ALCATEL-LUCENT RAINBOW™

Network Requirements

GETTING STARTED GUIDE Ed 9

MAY 2018

Author: Operations - Cloud Services

Alcatel·Lucent Enterprise

Disclaimer

This documentation is provided for reference purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this documentation, this documentation is provided “as is” without any warranty whatsoever and to the maximum extent permitted.

In the interest of continued product development, ALE International reserves the right to make improvements to this document and the products it describes at any time without notice or obligation.

Copyright

Contents

Glossary

ALE: Alcatel-Lucent Enterprise

PBX: Private Branch Exchange

HTTP: Hyper Text Transfer Protocol

HTTPS: Hyper Text Transfer Protocol Secured

ICE: Interactive Connectivity Establishment – RFC 5245

STUN: Simple Traversal of UDP through NAT – RFC 5389

TURN: Traversal Using Relays around NAT – RFC 5766

DTLS-SRTP: Datagram Transport Layer Security – Secured Real Time Protocol

1 Introduction

This guide provides technical requirements to connect Rainbow clients and Agent to Rainbow Cloud services.

2 Overview

Alcatel-Lucent Enterprise (ALE) is introducing Alcatel-Lucent Rainbow, an overlay cloud service operated by ALE. Rainbow offers contact management, presence, persistent messaging, audio/video, screen and file sharing, with PSTN termination and API openness to integrate with existing customer PBXs, machines and apps.

Rainbow's clients and agents connect to Rainbow cloud services using Web protocols.

More details about Web protocols used are provided in this document.

3 History

Modifications	Date	Edition
Video Bandwidth requirements update, WebRTC GW requirements and SDK specificities	05/17/2018	Ed 09
TURN endpoints update, bandwidth requirements update	04/07/2017	Ed 08
HTTP vs. HTTPS cleanup	04/04/2017	Ed 05
Minor change (legacy PBX Agent removed)	31/03/2017	Ed 04
Information on bandwidth added (chapter 5.5)	08/03/2017	Ed 03
Chapter 0 added, Chapter 6.1 modified	05/01/2017	Ed 02
Creation of document	27/10/2016	Ed 01

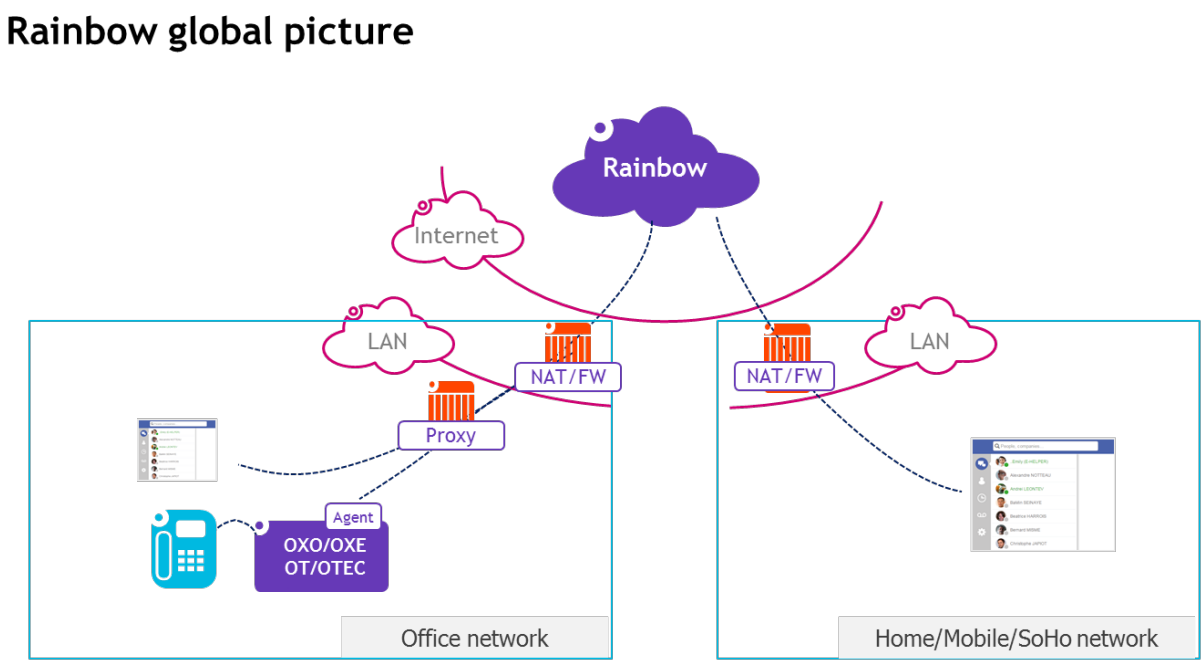
4 Related documents

None

5 Requirements

5.1 Global Overview

The following picture provides the global overview of Rainbow from network perspective:



5.2 Used Protocols

The Rainbow solution provides multiple client-side applications to connect to the service:

- A Web-based Qt-contained Desktop application for Windows and OSX.
- A Web application for Chrome/Firefox browsers.
- An iOS native application.
- An Android native application.
- An Agent to connect the PBX (can be integrated with the PBX)
- A WebRTC Gateway to establish multimedia calls between PBX and Rainbow

All applications aim at providing the same level of services and features and interact with server-side components the following way:

- Through HTTPS (443) for all REST API communications and resources loading.
- Through secure Web Sockets (WSS, 443) for all XMPP messages and notifications.
- Through DTLS and SRTP with or without STUN/TURN for WebRTC-based audio/video media streams.

5.3 Connections and Ports used

5.3.1 Rainbow Desktop and Web clients and Web SDK

Depending on the presence of an HTTP Proxy (Web Proxy), following connections take place between Rainbow client/Agent and Rainbow Cloud Services

Protocols	no Proxy, or Proxy not used	HTTP Proxy configured and used
HTTPS (Resources and REST API)	Direct, TLS to Rainbow servers, TCP port 443	Through the Proxy, TLS to Rainbow servers on TCP port 443
Web Sockets (XMPP)	Direct, TLS to Rainbow servers, TCP port 443	Through the Proxy, via HTTP switch, TLS to Rainbow servers on TCP port 443
ICE/TURN(s)	As DTLS-SRTP	
DTLS-SRTP to Rainbow clients on local network	Direct, to Rainbow clients on TCP or UDP, dynamic port range (any ports allowed by the OS)	
DTLS-SRTP to Rainbow clients thru Internet	Direct, TLS to Rainbow TURN servers, on TCP port 443/80 or UDP port 3478.	Through the Proxy, via HTTP Connect, TLS to Rainbow TURN servers on TCP port 443 or TCP port 80 (See notes)
DTLS-SRTP to WebRTC Conference thru Internet	Direct, to Rainbow conference servers, dynamic UDP port range (49152 – 65535). Backup, Direct, TLS to Rainbow TURN servers, on TCP port 443/80 or UDP port 3478.	Through the Proxy, via HTTP Connect, TLS to Rainbow TURN servers on TCP port 443 or TCP port 80 (See notes)

Note1: Firefox does not correctly support TURN-TLS thru proxy at present time, ie version 59 for this document edition. TCP-80 will be used, and port 80 must therefore be opened in firewalls for outgoing traffic when Firefox is being used. It is reminded that the usage of port TCP-80 does not imply clear media traffic. This port is only used as transport channel to the TURN server, and the applicative flow conveyed over it is encrypted end-to-end with DTLS-SRTP

Note2: The connection between a browser and the TURN Server, using TCP-80 or TCP-443 ports, are not using HTTP protocol beyond the HTTP CONNECT allowing the proxy to open the tunnel, but STUN/TURN and DTLS-SRTP protocols. In case Deep Packet Inspection is applied on the customer network and expects to examine HTTP traffic, exception rules must be applied for traffic to Rainbow TURN IP addresses, so the DPI gear allows this legitimate Rainbow TURN connections without attempt for intermediate decryption neither HTTP inspection.

5.3.2 Rainbow Android and iOS clients and associated SDKs

<table>
<tr><th>Protocols</th><th>no Proxy,
or Proxy not used</th><th>HTTP Proxy
configured and used</th></tr>
<tr><td>HTTPS (Resources and REST API)</td><td colspan="2" rowspan="4">As for 5.3.1Rainbow Desktop and Web clients, see 5.3.1</td></tr>
<tr><td>Web Sockets (XMPP)</td></tr>
<tr><td>ICE/TURN(s)</td></tr>
<tr><td>DTLS-SRTP to Rainbow clients on local network</td></tr>
<tr><td>DTLS-SRTP to Rainbow clients thru Internet</td><td rowspan="2">As for 5.3.1Rainbow Desktop and Web clients, see 5.3.1</td><td rowspan="2">Proxy Not supported for media, but the app automatically falls back to mobile data network is done in case outgoing traffic to UDP-3478 or TCP-443 are not allowed by the local firewall</td></tr>
<tr><td>DTLS-SRTP to WebRTC Conference thru Internet</td></tr>
</table>

5.3.3 WebRTC gateway

<table>
<tr><th>Protocols</th><th>no Proxy,
or Proxy not used</th><th>HTTP Proxy
configured and used</th></tr>
<tr><td>HTTPS (Resources and REST API)</td><td colspan="2" rowspan="3">As for 5.3.1Rainbow Desktop and Web clients, see 5.3.1</td></tr>
<tr><td>Web Sockets (XMPP)</td></tr>
<tr><td>ICE/TURN(s)</td></tr>
<tr><td>DTLS-SRTP to Rainbow clients thru Internet</td><td>As for 5.3.1Rainbow Desktop and Web clients, see 5.3.1</td><td>Not supported through the proxy, <u>UDP port 3478 to Rainbow TURN servers has to be opened</u></td></tr>
</table>

5.4 Domain used

Rainbow cloud services use the following domains:

Used by	Purpose	Domains
Rainbow Clients	Resources (website, images, client package, Agent package, …)	web.openrainbow.com cdn.openrainbow.com

Used by	Purpose	Domains
Rainbow Clients/SDK	REST API	openrainbow.com
Rainbow Clients/SDK	XMPP over Secured WebSockets	openrainbow.com
Rainbow Clients/SDK	STUN/TURN	turn-eu1.openrainbow.com (France), turn-eu2.openrainbow.com (Germany), turn-na1.openrainbow.com (Canada), turn-as1.openrainbow.com (Singapore), turn-oc1.openrainbow.com (Australia)
PBX agent	PBX connection to Rainbow	agent.openrainbow.com
WebRTC GW	PBX media connection to Rainbow	As for Rainbow clients
Node Cli and SDKs in development mode	Sandbox connections for applications development tests	sandbox.openrainbow.com web-sandbox.openrainbow.com

5.5 Bandwidth requirement

5.5.1 WebRTC

Rainbow WebRTC communication currently rely on the following codecs:

WebRTC P2P:

- OPUS for audio
- VP8 or H.264 for Video and Screen Sharing,

WebRTC Conference:

- OPUS for audio
- VP8 for Video and Screen Sharing

WebRTC GW

- G711 or G722 for audio

WebRTC codecs are able to dynamically throttle both their resolution and bitrates, depending on network performance observed. Peer to peer (P2P) WebRTC communications maximal resolution is 720p. In a WebRTC conference, maximal resolution is 480p. The following table provides bandwidth requirement per media:

Media Type	Maximal Bandwidth	Average Bandwidth	Lowest Bandwidth	Comment
Audio	100 kbps	80kbps	15kbps	

Video	1.2 Mbps (P2P - 720p) 500 kbps (Conference - 480p)	Depends on video resolution / quality
Screen Sharing	15 – 1500 kbps (P2P) 15 – 800 kbps (Conference)	Depends on screen motion

Note: In a WebRTC conference a participant can receive up to 5 video media streams (4 video from other participants and 1 desktop sharing). For a given participant, the maximum upstream bandwidth is 1.3 Mbps (0.5 + 0.8 Mbps) if the participant shares his video and his desktop. The maximum downstream is 2.8 Mbps (4 x 0.5 + 0.8 Mbps).

5.6 Configuration of border elements in enterprise

To allow Rainbow to operate properly, border elements like DNS, HTTP Proxy or Firewall must be configured to allow accessing domains and protocols listed in the table chapter 5.3 and 5.4.

Regarding range of ports, if you are in one of the two following cases, there is no specific port range to configure in border elements:

If an HTTP Proxy is configured, all the traffic will be directed through the proxy (HTTP Connect method),

If no HTTP Proxy is configured, that probably means that any direct traffic over arbitrary TCP/UDP ports is allowed.

If you are in the second case, but traffic is filtered by a border element, please, check chapter 5.3 and 5.4.

In case Deep Packet Inspection is in place on the network, some exceptions might have to be configured according to the Note of 5.3.1.

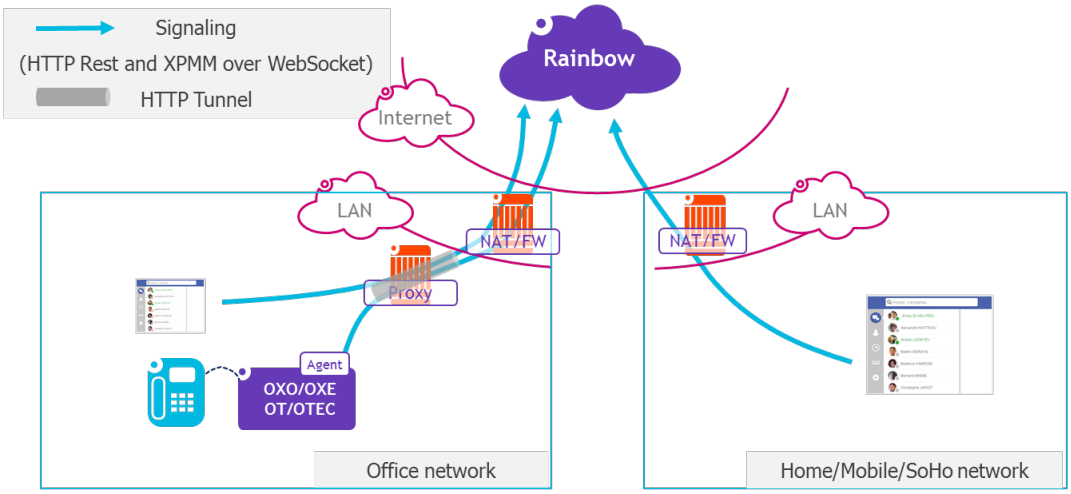
5.7 Connections illustrated

5.7.1 Signaling

For signaling, HTTPS/REST and Secured Web Sockets protocols are used.

If a HTTP Proxy is configured, HTTP Proxy is used. In such case, HTTP Proxy must support Secured WebSocket (HTTP Upgrade to switch to wss protocol).

Signaling



5.7.2 WebRTC/Media

ICE (Internet Connectivity Establishment) procedure and STUN/TURN protocols are used to dynamically determine how the media will be routed between two Rainbow clients.

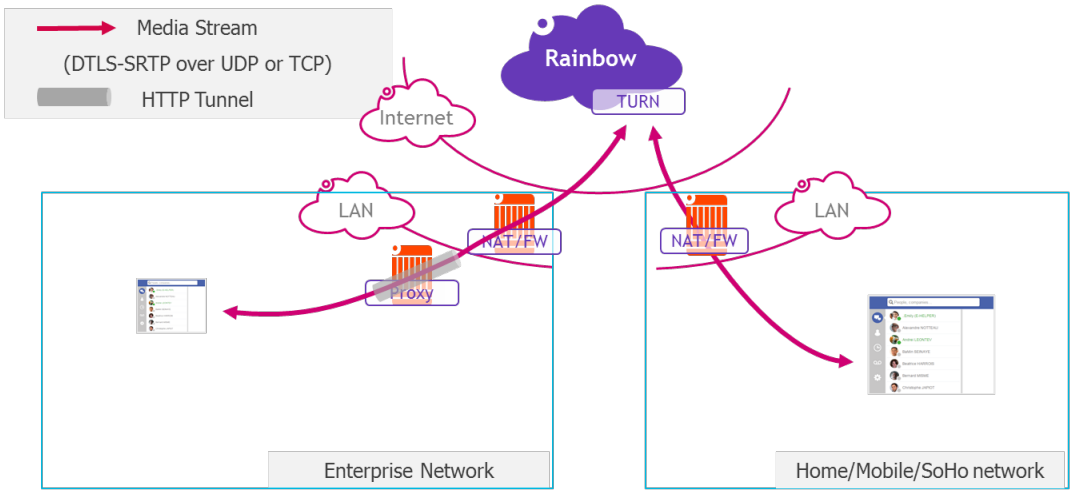
Basically, when a WebRTC communication takes place, client proceeds to the following steps:

- For each client, gather candidates addresses (A candidate is a transport address, a combination of IP address and port for a particular transport protocol, allocated on local interface and on TURN server in case of TURN is required, local interface are for example wired Ethernet interface or WiFi interface for a PC),
- Exchange candidates with the peer client,
- Check connectivity for candidates between both clients and select a working peer of candidates.

Example : P2P WebRTC (Enterprise network)

When it comes to enterprise network, most likely an HTTP Proxy will be there. In such case, HTTP Tunnel and TURN are used to establish the media path.

P2P WebRTC (Enterprise Network)



Note: to simplify figures, only one TURN server is illustrated. For a P2P communication, depending on geography and network performance, up to two TURN servers could be used to establish a communication.

6 Limitations, Restrictions and workarounds

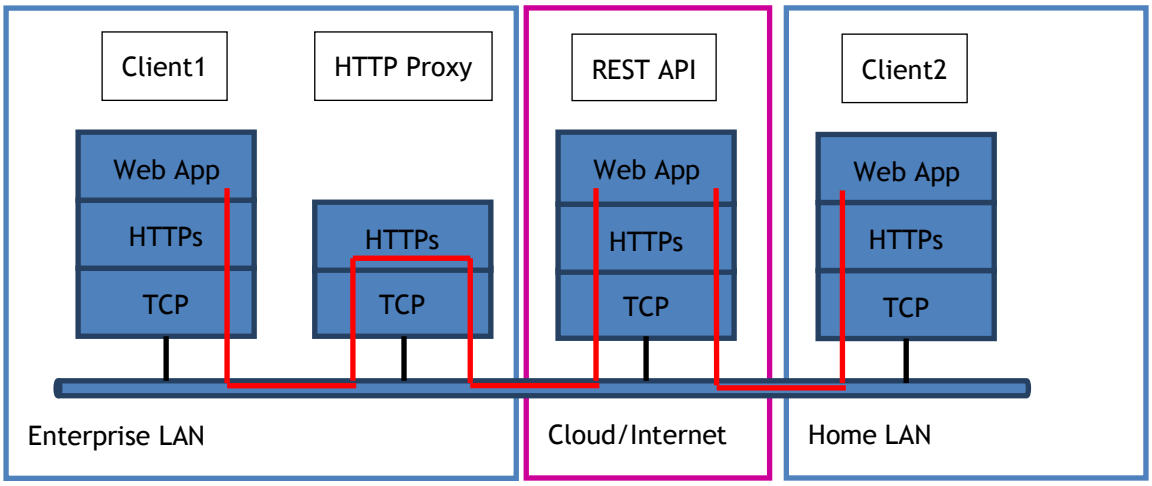
6.1 HTTP Proxy

If an HTTP Proxy is configured and/or selected at OS level (fixed configuration, auto-detect, script, …), by design, Browsers and Rainbow desktop client as well will always rely on this HTTP Proxy to reach Rainbow cloud services (for all protocols used, including HTTPS/REST, XMPP over Secured Web Sockets and TURN). It could append that the proxy in place in enterprise is not able or is not willing to pass-through some protocols (TURN for example). Capability and willingness of the HTTP Proxy must be checked by IS/IT team in case of trouble to start Rainbow clients or to establish Audio/Video call between Enterprise Lan and external users.

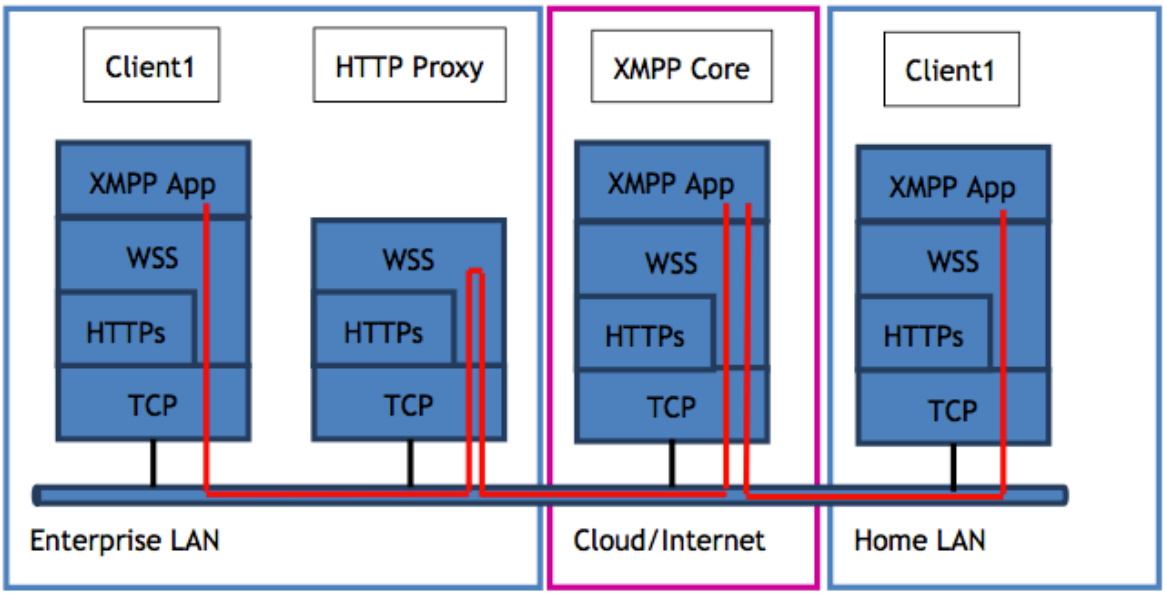
7 Annexes: Detailed call-flow of HTTPS/REST, XMPP and ICE connections

The following figures illustrate a case where Rainbow Client1 make an Audio/Video call to Rainbow Client2. Rainbow Client1 is in an enterprise environment with NAT/FW and HTTP Proxy border elements. Rainbow Client2 is in a Home network with simple NAT/FW as border element (home router/box).

Network layers (Rest API):

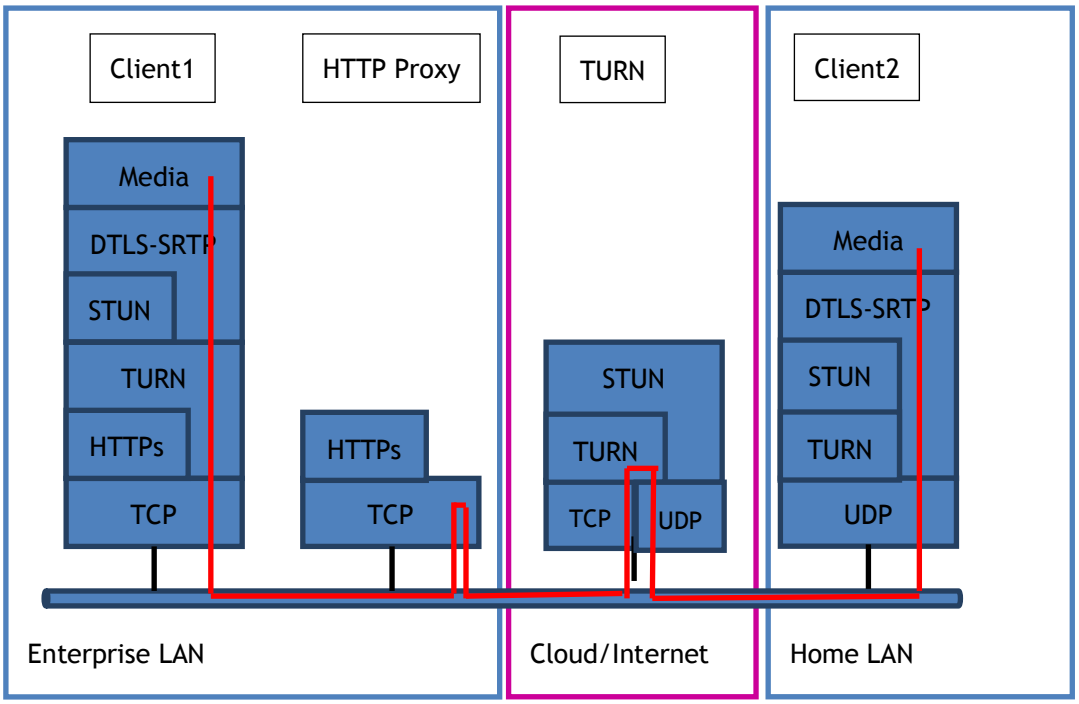


Network layers (XMPP):



HTTPs is used to setup WSS protocol (RFC6455)

Network layers (Media):



HTTPs is used to setup HTTP Tunnel to TURN server

STUN is a protocol helper to check connectivity

TURN is a protocol helper to pass-through NAT/FW/HTTP Proxy

Call Flow:

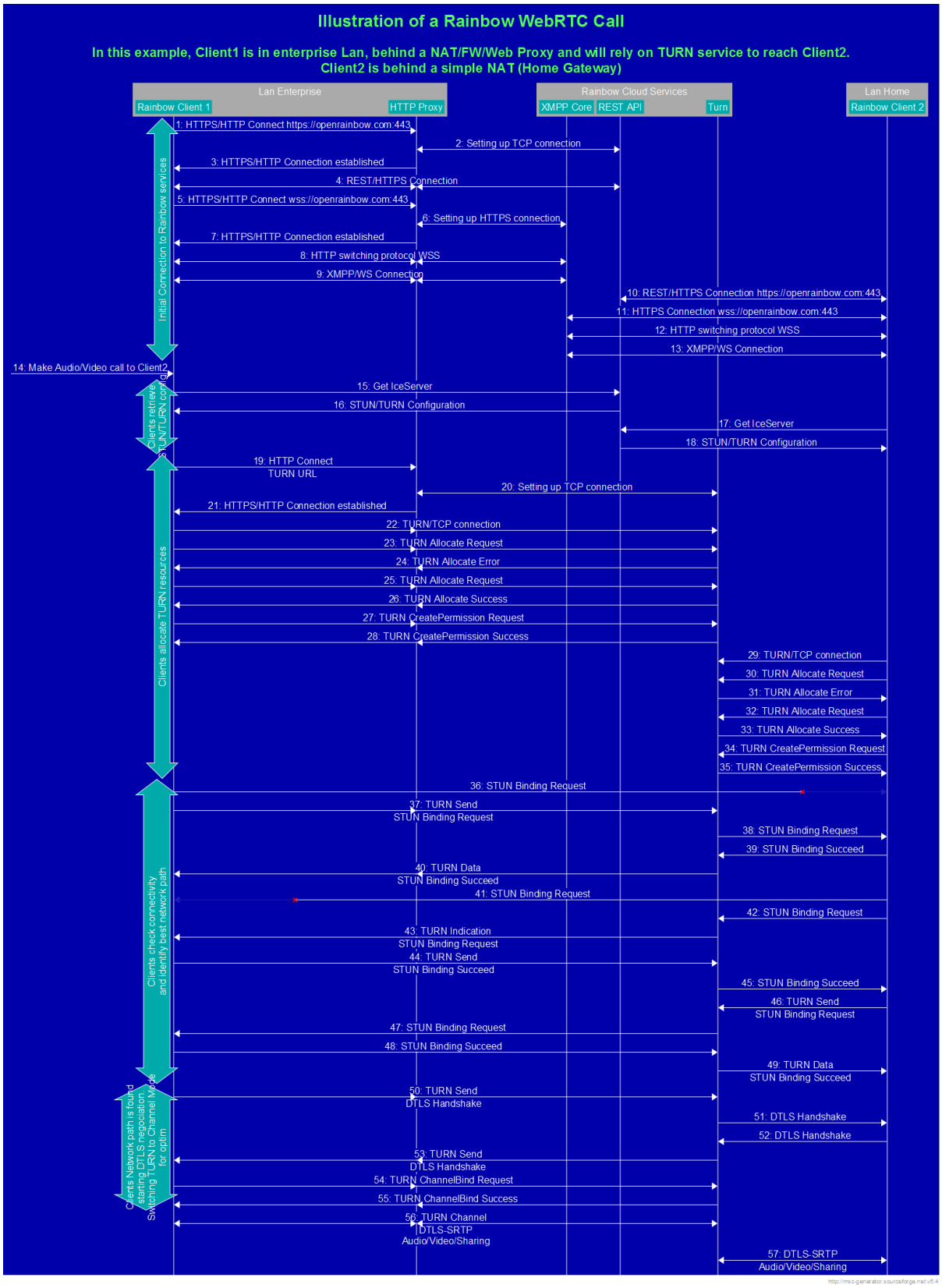


Illustration of a Rainbow WebRTC Call
In this example, Client1 is in enterprise Lan, behind a NAT/FW/Web Proxy and will rely on TURN service to reach Client2.
Client2 is behind a simple NAT (Home Gateway)
Lan Enterprise
Rainbow Client 1
HTTP Proxy
Rainbow Cloud Services
XMPP Core
REST API
Turn
Lan Home
Rainbow Client 2
Initial Connection to Rainbow services
1: HTTPS/HTTP Connect https://openrainbow.com:443
2: Setting up TCP connection
3: HTTPS/HTTP Connection established
4: REST/HTTPS Connection
5: HTTPS/HTTP Connect wss://openrainbow.com:443
6: Setting up HTTPS connection
7: HTTPS/HTTP Connection established
8: HTTP switching protocol WSS
9: XMPP/WS Connection
10: REST/HTTPS Connection https://openrainbow.com:443
11: HTTPS Connection wss://openrainbow.com:443
12: HTTP switching protocol WSS
13: XMPP/WS Connection
14: Make Audio/Video call to Client2
Clients retrieve STUN/TURN config
15: Get IceServer
16: STUN/TURN Configuration
17: Get IceServer
18: STUN/TURN Configuration
Clients allocate TURN resources
19: HTTP Connect
TURN URL
20: Setting up TCP connection
21: HTTPS/HTTP Connection established
22: TURN/TCP connection
23: TURN Allocate Request
24: TURN Allocate Error
25: TURN Allocate Request
26: TURN Allocate Success
27: TURN CreatePermission Request
28: TURN CreatePermission Success
29: TURN/TCP connection
30: TURN Allocate Request
31: TURN Allocate Error
32: TURN Allocate Request
33: TURN Allocate Success
34: TURN CreatePermission Request
35: TURN CreatePermission Success
Clients check connectivity and identify best network path
36: STUN Binding Request
37: TURN Send
STUN Binding Request
38: STUN Binding Request
39: STUN Binding Succeed
40: TURN Data
STUN Binding Succeed
41: STUN Binding Request
42: STUN Binding Request
43: TURN Indication
STUN Binding Request
44: TURN Send
STUN Binding Succeed
45: STUN Binding Succeed
46: TURN Send
STUN Binding Request
47: STUN Binding Request
48: STUN Binding Succeed
49: TURN Data
STUN Binding Succeed
Clients Network path is found starting DTLS negociation Switching TURN to Channel Mode for optim
50: TURN Send
DTLS Handshake
51: DTLS Handshake
52: DTLS Handshake
53: TURN Send
DTLS Handshake
54: TURN ChannelBind Request
55: TURN ChannelBind Success
56: TURN Channel
DTLS-SRTP
Audio/Video/Sharing
57: DTLS-SRTP
Audio/Video/Sharing

- Steps 1-9: Client1, behind HTTP Proxy, establish HTTPs sockets and Secured Web Sockets (for XMPP) through the HTTP Proxy.
- Steps 10-13: Client2, behind simple NAT, establish regular HTTPs sockets and Secured Web Sockets (for XMPP) directly to Rainbow Cloud Services.
- Step 14: Client1 make an Audio/Video call to Client2
- Steps 15-18: Client1 and Client2 retrieves ICE configuration (list of TURN servers)
- Steps 19-28: Client1 allocates TURN resources (through HTTP Proxy)
- Steps 29-35: Client2 allocates TURN resources (direct connection to TURN)
- Steps 36-49: Client1 and Client2 perform STUN connectivity check for various options (in this example, we illustrate the case where direct STUN connectivity check would failed)
- Steps 50-53: Client1 and Client2 have found a network path through the TURN server. They start to initiate the DTLS-SRTP handshake.
- Steps 54-55: Client1 ask TURN to switch to Channel mode to optimize network bandwidth (reduce TURN header overhead).
- Steps 56-57: DTLS-SRTP is established, Audio/Video media starts to flow between Client1 and Client2

End of Document