# Identity Management in WebRTC domains

J. Murányi and I. Kotuliak*
* Slovak University of Technology in Bratislava, Bratislava, Slovakia
jan.muranyi@stuba.sk, ivan.kotuliak@stuba.sk

***Abstract*—In this paper we present possibilities of development in the field of web browsers in interconnection with existing 3GPP architecture. We utilized one of the most discussed web browser technology nowadays WebRTC in a way so it can supplement existing and proprietary Flash based technologies. To bring the customer's quality of experience even further we interconnected this system with a Single Sign On mechanism. System like this can simulate the usage of our work in the 3GPP IMS (IP Multimedia Subsystem) networks, where Single Sign On mechanism can be easily implemented into Generic Bootstrapping Architecture. Our streaming platform then acts as a Network Application Function. For this purpose we used OpenId standard. The motivation for this work is to show the potential of modern browser technologies in an existing 3GPP network. In this paper we discuss this interconnection and show the WebRTC streaming can be a working alternative to existing Flash based systems.**

## I. INTRODUCTION

Multimedia over Internet protocol has developed as a mainstream platform. According to [1] a demand on multimedia content is constantly growing. As [2] shows the main system used nowadays to deliver multimedia content is Adobe Flash Platform. This platform can be installed into every major browser and therefore its integration is very widespread. This platform is however proprietary. As an opened community does not develop this platform, developers started to look for alternatives. Now W3C and IEEE introduced a WebRTC to overcome the need of additional applications, modules or extensions.

IMS by 3GPP was introduced as a new architecture for offering telecommunication services. Soon this architecture become more complex and many new modules were introduced to satisfy users demand. 3GPP adopted SIP as a signaling protocol, but the demand of HTTP based services brought the need of new entity. As the security and authentication is essential 3GPP introduced in [3] GBA (Generic Bootstrapping Architecture). This allowed to use HTTP based services in the IMS network.

In this paper we simulate the interconnection between browser based multimedia content delivery and telecommunication architecture IP Multimedia Subsystem using Single Sing On application. The WebRTC technology is used as a gateway between web based services and SIP-based environment.

The rest of the paper is organized as follows: In Section 2 we present the State of Art. Section 3 proposes our solution. Possible IMS interconnection is described in section 4. Session 5 discusses proposed test-beds. Concluding remarks and ideas for a future work are given in the Section 6.

## II. STATE OF ART

### A. RTMP

Nowadays for streaming videos the Adobe Flash Media Server and Flash Player web browser plugin has to be used.

"The Real-Time Messaging Protocol (RTMP) was designed for high-performance transmission of audio, video, and data between Adobe Flash Platform technologies, including Adobe Flash Player and Adobe AIR" [4]. RTMP is used to demiltuplex and multiplex to "chunks" [5], packetize and deliver them using lower protocols.

RTMP was designed to deliver real-time data such video, however it is possible to deliver any other information. Such example can be collaborating learning where there are many possibilities for screen sharing, multiple video conferencing and file sharing [6]. These streams are demultiplexed to chunks and sent to receiver where they are again multiplexed. The size of these chunks is a matter of negotiation at the initial handshake.

RTMP protocol can be streamed directly from the streaming server over the TCP on port 1935. However in many cases such in hotels, free or public wireless hotspots routers are protected by a firewall, which is often limited only to basic protocols. As most of SOHO firewalls implement port based filtering the set of ports does not contain more than ports 80 and 443. In such cases RTMP protocol can be encapsulated into HTTP protocol. Solutions offered by Adobe systems contains mechanisms for NAPT and firewall connection checks and in case firewall blocks a connection it automatically fallbacks to the HTTP encapsulation [7].

RTMP protocol was created by Macromedia, which was later acquired by Adobe Systems. This proprietary protocol is built to work with Adobe Flash Platform. Generally user has two possibilities to access a Flash content on the Internet. The only prerequisite is to have installed local flash player in a user's browser.

First option for a service provider is to directly share a flash multimedia content as Flash Video (FLV) or ShockWave Flash (SWF) on a website. A user downloads this content using HTTP and plays it locally. No RTMP protocol is used as all data is delivered as any other file via HTTP. This approach lacks the control over the content by a provider, as it allows user to manipulate it locally. Another drawback arises when a user wants to watch long video content. In such case when a user wants to watch only last few minutes he has to wait till the whole file will be downloaded. An advantage of this approach lies in easiness of the implementation and management.

978-1-4799-2162-1/13/$31.00 ©2013 IEEE

When a service provider requires the content control, for legal or lawful reasons, or wants to offer better Quality of Experience (QoE) for users [8], the RTMP protocol is needed. A service provider shares the SWF file on a web page. A user afterwards accesses this file. At first SFW player creates a NetConnection instance. The actual content of the video is now delivered by RTMP from Flash Media Server (FMS) located somewhere in the service provider network [9]. A user is now able to control an incoming stream by set of commands. Advantages of using RTMP are the control over the content, "near-instantaneous playback of video" [9], and possibility to view any part of the multimedia stream. However FMS is licensed and expensive to acquire for small providers. For this reason a solution using RTMP might be far more complex to implement.

### *B. OpenID*

OpenId is a technology that allows users to use one profile in multiple services. This technology is decentralized. Multiple identity providers can offer authentication and authorization for multiple service providers known as relaying parties. The user can choose any identity provider to connect to a relaying party. During the information exchange relaying party may receive many attributes like name, age or any personal information.

The communication during an authorization and an authentication process is between the user and the identity provider. As depicted on Fig. 1 the relaying party only offers a shared secret to the identity provider, which is after user's successful login accessed to the user, who can login into a relaying party's service.

### *C. WebRTC*

WebRTC (Web Real Time Communication) is the joined initiative by World Wide Web Consortium (W3C) and Internet Engineering Task Force (IETF) to define API and protocols to enable deploying of real-time applications in web browsers. Whole WebRTC mechanism is divided into two parts. First one is the protocol suite definition, codec dealing, security problems, etc. Second part of the effort is the API. The API deals with a direct browsers access to the RTP stack, brings the functions and methods for implementing the solutions, access the media information, etc. The first part of the solution is being discussed at IETF, while W3C deals with API.

The aim of WebRTC is to bring RTP multimedia sessions into web browsers, without a need of any extensions, plugins or addons. WebRTC is rapidly growing as almost every major browser developer is contributing to the project.

WebRTC defines API and functions to enable end-to-end multimedia communication. Fig. 3 depicts a browser model of WebRTC. This model defines a Browser Real-Time Communication Function, which is a part of each WebRTC enabled browser. Web developer can directly control this function through RTC API using JavaScript functions. These functions are defined by W3C and currently variously implemented by vendors.

Nowadays there are several signaling protocols used. Most widely spread protocol is SIP [10]. Even 3GPP has adopted this protocol as a signaling protocol to IP Multimedia Subsystem (IMS) in [11].

At the beginning two protocols were proposed for WebRTC. Both Extensible Messaging and Presence Protocol (XMPP) with Jingle extension and SIP are widely used on the opened Internet. However none of these were standardized into WebRTC. As there is no signalization defined in the matter of interconnection, this part of the communication has to be implemented directly by web developers. Fig. 5 depicts the WebRTC trapezoid of signaling and media paths. As shown the communication between browsers and web servers is left for developers. Also the signaling path between web servers is undefined. As main use cases of the WebRTC are not the standard telephony between two consecutive users, standardization of this path would make implementations more complex. In cases where two service providers would like to interconnect their services the signaling protocol have to be dealt on demand. This can be any proprietary or well-known protocol.

The multimedia negotiation is done using SDP encapsulated in JavaScript Object Notation (JSON) object. This information is encapsulated in signaling protocol and delivered to consecutive endpoint.

Another important part in the media negotiation is the ICE (Interactive Connectivity Establishment) technique. It solves NAPT related issues and security problems, mainly the DoS (Denial Of Service) attacks that might be very spread in the Internet environment. An attacker can place a malicious code into his webpage and every user's browser can create an RTP session to a victim's user agent. This can be preceded by ICE, where both endpoints have to at first agree on the session.

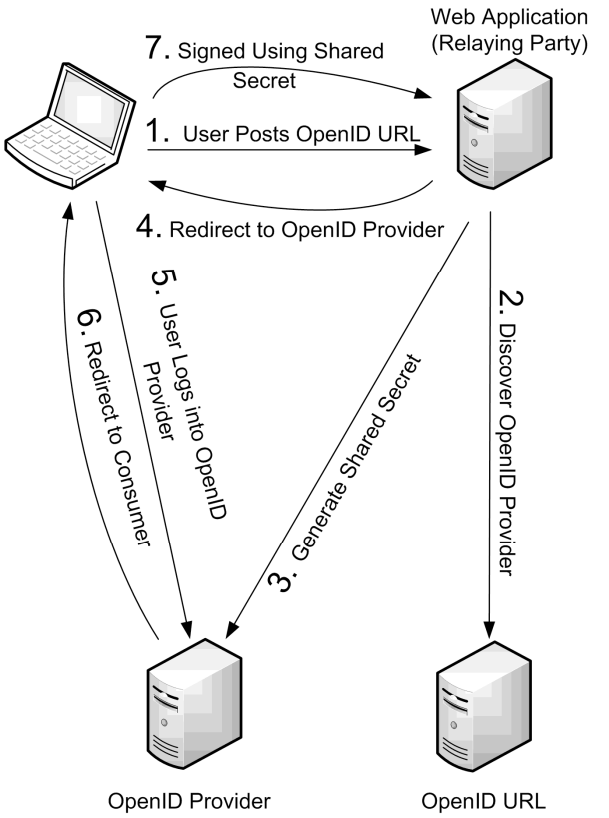


Figure 1. Communication flow during the OpenId authentication

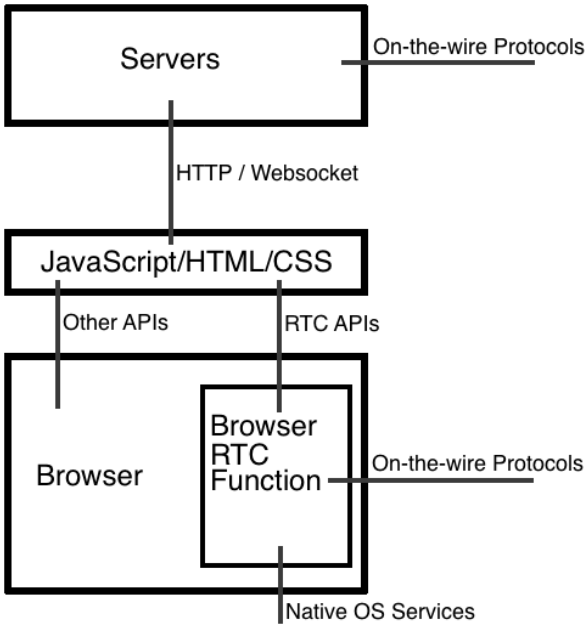


Figure 3. Browser model of WebRTC with interfaces

## III. SOLUTION DESCRIPTION

In our solution we used WebRTC to receive an RTP stream from the service provider directly into a browser. The most used solution in todays Internet is the Flash platform [2] with media delivery using RTMP protocol. As opening RTP stack in browsers is the main aim of the WebRTC initiative we will use this protocol to deliver multimedia to users. Ref. [12] states "the overhead is approximately the same for both transports and accounts for about 9 percent of the total bandwidth, both for HTTP/TCP and the RTP/UDP." According to [2] the overhead in multiple multimedia protocols such HTTP or RTMP might be significant. This brings a space for the WebRTC to utilize the bandwidth between service provider and a user.

OpenId mechanism was implemented using Passport framework in the Node.js environment. During the first login the system records the user's username. For every other login this user is identified using this OpenId identifier. All accounting is being done only based on this identifier.

OpenId allows us to offer such services to any user with account at any identity provider. This allowed us to perform AAA for all users. Mainly billing becomes easier. The main reason to use OpenId is its possibility to interconnect it with Generic Authentication Architecture (GAA) as defined by 3GPP in [3]. This interconnection is described in [13]. To the best of our knowledge no interconnection in a way this paper describes has been done yet. This brings a possibility to subscribe to services without any need of login, because based on [14] the network would be able to provide the user's identity based on its connection to the IMS network. This would ease the service usability for users and together with WebRTC would offer service providers a new way to implement services based on a web and multimedia content.

As depicted on the Fig. 6 we have built a platform to offer streaming multimedia content for WebRTC enabled browsers and OpenId as authentication and authorization

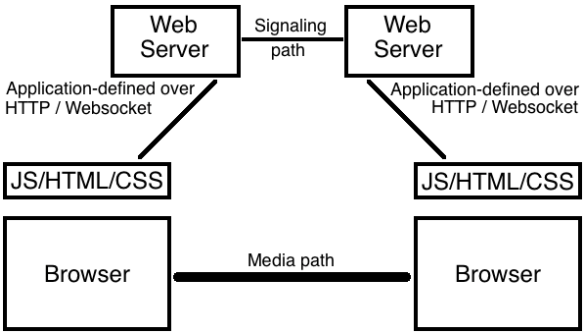


Figure 5. Signaling and media paths in WebRTC trapezoid

backend to simulate the interconnection with IMS network.

To successfully access the multimedia content provided by the platform client has to receive it using an RTP stream. After a client opens a requested content, an user's browser calls a getUserMedia function. This function reads and opens browser's multimedia capabilities. To receive an RTP stream the browser has to propagate his multimedia capabilities. The browser creates this description using a PeerConnection function. The result is the session description in the Session description protocol encapsulated in an JSON object. This object is prepared to be delivered to the consecutive endpoint as an SDP answer. During a page load user does not just downloads the source code of the webpage and calls mentioned WebRTC functions, but also connects to WebRTC server using a websocket. This creates a both way communication channel. The WebRTC server contacts a streaming server chosen by an algorithm for load sharing. The WebRTC server using similar functions as a client creates an SDP offer based on the available resources of the chosen streaming server. This offer is stored in a JSON object. Created JSON object is sent using the websocket from the WebRTC server to the client. The client processes the offer and sends back the SDP answer. The SDP answer is delivered to the WebRTC server using the same path the SDP offer came from to the browser. As the ICE technique is mandatory both endpoints creates their ICE candidate pairs and afterwards perform checks to see which pairs are valid. If there is any valid pair, the server picks the pair with the highest priority and starts to send the RTP stream. The browser displays this stream as an incoming video. The browser does not send any video to the server expect the regular RTCP datagrams to avoid dropping the session.

The streaming server is based on the ffmpeg library and is used to demultiplex and decode the multimedia files into RTP stream. Multimedia files are stored directly on the streaming servers and decoded during a streaming. The WebRTC server due to interconnection problems and ongoing development of the WebRTC standard was implemented as our own C++ application. This application uses libjingle library to communicate with a WebRTC client via websockets. This application simulates the consecutive endpoint for the real time communication between server and user's browser.

The communication between browser and WebRTC server is handled using websockets. This allows us to asynchronously communicate with browser. Many task in the session establishment like session information negotiation or ICE candidates exchange are eased.

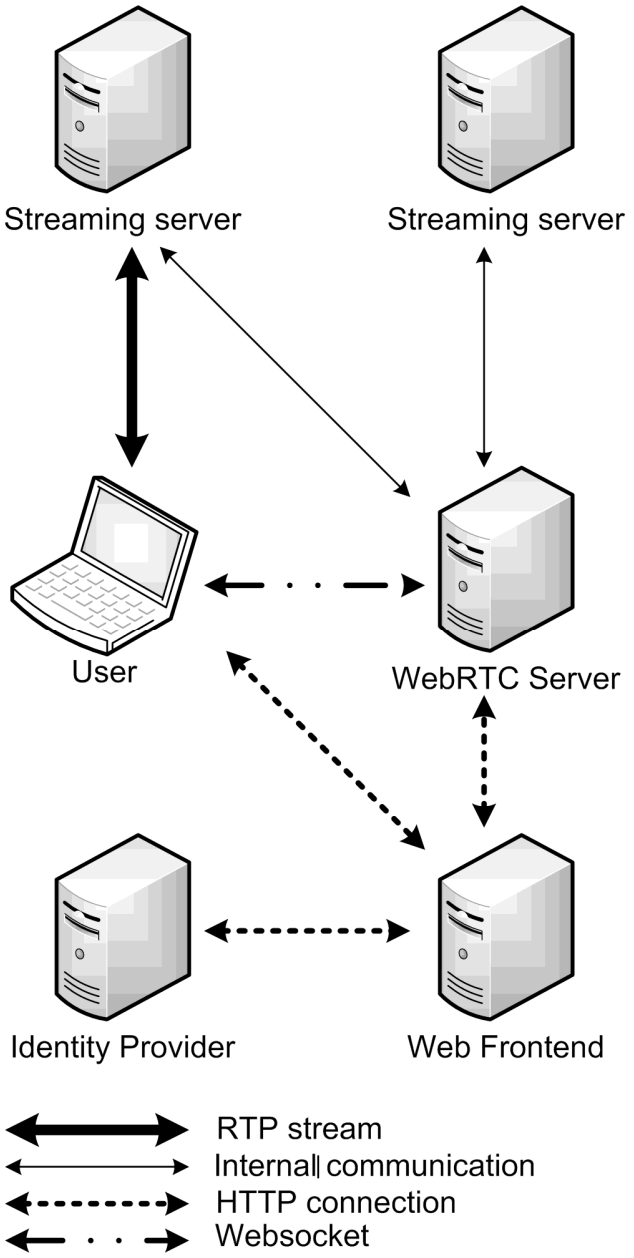


Figure 6. Architecture for RTP streaming using WebRTC

## IV. IMS INTERCONNECTION

As mentioned the reason we used OpenId mechanism in this work is to simulate the implementation in the IMS network. Ref. [13] states "SIP Digest is arguably the most commonly used authentication mechanism in current IMS deployments." Based on this statement we can assume the potential usage Single Sign On mechanism can become widespread, as the authentication is not based on USIM (Universal Subscriber Identity Module) but on a standard SIP credentials. Together with WebRTC this allows service providers to offer new services based on an existing accounting and authorization. Users will not be asked to register to every new service. They will simply be connected to the service right after accessing it. The identity of the user will be provided during a registration into the network. As any Single Sign On mechanism can authorize user during an existing registration in the network, during this period user will be able to provide his identity to any service.

3GPP defines Generic Bootstrapping Architecture (GBA). GBA defines possible authentication process of the user to the IMS network. When a user wants to use a service provided by the network, he does not have to directly authenticate with the service. In GBA user authenticate with BSF (Bootstrapping Server Function). Authentication is done using AKA (Authentication and Key Agreement Protocol). During this process both sides creates derivations of keys used in this process. These keys can now be used for any service the operator offers. Generally they are used to connect via Ua reference point to NAF (Network Application Function). NAF can be any server offering services. To retrieve key derivations NAF uses Zn reference point to communicate with BSF. This reference point is using Diameter protocol.

Any Single Sign On mechanism can simulate bootstrapping as described. To prove the usability of WebRTC technology in an IMS network we used OpenId. In this case our web frontend, WebRTC server and streaming servers act as NAF.

## V. TEST-BEDS

To verify the functionality and usability of our system we prepared several test-beds. In our testing topology we used web browser Google Chrome 29.0. on Mac OS X operating system. Google Chrome has WebRTC functionality available in a stable version and no other configuration is needed to access its tools. To verify our approach we need to test OpenId authentication and multimedia streaming. In a first use case scenario the user logs into our system using existing OpenId identification. The identity provider in our case is myopenid.com. After a successful login the system creates a new account for the user based on his unique OpenId identity. This information is sufficient to create a user in our system. To access other services the user would be required to fill his billing information so the accounting would be reliable. As OpenId allows us to query for other available information about a user, this information might be acquired from the identity provider. In the IMS network the identity provider would be an operator, hence this information would be available and a user would be able to use the service without providing any information. This scenario shows the working concept of the Single Sign On authentication.

In the second use case scenario we access the multimedia content of the service provider. In this scenario user sits behind NAPT (Network Address and Port Translator) router. After successful login, when a service provider has all required information, the user is allowed to access the multimedia content. Using a web browser the user selects his favorite movie. After opening the movie, client's browser creates a websocket to the WebRTC server and asks client whether he wants to open a WebRTC session. This restriction is built in Google Chrome browser to enhance the security of WebRTC. If the user agrees, user's client contacts the STUN (Session Traversal Utilities for NAT) server to get his external IP address. This IP address is one of the browser's candidates. Both sides exchange ICE candidates and

perform all required checks. After the correct pair is selected both client and server are prepared for the multimedia session. All these actions are done in a background without any interaction from the user's side. The streaming starts right after. The client is now able to watch a requested video. The RTP and RTCP streams flow from the IP address of the streaming server to the IP address of the browser. On both sides are two ports used for the communication, as RTCP requires its own port. According to the [15], these ports should be odd and even following one by another. From the client's IP address only RTCP packets are being sent. If the network between client's browser and a streaming server is stable, the quality of the video is very good. However due to the RTP nature, the stream might be sensitive to jitter and packet loss. Choosing a right multimedia codec with sufficient FEC (Forward Error Correction) might improve the quality.

This test-bed has shown the streaming the multimedia content using RTP streams into web browsers is possible. No issues were observed, not even when the stream passed through NAPT router.

## VI. CONCLUSION

The aim of this paper is to show the potential of WebRTC and possibilities in its interconnection with other systems. We have created a streaming platform for delivering RTP streams directly into web browsers without any need of browser's extensions or special programs. The multimedia session was established even though a user's browser was behind a NAPT device.

We simulated the integration of this system into the IMS network where we used Single Sign On authentication as GBA. The Single Sign On mechanism simplified accounting, authentication and user registration into the service. We have shown that user experience we observed in our system can be easily introduced into the 3GPP network.

Both proposed test-beds showed the validity of the presented concept and verified its usability in a real network environment. We have shown the RTP protocol and WebRTC can be a reliable alternative to proprietary Flash based platforms.

For the future work we would like to implement Bootstrapping Server Function into an existing Open IMS core implementation and verify our concept directly with bootstrapping process.

## ACKNOWLEDGMENT

This work has been supported by the Slovak National Grant agency under the number 1/0676/12, 7FP project HBB- Next and ITMS 26240120029.

## REFERENCES

[1] M. Li, M. Claypook, R. Kinicki, and J. Nichols, "Characteris- tics of streaming media stored on the web," ACM Transactions on Internet Technology, vol. 5, no. 4, pp. 601–626, November 2005.

[2] McAlarney, J.; Haddad, R.; McGarry, M.P.; , "Modeling Network Protocol Overhead for Video," Computer Modeling and Simulation (EMS), 2010 Fourth UKSim European Symposium on , vol., no., pp.375-380, 17-19 Nov. 2010

[3] 3GPP TS 33.220 V11.3.0 (2012-06), Technical Specification, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA), (Release 11)

[4] Real-Time Messaging Protocol (RTMP) specification | Adobe Developer Connection. Available: http://www.adobe.com/devnet/rtmp.html [online on 17th of May 2012]

[5] Real Time Messaging Protocol Chunk Stream | Adobe Developer Connection. Available: http://www.adobe.com/devnet/rtmp.html [online on 17th of May 2012]

[6] Premchaiswadi, W.; Tungkasthan, A.; Jongsawat, N.; , "Enhancing learning systems by using virtual interactive classrooms and web-based collaborative work," Education Engineering (EDUCON), 2010 IEEE , vol., no., pp.1531-1537, 14-16 April 2010

[7] Flash Media Rights Management Server Help | Ports and firewalls | Flash Media Server. Available: http://helpx.adobe.com/flash-media-server/kb/ports-firewalls-flash-media-server.html [online at 18th of May 2012]

[8] French, H.; Jie Lin; Tung Phan; Dalal, A.C.; , "Real time video QoE analysis of RTMP streams," Performance Computing and Communications Conference (IPCCC), 2011 IEEE 30th International , vol., no., pp.1-2, 17-19 Nov. 2011

[9] Reinhardt, J.: Beginner's Guide to Distributing Flash Video. Available: http://www.adobepress.com/articles/article.asp?p=1014968&seqNum=2 [online on 19 of May 2012]

[10] Rosenberg, J., Schooler, E.: SIP: Session Initiation Protocol. RFC 3261 (2002)

[11] 3GPP TS 23.002 V11.2.0 (2012-03), Technical Specification, 3rd Generation Partnership Project; Technical Specification Group Services and Systems Aspects; Network architecture, (Release 11)

[12] M. Johanson, "An rtp to http video gateway," in WWW '01: Proceedings of the 10th international conference on World Wide Web, 2001, pp. 499–503

[13] 3GPP TR 33.804 V12.0.0 (2013-06), Technical Report, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Single Sign On (SSO) application security for Common IP Multimedia Subsystem (IMS) based on Session Initiation Protocol (SIP) Digest, (Release 12)

[14] 3GPP TR 33.924 V11.0.0 (2012-09), Technical Report, 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Identity management and 3GPP security interworking; Identity management and Generic Authentication Architecture (GAA) interworking, (Release 11)

[15] Schulzrinne, H. et al.: RTP: A Transport Protocol for Real-Time Applications, RFC 3550 (2003)