INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2012, VOL. 58, NO. 1, PP. 77–83
Manuscript received January 15, 2012; revised March 2012. DOI: 10.2478/v10177-012-0011-9

A Hybrid Extension for IP Multicasting in Push-to-talk over Cellular Systems

Tomasz Gębarowski

***Abstract*—OMA Push-to-talk over Cellular (abbr. PoC or PTT) as an IP-based service can be considered the first Voice-over-IP technology delivered in a broad manner in cellular networks world wide. Two key differences between PoC and regular voice communication systems are its half-duplex nature, which converts it into 'walkie-talkie' style of service and ability to establish group sessions between large groups of participants. As a matter of fact, majority of available PoC solutions are inefficient in handling groups of large size because they rely on IP unicasting for delivering media streams. This introduces not only redundancy and greater packet loss, but also increases computational and transmission effort.**

This paper proposes a hybrid IP multicasting extension for Push-to-talk technology allowing to reduce group size limitation significantly and increase the performance of large scale Push-to-talk systems. The proposed enhancement is elaborated and verified basing on an automatic testing framework feedback and some statistical measurements.

***Keywords*—Push-to-talk, PoC, OMA PoC, PTT, IMS, SIP, RTP, IP Multicasting, VoIP.**

I. Introduction and Related Work

In early 2000's Nokia has released its first terminal supporting proprietary version of Push-to-talk technology, based on SIP architecture. Since that time other mobile phone vendors such as Motorola and Ericsson started working on integrating their phones with similar but also proprietary VoIP based solutions. Due to incompatibility issues and fragmentation of the market none of the standards gained full acceptance of the customers. Therefore, in the late 2003 Nokia, Ericsson, Motorola and Siemens joined their forces to create Push-to-talk over Cellular standard specifications. This set of definitions was the core and main contribution to *OMA PoC Release 1* standardization effort ([1], [2], [3], [4], [5]), finally approved by *Open Mobile Alliance* (OMA) in 2006.

Nowadays Push-to-talk is a rather niche part of the VoIP market applicable mainly locally in enterprise sector. Fortunately due to previously mentioned standardization effort, smaller market players started releasing their own OMA PoC Release 1 compliant solutions, which may be easily integrated with already existing IMS infrastructure and which provide advanced features such as geo-location and presence information sharing. This newly defined features match perfectly the needs of taxi companies, building, military, transportation and logistic sectors. For large scale commercial deployments reliability, maintenance and quality of service are significant. Certain applications require communication with large groups of participants who are listening to the same content. With majority of currently available VoIP solutions (using unicast packet routing) this is not easily achievable. On the other hand for typical large scale business applications PoC is usually deployed using a dedicated mobile operator Access Point Name (APN) directly connected to enterprise network infrastructure, thus usage of IP multicasting is worth considering and achievable.

Despite the fact that Push-to-talk over Cellular is not a widely used technology, there are publications elaborating PoC related subjects in more details. Most of the papers [6], [7] focused on investigating service deployment within IP Multimedia Subsystem infrastructure. However, from the perspective of this paper the major concern is given to articles focusing on possible enhancements of Push-to-talk over Cellular technology. There are various approaches to the subject, some papers like [8] concentrate on a network level and efficient management of call setup time, others like [9] try to enhance floor control protocol. An interesting approach for reducing Push-to-talk floor granted response time has been presented in [10], which proposed simplifying SIP message flow and what is even more interesting used a concept of a Snoop Agent to transparently speed up session build-up without violating existing PoC architecture. Finally [11] discussed Quality of Service aspects of PoC with respect to MOS (Mean Opinion Score), PoC system design and experimentally obtained jitter, latency and packet loss values. From the point of view of all available papers, this article investigates a new area of possible PoC Release 1 service enhancement. With the proposed IP multicasting improvement it should be feasible not only to reduce packet delivery time but also decrease the bandwidth usage by a significant factor. So far, none of existing commercial Push-to-talk over Cellular solutions are IP multicast enabled. The OMA PoC 1.0 specification itself is not saying anything about multicasting, the version 2.0 defines IP multicast support however there are no available solutions supporting this relatively new and not accepted by market standard. The author, himself managed to extend OMA PoC Release 1.0 based client and server to support IP multicasting and performed numerous simulations. As a result, the paper proposes a complete design for PoC testing infrastructure and delivers experimentally obtained quality feedback comparing both multicast and unicast packet delivery.

The paper has been divided into four parts, in the first one the author introduces key concepts of Push-to-talk technology with emphasis on audio distribution and IP multicasting. The second and third parts are author's original contribution. Respectively, they describe author's aproach towards extending existing PoC systems with IP multicasting and complete application specific procedure for verifying the proposed solution

Tomasz Gębarowski is the PhD student at the Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland (e-mail: T.Gebarowski@stud.elka.pw.edu.pl).

with advanced distributed testing framework. The latter part, presents an overview of simulation results and prospective enhancements.

II. Technical Background

From the logical point of view, PoC system may be divided into control, policy/access and media/user plane. The first two; which are beyond the scope of this article, are responsible for controlling session (with Session Initiation Protocol, abbr. SIP) and managing access control lists and policies (XML Configuration Access Protocol, abbr. XCAP). The latter, the media plane is the matter of this paper elaboration and will be presented in more details in this section.

As mentioned before, due to half-duplex nature of Push-to-talk service, only one user is allowed to talk in the same group at a time. Every voice stream generated by a PoC client is referred as a *talk burst* and is composed of a sequence of RTP packets embedding a given number of AMR (encoded using Adaptive Multi-Rate codec) frames. In order to manage talk bursts and avoid intentional user interruptions, a specific talk burst (or floor) control mechanism is used. It is based on a simple request and grant permission model. Therefore, prior to every talk burst a user has to ask for permission to talk. The controlling PoC function has to ensure that only one user is sending media stream and can correspondingly grant or deny the floor permission. According to [4] floor control is realized independently of the SIP layer, as it is strongly related to media handling routines. Hence, talk burst control is realized together with feedback reporting (with Real-time Transport Control Protocol, abbr. RTCP) through a separate UDP unicast channel created between PoC client and PoC server. On the top of this communication a Talk Burst Control Protocol (abbr. TBCP) is placed. TBCP is based on RTCP Application Packets (abbr. RTCP:APP) as defined in RFC-3550 [12] and it does not conform the rules for compound RTCP packets.

Each talk burst is forwarded as Real-time Transport Protocol (RTP) media stream through a separate UDP unicast channel. The voice stream which is sent through this channel is first fragmented into smaller RTP packets and then wrapped as a payload for UDP/IP protocols. Ports for both control and media channels are negotiated during session establishment (using Session Description Protocol offer-answer model). For each group participant PoC session is identified by an IP address of both entities (client and group) and two pairs of UDP ports:

- PoC Client Remote Media Port
- PoC Client Remote Control Port
- Server allocated Group Media Port
- Server allocated Group Control Port

Figure 1 depicts interconnections between clients belonging to the same group and PoC server media plane. Due to convention media ports have even number value, while control ports are greater by one from corresponding media port.

III. Hybrid Approach towards IP Multicasting in PoC

In majority of Push-to-talk use cases arbitrary PoC session consists of a large number of group participants. Each member receives exactly the same media content, which is distributed by the user currently possessing floor permission. In standard Push-to-talk service implementation user acting as a source of the content, forwards media stream to previously negotiated media port allocated on the server side. The role of the server is to distribute the content using N-unicast UDP channels, where N stands for the number of listeners (see Fig. 1 for more details). This approach is not efficient at all, as the same content is sent in a redundant manner generating excessive amount of traffic in the network. For small scale Push-to-talk systems it may be still acceptable, however in case of hundreds of groups having hundreds of participant the original idea is invalid. In this respect, IP multicasting seems to be more appropriate solution leading to better network usage and allowing for possible reduction of network maintenance costs. The IP multicasting concept itself is much easier to deploy in case of Push-to-talk service, comparing to any other VoIP solutions. First of all, Push-to-talk service is often using dedicated network infrastructure within a given Access Point Name, allowing to avoid IP multicasting problems known from Internet (i.e. avoiding IP multicast address collisions, spare content distribution trees and incompatible network components). Secondly due to mentioned network isolation, all the multicast group participants are always distributed densely and what is more, multicast IP addresses may be selected arbitrarily. Finally due to entire control over the end-to-end PoC network, one may adjust the whole multicasting routing protocols and network entities, so that they always create efficient content distribution trees.

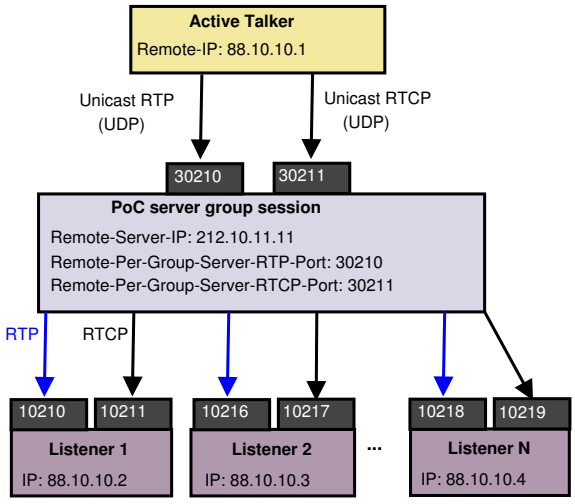

Fig. 1. Original PoC media stream flow.

The concept hidden behind this article combines both multicast and unicast packet distribution channels. This explains the first words of the paper referring to a *hybrid* approach towards IP multicasting in PoC. Generally speaking, in the proposed enhancement IP multicasting is used to distribute all RTP

media packets, while the control flow is still realized using standard unicast channels. From the first point of view, the idea may appear to introduce unnecessary complexity, however after brief explanation the motivation should be clear.

In the proposed approach (as depicted in the Figure 2), each member of a certain PoC session is listening on a standard RTCP interface bound to a publicly available unique IP address. Moreover, it listens on a common port of a general per group multicast interface. It is worth to point out that IP multicasting does not oblige the source (talker) to be a member of a multicast group. This works well when the source activity is rather static and does change its role in the service every minute or even couple of seconds. Unfortunately, such a behavior is impossible to achieve with PoC service. Hence to avoid constantly sent IGMP messages[1], it is assumed that all group participants including current floor owner (talker) are members of the multicast group throughout their activity in the PoC group session.

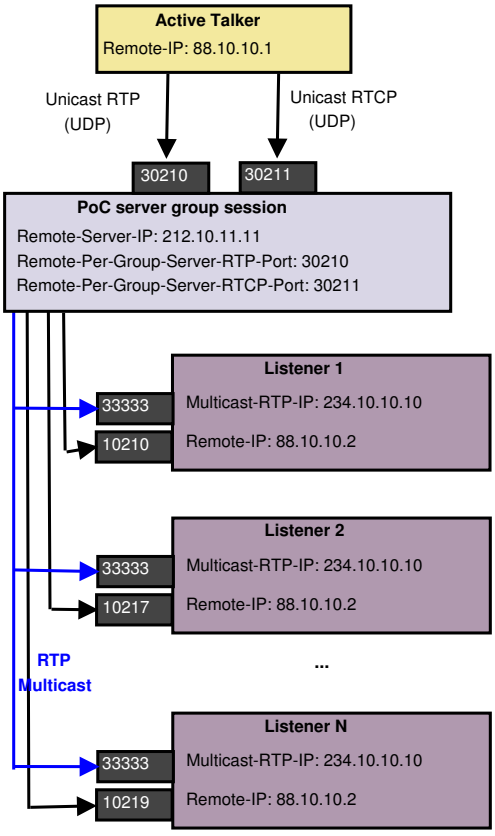

Fig. 2. A Hybrid Multicast approach to PoC stream flow.

The concept described above is not perfect. First of all each source will receive back all its media packets. Therefore it must be taught to ignore those packets having its own origination point. This could be realized by analyzing Synchronization Source (SSRC) field of the incoming RTP packets, but unfortunately is not fully reliable as there may be some SSRC collisions within the same session. Having an SSRC collision is not very probable, but to decrease that probability by a significant factor it should be possible to re-use Contributing Source field (CSRC) by putting there another randomly (per talk burst) generated value. This is an idea which goes beyond the standard, but by comparing these two values it will be almost certain that only RTP packets that has been looped back to the source are ignored. Apart from that, CSRC field is not used in any PoC deployments and should not cause any system malfunctions. The above drawback may be eliminated by using IGMPv3, in which the source may be excluded when joining multicast group. However, this approach has not been selected due to limitation in 3rd party library and testbed itself. From the performance point of view it should not have any impact on the results.

Applying the same concept for RTCP sessions is not that trivial and probably not even worth considering. First of all, RTCP packet has no CSRC field, secondly there are control message flows, which imply distribution to one participant only. Finally, when reasoning in terms of performance, implementing RTCP packet multicasting is not a critical enhancement. In case of PoC technology and probably a great majority of VoIP applications more than 90% of PoC traffic is generated by RTP streams. Hence if there is anything to improve it must be within the RTP packet flow.

As it was mentioned previously in the paper, the usage of IP multicasting is really efficient inside closed operator networks, where all the end terminals are creating a dense distribution tree. Because PoC is mainly deployed within such an infrastructure, this specific architectural feature allows to select IP multicast addresses almost freely. Unfortunately, each PoC client joining a group session should know its IP address before sending initial SIP INVITE message. Therefore, the IP address must be appended to Session Description Protocol (abbr. SDP) offer and should be unique within PoC group session. To provide efficient way of selecting unique IP multicast address for arbitrary PoC chat group, the paper proposes a method described below.

Each PoC client manages a hash table of all discovered multicast PoC sessions, the indices of the table are Group URIs, while corresponding values are multicast IP addresses. Before joining a PoC chat group, the client checks for existence of a Group URI and related multicast IP address inside its own cache (hash table). If the group is already there, the corresponding IP address is used to formulate SDP offer. Otherwise, the joining client selects first not-taken[2] multicast IP address and announces its selection via Session Announcement Protocol (SAP). The SAP message is sent to a predefined multicast group, where all PoC clients are automatically subscribed on registration. When receiving the SAP message each client updates its cache and checks for conflicts. If the conflict occurs it immediately sends invalidating SAP message to notify the original source about incorrect address choice. On reception of invalidating SAP message, the joining clients repeats the selection procedure until it does not discover any conflicts. Note that such conflicts may occur mainly to newly registered PoC clients, when they do not have a cache

[1]Note this does not refer to IGMPv1 where the user cannot explicitly leave the multicast session.

[2]Ideally a hash algorithm should be used to provide efficient multicast IP address mapping with respect to group URIs.

of Group URIs mapped to multicast IP addresses and they need to discover it from scratch.

When per session unique multicast IP address is properly appended to SDP offer embedded in the SIP INVITE, the whole message is forwarded towards PoC server. To confirm successful call establishment SIP 200 OK reply is sent back to the client. On receiving it, the client should start listening on corresponding RTP and RTCP ports and send an IGMP Group Join message to attach to selected multicast group. Similarly, on session termination the IGMP Group Leave message is sent to unsubscribe from the multicast stream.

All the proposed enhancements allowing to introduce IP multicasting in Push-to-talk service were realized as transparently as possible. However, to successfully deploy the solution stated in this paper, certain PoC architecture components needs to be re-adjusted:

- **Push-to-talk client** : Most of the improvements must be realized inside the client itself. First of all the IP multicast address selection mechanism with conflicts resolution should be implemented. Secondly, the SDP offer embedded in the SIP INVITE message must contain the selected IP multicast address. Finally the client should handle and generate IGMP messages to indicate multicast group participation.
- **Push-to-talk server**: Depending on the PoC server internal implementation, some effort may be required to allow for IP multicasting. The PoC server itself should detect the IP multicast addresses of group members and forward all media packets to a single multicast IP address only. Oppositely, the control flow is not to be modified and unicast packet delivery should be used as in the original case.
- **Network Architecture**: Modifying network infrastructure is beyond the scope of this paper, although there are some articles (including [13]), which describe deployment of already existing IP multicasting solutions on the top of UMTS. It is worth to mention that for testing purposes it should be enough to use a local network with WiFi access and corresponding multicasting enabled routers.

IV. Testbed

In order to verify efficiency of the solution proposed in the previous section, a software testbed system was developed. Its main task is to generate PoC traffic according to a predefined scenario and collect feedback from all the PoC clients. As depicted in the Figure 4, the testbed is composed of two functional elements:

- **Many Push-to-talk client simulation processes** : Each working as a standalone process running a predefined number of PoC clients. Individual client is able to perform simulation according to its own execution scenario
- **Testbed Controller**: A single controlling process, detecting new PoC clients, distributing tasks and collecting their feedback

All the testbed elements are located inside an isolated multicast enabled network (see Fig. 3), due to dense distribution of multicast receivers PIM-DM protocol has been chosen for multicast routing. Despite rather simple network design, several distribution trees can be found within the network.

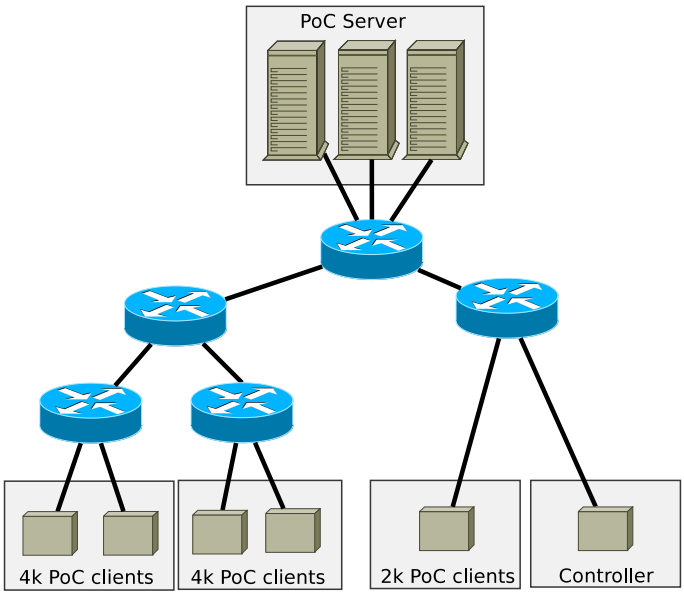

Fig. 3. Multicast-enabled network design for testbed purposes.

The proposed architecture is fully distributed, simulation processes are independent of each other and are connected to the controller through a two way TCP-based communication channel. Therefore, it is possible to scale the whole testbed solution among almost arbitrary number of nodes, allowing to generate the traffic from tens of thousands of simultaneously working PoC clients.

The testbed design allows to automate the whole simulation procedure, so that even largely distributed and most complex scenarios can be realized easily. All of that is achieved with the control unit, which keeps track of PoC client process activity and facilitates their work. The simulation procedure itself is divided into five phases:

- **Test Scenario Build-up** – The controller loads a generic XML configuration template specifying step by step behavior of PoC clients. The content of the template may be randomized so that each client gets different simulation scenario
- **Detecting PoC process availability** – Each PoC client process sends a periodic keepalive message towards the controller. The controller keeps track of that messages and knows where the client is located and what its current status is
- **Configuration Preloading** – The controller loads the scenario into each PoC client and initializes its settings
- **Simulation Start-up** – The controller initiates simulation procedure in every PoC client
- **Simulation phase** – Individual PoC clients are working independently, generating their own statistics and sending

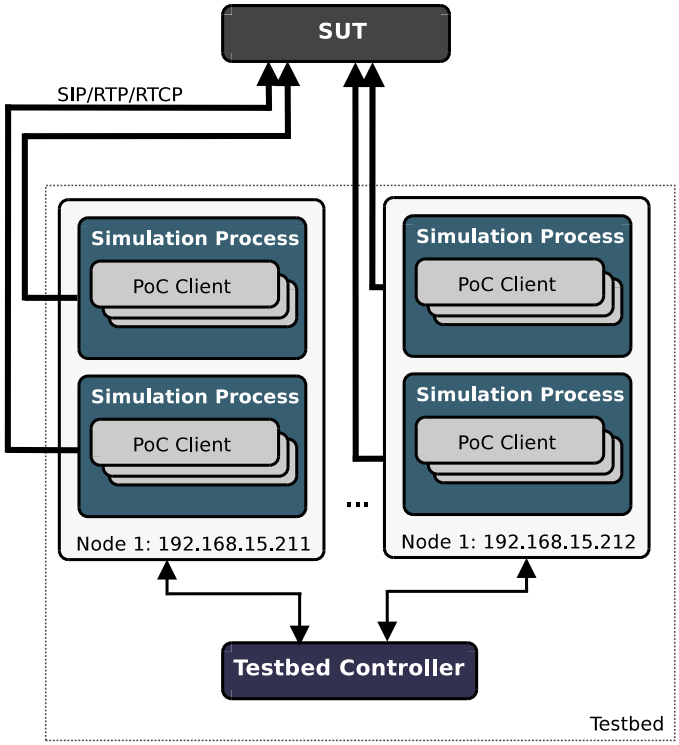

Fig. 4. Push-to-talk testbed.

them back as a feedback to the controller

The proposed testbed may be used not only for load testing, but also for providing detailed performance analysis. The previously mentioned feedback reporting mechanism is a key component of the proposed architecture. The concept hidden behind the idea is relatively easy and scales well. Each PoC client process manages a data structure called a feedback collector, which is responsible for in memory storage of different statistical information. Every once in a while, when a predefined timer is fired, a previously gathered data is used to generate minimum, maximum, sum and mean components with respect to each stored data set type. The cumulative components are sent towards the control unit, clearing at the same time corresponding PoC client feedback collector entries. It is worth to mention that the control unit is managing a similar feedback collector component, which additionally may compute various statistical information basing on feedback probes' timestamps.

From the perspective of this paper only a limited number of performance indicators is worth considering. Those of the major interest are calculated by the feedback collector mechanism using two methods. First one, referred as timer measurement, calculating a delay as a difference of timestamps captured at the time boundaries of the event. Second based on RTCP Receiver Report and Sender Report packet analysis.

Timer measurement method is used to estimate Round Trip Time (RTT) of talk burst request packets[3]. Additionally it helps to calculate a number of incidents when consecutive RTP packets are received after they playtime (hence introducing jitter). Finally the timer measurement is used to calculate first media packet in a talk burst propagation time. It is possible because each active talker sends to the controller a *speaker report* (including a timestamp value of emitted talk burst and its unique identifier). The unique identifier is also appended as a first CSRC field of RTP packet. On reception of such a packet the receiver generates similar *reception report*[4] and sends it to the controller. By comparing unique identifiers and timestamp difference, the control unit may calculate media packet propagation time for each client. All those reports are calculated only for the first packet in a talk burst. What is more they are cumulated inside a feedback collector and sent towards the control unit every random period of time. Therefore their load impact on the controller is not that significant.

Furthermore, to provide even more detailed source of information, an RTCP packet analysis is applied to determine a number of lost packets by each client and the value of jitter itself.

Basing on all the mentioned performance indicators and real simulation using described testbed, the following section will study certain performance aspects of a hybrid IP multicasting PoC enhancement with respect to an original unicast concept.

V. Results

Series of tests were performed in order to compare unicast PoC and proposed hybrid multicasting approach. Each test was designed to measure a different performance and/or reliability factor.

A. Packet loss

A total number of 10.000 users was registered in PoC system, creating PoC groups of predefined sizes (50, 100 ... 500) in a manner that a single simulation scenario handled group of one type of size only. Each group consisted of five predefined talkers, each of them emitting 250 talk bursts of the length of 62 RTP packets. Figure 5 depicts a relation between the number of lost packets and group size. There are two measurement series, green indicating unicast approach and blue for hybrid multicast PoC. Basing on the research from [14], an acceptable packet loss boundary has been defined. The boundary indicates Mean Opinion Score of the level three[5] for AMR 4.75 codec. It is worth to point out that for unicast media packet forwarding groups approaching 100 members are slightly above the fair quality boundary.

B. Media propagation delay

The same simulation scenarios were performed to compare a media propagation time (according to the method described in Testbed section). In case of original unicast PoC approach, latencies for consecutive PoC group members resembled some kind of increasing linear relationship (the same packet was sent with some small and more or less constant delay to next user).

[3]Calculated as a difference of timestamps between generating *Talk Burst Request* and receiving *Talk Burst Granted* response.

[4]Note the speaker and reception reports mentioned here are not of RTCP type, but are invented for the purpose of the testbed.

[5]Fair level with small disruptions.

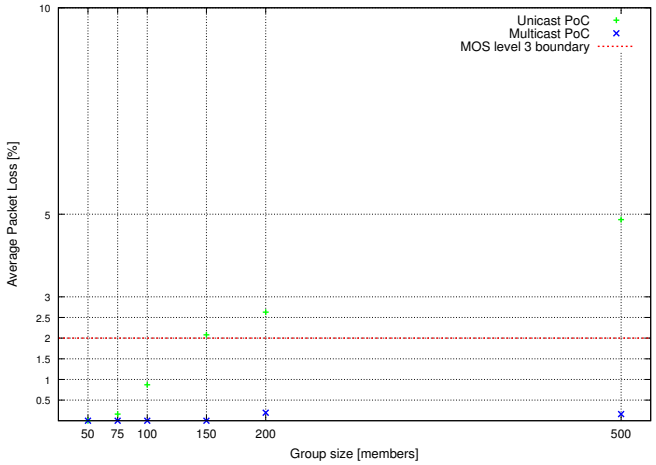

Fig. 5. Cumulative percentage of lost packets versus group size.

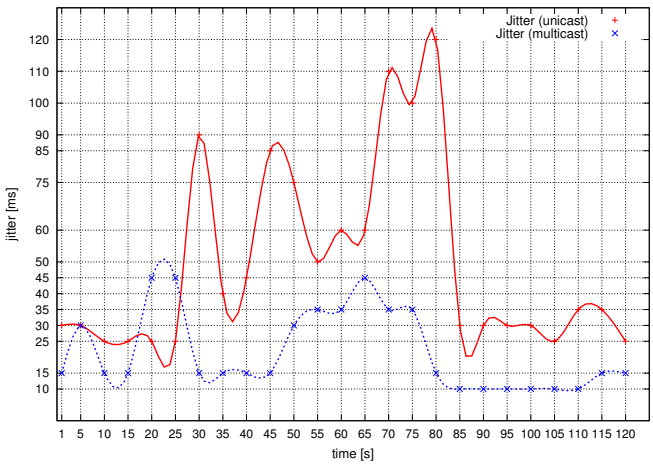

Fig. 7. Media packet jitter for unicast PoC session.

This had a strong impact on the overall average propagation delay and its standard deviation. For hybrid multicasting approach this pattern has been eliminated and the propagation time seemed to vary within some limited boundary. Figure 6 indicates the observed relationship between media propagation time. Red points reflect measured average propagation times for groups of specific size in case of unicast approach. The error bars show the observed deviation ranges. Similarly blue points refer to the hybrid multicast approach.

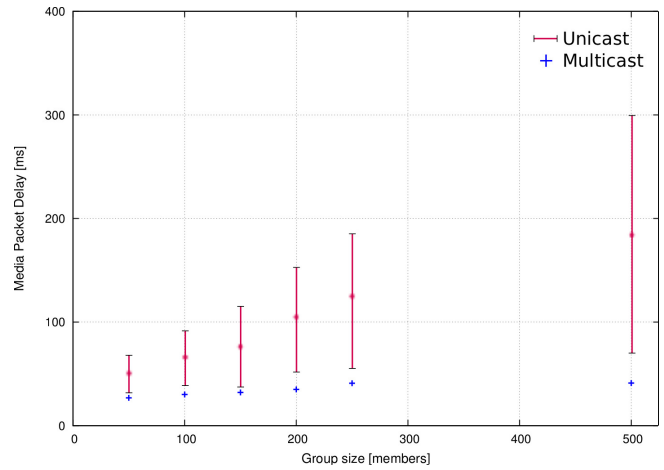

Fig. 6. Average first media packet delay versus group size.

C. Jitter

Finally a jitter relationship was determined. In this case, the simulation scenario was different. Instead of calculating total and overall jitter for each predefined group size, a group of 100 members was selected and corresponding average jitter measurements were estimated for a given time frames. Figure 7 depicts jitter variations within tested period.

From the presented jitter graph it is hard to observe any specific pattern. However, one may conclude that jitter for multicasting approach is more stable and limited within smaller boundary. Tests for group of larger size were intentionally omitted due to greater packet loss (for unicast approach) having serious impact on jitter.

VI. Conclusions and Future Work

The intention of this paper was to present a possible enhancement of Push-to-talk service. The concept has been proved by performing a series of simulations allowing to compare two methods of distributing media packets in the PoC environment. It was demonstrated that IP multicasting approach is recommended in PoC groups of size larger than 100-150 users. Additionally, from the operator perspective such a solution may be considered due to much smaller bandwidth consumption and lower packet loss. It is worth to point out that together with the increase of a single group size, the number of possible Push-to-talk use cases increases. This is really important in so called location based oriented dispatcher solutions, where large groups of users are managed by a dispatcher unit running on a standalone PC. The dispatcher knows the position of all the units which are logged into certain group and may establish adhoc sessions based on current terminal position. With large group support it is possible to have global groups of client's fleet or for example taxi drivers and contact all of them with a single button press.

One may notice, that the major difficulty regarding usage of the proposed solution in PoC environment is the necessity of having a dedicated multicasting oriented APN (implying strict mobile operator cooperation) or Wireless Local Area Networks restraint (limited area).

Due to limited testing possibilities and hardware resources, the presented testbed operated inside Local Area Network only. A possible future research area may be related to further exploration of IP multicasting PoC behavior in Wireless LAN and within 3G networks. One may also consider comparing a full IP multicasting PoC approach with the hybrid version stated in this paper.

Finally, although the paper presented a scheme for extending PoC chat group call with IP multicasting, a similar approach may be still derived for other types of PoC group calls (i.e. prearranged or adhoc).

REFERENCES

[1] "Push To Talk over Cellular requirements," Open Mobile Alliance (OMA), SPECIFICATION, Jun. 2006, requirements Document OMA-RD-PoC-V1-0.

[2] "Push To Talk over Cellular (PoC) - Architecture," Open Mobile Alliance (OMA), SPECIFICATION, Jun. 2006, architecture Document OMA-AD-PoC-V1-0.

[3] "Push To Talk over Cellular control plane," Open Mobile Alliance (OMA), SPECIFICATION, Jun. 2006, technical Specification OMA-TS-PoC-ControlPlane-V1-0.

[4] "Push To Talk over Cellular user plane," Open Mobile Alliance (OMA), SPECIFICATION, Sep. 2007, technical Specification OMA-TS-PoC-UserPlane-V1-0.

[5] "Push To Talk over Cellular user plane," Open Mobile Alliance (OMA), SPECIFICATION, Sep. 2007, technical Specification OMA-TS-PoC-XDM-V1-0.

[6] R. S. Cruz, M. S. Nunes, G. Varatojo, and L. Reis, "Push-To-Talk in IMS Mobile Environment," Fifth International Conference on Networking and Services, CONFERENCE, 2009.

[7] P. Kim, A. Balazs, E. van den Broek, G. Kieselmann, and W. Bohm, "IMS-based Push-to-Talk over GPRS/UMTS," Wireless Communication and Networking Conference, CONFERENCE, 2005.

[8] K. Pillai and H. Akhtar, "Optimizations for Push-To-Talk in Wireless Networks Efficient Management of Call Setup Latency," Third International Conference on Next Generation Mobile Applications, Services and Technologies, CONFERENCE, 2009.

[9] H. Jiang, A. K. Wong, V. W. Luk, X. Duan, and J. L. C. Ma, "Enhanced Floor Control Protocol for PoC Application in Data Packet Voice Communication," Third International Conference on Next Generation Mobile Applications, Services and Technologies, CONFERENCE, 2009.

[10] D.-J. Tsaur and C.-L. Liu, "Efficiency Analysis and Improvement of SIP-based Push-to-Talk over Cellular," Fourth International Conference on Genetic and Evolutionary Computing, CONFERENCE, 2010.

[11] K.-Y. Tsai, Yung-Feng, L. A.-C. Pang, and T.-W. Kuo, "The Speech Quality Analysis of Push-to-Talk Services," Wireless Communication and Networking Conference, CONFERENCE, 2009.

[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: a transport protocol for Real-Time applications," Internet Engineering Task Force, RFC 3550, Jul. 2003. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3550.txt

[13] M. Hauge and O. Kure, "Multicast in 3G Networks: Employment of Existing IP Multicast Protocols in UMTS," World of Wireless, Mobile and Multimedia Networks, CONFERENCE, 2002.

[14] I.-H. Mkwawa, E. Jammeh, L. Sun, A. Khan, and E. Ifeachor, "Open IMS Core with VoIP Quality Adaptation," CONFERENCE, 2002.