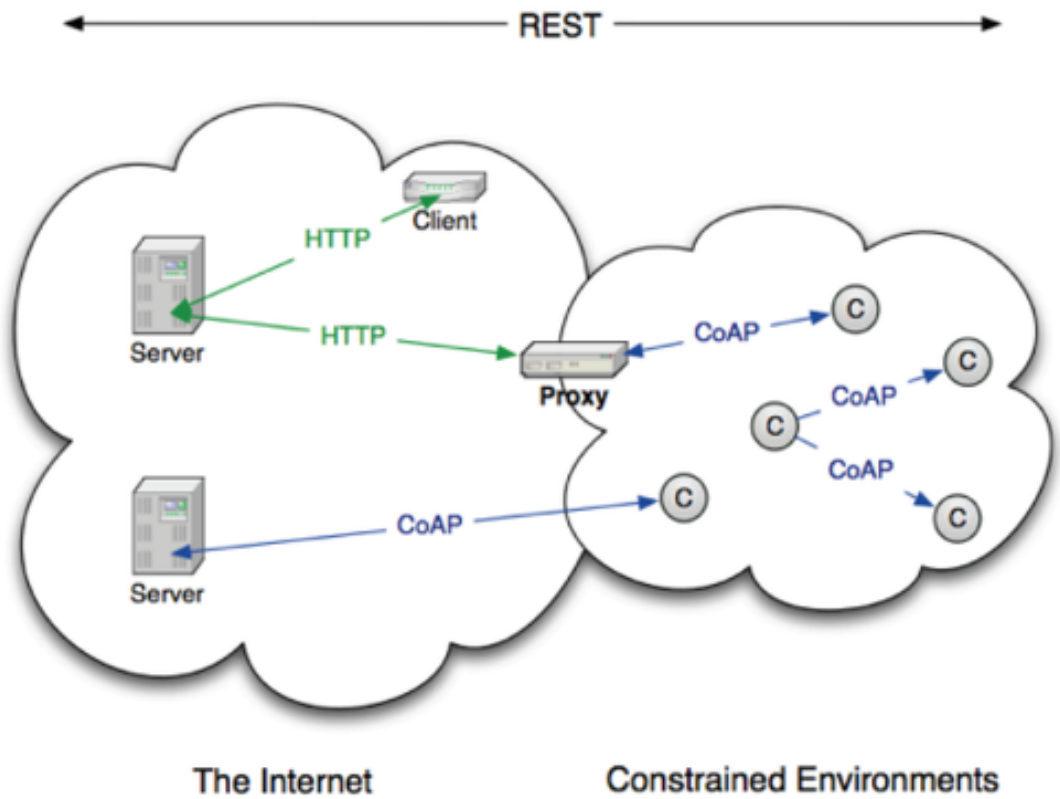
Application layer

Come descritto giá in precedenza, il protocollo CoAP riesce ad ottimizzare i *RESTful Web Services* per le reti ed i dispositivi con risorse limitate (IoT/Wot e applicazioni M2M). Oltre a ció garantisce:

- **Affidabilitá**, anche senza l'utilizzo di TCP come livello di trasporto tramite timeout di default e ritrasmissione di back-off.
- Mantiene l'**overhead** dei messaggi il piú **basso** possibile.
- Comparato al 6LoWPAN, é meno dispendioso e garantisce un **incremento della durata della batteria** per i dispositvi embedded che dispongono di risorse limitate.

Architettura CoRE (Coinstrained RESTful Environments)

La figura sottostante rappresenta un'architettura end-to-end per l'integrazione dei dispositivi embedded col Web basata su IP e RESTful Web Services. L'architettura permette di realizzare un'unica interfaccia sia per HTTP che CoAP,in quanto entrambi i protocolli implementano il paradigma REST.



Indice

AWS IoT

- AWS é una intera classe di servizi offerti ed amministrati tramite una interfaccia comune.

AWS IoT

- AWS é una intera classe di servizi offerti ed amministrati tramite una interfaccia comune.
- L'offerta di AWS comprende sia soluzioni IAAS, sia stack applicativi di tipo PAAS.

AWS IoT

- AWS é una intera classe di servizi offerti ed amministrati tramite una interfaccia comune.
- L'offerta di AWS comprende sia soluzioni IAAS, sia stack applicativi di tipo PAAS.
- I servizi offerti sono suddivisi in cinque macroclassi:
  - Compute & Networking
  - Storage & Content Delivery
  - Database
  - Deployment and Management
  - App Services

Servizi di AWT

Ogni macroaree sono divise a loro volta da molteplici servizi, di seguito saranno citati quelli piú importanti.

Servizi di AWT

Ogni macroaree sono divise a loro volta da molteplici servizi, di seguito saranno citati quelli piú importanti.

- **Compute & Networking:**
  - **EC2:** Servizio dedicato alla creazione e configurazione di istanze virtuali;
  - **Elastic MapReduce:** Servizio dedicato alla manipolazione di grandi moli di dati;
  - **VPC:** Permette la creazione di una vera e propria VPN nel cloud di AWS.

Servizi di AWT

Ogni macroaree sono divise a loro volta da molteplici servizi, di seguito saranno citati quelli piú importanti.

- **Compute & Networking:**
  - **EC2:** Servizio dedicato alla creazione e configurazione di istanze virtuali;
  - **Elastic MapReduce:** Servizio dedicato alla manipolazione di grandi moli di dati;
  - **VPC:** Permette la creazione di una vera e propria VPN nel cloud di AWS.
- **Storage & Content Delivery:**
  - **CloudFront:** Servizio dedicato alla creazione di reti di content delivery, ovvero un modello distribuito per mettere a disposizione contenuti sulla rete globale permettendo download veloci e bassa latenza.
  - **S3 (Simple Storage Service)** Un sistema di storage per il cloud.

Servizi di AWT

- **Database:**
  - **DynamoDB:** Servizio dedicato al data store 'NoSql', una soluzione completamente controllabile da console web;
  - **ElastiCache:** Servizio dedicato al sistema di caching in memoria dei dati;
  - **RDS:** Servizio dedicato ai Database tradizionali (MySQL, Oracle, MS SQL Server).

Servizi di AWT

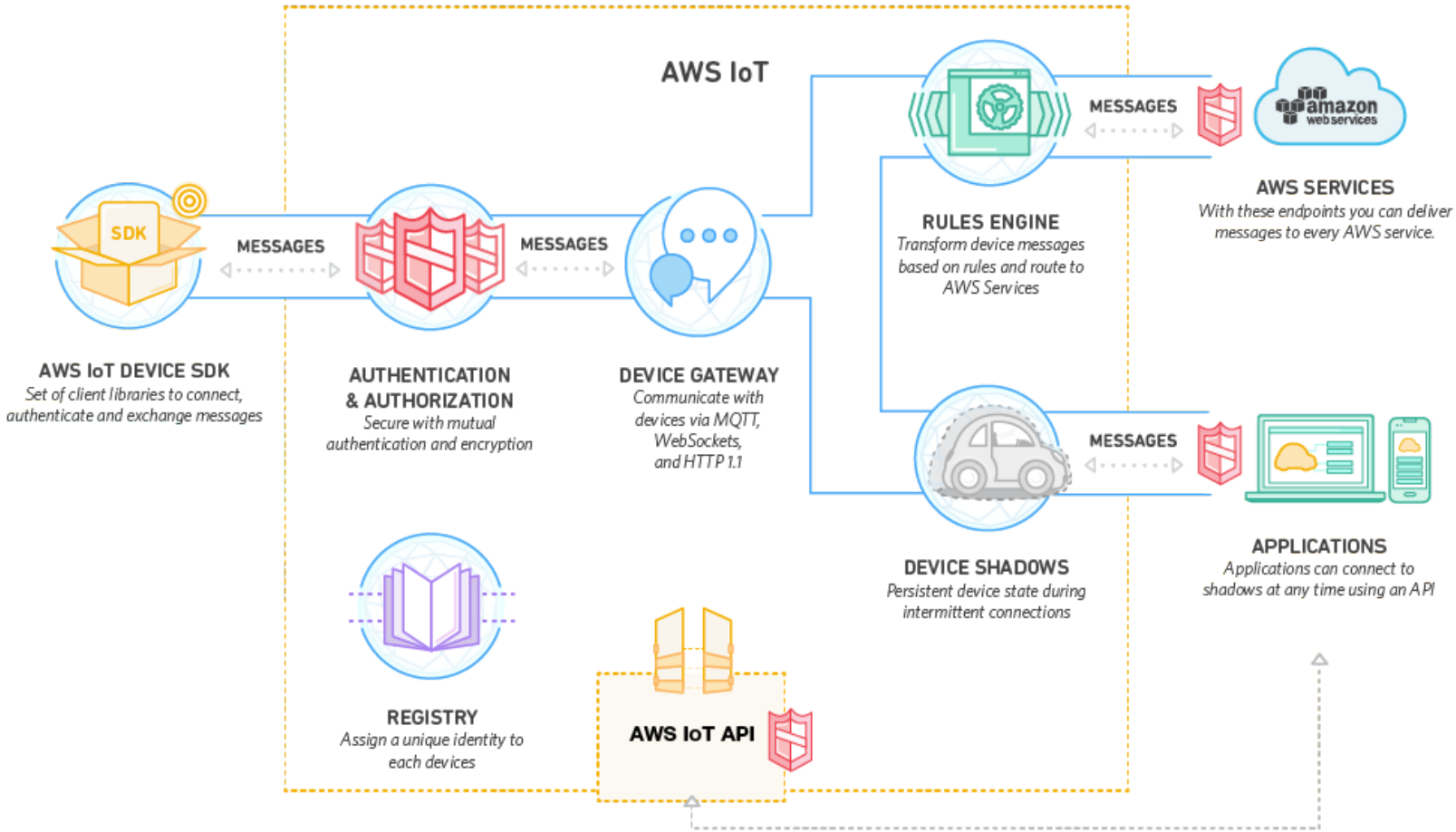
- **Database:**
  - **DynamoDB:** Servizio dedicato al data store 'NoSql', una soluzione completamente controllabile da console web;
  - **ElastiCache:** Servizio dedicato al sistema di caching in memoria dei dati;
  - **RDS:** Servizio dedicato ai Database tradizionali (MySQL, Oracle, MS SQL Server).
- **Deployment and Management:**
  - **CloudFormation:** Essenzialmente una serie di macchine virtuali template da utilizzare per prodotti in configurazioni tipiche (es. WordPress);
  - **CloudWatch:** Servizio utilizzato per il monitoring delle istanze;
  - **Elastic Beanstalk:** per il deployment di applicazioni Java in chiaro stile PAAS.

Servizi di AWT

- **App Services**
  - **CloudSearch:** un servizio che consente di integrare facilmente un motore di ricerca;
  - **SES (Simple Email Service):** un servizio per linvio massivo di email con gestione della transazione;
  - **SNS (Simple Notification Service):** un servizio dedicato alle notifiche istantanee.
  - **SQS (Simple Queue Service):** implementa un sistema per la gestione di code di messaggi.
  - **SWF (Simple Workflow Service):** un servizio per la realizzazione di workflow .

AWT IoT

AWS IoT é un servizio di Amazon AWS che permette la comunicazione **sicura e bi-direzionale** tra i device connessi ad Internet e gli altri servizi AWS.



AWT IoT

- **Device Gataway**: E linterfaccia verso il device. Permette di comunicare in sicurezza ed efficienza con AWS IoT.
- **Message Broker:** il meccanismo che consente la trasmissione e la ricezione tra device e AWS IoT in entrambe le direzioni. Utilizza MQQT per registrare e trasmettere (pu essere utilizzanto HTTP REST).
- **Rules Engine:** Permette lelaborazione dei messaggi e linvio ad altri servizi AWS.
- **Security e Indentity Service:** Identifica il device e provvede alla trasmissione sicura dei dati.

AWT IoT

- **Thing Registry:** Organizza le risorse associate a ciascun device. E' possibile avere fino a 3 attributi che ne indicano lo stato. E anche possibile includere un certificato e un MQQT client ID.
- **Thing Shadow:** Una copia logica del registro vero e proprio relativo al device. (ad esempio potrebbe succedere che il device perde la comunicazione e lo stato reale nel frattempo cambia, il registro shadow mantiene il valore che aveva prima della perdita di comunicazione)
- **Thing Shadow Serivce:** Mantiene persistente la rappresentazione dello stato reale del device nel Cloud. Si occupa di sincronizzare lo stato del device con il thing shadow; viceversa, i device possono aggiornare il thing shadow.

Google Cloud Platform

- Google Cloud Platform un servizio sviluppato da Google il quale fornisce diversi prodotti per permettere ai programmatori di sviluppare rapidamente, utilizzando i servizi di cui hanno bisogno.

Google Cloud Platform

- Google Cloud Platform un servizio sviluppato da Google il quale fornisce diversi prodotti per permettere ai programmatori di sviluppare rapidamente, utilizzando i servizi di cui hanno bisogno.
- Come il giá citato AWS, Gogole Cloud Platform divide i suoi serivizi in quattro macroaree:

Google Cloud Platform

- **Compute:** Permette di eseguire carichi di lavoro su larga scala e di sviluppare app attraverso i servizi integrati.

Google Cloud Platform

- **Compute:** Permette di eseguire carichi di lavoro su larga scala e di sviluppare app attraverso i servizi integrati.
- **Storage:** Utilizzato per salvare dati relazionali e non (NoSQL e MySQL).

Google Cloud Platform

- **Compute:** Permette di eseguire carichi di lavoro su larga scala e di sviluppare app attraverso i servizi integrati.
- **Storage:** Utilizzato per salvare dati relazionali e non (NoSQL e MySQL).
- **Big Data:** Utilizzato per eseguire query SQL-like in tempo reale

Google Cloud Platform

- **Compute:** Permette di eseguire carichi di lavoro su larga scala e di sviluppare app attraverso i servizi integrati.
- **Storage:** Utilizzato per salvare dati relazionali e non (NoSQL e MySQL).
- **Big Data:** Utilizzato per eseguire query SQL-like in tempo reale
- **Services:** Utilizzato mer connettere i serivzi con messaggistica asincrona, per creare DNS recordm per creare servizi RESTful, per creare app multilingua e per creare algoritmi di apprendimento automatico

AWT e Google Cloud: il confronto

Servizio	Vantaggi	Svantaggi
AWS Amazon	AWS si distingue per opzioni di configurazione della piattaforma, funzioni di monitoraggio e di policy, sicurezza e affidabilitá. Un altro vantaggio del cloud AWS é la sua apertura e flessibilitá.	Cloud Ibrido e grandezza eccessiva della scelta di tecnologie e servizi.
Google Cloud	La sua strategia go-to-market é stata focalizzata su progetti piccoli e innovativi e funzionalitá innovative nel mondo del Machine Learning	Puntando sul machine learning Google non si sta concentrando sui serivizi piú comuni che un Cloud dovrebbe avere.

AWT e Google Cloud: il confronto

In conclusione, non esiste un vero e proprio servizio che racchiude tutti i possibili servizi. L'utente in base alle proprie esigenze sceglierá un serivzio al posto che l'altro. Google Cloud Platform ha virato su servizi mirati al Machine Learning e sulla gestione dei database; mentre AWS Amazon ha optato sulla gestione completa di un vero servizio cloud pur avendo i difetti giá citati.

Indice

Problemi di Sicurezza

svantaggi del WS nel Iot

- Obiettivo alla base dei WS é mettere a disposizione un set di tecnologie che siano in grado di facilitare l interoperabilitá tra sistemi eterogenei sfruttare al meglio le risorse distribuite senza preoccupazioni per il tipo di applicazioni, linguaggi di programmazione o sistemi operativi coinvolti.
- I Web services si basano su uninterfaccia descritta in un formato processabile da un elaboratore, il WSDL (Web Services Description Language), scambiano messaggi SOAP (Simple Object Access Protocol) con i sistemi che li invocano e tipicamente si basano su una comunicazione HTTP.

Problemi di Sicurezza

svantaggi del WS nel Iot

- L'approccio alla base dei sistemi WS ha avuto negli anni una grande successo e una discreta diffusione
- Tutto questo ha portato anche alla necessitá di una maggiore sicurezza dell' integritá e confidenzialitá dei dati e dei rispettivi sistemi.

Problemi di Sicurezza

SICUREZZA

La sicurezza e' il percorso non la destinazione finale da raggiungere.

Quando ci si vuole rendere sicuro un sistema é importante infatti:

- valutare inizialmente l'infrastruttura delle applicazioni: le minacce e i rischi che possono incorrere,
- poi studiare e implementare le possibili contromisure da adottare.

Problemi di Sicurezza

Alcune criticitá possono essere

Criticità	Soluzione
Integrità del messaggio	Utilizzare una firma può evitare il rischio di processare messaggi modificati lungo il tragitto.
Confidenzialità	Occorre evitare che entità non autorizzate siano in grado di ottenere accesso a informazioni contenute nei messaggi.
Man in the middle	Tramite meccanismi di mutua autenticazione si può evitare che un attaccante si frapponga tra mittente e destinatario alterando i messaggi.
Spoofing	Tramite tecniche di autenticazione avanzate si può evitare che un attaccante assuma l'identità di un'entità fidata.
Replay Attacks	Utilizzando `timestamp` e numerando le richieste, si può evitare che attacchi basati su reinvio di richieste di elaborazione vadano ad intasare i servizi.

Requisiti di Sicurezza

Ogni sistema deve avere

Requisiti	Descrizione
Autenticazione	Il processo che consente di identificare univocamente l'utente di un'applicazione o un servizio.
Non ripudio	*Auditing* e *logging* sono gli aspetti chiave delle politiche di non ripudio che garantiscono che l'utente non possa negare di avere cominciato e/o concluso operazioni o transazioni.
Autorizzazione	Il processo che governa l'accesso a risorse e operazioni da parte di utenti autenticati.
Confidenzialità	Il processo che garantisce che i dati rimangano privati e non possano essere acceduti da terze parti non autorizzate.
Integrità	Il processo che garantisce che i dati siano protetti da modifiche accidentali o deliberate.
Integrità e confidenzialità End-to-End	Integrità e confidenzialità vanno garantite anche in presenza di intermediari.
Disponibilità	Mantiene il sistema utilizzabile per utenti legittimati, evitando che venga messo fuori servizio tramite crash o attacchi volti a minarne l'accessibilità (come gli attacchi DOS).

Indice

Come creare una soluzione?

L' idea di base quella di dotare i web service di servizi di sicurezza che sia in grado di facilitare e supportare l interoperabilitá e che siano indipendenti da sistemi operativi, infrastrutture applicative e linguaggi di programmazione.

Come creare una soluzione?

L' idea di base quella di dotare i web service di servizi di sicurezza che sia in grado di facilitare e supportare l interoperabilitá e che siano indipendenti da sistemi operativi, infrastrutture applicative e linguaggi di programmazione.

Per raggiungere questo obiettivo sono state studiate una serie di soluzioni tra cui:

- a livello di trasporto
- a livello di messaggio

Sicurezza a livello di Trasporto

A livello trasporto é possibile usare diversi protocolli di sicurezza ma i due principali sono:

- **SSL:** utilizza un sistema di crittografia basato su chiave pubblica e privata. Lautenticazione tra client e server consente di stabilire l'identitá e l'autenticitá delle due entitá. Tramite firma digitale si punta a evitare modifiche non autorizzate dei dati.

Sicurezza a livello di Trasporto

A livello trasporto é possibile usare diversi protocolli di sicurezza ma i due principali sono:

- **SSL:** utilizza un sistema di crittografia basato su chiave pubblica e privata. Lautenticazione tra client e server consente di stabilire l'identitá e l'autenticitá delle due entitá. Tramite firma digitale si punta a evitare modifiche non autorizzate dei dati.
- **TLS:** é un protocollo basato su SSL. I client adottano chiavi pubbliche fornite dal server per criptare un numero casuale. Quest'ultimo viene impiegato per garantire una chiave di sessione utilizzato per criptare il successivo scambio di messaggi. Gli URL che richiedono una connessione SSL cominciano per https invece che per http.

Sicurezza a livello di Trasporto

- Per ricevere ed inviare messaggi SOAP, il client di un Web Service dovrá instaurare una socket sicura attraverso il protocollo SSL.

Sicurezza a livello di Trasporto

- Per ricevere ed inviare messaggi SOAP, il client di un Web Service dovrá instaurare una socket sicura attraverso il protocollo SSL.
- Basandosi su un'autoritá di certificazione per autenticare il Web Service e il client, il protocollo si occupa di assicurare privacy e autenticazione.

Sicurezza a livello di Trasporto

- Per ricevere ed inviare messaggi SOAP, il client di un Web Service dovrá instaurare una socket sicura attraverso il protocollo SSL.
- Basandosi su un'autoritá di certificazione per autenticare il Web Service e il client, il protocollo si occupa di assicurare privacy e autenticazione.
- Utilizzando HTTPS, il quale opera con certificati a lato server, si dovrá implmentare un sistema di autenticazione da parte del client. Il meccanissmo piú basico é quello di associare a ogni client un nome utente e password univoco.

Limiti di SSL

- É un meccanismo di sicurezza *punto-punto*, mentre nei Web services si neccessita di un *end-to-end*;

Limiti di SSL

- É un meccanismo di sicurezza *punto-punto*, mentre nei Web services si neccessita di un *end-to-end*;
- Opera a livello trasporto e non messaggio una volta a destinazione i dati saranno registrati in chiaro (non sicuro);

Limiti di SSL

- É un meccanismo di sicurezza *punto-punto*, mentre nei Web services si neccessita di un *end-to-end*;
- Opera a livello trasporto e non messaggio una volta a destinazione i dati saranno registrati in chiaro (non sicuro);
- Siccome opera a livello trasporto, la granularitá offerta sará minore rispetto a quella offerta da un sistema basato su messaggio.

Limiti di SSL

- É un meccanismo di sicurezza *punto-punto*, mentre nei Web services si neccessita di un *end-to-end*;
- Opera a livello trasporto e non messaggio una volta a destinazione i dati saranno registrati in chiaro (non sicuro);
- Siccome opera a livello trasporto, la granularitá offerta sará minore rispetto a quella offerta da un sistema basato su messaggio.

Esempio: si potrebbe desiderare di criptare parti diverse del messaggio con chiavi differenti per distribuirlo a pi utenti, o criptare solo alcune sezioni lasciando in chiaro il grosso del messaggio, ma ció non fattibile se si lavora con il messaggio preso a scatola chiusa.

Sicurezza a livello di Messaggio

In alcuni meccanismi non é sempre possibile applicare una sicurezza a livello di trasporto e altre volte questa non si rivela sufficiente per fa fronte alle esigenze.

- Nel caso di ambienti popolati da intermediari diversi che operano sugli stessi messaggi o che necessitano
- in alcuni casi pu nascere la necessit di criptare campi diversi con chiavi diverse
- ...

In questi casi sorge la necessitá di nuove tecnologie di sicurezza che operano a livello di messaggio.

Sicurezza a livello di Messaggio

Come?

Una modalitá é quello di criptare il contenuto del messaggio con la chiave pubblica del destinatario, in modo che esso possa viaggiare normalmente, ma senza che il contenuto sia leggibile da soggetti che non siano i destinatari finali.

In questo modo peró si va ad impattare sulla struttura implementativa dei Web services.

Sicurezza a livello di Messaggio

Come?

Una modalitá é quello di criptare il contenuto del messaggio con la chiave pubblica del destinatario, in modo che esso possa viaggiare normalmente, ma senza che il contenuto sia leggibile da soggetti che non siano i destinatari finali.
In questo modo peró si va ad impattare sulla struttura implementativa dei Web services.

Come prima cosa é importante scegliere su quale aspetto implementare i meccanismi di sicurezza e quindi quale ambiente rendere maggiormente sicuro. Se stabilire regole che debbano avere impatto su WSDL oppure agire direttamente sul messaggio SOAP avendo un impatto piú significativo.

Sicurezza a livello di Messaggio

L'organizzazione dell'OASIS (*Organization for the Advancement of Structured Information Standards*) ha emanato due standard:

- **Web-security**: un estensione di SOAP che consente di stabilire regole di sicurezza a livello messaggio permettendo di sfruttare la granularitá dell XML;

Sicurezza a livello di Messaggio

L'organizzazione dell'OASIS (*Organization for the Advancement of Structured Information Standards*) ha emanato due standard:

- **Web-security**: un estensione di SOAP che consente di stabilire regole di sicurezza a livello messaggio permettendo di sfruttare la granularitá dell XML;
- **WS-SecurityPolicy**: estensione del WSDL che tramite una serie di policy permette di descrivere nel WSDL, quali meccanismi di sicurezza vengono implementati nei messaggi scambiati, con una ricchezza espressiva tale da avere un impatto non indifferente sui frameworks che implementano i Web services

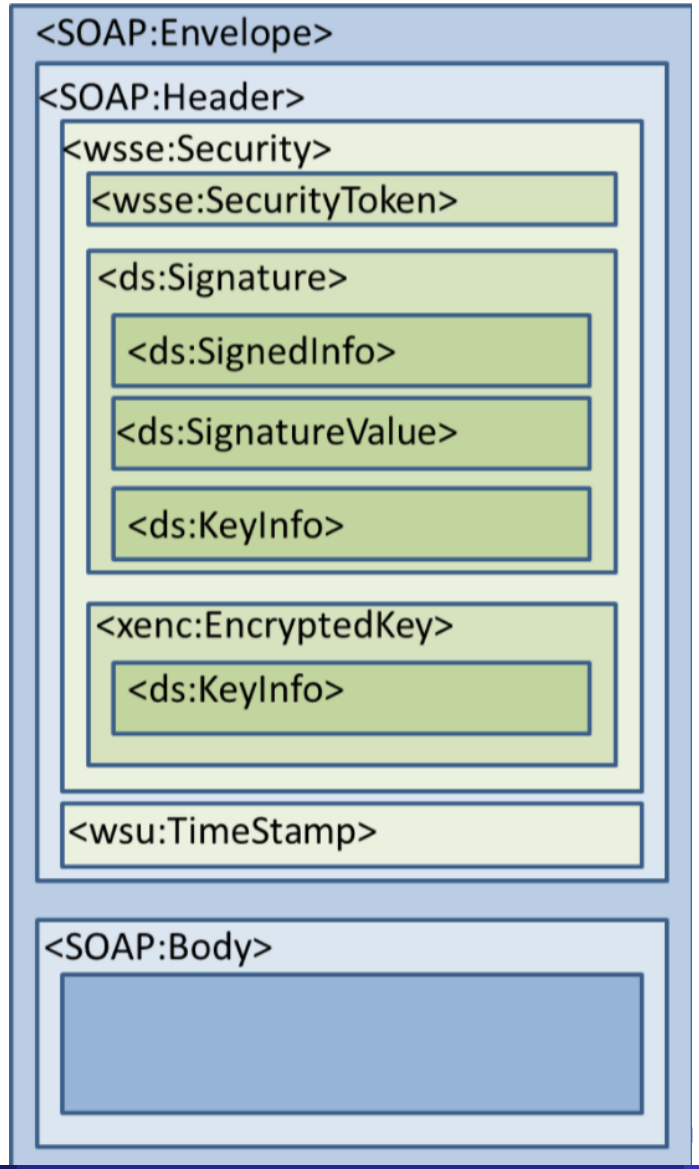
WS-Security

Web Services Security (WS-Security, WSS)

- *é una estensione di SOAP per estendere la sicurezza ai servizi web ed é una specifica redatta da OASIS*
- specifica come puó essere rafforzata l'integritá e la confidenzialitá e permette la comunicazione di vari token di sicurezza, come SAML, Kerberos e X.509
- il principale obiettivo é l'utilizzo della firma XML e della crittazione XML per garantire la sicurezza end-to-end.

Componenti logiche del WSS

Ogni messaggio SOAP sará composto da:



Componenti logiche del WSS

dove saranno presenti le diverse componenti logiche del WSS:

Sezione	Descrizione
Security Tokens	Sezione destinata a gestire in particolare le informazioni d'autenticazione
Message confidentiality	Sezione destinata a gestire le informazioni di cifratura del messaggio (*XML Encryption*)
Message integrity	Sezione destinata a gestire le informazioni di firma del messaggio o relative parti (*XML Signature*)
Security timestamp	Sezione destinata a gestire le informazioni temporali del messaggio, utili al destinatario a stabilire la validità dell'informazione contenuta nel messaggio

Security Tocken

Come illustrato nella slide precedente i **SecurityTockens** sono combinazioni di uno o piú *claim* (dichiarazione fatta da una entitá composta di solito racchiude una serie di parametri).

- Essi infatti contengono una serie di dichiarazioni che possono servire a fornire informazioni di autenticazione e, combinandoli con gli standard per la criptazione e la firma, possono permettere di instaurare confidenzialitá e integritá.

Tipologie

Secondo la specifica OASIS per WS-security esistono 3 diverse tipologie di Security-Tockens:

- **UserName Tokens**: dove lídentitá dellútente, composta solitamente da un username e una password;
- **Binary Security Tokens**: modalitá utilizzata per messaggi non formattati in XML e richiedono una particolare codifica per l'inclusione in messaggi XML;
- **XML Tokens**: base per utilizzare altri security tokens tramite in XML

UserName Tokens

Ogni volta che viene svolta una funzione i **SecurityTokens** passare un username e una password.

In questo modo:

- l'utente che richiede il servizio avrá la possibilitá di fornire la sua identitá e il ricevente potrá verificarne la validitá e scegliere come rispondere all'invocazione.
- Il server poi verificherá se il timestamp ancora valido, se lusername é previsto nel proprio repository, se la password relativa valida.

Password come STRINGA TESTUALE

La *password* a sua volta puó essere sotto forma di semplice **stringa testuale**, in questo caso il server utilizzerá l' attributo **Type**:

```
<wsse:UsernameToken>
 <wsse:Username>user</wsse:Username>
 <wsse:Password Type="wsse:PasswordText">password</wsse:Password>
</wsse:UsernameToken>
```

Password come FIRMA

Oppure sotto forma di **firma** garantendo in questo modo un maggior grado di sicurezza, grazie anche alla modalitá di "oscuramento" SHA1:

```
<wsse:UsernameToken>
    <wsse:Username>scott</wsse:Username>
    <wsse:Password Type="wsse:PasswordDigest">
        KE6QugOpkPyT3Eo0SEgT30W4Keg=</wsse:Password>
    <wsse:Nonce>5uW4ABku/m6/S5rnE+L7vg==</wsse:Nonce>
    <wsu:Created xmlns:wsu=
        "http://schemas.xmlsoap.org/ws/2002/07/utility">
            2002-08-19T00:44:02Z
    </wsu:Created>
</wsse:UsernameToken>
```

Password come FIRMA

dalla figura sopra riportata si possono notare alcuni attributi, tra cui:

- **PasswordDigest** che é la concatenazione del parametro **nonce** (un long da 16 bytes passato come valore codificato in base64), del tempo di creazione e della password.

Il server partendo dal tempo di creazione e dal nonce, oltre alla password memorizzata localmente per lo user, dovrebbe essere in grado di riottenere il valore esatto passato, confermando l hash.

Password come FIRMA

ATTENZIONE: Nonostante il maggior grado di difesa che questa tipologia di password offre rispetto a una piú semplice password testuale, la firma non é in grado di difendere il sistema da attacchi di tipo REPLAY.

- Spesso infatti é consigliabile combinare a questo tipo di meccanismo un *timestamp* sufficientemente breve cosí da prevenire simili attacchi.

Binary Security Tokens

Questa modalitá di autenticazione consiste semplicemente nel passare direttamente, tramite il security token, un certificato (come X.509).

- Dato che sono certificati non basati sullXML, vengono aggregati al messaggio dopo essere stati convertiti in binario.

Binary Security Tokens

Questa modalitá di autenticazione consiste semplicemente nel passare direttamente, tramite il security token, un certificato (come X.509).

- Dato che sono certificati non basati sullXML, vengono aggregati al messaggio dopo essere stati convertiti in binario.
- Questo genere di certificati presenta un grado di fiducia maggiore rispetto a un autenticazione basata su user e password.

Binary Security Tokens

Questa modalitá di autenticazione consiste semplicemente nel passare direttamente, tramite il security token, un certificato (come X.509).

- Dato che sono certificati non basati sullXML, vengono aggregati al messaggio dopo essere stati convertiti in binario.
- Questo genere di certificati presenta un grado di fiducia maggiore rispetto a un autenticazione basata su user e password.

```
<wsse:BinarySecurityToken
    ValueType="wsse:X509v3"
    EncodingType="wsse:Base64Binary"
    Id="SecurityToken-f49bd662-59a0-401a-ab23-1aa12764184f">MIIHdjCCB..
</wsse:BinarySecurityToken>
```

Binary Security Tokens VS Replay

Grazie alla combinazione dei diversi certificati implementati nel messaggio con la firma dell'utente stesso é possibile eliminare il rischio di attacchi di tipo replay.

Binary Security Tokens VS Replay

Grazie alla combinazione dei diversi certificati implementati nel messaggio con la firma dell'utente stesso é possibile eliminare il rischio di attacchi di tipo replay.

L'immagine che viene riportata ne illustra un esempio pratico:

```
<wsu:Timestamp>
    <wsu:Created
        wsu:Id="Id-3beeb885-16a4-4b65-b14c-0cfe6ad26800"
        >2002-08-22T00:26:15Z</wsu:Created>
    <wsu:Expires
         wsu:Id="Id-10c46143-cb53-4a8e-9e83-ef374e40aa54"
         >2002-08-22T00:31:15Z</wsu:Expires>
</wsu:Timestamp>
```

utilizzando un timestamp che indichi il tempo di creazione e il limite oltre il quale un determinato messaggio cessa di essere valido, é possibile stabile una politica in base alla quale scegliere se e quando eliminarlo.

FIRMA DIGITALE

La **Firma Digitale** é uno schema matematico per dimostrare l'autenticit di un messaggio o di un documento digitale inviato attraverso un canale non sicuro: una firma digitale valida garantisce:

- al destinatario che il mittente del messaggio sia chi dice di essere (autenticazione),
- che il mittente non possa negare di averlo inviato (non ripudio),
- che il messaggio non sia stato alterato lungo il percorso dal mittente al destinatario (integritá).

FIRMA DIGITALE

Come si é visto nella precedenti slide sono diversi i meccanismi di sicurezza che prevedono lo strumento della firma digitale:

- negli UserToken la firma viene basata sulla password
- i certificati abilitano gli utenti a porre la firma sui messaggi come chiave privata
- i binaryToken forniscono la chiave di sessione come un ticket che solo il destinatario é in grado di decifrare

CRITTOGRAFIA

La **Crittografia** é una dei requisiti fondamentali della sicurezza informatica. Garantendo la confidenzialitá dei dati, impedisce la realizzazione di diversi attacchi ai dati piú sensibili del sistema.

CRITTOGRAFIA

La **Crittografia** é una dei requisiti fondamentali della sicurezza informatica. Garantendo la confidenzialitá dei dati, impedisce la realizzazione di diversi attacchi ai dati piú sensibili del sistema.

Cosi anche nell' ambito del WSS, questa caratteristica ricompre uno ruolo essenziale. Dove a volte lautenticazione e la garanzia della non corruzione del messaggio non sono sufficienti, occorre garantire che il messaggio non sia letto.

CRITTOGRAFIA

La **Crittografia** é una dei requisiti fondamentali della sicurezza informatica. Garantendo la confidenzialitá dei dati, impedisce la realizzazione di diversi attacchi ai dati piú sensibili del sistema.

Cosi anche nell' ambito del WSS, questa caratteristica ricompre uno ruolo essenziale. Dove a volte lautenticazione e la garanzia della non corruzione del messaggio non sono sufficienti, occorre garantire che il messaggio non sia letto.

Tramite IXML Encryption, WS-Security consente di combinare la criptazione di blocchi e sottostrutture dellheader e del body di un messaggio SOAP facendo fronte a questa esigenza. Nello specifico vengono scelte chiavi trasportate nel messaggio e criptate a loro volta oppure condivise tra mittenti e destinatari.

CRITTOGRAFIA esempio

Nel caso in cui il mittente e destinatario utilizzino un unica chiave segreta si avrá:

```
<S11:Header>
  <wsse:Security>
    <xenc:ReferenceList>
      <xenc:DataReference URI="#bodyID"/>
    </xenc:ReferenceList>
  </wsse:Security>
</S11:Header>
<S11:Body>
  <xenc:EncryptedData Id="bodyID">
    <ds:KeyInfo>
      <ds:KeyName>CN=Mario Rossi, C=IT</ds:KeyName>
    </ds:KeyInfo>
    <xenc:CipherData>
      <xenc:CipherValue>...</xenc:CipherValue>
    </xenc:CipherData>
  </xenc:EncryptedData>
</S11:Body>
```

Da WS-Security, migliorando..

Nelle ultime slide si é compreso come l'estensione del protocollo SOAP, WS-Security, abbia lo scopo principale di garantire la sicurezza del Web Service senza imbattere il WSDL.
Questo puó essere allo stesso tempo un vantaggio e uno svantaggio, in quanto un utente che consulta il WSDL non é in grado di conoscere le regole di sicurezza scelte e quindi di conseguenza diventa piú complicato apportare modifiche che ottimizzino il sistema.

Da WS-Security, migliorando..

Per questo motivo sono stati sviluppati una serie di standard da affiancare al WSS:

- **WS-Trust**: standard OASIS che fornisce unestensione di WS-Security specifica per le tematiche che ruotano attorno ai security tokens, volta a garantire i processi utili a stabilire una relazione di fiducia tra utenti e providers di servizi.
- **WS-Policy**: specifica che permette ai Web services di usare IXML per descrivere e condividere le politiche adottate (politiche di sicurezza, quality of services etc.) e ai clients di descrivere le politiche richieste.
- **WS-SecureConversation**: specifica creata da IBM e altri che lavora assieme a WS-Security, WS-Trust e WS-Policy, per abilitare la creazione e la condivisione di contesti di sicurezza. Lo scopo di WS-SecureConversation quello di stabilire contesti di sicurezza per scambi multipli di messaggi SOAP, riducendo loverhead derivante dallo scambio delle chiavi attraverso linstaurazione di chiavi di sessione.

WS-SecurityPoliciy

A questi ne é stato aggiungo uno:

WS-SecurityPolicy:

una specifica creata da IBM e divenuta standard OASIS che estende alcuni protocolli per Web services in ambito sicurezza (WS-Security, WS-Trust e WS-SecureConversation).

- WS-SecurityPolicies fornisce un modo semplice per configurare e controllare i requisiti di sicurezza estendendo il WSDL in modo da poter descrivere direttamente in esso le politiche di sicurezza adottate da un Web service.

WS-Security VS WS-SecurityPolicy

WS-Security é estensione del SOAP quindi dei messaggi che viaggiano

WS-SecurityPolicy é estensione del WSDL quindi non solo di applicare ai messaggi SOAP le regole adottate ma di renderle visibili anche a chi volesse consultare il WSDL.

WS-Security VS WS-SecurityPolicy

WS-Security é estensione del SOAP quindi dei messaggi che viaggiano

WS-SecurityPolicy é estensione del WSDL quindi non solo di applicare ai messaggi SOAP le regole adottate ma di renderle visibili anche a chi volesse consultare il WSDL.

- esso fornisce un modo piú semplice e piú basato su standards per configurare e controllare i requisiti di sicurezza, permettendo a svariati tool e framework come .NET o CXF di auto configurarsi semplicemente per interoperare con i servizi esposti.

WS-Security VS WS-SecurityPolicy (1)

- il binding e/o le operazioni del WSDL possono far riferimento a sezioni WS-Policy che descrivono i requisiti di sicurezza di base per poter interagire con il Web service. É indicare aspetti quali chiavi simmetriche o asimmetriche, utilizzo del protocollo HTTPS, parti da criptare o firmare, se includere il timestamp o meno e cos via.
- tra tutte le azioni esercitate per interagire in modo sicuro, non tutti gli aspetti vengono specificati nelle policies adottate, ad esempio non viene descritto dove trovare keystores, username e password e altre informazioni comunque utili a run-time, cos da garantirne la riservatezza.

Esempio di WS-SecurityPolicy

```
xmlns:wsaws="http://www.w3.org/2005/08/addressing"
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">

<types> … </types>
<message name="PresentazioneReq">   …  </message>
<message name="PresentazioneResp">  … </message>
<portType name="ServizioPresentazioneIF">  … /portType>

<binding name="ServizioPresentazioneImplPortBinding" type="tns:ServizioPrese
  <wsp:PolicyReference URI="#SecurityServiceBindingPolicy" />
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/
    <operation name="Presentazione"> … </operation>
</binding>

<wsp:Policy wsu:Id="SecurityServiceBindingPolicy">
  <wsp:ExactlyOne>
    <wsp:All>
      <sp:SupportingTokens xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-secu
        <wsp:Policy>
          <sp:UsernameToken sp:IncludeToken="http://docs.oasis-open.org/ws-s
            <wsp:Policy>
              <sp:WssUsernameToken10/>
            </wsp:Policy>
          </sp:UsernameToken>
        </wsp:Policy>
      </sp:SupportingTokens>
    </wsp:All>
  </wsp:ExactlyOne>
</wsp:Policy>

<service name="PresentazioneService">  … </service>
```

Esempio di WS-SecurityPolicy

Come si enuncia dall' immagine precende, si andrá ad operare sul WSDL inserendo ad esempio un nuovo elemento *Policy*, descrivendo le policies che si stanno aggiungendo.

In questo semplice esempio si tratta di una policy basata su UsernameToken, descritta da una proprietá del tipo *AlwaysToRecipient*, essa stabilisce che il client ha lobbligo di specificare il token nellheader.

- Si avrá un token che nel caso descritto sará in chiaro. Segue quindi lassociazione della policy al binding per mezzo dellID della policy. Nota particolarmente interessante é che per il WSDL non servirá fare altro, nemmeno ricompilarlo. Sará infatti cura dei frameworks gestirne gli aspetti relativi.

Test Finali

Come ultima fase sará importante costruire dei test e sottoporli all' esecuzione su server precedentemente configurati appositamente, attenderne il corretto avvio e invocando i test del servizio con il client impostato sui parametri (protocollo, host, porto) corretti.

In questa fase sará importante concentrasi su **quanto il messaggio inviato dal client stia viaggiando verso il server**, quindi controllare quando descritto nell header di sicurezza e relative regole di implementate nel messaggio.

Test Finali

Lato client sará utile la presenza di un WSDL che contempli le policies di sicurezza adottate lato server, cosí da avere la possibilitá, ad esempio di applicare controlli di sicurezza che vadano a verificare le regole di sicurezza implementate evitando di far partire messaggi che non le rispettino (ed evitando di conseguenza, che messaggi non conformi viaggino sulla rete solo per essere rifiutati).

NB. Per ottimizzare..

..ed evitare di dover mettere mano al WSDL ogni volta che si modificano le policies, é possibile descrivere queste ultime in documenti esterni richiamati poi nel WSDL (come descritto nello standard *Web Service Policy 1.5 Attachment*).

Indice

Esempi di tecnologie WS sicuri nell'ambito di IoT

Diversi sono gli esempi di Web Service e di WS applicati nell'ambito dell'Iot (come quelli sopra citati).

Per approfondire il tema della sicurezza di questi servizi é stato preso in esame la nuova tecnologia delle reti **Blockchain**.

Blockchain é

l'applicazione di una tecnologia in cui una lista di record, o blocchi, sono collegati l'uno all'altro tramite timestamp e sono resi sicuri mediante l'uso della crittografia.

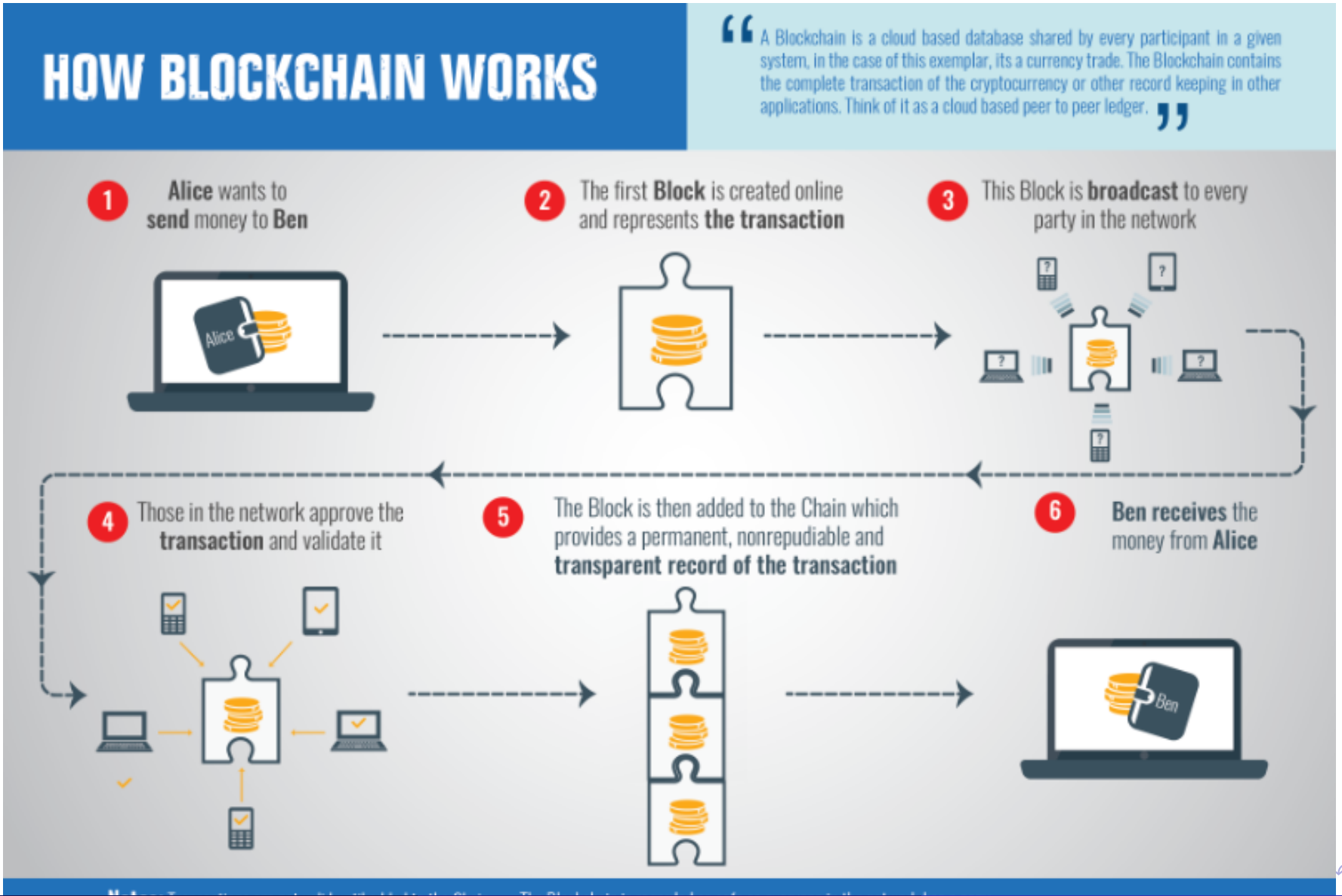
- Le blockchain sono resistenti alla modifica dei dati poiché l'alterazione di un blocco richiede il consenso attraverso la catena o il libro mastro registrati.

Oggi giorno questa tecnologia innovativa viene sfruttata principalmente in campo finanziario come infrastruttura tecnica per gestire le numerose transizioni online relative all'uso dei Bitcoin (innovative criptomonete/criptovalute).

- La figura nella prossima slide spiega come lavora il meccanismo della tecnologia nello specifico.

Blockchain

Esempi di tecnologie WS sicuri



Blockchain

Negli ultimi anni sono state diverse le societá che hanno voluto approfondire questa nuova tecnologie.

- Aziende e banche come Amazon, Google, IMB ecc.. in collaborazione di grandi istituzioni finanziarie come Mastercard, Visa, il Nasdaq e Western Union stanno progettando protocolli per il processing sicuro e rapido di decine di migliaia di transazioni allinterno di una rete Blockchain.

Perché utilizzare una Blockchain?

Un ledger crittograficamente sicuro, condiviso e distribuito

I principali vantaggi di questa particolare tecnologia infatti sono:

- **Ledger**:é un database WORM (Write Once, Read Many) come record non modificabile di ogni transazione. Se viene introdotto un errore, é necessario inserire una transazione di compensazione per correggerlo, dato che l'aggiornamento e l'eliminazione non sono consentiti.

Perché utilizzare una Blockchain?

Un ledger crittograficamente sicuro, condiviso e distribuito

I principali vantaggi di questa particolare tecnologia infatti sono:

- **Ledger**:é un database WORM (Write Once, Read Many) come record non modificabile di ogni transazione. Se viene introdotto un errore, é necessario inserire una transazione di compensazione per correggerlo, dato che l'aggiornamento e l'eliminazione non sono consentiti.
- **Sicurezza crittografica**: Una blockchain applica una tecnologia di firma digitale autentica e collaudata per creare transazioni che riducono le frodi, garantendo affidabilitá e attendibilitá.

Perché utilizzare una Blockchain?

Un ledger crittograficamente sicuro, condiviso e distribuito

I principali vantaggi di questa particolare tecnologia infatti sono:

- **Ledger**:é un database WORM (Write Once, Read Many) come record non modificabile di ogni transazione. Se viene introdotto un errore, é necessario inserire una transazione di compensazione per correggerlo, dato che l'aggiornamento e l'eliminazione non sono consentiti.
- **Sicurezza crittografica**: Una blockchain applica una tecnologia di firma digitale autentica e collaudata per creare transazioni che riducono le frodi, garantendo affidabilitá e attendibilitá.
- **Condivisione**:Le blockchain acquistano valore grazie alla condivisione. Maggiore é il numero di organizzazioni e di societá partecipanti, anche se in concorrenza, piú uniforme sará il processo e maggiore il valore.

Perché utilizzare una Blockchain?

Un ledger crittograficamente sicuro, condiviso e distribuito

I principali vantaggi di questa particolare tecnologia infatti sono:

- **Ledger**:é un database WORM (Write Once, Read Many) come record non modificabile di ogni transazione. Se viene introdotto un errore, é necessario inserire una transazione di compensazione per correggerlo, dato che l'aggiornamento e l'eliminazione non sono consentiti.
- **Sicurezza crittografica**: Una blockchain applica una tecnologia di firma digitale autentica e collaudata per creare transazioni che riducono le frodi, garantendo affidabilitá e attendibilitá.
- **Condivisione**:Le blockchain acquistano valore grazie alla condivisione. Maggiore é il numero di organizzazioni e di societá partecipanti, anche se in concorrenza, piú uniforme sará il processo e maggiore il valore.
- **Sistema distribuito**:Maggiore é il numero di repliche, piú autentico sará il ledger.

"Blockchain as a service"

Grazie alle diverse applicazioni che sono state sviluppate fino ad oggi e quelle che sono in fase ancora progettuale é possibile qualificare la blockchain come **blockchain as a service**.

La sua potenzialitá infatti consente lo sviluppo di numerose applicazioni, anche per l erogazione di servizi diversi e non necessariamente legati strettamente al mercato finanziario online.

Una delle evoluzioni future potrebbe essere quello di approfondire lo sviluppo della tecnologia nel settore della **Industry 4.0** come supporto alle applicazioni che sfruttano l' Intelligenza Artificiale (AI). Questo porterá a ulteriori enormi potenzialitá sia per le industrie sia per gli utenti che fruiranno dei servizi erogati.