**Standard:**

È un concetto per il quale si indica un protocollo, il quale è un insieme di regole comuni, che tutti quindi lo riconoscono. Per quello sono dette standardizzate. In virtù di questo standard ci permetterà di lavorare con molteplici dispositivi con HW e SO diversi.

PDU 🡺 protocol data unit. Esso è la struttura del messaggio del protocollo formato da campi. Questo viene definito come device software. Hardware invece si intende la codifica/decodifica del segnale.

I protocolli hanno delle loro caratteristiche proprie:

* Function: esempio di funzione all’interno del protocollo: Trasferimento della pagina web (HTTP), esso è all’interno di una pila di protocolli definita SUITE. Nella suite si inizia dal protocollo più basso (fisico) fino ad arrivare a quelli più alti. (Vedi seconda tabella)
* Format: Formato del messaggio del protocollo [PDU] a volte essi nei campi contengono delle informazioni ridondanti. Quindi il formato definisce i campi del messaggio.
* Regole: Come avviene regolamentata la comunicazione.

Esempi di alcune tipologie di protocolli:

|  |  |
| --- | --- |
| **Comunicazioni** all’interno della rete | **Abilitare** due o più device per comunicare nella rete/tra reti |
| **Sicurezza** all’interno della rete | **Autentificazione** [tramite credenziali], **integrità** dei dati [salvati nel Cloud e verificare che essi non sono stati modificati, in maniera maliziosa o non-intenzionale], **cifratura** dei dati |
| **Instradamento** | **Struttura** della rete. Tramite il router i vari dispositivi possono comunicare fra di loro, e creano una strada. **Routing table** è la tabella in cui il dispositivo andrà a decidere su quale della interfacce dovrà instradare il dato affinché arrivi a destinazione. Quindi il **router** individua il cammino corretto. |
| **Service Discovery** | Il router ha la capacità di trovare **automaticamente** i suoi device vicini. |

A livello di funzioni dei protocolli ne troviamo molteplici, qui di seguito ne sono elencate alcune:

* **Addressing (indirizzamento):** Identificare il mittente e il ricevente, in maniera univoca. Esistono vari livelli di indirizzamento, ad esempio: quando si cambia la rete si cambia indirizzo ecc. Indirizzi IP ad esempio sono logici, invece quelli MAC sono fisici, il quale è univoco e NON è possibile cambiarlo.
* **Affidabilità (Reliability):** proprietà per cui si rilasciano i dati, con avvenuta **conferma**. Essi sono affidabili, siccome arriva ma SENZA modifiche. **Unreliability** il contrario dei Reliability, quindi non sono affidabili a livello di sicurezza nella modifica dei dati, cosa molto utile se si sta parlando di servizi di streaming. Siccome anche se ci sono errori nella trasmissioni non si rimanda il dato ma si lascia così, cosa assolutamente inversa nei protocolli Reliability. Si preferisce fare così siccome si preferisce la velocità che la correttezza dei dati mandati.
* **Controllo di flusso (Flow control):** operazione che consiste nella trasmissione seriale dei bit. Per fare questo abbiamo bisogno di avere una velocità di trasmissione e ricezione equivalente. Buffferizza, controlla l’errore e poi viene passata il dato 🡺 questi sono i passaggi che effettua il device ricevente. Passaggi più lunghi di chi lo manda solamente. Se non viene gestito correttamente si può andare in OVERRUN quindi si andrebbero a perdere i dati inviati. Il controllo di flusso controlla che questo non succeda. (Per maggiori appunti vedere lezione 3).
* **Sequencing**: Non sempre i dati prendono lo stesso cammino, avendo quindi varie tratte. Quindi siccome i dati possono arrivare in ordine sparso, viene utilizzato questo concetto di Sequencing, che tramite l’identificatore va a costruire la corretta sequenza del messaggio.
* **Error Detection**: controllo delle errore viene fatto dalla scheda di rete, componente HW quindi mi dà più tempo di esecuzione e non è flessibile. L’algoritmo che **controlla l’errore** è CRC 32bit. A fronte della verifica dell’errore per correggerlo semplicemente lo scarta, senza recuperarlo e non richiedendo l’invio. Nel caso di protocollo confermato, esso va a creare un TIMEOUT che viene attivato quando mandiamo il messaggio, se la conferma del messaggio non arriva entro la fine del time-out allora manda un messaggio di errore, così il mittente può capire se ci sono o meno errori nei dati inviati.

Come sono strutturati le interazioni tra protocolli?

I protocolli possono interagire con i protocolli in alto e in basso rispetto alla sua posizione. Sul nostro device è presente il modello TCP – IP (modello a livelli). Siccome ogni livello è programmato per un effettuare una determinata funzione. Questo modello deriva dal modello OSI (7 livelli rispetto ai 4 del TCP – IP).

**Ci sono diversi livelli:**

* Livello **4**: livelli applicativi (che si interfacciano con le applicazioni) come l’HTTP (trasferimento di protocolli, che gestisce la navigazione). Esso ad esempio gestisce come interagiscono web server e client. Il browser, o i client di tipo Host ad esempio: client, però non è all’interno del protocollo siccome NON sono standardizzati. Comunicazione END-TO-END.
* Livello **3**: Gestisce le conversazioni individuali, anche se si sta utilizzando più servizi, colui che traccia le varie conversazioni è il TCP. Garantisce l’invio e controlla il Flow Control, controllano l’OVERRUN. Esso è un protocollo di trasporto.
* Livello **2**: Questo livello, IP, si occupa dell’instradamento: al fine che possa raggiungere la destinazione correttamente.
* Livello **1**: Protocollo Ethernet, quello cablato, che gestisce il collegamento con la rete. Esso prende i bit dal livello 2 e li inserisce nella rete. Comunicazione POINT-TO-POINT.

I livelli più alti fanno delle funzioni molto più complesse rispetto che i livelli più bassi, vedi il primo che va a gestire il collegamento FISICO.

**Protocol Suites:**

È un insieme di protocolli che sono correlati fra di loro, in particolare tra di loro comunicano. Essi comunicano PEER-TO-PEER quando bisogna fare una comunicazione allo stesso livello ma tra nodi diversi.

Esempi di Suites:

* TCP/IP: gestito dall’IETF (ente che gestisce le caratteristiche di internet).
* OSI: sistemi eterogeni possono comunicare fra di loro e non come quelli proprietario. Esso è stato emanato dall’ente ISO (un organizzazione per Standard).
* Apple Talk: proprietario Apple, non più usato.
* Novell NetWare

TCP IP: (vedi presentazione)

* Livello 4:
* Livello 3: Transport
* Livello 2: internet
* Livello 1: Network access, il quale è Ethernet.

Esso include diversi protocolli. TCP/IP è una suite aperta che può essere utilizzato da qualsiasi venditore siccome accesso al pubblico.

Interoperabilità: Capacità di due o più sistemi, reti, mezzi, applicazioni o componenti, di scambiare informazioni tra loro e di essere poi in grado di utilizzarle.

Esempi di protocolli nel livello 4:

* DNS: converte il nome ipertestuale ad un numero formato da bit.
* DHSP
* POP3
* IMAP
* TFTP, HTTP, HTTPS.

Esempi di protocolli nel livelli 3:

* TCP
* UDP (non Reliability rispetto TCP).

Esempi di protocollo nel livello 2:

* Ipv4: 32 bit (232)
* Ipv6: 128 bit (2128) 🡺 utilizzato nell’IoT.
* Nat (permette di salvare gli indirizzi IP a IPv4)
* ICMP: controlla se c’è una corretta comunicazione, ecc.
* ROUTING (es:BGP): serve per instradare

Esempi di protocollo nel livello 1:

* ARP: usa gli indirizzi logici in fisici (MAC)