GÉANT
Networks • Services • People

Installation and configuration of H.323 Gatekeeper

Best Practice Document

Produced by the AMRES-led Multimedia – VoIP working group

Author: Ognjen Milosavljevic (RCUB)

April 2016

Document No: GN4-1-NA3-T2-AMRES-BDP-116
Version / date: V1.2 / 20-04-2016
Original language : Serbian
Original title: "Instalacija i konfiguracija H.323 Gatekeeper-a"
Original version / date: Version 1 / 8. December 2014
Contact: ognjen.milosavljevic@rcub.bg.ac.rs

AMRES/RCUB is responsible for the contents of this document. The document was developed by the AMRES Multimedia – VoIP (AMRES BPD 116 topic) working group.

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 691567 (GN4-1).

Table of Contents

1 Executive Summary 1

2 A Description of the H323 Technology 2
- 2.1 H.323 2
- 2.2 H.323 Gatekeeper 4
- 2.3 Global Dialling Scheme (GDS) 7

3 A Description of the Solution 9
- 3.1 The National Gatekeeper 9
- 3.2 The NREN Gatekeeper 10
- 3.3 The Institutional Gatekeeper 11

4 Installing the GNU Gatekeeper 12

5 Configuring the Gatekeeper 13
- 5.1.1 The National Gatekeeper 13
- 5.1.2 The NREN Gatekeeper 16
- 5.1.3 The Institutional Gatekeeper 19
- 5.2 Configuring the *iptables* Tool 20

Appendix A GNU-GK installation script 21

References 27

Glossary 28

Table of Figures

Figure 2.1: Logical elements in the H.323 environment (the figure taken from [1]) 2
Figure 2.2: Routing the call signalling through the gatekeeper 6
Figure 2.3: Calling a terminal from another zone 6
Figure 2.4: Hierarchy in the GDS network 8
Figure 3.1: Structural scheme of the design 9

1 Executive Summary

This paper describes the theoretical basis of the H.323 protocol with the aim of presenting a solution for the software implementation and integration of H.323 Gatekeeper into the existing infrastructure of the Academic Network of Serbia (AMRES).

A gatekeeper is the central controlling device in the H.323 environment and includes functions such as admission control, translation of H.323 addresses to IP addresses, bandwidth control, zone management, call control and additional services during the call. The gatekeeper can also serve as a proxy server, i.e. the entire traffic between the endpoints (terminals) participating in the communication can be routed through the gatekeeper for security or quality control purposes.

Summary (in Serbian)

U ovom radu je opisana teorijska osnova H.323 protokola, sa ciljem predstavljanja rešenja za implementaciju i integraciju H.323 gatekeeper-a u postojeću infrastrukturu Akademske mreže Srbije (AMRES). Prikazana je izabrana softverska implementacija rešenja za gatekeeper.

2 A Description of the H323 Technology

This section describes the technologies that are necessary for understanding the configuration of the gatekeeper. A short overview of the H.323 protocol is presented, with a special emphasis on the H.323. gatekeeper. The description also includes the hierarchy of the GDS (Global Dialling Scheme) and information on GDS numbering.

2.1 H.323

H.323 is a recommendation of the ITU-T (ITU Telecommunication Standardisation Sector), the body that defines protocols for audio-visual communication in IP-based networks. H.323 defines several logical elements shown Figure 2.1, and their roles are explained below:

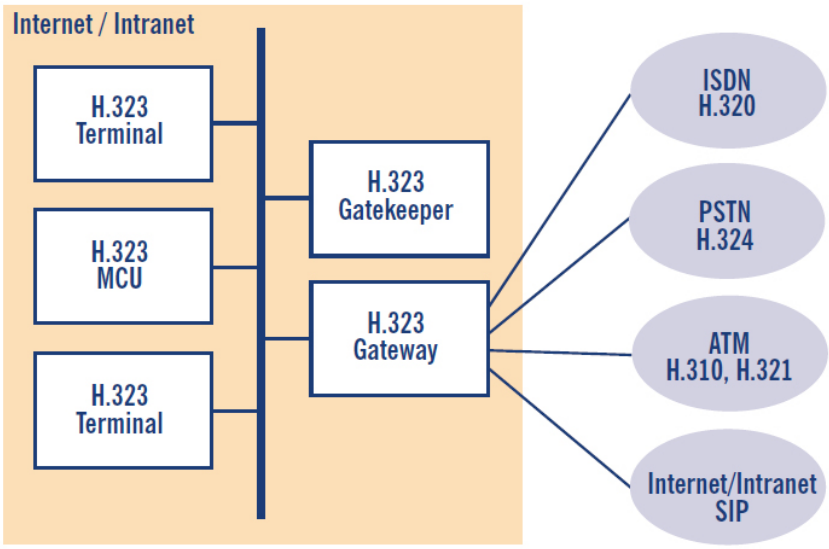

Figure 2.1: Logical elements in the H.323 environment (the figure taken from [1])

- **H.323 Gatekeeper** is the central controlling device in the H.323 environment, which includes a set of functionalities such as admission control, translation of H.323 addresses to IP

addresses, bandwidth control, zone management, call control, additional services during the call, etc. Besides this, the gatekeeper can also act as a proxy server, i.e. the entire traffic between the endpoints (terminals) participating in the communication can be routed through the gatekeeper for security or quality control purposes.

- **H.323 Terminal** is an H.323-compatible user device accessing the H.323 network in order to establish a video conference call. This can be achieved by way of software, i.e. as a program installed on a certain computer, or by way of hardware, in the form of a separate device (e.g. an H.323 video conference device). The terminals have their respective identifications in the form of a username and password and/or number, whereby they are uniquely identified on the H.323 network.
- **H.323 MCU** (Multipoint Control Unit) is a combination of an MC (Multipoint Controller) and an MP (Multipoint Processor) within one device and it is used for establishing conference calls involving more than two clients.
- **MC** (Multipoint Controller) is a device that connects the signalling channels of two or more H.323 terminals in a star-shaped fashion. The MC performs the negotiation process about the audio-visual capabilities of the H.323 terminals in order to maintain the best possible communication. Also, the MC controls resource consumption during a multicast video conference. Besides the MCU, an MC can be used as part of any H.323 device (terminal, gateway, gatekeeper), and it can be implemented as a separate device.
- **MP** (Multipoint Processor) serves for receiving and processing audio-visual data from individual endpoints, and for sending the data to other endpoints. The MP performs the conversion between different codecs and bandwidths.
- **H.323 Gateway** is a device that connects the H.323 environment with other protocols and/or networks, such as ISDN, PSTN, ATM and other telephone networks.

The following three control protocols are used for the communication between the components within the H.323 environment:

- **H.225.0 RAS** (Registration, Admission and Status) - this control protocol is used for communication between the endpoints and the gatekeeper, and for communication between different gatekeepers. The endpoints rely on RAS for registering with the gatekeeper, for requesting permission to use the system resources, to send requests for the IP address translation of another endpoint etc. The gatekeeper uses RAS for maintaining the list of registered endpoints and for gathering the resource information at the end of the call. RAS provides a user authentication and call authorisation mechanism.
- **H.225.0 Call Signalling** - this channel serves for establishing calls and for messages such as ringing, successfully established or failed connections, and for additional services during the call. These messages are derived from the Q.931 signalling (IDSN call signalling), but the procedure is simplified and only a subset of the messages is used. The signalling can be performed directly between the endpoints as well as through one or more gatekeepers.
- **H.245 Conference Control** - this channel is used for controlling the audio-visual format. It enables two or more participants in the conference to agree on a multimedia format that will be understood by all the participants. This concerns the compression, decompression and auto-configuration of multimedia content between the users. The control of the audio-visual format can be performed directly between the endpoints (terminal-terminal, terminal-MCU), as well as through one or more gatekeepers.

The audio-visual content is directly exchanged between the endpoints and the following two protocols are used for that purpose: RTP (Real Time Protocol) and RTCP (Real Time Control Protocol). An RTP session is established for each multimedia content separately. The session contains an IP address and a pair of ports, the one port for the RTP, and the other port for the RTCP. According to the specification under Reference [2], the RTP port should be an even number, while a larger odd number should be used for the RTCP. Available UDP ports (1024-65565) are usually used for that. The RTP uses the UDP protocol to establish a number of connections between two points. The RTP header contains the ordinal number of the packet, the time of its creation, the source that generated the packet, etc. The RTCP contains the information on data quality, the number of listeners, the identification of listeners and senders, etc. Moreover, the SRTP (Secure RTP) can be used for additional data security.

There are several types of H.323 addresses, i.e. several ways in which endpoints can be identified in the H.323 environment. They include E.164 numbers, H.323 URLs having the h323:user@domain.com structure, various symbolic names, and IP addresses.

2.2 H.323 Gatekeeper

As noted above, the gatekeeper is the central control and management device in the H.323 environment. According to the ITU-T H.323 recommendations, the gatekeeper should enable the following:

- Address translation.
- Admission control.
- Bandwidth control.
- Zone management.
- Call control signalling.
- Call authorisation.
- Bandwidth control
- Call management.

The most common way of setting up the client device for a gatekeeper is through a static configuration of the IP address, which is the default manner referred to in this paper. Another way of configuration involves sending multicast messages to the 224.0.1.41 address in order to find the gatekeeper.

Signalling messages can be sent directly between the endpoints or through the gatekeeper. In the first case, the gatekeeper cannot monitor the course of the call because the signalling, or part thereof, does not go through it, while in the second case the gatekeeper can control the entire course of the call, or parts thereof, depending on the signalling messages that pass through it. The gatekeeper is configured to work in one of the following three signalling models, depending on the type of messages that need to pass through it:

- Direct signalling. In this model, only the H.225.0 RAS messages pass through the gatekeeper, while the other messages go directly between the endpoints.

- Routing the call signalling through the gatekeeper. In this model the H.225.0 RAS and H.225.0 signalling messages pass through the gatekeeper, while the H.245 signalling goes directly between the endpoints.
- Routing the H.245, H.225.0 RAS and H.225.0 through the gatekeeper. In this model, all the signalling messages go through the gatekeeper.

In all three models, the audio-visual content is exchanged directly between the endpoints.

On establishing a call, the calling endpoint sends the ARQ (Admission Request) message to the gatekeeper and requests access to the network; the gatekeeper answers with an ACF (Admission Confirmation) if access is granted, or with an ARJ (Admission Reject) if access is rejected. These messages belong to the H.225.0 RAS control protocol. Upon approval, there ensues an exchange of SETUP messages that are part of the H.225.0 call signalling protocol. The IP address to which the SETUP messages are sent is contained in the ACF message. Depending on the signalling model, this address is either the address of the calling terminal or the address of the gatekeeper. Following the exchange of the SETUP messages, the called side addresses the gatekeeper with an ARQ message and, if it receives an ACF response, it sends the CONNECT message to the calling terminal or to the gatekeeper, depending on the signalling model. When the call is established, the H.245 conference control channel is opened through which the terminals negotiate the format of the audio-visual content. This channel can also go through the gatekeeper. At the end of the call, the terminal sends the RELEASE COMPLETE message to the other terminal (through the gatekeeper or directly), which is part of the H.225.0 call signalling protocol. The gatekeeper must be informed when the call is ended, and it is achieved by each of the endpoints that participated in the conversation sending a DRQ (Disengage Request) message to the gatekeeper. In response, the gatekeeper sends a DCF (Disengage Confirmation) message. Figure 2.2 shows the exchange of signalling messages when the model of routing the call signalling through the gatekeeper is applied. It shows that the H.225.0 RAS and H.225.0 messages go through the gatekeeper, wherefore the SETUP AND CONNECT messages are sent to the gatekeeper in addition to the ARQ and ACF messages. The H.245 conference control channel is established directly between the terminals. After the call is ended (following receipt of the RELEASE COMPLETE message), the terminals and the gatekeeper exchange the DRQ and DCF messages.

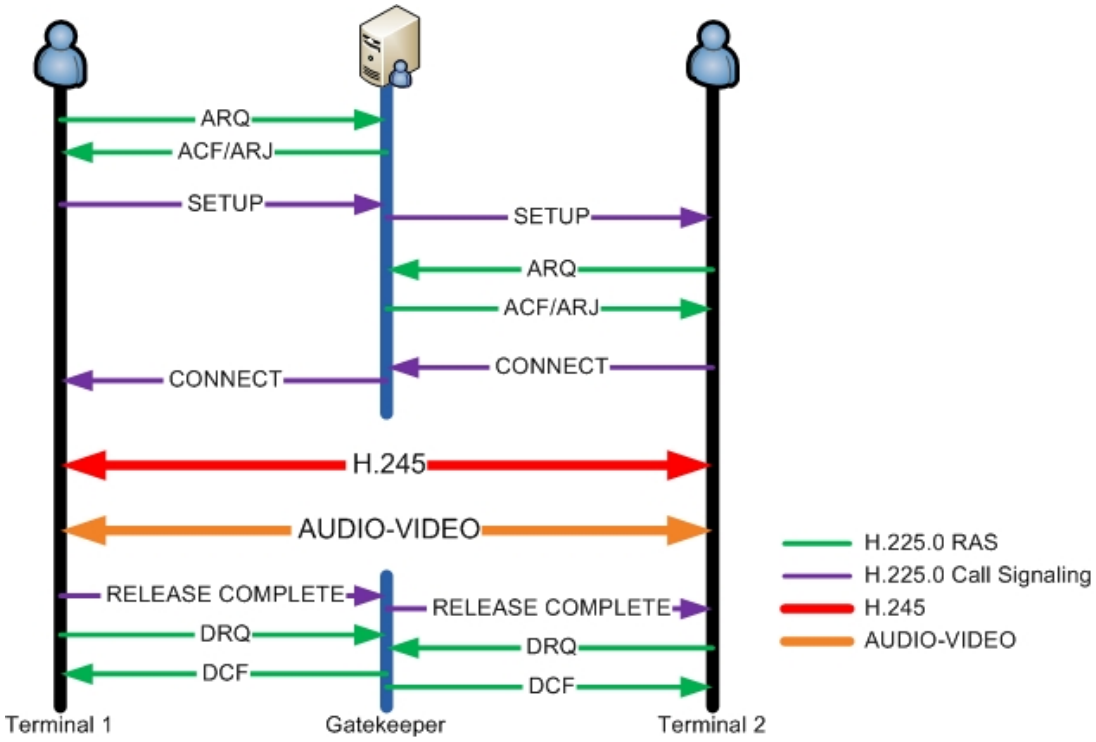

Figure 2.2: Routing the call signalling through the gatekeeper

Gatekeepers are organised into zones. A zone comprises all endpoints (terminals, MCUs and gateways) registered with one gatekeeper. Communication between the zones amounts to communication between the gatekeepers.

An example of this communication is the calling of an endpoint from another zone, as shown in Figure 2.3. In this case, gatekeeper 1 from the zone initiating the call sends an LRQ (Location Request) message to gatekeeper 2 of the calling zone, and it returns an LCF (Location Confirm) message with the calling terminal information, or an LRJ (Location Reject) message if the terminal is not registered in that zone. If the LRQ is forwarded between several gatekeepers, the LCF message is sent directly to the gatekeeper that sent the LRQ request.

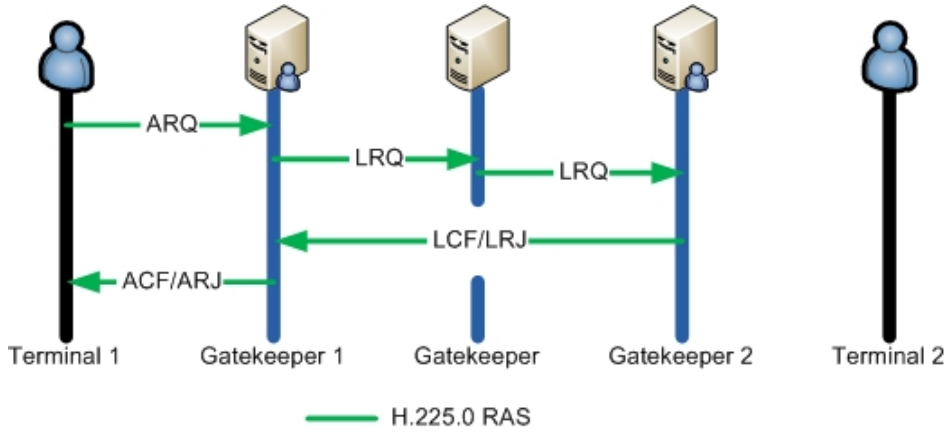

Figure 2.3: Calling a terminal from another zone

2.3 Global Dialling Scheme (GDS)

The GDS is a numbering plan for global audio-visual communication over IP networks developed by several NREN (National Research and Education Network) organisations. The GDS was modelled on the existing international numbering plan used in traditional telephony with a few exceptions. eduCONF, as part of the GEANT project, is aimed at perfecting videoconferencing services and thereby the GDS as its integral part.

The GDS makes it possible for each videoconferencing device, MCU or gateway to be allocated a unique number.

Each number contains four parts: <IAC><CC><OP><EN>

- IAC (International Access Code) – international number, also called the world gatekeeper prefix. It is defined as 00.
- CC (Country Code) – this code follows the ITU international country codes so the CC for Serbia is 381.
- OP (Organisation Prefix) – this is an organisational prefix. Many NREN organisations in certain countries follow the national number allocation system and take the numbers as their own organisational code; in Serbia, this number would be 11 for Belgrade. Some, however, use their own administration numbers, while others choose numbers randomly. A limitation as to the code allocation is in that it needs to be unique at the state level, which is taken care of by an organisation possessing the national gatekeeper (usually it is NREN, as will be the case with AMRES in Serbia).
- EN (Endpoint Number) – this is the number of the endpoint. It is allocated at the level of organisation and it can be any number at the level of the given organisation. It is recommended that the EN number is no longer than seven digits.

The GDS also defines an alphanumeric identification plan, where identification is expressed in the form of <user_name>@<domain_organisation>.

The gatekeepers in the GDS are hierarchically connected at several levels as shown in Figure 2.4.

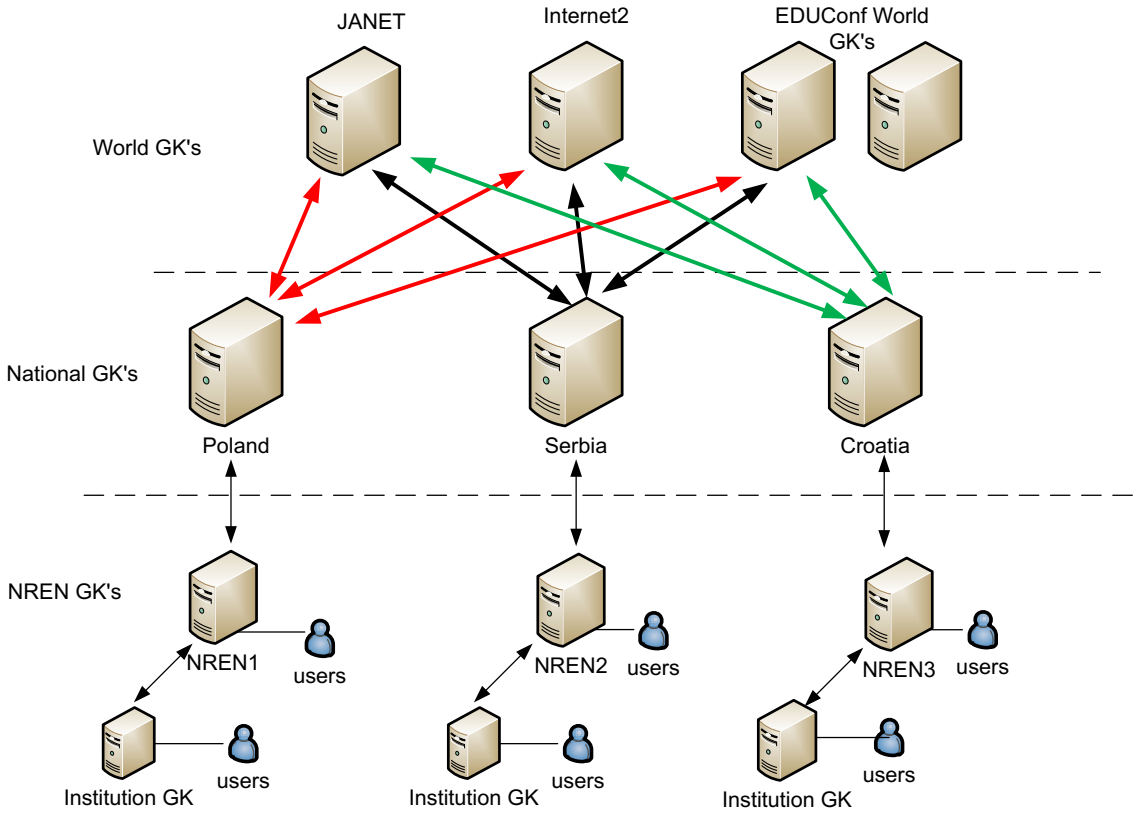

Figure 2.4: Hierarchy in the GDS network

At the top of the hierarchy is a network of world gatekeepers whose zone prefix (IAC) is 00, which are therefore responsible for resolving international calls. eduCONF, Internet2 and JANET secure and maintain some of the world gatekeepers. Below them are the national gatekeepers with their own zone prefixes, i.e. national prefixes (CC) (e.g. for Serbia this prefix is 381, for Croatia 385). These gatekeepers are in charge of calls between organisations within a country and the forwarding of requests for international calls to the world's gatekeepers. The next level involves the gatekeeper and the organisation and they have their zone prefixes, i.e. the OP prefixes from the GDS plan. Endpoint numbers (EN) with their identification numbers and institution gatekeepers are registered at the organisation's gatekeeper. The above structure is convenient because it provides a clear hierarchy and division of responsibilities.

3 A Description of the Solution

The AMRES solution (Figure 3.1) incorporates three installed gatekeepers: the first one for the functionality of the institution's local videoconferencing calls – Institutional gatekeeper; the second one for the functionality of the local videoconferencing calls of all AMRES users that do not have a gatekeeper of their own institution – NREN gatekeeper called AMRES_GK; and the third one for connecting to the world's GDS network – the national gatekeeper called SERBIA_GK. In order to connect to the world's GDS network, according to the instructions of the GDS community, it is necessary to contact the organisation responsible for connecting national gatekeepers with the world's gatekeepers. If greater gatekeeper reliability is required, it is necessary to install a redundant gatekeeper that would take over the service in the event of a failure of the main gatekeeper.

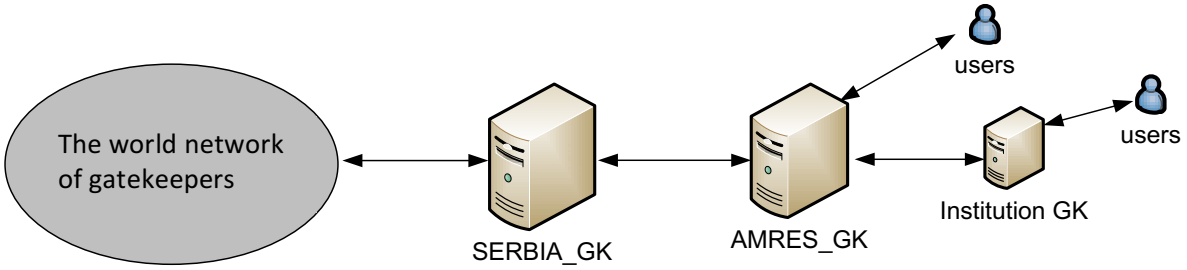

Figure 3.1: Structural scheme of the design

Videoconferencing terminals at institutions that do not require an institutional gatekeeper can use the AMRES_GK gatekeeper. If the institution requires a separate institutional gatekeeper, that gatekeeper can be connected to the GDS network via the AMRES_GK gatekeeper of the Academic Network.

3.1 The National Gatekeeper

The function of the national gatekeeper (SERBIA_GK) is to receive LRQ (Location Request) messages from the AMRES_GK gatekeeper and forward them to the gatekeeper that can resolve them.

The SERBIA_GK gatekeeper should receive LRQ requests from the world's gatekeepers for resolving the 00381 prefix and forward such requests to the gatekeeper in the national network responsible for the OP (organisational prefix), i.e. the number following the 00381 prefix in the GDS number. For instance, the SERBIA_GK gatekeeper will forward an LRQ sent by the world's gatekeepers with a number that contains the 003819 prefix to the AMRES_GK gatekeeper because the OP number of AMRES is 9.

The LRQ requests that the SERBIA_GK gatekeeper receives from the gatekeepers in the national network include all requests with the 00 prefix and they are forwarded to the neighbouring gatekeepers depending on whether the request pertains to resolving a number from Serbia or another country, i.e. on whether the call was directed to a terminal in Serbia or is an international call.

If it is an international call (if the number does not begin with 00381), the SERBIA_GK gatekeeper will forward the LRQ request received from a gatekeeper in the national network to the world's gatekeeper network, i.e. to the network of gatekeepers at a higher hierarchical level. For example, if the 00385 prefix is dialled, the LRQ will be forwarded to the world's gatekeeper network since it is a call to a terminal in Croatia, and according to the GDS hierarchy it can only be reached through the world's gatekeeper network.

If the call was sent to a terminal in Serbia, the LRQ request with the 00381 prefix received by a gatekeeper from the national network will be forwarded to the gatekeeper in the national network responsible for the OP of the given GDS number. For instance, in the case of a call to a number with the prefix 003819 initiated from a terminal in the national network, but not from AMRES, the LRQ request received by the SERBIA_GK gatekeeper will be forwarded to the AMRES_GK gatekeeper because 9 is the OP number of AMRES.

The SREBIA_GK gatekeeper will not receive LRQ requests from organisational gatekeepers with an organisation's prefix that are below the national gatekeeper in the hierarchy because such calls are actually calls within the given organisation, nor will it receive LRQ requests from the world's gatekeepers with a prefix that is not 00381, which are above the national gatekeeper in the hierarchy. For instance, the SERBIA_GK gatekeeper will not receive LRQ requests from the AMRES_GK gatekeeper with the prefix 003819 since these are local calls within AMRES; if the AMRES_GK gatekeeper is not able to locate the given number in its registration table, then the terminal with that number is not registered and there is no need to send an LRQ request.

3.2 The NREN Gatekeeper

The NREN_GK gatekeeper serves the needs of the NREN users. Its role is to secure local video calls and resolve national and international calls (resolution of GDS numbers to specific IP addresses). Also, in cases where the national gatekeeper is connected to the GDS network via the NREN gatekeeper, it should ensure call forwarding from and to the institutional gatekeeper.

The gatekeeper user is registered to the NREN gatekeeper of his/her institution. GDS numbers are used for the registration of users as described in section 2.3. The user is registered with the gatekeeper by way of the identification number. On registration, the user's IP address and identification number are entered into the registration table of the gatekeeper. If the registration table lacks the information on a given number, we can have the following three scenarios (the example concerns the AMRES gatekeeper in charge of the prefix 003819):

- If the prefix is 00XYZ - the LRQ request is sent to the national gatekeeper.
- If the prefix is 003819 (the prefix selected for the AMRES GDS numbers) - the request is not forwarded to the national gatekeeper because it is a local call within the AMRES, and the number is not registered in that case.
- If the prefix is 003819XY (where XY is the OP of the institution whose gatekeeper is connected to the GDS network via the AMRES_GK) - the request is forwarded to the institutional gatekeeper.

The signalling method used is the routing of the call signalling through the gatekeeper, as described in section 2.2. This method ensures that the H.225.0 signalling messages pass through the gatekeeper, so that the gatekeeper can monitor the course of the call and eventually gather the information about the call. The call information - CDR (Call Detail Record) – is recorded by the gatekeeper in the corresponding log file (cdr.log). The information contained in these messages includes: the IP address and GDS number of the calling terminal, the IP address and GDS number of the called terminal, the start and end time of the call, call duration and other data.

3.3 The Institutional Gatekeeper

If an institution wants to have its own prefix, and to monitor and control the calls and access, it should have its own gatekeeper within the institution. The institutional gatekeeper should provide for local video calls and the resolution of national and international calls (translation of GDS numbers to IP addresses).

A user is registered with the gatekeeper by way of an identification number, which has the prefix of the institution (OP) XY. During registration, the user IP address and identification number are entered into the registration table of the institutional gatekeeper.

If the registration table does not contain the information on a given number, we can have the following three scenarios:

- If the prefix is 00 - the LRQ request is sent to the AMRES gatekeeper, which forwards the LRQ to the national gatekeeper.
- If the prefix is 003819, but not 003819XY (where XY is the OP of the institution) - the LRQ request is sent to the AMRES gatekeeper, which searches for the called number in the registration table.
- If the prefix is 003819XY (where XY is the OP of the institution whose gatekeeper is connected to the GDS network via the AMRES_GK) - the LRQ request is not forwarded to the AMRES gatekeeper because it is a local call within the institution, and the number is not registered in that case.

4 Installing the GNU Gatekeeper

A GNU gatekeeper (usually abbreviated as GnuGk) has been chosen as the gatekeeper in AMRES because it reliably provides all the H.323 gatekeeper functions and it is available under the GPL licence. The GNU gatekeeper has been chosen as the solution for the needs of the eduCONF world gatekeeper.

The Linux operating system has been chosen for the installation of the GNU gatekeeper. The installation relies on the script written for the Red Hat and Debian distributions of the operating system. The script was prepared as part of the GN3plus eduCONF task to automate the process of gatekeeper installation. In order to run the script successfully, the user who runs it needs to have root privileges. The script is provided in Appendix 1 and it can also be downloaded from: [GatekeeperScript]. Note that the permissions of the script file should be set to r-x------.

At the beginning of the script, it is established which Linux distribution runs the script. Based on the established distribution, the following packets are installed by way of the *apt-get* or *yum* command: *flex*, *bison*, *pcg-donfig*, *pkgconfig*, *gcc*, *gcc-c++*, *g++*, *make*, *autoconf*, *automake* and *wget*. If the packets have already been installed, they will be updated by running the script.

After installing the necessary packets, the *gnugkinstall.tar.gz* packet is downloaded from the eduCONF GÉANT site. The packet contains the GnuGk gatekeeper and all the libraries necessary for compiling it, PTLib and H323Plus. The versions of the libraries have been tested and they are compatible with the version of the GnuGk gatekeeper. The downloaded packet is unpacked in the temporary folder containing the script, whereupon the PTLib library is installed first and then the H323Plus. During the installation/compilation, all the log data related to the compiling is written in the corresponding files in the log folder.

After compiling the necessary libraries, the GNU gatekeeper will be compiled. In the course of compiling the gatekeeper, the script offers the installation of add-ons necessary for using *mysql*, *radius*, *postgresql*.

Following the installation, the GnuGk daemon will be located in the /etc/init.d/ folder. The Gatekeeper is started using the *gnugkd service start* command and stopped using the *gnugkd service stop* command. In order to properly start the gatekeeper daemon with the *gnugkd service start* command, the *gnugk.ini* configuration file should be located in the /etc/ folder.

5 Configuring the Gatekeeper

Section 3, which describes the structure of the solution, also refers to the functionalities that should be provided by the National, NREN and Institutional gatekeepers. Accordingly, below is a detailed presentation and explanation of the contents of the configuration files of the three GNUGKs, using the example of the AMRES solution and in accordance with References [3] and [4].

The configuration file is a standard text file, with a basic format as follows:

- [Section String]
- Key Name=Value String

The first line contains the name of the section and the second line lists the configuration parameter and its value. Comments in the file begin with the sign ; or #.

Of all the sections in the configuration file, only those necessary for meeting the needs of AMRES will be discussed here; they include global H.323 communications via the IPv4 and IPv6 protocols and the logging of H.323 communications. The complete configuration file with all the sections and parameters - *complete.ini* - can be found in the */gnugkinstall/gnugk/etc/* folder.

5.1.1 The National Gatekeeper

The configuration for the national gatekeeper is provided through the example of SERBIA_GK. Each configuration file contains the *main* section in which the basic configuration parameters are set. Below is an example of the section.

- [Gatekeeper::Main]
- Fortytwo=42 indicates that the gatekeeper.ini configuration file is present. If this line did not exist at the start of the application, a warning would be displayed.
- Name=SERBIA_GK is the name (identifier) of the gatekeeper. Terminals and gateways get this parameter during the "discovery" of available gatekeepers. After the "discovery", the terminals and gateways are registered with only one gatekeeper using the identifier they receive.
- EnableIPv6=1 enables the use of IPv6 on the gatekeeper.
- Home=128.66.4.205, [2001:DB8:0:1::205] are the IPv4 (128.66.4.205) and/or IPv6 (2001:DB8:0:1::205) addresses at which the gatekeeper listens for requests.
- TraceLevel=2 is the setting of the trace level of information written in the log file. There are 6 trace levels, and a detailed list of what each level shows can be found in Reference [4].
- TotalBandwidth=100000 is the bandwidth allowed for all the endpoints, expressed in 100 bits, i.e. 100000 equals 1Mb/s.
- StatusPort=7000 is the port through which the activities on the gatekeeper can be monitored; it is called the status port.
- StatusTraceLevel=2 is the trace level of information written to the status port; it should not be confused with the previous trace level. There are three trace levels of information

presented to the status port (0, where the information and answers to the entered commands are shown; 1, which shows the routing data and call details in addition to the information presented for 0; and 2, which shows all activities).

- UseBroadcastListener=0 turns off the listening for broadcast RAS requests.
- rule=explicit is the rule according to which the access selection is performed. There are several such rules, including forbid (denies access to all), allow (grants access to all), explicit (specific IP addresses are granted or denied access), regex (a set of IP addresses is granted or denied access), and password (access is granted based on username and password).
- 127.0.0.1=allow the right to access the status port is assigned only from the local server. This rule is selected because it enables us to grant access to the status port from remote computers by simply adding an IP addresses.
- default=forbid denies access to all other IP addresses.
- Shutdown=forbid prevents the gatekeeper being shut down through the status port.
- DelayReject=5 is the time, expressed in seconds, after which the user who entered an incorrect username or password is denied access to the status port.

The following two sections serve for the authentication of users at the gatekeeper; there are several rules that may be used for user authentication, such as SimplePasswordAuth, AliasAuth, FileIPAuth, PrefixAuth, SQLAuth, etc. It has been selected to impose access limitations based on IP address, which is ensured through the FileIPAuth rule.

- [Gatekeeper::Auth]
- FileIPAuth=required;RRQ,LRQ,Setup indicates that a user who attempts to register must meet the requirements set in this rule or his/her registration will be denied.
- any=reject indicates that no IP address has the right of access, which conforms to the solution because the national gatekeeper serves exclusively for forwarding LRQ messages (no endpoints are registered with it).

The next section contains the configuration of the gatekeeper log file, specifying the path to the log file, the time span and the exact time of replacement of this file with the new one.

- [LogFile]
- Rotate=Weekly
- RotateDay=Sun
- RotateTime=03:59
- Filename=/var/log/gnugk/gnugk.log

The above shows that the replacement will take place weekly, on Sundays at 03:59; at that time the gatekeeper will save the old file as gnugk.log and add the date to the name, then make a new gnugk.log file.

The *RoutedMode* section was only assigned the TCP port for signalling messages. There is no need for detailed settings because this gatekeeper serves only for forwarding LRQ messages, as explained above.

- [RoutedMode]

- CallSignalPort=1720

The following is the set of sections for configuring the relationship with the neighbouring gatekeepers, including the data on each neighbour.

The identifiers of all the neighbours are specified in the [RasSrv::Neighbors] section, regardless of their position in the hierarchy. The type of gatekeeper installed on the neighbours is also listed.

- [RasSrv::Neighbors]
- AMRES_GK=GnuGk
- eduCONF=GnuGk
- Janet=CiscoGk

The next section contains data on the neighbouring gatekeeper (in this case it is the AMRES_GK).

- [Neighbor::AMRES_GK]
- `GatekeeperIdentifier=AMRES_GK` is the name of the gatekeeper.
- `Host=128.66.4.79`, is the IPv4 address assigned to the neighbouring gatekeeper.
- `SendPrefixes=003819` is the prefix of the zone the neighbouring gatekeeper is in charge of. All calls containing this prefix will be forwarded to this gatekeeper for resolution.
- `AcceptPrefixes=00,!003819` are the prefixes of the zones that cannot be resolved by the neighbouring gatekeeper. All calls containing these prefixes are accepted by this gatekeeper in order to be resolved or forwarded to the next neighbour for resolution. !003819 indicates that the prefix 003819 is rejected by this neighbour.
- `ForwardLRQ=always` is the rule requiring that the LRQ request be unconditionally forwarded to other gatekeepers. If it is set to *depends*, the option *ForwardHopCount* will be checked.
- `ForwardHopCount=10` is the number that the gatekeeper decreases by 1 if it forwards the LRQ request to another gatekeeper. If it is set to *ForwardLRQ=depends* and this number is 0 and the LRQ request will not be forwarded further.

The following sections contain the data on two gatekeepers from the world network. As is the case with all the neighbours, it specifies the data on identifications, IP addresses, prefixes and LRQ forwarding. The IP addresses in these examples are fictitious.

- [Neighbor::eduCONF]
- GatekeeperIdentifier=eduCONF
- Host=128.66.31.225
- SendPrefixes=00,!00381
- AcceptPrefixes=00381
- ForwardLRQ=always
- ForwardHopCount=10

- [Neighbor::Janet]
- GatekeeperIdentifier=USA

- Host=128.66.240.220
- SendPrefixes=00,!00381
- AcceptPrefixes=00381
- ForwardLRQ=always
- ForwardHopCount=10

The world gatekeepers are sent the 00 prefix (exclusive of 00381) for resolution, and requests for resolution of the 00381 prefix are received from them.

The last section of the configuration file contains the settings for LRQ and LCF messages that are common for all the neighbours.

- [RasSrv::LRQFeatures]
- NeighborTimeout=5
- SendRetries=3
- AcceptForwardedLRQ=1
- AcceptNonNeighborLCF=1
- AcceptNonNeighborLRQ=1
- IncludeDestinationInfoInLCF=0
- CiscoGKCompatible=1

The first two lines specify the time, expressed in seconds, within which the answer must arrive from the neighbour or an ARJ message will be sent to the caller, and the number of retries for sending the LRQ request to the neighbour before the LRJ is sent to the sender of the request. The next three lines indicate that the gatekeeper will accept forwarded LCF and LRQ messages from gatekeepers that are not its neighbours. The last two lines refer to compatibility with other gatekeepers and certain terminals.

5.1.2 The NREN Gatekeeper

The NREN gatekeeper configuration is provided using the example of the AMRES_GK. This gatekeeper is envisaged for work with endpoints and the institutional gatekeeper, so its configuration differs in some aspects from that of the national gatekeeper. As regards the *main* and admission control sections, they are the same, except that the identification and the IP address are set for this gatekeeper.

- [Gatekeeper::Main]
- Fortytwo=42
- Name=AMRES_GK
- EnableIPv6=1
- Home=128.66.4.79, [2001:DB8:0:4::79]
- TimeToLive=1000
- TraceLevel=2

- StatusPort=7000
- StatusTraceLevel=2
- UseBroadcastListener=0

- [GkStatus::Auth]
- rule=explicit
- 127.0.0.1=allow
- default=forbid
- Shutdown=forbid

The FileIPAuth option is used for user authentication. In our case it is allowed to register from any IP address because mobile clients are required to register when they are not in the AMRES network.

- [Gatekeeper::Auth]
- FileIPAuth=required;RRQ,LRQ,Setup

- [FileIPAuth]
- any=allow

The gatekeeper log file configuration is the same as the one at the SERBIA_GK.

- [LogFile]
- Rotate=Weekly
- RotateDay=Sun
- RotateTime=03:59
- Filename=/var/log/gnugk/gnugk.log

The *Acct* section is used for reporting on the performed calls, and this information is provided in the form of CDR strings.

- [Gatekeeper::Acct]
- FileAcct=alternative;start,stop is the rule by which the CDR strings are collected, and it regulates that they are to be collected into a file; the start and the end of the call are the events monitored.
- [FileAcct]
- DetailFile=/var/log/gnugk/cdr.log are the path and name of the file into which the logs are collected, while the other rules have the same meaning as with log files.
- Rotate=weekly
- RotateDay=Sun
- RotateTime=04:59

It should be noted that the gathering of information on calls would not be possible if the direct signalling mode were selected since in that case the H.225.0 signalling messages would not go through the gatekeeper.

The *Routing Mode* section sets the signalling mode to be used, and in this case the routing of the call signalling through the gatekeeper has been selected as the mode in accordance with the structure of the solution.

- [RoutedMode]
- GKRouted=1 enables the routing of signalling messages through the gatekeeper.
- H245Routed=0 disables the routing of H.245 control messages (these messages are exchanged directly between the terminals).
- AcceptNeighborsCalls=1
- AcceptUnregisteredCalls=1 enable the acceptance of calls from other zones and calls by unregistered endpoints.
- SupportNATedEndpoints=1 enables access support for the endpoints located behind the NAT.

The following section serves for setting up the relationship with neighbours, and it has some minor differences compared to the settings of the SERBIA_GK.

- [RasSrv::Neighbors]
- SERBIA_GK=GnuGk
- INST_GK=GnuGk

- [Neighbor::SERBIA_GK]
- GatekeeperIdentifier=SERBIA_GK
- Host=128.66.4.205
- SendPrefixes=00,!003819
- AcceptPrefixes=003819

- [Neighbor::INST_GK]
- GatekeeperIdentifier=INST_GK
- Host=128.66.4.102
- SendPrefixes=0038191
- AcceptPrefixes=*,!0038191
- ForwardLRQ=always
- ForwardHopCount=10

- [RasSrv::LRQFeatures]

- NeighborTimeout=5 is the time to wait for the response by the neighbouring gatekeeper before the LRJ message is returned to the terminal.
- SendRetries=3 is the number of attempts at sending the LRQ to the neighbouring gatekeeper, after which the terminal is sent the LRJ message.
- AcceptForwardedLRQ=1 enables the acceptance of LRQ requests received from the neighbouring gatekeepers.
- AcceptNonNeighborLCF=1 enables the gatekeeper to accept LCF messages from gatekeepers that are not on the list of neighbours. This poses a significant security risk and one needs to be extremely careful if this option is enabled.
- AcceptNonNeighborLRQ=1 enables the gatekeeper to accept LRQ messages from gatekeepers that are not on the list of neighbours.
- IncludeDestinationInfoInLCF=0 this option needs to be set to 0 in order to prevent the sending of information. This would enable interoperability between different terminals and gatekeepers.
- CiscoGKCompatible=1 enables compatibility between the Cisco and Gnu gatekeepers.

The neighbours are SERBIA_GK (with the IP address 128.66.4.205) and INST_GK (with the IP address 128.66.4.102); the IP addresses in the examples are fictitious. The SERBIA_GK gatekeeper receives the prefix 00 (exclusive of 003819) for resolution, and the prefix AMRES_GK 003819 is received by it. The remaining settings have the same meaning as in the SERBIA_GK.

5.1.3 The Institutional Gatekeeper

Bearing in mind that the institutional gatekeeper INST_GK is the last one in the existing hierarchy, it has only one neighbour, namely the AMRES_GK. This gatekeeper is envisaged for working with endpoints of the institution whose gatekeeper it is. Most of the settings are the same as in the case of the AMRES_GK. Within the INST_GK configuration file, there is a difference regarding the neighbour settings, where the AMRES_GK is set as the neighbour as shown in the following extract:

- [RasSrv::Neighbors]
- AMRES_GK=GnuGk

- [Neighbor::AMRES_GK]
- GatekeeperIdentifier=AMRES_GK
- Host=128.66.4.79
- SendPrefixes=*,!0038191
- AcceptPrefixes=0038191

5.2 Configuring the *iptables* Tool

It is necessary to protect the gatekeeper server in order to prevent unauthorised access to the server and attack attempts. The *iptables* tool is used for filtering the traffic on the server, which is based on the source and destination IP addresses and the TCP/UDP ports.

No destination IP addresses have been filtered within the configuration file of the gatekeepers described in this paper, because it is necessary to have endpoints registered from any networks. Therefore, it is required to filter the traffic using the *iptables* tool on destination ports. The document recommends that access be granted on the TCP and UDP ports 1718, 1719 and 1720.

Appendix A GNU-GK installation script

```
# Bash script for GNUGK
# created by ognjenm@rcub.bg.ac.rs
# Clear screen

clear
echo " Running script for GNUGK compiling"
echo " script has been run by $USER"

echo
"**********************************************************************
**"

if [ -a "$PWD/.build_run" ]
   then
         echo "Updates are all installed and has been run"
         read -p "Do you want me to run Updates again? (y/n)"
         if [ "$REPLY" == "y" ]
               then
               echo "Running update system"
               PACKAGES="flex bison pkg-config pkgconfig gcc gcc-c++ g++
make autoconf automake wget"
               if which apt-get &> /dev/null; then
               apt-get update;
               for PACKAGE in $PACKAGES; do apt-get install -y
"$PACKAGE"; done
               elif which yum &> /dev/null; then
               for PACKAGE in $PACKAGES; do yum install -y "$PACKAGE";
done
                     else
                     echo "ERROR: Unknown package-management utility."
               fi
               echo "Done..."
         fi

         date >> .build_run
   else

   PACKAGES="flex bison pkg-config pkgconfig gcc gcc-c++ g++ make
autoconf automake wget"

   if which apt-get &> /dev/null; then
         apt-get update;
   for PACKAGE in $PACKAGES; do apt-get install -y "$PACKAGE"; done
   elif which yum &> /dev/null; then
   for PACKAGE in $PACKAGES; do yum install -y "$PACKAGE"; done
```

```
          else
          echo "ERROR: Unknown package-management utility."
   fi
fi

wget http://educonf-directory.geant.net/gnugkinstall/gnugkinstall.tar.gz
tar -zxvf gnugkinstall.tar.gz
chmod -R 755 gnugkinstall
sleep 10
cd $PWD/gnugkinstall

if [ -d "$PWD/log/" ]
   then
          echo "Log directory is found"
   else
          echo "Creating log directory"
          mkdir log
fi

if [ -d "$PWD/log/" ]
   then
          echo "Log directory is found"
   else
          echo "Creating log directory"
          mkdir log
fi

echo "Running PTLIB configuration"

if [ -d "$PWD/ptlib/" ]
   then
          echo "Directory PTLIB exists"
          echo "running configure"
          echo
"************************************************************"
          echo "                PTLIB"
          echo
"************************************************************"
          echo
          echo "running configure"
          echo "please wait....."

          cd $PWD/ptlib/

       make clean >> ../log/ptlibmake.log

          ./configure --enable-ipv6 --disable-odbc
>> ../log/ptlibconf.log

          echo
"************************************************************"
          echo "Running make ......"
          echo "please wait....."
          echo

          make >> ../log/ptlibmake.log

          echo " PTLIB has been compiled continuing "
   else
```

```
         echo " There is NO PTLIB folder. Quit "
         exit
fi

cd ..
export PTLIBDIR=$PWD/ptlib/
sleep 10

echo " Running h323 Plus configuration"

if [ -d "$PWD/h323plus/" ]
   then
   cd $PWD/h323plus

         echo "running configure"
         echo
"************************************************************"
         echo "               H323"
         echo
"************************************************************"
         echo "Running compiler ..... "
         echo "please wait....."

         make clean >> ../log/h323make.log

         ./configure >> ../log/h323conf.log


         echo
"************************************************************"
         echo "Running make ......"
         echo "please wait....."
         echo

         make >> ../log/h323make.log

         echo " H323plus has been compiled continuing "
   else

         echo " There is NO H323PLUS folder. Quit "
         exit
fi

cd ..
export OPENH323DIR=$PWD/h323plus

sleep 10

echo " Running gnugk configuration"


if [ -d "$PWD/gnugk/" ]
   then
         echo  " Directory gnugk exists "

         echo

         cd $PWD/gnugk
```

```
fi

if [ -d "$PWD/gnugk/" ]
   then
         echo  " Directory gnugk exists "

         echo

         cd $PWD/gnugk
fi
         echo "Make clean from previous compile..."

         echo "running again configure"
         echo
"************************************************************"
         echo "               GNUGK"
         echo
"************************************************************"
         read -p "Do you want an advance configuration od GNUGK? (y/n)"
         if [ "$REPLY" == "y" ]
               then
                     read -p "Do you want to use MySQL (y/n)"
                     if [ "$REPLY" == "y" ]
                           then
                                 OPTIONA="--enable-mysql"
                           else
                                 OPTIONA="--disable-mysql"
                     fi

                     read -p "Do you want to use RADIUS (y/n)"
                     if [ "$REPLY" == "y" ]
                           then
                                 OPTIONB="--enable-radius"
                           else
                                 OPTIONB="--disable-radius"
                     fi

                     read -p "Do you want to use PostgreSQL (y/n)"
                     if [ "$REPLY" == "y" ]
                           then
                                 OPTIONC="--enable-pgsql"
                           else
                                 OPTIONC="--disable-pgsql"
                     fi


         else

         OPTIONH="--disable-netsnmp "
         OPTIONG="--disable-libssh"
         OPTIONF="--disable-sqlite"
         OPTIONE="--disable-unixodbc"
         OPTIONC="--disable-pgsql"
         OPTIONB="--disable-radius"
         OPTIONA="--disable-mysql"

         fi

if which apt-get &> /dev/null; then
```

```
   if [ "$OPTIONA=" == "--enable-mysql" ]
         then
         echo "Installing MYSQL dependences"
         apt-get install -y mysql 2>&1 >> ../log/gnugk.log
         echo "Done..."
   fi

   if [ "$OPTIONB=" == "--enable-radius" ]
         then
         echo "Installing RADIUS dependences"
         apt-get install -y libfreeradius-dev 2>&1 >> ../log/gnugk.log
         echo "Done..."
   fi

   if [ "$OPTIONC=" == "--enable-pgsql" ]
         then
         echo "Installing PostgreSQL dependences"
         apt-get install -y libpostgresql-ocaml-dev 2>&1
>> ../log/gnugk.log
         echo "Done..."
   fi

elif which yum &> /dev/null; then
   if [ "$OPTIONA=" == "--enable-mysql" ]
         then
         echo "Installing MYSQL dependences"
         yum install -y mysql 2>&1 >> ../log/gnugk.log
         echo "Done..."
   fi

   if [ "$OPTIONB=" == "--enable-radius" ]
         then
         echo "Installing RADIUS dependences"
         yum install -y freeradius 2>&1 >> ../log/gnugk.log
         echo "Done..."
   fi

   if [ "$OPTIONC=" == "--enable-pgsql" ]
         then
         echo "Installing PostgreSQL dependences"
         yum install -y postgresql libpostgresql-ocaml-dev 2>&1
>> ../log/gnugk.log
         echo "Done..."
   fi
else
   echo "ERROR: Unknown package-management utility."
fi

         echo "Running compiler ..... "
         echo "please wait....."

         ./configure --enable-h46018 --enable-h46023 --enable-h46017
$OPTIONA $OPTIONB $OPTIONC --disable-firebird 2>&1 >> ../log/gnugk.log

         echo
"************************************************************"
         echo "Running make ......"
         echo "please wait....."
         echo
```

```
        make optnoshared 2>&1 >> ../log/gnugk.log

   if [ -x obj_linux_x86_s/gnugk ]
         then
               cp /root/gnugkinstall/gnugk/obj_linux_x86_s/gnugk
/usr/local/bin/
               echo
"************************************************************"
               echo  "GNU GK compiled properly."
               echo
"************************************************************"
         else
               echo
"************************************************************"
               echo  "GNU GK is NOT compiled properly"
               echo
"************************************************************"
               exit 1
   fi
   if [ -d "/var/log/gnugk/" ]
         then
               echo "gnugk log directory is found"
         else
               echo "Creating gnugk log directory"
               mkdir /var/log/gnugk
   fi

   if which apt-get &> /dev/null; then
         [ ! -x /etc/init.d/gnugkd ] && wget -P /etc/init.d/
http://educonf-directory.geant.net/gnugk/startstop/debian/gnugkd
         chmod 755 /etc/init.d/gnugkd
         echo "START_ON_BOOT=yes" > /etc/default/gnugk
         update-rc.d gnugkd defaults
   elif which yum &> /dev/null; then
         [ ! -x /etc/init.d/gnugkd ] && wget -P /etc/init.d/
http://educonf-directory.geant.net/gnugk/startstop/centos/gnugkd
         chmod 755 /etc/init.d/gnugkd
         chkconfig --add gnugkd
   fi
   echo "************************************************************"
   echo  "Configuration file gnugk.ini placed in /etc/ folder"
   echo "************************************************************"
   cp etc/gnugk.ini /etc/
exit 0
```

References

[1] TERENA: IP Telephony Cookbook, mart 2004.
http://www.terena.org/activities/iptel/contents1.html

[2] Collins, Daniel (2002). "Transporting Voice by using IP". Carrier grade voice over IP. McGraw-Hill Professional
http://course.ipv6.club.tw/VoIP.941/chap2-Transporting_Voice_By_Using_IP.pdf

[3] GNU Gatekeeper Manual
http://www.gnugk.org/gnugk-manual.html

[4] Using trace levels within the GNU Gatekeeper
http://www.gnugk.org/trace-levels.html

[GatekeeperScript] https://educonf-directory.geant.net/gnugk/educonf_gnugk_builder.sh

Glossary

GDS	Global Dialling Scheme
GnuGk	GNU Gatekeeper. An open source gatekeeper available under a GNU General Public License.
ITU-T	International Telecommunication Union Telecommunication Standardization Sector
MCU	Multipoint Control Unit
NREN	National Research and Education Network
RAS	Registration, Admission and Status
RCUB	Računarski centar Univerziteta u Beogradu (Belgrade University Computer Centre)