Janus: a general purpose WebRTC gateway

Alessandro Amirante alex@meetecho.com
Tobia Castaldi tcastaldi@meetecho.com
Lorenzo Miniero lorenzo@meetecho.com
Simon Pietro Romano spromano@unina.it

September 26, 2014
Outline

1. Some context
   WebRTC and standardization activities

2. Writing a WebRTC gateway from scratch
   Programmable Real-time Media Components

3. Janus: a general purpose WebRTC gateway
   Modular architecture

4. Next steps
Real-time media in browsers

- No standard solution!
  - No interoperability
  - Plugins need to be installed anyway

**WebRTC = Joint standardization efforts**

- Internet Engineering Task Force (IETF)
- World Wide Web Consortium (W3C)

**RTCWEB (IETF)**

- Real-Time Communication in WEB browsers WG
- Defines protocols and formats to use

**WEBRTC (W3C)**

- Web Real-Time Communications WG
- Defines UI and API to access devices
WebRTC reference architecture
Involving a gateway (and applications)
Involving different technologies as well
Do we really need a gateway?

• Several reasons for a YES, here
  • Relieve full-meshes (heavy on the client side)
  • Leveraging widespread technologies (e.g., SIP infrastructures)
  • Fixing things between implementations

• Reason for a NO?
  • You won’t go beyond 1-1 WebRTC communication
  • You don’t want an infrastructure
  • You don’t care about legacy stuff

“What is a WebRTC Gateway anyway?”
• http://webrtchacks.com/webrtc-gw/
Real-time Media Components

- Writing a gateway from scratch is a heavy task
  - Implementation of the WebRTC protocol suite
- Bridge between “legacy” stuff (SIP, RTMP, etc.) and WebRTC
  - Needs to support both (WebRTC gateway) → J1
  - What about statistics? → D1
  - Reachability may be an issue → D2
- Programmable interface
  - Different applications/technologies, different requirements
  - Dynamic management of media flows and users
  - Something a-la MEDIACTRL? → W1, B1, R1, R2
The WebRTC protocol suite

- Signalling (well, sort of) and Negotiation
  - Javascript Session Establishment Protocol (JSEP)
  - Session Description Protocol (SDP) adaptation
- Connection Establishment and NAT Traversal
  - Session Traversal Utilities for NAT (STUN)
  - Traversal Using Relay NAT (TURN)
  - Interactive Connectivity Establishment (ICE)
- Media Transport and Control
  - Real-time Transport (and Control) Protocol (RTP/RTCP)
  - Secure Extensions to RTP (SRTP)
  - Datagram Transport Layer Security (DTLS)
- Media Transport and Control
  - Opus audio codec (MTI, Mandatory-to-implement)
  - VP8 video codec (MTI candidate)
- Generic Data
  - WebRTC Data Channels (SCTP)
Janus: a general purpose WebRTC gateway

“In ancient Roman religion and myth, Janus [..] is the god of beginnings and transitions, and thereby of gates, doors, passages, endings and time. He is usually depicted as having two faces, since he looks to the future and to the past.”

Janus: a general purpose WebRTC gateway

- A door between the communications past and future
  - Legacy technologies (the “past”)
  - WebRTC (the “future”)

Janus

General purpose, open source WebRTC gateway

- [https://github.com/meetecho/janus-gateway](https://github.com/meetecho/janus-gateway)
- Demos and documentation: [http://janus.conf.meetecho.com](http://janus.conf.meetecho.com)

- Design and implementation of the gateway
  - WebRTC suite re-implemented (core)
  - Modular architecture rewritten from scratch
  - Plugins as the MEDIActRL “packages”
Modular architecture

- The core only implements the WebRTC stack
  - JSEP/SDP, ICE, DTLS-SRTP, Data Channels, ...
  - Integrated web-server for communication with browsers

- Application logic implemented in server side plugins
  - Users attach to plugins via the gateway core
  - The gateway handles the WebRTC stuff
  - Plugins route/manipulate the media/data

- Some proof of concept plugins implemented
  - Echo Test
  - Streaming (→ SOLEIL!)
  - Conferencing (→ Meetecho!)
  - SIP Gateway (→ Tiscali Indoona!)
  - ...

Extensible Architecture and API
Extensible Architecture and API

Janus Gateway

Core

Plugin 1

Plugin 2

Plugin N

Protocol messages

Janus
S. P. Romano
WebRTC
Standardization
Gateways
Requirements
Janus
Modular
Next steps
Plugins as “bricks”

- Each plugin is a feature, not an application
- Application can be composed out of different features
  - Features as “bricks” for a complex scenario
- A few examples...
  - Screensharing with Q&A
    - Video MCU (screen) + Video MCU (speakers) + Audio Bridge (questions)
  - Video communication in social networks
    - SIP plugin (calls) + Echo Test (diagnostics) + Voice Mail (messaging)
  - Social TV
    - Streaming (TV channel) + Video MCU (interaction)
Screen Sharing with Q/A

Audio bridge plugin

Video MCU plugin

Audio

Video

Text chat

Shared screen

External feature
(not provided by Janus)

Video MCU plugin

Janus

S. P. Romano

WebRTC

Standardization

Gateways

Requirements

Janus

Modular

Next steps
Social TV

TV live broadcast

User 1 video
User 2 video
User 3 video
User 4 video

→ Video MCU plugin

→ Streaming plugin
Benchmarks Janus

• Measuring Janus performance is not trivial...
  • Different plugins have different behaviours
  • Different technologies have different impact

• Need to focus on a single “application”
  • Video MCU is a typical usage (e.g., conferencing)
  • Other open source implementations available
    • Lynckia, Jitsi, Medooze, etc.

• Compared stress testing done in lab
  • Janus Video MCU, JitMeet, Lynckia
  • Scripts to automate and stress client and server sides
  • R used to compare results
Video MCU performances comparison

Server CPU usage comparison

- Janus
- Jitsi
- Licode
Video MCU performances comparison

Server memory usage comparison

- Janus
- Jitsi
- Licode
Video MCU performances comparison

Client CPU usage comparison

- Janus
- Jitsi
- Licode

CPU usage (%) vs n of peers
Video MCU performances comparison

Client memory usage comparison

- Janus
- Jitsi
- Licode

n of peers | Mem usage (%)
---|---
2 | 2
4 | 4
6 | 6
8 | 8
10 | 10
12 | 12
What to do next?

- Finalize the WebRTC implementation
  - SSRC multiplexing
- Integrate in our WebRTC-related projects
  - Meetecho (web conferencing)
  - SOLEIL (large scale streaming)
- Mobile access
  - Implementation for Android basically done, to refine
- Improve the pluggable architecture
  - Plugins as “filters”, not only sinks (e.g., transcoders)
  - Plugins in series and/or in parallel

- **Test test test!**
  - More numbers are important
    - Hoping the community can help there
    - Several people already trying it in the field
  - Can the architecture be improved somewhere?
Questions? Comments?
Related Publications

- International journals

- Book chapters

- Conferences and workshops
  W1 L. Miniero, “Improving the scalability of real-time multimedia applications using brokering of media resources”, InfQ 2013, June 13-14, 2013, Sorrento, Italy
  W2 A. Amirante, T. Castaldi, L. Miniero and S. P. Romano, “SOLEIL: Streaming Of Large-scale Events over Internet cLouds”, 11th Italian Networking Workshop, January 15-17, 2014, Cortina d’Ampezzo, Italy

- Request For Comments (RFC)
  R1 C. Boulton, L. Miniero, G. Munson, “Media Resource Brokering”, RFC6917, April 2013

- Internet Drafts
  D1 L. Miniero, S. Garcia Murillo, V. Pascual “Guidelines to support RTCP end-to-end in Back-to-Back User Agents (B2BUAs)”, draft-ietf-straw-b2bua-rtcp-01