Appunti di Cybersecurity

Andrea Bellu

20 ottobre 2025

1 Introduzione alla Cybersecurity

1.1 Perché Cybersecurity? Il Contesto di Industria 4.0

Il bisogno di cybersecurity è cresciuto esponenzialmente con l'arrivo dell'**Industria 4.0** (ora "Impresa 4.0"). Questo paradigma si basa su tecnologie abilitanti che interconnettono l'intera filiera produttiva.

- Questa interconnessione (es. feedback dalle vendite alla produzione) crea **molte nuove possibilità per fare danni**.
- L'elemento fondamentale è la necessità di **meccanismi per garantire la sicurezza nelle comunicazioni**.

1.2 Esempi di CyberThreats

Le minacce informatiche (cyberthreats) dimostrano la necessità di protezione in vari ambiti:

- **Robot Industriali**:
  - Attacco tramite smartphone infetto che, una volta nella rete aziendale, si finge il repository per gli aggiornamenti.
  - Il robot scarica così codice malevolo.
  - **Problema di fondo:** Rete aziendale per i client non separata dalla rete di produzione.
  - **Danno subdolo:** Modificare il movimento di pochi mm, creando migliaia di pezzi difettosi.
  - **Stuxnet**:
    - Virus sviluppato da governi (US contro Iran).
    - È entrato nella rete isolata della centrale nucleare di Natanz tramite una **chiavetta USB**.
    - **Obiettivo:** Attaccare i PLC di controllo delle centrifughe (Siemens).
  - **Domino's Pizza**:
    - L'app per smartphone gestiva tutto il processing, **senza un double-check lato server**.

    * Era possibile inviare un ordine alla produzione senza aver pagato.
    * **Lezione:** L'app deve essere un'interfaccia, il processing va fatto sul server.
- **Chipset Bluetooth Broadcom**:
    * È stato scoperto un comando di debug via wireless.
    * Esempio di fallimento della "Security by Obscurity".
- **WhatsApp (CVE-2019-3568)**:
    * Vulnerabilità di tipo **Buffer Overflow** durante l'instaurazione di una sessione VoIP.
    * Inviando un messaggio "opportuno" (più lungo del buffer), l'attaccante poteva sovrascrivere altre parti di memoria.
    * **Danno:** Installare software simile a Pegasus per spiare l'utente (camera, microfono, ecc.).
- **Virgin Media O2**:
    * Configurazione errata del software IMS (IP Multimedia Subsystem).
    * Informazioni sensibili (Cell ID, IMSI/IMEI) venivano incluse negli **header del protocollo SIP**.
    * **Danno:** Permetteva a un attaccante di mappare la posizione geografica degli utenti.

1.3 Il fattore umano e la convergenza delle reti

Molti problemi di sicurezza sono legati alla "pigrizia" (*laziness*):

- Protocolli senza autenticazione.
- Paradigma "security by obscurity" (es. GSM).
- Password fisse (magari appuntate su una lavagna).
- Mancanza di budget/tempo per aggiornare hw/sw (WannaCry docet).

2 Reti a Pacchetti e Convergenza

2.1 Concetti Base delle Reti a Pacchetti

- Una rete a pacchetto è un insieme di nodi connessi da canali, dove l'informazione è divisa in pacchetti.
- Internet è un'unione di sottoreti tra loro interconnesse.
- **Concetto Chiave:** Nello schema originale di Internet, i nodi intermedi non offrono **nessun servizio di sicurezza**.
- Tutta la sicurezza deve essere gestita **dai nodi terminali** (principio "end-to-end").

2.2 Evoluzione e Convergenza delle Reti

- **Reti LAN:** Si è passati da reti cablate (switch) a reti wireless (Access Point). Questo introduce il rischio di "ascolto" (*eavesdropping*).
- **Reti Industriali:** Si è passati da controlli punto-punto (cablaggio complesso) a Bus di campo (Fieldbus) e infine a **sistemi a pacchetto** (es. Profinet, Wi-Fi).
- **Convergenza:** Oggi, i problemi di sicurezza sono gli stessi ovunque (casa, azienda, produzione) a causa dell'uso delle stesse tecnologie di rete.

3 Meccanismi e Pilastri della Sicurezza

3.1 I Servizi di Sicurezza (Pilastri)

Per proteggere le comunicazioni servono "servizi" specifici:

1. **Autenticazione:** Identificare *chi* partecipa alla comunicazione (utente, nodo, applicazione) chiedendo una "prova".
2. **Controllo di Accesso:** Verificare i *diritti* di un partecipante (già autenticato) ad accedere a una risorsa. È *successivo* all'autenticazione.
3. **Confidenzialità:** Garantire che solo chi è autorizzato possa *leggere* le informazioni (sia in transito che memorizzate).
4. **Integrità (Integrity Protection):** Garantire che chi non è autorizzato non possa *modificare* l'informazione senza essere scoperto.
5. **Non Ripudio:** Garantire che un'entità non possa *negare* in seguito di aver generato un'informazione o partecipato a un processo.

3.2 Crittografia

- La crittografia è il "blocco fondamentale" per ottenere meccanismi di sicurezza "forti".
- Include sistemi a chiave simmetrica (private key), asimmetrica (public key), Hash e MAC.
- **Principio di Kerckhoffs:** La sicurezza di un sistema deve dipendere dalla **segretezza della chiave**, e non dalla segretezza del sistema/algoritmo (come invece si pensava per la macchina Enigma).

3.3 Tipi di Attacchi

- **Attacchi Passivi:** L'attaccante può solo catturare e analizzare i dati (es. intercettazione).
- **Attacchi Attivi:** L'attaccante può ricevere, *modificare* e re-immettere i dati in rete.

4 Conclusioni: Relatività e Standard

- **La Sicurezza Assoluta Non Esiste:** È sempre *relativa* all'ambito applicativo e al *valore* di ciò che si protegge.
- **Compromesso:** Ogni meccanismo è un compromesso tra costo, complessità e livello di protezione.
- **Sicurezza di Sistema vs. di Rete:** La prima riguarda il singolo nodo (HW, SW, OS), la seconda la comunicazione *tra* i nodi (protocolli).
- **Normative (EU):** Dal 2022 in Europa è presente la **NIS2** (Network and Information Security 2). Ha l'obiettivo di aumentare il livello di sicurezza in settori critici (energia, sanità, PA, ecc.) imponendo obblighi di gestione del rischio.
- **Framework:** Il **NIST Cybersecurity Framework (CSF)** è un esempio di approccio sistematico per gestire i rischi (e può essere usato per adeguarsi alla NIS2).

5 Modelli a Strati e Architetture di Rete

5.1 Comunicazione Logica vs. Fisica

Quando due sistemi comunicano (es. una webcam e uno smartphone), si possono identificare due tipi di comunicazione:

- **Comunicazione Logica:** La comunicazione "diretta" percepita tra i due dispositivi finali (la webcam invia immagini allo smartphone).
- **Comunicazione Fisica:** Il percorso reale che i dati compiono attraverso la rete, saltando da un nodo intermedio (router, switch) all'altro.

5.2 Pacchettizzazione e Indirizzamento

Per funzionare, la comunicazione si basa su due problemi fondamentali:

1. **Pacchettizzazione:** La sequenza di dati prodotta da un'applicazione (es. un video) è troppo grande per essere inviata in un blocco unico. Viene "spezzettata" in blocchetti più piccoli, detti **pacchetti**.

2. **Indirizzamento:** Ogni pacchetto deve contenere le informazioni per raggiungere sia il *nodo* (computer) corretto, sia l'applicazione corretta all'interno di quel nodo.

Questa divisione in problemi viene gestita da un **modello a strati** (o stack protocollare), dove ogni livello risolve un sotto-problema, aggiungendo le proprie informazioni di controllo (header).

5.3 Il Modello a Strati (Stack Protocollare)

L'approccio "divide et impera" della rete. I due modelli principali sono OSI (astratto) e TCP/IP (in uso).

- **Comunicazione Fisica (Verticale):** Su un singolo nodo, ogni livello interagisce solo con il livello immediatamente superiore o inferiore.
- **Comunicazione Logica (Orizzontale):** Ogni livello (es. L4) su un nodo "parla" logicamente con il suo livello omologo (L4) sul nodo di destinazione. L'insieme di regole di questo dialogo si chiama **protocollo**.
- **Incapsulamento:** Scendendo nello stack (dall'Applicazione al Fisico), ogni livello "imbusta" i dati ricevuti dal livello superiore aggiungendo il proprio **header**.

5.4 L'Architettura TCP/IP (Modello a 5 Livelli)

Questo è il modello operativo su cui si basa Internet.

Livello 5: Applicazione Fornisce servizi all'utente (es. protocolli HTTP per il web, SMTP per l'email).

Livello 4: Trasporto Fornisce un canale di comunicazione *end-to-end* tra le **applicazioni**.

- Identifica le applicazioni tramite **Porte**.
- Protocolli principali: **TCP** (affidabile) e **UDP** (non affidabile).
- Funzioni: Riordino pacchetti, controllo errori, controllo di flusso.

Livello 3: Rete Gestisce il trasferimento di pacchetti tra *nodi* attraverso la rete (anche tra reti diverse).

- Protocollo chiave: **IP (Internet Protocol)**.
- Identifica i nodi tramite **Indirizzi IP**.
- Funzione principale: **Routing (Instradamento)**, ovvero decidere il percorso migliore per i pacchetti.

Livello 2: Data-Link Gestisce il trasferimento di dati (detti **trame** o *frames*) tra nodi *adiacenti* (sullo stesso cavo o stessa rete Wi-Fi).

- Identifica i dispositivi tramite **Indirizzi MAC**.
- Funzioni: Framing (delimitazione trame), controllo errori (Checksum/CRC), accesso al mezzo.

Livello 1: Fisico Gestisce la trasmissione del singolo **bit** sul mezzo fisico (cavo in rame, fibra ottica, onde radio).

6 Approfondimento Livelli Chiave

6.1 Livello 2: Data-Link

Gestisce la comunicazione tra nodi direttamente connessi.

- **Framing:** Crea le "trame" aggiungendo un Header (H) e un Trailer (T) al pacchetto L3. Il trailer contiene tipicamente un **Checksum (CRC)** per il controllo degli errori.
- **Affidabilità:** Può implementare meccanismi di riscontro (**ACK**) e ritrasmissione per correggere gli errori. Serve un *sequence number* per gestire ACK persi e duplicati.
- **Accesso al Mezzo:** Fondamentale se il mezzo è *condiviso* (es. Wi-Fi).
  - **CSMA/CD (Obsoleto):** Usato nelle vecchie Ethernet. Rileva le collisioni e ritrasmette.
  - **CSMA/CA (In uso):** Usato nel **Wi-Fi (802.11)**. Cerca di *evitare* le collisioni prima di trasmettere.
- **Indirizzamento L2:** Avviene tramite **Indirizzo MAC** (48 bit), un identificativo hardware univoco della scheda di rete (NIC).

6.2 Livello 3: Rete (IP)

Il "collante" di Internet.

- **Protocollo IP:** Offre un servizio **best-effort** (fa del suo meglio) e **non affidabile**. I pacchetti possono essere persi, duplicati o arrivare fuori ordine. L'affidabilità è compito del Livello 4 (TCP).
- **Internetworking:** Permette a reti eterogenee (es. Wi-Fi ed Ethernet) di comunicare tra loro.
- **Router:** È il dispositivo chiave del L3. A differenza di un host, un router non "spacchetta" i dati oltre il L3.
  - 1. Riceve una trama L2 (es. Ethernet).
  - 2. Estrae il pacchetto IP (L3).
  - 3. Legge l'indirizzo IP di destinazione.

    - 4. Consulta la sua **tabella di routing** per decidere il "next-hop" (prossimo salto).
    - 5. Re-incapsula il pacchetto IP in una *nuova* trama L2 (es. Wi-Fi) e lo invia.
- **Header IP:** Contiene l'indirizzo IP sorgente e destinazione (32 bit in IPv4), che **non cambiano** per tutto il viaggio del pacchetto.
- **Indirizzi Privati:** Alcuni intervalli IP (es. `192.168.0.0/16`, `10.0.0.0/8`) sono riservati per reti locali e non sono instradati su Internet.

6.3 Livello 4: Trasporto (TCP e UDP)

Gestisce la comunicazione *end-to-end* tra le applicazioni.

- **Multiplazione (Porte):** Usa i numeri di **porta** per distinguere a quale applicazione (es. browser, email) su un host sono destinati i dati.
- **UDP (User Datagram Protocol):**
    - *Connectionless* e *non affidabile*.
    - Non ha conferme, né riordino, né controllo di congestione.
    - È molto veloce e leggero. Usato per streaming video, gaming, DNS.
- **TCP (Transmission Control Protocol):**
    - *Connection-oriented* e *affidabile*.
    - Stabilisce una connessione (handshake) prima di inviare dati.
    - Garantisce che tutti i segmenti arrivino, senza errori e nell'ordine corretto, tramite **ACK** e **numeri di sequenza**.
    - Implementa controllo di flusso e di congestione. Usato per web (HTTP), email (SMTP), file transfer (FTP).

7 Introduzione alla Crittografia

La crittografia è il "building-block" fondamentale per garantire in forma "forte" le proprietà di sicurezza come autenticazione, confidenzialità, integrità e non ripudio.

7.1 Crittologia: Definizioni

La **Crittologia** è il filone di ricerca che si occupa della segretezza delle comunicazioni. Si divide in tre branche:

- **Crittografia**: lo studio dei metodi matematici (algoritmi) per trasformare i dati in modo da renderli incomprensibili a chi non è autorizzato.

- **Protocolli crittografici**: i meccanismi pratici che usano gli algoritmi crittografici per raggiungere un obiettivo (es. autenticazione).
- **Crittanalisi**: lo studio dei meccanismi per "rompere" (circonvenire) i metodi e i protocolli crittografici.

7.2 Terminologia: Gli Attori

Nei protocolli crittografici, si usano nomi convenzionali per identificare i partecipanti:

- **Alice** e **Bob**: i due partecipanti legittimi che vogliono comunicare in modo sicuro.
- **Eve** (Eavesdropper): un'intrusa **passiva**. Si limita ad ascoltare il canale di comunicazione.
- **Trudy** (Intruder): un'intrusa **attiva**. Può intercettare, modificare, eliminare o creare nuovi messaggi.

7.3 Terminologia: Notazione e Operatori

m **(plaintext)** È il messaggio in chiaro, appartenente allo spazio dei messaggi M.

c **(ciphertext)** È il messaggio criptato, appartenente allo spazio C.

k **(key)** È la chiave, appartenente allo spazio delle chiavi K.

Nelle reti moderne, si usano quasi esclusivamente alfabeti binari (stringhe di 0 e 1). I dati sono spesso rappresentati in **esadecimale (hex)**.

L'operatore fondamentale è lo **XOR** (scritto come $\oplus$), che è una somma binaria a 1-bit *senza riporto*.

$\oplus$	**0**	**1**
0	0	1
1	1	0

Tabella 1: Tavola di verità dell'operatore XOR.

8 Schema Crittografico Generale

Un sistema crittografico è definito da due funzioni:

- **Cifratura**: $E_{k1}(m) = c$ (dove $k1$ è la chiave di cifratura).
- **Decifratura**: $D_{k2}(c) = m$ (dove $k2$ è la chiave di decifratura).

La funzione di decifratura D_{k2} deve essere l'inversa di E_{k1}.

8.1 Classi di Algoritmi Crittografici

8.1.1 Block Cipher (Cifrario a Blocchi)

- L'algoritmo opera su blocchi di dati di dimensione fissa b (es. 64 o 128 bit).
- La funzione E_{k1} è **deterministica**: lo stesso blocco di plaintext m_i, se cifrato con la stessa chiave $k1$, produce **sempre** lo stesso blocco di ciphertext c_i.
- **Problema:** Se un blocco di plaintext si ripete nella comunicazione ($m_i = m_j$), anche il blocco di ciphertext si ripeterà ($c_i = c_j$). Questo fa trapelare informazioni (pattern) all'attaccante.

8.1.2 Stream Cipher (Cifrario a Flusso)

- Opera su unità di dati molto piccole (es. 1-8 bit).
- Mantiene uno **stato interno** ($Stato_i$) che viene aggiornato continuamente.
- La funzione di cifratura dipende anche da questo stato: $E_{k1}(m_i, Stato_i) = c_i$.
- **Vantaggio:** Anche se $m_i = m_j$, lo stato interno sarà diverso, quindi $c_i \neq c_j$. Questo nasconde i pattern statistici del plaintext.
- Sono generalmente più veloci e richiedono meno memoria dei block cipher.

9 Crittanalisi e Sicurezza

9.1 Ipotesi di Kerckhoffs

Questo è un principio fondamentale della crittografia moderna: **la sicurezza di un sistema crittografico deve risiedere esclusivamente nella segretezza della chiave (k), non nella segretezza dell'algoritmo (E, D)**. Si deve sempre assumere che l'attaccante (Trudy) conosca perfettamente l'algoritmo che stiamo usando.

9.2 Penetrabilità e Tipi di Attacco

Un algoritmo è **penetrabile** (breakable) se un crittanalista può recuperare m da c (o peggio, k) in un tempo ragionevole. L'obiettivo dei progettisti è fare in modo che l'attacco più veloce sia la **ricerca esaustiva** (brute-force) di tutte le possibili chiavi.

Gli attacchi di crittanalisi si classificano in base all'informazione disponibile all'attaccante:

Ciphertext-only L'attaccante possiede solo messaggi cifrati (c). È l'attacco più difficile.

Known plaintext L'attaccante possiede alcune coppie (m, c) di messaggi in chiaro e cifrati corrispondenti. L'obiettivo è trovare k.

Chosen plaintext L'attaccante può scegliere un m arbitrario e ottenere dall'oracolo (il sistema) il c corrispondente. Un buon algoritmo deve resistere anche a questo scenario.

9.3 Segretezza Perfetta (Shannon)

Un algoritmo offre **segretezza perfetta** se il ciphertext c e il plaintext m sono **statisticamente indipendenti**.

- In pratica: osservare c non dà **alcuna informazione** in più su m.
- **Condizione di Shannon:** La chiave deve essere lunga almeno quanto il messaggio ($H(k) \geq H(m)$).

9.3.1 One-Time-Pad (OTP)

È l'unico algoritmo noto a garantire segretezza perfetta.

- **Algoritmo:** Cifratura $c = m \oplus k$ e Decifratura $m = c \oplus k$.
- **Ipotesi fondamentali:** La chiave k deve essere:
  1. Lunga **esattamente** quanto il messaggio m.
  2. Generata in modo **perfettamente casuale**.
  3. Utilizzata **una sola volta** (da cui il nome "one-time").
- **Problema:** La generazione e la distribuzione sicura di una chiave così lunga (e usarla una sola volta) è estremamente difficile nella pratica.

10 Tipologie Base di Algoritmi Crittografici

Gli algoritmi crittografici si dividono in tre famiglie principali:

Simmetrici (a chiave privata) Si usa una singola chiave segreta condivisa: $k = k1 = k2$.

Asimmetrici (a chiave pubblica) Si usa una coppia di chiavi (una pubblica e una privata): $k1 \neq k2$.

Algoritmi di Hash Non usano chiavi per la cifratura, ma producono un "fingerprint" dei dati.