Università di Verona, Facoltà di Scienze MM.FF.NN.

Insegnamento di Reti di Calcolatori

Applicazioni TCP/IP

Davide Quaglia

Scopo di questa esercitazione è:
1) illustrare il funzionamento del Domain Name Service
2) illustrare il funzionamento dei protocolli per il World Wide Web e la posta elettronica

1 Domain Name Service (DNS)

1.1 Introduzione

Domain Name Service (DNS) è un servizio utilizzato per la risoluzione di nomi di host in indirizzi IP. Il servizio permette così di utilizzare i nomi e le parole di uso comune per fare riferimento ad host su cui risiedono delle applicazioni, ad esempio un sito Internet. In pratica un nome di un host viene associato al suo indirizzo IP. Ad esempio il sito di Novell `www.novell.com` in realtà è solo un modo facile per identificare l'host residente all'indirizzo 130.57.5.25. La possibilità di attribuire nomi simbolici agli indirizzi IP degli host è essenziale per l'usabilità di Internet, perché gli esseri umani trovano più facile ricordare nomi testuali, mentre gli host ed i router sono raggiungibili solo utilizzando gli indirizzi IP numerici.
Il servizio di associazione nomi/IP è realizzato tramite un database distribuito, costituito dai server DNS.
Un nome è costituito da una serie di stringhe separate da punti, ad esempio `it.wikipedia.org`. La stringa più a destra è detta *dominio di primo livello* (o TLD, Top Level Domain), per esempio `.org` o `.it`. Un dominio di secondo livello consiste in due parti, per esempio `wikipedia.org`, e così via. Ogni ulteriore stringa a sinistra specifica un'ulteriore suddivisione. Quando un dominio di secondo livello viene registrato all'assegnatario, questo è autorizzato a usare i nomi di dominio relativi ai successivi livelli come `it.wikipedia.org` (dominio di terzo livello) e altri come `some.other.stuff.wikipedia.org` (dominio di quinto livello) e così via; questo meccanismo garantisce l'unicità dei nomi.
Ogni volta che si fa accesso ad un host su Internet specificandone il nome (ad es. un sito web o un server di posta) il nostro PC chiede ad un server DNS l'indirizzo IP corrispondente che sarà poi usato per spedirgli i pacchetti.

1.2 Configurazione Linux

La lista con il o i server DNS da contattare deve essere impostata dall'amministratore di rete per ogni host. Su Linux tale lista è contenuta nel file `/etc/resolv.conf` che può essere visualizzato nel modo seguente:

```
$ cat /etc/resolv.conf
```

In laboratorio tale file dovrebbe essere impostato come segue:

```
$ sudo gedit /etc/resolv.conf

search sci.univr.it
nameserver 157.27.10.10
```

La direttiva *search* indica quale dominio usare quando nelle richieste viene indicato solo il nome dell'host (dominio sottointeso). La direttiva *nameserver* invece indica l'indirizzo IP del server DNS.

Un programma per shell che risolve i nomi in indirizzi IP è `dig` che invoca un server DNS e stampa a video la risposta.

Esempio:

```
$ dig www.univr.it
```

mostra l'IP del server web di Ateneo cioè 157.27.6.235

```
;; ANSWER SECTION:
www.univr.it.           2219    IN      CNAME   websrv.univr.it.
websrv.univr.it.        1913    IN      A       157.27.6.235
```

2 Analisi mediante `telnet` di alcuni protocolli di livello applicazione

I protocolli di livello applicazione dell'architettura TCP/IP possono imbustare i dati in pacchetti TCP oppure UDP. Le applicazioni che richiedono una trasmissione affidabile (ad es. WEB e posta elettronica) utilizzano TCP che permette di vedere il collegamento tra due host come un "tubo" virtuale in cui i byte sono trasferiti in maniera ordinata ed affidabile. In tale caso i protocolli di livello applicazione dell'architettura TCP/IP sono realizzati mediante scambio di messaggi testuali in modalità client/server; un programma client (ad esempio il browser Web oppure il programma per leggere la posta) apre una connessione TCP verso il server (Web o di posta) e invia richieste in formato testuale ricevendo le corrispondenti risposte.

Scopo dell'esercizio è sperimentare l'uso del tool `telnet` per interagire con programmi server che utilizzino connessioni di tipo TCP. Il tool `telnet` è un semplice client che apre una connessione TCP con il server e permette all'utente che voglia fare il debug di colloquiare con il server tramite un'interfaccia a caratteri.
Per aprire una connessione con un server in attesa sulla porta N ad un host di nome hostname è sufficiente dare il comando:

```
$ telnet hostname N
```

A questo punto, ammesso che il tentativo di connessione sia andato a buon fine, si interagisce direttamente con il server tramite il videoterminale.

2.1 Il protocollo Hyper Text Transfer Protocol (HTTP)

HTTP è l'acronimo di Hyper Text Transfer Protocol (protocollo di trasferimento di un ipertesto). Usato come principale sistema per la trasmissione di informazioni sul web. Le specifiche del protocollo sono attualmente in carica al W3C (World Wide Web Consortium). La prima versione, la 0.9, dell'HTTP risale alla fine degli anni '80 del XX secolo e costituiva, insieme con l'HTML e gli URL, il nucleo base della "World-Wide Web WWW global information initiative" portata avanti da Tim Berners-Lee al CERN di Ginevra per la condivisione delle informazioni tra la comunità dei fisici delle alte energie. La prima versione effettivamente disponibile del protocollo, la HTTP/1.0, venne implementata dallo stesso Berners-Lee nel 1991 e proposta come RFC 1945 all'ente normatore IETF nel 1996. Con la diffusione di NCSA Mosaic, un browser grafico di facile uso, il WWW conobbe un successo crescente e divennero evidenti alcuni limiti della versione 1.0 del protocollo, in particolare:

- l'impossibilità di ospitare più siti web sullo stesso indirizzo IP (virtual hosting)
- il mancato riuso della connessione TCP già aperta
- l'insufficienza dei meccanismi di sicurezza

Il protocollo venne quindi esteso nella versione HTTP/1.1, presentato come RFC 2068 nel 1997 e successivamente aggiornato nel 1999 come descritto dal RFC 2616

L'HTTP funziona su un meccanismo richiesta/risposta (client/server): il client esegue una richiesta ed il server restituisce la risposta. Nell'uso comune il client corrisponde al browser ed il server al sito web. Vi sono quindi due tipi di messaggi HTTP: messaggi richiesta e messaggi risposta.

Il messaggio richiesta è composto di tre parti:

- Riga di richiesta (request line)
- Sezione Header (informazioni aggiuntive)
- Body (corpo del messaggio)

La riga di richiesta è composta dal
- metodo,
- URI
- versione del protocollo.

Il metodo di richiesta, per la versione 1.1, può essere uno dei seguenti:

- GET
- POST
- HEAD
- PUT
- DELETE
- TRACE
- OPTIONS

l'URI sta per *Uniform Resource Identifier* ed indica l'oggetto della richiesta (ad esempio la pagina web che si intende ottenere).

I metodi HTTP più comuni sono GET, HEAD e POST. Il metodo GET è usato per ottenere il contenuto della risorsa indicata come URI (come può essere il contenuto di una pagina HTML). HEAD è analogo a GET, ma restituisce solo i campi dell'header, ad esempio per verificare la data di modifica del file. Una richiesta con metodo HEAD non prevede l'uso del body.

Il metodo POST è usato di norma per inviare informazioni al server (ad esempio i dati di un form). In questo caso l'URI indica che cosa si sta inviando e il body ne indica il contenuto.

Gli header di richiesta più comuni sono:

Host: Nome del server a cui si riferisce l'URI. È obbligatorio nelle richieste conformi HTTP/1.1 perché permette l'uso dei virtual host basati sui nomi.

User-Agent: Identificazione del tipo di client: tipo browser, produttore, versione...

Il messaggio di risposta è composto dalle seguenti tre parti:

- Riga di stato (status-line)
- Sezione header
- Body (contenuto della risposta)

La riga di stato riporta un codice a tre cifre catalogato nel seguente modo:

- 1xx: Informational (messaggi informativi)
- 2xx: Success (la richiesta è stata soddisfatta)
- 3xx: Redirection (non c'è risposta immediata, ma la richiesta è sensata e viene detto come

ottenere la risposta)
- 4xx: Client error (la richiesta non può essere soddisfatta perché sbagliata)
- 5xx: Server error (la richiesta non può essere soddisfatta per un problema interno del server)

Nel caso più comune il server risponde con un codice 200 (OK) e fornisce il contenuto nella sezione body. Altri casi comuni sono:

- *301 Moved Permanently*. La risorsa che abbiamo richiesto non è raggiungibile perché è stata spostata in modo permanente.
- *302 Found*. La risorsa è raggiungibile con un altro URI indicato nel header Location. Di norma i browser eseguono la richiesta all'URI indicato in modo automatico senza interazione dell'utente.
- *400 Bad Request*. La risorsa richiesta non è comprensibile al server.
- *404 Not Found*. La risorsa richiesta non è stata trovata e non se ne conosce l'ubicazione. Di solito avviene quando l'URI è stato indicato in modo incorretto, oppure è stato rimosso il contenuto dal server.
- *500 Internal Server Error*. Il server non è in grado di rispondere alla richiesta per un suo problema interno.
- *505 HTTP Version Not Supported*. La versione di http non è supportata.

Gli header della risposta più comuni sono:

- *Server*. Indica il tipo e la versione del server. Può essere visto come l'equivalente dell'header di richiesta User-Agent
- *Content-Type*. Indica il tipo di contenuto restituito. La codifica di tali tipi (detti Media type) è registrata presso lo IANA (Internet Assigned Number Authority); essi sono detti tipi MIME (Multimedia Internet Message Extensions), la cui codifica è descritta nel documento RFC 1521. Alcuni tipi usuali di tipi MIME incontrati in una risposta HTML sono:
    - text/html. Documento HTML
    - text/plain. Documento di testo non formattato
    - image/jpeg. Immagine di formato JPEG

Quando un client (ad es. un web browser) richiede una pagina ad un server web, apre la connessione TCP sull'IP del server (eventualmente ottenuto tramite il nome) sulla porta 80. A questo punto scrive nella connessione TCP (esattamente come fosse un file) la riga di richiesta e la sezione header seguita da una riga vuota (carattere di *new_line* ottenibile da tastiera premendo INVIO).

Ad esempio, se si vuole emulare il comportamento di un browser nell'interrogare il server Web del PC del docente occorre scrivere:

```
$ telnet 157.27.241.111 80
```

```
GET / HTTP/1.0[INVIO]
[INVIO]
```

A questo punto il server risponde scrivendo nella connessione TCP in caratteri testuali l'header della risposta e la pagina web richiesta (in formato Hyper Text Markup Language – HTML).

Il precedente esempio funziona per server locali o nelle reti in cui non è obbligatorio l'uso di un proxy web per navigare in Internet. Siccome nei laboratori di Facoltà tutte le richieste web devono passare attraverso il proxy `proxy.sci.univr.it` allora per fare accesso ad un server Web esterno il precedente esempio deve essere trasformato.

Prima occorre collegarsi al proxy della Facoltà

```
$ telnet proxy.sci.univr.it 8080
```

```
CONNECT it.wikipedia.org:80 HTTP/1.1[INVIO]
[INVIO]
```

a questo punto il proxy effettua la connessione per noi e ci manda il messaggio

```
HTTP/1.0 200 Connection established
```

a questo punto è possibile richiedere la pagina nel modo consueto (ma tramite il proxy)

```
GET /wiki/Pagina_principale HTTP/1.1[INVIO]
User-Agent: Mozilla/5.0 (KHTML, like Gecko)[INVIO]
Accept: text/html[INVIO]
Accept-Charset: iso-8859-1, utf-8;q=0.5, *;q=0.5[INVIO]
Accept-Language: en[INVIO]
Host: it.wikipedia.org[INVIO]
[INVIO]
```

2.2 Il Simple Mail Transfer Protocol (SMTP)

Il Simple Mail Transfer Protocol (SMTP) è il protocollo standard per la trasmissione di e-mail via Internet. È un protocollo relativamente semplice, testuale, nel quale vengono specificati uno o più destinatari di un messaggio, verificata la loro esistenza, il messaggio viene trasferito. È abbastanza facile verificare come funziona un server SMTP mediante un client telnet che simula il funzionamento di un client di posta come Mozilla-Thunderbird. L'SMTP usa il protocollo di trasmissione TCP e, per accedervi, la porta 25. Per associare il server SMTP a un dato nome di dominio (DNS) si usa un record denominato MX (Mail eXchange).

L'SMTP iniziò a diffondersi nei primi anni '80. A quel tempo era un'alternativa all'UUCP, che era più adatto a gestire il trasferimento di e-mail fra computer la cui connessione era intermittente. L'SMTP, d'altra parte, funziona meglio se i computer sono sempre collegati alla rete.

Sendmail fu uno dei primi (se non proprio il primo) programma ad implementare il protocollo SMTP. Fino al 2001 sono stati scritti almeno 50 programmi che implementano il protocollo SMTP come client (mittente dei messaggi) o server (destinatario del messaggio). Altri server molto diffusi sono *Exim* di Philip Hazel, *Postfix* di Wietse Venema, *qmail* di D. J. Bernstein, *Courier* di Sam Varshavchik e *Microsoft Exchange Server*.

Poiché SMTP è un protocollo testuale basato sulla codifica ASCII, non è permesso trasmettere direttamente testo composto con un diverso set di caratteri e tantomeno file binari. Lo standard MIME permette di estendere il formato dei messaggi mantenendo la compatibilità col software esistente. Per esempio, al giorno d'oggi molti server SMTP supportano l'estensione 8BITMIME, la quale permette un trasferimento di un testo che contiene caratteri accentati (non-ASCII) senza bisogno di trascodificarlo. Altri limiti di SMTP, quale la lunghezza massima di una riga, impediscono la spedizione di file binari senza trascodifica. (Nota che per i file binari inviati con HTTP si utilizza il formato MIME senza bisogno di una trascodifica.)

L'SMTP è un protocollo che permette soltanto di inviare messaggi di posta, ma non di richiederli ad un server: per fare questo il client di posta deve usare altri protocolli, quali il POP3 (Post Office Protocol) e l'IMAP (Internet Message Access Protocol).

Quella che segue è una transazione SMTP valida. Le righe inviate dal client sono precedute da "C:", mentre quelle inviate dal server da "S:". Su molti computer si può stabilire una connessione mediante il comando telnet:

```
telnet 157.27.241.111 25
```

Questo comando apre una connessione al PC del docente sulla porta SMTP.

```
S: 220 localhost.localdomain ESMTP Postfix (Debian/GNU)
C: HELO localhost.localdomain
S: 250 localhost.localdomain
C: MAIL FROM: <mionome@miodominio.it>
S: 250 Ok
C: RCPT TO: <reti@localhost.localdomain>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: Subject: messaggio di prova
C: From: mionome@miodominio.it
C: To: reti@localhost.localdomain
C:
C: Questa e' una prova
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
```

Notare che posso usare come mittente l'indirizzo che voglio e questa è una grave vulnerabilità del protocollo SMTP. Per ovviare si usa Secure SMTP che oltre a cifrare la connessione permette anche di autenticare il client.

2.3 Il Post Office Protocol (POP)

Il Post Office Protocol (detto anche POP) è un protocollo che ha il compito di permettere, mediante autenticazione, l'accesso ad un account di posta elettronica presente su di un host per scaricare le e-mail del relativo account. Il demone pop (nella versione 3) rimane in attesa sulla porta 110 dell'host (di default, ma può anche essere diversa) per una connessione TCP da parte di un client. I messaggi di posta elettronica, per essere letti, devono essere scaricati sul computer (questa è una notevole differenza rispetto all'IMAP), anche se è possibile lasciarne una copia sull'host. Il protocollo POP3 non prevede alcun tipo di cifratura, quindi le password utilizzate per l'autenticazione fra server e client passano in chiaro. Per risolvere questo possibile problema è stato sviluppata l'estensione APOP che utilizza MD5.

Esercizio: si simuli il comportamento di un client di posta utilizzando il programma telnet e aprendo una connessione verso 157.27.241.111 sulla porta 110. Non appena il server risponde dare i seguenti comandi attendendo per ognuno la risposta del server.

```
USER reti
PASS segreta
LIST
RETR 1
QUIT
```

2.3.1 Analisi di una connessione al server di posta

Mentre `wireshark` stava catturando è stato svolto l'esercizio della Sezione 2.3 (accesso al server di posta POP). Come si può notare, il file contiene un sacco di pacchetti che spesso non fanno riferimento all'host da cui si è effettuata la lettura della posta.

Un modo per diminuire il volume di pacchetti visualizzati è impostare un filtro di lettura sul MAC del mio host nel modo seguente

```
eth.addr == mioMAC
```

in questo modo è possibile notare uno scambio preliminare di messaggi ARP.

Se invece voglio isolare solo i pacchetti scambiati con il server di posta (`157.27.241.111`) allora occorre usare un filtro con l'IP di tale host (se si conosce solo il nome si può ricavare l'IP con il comando `dig`)

```
ip.addr == 157.27.241.111
```

è possibile esaminare anche il contenuto dei pacchetti (si noti la password dell'account di posta ben visibile a occhi indiscreti).