Internet of Things: protocolli e criticità nella sicurezza di rete (Seconda parte)

Sommario

Oggi quando si parla di Internet si tende a considerare la rete come un insieme informe di connessioni e contenitori di informazioni. Il web rappresenta la biblioteca, l'archivio video e l'album fotografico personale del XXI secolo. Con la nascita e la diffusione di Internet of Things, la rete Internet non è solo il mondo delle ricerche e delle informazioni, ma diventa teatro d'azione, di attività di droni comandati da remoto. In tale contesto si comprende che la rete comincia a subire dei rischi notevoli, soprattutto da parte di hacker che potrebbero utilizzare l'Internet of Things come la più potente fabbrica del male. In questa seconda parte della trattazione vediamo quali sono state le evoluzioni degli standard, come si intreccia oggi il mondo dei sensori con Internet e un approccio al mondo della sicurezza per difenderci da eventuali attacchi provenienti dal cyberspazio.

Lo standard IEEE 802.15.4a

Nella prima parte della dissertazione è stato presentato lo standard IEEE 802.15.4. Ne vediamo ora un'evoluzione: lo standard IEEE 802.15.4a [a].

Questo standard si arricchisce sul livello fisico (livello 1) definendo due diversi strati fisici. Il primo, chiamato Chirp Spread Spectrum (CSS), offre una velocità di trasmissione dati fino a 2 Mbps e offre una maggiore efficienza nella gestione

delle interferenze. Il secondo è basato su UWB (Ultra Wide Band). Questa tecnologia è sviluppata per trasmettere e ricevere segnali attraverso l'uso di impulsi a radiofrequenza con durata estremamente ridotta. Ha una velocità di trasmissione dati massima di 10 Mbps entro un'area di copertura inferiore a 10 metri. In ogni caso entrambe le modalità non sono compatibili con lo standard IEEE 802.15.4.

Standard IEEE 802.15.4e

Altro standard che nel 2012 fu definito dal Task Group 4e dell'IEEE, fu l'IEEE 802.15.4e.

Le novità sono:

- più bassi assorbimenti di energia;
- più elementi informativi del frame;
- struttura dei *beacons* più complessa;
- frame multipurpose (per ogni tipo di ambiente applicativo);
- misure di performance del livello MAC;
- associazione più veloce dei dispositivi;

Sono state migliorate ed incrementate le funzioni al livello MAC:

- identificazione a radiofrequenza (BLINK): identificazione e localizzazione di oggetti e persone;
- AMCA (Asynchronous Multi-Channel Adaptation): funzionalità che consente l'implementazione di domini di applicazione complessi (ad es. gestione del processo, automazione e controllo, monitoraggio dell'infrastruttura, ecc.);
- DSME (Deterministic and Synchronous Multi-channel Extension): funzionalità di supporto agli ambienti industriali e commerciali con rigidi requisiti di tempestività e affidabilità;
- LLDN (Low Latency Deterministic Network): funzionalità

di supporto ad applicazioni real-time che richiedono requisiti di latenza molto bassi (ad es. Automazione di fabbrica, controllo robot);
- l'innovazione più importante è il TSCH (Time Slotted Channel Hopping), un meccanismo ad accesso multiplo in grado di fornire una maggiore robustezza alle interferenze elettromagnetiche ed una riduzione della probabilità della collisione. È una combinazione tra l'accesso a time-slot, combina l'accesso a fessure temporali, la comunicazione multi-channel ed il channel-hopping.

Protocollo ZigBee

ZigBee è uno dei principali protocolli di comunicazione, edito dalla ZigBee Alliance, nato per le reti wireless a bassa potenza e a bassa velocità di trasmissione dati [b] [c]. È uno standard per il monitoraggio flessibile e il controllo degli apparecchi negli ambienti domestici e commerciali.

Zigbee utilizza le specifiche MAC/Phy IEEE 802.15.4 (Fig.3). L'ultima versione, la 3.0, è stata creata per facilitare la scelta e l'utilizzo del prodotto da parte dell'utente finale nelle varie implementazioni. La connessione delle reti ZigBee 3.0 al protocollo IP ha permesso il decollo dell'IoT. Le *isole* di sensori ed attuatori sono oggi connesse ad Internet: il monitoraggio e la gestione remota sono attività possibili grazie al connubio ZigBee e mondo IP. Il mondo delle app degli smartphone e tablet si rivolge sempre più verso la gestione di questi dispositivi. I dispositivi IoT vengono quindi corredati di app per smartphone e tablet.

Fig. 3 Zigbee Protocol Stack (RF Wireless World)

Associazione sicura tra dispositivi IoT

La letteratura presentata in questo articolo mostra punti di forza e debolezza delle soluzioni più importanti per implementare un'associazione sicura tra i dispositivi nell'IoT. Parleremo ora degli approcci più noti per progettare sistemi di scambio di chiavi e il modo migliore per implementare la sicurezza.
Il Servizio di Gestione delle Chiavi, KMS (Key Management System) è il centro di sicurezza del sistema basato sulle chiavi. Questo sistema definisce come generare le chiavi, come scambiarle, come memorizzarle e come sostituirle. Il KMS include la progettazione del protocollo crittografico, l'implementazione del server principale, le procedure utente e altri protocolli rilevanti [d]. Con l'Internet of Things, il KMS diventa l'elemento principale per la sicurezza dei dispositivi connessi.

Tipi di chiave

I sistemi crittografici utilizzano diversi tipi di chiavi: sistemi a chiave simmetrica e sistemi a chiave asimmetrica. In un algoritmo a chiave simmetrica, la stessa chiave è utilizzata per crittografare e decrittografare un messaggio. Le chiavi devono essere scelte con cura, distribuite e archiviate in modo sicuro. In un sistema di chiavi asimmetriche sono utilizzate due chiavi diverse, ma sono collegate matematicamente l'una all'altra. La chiave utilizzata per crittografare è diversa dall'altra utilizzata per decrittografare. Ogni dispositivo ha una coppia di chiavi (una chiave pubblica e una chiave privata) associate. La chiave pubblica deve essere distribuita sulla rete in modo sicuro, mentre quella privata deve essere consegnata su un canale diverso e ancora più sicuro.

Algoritmi di scambio chiave

L’approccio più semplice è una chiave pubblica scambiata prima della trasmissione dei dati tra i dispositivi. Se usiamo solo una chiave pubblica tra i dispositivi, abbiamo lo svantaggio che se questa chiave diviene sfortunatamente nota, rimuoverebbe la sicurezza dall’intero sistema. Dall’altra parte, se utilizziamo un paio di chiavi per ogni coppia di dispositivi, il sistema non diventa più scalabile, perché le reti IoT sono caratterizzate da un numero elevato di dispositivi [e]. Inoltre, per ogni coppia di chiavi, dobbiamo sempre garantire la riservatezza, l’integrità, la disponibilità e l’autenticazione della fonte [f].

Meccanismi più sofisticati possono risolvere questi tipi di problemi. La maggior parte delle tecniche si basa sull’algoritmo Diffie-Hellman (DH) del 1975 e sugli approcci con l’algoritmo ECDH (Algoritmo di Diffie-Hellman con le curve ellittiche) [g]. Il primo algoritmo è particolarmente affidabile perché è difficile risolvere il problema del logaritmo discreto. Il secondo, ECDH, tuttavia, sfrutta lo scambio di chiavi attraverso la primitiva tipica della “crittografia a curva ellittica” (ECC), che offre lo stesso livello di sicurezza dell’algoritmo DH ma con una lunghezza della chiave significativamente più piccola.

Algoritmo Diffie-Hellman e ECDH, Elliptic Curve Cryptography (ECC)

La caratteristica principale che un sistema a chiave pubblica/privata deve fornire è che non si deve poter ricavare una chiave dall’altra.

Nella teoria di Diffie-Hellman l’algoritmo di crittografia è chiamato $E(\cdot)$ e quello di decrittografia $D(\cdot)$.

L’algoritmo deve soddisfare due condizioni:

Se M è il messaggio

- $D\ (E(M)) = M$;
- È molto difficile calcolare $D(\cdot)$ da $E\ (\cdot)$

Questo algoritmo funziona come segue, vediamo l'esempio.

Alice, che desidera ricevere messaggi segreti da Bob, implementa due funzioni $E\ (\cdot)$ e $D\ (\cdot)$, parametrizzate rispettivamente dalla chiave di crittografia e dalla chiave di decrittografia, che soddisfano i suddetti requisiti. La chiave di crittografia è resa pubblica, mentre quella di decrittografia viene tenuta segreta da Alice. Se Bob vuole inviare un messaggio M ad Alice, deve ottenere la chiave pubblica di Alice, calcolare C = E(M), che è il suo messaggio crittografato, e inviarlo ad Alice. Alice, dopo aver ricevuto un messaggio, usa la sua chiave privata per decifrare il messaggio originale M = D (C).

Pertanto, la crittografia a chiave pubblica richiede che l'utente abbia due chiavi: una chiave pubblica utilizzata per crittografare i messaggi e una chiave privata per decrittografare i messaggi ricevuti. Queste due chiavi devono essere correlate tra loro, ma deve essere particolarmente complicata e difficile da calcolare la chiave privata da quella pubblica. Questa difficoltà si basa principalmente su problemi matematici.

I problemi matematici più comuni su cui si basa un sistema crittografico a chiave pubblica sono:

- problema di fattorizzazione degli interi (IFP): dato un numero intero $n = p \cdot q$, con p e q numeri primi grandi, trova p e q. Il sistema RSA sfrutta questo problema;
- Discrete Logarithm Problem (DLP): calcola il logaritmo x di un elemento $b=a^x$ in un campo finito. Il sistema DSA (Digital Signature Algorithm) si basa su questo problema;

- problema del logaritmo discreto della curva ellittica: è una forma generalizzata di DLP sui punti di una curva ellittica. Questa categoria include l'algoritmo di Diffie-Hellman sulle Curve Ellittiche (ECDH).

Conclusione

Rispetto ad un sistema di crittografia simmetrica, un sistema di chiavi asimmetriche presenta prestazioni peggiori perché è più lento, ma fornisce la soluzione migliore per il problema di distribuzione delle chiavi. Date le enormi differenze tra i sistemi utilizzati e il relativo problema matematico, il confronto e l'analisi sono basati essenzialmente sulla dimensione della chiave. Il giusto compromesso è trovare la corretta *dimensione* della chiave per un livello di sicurezza accettabile dell'intero sistema. Questo concetto è particolarmente importante in IoT, in quanto, trattandosi di reti con un basso bit-rate, la trasmissione di un pacchetto di dati di dimensioni ridotte migliora l'efficienza dell'intero sistema. È anche utile considerare che, a seconda della situazione, una chiave eccessivamente grande può influire negativamente sulle prestazioni di un algoritmo ed una troppo piccola può portare allo sviluppo di un sistema non sicuro.

Pertanto, è necessario determinare il livello minimo della dimensione della chiave che garantisce un livello sufficiente di sicurezza. Il confronto può anche essere fatto in relazione ad un altro fattore: il tempo di esecuzione. I problemi IFP e DLP sono sostanzialmente equivalenti, quindi sono paragonabili per sicurezza e tempo di esecuzione. Il problema del logaritmo discreto su curve ellittiche (ECDLP) è invece considerevolmente più complesso ma più sicuro di IFP e DLP. Se abbiamo la stessa lunghezza della chiave, un sistema basato su ECDLP è sostanzialmente più sicuro dei sistemi RSA/DSA. È quindi naturale concludere che ECDLP fornisce un miglioramento in termini di prestazioni e sicurezza nell'ambiente IoT.

Riferimenti

- [a] Eirini karapistoli, Fotini-Niovi Pavlidou, Ioannis Gragopoulos. *An overview of the IEEE 802.15.4a Standard* **DOI:** 10.1109/MCOM.2010.5394030
- [b] http://www.zigbee.org/
- [c] C. Wang, T. Jiang, and Q. Zhang, ZigBee Network Protocols and Applications. Auerbach Publications, 2014.
- [d] Turner, Dawn M. “What Is Key Management? A CISO Perspective”. Cryptomathic. Retrieved (Feb. 2017).
- [e] Eschenauer, L., and Gligor, V. D. A Key-management Scheme for Distributed Sensor Networks. In Proc. of the ACM Conf. on Computer and Commun. Security (2002), ACM.
- [f] Barker, Elaine; Smid, Miles; Branstad, Dennis; Chokhani, Santosh. “NIST Special Publication 800 -130: A Framework for Designing Cryptographic Key Management Systems”. National Institute of Standards and Technology.
- [g] Mirzadeh, S., Cruickshank, H., and Tafazolli, R. Secure Device Pairing: A Survey. IEEE Commun. Surveys Tuts. 16, 1 (First 2014).

A cura di: **Crescenzio Gallo, Michele Perilli, Michelangelo De Bonis**
Università di Foggia

Il dott. Crescenzio Gallo ha conseguito la laurea in Scienze dell’Informazione presso l’Università di Bari (1978) e la laurea specialistica in Scienze e Tecnologie Informatiche presso l’Università di Milano (2007).

Negli anni 1978–1980 ha svolto attività di ricerca in Sistemi Informativi presso l’Istituto di Informatica dell’Università di Bari ed è stato consulente tecnico-scientifico della società Telespazio SpA di Roma per i progetti satellitari TERRA-Landsat della NASA e dell’Agenzia Spaziale Europea (ESA).

Dal 1992 al 2003 è stato docente a contratto presso le Università di Bari e Foggia, e dal 2004 al 2007 è stato ricercatore di ruolo in Matematica Applicata presso l'Università di Foggia.

Dal 2007 è Professore Aggregato di Informatica e Sistemi di Elaborazione delle Informazioni, ed attualmente svolge la sua attività di ricerca e docenza presso il Dipartimento di Medicina Clinica e Sperimentale dell'Università di Foggia.

I suoi interessi scientifici primari includono l'analisi dei dati e il machine learning applicati alla bioinformatica, e l'analisi delle problematiche di sicurezza delle reti.

L'Ing. Michele Perilli è nato a Foggia, nel 1962. Si è laureato in Informatica presso l'Università degli Studi di Bari nel 1987, e in Scienze e Tecnologie dell' Informazione (2005) e Tecnologie dell'Informazione e della Comunicazione (2006) presso l'Università di Milano.

Nel periodo 1988-2000 ha lavorato in Telecom Italia SpA con compiti di direzione tecnica. È docente di ruolo di Informatica dal 2001, professore a contratto presso l'Università di Foggia dal 2007. È membro IEEE e istruttore certificato Cisco Systems.

Il suo principale interesse nella ricerca riguarda il networking, i protocolli e il routing delle reti.

https://www.ictsecuritymagazine.com/articoli/internet-of-things-protocolli-e-criticita-nella-sicurezza-di-rete-seconda-parte/