# 1. SICUREZZA WIRELESS E PROTOCOLLI DI RETE

## 1.1 Sicurezza wireless

### 1.1.1 WEP (Wired Equivalent Privacy)

- Primo protocollo di sicurezza per 802.11 (introdotto nel 1999)
- Utilizza algoritmo RC4 con chiavi da 64 o 128 bit
- Problemi:
  - Vettore di inizializzazione (IV) troppo corto (24 bit)
  - Riutilizzo di chiavi
  - Autenticazione debole
- Considerato completamente insicuro oggi
- Vulnerabilità principali:
  - Collisioni degli IV (dopo circa 5000 pacchetti)
  - Attacchi statistici (FMS)
  - Possibile decifrazione completa in pochi minuti

### 1.1.2 WPA (Wi-Fi Protected Access)

- Soluzione intermedia introdotta nel 2003 dopo i problemi di WEP
- Usa TKIP (Temporal Key Integrity Protocol)
- Miglioramenti rispetto a WEP:
  - IV più lungo (48 bit)
  - Mixing function per le chiavi
  - Message Integrity Check (MIC)
  - Distribuzione chiavi dinamica
- Ancora vulnerabile ad alcuni attacchi (crack di password PSK con dizionario)

### 1.1.3 WPA2

- Standard IEEE 802.11i rilasciato nel 2004
- Usa CCMP basato su AES invece di RC4/TKIP
- Modalità:
  - **Personal (PSK)**: chiave precondivisa, adatta per reti domestiche/piccoli uffici
  - **Enterprise**: autenticazione basata su 802.1X e RADIUS, per organizzazioni più grandi
- Considerato sicuro se configurato correttamente

- Vulnerabilità note:
  - KRACK (Key Reinstallation Attack) scoperto nel 2017
  - Attacchi di forza bruta su password deboli

### 1.1.4 WPA3

- Introdotto nel 2018 come successore di WPA2
- Caratteristiche principali:
  - SAE (Simultaneous Authentication of Equals) sostituisce il PSK
  - Protezione dagli attacchi di dizionario offline
  - Forward secrecy
  - Crittografia a 192 bit per reti enterprise
  - Protezione migliorata per reti pubbliche (OWE)

## 1.2 HTTPS (HTTP Secure) e SSL/TLS

- HTTPS = HTTP su connessione crittografata (SSL/TLS)
- Funzionamento:
  1. Client richiede connessione sicura al server
  2. Server invia il suo certificato
  3. Client verifica il certificato e genera una chiave di sessione
  4. La chiave viene scambiata in modo sicuro
  5. La comunicazione prosegue cifrata con la chiave di sessione
- Vantaggi:
  - Protezione contro intercettazioni (confidenzialità)
  - Verifica dell'identità del server (autenticazione)
  - Garantisce l'integrità dei dati
- Evoluzione dei protocolli:
  - SSL 2.0/3.0: obsoleti e vulnerabili
  - TLS 1.0/1.1: deprecati
  - TLS 1.2: ancora ampiamente utilizzato
  - TLS 1.3 (2018): più veloce e sicuro

## 1.3 Altri protocolli di sicurezza

### 1.3.1 IPsec (IP Security)

- Suite di protocolli per sicurezza a livello IP
- Componenti:
  - **AH (Authentication Header)**: integrità e autenticazione
  - **ESP (Encapsulating Security Payload)**: confidenzialità, integrità, autenticazione
  - **IKE (Internet Key Exchange)**: gestione delle chiavi

- Modalità:
  - **Transport mode**: protegge solo il payload
  - **Tunnel mode**: protegge l'intero pacchetto IP
- Utilizzi comuni:
  - VPN site-to-site
  - Protezione del traffico sensibile
  - Implementazione del modello di sicurezza end-to-end

### 1.3.2 VPN (Virtual Private Network)

- Crea un tunnel sicuro attraverso una rete non sicura (Internet)
- Tipi di VPN:
  - **Remote Access**: connette un singolo utente a una rete
  - **Site-to-Site**: connette intere reti tra loro
- Protocolli comuni:
  - **IPsec**: sicuro, supportato da molti dispositivi
  - **SSL/TLS**: più facile da attraversare firewall
  - **OpenVPN**: soluzione open-source flessibile
  - **WireGuard**: moderno, veloce, codice compatto
- Usi:
  - Accesso remoto alle risorse aziendali
  - Protezione su reti Wi-Fi pubbliche
  - Connessione sicura tra sedi distaccate

## 1.4 Tecniche di attacco e difesa

### 1.4.1 Man in the Middle (MITM)

- Attacco in cui l'aggressore si posiziona tra due parti comunicanti
- Metodi:
  - ARP spoofing/poisoning
  - DNS spoofing
  - Rogue access point
  - SSL stripping
- Difese:
  - HTTPS (certificati validi)
  - HSTS (HTTP Strict Transport Security)
  - Mutual authentication
  - Certificate pinning

### 1.4.2 DOS/DDOS (Denial of Service)

- Attacco che mira a rendere un servizio non disponibile
- Tipi:
  - **Volumetric**: sovraccarica la banda (UDP flood, ICMP flood)
  - **Protocol**: consuma risorse server (SYN flood)
  - **Application Layer**: attacca vulnerabilità applicazioni (HTTP flood, Slowloris)
- Difese:
  - Filtraggio del traffico
  - Rate limiting
  - Load balancing
  - Servizi anti-DDoS

### 1.4.3 Firewall

- Sistema che filtra il traffico di rete in base a regole predefinite
- Tipi:
  - **Packet filtering**: filtra in base a header (livello 3-4)
  - **Stateful inspection**: tiene traccia delle connessioni
  - **Application layer**: analizza il traffico a livello applicativo
  - **Next-gen**: include IPS, antivirus, deep packet inspection
- Regole tipiche:
  - Default deny (blocca tutto tranne il permesso)
  - Allow/Block in base a indirizzo IP, porta, protocollo
  - Limitazione delle connessioni
  - Content filtering

---

# 2. CRITTOGRAFIA E SICUREZZA DEI DATI

## 2.1 Crittografia simmetrica

- Usa la stessa chiave per cifrare e decifrare
- Veloce ma richiede scambio sicuro della chiave
- Algoritmi principali:
  - **DES** (Data Encryption Standard): obsoleto, chiave 56 bit
  - **3DES** (Triple DES): applica DES tre volte, più sicuro ma lento
  - **AES** (Advanced Encryption Standard): standard attuale, chiavi 128/192/256 bit
  - **ChaCha20**: alternativa veloce ad AES, usata in TLS e applicazioni mobili

## 2.2 Crittografia asimmetrica

- Utilizza coppia di chiavi: pubblica (per cifrare) e privata (per decifrare)

- Più lenta della simmetrica ma risolve il problema dello scambio chiavi
- Algoritmi principali:
  - **RSA**: basato sulla fattorizzazione di numeri primi grandi
    - Chiavi tipiche: 2048 o 4096 bit
    - Usato per cifratura e firma digitale
  - **Diffie-Hellman**: metodo per scambio chiavi su canale insicuro
  - **ECC** (Elliptic Curve Cryptography): più efficiente di RSA con chiavi più corte

### 2.2.1 Esempio di funzionamento RSA (semplificato)

1. **Generazione chiavi**:
   - Scegliere due numeri primi p e q (es. p=3, q=11)
   - Calcolare n = p×q = 33
   - Calcolare φ(n) = (p-1)×(q-1) = 2×10 = 20
   - Scegliere e (chiave pubblica) coprimo con φ(n), es. e=7
   - Calcolare d (chiave privata) tale che e×d ≡ 1 mod φ(n), es. d=3
   - Chiave pubblica: (n,e) = (33,7)
   - Chiave privata: (n,d) = (33,3)
2. **Cifratura**:
   - Messaggio m = 2
   - Cifrato c = m^e mod n = 2^7 mod 33 = 128 mod 33 = 29
3. **Decifratura**:
   - Cifrato c = 29
   - Messaggio m = c^d mod n = 29^3 mod 33 = 24389 mod 33 = 2

## 2.3 Funzioni di hash

- Trasformano input di lunghezza arbitraria in output di lunghezza fissa
- Proprietà:
  - One-way (non invertibile)
  - Resistenza alle collisioni
  - Effetto valanga (piccole modifiche causano output molto diversi)
- Algoritmi:
  - **MD5**: 128 bit, obsoleto e vulnerabile
  - **SHA-1**: 160 bit, considerato non sicuro
  - **SHA-256/SHA-512**: parte della famiglia SHA-2, standard attuali
  - **SHA-3**: nuova generazione, basata su costruzione a spugna

## 2.4 Firma digitale

- Garantisce autenticità e non ripudio

- Processo:
  1. Creazione hash del documento
  2. Cifratura dell'hash con la chiave privata del mittente
  3. Invio del documento e della firma
  4. Verifica decifrando la firma con la chiave pubblica del mittente
- Applicazioni:
  - **PEC** (Posta Elettronica Certificata)
  - **Documenti XML firmati**
  - **Certificati digitali**
  - **Smart contract**

## 2.5 Politiche di accesso

### 2.5.1 DAC (Discretionary Access Control)

- Il proprietario della risorsa decide chi può accedervi
- Esempi: permessi file in Windows/Unix
- Caratteristiche:
  - Flessibile
  - Facile da gestire per piccoli sistemi
  - Vulnerabile a errori umani
  - Non adatto a policy di sicurezza complesse

### 2.5.2 MAC (Mandatory Access Control)

- Sistema centrale impone regole di sicurezza
- Basato su etichette di sicurezza
- Esempi: SELinux, AppArmor
- Caratteristiche:
  - Maggiore sicurezza
  - Meno flessibilità
  - Complessità di configurazione
  - Adatto ad ambienti ad alta sicurezza

---

# 3. LIVELLO APPLICATIVO (LAYER 7)

## 3.1 DNS (Domain Name System)

- Risolve nomi di dominio in indirizzi IP
- Gerarchia di server DNS:

    - Root servers (.)
    - TLD (Top-Level Domain) servers (.com, .org, .it)
    - Authoritative servers (per specifici domini)
    - Resolver locali (ISP, Google 8.8.8.8)
- Processo di risoluzione:
    1. Client interroga resolver locale
    2. Se non in cache, resolver interroga server root
    3. Root indica TLD appropriato
    4. TLD indica server autoritativo
    5. Server autoritativo fornisce risposta
- Record principali:
    - A: indirizzo IPv4
    - AAAA: indirizzo IPv6
    - CNAME: alias
    - MX: mail server
    - NS: name server
    - TXT: informazioni testuali
    - SOA: informazioni di autorità
- Query:
    - Ricorsive: resolver fa tutto il lavoro
    - Iterative: client segue i riferimenti da solo

# 3.2 Protocolli di posta elettronica

## 3.2.1 SMTP (Simple Mail Transfer Protocol)

- Protocollo per l'invio di email
- Porta standard: 25 (non cifrata), 587 (TLS)
- Comandi principali:
    - HELO/EHLO: identificazione client
    - MAIL FROM: mittente
    - RCPT TO: destinatario
    - DATA: corpo del messaggio
    - QUIT: termina la sessione
- Esempio di sessione SMTP:

```
C: EHLO client.example.com
S: 250 Hello client.example.com
C: MAIL FROM:<mittente@example.com>
S: 250 OK
C: RCPT TO:<destinatario@example.org>
```

```
S: 250 OK
C: DATA
S: 354 Start mail input; end with <CRLF>.<CRLF>
C: From: "Mittente" <mittente@example.com>
C: To: "Destinatario" <destinatario@example.org>
C: Subject: Test email
C:
C: Questo è un test.
C: .
S: 250 OK
C: QUIT
S: 221 Bye
```

### 3.2.2 POP3 (Post Office Protocol v3)

- Protocollo per il recupero di email
- Porta standard: 110 (non cifrata), 995 (TLS)
- Caratteristiche:
  - Semplice, stateless
  - Tipicamente scarica e rimuove messaggi dal server
  - Adatto a connessioni intermittenti
- Comandi principali: USER, PASS, LIST, RETR, DELE
- Esempio di sessione POP3:

```
C: USER mario@example.com
S: +OK
C: PASS password
S: +OK Mailbox locked and ready
C: LIST
S: +OK 2 messages (320 octets)
S: 1 120
S: 2 200
S: .
C: RETR 1
S: +OK 120 octets
S: [contenuto del messaggio]
S: .
C: DELE 1
S: +OK message 1 deleted
C: QUIT
S: +OK Bye
```

### 3.2.3 IMAP (Internet Message Access Protocol)

- Alternativa più avanzata a POP3
- Porta standard: 143 (non cifrata), 993 (TLS)
- Caratteristiche:
  - Mantiene i messaggi sul server
  - Supporta cartelle e flag
  - Sincronizzazione tra dispositivi
  - Ricerca sul server
  - Download parziale dei messaggi
- Vantaggi rispetto a POP3:
  - Accessibilità da più dispositivi
  - Backup centralizzato
  - Organizzazione migliore
  - Funzionalità avanzate (ricerca, filtri)

## 3.3 Accesso remoto

### 3.3.1 SSH (Secure Shell)

- Protocollo per accesso remoto sicuro
- Sostituisce Telnet, rsh, rlogin
- Caratteristiche:
  - Crittografia forte
  - Autenticazione a chiave pubblica
  - Tunneling e inoltro porte
  - SFTP (SSH File Transfer Protocol)
  - SCP (Secure Copy)
- Porta standard: 22
- Comandi base:
  - Connessione: `ssh utente@host`
  - Copia file: `scp file.txt utente@host:/path/`
  - Tunneling: `ssh -L 8080:localhost:80 utente@host`

### 3.3.2 TELNET

- Protocollo legacy per accesso remoto
- Non sicuro: trasmette dati in chiaro
- Porta standard: 23
- Sostituito da SSH per quasi tutti gli usi
- Ancora utilizzato per debug di servizi di rete e dispositivi legacy

# 3.4 Scambio file e protocolli peer-to-peer

## 3.4.1 FTP (File Transfer Protocol)

- Protocollo classico per trasferimento file
- Utilizza due connessioni:
  - Controllo (porta 21)
  - Dati (porta 20 o dinamica)
- Modalità:
  - Attiva: server inizia connessione dati
  - Passiva: client inizia connessione dati (più sicura con firewall)
- Stato: autenticato o anonimo
- Comandi principali: USER, PASS, CWD, PWD, LIST, RETR, STOR
- Svantaggi: nessuna crittografia nativa, problemi con firewall

## 3.4.2 FTPS (FTP Secure)

- FTP con SSL/TLS
- Due varianti:
  - Implicito: sempre cifrato (porta 990)
  - Esplicito: negoziazione con comando AUTH (porta 21)
- Vantaggi: compatibile con client FTP esistenti, sicuro
- Svantaggi: problemi con firewall, complessità dei certificati

## 3.4.3 SFTP (SSH File Transfer Protocol)

- Protocollo di trasferimento file su SSH
- Singola connessione cifrata (porta 22)
- Caratteristiche:
  - Autenticazione forte
  - Integrità dei dati
  - Resume transfer
  - Directory listing
- Vantaggi: sicuro, attraversa firewall facilmente
- Svantaggi: non compatibile con client FTP tradizionali

## 3.4.4 BitTorrent

- Protocollo P2P per condivisione file
- Caratteristiche:
  - File suddivisi in pezzi (chunks)
  - Distribuzione parallela da più fonti

    - Algoritmo "rarest first" per ottimizzare distribuzione
    - Tit-for-tat per incentivare upload
- Componenti:
    - Tracker: coordina i peer
    - Seeder: utente con file completo
    - Leecher: utente che sta scaricando
    - Torrent file: metadati (hash dei pezzi, tracker, ecc.)
- Evoluzione:
    - DHT (Distributed Hash Table) per operare senza tracker
    - PEX (Peer Exchange) per scoprire nuovi peer
    - Magnet link: alternativa ai file .torrent

### 3.4.5 Gnutella

- Rete P2P decentralizzata (prima generazione)
- Funzionamento:
    - Messaggi di query inoltrati a tutti i peer connessi
    - Risposte inviate direttamente al richiedente
    - Trasferimento file avviene direttamente tra peer
- Caratteristiche:
    - Nessun server centrale
    - Tolleranza ai guasti
    - Problemi di scalabilità (flooding)
- Client storici: LimeWire, Morpheus, BearShare

## 3.5 API e microservizi

### 3.5.1 Concetto di API (Application Programming Interface)

- Interfaccia che permette la comunicazione tra diverse applicazioni
- Tipi di API:
    - **REST**: basata su HTTP, stateless, risorse identificate da URL
    - **SOAP**: protocollo più complesso basato su XML
    - **GraphQL**: permette al client di specificare i dati richiesti
    - **gRPC**: usa Protocol Buffers e HTTP/2 per comunicazione efficiente

### 3.5.2 REST (Representational State Transfer)

- Architettura per API web basata su HTTP
- Principi:
    - Risorse identificate da URI
    - Operazioni CRUD tramite metodi HTTP:

        - GET: lettura
        - POST: creazione
        - PUT/PATCH: aggiornamento
        - DELETE: eliminazione
    - Stateless (ogni richiesta contiene tutte le informazioni necessarie)
    - Interfaccia uniforme
- Formati comuni: JSON, XML
- Vantaggi: semplice, ampiamente supportato, cacheable
- Svantaggi: over-fetching/under-fetching, multiple richieste

### 3.5.3 Microservizi

- Architettura software che divide l'applicazione in servizi piccoli e indipendenti
- Caratteristiche:
    - Ogni servizio ha una singola responsabilità
    - Servizi indipendentemente sviluppabili e deployabili
    - Comunicazione tramite API (spesso REST)
    - Scalabilità e resilienza migliorate
- Pattern comuni:
    - API Gateway: punto di ingresso unificato
    - Service Registry: discovery dei servizi
    - Circuit Breaker: gestione fallimenti
- Vantaggi:
    - Deployment indipendente
    - Scalabilità granulare
    - Tecnologie eterogenee
    - Team autonomi
- Svantaggi:
    - Complessità distribuita
    - Overhead di comunicazione
    - Difficoltà nel debug

## 3.6 Architetture di rete

### 3.6.1 Client-Server

- Server centralizzato fornisce servizi a più client
- Vantaggi:
    - Controllo centralizzato
    - Sicurezza più facile da implementare
    - Backup e manutenzione semplificati

- Svantaggi:
  - Single point of failure
  - Limitazioni di scalabilità
  - Costi infrastruttura centralizzata
- Esempi: web server, email, database server

### 3.6.2 Peer-to-Peer (P2P)

- Ogni nodo può fungere sia da client che da server
- Tipi:
  - **P2P puro**: tutti i nodi sono equivalenti
  - **P2P ibrido**: alcuni nodi (supernodi) hanno ruoli speciali
- Vantaggi:
  - Resilienza (nessun single point of failure)
  - Scalabilità naturale
  - Costi distribuiti
- Svantaggi:
  - Sicurezza più difficile da implementare
  - Prestazioni variabili
  - Maggiore complessità
- Esempi: BitTorrent, blockchain, comunicazioni dirette

---

# 4. MALWARE E SICUREZZA DEL SOFTWARE

## 4.1 Tipi di malware

### 4.1.1 Virus

- Software malevolo che si attacca a file legittimi
- Caratteristiche:
  - Richiede azione umana per diffondersi
  - Si replica infettando altri file
  - Può restare dormiente
- Tipi:
  - **File infector**: infetta file eseguibili
  - **Boot sector**: infetta il settore di avvio
  - **Macro virus**: utilizza macro in documenti
  - **Polymorphic**: cambia forma per evitare rilevamento

### 4.1.2 Worm

- Programma autonomo che si propaga automaticamente
- Caratteristiche:
  - Non richiede intervento umano
  - Sfrutta vulnerabilità di rete
  - Non necessita di file host
- Esempi storici:
  - Morris Worm (1988)
  - ILOVEYOU (2000)
  - Conficker (2008)
  - WannaCry (2017)

### 4.1.3 Trojan

- Malware mascherato da software legittimo
- Caratteristiche:
  - Non si replica
  - Induce l'utente all'installazione
  - Spesso crea backdoor
- Categorie:
  - **Backdoor**: fornisce accesso remoto
  - **Downloader**: scarica altro malware
  - **Banker**: ruba credenziali bancarie
  - **RAT** (Remote Access Trojan): controllo completo
  - **Spyware**: raccoglie informazioni

### 4.1.4 Ransomware

- Crittografa i dati dell'utente e chiede un riscatto
- Fasi tipiche:
  1. Infezione (phishing, vulnerabilità)
  2. Scansione file
  3. Crittografia
  4. Richiesta riscatto
- Esempi:
  - CryptoLocker
  - WannaCry
  - Ryuk
  - REvil

## 4.2 Vettori di infezione

### 4.2.1 Email e allegati

- Phishing: impersonifica entità legittime
- Allegati malevoli (documenti con macro, eseguibili mascherati)
- Link a siti malevoli
- Indicatori di rischio:
  - Richieste urgenti
  - Errori grammaticali
  - Indirizzi sospetti
  - Allegati inaspettati

### 4.2.2 Download da Internet

- Software da fonti non affidabili
- Drive-by download: download automatico visitando siti compromessi
- Bundleware: software aggiuntivo installato con programmi legittimi
- Crack e keygen: spesso contengono malware

### 4.2.3 Dispositivi rimovibili

- USB infetti
- AutoRun/AutoPlay
- Firmware modificato (BadUSB)
- Cross-infezione tra sistemi

## 4.3 Protezione e prevenzione

### 4.3.1 Antivirus e antimalware

- Funzionamento:
  - Signature-based: confronto con database di firme note
  - Heuristic-based: analisi comportamentale
  - Cloud-based: analisi in tempo reale
- Limitazioni:
  - Inefficaci contro malware avanzato/nuovo
  - Falsi positivi/negativi
  - Impatto sulle prestazioni

### 4.3.2 Best practices

- Aggiornamenti regolari (sistema operativo e software)
- Backup frequenti (regola 3-2-1)
- Principio del privilegio minimo

- Email filtering
- Firewall personale
- Formazione degli utenti

### 4.3.3 Sandbox e virtualizzazione

- Esecuzione di software sospetto in ambiente isolato
- Tipi:
  - Application sandbox: limita le capacità di un'applicazione
  - System sandbox: simula un intero sistema
  - Browser sandbox: isola il browser dal sistema
- Vantaggi:
  - Protezione del sistema principale
  - Analisi sicura
  - Ripristino rapido

---