Summary. The paper describes a complete voice communication system based on the implementation of Asterisk software PBX and additionally employing classical SIP, and novel WebRTC solutions to create a signaling system. The implementation presented explores many functions of the system with an emphasis on enhancing its didactic impact and promoting understanding of configuration and signaling aspects.

Keywords: SIP, WebRTC, PBX, VoIP, Asterisk

ANALYZA SYGNALIZACJI W HYBRYDOWYM ŚRODOWISKU WebRTC I SIP

Streszczenie. W artykule przedstawiono kompletny system komunikacji głosowej wykorzystujący zarówno programową centralę Asterisk, jak i jednocześnie dwa systemy sygnalizacji: klasyczny SIP i nowy stosowany w technologii WebRTC. Przedstawione rozwiązanie wykorzystuje jednocześnie wiele funkcji systemów, pozwalając na wyeksponowanie istotnych z punktu widzenia dydaktyki aspektów budowy systemów VoIP.

Słowa kluczowe: SIP, WebRTC, PBX, VolP, Asterisk

---

1 This work was performed under Contract No. 11.11.230.018.
1. Introduction

The current impressive developments in high-speed networks are occurring in the interest of customers who are keen to use a plethora of services and consume multigigabit traffic streams. The softwarization and virtualization of networks make it quite straightforward to offer customized, flexibly configured applications and services. Despite that observation, voice communication is still very important and widely deployed. However, with solutions such as Web Real Time Communication (WebRTC), allowing for integration of voice services with Web browsers, it is possible to introduce service intelligence, service scenario, and customization by users.

Session Initiation Protocol (SIP) protocol is introduced in order to invite a user to join established session and to efficiently and flexibly manage traffic streams for multimedia transmission in the network. The motivation being behind the introduction of WebRTC is to allow using typical Web browsers for multimedia transmission, such as voice, video, gaming, and also supporting remote collaboration among users. Second, important feature is to make such transmission in a peer-to-peer manner.

The paper is organized as follows. Section 2 briefly describes building blocks of modern softwarized voice service, such as SIP and WebRTC. Section 3 includes description of Asterisk, that can be considered as flagship open source application for voice transmission, and additionally remarks about user voice applications. Implementation and configuration aspects for voice service, are presented in Section 4, with implementation and testing results documented in Section 5. In summary, section 6 gives concise conclusions and draws further steps towards more comprehensible offer for students, and also to give the testbed installation for further research.

2. Building blocks: SIP and WebRTC

2.1. SIP architecture

SIP is a concept of session control in Internet and to invite a user to join the session. This is made by cooperating with user agents and user agent servers. Since its initial proposal SIP evolved as universal and attractive solution for VoIP systems in modern packet-based networks.

There are few features which make SIP an attractive support for VoIP systems:
- usage of syntax and formats of HTTP (Hypertext Transfer Protocol),
An Analysis of Signaling in a Hybrid WebRTC and SIP Environment

- openness of SIP message format – using e.g. XML or MIME formats,
- usage of a URI (Uniform Resource Identifier) what enables integration with Web browsers, item good scalability.

2.2. WebRTC

The motivation being behind the introduction of WebRTC is to allow using typical web browsers for multimedia transmission, such as voice, video, gaming, and also supporting remote collaboration among users. Typically, transmission of media information is done based on dedicated systems, thus preventing from flexible transmission in newer installations. The work is organized by IETF RTCWeb WG, supported by collaboration with W3C. There are many tasks declared by RTCWeb WG, such as:
- definition of communication model with special emphasis on session management,
- definition of security goals, with mechanisms of security protocol,
- definition of solutions for firewall and NAT traversal,
- definition of necessary support in client system for media-oriented functions and extensions, such as media formats and codecs,
- transmission of non-media information among browsers etc.

The outcome of IETF RTCWeb WG is summarized in basic RFCs defining WebRTC use cases [6], or reporting used audio codecs, or video codecs but significant work is still ongoing and documented in working drafts. Brief description of WebRTC features and architecture can be found in [7-9].

3. Asterisk

Asterisk is an open source software-based PBX (Private Branch Exchange) allowing to create a coherent telephony system able to connect traditional PSTN (Public Switched Telephone Network) and modern VoIP (Voice over IP) systems [1, 4]. It allows to flexible configuration of any incoming or outgoing call. Asterisk gives the user possibility to create both, simple (supporting only basic functionality) and very complex (e.g. integrating many environments) telephony systems. Its innovation resulted a very large increase in the interest of software-based PBX, originally developed by Mark Spencer, and next extended by the open source development community. According to the information provided on the project web page each year arrives about 1.3 million of new users [3].
3.1. Asterisk architecture

Asterisk architecture is far away from the traditional PBX. Unlike in the traditional PBX, where exist a logical difference between channels created between end-user and PBX and channels interconnecting different PBXes (trunk) the Asterisk’s dialplan treats all channels in the same way [4]. Moreover, there is no distinction among links. There are only different type of channels but they are handled by dialplan in the same way, e.g. end-user may be connected by external link (e.g. using cellular network) but will be treated as directly connected.

Asterisk architecture consists of modules. Single module is a component that allows to execute programmed functions, e.g. support of various channels, support of external communication, etc. The PBX can be started without any module, but in such case it will not provide any functionality. Modules can be loaded during boot process of PBX or can be started when Asterisk works from the CLI (Command Line Interface). Thanks the fact that functions are gathered in groups (modules) the entire system is more transparent.

3.2. Dialplan

All channels in the system are handled in accordance with a predetermined logic. The dialplan is a heart of the Asterisk containing a script-based flow of all calls [2]. It allows to interpret the behavior of PBX in a given situation. There are thee methods of writing above mentioned dialplan: using traditional syntax as contained in the file /etc/asterisk/extensions.conf, using an extended Asterisk logic (AEL — Asterisk Extension Logic) located in the file /etc/asterisk/extensions.ael, benefiting from a scripting programming language LUA presented in the file /etc/asterisk/extensions.lua.

4. Implementation and configuration aspects

In this section we present network topology and tools used during experiments in configuration of voice system SIP/WebRTC.

4.1. Network topology

During laboratory classes students should get knowledge how to use, deploy and implement both classical SIP based solution and novel WebRTC. Laboratory classes will be divided into three parts. The first one is devoted to the SIP protocol and all components they
made a SIP system. The second one is dedicated to WebRTC technology. The last part of laboratory classes is devoted to cooperation of both systems.

For all parts of laboratory classes students will set up the individual components of analyzed system and build the network topology using various network devices. In the laboratory a common topology described in Fig. 1 is used. In the first part students setup and connect PC with software VoIP telephones (Linphone, X-Lite) and PC with Asterisk software-based PBX. In the second students setup and connect PC with web browsers (Firefox or Chrome) and PCs with web server and signaling server. During the last part of laboratories students connects both solutions with additional software component.

4.2. Asterisk setup

Asterisk is a complex system with many resources. These resources are accessing the file system in a different way. All configuration files in Linux are located in the /etc/asterisk directory. Modules described in previous section are commonly installed (if the path is not changed by the system administrator during installation) in the /usr/lib/asterisk/modules directory. Additionally, Asterisk uses many external resources placed in the /var/lib/asterisk directory. Files often used and changed are located by default in the /var/spool/asterisk directory.

There are three ways to install Asterisk PBX: build it from the sources, install pre-built Asterisk application or install entire OS called AsteriskNOW. AsteriskNOW is preconfigured Linux-based image that can be installed like other operating systems [5]. It contains all needed drivers and packages. AsteriskNOW is recommended to use on virtual machines.

During laboratory exercises students learn how to build Asterisk from the sources on Linux Debian 7. This way gives them the greatest ability to accurately get the knowledge
about Asterisk architecture. Moreover, process of modules’ selections gives them more insight into structure of files and modules.

Aster installation all configuration files are located in the /etc/asterisk directory. In order to make first call using SIP protocol only two of them have to be edited: the sip.conf file – containing basic configuration of groups of users and users belonging to them and the extensions.conf file containing the dialplan. Sample configuration of the sip.conf file is presented in listing:

```
[general]
dtmfmode = rfc2833
bindport = 5060
bindaddr = 0.0.0.0
tcpbindaddr = 0.0.0.0
tcpenable = yes
dtmfmode = rfc2833
disallow = all
allow = ulaw
transport = udp
[1001](group1)
callerid = OneOne <1001>
[1002](group1)
callerid = OneTwo <1002>
[1003](group1)
callerid = OneThree <1003>
```

4.3. Softphone installation and configuration

In order to make a call softphone supporting SIP protocol have to be installed. We selected X-Lite application that can be installed on Windows OS. Sample configuration of the X-Lite softphone is shown in the Fig. 2.

![Fig. 2. Example X-Lite configuration](image)

Rys. 2. Przykładowa konfiguracja telefonu X-Lite
4.4. WebRTC

As a WebRTC terminal can be used Mozilla Firefox, Google Chrome or Microsoft Edge web browser. Good side of the WebRTC is that solution not required any plugin to the web browsers. Therefore users can just simply open web page which contains code (in JavaScript) calling WebRTC functions and use it.

Web pages with code for WebRTC is loaded from simple web server. As a web server in laboratory we use Node.js. It is asynchronous event driven JavaScript runtime which allows to run on the server side a network application. Node.js is also used as a signaling server which running software with implementation of the signaling subsystem.

4.5. Integration of SIP and WebRTC

SIP and WebRTC has the different method to exchange SDP signaling messages. Therefore is necessary to create a signaling gateway which converts signaling messages between WebRTC and SIP systems. In practice, the gateway is formed of two components. One of them is a SIP client, runs on a browser, which converts SDP (Session Description Protocol) messages from the WebRTC system to SIP format and sends it via the WebSocket interface. The second component, running on middleware, converts messages received via WebSocket interface to standard SIP messages and sent it using UDP or TCP transport protocols.

5. Results

The most important point of part of the laboratory dedicated to SIP protocol analysis and configuration is traffic observation. It requires cooperation of three students. One of them is an Asterisk administrator and implements PBX configuration. The other two are PBX’s subscribers that test four scenarios: subscriber registration, successful call establishment between subscriber A and subscriber B, and unsuccessful call establishment between subscriber A and subscriber B (call rejected by subscriber B). Students capture packets passing PBX using Wireshark or tcpdump application and next draw flow graphs presented in the Fig. 3.

Moreover, Wireshark application provides nice tool able to convert captured packets into audio signal. It shows how easy is to eavesdrop VoIP call in local network. Thanks that discussion about security issues can be opened.
During tests of the WebRTC system a signaling messages between web browsers can be observed using Web Console (Web Console is integrated with Mozilla Forefox, Google Chrome or Microsoft Edge web browsers). During integration tests of the WebRTC and the SIP systems a traffic between a web browser and the SIP gateway and the SIP gateway and a SIP terminal can be observed using Wireshark. Example results from integration tests is visible in the Fig. 4.
6. Conclusions

In this paper we presented deployment and implementation of VoIP system using SIP and WebRTC protocols and Asterisk as a software switching module. Besides documenting formal description of architecture components and functions there are also illustrated efforts to make clearly visible all necessary settings and flows of messages. This is educationally important attitude in order to present all implementation aspects of VoIP system.

As a future work we plan to investigate the effectiveness of different codecs and further cooperation with MPLS protocol and SDN concept.

BIBLIOGRAPHY

Omówienie

Sygnalizacja w heterogenicznym środowisku, w którym integrowane są systemy wykorzystujące protokół SIP (Session Initiation Protocol) oraz technologię WebRTC (ang. Web Real-Time Communications) [6, 7] jest ważnym zagadnieniem. Artykuł prezentuje kompletny system komunikacji głosowej wykorzystujący zarówno programową centralę Asterisk, jak i jednocześnie dwa systemy sygnalizacji: klasyczny SIP i nowy stosowany w technologii WebRTC. Jako terminali użyto telefonów programowych X-Lite.

Przedstawione na rys. 1 rozwiązanie wykorzystuje jednocześnie wiele funkcji systemów, pozwalając na wyeksponowanie istotnych z punktu widzenia dydaktyki aspektów budowy systemów VoIP. Zamieszczone zostały przykładowe konfiguracje systemów: X-Lite (na rys. 2), a na umieszczonym dalej listingu – modułu SIP centrali Asterisk.

W artykule przedstawiono, w jaki sposób integrowane są rozwiązania implementujące technologie WebRTC z rozwiązaniami wykorzystującymi protokół SIP.

Zamieszczone wyniki wskazują przepływ komunikatów pomiędzy systemami w różnych sytuacjach (rys. 3). Uwzględniono zarówno poprawne połączenie (rys. 3b), jak i odrzucenie połączenia (rys. 3c). W przypadku integracji WebRTC z rozwiązaniami korzystającymi z protokołu SIP wskazano, w jaki sposób są one enkapsułowane w WebSocket (rys. 4).

Addresses

Robert Ryszard CHODOREK, Grzegorz RZYM, Krzysztof WAJDA: AGH Akademia Górniczo-Hutnicza, Katedra Telekomunikacji, al. A. Mickiewicza 30, 30-059 Kraków, chodorek@agh.edu.pl, rzym@agh.edu.pl, wajda@kt.agh.edu.pl
Agnieszka CHODOREK: Politechnika Świętokrzyska, Katedra Systemów Informatycznych, al. Tysiąclecia Państwa Polskiego 7, 25-314 Kielce, a.chodorek@tu.kielce.pl