Communication Networks 2 --- VoIP measurement

Measurement guide

Introduction

During this measurement we will get to know: IP based telephones, IP based software phone applications, telephone based services, computer virtualization and network based packet-analysis. The measurement guide reading is needful for good understanding of the applied technics and for the successful measuring. Control questions have to be also answered for clear understanding.

Audio Codecs

Codecs are used to convert an analogue voice signal to digitally encoded version. Codecs try to provide better sound quality with less bandwidth requirement and computational requirements.

Features of codecs:

- **Required bandwidth:** the bandwidth demand of the coded voice. In some cases of codecs this could be adaptive, for example AMR (Adaptive Multi-Rate audio codec)
- **Audio quality:** In voice communication quality usually dictates whether the experience is a good or bad one, numerical method of expressing voice and video quality, called Mean Opinion Score (MOS). MOS gives a numerical indication of the perceived quality of the received media. MOS is expressed in one number, from 1 to 5. 1 being the worst and 5 the best. MOS is quite subjective, as it is calculated based on results perceived by people and no by software. Then an arithmetic mean (average) is calculated based on the gathered individual scores, giving the Mean Opinion Score. The MOS number are graded:
  5 – Excellent Like face-to-face conversation or radio reception.
  4 – Good Imperfections can be perceived, but sound still clear.
  3 – Fair Slightly annoying.
  2 – Annoying Nearly impossible to communicate.
  1 – **Very annoying** Impossible to communicate.
- **Complexity:** How deep calculating is needed for the given codec. It is an important criteria in case of the embedded and limited energy systems.
- **Packet loss sensitivity:** This feature speaks about a possibility of the sample skipping still with a good understanding. A prediction existing could help to make the missing part up.
- **Delay:** Time amount needed by the codec algorithm.

We will use the G.711 (PCM - Pulse code modulation) codec during our measurements. The G7.11 is a high bite rate (64 kbit/s) ITU-T (ITU Telecommunication Standardization Sector) standard codec. It is a native language of the modern digital telephone network and is supported by most VoIP providers. This is used for example between the IP and ISDN audio communication, so, the recoding is not needed. The G.711 has got two types: μ-Law (U-Law) used in North America, and a-Law (A-Law) used in other part of the world (also in Europe).
Parameters of the G7.11:

- Bandwidth: 64 kbit/s
- Delay: 0.125 msec
- MOS value: 4.1

- Time consuming: 0.52 MIPS (500 000 instructions/sec), is a general measure of computing performance.

The G.711 offers good audio-quality with small delay and low complexity. But bandwidth demand could be more optimal in case of other codecs. For example the GSM 06.10 codec used in old 2G mobile-phones needs only 7 kbit/s or 13 kbit/s (using mode depending), but its MOS value is only 3.5-3.9. The G.729 codec needs only 8 kbit/s bandwidth and its MOS value is3.9, but its time consuming is high and protected by patent.

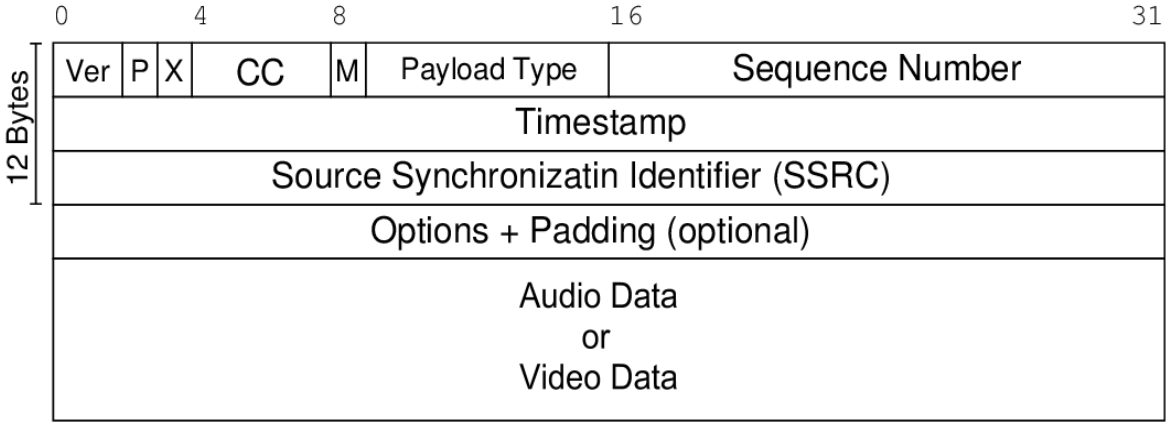
RTP

The RTP (Real-time Transport Protocol) offers transport service for the real-time traffic, for delivering audio and video over IP networks. RTP typically runs over UDP (User Datagram Protocol) with unicast or multicast addressing.

Protocol has two components:

- RTP: transports real-time data
- RTCP (RTP Control Protocol): monitors the transmission quality and carries information about the participants.

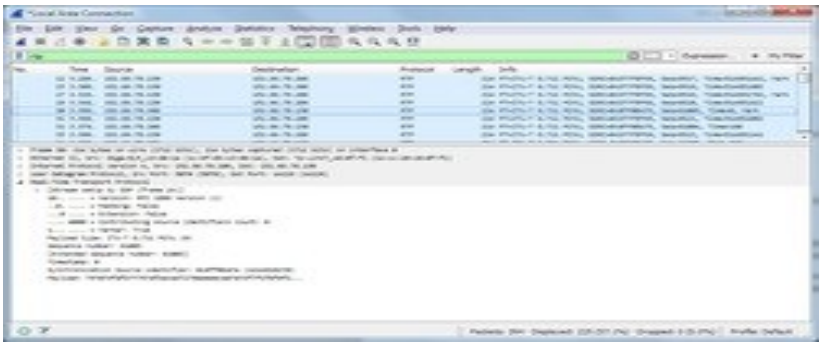
The RTP header with data:



We do not need to analyse all fields of the header, because some are only technologic and are needed only due to data representation. The header also includes information about the sender and receiver session – which are also not important for us at this time. The NEXT fields and their contents are important for us:

- **Payload Type:** (7 bits) Indicates the format of the payload and determines its interpretation by the application determining which codec is used. In case of the G7.11 two values exists: 0 -> μ-Law, 8 -> a-Law.
- **Sequence Number:** (16 bits) sequence number is incremented by one for each RTP data packet sent and is to be used by the receiver to detect packet loss and to restore packet sequence. The initial value of the sequence number is random.
- **Timestamp:** (32 bits) Used to enable the receiver to play back the received samples at appropriate intervals. The initial value of the timestamp is random.

Important: The RTP is not trying to repair packet loss!



1. Figure RTP packets analysing, Payload type:0

The RTCP describes the quality of RTP streams by the receiver (RR) and sender (SR) reports. But the RTCP can carry connection release information and application representative data as well. We can also get information about the packet loss and jitter (which means delay fluctuation) by the RTCP as well.
During the measurement, when the bandwidth requirement needs to be determined, do not forget the "protocol-layering" which means all protocols have got their headers as well (together 40 bytes):

- IP: 20 bytes
- UDP: 8 bytes
- RTP: 12 bytes

We can notice that the bandwidth demand also depends on the duration of the sample in the packet. Longer samples mean shorter overhead, which means smaller bandwidth demand. However, in case of packet samples increasing, stored in a package, the size of a delay also increases, because these packages could be sent only later.

SIP

At the end of last century, cheap, "long distance" packet-switched networks (like the global Internet) started to take the role of the circuit-switched based audio-transmission. Recommended by the ITU-T, the H.323 protocol group was used for the audio transmission through the packet-switched networks. This strongly was built on the logic of thecircuit-switched networks and many terminologies were taken from them as well. As a binary protocol had a very efficient bandwidth requirement, however, the implementation of this was a very complex task.

In the same time the IETF (Internet Engineering Task Force) organization also started to work on a VoIP protocol, which is simple and transparent as well. The IETF used mechanisms which have been accepted on the Internet. This protocol was named SIP (Session Initiation Protocol).

The SIP is a communications protocol for signalling and controlling multimedia communication sessions between the end points. Unlike the H.323, SIP is readable and clear even without any packet-analyser. The protocol structure is very similar to the HTTP protocol many times even the error messages are same. The SIP URI is doing clients identifications, typically in name@address form – in case of IP telephone exchange - address is the IP address or the domain name of this exchange. This name format is derived from email systems.

The most important participants of the SIP based systems are:

- User agent (UA), which can be:
  - a normal line IP phone or a wireless IP phone
  - sub-exchange
  - VoIP client program on the PC (softphone)
- Server, which can be:
  - Proxy server: is an intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients.
  - Redirect server: doing determination of forwarding direction
  - Location server: storing of customers current whereabouts
  - Registar: doing user agent registration

The SIP is a client-server based communication but the caller and the called, are working in both roles. At the server side the port number of the SIP is 5060 UDP or TCP port, and the client side is connecting to this port.

Important SIP messages:

- INVITE - is used to initiate a session with a user agent.
- ACK - is used to acknowledge the final responses to an INVITE method.
- OPTIONS - method is used to query a user agent or a proxy server about its capabilities and discover its current availability.
- REGISTER - performs the registration of a user agent. Usually customer identifier and password is needed and periodical repeating is necessary.
- BYE - is the method used to terminate an established a session.
- CANCEL - is used to terminate a session which is not established. For example if a calling is ringing on a group of phones and somebody is picking up. If other phones don't get a CANCEL message for them it is a missed call.
- SUBSCRIBE - is used by user agents to establish a subscription for the purpose of getting notification about a particular event.
- NOTIFY - is used by user agents to get the occurrence of a particular event.

For every SIP request message as an answer comes a respond message. Respond codes can be next:

- 1xx – provisional responses, which need to be followed by a final respond. 100 – Trying, 180 - Ringing
- 2xx – successful responses. 200 – OK
- 3xx – redirection responses. Other actions are needed for the calling. 302 – Moved temporary
- 4xx – client failure responses. The request cannot be done, error. 401 - Unauthorized
- 5xx and 6xx – server or global failure responses. Error

INVITE message example:

```
INVITE sip:kitti@172.16.0.75:5060 SIP/2.0
Via: SIP/2.0/UDP 172.16.0.135:5060;branch=z9hG4bK0017530a;rport
Max-Forwards: 70
```

From: <sip:101@172.16.0.135>;tag=as010dc147
To: <sip:kitti@172.16.0.135>;tag=f7eb8d16-dd11-1910-817c-4437e6aa81bd
Contact: <sip:101@172.16.0.135:5060>
Call-ID: f7eb8d16-dd11-1910-817d-4437e6aa81bd@majdan-THINK
CSeq: 102 INVITE
User-Agent: Asterisk PBX 1.8.32.3
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE, NOTIFY, INFO, PUBLISH, MESSAGE
Supported: replaces, timer
Content-Type: application/sdp
Content-Length: 264

v=0
o=root 1007958305 1007958306 IN IP4 172.16.0.153
s=Asterisk PBX 1.8.32.3
c=IN IP4 172.16.0.153
t=0 0
m=audio 16000 RTP/AVP 0 101
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=silenceSupp:off - - - -
a=ptime:20
a=sendrecv

We can see from above messages that the called customer is kitti with the IP address 172.16.0.75. The Via field shows which Proxy server will forward messages. The From fields shows that the caller is "101" at the telephone exchange with IP address 172.16.0.135. One character fields at the end of the package are part of the SDP (Session Description Protocol) block. We can read from this information: the codec type is PCMU/8000 for RTP packets – which means the G.711 PCM μ-law codec. Direct connection exists between customers. This can be concluded from the packets route, because the IP address 172.16.0.153 in the SDP block is not equivalent with the Proxy IP address. So, the Proxy server is not a part of the connection. The field m describes that the port number=1600 must be used at the 172.16.0.153 IP address.

SIP/2.0 200 OK
CSeq: 103 INVITE
Via: SIP/2.0/UDP 172.16.0.135:5060;branch=z9hG4bK5d0f2266;rport=5060;received=172.16.0.135
User-Agent: Ekiga/4.0.2
From: <sip:101@172.16.0.135>;tag=as010dc147
Call-ID: f7eb8d16-dd11-1910-817d-4437e6aa81bd@majdan-THINK
To: <sip:kitti@172.16.0.135>;tag=f7eb8d16-dd11-1910-817c-4437e6aa81bd
Contact: <sip:kitti@172.16.0.75:5060>
Allow: INVITE,ACK,OPTIONS,BYE,CANCEL,SUBSCRIBE,NOTIFY,REFER,MESSAGE,INFO,PING,PRACK
Content-Length: 218
Content-Type: application/sdp

v=0
o=- 1456043216 3 IN IP4 172.16.0.75

s=Ekiga/4.0.2
c=IN IP4 172.16.0.75
t=0 0
m=audio 5064 RTP/AVP 0 101
a=sendrecv
a=rtpmap:0 PCMU/8000/1
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=maxptime:240

Kitti picked up the call, audio packets need to be sent through the RTP protocol to the IP address 172.16.0.75 and port=5064.

Asterisk

Asterisk is a free, open source framework for building communications applications. Asterisk turns an ordinary computer into an IP PBX phone system, VoIP gateway, conference server. We are using it as an IP PBX phone system therefore we refer to it in this way. But various other customizable availabilities exist as well. Mark Spencer was the creator of the Asterisk software, released in 1999. Spreading of Asterisk was evident, thanks to its gateway possibility between analogue and digital technologies. They did not use expensive DSP (digital signal processor) cards but - thanks to the Zapata Telephony Project – they used increasingly stronger processors and only minimal current circuit was used for analogue (PSTN) connection.

2. Figure Digium IP-telephone 3. Figure Digium 4-Port analogue card

Asterisk is a Linux based system. From the beginning it has been built to its structure and services. So, Asterisk can be run under Windows based system only by a virtual machine (However, in the past also existed experiments for transforming this system under the Windows.)

Large community support can be founded on forums and dedicated VoIP sites (for example http://www.voip-info.org). The business support is also sponsored by Digium, which is the developer company of Asterisk now. Such telephone sub-exchanges, IP telephones, ISDN and analogue cards can be bought which compatibilities are guaranteed with Asterisk.

Next description is talking about the Asterisk version 1.8, which is used during our measurements.

Asterisk is a modular system. Modules can be downloaded one by one and all of them are responsible for other functions. For example, the SIP channel driver is one module (chan_sip.so), without this

could not be possible any SIP based VoIP communication. The file `modules.conf` has to include a list of modules which have to be loaded. Default setting means, all modules will be loaded automatically (`autoload=yes`), which are not forbidden (`noload something.so`).

Types of modules:

- **Applications**: For example Dial() application which connect particular channels (call set-up), or Playback() for replay of already recorded audio streams.
- **Bridging modules**: Modules needed in case of conference calls.
- **Call detail recording (CDR) modules**: Modules needed for storage of calling information. (into csv file or database).
- **Channel event logging (CEL) modules**: Modules needed to archieve channel events.
- **Channel drivers**: Each driver supports one protocol, respectively one type of channel. For example SIP, H.323, GTalk, Skinny, ISDN, DAHDI (analogue).
- **Codec translators**: Translators are capable converting between various codecs. For example, if the ISDN uses G.711 codec and this channel need to be connected with a SIP channel using G.729 channel, converting is needed between them. This is using processor acutely, so we will use the same type of codec everywhere.
- **Format interpreters**: This means various file format supporting in case of audio (or video) contents. For example, this module is needed in case of audio record or holding music replaying. We are using the simplest wav format in the laboratory. The holding music (Für Elise) is stored in this format.
- **Dialplan functions**: Modules which expand applications with functions connected with Dialplan. For example, the CallerID (caller customer number or in some cases even his name) can be asked or set, or even a linux shell command can be run in case of a number dialling.
- **PBX modules**: Modules of the telephone exchange for scripting possibility (LUA - scripting language).
- **Resource modules**: Modules needed for access of peripheral resources. For example, access to LADP (Lightweight Directory Access Protocol) database, which includes data of customers.
- **Addons modules**: Modules written by members of a community, which are not part of the basic Asterisk. These modules are separated for using or license reason. Module for MP3 files replaying is such an example.
- **Test modules**: Developers use these types of modules for testing.

We don't have more space for further description of modules. More information can be found about the Asterisk in The Definitive Guide, 3rd Edition book, pages 10 – 24.

Because our measurement tasks are built on the SIP signalling protocol, we describe basic set-up of the SIP channel driver, which need to be written into the `sip.conf` file.

The simplest way is to show it by an example:

```
[100]
type=friend
callerid="John Smith" <100>
host=dynamic
secret=strongpassword
```

```
context=pbx
canreinvite=no
```

Meaning of these fields:

- **user identifier**: this is between the square brackets. This identifier is not recommended to be set to the same value like the call number, (because this number identifies the telephone device. And later the call number changing could make some trouble.) But here these numbers are setting to same values for simply using. In case of VoIP client this value has to be set for the userID. In case of a real system, the telephone MAC address should be set as an identifier.
- **type**: How Asterisk has to understand incoming SIP requests, which come from this user
    - **peer**: Incoming requests are identified by the IP address and port number
    - **user**: user is identified based on the From: header of SIP message, and this has to be corresponded to the user identifier
    - **friend**: identification could be either by peer mode or user mode
- **callerid**: determination of the caller in case of calling, what is texted on the called side phone
- **host**: by which IP address is accessible this station. Dynamic option means, that this IP address is determined when this station will registered at the Asterisk
- **secret**: password which has to be set for the VoIP client
- **context**: in case of calling (initialized from this side), which type of dial-plan context has to be used
- **canreinvite**: information about creation of direct RTP voice connection with the other side (=no means, it is not allowed)

The **canreinvite** field needs more explanation. The calling customer (in case of SIP communication) is sending an INVITE message to the Proxy server, which is forwarding (even by more proxy servers) to the called customer. The SDP part of this message includes information about where and by which encoding type needs to be transferred voice packets. So, this includes the IP address and RTP port of the called customer and the supported codec types as well. We can see this method violate the OSI level rules, because it includes an addressing of Layer 3 (network level). The called customer is sending back a "200 OK" message, selecting the codec type and sending back its own IP address and RTP port - information where voice packets have to be sent. In particular cases this IP address is not its address of its SIP signalling method, but in case of a bigger telephone exchange, this is the IP address and RTP port of another DSP card of this device. In this case voice connection can be created directly between the customers, and SIP signalling messages will be transported through the SIP proxy server. Asterisk functioning is a little bit difference from this method. The INVITE message is not forwarded directly but it itself is making a new session with the called side (also by INVITE message) therefore we can recognize different identifiers in the packets. In this way Asterisk is able to allow or deny direct RTP connection between two customers. If the Asterisk denies direct connection, it puts its own IP address and RTP port into the SDP part during the SIP communication between customers, and connects voice channel for data packets.

- **recording of calling (voices recording)**
- **central hold music**
- **conference**
- **codec converting**
- **in-band DTMF tones detection**
- **speech recognizing** (bomb attack …)

SIP channel has much more setting options. For example, Asterisk is available for its own registration into a SIP proxy server and receives incoming calls from it, and forwards calls toward to it as well.

If we want to create an active IP telephone exchange, we need to get to know an `extensions.conf` file structure. Here is stored a Dialplan which includes calls ending logic.

extensions.conf example:

```
[pbx]
exten => 110,1,Dial(SIP/110)
exten => 200,1,Dial(SIP/101&SIP/102)
exten => 300,1,Dial(SIP/300,10)
exten => 300,2,Dial(SIP/301,10)
exten => 555,1,Answer()
exten => 555,2,MusicOnHold()
```

Rules under [pbx] are used only for dialling through the channels which are used only by pbx context. If somebody is calling the number 110, the channel with the identificator 110 is ringing. Number 1 means the priority level of this rule. If somebody is calling the number 200, channels with the identificators 101 and 102 are ringing in parallel both of them can be picked up. If somebody is calling the number 300 the channel with the identificator 300 is ringing for 10 seconds. And if during this period is not picked up the phone, the next rule, with the priority level 2, is done and the channel with the identificator 301 is ringing for 10 seconds. So, the Dial modul provides connection support between channels.

If the modul named MusicOnHold is active, in case of the number 555 dialling music for holding will be given. We can recognize, modules Dial and MusicOnHold are only two from many modules which can be used. There is also possible writing a new modul and fitting it into the system.

Format of the Dial service:

Dial(Technology/Resource&[Technology2/Resource2[&...]],[timeout,[options,[URL]]])

Rules can be also shorted but we will not use shorted rules.

An IP telephone exchange can work with this setup, but settings need to be done for every calling number - too much work for the operator. Therefore are used patterns for the calling number definition:

```
[pbx]
exten => _1XX,1,Dial(SIP/${EXTEN})
exten => _2XX,1,Dial(SIP/200)
exten => _906!,1,Dial(SIP/upc/+36${EXTEN:3})
```

The first rule says: Every call number which begins with number 1 and is 3 numbers long dials the channel with the customer identificator. So, if I am calling the number 105, the channel with the identificator 105 will be dialled. The second rule says: Every phone with the first number 2 and is 3 numbers long will be ringing when the customer identificator is 200. The last rule refers to all numbers which start with number 906 and can be whatever long. In this case number 9 is the “out-line” prefix and 06 is the dialling code in Hungary. Calling will be sent to the channel with the identificator named upc in the ITU standard form which means, first 3 number will be cut and substituted with +36 ITU front-number for Hungary. The upc will provide “call termination”.

Patterns are signed with an underscore. Next special characters can be used:

- **X** : 0-9 number
- **Z** : 1-9 number
- **N** : 2-9 number
- **[04-6]** : either a number or a range which is in the square bracket. In this case: 0,4,5,6.
- **.** : at least one character
- **!** : any number of character

During this measurement the next developed telephone services will be tested:

- **do not disturb** (DND): Telephone will not receive calls. The mode of rejection depends on the telephone exchange setup (also a message can be announced)
- **call forward** (CFWD): incoming calls are forwarding automatically to another number at the same or a separate location. Version of call forwarding can be conditional or unconditional. Conditions can be:
    - line busy
    - device is unavailable (for example the device is not connected to the network)
    - device is not responding (nobody is picking up within a certain period)
    - in case of a customizable IP based telephone exchange its conditions can be definable.
- **redial**: allows a user to redial the last number
- **hold**: allows you to keep your callers on hold with entertaining music or custom recordings. During this time is possible (for example, in case of customer service) to discuss about a specific issue with fellow worker in backround, or make/recieve a call from that device. A flashing button indicates the holding status and by pressing it the caller can be taken back. Many times other have also right to take over this call by a special phone number or button.
- **transfer**: Incoming call can be redirected to another customer phone-number. Usually the called side is doing this. For example a “customer service support” switches over the customer to the appropriate number when he decided who can help to solve the issue. Usually during this redirection holding music is used to fill the silence. Two type of transfer can be distinguished:
    - **supervised transfer** (announced, attended): Announced call transfer. Customer is announced where the call will be redirected before this action. Usually when the phone is hanged up customers are automatically connected and after the holding music the caller does not hear ringing. Many times an appropriate button need to be pressed for this new connection.
    - **blind transfer** (unannounced, unsupervised):. In case of call-redirection to another number this action is without any interaction between the person transferring the call and the new number receiving the transfer. Usually holding music is played at the side of the caller during the actual receiver dialling and after this only ringing is audiable unless the pick up at the receiving side.
- **conference**: Conference call. More customers can communicate together in the same time. Connection is usually made by the called part by pressing a special button or dialling a special number. Features like: number of participants, participant removing, new participant adding, preliminary talking with a new participant, all dependent on the type of used technology and settings.
- **call pickup** (directed pickup): Possibility to hang up a phone where the call was not directed. This could happen by pressing a special button or by dialling a special number. Two types

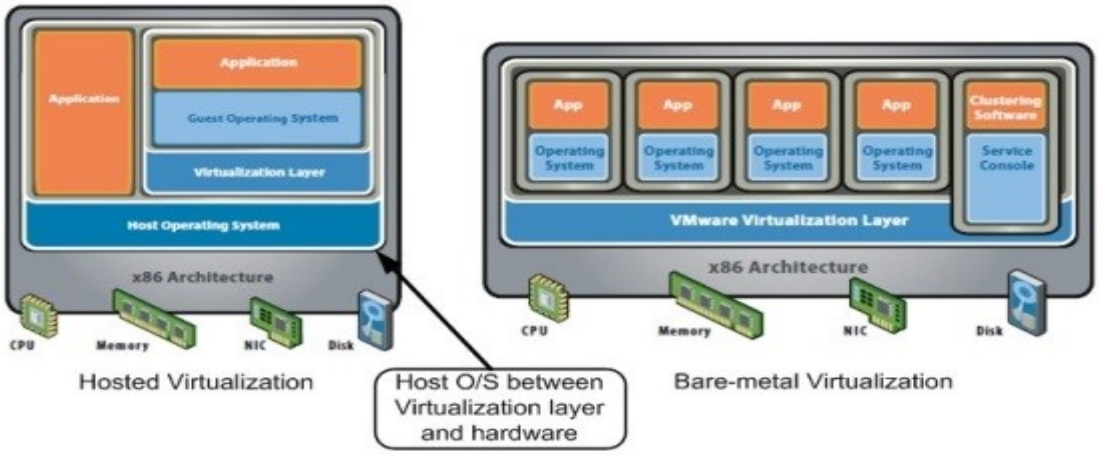
exist of this service: Pickup of an appropriate number call, or pickup a call which was directed to a pickup group including our phone number as well. This also dependences on the type of technology and settings. Even only one phone number can be a member of a group.

More information about the Astersik: http://www.asterisk.org/

Computer's hardware virtualization

Early computers have got only narrow resources. Many programs'optimalizations had to be done at the low level because of the sufficient run rate and memory capacity limitation. These devices were very expensive and programs were run one after other in batch processing, time-slices have got their prices. At first, various virtualization techniques of resources appeared on the mainframe computers, which later extended to the complete computer virtualization due to the technology development. Mainly security and backward compatibilities were reasons of this solution. Later new computers appeared with a so high computational capacity, that they were not utilized fully. Server capacity sharing became necessary because of the economic operation on these computers. Since this sharing technique only "resources" were sold, which nevertheless could be recognized as separated physical computers. Distribution of these resources in case of sharing (for example how many processors can be used by a virtual machine -> computing capacity) is based on a negotiation.
Computers' computing- and memory capacities had got so high level after the millennium, that hardware virtualization became possible to use even on PC machines. But the x86 architectures of these PCs made more difficult the creation of hypervisor programs for virtualization. (This specialized software, called a hypervisor, emulates the PC client or server's CPU, memory, hard disk network and other hardware resources completely enabling virtual machines to share resources.) But main processor companies (AMD, INTEL) noticed a rising demand of virtualization and they expanded set of instructions with some commands (in 2006), which helped in hardware virtualization (AMD-V, VT-x). Since this point the hardware virtualization on PCs is possible for everyone.
The computer where the hypervisor is running is called the **host** and virtualized machines are called the **guests**. Today **nested virtualization** also exist, which means that a Hyper-V host itself can be virtualized, and the host becomes also a guest or more guests in the same time. Two type of the virtualiztion exists based on the "virtualization level":

- **bare-metal (Type-1, native)**: the hypervisor is directly running on the hardware, so, it is a mini operating system.
- **hosted (Type-2):** the hypervisor is running on the operating system, like other programs and applications. We are using this method during our measurements. We are starting the virtual machine on the Windows operating system.



4. Figure Bare-metal and hosted virtualization

Why are we using virtualization on the simple PC?

Many times we need to test or use programs at the university (or in companies as well) which in the used environment (operating system - OS) would not run. For example, Windows 7 is running on the PC but the Asterisk IP telephone exchange is designed for linux based operating system. We do not need to buy a new PC and install a linux based OS on it, we can use virtual machine. We have to install an appropriate operating system (linux) on the virtual machine and after that we can install an Asterisk, what we want to use.

Feature named **snapshot** is also a good possibility in case of virtualization. Snapshot of status can be made on some particular points of a running or turned off virtual machine. And we can return to this status of machine in any time. For example before a "suspicious program" installation or running a command with a catastrophic effect we should make a snapshot.

Many students are using our virtual machines but we would not want always copy the whole virtual machine (make a **clone**). There is a possibility to create a **linked clone**, **which** is only a copy of changes compared to the starting image what was made originally. This is very useful in case of virtual machine with a large amount, because starting image storing is enough only once. And only changes made by the students will be stored at the students' PCs. The starting image will be always unchanged.

We are using a free Player software of the VMware (industry-leading virtualization software company), but also exist free hypervisors (VirtualBox, Xen, KVM).

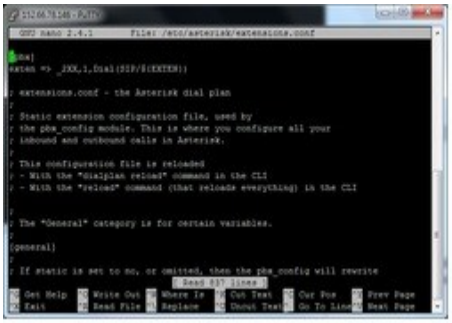
OpenWrt

OpenWrt is a free and open-source embedded operating system based on Linux kernel, used on various embedded devices (commonly routers). Previously it was used for expanding of ability of existing routers (by changing of firmware). Today devices exist directly based on OpenWrt or its variant. Initial router capabilities were expanded very quickly. Today we can get a general platform where various packages, network applications can be run. Because it is for embedded devices only few resources are needed for running, because these routers usually includes 4-16 MB flash memory and 16-64 MB RAM memory. This feature makes it ideal for a virtual machine application because even dozen are execable on one host. In case of different architectures different OpenWrt-s are used but only small difference exists in their using, but their steps of installation and used file system could be

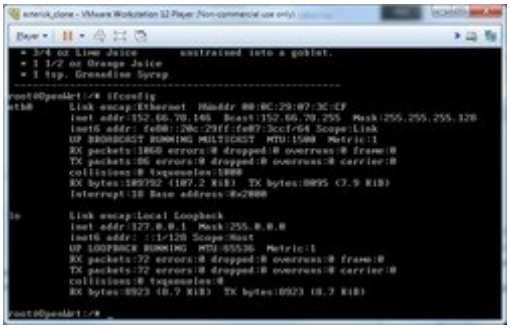
different. Various programs – like in case of other Linux distribution – can be installed by a package manager, the Asterisk IP telephone exchange is an included package as well. There is not necessary to care about downloading, un-packing and setup of automatic starting. Package manager is doing it itself.
The OpenWrt has web desktop (LuCI), but because we do not use OpenWrt like a router, we will not use this web desktop environment. Shell command is reachable through an encrypted channel. It is a secure shell (SSH), a cryptographic network protocol for secure operating network services over an unsecure network. But locally it is also reachable if the device includes an appropriate port. In case of embedded systems it is usually a serial port, keyboard + monitor connection is usually also possible. Like in case of different linux distribution, various shell can be used, default is the ash shell.

We are using next commands:

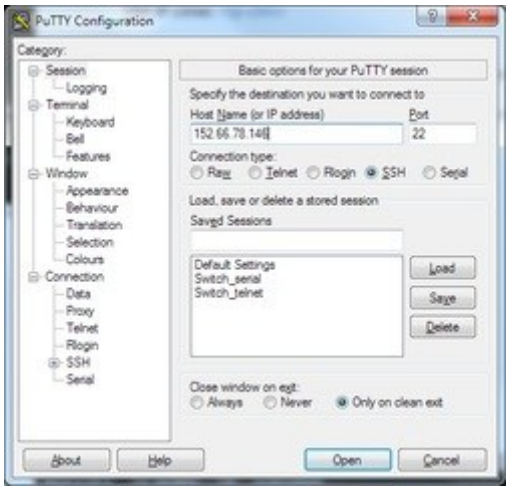
- **cd** – change directory (enter to directory)
- **cd ..** – change directory up (exit from a directory and go one level upper)
- **ls** – listing of files and directories
- **nano** – start of the nano editor for file editing
- **ifconfig** – listing of the network interfaces
- **/etc/init.d/asterisk restart** – restart of the Asterisk telephone exchange



5. Figure Nano text-editor



6. Figure IP address



7. Figure SSHconnection by the PuTTY

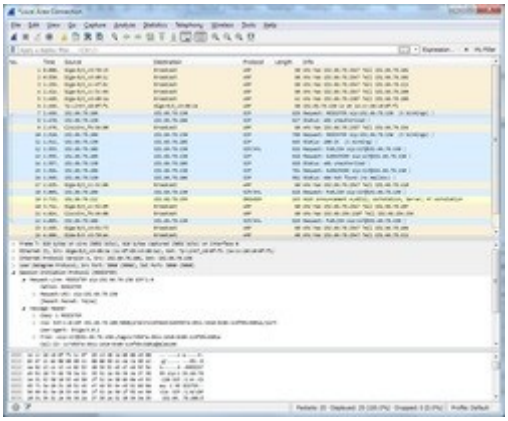
Key-combinations used in the nano editor:

- **ctrl+o** – file saving
- **ctrl+x** – exit from the editor

The Asterisk IP telephone exchange is running on a TP-LINK TL-WR1043ND router, where we have changed firmware to an OpenWrt. This router is very common home device, in 2016 its brutto price is around 14000-15000 Ft, including 8 MB flash memory, 64 MB RAM memory and 720 MHz processor (with MIPS architekture). A compressed OpenWrt file system is applied on this device due to narrow flash memory size. Still this device can support 24 IP phones (desk phone+ softphone) in the laboratory.
Students' desk-PC includes OpenWrt virtualization with 50 MB flash memory, 256 MB RAM memory and much faster 3.3 GHz processor 3.3 GHz.
More information about the OpenWrt: https://openwrt.org/

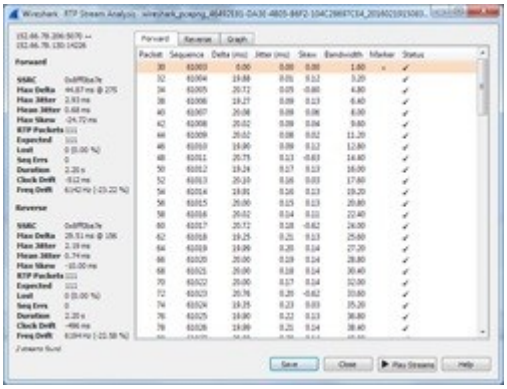
Wireshark

Wireshark (older name Ethereal) is a free, open-source network package-analyzator. It is widely used in the industry for the solution, analyzing of network issues, and it is very useful also in case of network protocols development and testing. It is available for every popular desktop and server operating system. Wireshark uses packet capture (pcap) for traffic monitoring, but already gathered traffic data can be also analysed. This offline method is helpful when the Wireshark is unexecable on the system, but traffic still can be stored by some program (for example by tcpdump). The OpenWrt and almost all IP telephone exchanges and IP media gateways (for example SIPGSM) have similar function.



Wireshark can not only show content of packets, but knowing the protocol, packets can be disassemblied to its fields. And based on rules of this protocol a clean understandable content visualization of packet structure is possible. In case of some protocol "packet-stream testing" is also possible. Wireshark can determine related packets and make some analyzis on all of them. In case of RTP protocol even the voice replay is possible.

8. Figure SIP registration message – analysis



9. Figure RTP stream analysis

In case of packet capture, select the Capture pull down menu and select Interfaces. Network interfaces (i.e., the physical connections) of the computer, connected to the network, are shown. Select the proper one, through which packets will be transmitted and received and on which interface you want to do traffic investigation. You need only traffic that's being sent to or from this networkcard.

10. Figure RTP replay

In default case packets have not addressed to this networkcard's physical address (MAC address) or packets which are not broadcast packets will be discarded. If you want to capture all network traffic packets recognized by this networkcard, you need to use **promiscuous mode**. Every device cannot work in this mode. For example some wireless devices' programs block this mode.

(see http://www.aircrack-ng.org/doku.php?id=compatibility_drivers).

Why could we receive packets which were not addressed to us? This can happen in more cases:

- **Shared physical media**: we are communicating on the same cable or wireless medium. Long time ago simple hubs function as a central connection point in the computer network, and/or a signal amplifier. When a hub receives packets or data at one of its ports, transmits them to every port without any selection. Today this type of hub is still using in case of network analysis, when this hub is connected between two tested points (computers), and the third computer with wireshark program is also connected into this hub.
- **Port mirroring**: Port mirroring is used on a recent network switch to send a copy of network packets seen on a certain switch port(s) to a network monitoring connection on another switch port. In this case computer is connected with Wireshark program to this "another port".

Due to the network speed increasing today, less attention is paid to minimalization of network traffics. So, unnecessary broadcast and traffics background applications' messages are overflowing networks. In case of network packets analysis we need to use a filter for only important packets showing. Wireshark has two possibilities to do this:

- **Capture filter (filtering at the level of packets saving)**: filtered packets are not received by the Wireshark. The library pcap provides this function.
- **Display filter (filtering at the level of displaying)**: Wirshark receives packets but based on filter's rules are not displayed. We can change this filter anytime during our measurement depending on our decision to display packets of a certain protocol. Specific rules can be used as well, but the Wireshark is able to create filter rules based on the packets' type automatically.

These two methods use different syntax. We will exploit packet-stream analysing possibility of the Wireshark (when it can automatically specify a display filter) and we will set up a certain filter at the level of displaying. We will use only two types of display filters: when we need to write the protocol name "rtp"or "sip" depeding on type of our investigation (data investigation or signal).

More information about filter rules: https://wiki.wireshark.org/CaptureFilters or https://wiki.wireshark.org/DisplayFilters

More information about the Wireshark: https://www.wireshark.org

Practical information

Information about IP telephones

During the measurement we will use 2 types of IP telephones: *Cisco SPA303-G2* and *Linksys SPA921*.

11. Figure Cisco SPA303-G2

Line buttons/Customer ID, (now also phone-number)

Function buttons

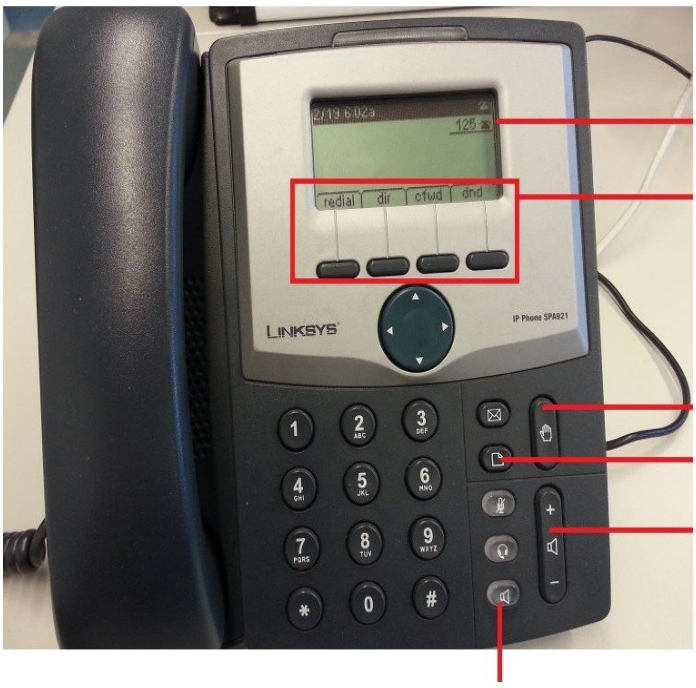
Call holding

Menu access

Volume-control

Speakerphone

12. Figure Linksys SPA921



Line buttons/Customer ID, (now also phone-number)

Function buttons

Call holding

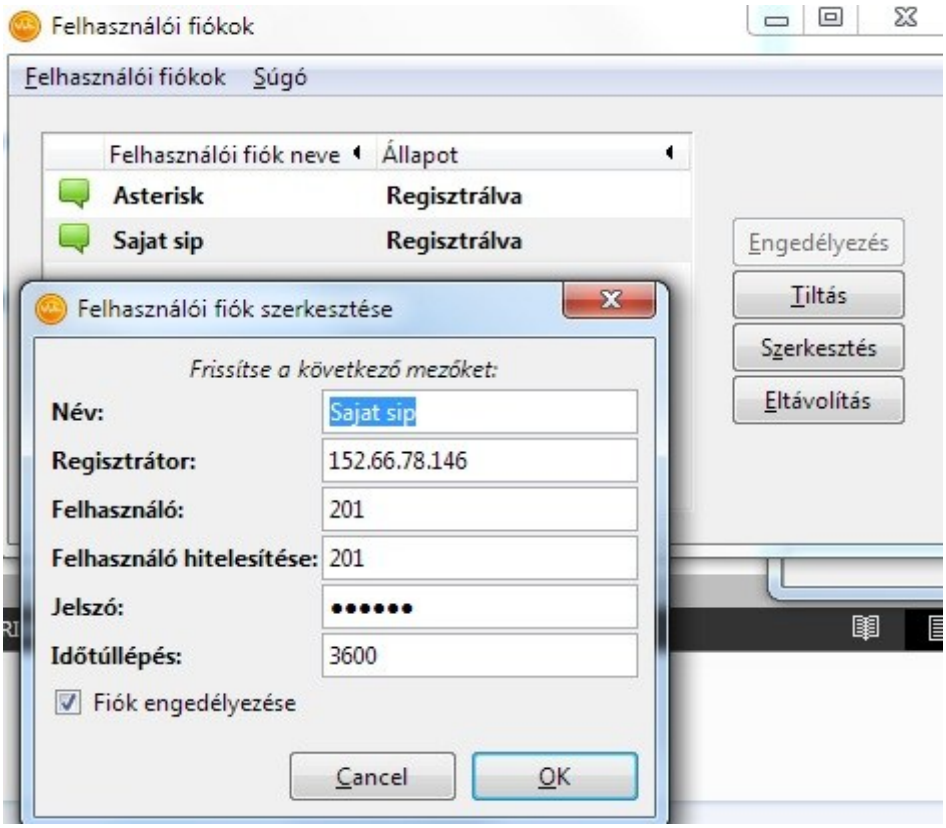
Menu access

Volume-control

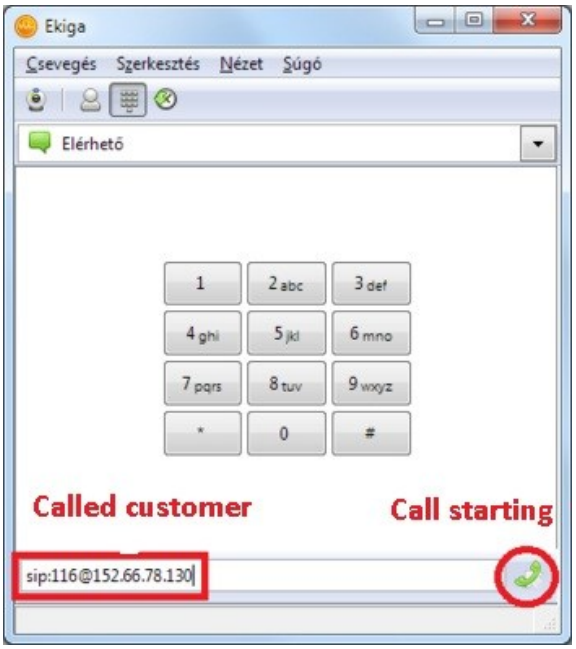
Speakerphone

Information about the Ekiga software

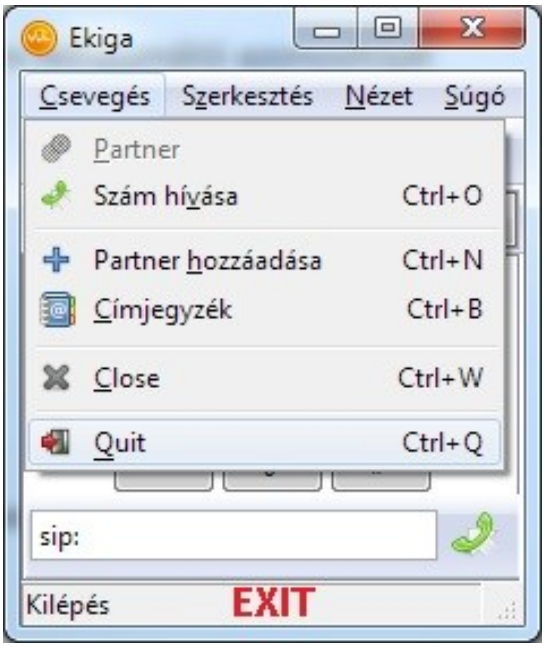
Exit and close the Ekiga: X pushing is not enough, you need to to close from the menu (see 15. Figure)
When you are calling to a number, you need to write not only the phonenumber but also the IP address of the used Proxy server. For example: 101@152.66.78.130



13. Figure Ekiga: Customer setting



14. Figure Ekiga: Call starting



15. Figure Definitive closure