



TÉCNICO
LISBOA

REDES E SISTEMAS DE INTERNET

Relatório de Projecto VOIP

Autores: Grupo 9

Sophie Taboada 84187
Helena Oliveira 87017
Tiago Cabecinha 87124

Professor:

Paulo Rogério Barreiros D'Almeida
Pereira

22 de Dezembro de 2019

Conteúdo

1	Introdução e Objectivos	2
2	Instruções para correr o projecto	2
2.1	Asterisk files	2
2.2	sound files	2
2.3	scriptAGI files	3
2.4	googletts files	3
2.5	Correr o programa	3
3	Ilustração das pilhas de protocolos utilizados	4
3.1	Sinalização	4
3.2	Transferência de fluxos de Som e Vídeo	4
4	Descrição dos protocolos com base em diagramas temporais	5
4.1	Autenticação de um utilizador SIP e arranque da aplicação telefónica	5
4.1.1	Telefonar para o sistema de menus e ser atendido	7
4.1.2	Colocar a chamada em espera	9
4.1.3	Retomar chamada em espera	10
4.1.4	Desligar chamada	11
4.1.5	Escolher a opção “0” do menu	12
4.1.6	Escolher a opção inválida do menu	15
4.1.7	Cancelar a chamada antes de ser atendida	16
4.1.8	Utilizador liga para um utilizador inexistente	17
4.1.9	Aplicação telefónica termina	18
4.2	Alteração dos parâmetros DTMF nas chamadas do X-Lite	19
4.2.1	“Send via RFC 2833”	19
4.2.2	“Send via INFO”	21
4.3	Ligar ao operador	22
4.3.1	Sem o uso da linha “directmedia=no”	22
4.3.2	Com o uso da linha “directmedia=no”	24
5	Conclusão	26

1 Introdução e Objectivos

Este trabalho tem como objectivo o desenvolvimento de um sistema de *Interactive Voice Response* responsável pelo atendimento de chamadas automático baseado num PBX Asterisk. De facto, é desenvolvido um menu de interface com o utilizador que permite a este ter acesso a diferentes funcionalidades como aceder a jogos, votar em equipas de futebol, obter informações e efectuar chamadas ao operador 4000, conforme os números seleccionados por este.

São também estudados os diferentes protocolos utilizados para a sinalização como SIP e para a transmissão de fluxos de sons e/ou vídeos como RTP e/ou UDP, analisando o seu funcionamento e ilustrando-o com diagramas temporais. Para esta análise é utilizada a ferramenta Wireshark .

2 Instruções para correr o projecto

2.1 Asterisk files

No ficheiro comprimido disponibilizado na submissão deste projecto, encontra-se o ficheiro *extensions.conf* que contem todas as funcionalidades do sistema de *Interactive Voice Response*. O menu para o nosso projecto, "project_menu" , situa-se no final deste ficheiro. Este menu contém 5 funcionalidades e, segundo as extensões seleccionadas pelo utilizador, este pode:

- "Vote": votar em uma de duas equipas de futebol, uma vez por chamada;
- "HearVote": ouvir o número de votos de cada equipa;
- "ArtilleryGame": jogar ao jogo da artilharia;
- "PostalCode": inserir um código postal e receber a morada respetiva;
- Ligar ao operador 4000.

Uma vez na pasta do RSI-VoIP-84187-87017-87124.zip, o comando para guardar os ficheiros *extension.conf* e *sip.conf* na directória "/etc/asterisk/" , é o seguinte:

```
sudo mv sip.conf extensions.conf /etc/asterisk
```

Caso não tenha permissão para alterar os ficheiros desta pasta é necessário a utilização do comando "chmod".

Na secção "[globals]" do ficheiro *extensions.conf* é possível encontrar as localizações a que o programa vai buscar os ficheiros utilizados para a execução do VoIP.

2.2 sound files

Os ficheiros de som para a execução do nosso programa situam-se na pasta "sound" da pasta comprimida. A localização desta pasta segundo o ficheiro *extensions.conf* é "PATH=/home/rsi/sound". Caso o ficheiro não seja colocado numa directória com os mesmos nomes então terá de se efectuar o comando:

```
sudo cp -r sound /home/rsi
```

Na pasta "sound" existem diferentes pastas com os sons usados nas diferentes funcionalidades do nosso programa: "menu_options", "voting", "game" e "postal".

2.3 scriptAGI files

O script AGI desenvolvido em python encontra-se na pasta "scriptAGI" e chama-se "code.py". Este permite interagir com o Asterisk e fazer pedidos HTTP. A localização desta pasta segundo o ficheiro *extensions.conf* é "PATH_Python=/home/rsi/scriptAGI". Caso o ficheiro não seja colocado numa directória com os mesmos nomes então será efectuado o comando:

```
sudo cp -r scriptAGI /home/rsi
```

É necessário instalar a biblioteca "request-unixsocket" no sistema através do comando:

```
sudo apt-get install python-requests-unixsocket
```

Esta biblioteca é responsável por efectuar o pedido POST HTTP para o endereço: http://www.ctt.pt/feapl_2/app/open/postalCodeSearch/postalCodeSearch.jspx com argumentos correspondendo ao código postal inserido pelo utilizador no Asterisk.

2.4 googletts files

De modo ao Asterisk poder enunciar ao utilizador as respostas provenientes do código AGI sobre a morada respectiva ao código postal inserido por este, foi escolhida a utilização dos ficheiros da pasta "googletts". O googletts permite a tradução em tempo real de texto disponível em vários idiomas, dos quais um deles o português. A localização desta pasta segundo o ficheiro *extensions.conf* é "PATH_GOOGLERTS=/home/rsi/googletts". Caso o ficheiro não seja colocado numa directória com os mesmos nomes então será efectuado o comando:

```
sudo cp -r googletts /home/rsi
```

2.5 Correr o programa

Recomenda-se correr o servidor Asterisk na máquina nscotia da rede do laboratório ou um servidor Asterisk numa máquina virtual Linux. Se o Asterisk estiver a correr como daemon, pode-se criar uma consola com o comando (com sudo, excepto na máquina nscotia):

```
sudo asterisk -vvvvr
```

Para carregar o ficheiro *extensions.conf*, necessário para a execução do programa é usado o comando:

```
dialplan reload
```

E finalmente, para carregar o ficheiro *sip.conf*, necessário para a execução do programa é usado o comando:

```
sip reload
```

Do lado do utilizador, utiliza-se o programa X-Lite (versão 3 ou 5) no windows, utilizando a *account setting* de um *peer* presente no ficheiro *sip.conf* e ligando para o número 84187.

3 Ilustração das pilhas de protocolos utilizados

3.1 Sinalização

O protocolo utilizado para estabelecer, modificar e terminar sessões de comunicação através de um endereço IP consiste no protocolo de sinalização multimédia SIP (Session Initiation Protocol) [1] usado para VoIP.

O SIP é um protocolo baseado no protocolo HTTP, no que toca à gestão das chamadas, permite efectuar as funções enunciadas, de maneira totalmente independente do conteúdo de dados que estas transportam. Tem mensagens de pedido e de respostas semelhantes: as mensagens de pedido começam com uma linha que identifica o tipo de mensagem, o domínio SIP e a versão do protocolo. Depois seguem-se linhas de cabeçalho, uma linha em branco e um corpo de mensagem opcional. As mensagens de resposta têm um código de resposta e uma explicação textual.

Considerando que uma sessão corresponde a uma troca de dados entre diferentes *Endpoints*, é necessário que existam pelo menos dois participantes. Neste projecto, o tráfico de dados que circula entre participantes sob protocolo SIP corresponde a chamadas de voz. Estes dados são transportados por TCP ou UDP. A pilha das camadas utilizadas para a execução deste protocolo estão representadas na figura 1.

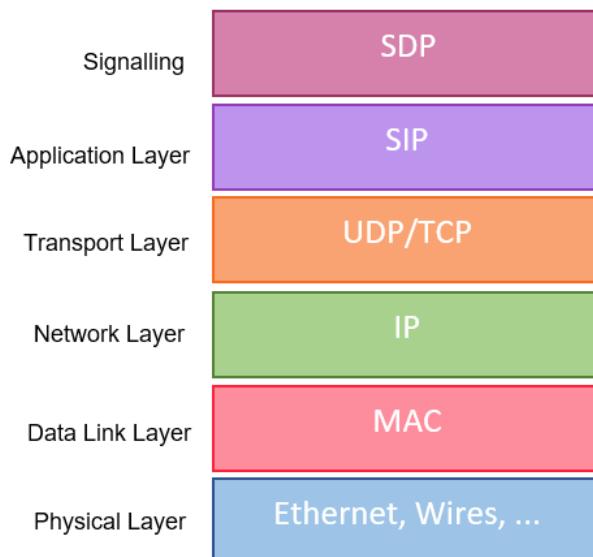


Figura 1: Pilha do Protocolo SIP

O protocolo SDP (Session Description Protocol) é então utilizado para descrever as sessões SIP com uma linguagem estruturada. Este utiliza um padrão para definir os parâmetros numa troca de dados entre participantes durante a chamada.

3.2 Transferência de fluxos de Som e Vídeo

O protocolo utilizado para o fluxo de som e vídeos é o protocolo RTP (Real-Time Transport). Este protocolo fornece funções de transporte de rede, adequadas para aplicações que transmitem dados em tempo real e para um único ou múltiplos destinos, sendo o caso

para a transmissão de áudio ou vídeo. O RTP não oferece por si mecanismos de garantia de entrega, entrega ordenada ou qualidade de serviço (QoS), funcionando normalmente sob UDP. Contrariamente ao protocolo SIP, o RTP não necessita da resposta de recepção dos seus pacotes, pelo que permite ter menos atrasados na transmissão de pacotes.

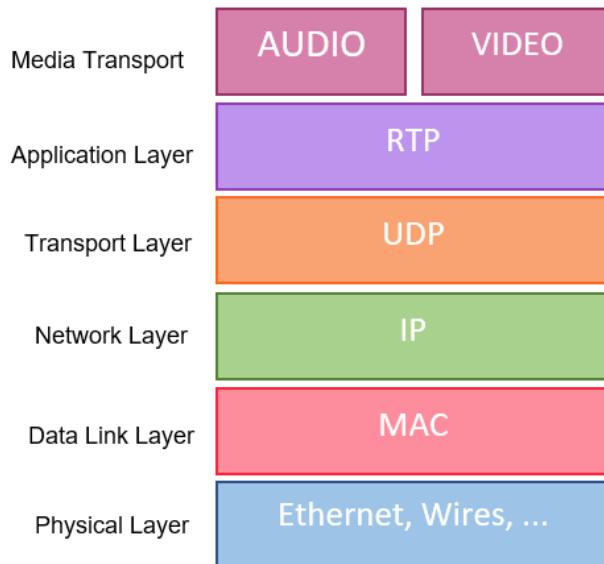


Figura 2: Pilha do Protocolo RTP

4 Descrição dos protocolos com base em diagramas temporais

As comunicações realizadas através dos protocolos descritos acima envolvem no mínimo um utilizador, um telefone e um PBX asterisk. Um utilizador, através de um telefone (aplicação telefónica) consegue registar-se no PBX e usufruir das funcionalidades desenvolvidas neste projeto. O utilizador apenas interage com a sua aplicação telefónica, e a aplicação telefónica e o PBX interagem entre si.

Na próxima secção será demonstrado o funcionamento dos protocolos utilizados e necessários para um conjunto de situações através de diagramas temporais, capturas de pacotes pelo programa *Wireshark* e também através de uma explicação textual.

4.1 Autenticação de um utilizador SIP e arranque da aplicação telefónica

Uma vez que a autenticação de um utilizador SIP ocorre aquando do arranque da aplicação telefónica por cada utilizador, tomou-se liberdade para representar o diagrama de tempo relativo a estas dois procedimentos num só.

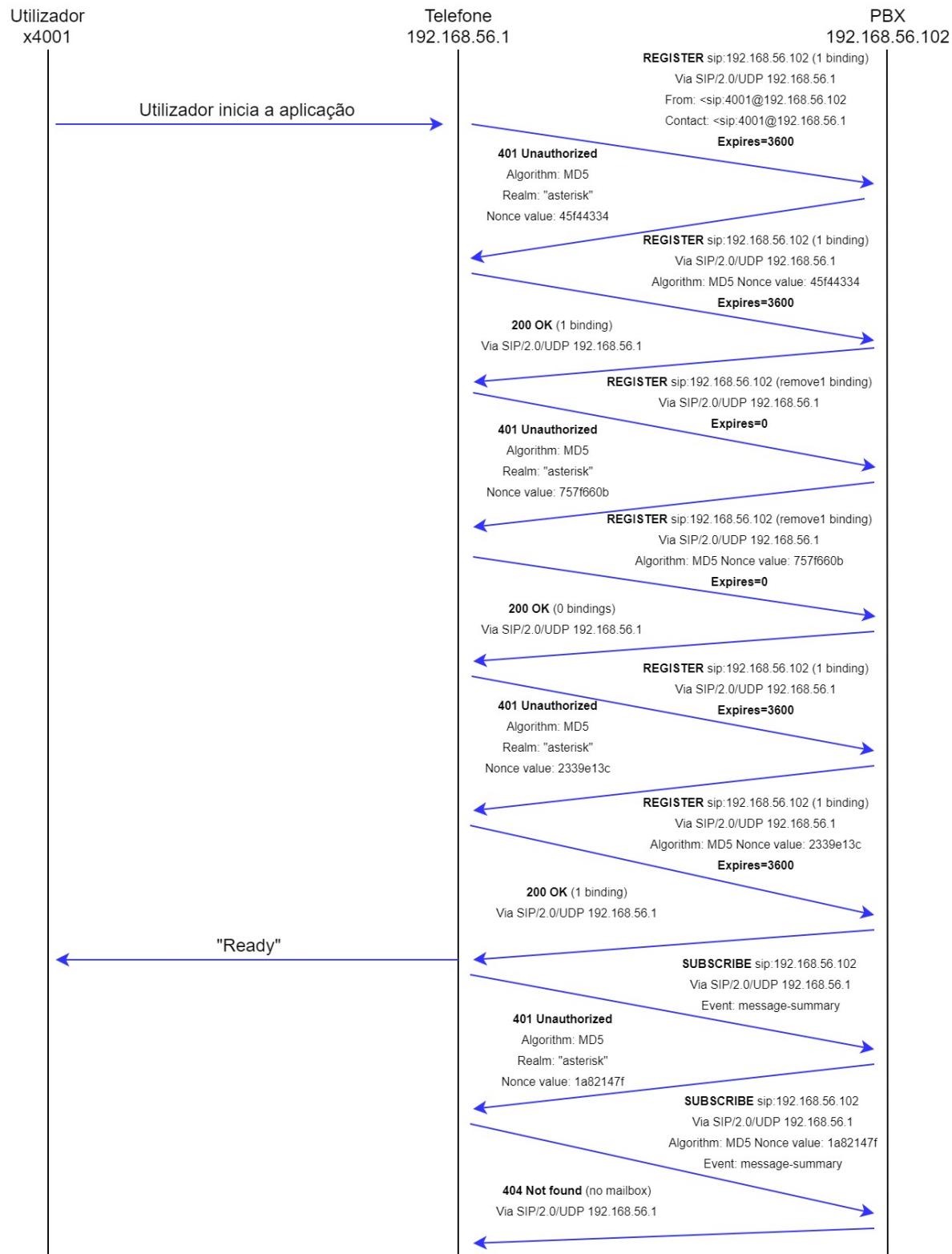


Figura 3: Diagrama temporal representando o registo de um utilizador SIP no arranque da aplicação.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.56.1	192.168.56.102	SIP	571	Request: REGISTER sip:192.168.56.102 (1 binding)
2	0.000256	192.168.56.102	192.168.56.1	SIP	613	Status: 401 Unauthorized
3	0.005624	192.168.56.1	192.168.56.102	SIP	727	Request: REGISTER sip:192.168.56.102 (1 binding)
4	0.006012	192.168.56.102	192.168.56.1	SIP	659	Status: 200 OK (1 binding)
5	0.013339	192.168.56.1	192.168.56.102	SIP	722	Request: REGISTER sip:192.168.56.102 (remove 1 binding)
6	0.013612	192.168.56.102	192.168.56.1	SIP	625	Status: 401 Unauthorized
7	0.018694	192.168.56.1	192.168.56.102	SIP	722	Request: REGISTER sip:192.168.56.102 (remove 1 binding)
8	0.380657	192.168.56.102	192.168.56.1	SIP	576	Status: 200 OK (0 bindings)
9	0.388615	192.168.56.1	192.168.56.102	SIP	727	Request: REGISTER sip:192.168.56.102 (1 binding)
10	0.388952	192.168.56.102	192.168.56.1	SIP	625	Status: 401 Unauthorized
11	0.395221	192.168.56.1	192.168.56.102	SIP	727	Request: REGISTER sip:192.168.56.102 (1 binding)
12	0.395540	192.168.56.102	192.168.56.1	SIP	659	Status: 200 OK (1 binding)
13	0.759888	192.168.56.1	192.168.56.102	SIP	645	Request: SUBSCRIBE sip:4001@192.168.56.102
14	0.760402	192.168.56.102	192.168.56.1	SIP	614	Status: 401 Unauthorized
15	0.765722	192.168.56.1	192.168.56.102	SIP	806	Request: SUBSCRIBE sip:4001@192.168.56.102
16	0.766098	192.168.56.102	192.168.56.1	SIP	548	Status: 404 Not found (no mailbox)

Figura 4: Captura Wireshark representando o registo de um utilizador SIP no arranque da aplicação.

Na abertura da aplicação XLite pela primeira vez por um utilizador, é quando ocorre o registo desse mesmo utilizador, através do protocolo de sinalização SIP, no PBX Asterisk.

Verifica-se na captura da figura 4 que o primeiro pacote capturado é uma tentativa de registo do utilizador com a mensagem REGISTER que através de uma variável chamada *expires* indica se este se está a tentar registar (*expires* = 3600) ou a fechar um registo anterior (*expires* = 0). Como se pode ver tanto na captura do Wireshark da figura 4 como no diagrama temporal da figura 3, a este pedido é recebida a mensagem *401 Unauthorized*, informando com que algoritmo de descodificação (MD5) e que Nonce o utilizador deve enviar na mensagem de registo. O Nonce é um valor aleatório utilizado para a cifra e é sempre diferente para cada mensagem de modo a preservar a segurança. É indicado nesta mensagem também qual o domínio (asterisk) e o esquema de autenticação (digest). De seguida, o utilizador envia então a mesma mensagem inicial mas com estas informações, o seu pedido é correspondido e o utilizador regista-se recebendo uma mensagem *200 OK*.

Após este registo, a aplicação reenvia de novo uma mensagem REGISTER mas desta com o *expires* a 0 de modo a terminar o registo realizado agora mesmo. Isto serve para limpar quaisquer informações que pudessem ter ficado numa sessão/registo anterior. Após o término deste registo, o processo repete-se exatamente da mesma forma e o utilizador regista-se. Essa informação pode ser vista no PBX Asterisk também, na figura 5.

```
=====
Connected to Asterisk 13.18.3-dfsg-1ubuntu4 currently running on ubuntu18 (pid = 869)
-- Registered SIP '4001' at 192.168.56.1:49580
-- Unregistered SIP '4001'
-- Registered SIP '4001' at 192.168.56.1:49580
[Dec 10 11:44:04] NOTICE[1268]: chan_sip.c:28410 handle_request_subscribe: Received SIP subscribe for peer without mailbox: 4001
ubuntu18*CLI>
```

Figura 5: Informações de registo no PBX Asterisk.

Por fim, o utilizador com a mensagem SUBSCRIBE tenta subscrever/ter acesso ao *voicemail* especificando o evento *message-summary*. A este pedido, uma vez que não foi desenvolvida esta funcionalidade, o PBX responde com uma mensagem de erro *404 not found*. Também é possível observar na figura 5 a tentativa de subscrição do *voicemail*.

4.1.1 Telefonar para o sistema de menus e ser atendido

Nesta secção é estudada a situação em que o utilizador liga para o sistema de menus e é atendido.

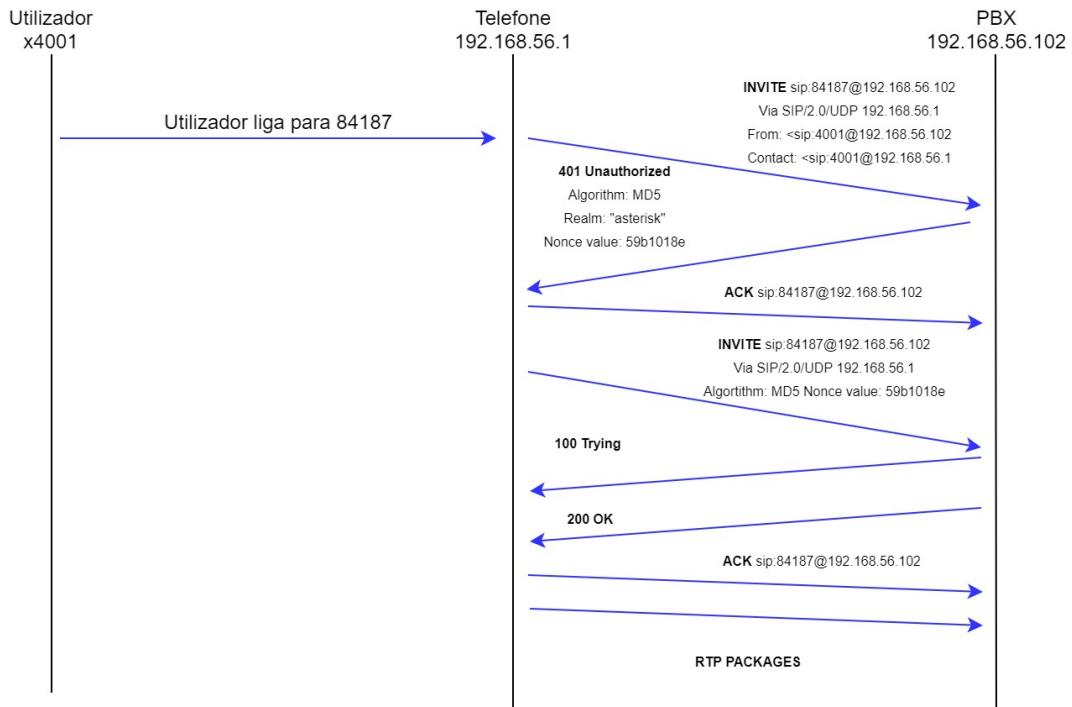


Figura 6: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador liga para o sistema de menus e é atendido.

sip or rtp or sdp						
No.	Time	Source	Destination	Protocol	Length	Info
43	16.871385	192.168.56.1	192.168.56.102	SIP/SDP	948	Request: INVITE sip:84187@192.168.56.102
44	16.871750	192.168.56.102	192.168.56.1	SIP	612	Status: 401 Unauthorized
45	16.872608	192.168.56.1	192.168.56.102	SIP	371	Request: ACK sip:84187@192.168.56.102
46	16.877968	192.168.56.1	192.168.56.102	SIP/SDP	1110	Request: INVITE sip:84187@192.168.56.102
47	16.878905	192.168.56.102	192.168.56.1	SIP	557	Status: 100 Trying
48	16.879712	192.168.56.102	192.168.56.1	SIP/SDP	881	Status: 200 OK
49	16.979519	192.168.56.1	192.168.56.102	SIP	490	Request: ACK sip:84187@192.168.56.102:5060
50	16.983659	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31030, Time=1
51	16.983699	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31031, Time=2
52	16.983716	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31032, Time=3
53	16.983734	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31033, Time=4
54	16.983754	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31034, Time=5
55	16.983769	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31035, Time=6
56	16.983789	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31036, Time=7
57	16.983802	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31037, Time=8
58	16.983819	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31038, Time=9
59	16.983830	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31039, Time=10
60	16.983846	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31040, Time=11
61	16.983869	192.168.56.1	192.168.56.102	RTP	54	PT=DynamicRTP-Type-126, SSRC=0xFF11142, Seq=31041, Time=12
62	16.993123	192.168.56.1	192.168.56.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xFF11142, Seq=31042, Time=0, Mark

Figura 7: Captura Wireshark que exemplifica a troca de pacotes quando o utilizador liga para o sistema de menus e é atendido.

O utilizador para aceder ao sistema de menus, terá de marcar o menor número de aluno dos elementos do grupo, neste caso, 84187. Assim que digita este número e pressiona para chamar, a troca de pacotes entre a aplicação telefónica e o PBX Asterisk começa. É enviado uma mensagem INVITE para a extensão 84187. Da mesma forma que é efetuado o registo, este processo necessita também de ter uma codificação específica. Esta codificação vem na mensagem de erro **401 Unauthorized**. De seguida, é enviada uma mensagem ACK de recepção da mensagem de erro e de novo uma mensagem INVITE com a codificação correta. A chamada é atendida quase instantaneamente. É recebido pela aplicação telefónica uma mensagem STATUS TRYING em que se entende que se está a tentar estabelecer ligação e logo de seguida, é recebida uma mensagem que confirma

a conexão efetuada entre o utilizador e o sistema de menus, uma mensagem que indica STATUS OK. Por último é enviado novamente pela aplicação um ACK que confirma a recepção destas duas mensagens e o utilizador começa a enviar os pacotes de dados relacionados com o sistema de menus, através do protocolo RTP. O fluxo de dados RTP inicia-se. Toda esta descrição pode ser observada quer na captura presente na figura 7 do *Wireshark* ou no diagrama temporal na figura 6.

4.1.2 Colocar a chamada em espera

Neste caso analisar-se-á a situação que o utilizador coloca a chamada em espera. Para tal, analisaremos os pacotes capturados pelo *Wireshark* na figura 9.

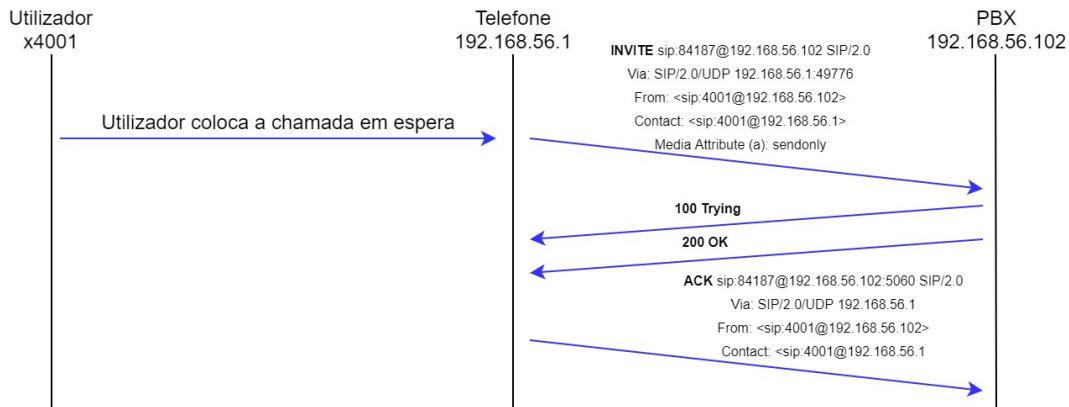


Figura 8: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador põe a chamada em espera.

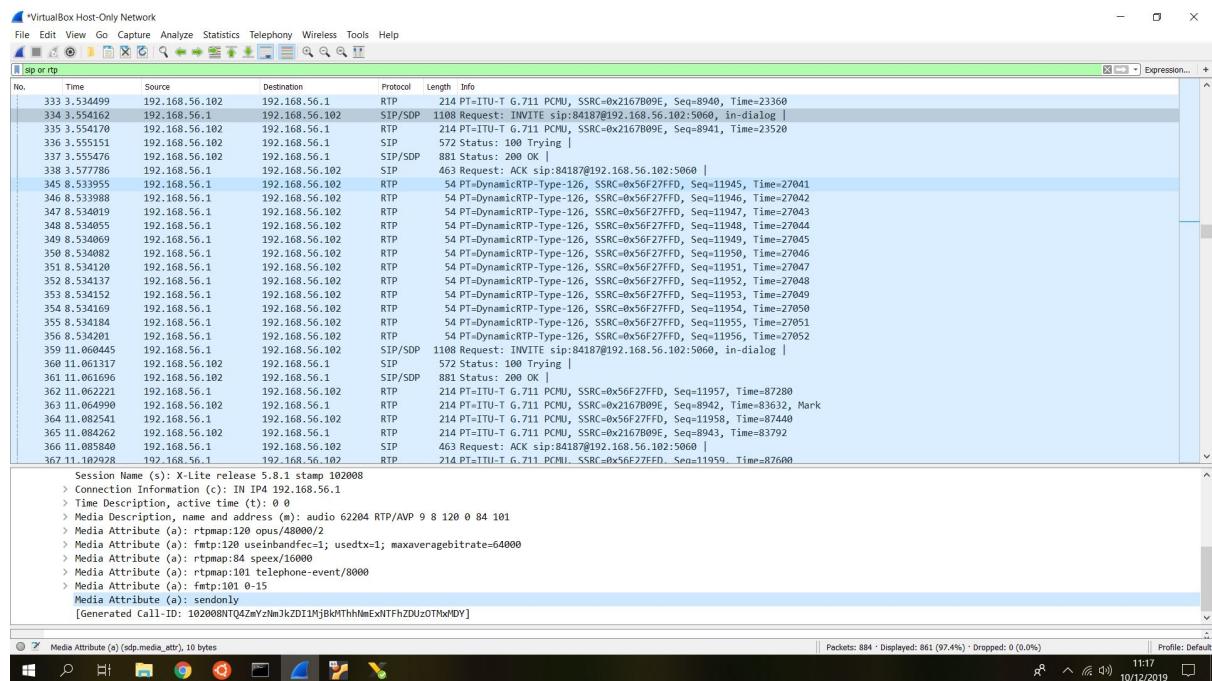


Figura 9: Captura de Wireshark na altura que a chamada entra em espera.

Até ao momento que o utilizador coloca a chamada em espera, o tráfego é realizado nos dois sentidos. A partir do momento em que é enviada uma mensagem INVITE por

parte do utilizador para a extensão 84187 com um atributo de *media sendonly*, significa que se pretende colocar o servidor em espera, ou seja, não receber pacotes do mesmo, e o tráfego de pacotes RTP pára de circular. A esta mensagem, o servidor responde com um mensagem *100 Trying* seguida de *200 OK* com um atributo *media recvonly*. É enviado um ACK por parte do utilizador e a partir desse momento, a chamada entra em espera. Daqui em diante o tráfego é realizado apenas num sentido, do utilizador para o servidor PBX como se pode observar no *Wireshark* da figura 9. É apresentado na figura 8 o diagrama temporal que mostra a troca de pacotes entre utilizador e servidor quando a chamada é colocada em espera. De seguida é analisada a situação em que se retoma a chamada.

4.1.3 Retomar chamada em espera

Para se identificar o momento em que a chamada é retomada após estar em espera, analisa-se mais uma vez as capturas de pacotes do *Wireshark* na figura 11.

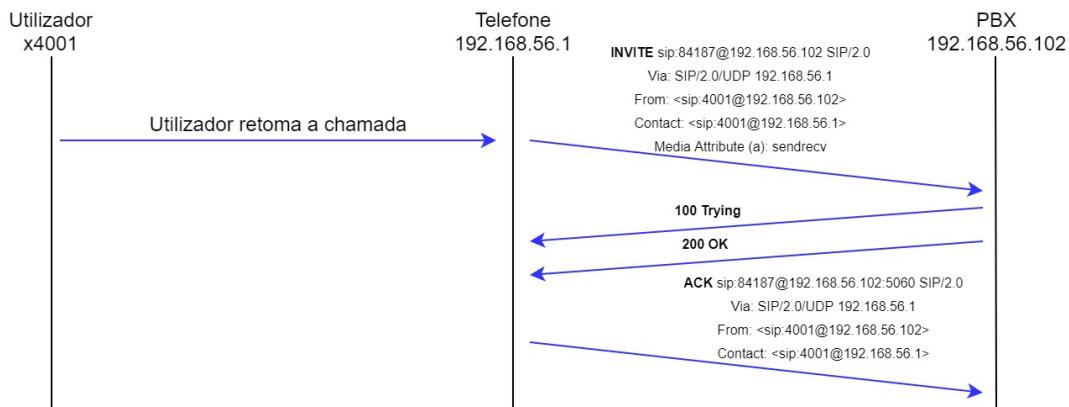


Figura 10: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador retoma a chamada.

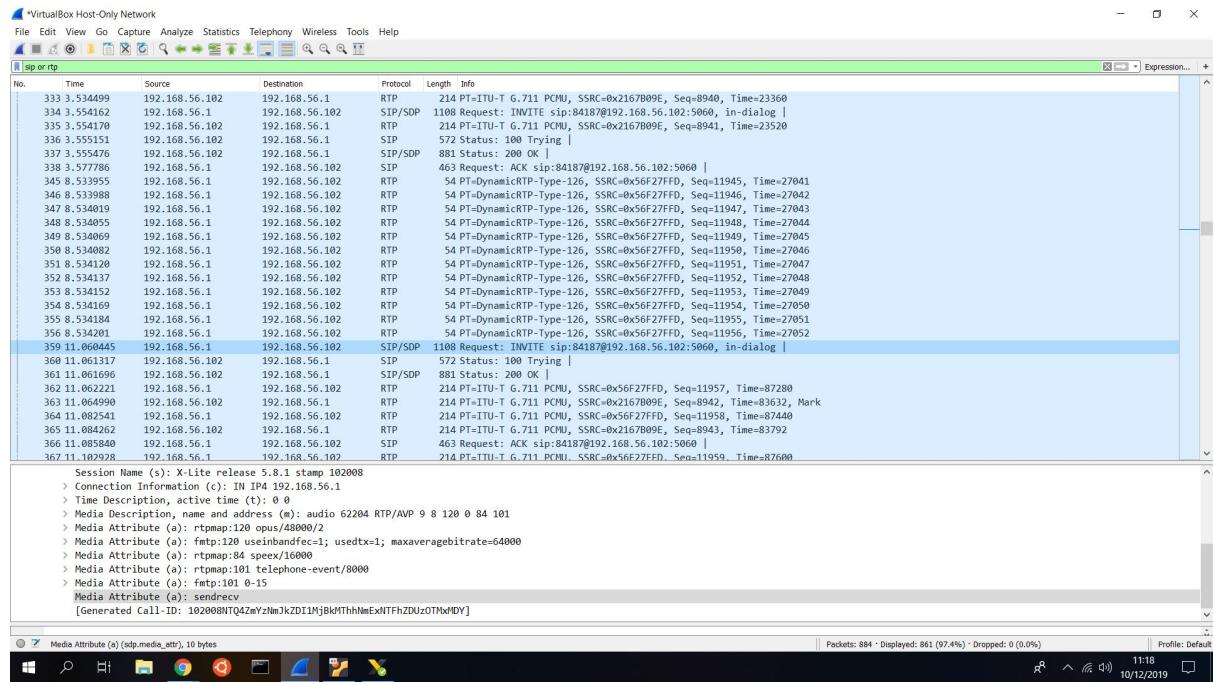


Figura 11: Captura de Wireshark na altura que a chamada é retomada.

Verifica-se que o fluxo, como referido anteriormente, realiza-se apenas num sentido enquanto a chamada se encontra em espera, do utilizador para o servidor PBX Asterisk.

Esta situação altera-se quando o utilizador envia novamente uma mensagem INVITE para a extensão 84187 com o atributo *media* alterado para *sendrcv*, significando que o utilizador quer voltar ao estado da chamada de mandar e receber pacotes. Isto pode ser verificado na figura 11, em que está destacado exatamente o pacote relativo a esta mensagem. Daí em diante, após ser recebido por parte do utilizador uma mensagem *100 Trying* e *200 OK* também com o atributo *media sendrcv* e este enviar um ACK, a chamada deixa de se encontrar em espera e retomam as comunicações em ambos os sentidos entre utilizador e servidor através da aplicação telefónica. É possível observar o diagrama temporal na figura 10 relativo a esta situação que explica visualmente os pacotes trocados nesta situação.

4.1.4 Desligar chamada

Quando o utilizador desliga a chamada, verifica-se a seguinte troca de pacotes observada na captura do Wireshark apresentada na figura 12.

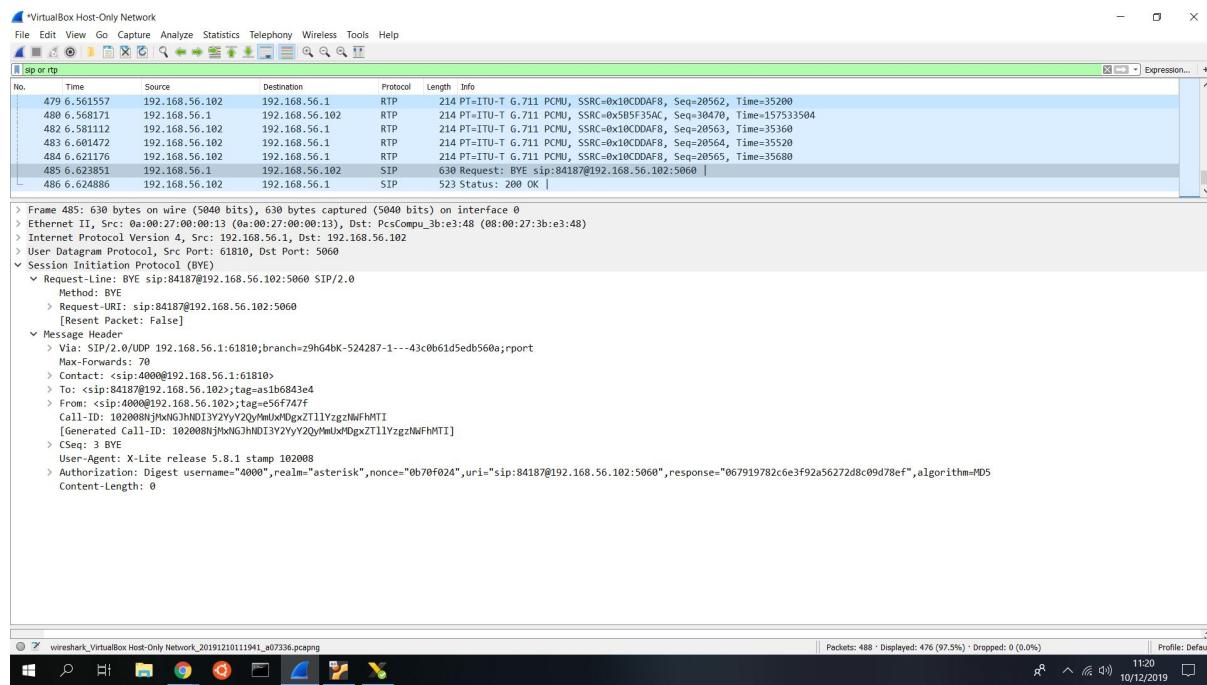


Figura 12: Captura de Wireshark na altura em que a chamada é desligada.

É simplesmente enviado um pacote, por parte do utilizador, com *Request BYE* para o servidor PBX Asterisk. A este pedido, o servidor responde com um pacote com *200 OK*. A partir deste momento, a chamada é encerrada e a troca de pacotes entre os dois termina.

É representado na figura 13 o diagrama temporal correspondente a esta situação.

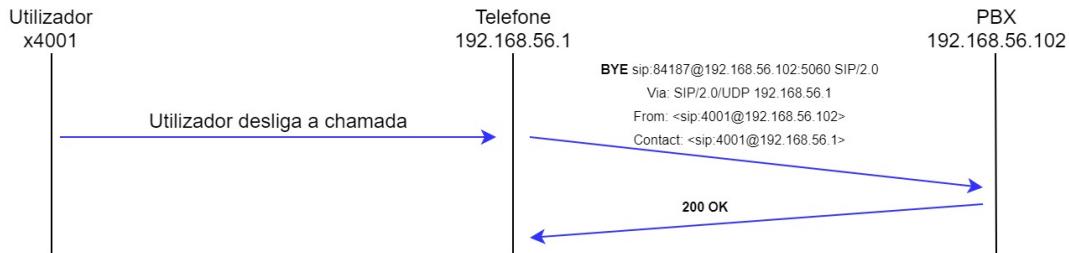


Figura 13: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador desliga a chamada.

4.1.5 Escolher a opção “0” do menu

Quando comparado o resultado de desligar a chamada no botão de desligar ou premindo o número 0, não existe diferença nenhuma, a chamada é desligada de qualquer maneira.

Mas se formos comparar o que acontece em cada situação analisando os pacotes capturados, verifica-se uma grande diferença.

A situação em que se desliga a chamada no botão já foi analisada na secção anterior. Analisar-se-á agora a opção do utilizador premir o número 0 a partir das figuras 14 e 15.

No.	Time	Source	Destination	Protocol	Length	Info
647	12.299899	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18182, Time=48800
648	12.317036	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
649	12.317120	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
650	12.317137	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
651	12.317730	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18183, Time=48960
652	12.331609	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18184, Time=49120
653	12.337224	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
654	12.353413	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18185, Time=49280
655	12.357380	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
656	12.374728	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18186, Time=49440
657	12.377238	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
658	12.396238	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18187, Time=49600
659	12.396988	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
660	12.417520	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
661	12.418028	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18188, Time=49760
662	12.434783	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18189, Time=49920
663	12.437137	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0
664	12.457363	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0 (end)
665	12.457410	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0 (end)
666	12.457451	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Zero 0 (end)
667	12.458080	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x29A4754F, Seq=18190, Time=50080
668	12.459727	192.168.56.102	192.168.56.1	SIP	672	Request: BYE sip:4000@192.168.56.1:52130;rinstance=437ebbe5953396a7
669	12.466142	192.168.56.1	192.168.56.102	RTCP	130	Receiver Report Source description Extended report (RFC 3611) Goodbye
670	12.515281	192.168.56.1	192.168.56.102	SIP	433	Status: 200 OK
671	25.286843	192.168.56.1	192.168.56.102	UDP	46	52130 → 5060 Len=4

> Frame 648: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface 0
> Ethernet II, Src: 0a:00:27:00:00:13 (0a:00:27:00:00:13), Dst: PcsCompu_3b:e3:48 (08:00:27:3b:e3:48)
> Internet Protocol Version 4, Src: 192.168.56.1, Dst: 192.168.56.102
> User Datagram Protocol, Src Port: 55356, Dst Port: 13288
> Real-Time Transport Protocol
> RFC 2833 RTP Event

0000	08 00 27 3b e3 48	0a 00 27 00 00 13	08 00 45 00	..';1...E-
0010	00 2c 3e 69 00 00 80 11	0a a0 c0 a8 38 01 c0 a8		,>i..... .8...
0020	38 66 d8 3c 33 e8 00 18	2b 9f 80 e5 16 00 00 00	8f-<3... +.....	
0030	cd 00 19 9d 58 14 00 0a	00 a0X... ..	

Figura 14: Captura de Wireshark na altura em que o utilizador seleciona a opção “0”.

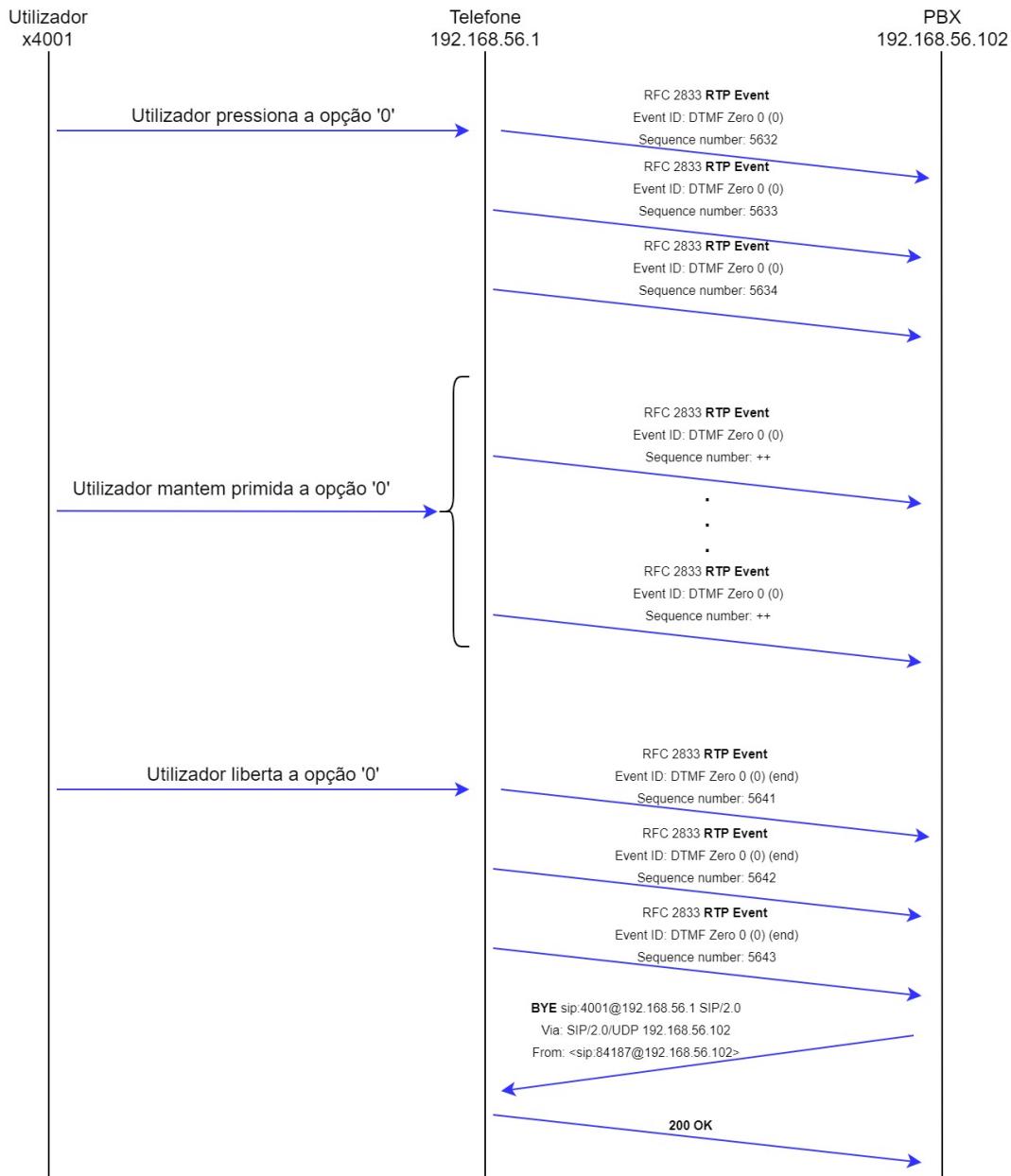


Figura 15: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador seleciona a opção “0”.

Neste caso, verifica-se que o utilizador envia 3 pacotes *RTP Event* indicando o botão clicado. De seguida, serão enviados periodicamente, enquanto o botão continuar premido pelo utilizador (mesmo que se tratem de pouquíssimos segundos), o mesmo tipo de pacotes até o utilizador deixar de pressionar o botão 0. Esta situação é identificada pelo PBX asterisk uma vez que quando o utilizador deixa de premir o botão, são enviados novamente 3 pacotes *RTP Event* indicando o botão clicado com uma descrição *end*.

Uma vez que ao escolher a opção 0, esta associa-se ao término da chamada, a chamada é encerrada, mas desta vez por parte do PBX Asterisk com um pacote *BYE* para o utilizador. Este responde com um pacote *200 OK* e a chamada é terminada.

4.1.6 Escolher a opção inválida do menu

Nesta secção é estudada a situação em que o utilizador escolhe uma opção inválida do sistema de menu. Para tal, observa-se a captura *Wireshark* presente na figura 16.

No.	Time	Source	Destination	Protocol	Length	Info
267	6.030763	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13402, Time=18880
268	6.051206	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13403, Time=19040
269	6.063969	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
270	6.064062	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
271	6.064113	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
272	6.071488	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13404, Time=19200
273	6.082071	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
274	6.091376	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13405, Time=19360
275	6.100405	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
276	6.110745	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13406, Time=19520
277	6.126894	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
278	6.130852	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13407, Time=19680
279	6.144815	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
280	6.151090	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13408, Time=19840
281	6.163775	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
282	6.171172	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13409, Time=20000
283	6.180832	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6
284	6.190876	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13410, Time=20160
285	6.198822	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6 (end)
286	6.198897	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6 (end)
287	6.198938	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Six 6 (end)
288	6.202184	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13411, Time=20320
289	6.223026	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x4F237274, Seq=13412, Time=20480

> Frame 269: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface 0
> Ethernet II, Src: 0a:00:27:00:00:13 (0a:00:27:00:00:13), Dst: PcsCompu_3b:e3:48 (08:00:27:3b:e3:48)
> Internet Protocol Version 4, Src: 192.168.56.1, Dst: 192.168.56.102
> User Datagram Protocol, Src Port: 57094, Dst Port: 16698
> Real-Time Transport Protocol
`> RFC 2833 RTP Event
 Event ID: DTMF Six 6 (6)
 0... = End of Event: False
 .0... = Reserved: False
 ..00 1010 = Volume: 10
 Event Duration: 160

Figura 16: Captura de *Wireshark* na altura em que o utilizador pressiona uma tecla inválida.

Neste caso, acontece exatamente o mesmo que se sucedeu após se clicar no número 0, mas neste caso numa opção inválida. Ou seja, o utilizador envia 3 pacotes *RTP Event* indicando o botão clicado, neste caso, o número 6, que não corresponde a nenhum número válido do menu. De seguida, enquanto o botão está premido, são enviados periodicamente mais pacotes deste tipo até o utilizador deixar de premir o botão. Mais uma vez, esta situação é identificada pelo utilizador através dos 3 pacotes *RTP Event* indicando que o botão clicado é o 6 mas desta vez com um atributo *end*.

Não acontece nada visto que a tecla 6 não está associada a nenhuma extensão do sistema de menu e o servidor PBX Asterisk continua no sistema de menu.

A figura 17 representa o diagrama temporal para a situação descrita.

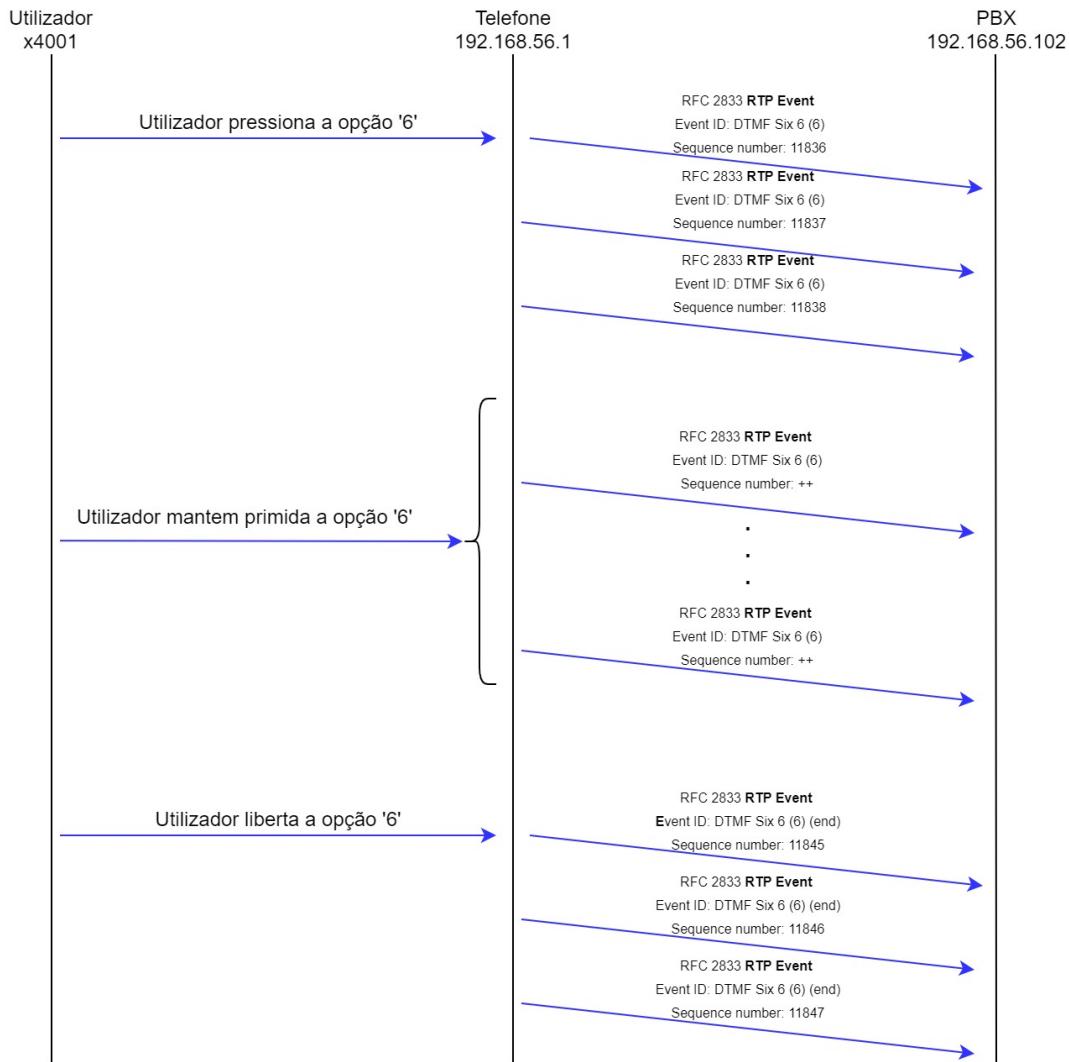


Figura 17: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador pressiona uma tecla inválida.

4.1.7 Cancelar a chamada antes de ser atendida

Irá ser analisada o caso em que o utilizador realiza a chamada para o sistema de menus mas termina a chamada antes de esta poder ser atendida. À semelhança da secção em que se liga para o sistema de menus e a chamada é atendida, é enviado um pacote INVITE para a extensão 84187. Ora, tal como anteriormente, este pacote necessita de ter uma codificação específica. Esta informação chega na mensagem que o servidor envia com *401 Unauthorized* com o respetivo algoritmo (MD5) e o um Nonce value aleatório, que o utilizador terá de enviar no INVITE seguinte. Assim acontece. De seguida o servidor responde com um pacote *100 Trying* e antes de este conseguir enviar o pacote *200 OK*, o utilizador envia um pacote CANCEL para a extensão 84187 novamente. A esta situação, o servidor envia um pacote *Request Terminated* seguido de um pacote *200 OK*. Por último, o utilizador envia um ACK confirmado a recepção destes pacotes. Foi possível a compreensão deste caso através da análise dos pacotes capturados na figura 18 e posteriormente representados no diagrama temporal da figura 19.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.56.1	192.168.56.102	SIP/SDP	948	Request: INVITE sip:84187@192.168.56.102
2	0.000400	192.168.56.102	192.168.56.1	SIP	612	Status: 401 Unauthorized
3	0.001059	192.168.56.1	192.168.56.102	SIP	371	Request: ACK sip:84187@192.168.56.102
4	0.006179	192.168.56.1	192.168.56.102	SIP/SDP	1110	Request: INVITE sip:84187@192.168.56.102
5	0.007308	192.168.56.102	192.168.56.1	SIP	557	Status: 100 Trying
6	2.9992847	192.168.56.1	192.168.56.102	SIP	571	Request: CANCEL sip:84187@192.168.56.102
7	2.9993296	192.168.56.102	192.168.56.1	SIP	542	Status: 487 Request Terminated
8	2.9993326	192.168.56.102	192.168.56.1	SIP	526	Status: 200 OK
9	3.050772	192.168.56.1	192.168.56.102	SIP	371	Request: ACK sip:84187@192.168.56.102
12	5.355246	192.168.56.1	192.168.56.102	UDP	46	53645 → 5060 Len=4
13	5.604471	192.168.56.1	192.168.56.102	UDP	46	53552 → 5060 Len=4

```

> Frame 6: 571 bytes on wire (4568 bits), 571 bytes captured (4568 bits) on interface 0
> Ethernet II, Src: 0a:00:27:00:00:13 (0a:00:27:00:00:13), Dst: PcsCompu_3b:e3:48 (08:00:27:3b:e3:48)
> Internet Protocol Version 4, Src: 192.168.56.1, Dst: 192.168.56.102
> User Datagram Protocol, Src Port: 53552, Dst Port: 5060
`- Session Initiation Protocol (CANCEL)
  `- Request-Line: CANCEL sip:84187@192.168.56.102 SIP/2.0
    Method: CANCEL
    Request-URI: sip:84187@192.168.56.102
    [Resent Packet: False]
  `-- Message Header
    > Via: SIP/2.0/UDP 192.168.56.1:53552;branch=z9hG4bK-524287-1---7e8789441b5f447e;rport
    Max-Forwards: 70
    > To: <sip:84187@192.168.56.102>
    > From: <sip:4001@192.168.56.102>;tag=21c5d25b
    Call-ID: 102008MzMmMTUwYjk2MTE1YmJjODhjMmZ1MDM5YWUwOWNkMDY
    [Generated Call-ID: 102008MzMmMTUwYjk2MTE1YmJjODhjMmZ1MDM5YWUwOWNkMDY]
  `-- CSeq: 2 CANCEL
    Sequence Number: 2
    Method: CANCEL
    User-Agent: X-Lite release 5.8.1 stamp 102008
    Authorization: Digest username="4001",realm="asterisk",nonce="56cf00d8",uri="sip:84187@192.168.56.102",response="c1b74e54e30535ba9953af7dc3dc352e",Content-Length: 0
  
```

Figura 18: Captura de Wireshark na altura em que o utilizador cancela a chamada antes de esta ser atendida.

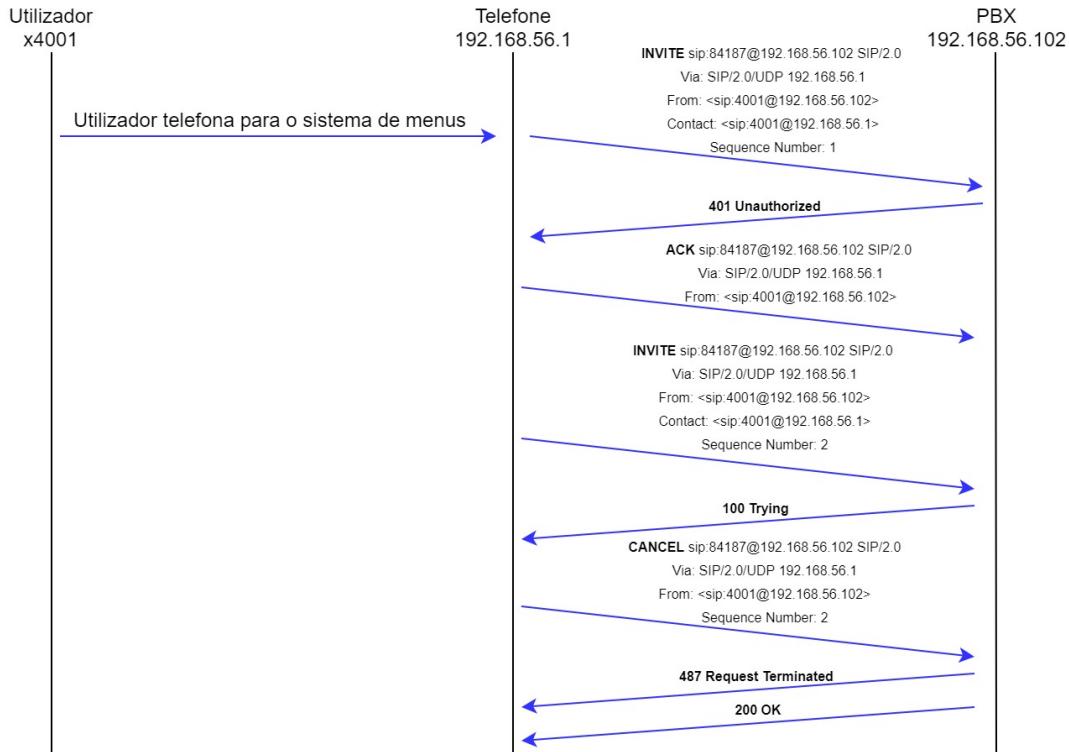


Figura 19: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador cancela a chamada antes de esta ser atendida.

4.1.8 Utilizador liga para um utilizador inexistente

Nesta secção estuda-se a situação em que se liga para um utilizador inexistente.

43 24.127386	192.168.56.1	192.168.56.102	SIP/SDP	950 Request: INVITE sip:852258@192.168.56.102
44 24.127963	192.168.56.102	192.168.56.1	SIP	613 Status: 401 Unauthorized
45 24.129011	192.168.56.1	192.168.56.102	SIP	373 Request: ACK sip:852258@192.168.56.102
46 24.135256	192.168.56.1	192.168.56.102	SIP/SDP	1113 Request: INVITE sip:852258@192.168.56.102
47 24.178555	192.168.56.102	192.168.56.1	SIP	534 Status: 404 Not Found
48 24.212138	192.168.56.1	192.168.56.102	SIP	373 Request: ACK sip:852258@192.168.56.102

Figura 20: Captura de Wireshark na altura em que se liga para um utilizador inexistente.

Verifica-se pela troca de pacotes, que inicialmente o utilizador envia um pacote INVITE para a extensão 852258. Novamente este pacote necessita de estar codificado pelo que após o envio da codificação com a mensagem *401 Unauthorized* e o utilizador envia de novo o INVITE, recebe-se um pacote com a informação STATUS *404 Not Found*. Este utilizador não existe e o servidor não conseguiu estabelecer ligação. O utilizador manda um pacote ACK dizendo que recebeu a informação.

É mostrado de seguida o diagrama temporal para esta situação na figura 21.

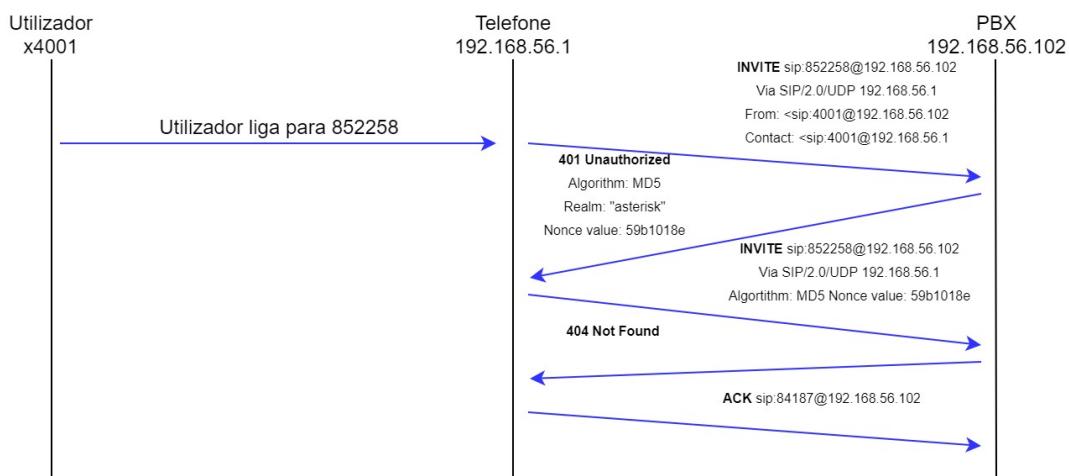


Figura 21: Diagrama temporal que exemplifica a troca de pacotes quando se liga para um utilizador inexistente.

4.1.9 Aplicação telefónica termina

Para capturar pacotes quando a aplicação telefónica termina, é necessário terminar a mesma carregando em *Softphone* e de seguida em *Exit*, e não apenas carregando no botão de fechar a janela da aplicação.

Este caso é também semelhante ao processo de registo de um utilizador. É enviado para o servidor PBX Asterisk um pacote REGISTER com o atributo *expires* com o valor 0. Este é primeiramente rejeitado uma vez que a mensagem não está devidamente codificada. Na mensagem *401 Unauthorized* são enviadas as informações relativas a esta codificação. Repete-se o processo com a codificação correta e o utilizador através da aplicação telefónica deixa de estar registado recebendo uma mensagem *200 OK* terminando assim a conexão total com o servidor. Mostra-se de seguida o diagrama temporal desenvolvido e as capturas realizadas pelo *Wireshark* nesta situação, nas figuras 22 e 23.

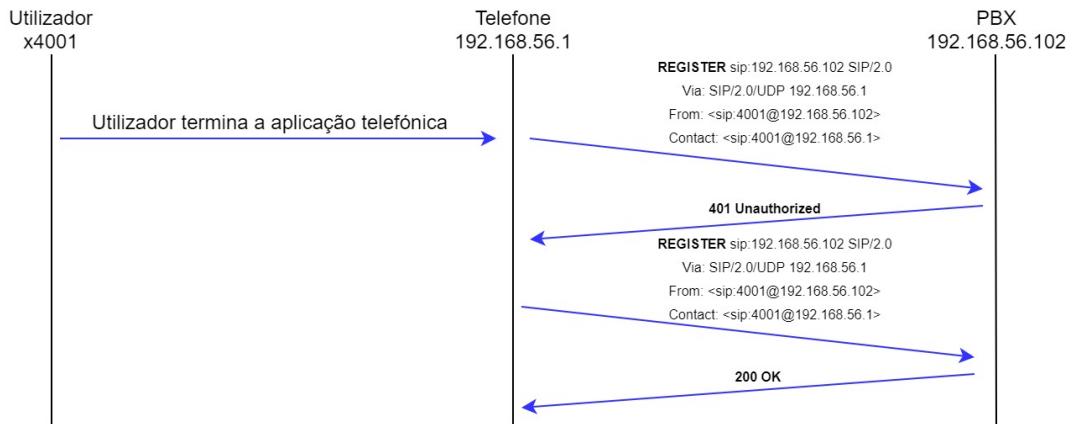


Figura 22: Diagrama temporal que exemplifica a troca de pacotes quando a aplicação telefónica termina.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.56.1	192.168.56.102	SIP	723	Request: REGISTER sip:192.168.56.102 (remove 1 binding)
2	0.000976	192.168.56.102	192.168.56.1	SIP	614	Status: 401 Unauthorized
3	0.014764	192.168.56.1	192.168.56.102	SIP	723	Request: REGISTER sip:192.168.56.102 (remove 1 binding)
4	0.015816	192.168.56.102	192.168.56.1	SIP	577	Status: 200 OK (0 bindings)
5	1.205724	192.168.56.1	192.168.56.255	UDP	86	57621 → 57621 Len=44

Figura 23: Captura de Wireshark na altura em que a aplicação telefónica termina.

4.2 Alteração dos parâmetros DTMF nas chamadas do X-Lite

Nesta secção será abordado a influência na alteração dos parâmetros DTMF nas chamadas do X-Lite, observando as suas diferenças analisando capturas realizadas pelo programa *Wireshark*.

4.2.1 “Send via RFC 2833”

Nesta primeira análise, verificaremos o que acontece quando o parâmetro de envio de pacotes pela aplicação X-Lite é realizado com *Send via RFC 2833*. Na figura 24 observamos os pacotes trocados entre a aplicação telefónica e o servidor PBX Asterisk.

No.	Time	Source	Destination	Protocol	Length	Info
759	8.933301	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27945, Time=57600
760	8.958003	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27946, Time=57760
761	8.970900	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
762	8.971002	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
763	8.971024	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
764	8.973339	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27947, Time=57920
765	8.991516	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
766	8.993514	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27948, Time=58080
767	9.010579	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
768	9.013423	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27949, Time=58240
769	9.031119	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
770	9.032897	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27950, Time=58400
771	9.051215	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
772	9.058472	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27951, Time=58560
773	9.071115	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
774	9.073628	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27952, Time=58720
775	9.090894	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2
776	9.093035	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27953, Time=58880
777	9.111670	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2 (end)
778	9.111751	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2 (end)
779	9.111769	192.168.56.1	192.168.56.102	RTP EVENT	58	Payload type=RTP Event, DTMF Two 2 (end)
780	9.113763	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27954, Time=59040
781	9.131223	192.168.56.1	192.168.56.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE5D4991, Seq=20601, Time=62880
782	9.133889	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x43491D48, Seq=27955, Time=59200
783	9.150904	192.168.56.1	192.168.56.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0xE5D4991. Seq=20602. Time=63040

> Frame 761: 58 bytes on wire (464 bits), 58 bytes captured (464 bits) on interface 0
 > Ethernet II, Src: 0a:00:27:00:00:13 (0a:00:27:00:00:13), Dst: PcsCompu_3b:e3:48 (08:00:27:3b:e3:48)
 > Internet Protocol Version 4, Src: 192.168.56.1, Dst: 192.168.56.102
 > User Datagram Protocol, Src Port: 49770, Dst Port: 17934
 > Real-Time Transport Protocol
 > RFC 2833 RTP Event

0000	08 00 27 3b e3 48 0a 00	27 00 00 13 08 00 45 00	...'; H.. '....E-
0010	00 2c 44 8c 00 00 80 11	04 7d c0 a8 38 01 c0 a8	..,D..... } -8 ...
0020	38 66 c2 6a 46 0e 00 18	e9 a0 80 e5 50 6d 00 00	8f .jF..... -Pm...
0030	f0 00 0e 5d 49 91 02 0a	00 a0	...]I.....

Figura 24: Captura de Wireshark com a configuração “Send via RFC 2833”.

Para uma qualquer acção do utilizador, neste caso, clicar no número 2, a aplicação telefónica começa por enviar 3 pacotes iniciais RTP EVENT indicando a tecla clicada. De seguida, é enviado periodicamente (50ms) pela aplicação telefónica. No entanto, o asterisk realiza actualizações de 20ms em 20ms enviando então o pacote neste período e não de 50ms em 50ms como seria de esperar. O envio de tantos pacotes periodicamente acontece porque estas comunicações são realizadas com meio de transporte UDP, que como se sabe, não é fiável, não garante a entrega dos pacotes. Assim, são enviados o maior número de pacotes possíveis de modo a garantir que pelo menos um pacote chegará ao destino. Não existe nenhuma mensagem que confirme a sua recepção. Por fim, como já foi visto noutras casas abordados, o Asterisk deteta o final de uma ação do utilizador quando recebe novamente 3 pacotes de seguida, indicando a tecla pressionada, mas desta vez com um atributo *end*.

É também apresentado o diagrama temporal que representa esta situação na figura 25.

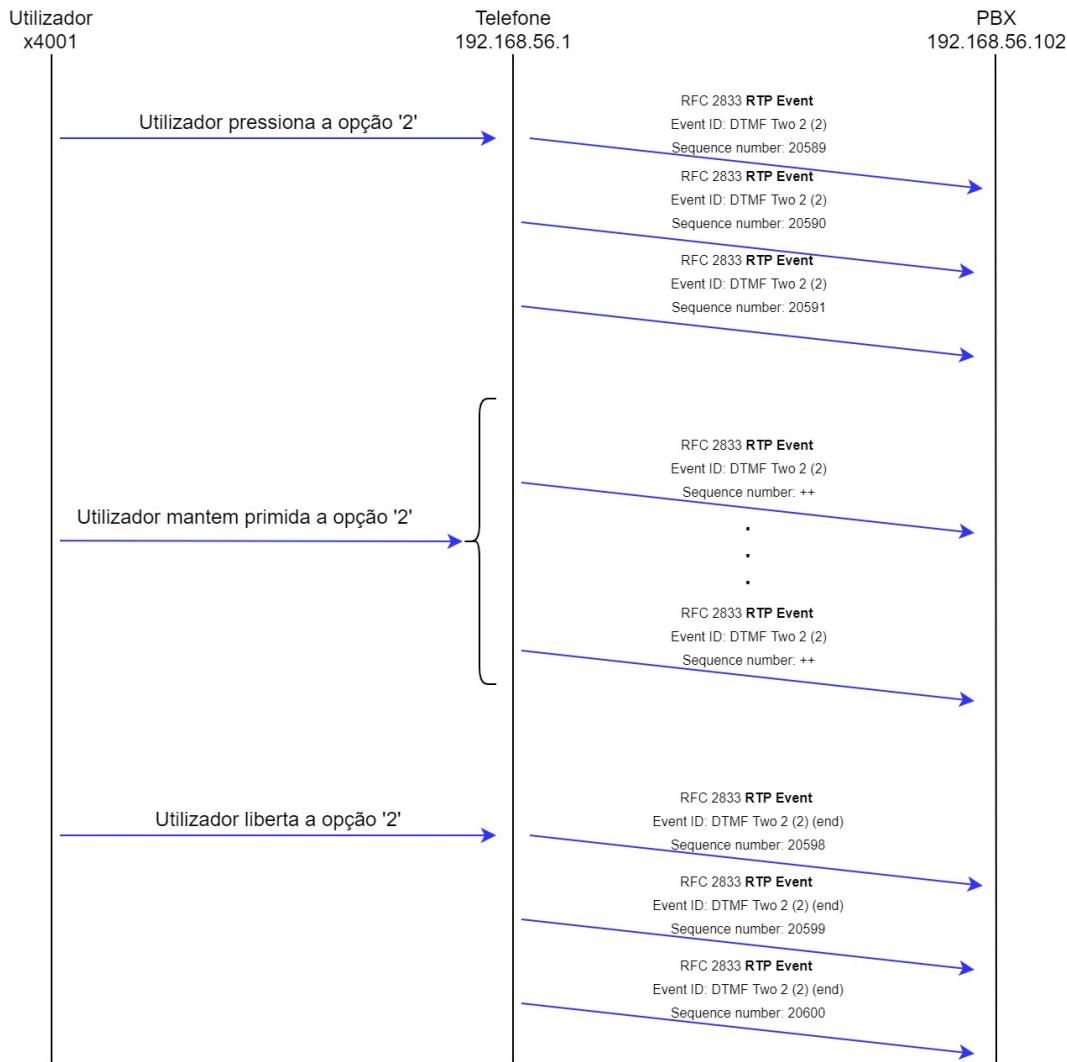


Figura 25: Diagrama temporal que exemplifica a troca de pacotes com a configuração “Send via RFC 2833”.

4.2.2 “Send via INFO”

Nesta segunda análise, é substituído o parâmetro anterior da aplicação X-Lite por *Send via INFO*.

Na figura 26 observamos os pacotes trocados entre a aplicação telefónica e o servidor PBX asterisk.

No.	Time	Source	Destination	Protocol	Length	Info
3177	102.505468	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x2BA95EB1, Seq=19123, Time=160064
3178	102.510496	192.168.56.1	192.168.56.102	SIP	720	Request: INFO sip:84187@192.168.56.102:5060
3179	102.511060	192.168.56.102	192.168.56.1	SIP	524	Status: 200 OK
3180	102.512563	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x2BA95EB1, Seq=19124, Time=160224
3181	102.518515	192.168.56.1	192.168.56.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x59D92CC2, Seq=12175, Time=1284321515
3182	102.532931	192.168.56.102	192.168.56.1	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x2BA95EB1, Seq=19125, Time=160384
3183	102.538751	192.168.56.1	192.168.56.102	RTP	214	PT=ITU-T G.711 PCMU, SSRC=0x59D92CC2, Seq=12176, Time=1284321675

```

> Frame 3178: 720 bytes on wire (5760 bits), 720 bytes captured (5760 bits) on interface 0
> Ethernet II, Src: 0a:00:27:00:00:13 (0a:00:27:00:00:13), Dst: PcsCompu_3b:e3:48 (08:00:27:3b:e3:48)
> Internet Protocol Version 4, Src: 192.168.56.1, Dst: 192.168.56.102
> User Datagram Protocol, Src Port: 52130, Dst Port: 5060
└ Session Initiation Protocol (INFO)
  > Request-Line: INFO sip:84187@192.168.56.102:5060 SIP/2.0
  > Message Header
  < Message Body
    Signal=2\r\n
    Duration=250
  
```

Figura 26: Captura de Wireshark com a configuração “Send via INFO”.

A quantidade de pacotes trocados entre o utilizador e o servidor PBX diminui abruptamente. Porquê? Usando este parâmetro, garante-se a fiabilidade no que toca à troca de pacotes. É um pacote enviado pelo protocolo SIP que garante uma recepção de uma resposta do tipo *200 OK* caso o pacote com a informação da tecla clicada tenha sido corretamente entregue. Caso o pacote não seja entregue (caso não se receba a mensagem de confirmação de entrega), aí sim, será reenviado o pacote inicial. Neste pacote, a tecla premida pelo utilizador está descrita no corpo da mensagem, no atributo *Signal*.

Também se pode observar a diferença das situações verificando o diagrama temporal apresentado na figura 27.

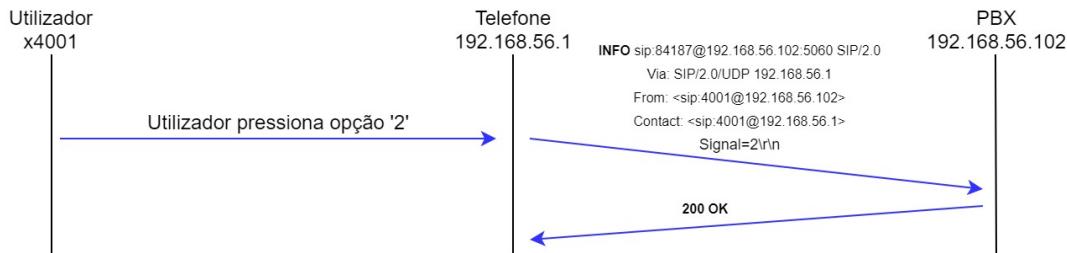


Figura 27: Diagrama temporal que exemplifica a troca de pacotes com a configuração “Send via INFO”.

4.3 Ligar ao operador

Nesta secção, é abordada a situação em que se liga para um utilizador que também se encontra registado no servidor PBX Asterisk. No caso em que a chamada é atendida, analisar-se-á o uso do atributo *directmedia* e a influência deste no que diz respeito à troca de pacotes numa chamada entre dois utilizadores.

4.3.1 Sem o uso da linha “directmedia=no”

Primeiramente, analisa-se o caso em que não se coloca o linha “*directmedia=no*” no ficheiro *sip.conf*. Apresenta-se de seguida, na figura 28, um diagrama temporal ilustrativo desta situação.

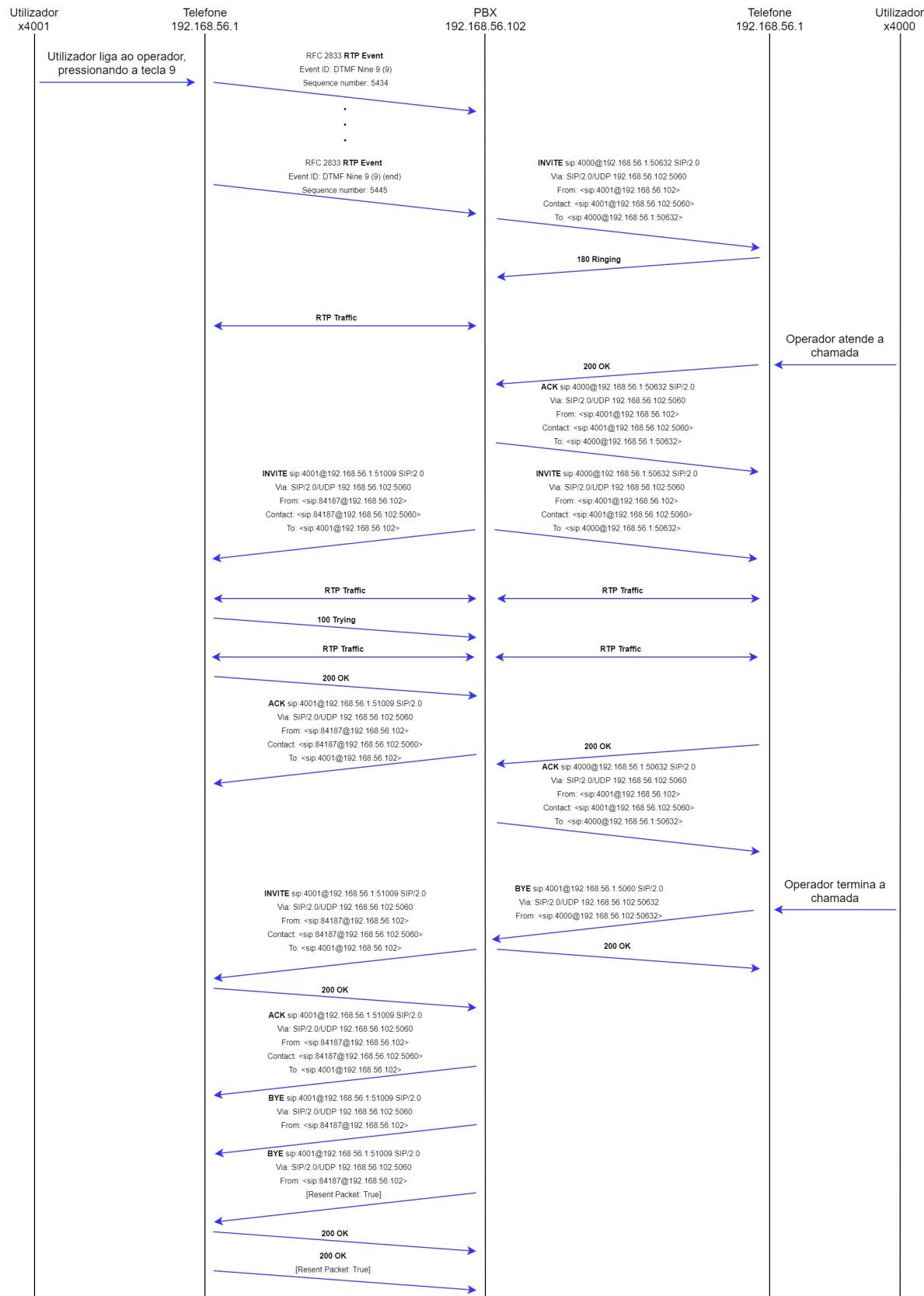


Figura 28: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador liga ao operador.

Inicialmente, o utilizador toma a iniciativa de começar a chamada, pressionando a te-

cla 9 do *softphone*. Este evento RTP é traduzido pelo PBX como a ligação ao operador, que possui o número 4000. Assim, é enviado um pacote pelo servidor para o utilizador que recebe a chamada. O PBX recebe o pacote *180 Ringing*, demonstrando que se está a tentar estabelecer uma ligação. Ao mesmo tempo, o servidor continua a trocar pacotes RTP com o utilizador inicial. De seguida, o operador atende a chamada, e quando o Asterisk aprende que os seus codecs são compatíveis aos do utilizador, conclui então que a informação pode simplesmente fluir entre os dois utilizadores e não passar pelo intermediário Asterisk. Então este manda um pacote *INVITE* para cada utilizador de modo a estes poderem comunicar para um endereço e porto específico, diretamente. Enquanto os utilizadores não aceitam o pedido, o Asterisk continua a servir de intermediário, mas após receber a resposta *200 OK* de cada um, o trânsito RTP é reencaminhado, até ao final da chamada, sem passar pelo Asterisk. Deixa-se então de visualizar os pacotes trocados entre os dois utilizadores, visto que o Wireshark estava a capturar pacotes que entram e saem da máquina virtual do computador, onde o Asterisk opera. Os pacotes que se captura a seguir, são apenas quando um dos utilizadores decide terminar a chamada. Esta definição está definida por padrão pelo asterisk como "directmedia = yes".

É apresentado de seguida, na figura 29 a captura Wireshark do sucedido.

259 7.359277	192.168.56.102	192.168.56.1	SIP/SDP	986 Request: INVITE sip:4000@192.168.56.1:50632;rinstance=c5d0ff4bfcaefaf83
270 7.464333	192.168.56.1	192.168.56.102	SIP	462 Status: 180 Ringing
970 14.301742	192.168.56.1	192.168.56.102	SIP/SDP	814 Status: 200 OK
971 14.302304	192.168.56.102	192.168.56.1	SIP	500 Request: ACK sip:4000@192.168.56.1:50632;rinstance=c5d0ff4bfcaefaf83
972 14.304151	192.168.56.102	192.168.56.1	SIP/SDP	889 Request: INVITE sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d, in-dialog
973 14.304237	192.168.56.102	192.168.56.1	SIP/SDP	960 Request: INVITE sip:4000@192.168.56.1:50632;rinstance=c5d0ff4bfcaefaf83, in-dialog
982 14.391806	192.168.56.1	192.168.56.102	SIP	326 Status: 100 Trying
1016 14.690961	192.168.56.1	192.168.56.102	SIP/SDP	783 Status: 200 OK
1017 14.691391	192.168.56.102	192.168.56.1	SIP	474 Request: ACK sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d
1018 14.713043	192.168.56.1	192.168.56.102	SIP/SDP	814 Status: 200 OK
1019 14.713567	192.168.56.102	192.168.56.1	SIP	500 Request: ACK sip:4000@192.168.56.1:50632;rinstance=c5d0ff4bfcaefaf83
1028 20.147612	192.168.56.1	192.168.56.102	SIP	568 Request: BYE sip:4001@192.168.56.102:5060
1029 20.147993	192.168.56.102	192.168.56.1	SIP	563 Status: 200 OK
1030 20.148324	192.168.56.102	192.168.56.1	SIP/SDP	891 Request: INVITE sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d, in-dialog
1031 20.169887	192.168.56.1	192.168.56.102	SIP/SDP	783 Status: 200 OK
1032 20.169424	192.168.56.102	192.168.56.1	SIP	474 Request: ACK sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d
1033 20.169463	192.168.56.102	192.168.56.1	SIP	672 Request: BYE sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d
1037 20.669923	192.168.56.102	192.168.56.1	SIP	672 Request: BYE sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d
1039 20.891403	192.168.56.1	192.168.56.102	SIP	433 Status: 200 OK
1040 20.896459	192.168.56.1	192.168.56.102	SIP	433 Status: 200 OK

Figura 29: Captura de Wireshark quando o utilizador liga ao operador.

Foram omitidos os pacotes RTP devido ao elevado número destes mesmos. É então possível verificar que o Servidor PBX Asterisk envia dois pedidos *INVITE* (números 972 e 973) para os dois utilizadores SIP e desde então, a troca de pacotes deixa de existir passando pelo servidor. É possível observar tal acontecimento analisando o número de sequência dos pacotes. Verifica-se no final que para terminar a chamada com o utilizador inicial foi necessário mandar dois pacotes *BYE*, visto que a resposta não foi imediata. Estes pacotes foram enviados com um intervalo de tempo de 0.5 segundos, que será o *timeout* imposto, em caso de não haver resposta. No entanto, são recebidos na mesma dois pacotes *200 OK*, indicando que o utilizador recebeu o primeiro pacote de conclusão da chamada, mas talvez este tenha demorado mais tempo do que o esperado a chegar ao destinatário. Ambos os segundos pacotes contêm a etiqueta *Resent Packet: TRUE*.

4.3.2 Com o uso da linha “directmedia=no”

No segundo caso, é acrescentada ao código do ficheiro *sip.conf* a linha de código *directmedia=no*. Esta mudança é acrescentada na linha 115 deste ficheiro, logo após começar a secção [general]. Verifica-se de seguida, na figura 30, o diagrama temporal correspondente a esta situação.

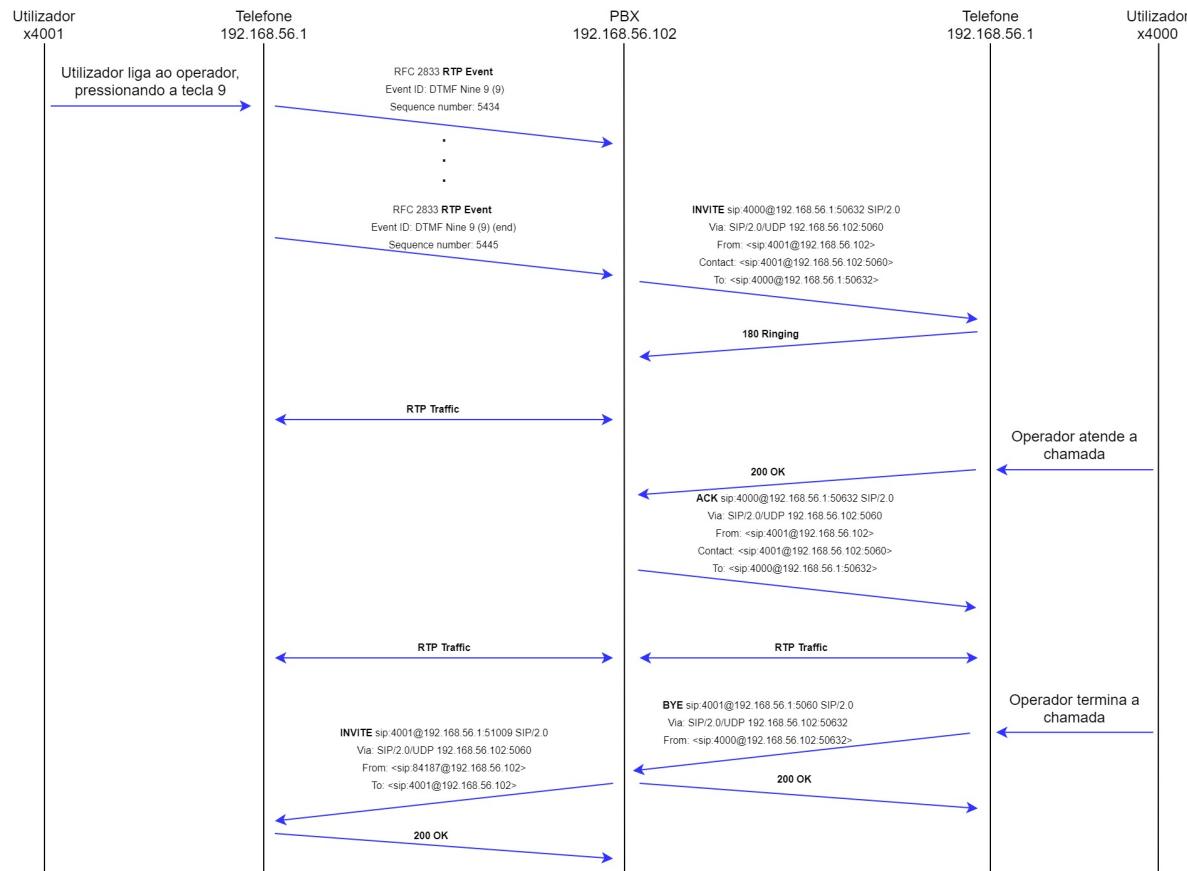


Figura 30: Diagrama temporal que exemplifica a troca de pacotes quando o utilizador liga ao operador, com a alteração *directmedia* = no.

Com esta funcionalidade o Asterisk nunca envia dois INVITES, um para cada utilizador, uma vez que esta linha de código desativou a visibilidade que o Asterisk tem aquando a análise dos codecs. Assim, o servidor PBX não consegue identificar quando estes são iguais e o tráfego continua todo a circular passando pelo Asterisk. Todos os pacotes RTP relativos à conversação dos dois utilizadores passam primeiro pelo servidor PBX asterisk.

347 16.713266	192.168.56.102	192.168.56.1	SIP/SDP	986 Request: INVITE sip:4000@192.168.56.1:50632;rinstance=c5d0ff4bfcaefafa83
358 16.818076	192.168.56.1	192.168.56.102	SIP	462 Status: 180 Ringing
764 20.845221	192.168.56.1	192.168.56.102	SIP/SDP	814 Status: 200 OK
765 20.845815	192.168.56.102	192.168.56.1	SIP	500 Request: ACK sip:4000@192.168.56.1:50632;rinstance=c5d0ff4bfcaefafa83
2184 27.783148	192.168.56.1	192.168.56.102	SIP	568 Request: BYE sip:4001@192.168.56.102:5060
2185 27.783879	192.168.56.102	192.168.56.1	SIP	563 Status: 200 OK
2106 27.785291	192.168.56.102	192.168.56.1	SIP	672 Request: BYE sip:4001@192.168.56.1:51009;rinstance=b4ba3e63a285b37d
2108 27.882436	192.168.56.1	192.168.56.102	SIP	433 Status: 200 OK

Figura 31: Captura de Wireshark quando o utilizador liga ao operador, com a alteração *directmedia* = no.

Pode-se observar pela figura 31, que mostra a captura de Wireshark feita, novamente apenas através dos pacotes com protocolo SIP, que o servidor desta vez não envia os dois pacotes INVITE. Desta vez, estes dois continuam a comunicar passando o tráfego RTP pelo Asterisk. Foram omitidos os pacotes do protocolo RTP mas pode-se observar pelo número de sequência dos pacotes que existe bastante comunicação entre os dois utilizadores até à existência de um pacote BYE.

5 Conclusão

Para este projeto foi numa primeira parte compreendido o funcionamento dos principais protocolos necessários para a sua realização. Compreendeu-se a sua importância principalmente através da análise de pacotes capturados pela aplicação *Wireshark* aquando da realização do projeto. Como seria de esperar, os pacotes do protocolo SIP são relacionados com o estabelecimento de sessões, estado de sessões e a comunicação entre vários utilizadores e os pacotes do protocolo RTP são relacionados com a troca de fluxos de áudio e informações entre utilizadores.

Este foi um projeto desafiante no qual foi possível explorar as funcionalidades e potencialidades que o servidor PBX Asterisk oferece aos seus utilizadores. Poderia ter sido explorado mais situações tais como desenvolver uma caixa de correio para chamadas não atendidas e o posterior envio de mensagens, sons de toque de chamada, chamadas de vídeo e tantas outras funcionalidades. No entanto, os objetivos propostos para este trabalho foram todos corretamente desenvolvidos incluindo o mais abstrato dos pontos, a busca por um código postal ZIP através do uso de um script Python. Foi um projeto que deu a conhecer novas tecnologias que ainda não tinham sido exploradas ao longo do curso.

Referências

- [1] SIP: Session Initiation Protocol. J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, E. Schooler. IETF RFC 3261, Junho de 2002.