POLITECNICO DI MILANO
FACOLTÀ DI INGEGNERIA
Corso di Laurea in Ingegneria delle Telecomunicazioni
Dipartimento di Elettronica e Informazione

IDENTIFICAZIONE AUTOMATICA DI ANOMALIE NEL TRAFFICO SU UN SIP TRUNK

Relatore: Prof. **Giacomo Verticale**

Correlatore: Ing. **Antonio Corghi**

Tesi di Laurea di:

Giorgio Adami
Matricola 751771

ANNO ACCADEMICO 2010/2011

SOMMARIO

In questo lavoro di tesi vengono proposti tre possibili approcci al problema dell'individuazione di eventi anomali che portano ad un degrado della qualità del servizio di SIP Trunking, attraverso l'analisi del traffico di segnalazione SIP presente sul trunk.

A partire dall'estrazione di parametri volumetrici dal traffico di segnalazione, sono illustrate tre implementazioni di Anomaly Detection Systems (ADS), basati rispettivamente sull'applicazione di soglie, sull'impiego di una Support Vector Machine (SVM) in modalità "one-class" e sulla combinazione di quest'ultima con una previa analisi multirisoluzione tramite trasformata Wavelet dell'andamento dei parametri estratti. Le prestazioni di queste implementazioni sono valutate rispetto a tre tipi di anomalie: "link-flap", "CPU-hog" e "loop". Queste sono riprodotte all'interno di un ambiente di emulazione di traffico su SIP Trunk.

Le prove effettuate mostrano che l'ADS che combina l'analisi multirisoluzione dell'andamento dei parametri e la SVM, ottiene risultati migliori delle altre due tecniche nell'individuazione di anomalie "CPU-hog" e di "link-flap", mentre è meno efficace nell'individuazione di "loop" di entità contenuta rispetto al volume di traffico presente sul trunk.

ABSTRACT

In this thesis three approaches to the issue of detecting anomalies in a SIP Trunk environment through the direct monitoring of the SIP traffic are presented.

Using volumetric parameters extracted form signaling traffic, three implementations of Anomaly Detection Systems (ADS) are shown, based respectively on thresholding, on the use of a one-class Support Vector Machine (SVM), and on the combination of a multiresolution analysis with Wavelet transform and the previous one-class SVM.

The performance of each implementation is evaluated with respect to three types of anomalous events: “link-flap”, “CPU-hog” and “loop”. These anomalies are reproduced in an emulation environment of SIP Trunk signaling traffic.

The test results show that the ADS using multiresolution analysis combined with SVM performs better than the others with regard to “link-flap” and “CPU-hog” anomalies, whereas it is less effective in detecting “loop” anomalies if their entity is smaller than the volume of the traffic on the trunk.

INDICE

SOMMARIO 3

ABSTRACT 4

INDICE 5

1. INTRODUZIONE 8

2. SCENARIO DI RIFERIMENTO 11

2.1 SIP Trunk 11

2.1.1 Vantaggi del SIP Trunking 13

2.1.2 Infrastruttura 14

2.2 Lo standard SIP 16

2.3 SIPconnect 17

2.4 SIPp 20

2.5 Anomaly Detection Systems 22

2.5.1 ADS “Classification Based” 24

2.5.2 ADS basati sul Nearest Neighbor 26

2.5.3 ADS basati sul Clustering 28

2.5.4 ADS basati su analisi statistica 29

2.5.5 ADS basati sulla teoria dell’informazione 30

2.5.6 ADS basati su analisi spettrale 30

2.5.7 Individuazione di anomalie “collettive” 31

2.5.8 ADS in sistemi con traffico a pacchetto 31

2.6 Curve di ROC 33

3. PROBLEMATICA E SOLUZIONI PROPOSTE 37

3.1 Problema da affrontare 37

3.2 Approccio 42

3.3 Estrazione delle informazioni da monitorare 43
3.4 Classificatore a soglia 45
3.5 Classificatore Support Vector Machine 49
3.6 Trasformata Wavelet e classificatore Support Vector Machine 52
4. EMULATORE 57
4.1 Call flow 58
4.2 Tempi di interarrivo 60
4.3 Implementazione 62
4.3.1 Chiamate entranti 63
4.3.2 Chiamate uscenti 68
4.4 Ambiente di emulazione 72
5. ANOMALIE 73
5.1 Link-Flap 73
5.2 CPU-Hog 76
5.3 Loop 79
6. RISULTATI 84
6.1 Parametri degli ADS 84
6.2 Emulazione di anomalia “Link-Flap” 85
6.2.1 Esperimento a 40 cps 85
6.2.2 Esperimento a 20 cps 88
6.2.3 Esperimento con volume di traffico presente su SIP Trunk reale 91
6.3 Emulazione di anomalia “CPU-Hog” 94
6.3.1 Esperimento a 40 cps 94
6.4 Emulazione di anomalia “Loop” 97
6.4.1 Esperimento a 40 cps 98
6.4.2 Esperimento a 20 cps 100

6.4.3 Esperimento con volume di traffico presente su SIP Trunk reale 102

7. CONCLUSIONI 107

APPENDICI 109

A. Support Vector Machine 109

B. La trasformata Wavelet 111

C. Parametri emulatore 113

INDICE DELLE FIGURE 118

INDICE DELLE TABELLE 121

BIBLIOGRAFIA 123

1. INTRODUZIONE

La crescente diffusione della tecnologia VoIP, soprattutto in ambito business, unitamente all'affermazione dello standard protocollare SIP per l'interoperabilità applicativa, ha incentivato l'industria delle telecomunicazioni allo sviluppo di nuovi modelli di offerta dei servizi di telefonia pubblica, introducendo il concetto di SIP Trunking.

A differenza della telefonia tradizionale, in cui i punti di accesso alla rete pubblica sono costituiti da circuiti dedicati in tecnologia TDM, il servizio di SIP Trunking permette l'utilizzo di una connessione logica, comunemente denominata SIP Trunk, finalizzata allo scambio di traffico VoIP tra due end-point SIP, rispettivamente localizzati all'interno della rete cliente e del Service Provider. Tale connessione può basarsi su collegamenti IP dedicati, oppure condivisi con altre tipologie di servizio dati.

E' sempre maggiore la tendenza per utenti business di tipo "Large Enterprise" a migrare l'architettura d'accesso alla propria rete telefonica verso la soluzione SIP Trunk, spinti principalmente da ragioni economiche. Tale soluzione, infatti, è in grado di assicurare notevoli risparmi rispetto ai trunk tradizionali, a fronte di un'offerta qualitativamente elevata. Al posto della costosa architettura d'accesso tradizionale, distribuita geograficamente su tutte le sedi territoriali dell'azienda cliente, in un'architettura d'accesso basata su SIP Trunk il traffico telefonico viene raccolto in un numero limitato di siti, nei quali è presente l'interconnessione con la rete telefonica del Service Provider; in questo modo si realizza un sistema che prevede un numero limitato di connessioni, su cui transita l'intero traffico telefonico aziendale.

Diventa necessaria l'individuazione tempestiva degli eventi anomali che possono verificarsi nel sistema e che portano ad un degrado del servizio offerto. Per fare ciò sono attualmente implementati nei dispositivi di rete in commercio, dei sistemi di monitoraggio basati principalmente sulla raccolta di informazioni di autodiagnosi dei dispositivi stessi. Questo, unitamente ad un'architettura di rete completamente ridondata e priva di "Single Point Of Failure", garantisce un'elevata protezione dai guasti di tipo hardware che possono verificarsi nei dispositivi dispiegati.

Tuttavia accade spesso che eventi anomali di diversa natura, dovuti principalmente a guasti di tipo software o errori di configurazione del sistema, non sono identificati dai sistemi di monitoraggio esistenti, ma vengono segnalati direttamente dagli utenti che sperimentano un disservizio.

Avere a disposizione uno strumento in grado di identificare in tempo reale tali eventi anomali, analizzando direttamente il traffico di segnalazione presente sul trunk, aggiunge valore al servizio di SIP Trunking, a beneficio sia del Service Provider che lo offre, che dell'azienda cliente che ne usufruisce.

In letteratura sono presenti numerosi lavori in ambito "Anomaly Detection", ed in particolare riguardanti Anomaly Detection Systems (ADS) per l'individuazione di anomalie di traffico a pacchetto. Questi si distinguono in primo luogo per l'approccio al problema, "signature-based" se viene caratterizzata l'anomalia e si considera normale qualunque comportamento diverso da questa, o "anomaly-based" se viene caratterizzato il traffico di normale esercizio del sistema e si considera anomalo qualunque discostamento da esso. In secondo luogo gli ADS si possono distinguere per la tipologia di parametri utilizzati per caratterizzare il traffico del sistema e per la tipologia di decisione impiegata.

A partire da un'analisi dei lavori esistenti, in questo documento sono proposte tre implementazioni di Anomaly Detection Systems per il monitoraggio real-time di parametri volumetrici estratti dal traffico presente sul SIP Trunk. La prima implementazione è basata sull'applicazione di soglie ai parametri estratti mentre la seconda fa uso di una Support Vector Machine in modalità "one-class"; la terza soluzione proposta applica un'analisi multirisoluzione, basata su trasformata wavelet discreta, su una finestra temporale di osservazione dei parametri estratti, impiegando poi una SVM "one-class".

Al fine di valutare le prestazioni degli Anomaly Detection Systems implementati, è stato realizzato un emulatore di traffico telefonico su SIP Trunk con l'ausilio di SIPp, un generatore di traffico SIP molto diffuso. Il traffico è riprodotto secondo parametri estratti da tracce di traffico reale su SIP Trunk.

Le performance delle soluzioni proposte sono valutate rispetto a tre tipologie di eventi anomali che il sistema SIP Trunk sperimenta con maggior frequenza, ovvero il “link-flap”, il “CPU-hog” e il “loop”.

Nel secondo capitolo di questo documento di tesi è introdotto il concetto di SIP Trunk, i modelli architetturali e gli standard protocollari che caratterizzano un servizio di SIP Trunking; inoltre è presentato SIPp, il software utilizzato per la realizzazione dell’ambiente di emulazione, una panoramica dello stato dell’arte delle principali tecniche di “Anomaly Detection” presenti in letteratura ed infine sono brevemente presentate le curve di ROC, strumento grafico per la valutazione delle prestazioni degli Anomaly Detection Systems.

Nel terzo capitolo è innanzitutto presentato il problema che questo lavoro di tesi intende affrontare, in seguito sono illustrati i tre differenti approcci di ADS.

Nel quarto capitolo è presentato l’ambiente di emulazione, realizzato sulla base di alcuni parametri e caratteristiche estratte da tracce di traffico catturate in un SIP Trunk in esercizio.

Il quinto capitolo illustra i tre tipi di anomalie presi come riferimento per la valutazione delle prestazioni degli ADS proposti, riportando le tecniche utilizzate per riprodurli all’interno dell’ambiente di emulazione realizzato.

Nel sesto capitolo sono riportati i risultati dei test sperimentali eseguiti, analizzando le prestazioni delle soluzioni proposte per ogni situazione anomala riprodotta.

2. SCENARIO DI RIFERIMENTO

In questo capitolo è presentato il contesto di riferimento per il presente lavoro di tesi, illustrando brevemente l'architettura della soluzione SIP Trunk, gli standard protocollari in uso ed una panoramica sullo stato dell'arte degli Anomaly Detection Systems.

2.1 SIP Trunk

Il termine "trunk" deriva dal mondo legacy della telefonia a commutazione di circuito, dove esso rappresenta un collegamento fisico tra due switch, ed instaura tra di essi una relazione uno-a-uno. Jonathan Rosenberg, autore di gran parte degli standard IETF che definiscono il protocollo SIP (Session Initiation Protocol), sottolinea come questo termine sia spesso utilizzato nel mondo SIP senza i dovuti chiarimenti. Egli definisce il SIP Trunk come un'entità SIP virtuale posta su di un server, che può essere sia un UAS (User Agent Server) che un UAC (User Agent Client) che un proxy; tale entità è limitata da un insieme predefinito di policy e regole che determinano le modalità di elaborazione delle richieste che pervengono ad essa [1]. E' quindi evidente la differenza tra le due realtà in cui lo stesso termine è utilizzato.
Attualmente il termine "SIP Trunk" è utilizzato per indicare:

- Un servizio offerto da un Service Provider verso un cliente Enterprise, atto all'interconnessione PSTN, al fine di sostituire una connessione a commutazione di circuito.
- Una porta SIP su un server Enterprise, con lo scopo di interconnessione ad altri sistemi "server-based" quali server voicemail, call-centers e application-servers.
- Una interconnessione SIP tra IP-PBXs che sostituisce una linea TDM tradizionale.

Il SIP Trunk è quindi, concretamente, una connessione logica tra due end-point SIP, finalizzata allo scambio di traffico VoIP (Voice over IP). I due end-point sono dei Back-to-Back User Agent (B2BUA), ovvero si comportano come UAS nei confronti di altri UAC e viceversa, e sono situati uno all'interno della rete Enterprise, l'altro nella rete

del Service Provider. Sono situati al “bordo” della rete in cui si trovano, perciò vengono spesso definiti Border Element.

Quando un utente all’interno della rete Enterprise, tipicamente rete aziendale, desidera instaurare una chiamata verso una numerazione esterna, l’end-point aziendale fornisce al suo pari, lato Service Provider, tutte le informazioni necessarie ad instaurare la chiamata verso il numero digitato, comportandosi come intermediario per la chiamata stessa. Segnalazione e traffico voce sono scambiati tra i due Border Element utilizzando rispettivamente il protocollo SIP e il protocollo RTP (Real Time Protocol).

Se la numerazione chiamata corrisponde ad un dispositivo appartenente ad una rete telefonica PSTN tradizionale, l’end-point del Service Provider instraderà i pacchetti verso il gateway più vicino al numero chiamato, in modo da minimizzare le spese dovute alla distanza geografica tra chiamante e chiamato. Lo stesso end-point situato a bordo della rete del Service Provider, oltre a gestire le chiamate in uscita dalla rete Enterprise, è in grado di instradare sul trunk le chiamate verso utenze appartenenti alla rete Enterprise. In questo modo è possibile, per l’azienda che acquista il servizio di SIP Trunking, mantenere la numerazione geografica servendo, da un unico sito, numeri telefonici locali situati in diverse aree.

Se la chiamata non necessita di essere instradata sulla rete PSTN, viene trasportata end-to-end sulla rete IP, sfruttando il SIP Trunk, riducendo così il numero di accessi alla rete telefonica tradizionale. Così facendo i Provider riescono a recapitare queste chiamate ai loro clienti con costi molto bassi, potendosi quindi permettere di offrire gratuitamente questa tipologia di chiamate. Inoltre alcuni Service Provider stipulano tra di loro accordi commerciali per lo scambio di chiamate dirette alla propria rete IP o provenienti da essa. Una chiamata che coinvolge due Provider i quali non hanno stabilito alcun accordo di questo tipo, verrà instradata attraverso la rete PSTN.

Il SIP Trunk può essere fornito sia dall’Internet Service Provider (ISP) che da un Internet Telephony Service Provider (ITSP) indipendente.

Come spiegato precedentemente, il SIP Trunk non è una connessione fisica vera e propria, perciò non esiste un limite esplicito al numero di chiamate concorrenti che

possono essere trasportate su un singolo Trunk. Visto che ogni chiamata occupa una porzione di banda, l'unico limite è rappresentato dalla capacità fisica messa a disposizione per il servizio voce. Altrimenti il numero di chiamate concorrenti potrebbe essere limitato da accordi commerciali, solitamente imposti dal Provider nell'offerta del servizio di SIP Trunk. Invece il tradizionale trunk TDM (Time Division Multiplexing), essendo un collegamento fisico, possiede un numero limitato di canali, supportando quindi un numero non superiore di chiamate concorrenti.

Il SIP Trunk facilita quindi l'instaurazione di comunicazioni real-time, sia voce che video, tramite la rete IP, e sono ampiamente utilizzati per sostituire i tradizionali Trunk TDM [2].

2.1.1 Vantaggi del SIP Trunking

Molte aziende utilizzano già la tecnologia VoIP per le comunicazioni interne, sfruttando la rete LAN aziendale. Così facendo rimane necessaria la presenza di un gateway PSTN aziendale di bordo rete per le comunicazioni dirette all'esterno dell'azienda, e seppur si riducono i costi delle chiamate interne, non si hanno sostanziali risparmi per l'azienda. Utilizzando una soluzione SIP Trunk è possibile portare la tecnologia VoIP anche fuori dai confini della rete LAN aziendale, abbattendo considerevolmente i costi [3]. Per questo motivo è in corso una progressiva migrazione verso questa soluzione tecnologica.

Di seguito sono riportati i principali benefici riscontrabili da un'azienda che usufruisce di servizi SIP Trunking:

- Nessun costo aggiuntivo dovuto ai canoni di BRI (Basic Rate Interface) e PRI (Primary Rate Interface).
- Nessun investimento necessario per gateway PSTN.
- Scalabilità semplificata ed economicamente vantaggiosa.
- Utilizzo ottimizzato della banda, inviando dati e voce sulla stessa connessione.
- Non è necessario l'acquisto di canali telefonici in gruppo, godendo quindi di massima flessibilità nel dimensionamento ed utilizzo del canale.
- Risparmio sui costi delle chiamate; telefonate ovunque dirette hanno costi comparabili a chiamate locali.

- Ridondanza offerta da provider e connessioni multiple.

I vantaggi di una soluzione SIP Trunk, rispetto ad una connessione telefonica tradizionale, si possono riassumere in un risparmio economico a parità di qualità del servizio offerto. Infatti i costi delle infrastrutture necessarie alla realizzazione di un SIP Trunk possono arrivare ad essere il 50% inferiori di un collegamento tradizionale, e la tariffazione per le comunicazioni a lunga distanza è notevolmente inferiore, essendo paragonabile a quelle locali. Inoltre una soluzione SIP Trunk permette l'acquisto della sola capacità necessaria, garantendo anche un'ottima scalabilità.

2.1.2 Infrastruttura

In generale solo un numero limitato di componenti caratterizza i sistemi che includono un servizio di SIP Trunking; tali componenti possono essere implementati sia lato Service Provider che lato Enterprise.

Un Service Provider che offre un servizio di SIP Trunking, deve garantire caratteristiche funzionali e qualitative tali da convincere il cliente e allo stesso tempo generare profitto.

I componenti di rete dedicati al SIP Trunking sono sostanzialmente gli stessi sia lato Service Provider che lato cliente Enterprise e si differenziano principalmente per dimensione e complessità di dispiegamento. I componenti lato Provider sono ottimizzati in modo da necessitare della minima manutenzione possibile, rendendo il servizio redditizio, mentre i componenti lato Enterprise sono progettati al fine di offrire servizi a valore aggiunto all'azienda cliente.

Il principale componente di rete per l'offerta di un servizio SIP Trunk lato Service Provider è l'Edge SBC (Session Border Controller). In generale un SBC è dispiegato nelle reti che offrono servizi VoIP per poter esercitare controllo sulla segnalazione e generalmente anche sui flussi media, figurando da intermediario per ogni comunicazione tra la rete che offre il servizio e il cliente che ne usufruisce (figura 1). L'Edge SBC si distingue da un normale SBC per la piena interoperabilità tra tutti gli indirizzi IP della rete aziendale del cliente e lo spazio di indirizzamento del Service Provider.

Inoltre garantisce la sicurezza alla rete del Service Provider proteggendola da eventuali attacchi provenienti dalla rete cliente, incorporando funzionalità di firewall di bordo rete e di Intrusion Protection. L'Edge SBC deve essere resiliente, offrendo una disponibilità prossima al 100%.

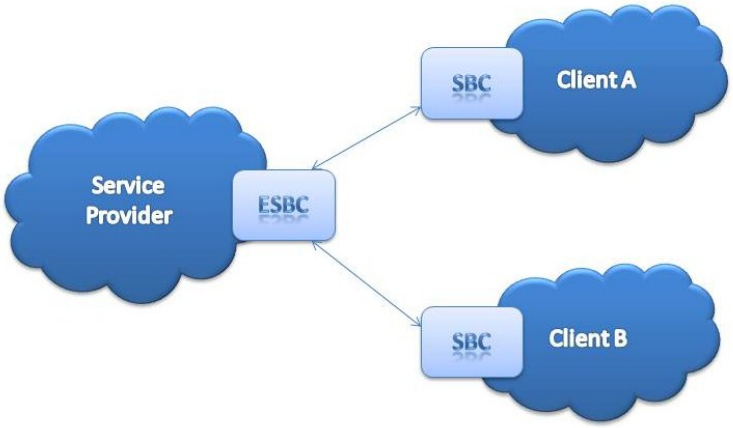


Figura 1 - Architettura generale SBC di bordo rete

Una componente basilare, solitamente data per scontata, è la disponibilità di un'infrastruttura IP di qualità. Questo perché deve poter offrire qualità ed affidabilità uguale o superiore a quella della rete PSTN.
Lato Service Provider sono presenti anche dei Session Border Controller per il "peering SBC", ovvero lo scambio di traffico telefonico tra diversi Service Provider direttamente tra le loro reti IP, in modo da non doverlo tradurre in TDM per scambiarselo.

Anche lato Enterprise il componente principale che si occupa della sicurezza della rete e della gestione delle sessioni, quindi che si comporta da intermediario, è costituito da un Session Border Controller. In questo caso l'aspetto principale del SBC deve essere la stretta integrazione di esso con l'architettura di rete pre-esistente, in modo da offrire un'esperienza di elevata qualità all'utente del servizio.
Anche per quanto riguarda la rete Enterprise, è necessaria la disponibilità di un'infrastruttura di rete IP di buona qualità.

Esistono diverse possibilità per connettere la rete di un cliente a quella di un Service Provider tramite l'utilizzo di un SIP Trunk; di seguito sono illustrate le due estreme strategie di dispiegamento secondo due modelli contrapposti:

- Modello centralizzato: tutte le chiamate originate all'interno della rete telefonica aziendale vengono instradate verso un unico sito centralizzato tramite una WAN (Wide Area Network) e vengono servite per mezzo di un unico SIP Trunk, condiviso da tutti i siti aziendali.
- Modello distribuito: le chiamate provenienti dai diversi siti aziendali sono gestite da SIP Trunk dedicati, uno per ogni sito.

Adottando la soluzione centralizzata è possibile abbattere i costi operazionali (OPEX) in quanto è necessario un solo SIP Trunk, attivo su un unico collegamento fisico. L'aspetto negativo di questo approccio è che spesso è necessario stravolgere l'architettura telefonica pre-esistente.

La soluzione distribuita prevede che ogni sito aziendale possieda una terminazione dedicata sul trunk, permettendo una migliore pianificazione delle risorse. Dal punto di vista delle modifiche architetturali, quest'ultimo è un approccio molto più conservativo di quello precedente, al prezzo di un minore risparmio sui costi operazionali.

Solitamente la soluzione migliore è un approccio intermedio tra i due modelli.

2.2 Lo standard SIP

Il Session Initiation Protocol (SIP) è un protocollo di segnalazione standardizzato da IETF nella RFC3261 [4]. E' principalmente impiegato per il controllo di sessioni di chiamate voce e video su protocollo IP. Il suo scopo è la creazione, la modifica e il rilascio di sessioni di comunicazione tra due o più entità.

Il SIP è un protocollo di livello applicativo e "text-based", incorpora molti elementi di http (Hypertext Transfer Protocol) e di SNMP (Simple Mail Transfer Protocol) ed è studiato appositamente per essere indipendente dal protocollo di trasporto sottostante, perciò può essere impiegato sia con UDP (User Datagram Protocol) che con TCP (Transmission Control Protocol).

E' stato sviluppato inizialmente da Henning Schulzrinne e Mark Handley nel 1996; nel 2000 è stato scelto come protocollo di segnalazione nell'architettura IP Multimedia Subsystem (IMS) e l'attuale versione, contenuta nella RFC3261, è del 2002.

E’ basato su una struttura richiesta/risposta in cui ogni transazione consiste in una richiesta da parte di un’entità client verso un’entità server e nella relativa risposta. Le entità previste dal protocollo sono diverse, la principale è costituita dagli User Agent, che possono comportarsi da client (UAC) quando inviano una richiesta, o da server (UAS) quando forniscono una risposta. Sono identificate da una SIP URI (Uniform Resource Identifier) simile a quelle che identificano indirizzi e-mail. Inoltre sono previste entità quali i proxy server e i registrar server.

Le richieste vengono anche chiamate “metodi” e ciascuna corrisponde ad una richiesta di informazioni o di un servizio da parte del UAC mittente. Tra le principali richieste si trova il messaggio “INVTE”, utilizzato per stabilire una sessione tra due User Agent, il “BYE”, per terminarne una già attiva, il “CANCEL”, l’”ACK” ed altre ancora.

Le risposte SIP sono identificate da un numero a tre cifre, di cui il primo indica la tipologia di risposta. Le risposte provisional sono identificate dal codice 1XX, quelle che indicano successo della richiesta con 2XX, errori dovuti al client sono indicati con 4XX mentre errori server o globali con 5XX e 6XX.

2.3 SIPconnect

SIPconnect Technical Recommendation [5] è un documento redatto da SIP Forum; esso raccoglie un insieme di Best Practice per l’interfacciamento di un’implementazione PBX aziendale con un ITSP. Questo lavoro è nato dal bisogno di eliminare incertezze ed incompatibilità riscontrabili tra i diversi e numerosi documenti di standardizzazione IETF a riguardo; Infatti nonostante le ITU-T Recommendation e le IETF RFC offrano un insieme completo di direttive finalizzate alla connettività diretta tra reti SIP-Enabled, i produttori di dispositivi non hanno uno standard principale di riferimento che definisca con chiarezza quali direttive debbano necessariamente essere supportate.

Tale documento:

- Specifica un insieme minimo di standard IETF e ITU-T che devono essere supportati da tutte le implementazioni.
- Fornisce precise linee guida in aree nelle quali gli standard permettono molteplici possibilità d’implementazione.

- Specifica un insieme minimo di capacità che dovrebbero essere supportate dalla rete sia lato Service Provider che lato Enterprise.

In questo documento si trovano quindi le direttive, seppur generiche, per la registrazione, la privacy, l'instaurazione e la terminazione delle chiamate. Inoltre vi sono disposizioni aggiuntive in merito alla gestione di servizi avanzati d'interworking come la voicemail e il trasferimento di chiamata.

L'architettura di riferimento per l'interconnessione IP tra una rete aziendale ed una di un Service Provider (figura 2), entrambe SIP-Enabled, presenta due diversi collegamenti:

- Il primo è destinato al trasporto della segnalazione SIP, ed è costituito dall'interfacciamento tra un SIP-PBX aziendale ed un Service Provider SIP Signaling Entity (SP-SSE).
- Il secondo è dedicato al trasporto dei pacchetti RTP e RTCP tra i Media Endpoint del SP e della rete aziendale. Lato aziendale il Media Endpoint potrebbe essere parte del SIP-PBX, oppure un dispositivo d'utente come un telefono SIP. Lato Service Provider il Media Endpoint potrebbe essere ad esempio un Gateway PSTN, un Media Server o un altro dispositivo endpoint IP-based d'utente.

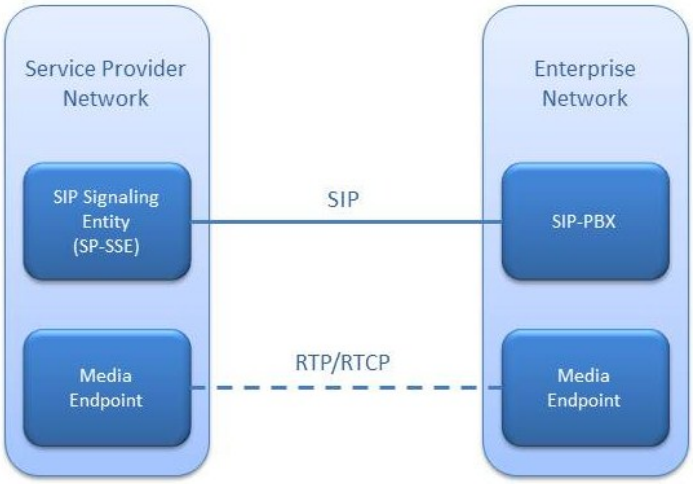


Figura 2 - Architettura di riferimento SIPconnect

L'architettura appena illustrata è da considerarsi generale, e nonostante tali funzioni siano sempre logicamente distinte, un costruttore può realizzare prodotti che le

integrano in soluzioni uniche, ad esempio incorporando SIP-PBX e Media Endpoint, oppure un singolo SIP-PBX può servire diversi Media Endpoint distribuiti geograficamente in diversi siti.

Il documento definisce quindi un modello d'architettura in cui il Service Provider è in grado di offrire servizi di SIP Trunking alla rete aziendale, permettendo la comunicazione tra gli utenti interni alla compagnia ed esterni ad essa.

All'interno di questo documento è possibile individuare alcune principali aree sulle quali si focalizza l'analisi svolta dal SIP Forum:

- I metodi di funzionamento: il Registration Mode, in cui il SIP-PBX si registra presso la rete SP, godendo di una facilità di installazione definibile "plug-and-play"; lo Static Mode, in cui le due reti sono considerate di pari livello ed è necessaria una "pre-configurazione" degli elementi di bordo.
- I protocolli di segnalazione e trasporto supportati: TCP, oltre ad UDP, è obbligatorio per supportare TLS (Transport Layer Security).
- L'instaurazione di chiamate base bidirezionali: vengono specificate modalità di riempimento dei campi di richieste e risposte SIP.
- L'inoltro di chiamata: è possibile implementarlo in due modalità, sfruttando rispettivamente la risposta "Moved Temporaly" o la richiesta "INVITE".
- Il trasferimento di chiamata: anche in questo caso è possibile utilizzare la richiesta "INVITE", oppure "REFER".
- I servizi d'emergenza: il SIP-PBX deve prevedere una differente gestione delle sessioni d'emergenza.
- I servizi di voicemail: vi è la possibilità di localizzazione del servizio sia presso la rete aziendale che presso il Service Provider.
- I limiti delle sessioni: il SP-SSE può prevedere limiti configurabili sul numero di sessioni provenienti o destinate al SIP-PBX. Limiti previsti dall'accordo commerciale tra azienda e SP.
- Le interazioni livello media delle sessioni: deve essere utilizzato il Session Description Protocol (SDP) [6] per la negoziazione dei media, che deve seguire un predefinito modello di negoziazione [7].

2.4 SIPp

SIPp [8], software open source sviluppato da Hewlett Packard, è sia un performance testing tool, che un generatore di traffico di segnalazione SIP.

E' strutturato secondo un'architettura client/server, pur essendo un unico software. All'avvio è necessario specificare se si desidera eseguire il programma in modalità client o in modalità server, a seconda se si vuole generare chiamate, client, o rispondere a chiamate in arrivo, server.

Al suo interno possiede alcuni scenari di chiamata di default, come uac.xml e uas.xml, attivabili da riga di comando all'avvio del programma specificando quale di questi eseguire. Attivando SIPp in modalità uac (user agent client), è necessario specificare l'indirizzo IP di destinazione delle chiamate che esso genererà, e il rate di chiamate espresso in cps (call per second); il programma creerà quindi tante chiamate al secondo quante specificate, verso l'indirizzo IP specificato, secondo il call flow di segnalazione specificato nel file di default uac.xml e riportato nella figura 3 sottostante. Il programma avviato genererà le SIP requests e attenderà le SIP responses.

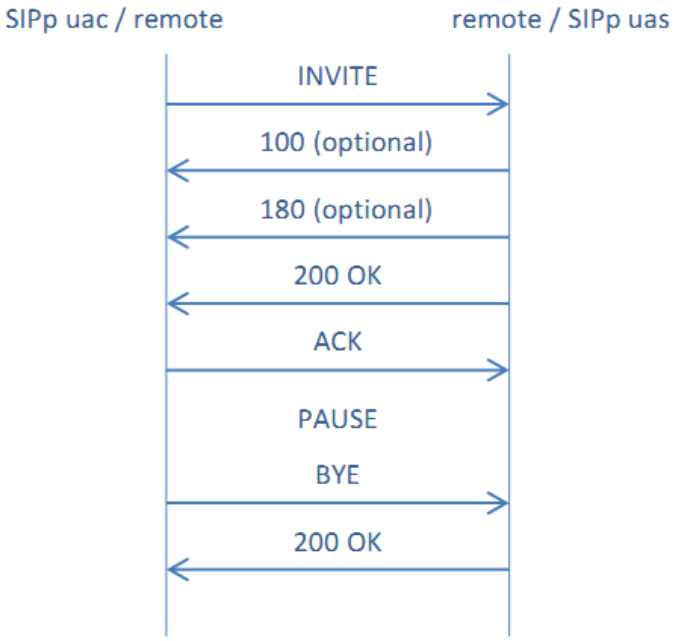


Figura 3 - Scenario di default per SIPp sia in modalità client che server

Lo stesso call flow appena illustrato può essere eseguito in modalità server specificando l'opzione uas (user agent server) all'avvio di SIPp; in questo modo il programma risponderà con SIP responses alle SIP requests in arrivo.

La finestra di dialogo di SIPp avviato in modalità client si presenta come in figura 4.

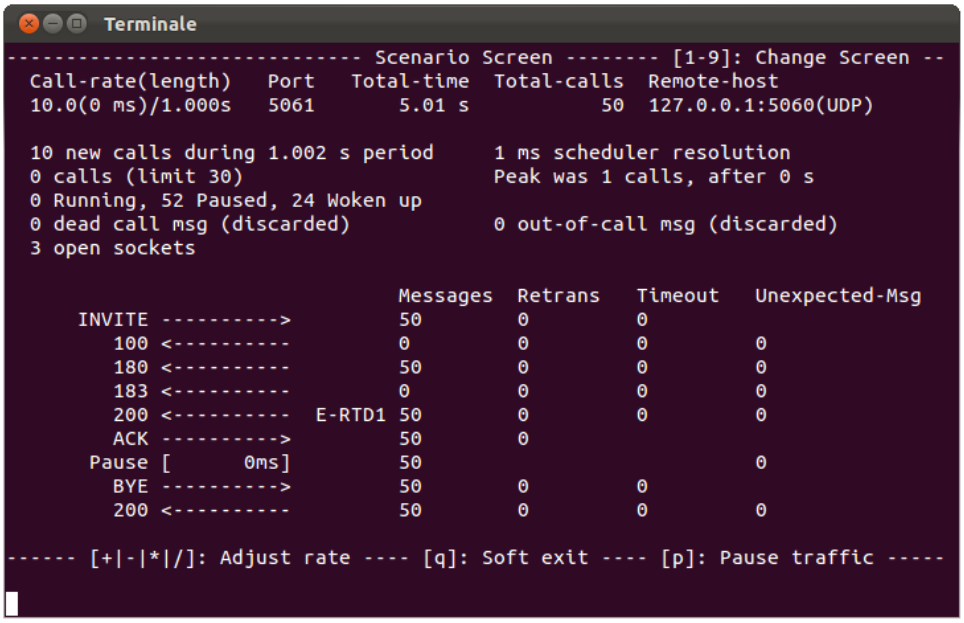


Figura 4 - Esecuzione dello scenario standard SIPp uac.xml

E' possibile far eseguire, sia in modalità client che server, degli scenari di chiamata creati ad hoc, anziché gli scenari di default. Per far questo è necessario costruire un file xml che descriva, attraverso le istruzioni e la sintassi specificate nella guida del tool, il call flow che si vuole realizzare. All'avvio del programma, da riga di comando, è necessario indicare il nome del file xml, completo di percorso, relativo allo scenario di chiamata che si vuole eseguire.

Il programma permette la raccolta di statistiche di test durante la sua esecuzione, riguardanti ad esempio il tempo di risposta ad una chiamata, il numero di chiamate fallite e di errori riscontrati. Inoltre è possibile impostare il salvataggio periodico delle statistiche raccolte in un file di report in formato csv.

Il trasporto della segnalazione può essere effettuato per mezzo di TCP o UDP. Nel primo caso è possibile attivare, previa inclusione delle apposite librerie, il trasporto sicuro TLS (Transport Layer Security), mentre nel secondo caso è possibile attivare il "retransmission management" dei messaggi.

Negli scenari xml creati ad hoc, è possibile inserire all'interno di alcune istruzioni, delle regular expression (regexp), attraverso le quali si può ad esempio controllare l'esattezza di alcuni campi dei messaggi ricevuti, oppure estrarre informazioni da un messaggio per riutilizzarle in seguito.

Se il software è avviato in modalità client, è possibile variare il rate delle chiamate generate, parametro specificato all'inizio da riga di comando, durante l'esecuzione del programma, sia che stia eseguendo uno scenario di default che uno personalizzato. Questo è possibile attraverso i comandi "+" ,"-" ,"*" e "/", aumentando o diminuendo di un unità il rate rispettivamente con il "+" o il "-", mentre lo si aumenta o diminuisce di dieci unità rispettivamente con il "*" o il "/".

Durante il funzionamento, se il programma riceve un messaggio che non rispetta il protocollo SIP, lo ignora, mentre se si tratta di un messaggio SIP inaspettato, ovvero non atteso da alcuna chiamata in corso, risponde con un appropriato messaggio. Se tale messaggio inatteso non può essere correlato ad alcuna chiamata attiva, viene inviato un messaggio "BYE". Se invece il messaggio è comunque relativo ad una chiamata attiva, anche se inaspettato, SIPp risponde con un "CANCEL" se la chiamata non ha ancora ricevuto risposta tramite un "200 OK" su "INVITE", altrimenti risponde con un "BYE". In entrambi i casi la chiamata viene terminata e marcata come "fallita". Se il messaggio inaspettato è una SIP response di tipo "4XX" o "5XX", SIPp risponde con un "ACK" e termina la chiamata, anche in questo caso marcandola come "fallita".

Se si utilizza UDP come protocollo di trasporto per la segnalazione, è possibile utilizzare il "retransmission management" di SIPp, attivabile singolarmente sui messaggi da trasmettere indicati nel file xml che descrive lo scenario. Esso implementa il meccanismo di ritrasmissione dei messaggi definito dallo standard SIP nella RFC3261 [4].

2.5 Anomaly Detection Systems

L'Anomaly Detection, ovvero la distinzione tra eventi o comportamenti normali o anomali di un sistema, è un argomento che tocca numerosi ambiti scientifici e non solo. Si tratta di un aspetto importante in medicina, in economia, in automazione e non da ultimo nell'informatica. E' un tema che ha applicazioni eterogenee, dalla

diagnosi di malattie, al controllo dell'andamento dei mercati finanziari fino ad aspetti di sicurezza in sistemi automatici ed informatici.

In ambito informatico, un sistema può sperimentare eventi o comportamenti anomali di varia natura. In una rete di calcolatori, le anomalie sono riconducibili sostanzialmente ad azioni dolose, ovvero attacchi, o a non meno pericolosi eventi involontari come guasti di apparati o collegamenti.
Sistemi volti all'individuazione del primo tipo di anomalie sono solitamente chiamati Intrusion Detection System (IDS).

Lo scopo degli Anomaly Detection System (ADS) è appunto la distinzione tra comportamenti normali ed anomali del sistema che esso monitora. Per fare ciò è possibile seguire due diverse strategie d'implementazione; in un caso si definisce e si caratterizza il comportamento normale del sistema, rilevando come anomalo qualunque comportamento che si presenta in modo diverso dalla caratterizzazione effettuata; nell'altro caso si procede in modo opposto, ovvero si caratterizza l'anomalia, o le anomalie, ritenendo normale qualunque altro comportamento non riconducibile ad essa. Questi due possibili approcci sono conosciuti in letteratura rispettivamente con i nomi "anomaly-based" o "behaviour-based" e con "signature-based" o "misuse-based" [9].

Nell'individuazione di una specifica anomalia, o di un insieme ristretto di queste, con un ADS implementato secondo un metodo "signature-based" è certamente possibile raggiungere prestazioni migliori rispetto ad uno di tipo "anomaly-based". Tuttavia con tale metodo non è possibile rilevare anomalie diverse da quelle per cui l'ADS è preposto, né eventi o comportamenti nuovi, ossia non è possibile il "novelty detection", mentre con un modello "anomaly-based" questo è realizzabile.

La scelta del tipo di approccio dipende in gran parte dallo scopo che si vuole raggiungere con l'applicazione dell'ADS al sistema, nonché dalle informazioni disponibili riguardo il funzionamento normale ed anomalo di quest'ultimo.

Durante il funzionamento, l'ADS decide se il comportamento del sistema che sta osservando è riconducibile ad una situazione normale od anomala, sulla base di come è stato addestrato o di parametri impostati. E' possibile incorrere in due diversi errori:

- Falsi positivi: con questo termine viene indicato l'errore in cui il sistema di monitoraggio segnala un evento o comportamento come anomalo quando in realtà esso è normale, ovvero quando viene segnalata la presenza di un'anomalia dove questa non risulta essere.
- Falsi negativi: con il termine falso negativo si indica l'errore opposto, ovvero la classificazione di un comportamento come normale quando in realtà è anomalo, ovvero la mancata segnalazione di un'anomalia quando questa è presente.

Il secondo tipo di errore è certamente il più critico per un ADS, infatti verrebbe meno al suo scopo. E' solitamente preferibile evitare di incorrere in questo tipo di errore, anche al prezzo di qualche falso allarme, ovvero falso positivo, in più. Tuttavia anche i falsi positivi sono dannosi per l'ADS, in quanto se frequentemente ricorrenti portano ad ignorarne l'output, rendendo inutile la sua applicazione al sistema.

Da un'analisi dei lavori presenti in letteratura riguardanti l'Anomaly Detection, si possono identificare diverse metodologie implementative, ovvero diversi modi per distinguere il corretto funzionamento del sistema da un'anomalia [10]. Si possono individuare:

- ADS "Classification Based".
- ADS basati sul Nearest Neighbor.
- ADS basati sul Clustering.
- ADS basati su analisi statistica.
- ADS basati sulla teoria dell'informazione.
- ADS basati su analisi spettrale.

2.5.1 ADS "Classification Based"

Gli ADS basati sulla classificazione prevedono l'apprendimento di un modello da un set di dati "etichettati", per poi classificare i dati di test a seconda del modello stesso. Sono quindi previste due fasi di funzionamento dell'ADS: la fase di training e quella di test.

A seconda della tipologia di dati a disposizione per la fase di training si possono utilizzare due differenti tipi di classificatori:

- “Multi-class” se i dati a disposizione appartengono a più classi “normali”, in questo caso viene segnalata un’anomalia se un campione di test non appartiene ad alcuna di queste.
- “One-Class” se i dati di training di corretto funzionamento appartengono tutti ad una classe. In questo caso un campione di test che non appartiene alla classe precedentemente individuata viene considerato anomalo.

Nella figura 5 sottostante è riportato un esempio grafico delle due tipologie di classificatori.

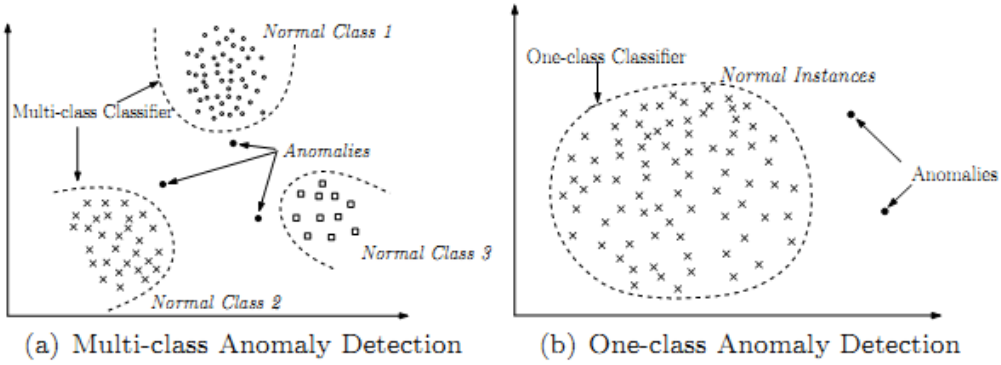


Figura 5 - Esempi di classificatori multi-classe e one-class

Le principali tecniche per implementare un ADS basato sulla classificazione sono:

- Reti Neurali, come i “multi layered perceptrons” per classificazione multi-classe, o il “Replicator Neural Network” per la one-class [11]. Quest’ultimo possiede lo stesso numero di neuroni in input ed output ed è addestrato a scomporre i dati in ingresso e ricomporli in uscita; in fase di test la differenza tra i dati ricostruiti e quelli in input fornisce un tasso d’anomalia del campione analizzato.
- Reti Bayesiane, usate esclusivamente per il caso multi classe.
- Support Vector Machines (SVM), utilizzate come one-class SVM. In fase di addestramento viene costruita una regione complessa che racchiude gran parte dei dati di training, separandoli dalla restante percentuale. La percentuale di elementi da non includere nella regione è un parametro di training. In fase di test un campione è definito anomalo se si trova all’esterno di questa regione.

- Tecniche basate su applicazione di regole, definiscono delle regole che descrivono il normale funzionamento del sistema monitorato, ovvero del training set. Se un campione di test non soddisfa alcuna regola è considerato anomalo. Gli alberi di decisione sono un esempio di questo tipo di tecniche.

Esistono ADS che combinano queste tecniche [12], ad esempio utilizzando una SVM per individuare un fault, ed un blocco Rule-Based Reasoning (RBR) per classificarlo.

La complessità computazionale di questi ADS dipende dalla specifica tecnica che essi implementano. Nonostante le Support Vector Machine siano lente in fase di training, la fase di test è sempre molto rapida.

I principali vantaggi di questo tipo di ADS sono la velocità di decisione in fase di test e la potenza degli algoritmi di separazione dei dati che implementano. Gli svantaggi sono la necessità di un buon numero di campioni per classe, nel caso di classificatori multi-classe, e il fatto che la decisione è di tipo "hard", ovvero quasi nessuna tecnica di questo tipo è in grado di fornire un tasso d'anomalia del campione di test.

2.5.2 ADS basati sul Nearest Neighbor

Gli ADS basati su questa tecnica seguono il principio per cui un dato di corretto funzionamento del sistema è situato in una regione con denso "vicinato", mentre un dato relativo ad un'anomalia si trova lontano anche dai più vicini ad esso. Questa tecnica richiede la definizione di una funzione di "distanza" o di dissimilarità tra due campioni dati; solitamente è utilizzata la distanza Euclidea per grandezze continue.

Le tecniche di Nearest Neighbor implementate per l'Anomaly Detection sono riconducibili a due tipi:

- Tecniche che considerano la distanza tra il campione in esame ed il suo k-esimo vicinato al fine di calcolarne il tasso d'anomalia.
- Tecniche che considerano la densità del vicinato del campione in esame per calcolarne il tasso d'anomalia.

Nel primo caso il tasso d'anomalia di un campione è calcolato sulla base della funzione distanza rispetto ai k campioni più vicini a quello considerato. In realtà esistono diverse soluzioni simili per il calcolo del tasso d'anomalia, come ad esempio il numero di

campioni la cui funzione distanza rispetto al campione in esame è inferiore ad un valore determinato. Inoltre esistono diverse tecniche per aumentare l'efficienza del metodo base appena descritto, ad esempio tecniche di "pruning".

Nel secondo caso viene calcolata la densità del vicinato del campione in esame, definendolo anomalo nel caso in cui questa sia bassa, mentre viene dichiarato normale se il suo vicinato, ovvero la zona circostante la sua posizione, è ad alta densità. Questa tecnica risulta poco performante nel caso in cui le regioni di corretto funzionamento abbiano densità differenti (vedi figura 6 sottostante); in questo caso anomalie che rispetto ad una zona densa sono distanti tanto quanto sono distanti i campioni di un'altra zona di corretto funzionamento a minor densità, non saranno rilevate.

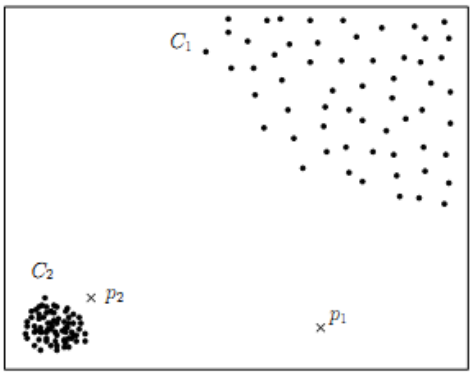


Figura 6 - Esempio di dataset con zone di corretto funzionamento a diversa densità

Per ovviare a questo problema sono state sviluppate diverse tecniche di "densità relativa", ovvero tecniche che considerano la densità del vicinato di un campione relativamente alla posizione del campione stesso.

La complessità computazionale di questo tipo di tecniche è solitamente più elevata delle altre.

I vantaggi di questa tipologia di ADS sono la possibilità di funzionamento sia in modo non supervisionato che semi-supervisionato e la non specificità dell'implementazione rispetto alle grandezze da monitorare.

Per contro ha la possibilità di incappare in un alto numero di falsi negativi nel caso in cui gli eventi anomali siano simili, creando così una zona anomala ad alta densità, non rilevabile da questo tipo di tecnica. Inoltre soffre la complessità computazionale anche in fase di test, rendendone difficile l'utilizzo in applicazioni real-time. Per ultimo la

definizione di una soglia oltre la quale una distanza è da considerarsi anomala spesso non è cosa banale.

2.5.3 ADS basati sul Clustering

Questa tipologia di ADS si basa su algoritmi di clustering. Questi sono nati per definire all'interno di un dataset dei gruppi di dati "simili" tra loro e solo in un secondo momento sono stati applicati all'Anomaly Detection. Si possono distinguere tre differenti approcci per questo tipo di tecnica.

Il primo approccio consiste nel considerare campioni di corretto funzionamento quelli appartenenti ad un cluster, mentre campioni anomali quelli che non appartengono ad alcun cluster. Questo approccio è percorribile solo con l'utilizzo di algoritmi di clustering che non associano forzatamente tutti i campioni a dei cluster.

Nel secondo approccio vengono considerati corretti i campioni più vicini al centro del cluster cui appartengono, mentre sono anomali quelli più vicini al bordo di esso. In questo caso la distanza dal centro del cluster di appartenenza definisce un tasso d'anomalia del campione. Diversi algoritmi permettono l'utilizzo di questo metodo, come l'uso delle Self-Organizing Maps o del K-means.

Infine un possibile approccio prevede la classificazione di un campione come corretto se appartiene al cluster di dimensione e densità maggiore, come anomalo se appartenente ad uno degli altri cluster "minori".

La differenza tra il Clustering e il Nearest Neighbor è che nel primo il campione in esame è valutato rispetto all'intero cluster di appartenenza, nel secondo rispetto al solo vicinato.

Come per le tecniche di classificazione, anche in questo caso la fase di costruzione dei cluster è ad elevata complessità computazionale, mentre la fase di test è rapida in quanto la decisione sul campione in esame è presa rispetto ad un modello già costruito.

Il vantaggio di questa tipologia di ADS, oltre alla già menzionata rapidità di test, è la possibilità di inserire nel modello un cluster aggiuntivo a posteriori. I principali svantaggi, oltre alla elevata complessità computazionale per la costruzione del

modello, sono imputabili alle limitazioni dovute agli algoritmi, come la forzatura nell'associazione di ogni campione ad un cluster.

2.5.4 ADS basati su analisi statistica

Gli ADS che rientrano in questa categoria si basano sul principio per cui un'anomalia è costituita da un'osservazione che non può essere ricondotta al modello stocastico costruito per descrivere il sistema da monitorare. In questo modo il campione in esame viene classificato corretto se si trova in una regione ad alta probabilità di occorrenza per il modello stocastico, anomalo se in una regione a bassa probabilità.

Anche in questo caso l'ADS necessita di una fase preliminare in cui viene costruito il modello stocastico del sistema, ovvero viene calcolato un modello statistico che descrive al meglio i dati a disposizione riguardo il corretto funzionamento del sistema. In seguito, in fase di test, viene deciso se il campione in esame appartiene o meno al modello precedentemente costruito.

ADS di questo tipo possono basarsi su due differenti tecniche di analisi statistica:

- Tecniche parametriche.
- Tecniche non parametriche.

Tecniche parametriche assumono che i dati di corretto funzionamento del sistema sono generati da distribuzioni parametriche, perciò in una prima fase ne determinano i parametri, per poi ottenere un tasso d'anomalia di un campione test attraverso l'inverso della funzione densità di probabilità della distribuzione ottenuta. Un particolare tipo di tecnica parametrica prevede la costruzione di un modello autoregressivo del sistema, calcolando il tasso d'errore come il discostamento del dato osservato da quello atteso, calcolato secondo tale modello.

Tecniche non parametriche non prevedono la costruzione di un modello a priori, ma ne determinano alcune caratteristiche statistiche durante il funzionamento, ovvero stabiliscono se un campione in esame è coerente o meno con la distribuzione dei campioni osservati in precedenza, decidendo così se esso è rispettivamente normale o anomalo. Tipici esempi di tecniche di questo tipo prevedono l'applicazione di funzioni

kernel come il “parzen windows estimation”, oppure la costruzione di istogrammi per tenere traccia della frequenza delle occorrenze del processo in esame.

I vantaggi di questa tipologia di ADS sono dovuti alla possibilità di operare in modo non supervisionato, se dotati di una fase di definizione del modello robusta, e alla disponibilità di un intervallo di confidenza associato al tasso d’anomalia di ogni campione esaminato.

Tali tecniche si trovano in difficoltà nei casi in cui i dati ricavati dall’osservazione del sistema non seguono una semplice distribuzione statistica, rendendone complessa la modellizzazione. Inoltre la maggior parte di queste tecniche esamina individualmente ogni parametro che costituisce il campione in esame, senza tenere conto di una visione d’insieme.

2.5.5 ADS basati sulla teoria dell’informazione

Questi ADS si basano sul calcolo di grandezze quali l’entropia, l’entropia relativa o la “Kolmogorov Complexity”. Il principio alla base di questi ADS è l’assunzione che un anomalia introduce una irregolarità nell’informazione contenuta nei dati.

I principali vantaggi di questo tipo di tecniche sono la possibilità di lavorare in modalità non supervisionata ed il fatto che non sono necessarie assunzioni a priori sui dati da monitorare, invece necessarie per le tecniche di analisi statistica. Il principale svantaggio di questa tecnica è la possibile difficoltà di costruzione di strutture dati sulle quali calcolare le grandezze da monitorare.

2.5.6 ADS basati su analisi spettrale

ADS basati su tecniche di analisi spettrale, cercano una rappresentazione alternativa dei dati attraverso una combinazione dei loro attributi, al fine di mettere in evidenza la variabilità di questi. Il principio è quindi quello di riportare i dati in uno spazio differente in cui campioni normali e campioni anomali risultano maggiormente separabili.

La maggior parte degli ADS di questo tipo si basano sulla Principal Component Analysis (PCA), ma anche tecniche come Compact Matrix Decomposition (CMD) e la Singular Value Decomposition (SVD) sono frequentemente utilizzate.

I vantaggi di queste tecniche sono costituiti principalmente dalla riduzione delle dimensioni dello spazio delle features, rendendole particolarmente indicate per dataset con un elevato numero di dimensioni, e possono essere applicate unitamente ad altre tecniche come quelle viste in precedenza. Tuttavia trovare uno spazio di dimensione inferiore in cui i dati risultano separabili non è sempre possibile, e spesso comporta complessità computazionali elevate.

2.5.7 Individuazione di anomalie "collettive"

Esistono alcune tecniche che permettono di individuare anomalie composte da particolari occorrenze di campioni che se considerati singolarmente non sono necessariamente anomali.

Tra le più diffuse vi è l'applicazione della trasformata wavelet [13][14] a sequenze temporali di campioni, al fine di estrarre pattern da tale sequenza e ricondurre il problema di anomalia "collettiva" ad un più semplice problema di anomalia "singola"; in questo modo tale problema può essere affrontato attraverso l'applicazione di una delle tecniche illustrate in precedenza. Questo metodo è largamente utilizzato in molti campi [15][16][17][18].

2.5.8 ADS in sistemi con traffico a pacchetto

Nel campo specifico dell'individuazione di eventi anomali in sistemi informatici con traffico a pacchetto, è possibile introdurre ulteriori distinzioni tra le diverse tecniche di ADS presenti in letteratura.

In primo luogo gli ADS sono classificabili in base alla tipologia di analisi del sistema adottata [19]:

- Analisi del flusso dei dati.
- Analisi protocollare.
- Analisi a livello applicativo.

Nel primo caso vengono estratte dal traffico informazioni relative al flusso dei dati quali il numero di bytes, di pacchetti o di connessioni osservate in un determinato intervallo temporale. Nel secondo caso, invece, vengono analizzati parametri specifici

del protocollo utilizzato. L'analisi a livello applicativo prevede la conoscenza a priori della natura delle informazioni contenute nel traffico in analisi.
Inoltre si possono distinguere in base al metodo di costruzione del modello del sistema da monitorare, a seconda che sia ottenuto dall'analisi diretta del sistema stesso o per mezzo di specifiche fornite esternamente all'ADS. Nel primo caso si parla di "learnt model", e può essere ottenuto mediante una fase d'addestramento, supervisionato o non, precedente all'utilizzo dell'ADS; nel secondi caso il modello ottenuto è detto "specification-based" ed è costruito sulla base di informazioni fornite esternamente da personale esperto del sistema da monitorare.
Un'ulteriore distinzione è relativa alla scala di analisi, sia temporale che di dettaglio dei dati. Un ADS può infatti operare in "microscala" se campiona con frequenza i parametri che monitora, oppure se questi ultimi sono relativi ad informazioni di basso livello. Un ADS che analizza parametri relativi a lunghi intervalli di osservazione, nell'ordine delle ore, o che analizza informazioni di alto livello, opera in "macroscala".

Esistono numerosi ADS che monitorano le informazioni contenute nelle Management Information Base (MIB) degli apparati appartenenti al sistema in oggetto [20][21]. In questo caso il traffico è monitorato indirettamente, ovvero non vengono estratte informazioni dal traffico di rete ma queste sono derivate dalle statistiche prodotte dai dispositivi durante l'esercizio. Questo tipo di approccio offre buoni risultati nell'individuazione di errori server e di guasti hardware degli apparati.
Nel caso particolare di traffico voce, oltre alle informazioni presenti nelle MIB degli apparati vengono spesso considerati i "cartellini" delle telefonate, ovvero i Call Detail Records (CDR) e gli Internet Protocol Detail Records (IPDR) in caso di traffico VoIP [22]. Per ogni telefonata viene generato un cartellino contenente informazioni quali la durata della conversazione e la tipologia di apparecchi telefonici in uso.

I parametri estratti dagli ADS che monitorano il traffico in modo diretto appartengono solitamente a due tipologie:

- Parametri volumetrici.
- Parametri entropici.

Nel secondo caso sono solitamente estratte informazioni circa la distribuzione del traffico in rete, calcolando parametri come l'entropia di Shannon o di Tsallis degli indirizzi IP e delle porte sorgente e destinazione dei pacchetti. Lavori recenti evidenziano le ottime prestazioni ottenibili sulla base di questo tipo di metriche [23], soprattutto nell'individuazione di attacchi Denial of Service (DoS) e Port Scan. Questa tipologia di metrica è particolarmente indicata per il monitoraggio di reti di larga scala.

In ambito volumetrico sono numerosi i parametri estraibili dal traffico in rete. E' possibile implementare un semplice conteggio di byte osservati in un intervallo temporale, soluzione indipendente dalla tipologia di informazioni scambiante nel sistema, oppure estrarre parametri di più alto livello, tenendo conto del protocollo in uso o del contenuto informativo dei pacchetti. ADS destinati ad un uso in generiche reti basate su protocollo TCP/IP analizzano solitamente informazioni di più basso livello, distinguendo eventualmente il volume di traffico per flussi [24].

In ambito VoIP, e SIP in particolare, sono presenti numerosi lavori in ambito Anomaly Detection volti soprattutto all'individuazione di fenomeni intrusivi, come attacchi Denial of Service, o l'eliminazione del fastidioso fenomeno di "SPam over Internet Telephony" (SPIT).
In queste implementazioni è ricorrente l'uso di parametri volumetrici relativi a particolarità del protocollo SIP, come il conteggio dei messaggi tipici delle mimiche di registrazione, vedi il messaggio "REGISTER", per l'individuazione di attacchi DoS a Registrar Server [25]. E' diffuso anche l'uso di parametri relativi, come la percentuale di particolari metodi o status SIP rispetto al totale dei messaggi osservati [26].

2.6 Curve di ROC

Le curve di ROC, ovvero Receiver Operating Characteristic, inizialmente introdotte come strumento di analisi di immagini radar, sono uno strumento per la rappresentazione grafica delle prestazioni di un classificatore binario [27].

Attualmente sono molto utilizzate in ambito medico, nel "data mining" e nel "machine learning".

Si consideri un problema di predizione a due classi, in una sola dimensione, dove le distribuzioni delle occorrenze dei campioni di entrambe le classi risultano parzialmente

sovrapposte (vedi figura 7); sottoponendo un campione ad un classificatore binario, esso ne decide l'appartenenza ad una delle due classi, a seconda del valore del campione stesso rispetto ad una soglia, o cut-off, definita dal classificatore. Chiamando le due classi "normale" ed "anomala", se un campione appartenente alla classe "normale" viene classificato correttamente, si dice che è un "vero negativo" (true negative TN), se invece viene classificato "anomalo" si tratta di un "falso positivo" (false positive FP). Allo stesso modo se un campione appartenente alla classe "anomala" è classificato correttamente, si tratta di un "vero positivo" (true positive TP), altrimenti di un "falso negativo" (false negative FN). Nella tabella 1 è presente uno schema riassuntivo.

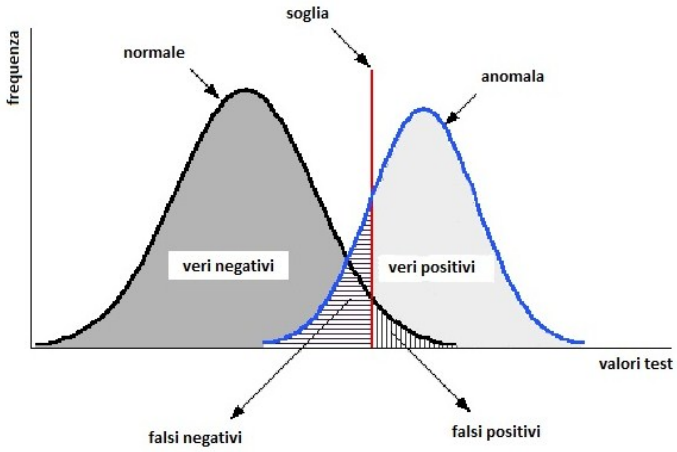


Figura 7 - Distribuzioni classi "normale" e "anomala" con soglia del classificatore e indicazione di veri e falsi

		Valore vero: Anomalo	Valore vero: Normale
Classificazione	Anomalo	Vero Positivo (TP)	Falso Positivo (FP)
	Normale	Falso Negativo (FN)	Vero Negativo (TN)

Tabella 1 - Tabella di contingenza per decisori binari

Se si sottopone al classificatore una serie di campioni, è possibile valutarne le prestazioni, ovvero la correttezza delle decisioni, calcolando due parametri: il "True

Positive Ratio" (TPR) e il "False Positive Ratio" (FPR), derivati direttamente dal numero di "True Positive", "True Negative", "False Positive" e "False Negative" ottenuti.

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

I due parametri così calcolati permettono di individuare un punto sul piano con in ascissa il FPR e ordinata il TPR. Variando il posizionamento della soglia del classificatore, è possibile costruire una curva, chiamata appunto "curva di ROC", che va dall'origine degli assi fino al punto di coordinate (1;1). L'origine degli assi rappresenta un classificatore che decide sempre per l'appartenenza del campione che gli viene sottoposto alla classe "normale", evitando di produrre falsi positivi, ma anche veri positivi. L'altro estremo, costituito dal punto (1;1), rappresenta un classificatore che si comporta nel modo opposto, ovvero che decide sempre per l'appartenenza del campione alla classe "anomala", classificando correttamente tutti i campioni realmente anomali, ma senza ottenere alcun vero negativo.

La curva che, passando per i due punti precedentemente presentati, taglia a 45° il piano, corrisponde alla curva prodotta da un classificatore casuale (figura 8"a").
Un classificatore ideale, invece, produrrebbe una curva "spezzata", che unisce in verticale l'origine con il punto di coordinata (0;1), ovvero FPR=0 e TPR=1, e questo punto con quello di coordinate (1;1) (figura 8"b").
Normalmente la curva prodotta da un classificatore si posiziona tra i due casi limite precedentemente descritti (figura 8"c").

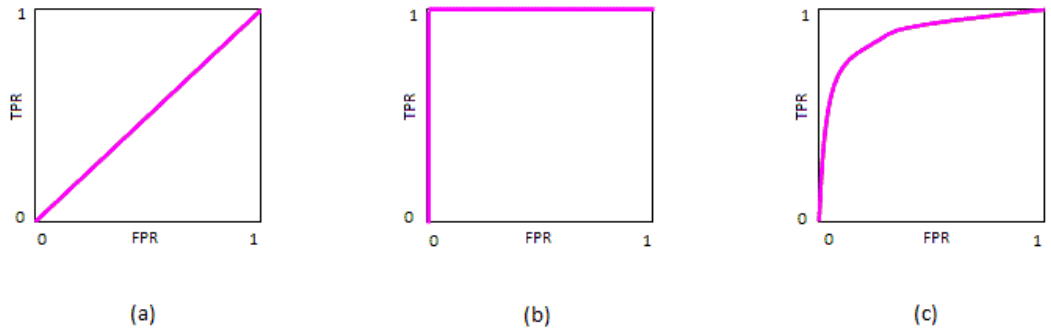


Figura 8 - Curve di ROC

Un indice di prestazione molto diffuso, in quanto sintetico ed immediato, è l'area sottesa dalla curva di ROC, "Area Under Curve" (AUC), e può valere da un minimo di 0,5 nel caso del classificatore casuale, ad un massimo di 1 per il classificatore ideale.

3. PROBLEMATICA E SOLUZIONI PROPOSTE

In questo capitolo è illustrata una problematica relativa all'architettura SIP Trunk e vengono presentati tre possibili approcci a tale problema.

3.1 Problema da affrontare

Lo scenario di riferimento per il lavoro presentato nei capitoli successivi è un SIP Trunk che interconnette la rete telefonica privata di un'azienda e la rete di un Service Provider, attraverso una connessione IP tra i rispettivi apparati di bordo rete, definiti in precedenza Session Border Controller. Su questo collegamento transita sia il traffico di segnalazione (SIP) che il traffico voce (RTP). Più in generale si può pensare lo scenario come l'interconnessione tra le reti VoIP di due soggetti, come mostrato in figura 9.

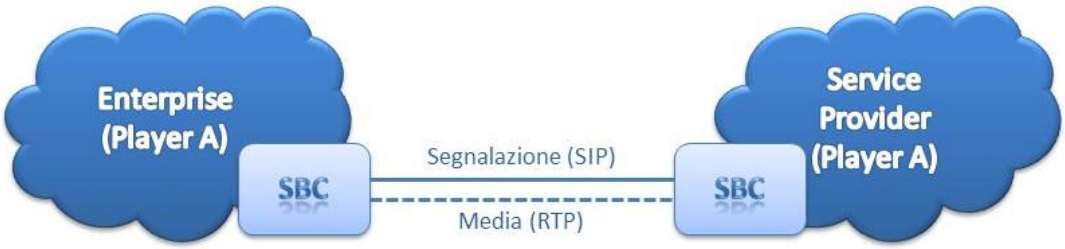


Figura 9 - Schema generale dello scenario di riferimento

Come illustrato nel capitolo precedente, è sempre maggiore la tendenza per i grossi clienti telefonici definiti "Large Enterprise" di migrare la propria architettura telefonica verso la tecnologia VoIP, in particolare SIP, sfruttando un'implementazione SIP Trunk come architettura di accesso alla Rete Telefonica Pubblica. Questo avviene principalmente per ragioni economiche, infatti tale soluzione d'accesso garantisce risparmi sia sui costi delle telefonate che sui costi d'acquisto delle linee telefoniche, concentrando in pochi punti geografici il traffico telefonico dell'intera rete aziendale.

Ad esempio, considerando una realtà aziendale costituita da 1000 sedi aventi 10 utenze telefoniche ciascuna, per un totale di 10000 utenti, l'interconnessione di ciascun sito aziendale alla Rete Telefonica Pubblica comporta l'acquisto di 4 linee tradizionali, per un totale di 4000 linee.

Il dimensionamento è infatti effettuato modellizzando il traffico offerto dagli utenti secondo una distribuzione binomiale, stimata una probabilità di chiamata da parte di

ciascun utente P_c = 0,08, ossia ciascun utente trascorre l'8% del suo tempo al telefono, ed una probabilità di blocco P_b = 0,001. Di seguito è riportata la probabilità di blocco per un singolo sito aziendale, ovvero la probabilità che *k* utenti desiderino telefonare contemporaneamente, con *k* minore del numero di utenze *N* per singolo sito e maggiore delle linee telefoniche *M* disponibili per il sito stesso.

$$P(k > M) = \sum_{k=M+1}^{N} \binom{N}{k} P_c{}^k (1 - P_c)^{N-k} < P_b$$

ovvero

$$P(k > 4) = \sum_{k=5}^{10} \binom{10}{k} 0{,}08^k (1 - 0{,}08)^{10-k} = 0{,}000544 < 0{,}001$$

Raccogliendo invece il traffico offerto da tutte le utenze aziendali in un unico sito e mantenendo i parametri stimati in precedenza, il numero di linee necessarie all'interconnessione alla Rete Telefonica Pubblica si riduce a 866, ottenuto mediante il modello Erlang-B, con volume di traffico offerto *A* = 800 Erlang = P_c**N*. E' quindi evidente la convenienza di una soluzione centralizzata.

$$P(k > M) = E_B(M, A) = \frac{A^M}{M!}\left(\sum_{i=0}^{M} \frac{A^i}{i!}\right)^{-1} = E_B(866, 800) = 0{,}000967 < 0{,}001$$

Una soluzione SIP Trunk per un'azienda "Large Enterprise" (figura 10) serve il traffico telefonico generato da un elevato numero di utenti, nell'ordine delle migliaia. Un'anomalia o un malfunzionamento del sistema d'accesso SIP Trunk può ripercuotersi su una quantità importante di utenze telefoniche; per questo motivo è necessario essere in grado di individuare tempestivamente situazioni anomale che portano ad un degrado della qualità del servizio offerto agli utenti del sistema.

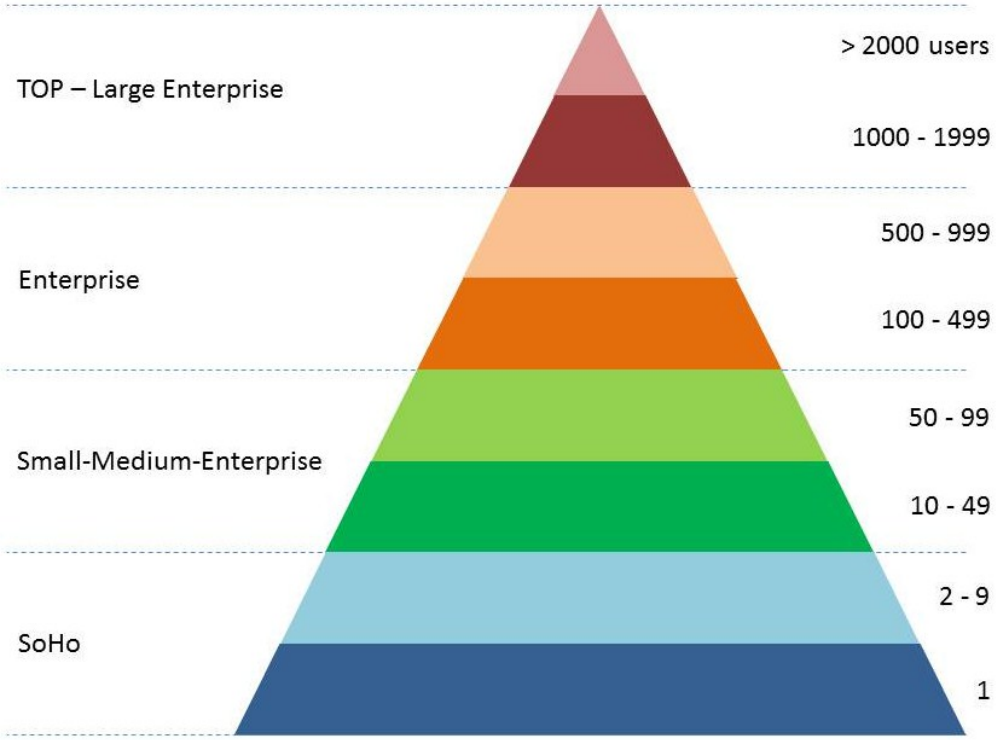


Figura 10 - Piramide di segmentazione del mercato telefonico business

Lo scenario di riferimento, illustrato in modo più dettagliato in figura 11, può essere soggetto a malfunzionamenti di varia natura, originati da attacchi, guasti di origine hardware o software, errori umani di gestione o malfunzionamenti di rete.

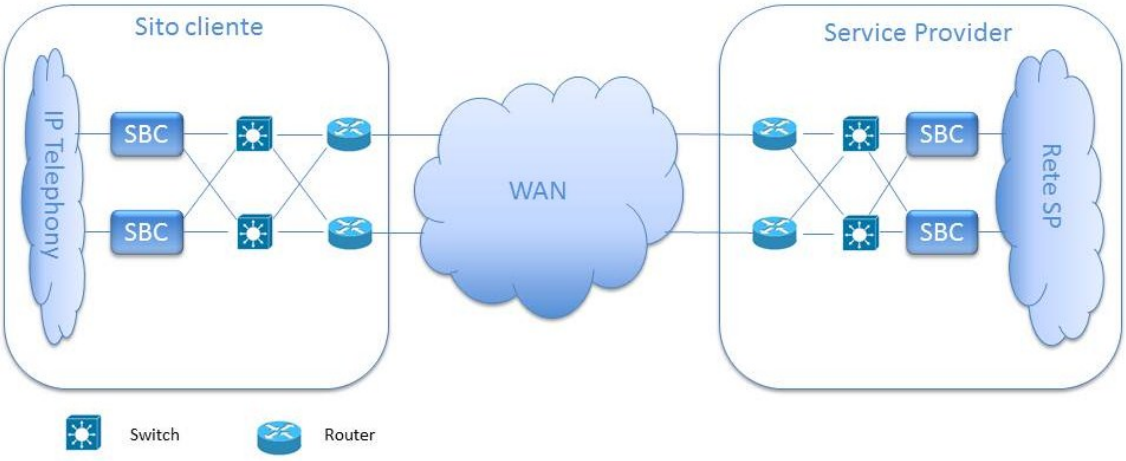


Figura 11 - Schema più dettagliato dello scenario di riferimento

I sistemi di monitoring attualmente disponibili in commercio si basano principalmente sulla raccolta di informazioni di autodiagnosi delle macchine dispiegate, attraverso due protocolli:

- Simple Network Management Protocol (SNMP).

- System Log (Syslog).

Sulla base di tali informazioni vengono derivati dei Key Performance Indicator (KPI) che forniscono una panoramica sullo stato di funzionamento del SIP Trunk.

L'impiego di queste tecniche, unitamente ad una architettura ridondata del sistema e priva di Single Point Of Failure (SPOF), garantisce un'elevata e tempestiva protezione da guasti di natura hardware degli apparati. Infatti ogni dispositivo è affiancato da un altro identico pronto a sostituirlo in caso di guasto; quest'ultimo solitamente è configurato in modo da eseguire le stesse operazioni svolte dal dispositivo primario, in modo da contenere esattamente le stesse informazioni ed essere pronto ad intervenire immediatamente. Questa modalità di funzionamento del dispositivo di backup è conosciuta con il nome "hot standby".

Tuttavia le informazioni di autodiagnosi che un dispositivo rende disponibili al sistema di monitoraggio, sono diverse a seconda delle scelte implementative del produttore del dispositivo; si rende perciò necessaria la realizzazione di sistemi dedicati.

Esistono inoltre altre tipologie di malfunzionamento che questi sistemi di monitoraggio non sono sempre in grado di rilevare e spesso i primi a segnalarne la presenza sono gli utenti stessi. Esempi di questi malfunzionamenti sono i guasti software, come la presenza di un processo malevolo che consuma le risorse a discapito del processo principale di gestione delle chiamate, oppure del codice errato che porta ad un malfunzionamento del dispositivo interessato; in quest'ultimo caso il dispositivo di backup non è in grado di sostituire il primario in quanto, eseguendo le stesse istruzioni simultaneamente, si trova anch'esso in errore. Un altro esempio è l'errata gestione dei dispositivi, come il disallineamento delle tabelle di routing; questo porta a loop di chiamate potenzialmente pericolosi per l'intero sistema SIP Trunk.

Avere un sistema di monitoring in grado di identificare la presenza di malfunzionamenti ed eventi anomali nel sistema, analizzando direttamente il traffico di segnalazione presente sul SIP Trunk, costituisce un valore aggiunto sia per il fornitore del servizio di SIP Trunking che per il cliente che ne usufruisce. Un sistema di questo tipo deve avere accesso diretto al traffico di segnalazione che transita sul trunk; per fare ciò, con riferimento alla figura 12 sottoriportata, si è ipotizzato di attestare

l'Anomaly Detection System su uno qualunque dei nodi di rete presenti tra il Session Border Controller del cliente ed i rispettivi del Service Provider, utilizzando porte di mirroring.

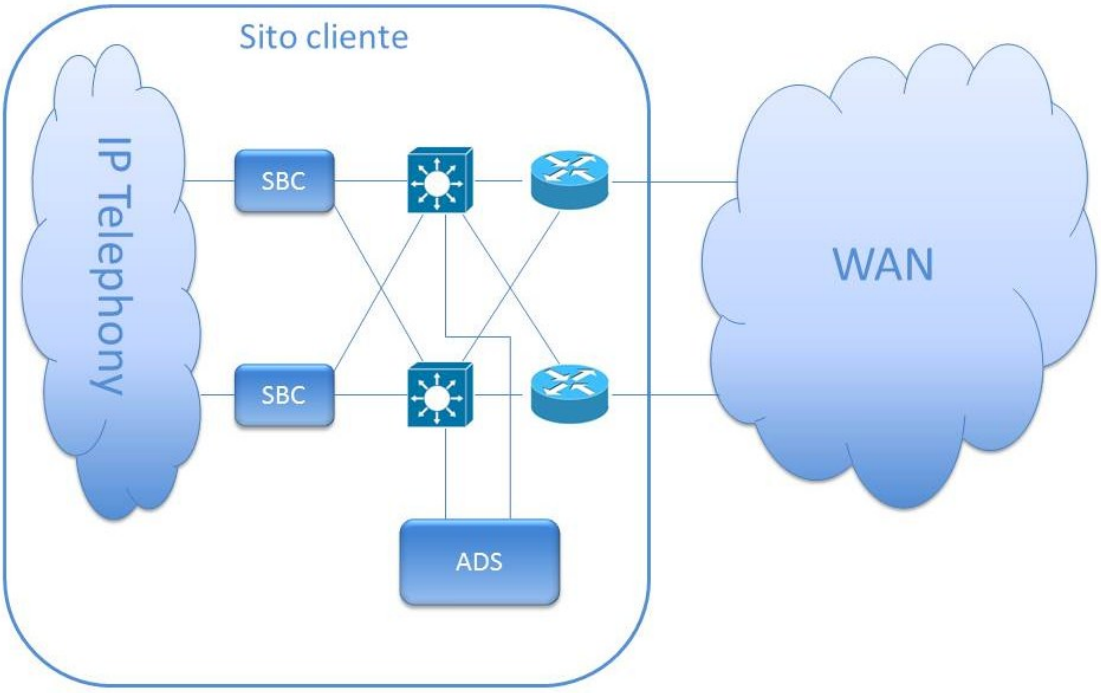


Figura 12 - Posizionamento del sistema di monitoring all'interno dello scenario di riferimento

Da un punto di vista operativo, i dispositivi presenti tra la rete telefonica aziendale ed il trunk possono essere gestiti sia dal Service Provider che direttamente dall'azienda stessa, o ancora possono essere affidati a terzi per una gestione in "outsourcing". Tale scelta deriva prettamente da logiche legate alla politica di "sourcing" dell'azienda, ma non influisce in alcun modo sull'implementazione dell'ADS.

L'approccio al problema della localizzazione di eventi anomali nel traffico di segnalazione presente sul trunk proposta in questo lavoro, non vuole essere orientato all'individuazione di una particolare tipologia di malfunzionamento del sistema. Tuttavia, a partire da un database di eventi anomali relativi a SIP Trunks messo a disposizione da ICT Consulting, si è deciso di valutare le prestazioni delle soluzioni proposte con riferimento a tre tipologie di anomalia, di particolare interesse per la frequenza con cui queste si verificano. Queste anomalie sono conosciute come "link-flap", "CPU-hog" e "loop".

3.2 Approccio

Dall'analisi della letteratura esistente riguardante problemi di "Anomaly Detection" si evince la distinzione tra due principali approcci, ovvero quello "signature based" e quello "anomaly based", solitamente basato su "apprendimento non supevisionato" [28]. Lo scopo di questo lavoro non è limitato al rilevamento e segnalazione di singoli eventi anomali ben caratterizzabili, ma potenzialmente di qualunque evento che porta il sistema a comportarsi in modo differente da quello che è il suo solito. Il secondo approccio, quello basato su "apprendimento non supervisionato", permette proprio di individuare qualunque discostamento del comportamento del sistema monitorato dal suo corretto funzionamento; per questo motivo la soluzione proposta è orientata proprio a quest'ultimo approccio.

Volendo monitorare il sistema SIP Trunk attraverso l'analisi del traffico di segnalazione SIP che transita su di esso, è necessario estrarre dal traffico stesso dei parametri che ben lo caratterizzano. Scegliendo i pattern da estrarre, è necessario tenere conto della complessità computazionale di estrazione, escludendo le soluzioni più dispendiose; questo perché la soluzione proposta vuole essere implementabile per il monitoraggio "real-time" del sistema.

Lavori piuttosto recenti evidenziano le buone prestazioni di Anomaly Detection System (ADS) per il monitoraggio di traffico di rete, basati sul calcolo dell'entropia degli indirizzi sorgente e destinazione dei pacchetti IP [23]. E' difficile riportare le buone prestazioni di questo principio nel sistema SIP Trunk perché il traffico di segnalazione presente su di esso è generato da una coppia si Session Border Controller, i quali agiscono da Back-to-Back User Agent (B2BUA), "nascondendo" tutte le informazioni di provenienza e destinazione dei messaggi, al di fuori del campo "user" delle SIP-URI. Questo campo è costituito dal numero di telefono dell'utente, ma l'estrazione di parametri entropici dal traffico di segnalazione basati su di esso produrrebbe pattern molto sensibili a fenomeni sociali. Infatti sul trunk è presente traffico da e verso le utenze telefoniche dell'azienda che acquista il servizio di SIP Trunking, perciò è prevedibile che l'entropia delle numerazioni che effettuano o ricevono telefonate in un determinato periodo di osservazione risenta delle attività che svolge l'azienda stessa. Ad esempio le numerazioni dell'ufficio contabile dell'azienda possono essere

notevolmente più attive nel periodo di pagamento degli stipendi rispetto agli altri giorni del mese.

In ambito SIP, è soluzione comune l'estrazione di parametri volumetrici dal traffico di segnalazione, perciò la ricerca di parametri per caratterizzare il comportamento del sistema SIP Trunk sarà orientata a soluzioni di questo tipo.

3.3 Estrazione delle informazioni da monitorare

In un recente articolo riguardante una proposta di Intrusion Detection System per reti VoIP SIP-based [26], sono presentate diverse possibili features di tipo statistico-volumetrico estraibili da traffico SIP. Queste sono divise in cinque gruppi, a seconda del tipo di grandezza estratta; è presente un insieme di parametri di tipo generico quali il numero di richieste SIP rispetto al totale di messaggi SIP osservati in un'unità di tempo, la quantità di messaggi con contenuto SDP (Session Description Protocol) e il tempo medio di interarrivo tra richieste. Un altri raggruppamenti comprendono grandezze calcolate in base alla distinzione dei messaggi per "Call-ID", quali il numero medio di messaggi per chiamata o calcolate in base al "final-state" della chiamata, come la quantità di chiamate andate in risposta e terminate con successo (COMPLETED) piuttosto che quelle ancora attive (IN-CALL). Infine due gruppi di parametri comprendenti rispettivamente i rate delle principali richieste SIP e delle principali risposte.

Tra questi, i parametri che più si adattano all'implementazione per lo scopo di questo lavoro sono gli ultimi due raggruppamenti proposti, in primo luogo per la semplicità di estrazione, caratteristica importante ai fini del funzionamento in real-time dell' ADS, in secondo luogo perché forniscono una rappresentazione immediata dell'andamento volumetrico del traffico SIP.

Siccome il sistema da monitorare non è una generica rete come nel caso dell'articolo precedentemente citato, ma si tratta di un SIP Trunk, perciò di un singolo link, è utile calcolare i rate dei messaggi distinguendoli non solo per tipologia, ma anche per verso di percorrenza.

La durata dell'intervallo di osservazione T, è un parametro dell'ADS da scegliere in modo empirico, tipicamente nell'ordine di qualche secondo, in modo da poter fornire una valutazione sullo stato del trunk in tempi brevi.

L'estrazione dei parametri che caratterizzano il traffico presente sul trunk è quindi effettuata per mezzo di un blocco di elaborazione, riportato in figura 13, il quale ha in ingresso la traccia di traffico di segnalazione SIP.

In uscita, ogni "slot" temporale t di durata T secondi, tale blocco produce in uscita M valori $r_m(t)$, dove m individua uno degli M parametri da monitorare, distinti per tipo di messaggio SIP, ad esempio "INVITE" o "200 OK", e per verso di percorrenza del trunk, dal Provider all'Enterprise o viceversa; $r_m(t)$ è quindi il rapporto tra le occorrenze del messaggio m-esimo e il totale dei messaggi SIP osservati nello slot t-esimo.

$$r_m(t) = \frac{\text{messaggi di tipo } m \text{ osservati nello slot } t}{\text{totale messaggi osservati nello slot } t}$$

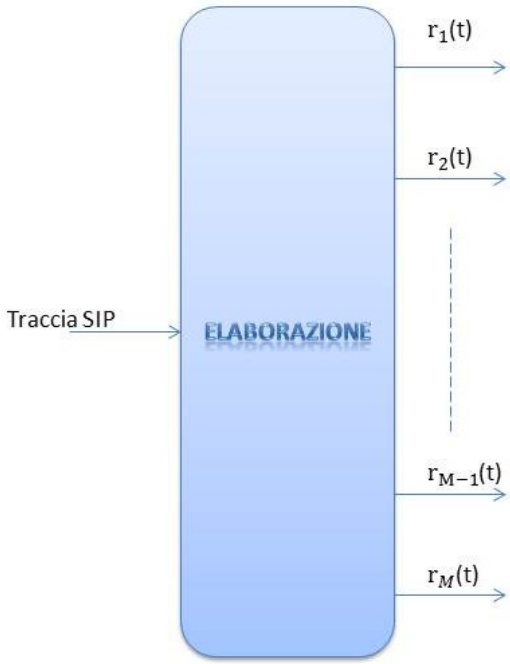


Figura 13 - Blocco di estrazione parametri dalla traccia di traffico SIP

3.4 Classificatore a soglia

Come già anticipato all'inizio di questo capitolo, l'approccio di questo lavoro al tema dell'Anomaly Detection vuole essere "Anomaly Based", ovvero il rilevamento di discostamenti del segnale monitorato dai valori che esso assume in situazioni di corretto funzionamento. Nel paragrafo precedente è stato individuato l'insieme delle serie temporali r_m come segnale caratterizzante l'andamento del traffico di segnalazione SIP presente sul trunk, e pertanto da monitorare. Volendo quindi controllare, slot per slot, l'andamento di questi parametri durante il funzionamento del sistema SIP Trunk, la soluzione concettualmente più semplice è quella di definire, per ogni parametro *m*, un intervallo di valori all'interno del quale $r_m(t)$ in situazione di corretto funzionamento, è solito collocarsi.

Per fare ciò, si introduce un blocco per ogni successione r_m, che riceve in ingresso ad ogni slot *t* il valore $r_m(t)$ e controlla che esso sia compreso tra una soglia massima $THup_m$ e una minima $THdown_m$. In caso affermativo produrrà in uscita $d_m(t)=1$, valore che esprime la "normalità" del parametro *m*-esimo nello slot temporale *t*, in caso contrario segnalerà il valore "anomalo" del parametro ponendo in uscita $d_m(t)=0$. Tale blocco è riportato nella figura 14 sottostante.

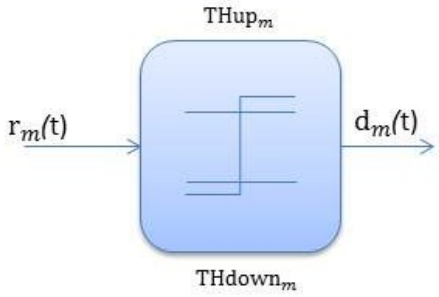


Figura 14 - Blocco di "thresholding"

A valle di questi blocchi di sogliatura, si pone un blocco di decisione, illustrato in figura 15, che fonde le informazioni in uscita da ognuno di essi. Questo blocco non fa altro che controllare, per ogni slot *t*, i valori $d_m(t)$ prodotti dai blocchi che lo precedono, decidendo di segnalare un evento anomalo ponendo la sua uscita al valore negativo $D(t)=-1$ se almeno uno di essi è nullo, ovvero se almeno uno dei parametri estratti dalla traccia di traffico SIP si discosta da quello che è il suo regolare valore; invece nel caso in cui tutti i parametri rientrano negli intervalli predefiniti, l'uscita del blocco finale di

decisione è posta al valore positivo D(*t*)=1, esprimendo coerenza tra i valori assunti dai parametri nello slot di osservazione e l'intervallo di corretto funzionamento predefinito.

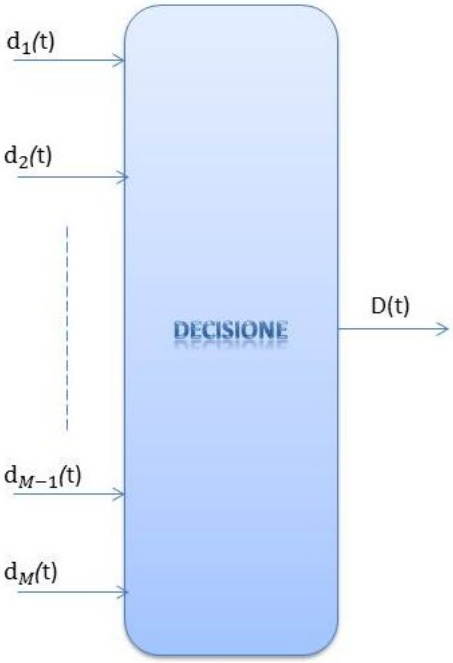


Figura 15 - Blocco di decisione

Definito lo schema a blocchi completo, riportato in seguito, è necessario stabilire il metodo più opportuno per definire le soglie $THup_m$ e $THdown_m$ che individuano l'intervallo di corretto funzionamento per ogni parametro *m* monitorato. A questo scopo la soluzione più intuitiva è quella di elaborare, attraverso il metodo presentato al paragrafo "estrazione features", una traccia di traffico SIP in una finestra temporale in cui il sistema SIP Trunk funziona correttamente. Così facendo si ottengono *m* serie temporali r_m, lunghe tanti campioni quanti sono gli slot temporali in cui è possibile suddividere la finestra di osservazione del traffico. E' quindi possibile porre le soglie $THup_m$ e $THdown_m$ rispettivamente al valore massimo e minimo assunto dalla *m*-esima serie temporale, ovvero $THup_m=Max_m=\max(r_m)$ e $THdown_m=Min_m=\min(r_m)$, dove $\max(r_m)$ e $\min(r_m)$ sono rispettivamente il valore massimo e minimo della successione r_m. In questo modo, in fase di monitoring, viene segnalato un evento anomalo solo nel caso in cui almeno uno dei parametri considerati superi il valore massimo osservato nella relativa serie temporale di riferimento, oppure nel caso sia inferiore al valore minimo della stessa serie.

Da prove sperimentali si riscontra che questo modo per calcolare le soglie dà origine ad un ADS inefficiente, infatti non vengono individuate gran parte delle situazioni anomale ad esso sottoposte, certamente a causa di un'eccessiva ampiezza dell'intervallo così definito. Osservando una traccia di corretto funzionamento è possibile che, seppur con frequenza molto bassa, il valore dei parametri estratti dia origine a dei picchi, in positivo o in negativo, rispetto a quella che è la normale oscillazione delle serie temporali dei parametri stessi. Questi picchi sono noti in letteratura con il nome di "outliers". E' necessario quindi tenere conto di questo fatto nella definizione delle soglie dell'ADS, calcolandole opportunamente in modo da escludere gli outliers dall'intervallo di corretto funzionamento.

A questo scopo, si è implementato un algoritmo che "stringe" progressivamente l'intervallo di corretto funzionamento di ciascun parametro, sottraendo e aggiungendo rispettivamente alla soglia superiore e a quella inferiore dell'intervallo stesso, una determinata quantità. Tale quantità è individuata, per ogni parametro *m*, come la millesima parte dell'ampiezza iniziale dell'intervallo, ovvero lo 0,1% della differenza tra il valore massimo e minimo assunti dalla serie temporale *m*-esima. Il numero di volte che viene eseguito il "restringimento" dell'intervallo di corretto funzionamento è un parametro dell'ADS; esso è strettamente legato alla percentuale di outliers che si desidera escludere dall'intervallo stesso.

L'ADS implementato lavora "offline", ovvero su tracce di traffico SIP catturate in precedenza, in formato ".pcap", calcolando le soglie di ogni parametro con un algoritmo descritto dallo pseudocodice seguente:

1. *Calcolo delle serie temporali P[m] attraverso il blocco di elaborazione*
2. *Individuazione per ogni parametro m (tipo messaggio - verso di percorrenza)* Max_m *e* Min_m
3. *Definizione soglie iniziali:* $THup_m = Max_m$ *e* $THdown_m = Min_m$*, calcolo:* $Step_m = (Max_m - Min_m)/1000$
4. *Processing della traccia stessa con l'ADS completo, individuando gli slot temporali "anomali", ovvero gli outliers (al primo passaggio si ottiene certamente lo 0% di outliers)*
5. *Restringimento dell'intervallo di corretto funzionamento di ogni parametro,* $THup_m = THup_m - Step_m$ *e* $THdown_m = THdown_m + Step_m$
6. *Ripetizione dei punti 4 e 5 per x volte, con* $x \leq 500$ *(THup=THdown)*

Una volta impostate le soglie dell'ADS con i valori calcolati grazie all'algoritmo sopra descritto, è possibile processare le tracce da monitorare tramite il sistema esposto in precedenza di cui riporto lo schema a blocchi completo in figura 16.

Per ogni traccia di traffico in ingresso all'ADS ottengo in uscita una serie temporale di ±1, lunga tanti valori quanti sono gli slot che è possibile individuare nella finestra temporale in cui è catturata la traccia stessa; tali valori rappresentano le decisioni prese dall'ADS slot per slot, gli slot ritenuti anomali sono segnalati con un -1, quelli ritenuti coerenti con la traccia di riferimento con un +1.

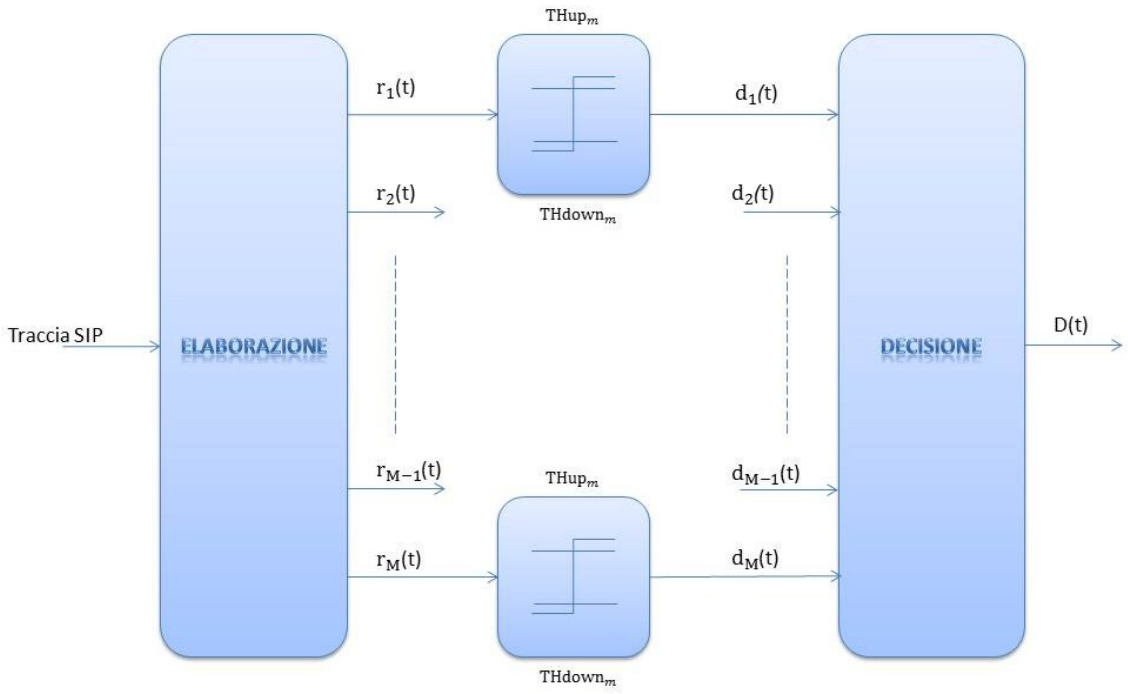


Figura 16 - Schema a blocchi completo

L'algoritmo di individuazione delle soglie e l'ADS completo sono realizzati con programmi dedicati in linguaggio C, sfruttando la libreria "libpcap" per la lettura delle tracce di traffico.

Concettualmente, l'utilizzo di questa tecnica equivale alla definizione di un parallelepipedo all'interno di un iperspazio di M dimensioni; infatti le decisioni vengono prese separatamente parametro per parametro, quindi dimensione per dimensione in modo indipendente. Ad ogni slot temporale viene calcolato il "punto di lavoro" del sistema all'interno dell'iperspazio M-dimensionale, e viene segnalata un'anomalia se tale punto è collocato all'esterno del parallelepipedo.

E' facile intuire che la superficie di separazione tra lo spazio di corretto funzionamento e quello in cui si considera il traffico come anomalo non corrisponde alla superficie ottima. Inoltre l'algoritmo di calcolo delle soglie così definito ha il limite di "stringere" contemporaneamente gli intervalli di tutti i parametri.

3.5 Classificatore Support Vector Machine

Per ovviare al problema esposto nel paragrafo precedente, è possibile utilizzare una soluzione molto diffusa e consolidata per quanto riguarda problemi di "one-class", una variante particolare di Support Vector Machine (SVM), la ν-SVM (nu-SVM), conosciuta anche con il nome di "One-Class SVM" proprio per la tipologia di problemi per cui essa è utilizzata.

Questa SVM è la tecnica più diffusa per ADS "anomaly-based", e rientra nella categoria delle macchine ad addestramento non supervisionato.

Come riportato in appendice, una SVM permette di individuare la superficie ottima di separazione tra campioni appartenenti a classi distinte, riportando i dati, per mezzo di una funzione kernel, in uno spazio di dimensione maggiore in cui essi risultano linearmente separabili. La variante one-class prevede la separazione ottima tra una predefinita percentuale di campioni del dataset di addestramento, definiti outliers, rispetto alla restante parte. La percentuale di outliers che si desidera escludere dalla zona in cui i campioni non vengono ritenuti anomali è un parametro d'ingresso della macchina, chiamato parametro ν (nu); da qui il nome ν-SVM.

I parametri di input della ν-SVM sono gli stessi utilizzati nella soluzione precedente e descritti nel paragrafo "estrazione features", ovvero i rate dei differenti messaggi SIP distinti per verso di percorrenza del trunk, rapportati al totale dei messaggi SIP osservati nella medesima unità di tempo. Per sfruttare al meglio le potenzialità della SVM, è necessario applicare il cosiddetto "scaling" dai parametri, ovvero riportare tutti i parametri in una scala di valori tra -1 e +1. Grazie a questa procedura è possibile evitare che un parametro che assume valori di ordine di grandezza superiore ai valori assunti da un altro parametro risulti dominante per la decisione della macchina. Per applicare quindi lo scaling, nel toolbox LibSVM [29] è presente un programma chiamato svm-scale, il quale riporta il dataset in ingresso al range di valori desiderato

(default ±1), salvando in un file a parte le trasformazioni eseguite, in modo da poter riapplicare la stessa trasformazione ai dati in input in fase di "test".

Come funzione kernel della ν-SVM è preferibile scegliere una Radial Basis Function (RBF), nota anche con il nome di Gaussian Kernel, in quanto largamente diffusa. Questa particolare funzione viene definita da due parametri, C e γ (gamma); il primo definisce il costo di errore di classificazione, ed esprime il trade-off tra un classificatore semplice ma con scarsa accuratezza ed uno molto accurato ma anche molto complesso; il secondo definisce l'ampiezza della funzione kernel gaussiana. La scelta di questi parametri è molto importante ai fini di evitare "underfitting" o "overfitting" del dataset di addestramento. Sempre nel toolbox LibSVM, è disponibile un programma chiamato svm-grid, il quale esegue una ricerca dei parametri C e γ più appropriati per lo specifico problema in esame. E' necessario inserire in ingresso a tale programma l'intervallo di valori in cui si vogliono cercare C e γ oltre ad un valore *n*, alla percentuale di outliers ν e al dataset di addestramento. Il tool esegue, per ogni possibile coppia di valori di C e γ all'interno degli intervalli specificati, un addestramento della SVM, testandone le performance per mezzo di una *n*-fold cross-validation e restituendo quindi la coppia di valori C e γ che ha evidenziato le prestazioni migliori.

Ottenuti i parametri dalla grid-search, è possibile addestrare la ν-SVM con il tool di LibSVM chiamato svm-train, fornendo in ingresso al programma i valori di C e γ ottenuti e lo stesso valore ν fornito in ingresso in precedenza al programma svm-grid, oltre ovviamente al dataset di addestramento, composto dai parametri estratti dalla traccia e scalati con svm-scale. In questo modo ottengo il blocco di decisione dell'ADS, riportato in figura 17, costituito dalla SVM così addestrata.

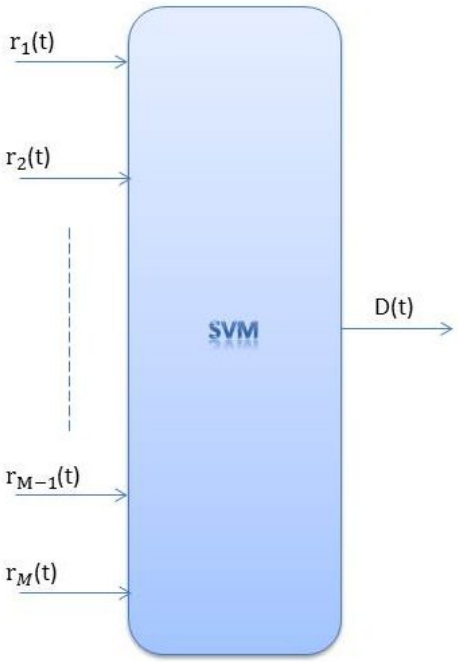


Figura 17 - Blocco di decisione composto dalla SVM

In fase di test va inserito in ingresso al blocco di decisione il vettore di parametri relativo allo slot di cui si vuole decidere la coerenza o meno con il dataset di addestramento della SVM. All'interno del blocco gli elementi del vettore vengono scalati, secondo i parametri utilizzati per lo scalamento del dataset di addestramento, e forniti in ingresso alla SVM precedentemente addestrata. Essa, per ogni vettore corrispondete ad uno slot temporale *t*, produce in uscita un valore D(*t*)=±1, positivo nel caso in cui il vettore appartenga alla regione di corretto funzionamento individuata in fase di training, negativo in caso contrario, ritenendo quindi lo slot "anomalo".

Riporto di seguito, in figura 18, lo schema a blocchi completo dell'ADS realizzato. Il blocco di estrazione dei pattern da tracce formato ".pcap" è realizzato mediante un programma dedicato in codice C, come nel caso precedente, mentre il blocco di decisione e la fase di grid-search e addestramento della SVM sono implementati mediante uno script bash che chiama al suo interno i tool di LibSVM.

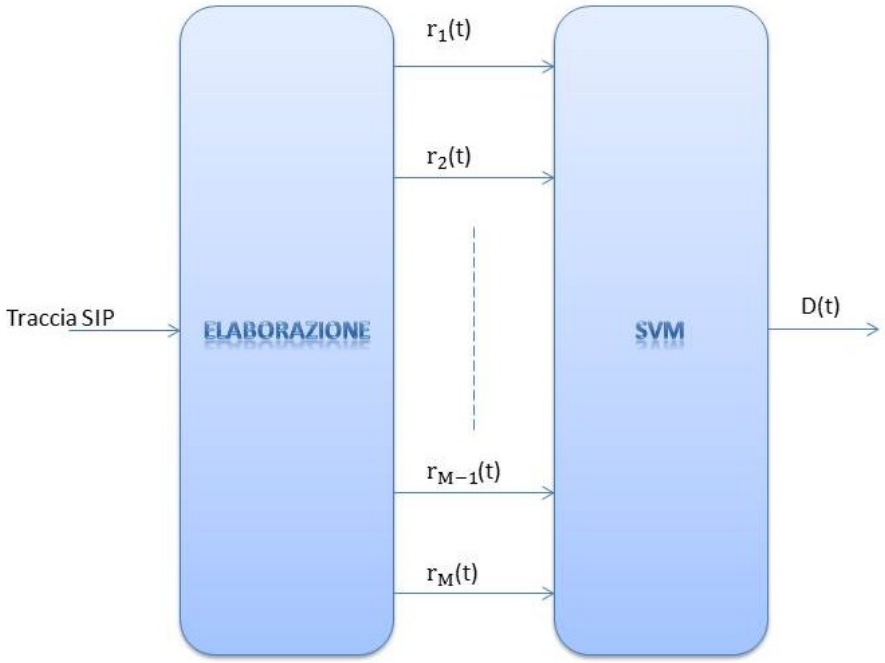


Figura 18 - Schema a blocchi completo

Utilizzando questo metodo basato su SVM, rispetto all'algoritmo a soglie illustrato in precedenza, è possibile eliminare il vincolo "lineare" sulle soglie di decisione, permettendo così la definizione, nell'iperspazio M-dimensionale dei parametri, di superfici di separazione anche complesse.

Il principale limite di questa tecnica, come di quella illustrata in precedenza, è la limitatezza della decisione che, essendo relativa all'osservazione di un solo slot, non tiene conto in alcun modo dell'andamento dei parametri nel tempo.

3.6 Trasformata Wavelet e classificatore Support Vector Machine

Ai fini di considerare l'andamento temporale dei parametri che caratterizzano il traffico SIP presente sul trunk, è possibile considerare, anziché un solo slot temporale, un vettore composto dai più recenti *k* slot temporali consecutivi. Per fare questo è sufficiente porre un buffer, riportato in figura 19, per ogni parametro *m*-esimo di uscita del blocco di elaborazione, il quale riceve in ingresso il valore del parametro *m*-esimo $r_m(t)$ del più recente slot temporale *t*, e pone in uscita $r_m(t, t\text{-}1, .. , t\text{-}k+1)$, ovvero la serie temporale degli ultimi *k* valori assunti dal parametro *m*-esimo.

Figura 19 - Buffer

Si ottiene quindi, ad ogni istante di campionamento, una serie temporale per ogni parametro estratto; tali serie descrivono l'andamento del traffico sul SIP Trunk in una finestra temporale lunga *k* volte il tempo di campionamento. E' necessario estrarre delle informazioni riassuntive in grado di descrivere sinteticamente tale andamento dei parametri.

In letteratura è diffuso l'impiego della Wavelet Decomposition ai fini di analizzare le componenti spettrali di un segnale o di una serie temporale [16]. Nel contesto in esame, è possibile utilizzare la versione discreta di tale trasformata, la cui struttura è riportata in appendice, impiegando come "Mother Wavelet" la semplice versione discreta di Haar a due campioni. Il blocco riportato nella figura 20 sottostante è realizzato sfruttando le funzioni della libreria gsl (GNU Scientific Library) per linguaggio C.

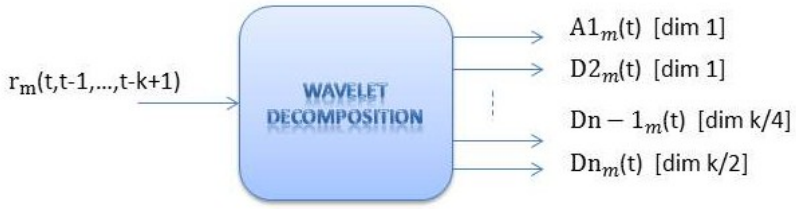


Figura 20 - Blocco che implementa la Wavelet Decomposition

Utilizzando questa trasformata è necessario vincolare la scelta del valore *k* ad una potenza di due. In uscita dal blocco wavelet si ottengono n = $\log_2(k)+1$ vettori di cui il primo, $A1_m(t)$, di dimensione unitaria, corrisponde all'approssimazione del vettore in ingresso, ovvero la componente continua della serie temporale, mentre l'ultimo, $Dn_m(t)$, di dimensione $k/2$, contiene i dettagli a frequenza più alta della serie stessa.
Nel caso in cui si sceglie $k = 8$ si ottengono n = 4 vettori in uscita, dove $A1_m(t)$ e $D2_m(t)$ sono composti da un unico elemento, il cui valore è rispettivamente la media degli 8 elementi del vettore in ingresso e la differenza tra la media dei primi 4 elementi e la media dei secondi 4. $D3_m(t)$ è composto da due elementi, di cui il primo è la differenza

tra le medie della prima e della seconda coppia di elementi del vettore in ingresso, il secondo la differenza tra le medie della terza e quarta coppia. Infine $D4_m(t)$ è composto da quattro elementi, ottenuti come differenza tra gli elementi del vettore d'ingresso.

Una volta applicata questa trasformazione, è possibile, come è solito in letteratura, estrarre informazioni quali l'energia del segnale in ogni sottobanda precedentemente definita [30], oppure media e varianza dei vettori ottenuti [16]. E' soluzione molto diffusa l'utilizzo dell'energia per ogni sotto-banda del segnale, calcolata come media del quadrato dei campioni della sotto-banda stessa. E' riportata di seguito la formula per il calcolo dell'energia E_{sb} di ogni sottobanda, dove N_{sb} è il numero di elementi che compongono il vettore relativo alla sottobanda considerata e d_x è l'elemento in posizione *x*-esima del vettore stesso.

$$E_{sb} = \frac{1}{N_{sb}} \sum_{x=1}^{N_{sb}} (d_x)^2$$

Si inserisce quindi, a valle di ogni blocco di trasformata wavelet, un ulteriore blocco, riportato in figura 21, che calcola questa informazione per ogni sottobanda in cui è suddivisa la serie temporale di ogni parametro.

Figura 21 - Blocco di estrazione medie-varianze o energie

Si ottiene così un nuovo set di parametri che non si limitano a descrivere solamente la situazione del traffico in un ristretto periodo di tempo come lo slot, fornendo un'informazione "istantanea", ma caratterizzano l'evolversi del traffico in una finestra temporale più ampia, seppur limitata; tale finestra, essendo derivata dalla somma di slot temporali dell'ordine dei secondi, sarà dell'ordine delle decine di secondi, conseguentemente alla scelta del parametro *k*.

Per definire la zona di corretto funzionamento del sistema, sulla base dei parametri appena descritti, è possibile mantenere la stessa soluzione proposta e adottata nell'ADS illustrato in precedenza, riportata in figura 22. Anche in questo caso, la fase di addestramento della ν-SVM consisterà nello scaling del dataset di addestramento, nella grid-search dei parametri della macchina e nell'addestramento vero e proprio di essa. L'unica differenza rispetto al caso precedente è la dimensione dei parametri in ingresso.

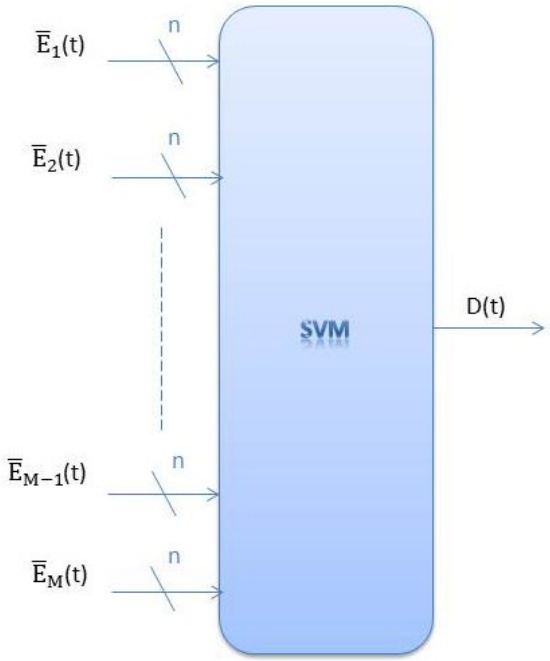


Figura 22 - Blocco di decisione SVM

Anche in questo caso, ad ogni campionamento, si ottiene in uscita un valore $D(t)=\pm 1$ corrispondente alla decisione presa dalla macchina rispetto al campione sottoposto; tale valore sarà positivo nel caso in cui il campione che descrive il traffico nella finestra temporale di osservazione è coerente con quelli appartenenti al dataset di addestramento, negativo in caso contrario, segnalando quindi la decisione di campione anomalo. Nella figura 23 sottostante è riportato lo schema a blocchi completo dell'ADS presentato.

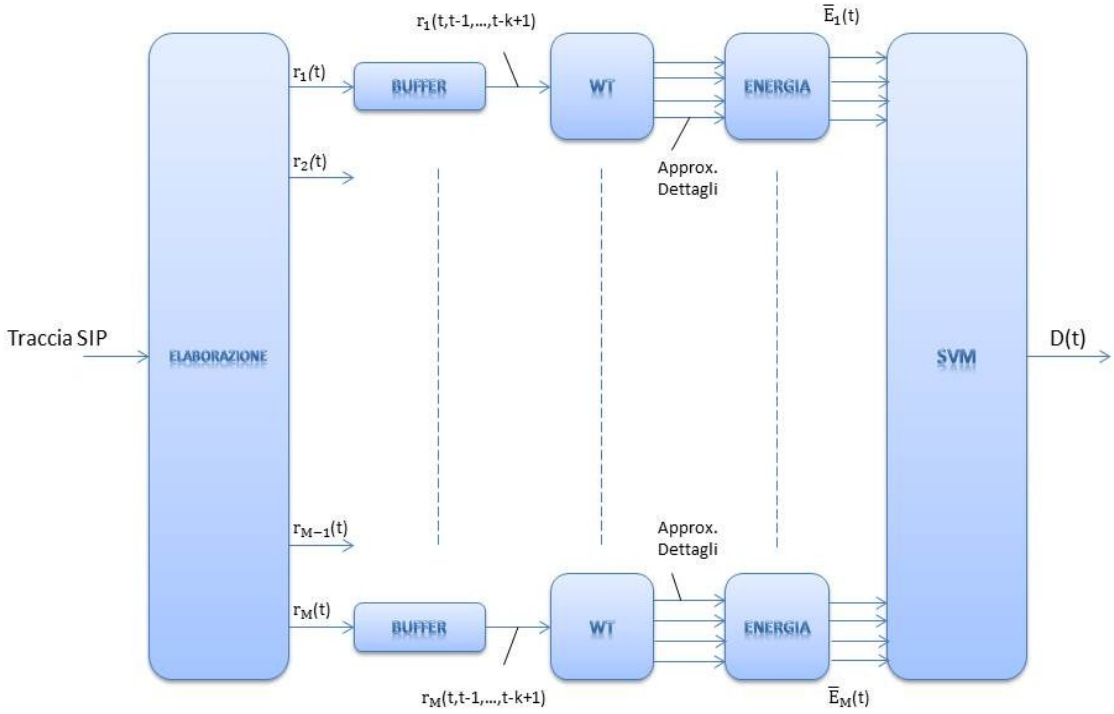


Figura 23 - Schema a blocchi completo

La porzione di ADS comprendente il blocco di estrazione, i buffer, le trasformate wavelet e l'estrazione dei parametri, sono implementate in linguaggio C, con l'ausilio delle librerie "libpcap" per la lettura delle tracce di traffico in formato ".pcap", e "lgsl" per il calcolo delle trasformate wavelet; il blocco finale di decisione che completa l'ADS è implementato, come nel caso precedente, attraverso uno script bash che richiama al suo interno i tool di LibSVM.

4. EMULATORE

Al fine di valutare le prestazioni delle soluzioni proposte nel capitolo precedente, è necessario disporre di tracce di traffico sia in situazioni di corretto funzionamento che in presenza di anomalie ben identificate temporalmente. Sono facilmente disponibili tracce di traffico catturate in situazioni di regolare funzionamento del sistema, mentre risulta difficile la cattura di tracce in cui sia presente e ben individuata temporalmente un'anomalia.

Per questo motivo risulta utile, se non necessario, ricreare un ambiente in cui è possibile emulare il sistema in situazione di corretto funzionamento, con la possibilità di ricreare artificialmente situazioni anomale.

SIPp, il test-tool sviluppato da Hewlett-Packard, presentato precedentemente, si presta facilmente ad essere utilizzato per generare traffico di segnalazione SIP secondo scenari anche complessi. Per questo motivo è adatto alla sintetizzazione del sistema di riferimento, programmando lo scenario eseguito dal software secondo opportuni parametri estratti dalle tracce reali in situazione di corretto funzionamento.

Vista l'architettura client-server di SIPp, è necessario distinguere il traffico delle tracce reali in chiamate entranti e chiamate uscenti, in modo da riprodurre tali componenti di traffico per mezzo di due coppie client-server del tool, come illustrato in figura 24.

Figura 24 - Schema emulazione chiamate entranti e chiamate uscenti.

I parametri estratti dalle tracce reali sono:

- Call flow.
- Tempo di inter-arrivo.

4.1 Call flow

I call flow sono la sequenza dei messaggi di segnalazione SIP per ogni Call-ID presente nelle tracce a disposizione.

Analizzando le tracce di traffico di riferimento, si possono trovare diversi call flow, solo alcuni sono ricorrenti, questi costituiscono circa il 70% del totale di quelli presenti nelle tracce stesse, perciò è una buona approssimazione limitarsi ad implementare questi, riportati in figura 25.

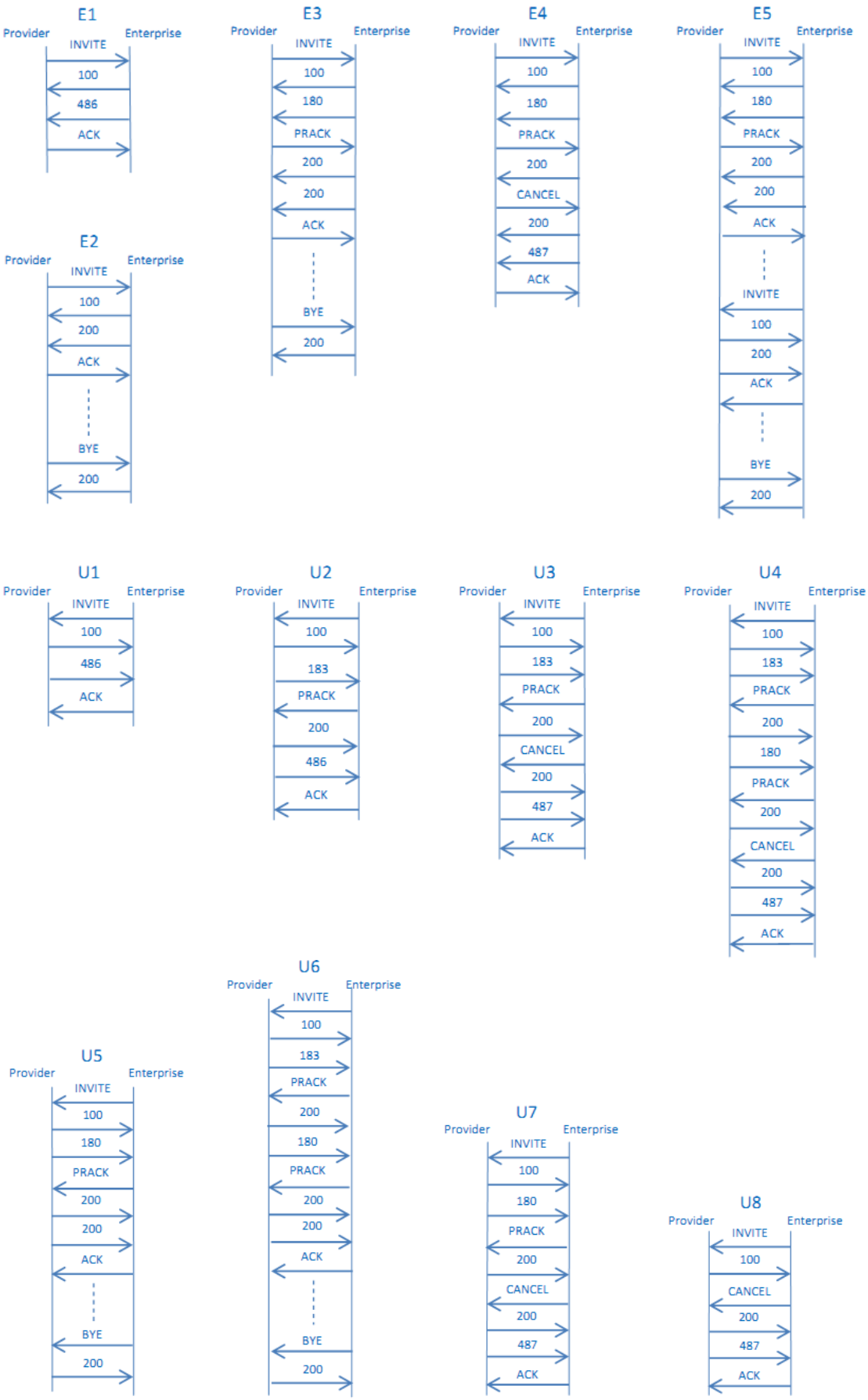


Figura 25 - Call flow considerati.

Partendo dall'immagine sopra riportata si nota che tali call flow corrispondono tutti a chiamate "COMPLETED", "REJECTED" o "CANCELLED", a seconda della presenza rispettivamente dei messaggi "BYE" (termina chiamate "andate in risposta"), "486 Busy Here" (l'utente chiamato è occupato) o "CANCEL" (il chiamante ha chiuso la chiamata prima di un'eventuale risposta del chiamato). Proseguendo nell'analisi, si osserva che a seguito di ogni risposta provisional come "183 Session Progress" o "180 Ringing" è presente il messaggio "PRACK", ovvero "provisional acknowledgement", e la relativa risposta "200"; questo perché il campo "Supported" del messaggio "INVITE" annovera l'opzione "100 rel", che richiede l'affidabilità di queste risposte.
E' presente anche un call flow in cui a seguito della risposta da parte del chiamato lato enterprise, si ripresenta un messaggio "INVITE"; questo perchè la chiamata viene trasferita ad altro utente, sempre lato enterprise, e vengono così fornite al chiamante le informazioni necessarie al trasferimento. E' anche presente la trasmissione di messaggi OPTIONS dopo ogni minuto in cui la chiamata si trova nello stato "IN CALL".

La tabella 2 mostra la percentuale di ricorrenza di ciascun call flow, rispetto al totale dei soli call flow considerati.

Call flow chiamate entranti	Percentuale sul totale dei call flow chiamate entranti
E1	22%
E2	11%
E3	28%
E4	19%
E5	20%

Call flow chiamate uscenti	Percentuale sul totale dei call flow chiamate uscenti
U1	10%
U2	4%
U3	18%
U4	6,3%
U5	31,5%
U6	11,7%
U7	13,5%
U8	5%

Tabella 2 - Percentuali di ricorrenza dei call flow, divisi tra entranti e uscenti.

4.2 Tempi di interarrivo

Il tempo di inter-arrivo è quello che intercorre tra l'arrivo di un messaggio e il successivo nella medesima chiamata, per ogni possibile coppia di messaggi consecutivi presenti nei call flow individuati.

L’estrazione dei tempi di inter-arrivo tra le coppie di messaggi permette di riprodurre i call flow sopra riportati con la giusta distribuzione dei messaggi nel tempo, quindi non solo la sequenzialità dei messaggi ma anche la tempistica.
Il tempo di inter-arrivo tra coppie di messaggi dipende anzitutto dalla dipendenza o meno dal comportamento d’utente; nel primo caso la media sarà nell’ordine dei secondi, nel secondo caso tra qualche millisecondo e qualche centinaio di millisecondi; influisce anche la differenza tra coppie in cui il secondo messaggio è una risposta immediata del nodo di rete a cui viene inoltrata la richiesta, ad esempio la coppia “INVITE-100 Trying”, o se la risposta è trasmessa nel momento in cui la richiesta raggiunge l’end-user , per esempio la coppia “100 Trying-180 Ringing”.
Inoltre si nota la differenza tra i tempi di inter-arrivo anche tra coppie identiche di messaggi, a seconda del verso di percorrenza di essi, questo perché lo strumento di cattura delle tracce si trova in prossimità di uno dei due Border Element, detto “Near End” (NE), perciò il tempo che trascorre tra una richiesta verso NE e risposta da NE non contempla il tempo di latenza della rete, seppur quest’ultimo sia minimo in quanto il layer fisico è prettamente una fibra ottica, mentre nel verso opposto il tempo tra una richiesta verso il “Far End” (FE) e la risposta da FE contempla due volte la latenza del link. Un esempio è la differenza tra la coppia “CANCEL-200” con “CANCEL” entrante nella rete enterprise, con tempo medio di 3 ms, e la stessa coppia con “CANCEL” uscente, con tempo medio di circa 7 ms.

Il tool SIPp mette a disposizione un numero limitato di distribuzioni statistiche utilizzabili nelle pause tra la ricezione di una richiesta e l’invio della relativa risposta; benché queste siano le più diffuse (normale, esponenziale negativa e uniforme) si rende necessaria un’approssimazione delle distribuzioni osservate.

Nella tabella 28, riportata in appendice, è presente l’elenco delle possibili coppie di messaggi SIP individuabili nei call-flow considerati; per ognuna di esse è riportato il grafico illustrante la distribuzione del tempo d’interarrivo, tra il primo ed il secondo messaggio della coppia considerata, osservata nelle tracce di traffico a disposizione. Inoltre è indicato il modello empirico scelto per riprodurre in modo approssimato nell’emulatore la distribuzione osservata. Per convenzione vengono indicati con “e” i messaggi entranti nella rete enterprise, e con “u” quelli uscenti.

4.3 Implementazione

L'emulatore è distinto in quattro componenti: una coppia client-server per l'emulazione delle chiamate generate dalla rete enterprise e instradate sul trunk verso il provider, ovvero le chiamate uscenti; e una seconda coppia client-server per le chiamate provenienti dal provider che vanno in terminazione nella rete enterprise, chiamate entranti.

In ogni coppia la componente client genera le chiamate verso la relativa componente server e la chiamata evolve secondo lo scenario prestabilito tramite file "xml"; programmando opportunamente tale file è possibile riprodurre i call flow individuati in precedenza rispettando le distribuzioni temporali dei messaggi sopra discusse.

La durata della conversazione è distribuita come una gaussiana a media 2 minuti con deviazione standard di 40 secondi. Per esigenze implementative, il meccanismo di "keep alive" della conversazione attraverso la trasmissione di messaggi "OPTIONS" ogni 60 secondi è implementata nella sola coppia client-server relativa alle chiamate uscenti.

4.3.1 Chiamate entranti

In figura 26 è riportato uno schema raffigurante le possibili evoluzioni della chiamata entrante, che ha inizio con un messaggio "INVITE" trasmesso dal client, costituito da SIPp che esegue lo scenario uacEntrante.xml (user agent client Entrante), al server, SIPp che esegue lo scenario uasEntrante.xml (user agent server Entrante).

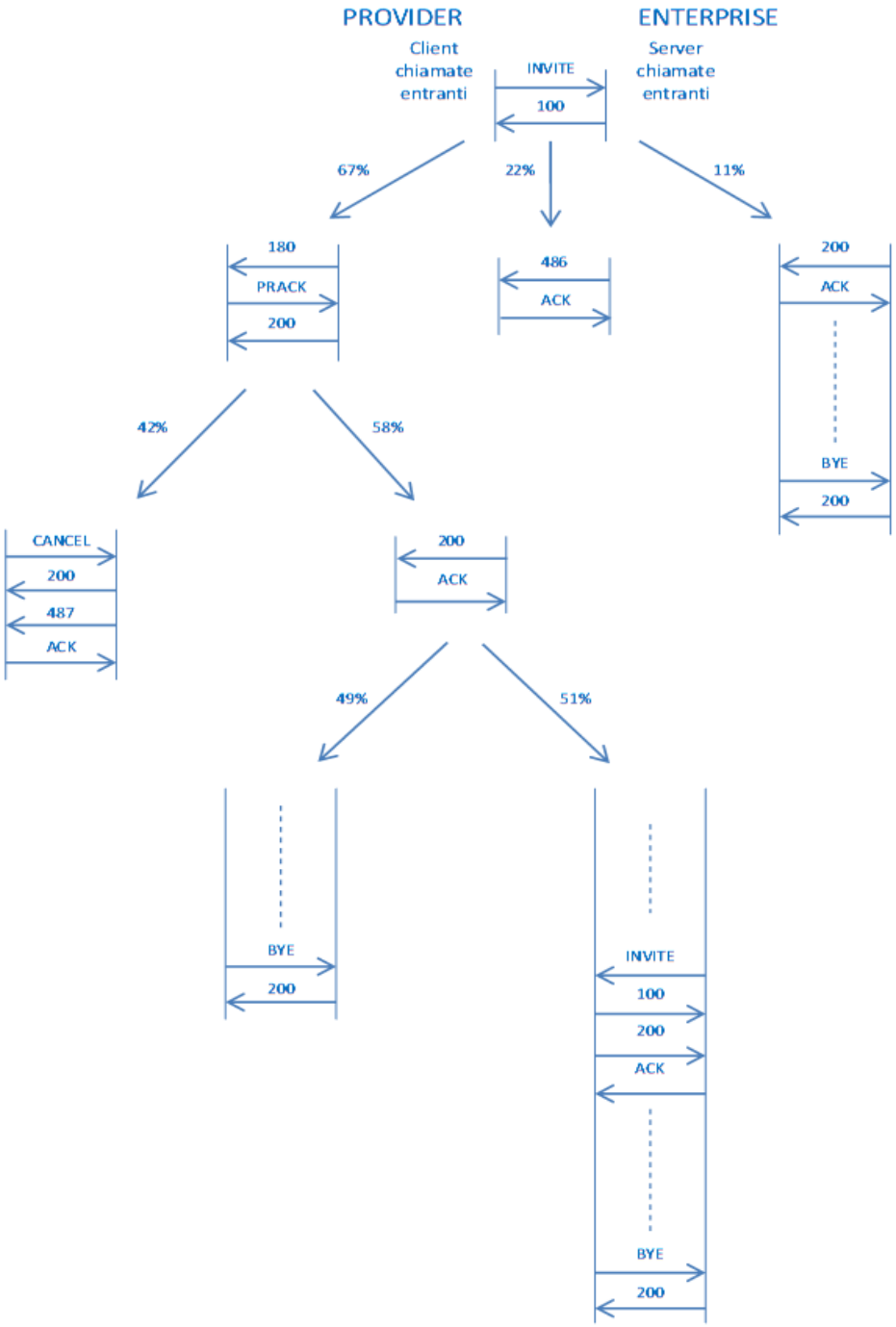


Figura 26 - Possibile evoluzione di una chiamata entrante nella rete enterprise.

Le decisioni di “branching” durante l’esecuzione dello scenario sono prese tutte dal lato server. All’interno di un’istruzione come “send”, “recv” o “nop” è possibile inserire due campi, “chance=*a*” e “next=*b*”, attraverso i quali è possibile far “saltare” l’esecuzione del codice all’istruzione posta dopo la label *b* con probabilità *a*, oppure eseguire l’istruzione successiva con probabilità 1-*a* .

Alla ricezione del messaggio “INVITE” il server decide se non risponderà alla chiamata, mettendosi in attesa del messaggio “CANCEL”, comunicandolo precedentemente al client, che entrerà a sua volta nel branch corretto, attraverso l’invio di un messaggio “180 Rynging” anziché “180 Ringing”. Questo automatismo è necessario a causa di un limite di SIPp che non consente elasticità nelle azioni condizionali (ad esempio non è possibile dal lato client decidere di inviare “CANCEL” se non si riceve il “200 OK” dal server entro un limite prefissato). Allo stesso modo il server decide se durante la conversazione ci sarà un trasferimento di chiamata ad altro utente accettando la chiamata con un “200 0K” (con zero “0” anziché “O”). Non è possibile programmare il tool in modo da metterlo in attesa di ricevere un messaggio, eseguendo azioni come la trasmissione di un altro messaggio se non ne viene ricevuto alcuno entro un determinato tempo di attesa. Il client, quindi, mettendosi in attesa di un messaggio di risposta “200 OK”, non può autonomamente decidere di terminare in anticipo la chiamata attraverso l’invio di un messaggio “CANCEL” se non riceve risposta entro un limite di tempo prefissato, ma è necessario che il server gli comunichi in anticipo la decisione di non trasmettere la risposta. In questo modo il client non si metterà in attesa di tale risposta, ma entrerà in un apposito branch in cui dopo un periodo di attesa variabile, invierà il messaggio “CANCEL”.

4.3.1.1Server

Il file “uasEntranti.xml” è “eseguito” tag per tag da SIPp, seguendo l’algoritmo descritto dallo pseudo-codice che segue:

1- *Ricevi “INVITE”*
2- *Pausa di 3 ms*
3- *Invia “100 Trying” e con probabilità 22% salta alla riga 29*
4- *Con probabilità 14% pausa con ddp gaussiana a media 30 ms e deviazione standard 3 ms poi salta alla riga 13*
5- *Pausa con distribuzione gaussiana a media 400 ms e deviazione standard 100 ms*

6- Con probabilità 42% salta alla riga 18

7- Invia "180 Ringing" con ritrasmissione (secondo standard SIP)

8- Ricevi "PRACK"

9- Pausa di 5 ms

10- Invia "200 OK"

11- Pausa con distribuzione esponenziale negativa a media 3 secondi

12- Con probabilità 51% salta alla riga 31

13- Invia "200 OK" con ritrasmissione

14- Ricevi "ACK"

15- Ricevi "BYE"

16- Pausa di 3 ms

17- Invia "200 OK" e TERMINA

18- Invia "180 Rynging" con ritrasmissione

19- Ricevi "PRACK"

20- .Pausa 5 ms

21- Invia "200 OK"

22- Ricevi "CANCEL"

23- Pausa di 3 ms

24- Invia "200 OK"

25- Pausa di 1 ms

26- Invia "487 Request Terminated" con ritrasmissione

27- Ricevi "ACK" e TERMINA

28- Pausa con distribuzione gaussiana a media 130 ms e deviazione 20 secondi

29- Invia "486 Busy Here" con ritrasmissione

30- Ricevi "ACK" e TERMINA

31- Invia "200 (zero)0K" con ritrasmissione

32- Ricevi "ACK"

33- Pausa con distribuzione esponenziale negativa a media 9 secondi

34- Invia "INVITE" con ritrasmissione

35- Ricevi "100 Trying"

36- Ricevi "200 OK"

37- Pausa con distribuzione gaussiana a media 130 ms e deviazione 50 ms

38- Invia "ACK" e salta a riga 15

La tabella 3 sottostante riporta le righe di pseudo-codice eseguite dall'emulatore per riprodurre ciascun call-flow individuato in precedenza e riportato nella figura 25.

Call flow	Righe di pseudo-codice eseguite
E1	1, 2, 3, 29, 30
E2	1 -> 4, 13 -> 17
E3	1 -> 17
E4	1 -> 6, 18 -> 27
E5	1 -> 12, 31 -> 38, 15 -> 17

Tabella 3 - Legame tra call-flow entranti e pseudo-codice server

4.3.1.2 Client

Di seguito è riportato lo pseudo-codice che descrive l'algoritmo contenuto nel file "uacEntranti.xml".

1- *Pausa esponenziale negativa con media 1 secondo*
2- *Invia "INVITE" con ritrasmissione (secondo standard SIP)*
3- *Ricevi "100 Trying"*
4- *Se ricevi "486 Busy Here" salta alla riga 22*
5- *Se ricevi "200 OK" salta alla riga 12*
6- *Ricevi "180 Ringing" o "180 Rynging"*
7- *Pausa di 11 ms o 21 ms*
8- *Invia "PRACK" con ritrasmissione*
9- *Ricevi "200 OK"*
10- *Se alla riga 6 hai ricevuto "180 Rynging" salta alla riga 18*
11- *Ricevi "200 OK" o "200 (zero)0K"*
12- *Pausa di 25 ms*
13- *Invia "ACK"*
14- *Se alla riga 11 hai ricevuto "200 (zero)0K" salta alla riga 24*
15- *Pausa con distribuzione gaussiana a media 2 minuti e deviazione standard 40 secondi (durata chiamata)*
16- *Invia "BYE" con ritrasmissione*
17- *Ricevi "200 OK" e TERMINA*
18- *Pausa con distribuzione uniforme tra 1 secondo e 59 secondi*
19- *Invia "CANCEL" con ritrasmissione*
20- *Ricevi "200 OK"*
21- *Ricevi "487 Request Terminated"*
22- *Pausa di 7 ms*
23- *Invia "ACK" e TERMINA*
24- *Ricevi "INVITE"*
25- *Pausa di 3 ms*

26- *Invia "100 Trying"*
27- *Pausa di 2,3 secondi*
28- *Invia "200 OK" con ritrasmissione*
29- *Ricevi "ACK"*
30- *Pausa con distribuzione gaussiana a media 2 minuti e deviazione standard 40 secondi (durata chiamata)*
31- *Invia "BYE" con ritrasmissione*
32- *Ricevi "200 OK" e TERMINA*

Nella tabella 4 sono riportate le righe di pseudo-codice client eseguite per la riproduzione dei call-flow relativi alle chiamate entranti.

Call flow	Righe di pseudo-codice eseguite
E1	1 -> 4, 22, 23
E2	1 -> 5, 12 -> 17
E3	1 -> 17
E4	1 -> 10, 18 -> 23
E5	1 -> 14, 24 -> 32

Tabella 4 - Legame tra call-flow entranti e pseudo-codice client

4.3.2 Chiamate uscenti

Come per le chiamate entranti, in figura 27 è mostrato lo schema relativo all'evoluzione di una chiamata uscente.

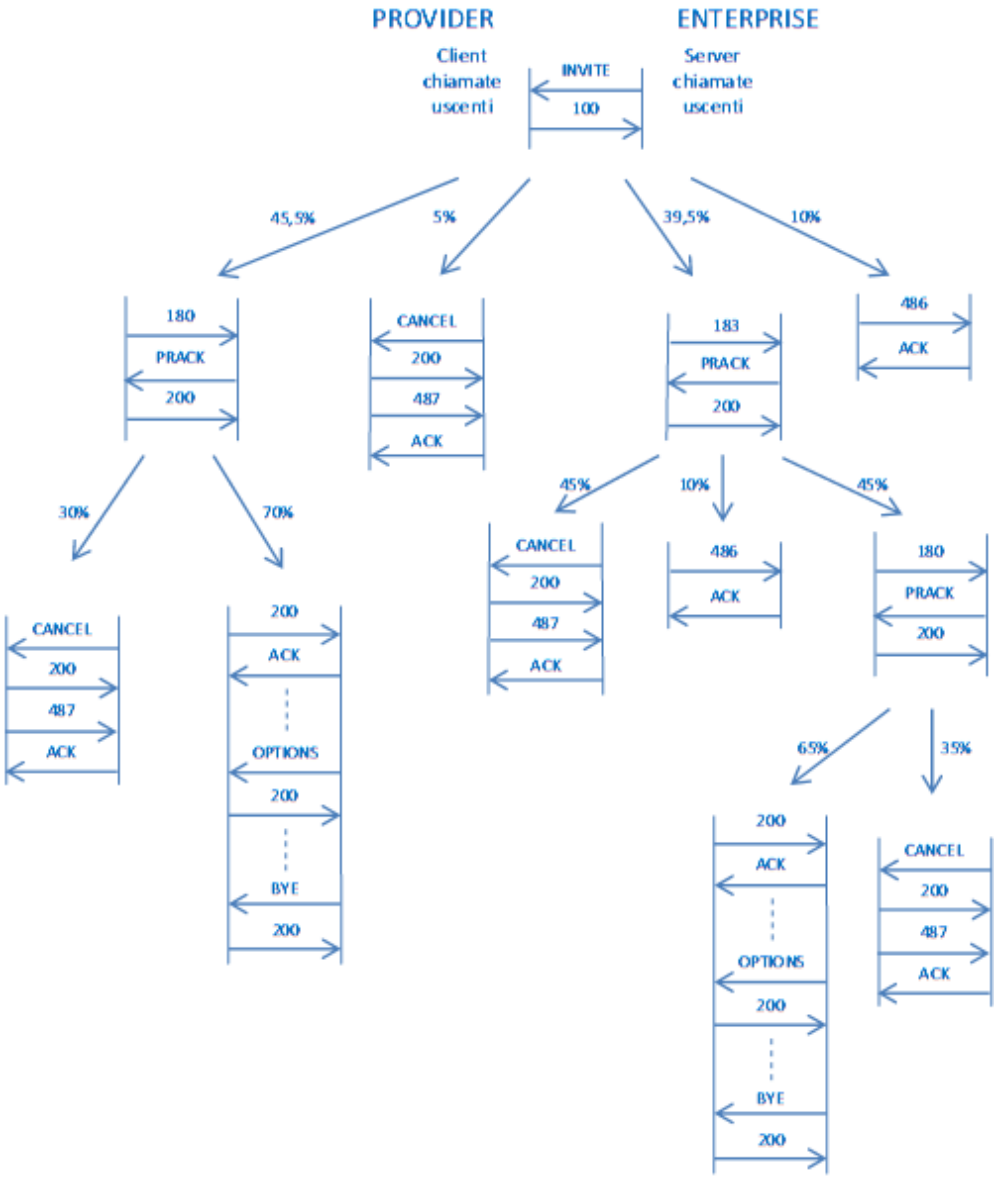


Figura 27 - Possibile evoluzione di una chiamata uscente dalla rete enterprise.

Anche in questo caso le decisioni di "branching" sono prese dal server, ad eccezione della trasmissione degli "OPTIONS".

4.3.2.1 Server

Lo pseudo-codice che segue descrive l'algoritmo contenuto nel file "uasUscenti.xml".

1- *Ricevi "INVITE", pausa di 7 ms, con probabilità 5% salta alla riga 17*
2- *Invia "100 Trying" e con probabilità 10.5% salta alla riga 24*

3- *Con probabilità 47% salta alla riga 27*
4- *Con probabilità 30% salta alla riga 37*
5- *Pausa di 3 secondi oppure pausa con distribuzione gaussiana a media 6 secondi e dev. standard 1 secondo*
6- *Invia "180 Ringing" con ritrasmissione (secondo standard SIP)*
7- *Ricevi "PRACK"*
8- *Pausa con distribuzione gaussiana a media 13 ms e dev. standard 2 ms*
9- *Invia "200 OK"*
10- *Pausa con distribuzione uniforme tra 1 secondo e 13 secondi*
11- *Invia "200 OK" con ritrasmissione*
12- *Ricevi "ACK"*
13- *Se ricevi "OPTIONS" salta alla riga 42*
14- *Ricevi "BYE"*
15- *Pausa di 11 ms*
16- *Invia "200 OK" e TERMINA*
17- *Invia "100 Traing"*
18- *Ricevi "CANCEL"*
19- *Pausa di 7 ms*
20- *Invia "200 OK"*
21- *Pausa di 6 ms*
22- *Invia "487 Request Terminated" con ritrasmissione*
23- *Ricevi "ACK" e TERMINA*
24- *Pausa con distribuzione gaussiana a media 2.2 s e dev. standard 300 ms*
25- *Invia "486 Busy Here" con ritrasmissione*
26- *Ricevi "ACK" e TERMINA*
27- *Pausa con distribuzione gaussiana a media 2.1 s e dev. standard 300 ms*
28- *Invia "183 Session Progress" con ritrasmissione*
29- *Ricevi "PRACK" e pausa con distribuzione gaussiana a media 13 ms e dev. standard 2 ms*
30- *Con probabilità 10% salta alla riga 40*
31- *Con probabilità 50% invia "200 (zero)0K" e salta alla riga 18*
32- *Invia "200 OK" e con probabilità 35% salta alla riga 37*
33- *Pausa di 1 ms e invia "180 Ringing" con ritrasmissione*
34- *Ricevi "PRACK" e pausa con distribuzione gaussiana a media 13 ms e dev. standard 2 ms*
35- *Invia "200 OK" e salta alla riga 11*
36- *Pausa di 3 secondi oppure pausa con distribuzione gaussiana a media 6 secondi e dev. standard 1 secondo*
37- *Invia "180 Rynging" con ritrasmissione*
38- *Ricevi "PRACK" e pausa con distribuzione gaussiana a media 13 ms e dev. standard 2 ms*

39- *Invia "200 OK" e salta a riga 18*

40- *Invia "200 OK"*

41- *Pausa con distribuzione uniforme tra 100 ms e 800 ms poi salta alla riga 25*

42- *Invia "200 OK" e salta alla riga 13*

La tabella 5 seguente riporta la relazione tra le righe di pseudo-codice e i call-flow uscenti riprodotti dall'emulatore.

Call flow	Righe di pseudo-codice eseguite
U1	1, 2, 24, 25, 26
U2	1, 2, 3, 27 -> 30, 40, 41, 25, 26
U3	1, 2, 3, 27 -> 31, 18 -> 23
U4	1, 2, 3, 27 -> 32, 37, 38, 39, 18 -> 23
U5	1 -> 12, ciclo 13 e 42, 14, 15, 16
U6	1, 2, 3, 27 -> 35, 11, 12, ciclo 13 e 42, 14, 15, 16
U7	1 -> 4, 37 -> 39, 18 -> 23
U8	1, 17 -> 23

Tabella 5 - Relazione tra call-flow uscenti e pseudo-codice server

4.3.2.2 Client

Infine presento lo pseudo-codice che descrive l'algoritmo contenuto nel file "uacUscenti.xml".

1- *Pausa esponenziale negativa con media 1 secondo*

2- *Invia "INVITE" con ritrasmissione (secondo standard SIP)*

3- *Se ricevi "100 Traing" salta alla riga 16*

4- *Ricevi "100 Trying"*

5- *Se ricevi "486 Busy Here" salta alla riga 21*

6- *Se ricevi "183 Session Progress" salta alla riga 22*

7- *Ricevi "180 Ringing" o "180 Rynging" e pausa di 7 ms o con distribuzione uniforme tra 280 ms e 390 ms*

8- *Invia "PRACK" con ritrasmissione*

9- *Ricevi "200 OK", se alla riga 7 hai ricevuto "180 Rynging" salta alla riga 17*

10- *Ricevi "200 OK" e pausa con distribuzione gaussiana a media 130 ms e dev. standard 50 ms*

11- *Invia "ACK"*

12- *Con probabilità 70% salta alla riga 31*

13- *Pausa con distribuzione uniforme tra 1 secondo e 60 secondi*

14- *Invia "BYE" con ritrasmissione*

15- *Ricevi "200 OK" e TERMINA*
16- *Pausa di 180 ms e salta alla riga 18*
17- *Pausa con distribuzione uniforme tra 1 secondo e 59 secondi*
18- *Invia "CANCEL" con ritrasmissione*
19- *Ricevi "200 OK"*
20- *Ricevi "487 Request Terminated"*
21- *Pausa di 2 ms, invia "ACK" e TERMINA*
22- *Pausa con distribuzione gaussiana a media 350 ms e dev. standard 40 ms*
23- *Invia "PRACK" con ritrasmissione*
24- *Se ricevi "200 (zero)0K" salta alla riga 17*
25- *Ricevi "200 OK"*
26- *Se ricevi "486 Busy Here" salta alla riga 21*
27- *Ricevi "180 Ringing" o "180 Rynging" , pausa di 7 ms o con distribuzione uniforme tra 280 ms e 390 ms*
28- *Invia "PRACK" con ritrasmissione*
29- *Ricevi "200 OK", se alla riga 27 hai ricevuto "180 Rynging" salta alla riga 17*
30- *Salta alla riga 10*
31- *Pausa di 60 secondi e invia "OPTIONS"*
32- *Ricevi "200 OK", se ha CSeq="7 OPTIONS" salta alla riga 13*
33- *Salta alla riga 12*

In tabella 6 sono riportati i call-flow uscenti e le rispettive righe di pseudo-codice eseguite dall'emulatore per riprodurli.

Call flow	Righe di pseudo-codice eseguite
U1	1 -> 5, 21
U2	1 -> 6, 22 -> 26, 21
U3	1 -> 6, 22 -> 24, 17 -> 21
U4	1 -> 6, 22 -> 29, 17 -> 21
U5	1 -> 11, ciclo 12 31 32 33, 13 -> 15
U6	1 -> 6, 22 -> 30, 10, 11, ciclo 12 31 32 33, 13 ->15
U7	1 -> 9, 17 -> 21
U8	1, 2, 3, 16 -> 21

Tabella 6 - Relazione tra call-flow uscenti e pseudo-codice client

4.4 Ambiente di emulazione

L'ambiente di emulazione è costituito da due PC (CPU: Intel Celeron 3.06GHz, RAM: 1Gb, S.O.: Ubuntu 11.10) entrambi connessi ad un hub Ethernet a 10 Mbit/s. Uno di questi esegue SIPp come client per le chiamate uscenti e server per quelle entranti rappresentando il Near End, o Border Element della rete enterprise; l'altro esegue SIPp come client per le chiamate entranti e server per quelle uscenti, figurando come il Far End, o Border Element della rete del Provider. Allo stesso hub è connesso un terzo PC che effettua la cattura della traccia di traffico. Lo schema dell'ambiente di emulazione è riportato in figura 28, mentre la figura 29 mostra uno schema dello scenario reale equivalente.

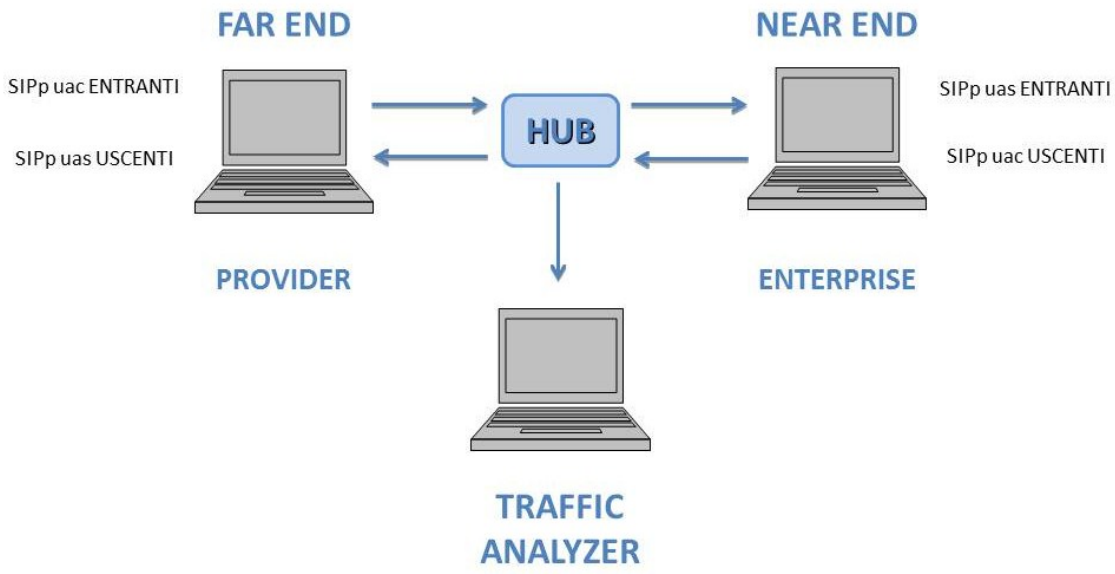


Figura 28 - Schema dell'ambiente di emulazione

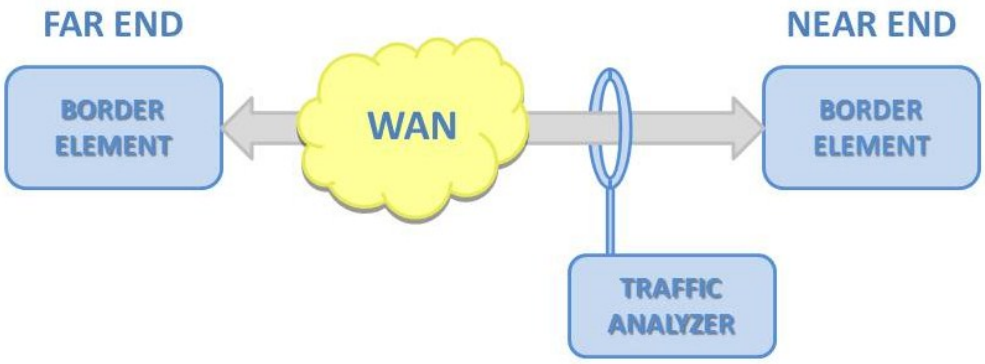


Figura 29 - Schema del sistema di riferimento.

5. ANOMALIE

La scelta della gamma di anomalie da riprodurre all'interno dell'ambiente di emulazione presentato nel capitolo precedente, è stata guidata dall'idea di testare le prestazioni degli approcci presentati con riferimento ad eventi relativamente frequenti e conosciuti, i cui effetti producono degrado della qualità del servizio offerto, senza comprometterlo totalmente. Sulla base di queste osservazioni e delle informazioni raccolte durante il periodo di tesi presso ICT Consulting, società leader nel settore della consulenza tecnologica che annovera tra i suoi clienti grosse realtà aziendali di tipo "Large Enterprise", è ragionevole considerare un buon test-bed la seguente gamma di anomalie:

- "Link-Flap"
- "CPU-Hog"
- "Loop"

5.1 Link-Flap

Una tipico e diffuso evento anomalo è il "flap" di un link, ovvero la ripetuta transizione dell'interfaccia da "up" a "down" e viceversa, generalmente causata da un guasto "hardware" dell'interfaccia stessa. Tipiche cause di link-flap sono quindi problemi di livello 1, come un cavo rovinato, un duplex mismatch o una scheda danneggiata.

Il "flapping" di un link, essendo a tutti gli effetti una ripetuta interruzione del collegamento, causa la perdita dei pacchetti trasmessi nelle frazioni di tempo in cui il link è inattivo. Questo si traduce nella mancata ricezione da parte di ciascun Border Element (BE) dei messaggi inviati dall'altro BE, e quindi la ritrasmissione di essi negli istanti successivi, secondo l'automatismo di ritrasmissione dei pacchetti definito dal protocollo SIP.

Esistono algoritmi di contenimento di questo fenomeno, ad esempio quello presente su gran parte degli apparati Cisco, attivo direttamente a livello link sullo switch. Si tratta di un algoritmo di detection che, una volta individuato il flap di una delle interfacce dello switch, agisce al fine di prevenire danni a livello di rete, disabilitando

automaticamente l'interfaccia compromessa. Si dice che la porta è posta nello stato "Err-disabled". Convenzionalmente l'algoritmo di detection presente sugli switch Cisco individua l'evento anomalo e disabilita l'interfaccia se quest'ultima cambia di stato, da attiva a inattiva e viceversa, per più di cinque volte in un lasso di tempo di dieci secondi [31]. Transizioni meno frequenti del link non sono quindi individuate da tale algoritmo.

Al fine di emulare un evento anomalo di tipo "link-flap" sul trunk, nell'ambiente di emulazione precedentemente presentato, è necessario isolare periodicamente il PC rappresentante il Far End (FE), in modo che il guasto figuri tra il FE stesso ed il Traffic Analyzer. In questo modo l'analizzatore di traffico vede i messaggi trasmessi dal Near End (NE) verso il FE, inclusi quelli che il FE non riceve a causa del flap, mentre non vede i messaggi provenienti dal FE e diretti al NE. Il periodico isolamento del PC-FE può essere realizzato con l'ausilio di due funzionalità di Ubuntu, il sistema operativo presente sul PC stesso. Per "droppare" i pacchetti in entrata è utile il firewall predefinito del sistema operativo, su cui si può agire direttamente da riga di comando, e quindi di facile utilizzo anche attraverso script. Per bloccare il traffico in uscita, invece, è utilizzabile il Traffic Control di Linux, il "tc", anch'esso utilizzabile da riga di comando.

Per emulare una transizione da "up" a "down" di un link è quindi sufficiente inserire simultaneamente una policy nel firewall attraverso il comando "iptables -P INPUT DROP" per il traffico in entrata, e una probabilità di "loss" unitaria sull'interfaccia di uscita (eth0) tramite il comando "tc qdisc add dev eth0 root netem loss 100%". Con il primo comando ogni pacchetto in ingresso verrà "droppato" a causa della policy del firewall, mentre con il secondo si agisce sulla coda dei pacchetti in uscita, eliminandoli sistematicamente anziché inviarli.

Per ripristinare le condizioni di corretto funzionamento del link, quindi la transizione da "down" ad "up", è necessario agire nuovamente sul firewall, modificando la policy precedente attraverso il comando "iptables -P INPUT ACCEPT", nonché in contemporanea eliminare la probabilità di "loss" sull'interfaccia d'uscita con il comando "tc qdisc del dev eth0 root netem".

Per evitare di riprodurre artificialmente un evento anomalo già comunemente rilevato dai più diffusi algoritmi esistenti, è possibile ricreare un flap composto da cinque transizioni da “up” a “down”, in cui il link rimane inattivo per un periodo di un secondo, intervallate da altrettanti periodi, anch’essi di un secondo, in cui il link è attivo. In figura 30 è illustrato il profilo temporale dell’anomalia riprodotta.

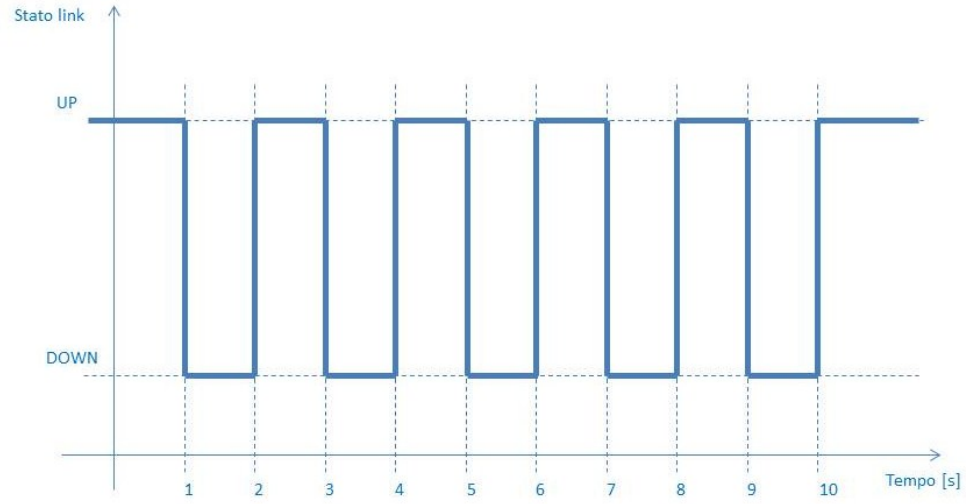


Figura 30 - Profilo temporale anomalia "Link-Flap"

Tecnicamente l’anomalia è riprodotta eseguendo come “super user” sul PC “Far End” uno script bash formato dalla ripetizione per cinque volte delle seguenti righe:

- iptables -P INPUT DROP
- tc qdisc add dev eth0 root netem loss 100%
- sleep 1
- iptables –P INPUT ACCEPT
- tc qdisc del dev eth0 root netem
- sleep 1

Nelle figure 31 e 32 sono riportati rispettivamente lo scenario di emulazione e l’equivalente reale, con localizzazione dell’evento anomalo.

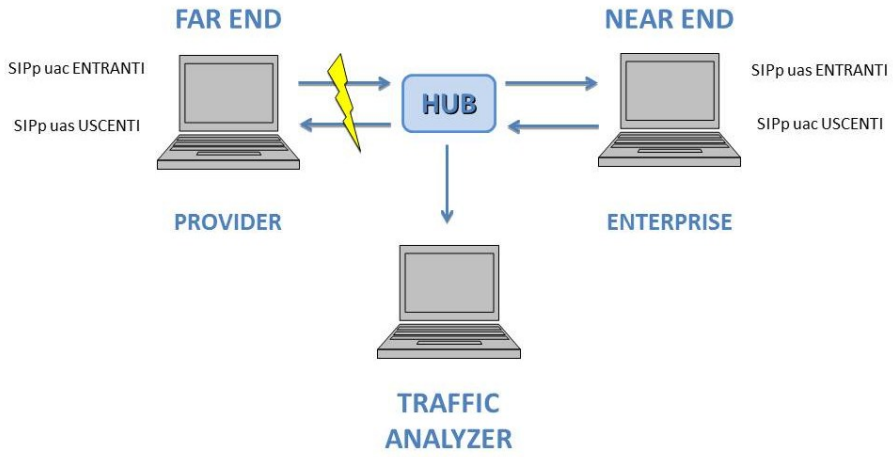


Figura 31 - Ambiente di emulazione con anomalia "link-flap"

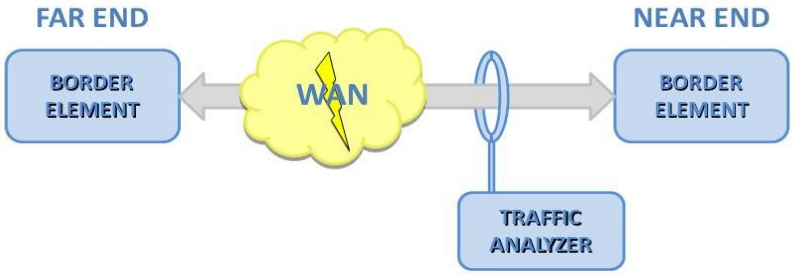


Figura 32 - Schema scenario reale con anomalia "link-flap"

5.2 CPU-Hog

Un altro evento anomalo si verifica quando il processo di gestione delle chiamate di un nodo di rete non ha più a disposizione le risorse che gli necessitano per funzionare correttamente perché “consumate” da un altro processo, detto “CPU-Hog”. La natura di questo processo malevolo può essere di diverso tipo; un esempio è la mancata chiusura del processo di “log”, per dimenticanza dell’operatore, al termine di un’attività effettuata sul nodo di rete; quel processo richiede sempre più risorse in termini di memoria e capacità di calcolo con il passare del tempo, fino ad inficiare il funzionamento del processo principale attivo sul nodo stesso, ovvero quello di gestione delle chiamate.

In generale un evento di questo tipo causa un notevole aumento del tempo di elaborazione dei messaggi in entrata e la possibile perdita di pacchetti in coda, qualora quest’ultima si saturi; l’insieme di questi eventi può facilmente dare origine a ritrasmissioni.

Per ricreare una condizione simile a quella sopradescritta, è possibile agire direttamente sul processo di gestione delle chiamate, limitando temporaneamente le risorse a sua disposizione ad una quota inferiore a quella che gli necessita. Questo permette di evitare l'avvio di uno o più processi "hog" sul PC su cui si vuole riprodurre l'evento anomalo, al fine di consumare anche parte delle risorse richieste dal software SIPp che gestisce le chiamate. E' possibile utilizzare un approccio di questo tipo grazie al software "cpu-limit" [32], un tool sviluppato inizialmente per hobby da un programmatore italiano, Angelo Marletta, e che ha avuto successo grazie alla sua efficacia e versatilità. Questo programma permette, se eseguito come "super user", di limitare l'utilizzo di CPU per un processo target ad una percentuale intera della disponibilità totale della/e CPU del PC. La riproduzione di questo evento anomalo nell'ambiente di emulazione è quindi attuata attivando il tool "cpu-limit" con target i due processi di gestione delle chiamate presenti sul PC interessato, SIPp client e SIPp server, e limitandone la disponibilità di CPU ad una percentuale intera inferiore a quella normalmente richiesta per un periodo di tempo di dieci secondi. In figura 33 è illustrato il profilo temporale delle risorse di CPU a disposizione dei processi SIPp. L'anomalia è stata riprodotta sul PC rappresentante il Far End, perciò i target di "cpu-limit" sono i due processi SIPp che eseguono gli scenari "uacEntranti.xml" e "uasUscenti.xml".

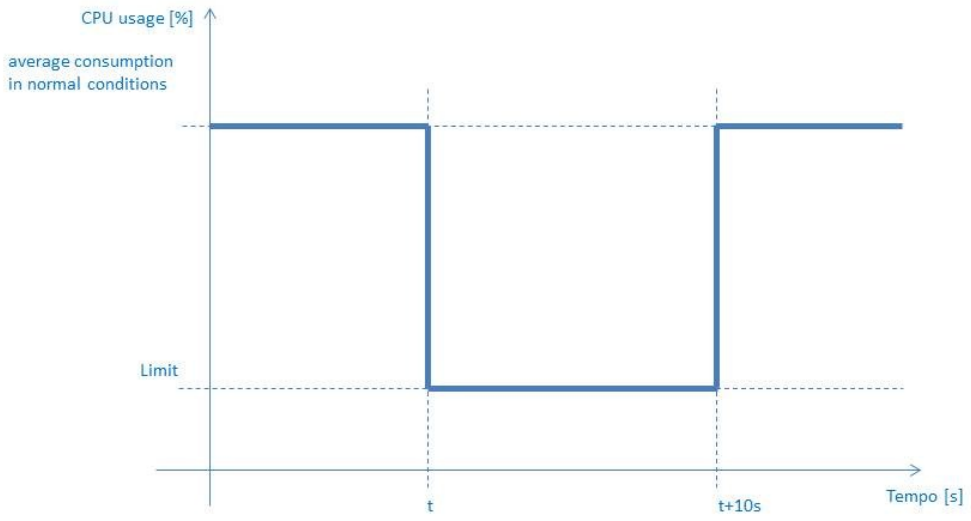


Figura 33 - Profilo temporale anomalia "CPU-Hog"

Anche questa anomalia è riprodotta attraverso l'esecuzione come "super user" sul PC "Far End" del seguente script bash:

- sip=$(pidof sipp) #acquisisco i pid dei due processi "sipp" attivi
- for i in $sip
- do
- cpulimit -p $i -l 2 & #limito la cpu disponibile al processo $i al limite 2 (2%)
- done
- cpul=$(pidof cpulimit) #acquisisco i pid dei due processi cpulimit appena creati
- sleep 10
- kill $cpul #elimino il limite di cpu sui processi sipp

Nelle figure 34 e 35 viene localizzata l'anomalia nell'ambiente di emulazione e nello schema dello scenario reale.

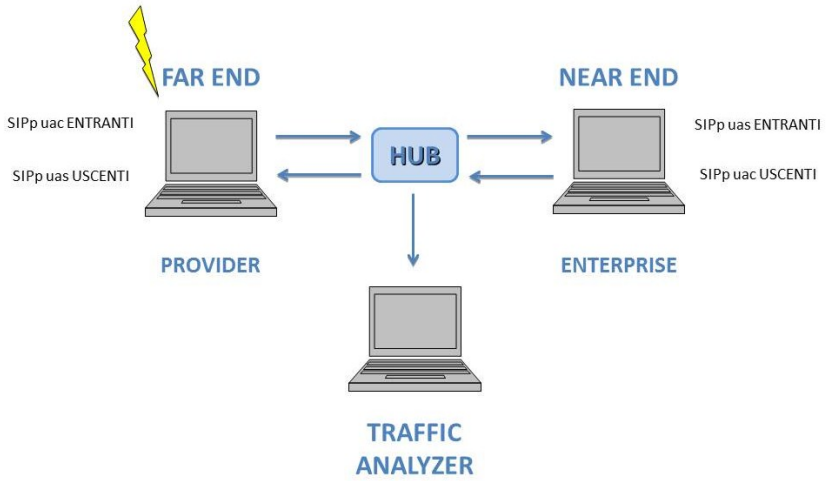


Figura 34 - Ambiente di emulazione con anomalia "CPU-hog"

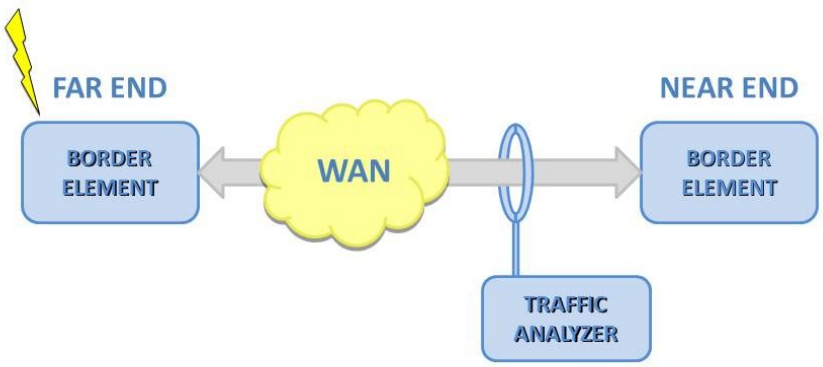


Figura 35 - Schema scenario reale con anomalia "CPU-hog"

5.3 Loop

Infine, considero l'evento anomalo di "loop" di numerazione. E' un fenomeno che consiste nel continuo re-instradamento di un messaggio, tipicamente un "INVITE" in fase di instaurazione di una chiamata, attraverso lo stesso collegamento. Questo può accadere a causa di un disallineamento nelle tabelle di routing dei nodi della rete, dovuto principalmente ad errori di configurazione delle tabelle stesse.

In una generica rete il "loop", o anello, può essere composto da numerosi "hop", ovvero il messaggio può attraversare diversi nodi di rete per giungere nuovamente in uno dei nodi già attraversati, e da lì venir reinstradato attraverso la stessa direzione del precedente passaggio. Nello specifico scenario di riferimento, ovvero un SIP Trunk, questo fenomeno può verificarsi attraverso il continuo scambio dello stesso messaggio tra i due Border Element, nel caso in cui le tabelle di instradamento di questi presentino una discordanza relativamente alla numerazione oggetto del messaggio "INVITE" coinvolto. Solo in questo caso il SIP Trunk risente di un fenomeno di "loop", a meno di disastri di configurazione altamente improbabili; infatti entrambi i Border Element possiedono tabelle di routing e conoscono quindi le numerazioni presenti all'interno della rete Enterprise; questo permette al Session Border Controller (SBC) lato Enterprise di bloccare l'errato instradamento verso il Trunk di un messaggio proveniente dalla rete aziendale nel caso in cui l'errore di routing si presenti in un nodo della rete Enterprise, mentre consente al SBC lato Provider di non inoltrare sul trunk messaggi provenienti dalla rete dell'operatore non destinati a numerazioni appartenenti alla rete Enterprise.

E' possibile attivare algoritmi di identificazione di questo fenomeno, principalmente basati sul monitoraggio dei messaggi che si presentano all'ingresso del nodo di rete al fine di individuare il ripresentarsi di messaggi identici e bloccarne la propagazione. Il principio alla base di questi algoritmi è il controllo di particolari campi di ciascun messaggio, quali la "Request-line", il "CSeq", i campi "Via" e le "tag" del campo "To"; questo al fine di evitare di confondere come se fosse un "loop", il riproporsi di messaggi apparentemente simili, frutto di un automatismo per nulla anomalo per il protocollo SIP, conosciuto con il nome di "spyral" [4]. Un tipico esempio di algoritmo di "loop detection" è il calcolo dell'hash MD5 della stringa composta dalle "tag" dei

campi “To” e “From”, dal “Call-ID”, dalla “Request-URI”, dal più recente dei campi “Via”, dal “sequence number” del campo “CSeq” e dagli eventuali campi di “Proxy-Require” e “Proxy-Authorization”.

Nei messaggi SIP è presente anche un campo chiamato “Max-Forwards” la cui funzionalità è quella di porre un limite di “hop” al messaggio stesso; una sorta di “Time To Live” del protocollo IP, trasposto al protocollo SIP. Questo campo, inizializzato con il valore di default 70, consente di porre un limite alla propagazione in rete dei messaggi SIP, e di conseguenza limita l’incidenza di eventi anomali come quello sopradescritto, interrompendo l’inoltro potenzialmente infinito di messaggi in “loop”.

L’obbligo di inserimento del campo “Max-Forwards” nei messaggi ha reso opzionale la presenza di algoritmi di “loop detection” negli elementi di rete. E’ quindi frequente l’utilizzo del solo campo “Max-Forwards”, o la combinazione di esso con altri campi equivalenti, allo scopo di individuare e limitare i fenomeni di “loop” [33].

Tipico esempio di “loop” è una chiamata proveniente da una numerazione appartenente alla rete Enterprise verso un utente esterno, correttamente instradata sul Trunk verso la rete dell’operatore, ma a causa di un errore nella tabella di routing del SBC lato Provider che dichiara la numerazione del chiamato appartenente alla rete Enterprise, viene nuovamente instradata sul Trunk nel verso opposto, innescando il “loop”.

La frequenza di eventi di questo tipo dipende dalla quantità di numerazioni interessate dal disallineamento delle tabelle di routing e dalla frequenza di chiamate verso tali numerazioni. Errori riguardanti vaste numerazioni saranno più facilmente individuabili attraverso i comuni indicatori di qualità di servizio della rete come l’Answer Seizure Ratio (ASR), mentre errori relativi ad una ristretta cerchia di numerazione avranno meno riscontro su tali indicatori. E’ quindi interessante riprodurre all’interno dell’ambiente di emulazione un evento in cui sia una singola chiamata ad andare in “loop”, anziché numerose chiamate contemporaneamente.

Ricreo quindi un’anomalia di “loop” in cui una sola chiamata viene generata dal Border Element della rete Enterprise attraverso l’invio sul trunk di un messaggio “INVITE” verso il Border Element della rete Provider, e quest’ultimo, dopo aver risposto al

messaggio con un “100 Trying”, inoltra nuovamente sul trunk il messaggio stesso. A sua volta il Border Element che aveva iniziato la chiamata risponde con un messaggio “100 Trying” e instrada nuovamente il messaggio “INVITE” sul trunk. Questo scambio di messaggi prosegue fino al raggiungimento del limite di “Max-Forwards”, ovvero dopo 70 passaggi sul trunk. Per fare questo è possibile utilizzare lo stesso tool con cui è realizzato l’ambiente di emulazione, ovvero SIPp. Nella figura 36 è riportato il call-flow di una chiamata in loop.

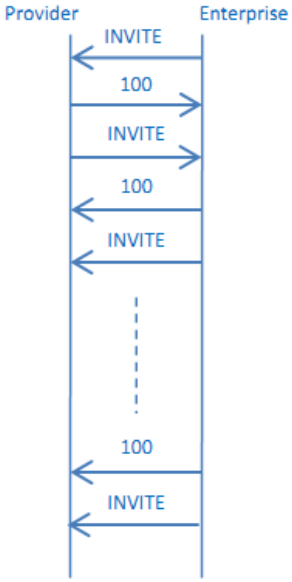


Figura 36 - Call flow di una chiamata in "loop"

La soluzione implementata consiste nella creazione di un ulteriore client SIPp e nella modifica di uno scenario server già utilizzato per l’emulazione della situazione di corretto funzionamento. Il nuovo client, il cui algoritmo è descritto nel file “uacUscentiLoop.xml” e di cui riporto sotto lo pseudo-codice, inizia una chiamata con un messaggio “INVITE” con una “Request-URI” particolare, e si mette in attesa di ricevere il messaggio “100 Trying” e lo stesso “INVITE” appena trasmesso; a questo punto risponde a sua volta con un “100 Trying” e ripete nuovamente le stesse azioni per trentacinque volte, ovvero finché il contatore del campo “Max-Forwards” non raggiunge lo zero. Questo scenario è eseguito da SIPp una sola volta, specificando un’apposita opzione nella riga di comando, in questo modo viene generata una singola chiamata anomala.

39- Invia “INVITE” con Request-URI “loop@...”

40- Ricevi "100 Trying"

41- Ricevi "INVITE"

42- Invia "100 Trying"

43- Se è la trentacinquesima volta che esegui questa riga, TERMINA, altrimenti salta alla riga 1

Per quanto attiene al lato server è sufficiente agire sul file "uasUscenti.xml", ovvero il server per l'emulazione delle chiamate uscenti dalla rete Enterprise posto sul PC che rappresenta il "Far End", aggiungendo una possibilità di "branching" specifica per interagire con il client di "loop" qualora questo venga attivato. Alla ricezione di un messaggio "INVITE", il server modificato, "uasUscentiLoop", controlla la "Request-URI", confrontandola con quella apposita del client di "loop"; nel caso quest'ultima corrisponda, il server entra nel "branch" aggiuntivo, in cui per prima cosa risponde al messaggio ricevuto con il "100 Trying" e successivamente invia lo stesso "INVITE" appena ricevuto, mettendosi in attesa della ricezione del relativo "100 Trying" e del successivo "INVITE", per poi ripetere di nuovo il "branch" fino alla trentacinquesima volta. Se invece la "Request-URI" non corrisponde a quella dei messaggi provenienti dal client di "loop", il server si comporta esattamente come descritto nel capitolo precedente per il server "uasUscenti.xml".

1- Ricevi "INVITE", se Request-URI è uguale a "loop@..." salta alla riga 43, altrimenti prosegui come per scenario riportato a pag. 68 (capitolo emulatore)

2- Vedi scenario riportato a pag. 68

42- Vedi scenario riportato a pag.68

43- Invia "100 Trying"

44- Invia "INVITE"

45- Ricevi "100 Trying"

46- Ricevi "INVITE"

47- Se è la trentaquattresima volta che esegui questa riga, TERMINA, altrimenti salta alla riga 42

Considerando il tempo di latenza della rete e quello di elaborazione dei messaggi, il tempo che intercorre tra l'invio del primo "INVITE" di "loop" e l'ultimo, ovvero quello con "Max-Forwards" nullo, è dell'ordine dei quattrocento o cinquecento millisecondi. In figura 37 è illustrato l'ambiente di emulazione in cui è presente l'ulteriore client SIPp.

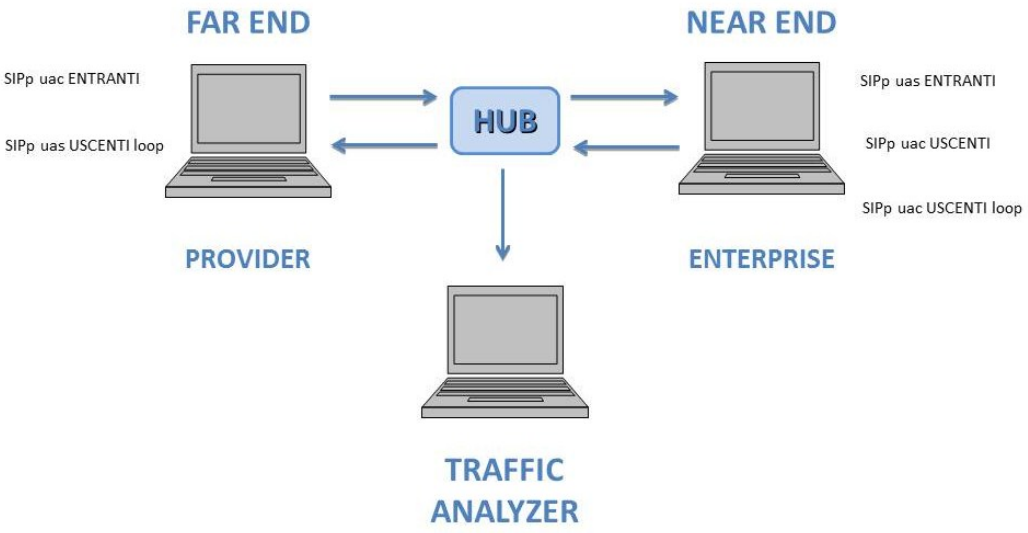


Figura 37 - Ambiente di emulazione con anomalia "loop"

6. RISULTATI

In questo capitolo sono riportate le prove sperimentali effettuate al fine di valutare le prestazioni delle tre implementazioni di ADS proposte.

Ogni prova consiste nella generazione di due tracce, la cui durata è specificata per ciascun esperimento effettuato, per mezzo dell'emulatore descritto precedentemente; la prima traccia viene generata priva di anomalie ed utilizzata per l'addestramento degli ADS, mentre nella seconda vengono ricreate le situazioni anomale descritte al capitolo "anomalie" di questo documento.

6.1 Parametri degli ADS

E' necessario impostare alcuni parametri degli ADS, comuni a tutte le soluzioni proposte.

Innanzi tutto occorre definire quali messaggi SIP si intende monitorare, ovvero, facendo riferimento al paragrafo "Dati in ingresso", il numero M di parametri estratti dalla traccia.
I messaggi SIP considerati dagli ADS nelle prove effettuate sono i seguenti:

- INVITE
- BYE
- CANCEL
- 486 BUSY
- 200 OK (come risposta ad un INVITE)
- PRACK
- OPTIONS

Tali messaggi vengono distinti anche per verso di percorrenza del trunk.

L'intervallo di tempo di osservazione, ovvero la durata T dello slot *t*, è impostata a 2 secondi, buon compromesso tra risoluzione temporale e rapidità di decisione dell'ADS. La percentuale di outliers ν da individuare nella traccia di riferimento è un parametro

empirico per gli ADS; variando il valore di questo parametro è possibile ottenere le curve di ROC in modo da valutare le prestazioni dell'ADS.

Il terzo algoritmo proposto, ovvero quello che fa uso della Wavelet decomposition, necessita l'impostazione di un ulteriore parametro oltre a quelli sopracitati: la dimensione del buffer, cioè la lunghezza della finestra di osservazione espressa in slot, definita precedentemente con la lettera *k*. Nelle prove effettuate tale parametro è stato impostato a 8, analizzando quindi l'andamento dei parametri in una finestra temporale di 16 secondi. In questo modo il blocco Wavelet Decomposition implementa tre stadi di trasformata Wavelet, fornendo in uscita una scomposizione del segnale in quattro sotto-bande.

6.2 Emulazione di anomalia "Link-Flap"

Di seguito sono presentate le prove effettuate con tracce di traffico in cui sono presenti anomalie di tipo link-flap.

Le tracce di traffico generate per mezzo dell'emulatore prevedono una serie di dieci eventi anomali di tipo link-flap, descritti nel capitolo precedente, distanziati tra loro di 90 secondi.

6.2.1 Esperimento a 40 cps

Questa prova è stata effettuata con entrambi i generatori di chiamate che compongono l'emulatore, sia chiamate uscenti che chiamate entranti, a 40 cps.

La traccia generata, lunga 1385 secondi, presenta le dieci anomalie negli intervalli temporali riportati nella tabella 7 seguente.

ANOMALIA	INIZIO [s]	FINE [s]
1	417	426
2	517	526
3	617	626
4	717	726
5	817	826
6	917	927
7	1018	1027
8	1118	1127
9	1218	1227
10	1318	1327

Tabella 7 - Intervalli temporali anomali nell'esperimento "Flap 40 cps"

Nell'intervallo di tempo in cui si presenta un'anomalia di questo tipo, l'ADS registra un aumento "a picco" dei messaggi SIP provenienti dal Near-End e destinati al Far-End,

questo perché avvengono ritrasmissioni dei messaggi persi durante i momenti in cui il link in “flap” si trova in stato “down”.

Di seguito, in figura 38, è riportato il profilo della traccia di traffico anomala; in ascissa il tempo espresso in secondi, mentre in ordinata il valore del parametro $r_m(t)$, ovvero la percentuale di messaggi di tipo *m* osservati nello slot *t*.

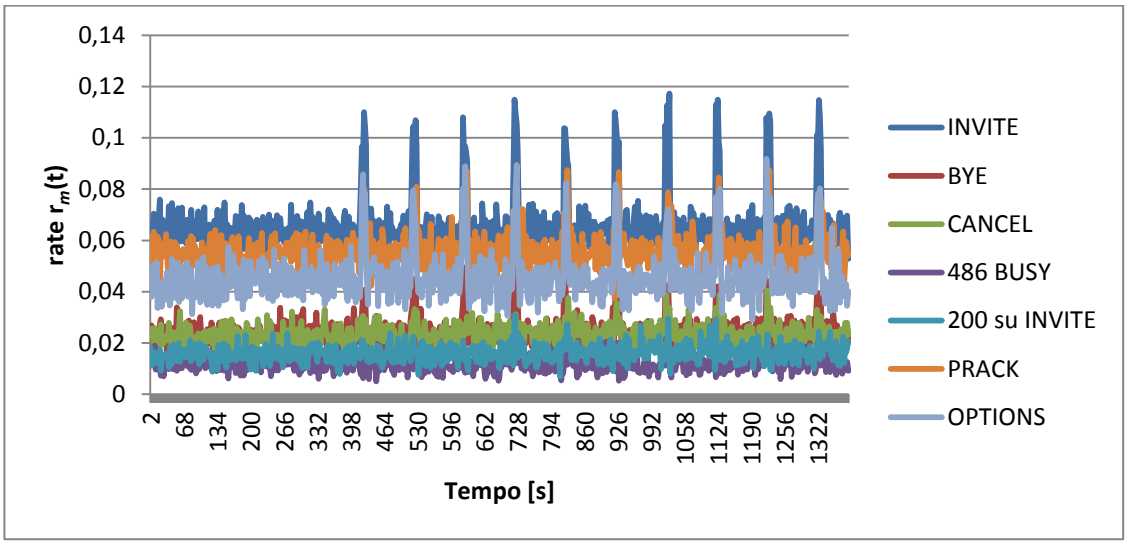


Figura 38 - Profilo temporale del traffico da Near-End a Far-End per test "flap 40 cps"

Allo stesso modo si registrano dei picchi per difetto nei messaggi da Far-End a Near-End, riportati in figura 39, infatti durante le fasi di “down” del link questi vengono persi prima di giungere in prossimità del Near-End, dove è posto l’analizzatore di traffico.

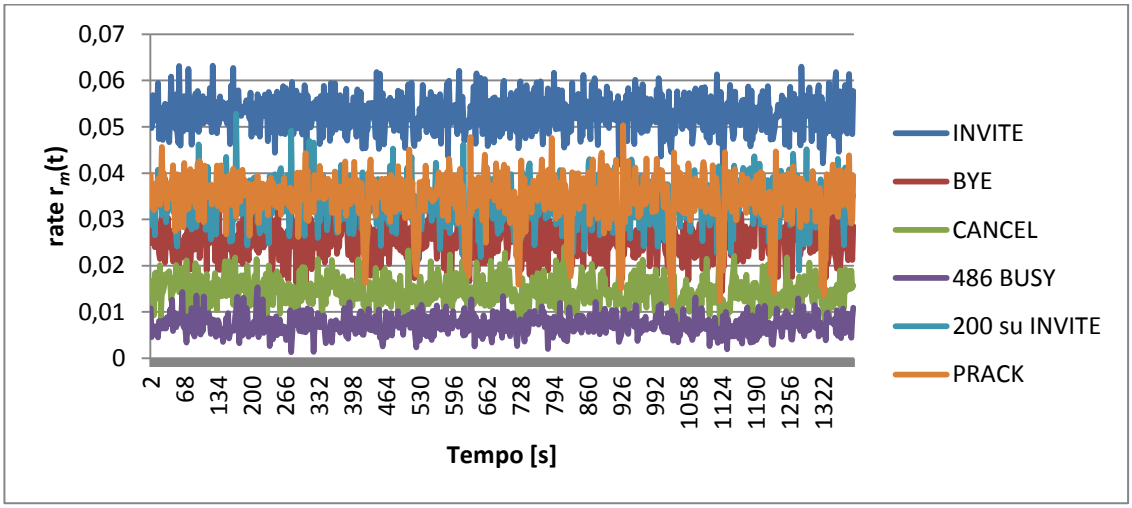


Figura 39 - Profilo temporale del traffico da Far-End a Near-End per test "flap 40 cps"

In figura 40 è riportata la curva di ROC dei tre ADS implementati, campionata con ν=0,025, ν=0,04, ν=0,08, ν=0,12, ν=0,20, ν=0,40, ν=0,60 e ν=0,80. I punti relativi all'ADS basato sulle soglie sono individuati sul grafico da triangoli verdi, quelli della tecnica che sfrutta la sola Support Vector Machine sono rappresentati con dei quadrati rossi, mentre la tecnica che combina Wavelet e SVM da dei rombi azzurri. Nel grafico, i punti più a sinistra sono ottenuti con i valori più bassi di ν, mentre quelli ottenuti con valori più alti si collocano progressivamente verso la destra, ovvero la zona con valori di FPR più elevati.

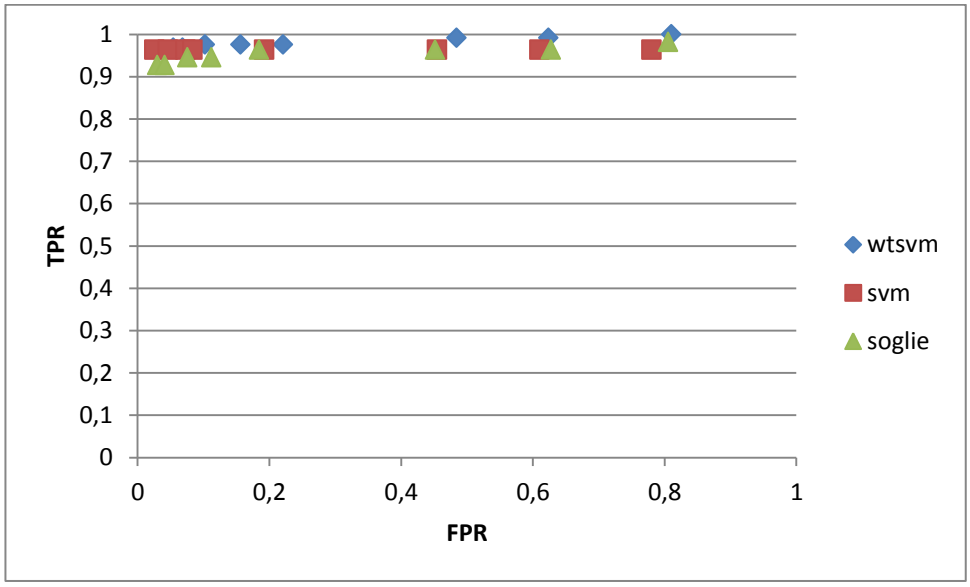


Figura 40 - Curva di ROC per test "flap 40 cps"

Dalla curva di ROC si nota che le dieci anomalie riprodotte vengono ben identificate da tutti e tre gli ADS implementati; si ottengono infatti dei True Positive Ratio piuttosto elevati, prossimi all'unità, mantenendo la percentuale di falsi positivi entro valori piuttosto bassi se si utilizza ν=0,025, dell'ordine di qualche punto percentuale. Nella tabella 8 sono riportati gli esatti valori di TPR ed FPR ottenuti dalle tre tecniche con i diversi valori di ν considerati. Questi sono le coordinate dei punti rappresentati nella curva di ROC.

% outliers	wtsvm		svm		soglie	
	FPR	TPR	FPR	TPR	FPR	TPR
2,50%	0,05	0,968	0,02	0,96	0,02	0,92
4%	0,06	0,968	0,04	0,96	0,04	0,92
8%	0,10	0,976	0,07	0,96	0,07	0,94
12%	0,15	0,976	0,08	0,96	0,11	0,94
20%	0,22	0,976	0,19	0,96	0,18	0,96
40%	0,48	0,992	0,45	0,96	0,45	0,96
60%	0,62	0,992	0,61	0,96	0,62	0,96
80%	0,81	1	0,77	0,96	0,80	0,98

Tabella 8 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Flap 40 cps"

Un ulteriore significativo indice di prestazione è il False Discovery Rate, ovvero la percentuale di falsi positivi rispetto al totale delle segnalazioni di anomalia fornite dall'ADS. In tabella 9 è riportato il valore di tale indice per le tre implementazioni considerate, calcolato rispetto all'output degli ADS quando questi sono addestrati con ν=0,025.

	wtsvm	svm	soglie
FDR	0,19	0,23	0,27

Tabella 9 - Valori di FDR per esperimento "Flap 40 cps"

La tecnica "Wavelet-SVM" è quella che fornisce il valore più basso di questo parametro, ovvero che segnala meno falsi positivi rispetto al numero di istanze segnalate anomale.

6.2.2 Esperimento a 20 cps

Rispetto al test appena presentato, questa prova è stata effettuata riproducendo un volume inferiore di traffico sul trunk, ovvero con generazione di chiamate da parte dei client SIPp a 20 cps.

La traccia generata è lunga 1290 secondi e presenta dieci anomalie in altrettanti intervalli temporali, riportati nella tabella sottostante (tabella 10).

ANOMALIA	INIZIO [s]	FINE [s]
1	306	315
2	406	415
3	506	515
4	606	615
5	706	715
6	806	815
7	906	915
8	1006	1015
9	1106	1115
10	1206	1215

Tabella 10 - Intervalli temporali anomali nell'esperimento "Flap 20 cps"

L'effetto del flap del link sul traffico SIP risulta immutato. Di seguito sono riportati i profili del traffico per i messaggi da Near-End a Far-End (figura 41) e viceversa (figura 42).

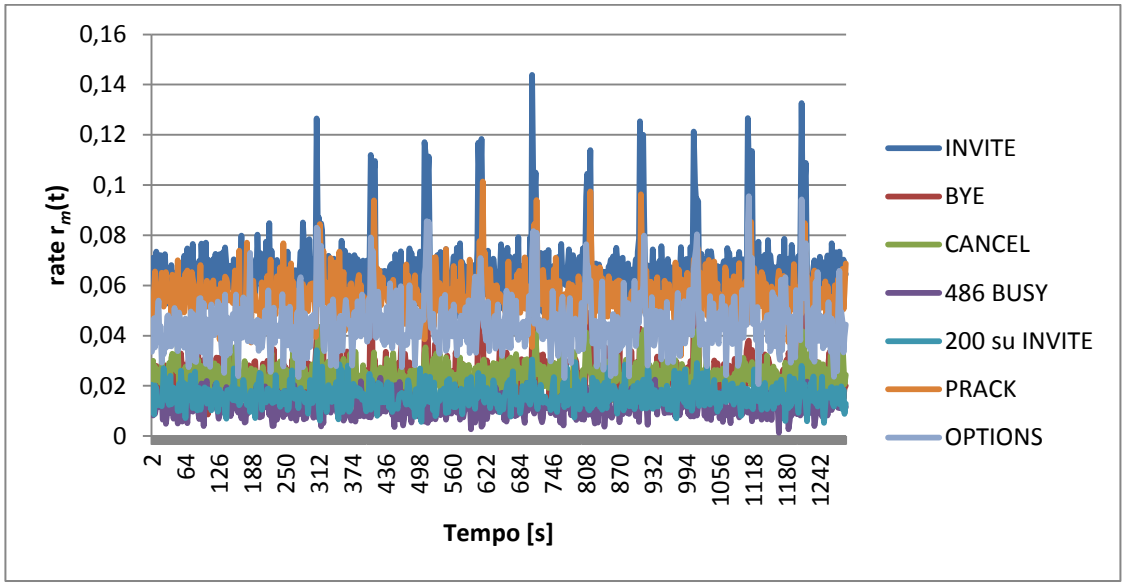


Figura 41 - Profilo temporale del traffico da Near-End a Far-End per test "flap 20"

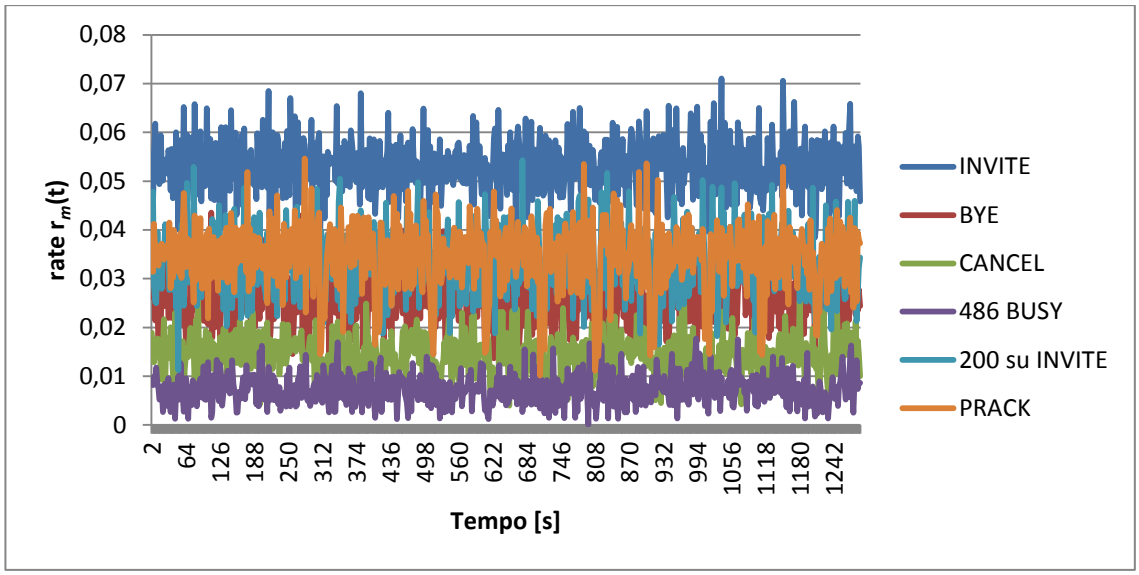


Figura 42 - Profilo temporale del traffico da Far-End a Near-End per test "flap 20"

La curva di ROC dei tre ADS, riportata in figura 43, conferma le buone prestazioni nel rilevamento di questo tipo di anomalia. In questo caso i dieci "link-flap" riprodotti sono identificati con precisione, TPR = 1, già con valori di FPR bassi.

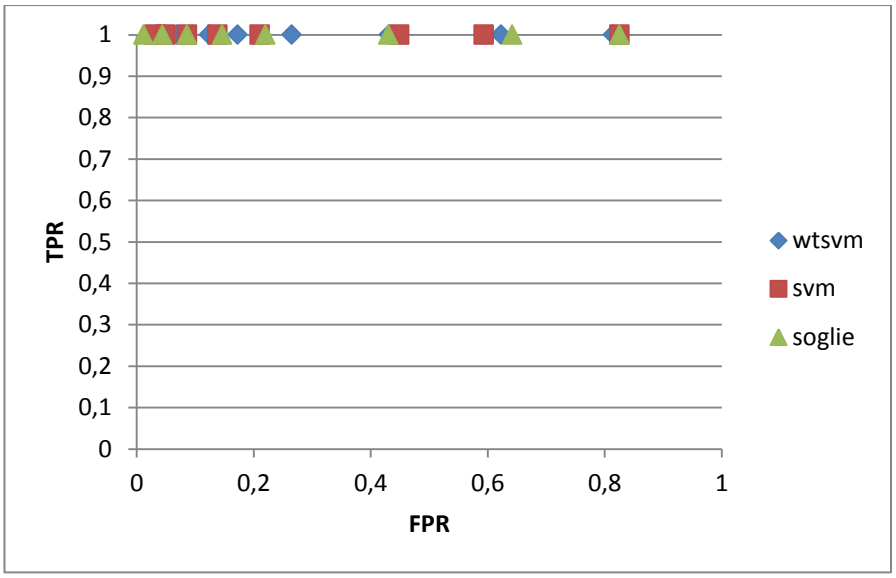


Figura 43 - Curva di ROC per test "flap 20"

La tabella 11 riporta le esatte coordinate dei punti individuati nella curva di ROC.

% outliers	wtsvm		svm		soglie	
	FPR	TPR	FPR	TPR	FPR	TPR
2,50%	0,04	1	0,03	1	0,01	1
4%	0,06	1	0,04	1	0,04	1
8%	0,12	1	0,08	1	0,08	1
12%	0,17	1	0,13	1	0,14	1
20%	0,26	1	0,21	1	0,22	1
40%	0,43	1	0,44	1	0,43	1
60%	0,62	1	0,59	1	0,64	1
80%	0,81	1	0,82	1	0,82	1

Tabella 11 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Flap 20 cps"

A questo volume di traffico è il classificatore a soglia che registra la miglior prestazione in termini di FDR (tabella 12). Infatti ottiene la percentuale inferiore di falsi positivi, rispetto al totale delle segnalazioni di anomalia effettuate, in confronto alle altre tecniche.

	wtsvm	svm	soglie
FDR	0,17	0,26	0,12

Tabella 12 - Valori di FDR per esperimento "Flap 20 cps"

6.2.3 Esperimento con volume di traffico presente su SIP Trunk reale

In questa prova è stato riprodotto il volume di traffico solitamente presente sul trunk telefonico da cui sono stati estratti i parametri dell'emulatore; i rate sono inferiori rispetto a quelli utilizzati nelle prove presentate in precedenza. Le chiamate entranti nella rete Enterprise, ovvero da Far-End a Near-End, sono generate a 2,5 cps, mentre le chiamate uscenti a 7 cps. Per semplicità i test svolti a questo volume di traffico sono indicati con "Tr".

Anche in questo caso sono state riprodotte dieci anomalie di tipo "flap" negli intervalli temporali indicati dalla tabella 13. La traccia generata è lunga 1578 secondi.

ANOMALIA	INIZIO [s]	FINE [s]
1	492	501
2	593	602
3	693	702
4	793	802
5	893	902
6	993	1002
7	1093	1102
8	1193	1202
9	1293	1302
10	1393	1402

Tabella 13 - Intervalli temporali anomali nell'esperimento "Flap Tr"

Gli effetti del flap sul traffico rimangono qualitativamente immutati, ma risultano più contenuti. Il profilo temporale del traffico è riportato nelle figure 44 e 45.

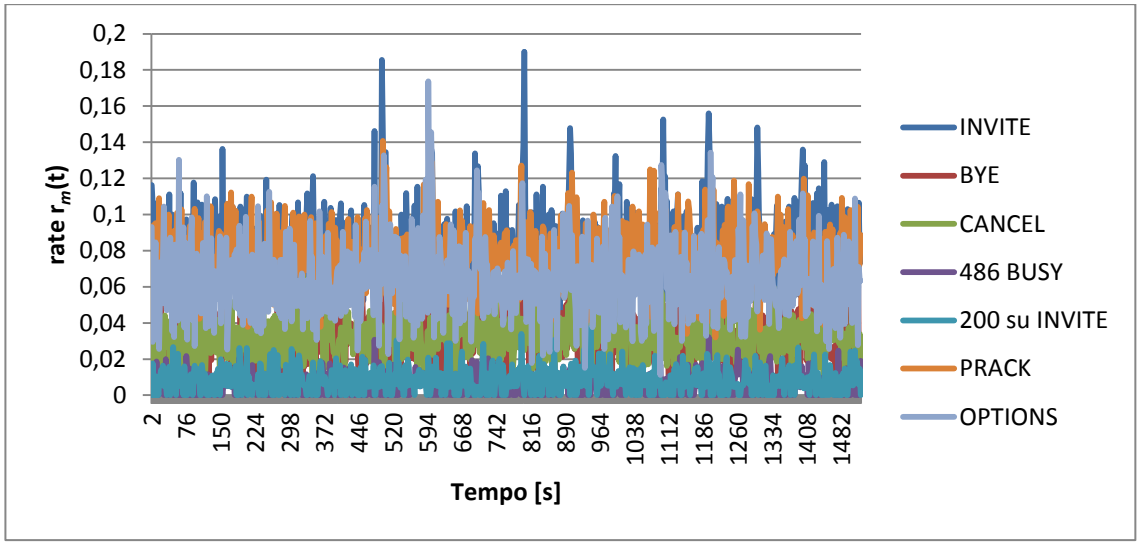


Figura 44 - Profilo temporale del traffico da Near-End a Far-End per test "flap Tr"

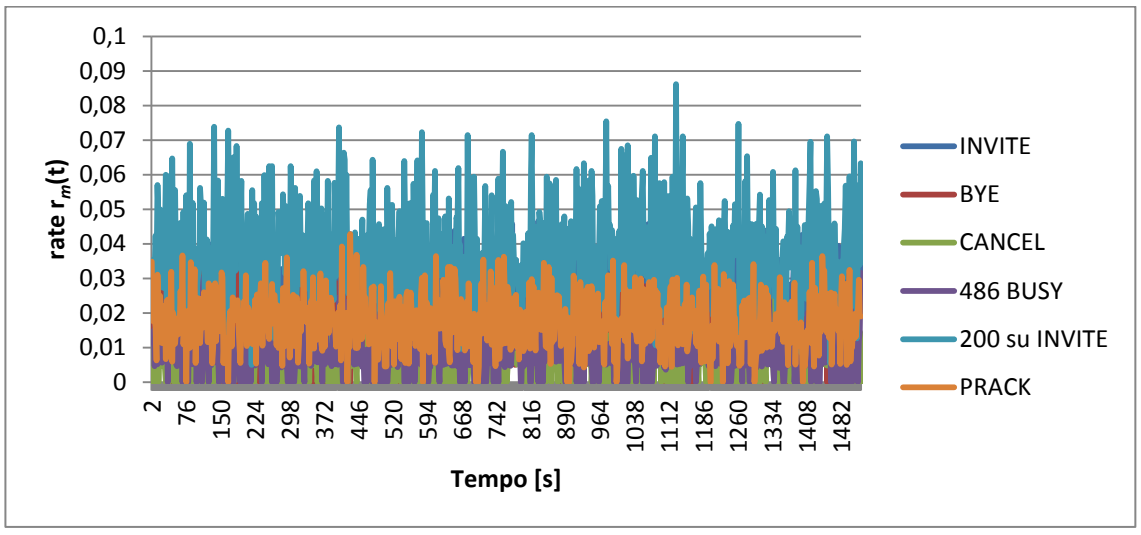


Figura 45 - Profilo temporale del traffico da Near-End a Far-End per test "flap Tr"

Dato il basso volume di traffico, alcuni parametri risultano spesso nulli all'interno dei singoli slot, come ad esempio i "486 BUSY" uscenti o i "CANCEL" entranti. Per questo motivo non è possibile utilizzare la prima tecnica implementata, in quanto non si riescono ad individuare delle soglie appropriate per escludere basse percentuali di outliers attraverso l'algoritmo presentato. Infatti già al primo "step" dell'algoritmo la soglia inferiore per il parametro relativo ai "CANCEL" entranti, preso d'esempio, è superiore allo zero, escludendo gran parte degli slot della traccia.

La prova effettuata evidenzia come le anomalie risultino, in questo caso, più difficili da individuare. La tecnica che utilizza la sola SVM non fornisce buone prestazioni, questo perché i valori dei parametri all'interno degli intervalli di anomalia sono prossimi ai valori di normale funzionamento; l'ADS prende una decisione sulla base di un singolo slot, individuando solo alcuni slot all'interno del periodo in cui il sistema sperimenta l'anomalia.

Questo non succede alla tecnica "Wavelet-SVM", che implementando un'analisi a finestra con overlap unitamente alla Wavelet Decomposition, segnala con maggior evidenza la presenza dell'anomalia all'interno del relativo intervallo temporale.

La tabella 14 riporta le coordinate dei punti rappresentati nella curva di ROC in figura 46.

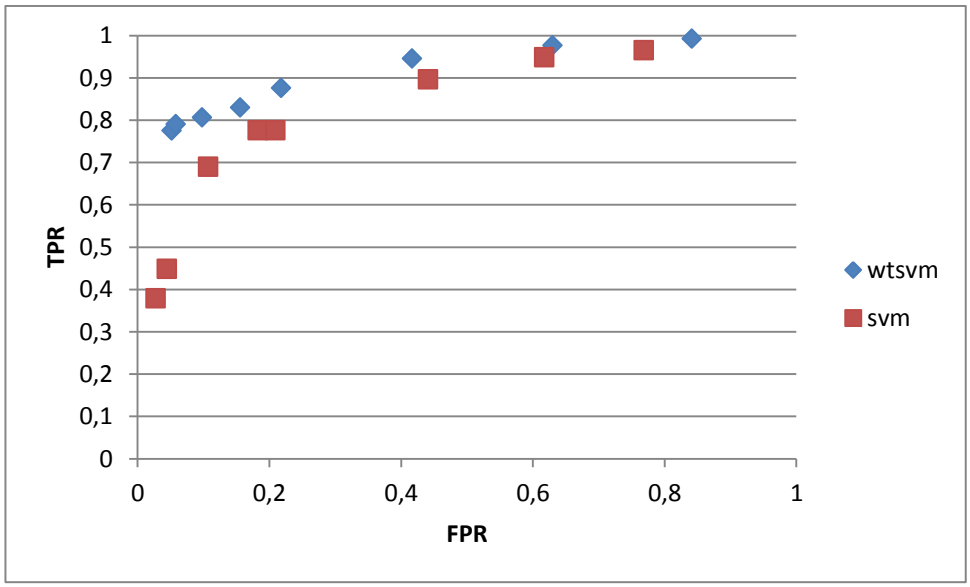


Figura 46 - Curva di ROC per test "flap Tr"

% outliers	wtsvm		svm	
	FPR	TPR	FPR	TPR
2,50%	0,05	0,77	0,02	0,37
4%	0,05	0,79	0,04	0,44
8%	0,09	0,80	0,10	0,68
12%	0,15	0,82	0,18	0,77
20%	0,21	0,87	0,20	0,77
40%	0,41	0,94	0,44	0,89
60%	0,62	0,97	0,61	0,94
80%	0,84	0,99	0,76	0,96

Tabella 14 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Flap Tr"

Il parametro FDR risulta inferiore per la tecnica "Wavelet-SVM", come riportato in tabella 15.

	wtsvm	svm
FDR	0,24	0,46

Tabella 15 - Valori di FDR per esperimento "Flap Tr"

6.3 Emulazione di anomalia "CPU-Hog"

Anche questo tipo di anomalia, descritta nel capitolo precedente, è stata riprodotta nella traccia di test per dieci volte, con intervalli di 90 secondi l'una dall'altra.

Test di performance, in questo caso, sono possibili solamente ad elevati volumi di traffico. Questo perché l'evento anomalo è riprodotto "strozzando" le risorse di CPU a disposizione del processo di gestione delle chiamate, applicando un limite in percentuale intera. Solo per valori elevati di cps generati, il traffico sul trunk risente della carenza di risorse, rendendo significativo il test di performance.

6.3.1 Esperimento a 40 cps

Questo test è stato effettuato con generazione di chiamate uscenti ed entranti entrambe a 40 cps, mentre la capacità di CPU a disposizione del processo di gestione delle chiamate, durante la riproduzione dell'evento anomalo "cpu-hog", è limitata al 2%. In condizioni di normale funzionamento e a questo volume di traffico, entrambi i processi SIPp attivi sulla macchina che rappresenta il Far-End richiedono mediamente il 6%.

La traccia generata è lunga 1206 secondi, all'interno dei quali sono state riprodotte dieci anomalie "hog" negli intervalli temporali riportati in tabella 16.

ANOMALIA	INIZIO [s]	FINE [s]
1	204	214
2	304	314
3	404	414
4	504	514
5	604	614
6	704	714
7	804	814
8	904	914
9	1004	1014
10	1104	1114

Tabella 16 - Intervalli temporali anomali nell'esperimento "Hog 40 cps"

Questo tipo di anomalia, ovvero la carenza di risorse per la gestione delle chiamate, si ripercuote principalmente sulle risposte SIP da Far-End a Near-End, trasmesse con ritardo e a burst. Per questo motivo si registrano ritrasmissioni nelle richieste SIP in senso opposto, in quanto le relative risposte non giungono entro i timer di ritrasmissione SIP. Di seguito sono riportati i profili del traffico in entrambi i versi di percorrenza del trunk (figura 47 e 48).

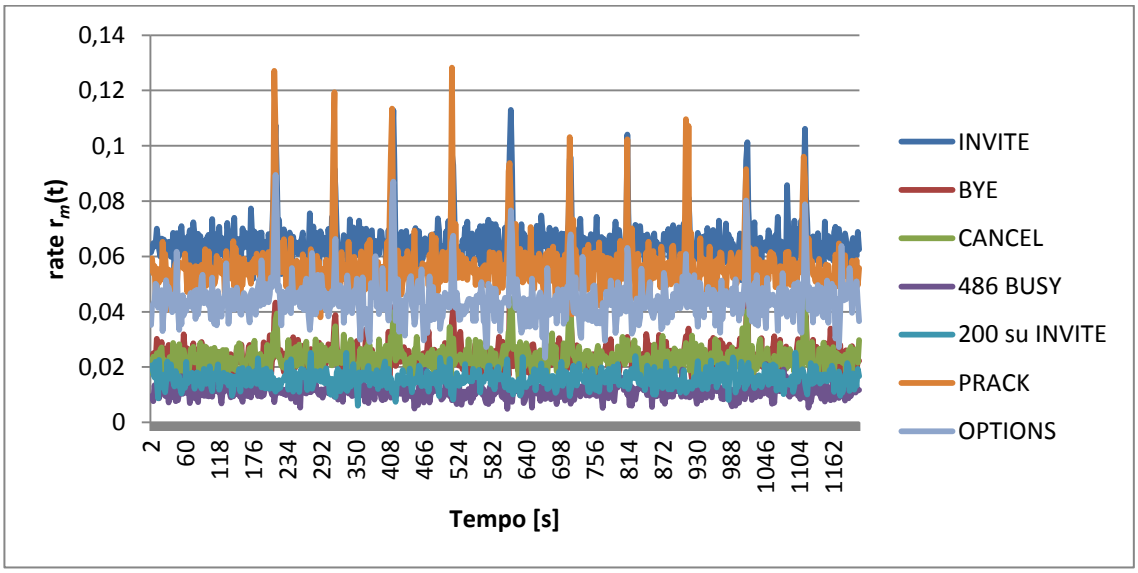


Figura 47 - Profilo temporale del traffico da Near-End a Far-End per test "hog 40 cps"

In verso opposto, da Far-End a Near-End, si notano dei picchi per difetto nei profili delle richieste SIP, indicanti la presenza di istanti in cui il processo di gestione delle chiamate rallenta notevolmente l'invio di messaggi.

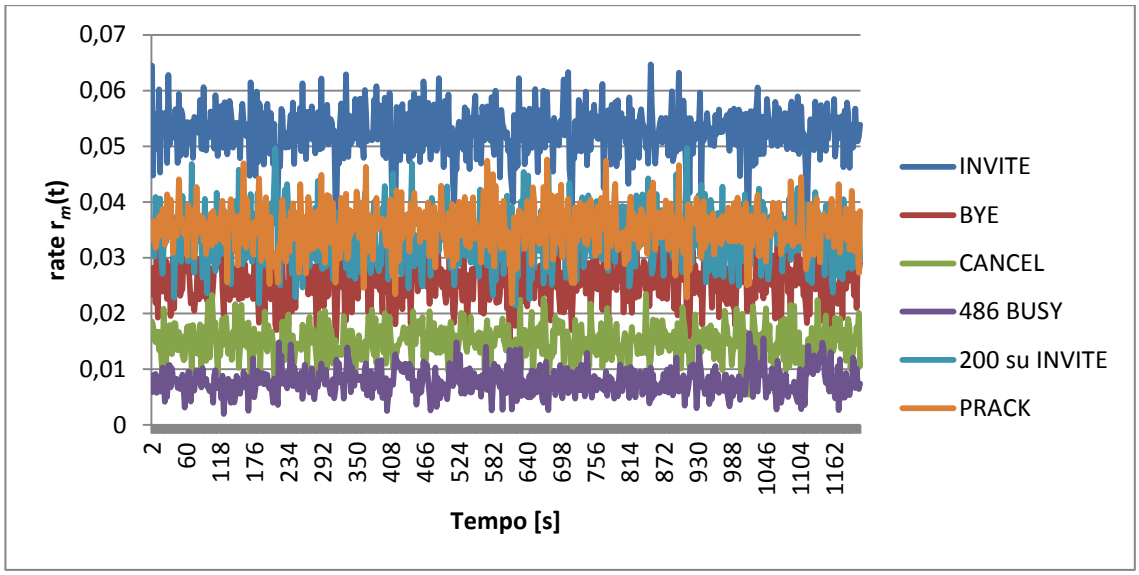


Figura 48 - Profilo temporale del traffico da Far-End a Near-End per test "hog 40 cps"

In questo caso, nella curva di ROC di figura 49, si nota come gli ADS basati sulle soglie e sulla sola Support Vector Machine forniscono prestazioni pressoché equivalenti, mentre l'ADS che utilizza la trasformata Wavelet ottiene valori di TPR più elevati, seppur con valori di FPR maggiori. Tutte le tre tecniche proposte identificano la presenza di un anomalia all'interno del relativo intervallo, ma la tecnica "Wavelet-SVM" riscontra un numero maggiore di slot anomali rispetto al totale di slot temporalmente interessati dall'anomalia stessa.

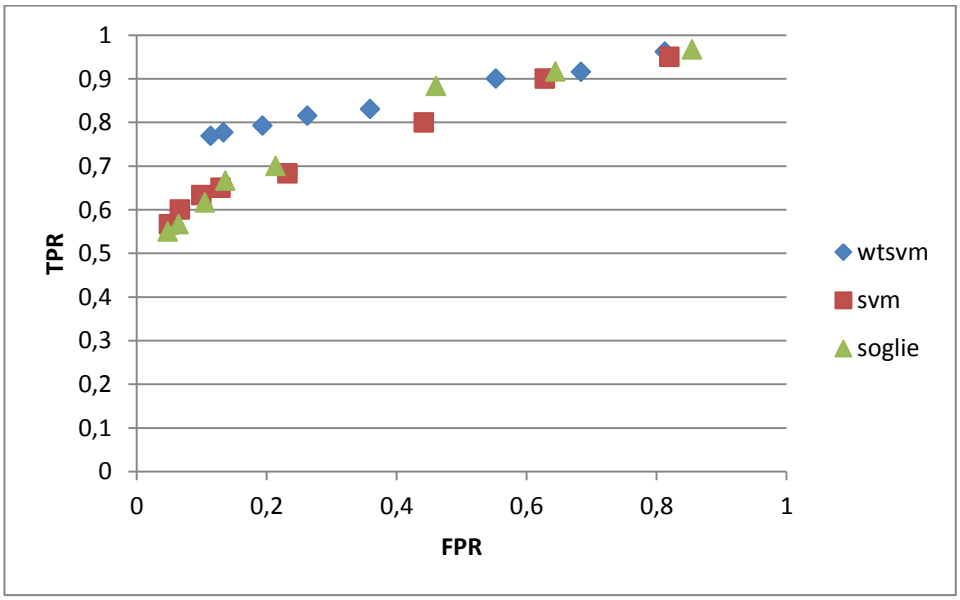


Figura 49 - Curva di ROC per test "hog 40 cps"

La tabella 17 riporta le coordinate dei punti riportati nella curva di ROC.

% outliers	wtsvm		svm		soglie	
	FPR	TPR	FPR	TPR	FPR	TPR
2,50%	0,11	0,76	0,04	0,56	0,04	0,55
4%	0,13	0,77	0,06	0,6	0,06	0,56
8%	0,19	0,79	0,09	0,63	0,10	0,61
12%	0,26	0,81	0,12	0,65	0,13	0,66
20%	0,35	0,83	0,23	0,68	0,21	0,7
40%	0,55	0,9	0,44	0,8	0,46	0,88
60%	0,68	0,91	0,62	0,9	0,64	0,91
80%	0,81	0,96	0,81	0,95	0,85	0,96

Tabella 17 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Hog 40 cps"

In questo caso non si raggiungono valori di TPR prossimi all'unità come per la precedente tipologia di anomalia. Questo è dovuto principalmente al fatto che l'anomalia si manifesta sul traffico con ritardo rispetto al momento in cui vengono ridotte le risorse a disposizione del processo di gestione delle chiamate, generando falsi negativi. Inoltre si registrano valori di FPR più elevati a causa del protrarsi nel tempo dell'effetto dell'anomalia sul traffico. Anche quando viene meno la strozzatura delle risorse al sistema, occorre del tempo, quantificabile in qualche secondo, prima che l'effetto dell'anomalia sia "riassorbito".

	wtsvm	svm	soglie
FDR	0,32	0,44	0,44

Tabella 18 - Valori di FDR per esperimento "Hog 40 cps"

Anche in questo esperimento la tecnica "Wavelet-SVM" ottiene il valore più basso di FDR rispetto alle altre due (tabella 18), seppur in questo caso tale valore sia quasi il doppio in confronto a quello calcolato nelle prove presentate in precedenza.

6.4 Emulazione di anomalia "Loop"

Per testare le performance degli ADS nell'individuazione di loop, sono state generate tracce a diverso call-rate in cui sono presenti 25 chiamate in loop, una alla volta; tali chiamate vengono effettuate a intervalli regolari di 40 secondi più il tempo di avvio del client SIPp che genera di volta in volta una chiamata verso una numerazione in loop, stimabile in circa 7 secondi.

6.4.1 Esperimento a 40 cps

Questa prova è effettuata ad un rate di 40 cps in entrambe le direzioni, generando una traccia di lunghezza 1544 secondi, in cui sono presenti 25 chiamate in loop, localizzate temporalmente negli intervalli riportati in tabella 19.

ANOMALIA	INIZIO	FINE	ANOMALIA	INIZIO	FINE
1	320	320	14	947	947
2	368	368	15	995	995
3	416	417	16	1043	1044
4	464	465	17	1091	1092
5	513	513	18	1139	1140
6	561	561	19	1188	1188
7	609	609	20	1236	1236
8	657	658	21	1284	1285
9	706	706	22	1332	1333
10	754	754	23	1380	1381
11	802	802	24	1429	1429
12	850	851	25	1477	1477
13	898	899			

Tabella 19 - Intervalli temporali anomali nell'esperimento "Loop 40 cps"

La presenza di una chiamata in loop ha l'unico effetto di aumentare i messaggi INVITE osservati dall'analizzatore di traffico. Questo si traduce nella presenza di picchi superiori nel profilo di traffico dei messaggi INVITE in corrispondenza degli slot in cui si verifica il loop.

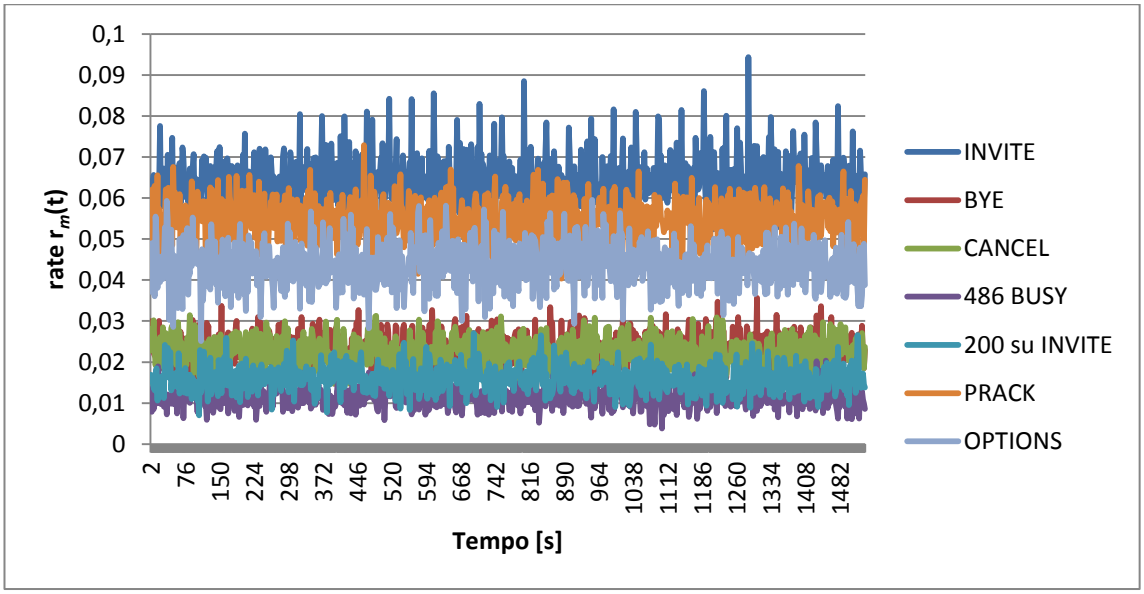


Figura 50 - Profilo temporale del traffico da Near-End a Far-End per test "loop 40 cps"

Tale effetto si presenta in entrambi i versi di percorrenza del trunk (figura 50 e 51).

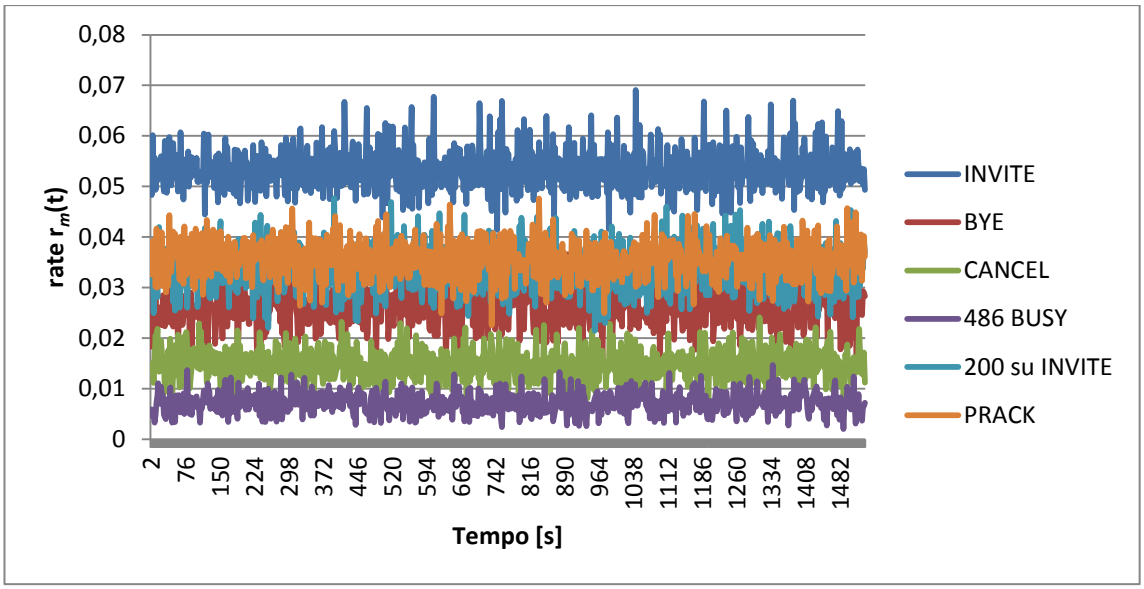


Figura 51 - Profilo temporale del traffico da Far-End a Near-End per test "loop 40 cps"

In questo caso, a bassi valori di FPR le tecniche che utilizzano le soglie e la sola SVM forniscono valori di TPR rispettivamente intorno a 0,85 e 0,75. Infatti decidendo sulla base del valore di un singolo slot alla volta sono in grado di individuare la maggior parte delle anomalie riprodotte. La tecnica "Wavelet-SVM" invece riesce ad individuarne meno, a causa della piccola variazione che tali picchi introducono nell'andamento degli "INVITE" nella finestra di osservazione. La curva di ROC relativa a questo esperimento è riportata in figura 52.

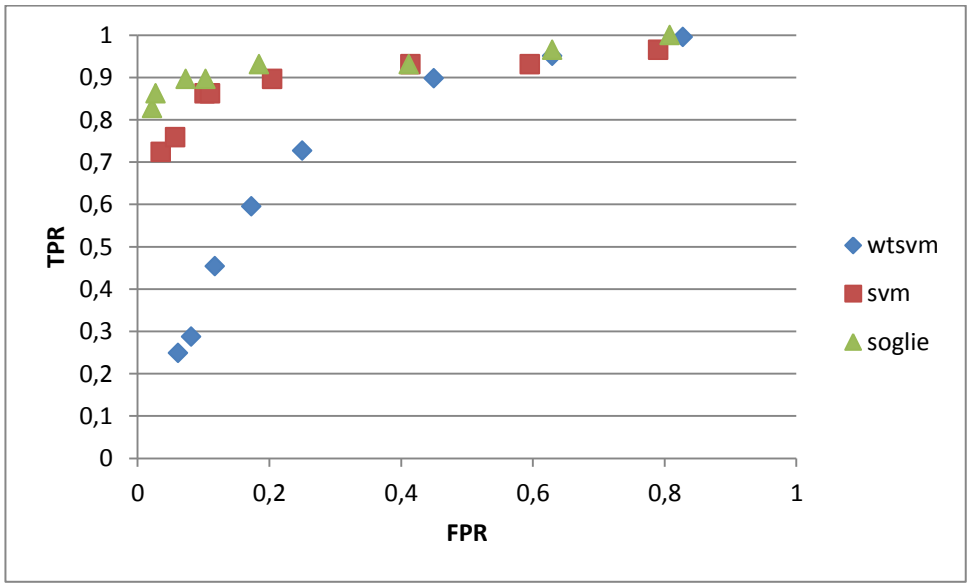


Figura 52 - Curva di ROC per test "loop 40 cps"

In tabella 20 sono riportate le coordinate dei punti individuati nella curva di ROC.

% outliers	wtsvm		svm		soglie	
	FPR	TPR	FPR	TPR	FPR	TPR
2,50%	0,06	0,24	0,03	0,72	0,02	0,82
4%	0,08	0,28	0,05	0,75	0,02	0,86
8%	0,11	0,45	0,10	0,86	0,07	0,89
12%	0,17	0,59	0,10	0,86	0,10	0,89
20%	0,25	0,72	0,20	0,89	0,18	0,93
40%	0,44	0,89	0,41	0,93	0,41	0,93
60%	0,62	0,95	0,59	0,93	0,62	0,96
80%	0,82	0,99	0,79	0,96	0,80	1

Tabella 20 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Loop 40 cps"

In questo esperimento il valore di FDR delle tre tecniche, calcolato per il test con ν=0,025 e riportato nella tabella 21, è decisamente elevato.

	wtsvm	svm	soglie
FDR	0,4	0,55	0,4

Tabella 21 - Valori di FDR per esperimento "Loop 40 cps"

6.4.2 Esperimento a 20 cps

In questo test entrambi i generatori di chiamata dell'emulatore lavorano a 20 cps. La traccia generata è lunga 1730 secondi e le 25 anomalie riprodotte al suo interno sono localizzate temporalmente secondo la tabella che segue (tabella 22).

ANOMALIA	INIZIO	FINE	ANOMALIA	INIZIO	FINE
1	515	516	14	1137	1138
2	563	564	15	1185	1186
3	611	612	16	1233	1233
4	659	659	17	1281	1281
5	707	707	18	1329	1329
6	755	755	19	1377	1377
7	802	803	20	1424	1425
8	850	851	21	1472	1473
9	898	899	22	1520	1521
10	946	946	23	1568	1568
11	994	994	24	1616	1616
12	1042	1042	25	1664	1664
13	1089	1090			

Tabella 22 - Intervalli temporali anomali nell'esperimento "Loop 20 cps"

In questa situazione, ovvero con mediamente 20 messaggi "INVITE" al secondo in entrambi i sensi di percorrenza del trunk, l'anomalia loop è causa di picchi più accentuati rispetto al caso precedente, in cui gli "INVITE" al secondo sono il doppio (figura 53 e 54).

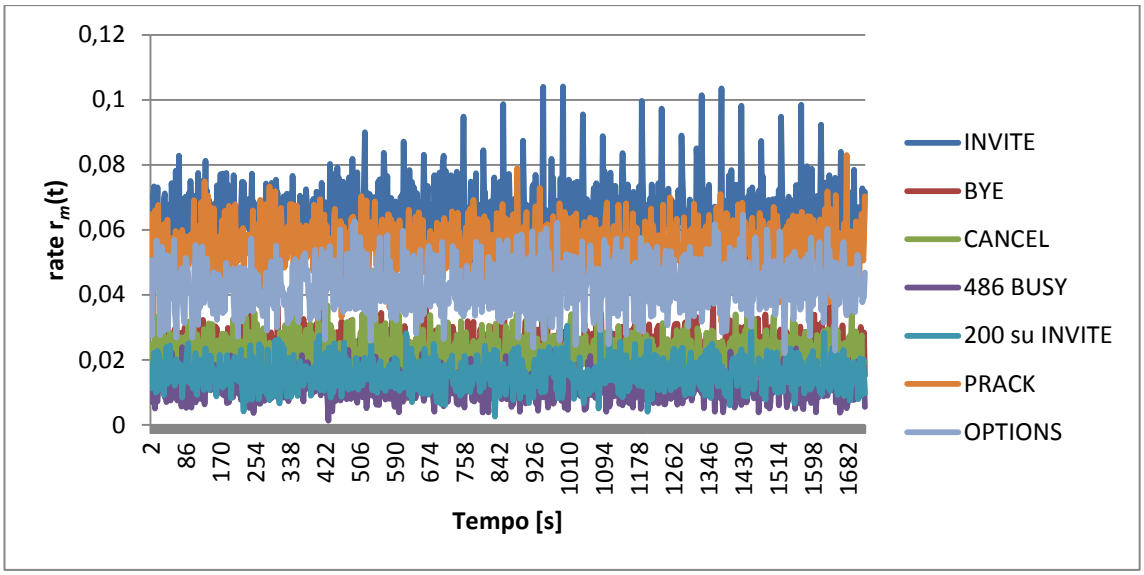


Figura 53 - Profilo temporale del traffico da Near-End a Far-End per test "loop 20 cps"

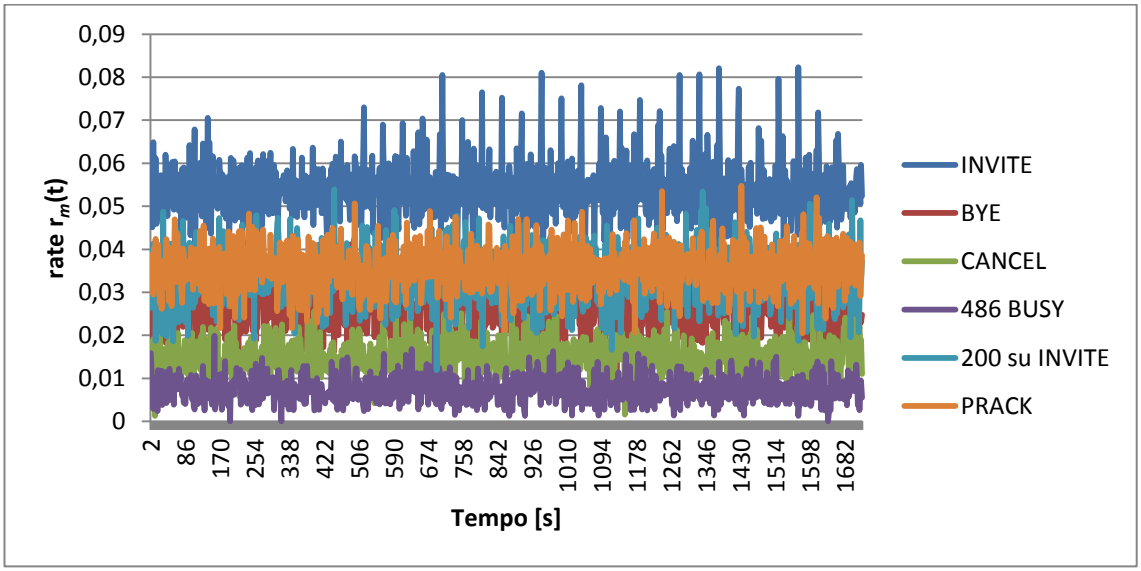


Figura 54 - Profilo temporale del traffico da Far-End a Near-End per test "loop 20 cps"

Le prestazioni degli ADS in questo caso sono equiparabili; a bassi valori di FPR la tecnica “SVM” registra un TPR di 0,6 , mentre le altre due 0,5. Non tutte le 25 anomalie sono rilevate dagli ADS, alcune di esse causano dei picchi che non superano quelli presenti regolarmente nella traccia di addestramento. Di seguito è illustrata la curva di ROC relativa a questo esperimento (figura 55); le esatte coordinate dei punti individuati in essa sono contenute nella tabella 23.

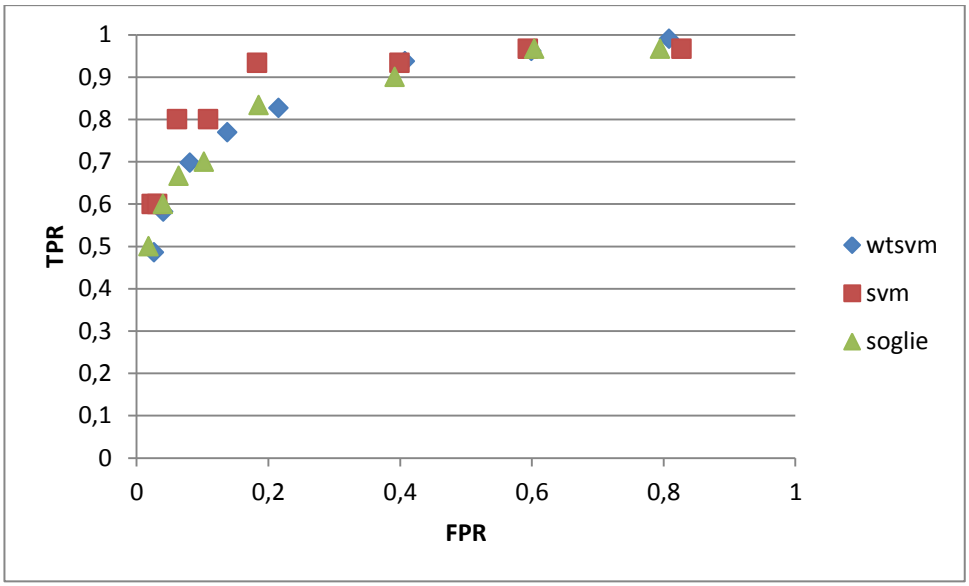


Figura 55 - Curva di ROC per test "loop 20cps"

% outliers	wtsvm		svm		soglie	
	FPR	TPR	FPR	TPR	FPR	TPR
2,50%	0,02	0,48	0,02	0,6	0,01	0,5
4%	0,04	0,58	0,03	0,6	0,03	0,6
8%	0,08	0,69	0,06	0,8	0,06	0,66
12%	0,13	0,76	0,10	0,8	0,10	0,7
20%	0,21	0,82	0,18	0,93	0,18	0,83
40%	0,40	0,93	0,39	0,93	0,39	0,9
60%	0,59	0,96	0,59	0,96	0,60	0,96
80%	0,80	0,99	0,82	0,96	0,79	0,96

Tabella 23 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Loop 20 cps"

Rispetto al caso precedente, mentre il valore di FDR delle tecniche "SVM" e "Soglie" rimangono elevati, il valore di FDR della tecnica "Wavelet-SVM" risulta di molto inferiore; questo perché a 20 cps tale tecnica registra valori di TPR più elevati a fronte di un minor numero di falsi positivi rispetto all'esperimento a 40 cps. I valori di FDR riportati nella tabella 24 sono relativi ai test effettuati con ν=0,025.

	wtsvm	svm	soglie
FDR	0,14	0,51	0,5

Tabella 24 - Valori di FDR per esperimento "Loop 20 cps"

6.4.3 Esperimento con volume di traffico presente su SIP Trunk reale

Il volume di traffico generato per questa prova è 7 cps per le chiamate uscenti, mentre 2,5 per quelle entranti. Nella traccia, lunga 1598 secondi, sono riprodotte 25 chiamate in loop negli intervalli temporali riportati di seguito (tabella 25).

ANOMALIA	INIZIO	FINE	ANOMALIA	INIZIO	FINE
1	396	397	14	1014	1014
2	444	444	15	1061	1062
3	491	492	16	1109	1109
4	539	539	17	1156	1157
5	586	587	18	1204	1204
6	634	634	19	1251	1252
7	681	682	20	1299	1299
8	729	729	21	1346	1347
9	776	777	22	1394	1394
10	824	824	23	1441	1442
11	871	872	24	1489	1489
12	919	919	25	1536	1537
13	966	967			

Tabella 25 - Intervalli temporali anomali nell'esperimento "Loop Tr"

Come nel caso precedente, lavorando a cps più bassi le anomalie "loop" producono picchi più accentuati nell'andamento degli "INVITE" (figura 56 e 57).

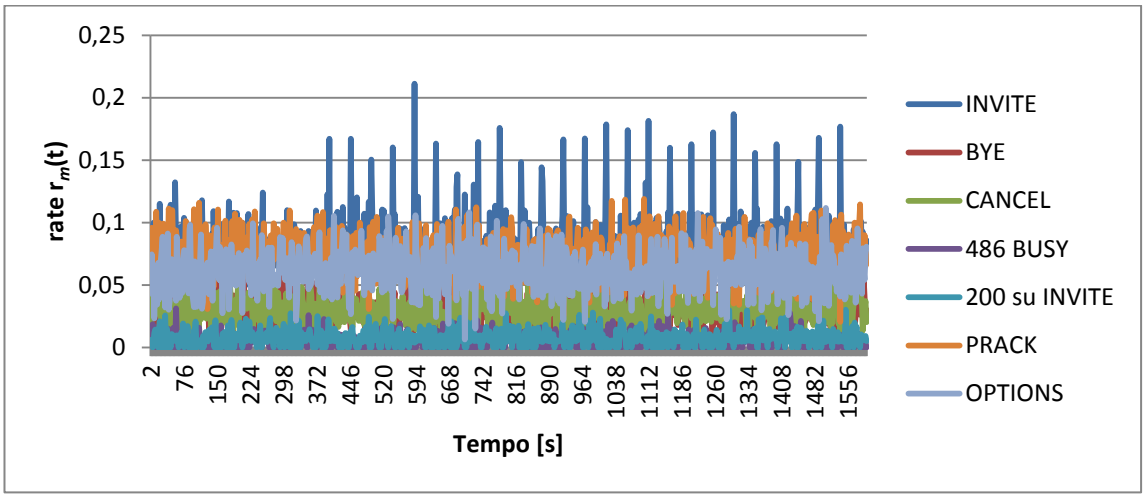


Figura 56 - Profilo temporale del traffico da Near-End a Far-End per test "loop Tr"

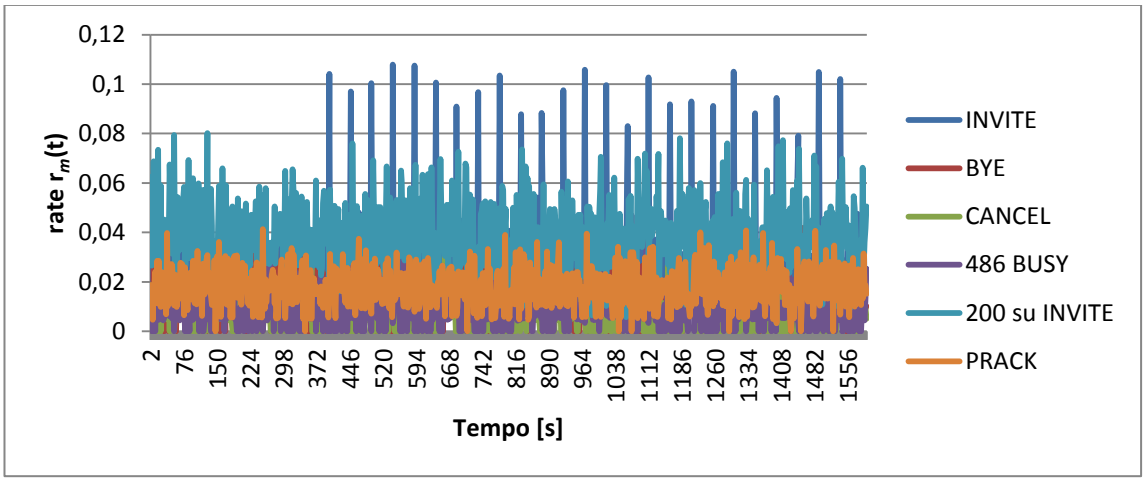


Figura 57 - Profilo temporale del traffico da Far-End a Near-End per test "loop Tr"

Sia la tecnica “SVM” che “Wavelet-SVM” forniscono ottimi risultati già a basse percentuali di falsi positivi, raggiungendo valori di TPR prossimi all’unità (figura 58). Tutte le anomalie riprodotte sono individuate correttamente da entrambi gli ADS.

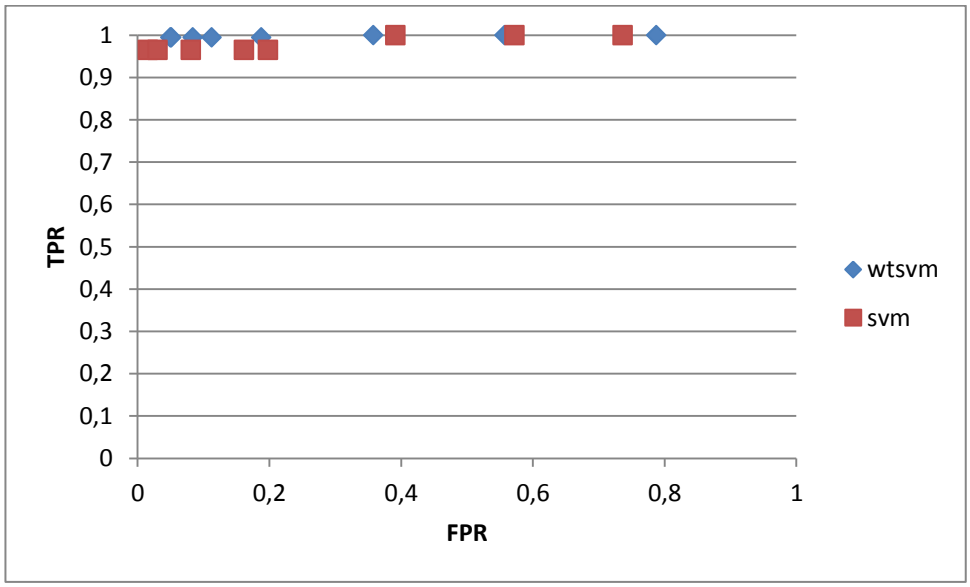


Figura 58 - Curva di ROC per test "loop Tr"

Nella tabella sottostante (tabella 26) sono riportate le coordinate dei punti individuati in nella curva di ROC.

% outliers	wtsvm		svm	
	FPR	TPR	FPR	TPR
2,50%	0,04	0,99	0,01	0,96
4%	0,05	0,99	0,02	0,96
8%	0,08	0,99	0,08	0,96
12%	0,11	0,99	0,16	0,96
20%	0,18	0,99	0,19	0,96
40%	0,35	1	0,39	1
60%	0,55	1	0,57	1
80%	0,78	1	0,73	1

Tabella 26 - Valori di TPR ed FPR al variare della percentuale di outliers nell’esperimento “Loop Tr”

Anche a questo volume di traffico la tecnica “Wavelet-SVM” offre il valore inferiore di FDR rispetto alla tecnica “SVM”. I valori, ottenuti con ν=0,025, sono riportati nella seguente tabella (tabella 27).

	wtsvm	svm
FDR	0,12	0,3

Tabella 27 - Valori di FDR per esperimento "Loop Tr"

Gli effetti dovuti ad un'anomalia di tipo "link-flap" tendono a ripercuotersi sui parametri volumetrici estratti in modo maggiore a volumi di traffico più consistenti; infatti le anomalie di questo tipo introdotte negli esperimenti a 40 e 20 cps vengono ben identificate da tutti i tre metodi proposti già con percentuali di outliers dell'ordine di qualche punto percentuale, ad esempio 2,5%, ovvero addestrando le SVM con ν=0,025. A volumi di traffico più contenuti, invece, i picchi nel profilo temporale dei parametri, dovuti alle ritrasmissioni dei messaggi a causa del "link-flap", sono prossimi ai picchi dovuti alla variabilità del numero di messaggi osservati in situazione di normale esercizio del sistema. Infatti la tecnica che utilizza la sola SVM per una decisione sulla base del valori dei parametri osservati in un singolo slot temporale, pur identificando come anomalo almeno uno slot all'interno di ogni intervallo in cui vi è riprodotta un'anomalia, non è in grado di identificare la maggior parte degli slot temporali interessati dall'evento anomalo. La tecnica "Wavelet-SVM" ottiene risultati migliori con i valori più bassi di ν, ovvero ν=0,025, ν=0,04 e ν=0,08, individuando come anomali la maggior parte degli slot interessati dall'evento anomalo.

L'esperimento effettuato a 40 cps in cui sono state riprodotte anomalie di tipo "CPU-hog" mostra che la tecnica "Wavelet-SVM" è in grado di fornire TPR più elevati rispetto alle altre due tecniche proposte, le quali offrono prestazioni pressoché equivalenti; questo perché viene identificato un numero maggiore di slot appartenenti agli intervalli temporali anomali.

Gli esperimenti in cui sono state riprodotte anomalie di tipo "loop", illustrati in precedenza, mostrano che questi eventi anomali, a parità di entità, ovvero di numero di chiamate in "loop" concorrenti, sono maggiormente identificabili più è contenuto il volume di traffico presente sul trunk. Questo perché i messaggi monitorati dagli ADS sono considerati in percentuale a tutti i messaggi presenti sul trunk, perciò gli "INVITE" dovuti alle chiamate in loop sbilanciano maggiormente le relative percentuali al decrescere del numero medio di "INVITE" al secondo. La prova effettuata a 40 cps mostra buoni risultati solo per le tecniche "Soglie" e "SVM" che individuano la maggior parte degli slot in cui si verifica la chiamata in "loop", mentre la tecnica "Wavelet-SVM" è in grado di identificarne solamente alcuni. Nella prova effettuata a basso

volume di traffico, 7 cps per le chiamate uscenti e 2,5 cps per quelle entranti, tutte le anomalie riprodotte sono ben identificate.

In generale dalle prove effettuate emerge che con la tecnica “Wavelet-SVM” si ottengono valori di False Discovery Rate più contenuti rispetto alle altre due implementazioni proposte. Questo significa che rispetto al totale delle segnalazioni di anomalia effettuate, la percentuale di falsi positivi presenti è inferiore per tale tecnica.

E’ da sottolineare un vantaggio non indifferente che si ottiene considerando una finestra di campioni con overlap anziché un campione singolo. In caso di anomalie di durata inferiore all’ampiezza temporale dello slot, se l’output di decisione dell’ADS si basa sul singolo slot, i campioni realmente anomali, true positive, e i falsi positivi dovuti all’individuazione di outliers nella traccia di addestramento non sono distinguibili. Non è così nel caso in cui lo slot nel quale è presente l’anomalia concorra alle decisioni di più istanti temporali consecutivi; in questo modo un’anomalia di breve durata dovrebbe ripercuotersi su più decisioni consecutive dell’ADS, difficilmente su una singola.

Empiricamente si nota che i falsi positivi dovuti all’esclusione di campioni buoni dalla traccia di riferimento, ovvero gli outliers, raramente si presentano consecutivamente. Grazie a questo si potrebbe pensare di implementare un algoritmo di post-processing in grado di eliminare una notevole quantità di falsi positivi per la tecnica “Wavelet-SVM”.

7. CONCLUSIONI

In questo lavoro di tesi sono stati proposti tre possibili approcci al problema dell'individuazione di eventi anomali sperimentabili da un sistema SIP Trunk, attraverso l'analisi volumetrica del traffico di segnalazione SIP presente su di esso. Il problema si pone perché gli strumenti di monitoraggio attualmente in esercizio si basano essenzialmente su informazioni di autodiagnosi dei dispositivi di rete; questi garantiscono una buona protezione da guasti principalmente di tipo hardware, mentre eventi anomali di diversa natura vengono difficilmente individuati e spesso sono segnalati direttamente dagli utenti che sperimentano un disservizio.

La prima implementazione proposta è basata sull'applicazione di soglie ai parametri monitorati; la seconda impiega una Support Vector Machine (SVM) in modalità "one-class" mentre la terza implementazione combina l'analisi multirisoluzione dell'andamento dei parametri estratti, mantenendo a valle la SVM "one-class".

Le prestazioni dei tre ADS proposti sono state valutate rispetto all'individuazione di tre tipologie di eventi anomali: "link-flap", "CPU-hog" e "loop", riprodotti all'interno di un ambiente di emulazione di traffico su SIP Trunk, di cui ne è illustrata la realizzazione.

Dagli esperimenti effettuati è emerso che l'ADS che combina l'analisi multirisoluzione con la SVM "one-class" fornisce prestazioni superiori alle altre due implementazioni nell'individuazione di anomalie "CPU-hog" e "link-flap"; è risultato invece meno efficace nella rilevazione di anomalie puntuali, come il "loop" di una numerazione, quando queste si presentano con entità limitata rispetto al volume di traffico presente sul trunk. In quest'ultimo caso le prime due tecniche proposte, che basano la decisione sul valore dei parametri di un singolo campione temporale, ottengono i risultati migliori; tuttavia le segnalazioni relative a questi eventi anomali puntuali risultano indistinguibili dai falsi positivi dovuti all'individuazione di outliers nei dati di addestramento degli ADS.

Nella terza tecnica proposta, invece, l'analisi multirisoluzione è effettuata su una finestra temporale con "overlap", composta da più campioni. In questo modo il singolo slot relativo ad un'anomalia puntuale concorre alla decisione di più istanti consecutivi,

rendendo più facilmente distinguibili i falsi positivi dovuti agli outliers, i quali il più delle volte si presentano isolati.

Un possibile sviluppo di queste tecniche può essere la realizzazione di un algoritmo di "post-processing" per la riduzione dei falsi positivi, a partire dal filtraggio di quelli dovuti all'individuazione di outliers nei dati di addestramento degli ADS.
E' ipotizzabile l'inserimento di parametri addizionali nella struttura degli ADS proposti, individuabili ad esempio entrando nella semantica dei messaggi SIP.
Inoltre sarebbe interessante l'estrazione di informazioni che evidenzino, in caso venga segnalata un'anomalia, l'impatto di ciascun parametro sulla decisione, in modo da affiancare una decisione "soft" riguardo ciascun parametro a quella "hard" fornita dalle attuali implementazioni.

APPENDICI

A. Support Vector Machine

Le Support Vector Machines (SVM) sono un insieme di algoritmi di apprendimento per l'analisi di dati al fine di classificarli. Esistono diverse varianti di SVM, a seconda dello scopo che si vuole perseguire, come la classificazione multi-classe, la classificazione one-class e l'analisi regressiva, ma il principio di funzionamento è comune.

Il funzionamento di base di una SVM [34] è il seguente. Fornito un insieme di dati di training, appartenenti a due classi distinte, l'algoritmo di apprendimento della SVM trova, nello spazio in cui i dati sono proiettati, il piano che separa al meglio i dati appartenenti alle due differenti classi, ovvero il piano più distante dai campioni di training più vicini, chiamati Support Vectors; è riportato un esempio grafico in figura 59. Per fare questo l'algoritmo prevede la soluzione di un problema di ottimizzazione "Quadratic Programming" (QP), ovvero la minimizzazione o la massimizzazione di una funzione quadratica di più variabili soggette a vincoli lineari.

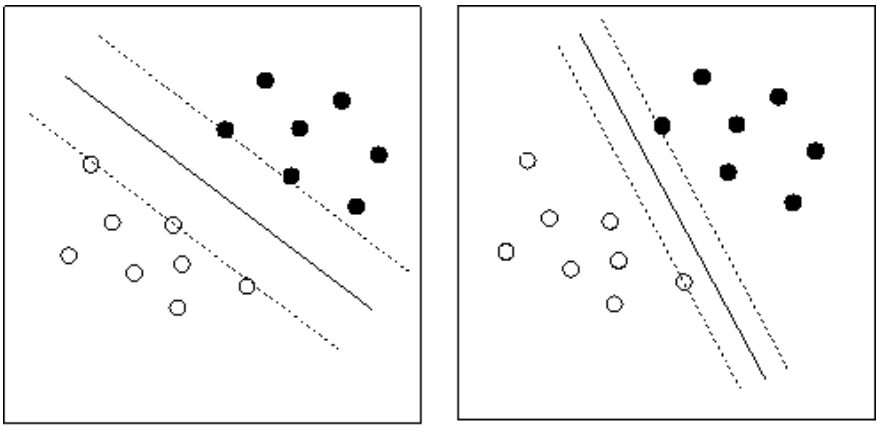


Figura 59 - Modelli di separazione

Talvolta i dati di training non risultano linearmente separabili nello spazio in cui sono definiti (input space). Una SVM può ovviare a questo problema proiettando i dati in uno spazio di dimensionalità maggiore (feature space) in cui essi risultano linearmente separabili (figura 60). Questo è possibile grazie a funzioni non lineari chiamate "kernel". Le funzioni kernel più diffuse sono di tipo polinomiale o gaussiano; funzioni di quest'ultimo tipo sono conosciute anche con il nome Radial Basis Function (RBF).

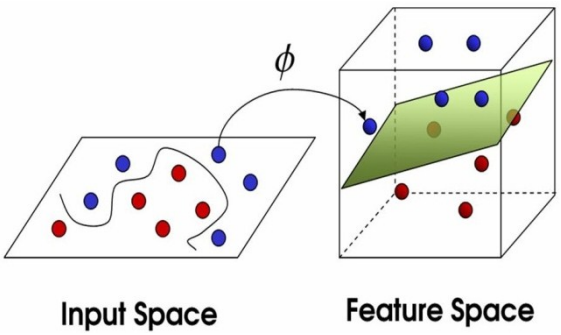


Figura 60 - Applicazione di una funzione kernel ai dati di input di una SVM

In alcuni casi, al fine di ottenere un piano di separazione con margine maggiore, è possibile consentire alla SVM di classificare in modo errato alcuni campioni di training. Questo tipo di variante all'algoritmo si chiama "soft margin" e prevede l'introduzione di un parametro, noto in letteratura come "C", che rappresenta il costo di errore di classificazione; esso bilancia la quantità di errori di classificazione in fase d'addestramento e l'ampiezza del margine. La scelta della giusta funzione di kernel, e del valore dei suoi parametri, oltre al giusto valore di C, è molto importante per ottenere un modello performante, evitando "underfitting" e "overfitting" del problema. Nel primo caso si ottiene un modello troppo semplice, non in grado di separare bene i dati di training nelle rispettive classi (figura 61a sottostante), mentre nel secondo caso si crea un modello troppo dettagliato dei dati di training, rischiando di classificare in modo errato i dati in fase di test (figura 61c sottostante). La giusta soluzione si ottiene con un bilanciamento dei parametri della SVM (figura 61b sottostante).

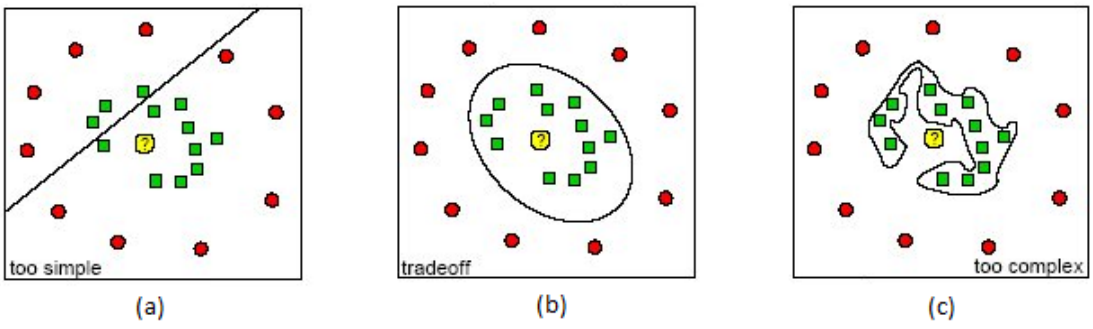


Figura 61 - Esempi di underfitting (a), corretto addestramento (b) e overfitting (c)

Per affrontare problemi di classificazione "one-class", è utilizzata una variante della SVM sopra presentata, nota con il nome v-SVM [35].

In questa versione il dataset di addestramento è composto esclusivamente da campioni appartenenti alla stessa classe. Addestrando una v-SVM con tale dataset, essa costruisce un modello che racchiude tali campioni, lasciando all'esterno una percentuale di essi definita dal parametro v, da cui il nome v-SVM. Nella figura sottostante è riportato il caso di un dataset utilizzato per addestrare una v-SVM con v = 0.01 (figura 62a) e v = 0.5 (figura 62b).

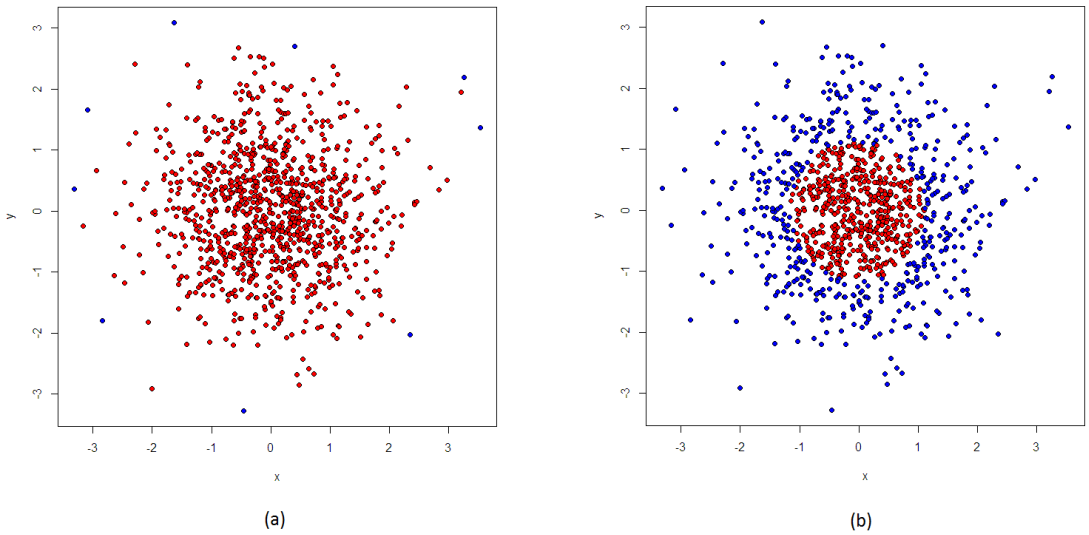


Figura 62 - Training set per l'addestramento di una v-SVM con v = 0.01 (a) e v = 0.5 (b)

B. La trasformata Wavelet

La trasformata wavelet di un segnale è la sua rappresentazione mediante l'uso di una forma d'onda oscillante di lunghezza finita o a decadimento rapido, scalata e traslata opportunamente. Tale forma d'onda, o "ondina" da cui il nome "wavelet", viene chiamata "wavelet madre"[36].

Il vantaggio della trasformata wavelet rispetto a quella di Fourier è la possibilità di mantenere informazioni temporali nella rappresentazione del segnale trasformato, oltre a fornire informazioni nel campo della frequenza del segnale oggetto della trasformazione.

Esistono due tipi di trasformata wavelet, la Continuous Wavelet Transfrom (CWT) e la Discrete Wavelet Transform (DWT), utilizzate a seconda della natura del segnale da rappresentare.

La DWT viene applicata a livelli, o stadi, ognuno dei quali consiste nel filtraggio del segnale discreto in ingresso con un filtro passa-basso ed uno passa-alto, entrambi ottenuti dalla mother wavelet scelta. L'uscita dei filtri viene "sottocampionata", eliminando un campione su due. In questo modo il numero di campioni in ingresso al livello wavelet corrisponde a quelli in uscita. La metà dei campioni ottenuti attraverso il filtro passa-basso forniscono un'approssimazione del segnale in ingresso, mentre la metà ottenuta con quello passa-alto rappresenta i dettagli in alta frequenza. In figura 63 è riportato uno schema di un singolo stadio. Le mother wavelet discrete più diffuse sono quelle di Haar e di Daubechies; esse sono ottenute campionando le loro versioni continue. In figura 64 è riportata la coppia di filtri della trasformata Wavelet Haar continua.

Figura 63 - Stadio wavelet, g[n] è un filtro passa-basso, h[n] è un filtro passa-alto

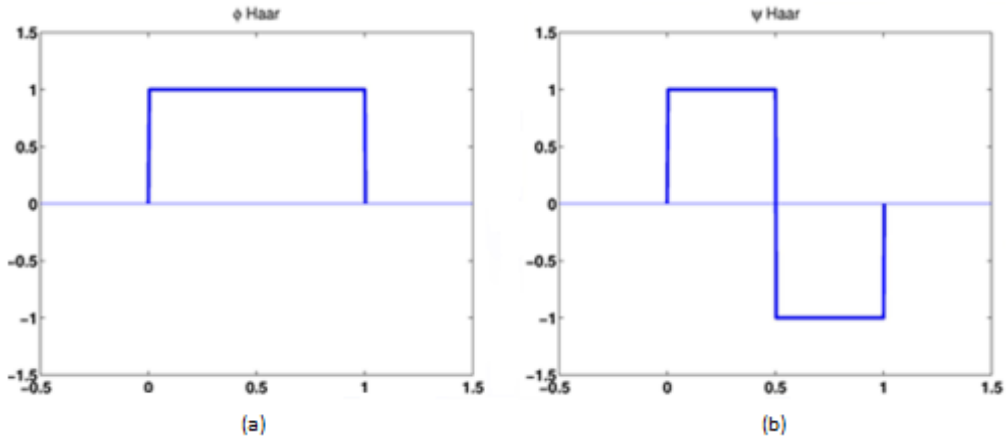


Figura 64 - Filtri passa-basso (a) e passa-alto (b) wavelet Haar continua

Generalmente tale trasformata viene applicata a più stadi consecutivamente, utilizzando i campioni di approssimazione come ingresso per un successivo stadio. Si

ottiene così la Wavelet Decomposition, che scompone il segnale discreto in ingresso in sotto-bande.

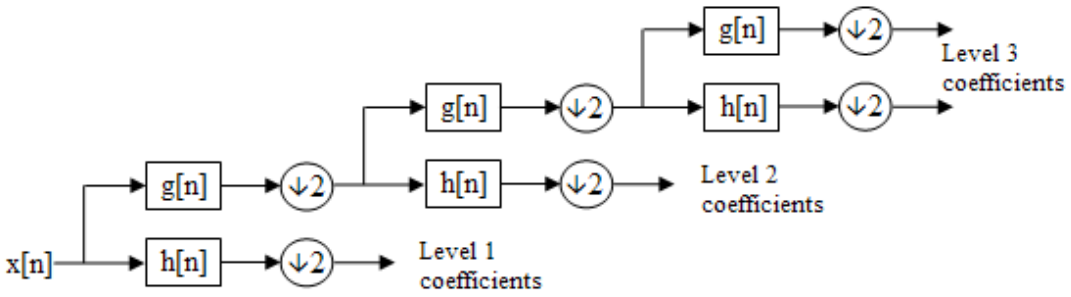


Figura 65 - Struttura Wavelet Decomposition a tre stadi

Facendo riferimento alla figura 65 soprastante, i coefficienti di approssimazione del livello tre rappresentano la componente più a bassa frequenza del segnale (vedi figura 66 sottostante), mentre i coefficienti degli altri livelli forniscono dettagli a frequenze più alte.

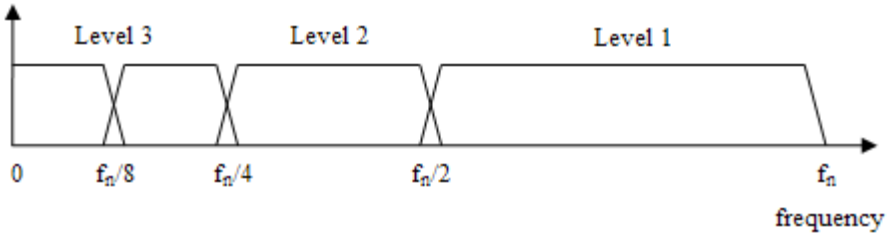
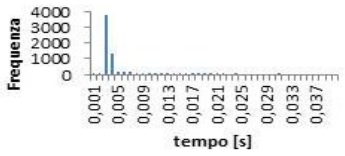
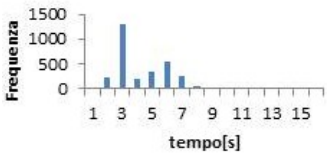
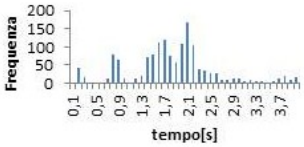
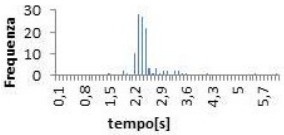
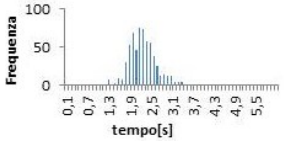
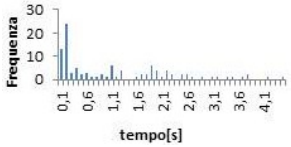
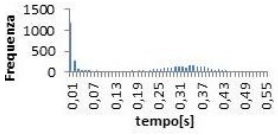
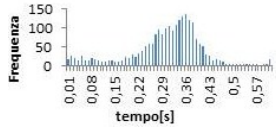
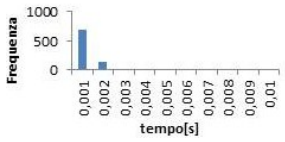
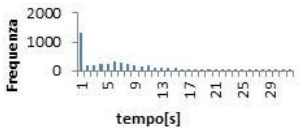
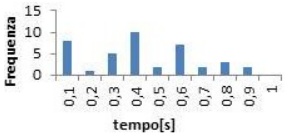
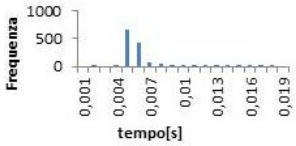
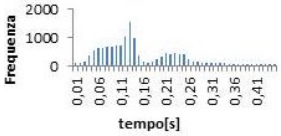
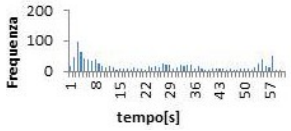
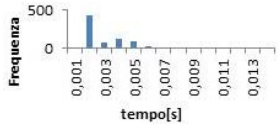
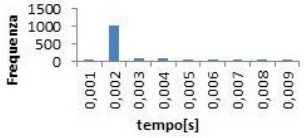
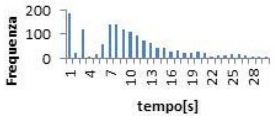
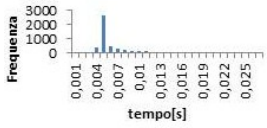
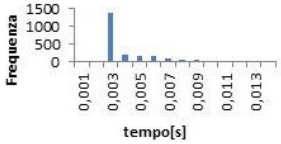
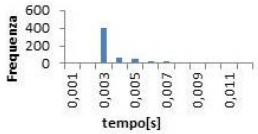


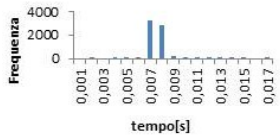
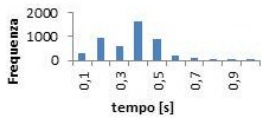
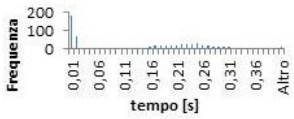
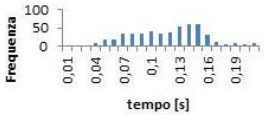
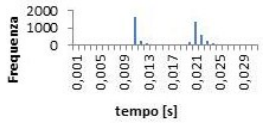
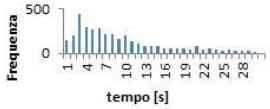
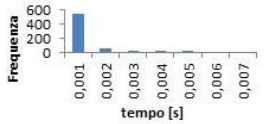
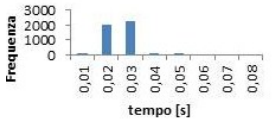
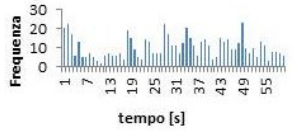
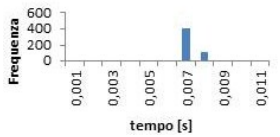
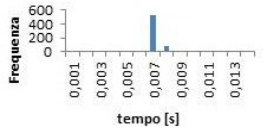
Figura 66 - Spettro del segnale "decomposto"

C. Parametri emulatore

La tabella seguente (tabella 28) riporta, per ogni coppia di messaggi SIP osservabile nei call flow considerati nella realizzazione dell'emulatore, la distribuzione del tempo di interarrivo tra i due messaggi, osservata in tracce di traffico su un SIP Trunk in esercizio. Inoltre riporta il modello empirico utilizzato per la riproduzione di tale tempo di interarrivo nell'emulatore.

COPPIA DI MESSAGGI	DISTRIBUZIONE OSSERVATA	MODELLO EMPIRICO
eINVITE-u100		dt = 3ms
e100-e180		50% dt = 3s ; 50% dt v.c. gaussiana con media = 6s e stdev = 1s
e100-e183		dt v.c. gaussiana con media = 2,1s e stdev = 300ms
e100-e200		dt = 2,3s
e100-e486		dt v.c. gaussiana con media = 2,2s e stdev = 300ms
e100-uCANCEL		dt = 180ms
e180-uPRACK		50% dt = 7ms ; 50% dt v.c. uniforme min = 280ms e max = 390ms
e183-uPRACK		dt v.c. gaussiana con media = 350ms e stdev = 40ms
e200-e180		dt = 1ms

e200-e200		dt v.c. uniforme min = 1ms e max = 13s
e200-e486		dt v.c. uniforme min = 100ms e max = 800ms
e200-e487		dt = 6ms
e200-uACK		dt v.c. gaussiana con media = 130ms e stdev = 50ms
e200-uCANCEL		dt v.c. uniforme min = 1s e max = 59s
e486-uACK		dt = 2ms
e487-uACK		dt = 2ms
eACK-uINVITE		dt v.c. esponenziale negativa con media = 9s
ePRACK-u200		dt = 5ms
eBYE-u200		dt = 3ms
eCANCEL-u200		dt = 3ms

uINVITE-e100		dt = 7ms
u100-u180		dt v.c. gaussiana con media = 400ms e stdev = 100ms
u100-u200		50% dt = 9ms ; 50% dt v.c. gaussiana con media = 240ms e stdev = 40ms
u100-u486		dt v.c. gaussiana con media = 130ms e stdev = 20ms
u180-ePRACK		50% dt = 11ms ; 50% dt = 21ms
u200-u200		dt v.c. esponenziale negativa con media = 3s
u200-u487		dt = 1ms
u200-eACK		dt = 25ms
u200-eCANCEL		dt v.c. uniforme min = 1s e max = 59s
u486-eACK		dt = 7ms
u487-eACK		dt = 7ms

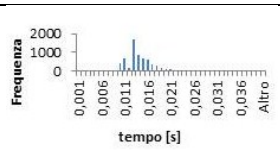
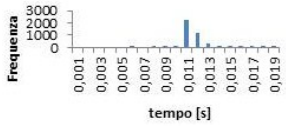
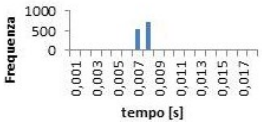
uPRACK-e200		dt v.c. gaussiana con media = 13ms e stdev = 2ms
uBYE-e200		dt = 11ms
uCANCEL-e200		dt = 7ms

Tabella 28 - Tempi di interarrivo osservati per ogni coppia di messaggi considerati e relativa scelta implementativa approssimata

INDICE DELLE FIGURE

Figura 1 - Architettura generale SBC di bordo rete 15
Figura 2 - Architettura di riferimento SIPconnect 18
Figura 3 - Scenario di default per SIPp sia in modalità client che server 20
Figura 4 - Esecuzione dello scenario standard SIPp uac.xml 21
Figura 5 - Esempi di classificatori multi-classe e one-class 25
Figura 6 - Esempio di dataset con zone di corretto funzionamento a diversa densità 27
Figura 7 - Distribuzioni classi "normale" e "anomala" con soglia del classificatore e indicazione di veri e falsi 34
Figura 8 - Curve di ROC 35
Figura 9 - Schema generale dello scenario di riferimento 37
Figura 10 - Piramide di segmentazione del mercato telefonico business 39
Figura 11 - Schema più dettagliato dello scenario di riferimento 39
Figura 12 - Posizionamento del sistema di monitoring all'interno dello scenario di riferimento 41
Figura 13 - Blocco di estrazione parametri dalla traccia di traffico SIP 44
Figura 14 - Blocco di "thresholding" 45
Figura 15 - Blocco di decisione 46
Figura 16 - Schema a blocchi completo 48
Figura 17 - Blocco di decisione composto dalla SVM 51
Figura 18 - Schema a blocchi completo 52
Figura 19 - Buffer 53
Figura 20 - Blocco che implementa la Wavelet Decomposition 53
Figura 21 - Blocco di estrazione medie-varianze o energie 54
Figura 22 - Blocco di decisione SVM 55
Figura 23 - Schema a blocchi completo 56
Figura 24 - Schema emulazione chiamate entranti e chiamate uscenti. 57
Figura 25 - Call flow considerati. 59
Figura 26 - Possibile evoluzione di una chiamata entrante nella rete enterprise. 63
Figura 27 - Possibile evoluzione di una chiamata uscente dalla rete enterprise. 68

Figura 28 - Schema dell'ambiente di emulazione 72
Figura 29 - Schema del sistema di riferimento. 72
Figura 30 - Profilo temporale anomalia "Link-Flap" 75
Figura 31 - Ambiente di emulazione con anomalia "link-flap" 76
Figura 32 - Schema scenario reale con anomalia "link-flap" 76
Figura 33 - Profilo temporale anomalia "CPU-Hog" 77
Figura 34 - Ambiente di emulazione con anomalia "CPU-hog" 78
Figura 35 - Schema scenario reale con anomalia "CPU-hog" 78
Figura 36 - Call flow di una chiamata in "loop" 81
Figura 37 - Ambiente di emulazione con anomalia "loop" 83
Figura 38 - Profilo temporale del traffico da Near-End a Far-End per test "flap 40 cps" 86
Figura 39 - Profilo temporale del traffico da Far-End a Near-End per test "flap 40 cps" 86
Figura 40 - Curva di ROC per test "flap 40 cps" 87
Figura 41 - Profilo temporale del traffico da Near-End a Far-End per test "flap 20" 89
Figura 42 - Profilo temporale del traffico da Far-End a Near-End per test "flap 20" 90
Figura 43 - Curva di ROC per test "flap 20" 90
Figura 44 - Profilo temporale del traffico da Near-End a Far-End per test "flap Tr" 92
Figura 45 - Profilo temporale del traffico da Near-End a Far-End per test "flap Tr" 92
Figura 46 - Curva di ROC per test "flap Tr" 93
Figura 47 - Profilo temporale del traffico da Near-End a Far-End per test "hog 40 cps" 95
Figura 48 - Profilo temporale del traffico da Far-End a Near-End per test "hog 40 cps" 96
Figura 49 - Curva di ROC per test "hog 40 cps" 96
Figura 50 - Profilo temporale del traffico da Near-End a Far-End per test "loop 40 cps" 98
Figura 51 - Profilo temporale del traffico da Far-End a Near-End per test "loop 40 cps" 99
Figura 52 - Curva di ROC per test "loop 40 cps" 99
Figura 53 - Profilo temporale del traffico da Near-End a Far-End per test "loop 20 cps" 101
Figura 54 - Profilo temporale del traffico da Far-End a Near-End per test "loop 20 cps" 101
Figura 55 - Curva di ROC per test "loop 20cps" 102

Figura 56 - Profilo temporale del traffico da Near-End a Far-End per test "loop Tr" ... 103
Figura 57 - Profilo temporale del traffico da Far-End a Near-End per test "loop Tr" ... 103
Figura 58 - Curva di ROC per test "loop Tr" ... 104
Figura 59 - Modelli di separazione ... 109
Figura 60 - Applicazione di una funzione kernel ai dati di input di una SVM ... 110
Figura 61 - Esempi di underfitting (a), corretto addestramento (b) e overfitting (c) ... 110
Figura 62 - Training set per l'addestramento di una v-SVM con v = 0.01 (a) e v = 0.5 (b) ... 111
Figura 63 - Stadio wavelet, g[n] è un filtro passa-basso, h[n] è un filtro passa-alto ... 112
Figura 64 - Filtri passa-basso (a) e passa-alto (b) wavelet Haar continua ... 112
Figura 65 - Struttura Wavelet Decomposition a tre stadi ... 113
Figura 66 - Spettro del segnale "decomposto" ... 113

INDICE DELLE TABELLE

Tabella 1 - Tabella di contingenza per decisori binari....34
Tabella 2 - Percentuali di ricorrenza dei call flow, divisi tra entranti e uscenti. 60
Tabella 3 - Legame tra call-flow entranti e pseudo-codice server....66
Tabella 4 - Legame tra call-flow entranti e pseudo-codice client....67
Tabella 5 - Relazione tra call-flow uscenti e pseudo-codice server....70
Tabella 6 - Relazione tra call-flow uscenti e pseudo-codice client....71
Tabella 7 - Intervalli temporali anomali nell'esperimento "Flap 40 cps" 85
Tabella 8 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Flap 40 cps" 88
Tabella 9 - Valori di FDR per esperimento "Flap 40 cps" 88
Tabella 10 - Intervalli temporali anomali nell'esperimento "Flap 20 cps" 89
Tabella 11 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Flap 20 cps" 91
Tabella 12 - Valori di FDR per esperimento "Flap 20 cps" 91
Tabella 13 - Intervalli temporali anomali nell'esperimento "Flap Tr" 92
Tabella 14 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Flap Tr" 94
Tabella 15 - Valori di FDR per esperimento "Flap Tr" 94
Tabella 16 - Intervalli temporali anomali nell'esperimento "Hog 40 cps" 95
Tabella 17 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Hog 40 cps" 97
Tabella 18 - Valori di FDR per esperimento "Hog 40 cps" 97
Tabella 19 - Intervalli temporali anomali nell'esperimento "Loop 40 cps" 98
Tabella 20 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Loop 40 cps" 100
Tabella 21 - Valori di FDR per esperimento "Loop 40 cps" 100
Tabella 22 - Intervalli temporali anomali nell'esperimento "Loop 20 cps" 100
Tabella 23 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Loop 20 cps" 102

Tabella 24 - Valori di FDR per esperimento "Loop 20 cps" 102
Tabella 25 - Intervalli temporali anomali nell'esperimento "Loop Tr" 103
Tabella 26 - Valori di TPR ed FPR al variare della percentuale di outliers nell'esperimento "Loop Tr" 104
Tabella 27 - Valori di FDR per esperimento "Loop Tr" 104
Tabella 28 - Tempi di interarrivo osservati per ogni coppia di messaggi considerati e relativa scelta implementativa approssimata 117

BIBLIOGRAFIA

[1] J. Rosenberg, *"What is a Session Initiation Protocol (SIP) Trunk Anyway?"*, IETF draft, 2008, http://tools.ietf.org/id/draft-rosenberg-sipping-siptrunk-00.txt .

[2] D. Sladden, z. Swapan, C. Hattingh, *SIP Trunking*, Cisco Press, 2010.

[3] J. Magnusson, *SIP Trunking Benefits and Best Practices*, Ingate Systems white paper, 2007, http://www.ingate.com/files/white_paper_What_is_SIP_Trunking_A.pdf .

[4] H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler, J. Rosenberg, *SIP: Session Initiation Protocol*, RFC 3261, 2002.

[5] SIP Forum, *SIPconnect 1.1 Technical Recommendation*, 2011, http://www.sipforum.org/sipconnect .

[6] M. Handley, V. Jacobson, C. Perkins, *SDP: Session Description Protocol*, RFC 4566, 2006.

[7] J. Rosenberg, H. Schulzrinne, *An Offer/Answer Model with the Session Description Protocol (SDP)*, RFC 3264, 2002.

[8] http://sipp.sourceforge.net

[9] A. K. Jones, R. S. Sielken, *Computer System Intrusion Detection: A Survey*, 1999, http://www.cs.virginia.edu/~jones/IDS-research/Documents/jones-sielken-survey-v11.pdf .

[10] V. Chandola, A. Banerjee, V. Kumar, *Anomaly Detection: A Survey*, ACM Computing Surveys, vol.41(3), 2009.

[11] S. Hawkins, H. He, G. Williams, R. Baxter, Outlier Detection Using Replicator Neural Networks, DaWaK 2000, 2002.

[12] L. Gao, Z. Ren, W. Tang, H. Wang, P. Chen, *Intelligent Gearbox Diagnosis Methods Based on SVM, Wavelet Lifting and RBR*, Sensors, vol. 10(5), pag. 4602-4621, 2010.

[13] Y. Bu, T. Leung, A. Fu, E. Keogh, J. Pei, S. Meshkin, *WAT: Finding Top-K Discords in Time Series Database*, in "Proceedings of 7th SIAM International Conference on Data Mining", 2007, http://www.ics.uci.edu/~yingyib/papers/sdm2007_discords_camera-ready.pdf .

[14] K. Chan, A. Fu, *Efficient Time Series Matching by Wavelets*, ICDE '99 "Proceedings of the 15th International Conference on Data Engineering", pag. 126-133, 1999.

[15] A. Kandaswamy, C. Kumar, R. P. Ramanathan, S. Jayaraman, N. Malmurugan, *Neural classification of lung sounds using wavelet coefficients*, Computers in Biology and Medicine, vol. 34(6), pag. 523-537, 2004.

[16] P. Jahankhani, V. Kodogiannis, K. Revett, *EEG Signal Classification Using Wavelet Feature Extraction and Neural Networks*, JVA '06, "IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing", pag. 120-124, 2006.

[17] L. Li, G. Lee, *DoS Attack Detection and Wavelets*, ICCCN 2003, "Proceedings the 12th International Conference on Computer Communications and Networks", pag. 421-427, 2003.

[18] A. Matsuyama, M. Jonkman, *The Application of Wavelet and Feature Vectors to ECG Signals*, "Australasian Physical & Engineering Sciences in Medicine, vol. 29(1), pag. 13-17, 2006.

[19] J. M. E. Tapiador, P. G. Teodoro, J. E. D. Verdejo, *Anomaly detection methods in wired networks: a survey and taxonomy*, Computer Communications, vol. 27(16), pag. 1569-1584, 2004.

[20] M. Thottan, C. Ji, *Anomaly Detection in IP Networks*, IEEE Transactions on Signal Processing, vol. 51(8), pag. 2191-2204, 2003.

[21] A. Magnaghi, T. Hamada, T. Katsuyama, *A Wavelet-Based Framework for Proactive Detection of Network Misconfigurations*, Net '04, "Proceedings of the ACM SIGCOMM workshop on Network troubleshooting: research, theory and operations practice meet malfunctioning reality", 2004.

[22] G. D. Breda, L. Mendes, *QoS monitoring and Failure Detection*, International Communication Symposium, pag. 243-248, 2006.

[23] B. Tellenbach, M. Burkhart, D. Schatzmann, D. Gugelmann, D. Sornette, *Accurate network anomaly classification with generalized entropy metrics*, Computer Networks 55, pag. 3485-3502, 2011.

[24] J. G. Dai, Z. Xu, *Network anomaly detection using dissimilarity-based one-class SVM classifier*, ICPPW '09, "International Conference on Parallel Processing Workshops 2009", pag. 409-414, 2009.

[25] A. Kumar, S. Tilagam, *A Novel Approach for Evaluating and Detecting Low Rate SIP Flooding Attack*, International Journal Computer Applications, vol. 26(1), 2011.

[26] M. Nassar, R. State, O. Festor, *Monitoring SIP Traffic Using Support Vector Machines*, RAID '08, pag. 311-330, 2008.

[27] T. Fawcett, *ROC Graphs: Notes and Practical Considerations for Researchers*, "HP Labs Tech Report", No. HPL-2003-4, 2004.

[28] G. Giacinto, R. Perdisci, M. Del Rio, F. Roli, *Intrusion Detection in Computer Networks by a Modular Ensemble of One-Class Classifier*, Information Fusion, vol. 9(1), pag. 69-82, 2008.

[29] LibSVM, http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[30] J. Wan, S. Zhou, *Features Extraction Based on Wavelet Packet Transform for B-mode Ultrasound Liver Images*, CISP, "3rd International Congress on Images and Signal Processing", pag. 949-955, 2010.

[31] *Errdisable Port State Recovery on the Cisco IOS Platforms*, Cisco, Document ID: 69980, 2009.

[32] http://cpulimit.sourceforge.net

[33] *Cisco BTS 10200 Softswitch SIP Feature and Provisioning Guide, Release 5.0*, Cisco, 2011.

[34] C. Cortes, V. Vapnik, *Support Vector Networks*, Machine Learning, vol. 20(3), pag. 273-297, 1995.

[35] P. H. Chen, C. J. Lin, B. Schölkopf, *A tutorial on ν-support vector machines*, Applied Stochastic Models in Business and Industry, vol. 21(2), pag. 111-136, 2005.

[36] S. G. Mallat, *A Theory for Multiresolution Signal Decomposition: The Wavelet Representation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11(7), pag.674-693, 1989.