Appendix D – Claim Chart for US Patent No. 11,576,046 Against Accused Cisco Products

Based on information presently available, Damaka Inc. ("Damaka") contends that Cisco Systems, Inc. ("Cisco" or "Defendant") infringes Claims 1-4, 7-15, 17, and 19-20 (the "Asserted Claims") of U.S. Patent No. 11,576,046 (the "'046 Patent") through the Accused Products which are manufactured, sold, offered for sale, used, imported into the United States, and/or exported from the United States by Cisco.

The Accused Products comprise all versions and variants of the Webex Application and Webex SDKs[1], including, but not limited to, Webex for iOS, Webex for Android, Webex for Tablet, Webex for Web Browser, Webex for the Windows operating system, Webex for the Mac operating system, and Webex for Linux that are made, used, sold, offered for sale, imported into the United States, and/or exported from the United States or otherwise provided, after May 30, 2019. Damaka reserves the right to amend this list of Accused Products as discovery progresses.

Cisco directly infringes each of the Asserted Claims by using, importing, testing, selling, offering for sale, importing into the United States, and/or exporting from the United States the Accused Products in violation of 35 U.S.C. § 271(a).

Cisco indirectly infringes the Asserted Claims in violation of 35 U.S.C. § 271(b) by inducing third parties, including Cisco's customers and end-users, to directly infringe through their operation and use of the Accused Products. Cisco has knowingly and intentionally induced this direct infringement by, *inter alia*, manufacturing, selling, distributing, and/or otherwise making available the Accused Products, and providing instructions, documentation, and other information to customers and end-users suggesting that they use the Accused Products in an infringing manner, including technical support, marketing, product manuals, advertisements, and online documentation. Cisco's marketing and promotional materials for the Accused Products are found, for example, on the Webex website (https://www.webex.com/). For example, the Webex website instructs customers to, among other things, use the services in the Accused Products. *See* https://www.Webex.com/frequently-asked-questions. The Webex website also offers support to customers, including instruction on, among other things, how to operate their products. On information and belief, Cisco knows that its actions will result in infringement of the Asserted Claims, or subjectively believes that there is a high probability that its actions will result in infringement of the Asserted Claims but has taken deliberate actions to avoid learning these facts.

Cisco also contributorily infringes each of the Asserted Claims in violation of 35 U.S.C. § 271(c) by selling, importing, offering for sale, and otherwise providing the Accused Products, which when used by third parties, such as customers and end-users, directly infringe the Asserted Claims. The Accused Products constitute a material part of the Asserted Claims, are not staple articles or commodities of commerce, have no substantial non-infringing uses, and are known by Cisco to be especially made or adapted for use in the infringement of the '046 Patent. On information and belief, Cisco knows that its actions will result in infringement of the Asserted Claims, or subjectively believes that there is a high probability that its actions will result in infringement of the Asserted Claims but has taken deliberate actions to avoid learning these facts.

On information and belief, the charted version of the Webex application and Webex SDK is representative of all the Accused Products as functional equivalents of all claims exist in iOS, Android, and "Web."[2]

Damaka does not concede that any claims of the '046 Patent that are not listed below are not infringed by the identified Accused Products. Moreover, the citations to certain documents and other information below are intended to be exemplary only and in no way foreclose Damaka from citing or relying on additional documents, information, source code, and/or testimony at a later time. These contentions are preliminary in nature, and an analysis of Cisco's products, internal documentation, source code, and/or testimony from relevant witnesses may more fully and accurately describe the infringing features of its Accused Products. Accordingly, Damaka reserves the right to supplement, correct, modify, and/or amend these contentions once such additional information is made available to Damaka. Furthermore, Damaka reserves the right to supplement, correct, modify, and/or amend

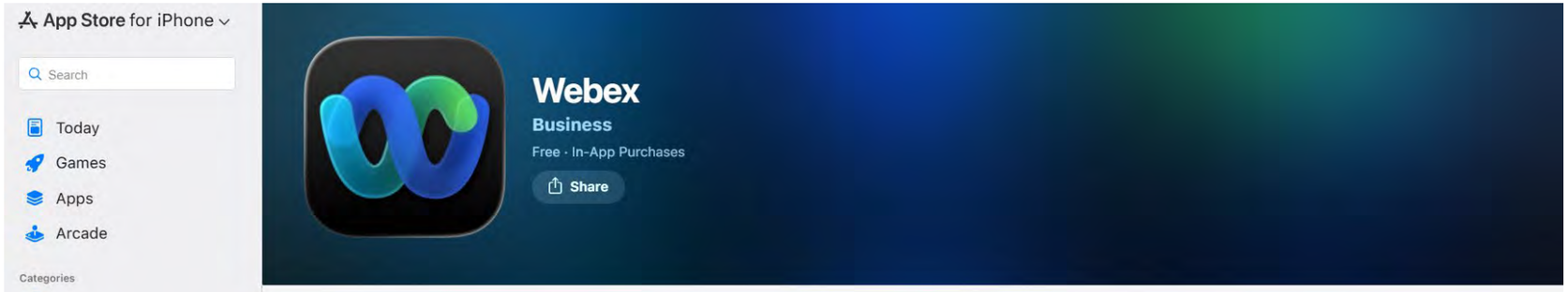
[1] The Webex SDK encompasses both the SDK as and associated Webex API. *See e.g.* https://developer.webex.com/meeting/docs/meetings

[2] *See e.g.* https://developer.webex.com/meeting/docs/meetings; *See also* https://help.webex.com/en-us/article/fz1e4b/System-requirements-for-Webex-services

these contentions as discovery in this case progresses; in view of the Court's claim construction order(s); in view of any positions taken by Cisco, including, but not limited to, positions on claim construction, invalidity, and/or non-infringement; and in connection with the preparation and exchange of expert reports.

To the extent Cisco contends that any element of the Accused Products is attributable to a third party, Damaka contends that the activities are attributable to Cisco such that they constitute direct infringement by Cisco under 271(a). The acts may be attributable to Cisco because Cisco directs or controls the others' performance, and because Cisco and the other entity form a joint enterprise. *Akamai Techs., Inc. v. Limelight Networks, Inc.*, 797 F.3d 1020, 1022 (Fed. Cir. 2015). Additionally, Cisco is vicariously liable for the activities of these other entities. *Centillion Data Sys., LLC v. Qwest Commc'ns Int'l, Inc.*, 631 F.3d 1279, 1286 (Fed. Cir. 2011). Further, the activities of these entities (including third party application developers, and users of the Accused Products consumers), are attributable to Cisco because Cisco (1) conditions participation in an activity or receipt of a benefit upon others' performance of one or more steps of a patented method, and (2) establishes the manner or timing of that performance. *Eli Lilly & Co. v. Teva Parenteral Medicines, Inc.*, 845 F.3d 1357, 1365 (Fed. Cir. 2017). These acts are also attributed to Cisco because it initiated the activities of their end-users. *SiRF Tech., Inc. v. Int'l Trade Comm'n*, 601 F.3d 1319, 1330 (Fed. Cir. 2010).

Claim	Cisco Accused Products
1p. A non-transitory computer readable medium embodying a computer program comprising instructions that include a function block and a superblock application, wherein the function block is accessible only to the superblock application on an electronic device, and wherein the function block is operative when executed by a processor of the electronic device to:	The preamble of claim 1 is nonlimiting. Nonetheless, to the extent the preamble limits the claim, each Accused Product meets the requirement of "[a] non-transitory computer readable medium embodying a computer program comprising instructions that include a function block and a superblock application, wherein the function block is accessible only to the superblock application on an electronic device." For example, Cisco's Webex (superblock application) includes the Cisco Webex SDK (function block). The Webex SDK is directly compiled into the Webex app, and is therefor only accessible by it. For example, Cisco's Webex is a computer program.

Claim	Cisco Accused Products
	 https://apps.apple.com/us/app/webex/id833967564 As shown below "Webex Meetings [SDK] offers integrated audio, video…" **Webex Meetings** Webex Meetings offers integrated audio, video, and content sharing with highly secure web meetings from the cloud. The Webex Meetings REST API allows developers to add basic Webex scheduling functionality to their custom applications or websites. You can: • Create and manage meetings • Add and manage meeting invitees • Get and update meeting preferences, options, and other details • Get, list, and delete recordings https://developer.webex.com/meeting/docs/meetings

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
1a. establish a connection with an external device via a network interface to provide a service requested by the superblock application without switching context from the superblock application to another application on the electronic device.	Each Accused Product literally meets the requirement "establish a connection with an external device via a network interface to provide a service requested by the superblock application without switching context from the superblock application to another application on the electronic device." For example, on can join meetings via the Webex SDK, as shown below. For example, the function block (Webex SDK) can receive a "Join_a_meeting" call. This allows the superblock application "to start or join a meeting"

Join a Meeting

Operation Id: *Join_a_Meeting*

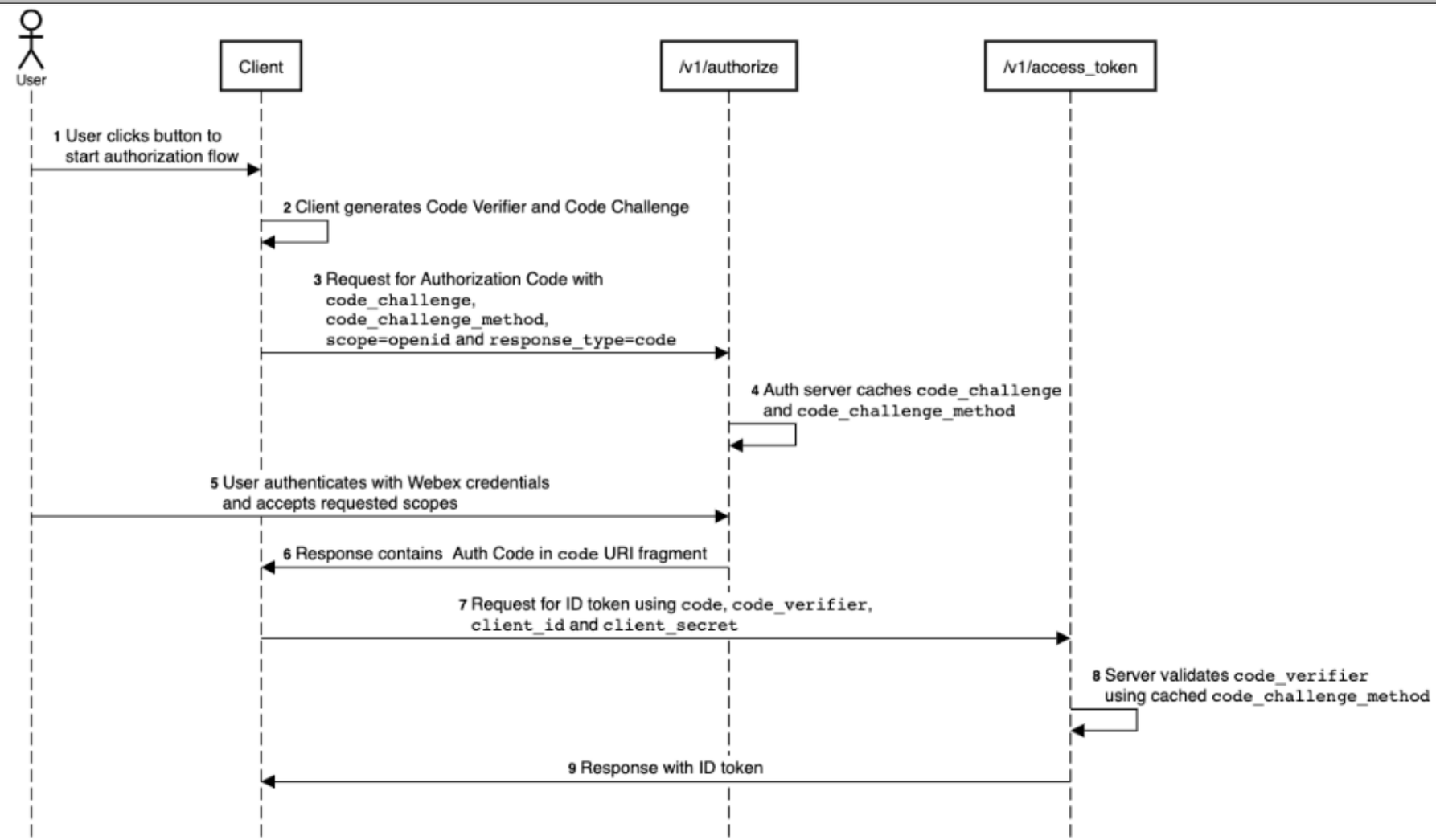
Description: *Retrieves links for a meeting with a specified* `meetingId`, `meetingNumber`, *or* `webLink` *that allow users to start or join the meeting directly without logging in and entering a password.*

- *Please note that* `meetingId`, `meetingNumber` *and* `webLink` *are mutually exclusive and they cannot be specified simultaneously.*
- *If* `joinDirectly` *is true or not specified, the response will have HTTP response code 302 and the request will be redirected to* `joinLink`*; otherwise, the response will have HTTP response code 200 and* `joinLink` *will be returned in response body.*
- *Only the meeting host or cohost can generate the* `startLink`.
- *An admin user or a Service App can generate the* `startLink` *and* `joinLink` *on behalf of another meeting host using the* `hostEmail` *parameter. When a Service App generates the* `startLink` *and* `joinLink`*, the* `hostEmail` *parameter is required. The* `hostEmail` *parameter only applies to meetings, not webinars.*
- *For Service Apps,* `hostEmail` *must be provided in the request.*
- *Generating a join link or a start link before the time specified by* `joinBeforeHostMinutes` *for a webinar is not supported.*
- *The* `joinLink` *and* `startLink` *generated by the API only work in a web browser, not in mobile apps. When the* `joinLink` *or* `startLink` *is used in a mobile app, the user can't join or start the meeting directly by the dial-in or dial-out phone numbers displayed in the app.*
- *When* `createJoinLinkAsWebLink` *or* `createStartLinkAsWebLink` *is set to true, a user cannot join or start the meeting using the* `joinLink` *or* `startLink` *returned in the response, and must complete the login flow. Those options are typically useful if mandatory user login is configured in Control Hub.*
- *Appending the parameter* `&launchApp=true` *to the weblink will launch directly into the browser experience without showing the user the download selection for the native client.*
- *Embedded Apps are not supported in a meeting which is started from the* `startLink` *generated by this API.*

Claim	Cisco Accused Products

	https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting
	
	There is also equivalent functionality in the formal SDK, as demonstrated by the below.
	
	Fetch a Meeting Instance
	
	A `meeting` instance is typically created once a meeting is initiated through the SDK. If a meeting has already been created on the SDK, you can retrieve it using the SDK's `meetings` object by passing a `meetingId`.
	
	This method does not fetch the list of all meetings from the Webex cloud. It only lists the meetings that are created either with `webex.meetings.create` or `webex.meetings.syncMeetings`.
	
	``
	const meeting = webex.meetings.getAllMeetings()[meetingId];
	``
	
	If participants have already joined the meeting, it can be retrieved from Webex Cloud. Refer to Sync Meetings.
	
	Join a Meeting
	
	Once you have a `Meeting` object, you can join the meeting using the `join()` method:
	
	``
	const join = await meeting.join(joinOptions);
	``
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-join-a-meeting

Claim	Cisco Accused Products
	The Webex SDK must connect to the internet, which it does via the devices network interface. For example, when authorizing the superblock application, the function block (Webex SDK) must contact Cisco servers that are external to the device the Webex application is on. As shown below, the function block (Webex SDK) makes a request to a Cisco server to “validate[].”

Claim	Cisco Accused Products
	

Claim	Cisco Accused Products

	https://developer.webex.com/create/docs/login-with-webex
	
	If authorization is not successful, the function block will throw a 401 error, as shown below.
	
	Status: 401
	
	Unauthorized: Authentication credentials were missing or incorrect.
	
	https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting
	
	There is analogous functionality within the formal Webex SDK.
	
	Initialize the SDK
	
	To use the SDK, you'll need a free Webex account; if you don't have a Cisco Webex account, visit Cisco Webex for Developers to create your account and retrieve your **access token** from Getting and Using your Personal Access Token.
	
	To create a new Webex instance:
	
	``
	const webex = Webex.init({
	  credentials: {
	  access_token: `<AUTHORIZATION/BOT TOKEN>`
	  }
	});
	``
	
	At this point, the Webex `Meetings` object is ready for use. The `Meetings` object contains a list of meetings as well as methods to perform specific operations or listen for certain events. Detailed information is included in this article.

Claim	Cisco Accused Products
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-authorization-device-registration For example, authentication credentials must be acquired to use the Webex SDK. Without proper authorization, Webex SDK will not function, as evidenced by the below. What Are Integrations? Integrations are how you request permission to invoke the Webex REST API on behalf of another Webex user. To do this in a secure way the API supports the OAuth 2 standard which allows third-party integrations to get a temporary access token for authenticating API calls instead of asking users for their password. If you're sure that your integrations require authenticating on behalf of another Webex user, read on, we'll get you there in a few easy steps: 1. Register your integration with Webex 2. Request permission using an OAuth Grant Flow 3. Exchange the resulting authorization code for an access token 4. Use the access token to make your API calls https://developer.webex.com/create/docs/integrations Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
2a. The non-transitory computer readable medium of claim 1 wherein the function block is further operative when executed by the processor to: receive from	Each Accused Product literally meets the requirement "wherein the function block is further operative when executed by the processor to: receive from the superblock application a request for the service, wherein the service is a communication session, and wherein the function block is configured to provide functionality for the communication session to the superblock application."

Claim	Cisco Accused Products
the superblock application a request for the service, wherein the service is a communication session, and wherein the function block is configured to provide functionality for the communication session to the superblock application; and	The Webex SDK is configured to receive a request from the superblock application for a communication. For example, the Webex SDK (function block) receives a call from Webex (the superblock application) for A/V capabilities. As shown below "Webex Meetings [SDK] offers integrated audio, video…" **Webex Meetings** Webex Meetings offers integrated audio, video, and content sharing with highly secure web meetings from the cloud. The Webex Meetings REST API allows developers to add basic Webex scheduling functionality to their custom applications or websites. You can: • Create and manage meetings • Add and manage meeting invitees • Get and update meeting preferences, options, and other details • Get, list, and delete recordings https://developer.webex.com/meeting/docs/meetings For example, the function block (Webex SDK) can receive a "Join_a_meeting" call. This will allows the superblock application "to start or join a meeting"

	Join a Meeting **Operation Id:** *Join_a_Meeting* **Description:** *Retrieves links for a meeting with a specified `meetingId`, `meetingNumber`, or `webLink` that allow users to start or join the meeting directly without logging in and entering a password.* • *Please note that `meetingId`, `meetingNumber` and `webLink` are mutually exclusive and they cannot be specified simultaneously.* • *If `joinDirectly` is true or not specified, the response will have HTTP response code 302 and the request will be redirected to `joinLink`; otherwise, the response will have HTTP response code 200 and `joinLink` will be returned in response body.* • *Only the meeting host or cohost can generate the `startLink`.* • *An admin user or a Service App can generate the `startLink` and `joinLink` on behalf of another meeting host using the `hostEmail` parameter. When a Service App generates the `startLink` and `joinLink`, the `hostEmail` parameter is required. The `hostEmail` parameter only applies to meetings, not webinars.* • *For Service Apps, `hostEmail` must be provided in the request.* • *Generating a join link or a start link before the time specified by `joinBeforeHostMinutes` for a webinar is not supported.* • *The `joinLink` and `startLink` generated by the API only work in a web browser, not in mobile apps. When the `joinLink` or `startLink` is used in a mobile app, the user can't join or start the meeting directly by the dial-in or dial-out phone numbers displayed in the app.* • *When `createJoinLinkAsWebLink` or `createStartLinkAsWebLink` is set to true, a user cannot join or start the meeting using the `joinLink` or `startLink` returned in the response, and must complete the login flow. Those options are typically useful if mandatory user login is configured in Control Hub.* • *Appending the parameter `&launchApp=true` to the weblink will launch directly into the browser experience without showing the user the download selection for the native client.* • *Embedded Apps are not supported in a meeting which is started from the `startLink` generated by this API.*

Claim	Cisco Accused Products

	https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting
	
	There is also equivalent functionality in the formal SDK, as demonstrated by the below.
	
	Fetch a Meeting Instance
	
	A `meeting` instance is typically created once a meeting is initiated through the SDK. If a meeting has already been created on the SDK, you can retrieve it using the SDK's `meetings` object by passing a `meetingId`.
	
	This method does not fetch the list of all meetings from the Webex cloud. It only lists the meetings that are created either with `webex.meetings.create` or `webex.meetings.syncMeetings`.
	
	``
	const meeting = webex.meetings.getAllMeetings()[meetingId];
	``
	
	If participants have already joined the meeting, it can be retrieved from Webex Cloud. Refer to Sync Meetings.
	
	Join a Meeting
	
	Once you have a `Meeting` object, you can join the meeting using the `join()` method:
	
	``
	const join = await meeting.join(joinOptions);
	``
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-join-a-meeting

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
2b. notify the superblock application that the communication session is being requested.	Each Accused Product literally meets the requirement "notify the superblock application that the communication session is being requested." For example, the Webex SDK has capabilities to notify a superblock application that a meeting has started (*i.e.* that a communication is being requested). The Webex SDK accomplishes this through the "Create_a_Meeting_Invitee" which invites another to a meeting. Invitees Create a Meeting Invitee **Operation Id:** *Create_a_Meeting_Invitee* **Description:** • *Invite a person to attend a meeting.* • *Identify the invitee in the request body, by email address.* • *The `sendEmail` parameter is `true` by default and the meeting emails will be sent to the invitee's `email`. Please set `sendEmail` to `false` to prevent the invitee from receiving emails.* POST /meetingInvitees https://developer.webex.com/meeting/docs/api/v1/meeting-invitees/create-a-meeting-invitee

Claim	Cisco Accused Products

	Additionally, the Webex SDK provides functionality to admit participants when they are ready to join the meeting session.
	
	Admit Participants
	
	Operation Id: *Admit_Participants*
	
	Description: *Admit multiple participants to a meeting in progress.*
	
	This API limits the maximum size of `items` in the request body to 100.
	
	Each `participantId` of `items` in the request body should have the same prefix of `meetingId`.
	
	https://developer.webex.com/meeting/docs/api/v1/meeting-participants/admit-participants
	
	For example, the SDK has a similar functionality, as demonstrated below.
	
	Invite another participant
	
	A participant can be invited to the meeting using the invite method within the meeting object. Only a host can invite another participant. Participants can be invited using an email address.
	
	Use the following code to invite another participant to an existing meeting. You can use either the `emailAddress` or the `phoneNumber` parameter.
	
	``
	meeting.invite(invitee: {
	  emailAddress: 'johndoe@abc.com',
	  phoneNumber: '1234',
	});
	``
	
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-migration-unified-spaces-meetings

Claim	Cisco Accused Products

	Additionally, the Webex SDK has numerous event handling options, as demonstrated below:
	
	``
	document.getElementById('destination').addEventListener('submit', (event) => {
	  // again, we don't want to reload when we try to join
	  event.preventDefault();
	
	  const destination = document.getElementById('invitee').value;
	
	  return webex.meetings.create(destination).then((meeting) => {
	  // Call our helper function for binding events to meetings
	  bindMeetingEvents(meeting);
	
	  return joinMeeting(meeting);
	  })
	  .catch((error) => {
	  // Report the error
	  console.error(error);
	  });
	});
	``
	
	https://github.com/WebexSamples/webex-meetings-quick-start/blob/main/index.js
	
	For example, Cisco servers sends various notifications to the function blocks. For example, the Webex SDK has the following webhooks, which are passed up to the superblock application:

Claim	Cisco Accused Products
	A webhook is an HTTP callback, or an HTTP POST, to a specified URL that notifies your app when a particular activity or "event" has occurred in one of your resources on the Webex platform. The benefit of using webhooks is that they allow your application to receive real-time data from Webex, so you can keep up with the state of your resources (i.e. rooms, messages, memberships, etc.).

Claim	Cisco Accused Products
	`meetings` `created` A new meeting was created. `id` – Meeting Id. `meetingNumber` – Meeting number. `meetingType` – Meeting type. `state` – Meeting state. `hostEmail` – Email address for the meeting host. `hostUserId` – Unique identifier for meeting host. `siteUrl` – Webex site URL to query. `meetings` `updated` A meeting that you're in was updated. `id` – Meeting Id. `meetingNumber` – Meeting number. `meetingType` – Meeting type. `state` – Meeting state. `hostEmail` – Email address for the meeting host. `hostUserId` – Unique identifier for meeting host. `siteUrl` – Webex site URL to query. `meetings` `deleted` A meeting that you're in was deleted. `id` – Meeting Id. `meetingNumber` – Meeting number. `meetingType` – Meeting type. `state` – Meeting state. `hostEmail` – Email address for the meeting host. `hostUserId` – Unique identifier for meeting host. `siteUrl` – Webex site URL to query. https://developer.webex.com/meeting/docs/api/guides/webhooks

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
3. The non-transitory computer readable medium of claim 2 wherein the steps of receiving, notifying, and establishing are performed without switching context from the superblock application to another application on the electronic device.	Each Accused Product literally meets the requirement "wherein the steps of receiving, notifying, and establishing are performed without switching context from the superblock application to another application on the electronic device." For example, none of the steps in claim 2 require leaving the Webex application. *See* support for claim 2. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
4. The non-transitory computer readable medium of claim 2 wherein the function block is further operative when executed by the processor to determine whether the function block has a capability to perform any rendering needed by	Each Accused Product literally meets the requirement "wherein the function block is further operative when executed by the processor to determine whether the function block has a capability to perform any rendering needed by the communication session." For example, the Webex SDK has the following resolution capabilities.

Claim	Cisco Accused Products
the communication session.	Prerequisites Keep the following prerequisites in mind: • The device camera must support **full HD (1080p)**; if not, the feature is downgraded to the camera's maximum resolution support. To check for FHD support, see Webcam Test Resolution. • Make sure that you're using version webex-js-sdk@2.32.0 or above. For SDK versions 2.32.0 and above, the value mappings used for resolutions are LOW: 480p, MEDIUM: 720p, and HIGH 1080p. For earlier SDK versions: LOW: 360p, MEDIUM: 480p, and HIGH: 720p. Local Resolution The sharpness, smoothness, and overall quality of the video are directly linked to the frame rate, bitrate, and other video settings. The following table specifies the different resolution settings available. **Supported Sending Video Resolutions** Video Profile | Resolution (Width×Height) | Frame Rate (fps) | Chrome | Firefox | Safari 360p | 640 × 360 | 30 | ✓ | ✓ | ✓ 480p | 640 × 480 | 30 | ✓ | ✓ | ✓ 720p | 1280 × 720 | 30 | ✓ | ✓ | ✓ 1080p | 1920 × 1080 | 30 | ✓ | ✓ | ✓

Claim	Cisco Accused Products
	https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-video-resolution For example, the Webex SDK will throw the following errors if either the superblock or function block cannot render the A/V call: **Remote Video Resolution Errors** The following table lists possible errors when working with remote video resolutions: Method / Error / Fix `setLocalVideoTrack()` / `Meeting:index#setLocalVideoTrack --> Local video quality of "${localQualityLevel}" not supported, downscaling to the highest possible resolution of "${height}p"` / Remote resolution is set to 720p. `setRemoteVideoConstraints()` / `setRemoteVideoConstraints: unable to set max framesize, value for level "${level}" is not defined` / Use the defined resolution values. `getMediaStreams()` / `ParameterError: ${audioVideo} not supported. Either pass level from pre-defined resolutions or pass complete audioVideo object` / Pass a level from the pre-defined resolutions or use a complete `audioVideo` object with appropriate video properties `getUserMedia()` / `Media:index#getMedia --> navigator.mediaDevices.getUserMedia failed - [object OverconstrainedError] (facingMode)` / If you request the rear camera on a laptop device, you'll get this error. Use either `ideal` instead of `exact` or request only cameras appropriate to the device. https://developer.webex.com/meeting/docs/sdks/webex-meetings-sdk-web-video-resolution

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
7. The non-transitory computer readable medium of claim 1 wherein the function block is further operative when executed by the processor to determine that the superblock application is authorized to access functionality provided by the function block for the service.	Each Accused Product literally meets the requirement "wherein the function block is further operative when executed by the processor to determine that the superblock application is authorized to access functionality provided by the function block for the service." For example, the function block determines if the function block and superblock are allowed to access the A/V resources. If either are not able to access the resources, a 401 or 403 error code is returned, as shown below.

Claim	Cisco Accused Products
	Status: 400 *Bad Request: The request was invalid or cannot be otherwise served. An accompanying error message will explain further.* **Status:** 401 *Unauthorized: Authentication credentials were missing or incorrect.* **Status:** 403 *Forbidden: The request is understood, but it has been refused or access is not allowed.* **Status:** 404 *Not Found: The URI requested is invalid or the resource requested, such as a user, does not exist. Also returned when the requested format is not supported by the requested method.* **Status:** 405 *Method Not Allowed: The request was made to a resource using an HTTP request method that is not supported.* https://developer.webex.com/meeting/docs/api/v1/meetings/join-a-meeting

Claim	Cisco Accused Products
	For example, authentication credentials must be acquired to use the Webex SDK. Without proper authorization, Webex SDK will not function, as evidenced by the below. What Are Integrations? Integrations are how you request permission to invoke the Webex REST API on behalf of another Webex user. To do this in a secure way the API supports the OAuth 2 standard which allows third-party integrations to get a temporary access token for authenticating API calls instead of asking users for their password. If you're sure that your integrations require authenticating on behalf of another Webex user, read on, we'll get you there in a few easy steps: 1. Register your integration with Webex 2. Request permission using an OAuth Grant Flow 3. Exchange the resulting authorization code for an access token 4. Use the access token to make your API calls https://developer.webex.com/create/docs/integrations *See* support for claim 1a. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
8. The non-transitory computer readable medium of claim 1 wherein the function block is further operative when executed by the processor to provide	Each Accused Product literally meets the requirement "wherein the function block is further operative when executed by the processor to provide authorization credentials to access the service, where the authorization credentials are received by the function block from the superblock application." When a user requests to join an A/V room from the superblock application of the Webex program, the request is passed to the Webex SDK function block with information associated with the user (i.e., authorization credentials). The Webex SDK function block provides the authorization credentials to an

Claim	Cisco Accused Products
authorization credentials to access the service, where the authorization credentials are received by the function block from the superblock application.	external device, such as a Cisco server (or other server), which then generates an access token for accessing the A/V room. Authorization credentials are also communicated between the Webex SDK function block and the superblock application when API keys are requested for accessing a Webex SDK account *See* support for claim 7. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
9. The non-transitory computer readable medium of claim 1 wherein the function block is further operative when executed by the processor to receive a request for the service from the superblock application as an application programming interface (API) call from the superblock application to the function block.	Each Accused Product literally meets the requirement "wherein the function block is further operative when executed by the processor to receive a request for the service from the superblock application as an application programming interface (API) call from the superblock application to the function block." For example, the Webex SDK includes both a traditional SDK and APIs. *See* support for claim 1a. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
10. The non-transitory computer readable medium of claim 1 wherein the function block is further operative when executed by the processor to determine	Each Accused Product literally meets the requirement "wherein the function block is further operative when executed by the processor to determine whether the service is available on the external device." For example,

Claim	Cisco Accused Products
whether the service is available on the external device.	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
11.p A method implemented in a function block associated with an electronic device having a network interface, comprising:	The preamble is nonlimiting. Nonetheless, to the extent that the preamble limits the claim, each Accused Product literally meets the requirement "[a] method implemented in a function block associated with an electronic device having a network interface." *See* support for elements 1p and 1a. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
11a. accessing a resource via the network interface to provide a service to a superblock application on the electronic device, wherein the service is not otherwise supported by the superblock application, and wherein the function block is accessible only to the superblock application on the electronic device; and	Each Accused Product literally performs the step of "accessing a resource via the network interface to provide a service to a superblock application on the electronic device, wherein the service is not otherwise supported by the superblock application, and wherein the function block is accessible only to the superblock application on the electronic device." For example, without the Webex SDK, A/V capabilities would not be available within the superblock application. As stated in claim 1p, because the code is compiled in the superblock application, it is only accessible via the superblock application. *See* support for claim 1p. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.

Claim	Cisco Accused Products
11b. providing the service to the superblock application without switching context from the superblock application to another application on the electronic device.	Each Accused Product literally performs the step of "providing the service to the superblock application without switching context from the superblock application to another application on the electronic device." *See* support for element 1a. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
12. The method of claim 11, further comprising determining that the superblock application is authorized to access the service.	Each Accused Product literally performs the step of "determining that the superblock application is authorized to access the service." *See* support for claim 7. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
13. The method of claim 11, further comprising determining that the service requires access to the resource, wherein the resource is external to the electronic device.	Each Accused Product literally performs the step of "determining that the service requires access to the resource, wherein the resource is external to the electronic device." *See* support for claims 1a and 7. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
14. The method of claim 11, further comprising operating as a bridge between the superblock application and the resource.	Each Accused Product literally performs the step of "operating as a bridge between the superblock application and the resource." *See* support for claim 11.

Claim	Cisco Accused Products
	Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
15. The method of claim 11, further comprising initializing the function block when the superblock application is loaded and executed.	Each Accused Product literally meets the requirement "initializing the function block when the superblock application is loaded and executed." For example, on information and belief the Cisco's Webex application (the superblock application), initializes the Webex SDK function block. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
17. The method of claim 11, further comprising determining whether the function block is capable of performing any rendering needed by the service.	Each Accused Product literally performs the step of "determining whether the function block is capable of performing any rendering needed by the service." *See* support for claim 4. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.
19. The method of claim 11, further comprising receiving a request for the service from the superblock application, wherein the request is received as an application programming interface (API) call from the superblock application to the function block.	Each Accused Product literally performs the step of "receiving a request for the service from the superblock application, wherein the request is received as an application programming interface (API) call from the superblock application to the function block." *See* support for claim 9.

Claim	Cisco Accused Products
20. The method of claim 11, further comprising determining whether the resource is available.	Each Accused Product literally performs the step of "determining whether the resource is available." *See* support for claim 10. Plaintiff may supplement these contentions pursuant to Section 3(a)(i) of the model Discovery Order once access to relevant source code is provided.