Considerations for deploying a geographically distributed video conferencing system

Boris Grozev
Atlassian and
University of Strasbourg
bgrozev@atlassian.com

George Politis
Atlassian and
University of Strasbourg
gpolitis@atlassian.com

Emil Ivov
Atlassian
eivov@atlassian.com

Thomas Noel
University of Strasbourg
noel@unistra.fr

***Abstract*—In this paper we report on our ongoing work to improve the user experience of video conferencing by using geo-location. We discuss the problem of selecting a media server for a video conference, introduce one state-of-the-art system which uses a simple method, and discuss a model for distributing a conference among a set of media servers. We perform a measurement study of a production service, and find that in many cases, contrary to common wisdom, connecting each participant to their closest server is not only costly, but counter-productive in terms of decreasing the round-trip time. While, the approach may still seem viable in some use cases, more research is needed in order to understand when that may be, and define specific algorithms for server selection in the case of a distributed media conference.**

***Index Terms*—Communication system software, Streaming media, Multimedia communication, Teleconferencing, Videoconferences**

I. Introduction

Modern cloud services such as Amazon AWS[1] make it easy to deploy a system comprising of servers in different geographic locations, and provide well developed tools, like Route53[2], for routing users to a server which is close to them. However, they are mostly designed for classical web applications and the scenario in which one consumer accesses a certain resource in the cloud.

The case of online video conferencing can also potentially greatly benefit from geo-location features, because it is interactive in nature, and it is very sensitive to network conditions such as packet loss, round-trip-time (RTT) and variations in RTT. However, this case is not as simple as connecting one consumer to the closest server, since it requires the interconnection of two or more user endpoints, and the problem of optimal selection of the server location(s) has not been researched much.

With the emergence of WebRTC [1] as a reliable and popular platform for real-time communication, conferencing applications are often implemented as web applications, running in unmodified web browsers. This makes reusing existing technologies for geo-location easier.

We are interested in exploring the ways in which we can use a geographically distributed system in order to optimize the user experience in video conferencing applications, and in particular those based on WebRTC.

In this paper we describe a state-of-the-art multi-party video conferencing system, which uses centralized media servers with a basic form of geo-location. We analyze its performance and propose the first steps towards improving it. More specifically, we explore the extent to which we can improve the system if we introduce a distributed mode, in which a single conference can be split to multiple servers.

We structure the remainder of this paper as follows: section II discusses other work related to our problem; section III describes how our video conferencing system works, how it uses geo-location, and introduces ways in which we propose to improve it; section IV describes some experimental results which we obtained for our system; section V interprets the results; and finally section VI concludes the paper and presents some ways in which we plan to continue this work.

II. Related Work

Voice-over-IP and video conferencing systems often use the Interactive Connectivity Establishment (ICE [2]) protocol to establish a session between endpoints. ICE works on the application level, and is designed to find the best possible connection (among the various possible [IP address, port] pairs) between two endpoints, falling back to the use of a relay server (TURN) if necessary. This is different from the problem of choosing the location of the server to be used, since it already assumes that the endpoints are known.

The International Telecommunication Union recommends that for conversational speech, a one-way delay of 400ms should not be exceeded, however some users are dissatisfied with a one-way delay of 300ms [3]. Nagy et al. propose a congestion control algorithm for multimedia conferencing, and detail the importance of the end-to-end delay for congestion control [4].

In [5] the authors explore the effects of routing transoceanic video telephony traffic through a privileged network instead of through the public internet. They report positive results, but their study is limited to one-to-one conversations between the US and China. Xu et al. measure the performance of some popular video conferencing systems, and explore their architecture, including the way in which they use geo-location [6].

[1] https://aws.amazon.com
[2] https://aws.amazon.com/route53/

978-1-5386-4649-6/18/$31.00

In a series of publications, Alvaro Alonso et al. describe a model for distributing a multimedia server among multiple machines, and detail it's use for optimal resource scheduling [7], [8], [9]. They specify geo-location as one of the use-cases for their system, but do not explore the details on how to do it, or perform any evaluations for this use. In their architecture for a distributed media server, a participant in the conference subscribes to each individual video stream that they want to receive. This is significantly different than our model, in which each participant uses a single connection towards a server (even if the conference is distributed among multiple servers).

In order to minimize packet loss and RTT Ahmed Elmokashfi et al. propose an overlay IP network [10] dedicated to transporting sensitive media traffic, coupled with control and management systems in order to provide a global Video Network Service (VNS). The key idea is that customers will experience a better video quality by sending their traffic through this tailored network than by sending it through the default path chosen by their normal Internet providers. To maximize control over the quality, they suggest that traffic should enter VNS network-wise as close as possible to the source. Once in VNS network, the traffic is carried inside VNS as close as possible to the destination, and then released on the Internet. User media traffic is pooled from arbitrary Internet locations into the VNS network using transport or application-layer media relays, such as TURN relays, SIP B2BUA, or Multipoint Conferencing Units.

III. Geo-location for video conferences

While we are interested in the general problem of using geo-location in video conferencing, we work with a specific software system which we analyze and for which we implement geo-location features. It is an open-source system called Jitsi Meet, and it is based on WebRTC [1]. Users connect to a conference using a web browser, and use a variety of devices including mobile, laptop, or dedicated conferencing hardware.

When clients join a conference, they initially contact a signaling server, which manages the presence of participants in the room and the server side resources. This communication uses the XMPP protocol [11], transported via BOSH[3] and ultimately HTTP. This makes it easy to re-use existing technologies such as Amazon Route53 and Elastic Load Balancer to perform geo-location on the signaling level.

The actual exchange of audio and video goes through a separate media server (jitsi-videobridge). WebRTC uses ICE [2] for connection establishment, and the Real-time Transport Protocol (RTP [12]), usually over UDP, for the transport of audio and video. With the current implementation each conference is hosted on one media server that all clients connect to, but in the section below we also consider a distributed model.

A. Current geo-location architecture

Our system[4] is hosted on Amazon AWS. It comprises of a set of independent shards in different regions. Each shard has one signaling node and one or more media servers.

The only place where geo-location is taken into account is at conference allocation time. The first client to join a conference determines which region the conference will be allocated in, and once allocated the conference is hosted there until it terminates. Participants joining afterwards are routed to the existing server regardless of their location.

This is not always optimal, because the location of the first client isn't always representative. It could be the case that the majority of the users are in one location, which is different than the location of the first participant. However, it is very straightforward to implement, which is the reason we chose this solution initially. All that is needed is a global mapping of a conference identifier to the shard which hosts the conference, and Route53 can be used for the initial selection of a shard (i.e. if a conference doesn't exist, the user is sent to an HTTP endpoint which resolves to a shard in its region (if possible)).

B. Distributed conferences

There is ongoing work for supporting a distributed mode in our media servers. That is, a conference will be hosted on more than one server, in potentially different locations. There are two different motivations for this. One is scalability – increasing the maximum number of participants that a single conference can have. And the second one is the potential for improved geo-location features.

In our proposed design a conference is distributed over a set of media servers. All media servers are connected together and forward the necessary audio and video streams to each other. A single client always connects to one of the servers, and uses that connection for both sending and receiving multimedia.

Servers from different regions can be dynamically added to a conference, which allows this scheme to be used for geo-location. There are multiple ways in which servers can be selected for a conference, and it is not clear which one is the best in terms of achieving the best possible user experience (or even something which we can measure, such as minimizing the RTT).

Using distributed media servers has its disadvantages. For one, it adds significant complexity to the system. But maybe more important is that it costs more because it requires more servers, and requires and uses significant network resources between the servers.

With this study we aim to begin to shed light on when it is appropriate to use distributed media servers for the purpose of geo-location, and what kinds of improvements we should expect.

C. Two-way conferences

We recently analyzed the special case of two-participant conferences [13]. We found that in this case we get better results by using a direct peer-to-peer connection (with a fallback to a TURN relay when necessary) than when using the media server. It also has a much lower cost for the provider, since the multimedia goes over the public Internet instead of through our infrastructure. We implement dynamic switching

[3] https://xmpp.org/extensions/xep-0206.html

[4] https://meet.jit.si

between a peer-to-peer connection used when there are two participants, and a connection to the media server used for three or more participants.

Because of this, we don't consider using a geo-located media server for conferences with two participants.

D. Three-way conferences

In this section we list the different ways in which three participants can be distributed in different regions, and discuss possible ways to connect them with one or more media servers. We do this because the number of cases for three participants is small enough to go through and it helps to illustrate the problem in the general case (and also three-way conferences are very common, so this is an important case). Below we refer to the cases labeled in Figure 1.

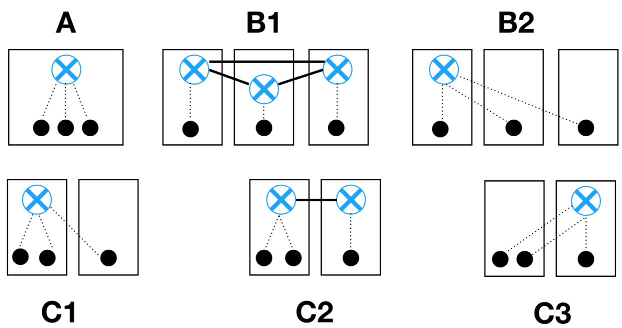

Fig. 1. The different ways in which three participants and a server can distributed in regions.

If all three participants are in the same region, clearly we should use a single media server in the same region (case "A").

If all three participants are in different regions, there are two sensible ways to connect them. We could use a single server, in one of the participant's regions (case "B2"). Or we could use three servers, one in each participant's region (case "B1").

If there are two participants in the same region, and the third is in a different region there are again two sensible ways to connect them (cases "C1" and "C2"). Case "C3" is not optimal, because if we use "C1" instead we are likely going to reduce the RTT between the two users in the same region.

This last case, "C3", illustrates the situation in which a distributed media server is most likely to have an advantage. Because of this when we analyze our service we look at how often similar situations (generalized to 3 or more participants) occur.

Note that with the geo-location algorithm described above any of the situations with one media server might occur, provided participants join and leave a conference in a particular order

IV. Results

We obtained measurements from a production service which runs on Amazon AWS. It is distributed in four AWS regions: us-east-1 (North Virginia), us-west-2 (Oregon), eu-west-1 (Ireland) and ap-southeast-2 (Sydney). The servers run on Amazon EC2 c4.xlarge instances[5].

Each conference is allocated a single server, with its region being based on the location of the first participant in the conference. This means that as new participants join, they do not always connect to a server in the region closest to them. This gives us an opportunity to measure the round trip time from users detected as being located in one region, to a server in another region.

A. Round Trip Time between regions

We measured the RTT between servers from different regions. We used ICMP ping, over a period of a week. We found that the packet loss and the variation in RTT were both negligible. Figure 2 shows the results.

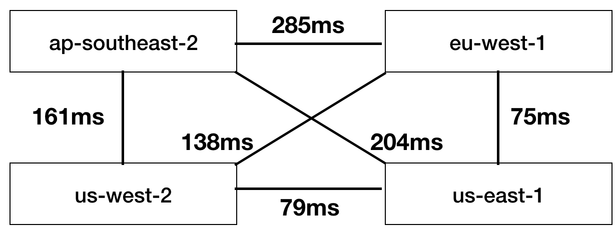

Fig. 2. The Round Trip Time in milliseconds between the four AWS regions that we use.

B. Round Trip Time (RTT) for users

We measured the average RTT between a user and a server, from actual user sessions on the service. For each user session, we obtained the RTT periodically during the conference, using the WebRTC statistics API [1]. We took the mean across these as the RTT for the session. We then grouped the sessions by the user region and the server region, and we show the mean in table I. The dataset comprises 40214 sessions, and the smallest group (users from us-west-2 connecting to ap-southeast-2) has 833 sessions.

TABLE I
THE AVERAGE RTT IN MILLISECONDS FROM USERS IN A GIVEN REGION TO A SERVER IN A GIVEN REGION. VERTICALLY: USER REGION; HORIZONTALLY: SERVER REGION.

	server region	ap	eu	us-e	us-w
user region					
ap		329	291	242	292
eu		378	110	171	216
us-e		307	189	107	116
us-w		241	213	137	81

[5]https://aws.amazon.com/ec2/instance-types/

C. Distribution of users

We analyzed the sizes of the conferences on our system in a period of 6 weeks, during which we recorded 56383 conferences. Since the size of a given conference changes with participants joining and leaving, we looked at the total amount of time that conferences of a given size were active. The results are shown in Figure 3.

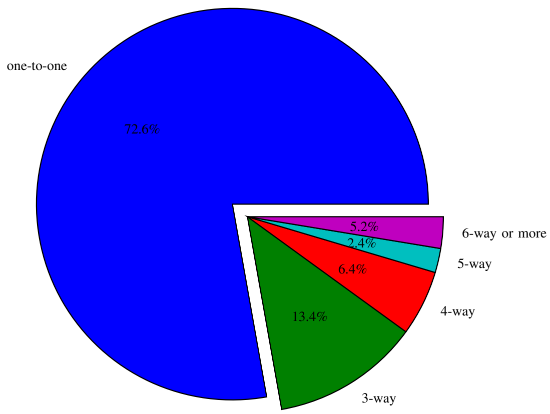

Fig. 3. The distribution of conference-time on our service by conference size.

In addition, we see that 72% of all conferences never grow to size 3 or more for more than two minutes (we introduce this threshold, because if a user reloads the web-page this might incorrectly register as a temporary third participant, until the old session times out). We refer to the remaining 28% (which have grown to size 3 or more for at least two minutes) as multi-party conferences.

Next, we look for situations in which a conference hosted on a server in one region has at least two participants from the same region, but different from the server region (this is similar to case "C3" from section III-D, which we hypothesize can be improved by using a distributed media server). We also only look for multi-party conferences, because for the one-to-one conference case we use a peer-to-peer connection (see section III).

For example, a conference hosted in us-east-1, with two participants from eu-west-1 and one in us-east-1 will qualify. But a conference hosted on us-east-1 with just two participants from eu-west-1 will not qualify. A conference hosted in us-east-1 and one participant in eu-west-1, us-east-1, and ap-southeast-2 will also not qualify.

We see that 33% of all multi-party conferences reach such a state at some point in their life cycle. Split by time, 13% of the time spent in multi-party conferences is spent in this situation (which is 3.6% of all conference-time).

V. Analysis

The first thing we notice is that in almost all cases, the average RTT from a user to a server will increase if we introduce an intermediary server in the user's region. As an example, the average RTT from users in Europe to our server in eu-west-1 is 104ms, and to us-east-1 it is 141ms. Taking into account the 75ms RTT between our servers in eu-west-1 and us-east-1, if we route European users through a server in eu-west-1 we should expect to see an average RTT (to the server in us-east-1) of 179ms. Figure 4 illustrates this. We see this pattern for all pairs of regions, with the sole exception of users in us-east-1 connecting to a server in eu-west-1, in which case we see a decrease in the average on 3ms.

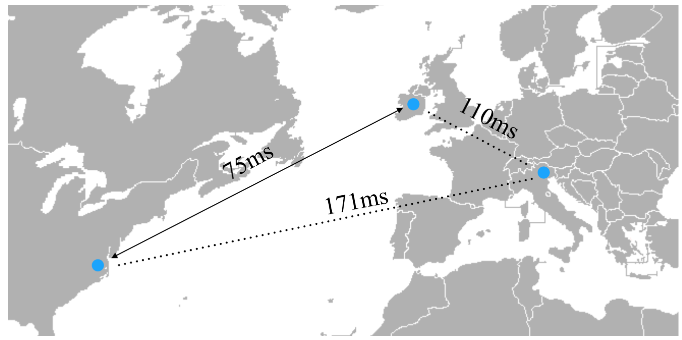

Fig. 4. The average RTT between users in Europe and our servers in Europe and the US, and the RTT between our two servers. The direct connection to the US has lower latency.

This means that using a distributed media server in a case like "B1" (see section III-D) will increase the RTT between all pairs of participants in the conference, i.e. it will be counterproductive in addition to being costly. It also means that case "C1" is to be preferred over "C2", and more generally, that a single participant in a remote region does not justify the use of a separate server in that region.

This effect is strongest in the ap-southeast-2 region, where it turns out that users from within the region have a better RTT to servers in any other region, then they do to the server in their own region. We suspect that this is because ap-southeast-2 covers a much larger geographic area, including both countries in Asia as well as Australia. In this case introducing an intermediary server in ap-southeast-2 (Sydney) will significantly increase the RTT for many users, to the point where it might exceed the ITU recommendations and start to inhibit the ability of users to communicate.

We expect that placing servers in more regions (we currently use 4 region, out of the 16 regions that AWS offers) will have a significant impact on the users average RTT. It will decrease it for local users, and it will also decrease the negative effect of using an intermediary server. However, this also has higher infrastructure costs.

We also notice that the majority of conferences on our service only have two participants. Our proposed distributed architecture would not affect these cases. Still, multi-party conferences are one of the core features in the system, and as such we consider them an important use-case. The situation which we describe in section III-D, which our distributed architecture has the most potential to improve (case "C3"), affects about one in three conferences. Because of this we consider it worthwhile to continue our study with this approach in future research.

VI. Conclusion and Future Work

Geo-location for video conferencing is more complicated than it is for simple resource-consumer systems. Following the common wisdom and connecting all users to the nearest to them server is not always optimal, and in some cases will have a negative impact on users. It is also technically challenging to implement and has other disadvantages, such as increased traffic for the infrastructure.

The effects of adding geo-location to a video conferencing system depend on the hosting/cloud provider, and on the users of the system. An in-depth analysis should be performed for the specific service under consideration, in order to get the most out of geo-location and not introduce harm.

The most simple approach for geo-location – using a single server and choosing a region based on the first participant in the conference – is easy to implement and works reasonably well. Using very large regions (e.g. a single region for both Asia and Australia) leads to significantly long RTT values.

We hypothesize that using more regions will improve users' RTT (and thus QoS), and we plan to test this idea experimentally in future work. We also plan to perform more detailed measurements, in which we record the RTT from one user to multiple potential server locations. Finally, we plan to define and test specific algorithms for selecting a server for the distributed server case.

References

[1] A. Bergkvist, D. Burnett, C. Jennings, A. Narayanan, B. Aboba and T. Brandstetter, "WebRTC 1.0: Real-time Communication Between Browsers," November 2017, W3C Editor's Draft 23. [Online]. Available: {https://www.w3.org/TR/2017/CR-webrtc-20171102/}

[2] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols," RFC 5245 (Proposed Standard), Internet Engineering Task Force, Apr. 2010, updated by RFC 6336. [Online]. Available: http://www.ietf.org/rfc/rfc5245.txt

[3] ITU-T, "ITU-T recommendation G.114," International Telecommunication Union, Tech. Rep., 2003.

[4] M. Nagy, V. Singh, J. Ott, and L. Eggert, "Congestion control using fec for conversational multimedia communication," in *Proceedings of the 5th ACM Multimedia Systems Conference*, ser. MMSys '14. ACM, 2014, pp. 191–202.

[5] J. Li, Z. Li, Y. Liu, and Z. L. Zhang, "Do twin clouds make smoothness for transoceanic video telephony?" in *2015 44th International Conference on Parallel Processing*, Sept 2015, pp. 260–269.

[6] Y. Xu, C. Yu, J. Li, and Y. Liu, "Video telephony for end-consumers: Measurement study of google+, ichat, and skype," *IEEE/ACM Transactions on Networking*, vol. 22, no. 3, pp. 826–839, June 2014.

[7] . Alonso, I. Aguado, J. Salvacha, and P. Rodrguez, "A methodology for designing and evaluating cloud scheduling strategies in distributed videoconferencing systems," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2282–2292, Oct 2017.

[8] . Alonso, P. Rodrguez, J. Salvacha, and J. Cervio, "Deploying a multipoint control unit in the cloud: Opportunities and challenges," *Cloud Computing 2013*, pp. 173–178, 2013.

[9] P. Rodrguez, . Alonso, J. Salvacha, and J. Cervio, "dotm: A mechanism for distributing centralized multi-party video conferencing in the cloud," in *2014 International Conference on Future Internet of Things and Cloud*, Aug 2014, pp. 61–67.

[10] A. Elmokashfi, E. Myakotnykh, J. M. Evang, A. Kvalbein, and T. Cicic, "Geography matters: Building an efficient transport network for a better video conferencing experience," in *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '13. New York, NY, USA: ACM, 2013, pp. 369–380. [Online]. Available: http://doi.acm.org/10.1145/2535372.2535395

[11] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," RFC 6120 (Proposed Standard), Internet Engineering Task Force, Mar. 2011, updated by RFC 7590. [Online]. Available: http://www.ietf.org/rfc/rfc6120.txt

[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550 (INTERNET STANDARD), Internet Engineering Task Force, Jul. 2003, updated by RFCs 5506, 5761, 6051, 6222, 7022, 7160, 7164. [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt

[13] G. Politis, B. Grozev, P. Domas, E. Ivov, and T. Noel, "Balancing peer-to-peer and relayed transport for media conferences," submitted.