Dispense Universitarie: Network Protocols & Security Fundamentals

Corso: Cyber Security & Ethical Hacking - Fondamenti di Network **Modulo:** Protocolli di Rete (S2L1)

---

1. Fondamenti e Introduzione ai Protocolli di Rete

1. 📖 Spiegazione Approfondita

Lo studio dei protocolli di rete costituisce la spina dorsale dell'informatica moderna e della sicurezza informatica. Un protocollo di rete non è altro che un **insieme di regole formalizzate e convenzioni** che disciplinano il formato, la tempistica, il sequenziamento e il controllo degli errori nello scambio di messaggi tra dispositivi elettronici.

Senza questi standard, dispositivi di produttori diversi (interoperabilità) non potrebbero comunicare. I protocolli garantiscono:

- **Interoperabilità:** Un sistema Linux può inviare dati a un sistema Windows.
- **Affidabilità:** Meccanismi per assicurare che i dati arrivino integri.
- **Efficienza:** Ottimizzazione delle risorse di rete.
- **Sicurezza:** Protezione contro accessi non autorizzati (se il protocollo prevede crittografia).

I protocolli operano a diversi livelli del modello OSI. Ad esempio:

- **Livello Fisico/Data Link:** Ethernet, Wi-Fi.
- **Livello di Rete:** IP, ICMP.
- **Livello di Trasporto:** TCP (affidabile), UDP (veloce, senza connessione).
- **Livello Applicazione:** HTTP, FTP, SMTP, ecc.

3. 🧠 Analisi Tecnica & Memorizzazione

- **Definizione Chiave:** "Un protocollo è un linguaggio standardizzato che definisce sintassi, semantica e sincronizzazione della comunicazione tra entità hardware o software."
- **Contesto Reale:** Ogni volta che apri un browser, invii una mail o ti connetti al Wi-Fi, stai attivando una pila (stack) di protocolli che lavorano in concerto.

5. ⚠ Focus Esame

- **Domanda:** Qual è la differenza tra un protocollo di trasporto e uno di applicazione?
  - *Risposta:* Il trasporto (TCP/UDP) gestisce *come* i dati vengono spostati affidabilmente o velocemente; l'applicazione (HTTP/FTP) definisce *cosa* sono quei dati e come l'utente li utilizza.

---

2. TELNET (Teletype Network)

1. 📖 Spiegazione Approfondita

Porta: 23 (TCP)

Telnet è uno dei protocolli più antichi (1969), progettato per fornire una comunicazione bidirezionale interattiva tramite interfaccia a riga di comando (CLI). Permette a un utente di collegarsi a una macchina remota ed eseguire comandi come se fosse seduto davanti al terminale locale.

Sebbene sia stato fondamentale per l'amministrazione remota e il debugging, Telnet soffre di una criticità fatale per gli standard moderni: **non è sicuro**.

Caratteristiche e Vulnerabilità:

- **Trasmissione in Chiaro (Cleartext):** Tutto il traffico, inclusi username e password, viaggia senza crittografia.
- **Mancanza di Autenticazione Forte:** Non supporta nativamente certificati o meccanismi moderni.
- **Sniffing & MITM:** Un attaccante sulla stessa rete può usare strumenti come Wireshark per leggere le credenziali passate via Telnet (Man-in-the-Middle).

2. 🛠 Sintassi & Comandi

Sebbene deprecato per l'amministrazione, il client Telnet è ancora usato per testare la connettività su porte specifiche (Banner Grabbing).

- **Comando Base:**

  ```
  telnet [indirizzo_ip_o_hostname] [porta]
  ```

- **Esempio di Debug:**

  `telnet google.com 80` *Spiegazione:* Apre una connessione TCP sulla porta 80 per verificare se il web server risponde, utile per diagnosticare problemi di firewall.

3. 🧠 Analisi Tecnica & Memorizzazione

- **Contesto Reale:** Oggi Telnet non si usa quasi mai per amministrare server (sostituito da SSH). Si trova ancora su vecchi apparati industriali (legacy) o router non configurati (cattiva pratica), o viene usato dagli amministratori per il *troubleshooting* rapido delle porte aperte.

4. 🔗 Cross-Connection Master (Pentesting)

Durante un Penetration Test, trovare la porta 23 aperta è una vulnerabilità critica. Permette di tentare attacchi di *Brute Force* o, se si ha accesso alla rete, di sniffare le credenziali "on the wire".

---

3. FTP (File Transfer Protocol)

1. 📖 Spiegazione Approfondita

Porte: 20 (Dati), 21 (Controllo) - TCP

FTP è il protocollo standard per il trasferimento di file tra client e server. È un protocollo complesso che utilizza **due canali separati**:

1. **Canale di Comando (Porta 21):** Dove passano i comandi (USER, PASS, LIST, RETR) e le risposte di stato.
2. **Canale Dati (Porta 20 o Random):** Dove passano effettivamente i file o le liste delle directory.

Modalità di Connessione:

- **Attiva:** Il client apre una porta random e il server si connette al client (problematica con i firewall lato client).
- **Passiva (PASV):** Il client chiede al server di aprire una porta per i dati. Il server risponde con IP e Porta, e il client si connette. Questa è la modalità standard moderna per superare problemi di NAT/Firewall.

Vulnerabilità: Come Telnet, FTP standard trasmette tutto in chiaro, rendendo facile il furto di credenziali e dati.

2. 🛠 Sintassi & Comandi (Interazione Manuale)

Le slide mostrano un esempio avanzato di interazione FTP usando `netcat` (nc) o comandi raw, utile per capire il protocollo "sotto il cofano".

Comandi FTP Interni (RFC 959):

- `USER [nome]`: Invia username.
- `PASS [password]`: Invia password.
- `PASV`: Richiede modalità passiva.
- `LIST`: Elenca i file.
- `RETR [nomefile]`: Scarica (Retrieve) un file.
- `STOR [nomefile]`: Carica (Store) un file.

Calcolo della Porta in Modalità Passiva: Quando il server risponde al comando `PASV` con: `227 Entering Passive Mode (44,241,66,173,4,9)` I numeri finali (4, 9) indicano la porta secondo la formula:

> **Porta = (P1 * 256) + P2** Esempio: `(4 * 256) + 9 = 1024 + 9 = 1033`

Scripting Bash (dalle slide):

> `nc ftp.dlptest.com $(($1 * 256 + $2))` *Spiegazione:* Usa la shell per calcolare dinamicamente la porta ricevuta e connettersi al canale dati.

5. ⚠ Focus Esame

- **Trabocchetto:** Quale porta usa FTP?
  - *Risposta Corretta:* 21 per il controllo, ma la porta dati varia (20 in attiva, >1023 in passiva). Non dire solo "21".
- **Sicurezza:** Perché FTP è insicuro? Perché non cifra né comandi né dati.

---

4. FTPS (FTP Secure)

1. 📖 Spiegazione Approfondita

Porte: 21 (Controllo), 989/990 (Dati/Controllo Implicito)

FTPS è l'evoluzione sicura di FTP. Aggiunge un livello di crittografia **SSL/TLS** al protocollo standard. Non va confuso con SFTP (che è basato su SSH).

Differenze Chiave con FTP:

- **Crittografia:** Utilizza certificati X.509 (SSL/TLS) per cifrare la sessione (simmetrica per i dati, asimmetrica per l'handshake).
- **Autenticazione:** Supporta username/password cifrati e, opzionalmente, autenticazione tramite certificato client.

Metodi di Negoziazione:

1. **Implicito (Legacy):** Si connette subito su una porta sicura (es. 990).
2. **Esplicito (Moderno):** Si connette sulla porta standard 21 e il client richiede esplicitamente la sicurezza inviando il comando `AUTH TLS`.

3. 🧠 Analisi Tecnica & Memorizzazione

- **Contesto Reale:** Usato in ambienti aziendali legacy dove si vuole rendere sicuro il trasferimento file senza cambiare l'infrastruttura server FTP esistente in server SSH.

---

5. DHCP (Dynamic Host Configuration Protocol)

1. 📖 Spiegazione Approfondita

Porte: 67 (Server), 68 (Client) - UDP

DHCP è un protocollo fondamentale che automatizza la configurazione di rete. Invece di configurare manualmente IP, Subnet Mask, Gateway e DNS su ogni dispositivo (Static IP), un server DHCP li assegna dinamicamente.

Concetto di "Lease" (Affitto): L'indirizzo IP non è assegnato per sempre, ma per un tempo limitato (Lease Time). Alla scadenza (o meglio, a metà tempo - T1 timer), il client deve chiedere il rinnovo.

Il Processo DORA (Memorizzare l'acronimo):

1. **D - Discovery:** Il client, appena acceso e senza IP, urla in broadcast (`0.0.0.0` -> `255.255.255.255`): "C'è un server DHCP qui?".
2. **O - Offer:** I server DHCP che ricevono la richiesta rispondono (unicast o broadcast): "Io sono disponibile, ecco un IP (es. 192.168.1.10)".
3. **R - Request:** Il client sceglie un'offerta e risponde: "Accetto l'IP 192.168.1.10 dal server X".
4. **A - Acknowledge:** Il server conferma: "Ok, l'IP è tuo per 24 ore. Ecco anche DNS e Gateway".

2. 🛠 Sintassi & Comandi

- `ipconfig /release` (Windows) / `dhclient -r` (Linux): Rilascia l'IP (invia pacchetto DHCPRELEASE).
- `ipconfig /renew` (Windows) / `dhclient` (Linux): Avvia il processo DORA per un nuovo IP.

5. ⚠ Focus Esame & Sicurezza (DHCP Attacks)

- **DHCP Starvation:** Un attaccante simula migliaia di client diversi (spoofing MAC address) e chiede tutti gli IP disponibili fino a esaurire il pool del server. I client legittimi non riceveranno più IP.
    - *Mitigazione:* Port Security sui switch.
- **Rogue DHCP Server:** Un attaccante installa un finto server DHCP. Se è più veloce del server legittimo, assegna ai client un Gateway malevolo, intercettando tutto il traffico (Man-in-the-Middle).
    - *Mitigazione:* DHCP Snooping sui switch.

---

6. HTTP (HyperText Transfer Protocol)

1. 📖 Spiegazione Approfondita

Porta: 80 (TCP)

HTTP è il protocollo alla base del World Wide Web. È un protocollo **stateless** (senza memoria tra una richiesta e l'altra) basato sul modello richiesta/risposta. Nato al CERN, si è evoluto dalla versione 0.9 alla 1.1 (standard de facto per anni), fino a HTTP/2 (multiplexing) e HTTP/3 (basato su UDP/QUIC).

Metodi HTTP (Verbi):

- **GET:** Richiede una risorsa (es. pagina web). Parametri in URL.
- **POST:** Invia dati al server (es. login, form). Dati nel corpo (body). Modifica lo stato.
- **PUT:** Carica/Sostituisce una risorsa specifica.
- **DELETE:** Cancella una risorsa.
- **HEAD:** Come GET, ma scarica solo gli header (utile per check veloci).
- **OPTIONS:** Chiede al server quali metodi sono supportati.

Codici di Stato (Status Codes):

- **1xx:** Informational (es. 100 Continue).
- **2xx:** Successo (es. **200 OK**, 201 Created).
- **3xx:** Redirezione (es. **301 Moved Permanently**, 302 Found/Temporary).
- **4xx:** Errore Client (es. **400 Bad Request**, **401 Unauthorized** [serve login], **403 Forbidden** [login ok ma no permessi], **404 Not Found**).
- **5xx:** Errore Server (es. **500 Internal Server Error**, **503 Service Unavailable**).

4. 🔗 Cross-Connection Master (Web Security)

HTTP trasmette in chiaro. Iniezioni (SQLi, XSS) e furto di sessione avvengono spesso manipolando i parametri GET/POST o gli header.

---

7. HTTPS (HTTP Secure)

1. 📖 Spiegazione Approfondita

Porta: 443 (TCP)

HTTPS non è un protocollo diverso, è HTTP incapsulato dentro un tunnel **TLS (Transport Layer Security)**, successore di SSL. Garantisce:

1. **Confidenzialità:** I dati sono cifrati (nessuno può leggerli sniffando).
2. **Integrità:** I dati non sono stati modificati in transito.
3. **Autenticazione:** Il certificato digitale garantisce che stai parlando col server vero (es. www.google.com) e non con un imitatore.

Il TLS Handshake (Semplificato):

1. **Client Hello:** Il client propone versioni TLS e cifrari (Cipher Suites).
2. **Server Hello:** Il server sceglie il cifrario e invia il suo **Certificato Digitale** (che contiene la Chiave Pubblica).
3. **Verifica:** Il client verifica se il certificato è valido e emesso da una CA (Certification Authority) fidata.
4. **Scambio Chiavi:** Usando crittografia asimmetrica (es. RSA o Diffie-Hellman), client e server concordano una **Chiave di Sessione Simmetrica**.
5. **Sessione Sicura:** Da qui in poi, tutto è cifrato velocemente con la chiave simmetrica.

5. ⚠ Focus Esame

- **SEO & Trust:** Google penalizza i siti non-HTTPS. Il lucchetto nel browser indica la validità del certificato.
- **Vulnerabilità:** HTTPS non protegge il server dagli attacchi web (SQL Injection passano attraverso il tunnel cifrato), protegge solo il *trasporto* dei dati.
- **HSTS (HTTP Strict Transport Security):** Un header che il server invia per forzare il browser a usare *sempre* HTTPS in futuro, prevenendo attacchi di Downgrade (SSL Stripping).

---

8. Protocolli Email (SMTP, POP, IMAP)

1. 📖 Spiegazione Approfondita

La posta elettronica si basa su tre protocolli distinti per invio e ricezione.

SMTP (Simple Mail Transfer Protocol):

- **Porta:** 25 (Standard), 587 (Submission/Secure).
- **Ruolo:** Il "Postino". Serve per **INVIARE** mail dal client al server, e per spostare mail da un server all'altro (Relaying).
- **Funzionamento:** Push protocol. Usa comandi testuali come `HELO`, `MAIL FROM`, `RCPT TO`, `DATA`.

POP3 (Post Office Protocol v3):

- **Porta:** 110 (Default), 995 (SSL/TLS).
- **Ruolo: RICEZIONE**.
- **Comportamento:** "Store and Forward". Scarica la mail sul dispositivo locale e (di solito) la cancella dal server.
- **Svantaggio:** Non sincronizza. Se leggi una mail sul PC, sul telefono risulta non letta o assente. Adatto per uso mono-dispositivo.

IMAP (Internet Message Access Protocol):

- **Porta:** 143 (Default), 993 (SSL/TLS).
- **Ruolo: RICEZIONE & GESTIONE**.

- **Comportamento:** Le mail restano sul server. Il client scarica solo intestazioni o copie temporanee.
- **Vantaggio:** Sincronizzazione totale (stati letto/non letto, cartelle, bozze) su tutti i dispositivi.

5. ⚠ Focus Esame

- **Differenza POP vs IMAP:** POP scarica e rimuove (locale), IMAP sincronizza e mantiene (server).
- **Sicurezza Email:** SMTP puro è in chiaro. Si usano SPF, DKIM e DMARC per prevenire lo spoofing (falsificazione mittente) e TLS per cifrare il trasporto.

---

9. SSH (Secure Shell)

1. 📖 Spiegazione Approfondita

Porta: 22 (TCP)

SSH è il sostituto sicuro di Telnet. Permette amministrazione remota crittografata. Oltre al terminale, offre funzionalità avanzate come:

- **Tunneling/Port Forwarding:** Incapsulare altri protocolli insicuri dentro SSH.
- **SFTP/SCP:** Trasferimento file sicuro.
- **X11 Forwarding:** Esecuzione di programmi grafici remoti.

Autenticazione:

1. **Password:** Metodo classico (sconsigliato per rischio Brute Force).
2. **Chiave Pubblica (Public Key Auth):** Si genera una coppia di chiavi (Privata sul PC, Pubblica sul Server). Il server invia una "sfida" cifrata con la pubblica che solo chi ha la privata può risolvere. Molto più sicuro.

3. 🧠 Analisi Tecnica & Memorizzazione

- **Best Practices:** Disabilitare login root via SSH, cambiare porta standard (security by obscurity, debole ma riduce il rumore dei log), usare solo chiavi SSH e disabilitare password.

---

10. NetBIOS & Samba (SMB)

1. 📖 Spiegazione Approfondita

NetBIOS (Network Basic Input/Output System):

- **Porte:** 137 (UDP - Name), 138 (UDP - Datagram), 139 (TCP - Session).
- **Descrizione:** API legacy per reti locali. Gestisce nomi computer (es. "PC-UFFICIO") e sessioni. Molto "chiacchierone" (broadcast), espone informazioni sulla rete interna.

Samba / SMB (Server Message Block):

- **Porta:** 445 (TCP - SMB over IP diretto).
- **Descrizione:** Protocollo per **condivisione di file e stampanti**. Samba è l'implementazione Open Source (Linux) che permette a macchine Unix di integrarsi in domini Windows (Active Directory).
- **Storia:** Nato per far parlare Windows e Unix.

5. ⚠ Focus Esame (Sicurezza)

- **SMB è critico:** Vulnerabilità storiche come **EternalBlue** (WannaCry Ransomware) sfruttavano bug nel protocollo SMBv1.
- **Enumerazione:** NetBIOS è una miniera d'oro per gli hacker (enumera nomi utenti, gruppi, share condivise). Va bloccato verso Internet.

---

11. SNMP (Simple Network Management Protocol)

1. 📖 Spiegazione Approfondita

Porte: 161 (Polling/Richieste - UDP), 162 (Trap/Notifiche - UDP).

Protocollo per monitorare e gestire apparati di rete (Router, Switch, Stampanti, Server). **Componenti:**

- **Manager (NMS):** Il server che controlla (es. Nagios, Zabbix).
- **Agent:** Il software che gira sul dispositivo da controllare.
- **MIB (Management Information Base):** Database gerarchico delle impostazioni configurabili.
- **OID (Object Identifier):** L'indirizzo numerico di una specifica impostazione (es. 1.3.6.1.2... indica "CPU Load").

Operazioni:

- **GET:** Il Manager chiede un dato.
- **SET:** Il Manager cambia una configurazione.
- **TRAP:** L'Agent segnala spontaneamente un problema (es. "Interfaccia Down").

Versioni & Sicurezza:

- **v1/v2c:** Usano "Community String" (una password) in **chiaro**. Se sai la stringa "public" (default sola lettura) o "private" (lettura/scrittura), controlli il router.
- **v3:** Introduce cifratura e autenticazione forte. È l'unico standard sicuro.

---

12. Laboratorio: Configurazione Ambiente di Sviluppo (VS Code & GCC su Kali)

1. 📖 Spiegazione Approfondita

[cite_start]Per affrontare i moduli successivi di programmazione (C e Python) e analisi del codice, è necessario predisporre un ambiente di sviluppo integrato (IDE) robusto all'interno della macchina virtuale Kali Linux[cite: 103].

Sebbene Kali Linux fornisca nativamente molti tool, l'installazione di **Visual Studio Code (VS Code)** è consigliata per la sua versatilità, il supporto al debugging e l'ampio ecosistema di estensioni.
[cite_start]Essendo Kali una distribuzione basata su **Debian**, utilizzeremo i pacchetti `.deb` ufficiali[cite: 103].

Gestione delle Architetture: È fondamentale scaricare la versione corretta per l'hardware in uso:

- [cite_start]**x64 / amd64:** Per PC standard (Intel/AMD) o VirtualBox su Mac Intel[cite: 103].

- [cite_start]**ARM64:** Per Mac con processori Apple Silicon (M1/M2/M3) che utilizzano virtualizzatori come UTM[cite: 103].

[cite_start]Oltre all'editor, installeremo **GDB (GNU Debugger)**, uno strumento essenziale per l'analisi dinamica dei programmi C, il reverse engineering e l'exploit development (es. buffer overflow)[cite: 106].

2. 🛠 Sintassi & Comandi (Cleaning)

Ecco la procedura corretta e pulita per l'installazione, correggendo gli errori di OCR presenti nelle slide.

Fase 1: Download e Posizionamento Scaricare il pacchetto `.deb` dal sito ufficiale. Successivamente, aprire il terminale e spostarsi nella cartella dei download:

> `cd ~/Downloads` [cite_start]*(Nota: Su sistemi localizzati in italiano potrebbe essere `cd ~/Scaricati`)* [cite: 104]

Fase 2: Installazione del Pacchetto Usare `apt` (Advanced Package Tool) per installare il file locale. Il percorso `./` è cruciale per indicare al gestore pacchetti che si tratta di un file nella directory corrente e non di un pacchetto remoto.

> `sudo apt install ./<nome_file_scaricato>.deb` [cite_start]*Esempio reale:* `sudo apt install ./code_1.85.1-1702462158_amd64.deb` [cite: 104]

Fase 3: Configurazione Repository Microsoft Durante l'installazione, potrebbe apparire una schermata interattiva (ncurses) che chiede se aggiungere il repository Microsoft.

- **Azione:** Selezionare **"Sì" (Yes)**.
- [cite_start]*Motivo:* Questo permette di ricevere gli aggiornamenti futuri di VS Code direttamente tramite il comando `sudo apt upgrade`, senza dover scaricare manualmente nuove versioni[cite: 105].

Fase 4: Installazione Debugger (GDB) VS Code ha bisogno di un debugger backend per analizzare codice C/C++.

> `sudo apt update sudo apt install -y gdb` [cite_start][cite: 106]

Fase 5: Avvio Per avviare VS Code direttamente nella cartella di lavoro corrente:

> `code .` [cite_start]Oppure cercare "Visual Studio Code" nel menu delle applicazioni[cite: 106].

3. 🧠 Analisi Tecnica & Memorizzazione

- **Perché GDB?** Non è solo un debugger per sviluppatori. In Cybersecurity, GDB è il bisturi per dissezionare la memoria, analizzare i registri della CPU e capire perché un exploit funziona o fallisce.
- **Repository APT:** Aggiungere un repository esterno (come quello Microsoft) modifica il file `/etc/apt/sources.list` o aggiunge un file in `/etc/apt/sources.list.d/`, istruendo il sistema su dove cercare gli aggiornamenti.

4. 🔗 Cross-Connection Master (Malware Analysis)

In scenari avanzati di **Malware Analysis**, useremo debugger più visuali o specifici (come x64dbg su Windows o Ghidra), ma GDB rimane lo standard de facto per l'analisi su sistemi Linux e per il debugging remoto di

dispositivi embedded/IoT.

5. ⚠ Focus Esame

- **Domanda:** Quale comando installa un pacchetto `.deb` locale risolvendo le dipendenze?
  - *Risposta:* `sudo apt install ./pacchetto.deb` (preferibile al vecchio `dpkg -i` perché `apt` gestisce meglio le dipendenze mancanti).
- **Architettura:** Ricordare la differenza tra `amd64` (Intel/AMD 64-bit) e `arm64` (Apple Silicon/Raspberry Pi), scaricare il pacchetto sbagliato impedirà l'installazione.