

Overview on CMS presentation sharing with Skype for Business using Expressway-E as TURN server - Cisco

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Components Used](#)

[Background Information](#)

[Scenario](#)

[Network Diagram](#)

[Working with packet captures](#)

[Wireshark filter](#)

[Looking for STUN packets in TCP payload](#)

[Using Wireshark to decode MSSTUN messages](#)

[Troubleshoot](#)

[User is not able to share](#)

Introduction

This document describes a detailed view on the TCP TURN message exchange between the CMS, Expressway and Skype for Business components.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Expressway server
- CMS (Cisco Meeting Server)
- Skype for Business (previously Lync) server

Components Used

The information in this document is based on these software and hardware versions:

- Expressway 8.9

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

Expressway version X8.9 introduced support for TCP TURN, allowing presentation sharing calls between CMS and Skype for Business (Lync) where CMS would use Expressway-E as its TURN server. Content media from the Skype client is then expected to flow towards Expressway-E, which then forwards it to CMS on premise.

This document is supposed to provide a detailed view on the TCP TURN message exchange between all the components to help troubleshoot the potential issues. It does not explain the fundamentals of TURN or the use of UDP TURN for regular audio or video call.

Tip: The TCP TURN is an extension to TURN documented under the following [RFC6062](#).

This document focuses on the TCP part, which is unique for Skype presentation sharing calls, and adds extra complexity to the classic TURN operation.

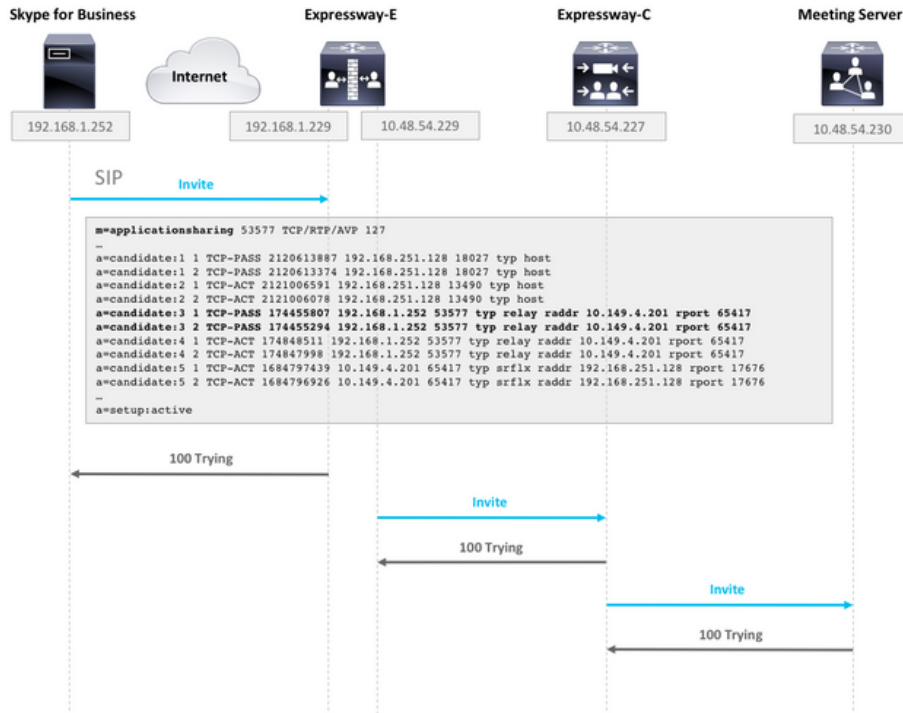
Scenario

In the test lab scenario described in this document, we have Skype client communicating to CMS over Skype Edge server, Expressway-E and Expressway-C. Expressway-E is configured in CMS as a TURN server. Additionally, the Skype client has no IP connectivity to the Expressway-E server, so we expect the only working media path to be over the Skype Edge towards the Expressway-E server.

Network Diagram

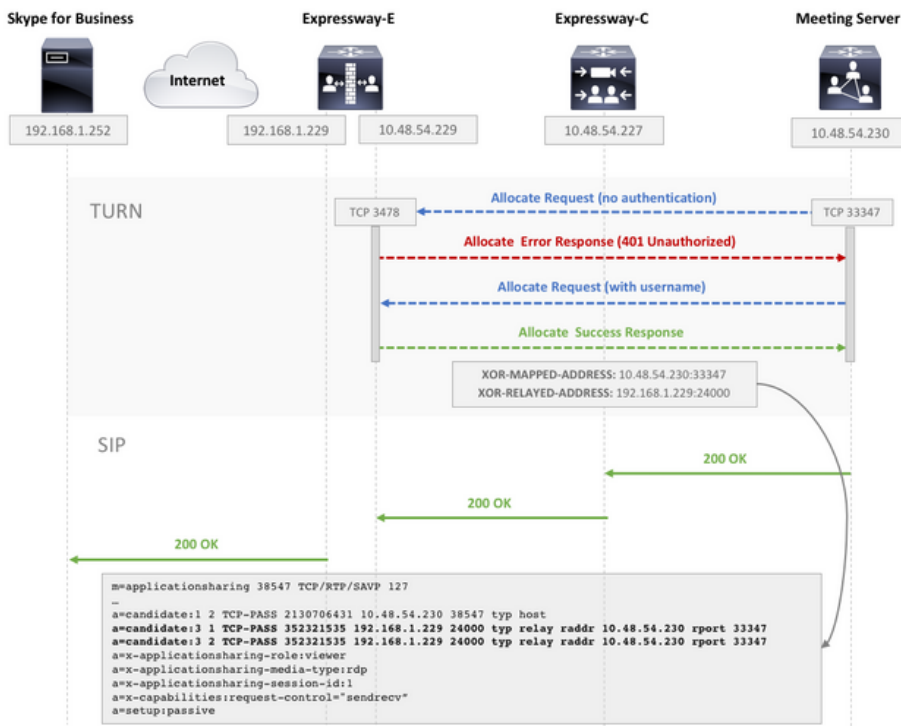
The following image shows the new **INVITE** with **m=applicationsharing** is sent from Skype to initiate the presentation sharing.

(it does not show the initial audio and video call invites, which are already negotiated at this stage):



SDP from Skype contains remote ICE candidates. Note the m=applicationsharing that indicates this is a call for sharing presentation. It will have a different SIP call-id than the initial audio/video call.

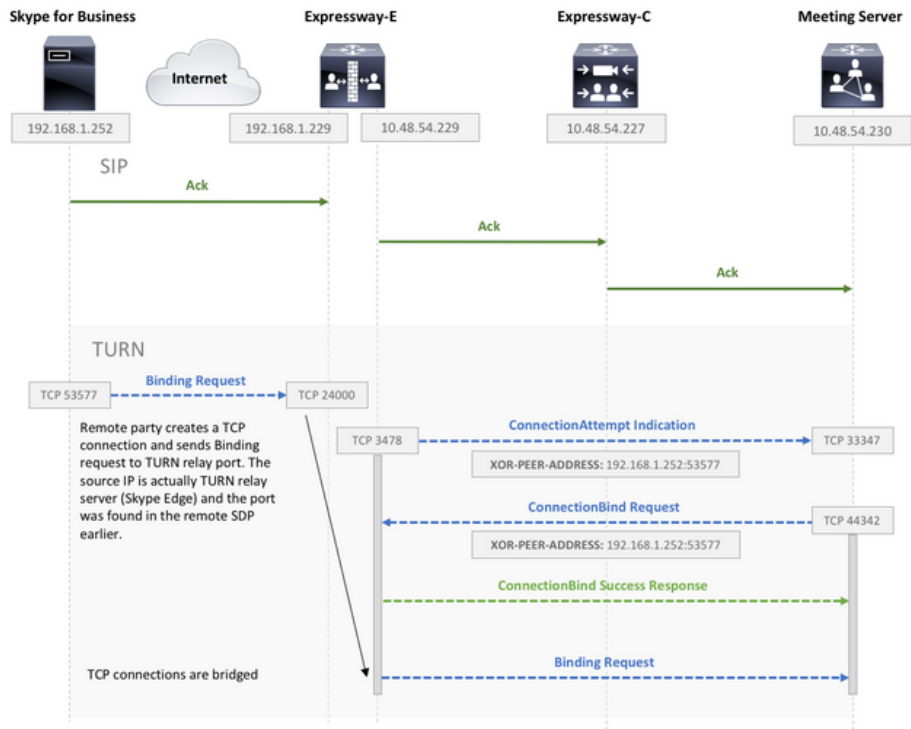
After CMS receives the call, it will reach out to its TURN server (Expressway-E) to get its own TURN relay candidates.



CMS make TCP connection to TURN server for TURN relay candidate allocation.

TURN server sends Allocate Success Response which contains the TURN relay candidate.

CMS adds TURN relay candidate to SDP in its 200 OK SIP response.



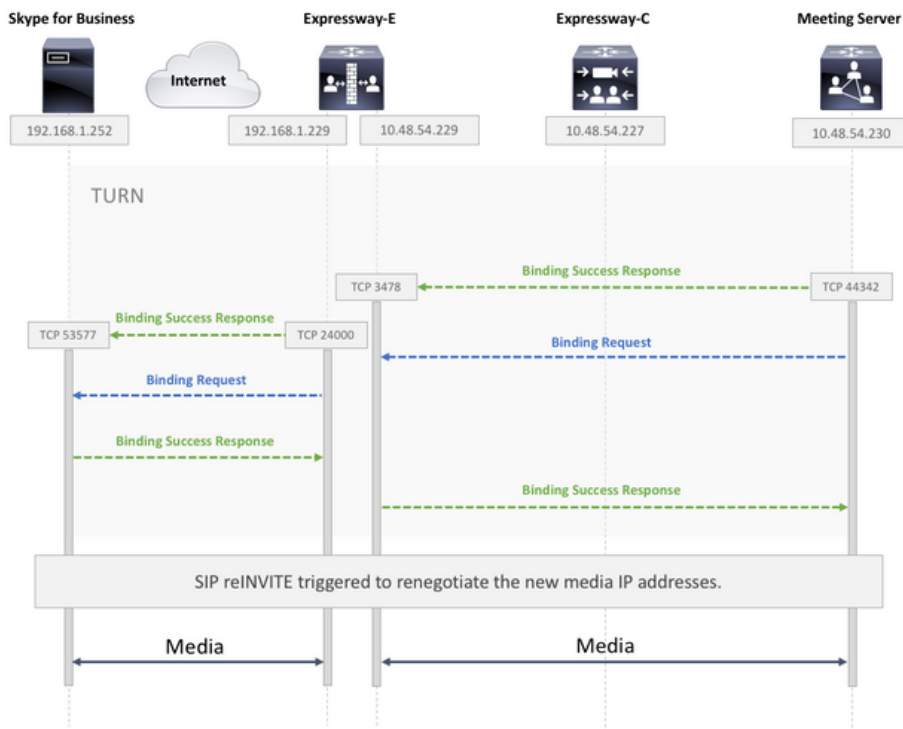
SIP dialog finishes with the ACK

TURN server notifies the TURN client about a connection made to the relay candidate address (XOR-PEER-ADDRESS attribute). This is done over the same TCP connection where Allocate Request was sent.

TURN client (CMS) creates a new TCP connection to TURN server to request the remote XOR-PEER-ADDRESS connection to be bridged to this new TCP connection.

TURN server confirms. From now on the traffic from remote peer 192.168.1.252:53577 hitting port 24000 on TURN server will be forwarded over this TCP connection to CMS.

Binding request from Skype is sent to CMS.



Bidirectional Binding Requests and Binding Success Responses are required for this candidate pair to be considered valid.

After Binding Success Response was received in both directions, there will be SIP reINVITE dialog between CMS and Skype to establish the new media route.

Working with packet captures

Wireshark filter

In some situations, in order to get quick overview of the STUN communication, it may be enough

to set a Wireshark filter as **tcp and stun**:

No.	Time	Source	Destination	Protocol	Length	Info
2394	2017-08-17 08:03:51.966175	10.48.54.230	10.48.54.229	STUN	98	Allocate Request TCP lifetime: 600
2397	2017-08-17 08:03:51.968443	10.48.54.229	10.48.54.230	STUN	230	Allocate Error Response with nonce realm: TANDBERG lifetime: 600
2399	2017-08-17 08:03:51.968947	10.48.54.230	10.48.54.229	STUN	202	Allocate Request user: turn realm: TANDBERG with nonce TCP
2427	2017-08-17 08:03:52.084888	10.48.54.229	10.48.54.230	STUN	166	Allocate Success Response lifetime: 600 XOR-MAPPED-ADDRESS: 10.48.
2428	2017-08-17 08:03:52.085424	10.48.54.230	10.48.54.229	STUN	190	Refresh Request user: turn realm: TANDBERG with nonce lifetime: 6.
2447	2017-08-17 08:03:52.172733	10.48.54.229	10.48.54.230	STUN	142	Refresh Success Response lifetime: 600
2526	2017-08-17 08:03:52.568097	10.48.54.229	10.48.54.230	STUN	154	ConnectionAttempt Indication XOR-PEER-ADDRESS: 192.168.1.252:53577
2540	2017-08-17 08:03:52.618906	10.48.54.230	10.48.54.229	STUN	190	ConnectionBind Request user: turn realm: TANDBERG with nonce
2552	2017-08-17 08:03:52.673050	10.48.54.229	10.48.54.230	STUN	142	ConnectionBind Success Response
3209	2017-08-17 08:03:57.084719	10.48.54.230	10.48.54.229	STUN	82	Binding Indication

Looking for STUN packets in TCP payload

Wireshark may not always decode the TCP communication as STUN.

You will have to filter out on the TCP port which is used for communication, look for TCP packets with **[PSH, ACK]** flag and investigate the TCP payload:

No.	Time	Source	Destination	Protocol	Length	Info
2596	2017-08-17 08:03:52.829644	10.48.54.229	10.48.54.230	TCP	144	3478->44342 [PSH, ACK] Seq=391 Ack=529 Win=31360 Len=90
2597	2017-08-17 08:03:52.829905	10.48.54.230	10.48.54.229	TCP	164	44342->3478 [PSH, ACK] Seq=529 Ack=481 Win=29312 Len=110
2608	2017-08-17 08:03:52.869391	10.48.54.229	10.48.54.230	TCP	54	3478->44342 [ACK] Seq=481 Ack=639 Win=31360 Len=0

Offset	Raw Data	ASCII
0000	00 0c 29 48 9e 5f 00 50 56 98 98 98 08 00 45 00	...H...P V....E.
0010	00 96 ba 17 40 00 40 06 fe 1f 0a 30 36 e6 0a 30	...@.@. ...06..0
0020	36 e5 ad 36 0d 96 f2 eb b4 ab 80 89 c7 5f 50 18	6..6.....P.
0030	00 e5 c7 82 00 00 00 6c 00 01 00 58 21 12 a4 42l ...X!..B
0040	a7 d4 2d 51 9e 4d 78 c5 93 81 95 21 00 25 00 00	..-Q.Mx. ...!%..
0050	00 24 00 04 6e ff ff ff 80 29 00 08 08 b2 67 4a	\$.n...)...gJ
0060	8b ee cd 68 00 06 00 0c 6c 30 4d 52 3a 41 6f 56	...h... l0MR:AoV
0070	79 00 00 00 00 54 00 04 33 00 00 00 00 70 00 04	y...T.. 3....p.
0080	00 00 00 02 00 08 00 14 1d a4 84 25 29 57 5b 38 (%)W[B
0090	e0 6b 72 ef 45 8c 3e 17 2b 65 c7 6c 80 28 00 04	.kr.E.>. +e.l.(..
00a0	ff 2f a7 18	/../

In the the image above the payload begins with data **00 6c 00 01**. The different values in the 3rd and 4th byte represent the following STUN packets:

00 01 - Binding Request

01 01 - Binding Success Response

In order for the STUN pair to work, there has to be one of each in each direction.

Using Wireshark to decode MSSTUN messages

Microsoft has made additions to the base IETF standards which are not recognised by Wireshark. You can install a plugin into Wireshark which will make these packet capture more readable.

More information on the plugin can be found [here](#).

Troubleshoot

This section provides information you can use in order to troubleshoot your configuration.

User is not able to share

- Check if the CMS logs contain the following entry: **ms-diagnostics-public: 21002;reason="Attendees cannot share in this conference";component="ASMCU"**
- Skype for Business Meetings are not setup to allow all to share by default. If you see the

above error, right click on the attendee from the Skype client and select **Make Presenter**