Vulnerabilità intrinseche IP/TCP

Prima di capire le vulnerabilità, è necessaria una revisione dei protocolli di rete.

0.1 Architettura di IP

Intenet non è altro che una inter-connessione di reti che possono avere differenti tecnologie.
La comunicazione fra i device è possibile con IP, che è implementato sui livelli 1 e 2 (che sono indipendenti).
Ogni dispositivo è identificato dall'indirizzo univoco a 32/128 bit (che seguono la nomenclatura CIDR), le diverse sotto-reti comunicano tramite i router ovvero un device IP che ha almeno 3 livelli dello stack Internet.
Ha diverse interfacce che possono essere in diverse tecnologie: ADSL, Ethernet, Fibra etc...
Le operazioni compiute dai router prevedono l'IP forwarding: longest prefix mathcing, manda un pacchetto verso la prossima interfaccia del percorso di rete.
È diretta se la destinazione è nella subnet, mentre indiretta se serve trovare il next hop, quindi andare fuori dalla propria sotto-rete.
Il next hop è un dispositivo nella propria sotto-rete che sa come arrivare alla destinazione.
Il goal di IP è quello di mandare il pacchetto a destinazione, l'Internet è diviso in Autonomous Systems all'interno dei quali si può configurare il routing come si vuole.
Ma poi serve scambiare informazioni fra AS, si parla quindi di protocolli intra-AS come OSPF, RIP e protocolli inter-AS, come BGP.
Oltre all'indirizzo IP, si usa anche la network mask, dove il bit i-esimo è impostato a:

- 0 se è nella parte host dell'IP;
- 1 se è nel prefisso di rete

dove due indirizzi della stessa sottorete hanno i primi X bit uguali ed i restanti 32-X (nel caso IPv4) diversi.
C'è l'header per ogni pacchetto che ha i diversi campi (TTL, src, dest, protocol, checksum etc...), diverso fra IPv4 ed IPv6, differenze fra i due:

- header più piccolo in IPv6
- no fragmentation in IPv6

0.1.1 Routing table

Struttura dati che associa una destinazione ad un next hop.
Per il routing si usa il longest prefix matching, l'IP più "specifico" è quello scelto.
La differenza principale fra host e router è che se un host riceve un pacchetto da una src non fra quelle locali, lo scarta; un router lo forwarda.

Il forwarding standard basico è fatto in base all'indirizzo di destinazione: dopo l'estrazione del pacchetto IP, se l'IP dest è locale, lo manda al livello superiore, altrimenti lo guarda nella tabella di routing.
Se non si trova un matching, il pacchetto è scartato, altrimenti se c'è un next hop, bisogna scoprire chi è, altrimenti basta scoprire il MAC associato all'IP.
La maggior parte dei pacchetti che vengono mandati vanno fuori dall'AS con cui si firma il contratto: possono esserci diversi AS di transito prima di arrivare al data center del sito a cui si cerca di connettersi.

0.2 Vulnerabilità di TCP/IP

Per design, IP e TCP non si preoccupano della sicurezza in quanto sono stati progettati quando la sicurezza non era un problema.
C'è un certo numero di vulnerabilità intrinseche nei protocolli, ed è ancora così in quanto sono stati progettati senza pensare all'aspetto di sicurezza:

- Identification: i device della rete possono essere falsificati, è possibile spoofare l'IP address per dire di essere, ad esempio, un certo server web.
  Questo vale sia per i pacchetti generati dall'utente che per quelli forwardati: se viene inviato un pacchetto da un legittimo server DNS, stiamo impersonificando il server stesso.
  Non ci sono meccanismi in IP per verificare l'autenticità di chi manda il pacchetto;

- Repudiation: come è possibile verificare che l'origine del pacchetto è effettivamente la sorgente?

- Confidentiality: non vogliamo che le informazioni scambiate sulla rete siano viste da 3e parti.
  Vorrei che solo il server veda i dati in uno scambio client-server, ma intercettare il pacchetto in maniera "malevola" è fattibile ed anche decodificarlo perché l'header e l'IP sono in chiaro;
  Il problema non è nella rete interna, ma passa per diversi AS (noi ci fidiamo solo del nostro): anche se il path fra sorgente e destinazione è fidato, è possibile fare hijaking su BGP, fra gli attacchi più pericolosi e in voga;

- Integrity: voglio essere sicuro che il pacchetto non sia modificato durante il percorso fra src e dest.
  C'è il checksum, ma è calcolato su dati in chiaro, quindi non va bene come "codice" per poter assicurare l'integrità;

- Packet replication: non c'è sequence number nell'IP header nella sessione, non ci sono meccanismi per proteggersi da questo threat

- Dynamic mapping: le cose si complicano perché i protocolli Internet usano diversi meccanismi per implementare il mapping: ad esempio il DSN, ARP, 802.3 bridging (lo switch impara in automatico quale MAC è dietro quale porta) e questo è dinamico in quanto può cambiare nel tempo.
  Questo problema è presente in ogni layer ed anche questi meccanismi non sono stati pensati per essere sicuri;

0.2.1 DNS spoofing

Possiamo spoofare una richiesta DNS, quindi fare un hijack della sessione.
Attacco triviale (e vecchio):

- sfruttiamo il mapping MAC-IP per diventare MiTM
- sfruttiamo il mapping fra il nome di dominio e l'IP per impersonare il server web
- mappiamo l'IP del sito al nostro

Lo scenario è che l'attaccante sia nella stessa sotto-rete.
L'idea è quella di mandare dei messaggi ARP spoofati per far credere alla vittima di essere il default gateway e di fra credere al default gateway di essere la vittima.
Per impersonare il sito target è possibile usare il comando `wget`, ed usare Apache2 per impersonare il server web.
L'attacco sfrutta 3 vulnerabilità:

- ARP spoofing
- DNS redirection
- Impersonification

Per emulare un server DNS è possibile configurare bind ma c'è un piccolo servizio DNS da configurare e che permette di forwardare all'attacker tutto ciò che sono è noto.

0.2.2 Security requirements per le vulnerabilità

Vediamo cosa è possibile risolvere questi problemi:

- per la confidenzialità si usano algoritmi a chiave simmetrica per garantirla.
  In un algoritmo a chiave simmetrica, entrambe le end della comunicazione condividono la stessa chiave, c'è anche la differenza fra:
  - stream ciphers
  - block ciphers, che usano concatenazione per evitare ECB, in quanto se la stessa chiave viene riusata per molto è possibile scoprire il contenuto in chiaro dei blocchi che hanno lo stesso contenuto cifrato
- per l'integrità, vogliamo una prova che nessuno modifichi il contenuto del messaggio, quindi si produce un MAC basato su hash functions: l'hash è calcolato non solo sui dati ma anche sulla chiave.
  L'Authenticated Encryption permette, con "la stessa chiave" di fornire sia confidenzialità che integrità.
  Si può ottenere un algoritmo del genere anche con le tecniche base di symmetric encyption;
- Per l'autenticità è possibile usare pub key cryptography per realizzare dei meccanismi di firma, come RSA e Diffie-Hellman.
  Serve comunque avere una certificazione che un utente è il vero proprietario di una chiave, quindi che sia crittograficamente legato alla chiave: Public Key Infrastructure.