Asterisk: Where is it going this year?

Matthew Fredrickson

@creslin287

Personal Background

Who are you and what have you done with Matt Jordan!!?

- **Worked at Digium since 2001 in various developmental capacities**
- **Worked on Asterisk at different times**
- **Maintained libpri and DAHDI for many years**
- **Wrote an SS7 stack for Asterisk (libss7)**
- **Worked on WebRTC related initiatives for the last few years**
- **Now Project Lead of the Asterisk project**

A little bit of history

Asterisk 1.0 - First major release, ISDN support, H.323, MGCP, AGI, SIP

[many changes later]

Asterisk 1.6 - Wideband audio, SS7 support

[…some time passes…]

A little bit of history

Asterisk 11 - Beginnings of WebRTC support in chan_sip

Asterisk 12 - chan_pjsip, ARI

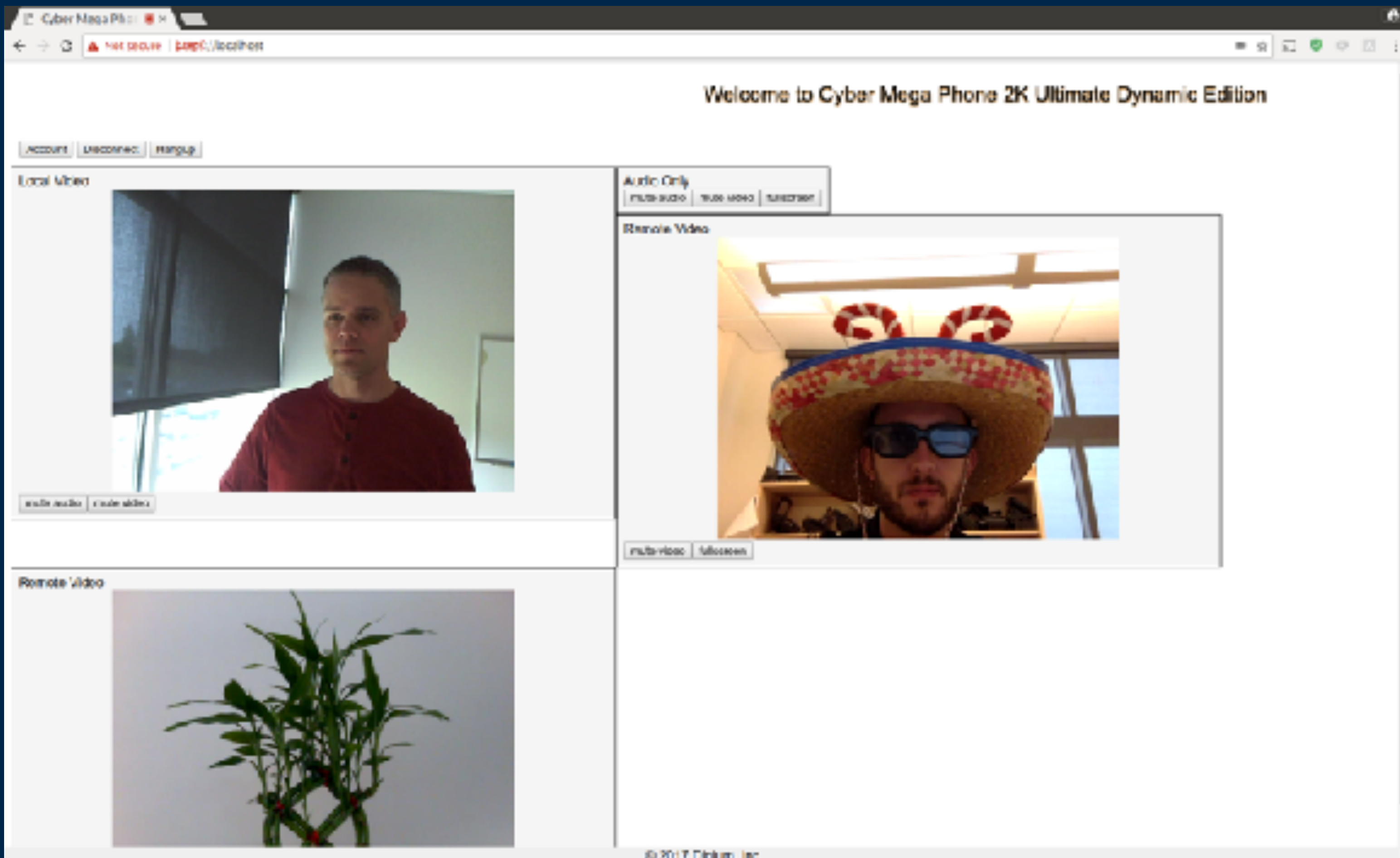
Asterisk 13 - more ARI, more PJSIP

Asterisk 14 - Async DNS, more ARI, publishing presence

Asterisk 15 - Big New Features

- **BUNDLE - RTP transport multiplexing**
- **Unified Plan - SDP schema for multi stream video**
- **RTP Layer Improvements - End to end sequence number preservation**
- **Asterisk core multistream enhancement**
- **Asterisk app_confbridge adding SFU support**

Asterisk 15 - Big New Features



Asterisk 16 - What's Next?

- WebRTC Video Quality Improvements
- WebRTC API Additions
- Improving Performance in chan_pjsip

Asterisk 16 - Improving Video Resilience

digium
The Asterisk Company

Why?

- Video is a lot more sensitive to packet loss than audio

- The loss of a single packet can have significant impact on the quality of the video stream

- In Asterisk 15 we took the sledgehammer approach, and just requested a new full video frame

- Browsers have incorporated a number of better technologies to help combat packet loss scenarios, particularly with regards to video

Asterisk 16 - Improving Video Resilience

NACK:

- RTCP-FB message

- Exactly what is sounds like - a proactive negative acknowledgement RTCP packet from the receiver of an RTP stream.

- NACK can be for one or more previous sequence numbers that may be missing.

RTP Sequence:

RX: 1, 2, 3, 6

TX: NACK(4, 5)

Asterisk 16 - Improving Video Resilience

NACK - caveats:

- Depends on low RTT time for feedback message

- NACKs quickly become impractical with longer round trip times.

- NACK’ing causes underlying packet buffers (end to end call latency) to potentially increase.

Asterisk 16 - Improving Video Resilience

REMB:

- Receiver Estimated Max Bitrate

- RTCP-FB message

- Support is negotiated in SDP

- Receiver tries to estimate it’s maximum received bitrate.

- Receiver tries to send this calculation to the remote video sender so that it can reduce or increase it’s video encoding bitrate accordingly

- Asterisk gathers these reports and tries to intelligently interpret them and inform senders to help them adjust their video encoding bandwidth.

Asterisk 16 - WebRTC API Improvements

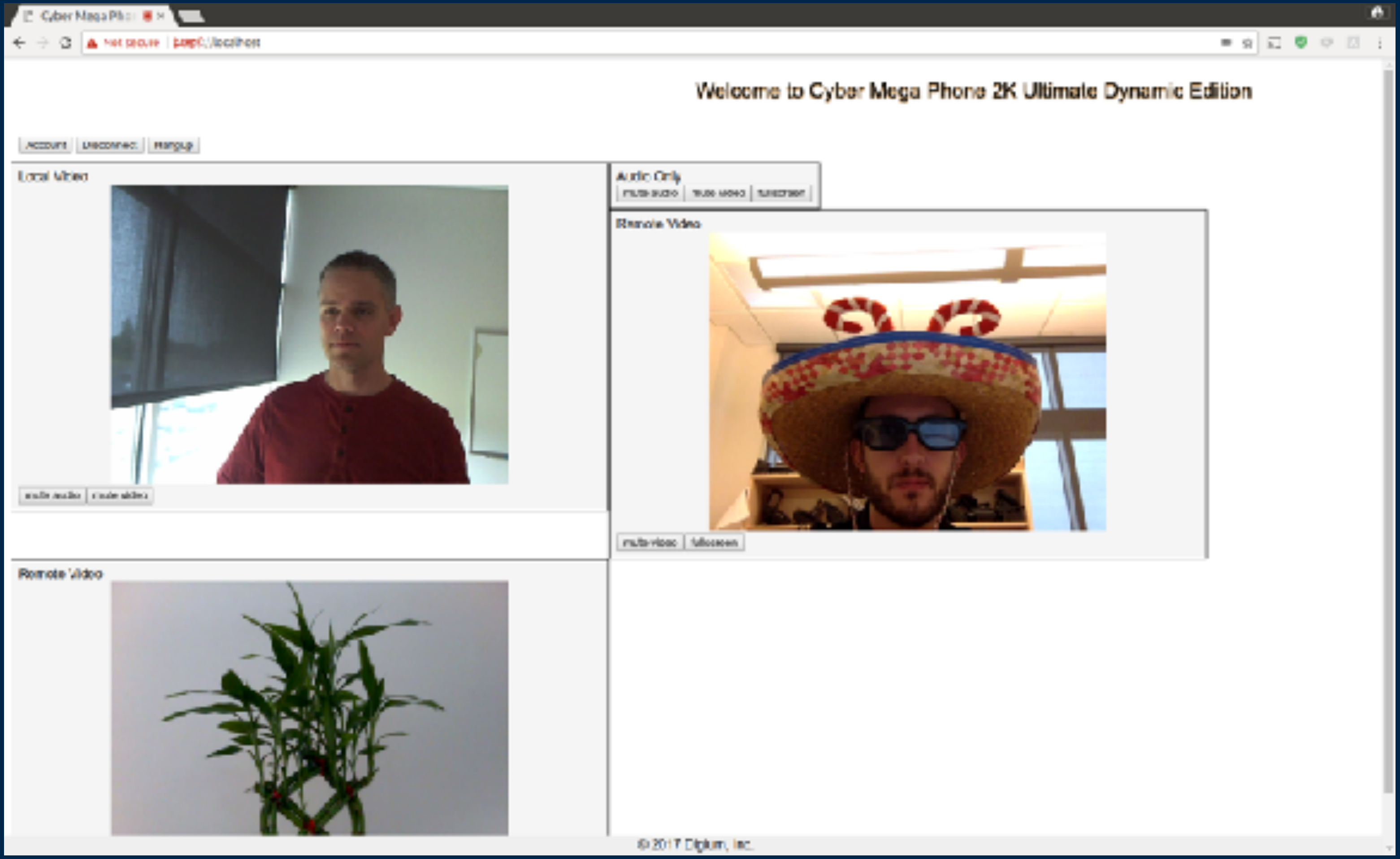
Background:

- Updated internal APIs in Asterisk to support multiple audio/video streams per call

- Built prototype web application of SFU using browser based SIP stack

- Added ARI (Asterisk REST Interface) support for new SFU style bridge mixing

Asterisk 16 - WebRTC API Improvements



Asterisk 16 - WebRTC API Improvements

Problem:

- Realized that it was challenging (from the browser) to correlate incoming streams on a call to participant metadata

- TL&DR: No good way to figure out the CallerID or other call related metadata programmatically of a particular video stream coming to the browser

Asterisk 16 - Improving Performance In chan_pjsip

- Good talk at Astricon last year about performance differences between chan_sip and chan_pjsip

- In many cases, chan_pjsip performed better

- There were a few cases where chan_sip had better performance that we'd like to see addressed

- Not significant difference - in the testing they did, it was on the order of one to two simultaneous calls more in some specific use cases.

Project Background

Asterisk 11 (LTS) was released in October of 2012

Asterisk 12 was released in December of 2013

Asterisk 13 (LTS) was released in October of 2014

Asterisk 14 was released in September of 2016

Asterisk 15 was released in October of 2017

LTS versus Standard release

- LTS - Long term support

- LTS releases (11, 13) - bug fixes for 4 years, followed by 1 year of only security fixes.

- Standard (12, 14, 15) - bug fixes for 1 year, followed by 1 year of only security fixes.

What about 16?

- Multistream extensions in core of Asterisk have been remarkably stable.

- Asterisk 16 will be an LTS

- 4 years of bug fixes

- 1 additional year of security fixes

Reminder

- 11 went EOL in October. No more security fixes or bug fix fixes. Get off that branch! (particularly if you run WebRTC)

- Keep track of what's happening in newer (non-LTS) major releases of Asterisk - if you don't, you potentially can experience big surprises when you move forward.

- Astricon is October 9-11 in Orlando! Use code "Community2018" for a 20% discount. Speakers and attendees are welcome!

Thanks!

THANK YOU!

Follow me @creslin287 on twitter.

digium

The Asterisk Company

Empowering Communication

www.digium.com • www.asterisk.org