Let's go

#CiscoLive

The bridge to possible

Troubleshooting Video Endpoint Third Party Meeting Integrations

Tim Kratzke (Technical Leader)

BRKCOL-3016

Cisco Webex App

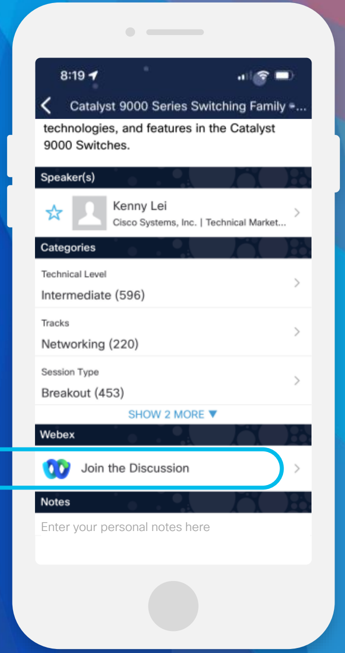
Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated
by the speaker until June 9, 2023.



https://ciscolive.ciscoevents.com/ciscolivebot/#BRKCOL-3016

Agenda

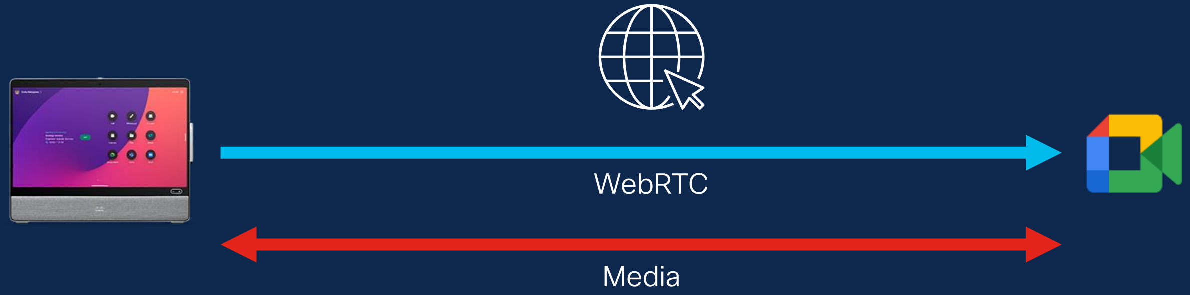
- 3rd Party Meeting Overview
- Log Collection Overview
- Troubleshooting WebRTC
- Troubleshooting MTR
- Conclusion

3rd Party Meetings, Where to Start?

- What service is being used?
- How does it connect?
- What features are available?

Google Meet

- WebRTC based
- Relies on the built in web browser capabilities



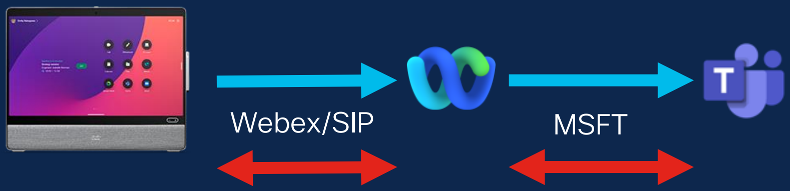
Microsoft Teams

- Three different options for interoperability with Microsoft Teams
  1. WebRTC
  2. VIMT / CVI
  3. Microsoft Teams Rooms (MTR)
- Options 1 and 2 work natively while MTR requires activating the device with Microsoft

Microsoft Teams

WebRTC

VIMT

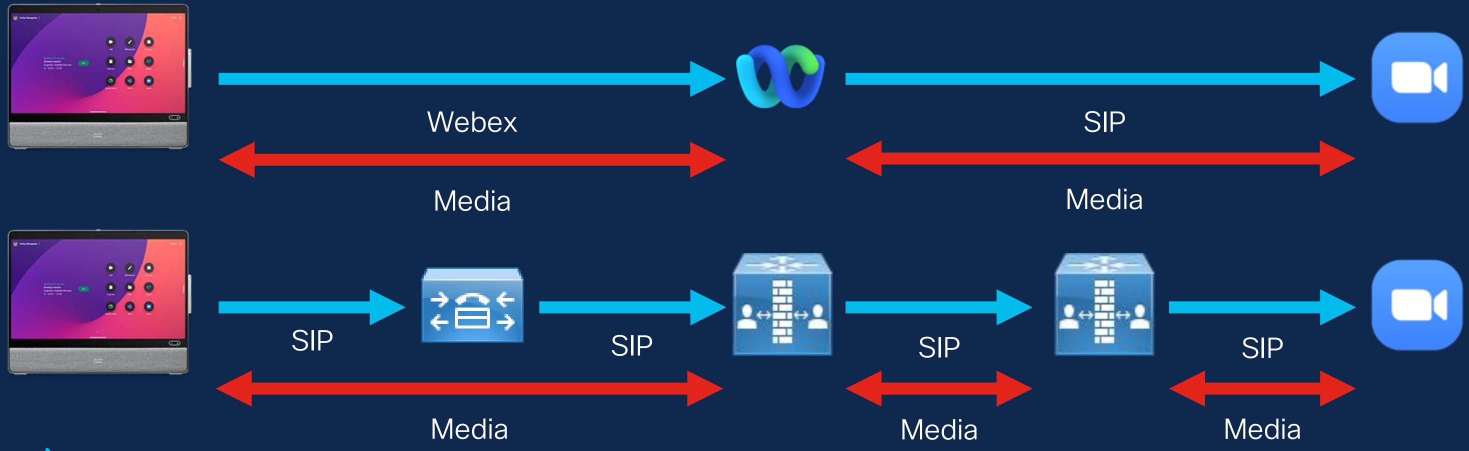


BRKCOL-2058: Troubleshooting the Cisco Webex Video Integration for Microsoft Teams

Zoom

BRKCOL-3004: Advanced Troubleshooting of Cisco Collaboration Video Endpoints

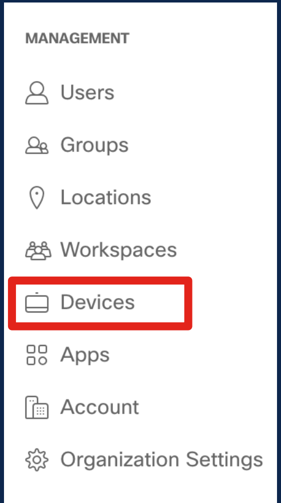
- Standard SIP call to ZoomCRC
- Video endpoint dials traditionally using SIP or Webex
- In call controls implemented using underlying DTMF commands

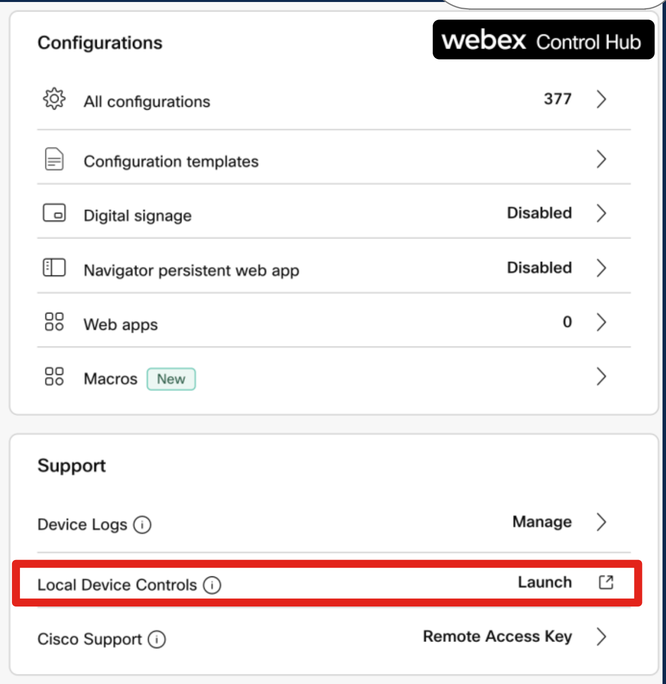


Agenda

- 3rd Party Meeting Overview
- Log Collection and Overview
- Troubleshooting WebRTC
- Troubleshooting MTR
- Conclusion

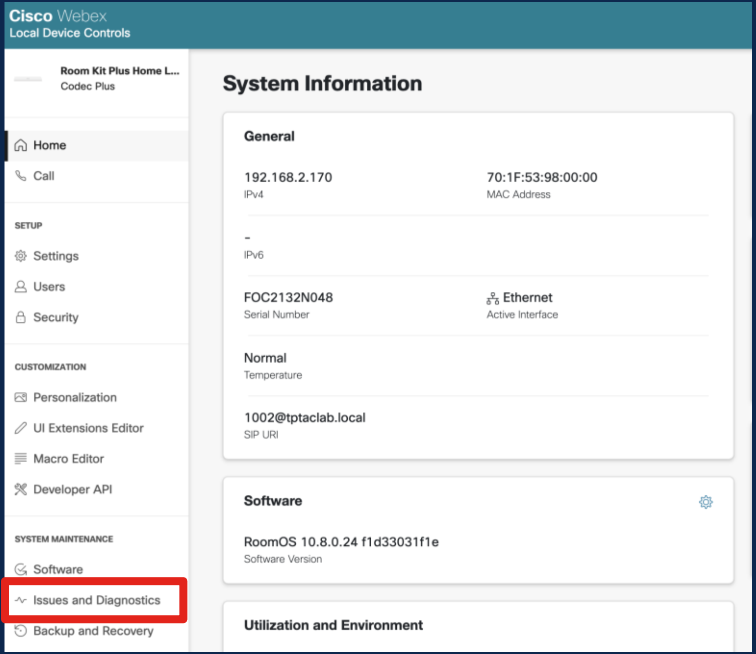
Log Collection

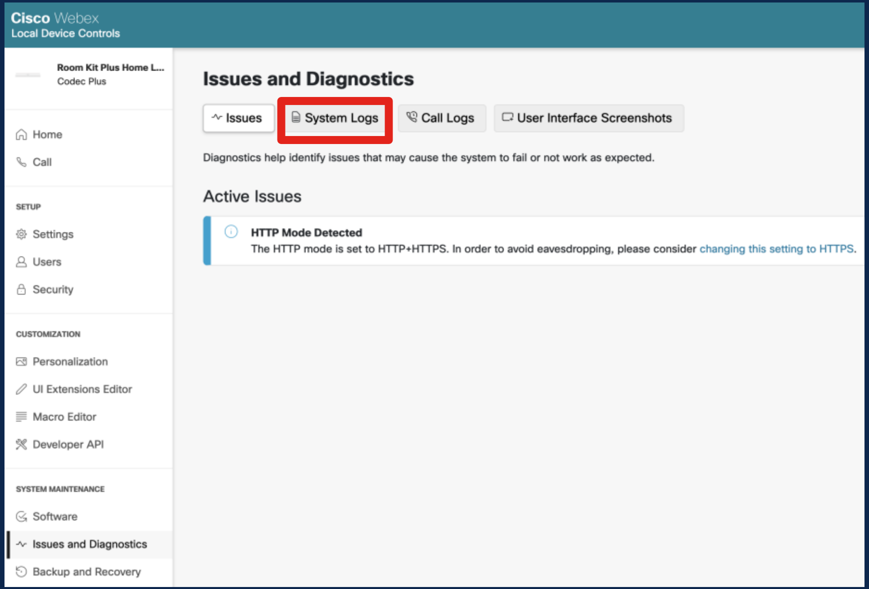




Devices → Local Device Controls

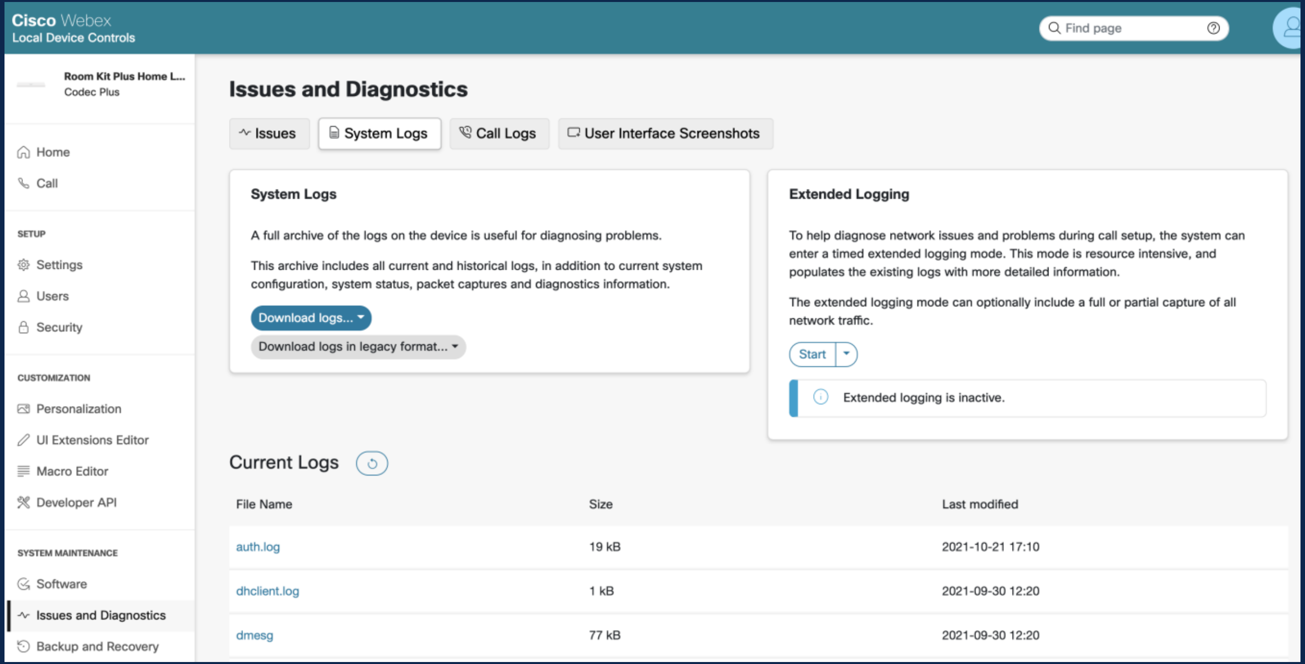
Log Collection





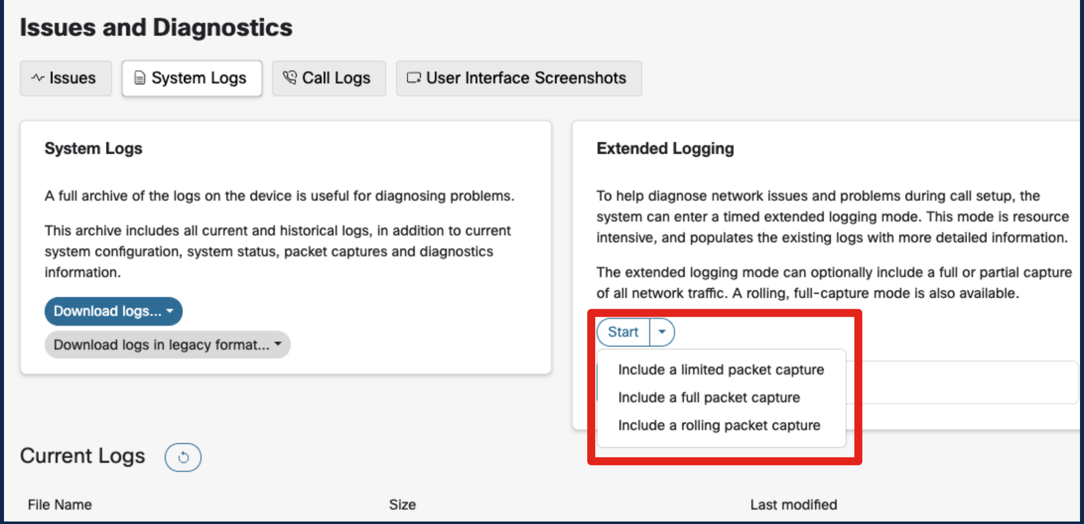
Issues and Diagnostics → System Logs

Log Collection



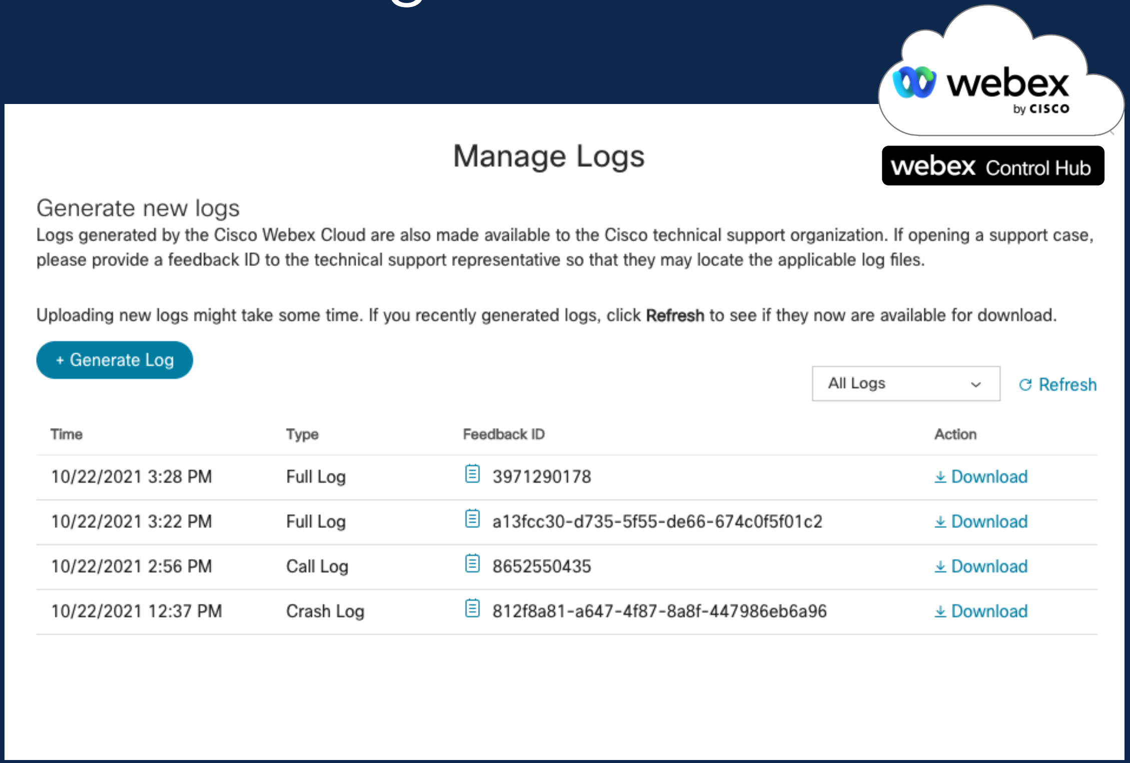
- Download Logs
- Start/Stop Extended Logging
- Browse Current Files

Extended Logging



- Extended Logging
  - Enables additional debugs
  - Lasts 10 minutes
- Include Limited Packet Capture
  - Starts pcap which will filter out RTP media
  - Lasts 10 minutes
- Include Full Packet Capture
  - Captures all traffic including RTP
  - Lasts 3 minutes
- Include Rolling Packet Capture
  - Pcaps will rotate based on file size
  - No time limit

Cloud Log Collection



- Send logs from device
- Request logs from control hub
- Automatic call logs
- Call-log contents
  - xconfig and xstatus
  - callhistory
  - all.log and main.log

Log Bundle Format

- config
- var
- wifi_extra_status.log
- bookings.txt
- callhistory.txt
- camera_presets.txt
- diagnostics.txt
- peripherals.txt
- xconfig.txt
- xstatus.txt

/var/log

- eventlog
- nginx
- peripherals
- dmesg
- osversion
- previous_version
- version.json
- auth.log
- dhclient.log
- hwmon.log
- kern.log
- kernerr.log
- messages.log
- network_state_logger.log
- performance_state_logger.log
- ptp.log
- system_state_logger.log
- wpa_supplicant.log
- vcinfo.txt
- platform-sanity-tests.xml

/eventlog

- airplay.log
- all.log
- application.log
- audioctrl.log
- audit.log
- blued.log
- camera.log
- diagnostics.log
- edid.log
- events.log
- headsetadministrator.log
- macros.log
- main.log
- metaproxy.log
- osd.log
- peripherals.log
- scriptbuffer.log
- softwareupgrade.log
- sysinfo.txt
- systemstate.txt

Agenda

- 3^{rd} Party Meeting Overview
- Log Collection Overview
- Troubleshooting WebRTC
- Troubleshooting MTR
- Conclusion

What is WebRTC?

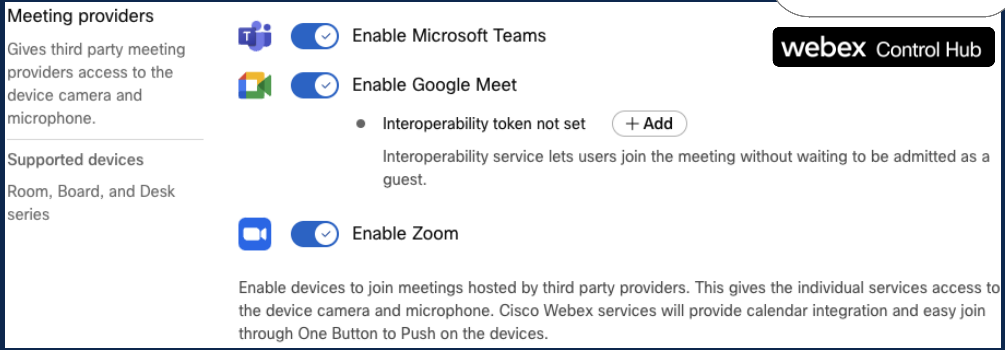
- HTML 5 specifications built to operate within modern web browsers and allow real-time communication between peers
- Set of javascript APIs to allow interaction with hardware such as cameras and microphones
- Eliminates the need for 3rd party plugins to use applications
- Applications are responsible for implementing the WebRTC standard to fit their specific needs

RoomOS and WebEngine

- WebEngine is the RoomOS browser framework based off of Chromium
- WebRTC applications run within the WebEngine and utilize its built in support
- Look and feel of these applications can vary significantly
- Media handling and limitations depending on the service handling the calls

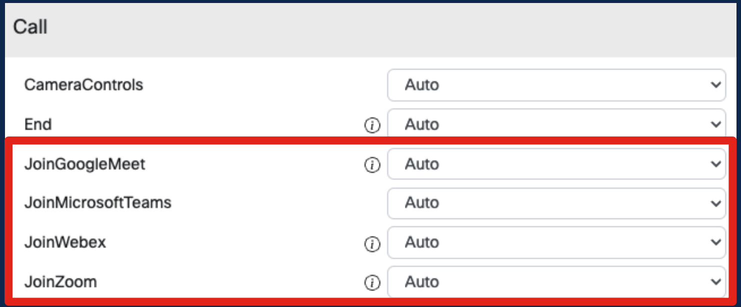
Third Party Meeting Settings

- Must be enabled in Control Hub under devices -> settings
- For WebRTC based calls, WebEngine must be enabled on the device
- For devices connected via Webex edge, ”Allow control hub to manage configuration” must also be enabled under device settings



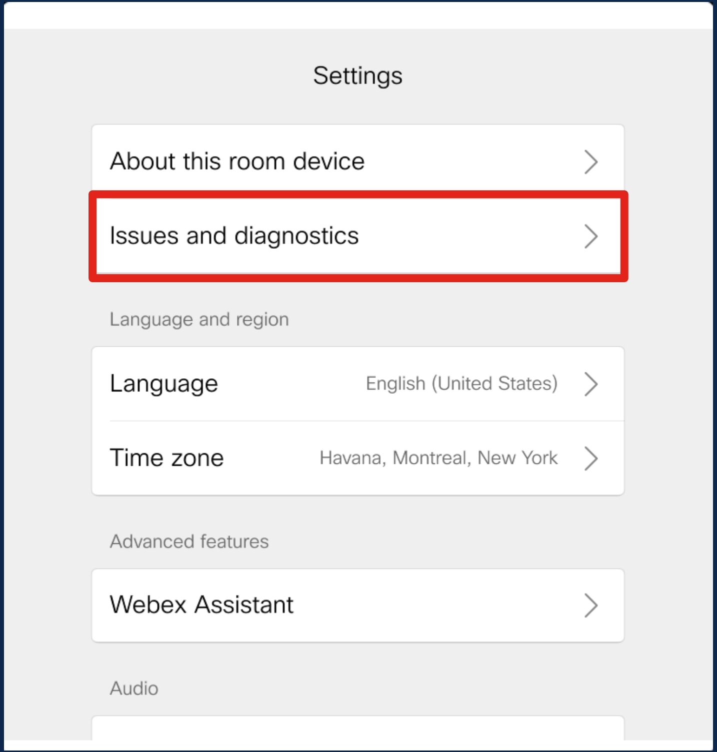
Third Party Meeting Settings

- Join shortcut icons can also be enabled/disabled on the device itself under UserInterface -> Features

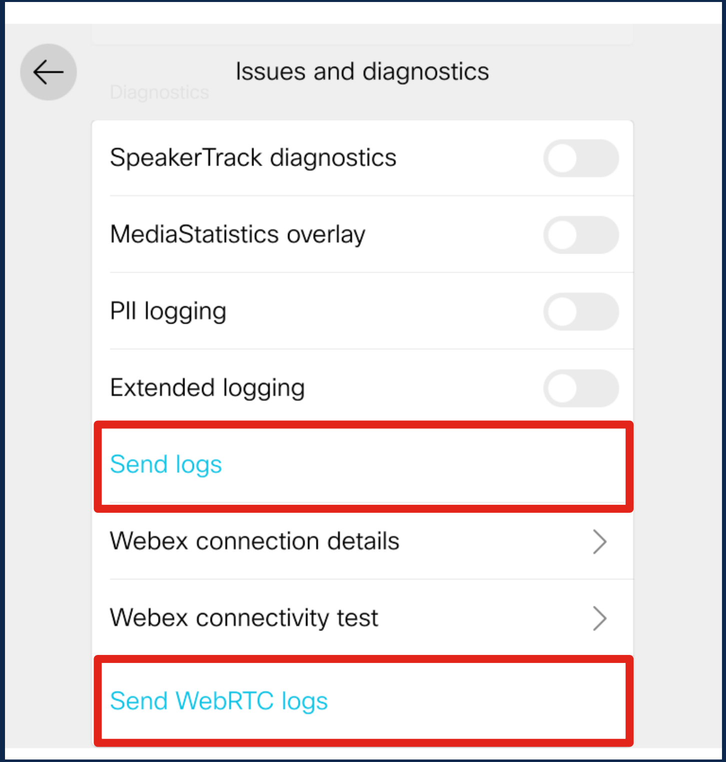


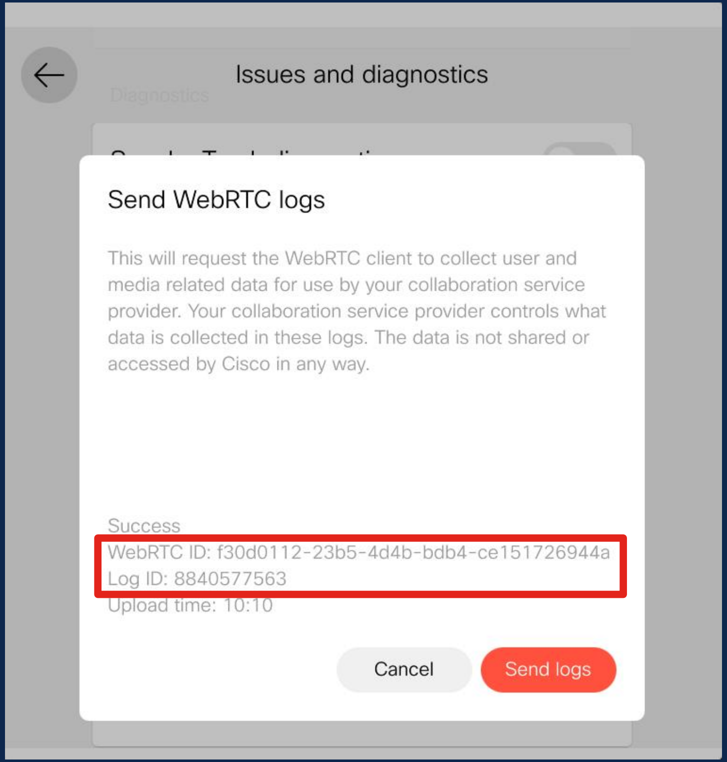
Send Logs to Provider

- Logs can be uploaded directly to control hub and sent to the WebRTC provider via the device diagnostics menu
- Cisco TAC does not have access to the WebRTC specific logs and the files that are collected are controlled by the meeting platform



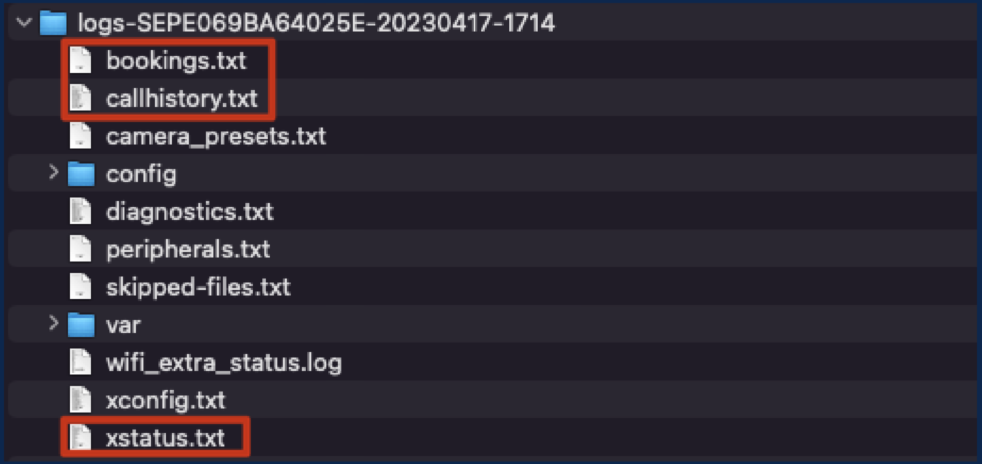
Send Logs to Provider





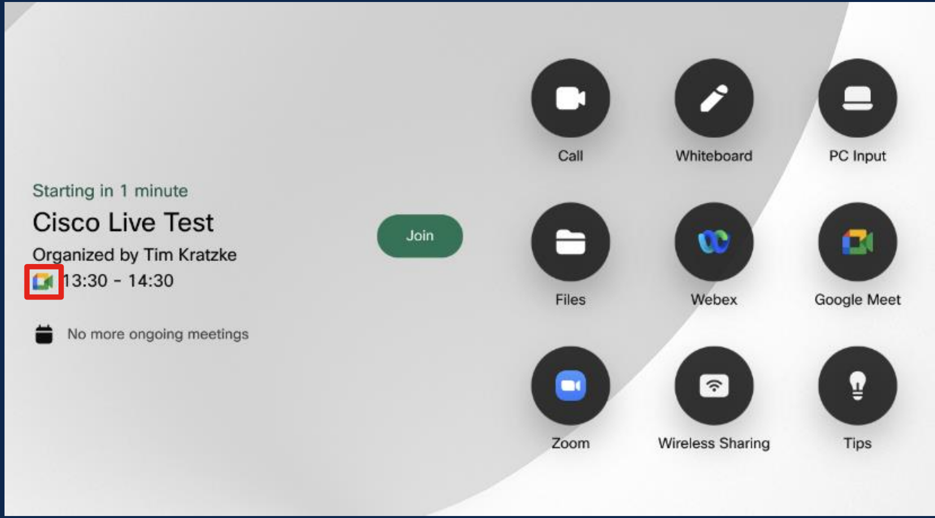
Log Troubleshooting Overview

- Log bundles can provide useful information to troubleshoot WebRTC calls depending on when they are collected
- Scheduling issues
- Connection problems
- General call info
- Media statistics and overview



Scheduling

- WebRTC based calls can be joined ad-hoc or scheduled just as a standard SIP or Webex meeting to leverage One Button to Push (OBTP)
- The bookings.txt file provides valuable information on how the call has been scheduled and can be used for troubleshooting purposes



Scheduling (booking.txt)

```
*r BookingsListResult Booking 1 Title: "Cisco Live Test"
...
*r BookingsListResult Booking 1 Organizer FirstName: "Tim Kratzke"
*r BookingsListResult Booking 1 Organizer LastName: ""
*r BookingsListResult Booking 1 Organizer Email: ""
*r BookingsListResult Booking 1 Organizer Id: "42174db8-07c1-4027-a9a1-784cfb56a8fb"
*r BookingsListResult Booking 1 Time StartTime: "2023-04-18T17:30:00Z"
*r BookingsListResult Booking 1 Time StartTimeBuffer: 300
*r BookingsListResult Booking 1 Time EndTime: "2023-04-18T18:30:00Z"
*r BookingsListResult Booking 1 Time EndTimeBuffer: 0
...
*r BookingsListResult Booking 1 BookingStatus: OK
*r BookingsListResult Booking 1 BookingStatusMessage: ""
*r BookingsListResult Booking 1 MeetingPlatform: "GoogleMeet"
*r BookingsListResult Booking 1 Cancellable: False
*r BookingsListResult Booking 1 Webex Enabled: False
...
*r BookingsListResult Booking 1 Encryption: BestEffort
*r BookingsListResult Booking 1 Recording: Disabled
*r BookingsListResult Booking 1 DialInfo Calls Call 1 Number: "https://meet.google.com/szp-mege-qtn?hs=224"
*r BookingsListResult Booking 1 DialInfo Calls Call 1 Protocol: WebRTC
*r BookingsListResult Booking 1 DialInfo ConnectMode: OBTP
```

Call History

- After a WebRTC call is ended, an entry is added to the callhistory.txt file with basic call information
- Notably absent from this is any media statistics or information commonly found for SIP or Webex calls
- This can provide a good overview of call connection information and times

Call History (callhistory.txt)

```
*r CallHistoryGetResult Entry 0 CorrelationId: "gzk-tvpj-xpe"
*r CallHistoryGetResult Entry 0 RemoteNumber: "https://meet.google.com/interopclient/gzk-tvpj-
xpe?correlationId=01b67814-1c5e-4098-8be6-b9a56cba81bd"
*r CallHistoryGetResult Entry 0 CallbackNumber: "https://meet.google.com/gzk-tvpj-xpe"
*r CallHistoryGetResult Entry 0 DisplayName: "gzk-tvpj-xpe"
*r CallHistoryGetResult Entry 0 Direction: Outgoing
*r CallHistoryGetResult Entry 0 Protocol: WebRTC
*r CallHistoryGetResult Entry 0 CallRate: 0
*r CallHistoryGetResult Entry 0 CallType: Video
*r CallHistoryGetResult Entry 0 VideoUsed: True
…
*r CallHistoryGetResult Entry 0 StartTime: "2023-04-17T17:13:01"
*r CallHistoryGetResult Entry 0 StartTimeUTC: "2023-04-17T21:13:01Z"
*r CallHistoryGetResult Entry 0 EndTime: "2023-04-17T17:14:39"
*r CallHistoryGetResult Entry 0 EndTimeUTC: "2023-04-17T21:14:39Z"
…
*r CallHistoryGetResult Entry 0 DisconnectCauseOrigin: Internal
*r CallHistoryGetResult Entry 0 DisconnectCauseType: LocalDisconnect
*r CallHistoryGetResult Entry 0 Video Incoming PacketLoss: "N/A"
*r CallHistoryGetResult Entry 0 Video Incoming PacketLossPercent: 0
*r CallHistoryGetResult Entry 0 Video Incoming MaxJitter: 0
```

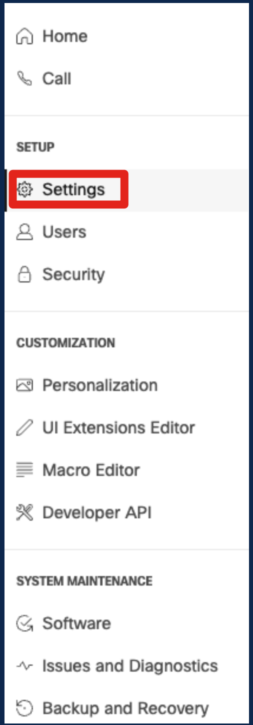
Media Troubleshooting

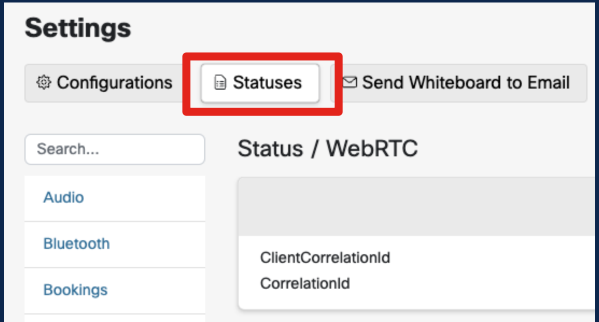
- While call history does not retain any media statistics, this info can be gathered by capturing logs while the call is in progress
- Overall media stats will be present in the xstatus.txt log output during a call
- This can be used in conjunction with a packet capture to take a detailed look at media information
- How media is handled depends on the application therefor can differ between vendor call solutions

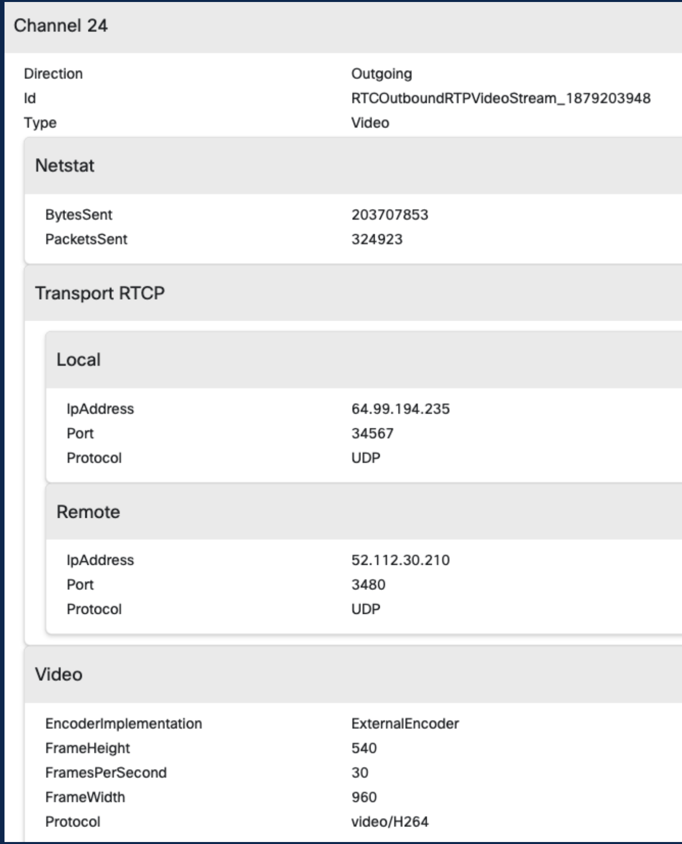
Media Troubleshooting (xstatus.txt)

```
*s WebRTC MediaChannels Client Channel 8 Direction: Incoming
*s WebRTC MediaChannels Client Channel 8 Id: "RTCInboundRTPVideoStream_1369139291"
*s WebRTC MediaChannels Client Channel 8 Netstat BytesReceived: 19478145
*s WebRTC MediaChannels Client Channel 8 Netstat JitterBufferDelay: 0
*s WebRTC MediaChannels Client Channel 8 Netstat JitterBufferEmittedCount: 0
*s WebRTC MediaChannels Client Channel 8 Netstat PacketsLost: 0
*s WebRTC MediaChannels Client Channel 8 Netstat PacketsReceived: 17605
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Local IpAddress: "64.99.194.235"
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Local Port: 53226
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Local Protocol: UDP
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Remote IpAddress: "142.250.82.113"
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Remote Port: 3478
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Remote Protocol: UDP
*s WebRTC MediaChannels Client Channel 8 Type: Video
*s WebRTC MediaChannels Client Channel 8 Video DecoderImplementation: "ExternalDecoder"
*s WebRTC MediaChannels Client Channel 8 Video FrameHeight: 720
*s WebRTC MediaChannels Client Channel 8 Video FrameWidth: 1280
*s WebRTC MediaChannels Client Channel 8 Video FramesDecoded: 1826
*s WebRTC MediaChannels Client Channel 8 Video FramesPerSecond: 30
*s WebRTC MediaChannels Client Channel 8 Video KeyFramesDecoded: 1
*s WebRTC MediaChannels Client Channel 8 Video Protocol: "video/VP8"
```

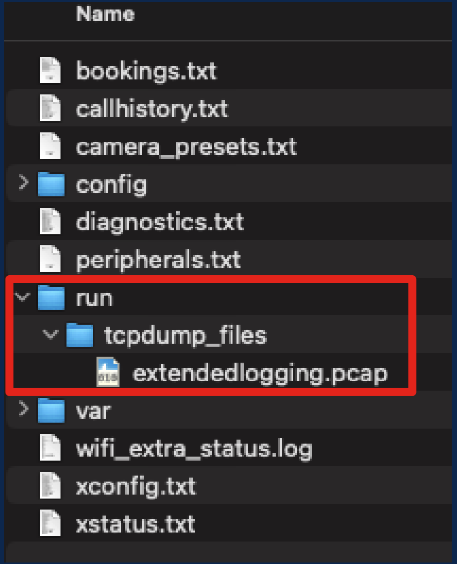
Media Troubleshooting (Web)





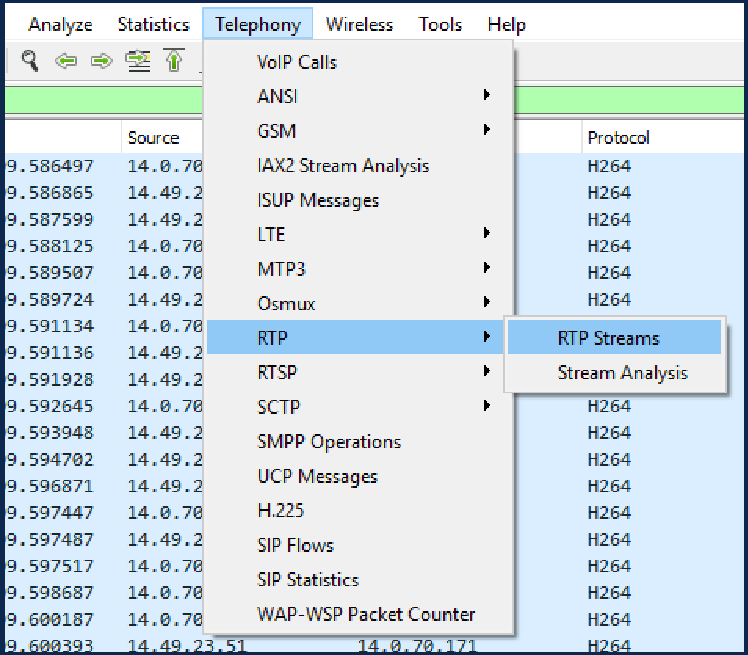


Packet Captures



- Packet captures are located in the “run” directory of log bundles
- In instances where the pcap file is too large it will not be included
- In those cases the pcap can be downloaded manually from the system logs page in the web interface

Packet Captures



Telephony → RTP → RTP Streams

Packet Captures (RTP Stream Overview)

Wireshark · RTP Streams · extendedlogging.pcap

Source Address	Source Port	Destination Address	Destination Port	SSRC	Start Time	Duration	Payload	Packets	Lost	Min Delta (ms)
142.250.82.113	3478	192.168.2.51	53226	0x519b685b	21.262999	56.74	RTPType-96	15504	0 (0.0%)	0.006000
142.250.82.113	3478	192.168.2.51	53226	0x63f9d036	21.262609	1.26	RTPType-97	890	0 (0.0%)	0.016000
142.250.82.113	3478	192.168.2.51	53226	0x1a0a	15.497057	62.40	RTPType-111	649	0 (0.0%)	0.015000
192.168.2.51	53226	142.250.82.113	3478	0xdce5b497	30.356936	0.00	RTPType-97	1	0 (0.0%)	-1.000000
192.168.2.51	53226	142.250.82.113	3478	0xc61f8231	21.504362	56.47	RTPType-96	13981	0 (0.0%)	0.060000
192.168.2.51	53226	142.250.82.113	3478	0xde2db488	21.336183	56.63	RTPType-96	5257	0 (0.0%)	0.064000
192.168.2.51	53226	142.250.82.113	3478	0x24635bce	21.327199	56.67	RTPType-96	1748	0 (0.0%)	0.116000
192.168.2.51	53226	142.250.82.113	3478	0x8345a013	18.738665	59.24	RTPType-111	1575	0 (0.0%)	10.281000

Inbound Media

Outbound Media

Logging (xstatus.txt)

```
*s WebRTC MediaChannels Client Channel 8 Direction: Incoming
*s WebRTC MediaChannels Client Channel 8 Id: "RTCInboundRTPVideoStream_1369139291"
*s WebRTC MediaChannels Client Channel 8 Netstat BytesReceived: 19478145
*s WebRTC MediaChannels Client Channel 8 Netstat JitterBufferDelay: 0
*s WebRTC MediaChannels Client Channel 8 Netstat JitterBufferEmittedCount: 0
*s WebRTC MediaChannels Client Channel 8 Netstat PacketsLost: 0
*s WebRTC MediaChannels Client Channel 8 Netstat PacketsReceived: 17605
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Local IpAddress: "64.99.194.235"
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Local Port: 53226
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Local Protocol: UDP
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Remote IpAddress: "142.250.82.113"
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Remote Port: 3478
*s WebRTC MediaChannels Client Channel 8 Transport RTCP Remote Protocol: UDP
*s WebRTC MediaChannels Client Channel 8 Type: Video
*s WebRTC MediaChannels Client Channel 8 Video DecoderImplementation: "ExternalDecoder"
*s WebRTC MediaChannels Client Channel 8 Video FrameHeight: 720
*s WebRTC MediaChannels Client Channel 8 Video FrameWidth: 1280
*s WebRTC MediaChannels Client Channel 8 Video FramesDecoded: 1826
*s WebRTC MediaChannels Client Channel 8 Video FramesPerSecond: 30
*s WebRTC MediaChannels Client Channel 8 Video KeyFramesDecoded: 1
*s WebRTC MediaChannels Client Channel 8 Video Protocol: "video/VP8"
```

Packet Captures

We can determine the SSRC by converting the channel ID from decimal to hex

```
WebRTC MediaChannels Client Channel 8 Id: "RTCInboundRTPVideoStream_1369139291"
```

```
1369139291 -> 0x519b685b
```

Wireshark · RTP Streams · extendedlogging.pcap

Source Address	Source Port	Destination Address	Destination Port	SSRC	Start Time	Duration	Payload	Packets	Lost	Min Delta (ms)
142.250.82.113	3478	192.168.2.51	53226	0x519b685b	21.262999	56.74	RTPType-96	15504	0 (0.0%)	0.006000
142.250.82.113	3478	192.168.2.51	53226	0x63f9d036	21.262609	1.26	RTPType-97	890	0 (0.0%)	0.016000
142.250.82.113	3478	192.168.2.51	53226	0x1a0a	15.497057	62.40	RTPType-111	649	0 (0.0%)	0.015000
192.168.2.51	53226	142.250.82.113	3478	0xdce5b497	30.356936	0.00	RTPType-97	1	0 (0.0%)	-1.000000
192.168.2.51	53226	142.250.82.113	3478	0xc61f8231	21.504362	56.47	RTPType-96	13981	0 (0.0%)	0.060000
192.168.2.51	53226	142.250.82.113	3478	0xde2db488	21.336183	56.63	RTPType-96	5257	0 (0.0%)	0.064000
192.168.2.51	53226	142.250.82.113	3478	0x24635bce	21.327199	56.67	RTPType-96	1748	0 (0.0%)	0.116000
192.168.2.51	53226	142.250.82.113	3478	0x8345a013	18.738665	59.24	RTPType-111	1575	0 (0.0%)	10.281000

Logging (osd.log)

- Relevant call setup and in call logging can be seen in the osd.log files for WebRTC
- Can be used to track media changes or see limited media information after a call
- Good when checking for error messages related to connection failures

```
gui[3781]: Io2: (S) xcom "WebRTC/Update State: \"connecting\"" ID=[ 166 ]
…
gui[3781]: Io2: (S) xcom "WebRTC/Update State: \"lobby\"" ID=[ 168 ]
…
gui[3781]: Io2: (S) xcom "WebRTC/Update State: \"connected\"" ID=[ 175 ]
```

Logging (osd.log)

xstatus.txt

```
*s WebRTC MediaChannels Client Channel 25 Direction: Outgoing
*s WebRTC MediaChannels Client Channel 25 Id: "RTCOutboundRTPVideoStream_997907521"
```

osd.log

```
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Direction:Outgoing
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Id:RTCOutboundRTPVideoStream_997907521
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local IpAddress
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local Port:0
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local Protocol:Unknown
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local IpAddress:64.99.194.235
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local Port:35751
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local Protocol:UDP
```

Logging (osd.log)

```
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FrameHeight:360
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FrameWidth:640
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:22
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Transport RTCP Local Port:47465
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:29
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:30
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:29
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FrameHeight:540
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FrameWidth:960
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:30
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:29
…
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FrameHeight:720
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FrameWidth:1280
Io2: (S) Status WebRTC MediaChannels Client Channel[25] Video FramesPerSecond:30
```

Manual WebRTC Calling

- For manual call testing, xcommands can be used to initiate a WebRTC based call to the provider of choice with the below commands
- See roomos.cisco.com for full API documentation

```
xCommand WebRTC Join Type: MSTeams Url: <join_url>
xCommand WebRTC Join Type: GoogleMeet Url: <join_url>
```

[API Documentation]

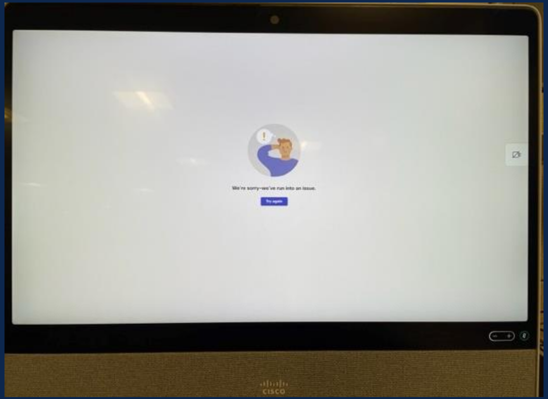
Example

Example 1: Unable to Join MS Teams WebRTC

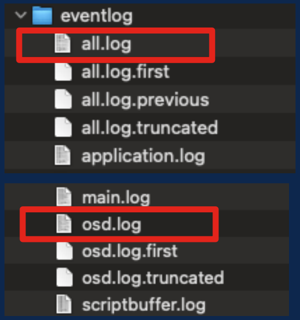
Problem Description:

We have multiple systems connected via Edge for Devices which fail when attempting to join a Microsoft Teams meeting via OBTP (WebRTC). The call attempts to connect and an error message is displayed saying “We’re sorry, we’ve run into an issue”.

Tested Webex and Zoom meetings and they are working fine.



Resolution: Unable to Join MS Teams WebRTC




```
14:46:53.280+00:00 appl[3888]: CuilApp[1]: User root/internal/phoenix-system
about to execute command '/WebRTC/Join BookingId: webex-2 Title: <removed> Type:
MSTeams Url: <removed> from localhost.
…
14:46:53.280+00:00 appl[3888]: APPL_Rpc I: Started WebRTC call CorrelationID=
<removed> URL= <removed> Title= <removed>
…
14:48:07.379+00:00 appl[3888]: WebRTCCall I: Ended WebRTC call
```

```
14:46:54.875+00:00 gui[3579]: INFO: WebEngine: Started loading "https://teams.microsoft.com/_#/l/meetup-
join/<meeting_url>”
14:46:54.876+00:00 gui[3579]: INFO: WebEngine: Succeeded loading "https://teams.microsoft.com/_#/l/meetup-
join/<meeting_url>"
…
14:46:55.237+00:00 gui[3579]: INFO: WebEngine: Started loading
"https://teams.microsoft.com/error/oops?errorMessage=cssloadfailed&errorDetails=rc:3;et:1682693215226;tc:1
3;https://statics.teams.cdn.office.net/hashed/stylesheets.theme-default.min-afc7381.css..."
```

Resolution: Unable to Join MS Teams WebRTC

- diagnostics.txt
- peripherals.txt
- run
  - tcpdump_files
    - extendedlogging.pcap

After the call is initiated and the WebRTC application is loaded, the web client tries to reach out to statics.teams.cdn.office.net. The DNS resolution can be seen working normally however any communication to the resolved address cannot complete the TCP handshake. Eventually the process times out and an error is presented in the Teams WebRTC application. In this instance traffic was only allowed externally to specific IP addresses and this one was missing from the allowed list.

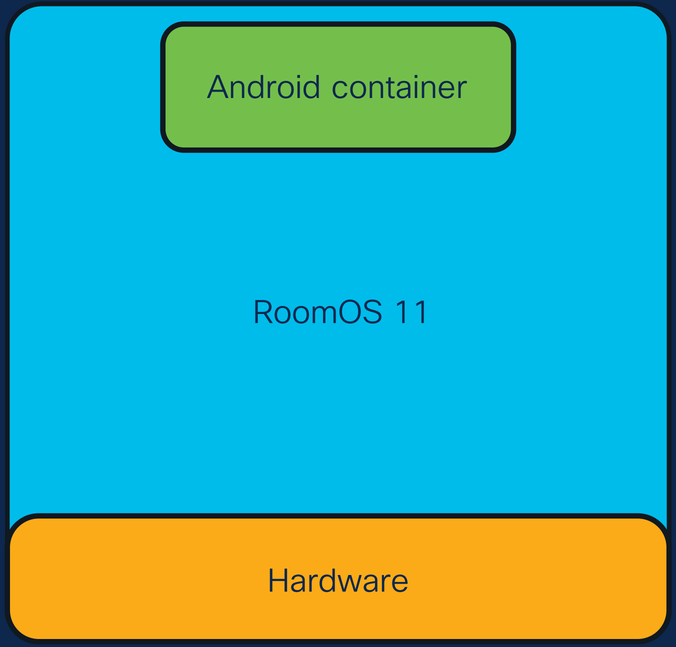
2023-04-28 14:46:54.168951	[redacted]	[redacted]	DNS	92	Standard query 0x7207 A statics.teams.cdn.office.net
2023-04-28 14:46:54.198255	[redacted]	[redacted]	DNS	248	Standard query response 0x7207 A statics.teams.cdn.office.net CNAME teams-staticscdn.traf
2023-04-28 14:46:54.228523	[redacted]	52.123.128.14	TCP	78	51850 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM TSval=38546078 TSecr=0 WS=128
2023-04-28 14:46:54.228708	[redacted]	52.123.128.14	TCP	78	51852 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM TSval=38546078 TSecr=0 WS=128
2023-04-28 14:46:54.229026	[redacted]	52.123.128.14	TCP	78	51854 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM TSval=38546078 TSecr=0 WS=128
2023-04-28 14:46:54.229342	[redacted]	52.123.128.14	TCP	78	51856 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM TSval=38546079 TSecr=0 WS=128
2023-04-28 14:46:54.229538	52.123.128.14	[redacted]	TCP	64	443 → 51850 [RST, ACK] Seq=1 Ack=1 Win=29200 Len=0
2023-04-28 14:46:54.229682	[redacted]	52.123.128.14	TCP	78	51858 → 443 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM TSval=38546079 TSecr=0 WS=128
2023-04-28 14:46:54.229695	52.123.128.14	[redacted]	TCP	64	443 → 51852 [RST, ACK] Seq=1 Ack=1 Win=29200 Len=0
2023-04-28 14:46:54.229723	52.123.128.14	[redacted]	TCP	64	443 → 51854 [RST, ACK] Seq=1 Ack=1 Win=29200 Len=0

Agenda

- 3rd Party Meeting Overview
- Log Collection Overview
- Troubleshooting WebRTC
- Troubleshooting MTR
- Conclusion

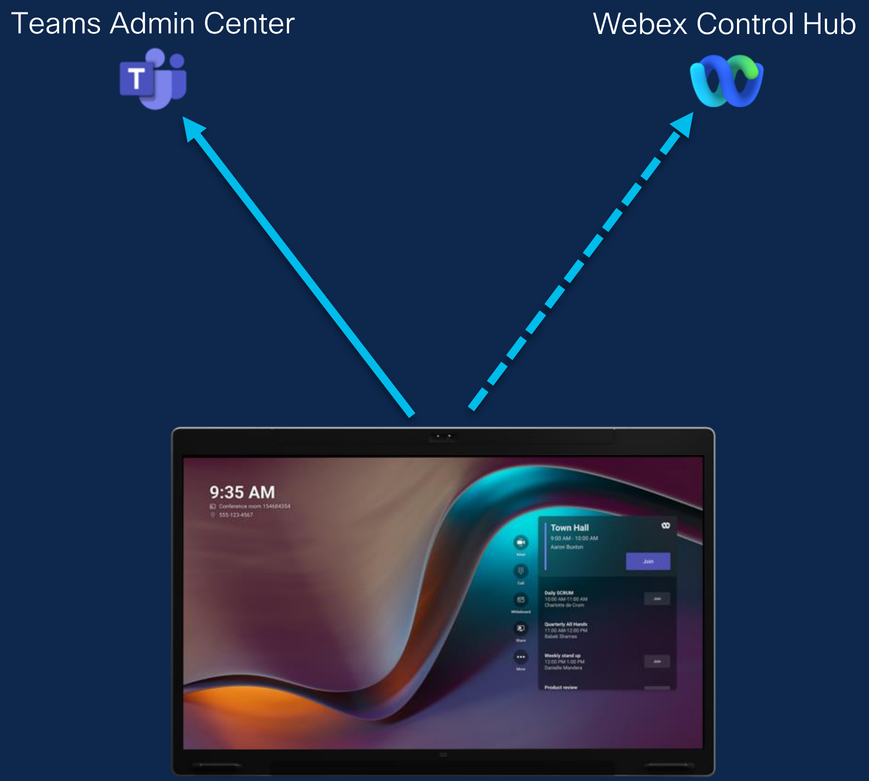
MTR vs RoomOS

- Containerized OS runs within RoomOS 11
- Android container runs MTR software
  - Handles all user interaction and calling
- RoomOS still maintains direct control over hardware and networking



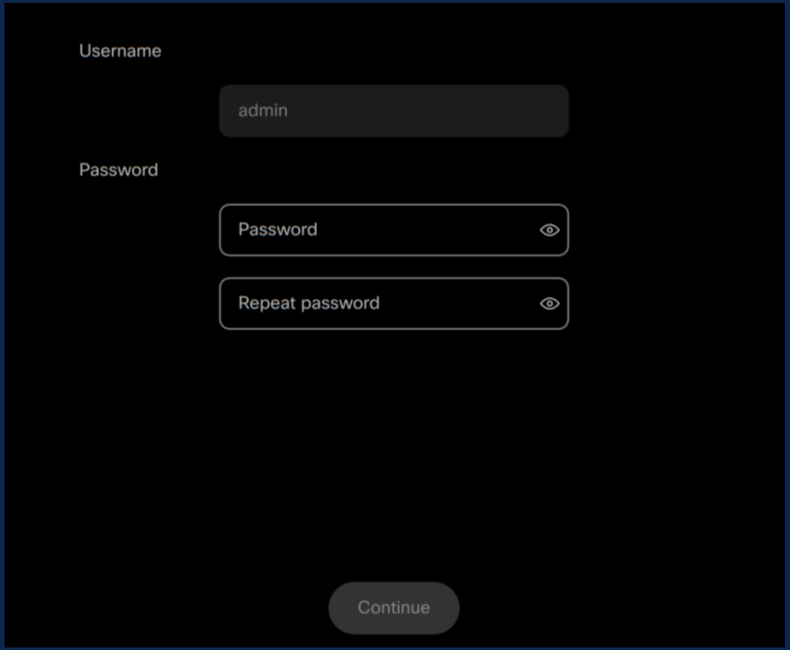
MTR Registration Options

- All MTR devices register to Microsoft Teams Admin Center (TAC)
- Devices can also be registered with control hub for additional features and management/troubleshooting options
- Control Hub registration can be done at the time of setup or at any point following on the device settings page
- Registration is done with the normal process using 16 digit activation key



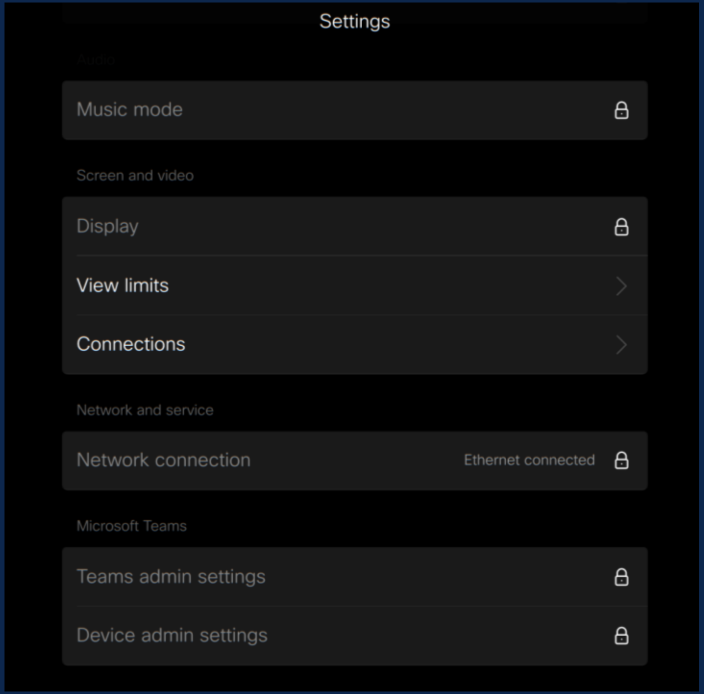
MTR Device Access

- Direct device access varies depending on if control hub is used for registration
- If Control Hub is not connected at the time of setup, user will be prompted to configure a username and password
- This can be changed at a later time in Teams Admin Center
- If the device is connected to Control Hub, the local admin account will be locked down and the device can be accessed directly from Control Hub



MTR Device Settings Lock

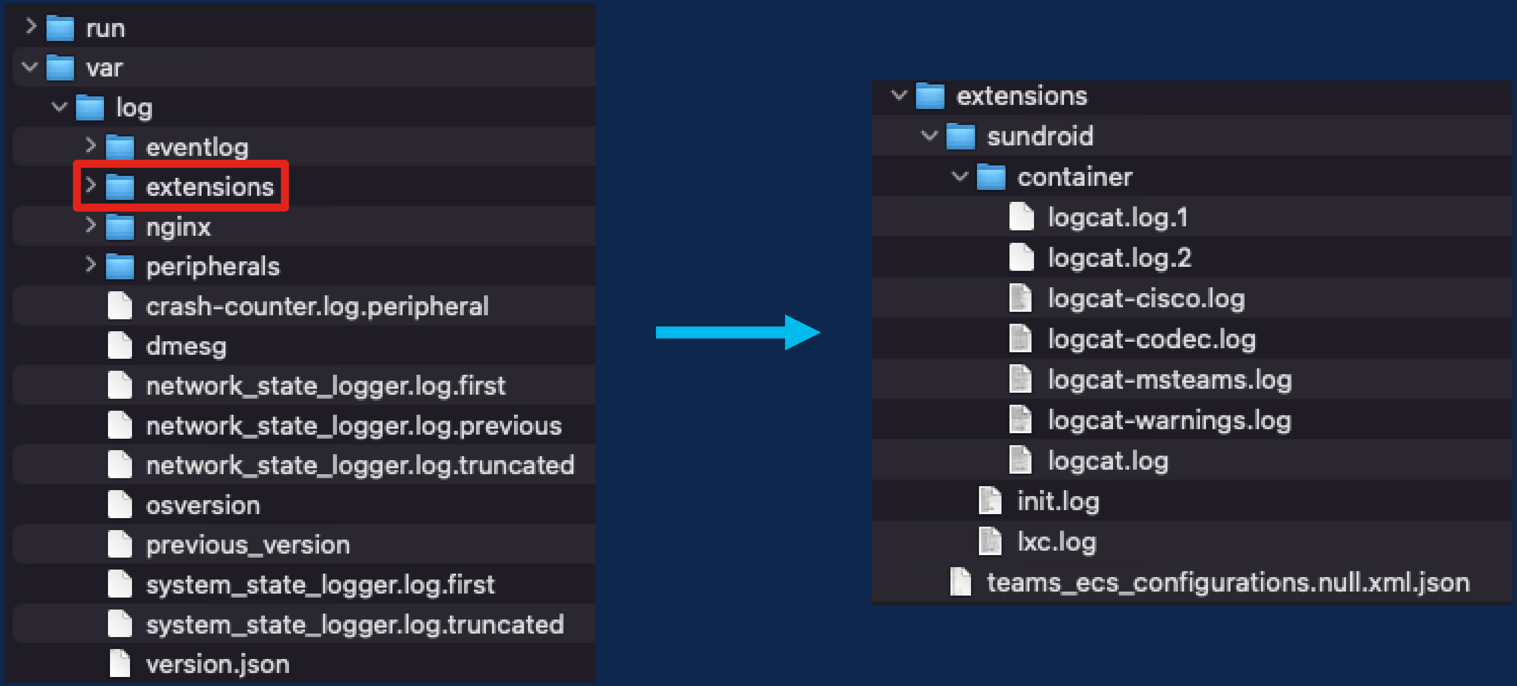
- By default all MTR systems lock the local UI from making any settings changes
- For non-Control Hub attached systems, these can be accessed by manually entering the admin credentials on the touch screen
- For control hub devices, these settings are locked completely until the configuration is changed
- This lock can be disabled by setting UserInterface/SettingsMenu/Mode to “unlocked”



MTR Log Collection

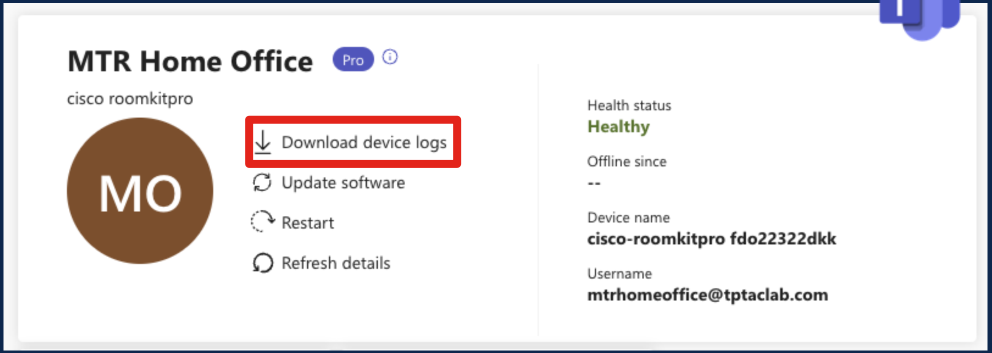
- MTR logs can be collected a variety of ways
  - On Device
  - Control Hub (if connected)
  - Teams Admin Center
- Device and Control Hub collected logs use the familiar bundle format with some additions
- Teams Admin Center logs contain a small subset of Microsoft Teams client specific logging

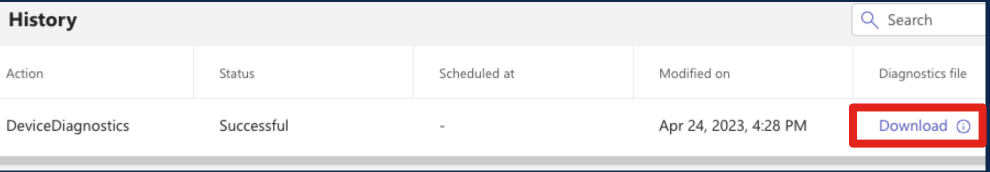
MTR Logging (Traditional Device Logs)



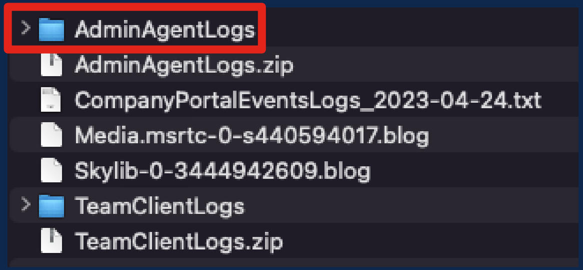
Teams Admin Center Logs

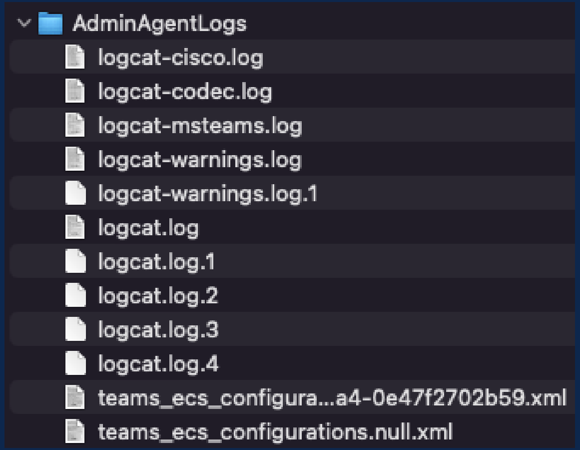
- Logs can be downloaded from Teams Admin Center under the "Teams Rooms for Android" devices page
- This will queue a download action which can take some time before completing
- Status can be checked and downloaded once available under the "History" tab





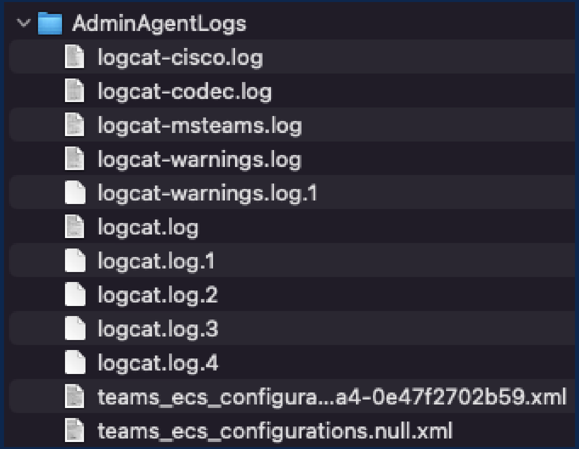
MTR Logging (Teams Admin Center Logs)





MTR Log File Overview (logcat.log)

- Essentially the ”all.log” for the Android instance of MTR
- All other more specific logcat files also dump to logcat.log
- Rotates from logcat.log (newest) to logcat.log.4 (oldest)
- No equivalent of x.first as seen in standard RoomOS logging



MTR Log File Overview (logcat-cisco.log)

- Logging of interaction between MTR and RoomOS
- Good place to track issues with settings changes initiated through the MTR UI

```
15:32:49.986  1783  3401 I MuteHandler: MTR -> CE: MTR mute: false
15:32:50.362  1783  1783 I VolumeHandler: MTR -> CE. Volume: 70
15:32:50.362  1783  1783 I VolumeHandler: Call status changed. deviceInCall: true
15:32:50.754  1783  1804 I VolumeHandler: CE -> MTR: Volume: 70
15:32:50.951  1783  3437 I VolumeHandler: MTR -> CE. Volume: 70
15:32:51.150  1783  3401 I MuteHandler: MTR -> CE: MTR mute: false
15:32:51.595  1783  1783 I VolumeHandler: MTR -> CE. Volume: 70
15:32:51.601  1783  1783 I VolumeHandler: Call status changed. deviceInCall: true
15:32:51.671  1783  3401 I MuteHandler: MTR -> CE: MTR mute: false
15:32:53.289  1783  3401 I MuteHandler: MTR -> CE: MTR mute: false
15:34:40.926  1783  1783 I VolumeHandler: MTR -> CE. Volume: 70
15:34:40.931  1783  1783 I VolumeHandler: Call status changed. deviceInCall: false
15:34:41.122  1783  3401 I MuteHandler: MTR -> CE: MTR mute: false
```

MTR Log File Overview (xstatus.txt)

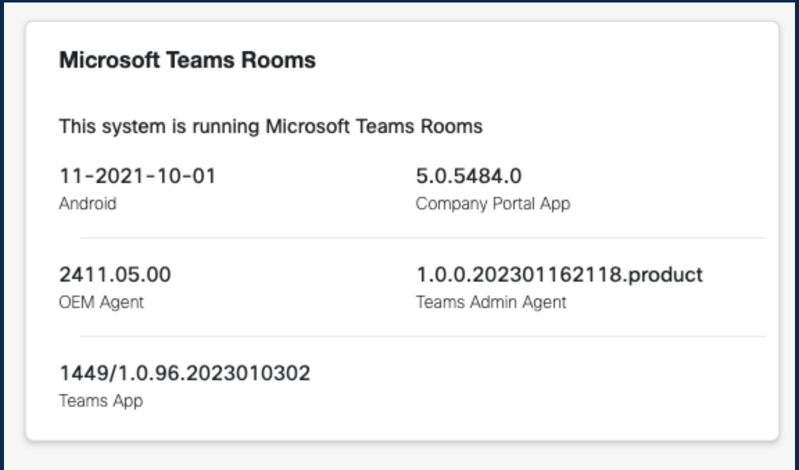
- New lines added for version information and configuration state
- Call state tracking to hook into other system functions

```
Apps FirstTimeWizard Topics Cisco SoftwareCompliance State: ToBeDone
Apps FirstTimeWizard Topics Cisco SoftwareUpdate State: ToBeDone
Apps FirstTimeWizard Topics MTR AdminPassphrase State: Finished
Apps FirstTimeWizard Topics MTR CiscoActivate State: Finished
Apps FirstTimeWizard Topics MTR DiagnosticsAndUsage State: Finished
Apps FirstTimeWizard Topics MTR InstallMTR State: CurrentlyOn
```

```
SystemUnit Extensions Microsoft InCall: False
SystemUnit Extensions Microsoft OEMAgentConnected: True
SystemUnit Extensions Microsoft State: Active
SystemUnit Extensions Microsoft Supported: True
SystemUnit Extensions Microsoft Version Android: "11-2021-10-01"
SystemUnit Extensions Microsoft Version CompanyPortalApp: "5.0.5484.0"
SystemUnit Extensions Microsoft Version OEMAgent: "2411.04.01"
SystemUnit Extensions Microsoft Version TeamsAdminAgent: "1.0.0.202209060820.product"
SystemUnit Extensions Microsoft Version TeamsApp: "1449/1.0.96.2023010302"
SystemUnit Extensions Microsoft VersionCode CompanyPortalApp: "5321940"
SystemUnit Extensions Microsoft VersionCode OEMAgent: "24110401"
SystemUnit Extensions Microsoft VersionCode TeamsAdminAgent: "382"
SystemUnit Extensions Microsoft VersionCode TeamsApp: "2023080011"
```

Web Interface

- Additional MTR section containing version information
- Access to config and status pages as normal
- Other pages for features that do not apply to MTR deployments removed



MTR Log Files Additional Notes

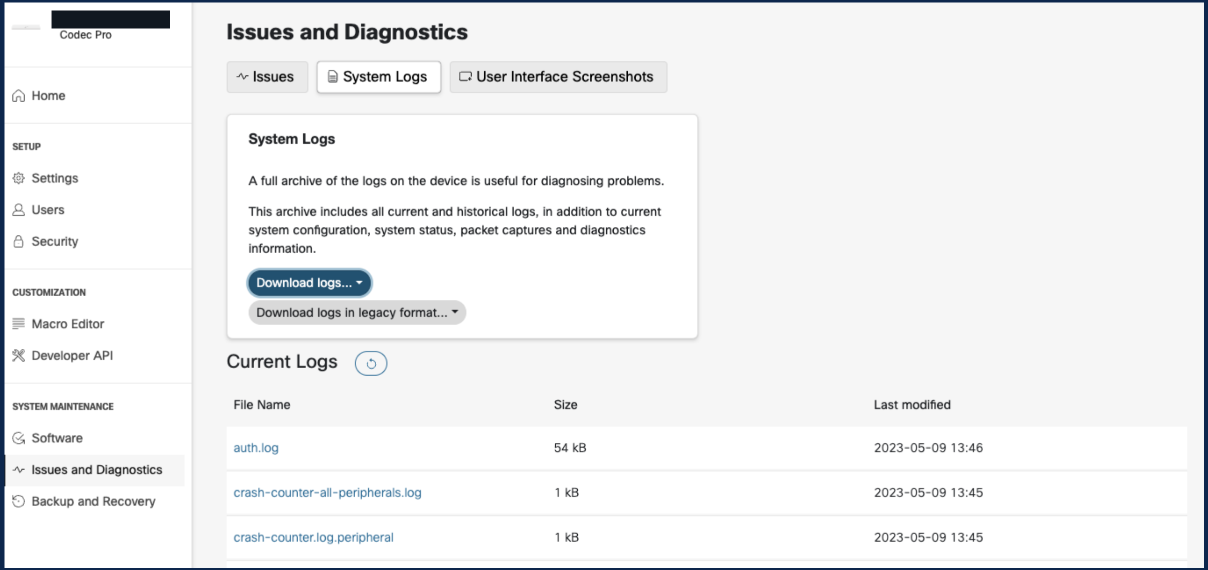
- Both callhistory.txt and bookings.txt are not populated as they normally would be for RoomOS meetings
- All other system logs function normally however you will not see most MTR specific actions such as calls or whiteboarding logged
- Anything in MTR that interacts with hardware (such as connecting a presentation source) will still be reflected in the regular logs

```
Welcome to
Cisco Codec Release RoomOS 11.4.1.7 3f0bc998202
SW Release Date: 2023-04-23
*r Login successful
OK


OK
*r CallHistoryGetResult (status=OK):
*r CallHistoryGetResult ResultInfo Offset: 0
*r CallHistoryGetResult ResultInfo Limit: 65534
** end
```

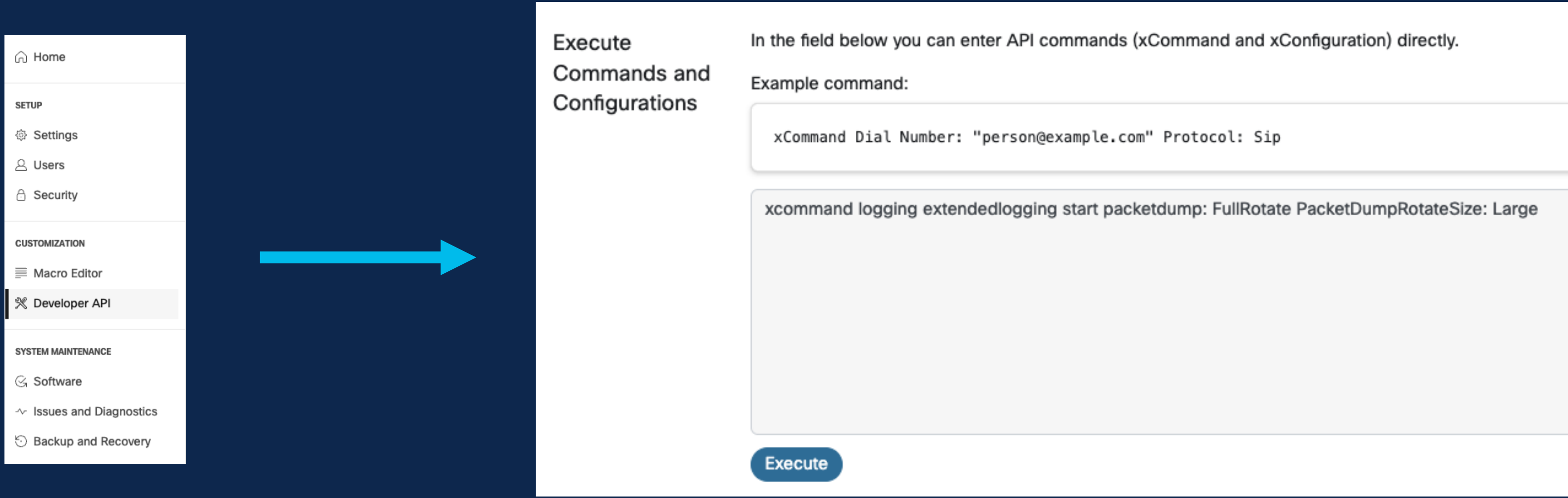
Debugging Limitations

- No extended logging
- No call control
- Direct log access still provided
- No native pcap ability



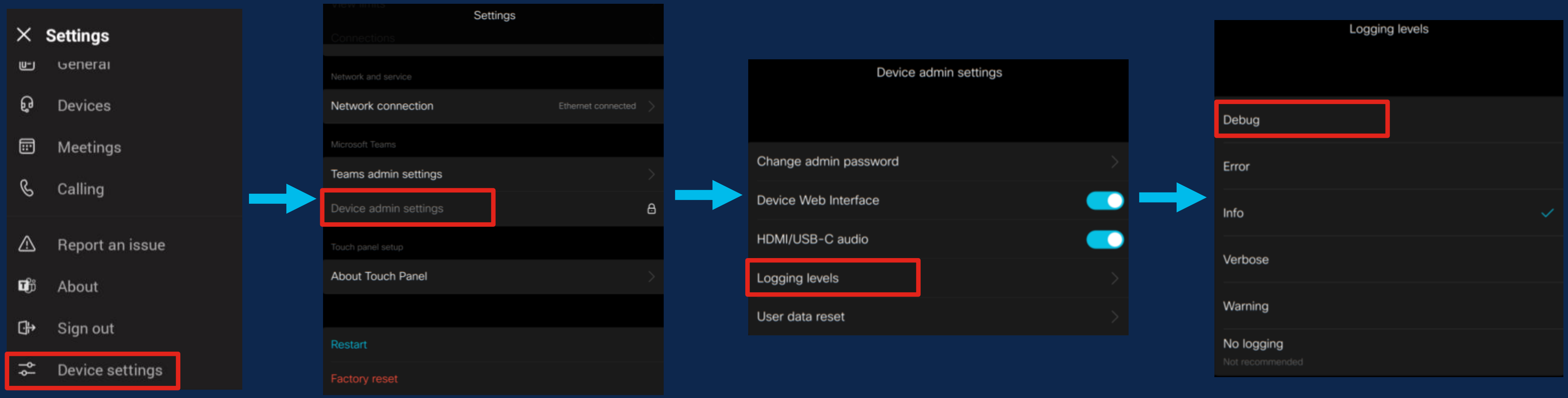
Manually Enable Extended Logging

- Extended logging and pcaps can be manually enabled through xAPI commands or via the "Developer API" section in the webUI
- See roomos.cisco.com for more info on options



MTR Logging Levels

- Additional debugging can be enabled for MTR logs via device menus

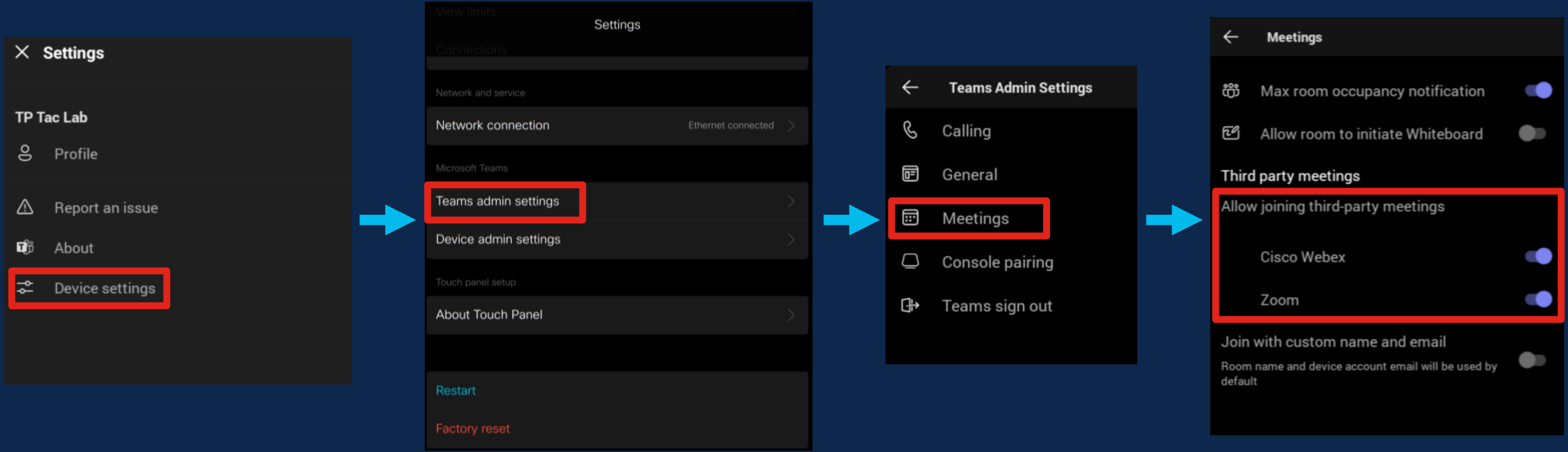


3rd Party Meeting Processing

- In order for Microsoft Exchange to process Webex or other 3rd party meeting invites to the resource mailbox, the following settings must be applied via PowerShell to the room mailbox
- `ProcessExternalMeetingMessages`
  - This must be set to `$true` to process meeting requests that originate outside of the Exchange organization. This is a requirement to process external Teams meetings and third-party meetings.
- `DeleteComments`
  - This must be set to `$false` to ensure that text in the message body of the incoming meeting request is not deleted. This is a requirement to process external Teams and third-party meetings to provide the One Touch Join experience.

3rd Party Meeting Processing

- Additionally, the individual services must be enabled on the device under



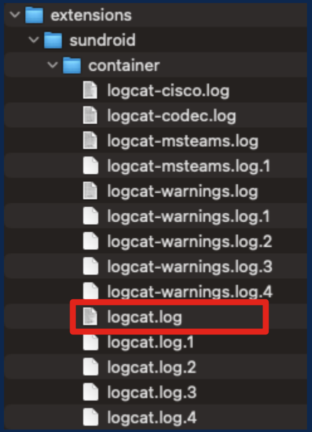
Examples

Example 1: MTR Calls Failing to Connect

Problem Description:

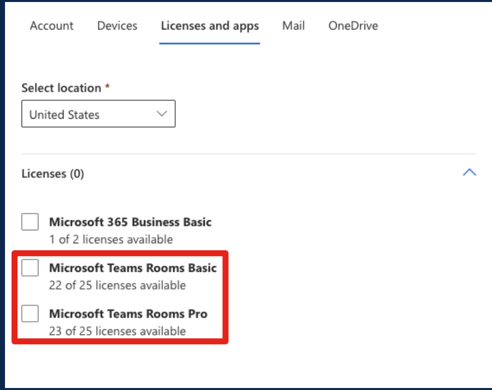
We have a Webex Board Pro in MTR mode that will no longer connects to incoming our outgoing calls. The call attempts to join but immediately disconnects. When calling to a person it says “We’re sorry, we couldn’t connect your call”.

Resolution: MTR Calls Failing to Connect




```
04-21 16:29:05.920 1165 4862 I Calling: CallNavigation: ProcessId: 1165,
Thread: Pool-CallInitialization-Thread-16, Showing fullscreen incoming call
: 125

04-21 16:29:08.779 1165 4908 E EndpointsAppData: ProcessId: 1165, Thread:
Pool-Auth-Thread-281, Failed to fetch SkypeToken from Authz endpoint.
{"errorCode":"UserLicenseNotPresentForbidden","message":"User Login. Teams
is disabled in user licenses"}
```



Solution: License had been removed from the device at some point after registration and needed to be re-added through the admin portal under Users → Active Users → Licensing and Apps

Example 2: System Freezing on MTR Install

Problem Description:

We are deploying MTR on new devices and the installation process is freezing when downloading the MTR-A image. Multiple users with devices in different locations are reporting the same issue.

Resolution: System Freezing on MTR Install

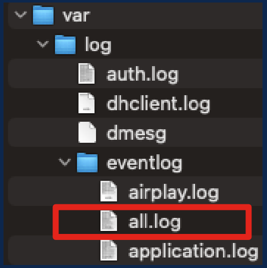
- bookings.txt
- callhistory.txt
- camera_presets.txt
- diagnostics.txt
- peripherals.txt
- var
- wifi_extra_status.log
- xconfig.txt
- xstatus.txt

- var
  - log
    - auth.log
    - dhclient.log
    - dmesg
    - eventlog
    - hwmon.log
    - kern.log
    - kernerr.log

```
*s Apps FirstTimeWizard Topics MTR InstallMTR State: CurrentlyOn
…
*s SystemUnit Extensions Microsoft InCall: False
*s SystemUnit Extensions Microsoft State: Disabled
*s SystemUnit Extensions Microsoft Supported: True
*s SystemUnit Extensions Supported: True
```

MTR extension is not present in the log bundle and not currently running according to the xstatus.txt output

Resolution: System Freezing on MTR Install



Solution: DNS lookup was failing when the system attempted to download the MTR extension, causing the download process to try and fail repetitively.

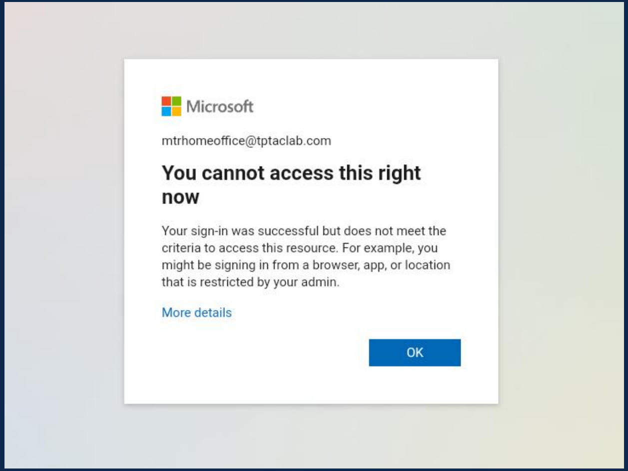
```
swupgrade[2551]: SoftwareUpgrade W: HTTP(1) Error: <DNS Lookup failure> (Could not resolve host: binaries.webex.com)

swupgrade[2551]: SoftwareUpgrade I: Download #1 failed: <DNS Lookup failure>: 'Could not resolve host: binaries.webex.com' url: https://binaries.webex.com/collaboration-endpoint-ce-production-beta/20230503142830/sundroid.ext
…
swupgrade[2551]: SoftwareUpgrade I: Download #15 failed: <DNS Lookup failure>: 'Could not resolve host: binaries.webex.com' url: https://binaries.webex.com/collaboration-endpoint-ce-production-beta/20230503142830/sundroid.ext
```

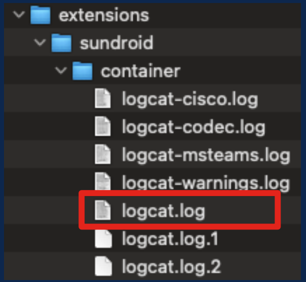

Example 3: MTR Registration Failing

Problem Description:

We are registering new MTR devices but the registration process is failing with an error stating Teams cannot be accessed.



Resolution: MTR Registration Failing



Solution: Conditional access policies have been enabled but inTune onboarding is not configured for this device thus the registration is denied. A similar error would also be present if onboarding was denied based on Intune parameters

```
LogonUserWatcher: received intent action: com.microsoft.teams.ipphone.admin.agent.CURRENT_LOGON_USER
LogonUserWatcher: userInfo :
{"authenticatedUsers":[{"accountType":"ENTERPRISE","cloudType":"PUBLIC_CLOUD …
LogonUserWatcher: teamsIdentifier : {"deviceId":"91ff0075-a133-376c-88ae-ad83e7d51277"}
LogonUserWatcher: userInfo value : UserInfo{usageMode='personalUser’ …
LogonUserWatcher: userid (9b844f15-c759-4a35-91a4-0e47f2702b59) with teamsIdentifier …
GetRecoveryCommands: canRunWithState: false, manageabilityLevel: Healthy
I EnrollOperation: Enroll operation started
…
MSALService: AADSTS53003: Access has been blocked by Conditional Access policies. The access policy
does not allow token issuance.
```

Agenda

- 3rd Party Meeting Overview
- Log Collection Overview
- Troubleshooting WebRTC
- Troubleshooting MTR
- Conclusion

Key Points to Remember

- Determine the meeting platform used
- For WebRTC based calls, pull logs while call is in progress
- Reference osd.log for more detailed WebRTC call information
- For MTR devices, remember to configure credentials and unlock settings
- MTR logs can be collected from device, Control Hub, and Teams Admin Center

Fill out your session surveys!

Attendees who fill out a minimum of four session surveys and the overall event survey will get **Cisco Live-branded socks** (while supplies last)!

Attendees will also earn 100 points in the **Cisco Live Challenge** for every survey completed.

These points help you get on the leaderboard and increase your chances of winning daily and grand prizes

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

The bridge to possible

Thank you

#CiscoLive

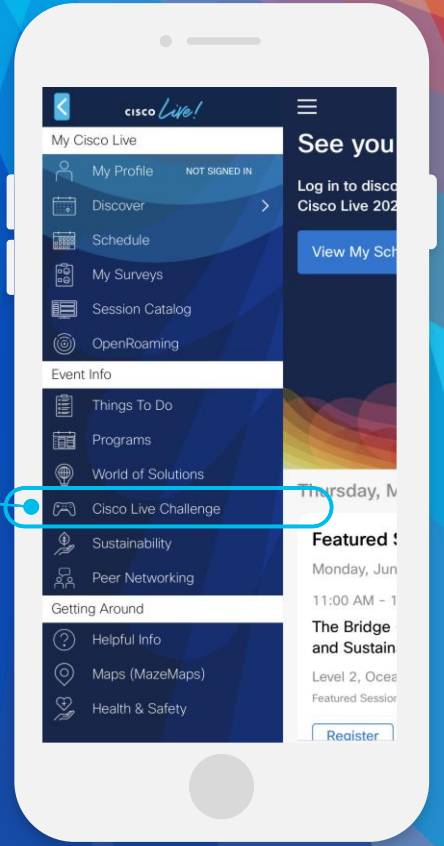
Cisco Live **Challenge**

Gamify your Cisco Live experience!

Get points for attending this session!

How:

1. Open the Cisco Events App.
2. Click on 'Cisco Live Challenge' in the side menu.
3. Click on View Your Badges at the top.
4. Click the + at the bottom of the screen and scan the QR code:



Let's go