
cisco Live!
TURN
IT
UP
#CiscoLive

The bridge to possible

Webex Calling deployment and media path optimization

Johannes Krohn, Principal Technical Marketing Engineer

BRKCOL-2046

Agenda

- Bulk Provisioning
  - User Provisioning
  - Device Provisioning
  - Provisioning APIs
- Media Path Optimization
  - Regional Media
  - Firewall Traversal and Media Path Optimization
  - Sample Media Flows
- Summary

Safe harbor statement

This presentation contains “forward-looking” statements that involve risks, uncertainties and assumptions. If the risks or uncertainties ever materialize or the assumptions prove incorrect, our results may differ materially from those expressed or implied by such forward-looking statements. All statements other than statements of historical fact could be deemed forward-looking, including, but not limited to, any projections of financial information; any statements about historical results that may suggest trends for our business; any statements of the plans, strategies, and objectives of management for future operations; any statements of expectation or belief regarding future events, technology developments, or enforceability of our intellectual property rights; and any statements of assumptions underlying any of the foregoing.

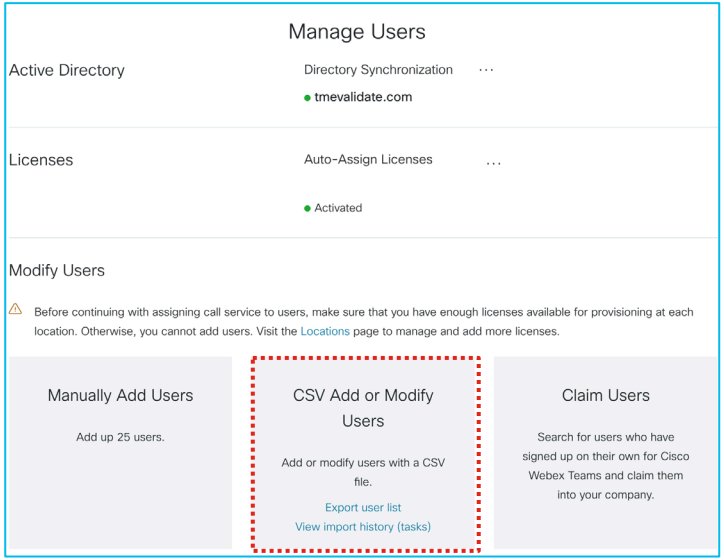
These statements are based on estimates and information available to us at the time of this presentation and are not guarantees of future performance. Actual results could differ materially from our current expectations as a results of many factors, including but not limited to: the unpredictable nature of our rapidly evolving market and quarterly fluctuations in our business; the effects of competition; and any adverse changes in our indirect channel relationships. These and other risks and uncertainties associated with our business are described in the company’s annual report on Form 10-K. The forward-looking statements in this presentation are made as of the date of the initial publication of this presentation, and we disclaim any obligation to update these statements at any time in the future.

Bulk Provisioning

User Provisioning

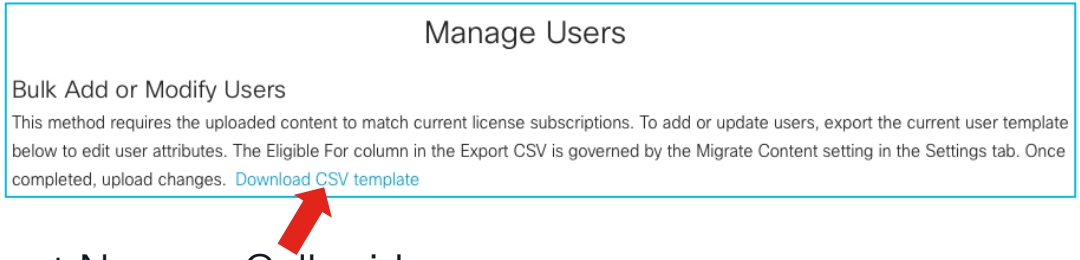
Bulk user operations

- Add/Update users
- Enable users for Webex Calling
- Assign user to location
- Assign DN (and DID) to user
- Define calling behavior
- Update caller ID



CSV file for bulk user operations

- Relevant Columns:
  - Extension- DN
  - Phone Number – DID
  - Caller ID Number/First Name/Last Name – Caller id
  - Location – Webex Calling Location
  - Calling behavior – user level calling behavior setting
  - Webex Calling VAR Enterprise – Enable for Webex Calling (true / false)

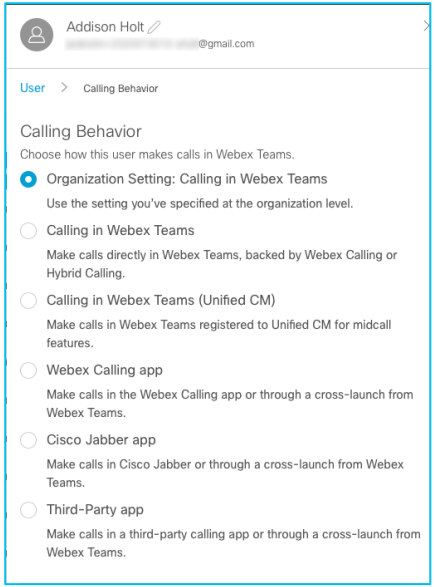


	A	B	C	D	E	H	I	J	S	AB
1	First Name	Last Name	Display Name	User ID/Email (Required)	User Status	Directory Number	Direct Line	Location	Calling Behavior	Webex Calling VAR Enterprise
3	Adrienne	Alvarado	Adrienne Alvarado	[redacted]	Not Verified	1101		Site1	USE_ORG_SETTINGS	TRUE
4	Amber	Kent	Amber Kent	[redacted]	Not Verified	1102		Site1	USE_ORG_SETTINGS	TRUE
5	Aretha	Mcintosh	Aretha Mcintosh	[redacted]	Not Verified	1103		Site1	USE_ORG_SETTINGS	TRUE
6	Adena	Villarreal	Adena Villarreal	[redacted]	Active	1104		Site1	USE_ORG_SETTINGS	TRUE
7	Beatrice	Rich	Beatrice Rich	[redacted]	Not Verified	1105		Site1	USE_ORG_SETTINGS	TRUE
8	Brenda	Sharp	Brenda Sharp	[redacted]	Not Verified	1106		Site1	USE_ORG_SETTINGS	TRUE
9	Cyrus	Adams	Cyrus Adams	[redacted]	Not Verified	1107		Site1	USE_ORG_SETTINGS	TRUE
10	Cameron	Baker	Cameron Baker	[redacted]	Not Verified	1108		Site1	USE_ORG_SETTINGS	TRUE
11	Chancellor	Bauer	Chancellor Bauer	[redacted]	Not Verified	1109		Site1	USE_ORG_SETTINGS	TRUE
12	Cailin	Holloway	Cailin Holloway	[redacted]	Not Verified	1110		Site1	USE_ORG_SETTINGS	TRUE
13	Ciaran	Martinez	Ciaran Martinez	[redacted]	Not Verified	1111		Site1	USE_ORG_SETTINGS	TRUE
14	Claire	Mcgowan	Claire Mcgowan	[redacted]	Not Verified	1112		Site1	USE_ORG_SETTINGS	TRUE
15	Charles	Ratliff	Charles Ratliff	[redacted]	Not Verified	1113		Site1	USE_ORG_SETTINGS	TRUE

CSV file for bulk user operations

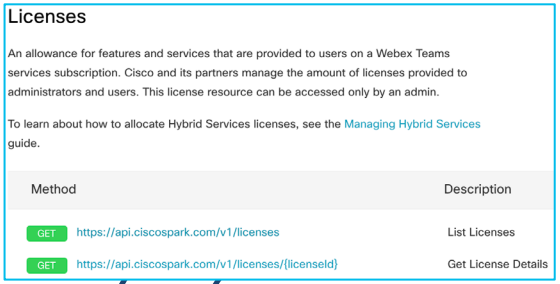
- Relevant Columns:
  - Extension- DN
  - Phone Number – DID
  - Caller ID Number/First Name/Last Name – Caller id
  - Location – Webex Calling Location
  - Calling behavior – user level calling behavior setting
  - Webex Calling VAR Enterprise – Enable for Webex Calling (true / false)
- Calling behavior option

USE_ORG_SETTINGS	Organization Setting
NATIVE_WEBEX_TEAMS_CALLING	Calling In Webex App
NATIVE_SIP_CALL_TO_UCM	Calling in Webex App (Unified CM)
CALL_WITH_APP_REGISTERED_FOR_WEBEXCALLTEL	Webex Calling App
CALL_WITH_APP_REGISTERED_FOR_CISCOTEL	Cisco Jabber App
CALL_WITH_APP_REGISTERED_FOR_TEL	Third-Party App



User provisioning using APIs

- Webex Calling locations can only be read (no creation or update)
- Webex Teams People API allows to provision users and to assign licenses (new: support to add Webex Calling entitlements)
- New: People API can be used to set location, DID and extension

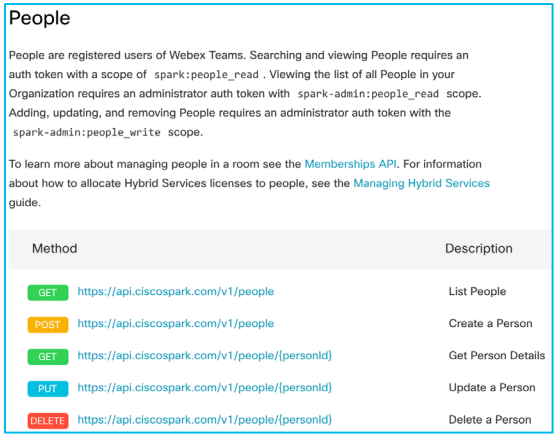


Licenses

An allowance for features and services that are provided to users on a Webex Teams services subscription. Cisco and its partners manage the amount of licenses provided to administrators and users. This license resource can be accessed only by an admin.

To learn about how to allocate Hybrid Services licenses, see the Managing Hybrid Services guide.

Method		Description
GET	https://api.ciscospark.com/v1/licenses	List Licenses
GET	https://api.ciscospark.com/v1/licenses/{licenseId}	Get License Details



People

People are registered users of Webex Teams. Searching and viewing People requires an auth token with a scope of `spark:people_read` . Viewing the list of all People in your Organization requires an administrator auth token with `spark-admin:people_read` scope. Adding, updating, and removing People requires an administrator auth token with the `spark-admin:people_write` scope.

To learn more about managing people in a room see the Memberships API. For information about how to allocate Hybrid Services licenses to people, see the Managing Hybrid Services guide.

Method		Description
GET	https://api.ciscospark.com/v1/people	List People
POST	https://api.ciscospark.com/v1/people	Create a Person
GET	https://api.ciscospark.com/v1/people/{personId}	Get Person Details
PUT	https://api.ciscospark.com/v1/people/{personId}	Update a Person
DELETE	https://api.ciscospark.com/v1/people/{personId}	Delete a Person


```
{
    "id": "Y2lzY29zcGFyazovL3VzL1BFT1BMRS80ZTIwMTdiNC1jNzdiLTQxZDMtYThhNS04OWE1Mzk1ZTYyNTg",
    "emails": [
        "jkrohn@[redacted]"
    ],
    "phoneNumbers": [
        {
            "type": "work",
            "value": "+1[redacted]"
        }
    ],
    "displayName": "Johannes Krohn",
    "nickName": "Johannes",
    "firstName": "Johannes",
    "lastName": "Krohn",
    "orgId": "Y2lzY29zcGFyazovL3VzL09SR0FOSVpBVElPTi9kMGU1Yjk3MC1jNzIwLTQxMDQtYmEzMi1mZTBkZTMwYTVkMjY",
    "roles": [
        "Y2lzY29zcGFyazovL3VzL1JPTEUvaWRfZnVsbF9hZG1pbg"
    ],
    "licenses": [
        "Y2lzY29zcGFyazovL3VzL0xJQ0VOU0UvZDBlNWI5NzAtYzcyMC00MTA0LWJhMzItZmUwZGUzMGE1ZDI2OkVFXzNiMjc3ODk2",
        "Y2lzY29zcGFyazovL3VzL0xJQ0VOU0UvZDBlNWI5NzAtYzcyMC00MTA0LWJhMzItZmUwZGUzMGE1ZDI2Ok1TXzc4ZGYyMmE1",
        "Y2lzY29zcGFyazovL3VzL0xJQ0VOU0UvZDBlNWI5NzAtYzcyMC00MTA0LWJhMzItZmUwZGUzMGE1ZDI2OkNNUl85ODhlYmYy",
        "Y2lzY29zcGFyazovL3VzL0xJQ0VOU0UvZDBlNWI5NzAtYzcyMC00MTA0LWJhMzItZmUwZGUzMGE1ZDI2OkJDU1REX2IyMjFm"
    ],
    "created": "2018-04-03T18:20:43.092Z",
    "lastModified": "2020-01-21T13:32:05.626Z",
    "lastActivity": "2020-01-21T11:06:37.635Z",
    "status": "inactive",
    "invitePending": false,
    "loginEnabled": true,
    "type": "person"
}
```

User provisioning – Summary

- Directory Connector
  - Automatic user provisioning
  - Linked to enterprise directory
  - Foundation for all Webex services (Messaging, Meeting, Calling)
- Bulk (or per user) operations for Webex Calling specific settings
  - DN, DID, Location, Calling Behavior
- Provisioning API support (People API)
  - Calling entitlement, DN, DID, Location

Device Provisioning

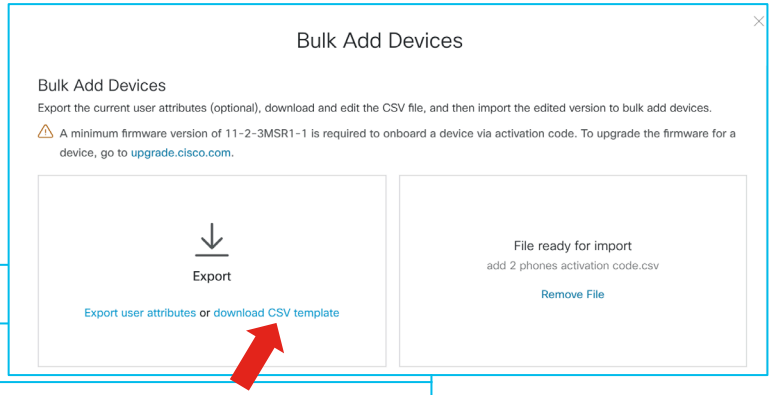
Device migration

- Firmware migration: Enterprise to MPP
  - Firmware migration required to register phones to Webex Calling
  - 7811, 7821*, 7841*, 7861*, 7832
  - Simplified cloud driven migration process**
- Device provisioning on Webex Calling
  - Per device
  - Bulk operations

*7821 V03+, 7841 V04+, 7861 V03+
**https://upgrade.cisco.com/e2m_converter

Phone bulk provisioning

- Bulk provisioning based on CSV file upload



Username	Reference to existing user or name of place
Type	USER or PLACE
Directory Number	Has to be empty for type USER
Direct Line	Has to be empty for type USER
Device Type	IP, WEBEX, or WEBEX_CALLING
Model	Only required for device type IP
MAC Address	For device type IP, activation code onboarding if empty
Location	Webex Calling location, empty for type USER

Username	Type	Directory Number	Direct Line	Device Type	Model	MAC Address	Location
bob@example.com	USER			IP	Cisco 8865	AB0971FA2967	
alice@example.com	USER			IP	Cisco 8865		
Reception	PLACE	101	4085550101	WEBEX_CALLING	Cisco 8865		Dallas

Phone bulk provisioning, type “User”

Username	Type	Directory Number	Direct Line	Device Type	Model	MAC Address	Location
alice@example.com	USER			IP	Cisco 8865	AFFEAFFE0001	
bob@example.com	USER			IP	Cisco 8865		
charlie@example.com	USER			WEBEX			
Barn	PLACE	1201		IP	Cisco 8841	AFFEAFFE0002	SJC
Shed	PLACE	1202		IP	Cisco 8841		SJC
Game Room	PLACE	1205		WEBEX_CALLING			SJC
Ranch House	PLACE			WEBEX			

Valid Device Types:
- IP: MPP
- WEBEX: Webex Device (personal mode)

No extension, DID, Location;
inherited from user referenced by username

Empty MAC Address: create activation code. Has to be empty for
Device Type “Webex”

Model required for Device Type “IP”

Phone bulk provisioning, type “Place”

Username	Type	Directory Number	Direct Line	Device Type	Model	MAC Address	Location
alice@example.com	USER			IP	Cisco 8865	AFFEAFFE0001	
bob@example.com	USER			IP	Cisco 8865		
charlie@example.com	USER			WEBEX			
Barn	PLACE	1201		IP	Cisco 8841	AFFEAFFE0002	SJC
Shed	PLACE	1202		IP	Cisco 8841		SJC
Game Room	PLACE	1205		WEBEX_CALLING			SJC
Ranch House	PLACE			WEBEX			

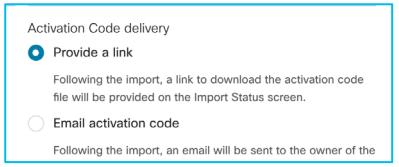
Valid Device Types:
- IP: MPP
- WEBEX: Webex Device (shared Mode)
- WEBEX_CALLING: Webex Device
- (shared Mode w/ Webex Calling)

Model required for Device Type “IP”

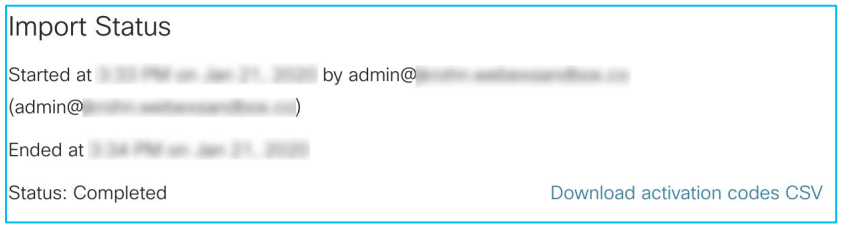
Extension,
Location required for Webex Calling
DID is optional

Empty MAC Address: create activation code.
Has to be empty for device type “WEBEX” and “WEBEX_CALLING

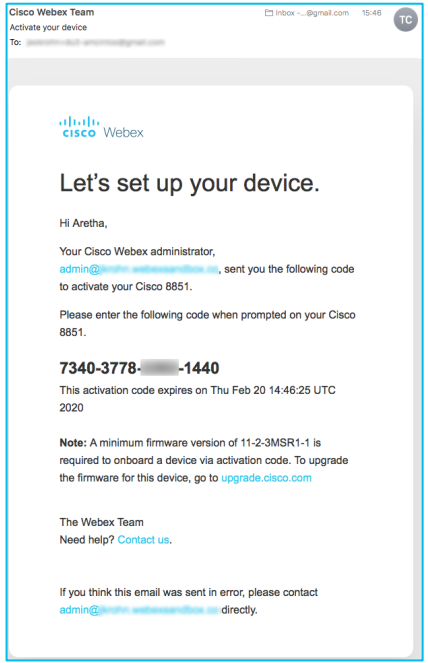
Bulk provisioning with activation codes



- Activation codes can be downloaded as CSV or emailed to the device owner (not for places)
- CSV download after completion of bulk transaction
- CSV contains activation codes together w/ device information
- Webex Devices show up in Webex Control Hub after activation



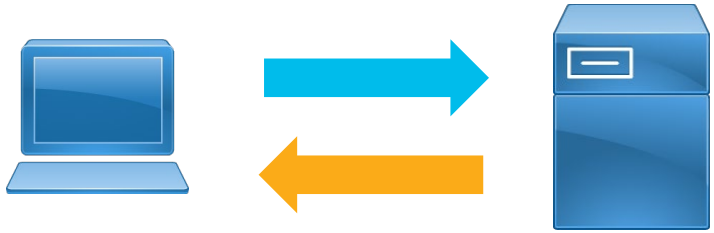
	A	B	C	D	E	F	G	H
1	Username	Activation Code	Expiry Time	Display Name	Model	Location	Owner Id	Type
2	[redacted]	0691-1426-[redacted]-7954	Thu Feb 20 14:33:58 UTC 2020	Adrienne Alvarado	Cisco8851	Site1	79078743-af71-4f31-ab06-b770301c9c7e	USER
3	[redacted]	5888-2820-[redacted]-3494	Thu Feb 20 14:34:04 UTC 2020	Amber Kent	Cisco8851	Site1	4b04d52c-3ec0-4179-9d91-2284fc6dc046	USER



Provisioning APIs

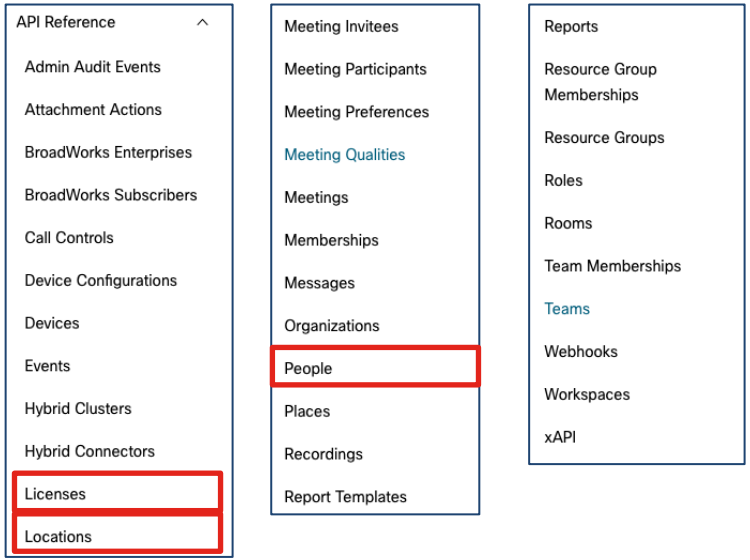
REST – Representational State Transfer

- Not really a standard – more an architecture
- Uses existing standards: for example HTTP(S) for transport
- All about client-server
- Conceptually similar to web browser accessing web server
- Resources
  - Every resource can be addressed by a URI
  - Methods: GET, PUT, POST, DELETE, HEAD
  - Uniform representation: typically JSON
- Protocol
  - Stateless
  - Client-server

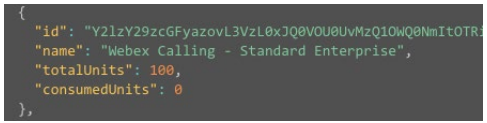


Webex APIs

- Documentation, References, …: https://developer.webex.com/



Webex APIs for Webex Calling Provisioning



- Licences
  - List licenses and determine Webex Calling License
- Locations
  - List exisiting Webex Calling Locations
- People
  - CRUD users
  - `callingData` parameter for Webex Calling
  - set/update location, extension, DID, (Webex Calling) license
- No APIs yet to provision locations, devices, ...

New APIs (Calling Features)*

User Features

- Barge-in (user & admin
- Call forwarding (user & admin)
- Call recording (user & admin)
- Voicemail, Greeting (user & admin)
- DND (user & admin)
- Intercept

Bulk provisioning

- Calling behavior
  - Get/set profile via API
- User Parameters
  - Caller ID: number, first/last name
  - Calling behavior: get/set profile
  - UC profile: assign profile per user

*To be released March ’21

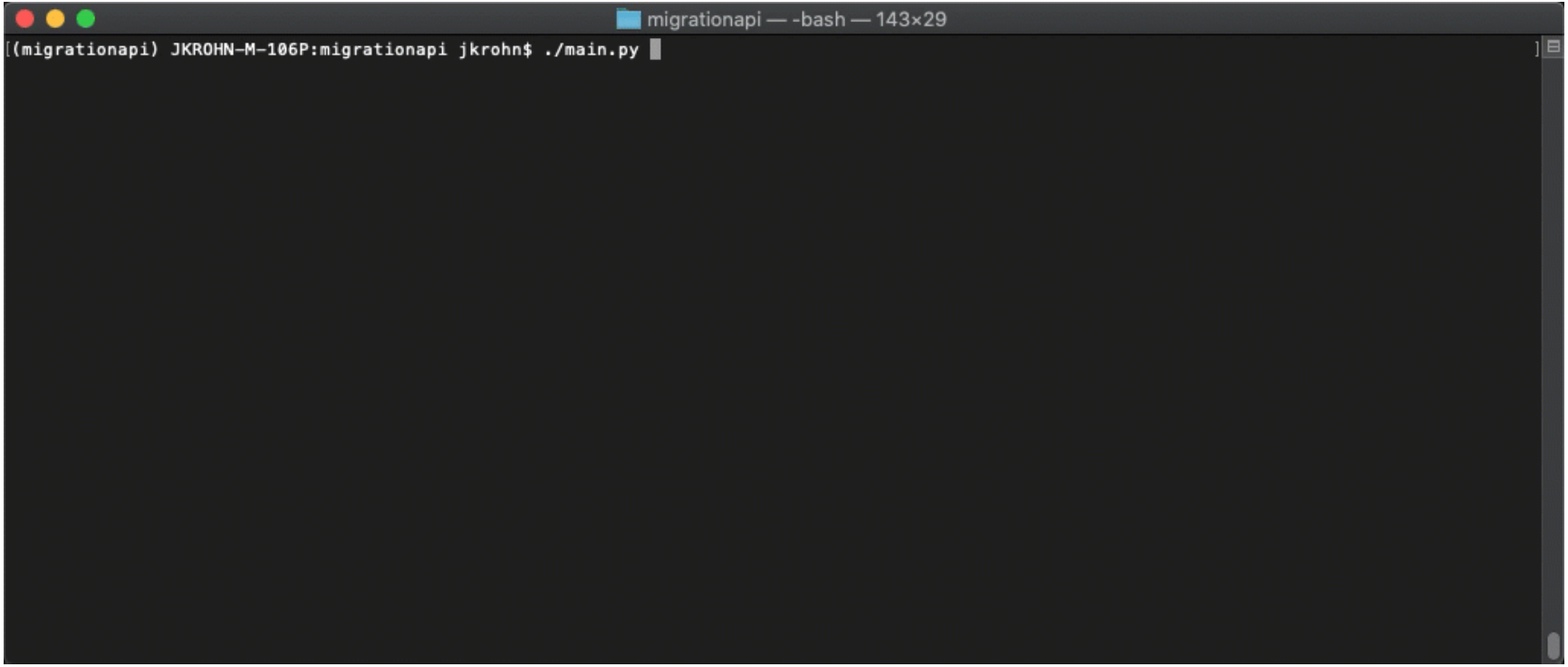
Demo: Automation Using Provisioning APIs

- Read users from Unified CM via AXL
- Select users with phone numbers in a specific range
- Provision these users for Webex Calling and assign their extension
  - Async calls b/c Webex Calling provisioning calls are slow
  - Async code allows concurrent execution of multiple REST API calls
- Access Token for Webex API has to be obtained from developer.cisco.com

Demo Framework

- https://github.com/jeokrohn/migrationapi
- Based on Python 3.7
- ucm_axl – ucm AXL helper
- ucm_reader – abstraction layer to read users, phones from Unified CM
  - Uses `pydantic` to create ”pythonic” data representation of Unified CM data types (user, phone, location)
- Webexteamsasyncapi
  - Rudimentary implementation of async API handled for Webex API
  - Handles 429 (rate limiting)
  - Handles spurious 500/502 results

Demo



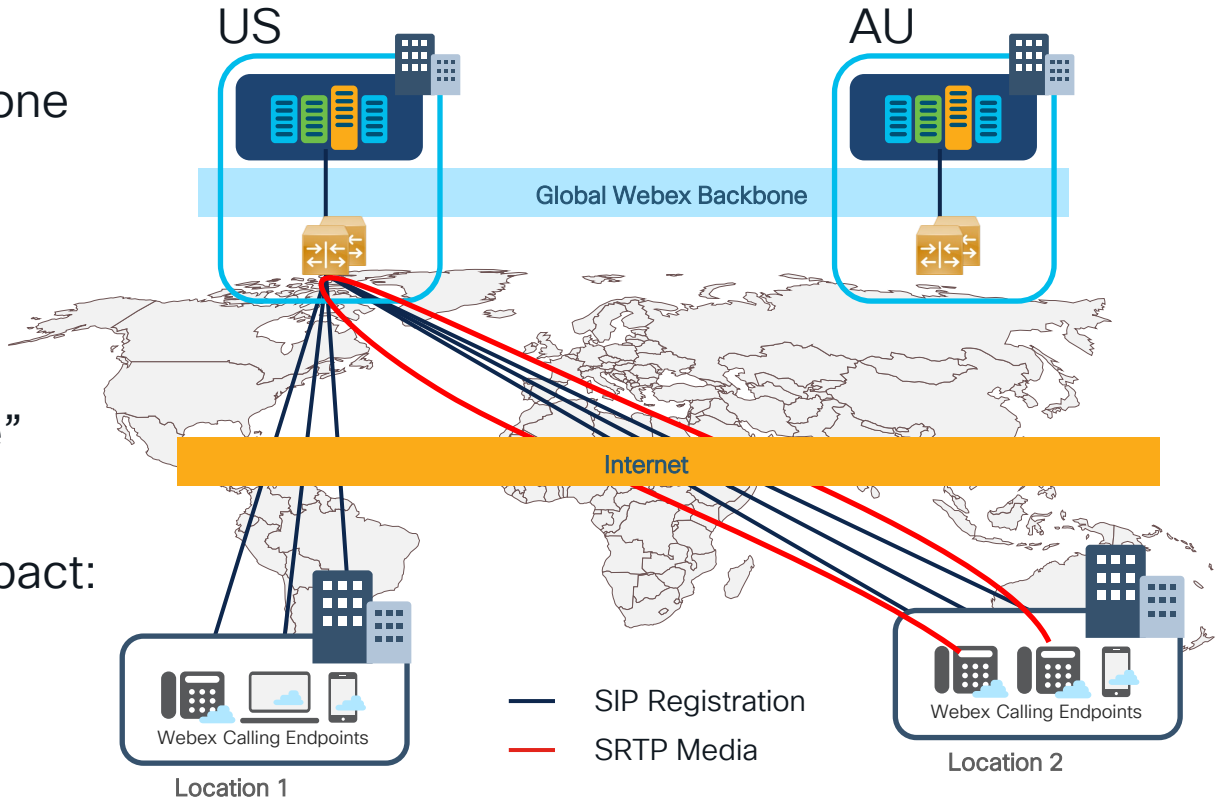
Media Path Optimization

cisco Live!

Regional Media

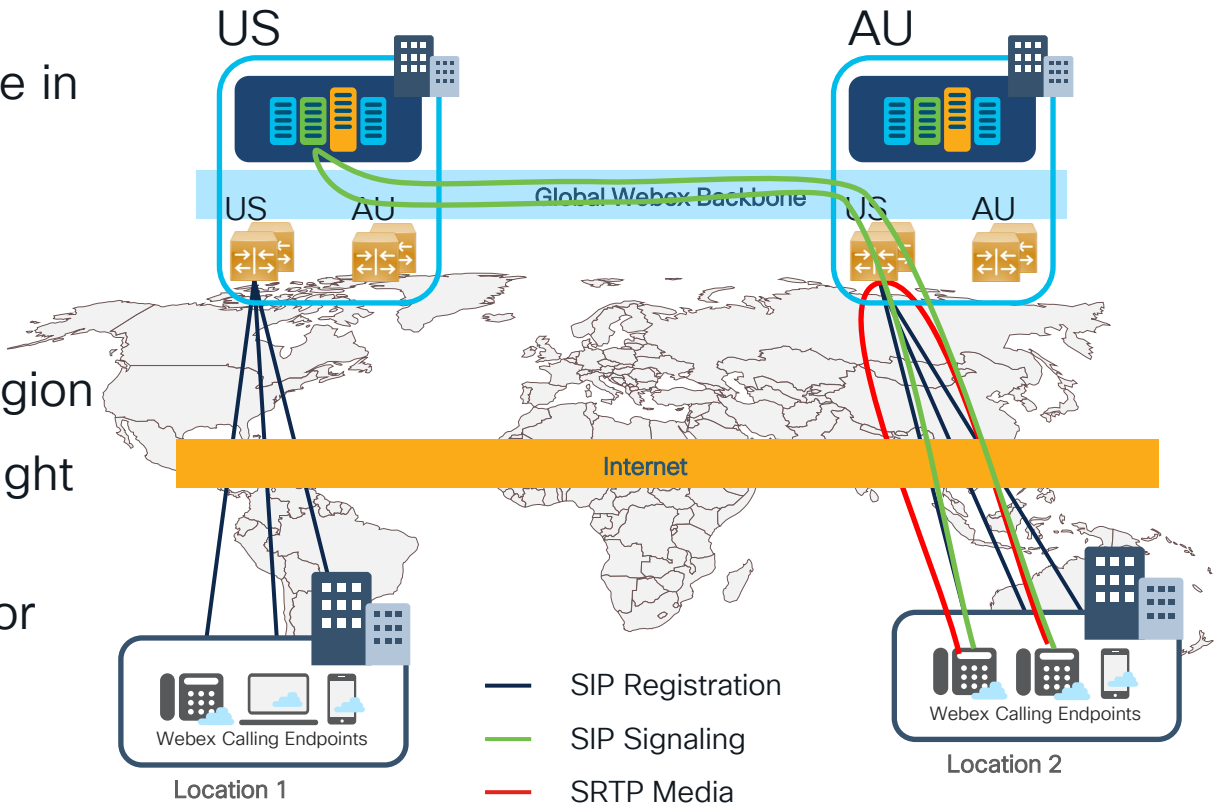
Multi-Region Customer

- Customers are tied to one region
- All registrations to that region
- Media anchored on access SBCs of “home” region
- Potentially negative impact:
  - Cut-through delay
  - Media RTT impact



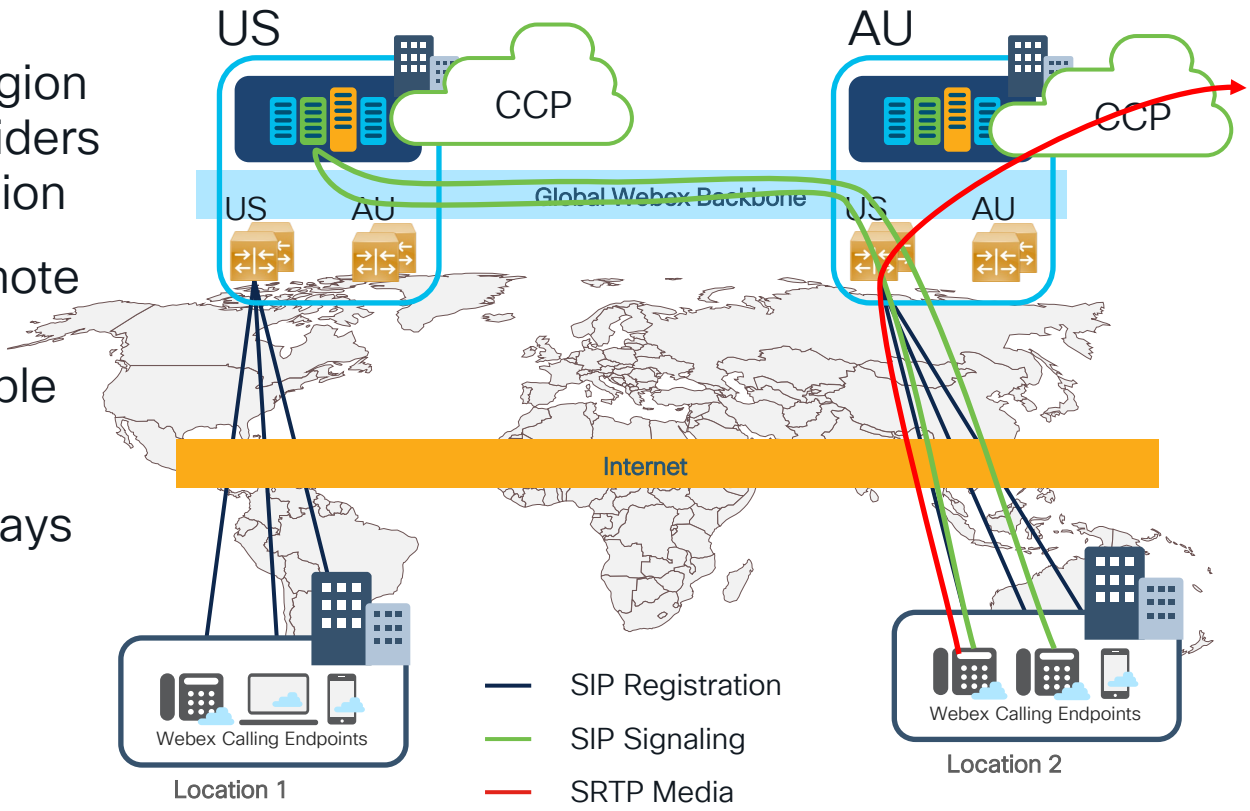
Multi-Region Customer: Regional Media

- Access SBC presence in other regions
- In-region registration
- In-region media
- Signaling still inter-region
- Cut-through delay might still occur
- Media RTT not a factor



Multi-Region Customer: Regional Media for Cloud Connected PSTN

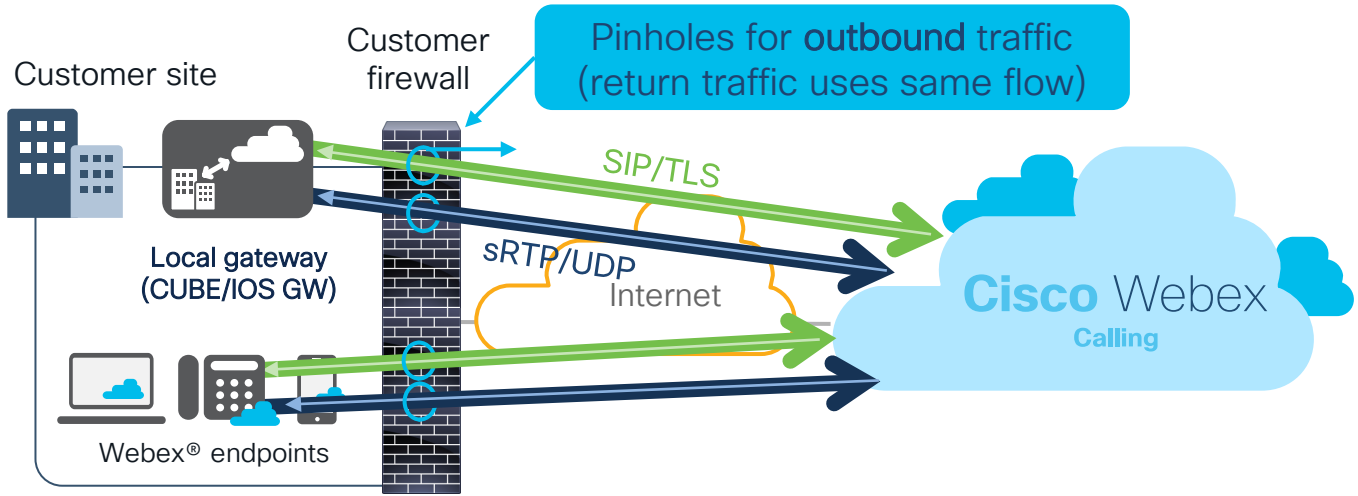
- Locations in home region can use all CCP providers available in home region
- New: locations in remote region can now use CCP providers available in remote region
- PSTN media traffic stays within region



Firewall Traversal and Media Path Optimization

CUBE as local gateway

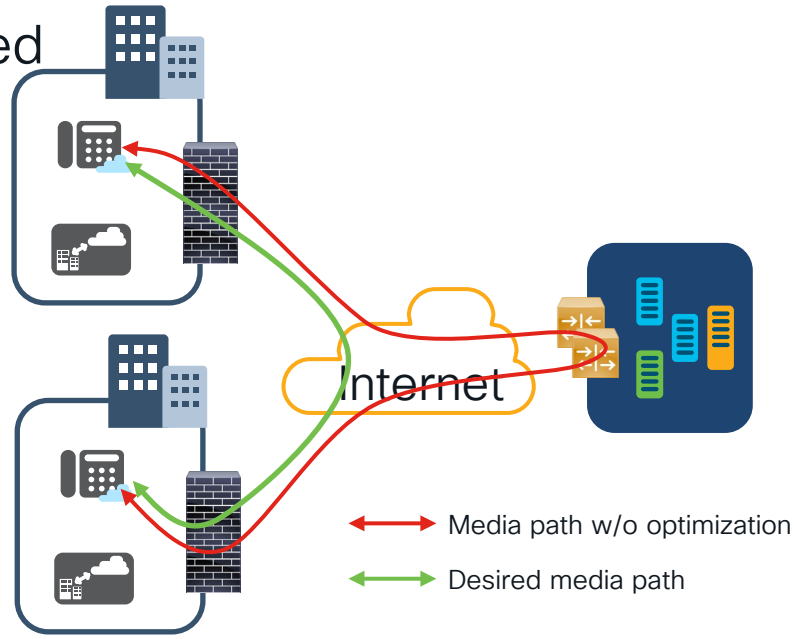
Firewall and NAT traversal



- In most cases, the local gateway and endpoints can sit on the internal customer network using private IP addresses (with NAT and PAT)
- Firewall needs to allow outbound traffic (SIP, RTP/UDP, HTTP) to specific IP addresses/ports (see updated Webex® Calling firewall and network configuration guide)
- Media Latching to establish downstream media path

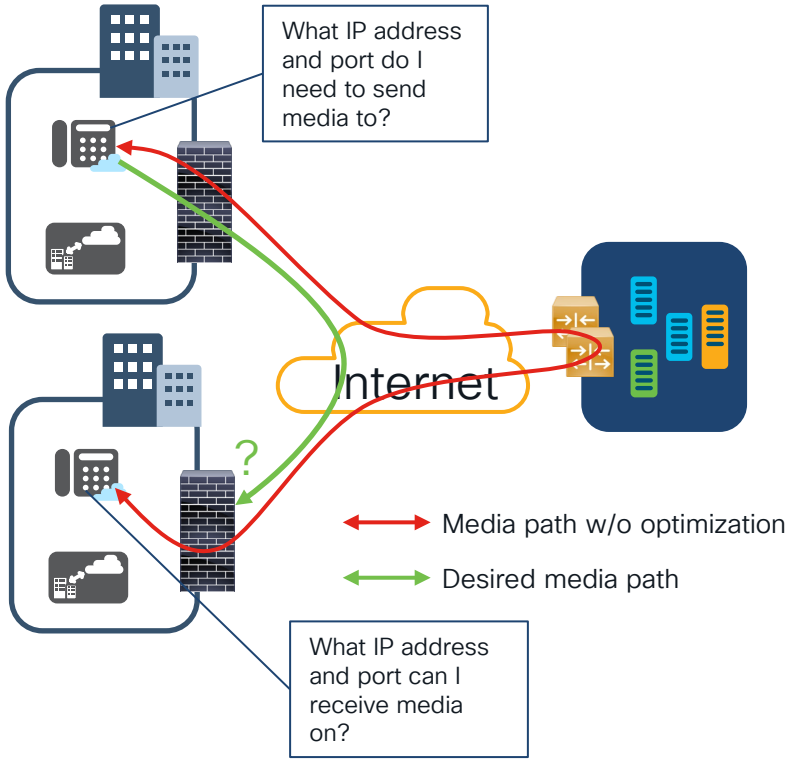
Media Path Optimization

- W/o optimization media is anchored at the Webex Calling access layer
- How can media path optimization be achieved?
- Media negotiation is part of SIP signaling (SDP)
- Endpoints need to know which transport addresses (IP address and port) to use to achieve optimized path



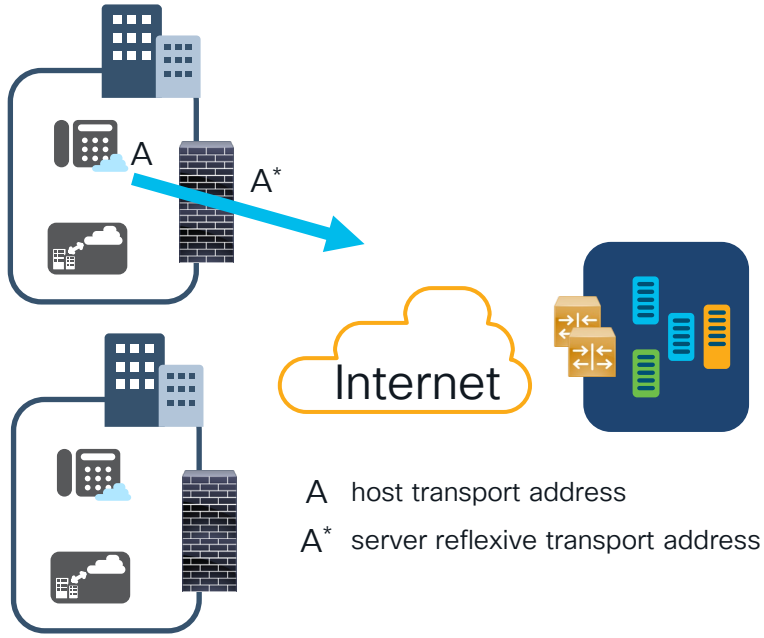
Media Path Optimization

- Required:
  - Devices need to determine useable transport address
  - Firewall needs to be opened
- First step:
  - Determine transport addresses



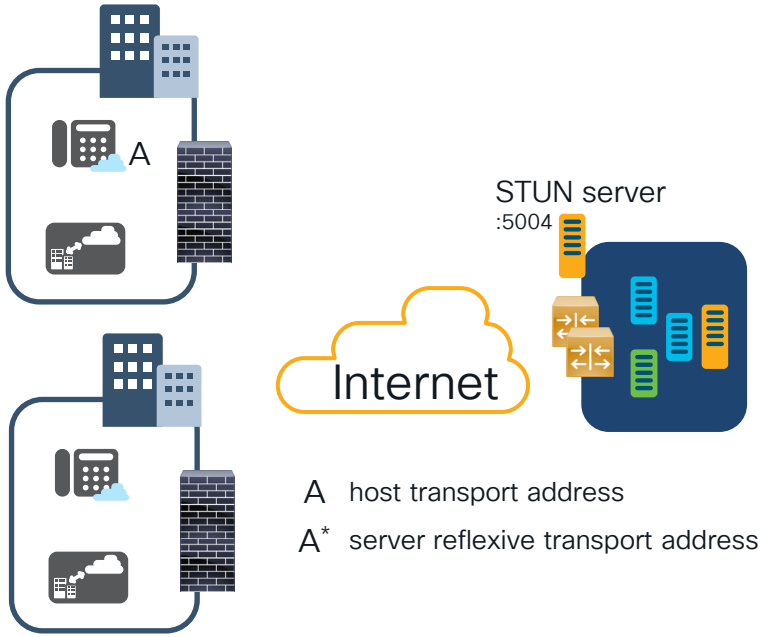
Transport addresses

- Required:
  - Devices need to determine useable transport address
  - Firewall needs to be opened
- First step:
  - Determine transport addresses
- Different types:
  - Host: address on local interface
  - Server reflexive: public (NATed)



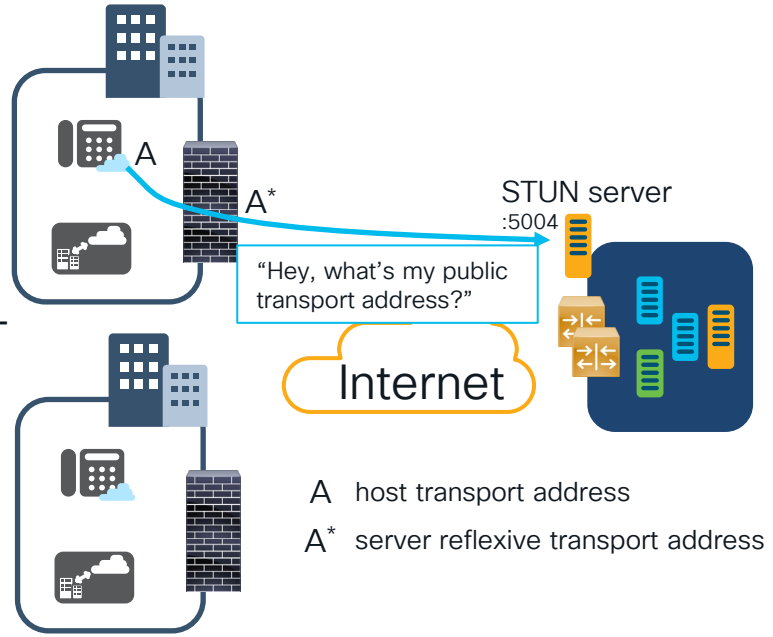
Determine Server Reflexive Address

- Need help: STUN server
  - Listening port: 5004 (standard 3478)



Determine Server Reflexive Address

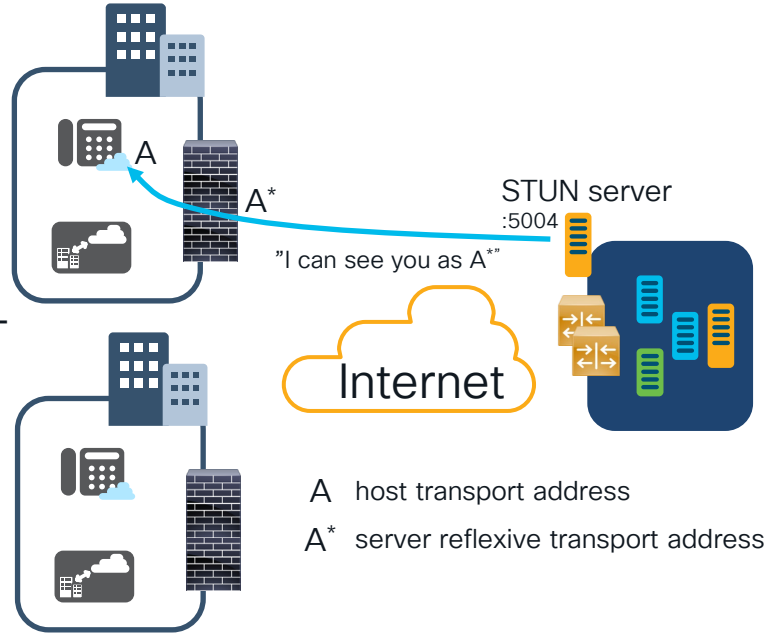
- Need help: STUN server
  - Listening port: 5004 (standard 3478)
- STUN binding request to STUN server
  - Source IP (and port) rewritten by NAT



Assumption here: all packets from the same source transport address are always mapped to the same public transport address → No symmetric NAT!

Determine Server Reflexive Address

- Need help: STUN server
  - Listening port: 5004 (standard 3478)
- STUN binding request to STUN server
  - Source IP (and port) rewritten by NAT
- STUN binding response contains server reflexive address
- Phone learns its public transport address(es)



Assumption here: all packets from the same source transport address are always mapped to the same public transport address → No symmetric NAT!

Candidate Exchange

- ICE clients exchange candidate transport addresses via SDP

```
a=candidate:1 1 UDP 10.10.10.1 19140 host
a=candidate:1 2 UDP 10.10.10.1 19141 host
a=candidate:3 1 UDP 192.88.99.101 23145 srflx raddr 10.10.10.1 rport 19140
a=candidate:3 2 UDP 192.88.99.101 23146 srflx raddr 10.10.10.1 rport 19141
```

Host candidate with private transport address

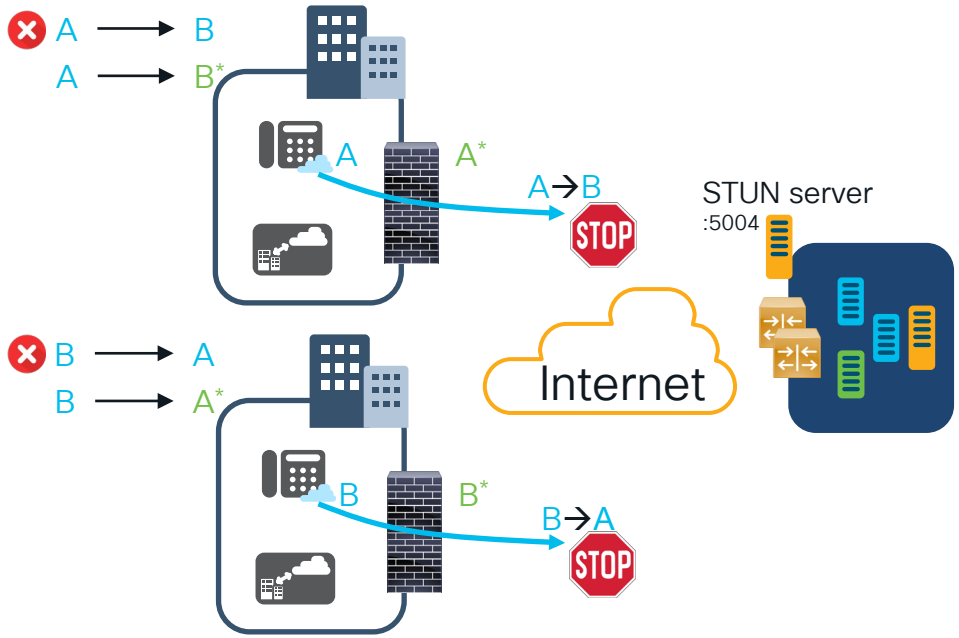
Server reflexive candidate

Public transport address

Private transport address

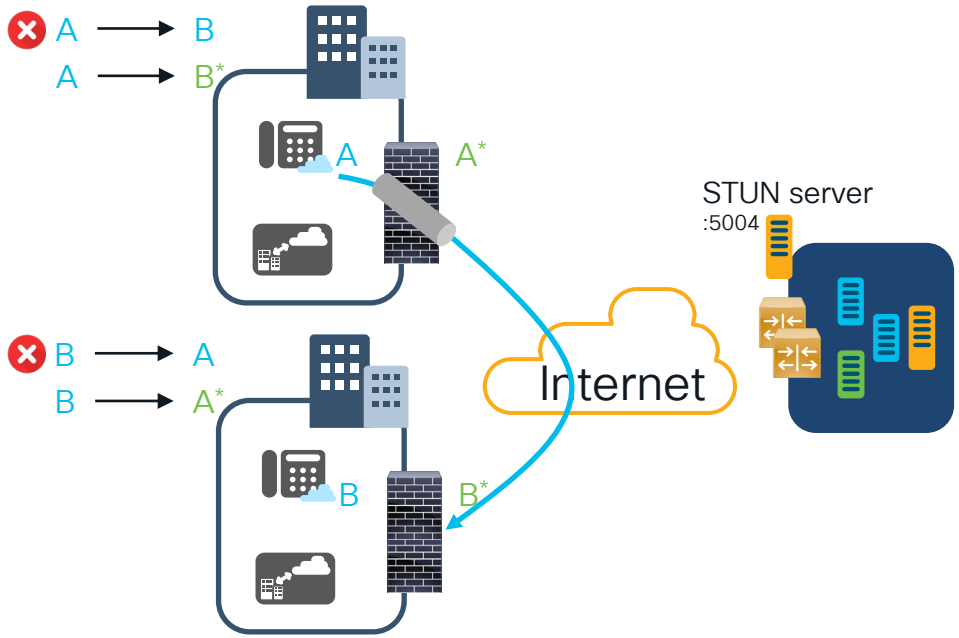
Connectivity Checks

- STUN bind requests
- Host to host fails
  - Private IP addresses
  - No response…



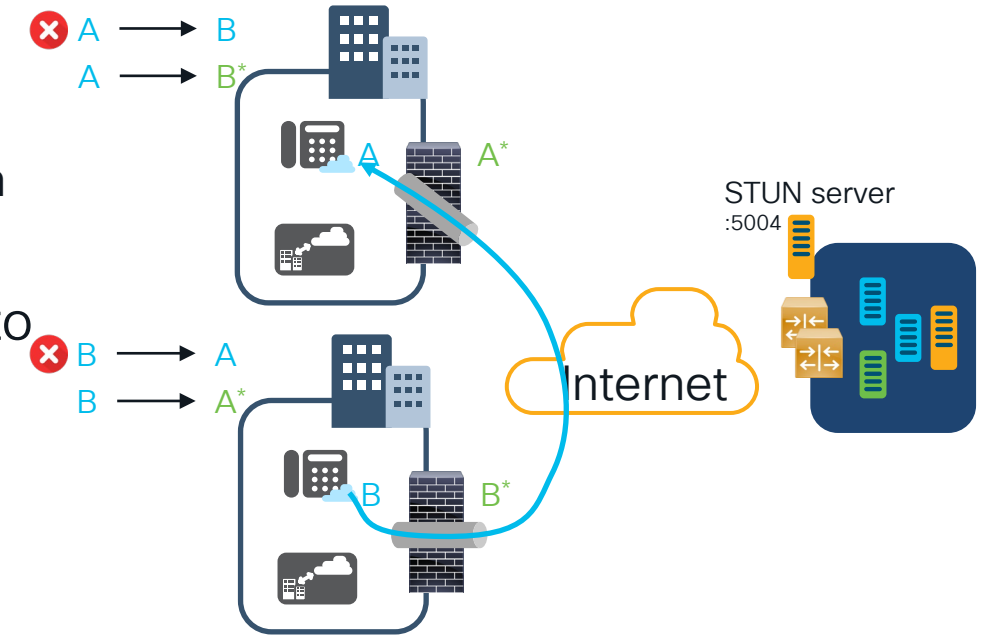
Connectivity Checks

- Bind request A→B* can't get through FW to B
  - Put punches a hole through the FW at A for A*



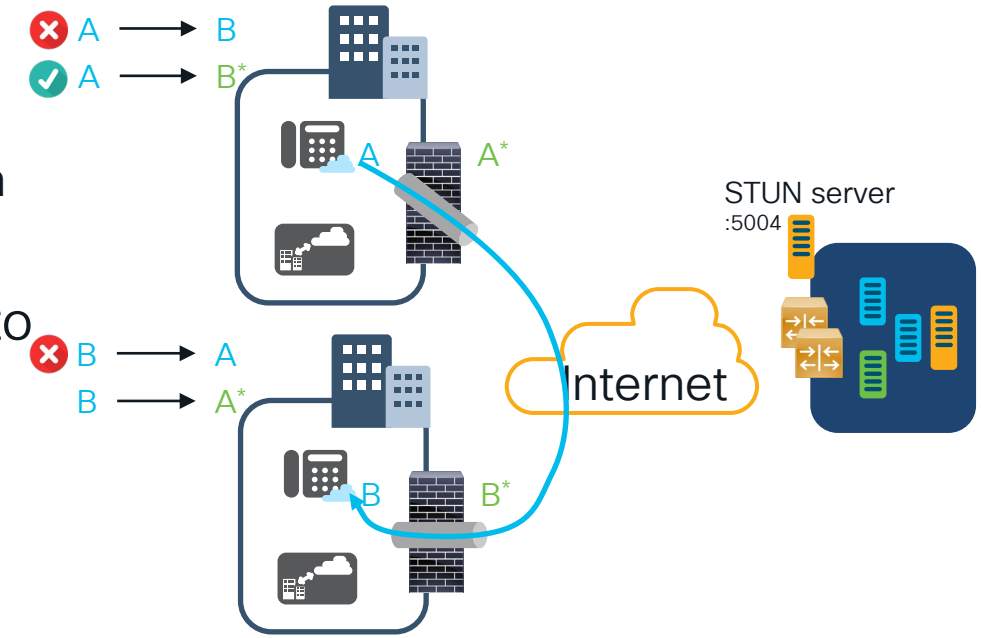
Connectivity Checks

- Bind request A→B* can't get through FW to B
  - Put punches a hole through the FW at A for A*
- Bind request B->A* gets to A via that hole



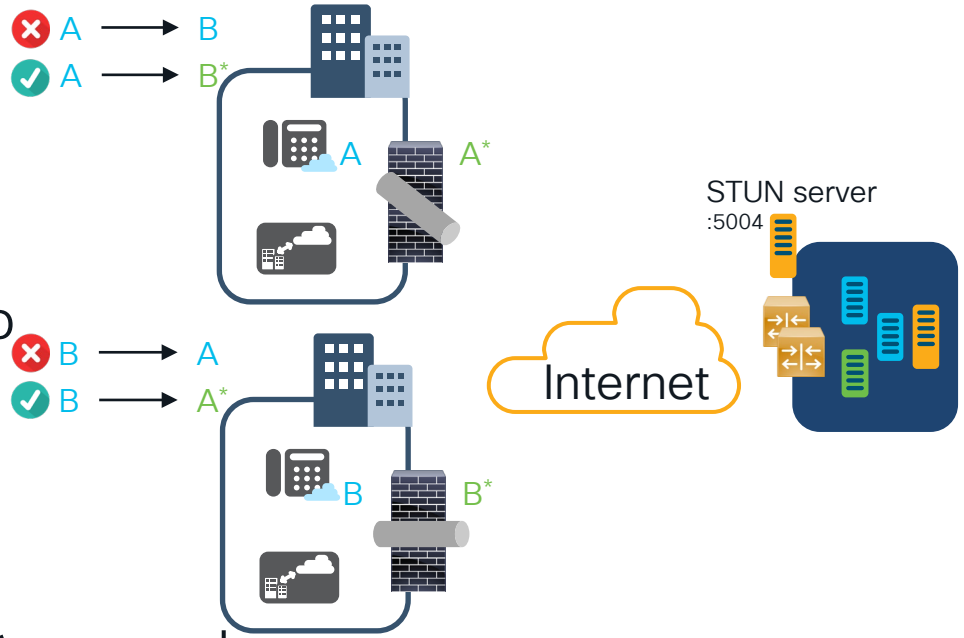
Connectivity Checks

- Bind request A→B* can't get through FW to B
  - Put punches a hole through the FW at A for A*
- Bind request B->A* gets to A via that hole
- A sends response
- Candidate found



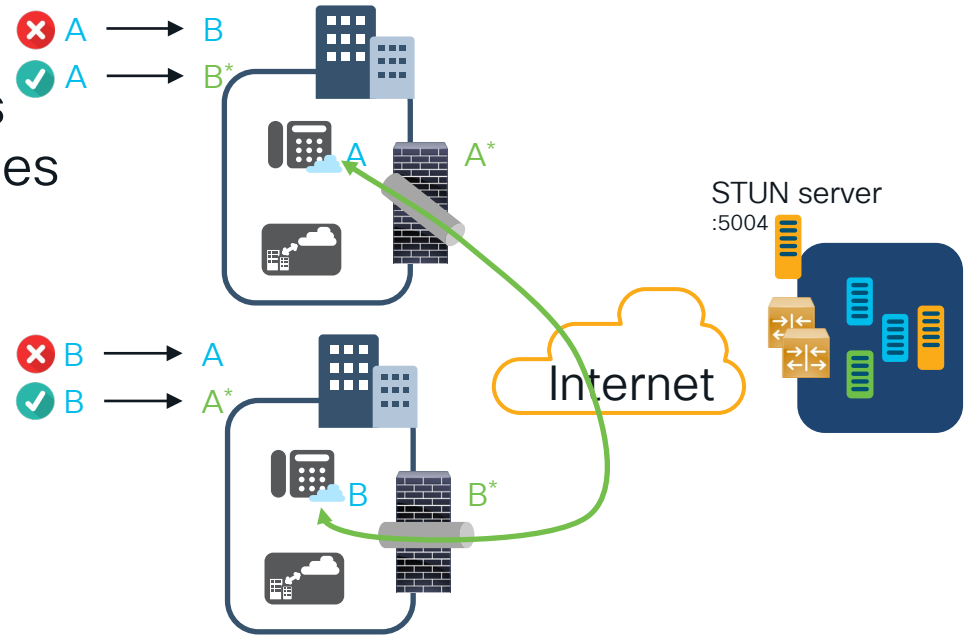
Connectivity Checks

- Bind request A→B* can't get through FW to B
  - Put punches a hole through the FW at A for A*
- Bind request B->A* gets to A via that hole
- A sends response
- Candidate found
- Subsequent request A→B* succeeds as well



Establish direct media

- Re-INVITE using the working candidate pairs as media addresses establishes direct media
- RTP stream uses holes punched through the FWs by STUN bind transactions



Local Gateway and “ICE Lite”

- Local Gateway only implements “ICE Lite”
- No STUN bind requests to determine server reflexive transport address
  - Only has host transport addresses
- Does not initiate connectivity checks
  - … but responds to STUN bind requests
- Consequence:
  - Media path optimization w/ Local Gateway can only use local host transport addresses of LGW

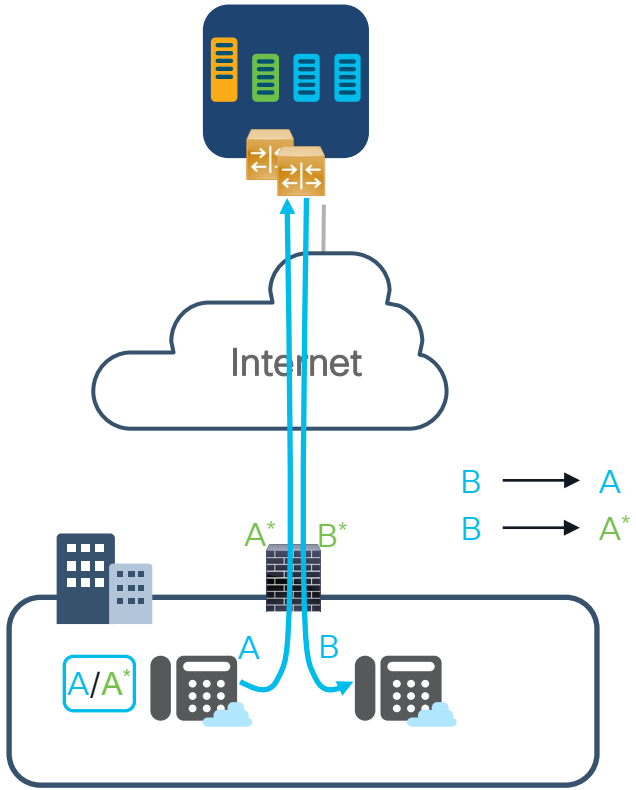
Summary

- Transport address: IP address and port
- STUN server to determine public server reflexive transport addresses
- Candidates
  - Host
  - Server reflexive
- Connectivity checks for all candidate pairs
  - STUN bind/response using transport addresses of candidate pairs
  - Host-Host
  - Host-Server reflexive
  - Punches holes through firewalls

Sample Media Flows

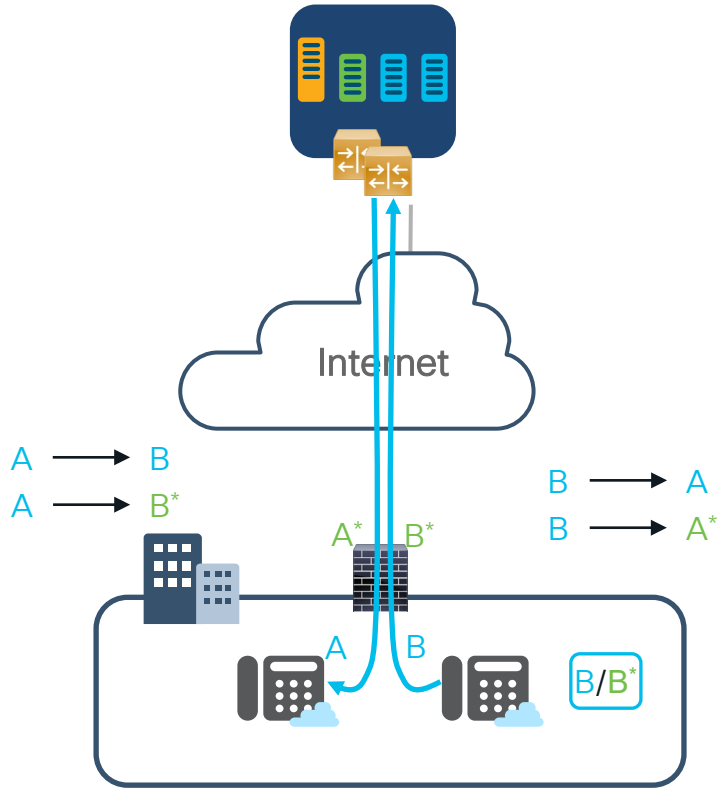
Webex Calling Phones in same Location

- A sends candidates in INVITE



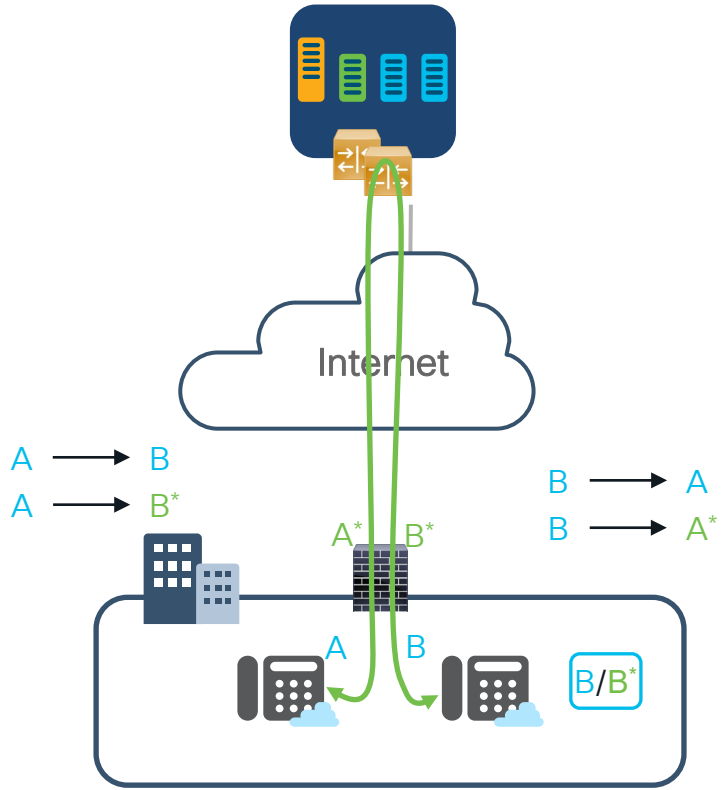
Webex Calling Phones in same Location

- A sends candidates in INVITE
- B's candidates in response



Webex Calling Phones in same Location

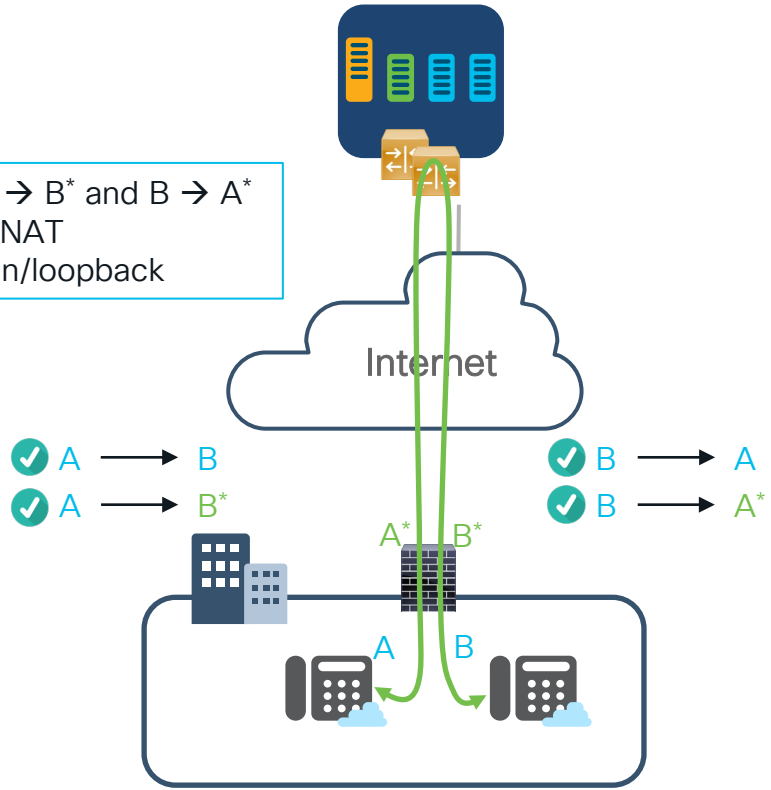
- A sends candidates in INVITE
- B's candidates in response
- Media established



Webex Calling Phones in same Location

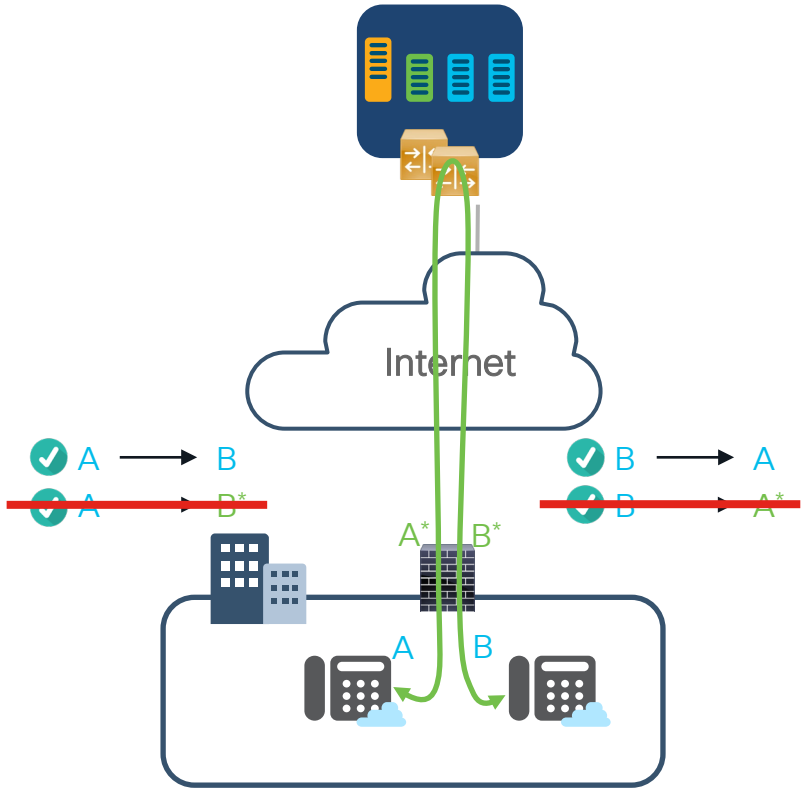
- A sends candidates in INVITE
- B's candidates in response
- Media established
- Connectivity checks for candidate pairs

Note: A → B* and B → A* fail w/o NAT reflection/loopback



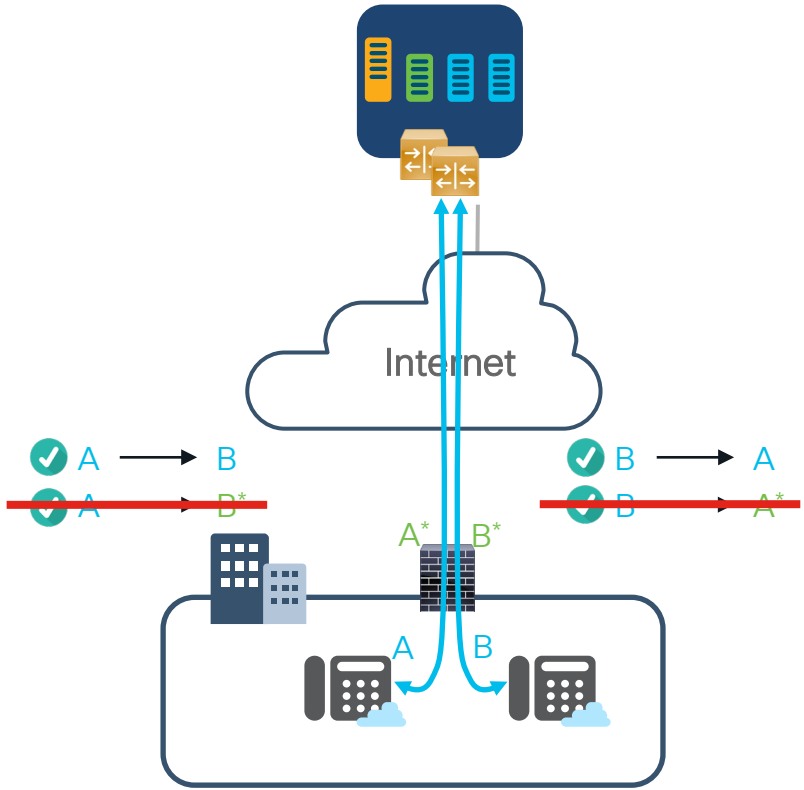
Webex Calling Phones in same Location

- A sends candidates in INVITE
- B's candidates in response
- Media established
- Connectivity checks for candidate pairs
- Best candidate pairs: host



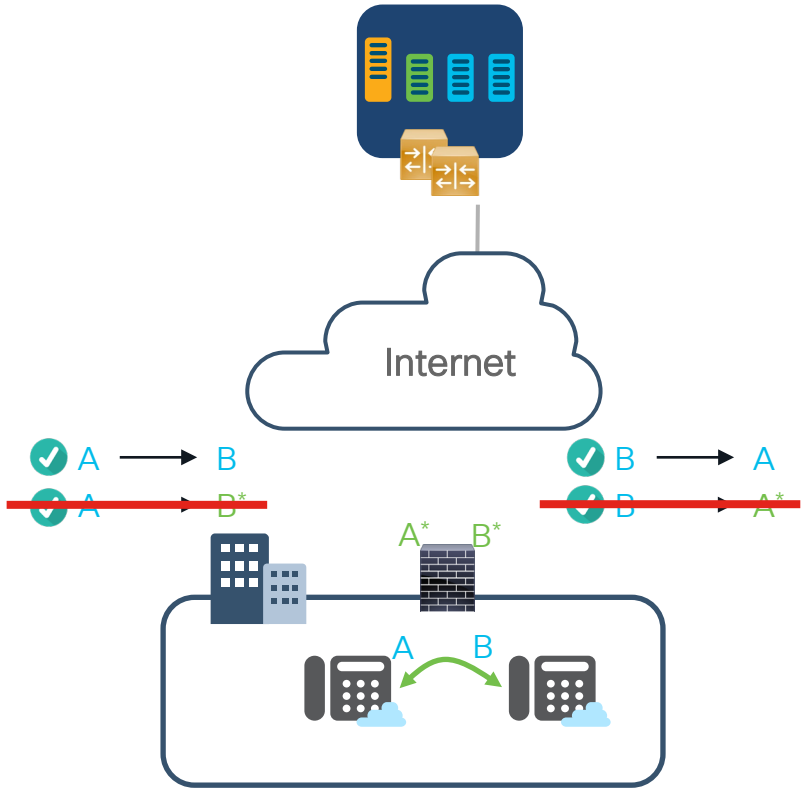
Webex Calling Phones in same Location

- A sends candidates in INVITE
- B's candidates in response
- Media established
- Connectivity checks for candidate pairs
- Best candidate pairs: host
- Re-INVITE to select optimized path



Webex Calling Phones in same Location

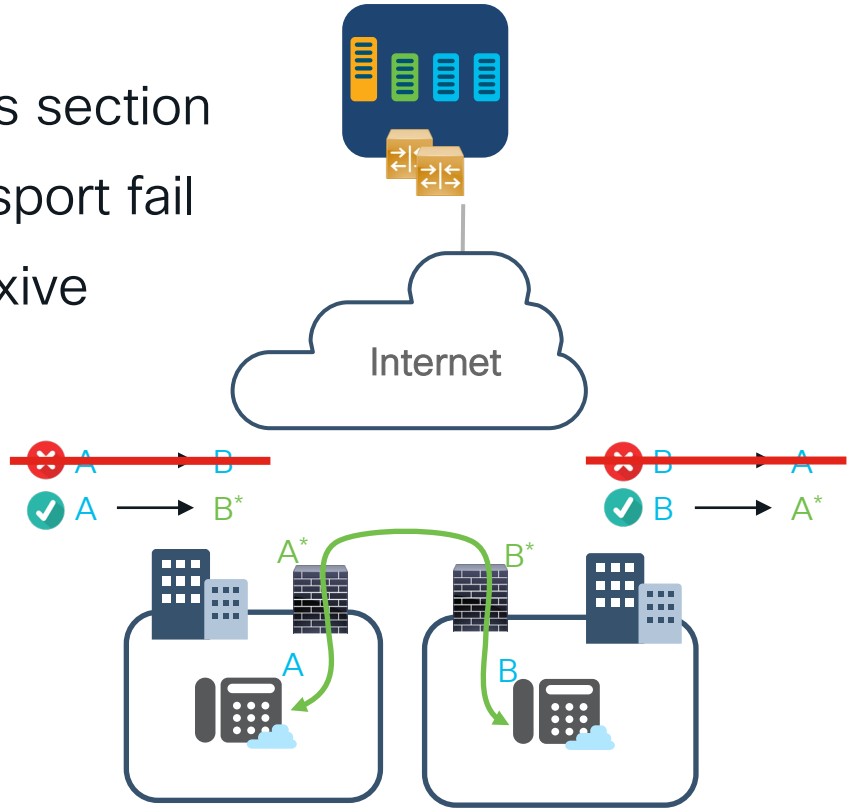
- A sends candidates in INVITE
- B's candidates in response
- Media established
- Connectivity checks for candidate pairs
- Best candidate pairs: host
- Re-INVITE to select optimized path
- Direct media is established



Phones in Different Locations

No WAN

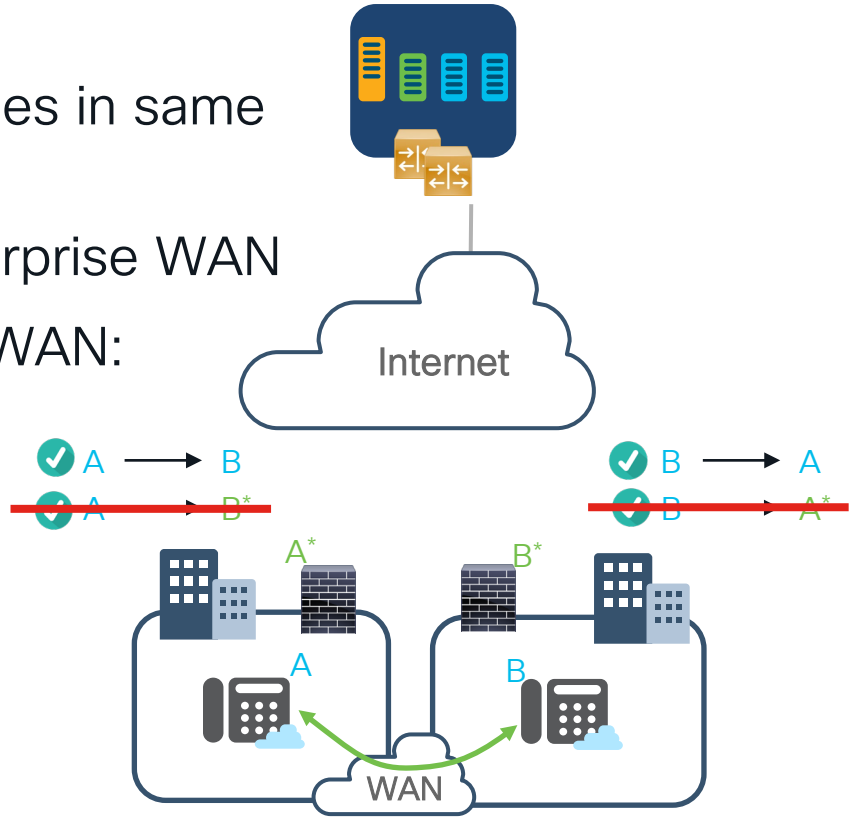
- Covered in more detail in previous section
- Connectivity checks for host transport fail
- Optimized path using server reflexive transport



Phones in Different Locations

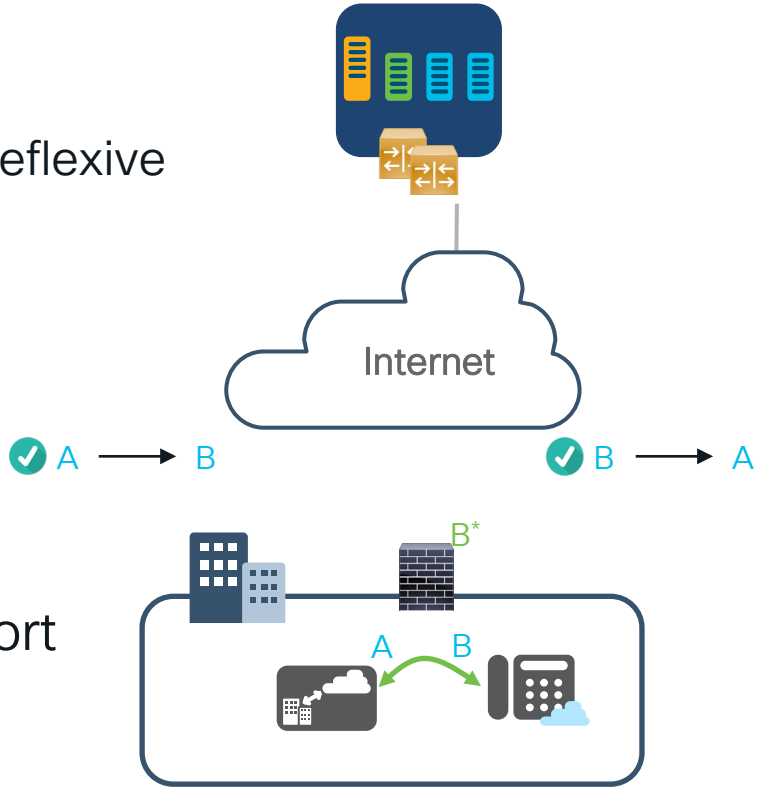
WAN

- Similar to scenario with two phones in same location
- Direct media established via enterprise WAN
- To keep real-time media off the WAN:
  - block UDP between phones on phone RTP media ports
  - This also blocks STUN binding



Phone and Local Gateway in same Location

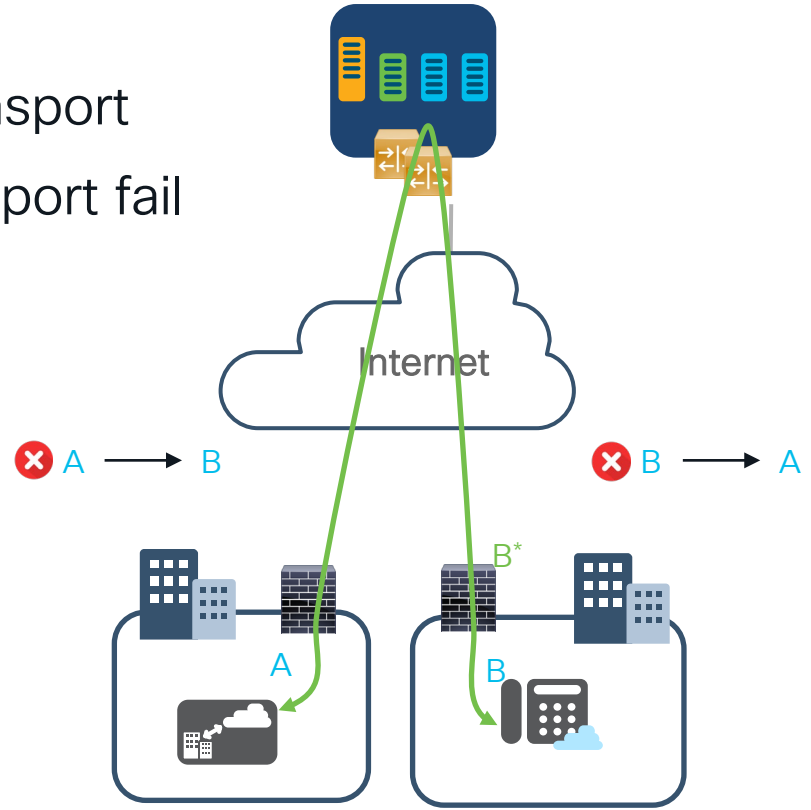
- ICE lite
  - Local Gateway does not have server reflexive address
- Limited set of candidate pairs
- Connectivity checks driven by B
- No STUN binding w/ any server reflexive addresses
- Connectivity checks for host transport succeeds



Phone and Local Gateway in Different Locations

Local Gateway behind NAT

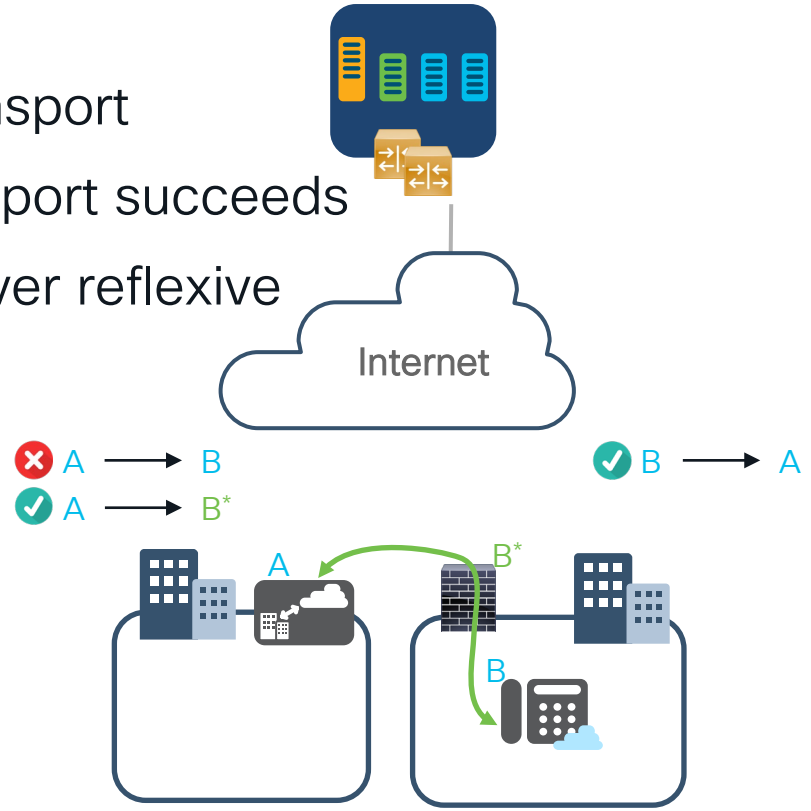
- Phone only has one pair: host transport
- Connectivity checks for host transport fail
- No media path optimization



Phone and Local Gateway in Different Locations

Local Gateway w/ public IP

- Phone only has one pair: host transport
- Connectivity checks for host transport succeeds
- Local Gateway uses pair host/server reflexive
- Local Gateway must have public IP address for media path optimization w/ phones in sites w/o WAN connectivity



Summary

Media Path Optimization

- Regional Media for in-region registrations
- Regional Media (Cloud Connected PSTN) to enable in-region Cloud Connected PSTN provider utilization
- Media Path Optimization (ICE)
  - MPP Phones, Local Gateway and Webex app
  - Decreased bandwidth utilization
  - Reduced latency
  - Improved media quality
  - Local Gateway consideration: public IP required for certain media flows

Analytics, Troubleshooting

- ICE status reported to platform as part of call metrics
- Analytics in Control Hub will be updated to allow reporting on ICE status
- Reduced RTT already is an indicator of successful media path optimization using ICE
- STUN BIND transactions can be used to monitor ICE negotiation at the network level

The bridge to possible

Thank you


cisco Live!
TURN
IT
UP
#CiscoLive