MODULARITÀ APPLICATA ALL'ELABORAZIONE DI PACCHETTI DI RETE: IL LINGUAGGIO NETPDL

M. Baldi, F. Risso
Dipartimento di Automatica e Informatica, Politecnico di Torino
{mario.baldi, fulvio.risso}@polito.it

Questo articolo presenta NetPDL, un linguaggio basato su XML che permette di descrivere il formato delle intestazioni e l'imbustamento dei protocolli di rete e la sua implementazione nella libreria open-source NetBee.

1. Introduzione

Questo articolo presenta il linguaggio NetPDL *(Network Protocol Description Language)*, che affronta il problema di identificare univocamente ogni campo di ogni protocollo di rete. In altre parole, definisce il formato di ogni singolo protocollo e si incarica di dare un nome univoco ad ogni campo (ad es. `ip.src` per quanto riguarda il campo "indirizzo IP sorgente"). Questo linguaggio è basato su XML che può essere una buona scelta per la sua flessibilità, la sua attuale diffusione e per la presenza di numerosi strumenti software (molti open-source) in grado di operare su file XML. Grazie a questi strumenti, un programmatore può concentrarsi maggiormente sulla semantica dei dati in quanto il controllo sintattico che viene fatto automaticamente da queste librerie XML. Inoltre, un linguaggio basato su XML può essere facilmente esteso (nuovi elementi o nuovi attributi di elementi esistenti) pur mantenendo la compatibilità con le precedenti versioni.

2. Network Protocol Description Language

Il *Network Protocol Description Language* (*NetPDL*) [1] è un linguaggio semplice e generico che mira a descrivere il formato di un pacchetto di rete dal punto di vista del formato dei dati (*packet header*) e dell'incapsulamento del protocollo stesso. La semplicità è dovuta soprattuto al fatto che il NetPDL non comprende primitive di descrizione temporale del protocollo (macchina a stati). Inoltre, è facilmente estendibile grazie al fatto di essere basato su XML.

2.1. Panoramica del linguaggio

NetPDL si compone di un certo numero di primitive composte da un elemento XML (es. `<proto>`) e caratterizzate da un insieme di attributi (es. `name="Ethernet"`). La Figura 1 mostra un estratto di una descrizione relativa ad un header Ethernet. Questo protocollo è composto da tre campi di lunghezza fissa (rispettivamente di sei, sei e due byte), ed è caratterizzato da contenere altri protocolli a seguire. Il formato della trama è contenuto nella sezione `<fields>`, mentre l'imbustamento è contenuto negli elementi `<protoref>` inclusi nella sezione `<nextproto>`. La scelta del protocollo successivo avviene attraverso il valore assunto dal campo `type-length`, che è utilizzato dall'espressione di valutazione contenuta nella sezione `<expr>`.

La maggioranza degli header dei protocolli utilizzano un insieme di campi con caratteristiche abbastastanza standard e che ricadono in cinque diverse categorie. La maggioranza dei campi sono a lunghezza fissa con lunghezza multipla di un byte (da cui l'elemento `<fixed>`), mentre in alcuni casi un campo è composto da bit non allineati al byte (da cui l'elemento `<masked>`). Altri campi sono caratterizzati dal fatto che la loro lunghezza può essere variabile (e ricavabile solamente a run-time) in base ad altre informazioni contenute nel pacchetto, da cui i campi di tipo `<variable>`. Infine, l'elemento `<line>` definisce un campo variabile terminato da un carattere "a capo" e `<padding>` è utilizzato per riallineare un header a multipli di 16 o 32 bit.

```
<proto name="Ethernet">
  <fields>
    <fixed name="dst" size="6"/>
    <fixed name="src" size="6"/>
    <fixed name="type-length" size="2"/>
  </fields>

  <nextproto>
    <switch>
      <expr type="int">
        <fieldref name="type-length">
      </expr>

      <case value="2048"><protoref name="IP"/></case>
      <case value="2054"><protoref name="ARP"/></case>
    </switch>
  </nextproto>
</proto>
```

Figura 1. Estratto della descrizione NetPDL relativa al protocollo Ethernet.

Un campo è completamente caratterizzato dal valori della sua *lunghezza*, del suo *numero di ripetizioni* e dalla sua *posizione* all'interno dell'header, anche se le ultime due informazioni sono spesso superflue (normalmente un campo è presente una sola volta nell'header e la sua posizione è "a seguire" il campo precedente) e pertanto vengono indicate solamente nel caso in cui si differenzino dai valori di default. Viceversa, la lunghezza di ogni campo è espressa da un un attributo `size` che deve essere sempre presente.

2.2. Caratteristiche avanzate del linguaggio NetPDL

Purtroppo, alcuni protocolli di rete hanno funzionalità particolari che non possono essere espresse attraverso i soli elementi precedenti. Ad esempio, perfino un protocollo comune come IP è contraddistinto da una parte obbligatoria (i primi 20 byte) ed una parte opzionale che è presente solamente a fronte del verificarsi di determinate condizioni nell'header iniziale. Il NetPDL gestisce situazioni di questo tipo attraverso la definizione di primitive per l'elaborazione condizionale: il formato dei pacchetti può essere differenziato a seconda della presenza o meno di alcuni valori in certi campi. Nel caso in cui queste funzionalità avanzate non siano sufficienti, il NetPDL prevede il meccanismo del *plug-in* per gestire uno specifico protocollo con codice nativo. Questo meccanismo aggiuntivo permette di mantenere semplice il NetPDL (non è necessario aggiungere primitive ad-hoc per protocolli specifici), pur garantendone il suo utilizzo anche in caso di protocolli "patologici". Ad esempio, il meccanismo dei *plug-in* è utilizzato per la definizione del formato del protocollo DNS, il quale utilizza un meccanismo di compattamento delle informazioni estremamente peculiare e che non si trova in altri protocolli.

2.3. Estensioni del linguaggio NetPDL

Una delle caratteristiche del linguaggio NetPDL, derivate dalla sua natura *XML-based*, è la sua capacità di essere *esteso*, ossia di aggiungere nuove primitive (sotto forma di elementi o attributi XML) al linguaggio stesso. Questo permette alle applicazioni di essere conformi almeno con un set base del linguaggio inn quanto gli elementi sconosciuti vengono ignorati.

La *NetPDL Visualization Extension* è la prima estensione al linguaggio e definisce le modalità con le quali il valore di un campo deve essere visualizzato da un applicativo. Ad esempio, il valore di un campo a 32 bit dovrà essere visualizzato come un numero esadecimale qualora ci si riferisca ad un CRC, mentre dovrà essere visualizzato in notazione decimale puntata qualora sia un indirizzo IP. Questa estensione definisce due diverse viste di un pacchetto: una vista di *sommario* relativamente ad ogni protocollo, nella quale vengono indicati i campi principali del protocollo stesso e i relativi valori assunti nel pacchetto in esame, e una vista di *dettaglio* che elenca tutti i campi (e i relativi valori) presenti all'interno del protocollo stesso. Queste estensioni definiscono nuovi elementi e attributi che vengono utilizzati all'interno di un *template di visualizzazione*, a cui si accede attraverso gli attributi `showsumtemplate` e `showtemplate`, come mostrato in Figura 2.

```
<proto name="Ethernet" longname="Ethernet 802.3“
          showsumtemplate="eth">
  <fields>
    <fixed name="dst" longname="MAC Destination" size="6"
          showtemplate="EthMAC"/>
    <fixed name="src" longname="MAC Source" size="6"
          showtemplate="EthMAC"/>
    <fixed name="type-length" longname="Ethertype - Length" size="2“
          showtemplate="FieldHex"/>
  </fields>
  ...
</proto>
...
<netpdlshow>
  <showtemplate name="FieldHex" showtype="hex"/>
  <showtemplate name="EthMAC" showtype="hex" showgrp="3" showsep="-"/>

  <showsumtemplate name="ethernet">
    <section name="next"/>
    <text value="Eth: "/>
    <pdmlfield name="src" attrib="show"/>
    <text value=" => "/>
    <pdmlfield name="dst" attrib="show"/>
  </showsumtemplate>
</netpdlshow>
```

Figura 2. NetPDL Visualization Extension relativamente all'header Ethernet.

Tra le informazioni più importanti contenute in un template di visualizzazione vi sono gli attributi `showtype`, `showgrp`, e `showsep`, che indicano rispettivamente il formato (esadecimale, decimale, ascii o binario) di ogni byte, come i byte devono essere raggruppati insieme, e il carattere di separazione tra i vari gruppi. Ad esempio, i campi *MAC Source* e *MAC Destination* in Figura 2 sono associati al template di visualizzazione `EthMAC`, il quale visualizza i valori dividendo l'indirizzo MAC in due parti da tre byte ciascuno (come specificato dall'attributo `showgrp`), visualizzando le due porzioni in esadecimale (attributo `showtype`) separate dal carattere "`-`" (attributo `showsep`), con un risultato finale ad esempio di `000800-AB34F9`.

In Figura 2 è mostrato anche il template di visualizzazione relativo alla vista di sommario relativa al protocollo Ethernet: ogni trama verrà associata ad una

vista repilogativa iniziante con la stringa "`Eth:`" seguita dall'indirizzo MAC sorgente, dalla stringa "=>" e dall'indirizzo MAC destinazione, con un risultato simile al seguente: `Eth: 0001C7-B75007 => 000629-393D7E`

3. Analisi prestazionale del linguaggio NetPDL

La critica maggiore al linguaggio NetPDL è relativa alle presunte minori prestazioni di uno strumento che utilizza questa tecnologia (completamente generica e svincolata da ogni protocollo) anzichè una tecnologia meno generale, ma più immediata e realizzabile con codice nativo (ad es. i primi 6 byte di una trama vengono assegnati all'indirizzo MAC destinazione). Intuitivamente, la seconda soluzione, ancorchè di validità limitata (non funzionerebbe in caso di una trame WiFi), sembra nettamente più efficiente.
Per smentire questa critica sono stati effettuati alcuni test prestazionali miranti a confrontare il modulo di decodifica dei pacchetti implementato nella libreria NetBee [2] (che utilizza NetPDL per la decodifica di pacchetti) con Tethereal [3], uno sniffer a riga di comando. I test, eseguiti su un PC Pentium IV – 2.4GHz, mirano a decodificare un insieme di pacchetti contenuti in alcuni file. I risultati indicano che le prestazioni ottenute da NetBee e Tethereal sono estremamente simili, con un tempo di processamento medio per pacchetto rispettivamente di 75 e 66 µs. Questi risultati forniscono una buona indicazione delle prestazioni ottenibili attraverso l'impiego della tecnologia NetPDL, dimostrando chiaramente come software basati su NetPDL possono essere estremamente competitivi anche quando vengono confrontati con software basati su codice nativo: in altre parole, la tecnologia NetPDL non inserisce, in sè, particolari penalizzazioni prestazionali in quanto la velocità di elaborazione dipende soprattutto dalla qualità del motore di elaborazione basato su questo linguaggio.

4. Conclusioni

Il linguaggio NetPDL può essere utilizzato per creare applicativi slegati da ogni protocollo di rete e pertanto potenzialmente capaci di supportare nuovi protocolli semplicemente aggiornando i relativi file NetPDL. Questo può essere estremamente interessante per applicativi legati alla sicurezza (firewall, IDS, monitor) nei quali è necessario essere estremamente rapidi a supportare nuovi protocolli di rete (anche a livello applicativo), ed è fondamentale supportare link-layer di diverso tipo (Ethernet, WiFi, ecc.); pertanto la possibilità di creare applicativi svincolati da specifici protocolli può essere di grande aiuto per gli sviluppatori. Questo articolo presenta anche la libreria NetBee [2], una implementazione estremamente efficiente del NetPDL per quanto riguarda le decodifica e il filtraggio di pacchetti, che è liberamente disponibile su Internet.

5. Bibliografia

[1] NetGroup, Politecnico di Torino, Il linguaggio NetPDL, Maggio 2003. Disponibile all'indirizzo http://www.nbee.org/NetPDL

[2] NetGroup, Politecnico di Torino, NetBee, Agosto 2004. Disponibile all'indirizzo http://www.nbee.org

[3] Ethereal, Analizzatore di rete. Disponibile all'indirizzo http://www.ethereal.com