WebRTC, Mobility, Cloud, and More...

IIT REAL-TIME COMMUNICATIONS

Conference & Expo ● Oct 17 - Oct 20, 2016 ● Chicago

Developing RTC Applications in Edge

Bernard Aboba, Microsoft Corporation

IIT RTC Conference
Chicago, IL
October 17, 2016

Edge Realtime Platform Evolution

https://dev.windows.com/en-us/microsoft-edge/platform/status/

Media Capture and Streams	Supported
WebRTC – Object RTC API	Supported
WebRTC – WebRTC v1.0 API	Preview
H.264/AVC for RTC	Preview
VP8 for RTC	In Development
Screen Capture	Under Consideration
MediaRecorder	Under Consideration

What Can You Do With Edge?

Build ORTC applications

Run WebRTC 1.0 Applications

- Using latest WebRTC 1.0 API over ORTC (adapter.js)
- Using legacy native WebRTC 1.0 API (in preview)

Edge ORTC Test Drive

https://developer.microsoft.com/en-us/microsoft-edge/testdrive/

Twilio and ORTC Phone Call

User Input, Misc • Updated Sep 18, 2015

Use Microsoft Edge's ORTC API and Twilio Client to place a voice call from your browser to any phone number.

View demo

SimpleWebRTC with ORTC

Media • Updated Sep 24, 2015

Use Microsoft Edge's ORTC API and the WebRTC APIs in Chrome and Firefox to make cross-browser conference calls.

View demo

ORTC Demo

User Input, Media, Misc • Updated Sep 18, 2015

This demo page allows you to create audio/video call via the ORTC API between Edge browsers. It demonstrates the basic code flow for a 1:1 connection between two peers.

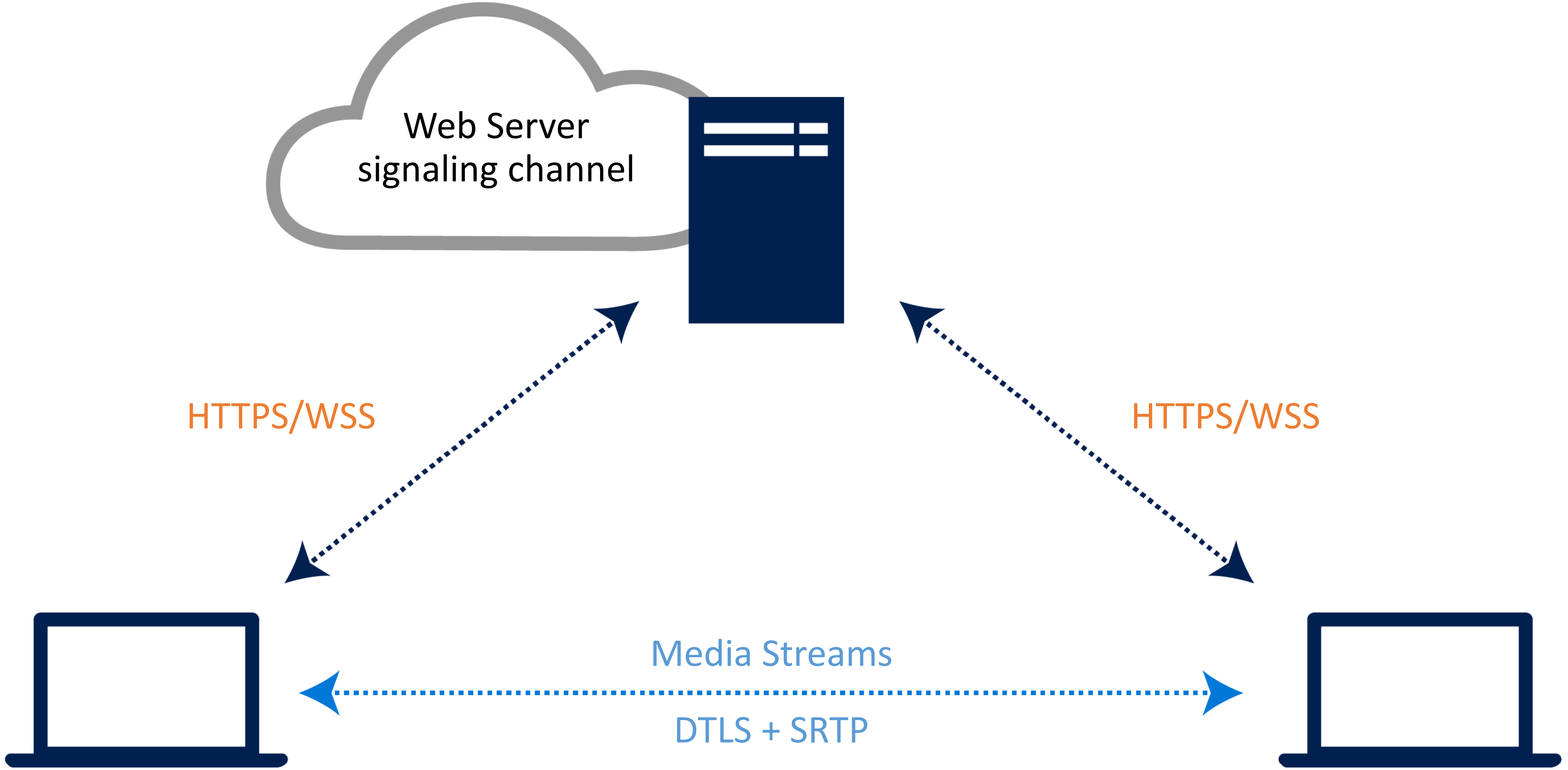
View demo

Twilio talk: https://channel9.msdn.com/Events/WebPlatformSummit/edgesummit2016/ES1611

Step-by-Step: Building an ORTC Application

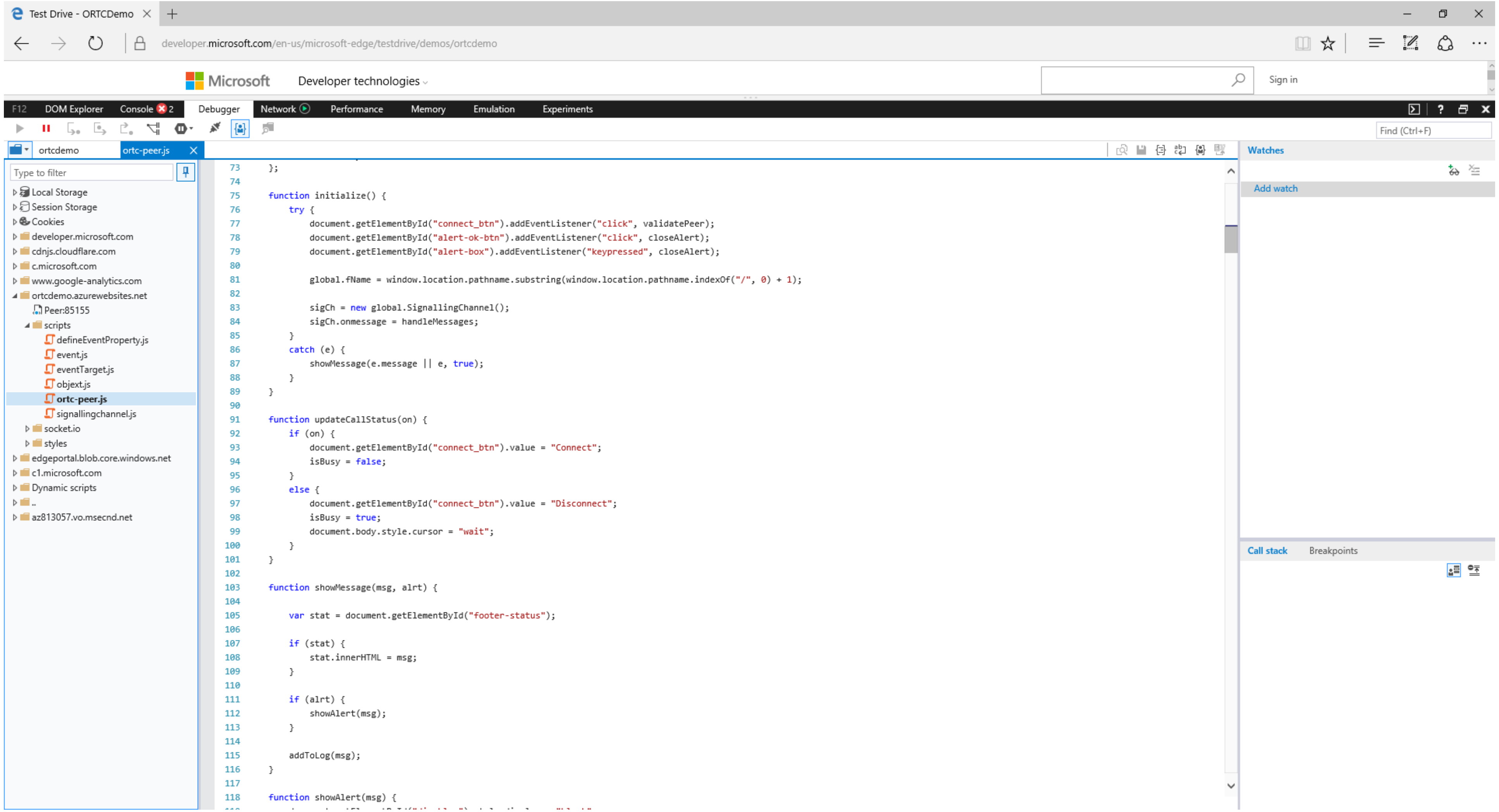
1. Establish a signaling channel
2. Create MediaStream object with audio and video tracks
3. Create transports and exchange parameters.
4. Create sender and receiver objects and extract and exchange capabilities
5. Start the ICE and DTLS transports, call send() and receive() methods
6. Connect incoming media tracks from RtpReceiver to audio and video tags

A Simple Topology



ORTC Demo Application

https://developer.microsoft.com/en-us/microsoft-edge/testdrive/demos/ortcdemo/



App level code flow

// 2. Create MediaStream object (i.e. Media Capture API) with one audio track and one video track

```
navigator.mediaDevices.getUserMedia({
      "audio": true,
      "video": {
         width: 640,
         height: 480,
         facingMode: "user"
      }
   }).then(
      gotMedia
    ).catch(
      gotMediaError
    );
```

// 3. Create transports (ICE gatherer, ICE transport, DTLS transports, etc.) and exchange parameters.

```
function initiateConnection() {
   updateCallStatus();

   var iceOptions = { "gatherPolicy": "all", "iceServers": [{ "urls": "turn:turn-
      testdrive.cloudapp.net:3478?transport=udp", "username": "redmond",
      "credential": "redmond123" }] };

    iceGathr = new RTCIceGatherer(iceOptions);
    iceTr = new RTCIceTransport();
    dtlsTr = new RTCDtlsTransport(iceTr);
...
   signalMessage(JSON.stringify({
      params: {
                "ice": iceGathr.getLocalParameters(),
                "dtls": dtlsTr.getLocalParameters()
              }
   }));

...
```

// 4. Create sender and receiver objects and extract and exchange capabilities

```
function gotMedia(stream) {
   var audioTracks = stream.getAudioTracks();

   if (audioTracks.length > 0) {
      var audioTrack = audioTracks[0];
      local_audio_MST = audioTrack;

      audioSender = new RTCRtpSender(audioTrack, dtlsTr);
      sendAudioCaps = RTCRtpSender.getCapabilities("audio");

      signalMessage(JSON.stringify({
         params: {
            "sendAudioCaps": sendAudioCaps
         }
      }));
}
...
```

// 5. Start the ICE and DTLS transports

```
if (message.params) {
    var remote = message.params;

    if (remote.ice) {
        remoteIceParams = remote.ice;
        remoteDtlsParams = remote.dtls;

        if(localCandidatesCreated){
           iceTr.start(iceGathr, remoteIceParams,(selfInfo.dtlsRole && selfInfo.dtlsRol
           === "client" ? "controlled" : "controlling" ));
                 dtlsTr.start(remoteDtlsParams);
           }
    }

...
```

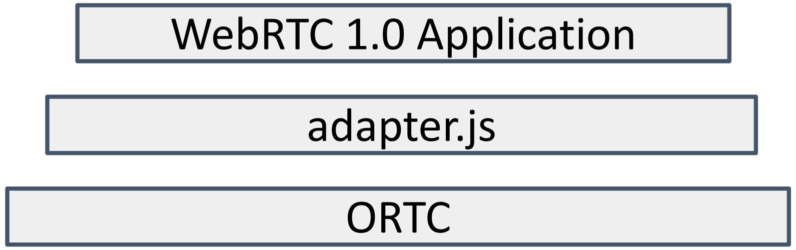
// 5. Call the send() and receive() methods

// 6. Connect incoming media tracks from RtpReceiver to audio and video tags

```
if(remote_audioRecvParams){
   var remote = remote_audioRecvParams;
   var audioRecvParams = util.myCapsToRecvParams(receiveAudioCaps, remote.sendAudioCaps);
   audioRecvParams.encodings.push(util.RTCRtpEncodingParameters(1001, 0, 0, 0, 1.0));
   audioReceiver.receive(audioRecvParams);
   trackCount++;
      if ( trackCount == 2) {
         videoRenderer.srcObject = renderStream;
      }
}
...
if( remote_audioSendParams ){
   var remote = remote_audioSendParams;
   var audioSendParams = util.myCapsToSendParams(sendAudioCaps, remote.receiveAudioCaps);
   audioSendParams.encodings.push(util.RTCRtpEncodingParameters(1001, 0, 0, 0, 1.0));
   audioSender.send(audioSendParams);
}
```

WebRTC 1.0 Support on ORTC

Using the adapter.js library, you can build audio and video applications that run on Edge, Chrome and Firefox using the latest WebRTC 1.0 API.



Latest release of adapter.js:
https://github.com/webrtc/adapter/tree/master/release

WebRTC 1.0 sample applications:
https://github.com/webrtc/samples

WebRTC 1.0 testbed (including H.264/AVC tests):
https://github.com/fippo/testbed

What Applications Can Use adapter.js?

- Currently, adapter.js supports getUserMedia as well as P2P use of RTCPeerConnection, including:
  - Single-stream video applications supporting H.264/AVC (and soon, VP8)
    - P2P audio or audio/video chat
  - Conferencing applications using an MCU
    - MCU receives multiple video streams from participants and outputs a composite video stream.
    - Avoids need for multi-stream support (not interoperable yet)
    - Example: FreeSwitch Verto Client

RTCPeerConnection Demo

- In November 2013, Sam Dutton of Google published a “WebRTC In the Real World” tutorial: https://www.html5rocks.com/en/tutorials/webrtc/infrastructure/
  - Tutorial example uses sockets.io and a node.js server to support a 1:1 audio/video session.
  - Due to changes in sockets.io and the https: requirement in Chrome getUserMedia, the original code no longer runs, but is available here:
  - http://simpl.info/rtcpeerconnection/
  - https://bitbucket.org/webrtc/codelab/src/master/complete/step5/
- Simon Pietro Romano updated the code and added data channel support, as available here:
  - https://github.com/spromano/WebRTC_Book
  - This code also no longer runs
- With a little “renovation” and addition of adapter.js, code now runs (and interoperates) on Chrome, Firefox and Edge.
  - Need to configure Chrome and Edge for H.264/AVC support

server code (runs in node.js):

http://internaut.com:8080/~baboba/iit-tutorial/adapter/server.js

client code

http://internaut.com:8080/~baboba/iit-tutorial/adapter/js/main.js

client renovations

- Provided a TURN server URI in IceServers
  - Edge does not support STUN URIs.
- Specified vga constraints in getUserMedia
  - Edge supports higher resolutions than other browsers, both in getUserMedia as well as in the encoder (e.g. H.264/AVC profile-level-id).
- New promise-based APIs
  - navigator.mediaDevices.getUserMedia
  - createOffer/createAnswer
  - setLocalDescription/setRemoteDescription
  - addIceCandidate
- End-of-candidates handling
  - Edge: calling addRemoteCandidate on a null event.candidate required for ICE processing to start.
  - adapter.js now emits a special “end-of-candidates” candidate
- Not fixed yet: APIs deprecated in Firefox
  - addStream (in favor of addTrack, but not in Chrome yet)
  - onaddstream (in favor of ontrack, but not in Chrome yet)
  - Can be supported in adapter.js once addTrack/ontrack supported in Chrome

For More Information

- Channel 9 video on Edge RTC platform
  - https://channel9.msdn.com/Events/WebPlatformSummit/edgesummit2016/ES1608
- Edge RTC FAQ
  - http://internaut.com:8080/~baboba/ms/edge.pdf
- Edge ORTC Developer documentation
  - http://internaut.com:8080/~baboba/ms/msortc-rs.html
- H.264/AVC in Edge (now on by default in Windows Insider Preview):
  - https://blogs.windows.com/msedgedev/2016/05/27/previewing-h-264avc-for-ortc/#LlVATlHe5GWtr10v.97
- Skype for Web:
  - https://web.skype.com/

Questions?