Sulla risoluzione degli indirizzi IP Parte IV – Domain Name System

Stefano Bonacina (*), Francesca Giuratrabocchetti (**), Davide Stefanoni (***)

(*) *Dottorato di Ricerca in Bioingegneria – XVI ciclo – Politecnico di Milano*
(**) *CILEA, Segrate*
(***) *già Politecnico di Milano e Medical Informatics Training Program of the National Library of Medicine (NLM) at the National Institute of Health (NIH), Bethesda, MD, USA*

Abstract
Con questo articolo si conclude la pubblicazione della monografia sul riconoscimento degli indirizzi IP. Il lavoro è originato dalla attività di dottorato di Stefano Bonacina, dottorando in Bioingegneria del Politecnico di Milano. Il documento originale è stato curato e rivisto nel suo complesso dagli specialisti di reti del CILEA, anche per tenere conto delle attuali evoluzioni delle soluzioni software.
Questa quarta e ultima parte, descrive un'applicazione reale dei concetti illustrati negli articoli precedenti: l'analisi degli accessi via rete ad una banca dati di immagini anatomiche. Il Visible Human Dataset, oggetto di tale analisi, è il più grande atlante di bioimmagini digitali del corpo umano, ospitato presso NLM (*National Library of Medicine, Bethesda,* Maryland, USA) [2], titolare dei diritti sulle immagini e presso il CILEA, uno dei quattro Mirror Site mondiali, indicato qui come MMS (*Milano Mirror Site*, CILEA, Milano, Italia). L'articolo descrive nel dettaglio un'analisi sugli accessi via rete all'atlante, utilizzando i concetti descritti negli articoli precedenti e alcuni pacchetti software liberamente disponibili.
Le prime tre puntate sono state pubblicate sui numeri 85,86 e 87 del Bollettino del CILEA.

Keywords: Telematica, TCP/IP, DNS, Reti.

Il caso dei log file del Visibile Human Dataset

Il Visibile Human Dataset (VHD) è il più grande atlante di bioimmagini digitali del corpo umano. È composto da circa quindici mila immagini provenienti da sezioni orizzontali di un cadavere maschile e di uno femminile, opportunamente selezionati [1]. Oltre alle immagini fotografiche ci sono quelle ottenute dalla risonanza magnetica (MR) e dalla tomografia assiale computerizzata (TAC) [2-3].
Il più semplice utilizzo del VHD consiste nella visualizzazione delle sezioni trasversali, sagittali, ed oblique del cadavere. Altri impieghi riguardano l'estrazione di organi e la creazione di dettagliati modelli tridimensionali dell'anatomia umana. Tutto questo costituisce una base per la costruzione di simulatori interattivi, sempre più richiesti non solo negli insegnamenti di Medicina.

La caratteristica principale dei dati contenuti in un database di bioimmagini è l'organizzazione dei file in gruppi. Tramite l'analisi degli accessi ai file che compongono la banca di bioimmagini è possibile monitorare e descrivere il comportamento degli utenti e precisamente:
1- verificare che degli utenti siano interessati al servizio offerto (quanti si collegano);
2- verificare che gli utenti siano in grado di accedere alle sezioni desiderate del sito;
3- studiare il comportamento dei visitatori (chi sono, da dove vengono, a quali informazioni sono interessati).
Tramite la risoluzione degli indirizzi IP in nomi a dominio si può stabilire la provenienza geografica del visitatore di un sito ed ottenere in questo modo informazioni ulteriori.

I file di log contengono le informazioni circa l'elenco delle azioni compiute dal generico visitatore sul Server Web in cui alloggia un determinato sito.

Dall'analisi del contenuto dei log file, disponibili per le statistiche sugli accessi al VHD Visible Human Dataset si nota che nei rispettivi server della NLM (National Library of Medicine, Bethesda, Maryland, USA) [2] e del MMS (Milano Mirror Site, CILEA, Milano, Italia) [3] è stata attivata la risoluzione degli indirizzi IP. La maggior parte degli utenti compare con il proprio nome a dominio, ma ciò non avviene necessariamente per tutti. Gli indirizzi che non compaiono con il nome a dominio, ma solamente con il loro indirizzo IP, sono definiti "unresolved", cioè non risolti. Nei file di log analizzati le quantità e le percentuali corrispondenti degli indirizzi di quel tipo sono:

- 53 IP non risolti su 318 per il MMS nel periodo Gennaio 1998 – Maggio 2001 pari al 16.66%;
- 658 IP non risolti su 2521 per la NLM nel periodo Gennaio 1998 – Febbraio 2001 pari al 26.1%.

Per superare il problema degli indirizzi non risolti si ricorre in genere all'esecuzione delle query inverse. Molti tra i software per l'analisi dei file di log, detti *Log Analyzer*, prevedono, tra le loro opzioni, questa possibilità. La risoluzione degli indirizzi IP è una procedura alquanto complessa e laboriosa che richiede solitamente una notevole quantità di tempo. Ovviamente qualsiasi programma per la risoluzione degli indirizzi IP fornisce gli stessi risultati, a patto di aver impostato un *time-out* sufficientemente ampio. La corrispondenza tra il campo PTR e il suo indirizzo IP è infatti sempre univoca. Le differenze si riscontrano invece nella velocità con la quale le risoluzioni vengono portate a termine; ad esempio è fondamentale che il programma in questione implementi delle query asincrone che possano essere lanciate contemporaneamente. Sfruttando il "DNS lookup", comando del sistema operativo Unix, la percentuale di indirizzi unresolved presente nei log file del VHD diminuisce sensibilmente:

- 30 IP non risolti su 313 per il MMS nel periodo Gennaio 1998 – Maggio 2001 pari al 9.58%;
- 485 IP non risolti su 2521 per la NLM nel periodo Gennaio 1998 – Febbraio 2001 pari al 19,23%.

Si osservi che nel caso del MMS compaiono 313 host name invece dei 318 del caso precedente: significa che alcuni degli indirizzi risolti corrispondevano a nomi già presenti in altre parti del file di log. Nonostante i risultati ottenuti, la percentuale di indirizzi non risolti rimane ancora troppo alta. Serve trovare strade alternative alle query di tipo PTR. Nei casi in cui il Resource Record (RR) di tipo PTR non è definito per un certo indirizzo IP non è possibile risalire in alcun modo al nome della macchina corrispondente, occorre quindi accontentarsi di informazioni più generiche. Prendere in considerazione i RR di tipo NS o SOA consente di venire a conoscenza del nome del Domain Name Server autoritario della zona a cui un certo indirizzo IP appartiene. In questo modo si ottiene quanto meno un'idea del tipo dominio dell'indirizzo cercato. JDResolve [4] è uno script in linguaggio Perl [5] capace di eseguire questo tipo di interrogazioni; tramite questa soluzione il numero degli indirizzi non risolti diventa trascurabile:

- 0 IP non risolti su 313 per il MMS nel periodo Gennaio 1998 – Maggio 2001 pari allo 0%;
- 83 IP non risolti su 2521 per la NLM nel periodo Gennaio 1998 – Febbraio 2001 pari al 3.29%.

Gli 83 indirizzi non risolti, caso della NLM, sono in realtà risolti, ma in modo non soddisfacente. Succede infatti che alcuni amministratori dei server DNS impostino erroneamente i RR di tipo PTR. Molto spesso non si tratta di un errore, ma di una deliberata volontà di non permettere la risoluzione inversa. In questi casi il DNS lookup giunge ad un risultato che è evidentemente privo di senso, ad esempio "baw-baw". A questo tipo di *unresolved* non esiste rimedio se la risoluzione errata viene effettuata dal server web. Nel log file verrà infatti registrato il nome falso perdendo l'indicazione del suo indirizzo IP e quindi ogni possibilità di poter risalire all'utente.

Lo script JDResolve

JDResolve [4] è uno script in linguaggio Perl [5] che consente di tradurre gli indirizzi IP numerici in Hostname (in nomi di Host). Jdresolve è stato scritto da John D. Rowell e rilasciato con licenza GPL (GNU GENERAL PUBLIC LICENSE). E' liberamente consentito l'uso, la copia, la modifica, la distribuzione del software stesso, ma la licenza impone il cosiddetto copyleft: ogni copia modificata può essere distribuita solo con licenza GPL [6].
In JDResolve è supportato qualsiasi formato di file di log, includendo quelli nei quali la linea non inizia con un indirizzo IP. Una delle caratteristiche più solide del programma è il supporto della ricorsione (iterazioni successive), che può ridurre drasticamente il numero degli host non

risolti simulando il nome di un host (hostname) basato sulla rete alla quale l'indirizzo IP appartiene. Le interrogazioni (queries) DNS sono spedite in parallelo, il che significa che si può ridurre il tempo di esecuzione (run time) incrementando il numero di socket simultanei utilizzati (disponendo di una macchina abbastanza veloce e di ampiezza di banda). Mediante l'uso del supporto database, usando i dati immagazzinati nel corso delle esecuzioni precedenti, le prestazioni possono essere incrementate ulteriormente.

Moduli impiegati

Lo script si basa sull'uso del modulo Net::DNS di CPAN (Comprehensive Perl Archive Network), dispone di una subroutine per query DNS in background. In appoggio al modulo appena citato il package IO::Select, per il controllo dei socket, ha permesso all'autore di ottenere un completo approccio non bloccante alle query multiple concorrenti.

Occorre notare che Net::DNS a differenza di IO::Select non fa parte delle librerie di base di Perl e quindi deve essere installato separatamente; inoltre il suo funzionamento è garantito solo su sistemi Unix, su Windows praticamente non funziona.

JDResolve fornisce un semplice supporto a database attraverso i file db (dbm, gdbm, sdbm, ecc.). Per usare il supporto a database, occorre fornire un nome di database (es. 'host.db') usando l'opzione --database. Se non esiste già, sarà creato un nuovo database con quel nome. Tutti gli host e le classi risolte durante l'esecuzione di JDResolve saranno depositati nel database. L'uso del database permetterà di risolvere gli indirizzi nel caso le query DNS falliscano per qualche motivo. In questo modo migliorano le prestazioni dal momento che è possibile diminuire il valore di timeout senza sacrificare la percentuale dei risolti.

L'algoritmo

L'algoritmo su cui si basa il programma, è mostrato nello schema a blocchi di figura 1 ed è composto dai seguenti passi:

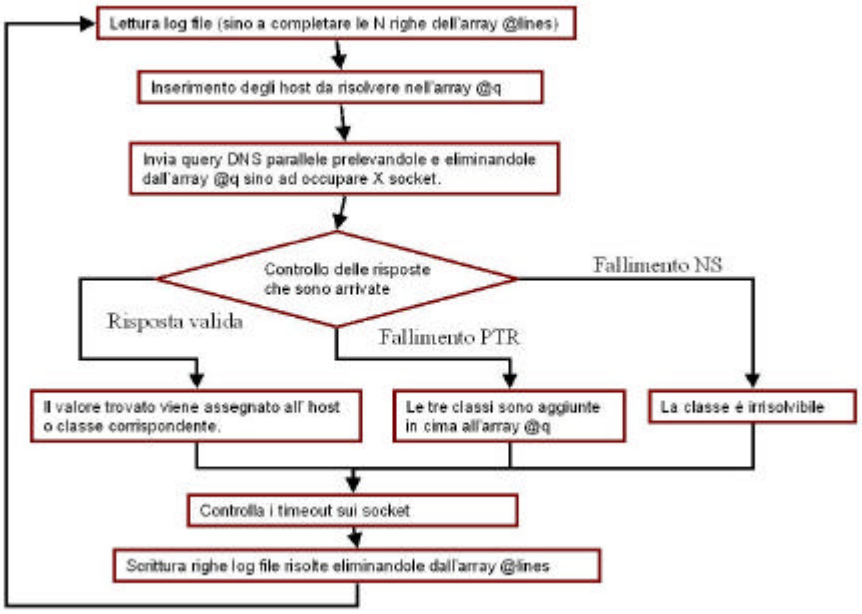

Figura 1 – Il diagramma di flusso dell'algoritmo implementato in JDResolve

Step 1: getlines
Legge le righe del file di log inserendole nell'array **@lines**. Il numero di righe lette dipende dal numero di posti disponibili nell'array. **@lines** ha dimensione N, dove N è un parametro che può essere definito dall'utente.
Quando nella riga viene individuato un indirizzo IP non risolto questo viene inserito in coda all'array **@q**. **@q** contiene l'elenco degli host e delle classi da risolvere in ordine di priorità.

Step 2: makequeries
Prende gli indirizzi IP e le classi dalla cima dell'array **@q** e li invia in query DNS. Ogni query viene eseguita su un socket diverso sino ad occupare tutti quelli disponibili. Il numero di query che vengono eseguite contemporaneamente dipende dal numero di socket che è stato impostato dall'utente in funzione della larghezza di banda disponibile.

Step 3: checkresponse
Controlla i socket per vedere quali risposte alle query DNS sono arrivate. Per ogni risposta raccolta viene liberato il socket corrispondente.

- Se la risposta ricevuta è valida viene registrato il valore del nome dell'host o della classe trovata. Il nome della classe viene desunto dal nome di dominio del server DNS, ritornato dal RR di tipo NS, eliminando il nome proprio della macchina (es.: nomeDNS.nomerete.es -> nomerete.es). Se la risposta NS contiene più di un server DNS, viene considerato l'ultimo della lista.
- Se la risposta ricevuta è nulla e il tipo di risposta è PTR allora vengono aggiunte all'array delle query **@q** le tre classi A, B, C corrispondenti all'indirizzo non risolto. Ad esempio se l'indirizzo IP è xxx.yyy.zzz.kkk verranno inserite le tre classi xxx, xxx.yyy, xxx.yyy.zzz. L'aggiunta avviene in cima alla coda in modo tale che abbia priorità più alta la classe A, poi B ed infine C.
- Se la risposta ricevuta è nulla e il tipo di risposta è NS la classe in questione è definitivamente registrata come irrisolta.

Step 4: checktimeout
Controlla i tempi di attesa dei socket e nel caso superino un certo limite di tempo che può essere definito dall'utente considera le risposte delle rispettive query come nulle.

Step 5: printresult
Scrive una alla volta sullo STDOUT le righe presenti nell'array **@lines**.

- Se la riga non contiene un indirizzo IP da risolvere viene riscritta tale e quale.
- Se la riga contiene un indirizzo IP che è stato risolto tramite una query PTR questo viene sostituito col nome di dominio trovato.
- Se la riga contiene un indirizzo IP che è stato risolto tramite una query NS viene sostituito col primo nome di classe a partire dalla C. Si cerca cioè se è disponibile una classe risolta di tipo C, nel caso non esista si ricorre alla B ed infine alla A.
- Se la riga contiene un indirizzo IP la cui risoluzione è ancora in corso si sospende la scrittura dell'array **@lines** tornando in ciclo.

Il ciclo ha termine quando la lettura del log file è terminata e l'array **@lines** è vuoto, ovvero tutti gli indirizzi sono stati risolti.
L'algoritmo sopra funziona estremamente bene eccetto per il caso di file di log di dimensioni superiori a 500Mbyte. Gli array contenenti indirizzi IP e la loro classe padre A/B/C diventano in questo caso di dimensioni elevatissime e richiedono grandi risorse di memoria per consentirne una gestione efficiente.

L'uso del campo SOA

In alternativa all'uso di query di tipo NS potrebbe anche essere usato il campo SOA. L'adozione di questa seconda soluzione porta alcuni vantaggi in termini di univocità dei risultati, mentre infatti i RR di tipo NS possono essere più di uno e quindi una query NS restituisce l'elenco dei name server preposti ad una certa zona, il RR di tipo SOA è unico e riporta solamente il nome del name server primario. Per includere in JDResolve questa soluzione occorre modificare le righe di codice che specificano il tipo di query da eseguire. Implementando queste due tipologie di query contemporaneamente, si può utilizzare il SOA nei casi in cui l'NS fallisce e viceversa.

Linea di comando

I parametri necessari alla corretta esecuzione di JDResolve devono essere tutti assegnati tramite la riga di comando. La sintassi è la seguente:

jdresolve [-h] [-v] [-n] [-r] [-a] [-d <level>] [-m <mask>]
[-l <line cache>] [-t <timeout>] [-p] [-s <number of sockets>]

[--database=<db path>] <<LOG FILE>>

Nella Tabella 1 sono indicati i parametri di JDResolve e la loro descrizione.

La seguente riga di comando ha prodotto risultati soddisfacenti nella analisi dei log file del VHD:

jdresolve –r –a --database vhd.db --dbfirst logfilein > logfileout

L'opzione –a è necessaria in quanto le righe dei log file non iniziano con gli indirizzi IP, ma con la data, per cui se questa opzione non fosse utilizzata, gli indirizzi non verrebbero nemmeno trovati. L'uso del database invece rende più veloce il processo di risoluzione.

Parametro	Descrizione
--help o -h	Mostra l'elenco dettagliato delle modalità d'uso
--version o -v	mostra il numero della versione di jdresolve.
--nostats o -n	non mostra le statistiche dopo il processo di risoluzione.
--recursive o -r	ricorre alle classi C,B, e A quando non è disponibile il campo PTR. Di default la ricorsione è disattivata.
--anywhere o -a	risolve tutti gli indirizzi IP presenti nel file, di default il programma risolve invece solamente quegli indirizzi IP che si trovano all'inizio di riga.
--progress o --p	mostra una barra di progressione che indica lo stato delle operazioni di risoluzione.
--debug=<level> o -d <level>	imposta la modalità di debug indicando il livello di messaggistica da mostrare a video durante l'esecuzione delle risoluzioni. (esistono tre livelli di debug).
--timeout=<timeout> o -t <timeout>	tempo di timeout (in secondi), oltre il quale la richiesta viene fatta cadere (per ciascuna risoluzione), il valore di default è 30 secondi.
--sockets=<sockets> o -s <sockets>	massimo numero di socket che possono essere usate contemporaneamente. Il valore di default è 64.
--mask=<mask> o -m <mask>	indica la maschera da usare per gli indirizzi risolti in modo ricorsivo. Valori validi per il parametro <mask> sono %i per l'IP e %c per il proprietario della classe. Es: "somewhere.in.%c" o "%i.in.%c". La maschera di default è "%i.%c".
--linecache=<lines> o -l <lines>	numero di linee che possono essere contemporaneamente caricate in memoria (il valore di default è 10.000).
--unresolved	mostra gli indirizzi non risolti nello STDOUT, senza risolvere nulla.
--database=<db path>	percorso del database che contiene gli indirizzi risolti.
--dbfirst	controlla l'eventuale presenza nel database dell'indirizzo cercato, prima di mandare una query DNS.
--dbonly	risolve gli indirizzi usando solamente le informazioni contenute nel database, senza inviare query DNS.
--dumpdb	mostra sullo STDOUT (standard output) il contenuto del database
--mergedb	unisce il file nel database indicato da –database (il formato del file di input deve essere lo stesso di database destinazione, esempio un ip/classe seguito da nomehost/nomeclasse, separati da spazi).
--expiredb=<hours>	elimina le voci del database più vecchie di <hours> ore.
<LOG FILE>	il nome del log file da esaminare o '-' per lo STDIN

Tabella 1- I parametri di JDResolve e la loro descrizione

Un esempio di esecuzione

Nella figura 2 è riportato una parte di file di log, in formato Common Log Format [7], riguardante gli accessi al sito VHD-MMS, durante il mese di agosto 2001, dell'entità di rete avente indirizzo 212.171.123.92.

Il formato Common Log Format contiene i seguenti attributi:

Remotehost: indirizzo IP della entità remota da cui è pervenuta la richiesta;

rfc931: nome di login remoto dell'utente. (Se non disponibile è solitamente inserito un trattino '-');

authuser: username con il quale l'utente si è autenticato. Questo attributo possiede un valore quando si accede a pagine www protette da password. (Se non disponibile è solitamente inserito un trattino '-');

[date]: Data e ora della richiesta (+ GMT offset).

"Request": richiesta d'accesso secondo uno dei metodi consentiti (tipicamente GET);

Status: codice HTTP di risposta alla richiesta (200 indica successo);**Bytes:** numero di byte eventualmente trasferiti se la richiesta è stata soddisfatta.

```
212.171.123.92 - - [01/Aug/2001:00:17:01 +0200] "GET /index.htm HTTP/1.1" 200 2479
212.171.123.92 - - [01/Aug/2001:00:17:02 +0200] "GET /cilea6.gif HTTP/1.1" 200 1803
212.171.123.92 - - [01/Aug/2001:00:17:02 +0200] "GET /vhd_logo.gif HTTP/1.1" 200 10482
212.171.123.92 - - [01/Aug/2001:00:17:03 +0200] "GET /POLI.jpg HTTP/1.1" 200 14519
212.171.123.92 - - [01/Aug/2001:00:17:04 +0200] "GET /logo.gif HTTP/1.1" 200 1425
212.171.123.92 - - [01/Aug/2001:00:17:04 +0200] "GET /hp1.gif HTTP/1.1" 200 2310
```

Figura 2 - Parte delle operazioni compiute sul sito VHD-MMS da parte dell'indirizzo IP 212.171.123.92.

Nella figura 3 è rappresentato la stessa parte del file di log di figura 2 aggiornato inserendo le informazioni riguardanti la classe di rete di appartenenza e il nome di dominio degli indirizzi IP non risolti. Le informazioni vengono ottenute utilizzando una versione modificata dello script JDResolve: si è fatto in modo di ottenere, dalle interrogazioni ai server DNS, la conoscenza della classe di rete di appartenenza dell'indirizzo in analisi.

Il nuovo file di log viene creato aggiungendo le informazioni riguardanti la **classe di rete di appartenenza** e il **nome di dominio**: per il caso in esempio classe A e nome ns.uu.net (indicati in grassetto nella figura 3). La modifica al codice sorgente del programma non è in disaccordo con la licenza, di tipo GPL, del programma originale. Anche questa versione modificata eredita il fatto di essere tutelata da una licenza GPL.

```
212.171.123.92 [A].ns.uu.net - - [01/Aug/2001:00:17:01 +0200] "GET /index.htm HTTP/1.1" 200 2479
212.171.123.92 [A].ns.uu.net - - [01/Aug/2001:00:17:02 +0200] "GET /cilea6.gif HTTP/1.1" 200 1803
212.171.123.92 [A].ns.uu.net - - [01/Aug/2001:00:17:02 +0200] "GET /vhd_logo.gif HTTP/1.1" 200 10482
212.171.123.92 [A].ns.uu.net - - [01/Aug/2001:00:17:03 +0200] "GET /POLI.jpg HTTP/1.1" 200 14519
212.171.123.92 [A].ns.uu.net - - [01/Aug/2001:00:17:04 +0200] "GET /logo.gif HTTP/1.1" 200 1425
212.171.123.92 [A].ns.uu.net - - [01/Aug/2001:00:17:04 +0200] "GET /hp1.gif HTTP/1.1" 200 2310
```

Figura 3 – L'IP in questione appartiene ad una rete di classe A ed il suo domain name è ns.uu.net

Bibliografia

[1] V. Spitzer, MJ. Ackerman, AL. Scherzinger, D Whitlock, *"The Visible Human male: a technical report"*, J. Amer Med Informatics Assoc. 1996, 3(2) pp. 118-130

[2] National Library of Medicine, *"The Visible Human Project®"*
Disponibile all'indirizzo URL:
http://www.nlm.nih.gov/research/visible/visible_human.html
Data dell'ultimo accesso: 10 Giugno 2003

[3] Visible Human Dataset, *"Milano Mirror Site®"*
Disponibile all'indirizzo URL:
http://vhd-mms.cilea.it/
Data dell'ultimo accesso: 10 Giugno 2003

[4] Jdresolve 0.6.1, *"Jd's homepage – jdresolve"*
Disponibile all'indirizzo URL:
http://jdrowell.com/projects/jdresolve
Data dell'ultimo accesso: 10 Giugno 2003

[5] O'Reilly®, Perl.com, *"The source for perl"*
Disponibile all'indirizzo URL:
http://www.perl.com
Data dell'ultimo accesso: 10 Giugno 2003

[6] GNU General Public License, *"GNU Project"*, Free Software Foundation (FSF)
Disponibile all'indirizzo URL:
http://www.gnu.org/copyleft/gpl.html
Data dell'ultimo accesso: 10 Giugno 2003

[7] Common Log Format, *"Bacus Laboratories"*
Disponibile all'indirizzo URL:
http://www.bacuslabs.com/WsvlCLF.html
Data dell'ultimo accesso: 10 Giugno 2003.