1. Livello Applicativo (Layer 7)

Il livello applicativo è il settimo e ultimo livello del modello ISO/OSI, nonché il livello più alto del modello TCP/IP. È il livello più vicino all'utente finale e fornisce i servizi di rete alle applicazioni.

1.1 DNS (Domain Name System)

Il DNS è un sistema fondamentale che risolve nomi di dominio in indirizzi IP, rendendo la navigazione in Internet più user-friendly.

1.1.1 Struttura gerarchica DNS

- **Root servers (.)**: 13 set di server gestiti da diverse organizzazioni
- **TLD (Top-Level Domain) servers**: gestiscono domini come .com, .org, .it
- **Authoritative servers**: contengono informazioni specifiche per un dominio
- **Resolver locali**: server DNS degli ISP o pubblici (es. Google 8.8.8.8, Cloudflare 1.1.1.1)

1.1.2 Processo di risoluzione DNS

1. L'utente richiede un sito (es. www.example.com)
2. Il client interroga il resolver locale DNS
3. Se la risposta non è in cache, il resolver interroga un server root
4. Il server root indica il server TLD appropriato (per .com)
5. Il server TLD indica il server autoritativo per example.com
6. Il server autoritativo fornisce l'IP di www.example.com
7. La risposta torna al resolver e viene memorizzata in cache
8. Il resolver fornisce l'IP al client

1.1.3 Record DNS principali

- **A**: mappa un nome a un indirizzo IPv4 (es. example.com → 93.184.216.34)
- **AAAA**: mappa un nome a un indirizzo IPv6 (es. example.com → 2606:2800:220:1:248:1893:25c8:1946)
- **CNAME**: alias per un altro nome (es. www.example.com → example.com)
- **MX**: indica i server di posta per un dominio
- **NS**: specifica i name server autoritativi per il dominio
- **TXT**: memorizza informazioni testuali (usato per SPF, DKIM, verifiche)
- **SOA**: contiene informazioni amministrative sul dominio
- **PTR**: reverse DNS, mappa IP a nomi (usato per verifiche di autenticità)

1.1.4 Tipi di query DNS

- **Ricorsive**: il resolver fa tutto il lavoro di risoluzione per il client
- **Iterative**: il client (o resolver) segue i riferimenti da solo
- **Inverse**: traduce un indirizzo IP in un nome di dominio

1.1.5 DNS sicuro

- **DNSSEC**: autentica le risposte DNS con firme digitali
- **DNS over HTTPS (DoH)**: cifra le query DNS usando HTTPS
- **DNS over TLS (DoT)**: cifra le query DNS usando TLS
- **Split DNS**: diversi server DNS per richieste interne ed esterne

1.2 Protocolli di posta elettronica

1.2.1 SMTP (Simple Mail Transfer Protocol)

- **Funzione**: Protocollo per l'invio di email
- **Porte**: 25 (non cifrata), 587 (TLS, submission), 465 (SSL, deprecata ma usata)
- **Caratteristiche**:
  - Protocollo testuale basato su comandi ASCII
  - Connessione persistente
  - Supporta diverse estensioni (ESMTP)
  - Non scarica messaggi, solo li invia

Comandi principali SMTP:

- `HELO/EHLO`: identificazione client
- `MAIL FROM`: specificare mittente
- `RCPT TO`: specificare destinatario/i
- `DATA`: iniziare trasmissione del corpo del messaggio
- `QUIT`: terminare 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
```

1.2.2 POP3 (Post Office Protocol v3)

- **Funzione**: Protocollo per scaricare email dal server
- **Porte**: 110 (non cifrata), 995 (SSL/TLS)
- **Caratteristiche**:
  - Semplice, stateless
  - Tipicamente scarica e rimuove messaggi dal server
  - Adatto a connessioni intermittenti
  - Generalmente non supporta cartelle
  - Approccio "store-and-download"

Comandi principali POP3:

- `USER`: specificare nome utente
- `PASS`: specificare password
- `LIST`: elencare messaggi disponibili
- `RETR`: scaricare un messaggio specifico
- `DELE`: cancellare un messaggio
- `QUIT`: terminare sessione e applicare modifiche

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
```

1.2.3 IMAP (Internet Message Access Protocol)

- **Funzione**: Protocollo avanzato per la gestione remota delle email
- **Porte**: 143 (non cifrata), 993 (SSL/TLS)
- **Caratteristiche**:
  - Mantiene i messaggi sul server
  - Supporta cartelle e flag (letto, risposto, importante)
  - Sincronizzazione tra dispositivi diversi
  - Ricerca sul server
  - Download parziale dei messaggi
  - Approccio "store-on-server"

Vantaggi rispetto a POP3:

- Accessibilità da più dispositivi
- Backup centralizzato
- Organizzazione migliore
- Funzionalità avanzate (ricerca, filtri)
- Gestione dello stato (letto/non letto, contrassegnato, ecc.)

Comandi principali IMAP:

- `LOGIN` : autenticazione
- `SELECT` : selezione cartella
- `FETCH` : recupero messaggi o parti di essi
- `STORE` : modifica flag dei messaggi
- `SEARCH` : ricerca messaggi
- `CREATE/DELETE` : gestione cartelle

1.2.4 Sicurezza email

- **SPF (Sender Policy Framework)**: verifica che il server di invio sia autorizzato
- **DKIM (DomainKeys Identified Mail)**: firma crittografica dei messaggi
- **DMARC (Domain-based Message Authentication, Reporting & Conformance)**: policy per gestire fallimenti SPF/DKIM
- **TLS per SMTP/POP/IMAP**: cifratura della connessione

- **S/MIME e PGP/GPG**: cifratura end-to-end e firma del contenuto

1.3 Accesso remoto

1.3.1 SSH (Secure Shell)

- **Funzione**: Protocollo sicuro per accesso remoto ai sistemi
- **Porta**: 22
- **Caratteristiche**:
  - Crittografia forte
  - Autenticazione a chiave pubblica/privata
  - Tunneling e inoltro porte
  - Compressione dati
  - SFTP (SSH File Transfer Protocol)
  - SCP (Secure Copy)

Funzionalità principali:

- **Remote login**: accesso shell sicuro
- **Port forwarding**: reindirizzamento porte locali/remote
- **X11 forwarding**: applicazioni grafiche remote
- **SOCKS proxy**: tunneling traffico
- **Agent forwarding**: autenticazione a cascata

Comandi base:

- Connessione: `ssh utente@host`
- Generazione chiavi: `ssh-keygen`
- Copia file: `scp file.txt utente@host:/path/`
- Tunneling: `ssh -L 8080:localhost:80 utente@host`

1.3.2 TELNET

- **Funzione**: Protocollo legacy per accesso remoto testuale
- **Porta**: 23
- **Caratteristiche**:
  - Non sicuro: trasmette dati in chiaro
  - Semplice e leggero
  - Interfaccia a linea di comando
  - Ancora utilizzato per debugging e dispositivi embedded
- **Uso attuale**: Sostituito quasi completamente da SSH
- **Rischi**: Vulnerabile a sniffing e man-in-the-middle

1.4 Scambio file e protocolli peer-to-peer

1.4.1 FTP (File Transfer Protocol)

- **Funzione**: Protocollo classico per trasferimento file
- **Architettura**: Utilizza due connessioni separate
  - Connessione di controllo (porta 21): comandi
  - Connessione dati (porta 20 o dinamica): trasferimento file
- **Modalità**:
  - **Active mode**: server inizia connessione dati
  - **Passive mode**: client inizia connessione dati (preferibile con firewall)
- **Tipi di trasferimento**:
  - ASCII: per file di testo (conversione fine riga)
  - Binary: per file binari (nessuna conversione)
- **Autenticazione**: nome utente/password o anonima
- **Svantaggi**: nessuna crittografia nativa, problemi con firewall

Comandi principali:

- `USER/PASS` : autenticazione
- `CWD/PWD` : navigazione directory
- `LIST` : elenco file
- `RETR` : download file
- `STOR` : upload file
- `PASV` : attiva modalità passiva

1.4.2 FTPS (FTP Secure)

- **Definizione**: FTP con layer di sicurezza SSL/TLS
- **Varianti**:
  - **Implicito**: sempre cifrato (porta 990)
  - **Esplicito (FTPES)**: negoziazione con comando AUTH TLS (porta 21)
- **Vantaggi**: compatibile con client FTP esistenti, sicuro
- **Svantaggi**: problemi con firewall, complessità dei certificati

1.4.3 SFTP (SSH File Transfer Protocol)

- **Definizione**: Protocollo di trasferimento file su SSH
- **Caratteristiche**:
  - Singola connessione cifrata (porta 22)
  - Autenticazione integrata con SSH
  - Trasferimenti binari sicuri

    - Supporto per resume, permessi file, simbolic links
    - Non è FTP su SSH, ma un protocollo completamente diverso
- **Vantaggi**: sicuro, attraversa firewall facilmente, gestione permessi
- **Svantaggi**: non compatibile con client FTP tradizionali

1.4.4 BitTorrent

- **Definizione**: Protocollo P2P per condivisione file distribuita
- **Caratteristiche**:
    - File suddivisi in pezzi (chunks) distribuiti tra peer
    - Distribuzione parallela da più fonti
    - Algoritmo "rarest first" per ottimizzare distribuzione
    - Tit-for-tat per incentivare upload (reciprocità)
    - Scalabilità eccellente

Componenti:

- **Tracker**: coordina i peer (centralizzato o distribuito)
- **Seeder**: utente con file completo
- **Leecher**: utente che sta scaricando
- **Torrent file/Magnet link**: metadati (hash dei pezzi, tracker, ecc.)
- **Swarm**: insieme di peer che condividono lo stesso file

Evoluzione:

- **DHT (Distributed Hash Table)**: permette operare senza tracker
- **PEX (Peer Exchange)**: peer condividono liste di altri peer
- **Magnet link**: alternativa ai file .torrent
- **BitTorrent v2**: nuova generazione con sicurezza migliorata

1.4.5 Gnutella

- **Definizione**: Rete P2P decentralizzata di 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 (completamente distribuita)
    - Tolleranza ai guasti
    - Flooding inefficiente (problemi di scalabilità)
- **Client storici**: LimeWire, Morpheus, BearShare

- **Evoluzione**: Varianti come Gnutella2 con architettura migliorata

1.5 API e microservizi

1.5.1 Concetto di API (Application Programming Interface)

- **Definizione**: Interfaccia che permette a diversi software di comunicare
- **Scopo**: Esporre funzionalità in modo standardizzato e controllato
- **Tipi**:
  - **Library API**: funzioni di una libreria software
  - **OS API**: interfacce del sistema operativo
  - **Web API**: servizi accessibili via HTTP/HTTPS

1.5.2 REST (Representational State Transfer)

- **Definizione**: Architettura per API web basata su HTTP e risorse
- **Principi**:
  - Risorse identificate da URI (Uniform Resource Identifiers)
  - Interfaccia uniforme (metodi HTTP standard)
  - Stateless (ogni richiesta contiene tutte le informazioni necessarie)
  - Rappresentazioni multiple (JSON, XML, etc.)
  - Sistema a livelli

Operazioni CRUD tramite metodi HTTP:

- **GET**: lettura (read)
- **POST**: creazione (create)
- **PUT/PATCH**: aggiornamento (update)
- **DELETE**: eliminazione (delete)

Caratteristiche:

- Formati comuni: JSON, XML
- Facile da usare e comprendere
- Supporto per caching
- Ampiamente supportato e diffuso
- Scalabile e performante
- Indipendente dal linguaggio e dalla piattaforma

Svantaggi:

- Over-fetching/under-fetching (troppe/poche informazioni)
- Multiple richieste per operazioni complesse
- Limitazioni per applicazioni in tempo reale

Esempio REST API:

```
# Ottenere tutti gli utenti
GET /api/users

# Ottenere un utente specifico
GET /api/users/123

# Creare un nuovo utente
POST /api/users
{
  "name": "Mario Rossi",
  "email": "mario@example.com"
}

# Aggiornare un utente
PUT /api/users/123
{
  "name": "Mario Rossi",
  "email": "mario.rossi@example.com"
}

# Eliminare un utente
DELETE /api/users/123
```

1.6 Architetture di rete

1.6.1 Client-Server

- **Definizione**: Architettura in cui i server forniscono risorse/servizi e i client li consumano
- **Caratteristiche**:
  - Separazione chiara dei ruoli
  - Server centralizzati gestiscono risorse condivise
  - Client richiedono servizi
  - Comunicazione iniziata dal client

Tipi di server:

- Web server
- Database server
- File server
- Mail server
- Application server

- Print server

Vantaggi:

- Controllo centralizzato
- Sicurezza più facile da implementare
- Backup e manutenzione semplificati
- Gestione delle risorse efficiente

Svantaggi:

- Single point of failure
- Limitazioni di scalabilità
- Costi infrastruttura centralizzata
- Dipendenza dalla connettività

1.6.2 Peer-to-Peer (P2P)

- **Definizione**: Architettura in cui 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
  - **P2P strutturato**: organizzazione basata su DHT
  - **P2P non strutturato**: connessioni casuali tra peer

Caratteristiche:

- Decentralizzazione
- Auto-organizzazione
- Risorse distribuite
- Tolleranza ai guasti

Vantaggi:

- Resilienza (nessun single point of failure)
- Scalabilità naturale
- Costi distribuiti
- Robustezza

Svantaggi:

- Sicurezza più difficile da implementare
- Prestazioni variabili
- Maggiore complessità

- Difficoltà di gestione

Esempi: BitTorrent, Blockchain, IPFS, Skype (ibrido)

1.6.3 Architetture ibride

- **Definizione**: Combinano elementi di client-server e P2P
- **Esempi**:
  - Content Delivery Networks (CDN)
  - Cloud distribuito
  - Edge computing
  - Fog computing

Caratteristiche:

- Bilanciamento tra centralizzazione e distribuzione
- Flessibilità di implementazione
- Ottimizzazione locale vs. globale
- Gerarchie di servizi

2. Malware e Sicurezza del Software

2.1 Tipi di malware

2.1.1 Virus

- **Definizione**: Software malevolo che si attacca a file legittimi e si replica infettando altri file
- **Caratteristiche**:
  - Richiede azione umana per diffondersi
  - Si replica infettando altri file
  - Può restare dormiente
  - Richiede un "ospite" (file eseguibile)

Tipi principali:

- **File infector**: infetta file eseguibili (.exe, .com)
- **Boot sector**: infetta il settore di avvio
- **Macro virus**: utilizza macro in documenti Office
- **Polymorphic**: cambia forma per evitare rilevamento
- **Metamorphic**: riscrive completamente il proprio codice
- **Multipartite**: combina più tecniche di infezione

Ciclo di vita:

1. Infezione
2. Fase dormiente (opzionale)
3. Attivazione (trigger)
4. Replicazione
5. Payload (azione dannosa)

2.1.2 Worm

- **Definizione**: Programma autonomo che si propaga automaticamente attraverso la rete
- **Caratteristiche**:
  - Non richiede intervento umano
  - Sfrutta vulnerabilità di rete
  - Non necessita di file host
  - Consuma risorse di rete e sistema

Metodi di propagazione:

- Vulnerabilità nei servizi di rete
- Email (allegati auto-eseguenti)
- Condivisione file
- Messaggistica istantanea
- Dispositivi rimovibili

Esempi storici:

- Morris Worm (1988): primo worm significativo
- ILOVEYOU (2000): si diffuse via email
- SQL Slammer (2003): sfruttava vulnerabilità SQL Server
- Conficker (2008): multiple vettori di infezione
- WannaCry (2017): ransomware con componente worm

2.1.3 Trojan

- **Definizione**: Malware mascherato da software legittimo o utile
- **Caratteristiche**:
  - Non si replica
  - Induce l'utente all'installazione
  - Spesso crea backdoor
  - Apparenza innocua, funzionalità malevole

Categorie:

- **Backdoor**: fornisce accesso remoto

- **Downloader**: scarica altro malware
- **Banker**: ruba credenziali bancarie
- **RAT** (Remote Access Trojan): controllo completo
- **Spyware**: raccoglie informazioni
- **Keylogger**: registra ciò che viene digitato
- **FakeAV**: finto antivirus

Vettori di infezione:

- Download da siti non affidabili
- Email di phishing
- Pubblicità malevola
- Software piratato
- Software bundle

2.1.4 Ransomware

- **Definizione**: Malware che cifra i dati dell'utente e chiede un riscatto per la decifratura
- **Fasi tipiche**:
  1. Infezione (phishing, vulnerabilità)
  2. Scansione file
  3. Crittografia (spesso asimmetrica)
  4. Richiesta riscatto
  5. Pagamento (spesso in cryptocurrency)
  6. Decifratura (non garantita)

Tipi:

- **Locker**: blocca l'accesso al sistema
- **Crypto**: cifra i file
- **Doxware/Leakware**: minaccia di pubblicare dati sensibili
- **Mobile ransomware**: colpisce dispositivi mobili

Esempi noti:

- CryptoLocker (2013): primo ransomware di grande impatto
- WannaCry (2017): pandemia globale
- Ryuk: prende di mira le aziende
- REvil/Sodinokibi: ransomware-as-a-service

2.1.5 Altri tipi di malware

- **Adware**: mostra pubblicità indesiderata

- **Spyware**: raccoglie informazioni sull'utente
- **Rootkit**: nasconde la presenza di malware
- **Bootkit**: infetta il boot loader
- **Keylogger**: registra input da tastiera
- **Cryptojacker**: usa risorse per mining di criptovalute
- **Botnet**: rete di dispositivi compromessi controllati remotamente
- **Fileless malware**: opera solo in memoria senza file su disco
- **Logic bomb**: si attiva quando si verificano determinate condizioni

2.2 Vettori di infezione

2.2.1 Email e allegati

- **Phishing**: impersonifica entità legittime
  - Spear phishing: mirato a individui specifici
  - Whaling: mirato a dirigenti o figure chiave
  - Clone phishing: replica di email legittime
- **Allegati malevoli**:
  - Documenti con macro
  - Eseguibili mascherati (.exe, .scr)
  - Archive con password contenenti malware
  - Script (.js, .vbs, .ps1)
- **Link a siti malevoli**:
  - URL ingannevoli (typosquatting)
  - Pagine di login false
  - Drive-by download

Indicatori di rischio:

- Richieste urgenti
- Errori grammaticali
- Indirizzi mittente sospetti
- Incongruenze tra nome visualizzato e email
- Allegati inaspettati
- Link hover che mostra URL diverso

2.2.2 Download da Internet

- **Software da fonti non affidabili**:
  - Siti di warez/pirateria
  - App store non ufficiali
  - Download da siti di annunci

- **Drive-by download**:
  - Download automatico visitando siti compromessi
  - Sfrutta vulnerabilità del browser/plugin
  - Non richiede interazione dell'utente
- **Bundleware/PUP**:
  - Software aggiuntivo installato con programmi legittimi
  - Spesso nascosto in opzioni avanzate/personalizzate
  - Cambio homepage/motore di ricerca
- **Crack e keygen**:
  - Spesso contengono malware
  - Disattivano protezioni antivirus

2.2.3 Dispositivi rimovibili

- **USB infetti**:
  - Autorun/AutoPlay (meno comune nei sistemi moderni)
  - File con icone ingannevoli
  - LNK exploit
  - DLL hijacking
- **BadUSB**:
  - Firmware USB modificato
  - Si presenta come tastiera/dispositivo HID
  - Esegue comandi automaticamente
- **Cross-infezione**:
  - Trasferimento tra sistemi fisicamente isolati
  - Vettore per attacchi ad air-gapped networks

Misure preventive:

- Disabilitare AutoPlay/AutoRun
- Scansione automatica dispositivi rimovibili
- Whitelisting dispositivi
- Formattazione prima dell'uso

2.3 Protezione e prevenzione

2.3.1 Antivirus e antimalware

- **Metodologie di rilevamento**:
  - **Signature-based**: confronto con database di firme note
  - **Heuristic-based**: analisi comportamentale
  - **Behavioral monitoring**: osservazione attività in tempo reale

    - **Sandbox**: esecuzione in ambiente isolato
    - **Cloud-based**: analisi in tempo reale via cloud
    - **Machine learning**: rilevamento modelli sospetti

Componenti di protezione:

- Scanner on-demand
- Protezione in tempo reale
- Web protection
- Email scanning
- Sandbox automation
- Firewall integrato
- HIPS (Host Intrusion Prevention System)

Limitazioni:

- Inefficaci contro malware avanzato/zero-day
- Falsi positivi/negativi
- Impatto sulle prestazioni
- Basati su reazione, non prevenzione

2.3.2 Best practices di sicurezza

- **Aggiornamenti regolari**:
    - Sistema operativo
    - Software applicativo
    - Driver
    - Firmware
- **Backup frequenti**:
    - Regola 3-2-1: 3 copie, 2 supporti diversi, 1 off-site
    - Backup incrementali/differenziali
    - Test di ripristino regolari
    - Backup air-gapped (disconnessi dalla rete)
- **Principio del privilegio minimo**:
    - Account utente standard per uso quotidiano
    - Elevazione temporanea per attività amministrative
    - Restrizione accessi basata sul ruolo
- **Email filtering**:
    - Filtri anti-spam
    - Scansione allegati
    - Link sanitization
    - SPF/DKIM/DMARC

- **Firewall personale**:
  - Filtraggio connessioni in entrata e uscita
  - Controllo applicazioni
  - Protezione da intrusioni
  - Segmentazione rete
- **Formazione degli utenti**:
  - Riconoscere phishing
  - Pratiche sicure password
  - Condivisione dati sicura
  - Social engineering awareness

2.3.3 Sandbox e virtualizzazione

- **Definizione**: Ambiente isolato per esecuzione sicura di software potenzialmente pericoloso
- **Tipi**:
  - **Application sandbox**: limita le capacità di un'applicazione
  - **System sandbox**: simula un intero sistema
  - **Browser sandbox**: isola il browser dal sistema
  - **Sandbox dispositivi**: iOS, Android

Implementazioni:

- **Virtual machines**: isolamento completo
- **Containers**: isolamento leggero
- **Jail** (FreeBSD): partizione di risorse
- **AppContainer/AppLocker**: Windows
- **SELinux/AppArmor**: Linux

Vantaggi:

- Protezione del sistema principale
- Analisi sicura di malware
- Ripristino rapido dopo infezione
- Test di software sospetto
- Isolamento di applicazioni critiche

2.4 Attacchi avanzati e APT

2.4.1 Advanced Persistent Threats (APT)

- **Definizione**: Attacchi prolungati e mirati condotti da attori sofisticati
- **Caratteristiche**:

- Lunga durata (mesi/anni)
- Obiettivi specifici (spionaggio, sabotaggio)
- Risorse significative (spesso state-sponsored)
- Multiple tecniche e vettori
- Evasione avanzata del rilevamento

Fasi tipiche:

1. **Ricognizione**: raccolta informazioni target
2. **Weaponization**: preparazione malware/exploit
3. **Delivery**: phishing mirato, watering hole
4. **Exploitation**: sfruttamento vulnerabilità
5. **Installation**: backdoor, rootkit
6. **Command & Control**: stabilire canale comunicazione
7. **Actions on Objectives**: esfiltrazione dati, sabotaggio

2.4.2 Zero-day exploit

- **Definizione**: Attacco che sfrutta vulnerabilità sconosciute al vendor
- **Caratteristiche**:
  - Nessuna patch disponibile
  - Altamente efficace
  - Spesso venduti sul dark web
  - Utilizzati in attacchi mirati/APT
  - Difficili da rilevare

Difese:

- Defense in depth
- Behavioral analysis
- Threat hunting
- Whitelisting applicazioni
- Segmentazione rete
- Principio del privilegio minimo

2.4.3 Supply chain attack

- **Definizione**: Compromissione di un'organizzazione attraverso partner fidati
- **Vettori**:
  - Software compromesso dal vendor
  - Hardware manomesso
  - Terze parti compromesse

- Aggiornamenti software infetti

Esempi noti:

- SolarWinds (2020)
- NotPetya via MEDoc
- CCleaner hack
- Target breach via HVAC vendor

Mitigazioni:

- Vendor assessment
- Verifica integrità software
- Code signing verification
- Zero trust approach