UNIVERSITA' DEGLI STUDI DEL SANNIO

FACOLTA' DI INGEGNERIA

CORSO DI LAUREA SPECIALISTICA IN INGEGNERIA INFORMATICA

Misure Su Reti di Calcolatori

ANALISI E SVILUPPO DI UN ANALYZER PER PROTOCOLLO SIP

PROF:

Dott. Ing. Luca De Vito

STUDENTI:

Francesco CIOFFI

Ausilio DI PRIZITO

Silvio GRECO

Giorgio MENNITTO

Luca PERROTTA

ANNO ACCADEMICO 2006/2007

Indice generale

1 Introduzione

1.1 Caratteristiche del protocollo

Il protocollo Session Initiation Protocol (SIP) è basato su IP ed è utilizzato principalmente per applicazioni di telefonia su IP o VoIP. La sua descrizione e definizione viene data dalla Request for comment (RFC) 3261.
Gestisce una sessione di comunicazione tra due o più entità: fornisce i meccanismi per instaurare, modificare e terminare una sessione. Il protocollo SIP permette il trasferimento di diverse tipologie di dati quali audio, video messaggistica testuale etc.
Oggi il SIP è il protocollo VoIP più diffuso sul mercato, anche grazie al fatto che esso favorisce un'architettura modulare e scalabile e quindi capace di crescere proporzionalmente al numero degli utilizzatori.
Il SIP nasce basato sul protocollo di trasporto UDP su porta 5060, ma negli ultimi tempi le revisioni di questo standard ne hanno permesso l'utilizzo anche attraverso TCP e TLS.
Le funzioni fondamentali del protocollo SIP possono essere così elencate:

- localizzazione degli utenti: acquisire le preferenze degli utenti;
- invito agli utenti per partecipare ad una sessione: negoziare le capability e trasportare una descrizione della sessione;
- instaurazione delle connessioni di sessione;
- gestione delle eventuali modifiche dei parametri di sessione;
- rilascio delle parti;
- cancellazione della sessione in qualunque momento si desideri.

La sintassi del SIP è basata su codifica ASCII. L'instaurazione di una sessione avviene tramite un three-way handshaking.

1.1.1 Messaggi SIP

Un messaggio SIP è costituito dalla chiamata di un metodo seguito da una serie di campi detti headers. Separati da una riga vuota ci sono gli headers del protocollo Session Description Protocol (SDP) (vedi RFC 2327), che segnalano, all'host contattato, il tipo di sessione multimediale che verrà instaurata per la comunicazione (es. RTP).

Un esempio di messaggio SIP è:

INVITE *sip:utente@domain.com SIP/2.0*
Via: *SIP/2.0/UDP 134.102.18.1*
From: *<sip:Pippo@dominio.com>; tag = 4711* ***"//identifica colui che da origine alla richiesta "***
To: *Topolino <sip:utente@domain.com>* ***"//identifica la destinazione logica di una richiesta"***
Call-Id: *12345678@134.102.18.1* ***"//è un valore costante che identifica l'invito"***
Cseq: *49 Invite* ***"//ordina le transazioni (la prossima richiesta avrà Cseq=50)"***
Content-Length: *117* ***"//il body consiste in 117 byte "***
Content-Type: *application /sdp* ***"//tipo di messaggio/protocollo che segue gli headers SDP"***
Subject: *felicitazioni!* ***"//l'oggetto del messaggio"***
Contact: *sip:Pippo@134.102.18.1:4050* ***"//l'indirizzo al quale si desidera ricevere richieste"***
transport *= udp* ***"//specifica il protocollo di trasporto, nell'esempio UDP***
v *= 0* ***"//indica la versione in uso"***
o *= jack 7564657 9823872 IN IP4 134.102.18.1* ***"//l'owner della risorsa con un ID di sessione"***
c *= IN IP4 134.102.18.1* ***"//tipo di rete, la versione del protocollo IP e l'IP stesso "***
t *= 0 0* ***"//tempo di start e di stop"***
m *= audio 4754 RTP/AVP 0* ***"//tipo di media, numero di porta, protocollo di trasporto e formato "***
a *= rtpmap: 0 PCMU/8000* ***"//indica eventualmente attributi audio\video"***
s *= festa* ***"//oggetto della sessione"***

Nei messaggi SIP, i metodi più utilizzati sono:

Register:	inviato da uno User Agent quando vuole registrare presso un Registrar Server il proprio punto di ancoraggio alla rete
Bye:	Utilizzato per porre fine al dialogo SIP
Cancel:	per terminare un dialogo se la sessione non ha ancora avuto inizio
Invite:	serve ad invitare un utente a partecipare ad una sessione
Ack:	messaggio di riscontro
Trying e Ringing:	messaggi provvisori; mantengono i parametri della richiesta a cui rispondono
Subscribe e Notify:	utilizzati per E-Presence

1.1.2 Architettura di una rete SIP

Le entità di una rete SIP possono essere divise in tre categorie:

- SIP User Agent: è un end-point che può fungere da client o da server con ruoli dinamici. Durante la sessione un client può fare da server e viceversa. Nel caso sia client genera richieste, mentre se funge da server ascolta le richieste e, se possibile, le serve.

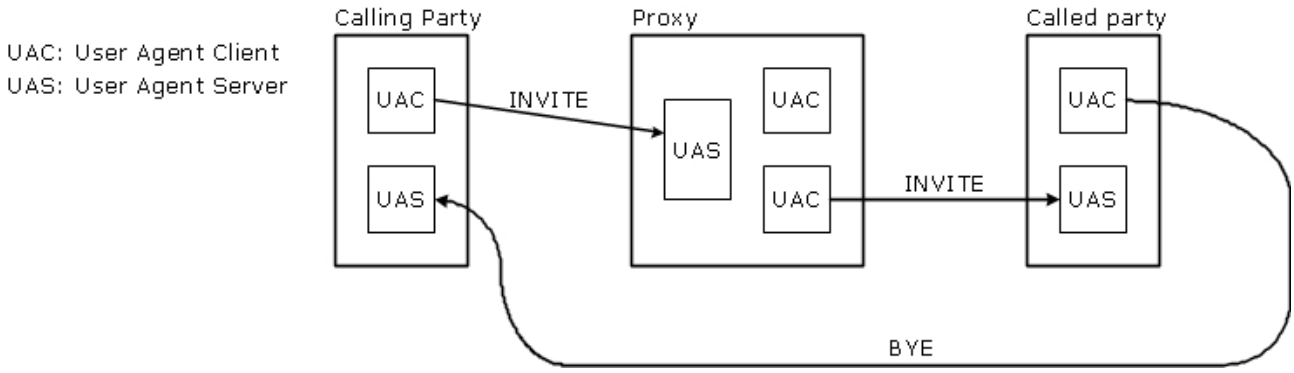


Illustrazione 1: Schema di chiamata tra User Agents

- **Registrar Server:** può essere un server dedicato o collocato in un proxy. Quando un utente è iscritto ad un dominio invia un messaggio di registrazione del suo attuale punto di ancoraggio alla rete ad un Registrar Server
- **Proxy Server:** è un server intermedio che si fa carico di rispondere direttamente alle richieste o di reindirizzarle ad un client, ad un server od ad un ulteriore proxy. Questi analizza i parametri di instradamento dei messaggi e nasconde la reale posizione del destinatario del messaggio, infatti esso sarà indirizzabile con un nome convenzionale del dominio di appartenenza.

  Si definisco anche diverse tipologie di proxy:
  - **Outbound-Proxy:** quando uno User Agent invia sistematicamente le proprie richieste ad un proxy vicino.
  - **Inbound-Proxy:** instrada le chiamate entranti in un dominio
  - **Forking-Proxy:** che può instradare una stessa richiesta in parallelo o in sequenza a più destinazioni
- **Redirect Server:** reinstrada le richieste SIP consentendo al chiamante di contattare un insieme alternativo di URI
- **Location Server:** è un database contenente informazioni sull'utente, come il profilo,l'indirizzo IP e l'URL.

1.2 Scopo del Lavoro

Lo scopo del lavoro è l'analisi ed il completamento di un software platform-independent per la memorizzazione e la verifica di messaggi SIP. Si è partiti dall'analisi di un software opensource distribuito in rete, chiamato SIPANAlyzer. Successivamente si è sviluppata una web-application in Java per compensare alle carenze del suddetto software.

2 Lo Stato dell'arte

Il software preso in esame è il SIPAnalyzer della Advanced Network Technology.
È composto di due parti: un Agent, che imposta la scheda di rete in modalità promiscua e cattura i messaggi SIP provenienti dalla rete per memorizzarli in un database MySQL; un Analyzer, che ha il compito di verificare i messaggi scritti nel database dall'agent. Il progetto, seppur incompleto, è stato abbandonato nel 2000. Questo ha creato alcune problematiche che saranno discusse in seguito.
Il nostro lavoro è partito dall'analisi dell'Agent, scritto in linguaggio C, con particolare attenzione alle problematiche di messa in opera.
Requisiti del sistema sono: server MySQL, ambiente Java.
Il primo problema affrontato è stato adattare il programma alle ultime versioni di MySQL.
Il programma scritto per MySQL 4 è stato quindi adattato a MySQL 5.
Il codice rilasciato per il SIPAnalyzer nella sua parte di Agent compie le seguenti azioni:

- apertura di una socket TCP/UDP in ascolto;
- apertura della connessione al database;
- cattura dei dati;
- analisi dei dati (esclusione di header non facenti parte di messaggi SIP)
- scrittura nel database di
  - no: numero identificativo della tupla all'interno della tabella
  - agentnum: identificativo dell'agent
  - buff: il messaggio inviato/ricevuto
  - sec, usec: paramteri usati per il timing
  - sourceip: indirizzo IP del mittente
  - sourceport: porta del mittente
  - desip: indirizzo IP di destinazione
  - desport: porta del destinatario

  - worknum: valore fisso a 99 che dovrebbe indicare il numero progressivo dell'elaborazione
  - protocoltype: valore fisso a 0 che dovrebbe indicare la tipologia del protocollo utilizzato.

In questa fase, il programma si limita a prendere i messaggi SIP provenienti sulle porte indicate e a memorizzarli nel database.
Come è evidente, non si può parlare di analizzatore di messaggi SIP, avendo solo memorizzato gli stessi.
L'analyzer fornito a supporto consiste in un sistema PHP, che permette la sola visualizzazione dei dati memorizzati. Dalle informazioni reperibili sul sito, sembra fossero in fase di sviluppo altre peculiarità, poi abbandonate insieme al progetto.

3 JSipAna2

Nasce quindi l'esigenza di completare il lavoro svolto dal gruppo di sviluppo del SIPAnalyzer, creando anche un Analyzer.

Per la progettazione ci si è orientati verso un linguaggio platform-independent, già ampiamente conosciuto dai componenti del team, che consentisse una semplice integrazione web, ed una forte modularizzazione sia in fase di sviluppo che di estensione.

JSipAna2 è stato sviluppato in Java.

La scelta del database, a differenza del linguaggio di programmazione, è stata obbligata da quelle che erano state le preferenze degli sviluppatori dell'Agent che abbiamo utilizzato.

3.1 Ambiente di sviluppo e classi

L'ambiente di sviluppo scelto per la programmazione dell'Analyzer è Eclipse, il quale ha consentito una semplice integrazione tra la parte di programmazione e quella di pubblicazione con Apache Tomcat 5.5.

Il lavoro svolto è facilmente comprensibile e modificabile grazie alla adattabilità di Java sia per quel che riguarda le piattaforme sottostanti, sia per quel che riguarda la progettazione.

Sono state create classi che riuscissero a dare la reale immagine del problema e che contemporaneamente consentissero una struttura ordinata.

Il programma si compone di oggetti detti Bean ognuno dei quali contiene le caratteristiche peculiari dell'oggetto reale:

- Message: definisce un messaggio con i relativi campi e il loro contenuto

- MessageField: i campi del messaggio sono definiti da un ulteriore oggetto che ne indica le caratteristiche
- MessageError: le caratteristiche degli errori sono memorizzate secondo regole prestabilite quali assenza di un campo, documento sconosciuto, campo non previsto etc.
- RulesField: ogni capo deve rispettare delle regole quindi in questo oggetto, associato al nome del capo ci sono le regole relative al campo stesso
- DynamicRule: è possibile definire regole sui campi in maniera dinamica, definite quindi dall'utente utilizzatore. Questo oggetto definirà appunto tali regole

Sono state successivamente definite classi che consentissero di estrarre i dati dal database, popolato dall'Agent, e memorizzarli come gli oggetti sopra elencati:

- DBHandler: questa classe si occupa di gestire il database aprendo e chiudendo la connessione realtiva. In essa sono anche contenuti i dati per la connessione stessa
- MessageDAO: grazie a questa classe viene prelevata una collezione dei messaggi che sono contenuti nel database
- RulesDAO: questa classe si occupa di prelevare i dati relativi alle regole statiche sui campi
- DynamicRulesDAO: preleva dal database le eventuali regole dinamiche in esso contenute e definite dall'utente

Fulcro centrale dell'analyzer sono le classi MessageAnalyzer e SessionAnalyzer che, sfruttando le classi precedentemente descritte, analizzano i messaggi e le sessioni durante le quali vengono scambiati i messaggi.

In particolare il MessageAnalyzer permette di effettuare l'analisi sintattica dei messaggi SIP secondo la RFC 3126.

Controlla tre regole:

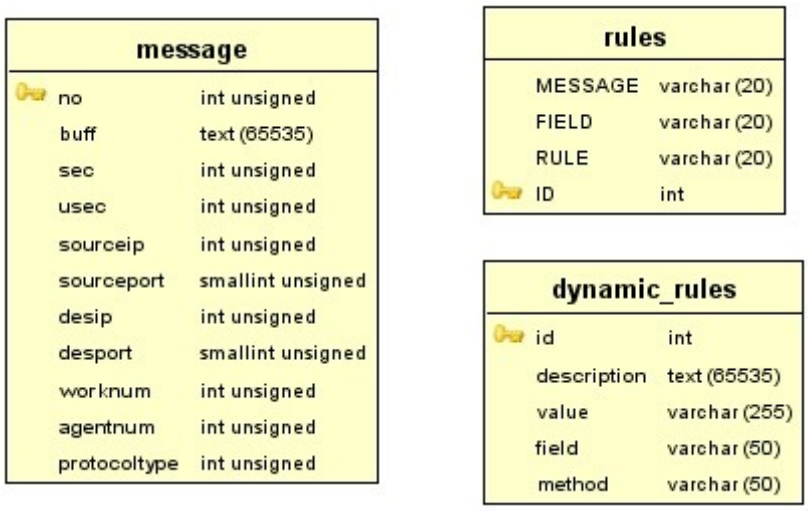
- campo mancante;
- campo sconosciuto;
- campo duplicato.

Inoltre verifica la regolarità dei campi obbligatori del messaggio SIP. Nello specifico:

- ogni campo deve obbligatoriamente avere come regola: MANDATORY (tranne se opzionale)
- se è presente una sola volta deve essere: SINGLE (es. To, From ...)

- se presente per più di una volta non deve avere come regola SINGLE, ma solo MANDATORY (es. Via)

3.2 Database



La struttura del database è data dalle tabelle sovraillustrate.

La tabella MESSAGE, è popolata dall'Agent del SIPAnalyzer, il quale inserisce in essa tutti i dati relativi al messaggio, come destinatario e mittente, i relativi indirizzi IP e porte e il messaggio stesso.

Un esempio di dati contenuti in essa è dato dal seguente schema:

no	buff	sec	usec	sourceip	sourceport	desip	desport	worknum	agentnum	protocoltype
4957	(MEMO)	1181296437	943529	3232235522	5061	3232235527	5060	99	3232235527	0
4958	(MEMO)	1181296437	944354	3232235527	5060	3232235522	5061	99	3232235527	0
4959	(MEMO)	1181297094	443200	3232235525	5063	3232235527	5060	99	3232235527	0
4960	(MEMO)	1181297094	445767	3232235527	5060	3232235525	5063	99	3232235527	0
4961	(MEMO)	1181297097	770997	3232235525	5063	3232235527	5060	99	3232235527	0
4962	(MEMO)	1181297107	533772	3232235525	5064	3232235527	5060	99	3232235527	0
4963	(MEMO)	1181297107	534407	3232235527	5060	3232235525	5064	99	3232235527	0
4964	(MEMO)	1181297147	211128	3232235525	5065	3232235527	5060	99	3232235527	0
4956	(MEMO)	1181297147	213901	3232235527	5060	3232235525	5065	99	3232235527	0

Come mostrato dalla tabella, Message contiene come chiave primaria un numero progressivo e i dati relativi al messaggio.

Nella colonna buff è contenuto il testo del messaggio in questa forma

INVITE sip:192.168.0.7 SIP/2.0

Date: Fri, 08 Jun 2007 09:52:30 GMT

CSeq: 1 INVITE

Via: SIP/2.0/UDP 192.168.0.2:5061;branch=z9hG4bK6c3fd8a3-1314-dc11-8937-000fb06abfda;rport

User-Agent: Ekiga/2.0.3

From: "LuCa" <sip:luca@192.168.0.2>;tag=c299d7a3-1314-dc11-8937-000fb06abfda

Call-ID: 62e1d6a3-1314-dc11-8937-000fb06abfda@none

To: <sip:192.168.0.7>

Contact: <sip:luca@192.168.0.2:5061;transport=udp>

Allow: INVITE,ACK,OPTIONS,BYE,CANCEL,NOTIFY,REFER,MESSAGE\r\nContent-Length: 336

Content-Type: application/sdp

Max-Forwards: 70

v=0

o=- 1181296350 1181296350 IN IP4 192.168.0.2

s=Opal SIP Session

c=IN IP4 192.168.0.2

t=0 0

m=audio 5000 RTP/AVP 101 96 3 107 110 0 8

a=rtpmap:101 telephone-event/8000

a=fmtp:101 0-15

a=rtpmap:96 SPEEX/16000

a=rtpmap:3 GSM/8000

a=rtpmap:107 MS-GSM/8000

a=rtpmap:110 SPEEX/8000

a=rtpmap:0 PCMU/8000

a=rtpmap:8 PCMA/8000

In questo caso si mostra un messaggio di INVITE.

Gli ulteriori dati in essa contenuti sono relativi all'agent utilizzato e agli indirizzi di mittente e ricevente.

La tabella RULES definisce le regole dettate dalla RFC alle quali un messaggio deve attenersi per essere considerato valido

MESSAGE	FIELD	RULE	ID
INVITE	VIA	MANDATORY	1
INVITE	MAX-FORWARDS	MANDATORY	3
INVITE	TO	MANDATORY	4
INVITE	FROM	MANDATORY	5
INVITE	CALL-ID	MANDATORY	6
INVITE	CSEQ	MANDATORY	7
INVITE	CONTACT	MANDATORY	8
INVITE	CONTENT-TYPE	MANDATORY	9
INVITE	CONTENT-LENGTH	MANDATORY	10
INVITE	DATE	MANDATORY	52

nella colonna MESSAGE si vedrà il tipo di messaggio, nella colonna FIELD i campi relativi a quel

messaggio, nella colonna RULE ci sarà la regola per quel campo.

I valori contenuti in questa colonna potranno essere:

- MANDATORY: che definisce che il campo all'interno del messaggio è obbligatorio e deve essere necessariamente presente
- OPTIONS: se il campo non è obbligatorio
- SINGLE: che definirà che il campo può essere presente una sola volta all'interno del messaggio

La tabella RULES consente all'analyzer di verificare la correttezza dei messaggi e all'occorrenza di segnalare eventuali errori.

La tabella DYNAMIC_RULES definisce regole esterne alla RFC che l'utente può dettare per consentire all'analyzer un ulteriore filtraggio dei messaggi.

id	description	value	field	method
2	(MEMO)	Ekiga	USER-AGENT	INVITE

Questa tabella è composta da una colonna per l'ID che fa anche da chiave primaria. Una per la descrizione (DESCRIPTION) nella quale può essere inserito del testo che meglio definisce questa regola, il campo METHOD nel quale si sottolinea la tipologia del messaggio. La colonna FIELD definisce quale dei campi del messaggio definito in METHOD avrà valore VALUE, ultima colonna della nostra tabella.

Nell'esempio, con la regola definita in tabella, saranno segnalati i messaggi INVITE che avranno nel campo USER-AGENT il valore Ekiga (nome di un linux free software per il VoIP).

4 Funzionalità

4.1 Motore di regole statiche

Parte centrale dell'analyzer sviluppato è il motore di regole statiche.

Il messaggio, contenuto nel database (MESSAGE), viene letto e suddiviso nei vari campi. Per ogni campo si effettua una ricerca all'interno delle regole (RULES) e si realizza un controllo di verifica del rispetto di queste regole. In un messaggio si rileva la presenza di uno dei seguenti errori:

- campo mancante: uno dei campi definiti MANDATORY per la specifica tipologia di messaggio non è presente all'interno del messaggio in analisi.
- campo sconosciuto: uno dei campi presenti nel messaggio non è presente nella tabella delle regole.
- campo singolo: uno dei campi definiti SINGLE all'interno della tabella delle regole, è ripetuto nel messaggio

Questa tabella consente di modificare l'analyzer con il cambiare del protocollo SIP. Sarà infatti

necessario aggiungere ovvero eliminare delle tuple oppure correggere alcuni campi, per modificare anche il comportamento dell'analyzer. Questo lo rende adatto a futuri utilizzi e sviluppi del protocollo.

4.2 Motore di regole dinamiche

Nell'eventualità che l'utente voglia filtrare i messaggi secondo una determinata regola da esso stabilita (vedi esempio precedente relativo all'USER-AGENT) potrà farlo definendo questa regola all'interno della tabella DYNAMIC-RULES.

In questo caso il messaggio prelevato dalla tabella MESSAGE, dopo essere stato esaminato secondo le regole statiche, sarà filtrato secondo le regole dinamiche definite nella tabella dall'utente.

4.3 Layout

Nell'immagine seguente si vedrà il layout dell'analizzatore, su pagina web: nella prima parte ci saranno tutti i dati relativi al messaggio, mentre evidenziati in rosso, in una prima riga, si vedranno eventuali errori presenti all'interno del messaggio dovuti alle regole statiche ed, in una seconda riga, gli errori dati dalle regole dinamiche definite dall'utente

In questo caso è stato rilevato un errore (STATIC RULES) dovuto alla mancanza di un campo (CONTENT-LENGTH) e un altro dovuto alla presenza del valore “Ekiga” all'interno del campo USER-AGENT (DYNAMIC RULES).

5 Sviluppi futuri

Come illustrato in questo documento, l'analyzer, progettato e sviluppato in tecnologia Java, consente, in accordo con l'evoluzione del protocollo e con le necessità dell'utente, infinite migliorie. In particolare è possibile:

- inserire, eliminare o modificare le regole statiche
- scegliere regole particolari e specifiche inserendo nella tabella delle regole dinamiche il tipo di messaggio, il campo da analizzare e il valore da ricercare.
- Unire il presente software ad analizzatori più generali, sfruttando sia la sua progettazione sia la possibilità dello stesso di essere richiamato via web.
- Utilizzarlo per altri protocolli, sempre basati su scambio di messaggi con codifica ASCII, specificando solamente le regole alle quali il messaggio scambiato deve sottostare.

6 Bibliografia

- http://it.wikipedia.org/wiki/Session_Initiation_Protocol
- http://ant.comm.ccu.edu.tw/sip/
- RFC 3261