ITESYS SRL

Versione 0.1 4 Luglio 2008
Revisione 0.2 8 Luglio 2008
Versione 0.3 25 Settembre 2008
Versione 0.4 20 Gennaio 2009 (aggiunto speex + correzioni su MTU e trunk IAX + revisione)

By Gianrico Fichera
Con la collaborazione di: Ing. Maurizio Intravaia, Dott. Aldo Schinina

Codec - Dimensionamento dei link dati per canali VoIP in reti WAN basate su Asterisk–

1.0 Introduzione

Questo articolo si pone il problema di come dimensionare correttamente un link dati per poter trasmettere un certo numero di conversazioni telefoniche in Voice Over IP. Un canale voce su IP occupa una banda dipendente dall'algoritmo di codifica della voce utilizzato e dai protocolli di linea. In questo articolo si citeranno principalmente i codec G711, ovvero voce non compressa, e i codec G729, Speex e GSM, ovvero voce compressa. Vi sono numerosi altri codec naturalmente, ma dagli elementi presentati si potra' facilmente dedurre l'impatto che un canale VoIP ha in una rete WAN indipendentemente dal tipo di codec utilizzato.

La trattazione di questo documento e' limitata alla VoIP. Non e' possibile utilizzare, se non parzialmente, questi stessi criteri di calcolo per valutare l'efficienza di trasmissione di dati di tipo differente.

2.0 Struttura di un pacchetto VoIP

La voce viaggia in pacchetti IP utilizzando UDP e normalmente RTP. Ecco un esempio di pacchetto voce:

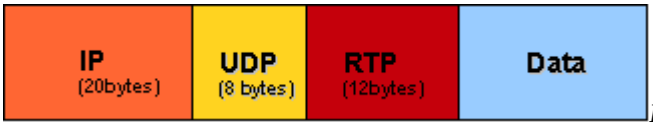


Figura 1

Come si vede gli header di ogni pacchetto occupano 40bytes = 20bytes(IP) + 8bytes(UDP) + 12bytes(RTP) (si puo' ridurre questo valore con l'utilizzo di qualche tipologia di compressione dell'header come '*ip rtp header compression*' RFC 2508 o '*compressed rtp cRTP*').

Un convertitore analogico/digitale trasforma la voce in pacchetti dati che vengono inseriti nel campo "Data" del pacchetto IP, chiamato **payload**. I dati campionati vengono raccolti e quindi spediti. Il numero di campioni voce inseriti nel payload e' chiamato **sample period.** L'algoritmo di conversione analogico/digitale e' chiamato **codec**.

Quindi per trasmettere la VoIP su un canale dati non basta considerare la banda utilizzata dalla voce stessa, ma bisogna considerare anche delle risorse aggiuntive proprie dei protocolli utilizzati per il trasporto. A queste risorse aggiuntive si da il nome di **overhead**. Ad esempio, in Figura 1 si ha un overhead di 40byte. Ma la figura 1 si limita al layer 3 della pila protocollare. Poiche' i pacchetti IP viaggiano su protocolli di Layer2 (Ethernet, ATM, Frame-relay) vi e' dell'overhead aggiuntivo rispetto quello di Figura 1, che e' mostrato in dettaglio nei prossimi paragrafi.

3.0 VoIP su layer 2 ATM: connessioni ADSL

Si abbia una connettivita' WAN di tipo ADSL. ADSL utilizza il protocollo di layer 2 ATM. Dobbiamo dunque trasmettere il pacchetto di figura 2 attraverso una connettivita' di tipo ADSL. Il problema pertanto e' quello di trasmettere il protocollo IP su ATM e vi sono varie modalita' differenti per farlo a seconda della configurazione che ci impone il fornitore di connettivita' **ISP**, che possiamo riassumere come IP over ATM, modalita' PPP over ATM (PPPoA) oppure Ethernet over ATM (PPPoE).

ATM e' un protocollo dove ogni pacchetto (chiamato **cella**) ha lunghezza fissa di 53byte. Di questi 5byte sono di header. ATM offre un livello di adattamento nella sua pila protocollare che consente di inserire pacchetti di layer 3 distribuendoli in celle differenti. Senza entrare nel dettaglio tale livello si chiama **AAL5**.

Sia assegnato un pacchetto IP allora e' detta **PDU** un gruppo di celle che hanno la caratteristica di trasportare una parte di questo pacchetto. Per ogni PDU AAL5 ha un overhead di 8byte che si inserisce alla fine della PDU. Attenzione: 8byte per ogni PDU e NON per ogni cella ATM.

Figura 2

RFC1483 e' la specifica per il protocol over ATM che definisce le modalita' per inviare un generico protocollo su ATM, tra cui IP. Consente di utilizzare due modalita': routing o bridging. Il routing e' la soluzione di uso piu' comune per collegamenti verso Internet. Il bridging aggiunge un layer MAC ed e' configurato, nel caso di un router Cisco, con i comandi "bridge" nelle interfacce coinvolte (normalmente ATM e Ethernet). Si utilizza anche la sigla RFC1483R per indicare la modalita' routing dell'RFC1483.

RFC1483 specifica due metodi per il trasporto su ATM: senza multiprotocollo, con multiprotocollo. Nel caso in cui si debba trasportare un solo protocollo su un collegamento ATM, ad esempio il protocollo IP, oppure il PPP, oppure Ethernet non ci sono header aggiuntivi al pacchetto ATM oltre il trailer AAL5. Nel caso di multiprotocollo ("encapsulation aal5snap" nei router cisco) si ha un overhead aggiuntivo pari a un header aggiuntivo di 8byte per PDU per il livello LLC/SNAP, che consente di utilizzare protocolli differenti sullo stesso collegamento (ad es. IP e IPX simultaneamente).

Poiche' gli ISP hanno necessita' di utilizzare un meccanismo di autenticazione utilizzano spesso il PPP aumentanto l'overhead. Se utilizziamo il protocollo PPP su ATM questo si indica con l'abbreviazione PPPoA. In tal caso le specifiche sono nell'**RFC2364** "PPP over AAL5" che completa RFC1483 per il PPP. Anche qui possiamo utilizzare MUX o LLC (per uno o piu' protocolli). In entrambi i casi per ogni PDU il PPP ha un overhead massimo di 2byte.

Se utilizziamo infine il protocollo PPPoE ci sono altri 6byte per PDU di PPPoE header. Poiche' per fare PPPoE si utilizza il bridging RFC1483 ci sono anche altri 18 byte di MAC layer e altri 8byte di LLC. Quindi nel caso peggiore, ovvero di PPPoE, per ognu PDU abbiamo quanto si evince dalla figura 2.1.

G729	20 bytes	
IP/UDP/RTP overhead	40 bytes	
PPP	2 bytes	Solo nel caso di PPPoA o PPPoE
PPPoE	6 bytes	Solo nel caso di PPPoE
MAC	18 bytes	Solo nel caso di PPPoE oppure RFC1483 bridged
LLC/SNAP	8 bytes	Solo nel caso di gestione multiprotocollo
AAL5	8 bytes	Per ogni PDU
ATM	5 bytes	Per ogni cella

Figura 2.1

Nota:

Per trattare tutti i casi particolari sarebbe necessario l'intero capitolo di un libro. Si tengano comunque presenti queste due ulteriori considerazioni:

1. Quando il protocollo ethernet e' coinvolto, nel bridging e nel PPPoE, si ricordi che un frame ethernet dev'essere di lunghezza minima di 64byte. Il payload G729+IP e' di 60bytes. Tuttavia abbiamo altro overhead nel caso di PPPoE, come si evince in figura 2.1, e questo risolve il problema, evitando altri byte sprecati per raggiungere i 64byte.
2. Il frame Ethernet e' 6byte + 6byte + 2byte + 4byte = MAC + MAC + TYPE + CRC

4.0 Il codec G711

Un codec consente una conversione analogico/digitale con un algoritmo. L'algoritmo piu' semplice e storicamente piu' utilizzato e' il **G711**. Questo non effettua compressione della voce ma ne effettua semplicemente un campionamento prelevando 8000 campioni in un secondo. Ogni campione e' in un byte di dati, pertanto ogni campione discretizza il livello vocale in uno tra 256 intervalli (**intervallo di quantizzazione**). L'**intervallo di campionamento** e' di 1s/8000=0,125ms.

Campionare a 8khz con 256 livelli vuol dire generare un flusso di 8000*8bit=64000bps. Pertanto un canale VoIP, senza overhead, con il codec G711 necessita di 64kbps. I canali voce delle linee telefoniche digitali POTS (ISDN BRI, PRI ad es.) utilizzano questo tipo di codec.

4.1 Banda per G711

Il codec G711 genera 8 campioni per ms (8000campioni/1000ms) pertanto con sample period a 20ms abbiamo 160byte di dati da inserire nel payload per completare un pacchetto IP. Poiche' gli headers del pacchetto sono 40bytes abbiamo 160byte+40bytes=200bytes per 20ms di voce.

Per 1 secondo di voce ci vogliono allora 1000ms / 20ms=50pacchetti, ovvero 200bytes / sec * 50 pacchetti = 80000bit/sec.

Allora un canale VoIP G711 occupa 80kbps al secondo per ogni direzione su IP (tranne in wireless e nelle reti full-duplex). Tuttavia un pacchetto IP non puo' essere trasmesso senza un protocollo di layer2. E allora bisogna aggiungere altro overhead dipendente dal media utilizzato.

5.0 Il codec G729

G729 e' specificato in ITU-T Recommendation G.729, "Coding of speech at 8 kbit/s using conjugate structure-algebraic code excited linear prediction (CS-ACELP)". *Il codec G729 genera ha un bit rate di 8kbps al secondo. I campioni sono di 80bit e codificano 10ms di voce cadauno. Per default la trasmissione avviene con due campioni per pacchetto, ovvero 20ms di voce per ogni pacchetto IP.* G729 e' protetto da brevetto e quindi non e' disponibile gratuitamente. Per asterisk e' possibile acquistarlo dall'azienda DIGIUM. Senza licenza G729 e' possibile utilizzare Asterisk solo come pass-through ovvero, se i telefoni SIP supportano il G729, per far transitare il canale voce senza effettuarne trascodifica. Pertanto non sara' possibile utilizzare i servizi della centrale, come IVR, Call-Conference, conversazione a tre etc. Vi sono diverse varianti di codec G729. Asterisk supporta G729 Annex A. Esiste anche una codifica G729 Annex B che comprende funzionalita' aggiuntive come il VAD/CNG/DTX che pertanto su Asterisk non sono supportate. Interessante e' allora il confronto con il codex speex che, pur essendo open source, e' dotato di VAD/CNG/DTX.

5.0.1 Trascodifica

Lavorando con i codec con compressione si pone il problema delle risorse di calcolo necessarie in caso di trascodifica. Infatti, a meno di non acquistare una scheda per la codifica in hardware come la Digium TC400B, l'eventuale trascodifica viene fatta in software ed utilizza risorse CPU. Per dare un'idea diciamo che la capacita' di processo di una CPU e' data dal prodotto del numero di core per la velocita' di clock. Al 95% di utilizzo CPU possiamo immaginare di gestire una trascodifica di 30 linee per GHz (per core) nel caso di un processore Intel Xeon e del codec G729. Per non forzare un dimensionamento corretto sara' in questo caso di 15 linee per GHz. E' interessante vedere le prestazioni per sistemi diametralmente opposti con processori Intel entry level. Nel caso di un Atom 230, a 1.6Ghz, con 30 transcodifiche si arriva al 95% di CPU, quindi realisticamente possiamo pensare di utilizzarlo per 15 trascodifiche G729. Senza trascodifica tutti i sistemi gestirebbero i codec per un numero di linee superiore di un fattore x3 o x4. Ovviamente questi sono valori indicativi, i processori hanno un carico di base che puo' dipendere fortemente dal sistema operativo utilizzato e dai servizi attivi sul sistema, oltre che dall'hardware su cui girano.

5.1 Banda per G729 Layer 3

In base a quanto visto in *Figura 1* si ha che 20ms di voce occupano 40bytes+20bytes quindi un pacchetto G729 si puo' rappresentare come in figura 3.

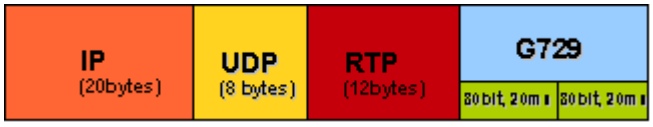


Figura 3

Con una semplice proporzione possiamo calcolare la larghezza di banda totale utilizzata:

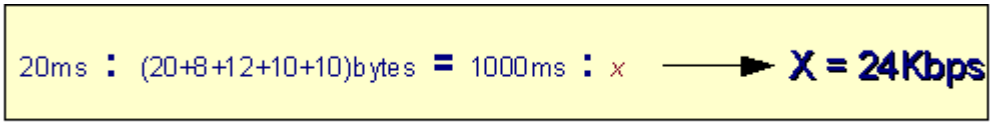


Figura 4

Pertanto un flusso G729 su IP occupa 24kbps di banda. Questo dato e' molto utile da un punto di vista teorico ma praticamente non serve a granche'.

Un pacchetto IP non puo' viaggiare in una rete senza un protocollo di livello2. Pertanto il nostro pacchetto di figura3 crescera' ulteriormente in dimensione prima di essere trasmesso.

Bisogna considerare l'header di un frame Ethernet, se siamo in una LAN Ethernet, oppure l'header ATM, se siamo, ad esempio, in un collegamento WAN di tipo XDSL. Stesso ragionamento dicasi per altri tipi di media di trasmissione.

5.2 Banda per G729 con Layer 2 IP over ATM

Abbiamo visto in figura 2 una cella ATM. In tutte le reti con connettivita' ATM i pacchetti IP vanno inseriti in una o piu' celle ATM. Un tipico utilizzo di una connettivita' WAN di tipo ATM abbiamo visto che e' nelle connessioni XDSL. Per ogni trasmissione dati/voce/video su una connessione tipo adsl o shdsl bisogna considerare l'overhead dato dalle celle ATM e dal metodo di incapsulamento di IP su ATM. Ricordiamo che una delle particolarita' di ATM e' che ogni pacchetto (chiamato cella) ha le dimensioni prefissate di 53byte. Non ci sono pacchetti di dimensione variabile come in IP o Ethernet ad esempio.

Ogni 53byte di cella ATM 5byte sono di header. Ben difficilmente un pacchetto IP entra in una cella ATM. Abbiamo visto infatti che il pacchetto IP G729 e' di 60byte (vedi figura 3). Ecco allora che ci viene in aiuto AAL5 ed ecco che piu' celle atm vengono utilizzate per trasportare i nostri 20ms di G729:

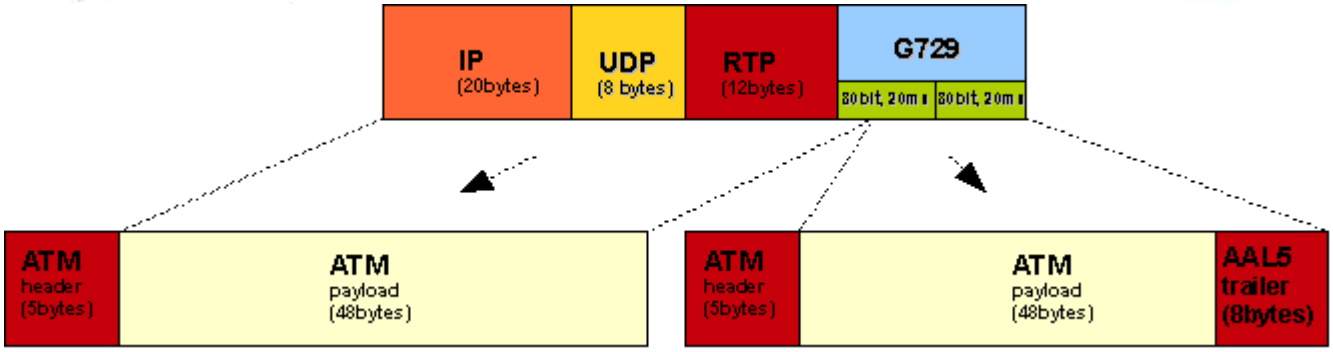


Figura 5

Pertanto considerando che ogni cella ATM brucia 5bytes, che ci servono due celle ATM e che abbiamo un trailer di 8bytes (AAL5):

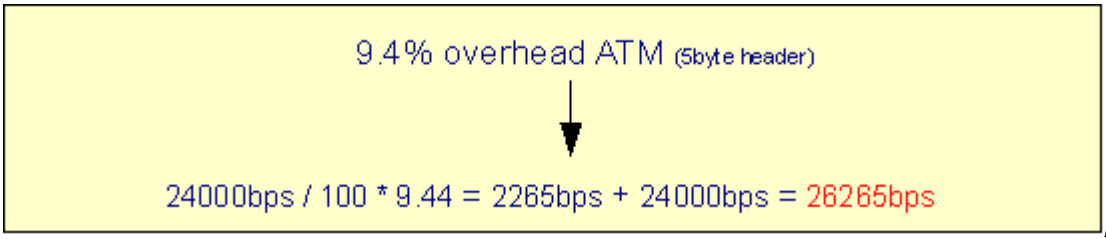


Figura 6

Concludiamo quindi che un canale G729 con 20ms di sample period, in una rete ATM RFC1483R VC-MUX IP utilizza 26kbps. Gli 8 bytes di trailer AAL5 non sono un problema e non sono neanche stati considerati nei calcoli. Questo perche' il pacchetto IP/G729 di 60byte lascia abbastanza spazio libero nelle due celle ATM per assorbire l'impatto dell'AAL5

trailer e anche di altri header che possono essere presenti in altri tipi di trasmissione. Il ragionamento qui fatto vale nel caso in cui la trasmissione venga effettuata con RFC1483R VC-MUX (questa e' una specifica per l'IP over ATM utilizzata di frequente ed e' la piu' semplice). Nel caso di un router Cisco stiamo parlando del comando "encapsulation aal5mux ip".

Nel caso di un collegamento ADSL di tipo PPPoE, in base a quanto indicato in figura 2.1 si ha:

PDU: *104 byte* -> 3 celle ATM in quanto 48byte x 2= 96.
Nell'ultima cella: *8byte AAL5 + 8 byte di dati + 32 byte* di **PADDING** (spazio vuoto)

TOTALE: 144byte per 20ms = **57600 bps**

Una bella differenza rispetto quanto indicato in Figura 6.

6.0 Il codec speex

Il codec speex e' OpenSource. Ovvero e' gratuito. Ecco alcuni vantaggi rispetto il G729a:

1. E' royalty-free;
2. Consente una compressione tra 2.15 e 44kbps;
3. Supporta il VAD e il CNG;
4. Supporta il VBR ovvero la compressione puo' essere dinamicamente modificata in base al tipo di frequenza da campionare.

7.0 Il protocollo layer 3 IAX

Abbiamo visto che il protocollo RTP incide per 12bytes per ogni 20ms di g729 da cui si evince che G729+RTP generano 12800bps. IAX e' un protocollo alternativo ad RTP utilizzato da Asterisk. Normalmente i trunk tra le centrali Asterisk si realizzano con IAX. Questo protocollo ha due modalita': trunk e no trunk. La modalita' trunk e' quella ottimale in quanto utilizza un unico header per trasmettere piu' canali voce. Pertanto si riduce significativamente l'overhead dovuto al pacchetto IP in quanto un pacchetto IP viene riempito nel payload con piu' frame voce relativi a differenti conversazioni. Al crescere del numero di conversazioni cresce la dimensione del pacchetto UDP. Nel momento in cui questo supera il valore di MTU in uso sulla rete la qualita' della voce puo' venire compromessa a causa della frammentazione del pacchetto IP.

Allora non bisogna superare l'MTU del percorso di rete. Nel file iax.conf c'e' un parametro che prende il nome di **trunkmtu** (introdotto definitivamente con asterisk 1.6) che di default e' a 1240bytes. Questo valore e' derivato dal valore 1516byte della ethernet, meno un frame g711 (ovvero *200byte di dati UDP/IP/G711 + 64byte di frame Eth + 12 di trunk IAX*). Questo vuol dire che asterisk non generera' pacchetti di dimensione superiore a 1240byte. Al raggiungere della soglia verranno inviati piu' pacchetti UDP. Con **trunkmtu=0** si disabilita questo comportamento. Nel caso di asterisk 1.2/1.4 non e' possibile fissare questo parametro e la dimensione del pacchetto crescera' lasciando gestire la frammentazione alla rete. In ogni caso esiste una patch sin da asterisk 1.2 per gestire l'mtu. Il trunk in ogni caso crescera' fino ad un limite massimo di 128000byte (per asterisk 1.4, invece per asterisk 1.6 fino ad un valore fissabile con **trunkmaxsize** in iax.conf).

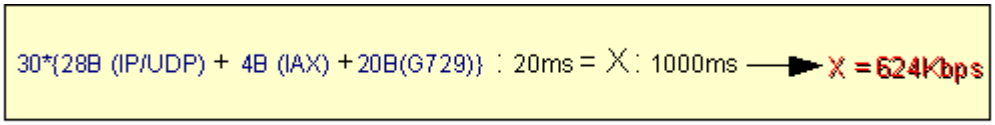
IAX non trunked ha 4bytes di overhead invece dei 12 di RTP. Pertanto G729+IAX=9600bps. Nel caso del trunk ci sono 12bytes di overhead che pero' rimane quasi costante all'aumentare del numero di canali (ci sono 4byte per canale aggiuntivi, vedi oltre). Solo nel caso di una chiamata singola IAX trunk pesa di piu' di IAX non trunk. Maggiore e il numero di canali e maggiori sono i benefici.

Per calcolare il traffico generato con piu' conversazioni attive facciamo un calcolo, che, come sempre, non considera casi particolari o pacchetti di servizio (Ad esempio se avete **trunktimestamps**=yes c'e' 1kbps per chiamata extra)

Con 30 canali G.729 con IAX in modalita' trunk la banda in uso e':

$$28B\ (IP/UDP) + 8B\ (IAX) + [20B(G729) + 4(IAX)] * 30 : 20ms = X : 1000ms \quad X = 302Kbps$$

Con 30 canali G.729 con IAX in modalita' no-trunk la banda in uso e':



Con G729 e RTP si ha 24kbps*30 = 720kbps. Si noti che i calcoli sono fatti senza considerare l'overhead eventuale dovuto alle problematiche sull'MTU di cui sopra.

Nota:

Non utilizzare il jitterbuffer con IAX in trunk mode a meno che non sia **trunktimestamps=yes.** Entrambi i lati devono essere configurati allo stesso modo.

1. Quando il protocollo ethernet e' coinvolto, nel bridging e nel PPPoE, si ricordi che un frame ethernet dev'essere di lunghezza minima di 64byte. Il payload G729+IP e' di 60bytes. Tuttavia abbiamo altro overhead nel caso di PPPoE, come si evince in figura 2.1 che risolve il problema, evitando altri byte sprecati per raggiungere i 64byte.

2. Nel caso di FCS attivo su Ethernet possono essere necessari altri 4byte.

3. Il parametro **trunkfreq**, impostato a 20ms, indica ogni quanto tempo inviare i pacchetti voce da spedire. Puo' essere aumentato, aumentando il delay, per migliorare ulteriormente l'uso di banda. Ovviamente dev'essere maggiore del delay del codec. Se ad esempio usiamo iLBC normalmente si usa 30ms come valore minimo.

7.0.1 Packet trace IAX no-trunk

Per finire ecco l'analisi di un pacchetto IAX no-trunk con codec GSM (20ms, 33byte, 13kbps):

tcpdump -s 2048 udp and port 4569 -A -xx <-- pacchetti di max 2048byte UDP sulla porta 4569

```
14:27:13.622550 IP 192.168.81.254.iax > 192.168.84.254.iax: UDP, length 37 <-- 37 e' il payload UDP

0x0000: (0015  171b  da28  001b  5483  2948  0800) (4500
0x0010:  0041  0bdd  0000  3e11  4882  c0a8  51fe   c0a8
0x0020:  54fe) 11d9  11d9  002d  b924 (3c55  a624   d5e7
0x0030:  54eb  637d  2d49  0ae9  baec  7d6b  8ecc   9a54
0x0040:  deba  eb16  eec9  8d16  e00c  2b73  42c7   5c)
```

In **verde** il frame ethernet con 6 byte di MAC sender, 6 byte di MAC destination, e "0800" che

indica che il pacchetto trasportato e' di tipo IP.

In **rosa** il pacchetto IP, dove gli ultimi 16 bit sono gli IP mittente e destinatario, il primo byte, "4" indica IPV4, il secondo byte, "5", indica 5 word di lunghezza dell'header.

In **grigio** il pacchetto UDP, dove "11d9" indica "4569" ovvero la porta dell'IAX, sia come mittente che come destinatario, e questo rende semplice la configurazione dei firewall nel caso di IAX. Segue il campo lenght e checksum (quest'ultimo puo' essere tolto in alcuni sistemi asterisk utilizzando l'opzione "nochecksums" in iax.conf.

In **azzurro** i 4byte di pacchetto IAX

In **nero** i 33byte di pacchetto voce GSM. Sono due campioni da 20ms.

7.0.2 Packet trace IAX trunk

In riferimento a quanto documentato nell'Internet-Draft "IAX: Inter-Asterisk eXchange Version 2, draft-guy-iax-05" e ad una versione di asterisk 1.4.21, ecco come vanno interpretati i pacchetti IAX, visualizzabili utilizzando il tool *tcpdump* con le le stesse indicazioni del paragrafo precedente.

Ci sono due diversi formati per i pacchetti del trunk IAX. La differenza sta nella presenza o meno del campo "time-stamp" relativamente ad ogni singola chiamata piuttosto che in un campo unico per l'intero pacchetto (la parte in rosso nella figura e' un'opzione). Tale campo e' presente per ogni singola chiamata nel caso in cui in iax.conf c'e' il valore "trunktimestamp=yes". E' consigliabile in quanto occupa poca banda ed aiuta per la sincronizzazione.

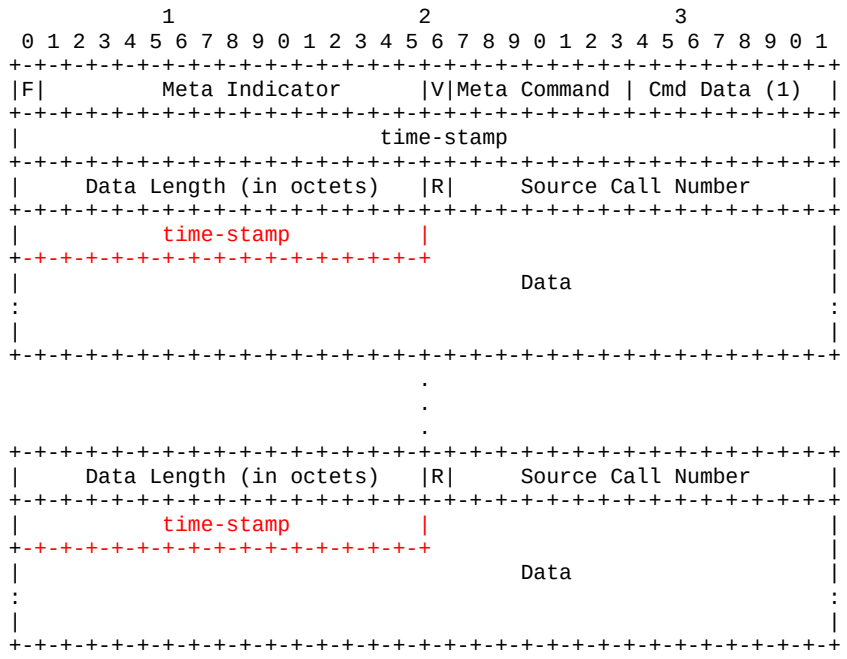


Figure 7: Meta Trunk Frame Binary Format (trunk time-stamps 1)

Fonte (IAX: Inter-Asterisk eXchange Version 2 draft-guy-iax-04)

Nella figura 7 osserviamo il formato di un pacchetto trunk.

Ecco un esempio nel caso di codec GSM e di una sola chiamata trasportata:

```
14:38:50.720082 IP 192.168.30.123.iax > 192.168.30.226.iax: UDP, length 45

0x0000:  000c 29ec  b254 0040 63e2 aae5 0800 4500
0x0010:  0049 b1ef  0000 4011 0a07 c0a8 1e7b c0a8
0x0020:  1ee2 11d9  11d9 0035 b806 0000 0100 0000
0x0030:  0988 2c06  0021 d931 74e2 5da8 c610 e49e          21 - campo data lenght IAX
0x0040:  9252 aff4  4924 6fda 5355 18ae 9256 6aa4
0x0050:  84b4 853b  4ecd 1b
```

Ecco un pacchetto IAX di trunk che trasporta due chiamate GSM

```
16:12:09.808905 IP 192.168.30.123.iax > 192.168.30.226.iax: UDP, length 82

0x0000:  000c 29ec b254 0040 63e2 aae5 0800 4500
0x0010:  006e e83c 0000 4011 d394 c0a8 1e7b c0a8
0x0020:  1ee2 11d9 11d9 005a 6a87 0000 0100 0000
0x0030:  6e27 2216 0021 d79f 9a29 6c8e a058 d476
0x0040:  2925 5a40 36db 89b7 24a0 c046 db91 c8ec
0x0050:  a880 391b 51b7 1440 0000 21d4 e284 6e21
0x0060:  98e5 7b6b 7606 21ba 8387 2a8d ba14 ecc3
0x0070:  fa22 9d89 5173 04a7 6ca1 eaf9
```

Possiamo notare la presenza di due gruppi di 33byte, che rappresentano due canali GSM, separati da 4byte con il "Source call number" e il "datalength". Come riferimento in rosso e segnata la fine del pacchetto IAX e l'inizio del GSM.

Ecco un altro esempio con "trunktimestamps=yes" e "trunkfreq=40" (per avere 4 campioni per singolo pacchetto IP piuttosto che due). Ecco il contenuto di un singolo pacchetto IP nel caso di due conversazioni attive:

```
08:06:38.772125 IP 192.168.2.3.iax > 192-85-14-169.b2b: UDP, length 164
            0x0000:  001b d449 2f38 0040 f47f d892 0800 4500
            0x0010:  00c0 1afa 0000 4011 3b8a c0a8 0203 5255
            0x0020:  0ea9 11d9 11d9 00ac cc30
            - qui finisce l'header UDP e inizia IAX-
            0000 0101
            0005 c300 32bit time-stamp
            --- prima conversazione primi 20ms ---
            0021 data lenght (33byte ovvero 10ms di gsm)
            4002 source call-number (id della chiamata, visibile con "iax2 show channels")
            da64 trunk-time-stamp
            daa4 73de 2270 a054 f348 bae3 86c0 172b 9245 11d8 a058 ea72
            46ec 86c0 366c b92d 6b     - pacchetto gsm 10ms di 33byte -
            --- seconda conversazione primi 20ms ---
            00 21   data lenght (33byte ovvero 10ms di gsm)
            40 01   source call-number (id della chiamata, visibile con "iax2 show channels")
            d7 0c   trunk-time-stamp
            d8 269c 1d5c b321 cc23 4958 edc8 81d9 130e 48f2 b8e1 c494
            ed27 27a6 e248 2445 c8b3
            --- seguono i secondi 20ms della prima e della seconda conversazione ---
            0x0080:  0021 4002 da78 da63 73a2 22c8 e027 334e
            0x0090:  5b4a bea0 b924 94d6 5cdf 205a 5acd 1899
```

```
0x00a0:  8680 24ef b9a8 9800 2140 01d7 20d7 e89b
0x00b0:  a5a5 5680 9d18 7b53 d6b6 81e3 5caa a539
0x00c0:  62c1 0b75 331a 8bb3 41ad 126d dd02
```

```
La dimensione totale di questo pacchetto e' 164byte
```

Nota: se volete delle statistiche dettagliate installate **tcpdump** e **tcptrace,** entrambi gratuiti. Generate un file di dump con tcpdump aggiungendo "-w nomefile". Quindi eseguite "tcptrace -l nomefile" e avrete anche la bandwidth coinvolta nello scambio dati.

Nota: utilizzate sempre **tcpdump** per verificare che le conversazioni sono effettivamente trasmesse in modalita' "trunk". Potrebbe capitare che il trunk non va up e automaticamente i dati viaggiano in modalita' "no-trunk", a volte anche in una sola direzione. Questo e' immediatamente visibile con tcpdump.

8.0 Conclusioni

La banda dati necessaria per trasmettere la VoIP puo' variare significativamente al variare dei protocolli e dei codec utilizzati.

Per ottimizzare la trasmissione di piu' canali voce si puo' aumentare il sample period se la vostra implementazione del codec ve lo permette, oppure utilizzare un protocollo di trunking come IAX. Attenzione al fatto che, se usate un buffer di **jitter** (ovvero di riordino di pacchetti voce se arrivati nella sequenza errata), questo deve essere un multiplo del sample period.

http://www.itesys.it/
Tel. 095434534
Fax. 0957164045

Saranno benvenuti commenti e suggerimenti. Potete inviare anche correzioni o materiale aggiuntivo purche' originali e frutto del vostro personale lavoro. L'indirizzo email e' gianrico@gianrico.com