



Using External and Local Policy with Pexip Infinity

Deployment Guide

Software Version 33

Document Version 33.a

October 2023

]pexip[

Contents

Using external and local policy to control Pexip Infinity behavior	4
External policy	4
Local policy	5
Whether to use external policy, local policy or both?	5
Configuration data request types	6
Policy profiles	6
Configuring policy profiles	7
Using the external policy server API with Pexip Infinity	10
Configuring policy profiles for external policy	10
Requests and responses when using the Pexip Infinity external policy API	12
Request types summary	12
Policy server responses — general information	13
Dialing out from conference	14
Service configuration requests	14
Response to a service configuration request	16
Example service configuration data responses	35
Participant properties requests	35
Response to a participant properties request	38
Example participant properties data responses	39
Media location requests	40
Media location response	41
Example media location data response	42
Participant avatar requests	42
Participant avatar response	44
Directory information requests	45
Directory information response	45
Example directory information responses	46
Registration alias requests	46
Registration alias response	47
Example registration alias responses	47
Summary of request parameters per request type	48
Enabling local policy	54
Configuring policy profiles for local policy	54
Writing local policy scripts	56
Types of configuration data	56
Writing a jinja2 script to control call behavior	56
Getting started in constructing a script	57
Supported variables	58

Response formats	62
Service configuration data responses	63
Example service configuration data responses	82
Participant configuration responses	82
Example participant properties data responses	83
Media location data responses	83
Example media location data response	84
Dialing out from conference	84
Using filters in local policy scripts	85
Supported jinja2 filters	85
Custom Pexip filters	85
Testing local policy scripts	88
Example local policy scripts	89
Minimum basic pass-through service configuration script	89
Minimum basic pass-through media location script	89
Basic pass-through service configuration script with call_info debug line	90
Unconditionally nominate media locations	90
Nominate a media location for RTMP streaming	91
Remove PIN based on location	91
Remove PIN if device is registered	91
Inject an ADP into every conference (with a debug line for service_config)	92
Reject specified User Agents	92
Test whether a participant's address is within a particular subnet	93
Lock a conference when the first participant connects	93
Use a different theme based on time of day	94
Limit VMR access to registered devices only	94
Disable encryption when calling a specific device	95
Change layout for Microsoft Teams gateway calls	96
Change layout for Google Meet gateway calls	97
Route incoming calls directly into a breakout room	97

Using external and local policy to control Pexip Infinity behavior

You can extend Pexip Infinity's built-in functionality by using external and/or local policy to apply bespoke call policy and routing decisions based on your own specific requirements.

For example, you might want to apply different PIN rules depending upon whether a conference participant is an on-site employee or a remote visitor, anonymize a participant's display name, or use a specific location for media handling for certain types of calls.

External policy

Pexip Infinity's external policy API allows a vast range of call policy decisions to be taken by an external system, based on the data sources that are available to that external system.

When external policy is enabled, rather than using its own database and systems to retrieve service and participant data, Pexip Infinity Conferencing Nodes send the external policy server a service request over a RESTful API and the server should respond by returning the requested data to the Conferencing Node.

For more information about using the external policy API, see [Using the external policy server API with Pexip Infinity](#).

Local policy

Local policy allows you to manipulate service configuration, some participant properties, and media location data (that has been provided either via the external policy API, or has been retrieved from Pexip Infinity's own database, or was provided by the endpoint in the call request) by running a jinja2 script against that data.

For more information about using local policy, see [Enabling local policy](#).

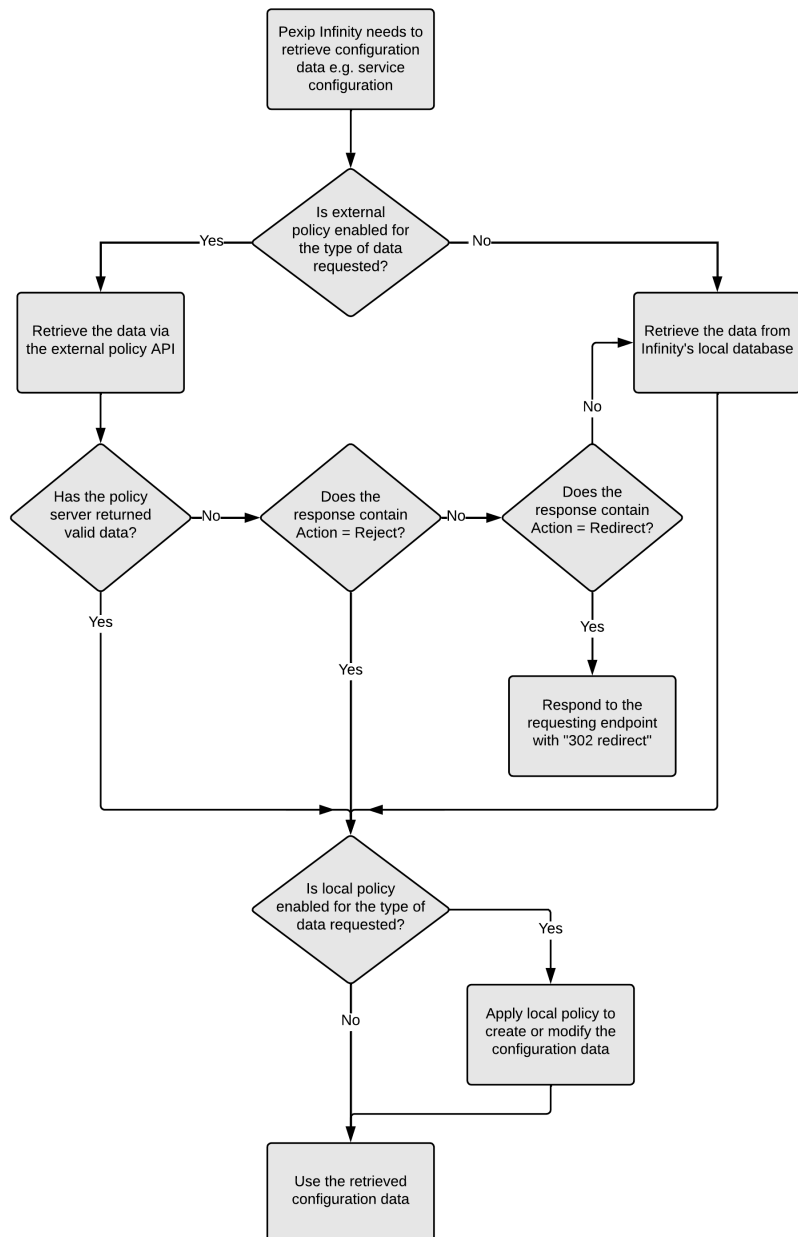
Whether to use external policy, local policy or both?

External policy provides the ultimate flexibility in implementing your own bespoke decision-making logic. However it requires more system development effort to implement and adds a dependency on a third-party system that must provide real-time responses to requests made by Pexip Infinity over the external policy API. Developing a high availability / redundant external policy server that works well in the face of network partitions can be complicated, especially if your Pexip Infinity deployment is geographically distributed.

Local policy has a more limited scope than external policy in what it can control, but it is easier to implement and runs locally on each Conferencing Node.

External policy can be more powerful than local policy: local policy is stateless and therefore, for example, does not have access to information about any running conferences or their participants, whereas external policy can query any databases or APIs it wants, including the Pexip Infinity APIs.

You can configure Pexip Infinity to use both external and local policy depending on your requirements. When both external and local policy are enabled, external policy is applied first to retrieve the configuration data from the external system, and then local policy is applied to that retrieved data (which can then conditionally modify that data). The flow chart shows Pexip Infinity's processing logic when policy profiles are used.



Configuration data request types

The following table shows the types of configuration data that can be controlled, and by which types of policy:

Type of data (policy requests)	Description	Controllable via external policy	Controllable via local policy
Service configuration	The configuration details of a service. Pexip Infinity typically requires this data when it: <ul style="list-style-type: none"> receives an incoming call request needs to place a call to a given alias 	✓	✓
Participant policy	This allows some of the participant's call properties, such as their display name or role, to be changed before they join the conference. This allows you, for example, to anonymize a participant's name or modify their role based on other properties of the call or their associated Identity Provider. <ul style="list-style-type: none"> For WebRTC participants it is applied after any PIN entry or SSO steps (and hence has access to Identity Provider attributes). For SIP/H.323 endpoints, it is applied before any PIN entry steps. Participant policy is applied after any service configuration policy, and before media location policy.	✓	✓
Media location	The system location to use for media allocation. A Conferencing Node assigned to that location will handle the media for the participant. A media location request is often made after a service configuration data request (and after any participant policy).	✓	✓
Participant avatar	Obtains the image to display to represent a conference participant or directory contact.	✓	†
Directory information	Obtains directory information — a list of device or VMR aliases and their associated names. This is used to provide phonebook information to Connect apps that are registered to a Pexip Infinity Conferencing Node.	✓	
Registration alias	Used to determine whether a device alias is allowed to register to a Conferencing Node.	✓	

† Pexip Infinity's local user records can be used to refer to an avatar URL.

Policy profiles

Policy profiles specify how Pexip Infinity uses external policy and/or local policy to control its call policy and routing decisions.

They give you individual control over which types of policy to apply to each type of configuration data. For example, you could use both external and local policy to manipulate service configuration data, only local policy to manipulate participant properties and media location data, external policy to provide participant avatars, and no policy at all (i.e. just use the data from Pexip Infinity's own database) for directory information and registration aliases.

Each system location is configured with a policy profile and that profile is then used by all of the Conferencing Nodes in that location whenever they need to retrieve configuration data. This means that you could use the same policy profile in all locations (and thus all Conferencing Nodes), or if required you can configure many different profiles with, for example, different local policy scripts or different external policy server URLs, and then assign different policy profiles to different system locations.

i You must assign policy profiles to locations otherwise they will never be used. If you want to configure just one policy profile to be used globally you need to assign it to all of your locations.

See [Configuring policy profiles](#) for more information about how to set up your policy profiles.

Configuring policy profiles


Policy profiles specify how Pexip Infinity uses external policy and/or local policy to control its call policy and routing decisions.

Each policy profile can be used to:

- control which types of data (e.g. service configuration, participant data, media location, participant avatars etc.) are managed via policy — either by external policy, or by local policy (where local policy is supported for that data type) or by both external and local policy
- nominate the address of an external policy server to which the external policy API requests are sent
- specify the local policy jinja2 script to be executed against the data (service configuration, participant and media location data types only).

You can configure Pexip Infinity to use both external and local policy depending on your requirements. When both external and local policy are enabled, external policy is applied first to retrieve the configuration data from the external system, and then local policy is applied to that retrieved data (which can then conditionally modify that data). See [Using external and local policy to control Pexip Infinity behavior](#) for more information.


Each system location is configured with a policy profile and that profile is then used by all of the Conferencing Nodes in that location whenever they need to retrieve configuration data. This means that you could use the same policy profile in all locations (and thus all Conferencing Nodes), or if required you can configure many different profiles with, for example, different local policy scripts or different external policy server URIs, and then assign different policy profiles to different system locations.

 You must assign policy profiles to locations otherwise they will never be used. If you want to configure just one policy profile to be used globally you need to assign it to all of your locations.

When using external policy within a system location, you must ensure that each Conferencing Node in that location is able to reach the nominated policy server.

To configure policy profiles:

1. Go to **Call Control > Policy Profiles**.
2. Select **Add Policy profile** and then configure that profile. The options are:

Option	Description
Name	The name used to refer to this policy profile in the Pexip Infinity Administrator interface.
Description	An optional description of the policy profile.
External policy server	
URL	<p>The URL of the policy server to use for all external policy API requests from this profile, for example https://policy.example.com/path.</p> <p>You can only configure one address URL per policy server.</p> <p>If the request is over HTTPS, Pexip Infinity must trust the certificate presented by the policy server.</p> <p> We strongly recommend that you use HTTPS (not HTTP) in production environments.</p>
Username	Optional fields where you can specify the credentials required to access the external policy server.
Password	External policy requests support Basic Authentication and basic ASCII-encoded usernames and passwords.
Avatar policy	
Use local avatar configuration	When Use local avatar configuration is enabled, requests to fetch avatar images to represent directory contacts and conference participants are sent to the Avatar URL associated with the user configured within Pexip Infinity.

Option	Description
Enable external avatar lookup	<p>If enabled, requests are sent to the external policy server to fetch avatar images to represent directory contacts and conference participants.</p> <p>If both Use local avatar configuration and Enable external avatar lookup are enabled, then the local avatar configuration takes precedence. However, if no matching user record is found, or the user record does not have a configured Avatar URL then a request is made to the external policy server instead. If there is an Avatar URL, and the request fails for any reason, Pexip Infinity will not fall back to external policy.</p>
Service configuration policy	
Enable external service configuration lookup	If enabled, requests are sent to the external policy server to fetch service configuration data (VMRs, Virtual Receptions, Infinity Gateway calls etc).
Apply local policy	If enabled, the service configuration retrieved from the local database or an external policy server is processed by the local policy script (which may change the service configuration or cause the call to be rejected).
Script	<p>Only applies if Apply local policy is selected.</p> <p>Enter a jinja2 script that takes the existing service configuration (if any) and optionally modifies or overrides the service settings.</p>
Participant policy	
Enable external participant lookup	If enabled, requests are sent to the external policy server to allow some of the participant's properties to be overridden, or the call to be rejected.
Apply local policy	If enabled, the original participant's call properties or any override properties returned from external policy are processed by the local policy script (which may itself override the participant properties or cause the call to be rejected).
Script	<p>Only applies if Apply local policy is selected.</p> <p>A Jinja2 script that takes the existing participant configuration and optionally overrides the participant settings</p>
Media location policy	
Enable external media location lookup	If enabled, requests are sent to the external policy server to fetch the system location to use for media allocation.
Apply local policy	If enabled, the media location configuration retrieved from the local database or an external policy server is processed by the local policy script (which may change the media location configuration).
Script	<p>Only applies if Apply local policy is selected.</p> <p>Enter a jinja2 script that takes the existing media location configuration and optionally modifies or overrides the location settings.</p>
Directory	
Enable external directory lookup	If enabled, requests are sent to the external policy server to fetch directory information (that can be used by some Connect app clients to display a phonebook).
Registration requests	
Enable external registration policy	If enabled, requests are sent to the external policy server to determine whether a device alias is allowed to register to a Conferencing Node.

3. Select **Save**.
4. Go to **Platform > Locations**.
5. Select each location in turn and specify the **Policy profile** that the Conferencing Nodes in that location should use when making policy decisions.

Using the external policy server API with Pexip Infinity

Pexip Infinity's external policy API allows a vast range of call policy decisions to be taken by an external system, based on the data sources that are available to that external system.

When external policy is enabled, rather than using its own database and systems to retrieve service and participant data, Pexip Infinity Conferencing Nodes send the external policy server a service request over a RESTful API and the server should respond by returning the requested data to the Conferencing Node.

You must configure Pexip Infinity with the details of one or more external policy servers to which it can send policy API requests. You do this by configuring [policy profiles](#) with the addresses of one or more external policy server URLs and then associating each system location with one of those policy profiles. This means that different locations can use different policy servers for scalability and routing efficiency, if required. When a Conferencing Node needs to obtain data that is supported by external policy (such as service configuration information) it will request information from the policy server associated with the location hosting that node.

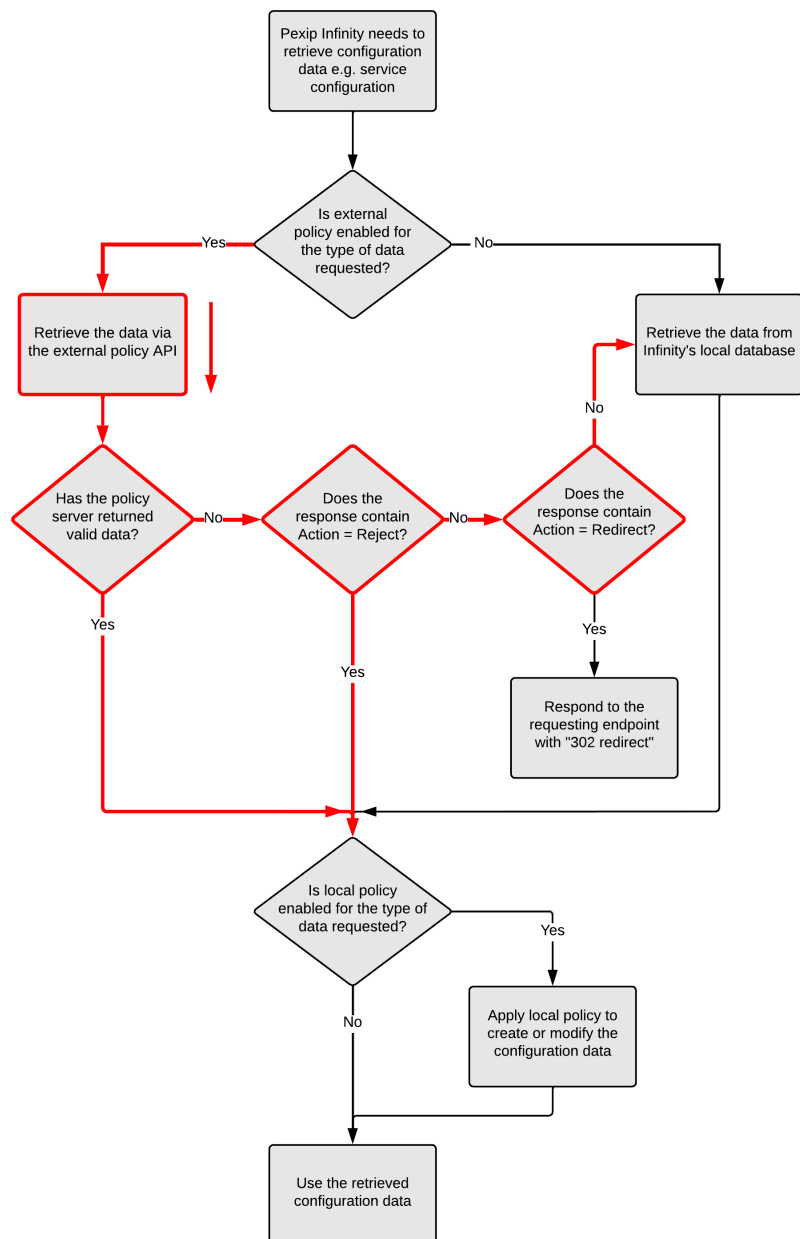
Within a policy profile you can enable or disable support for each individual request type (avatars, service configuration, participant properties, media location, directory and registration requests). The requests can be subject to basic username and password authentication. For redundancy, you may also use a http load balancer in front of a pool of policy servers.

See [Requests and responses when using the Pexip Infinity external policy API](#) for full information about how to use the external policy API.

We recommend that each policy server uses the same underlying decision-making logic or database, so that consistent responses are returned. However, a policy server can apply different rules on a per-location basis, if required, as the requesting location is one of the parameters passed to the policy server in policy requests. Note that if the policy server does not return the expected service information e.g. the policy server cannot be reached, times out or returns invalid or incomplete data, Pexip Infinity will then attempt to obtain the relevant information from its own internal database (i.e. perform its standard behavior as when external policy is not in use).


Configuring policy profiles for external policy

You can configure Pexip Infinity to use both external and local policy depending on your requirements. When both external and local policy are enabled, external policy is applied first to retrieve the configuration data from the external system, and then local policy is applied to that retrieved data (which can then



conditionally modify that data). The flow chart (right) shows Pexip Infinity's standard processing logic when policy profiles are used and highlights where external policy is applied.

To configure Pexip Infinity to use the external policy API:

1. Go to **Call Control > Policy Profiles**.
2. Select **Add Policy profile** and then configure that profile:
 - In the **External policy server** section, configure the URL of the policy server and any credentials required to access it.
 -  We strongly recommend that you use HTTPS (not HTTP) in production environments.
 - Enable the external lookup options for each type of data request (avatars, service configuration, participant properties, media location, directory and registration requests) that you want to submit to your external policy server, and save your changes.
3. Go to **Platform > Locations**.
4. Select each location in turn and specify the **Policy profile** that the Conferencing Nodes in that location should use when making policy decisions.

For more information on configuring policy profiles and how to combine external policy with local policy, see [Configuring policy profiles](#).

Requests and responses when using the Pexip Infinity external policy API

Pexip Infinity accesses the external policy API via a GET request over HTTP or HTTPS to the appropriate external policy server URL (as configured in the policy profile).

i We strongly recommend that you use HTTPS (not HTTP) in production environments.

A number of request parameters are added to the request URL according to the request type. This provides flexibility to the decision-making process within the policy server, and means, for example, that the policy server could return:

- different media location information based on the `remote_alias` or the call `protocol`
- a different avatar for the same participant based on the originally dialed `local_alias` or the `service_name`
- a different set of directory contacts depending on the `registered_alias` making the request.

This topic contains a [summary](#) of the supported requests and [response](#) guidelines. Each request type is then explained in more detail (see [service configuration](#), [participant properties](#), [media location](#), [participant avatar](#), [directory information](#) and [registration alias](#)) including the parameters sent to the policy server for that request, the expected response and some examples. Finally, there is a [summary matrix](#) showing which parameters are contained in each policy request type.

Request types summary

The following table shows the different API request types that a Conferencing Node can send to an external policy server.

Policy request type	Description and request URI
Service configuration	Obtains the configuration details of a service. Pexip Infinity typically makes this request when it: <ul style="list-style-type: none"> • receives an incoming call request • needs to place a call to a given alias Request URI: <code><policy_server_uri>/policy/v1/service/configuration</code>
Participant properties	This allows some of the participant's call properties, such as their display name or role, to be changed before they join the conference. This allows you, for example, to anonymize a participant's name or modify their role based on other properties of the call or their associated Identity Provider. <ul style="list-style-type: none"> • For WebRTC participants it is applied after any PIN entry or SSO steps (and hence has access to Identity Provider attributes). • For SIP/H.323 endpoints, it is applied before any PIN entry steps. Participant policy is applied after any service configuration policy, and before media location policy. Request URI: <code><policy_server_uri>/policy/v1/participant/properties</code>
Media location	Obtains the system location to use for media allocation. A Conferencing Node assigned to that location will handle the media for the participant. A media location request is often made after a service configuration data request (and after any participant policy). Request URI: <code><policy_server_uri>/policy/v1/participant/location</code>
Participant avatar	Obtains the image to display to represent a conference participant or directory contact. Request URI: <code><policy_server_uri>/policy/v1/participant/avatar/<alias></code> where <code><alias>*</code> is the alias of the participant/contact whose avatar is required.
Directory information	Obtains directory information — a list of device or VMR aliases and their associated names. This is used to provide phonebook information to Connect apps that are registered to a Pexip Infinity Conferencing Node. Request URI: <code><policy_server_uri>/policy/v1/registrations</code>

Policy request type	Description and request URI
<u>Registration alias</u>	Used to determine whether a device alias is allowed to register to a Conferencing Node. Request URI: <policy_server_uri>/policy/v1/registrations/<alias> where <alias>* is the alias of the device that is making the registration request.

* The alias may include scheme information, for example sip:alice@example.com. The policy server must be able to parse the received alias in an appropriate manner.

Note that:

- You can configure whether specific request types are sent or not, on a per policy profile basis, via **Call Control > Policy Profiles**. If a request type is disabled, Pexip Infinity will fall back to its own default behavior for that request type.
- Pexip Infinity has a non-configurable timeout of 5 seconds for each attempt it makes to contact a policy server.
- Most policy requests are sent from the Conferencing Node that is handling the call signaling or has received the registration request. However, requests are sent from the Conferencing Node that is handling the call media in the following situations:
 - It is a participant avatar request.
 - The participant is in a Virtual Reception (note that when the endpoint initially calls the Virtual Reception alias, the requests will come from the Conferencing Node handling the signaling — as usual, but after being placed into the Virtual Reception all subsequent requests i.e. when the participant enters the number of the VMR they want to join, are sent from the Conferencing Node that is handling the Virtual Reception call media).
- You may see multiple service configuration, participant and media location requests when handling Connect app participants.

Policy server responses — general information

The policy server response to a request should be a 200 OK message. The response header must include the Content-Type, and the message body must include the requested content.

The response for all requests, except for participant avatar requests, must return a **Content-Type** of **application/json**. The policy server can return an error code e.g. 404 if it wants Pexip Infinity to fall back to its own default behavior for a specific request. Pexip Infinity will not follow any 301/302 redirects received from the policy server.

The response to a request takes the basic format:

```
{
  "status": "success",
  "action": "reject|redirect|continue",
  "result": { <data> },
  "<other_keys>": "<other_values>"
}
```

where:

- if "status" is anything other than "success", the response is deemed to have failed and Pexip Infinity will fall back to its default behavior (which is to attempt to retrieve the relevant data from its own internal database), unless the "action" field is supported in the response for that request type, in which case the "action" field instructs Pexip Infinity how to proceed.
- "action" is an optional value that may be included in responses to service configuration and registration alias requests. When present, it instructs Pexip Infinity how to proceed in failure scenarios:
 - "reject" instructs Pexip Infinity to reject the request — for a service configuration or participant properties request this would mean to reject the call (i.e. return "conference not found"), and for a registration request it would mean to reject the registration. However, note that if you have local policy enabled for service configuration or participant properties, the local policy could change the action and provide updated service configuration / participant properties instead.
 - "redirect" instructs Pexip Infinity to send a SIP response with status "302 redirect" to the requesting endpoint, telling it to place an otherwise identical call to a different alias. The "result" field specifies the alias to be used, using the format {"new_alias": "sip:alias@example.com"}
 - "continue" (or any other value except "reject" or "redirect") instructs Pexip Infinity to fall back to its default behavior (which is to attempt to retrieve the service configuration or device alias data from its own internal database).
- "result" is a JSON object of key value pairs as appropriate for the request type. The specific requirements of what must be included in the result is included below in the descriptions of each request type. The fields in the JSON object can be supplied in any order.

- "<other_keys>" and "<other_values>" can be zero, one or more optional key value pairs. If included, they do not affect how Pexip Infinity processes the response but they will be included in any associated support log messages. They could, for example, indicate the version number of the software running on the policy server, or contain "reason" information for failure responses.

Note that responses to participant avatar requests have different requirements (see [Participant avatar response](#) for details).

Dialing out from conference

Pexip Infinity makes a service configuration request if it dials out from a conference to invite a participant to join (where the `call_direction` parameter will be `dial_out`). In these cases, the response to the service configuration request must match the existing service data (i.e. the same `name`, `service_type` and so on).

When dialing out, the only configuration you can control via policy is:

- The `prefer_ipv6` setting in the service configuration response.
- The media location — you can do this in the response to the media location request that follows the service configuration request.

Service configuration requests

Obtains the configuration details of a service. Pexip Infinity typically makes this request when it:

- receives an incoming call request
- needs to place a call to a given alias

Request URI: `<policy_server_uri>/policy/v1/service/configuration`

Pexip Infinity includes the following fields in a service configuration request:

Parameter	Description
<code>bandwidth</code>	The maximum requested bandwidth for the call. It is present on both inbound and outbound call requests but is meaningful only for inbound calls.
<code>breakout_uuid</code>	The uuid of a breakout room (if applicable).
<code>call_direction</code>	The direction of the call that triggered the request. Values can be: <ul style="list-style-type: none">• "dial_in": calls in to Pexip Infinity• "dial_out": calls dialed out from Pexip Infinity• "non_dial": when policy is triggered by other requests that are not related to incoming or outgoing call setup, such as when an H.323 LRQ or a SIP SUBSCRIBE or OPTIONS request is received.
<code>call_tag</code>	An optional call tag that is assigned to a participant.
<code>has_authenticated_display_name</code>	Boolean indicating if the participant's display name was provided and authenticated by an Identity Provider.
<code>idp_uuid</code>	The UUID of the Identity Provider who provided and authenticated the participant's display name, and provided any <code>idp_attributes</code> in participant policy.
<code>local_alias</code>	In the context of service and participant configuration requests, this is the incoming alias (typically the alias that the endpoint has dialed). This is the primary item of information that the policy server will use to return appropriate service configuration data.
<code>location</code>	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.
<code>ms-subnet</code> †	The sender's subnet address.

Parameter	Description
node_ip	The IP address of the Conferencing Node making the request.
p_Asserted-Identity [†]	The authenticated identity of the user sending the SIP message.
protocol	The call protocol. Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip". (Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)
pseudo_version_id	The Pexip Infinity software build number.
registered	Boolean indicating whether the remote participant is registered or not.
remote_address	The IP address of the remote participant.
remote_alias	The name of the user or the registered alias of the endpoint. The remote_alias may include scheme information, for example sip:alice@example.com.
remote_display_name	The display name of the remote participant.
remote_port	The IP port of the remote participant.
service_name ‡	The service name. This will match the name field returned by the policy server from the original service configuration request.
service_tag ††	The service tag associated with the service_name parameter.
supports_direct_media	Boolean indicating if the service supports direct media or not.
teams_tenant_id	Contains the Microsoft Teams tenant ID on an inbound Teams call to Pexip Infinity for Teams Rooms SIP/H.323 calling.
telehealth_request_id ◇	The telehealth call id.
trigger	The trigger for the policy request. Values: "web", "web_avatar_fetch", "invite", "options", "subscribe", "setup", "arq", "lrq" or "unspecified".
unique_service_name ‡	The unique name used by Pexip Infinity to identify the service: <ul style="list-style-type: none"> For "gateway" services this is the matching rule name followed by a unique identifier (so as to distinguish between separate calls that match the same rule). For all other services, this is the same as the service_name parameter.
vendor	System details about the remote participant, such as manufacturer name and version number for hard endpoints, or browser and operating system details for soft clients.
version_id	The Pexip Infinity software version number.

[†] Only included if the request was triggered by a SIP message, and the parameter is included as a field in the SIP message header. Note that the ms_subnet and p_asserted_identity fields use only underscores and lower case characters in their names when used in local policy (normally they are referenced as ms-subnet and p_Asserted-Identity).

‡ Only included in outbound call requests and for breakout rooms.

†† Only included in outbound call requests.

◇ Only included in Epic telehealth calls.

This example shows a GET request for service configuration data when alice@example.com has dialed meet.bob from a SIP endpoint:

```
GET /example.com/policy/v1/service/configuration?protocol=sip&node_ip=10.44.99.2&registered=False&remote_address=10.44.75.249&version_id=16&bandwidth=0&pseudo_version_id=36402.0.0&vendor=TANDBERG/518 (TC6.0.1.65adebe)&local_
```

```
alias=meet.bob&remote_port=58435&idp_uuid=&has_authenticated_display_name=False&supports_direct_media=False&call_direction=dial_in&call_tag=&remote_alias=sip:alice@example.com&remote_display_name=Alice&trigger=invite&location=London
```

Response to a service configuration request

The response to a service configuration request takes the basic format:

```
{
  "status": "success",
  "action": "reject|redirect|continue",
  "result": { <service_configuration_data> }
}
```

where:

- "action" is an optional value that, when included, instructs Pexip Infinity how to proceed in failure scenarios:
 - "reject" instructs Pexip Infinity to reject the call (i.e. return "conference not found"). However, note that if you have local policy enabled for service configuration, the local policy could still provide some service configuration instead.
 - "redirect" instructs Pexip Infinity to send a SIP response with status "302 redirect" to the requesting endpoint, telling it to place an otherwise identical call to a different alias. The "result" field specifies the alias to be used, using the format `{"new_alias": "sip:alias@example.com"}`
 - "continue" instructs Pexip Infinity to fall back to its default behavior (which is to attempt to retrieve the service configuration data from its own internal database).
- "result" is a JSON object of multiple key value pairs that describes the service:
 - Some data fields are required, and some are optional: the fields expected by Pexip Infinity in the response depend upon the returned service_type — either "conference" or "lecture" for a Virtual Meeting Room or Virtual Auditorium, "gateway" for an Infinity Gateway call, "two_stage_dialing" for a Virtual Reception, "media_playback" for a Media Playback Service, or "test_call" for a Test Call Service.
 - Note that the response configures the service and therefore each request for the same service should return the same configuration for that service. The only fields that can return a different value (per participant) are the PIN and bandwidth related fields (pin, guest_pin, max_callrate_in, max_callrate_out). You cannot change the properties of a participant calling into that service, such as their display name.
- the "action" field is ignored if "status" is "success" and "result" contains valid data (and it is a 200 OK message).

The fields expected in the response for each service type are described below:

Virtual Meeting Room / Virtual Auditorium service types response fields

Pexip Infinity expects the following fields to be returned for a service_type of "conference" (VMR) or "lecture" (Virtual Auditorium):

Field name	Required	Type	Description
name	Yes	String	The name of the service. You could, for example, use the local_alias received in the configuration request. Pexip Infinity will then use this name — as the service_name parameter — in subsequent requests to identify that service. If this configuration is defining a breakout room, then this is the name of the main VMR.
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	The type of service, in this case: <ul style="list-style-type: none">• "conference": a Virtual Meeting Room, or• "lecture": a Virtual Auditorium

Field name	Required	Type	Description
allow_guests	No	Boolean	Whether to distinguish between Host and Guest participants: <ul style="list-style-type: none"> true: the conference has two types of participants: Hosts and Guests. The pin to be used by Hosts must be specified. A guest_pin can optionally be specified; if a guest_pin is not specified, Guests can join without a PIN. false: all participants have Host privileges Default: false
automatic_participants	No	List	A list of participants to dial automatically when the conference starts. Each participant in the list contains a number of fields as described in Automatically dialed participants (ADP) fields .
breakout_rooms	No	Boolean	Whether Breakout Rooms are enabled for the VMR. See Breakout room fields below for configuration for a breakout room. Default: false
bypass_proxy	No	Boolean	Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node. <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. Default: false
call_type	No	String	The call capability of the conference. It can be limited to: <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only Default: "video"
crypto_mode	No	String	Controls the media encryption requirements for participants connecting to this service. <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) Default: <null> (use global setting)
description	No	String	A description of the service.

Field name	Required	Type	Description
direct_media	No	String	<p>Allows this VMR to use direct media between participants. When enabled, the VMR provides non-transcoded, encrypted, point-to-point calls between any two WebRTC participants. The options are:</p> <ul style="list-style-type: none"> "never": do not use direct media in this VMR. "best_effort": use direct media in this VMR where possible (when there are two WebRTC participants only), otherwise use standard, transcoded media connections via a Conferencing Node. <p>Default: "never"</p>
direct_media_notification_duration	No	Integer	<p>The number of seconds to show a notification to participants before being escalated into a transcoded call, or de-escalated into a direct media call. Range: 0 to 30 seconds.</p> <p>Default: 0 seconds</p>
enable_chat	No	String	<p>Whether chat messaging is enabled for the conference:</p> <ul style="list-style-type: none"> "default": as per the global configuration setting "yes": chat is enabled "no": chat is disabled <p>Default: "default"</p>
enable_active_speaker_indication	No	Boolean	<p>When active speaker display is enabled, the display name or alias of the current speaker is shown across the bottom of their video image. This option is not available in every layout.</p> <ul style="list-style-type: none"> true: active speaker is indicated false: active speaker is not indicated <p>Default: false</p>
enable_overlay_text	No	Boolean	<p>If participant name overlays are enabled, the display names or aliases of all participants are shown in a text overlay along the bottom of their video image.</p> <ul style="list-style-type: none"> true: participant names are shown false: participant names are not shown <p>Default: false</p>
force_presenter_into_main (applies to service_type of "lecture" only)	No	Boolean	<p>Controls whether the Host who is presenting is locked into the main video position:</p> <ul style="list-style-type: none"> true: the Host sending the presentation stream will always hold the main video position false: the main video position is voice-switched <p>Default: false</p>
guest_pin	No	String	Guest PIN — the secure access code for Guest participants.

Field name	Required	Type	Description
guest_view (applies to service_type of "lecture" only)	No	String	The layout seen by Guests: <ul style="list-style-type: none"> "one_main_zero_pips": full-screen main speaker only "one_main_seven_pips": large main speaker and up to 7 other participants "one_main_twentyone_pips": main speaker and up to 21 other participants "two_mains_twentyone_pips": 2 main speakers and up to 21 other participants "one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants "four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers "nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers "sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers "twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers "five_mains_seven_pips": Adaptive Composition layout (you must also set Host view to Adaptive Composition) "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants "teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) Default: "one_main_seven_pips" Only Hosts are displayed. Guests can hear but not see any of the other Guests.
guests_can_present	No	Boolean	Controls whether the Guests in the conference are allowed to present content. <ul style="list-style-type: none"> true: Guests and Hosts can present into the conference false: only Hosts can present Default: true
guest_identity_provider_group	No	String	The set of Identity Providers to be offered to Guests to authenticate with, in order to use the service. If this is blank, Guests are not required to authenticate.
host_identity_provider_group	No	String	The set of Identity Providers to be offered to Hosts to authenticate with, in order to use the service. If this is blank, Hosts are not required to authenticate.

Field name	Required	Type	Description
host_view (applies to service_type of "lecture" only)	No	String	<p>The layout seen by Hosts:</p> <ul style="list-style-type: none"> "one_main_zero_pips": full-screen main speaker only "one_main_seven_pips": large main speaker and up to 7 other participants "one_main_twentyone_pips": main speaker and up to 21 other participants "two_mains_twentyone_pips": 2 main speakers and up to 21 other participants "one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants "four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers "nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers "sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers "twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers "five_mains_seven_pips": Adaptive Composition layout "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants "teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) <p>Default: "one_main_seven_pips"</p>
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.
locked	No	Boolean	<p>Whether to lock the conference on creation:</p> <ul style="list-style-type: none"> true: the conference will be locked on creation false: the conference will not be locked <p>Note that this field has no effect on the conference if it is already running.</p> <p>Default: false</p>
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	<p>Controls the maximum call quality for participants connecting to this service:</p> <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. <p>Default: <null> (use global setting)</p>
mute_all_guests (applies to service_type of "lecture" only)	No	Boolean	<p>Controls whether to mute guests when they first join the conference:</p> <ul style="list-style-type: none"> true: mute Guests when they first join the conference false: do not mute Guests when they first join the conference <p>Default: false</p>

Field name	Required	Type	Description
non_idp_participants	No	String	<p>Determines whether participants joining a SSO-protected service from devices other than the Connect web app (for example SIP or H.323 endpoints) are allowed to dial in to the service.</p> <ul style="list-style-type: none"> "disallow_all": these devices are placed in a waiting room where they must wait to be admitted by a Host. "allow_if_trusted": these devices may join the service if they are locally registered. They must still enter a Host PIN or Guest PIN if either is required. All other devices are placed in a waiting room where they must wait to be admitted by a Host. <p>Default: "disallow_all"</p>
participant_limit	No	Integer	The maximum number of participants allowed to join the service.
pin	No	String	Host PIN — the secure access code for Host participants.
prefer_ipv6	No	String	<p>Whether to use IPv6 for SIP media when dialing out to any ADPs:</p> <ul style="list-style-type: none"> "default": use default Pexip Infinity behavior, which is to prefer IPv4 addresses for SIP media "yes": use IPv6 addresses for SIP media "no": do not use IPv6 addresses for SIP media <p>Default: "default"</p>
primary_owner_email_address	No	String	The email address of the owner of the VMR.
view (applies to service_type of "conference" only)	No	String	<p>The layout seen by all participants:</p> <ul style="list-style-type: none"> "one_main_zero_pips": full-screen main speaker only "one_main_seven_pips": large main speaker and up to 7 other participants "one_main_twentyone_pips": main speaker and up to 21 other participants "two_mains_twentyone_pips": 2 main speakers and up to 21 other participants "one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants "four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers "nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers "sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers "twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers "five_mains_seven_pips": Adaptive Composition layout "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants "teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) <p>Default: "one_main_seven_pips"</p>

Breakout room fields

If you want the configuration to represent a breakout room, then the following fields should also be supplied:

Field name	Required	Type	Description
breakout	Yes	Boolean	Indicates this is a breakout room. If <code>breakout</code> is true, then <code>breakout_rooms</code> should be false (you cannot create breakout rooms from within a breakout room). Also, note that name should be the main VMR room name (of which this is a breakout).
breakout_uuid	Yes	String	A uuid string that represents this breakout room, in the format "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX".
breakout_name	Yes	String	Name of the breakout room.
breakout_uuids	No	List	<p>When returning the main room configuration back to a Host you can include a list of uuid strings that represents multiple breakout rooms to be created.</p> <p>This causes the Host to attempt to join each of these <code>breakout_uuid</code> rooms as a control only participant. This will trigger another policy lookup that includes the following fields (either as external policy parameters, or as part of the <code>call_info</code> structure for internal policy):</p> <pre>"service_name": "<main_room_name>_breakout_<breakout_uuid>" "unique_service_name": same as service_name "breakout_uuid": breakout_uuid of breakout room being created</pre> <p>The response to these policy lookups should be a configuration block which represents a breakout room with the fields configured appropriately for each room.</p>
breakout_description	No	String	Long name / description of the breakout room.
end_action	No	String	<p>What to do with participants when the timer expires, or the breakout is closed:</p> <ul style="list-style-type: none"> "disconnect": disconnect them all "transfer": transfer them back to the main room <p>Default: "transfer"</p>
end_time	No	Integer	<p>Time when breakout ends, in seconds since epoch. 0 = no automatic end.</p> <p>Default: 0</p>

See [Route incoming calls directly into a breakout room](#) for an example local policy script that shows how you could use these fields.

Infinity Gateway service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "gateway":

Field name	Required	Type	Description
local_alias	Yes	String	The calling or "from" alias. This is the alias that the recipient would use to return the call.
name	Yes	String	<p>The name of the service. Ensure that all gateway service instances have a unique name to avoid conflicts between calls.</p> <p>Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that call.</p>

Field name	Required	Type	Description
outgoing_protocol	Yes	String	<p>The protocol to use to place the outgoing call:</p> <ul style="list-style-type: none"> "h323": an H.323 call; you can also optionally nominate an <code>h323_gatekeeper_name</code> "sip": a SIP call; you can also optionally nominate a <code>sip_proxy_name</code> "mssip": a Microsoft Skype for Business / Lync call; you can also optionally nominate an <code>mssip_proxy_name</code> and a <code>turn_server_name</code> "rtmp": uses the RTMP protocol; typically this is used for content streaming "gms": for calls to the Google Meet service. "teams": for calls to the Microsoft Teams service.
remote_alias	Yes	String	<p>The alias of the endpoint to call.</p> <p>The alias can include scheme information, for example <code>sip:alice@example.com</code>, but the call will always be made using the specified <code>outgoing_protocol</code>.</p>
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	<p>The type of service, in this case:</p> <ul style="list-style-type: none"> "gateway": an Infinity Gateway call
bypass_proxy	No	Boolean	<p>Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node.</p> <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. <p>Default: false</p>
call_type	No	String	<p>The call capability of the outbound call:</p> <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only "auto": match the capability of the outbound call to that of the inbound call <p>Default: "video"</p>
crypto_mode	No	String	<p>Controls the media encryption requirements for participants connecting to this service.</p> <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) <p>Default: <null> (use global setting)</p>

Field name	Required	Type	Description
called_device_type	No	String	<p>The device or system to which to route the call:</p> <ul style="list-style-type: none"> "external": route the call to a matching registered device if it is currently registered, otherwise attempt to route the call via an external system "registration": route the call to a matching registered device only (providing it is currently registered) "mssip_conference_id": route the call via a Skype for Business / Lync server to a Sfb/Lync meeting where the remote_alias is a Sfb/Lync meeting Conference ID "mssip_server": route the call via a Skype for Business / Lync server to a Sfb/Lync client or meeting "gms_conference": for calls to the Google Meet service. "teams_conference": for calls to the Microsoft Teams service. "teams_user": for direct calls to a Microsoft Teams Room. "telehealth_profile": for calls to an Epic telehealth system. <p>Default: "external"</p>
denoise_audio	No	Boolean	<p>Applies to Google Meet integrations only. Controls whether to remove background noise from audio streams as they pass through the infrastructure.</p> <p>Default: true</p>
description	No	String	A description of the service.
enable_active_speaker_indication	No	Boolean	<p>When active speaker display is enabled, the display name or alias of the current speaker is shown across the bottom of their video image. This option is not available in every layout.</p> <ul style="list-style-type: none"> true: active speaker is indicated false: active speaker is not indicated <p>Default: false</p>
enable_overlay_text	No	Boolean	<p>If participant name overlays are enabled, the display names or aliases of all participants are shown in a text overlay along the bottom of their video image.</p> <ul style="list-style-type: none"> true: participant names are shown false: participant names are not shown <p>Default: false</p>
external_participant_avatar_lookup	No	String	<p>Applies to Microsoft Teams integrations only. This determines whether or not the Teams Connector requests from Exchange Online an avatar for each participant in the Teams conference.</p> <ul style="list-style-type: none"> "default": use the global external participant avatar lookup setting. "yes": request the participant avatar from Exchange Online via the Teams Connector. "no": use default avatar behavior. <p>Default: "default" (use global setting)</p>
gms_access_token_name	No	String	<p>The name of the access token to use to resolve Google Meet IDs. You should select either a trusted or untrusted type of token, depending on whether you want to enable the device to be automatically admitted into the Google Meet conference (subject to also being a trusted endpoint from Pexip Infinity's perspective i.e. if the rule also has <code>treat_as_trusted</code> enabled).</p>

Field name	Required	Type	Description
h323_gatekeeper_name	No	String	The name* of the H.323 gatekeeper to use to place the outgoing call. DNS is used if no gatekeeper is specified.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)
mssip_proxy_name	No	String	The name* of the Skype for Business / Lync server to use to place the outgoing call. DNS is used if no server is specified.
outgoing_location_name	No	String	The name* of the location of a Conferencing Node from which to place the call.
prefer_ipv6	No	String	Whether to use IPv6 for SIP media for the outgoing call: <ul style="list-style-type: none"> "default": use default Pexip Infinity behavior, which is to prefer IPv4 addresses for SIP media "yes": use IPv6 addresses for SIP media "no": do not use IPv6 addresses for SIP media Default: "default"
sip_proxy_name	No	String	The name* of the SIP proxy to use to place the outgoing call. DNS is used if no proxy is specified.
stun_server_name	No	String	The name* of the STUN server to use when placing calls to external SIP and WebRTC endpoints that are ICE-enabled (such as Skype for Business / Lync clients and Connect app WebRTC clients).
teams_proxy_name	No	String	The name of the Teams Connector to handle the call to Microsoft Teams. If you do not specify anything, the Teams Connector associated with the outgoing location is used.
treat_as_trusted	No	Boolean	This indicates that the target of this Call Routing Rule may treat the caller as part of the target organization for trust purposes.
turn_server_name	No	String	The name* of the TURN server to offer to external SIP and WebRTC endpoints that are ICE-enabled (such as Skype for Business / Lync clients and Connect app WebRTC clients).

Field name	Required	Type	Description
view	No	String	<p>The layout seen by Pexip participants:</p> <ul style="list-style-type: none">"one_main_zero_pips": full-screen main speaker only"one_main_seven_pips": large main speaker and up to 7 other participants"one_main_twentyone_pips": main speaker and up to 21 other participants"two_mains_twentyone_pips": 2 main speakers and up to 21 other participants"one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants"four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers"nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers"sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers"twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers"five_mains_seven_pips": Adaptive Composition layout"one_main_twelve_around": large main speaker and up to 12 other participants"two_mains_eight_around": two main speakers and up to 8 other participants"teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) <p>Default: "one_main_seven_pips"</p>

* The returned "name" must match the name of the location, H.323 gatekeeper, SIP proxy etc. configured within Pexip Infinity.

Virtual Reception service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "two_stage_dialing" (Virtual Reception):

Field name	Required	Type	Description
name	Yes	String	The name of the service. You could, for example, use the <code>local_alias</code> received in the configuration request. Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that service.
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	The type of service, in this case: <ul style="list-style-type: none"> "two_stage_dialing": a Virtual Reception
bypass_proxy	No	Boolean	Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node. <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. Default: false
call_type	No	String	The call capability of the reception. It can be limited to: <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only Default: "video"
crypto_mode	No	String	Controls the media encryption requirements for participants connecting to this service. <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) Default: <null> (use global setting)
description	No	String	A description of the service.
gms_access_token_name	No	String	The name of the access token to use to resolve Google Meet IDs. When configuring a Virtual Reception it does not matter if you use a trusted or untrusted access token.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.

Field name	Required	Type	Description
match_string	No	String	<p>An optional regular expression used to match against the alias entered by the caller into the Virtual Reception. If the entered alias does not match the expression, the Virtual Reception will not route the call.</p> <p>If this field is left blank, any entered alias is permitted.</p>
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	<p>Controls the maximum call quality for participants connecting to this service:</p> <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. <p>Default: <null> (use global setting)</p>
mssip_proxy_name	No	String	The name of the Skype for Business / Lync server to use to resolve the SfB/Lync Conference ID entered by the user in the Virtual Reception.
post_match_string	No	String	<p>An optional regular expression used to match against the meeting code entered by the caller into the Virtual Reception. This is typically used in conjunction with the post_replace_string to transform the meeting code into a distinct alias pattern that will match a Call Routing Rule configured to route calls into external conferences.</p> <p>For example, you would typically set the regex match to <code>(.*)</code> (to match everything) and the replace string pattern to something like <code>meet.\1</code> which would prefix the meeting code entered into the Virtual Reception with "meet.". You would then configure an associated Call Routing Rule to match calls placed to aliases prefixed with <code>meet.</code> which then strips off that prefix (to leave just the meeting code again) before directing the call to the external conference.</p>
post_replace_string	No	String	An optional regular expression used in conjunction with the post_match_string field to transform the meeting code into a distinct alias pattern that will match a Call Routing Rule configured to route calls into external conferences. (Only applies if the post-lookup regex match string is also configured and the entered code matches that regex.)
replace_string	No	String	<p>An optional regular expression used to transform the alias entered by the caller into the Virtual Reception. (Only applies if a regex match string is also configured and the entered alias matches that regex.)</p> <p>Leave this field blank if you do not want to change the alias entered by the caller.</p>
system_location_name	No	String	This is an optional field used in conjunction with the two_stage_dial_type setting, when a type other than <i>regular</i> is selected. If specified, a Conferencing Node in this system location will perform the SfB/Lync Conference ID lookup on the SfB/Lync server, or the Microsoft Teams or Google Meet code lookup, as appropriate. We recommend that a location is specified here, otherwise the transcoding node hosting the Virtual Reception will perform the lookup (which may lead to routability issues).

Field name	Required	Type	Description
teams_proxy_name	No	String	The name of the Teams Connector to use to resolve Microsoft Teams meeting codes entered in a Virtual Reception. If you do not specify anything, the Teams Connector associated with the outgoing location is used.
two_stage_dial_type	No	String	<p>The type of Virtual Reception:</p> <ul style="list-style-type: none">"regular": the default type of Virtual Reception, used for routing calls to VMRs, or to other devices and call control systems via the Infinity Gateway."mssip": a special type of Virtual Reception, used when you want to provide an IVR gateway to scheduled and ad hoc Skype for Business / Lync meetings."gms": a special type of Virtual Reception, used when you want to provide an IVR gateway to scheduled and ad hoc Google Meet meetings."teams": a special type of Virtual Reception, used when you want to provide an IVR gateway to scheduled and ad hoc Microsoft Teams meetings. <p>Default: "regular"</p>

Media Playback Service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "media_playback" (Media Playback Service):

Field name	Required	Type	Description
name	Yes	String	The name of the service. You could, for example, use the <code>local_alias</code> received in the configuration request. Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that service.
media_playlist_name	Yes	String	The name of the media playlist.
service_type	Yes	String	The type of service, in this case: <ul style="list-style-type: none"> "media_playback": a Media Playback Service
service_tag	Yes	String	A unique identifier used to track usage of this service.
allow_guests	No	Boolean	Whether to distinguish between Host and Guest participants: <ul style="list-style-type: none"> true: the conference has two types of participants: Hosts and Guests. The <code>pin</code> to be used by Hosts must be specified. A <code>guest_pin</code> can optionally be specified; if a <code>guest_pin</code> is not specified, Guests can join without a PIN. false: all participants have Host privileges Default: false
description	No	String	A description of the service.
guest_pin	No	String	Guest PIN — the secure access code for Guest participants.
guest_identity_provider_group	No	String	The set of Identity Providers to be offered to Guests to authenticate with, in order to use the service. If this is blank, Guests are not required to authenticate.
host_identity_provider_group	No	String	The set of Identity Providers to be offered to Hosts to authenticate with, in order to use the service. If this is blank, Hosts are not required to authenticate.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)

Field name	Required	Type	Description
non_idp_participants	No	String	<p>Determines whether participants joining a SSO-protected service from devices other than the Connect web app (for example SIP or H.323 endpoints) are allowed to dial in to the service.</p> <ul style="list-style-type: none">"disallow_all": these devices are placed in a waiting room where they must wait to be admitted by a Host."allow_if_trusted": these devices may join the service if they are locally registered. They must still enter a Host PIN or Guest PIN if either is required. All other devices are placed in a waiting room where they must wait to be admitted by a Host. <p>Default: "disallow_all"</p>
on_completion	No	String	<p>Action to perform on playlist completion. This is defined as a JSON object.</p> <p>To disconnect the user, use the syntax:</p> <pre>"on_completion": {"disconnect": true }</pre> <p>To perform a transfer, use the syntax:</p> <pre>"on_completion": {"transfer": {"conference": "<alias>"}}</pre> <p>or (if you also want to specify the role):</p> <pre>"on_completion": {"transfer": {"conference": "<alias>", "role": "<role>" }}</pre>
pin	No	String	Host PIN — the secure access code for Host participants.

Test Call Service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "test_call" (Test Call Service):

Field name	Required	Type	Description
name	Yes	String	The name of the service. You could, for example, use the <code>local_alias</code> received in the configuration request. Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that service.
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	The type of service, in this case: <ul style="list-style-type: none"> "test_call": a Test Call Service
bypass_proxy	No	Boolean	Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node. <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. Default: false
call_type	No	String	The call capability of the service. It can be limited to: <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only Default: "video"
crypto_mode	No	String	Controls the media encryption requirements for participants connecting to this service. <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) Default: <null> (use global setting)
description	No	String	A description of the service.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.

Field name	Required	Type	Description
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"><null>: use the global maximum call quality setting."sd": each participant is limited to SD quality."hd": each participant is limited to HD (720p) quality."fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)

Automatically dialed participants (ADP) fields

The `automatic_participants` list field, which may be included in the service configuration response for a `service_type` of "conference" (VMR) or "lecture" (Virtual Auditorium), contains the following fields per participant:

Field name	Required	Type	Description
local_alias	Yes	String	The calling or "from" alias. This is the alias that the recipient would use to return the call.
protocol	Yes	String	The protocol to use to place the outgoing call: <ul style="list-style-type: none">"h323": an H.323 call"sip": a SIP call"mssip": a Microsoft Skype for Business / Lync call"rtmp": uses the RTMP protocol; typically this is used for content streaming Calls are routed to externally-located participants based on the configuration of the system location used to place the call.
remote_alias	Yes	String	The alias of the endpoint to call.
role	Yes	String	The level of privileges the participant has in the conference: <ul style="list-style-type: none">"chair": the participant has Host privileges"guest": the participant has Guest privileges
call_type	No	String	The call capability of the participant. It can be limited to: <ul style="list-style-type: none">"video": main video plus presentation"video-only": main video only"audio": audio-only Default: "video"
dtmf_sequence	No	String	An optional DTMF sequence to transmit after the call to the dialed participant starts.

Field name	Required	Type	Description
keep_conference_alive	No	String	<p>Determines whether the conference continues when all other non-ADP participants have disconnected:</p> <ul style="list-style-type: none">"keep_conference_alive": the conference continues to run until this participant disconnects (applies to Hosts only)."keep_conference_alive_if_multiple": the conference continues to run as long as there are two or more "keep_conference_alive_if_multiple" participants and at least one of them is a Host."keep_conference_alive_never": the conference terminates automatically if this is the only remaining participant. <p>Default: "keep_conference_alive_if_multiple"</p> <p>For more information, see https://docs.pexip.com/admin/automatically_terminate.htm.</p>
local_display_name	No	String	The display name of the calling or "from" alias.
presentation_url	No	String	This additional parameter can be specified for RTMP calls to send the presentation stream to a separate RTMP destination.
remote_display_name	No	String	An optional friendly name for this participant. This may be used instead of the participant's alias in participant lists and as a text overlay in some layout configurations.
routing	No	String	<p>Specifies how to route the call:</p> <ul style="list-style-type: none">"manual": uses the requested protocol and the defaults for the specified <code>system_location_name</code>."routing_rule": routes the call according to the configured Call Routing Rules. This means that the dialed alias must match an outgoing Call Routing Rule for the call to be placed (using the protocols, outgoing location and call control systems etc. as configured for that rule). <p>Default: "manual"</p>
streaming	No	Boolean	<p>Identifies the dialed participant as a streaming or recording device.</p> <ul style="list-style-type: none">true: the participant is a streaming or recording devicefalse: the participant is not a streaming or recording device <p>Default: false</p>
system_location_name	No	String	The location of the Conferencing Node from which to place the call.

Example service configuration data responses

Response containing service configuration data

This is an example response of service configuration data for a "conference" service type:

```
{
  "status": "success",
  "action": "continue",
  "result": {
    "service_type": "conference",
    "name": "Alice Jones",
    "service_tag": "abcd1234",
    "description": "Alice Jones personal VMR",
    "pin": "1234",
    "allow_guests": true,
    "guest_pin": "5678",
    "view": "one_main_zero_pips",
    "enable_overlay_text": true,
    "automatic_participants":
    [
      {
        "remote_alias": "sip:alice@example.com",
        "remote_display_name": "Alice",
        "local_alias": "meet.alice@example.com",
        "local_display_name": "Alice's VMR",
        "protocol": "sip",
        "role": "chair",
        "system_location_name": "London"
      },
      {
        "remote_alias": "rtmp://example.com/live/alice_vmr",
        "local_alias": "meet.alice@example.com",
        "local_display_name": "Alice's VMR",
        "protocol": "rtmp",
        "role": "guest",
        "streaming": true
      }
    ]
  },
  "xyz_version": "1.2"
}
```

Response instructing Pexip Infinity to reject the call

This is an example response that tells Pexip Infinity to reject a call. This could be sent in a 200 OK message, or in a 404 response:

```
{
  "status": "success",
  "action": "reject"
}
```

Participant properties requests

This allows some of the participant's call properties, such as their display name or role, to be changed before they join the conference. This allows you, for example, to anonymize a participant's name or modify their role based on other properties of the call or their associated Identity Provider.

- For WebRTC participants it is applied **after** any PIN entry or SSO steps (and hence has access to Identity Provider attributes).
- For SIP/H.323 endpoints, it is applied **before** any PIN entry steps.

Participant policy is applied after any service configuration policy, and before media location policy.

Request URI: <policy_server_uri>/policy/v1/participant/properties

Pexip Infinity includes the following fields in a participant properties request:

Parameter	Description
bandwidth	The maximum requested bandwidth for the call. It is present on both inbound and outbound call requests but is meaningful only for inbound calls.
call_direction	The direction of the call that triggered the request. Values can be: <ul style="list-style-type: none"> "dial_in": calls in to Pexip Infinity "dial_out": calls dialed out from Pexip Infinity "non_dial": when policy is triggered by other requests that are not related to incoming or outgoing call setup, such as when an H.323 LRQ or a SIP SUBSCRIBE or OPTIONS request is received.
call_tag	An optional call tag that is assigned to a participant.
call_uuid	A unique identifier for the participant's call.
has_authenticated_display_name	Boolean indicating if the participant's display name was provided and authenticated by an Identity Provider.
idp_attributes	This contains any custom Identity Provider attributes for that participant. <p>Each attribute is prefixed with <code>idp_attribute_</code> when it is referenced as a parameter in external policy. For example, a custom attribute of "locale" would be <code>idp_attribute_locale</code>.</p> <p>Custom attributes for use in participant policy are assigned to Identity Providers in the settings when configuring the Identity Provider (Users & Devices > Identity Providers > Advanced Options).</p>
idp_uuid	The UUID of the Identity Provider who provided and authenticated the participant's display name, and provided any <code>idp_attributes</code> in participant policy.
local_alias	In the context of service and participant configuration requests, this is the incoming alias (typically the alias that the endpoint has dialed). This is the primary item of information that the policy server will use to return appropriate service configuration data.
location	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.
ms-subnet †	The sender's subnet address.
node_ip	The IP address of the Conferencing Node making the request.
p_Asserted-Identity†	The authenticated identity of the user sending the SIP message.
participant_type	The participant type: It can be: <ul style="list-style-type: none"> "standard": used for most types of calls. "api": when a WebRTC client joins there are normally two requests to policy. The first is for the API participant which has the "api" type. Normally, immediately after the "api" request, a "standard" participant request is made. "api_host": this is a variation of the normal "api" type which also starts the conference if the participant is a Host.
participant_uuid	The uuid associated with the participant.

Parameter	Description
preauthenticated_role	<p>The participant's role, either "guest", "chair" or null (note that in any jinja statements you need to test for: <code>is None</code>). You should consider the sequence of events that could determine the role's value provided here:</p> <ul style="list-style-type: none"> For WebRTC participants, participant policy is applied after any PIN entry or SSO steps (which may determine the participant's role). For SIP/H.323 endpoints, participant policy is applied before any PIN entry steps. Local policy is applied after any external policy (which may have modified the role).
protocol	<p>The call protocol.</p> <p>Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip".</p> <p>(Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)</p>
pseudo_version_id	The Pexip Infinity software build number.
registered	Boolean indicating whether the remote participant is registered or not.
remote_address	The IP address of the remote participant.
remote_alias	The name of the user or the registered alias of the endpoint. The <code>remote_alias</code> may include scheme information, for example <code>sip:alice@example.com</code> .
remote_display_name	The display name of the remote participant.
remote_port	The IP port of the remote participant.
service_name	The service name. This will match the <code>name</code> field returned by the policy server from the original service configuration request.
service_tag	The service tag associated with the <code>service_name</code> parameter.
supports_direct_media	Boolean indicating if the service supports direct media or not.
teams_tenant_id	Contains the Microsoft Teams tenant ID on an inbound Teams call to Pexip Infinity for Teams Rooms SIP/H.323 calling.
telehealth_request_id 	The telehealth call id.
trigger	<p>The trigger for the policy request.</p> <p>Values: "web", "web_avatar_fetch", "invite", "options", "subscribe", "setup", "arq", "lrq" or "unspecified".</p>
unique_service_name	<p>The unique name used by Pexip Infinity to identify the service:</p> <ul style="list-style-type: none"> For "gateway" services this is the matching rule name followed by a unique identifier (so as to distinguish between separate calls that match the same rule). For all other services, this is the same as the <code>service_name</code> parameter.
vendor	System details about the remote participant, such as manufacturer name and version number for hard endpoints, or browser and operating system details for soft clients.

Parameter	Description
version_id	The Pexip Infinity software version number.
† Only included if the request was triggered by a SIP message, and the parameter is included as a field in the SIP message header. Note that the ms_subnet and p_asserted_identity fields use only underscores and lower case characters in their names when used in local policy (normally they are referenced as ms-subnet and p_Asserted-Identity).	
‡ Only included in outbound call requests and for breakout rooms.	
†† Only included in outbound call requests.	
◇ Only included in Epic telehealth calls.	

This example shows a GET request for participant properties when alice@example.com has dialed meet.bob from a WebRTC endpoint:

```
GET /example.com/policy/v1/participant/properties?call_direction=dial_in&protocol=api&bandwidth=0&vendor=Mozilla...&registered=False&trigger=web&remote_display_name=Alice&remote_alias=alice@example.com&remote_address=10.44.151.2&remote_port=64062&call_tag=idp_uuid=&has_authenticated_display_name=False&supports_direct_media=True&location=location+1&node_ip=10.44.161.21&version_id=33&pseudo_version_id=73905.0.0&preauthenticated_role=chair&participant_type=api&participant_uuid=8bc...&local_alias=meet.bob&call_uuid=8bc...&unique_service_name=meet.bob&service_name=meet.bob&service_tag="
```

Response to a participant properties request

The response to a participant properties request takes the basic format:

```
{
  "status": "success",
  "action": "reject|continue",
  "result": { <participant_configuration_data> }
}
```

where:

- "action" is an optional value that, when included, instructs Pexip Infinity how to proceed in failure scenarios:
 - "reject" instructs Pexip Infinity to reject the call (i.e. return "conference not found").
 - "continue" use the data supplied in the "result" field.
- "result" is a JSON object of key value pairs that override the participant properties provided in the original participant information. The fields that may be included are listed below. All data fields are optional.

Field name	Required	Description
preauthenticated_role	No	An override for the participant's role. This must be either "chair" or "guest", or null if the originally supplied value was also null. For SIP and H.323 participants the original value will be null as participant policy is applied before any PIN entry (unlike WebRTC). Thus, returning null in the response means the caller will still be prompted for a PIN, whereas chair/guest will behave as if the user had entered the PIN for the specified role.
remote_alias	No	An override for the name of the user or the alias of the endpoint. Note that the original caller alias/name will still be recorded in the Pexip Infinity call logs when the participant first tries to connect to a conference.
remote_display_name	No	An override for the display name of the participant. It may return null if the originally supplied value was also null. Note that the original caller display name will still be recorded in the Pexip Infinity call logs when the participant first tries to connect to a conference.

Example participant properties data responses

Response containing participant properties

This is an example response of participant properties:

```
{
  "status": "success",
  "action": "continue",
  "result": {
    "preauthenticated_role": "guest",
    "remote_alias": "Witness A alias",
    "remote_display_name": "Witness A"
  }
}
```

Response instructing Pexip Infinity to reject the call

This is an example response that tells Pexip Infinity to reject a call.

```
{
  "status": "success",
  "action": "reject"
}
```

Media location requests

Obtains the system location to use for media allocation. A Conferencing Node assigned to that location will handle the media for the participant. A media location request is often made after a service configuration data request (and after any participant policy).

Request URI: <policy_server_uri>/policy/v1/participant/location

Pexip Infinity includes the following fields in a media location request:

Parameter	Description
bandwidth	The maximum requested bandwidth for the call. It is present on both inbound and outbound call requests but is meaningful only for inbound calls.
call_direction	The direction of the call that triggered the request. Values can be: <ul style="list-style-type: none">"dial_in": calls in to Pexip Infinity"dial_out": calls dialed out from Pexip Infinity"non_dial": when policy is triggered by other requests that are not related to incoming or outgoing call setup, such as when an H.323 LRQ or a SIP SUBSCRIBE or OPTIONS request is received.
call_tag	An optional call tag that is assigned to a participant.
has_authenticated_display_name	Boolean indicating if the participant's display name was provided and authenticated by an Identity Provider.
idp_uuid	The UUID of the Identity Provider who provided and authenticated the participant's display name, and provided any <code>idp_attributes</code> in participant policy.
local_alias	For media location and participant avatar requests this contains the originally-dialed incoming alias for the associated service.
location	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.
ms-subnet †	The sender's subnet address.
node_ip	The IP address of the Conferencing Node making the request.
p_Asserted-Identity†	The authenticated identity of the user sending the SIP message.
protocol	The call protocol. Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip". (Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)
proxy_node_address	The address of the Proxying Edge Node that is handling the call, if applicable.
proxy_node_location	The system location of the Proxying Edge Node that is handling the call, if applicable.
pseudo_version_id	The Pexip Infinity software build number.
registered	Boolean indicating whether the remote participant is registered or not.
remote_address	The IP address of the remote participant.
remote_alias	The name of the user or the registered alias of the endpoint. The <code>remote_alias</code> may include scheme information, for example <code>sip:alice@example.com</code> .
remote_display_name	The display name of the remote participant.

Parameter	Description
remote_port	The IP port of the remote participant.
service_name	The service name. This will match the name field returned by the policy server from the original service configuration request.
service_tag	The service tag associated with the service_name parameter.
supports_direct_media	Boolean indicating if the service supports direct media or not.
teams_tenant_id	Contains the Microsoft Teams tenant ID on an inbound Teams call to Pexip Infinity for Teams Rooms SIP/H.323 calling.
telehealth_request_id [‡]	The telehealth call id.
trigger	The trigger for the policy request. Values: "web", "web_avatar_fetch", "invite", "options", "subscribe", "setup", "arq", "lrq" or "unspecified".
unique_service_name	The unique name used by Pexip Infinity to identify the service: <ul style="list-style-type: none">For "gateway" services this is the matching rule name followed by a unique identifier (so as to distinguish between separate calls that match the same rule).For all other services, this is the same as the service_name parameter.
vendor	System details about the remote participant, such as manufacturer name and version number for hard endpoints, or browser and operating system details for soft clients.
version_id	The Pexip Infinity software version number.

[‡] Only included if the request was triggered by a SIP message, and the parameter is included as a field in the SIP message header. Note that the ms_subnet and p_asserted_identity fields use only underscores and lower case characters in their names when used in local policy (normally they are referenced as ms-subnet and p_Asserted-Identity).

[‡] Only included in Epic telehealth calls.

This example shows a GET request for the media location to use for alice@example.com:

```
GET /example.com/policy/v1/participant/location?protocol=sip&node_ip=10.44.99.2&service_name=meet.bob&registered=False&remote_address=10.44.75.250&version_id=16&service_tag=&proxy_node_address=&bandwidth=0&pseudo_version_id=36402.0.0&vendor=TANDBERG/518(TC6.0.1.65adebe)&unique_service_name=meet.bob&local_alias=meet.bob&proxy_node_location=&remote_port=58426&idp_uid=&has_authenticated_display_name=False&supports_direct_media=True&call_direction=dial_in&remote_alias=sip:alice@example.com&remote_display_name=Alice&trigger=invite&location=London
```

Media location response

The response to a media location request takes the format:

```
{  
  "status": "success",  
  "result": { <location_data> }  
}
```

where "result" is a JSON object of key value pairs that describes the media location to use. The fields that may be included are:

Field name	Required	Description
location	Yes	The name* of the principal location to use for media allocation. A Conferencing Node assigned to that location will handle the media for the participant.

Field name	Required	Description
overflow_locations	No	<p>A list of one or more system location names* to handle the media if the principal location has reached its capacity. The order of the list defines the order that the locations are used.</p> <p>As of version 23.2, we recommend that you use overflow_locations instead of the primary_overflow_location and secondary_overflow_location fields. However, to ensure backwards compatibility with any previously-configured policy, any overflow locations specified here will be appended to the locations specified in the primary_overflow_location and secondary_overflow_location fields if they are also nominated.</p> <p>Note that if the overflow_locations field is set by external policy then these locations cannot subsequently be manipulated by local policy. Local policy only receives the content of the primary_overflow_location and secondary_overflow_location fields.</p>
primary_overflow_location	No	The name* of the system location to handle the media if the principal location has reached its capacity. As of version 23.2 we recommend that you use overflow_locations instead.
secondary_overflow_location	No	The name* of the system location to handle the media if both the principal location and the primary overflow location have reached their capacity. As of version 23.2 we recommend that you use overflow_locations instead.

* The returned "name" must match the name of a location configured within Pexip Infinity. If any of the location names included in the response do not match a configured location within Pexip Infinity, the entire response is deemed to have failed and Pexip Infinity will fall back to its own default behavior for that request.

Example media location data response

This is an example response containing media location data:

```
{
  "status": "success",
  "result": {
    "location": "Oslo",
    "overflow_locations": ["London", "Paris", "New York"]
  }
}
```

This sets the location to use for media to "Oslo", and sets "London", then "Paris" and finally "New York" as the overflow locations to use if "Oslo" is out of capacity.

Participant avatar requests

Obtains the image to display to represent a conference participant or directory contact.

Request URI: <policy_server_uri>/policy/v1/participant/avatar/<alias> where <alias> is the alias of the participant/contact whose avatar is required.

Pexip Infinity includes the following fields in a participant avatar request:

Parameter	Description
bandwidth	The maximum requested bandwidth for the call. It is present on both inbound and outbound call requests but is meaningful only for inbound calls.

Parameter	Description
call_direction	The direction of the call that triggered the request. Values can be: <ul style="list-style-type: none"> "dial_in": calls in to Pexip Infinity "dial_out": calls dialed out from Pexip Infinity "non_dial": when policy is triggered by other requests that are not related to incoming or outgoing call setup, such as when an H.323 LRQ or a SIP SUBSCRIBE or OPTIONS request is received.
call_tag	An optional call tag that is assigned to a participant.
has_authenticated_display_name	Boolean indicating if the participant's display name was provided and authenticated by an Identity Provider.
height *	The required height in pixels of the image to be returned.
idp_uuid	The UUID of the Identity Provider who provided and authenticated the participant's display name, and provided any idp_attributes in participant policy.
local_alias	For media location and participant avatar requests this contains the originally-dialed incoming alias for the associated service.
location *	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.
ms-subnet †	The sender's subnet address.
node_ip *	The IP address of the Conferencing Node making the request.
p_Asserted-Identity†	The authenticated identity of the user sending the SIP message.
protocol	The call protocol. Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip". (Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)
proxy_node_address	The address of the Proxying Edge Node that is handling the call, if applicable.
proxy_node_location	The system location of the Proxying Edge Node that is handling the call, if applicable.
pseudo_version_id *	The Pexip Infinity software build number.
registered *	Boolean indicating whether the remote participant is registered or not.
remote_address	The IP address of the remote participant.
remote_alias	The name of the user or the registered alias of the endpoint. The remote_alias may include scheme information, for example sip:alice@example.com.
remote_display_name	The display name of the remote participant.
remote_port	The IP port of the remote participant.
role	The role associated with the remote_alias. Values: "chair" (for Host participants), "guest", or "unknown" (a participant who is at the PIN entry screen, or who is at the Waiting for Host screen but their role has not yet been determined).

Parameter	Description
service_name	The service name. This will match the name field returned by the policy server from the original service configuration request.
service_tag	The service tag associated with the service_name parameter.
supports_direct_media	Boolean indicating if the service supports direct media or not.
teams_tenant_id	Contains the Microsoft Teams tenant ID on an inbound Teams call to Pexip Infinity for Teams Rooms SIP/H.323 calling.
trigger	The trigger for the policy request. Values: "web", "web_avatar_fetch", "invite", "options", "subscribe", "setup", "arq", "lrq" or "unspecified".
unique_service_name	The unique name used by Pexip Infinity to identify the service: <ul style="list-style-type: none"> For "gateway" services this is the matching rule name followed by a unique identifier (so as to distinguish between separate calls that match the same rule). For all other services, this is the same as the service_name parameter.
vendor	System details about the remote participant, such as manufacturer name and version number for hard endpoints, or browser and operating system details for soft clients.
version_id *	The Pexip Infinity software version number.
width *	The required width in pixels of the image to be returned.

* These are the only parameters included in a participant avatar request when Pexip Infinity is retrieving directory information (as opposed to service participant-related information where all parameters are included).

† Only included if the request was triggered by a SIP message, and the parameter is included as a field in the SIP message header. Note that the ms_subnet and p_asserted_identity fields use only underscores and lower case characters in their names when used in local policy (normally they are referenced as ms-subnet and p_Asserted-Identity).

This example shows a GET request for a participant avatar for the alias "alice" who is in the conference "meet.bob", where the participant device is an unregistered Connect app:

```
GET /example.com/policy/v1/participant/avatar/alice?protocol=webrtc&node_ip=10.44.99.2&service_name=meet.bob&registered=False&remote_address=10.44.75.250&version_id=16&service_tag=&bandwidth=0&pseudo_version_id=36402.0.0&vendor=Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36&height=100&unique_service_name=meet.bob&local_alias=meet.bob&remote_port=58426&idp_uuid=&has_authenticated_display_name=False&supports_direct_media=False&call_direction=dial_in&remote_alias=alice&remote_display_name=Alice&width=100&trigger=invite&role=chair&location=London
```

This example shows a GET request for a participant avatar for the alias "alice" that has been triggered in response to a previous directory information request:

```
GET /example.com/policy/v1/participant/avatar/alice?node_ip=10.47.2.46&registered=True&version_id=16&height=40&width=40&location=London&pseudo_version_id=36402.0.0
```

Participant avatar response

The response for an avatar request must return a **Content-Type** of **image/jpeg**.

All JPEG images must use the RGB or RGBA color space (CMYK is not supported), and be of the requested size (width, height).

If a valid JPEG image is not returned or the image is the wrong size, Pexip Infinity will return a 404 Not Found response to the caller (i.e. the Connect app) which will then use its standard placeholder participant image.

Directory information requests

Obtains directory information — a list of device or VMR aliases and their associated names. This is used to provide phonebook information to Connect apps that are registered to a Pexip Infinity Conferencing Node.

Request URI: <policy_server_uri>/policy/v1/registrations

Pexip Infinity includes the following fields in a directory information request:

Parameter	Description
call_tag	An optional call tag that is assigned to a participant.
limit	The maximum number of results to return (currently this is always set to 10).
location	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.
node_ip	The IP address of the Conferencing Node making the request.
pseudo_version_id	The Pexip Infinity software build number.
q	The string to lookup in the directory.
registration_alias	The alias of the registered client performing the directory lookup.
version_id	The Pexip Infinity software version number.

This example shows a GET request for directory information where the search string is "ali":

```
GET /example.com/policy/v1/registrations?q=ali&registration_alias=bob@example.com&limit=10&location=london&node_ip=10.44.155.21&pseudo_version_id=36402.0.0&version_id=16
```

Directory information response

The response to a directory information request takes the format:

```
{
  "status": "success",
  "result": [ <directory_data> ],
  "ignore_local": True|False
}
```

where:

- "result" is a JSON list containing objects representing individual directory entries. Each JSON object in the list contains the following fields:

Field name	Required	Description
alias	Yes	The alias of the device or VMR. This is the alias Pexip Infinity will use as the <alias> in the URI in a subsequent participant avatar request , and the alias that Connect apps will dial.
description	Yes	A description or name associated with the alias.
username	Yes	The username associated with the device or VMR. This is currently unused by Pexip Infinity.

The policy server should sort the list of aliases into the order in which it wants them to be presented.

- "ignore_local" is optional and defaults to False. If set to True it instructs Pexip Infinity to ignore the aliases of any local services or devices i.e. to only use the aliases returned from the policy server as the directory information. When "ignore_local" is False, Pexip Infinity adds the aliases of any local services or devices to the list of aliases returned by the policy server.

Example directory information responses

This is an example of a JSON dictionary of directory information data containing 2 devices:

```
{
  "status": "success",
  "result": [
    {
      "username": "alice",
      "alias": "alice@example.com",
      "description": "Alice's VMR"
    },
    {
      "username": "bob",
      "alias": "bob@example.com",
      "description": "Bob's VMR"
    }
  ]
}
```

This is an example response that instructs Pexip Infinity to not supply any directory information at all — no aliases are returned by the policy server and it also tells Pexip Infinity to ignore any local aliases ("ignore_local" is True):

```
{
  "status": "success",
  "result": [],
  "ignore_local": True
}
```

Registration alias requests

Used to determine whether a device alias is allowed to register to a Conferencing Node.

Request URI: <policy_server_uri>/policy/v1/registrations/<alias> where <alias> is the alias of the device that is making the registration request.

Pexip Infinity includes the following fields in a registration request:

Parameter	Description
call_tag	An optional call tag that is assigned to a participant.
location	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.
node_ip	The IP address of the Conferencing Node making the request.
protocol	The call protocol. Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip". (Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.) For registration requests this is the registration protocol. Values: "webrtc", "sip", or "h323".
pseudo_version_id	The Pexip Infinity software build number.
version_id	The Pexip Infinity software version number.

This example shows a GET request for a registration alias of "alice@example.com":

```
GET /example.com/policy/v1/registrations/alice@example.com?pseudo_version_id=36402.0.0&protocol=webrtc&location=london&version_id=16&node_ip=10.44.155.21
```

Registration alias response


The response to a registration alias request takes the format:

```
{
  "status": "success",
  "action": "reject|continue",
  "result": { <alias_data> }
}
```

where:

- "action" instructs Pexip Infinity to either reject the registration or proceed with the registration request:
 - "reject" instructs Pexip Infinity to reject the registration (note that "result": {} must also be included in the response).
 - "continue" (or any other value except "reject") instructs Pexip Infinity to proceed with the registration request.
- "result" is a JSON object of key value pairs that can contain the credentials to use to authenticate the registration request. The fields are:

Field name	Required	Description
username	No	The username associated with the device.
password	No	The password associated with the device and username.

 Note that the password is sent "in the clear" so we recommend using a secure https connection to your policy server.

The device alias may register if it supplies matching credentials.

Use "result": {} in reject responses, or to fall back to Pexip Infinity's default behavior (to check if the alias and optionally any associated credentials are configured in its local database of allowed aliases).

- If "status" is not "success" or an error code e.g. 404 is returned, then Pexip Infinity will fall back to its own default behavior.

Example registration alias responses

Allowed alias (with authentication): this is an example response containing the required credentials for an allowed alias (the device must then supply the matching credentials to be permitted to register):

```
{
  "status": "success",
  "result": {
    "username": "alice",
    "password": "password123"
  }
}
```

Allowed alias (without authentication): this is an example response that allows the alias to register without authentication (not recommended):

```
{
  "status": "success",
  "result": {
    "username": "",
    "password": ""
  }
}
```

The response must include username/password with empty fields.

Denied alias: this is an example response to reject the requested alias:

```
{
  "status": "success",
  "action": "reject",
  "result": {}
}
```

Fallback to local database: this is an example response to fall back to Pexip Infinity's default behavior,:

```
{
  "status": "success",
  "action": "continue",
  "result": {}
}
```

Summary of request parameters per request type

This table summarizes which fields are included by Pexip Infinity in each request type:

Parameter	Description	Service config	Participant config	Media location	Participant avatar	Directory info	Registration alias
bandwidth	The maximum requested bandwidth for the call. It is present on both inbound and outbound call requests but is meaningful only for inbound calls.	✓	✓	✓	✓		
breakout_uuid	The uuid of a breakout room (if applicable).	✓					
call_direction	The direction of the call that triggered the request. Values can be: <ul style="list-style-type: none">"dial_in": calls in to Pexip Infinity"dial_out": calls dialed out from Pexip Infinity"non_dial": when policy is triggered by other requests that are not related to incoming or outgoing call setup, such as when an H.323 LRQ or a SIP SUBSCRIBE or OPTIONS request is received.	✓	✓	✓	✓		
call_tag	An optional call tag that is assigned to a participant.	✓	✓	✓	✓	✓	✓
call_uuid	A unique identifier for the participant's call.		✓				
has_authenticated_display_name	Boolean indicating if the participant's display name was provided and authenticated by an Identity Provider.	✓	✓	✓	✓		
height *	The required height in pixels of the image to be returned.						

Parameter	Description	Service config	Participant config	Media location	Participant avatar	Directory info	Registration alias
idp_attributes	<p>This contains any custom Identity Provider attributes for that participant.</p> <p>Each attribute is prefixed with <code>idp_attribute_</code> when it is referenced as a parameter in external policy. For example, a custom attribute of "locale" would be <code>idp_attribute_locale</code>.</p> <p>Custom attributes for use in participant policy are assigned to Identity Providers in the settings when configuring the Identity Provider (Users & Devices > Identity Providers > Advanced Options).</p>		✓				
idp_uuid	The UUID of the Identity Provider who provided and authenticated the participant's display name, and provided any <code>idp_attributes</code> in participant policy.	✓	✓	✓	✓		
limit	The maximum number of results to return (currently this is always set to 10).					✓	
local_alias	<p>In the context of service and participant configuration requests, this is the incoming alias (typically the alias that the endpoint has dialed). This is the primary item of information that the policy server will use to return appropriate service configuration data.</p> <p>For media location and participant avatar requests this contains the originally-dialed incoming alias for the associated service.</p>	✓	✓	✓	✓		

Parameter	Description	Service config	Participant config	Media location	Participant avatar	Directory info	Registration alias
location *	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.	✓	✓	✓		✓	✓
ms-subnet †	The sender's subnet address.	✓	✓	✓	✓		
node_ip *	The IP address of the Conferencing Node making the request.	✓	✓	✓		✓	✓
p_Asserted-Identity†	The authenticated identity of the user sending the SIP message.	✓	✓	✓	✓		
participant_type	<p>The participant type: It can be:</p> <ul style="list-style-type: none"> "standard": used for most types of calls. "api": when a WebRTC client joins there are normally two requests to policy. The first is for the API participant which has the "api" type. Normally, immediately after the "api" request, a "standard" participant request is made. "api_host": this is a variation of the normal "api" type which also starts the conference if the participant is a Host. 		✓				
participant_uuid	The uuid associated with the participant.		✓				

Parameter	Description	Service config	Participant config	Media location	Participant avatar	Directory info	Registration alias
preauthenticated_role	<p>The participant's role, either "guest", "chair" or null (note that in any jinja statements you need to test for: <code>is None</code>).</p> <p>You should consider the sequence of events that could determine the role's value provided here:</p> <ul style="list-style-type: none"> For WebRTC participants, participant policy is applied after any PIN entry or SSO steps (which may determine the participant's role). For SIP/H.323 endpoints, participant policy is applied before any PIN entry steps. Local policy is applied after any external policy (which may have modified the role). 		✓				
protocol	<p>The call protocol.</p> <p>Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip".</p> <p>(Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)</p> <p>For registration requests this is the registration protocol.</p> <p>Values: "webrtc", "sip", or "h323".</p>	✓	✓	✓	✓		✓
proxy_node_address	The address of the Proxying Edge Node that is handling the call, if applicable.			✓	✓		
proxy_node_location	The system location of the Proxying Edge Node that is handling the call, if applicable.			✓	✓		
pseudo_version_id *	The Pexip Infinity software build number.	✓	✓	✓		✓	✓
q	The string to lookup in the directory.					✓	

Parameter	Description	Service config	Participant config	Media location	Participant avatar	Directory info	Registration alias
registered *	Boolean indicating whether the remote participant is registered or not.	✓	✓	✓			
registration_alias	The alias of the registered client performing the directory lookup.					✓	
remote_address	The IP address of the remote participant.	✓	✓	✓	✓		
remote_alias	The name of the user or the registered alias of the endpoint. The <code>remote_alias</code> may include scheme information, for example <code>sip:alice@example.com</code> .	✓	✓	✓	✓		
remote_display_name	The display name of the remote participant.	✓	✓	✓	✓		
remote_port	The IP port of the remote participant.	✓	✓	✓	✓		
role	The role associated with the <code>remote_alias</code> . Values: "chair" (for Host participants), "guest", or "unknown" (a participant who is at the PIN entry screen, or who is at the Waiting for Host screen but their role has not yet been determined).					✓	
service_name ‡	The service name. This will match the name field returned by the policy server from the original service configuration request.		✓	✓	✓		
service_tag ††	The service tag associated with the <code>service_name</code> parameter.		✓	✓	✓		
supports_direct_media	Boolean indicating if the service supports direct media or not.	✓	✓	✓	✓		
teams_tenant_id	Contains the Microsoft Teams tenant ID on an inbound Teams call to Pexip Infinity for Teams Rooms SIP/H.323 calling.	✓	✓	✓	✓		

Parameter	Description	Service config	Participant config	Media location	Participant avatar	Directory info	Registration alias
telehealth_request_id ◇	The telehealth call id.	✓	✓	✓			
trigger	The trigger for the policy request. Values: "web", "web_avatar_fetch", "invite", "options", "subscribe", "setup", "arq", "lrq" or "unspecified".	✓	✓	✓	✓		
unique_service_name ‡	The unique name used by Pexip Infinity to identify the service: <ul style="list-style-type: none"> For "gateway" services this is the matching rule name followed by a unique identifier (so as to distinguish between separate calls that match the same rule). For all other services, this is the same as the <code>service_name</code> parameter. 		✓	✓	✓		
vendor	System details about the remote participant, such as manufacturer name and version number for hard endpoints, or browser and operating system details for soft clients.	✓	✓	✓	✓		
version_id *	The Pexip Infinity software version number.	✓	✓	✓		✓	✓
width *	The required width in pixels of the image to be returned.						

* These are the only parameters included in a participant avatar request when Pexip Infinity is retrieving directory information (as opposed to service participant-related information where all parameters are included).

† Only included if the request was triggered by a SIP message, and the parameter is included as a field in the SIP message header. Note that the `ms_subnet` and `p_asserted_identity` fields use only underscores and lower case characters in their names when used in local policy (normally they are referenced as `ms-subnet` and `p_Asserted-Identity`).

‡ Only included in outbound call requests and for breakout rooms.

†† Only included in outbound call requests.

◇ Only included in Epic telehealth calls.

Enabling local policy

Local policy allows you to manipulate service configuration, some participant properties, and media location data (that has been provided either via the external policy API, or has been retrieved from Pexip Infinity's own database, or was provided by the endpoint in the call request) by running a jinja2 script against that data.

Local policy has a more limited scope than external policy in what it can control, but it is easier to implement and runs locally on each Conferencing Node.

Configuring policy profiles for local policy

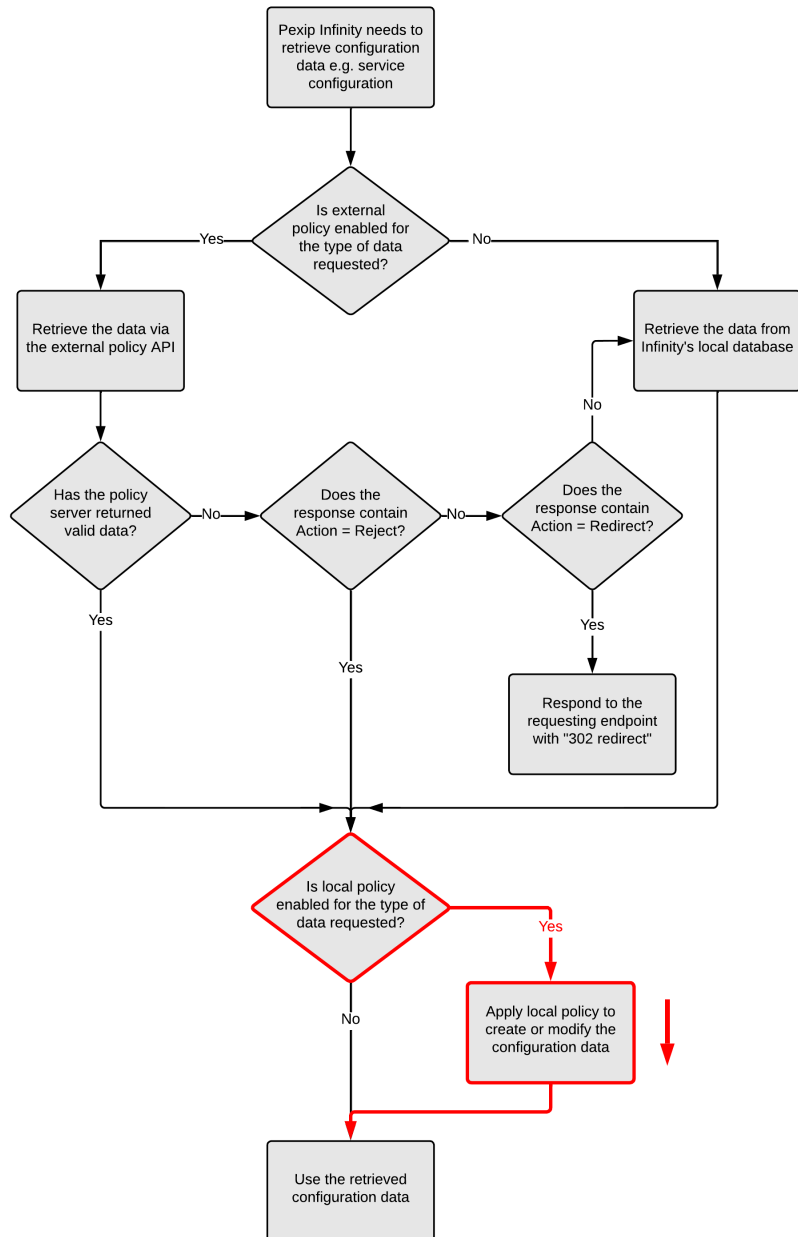
You can configure Pexip Infinity to use both external and local policy depending on your requirements. When both external and local policy are enabled, external policy is applied first to retrieve the configuration data from the external system, and then local policy is applied to that retrieved data (which can then conditionally modify that data). The flow chart (below) shows Pexip Infinity's standard processing logic when policy profiles are used and highlights where local policy is applied.

To configure Pexip Infinity to use local policy:

1. Go to **Call Control > Policy Profiles**.
2. Select **Add Policy profile** and then configure that profile:
 - Select **Apply local policy** for the types of configuration data (service configuration, participant and media location data types only) that you want to modify via local policy.
 - In the **Script** text box, enter the jinja2 script that you want to execute against that data, and save your changes.

See [Writing local policy scripts](#) for full information about how to construct your jinja2 scripts.
3. Go to **Platform > Locations**.
4. Select each location in turn and specify the **Policy profile** that the Conferencing Nodes in that location should use when making policy decisions.

For more information on configuring policy profiles and how to combine external policy with local policy, see [Configuring policy profiles](#).



Writing local policy scripts

Local policy scripts are configured as part of a policy profile and allow you to manipulate service configuration, some participant properties, and media location data. This topic explains how to write a jinja2 script to control Pexip Infinity's call behavior.

- i Using local policy requires proficient scripting skills. For large production environments we recommend that you contact your Pexip authorized representative.
- i We recommend that you test your policy scripts carefully with a variety of different inputs. We will attempt to maintain backwards compatibility in future releases of the Pexip Infinity platform but the introduction of new features may affect the behavior of existing scripts. We strongly recommend that you test your scripts in a lab environment before using them in a production system, particularly after an upgrade to the Pexip Infinity software.

This topic covers:

- [Types of configuration data](#)
- [Writing a jinja2 script to control call behavior](#)
- [Getting started in constructing a script](#)
- [Supported variables](#)
- [Response formats](#) ([Service configuration data responses](#), [Participant configuration responses](#) and [Media location data responses](#))

You can also use the Administrator interface's built-in facility for [Testing local policy scripts](#), and refer to our [Example local policy scripts](#) and our guide to [Using filters in local policy scripts](#).

Types of configuration data

The following table shows the types of configuration data that can be modified via local policy, and when that data is requested by Pexip Infinity:

Type of data	Purpose
Service configuration	The configuration details of a service. Pexip Infinity typically requires this data when it: <ul style="list-style-type: none">• receives an incoming call request• needs to place a call to a given alias
Participant properties	This allows some of the participant's call properties, such as their display name or role, to be changed before they join the conference. This allows you, for example, to anonymize a participant's name or modify their role based on other properties of the call or their associated Identity Provider. <ul style="list-style-type: none">• For WebRTC participants it is applied after any PIN entry or SSO steps (and hence has access to Identity Provider attributes).• For SIP/H.323 endpoints, it is applied before any PIN entry steps. Participant policy is applied after any service configuration policy, and before media location policy.
Media location	The system location to use for media allocation. A Conferencing Node assigned to that location will handle the media for the participant. A media location request is often made after a service configuration data request (and after any participant policy).

You may see multiple service configuration, participant and media location requests when handling Connect app participants.

Writing a jinja2 script to control call behavior

Your scripts are configured as part of a policy profile (**Call Control > Policy Profiles**), and that profile is then assigned to one or more system locations.

Pexip Infinity's local policy scripts use a subset of the jinja2 templating language (see [Template Designer Documentation](#)).

These scripts/templates consist of the following elements:

- **literal text** that you want to add to the output or result
- **variables** that are substituted with values from the source request (`call_info` and `participant`), or that contain the existing configuration data supplied by external policy or the Pexip Infinity database (`service_config` and `suggested_media_overflow_locations`)
- **filters** that can manipulate text or modify the content of variables or text strings, such as `pex_update` or `pex_to_json`
- **delimiters** such as `{{...}}` and pipes `|` which are used to define filter expressions
- **jinja statements and control structures** (see [Jinja Control Structures](#))

All scripts are subject to a maximum length of 49152 characters. If your scripts are getting close to this limit, we recommend that you consider using external policy instead.

Getting started in constructing a script

Local policy allows you to manipulate the service configuration, participant and media location data that has already been fetched from Pexip Infinity's own database, provided by the caller, or has been supplied via the external policy API. In the case of service configuration data, there may be no existing data available — for example if a call attempt was made to an alias that does not exist.

This existing data that can be manipulated is held in three variables:

- `service_config` for service configuration policy,
- `participant` for participant policy, and
- `suggested_media_overflow_locations` for media location policy.

Your local policy script will typically use [filters](#) to modify some elements of these variables, or replace the contents of those variables with completely different data, based on certain conditions, or it can choose to pass through the existing data unchanged. Those conditions will typically involve references to that existing data or to the original call information that led to the request to provide service configuration, participant and media location data. That existing call information is held in the `call_info` variable (which cannot be modified) and is explained in [Supported variables](#) below.

The objective of the script is to return the service configuration, participant or media location data to Pexip Infinity. That response — the output/result of the script — must be a JSON object that fully defines the service or media location i.e. it must include all of the mandatory [response fields](#) as a minimum.

An example basic response containing service configuration data is shown below. It simply returns the basic minimum set of configuration fields for a VMR (which has a `service_type` of "conference"). All other properties of the service will use Pexip Infinity's default values:

```
{
  "action": "continue",
  "result": {
    "service_type": "conference",
    "name": "Alice Jones",
    "service_tag": "abcd1234"
  }
}
```

A more typical script is likely to modify one or more elements of the existing service configuration data, for example:

```
{
  {% if service_config %}
    "action": "continue",
    "result": {{service_config|pex_update({"enable_chat": "no", "enable_overlay_text": True})|pex_to_json}}
  {% else %}
    "action": "reject",
    "result": {}
  {% endif %}
}
```

This script first checks if there is some existing service configuration data (which is held in the `service_config` variable) and then updates that data by disabling chat messages and turning on the display of participant names; otherwise (if there is no existing data) it rejects the request and returns null data. Note how the script uses the `pex_update` and the `pex_to_json` filters to update and then format the elements of the `service_config` data variable — you should use this as the model for your own scripts.

For full details of what may be contained in the policy response, see [Service configuration data responses](#) and [Media location data responses](#).

- i** To help you construct and test your scripts, Pexip Infinity has a built-in [script testing facility](#), and there are some [example scripts](#) that illustrate how to structure your scripts.

Supported variables

There is a limited set of system variables ([Configuration variables](#), [Call information variables](#) and [Participant properties variables](#)) that you can use within your script. Capitalization of variable names is important. The variables **must** be spelled exactly as shown.

Note that these variables are represented in a Python dictionary format (which is very similar, but not identical, to JSON). This is why the [pex_to_json filter](#) is required if you want to return the content of the configuration variables as the response to a service configuration or media location request.

Configuration variables

The service configuration and media location data that has already been fetched from Pexip Infinity's own database or has been supplied via the external policy API is held in the following variables:

Variable name	Description and example usage
<code>service_config</code>	<p>The <code>service_config</code> variable holds the existing service configuration data. This variable only applies to scripts run as service configuration policy.</p> <ul style="list-style-type: none"> When referring to specific service configuration data elements, you must use the <code>service_config</code> prefix. For example, use <code>service_config.pin</code> to refer to the service PIN, and <code>service_config.service_type</code> to refer to the type of service (either "conference" or "lecture" for a Virtual Meeting Room or Virtual Auditorium, "gateway" for an Infinity Gateway call, "two_stage_dialing" for a Virtual Reception, "media_playback" for a Media Playback Service, or "test_call" for a Test Call Service). If no service configuration data has been retrieved — for example if a call attempt was made to an alias that does not exist, this variable will be null, but it can still be updated to contain the required configuration data. See Service configuration data responses for the full set of service configuration elements within the <code>service_config</code> variable. (Note that if you use the <code>pex_debug_log</code> filter to log the content of the <code>service_config</code> variable, you may see additional fields reported to those listed here. Those additional fields are for information only and may change in future releases.) <p>Example <code>service_config</code> data (displayed here in JSON format):</p> <pre>"service_config": { "allow_guests": true, "automatic_participants": [{ "description": "Dial out to VMR owner", "local_alias": "meet.alice@example.com", "local_display_name": "Alice's VMR", "protocol": "sip", "remote_alias": "sip:alice@example.com", "role": "chair", "system_location_name": "London" }], "description": "Alice Jones personal VMR", "enable_overlay_text": true, "guest_pin": "5678", "name": "Alice Jones", "pin": "1234", "service_tag": "abcd1234", "service_type": "conference" }</pre>
<code>suggested_media_overflow_locations</code>	<p>The <code>suggested_media_overflow_locations</code> variable holds the existing media location data. This variable only applies to scripts run as media location policy. When referring to specific media location data elements, you must use the <code>suggested_media_overflow_locations</code> prefix followed by the element name — either <code>location</code>, <code>primary_overflow_location</code> or <code>secondary_overflow_location</code> which respectively refer to the principal location and the primary and secondary overflow locations that will handle the call media, for example <code>suggested_media_overflow_locations.location</code>.</p> <p>Example <code>suggested_media_overflow_locations</code> data (displayed here in JSON format):</p> <pre>"suggested_media_overflow_locations": { "location": "New York", "primary_overflow_location": "Paris", "secondary_overflow_location": "Peckham" }</pre>

Call information variables

The following table shows the call information variables that may be available to your script. These variables contain facts about the call and thus the data in these variables cannot be changed by policy.

For each variable the table indicates if it is available to service configuration data requests and/or media location data requests. When using these variables you must use the "call_info." prefix. For example to refer to the location associated with the Conferencing Node making the request, use `call_info.location` and to refer to the call protocol use `call_info.protocol`. (Note that the `call_info` variables are the same as those used in external policy API requests.)

Available call_info variables	Description	Service config	Participant config	Media location
bandwidth	The maximum requested bandwidth for the call. It is present on both inbound and outbound call requests but is meaningful only for inbound calls.	✓	✓	✓
breakout_uuid	The uuid of a breakout room (if applicable).	✓		
call_direction	The direction of the call that triggered the request. Values can be: <ul style="list-style-type: none"> "dial_in": calls in to Pexip Infinity "dial_out": calls dialed out from Pexip Infinity "non_dial": when policy is triggered by other requests that are not related to incoming or outgoing call setup, such as when an H.323 LRQ or a SIP SUBSCRIBE or OPTIONS request is received. 	✓	✓	✓
call_tag	An optional call tag that is assigned to a participant.	✓	✓	✓
external_policy_disposition	Indicates the result of a prior external policy request for that request type. <ul style="list-style-type: none"> For service configuration requests it can be either "CONTINUE", "REJECT" or "REDIRECT". For participant requests it can be "CONTINUE" or "REJECT". Note that it is only provided if external policy is enabled, the request was successful, and returned a valid action.	✓	✓	
has_authenticated_display_name	Boolean indicating if the participant's display name was provided and authenticated by an Identity Provider.	✓	✓	✓
idp_uuid	The UUID of the Identity Provider who provided and authenticated the participant's display name, and provided any <code>idp_attributes</code> in participant policy.	✓	✓	✓
local_alias	In the context of service and participant configuration requests, this is the incoming alias (typically the alias that the endpoint has dialed). This is the primary item of information that the policy script will use to return appropriate service configuration data. For media location and participant avatar requests this contains the originally-dialed incoming alias for the associated service.	✓	✓	✓
location	The name of the Pexip Infinity system location associated with the Conferencing Node making the request/notification.	✓	✓	✓
ms_subnet †	The sender's subnet address. Note that the value of "ms_subnet" is formatted as a JSON list, for example: ["10.47.5.0"]	✓	✓	✓
node_ip	The IP address of the Conferencing Node making the request.	✓	✓	✓

Available call_info variables	Description	Service config	Participant config	Media location
p_asserted_identity†	The authenticated identity of the user sending the SIP message. Note that the value of "p_asserted_identity" is formatted as a JSON list, for example: ["Alice"<sip:alice@example.com>"]	✓	✓	✓
protocol	The call protocol. Values: "api", "webrtc", "sip", "rtmp", "h323" or "mssip". (Note that the protocol is always reported as "api" when a Connect app dials in to Pexip Infinity.)	✓	✓	✓
proxy_node_address	The address of the Proxying Edge Node that is handling the call, if applicable.			✓
proxy_node_location	The system location of the Proxying Edge Node that is handling the call, if applicable.			✓
pseudo_version_id	The Pexip Infinity software build number.	✓	✓	✓
registered	Boolean indicating whether the remote participant is registered or not.	✓	✓	✓
remote_address	The IP address of the remote participant.	✓	✓	✓
remote_alias	The name of the user or the registered alias of the endpoint. The remote_alias may include scheme information, for example sip:alice@example.com.	✓	✓	✓
remote_display_name	The display name of the remote participant.	✓	✓	✓
remote_port	The IP port of the remote participant.	✓	✓	✓
service_name ‡	The service name. This will match the name field returned by the policy script from the original service configuration request.		✓	✓
service_tag ††	The service tag associated with the service_name parameter.		✓	✓
supports_direct_media	Boolean indicating if the service supports direct media or not.	✓	✓	✓
teams_tenant_id	Contains the Microsoft Teams tenant ID on an inbound Teams call to Pexip Infinity for Teams Rooms SIP/H.323 calling.	✓	✓	✓
telehealth_request_id ◊	The telehealth call id.	✓	✓	✓
trigger	The trigger for the policy request. Values: "web", "web_avatar_fetch", "invite", "options", "subscribe", "setup", "arq", "lrq" or "unspecified".	✓	✓	✓
unique_service_name ‡	The unique name used by Pexip Infinity to identify the service: <ul style="list-style-type: none"> For "gateway" services this is the matching rule name followed by a unique identifier (so as to distinguish between separate calls that match the same rule). For all other services, this is the same as the service_name parameter. 		✓	✓
vendor	System details about the remote participant, such as manufacturer name and version number for hard endpoints, or browser and operating system details for soft clients.	✓	✓	✓

Available call_info variables	Description	Service config	Participant config	Media location
version_id	The Pexip Infinity software version number.	✓	✓	✓

† Only included if the request was triggered by a SIP message, and the parameter is included as a field in the SIP message header. Note that the `ms_subnet` and `p_asserted_identity` fields use only underscores and lower case characters in their names when used in local policy (normally they are referenced as `ms-subnet` and `p_Asserted-Identity`).

‡ Only included in outbound call requests and for breakout rooms.

†† Only included in outbound call requests.

◇ Only included in Epic telehealth calls.

For example, the `call_info` variable could contain the following data (displayed here in JSON format):

```
{
  "call_info": {
    "bandwidth": 0,
    "call_direction": "dial_in",
    "call_tag": "wxyz789",
    "local_alias": "meet.alice.vmr@example.com",
    "location": "New York",
    "node_ip": "10.55.55.101",
    "protocol": "api",
    "pseudo_version_id": "36358.0.0",
    "remote_address": "10.55.55.250",
    "remote_alias": "bob@example.com",
    "remote_display_name": "Bob T. User",
    "remote_port": 64703,
    "trigger": "web",
    "vendor": "Mozilla/5.0 (X11; Linux x86_64) Chrome/54.0.2840.100 Safari/537.36",
    "version_id": "16"
  }
}
```

Participant properties variables

The `participant` variable holds the details of the participant who is attempting to join a service. This variable only applies to scripts run as participant policy.

- i** Note that it contains the `remote_alias` and `remote_display_name` fields, which are also included in the `call_info` variable. However, the content of these fields in the `participant` variable could have been modified by external policy. The original values, as supplied by the endpoint, are contained in the `call_info` variable.

The following table shows the participant configuration variables that are available to your script. These variables contain information about the participant, however the following fields can be changed by your local policy script: `preauthenticated_role`, `remote_alias` and `remote_display_name`. This allows you to change the participant's role or anonymize their identity during the conference.

When using these variables you must use the `"participant."` prefix. For example to refer to the display name associated with the participant making the request, use `participant.remote_display_name`. (Note that the participant variables are the same as those used in external policy API requests.)

Available participant variables	Description
call_uuid	A unique identifier for the participant's call.

Available participant variables	Description
idp_attributes	<p>This contains any custom Identity Provider attributes for that participant.</p> <p>All of the attributes are available within an <code>idp_attributes</code> dictionary, for example:</p> <pre>"idp_attributes": { "group": "super-admin", "locale": "en_GB" }</pre> <p>Thus, a custom attribute of "locale" would be referenced by <code>participants.idp_attributes.locale</code></p> <p>Note that if an attribute name contains a hyphen or spaces you must reference it via <code>.get()</code>, for example <code>participant.idp_attributes.get("X-wing pilot")</code></p> <p>Custom attributes for use in participant policy are assigned to Identity Providers in the settings when configuring the Identity Provider (Users & Devices > Identity Providers > Advanced Options).</p>
participant_type	<p>The participant type: It can be:</p> <ul style="list-style-type: none"> "standard": used for most types of calls. "api": when a WebRTC client joins there are normally two requests to policy. The first is for the API participant which has the "api" type. Normally, immediately after the "api" request, a "standard" participant request is made. "api_host": this is a variation of the normal "api" type which also starts the conference if the participant is a Host.
participant_uuid	The uuid associated with the participant.
preauthenticated_role	<p>The participant's role, either "guest", "chair" or null (note that in any jinja statements you need to test for: <code>is None</code>).</p> <p>You should consider the sequence of events that could determine the role's value provided here:</p> <ul style="list-style-type: none"> For WebRTC participants, participant policy is applied after any PIN entry or SSO steps (which may determine the participant's role). For SIP/H.323 endpoints, participant policy is applied before any PIN entry steps. Local policy is applied after any external policy (which may have modified the role).
remote_alias	The name of the user or the registered alias of the endpoint. The <code>remote_alias</code> may include scheme information, for example <code>sip:alice@example.com</code> .
remote_display_name	The display name of the remote participant.

Example **participant** data (displayed here in JSON format):

```
{
  "call_uuid": "69434387-e444-411c-afd4-dcc25bf328d3",
  "idp_attributes": {
    "group": "super-admin",
    "locale": "en_GB"
  },
  "participant_type": "standard",
  "participant_uuid": "91664d87-28e4-444b-b6f1-a3816547290",
  "preauthenticated_role": "chair",
  "remote_alias": "alice.jones@example.com",
  "remote_display_name": "Alice Jones"
}
```

Response formats

The response to Pexip Infinity i.e. the output/result of the script, must be a JSON object.

The following sections explain how that response must be structured for service configuration data, participant data, and for media location data. Note that the fields in a JSON object can be supplied in any order, and that the returned data takes the same format as the responses to external policy API requests.

Service configuration data responses

The response to a service configuration request takes the basic format:

```
{
  "action": "reject|redirect|continue",
  "result": { <service_configuration_data> }
}
```

where:

- "action" instructs Pexip Infinity how to proceed:
 - "reject" instructs Pexip Infinity to reject the call (i.e. return "conference not found")
 - "redirect" instructs Pexip Infinity to send a SIP response with status "302 redirect" to the requesting endpoint, telling it to place an otherwise identical call to a different alias. The "result" field specifies the alias to be used, using the format `{"new_alias": "sip:alias@example.com"}`
 - "continue" use the data supplied in the "result" field.
- "result" is a JSON object of multiple key value pairs that describes the service:
 - Some data fields are required, and some are optional: the fields expected by Pexip Infinity in the response depend upon the returned service_type — either "conference" or "lecture" for a Virtual Meeting Room or Virtual Auditorium, "gateway" for an Infinity Gateway call, "two_stage_dialing" for a Virtual Reception, "media_playback" for a Media Playback Service, or "test_call" for a Test Call Service.
 - Note that the response configures the service and therefore each request for the same service should return the same configuration for that service. The only fields that can return a different value (per participant) are the PIN and bandwidth related fields (pin, guest_pin, max_callrate_in, max_callrate_out). You cannot change the properties of a participant calling into that service, such as their display name.

The fields expected in the response for each service type are described below:

Virtual Meeting Room / Virtual Auditorium service types response fields

Pexip Infinity expects the following fields to be returned for a service_type of "conference" (VMR) or "lecture" (Virtual Auditorium):

Field name	Required	Type	Description
name	Yes	String	The name of the service. You could, for example, use the local_alias received in the configuration request. Pexip Infinity will then use this name — as the service_name parameter — in subsequent requests to identify that service. If this configuration is defining a breakout room, then this is the name of the main VMR.
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	The type of service, in this case: <ul style="list-style-type: none">• "conference": a Virtual Meeting Room, or• "lecture": a Virtual Auditorium

Field name	Required	Type	Description
allow_guests	No	Boolean	<p>Whether to distinguish between Host and Guest participants:</p> <ul style="list-style-type: none"> true: the conference has two types of participants: Hosts and Guests. The pin to be used by Hosts must be specified. A guest_pin can optionally be specified; if a guest_pin is not specified, Guests can join without a PIN. false: all participants have Host privileges <p>Default: false</p>
automatic_participants	No	List	<p>A list of participants to dial automatically when the conference starts. Each participant in the list contains a number of fields as described in Automatically dialed participants (ADP) fields.</p>
breakout_rooms	No	Boolean	<p>Whether Breakout Rooms are enabled for the VMR. See Breakout room fields below for configuration for a breakout room.</p> <p>Default: false</p>
bypass_proxy	No	Boolean	<p>Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node.</p> <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. <p>Default: false</p>
call_type	No	String	<p>The call capability of the conference. It can be limited to:</p> <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only <p>Default: "video"</p>
crypto_mode	No	String	<p>Controls the media encryption requirements for participants connecting to this service.</p> <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) <p>Default: <null> (use global setting)</p>
description	No	String	<p>A description of the service.</p>

Field name	Required	Type	Description
direct_media	No	String	<p>Allows this VMR to use direct media between participants. When enabled, the VMR provides non-transcoded, encrypted, point-to-point calls between any two WebRTC participants. The options are:</p> <ul style="list-style-type: none"> "never": do not use direct media in this VMR. "best_effort": use direct media in this VMR where possible (when there are two WebRTC participants only), otherwise use standard, transcoded media connections via a Conferencing Node. <p>Default: "never"</p>
direct_media_notification_duration	No	Integer	<p>The number of seconds to show a notification to participants before being escalated into a transcoded call, or de-escalated into a direct media call. Range: 0 to 30 seconds.</p> <p>Default: 0 seconds</p>
enable_chat	No	String	<p>Whether chat messaging is enabled for the conference:</p> <ul style="list-style-type: none"> "default": as per the global configuration setting "yes": chat is enabled "no": chat is disabled <p>Default: "default"</p>
enable_active_speaker_indication	No	Boolean	<p>When active speaker display is enabled, the display name or alias of the current speaker is shown across the bottom of their video image. This option is not available in every layout.</p> <ul style="list-style-type: none"> true: active speaker is indicated false: active speaker is not indicated <p>Default: false</p>
enable_overlay_text	No	Boolean	<p>If participant name overlays are enabled, the display names or aliases of all participants are shown in a text overlay along the bottom of their video image.</p> <ul style="list-style-type: none"> true: participant names are shown false: participant names are not shown <p>Default: false</p>
force_presenter_into_main (applies to service_type of "lecture" only)	No	Boolean	<p>Controls whether the Host who is presenting is locked into the main video position:</p> <ul style="list-style-type: none"> true: the Host sending the presentation stream will always hold the main video position false: the main video position is voice-switched <p>Default: false</p>
guest_pin	No	String	Guest PIN — the secure access code for Guest participants.

Field name	Required	Type	Description
guest_view (applies to service_type of "lecture" only)	No	String	The layout seen by Guests: <ul style="list-style-type: none"> "one_main_zero_pips": full-screen main speaker only "one_main_seven_pips": large main speaker and up to 7 other participants "one_main_twentyone_pips": main speaker and up to 21 other participants "two_mains_twentyone_pips": 2 main speakers and up to 21 other participants "one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants "four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers "nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers "sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers "twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers "five_mains_seven_pips": Adaptive Composition layout (you must also set Host view to Adaptive Composition) "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants Default: "one_main_seven_pips" Only Hosts are displayed. Guests can hear but not see any of the other Guests.
guests_can_present	No	Boolean	Controls whether the Guests in the conference are allowed to present content. <ul style="list-style-type: none"> true: Guests and Hosts can present into the conference false: only Hosts can present Default: true
guest_identity_provider_group	No	String	The set of Identity Providers to be offered to Guests to authenticate with, in order to use the service. If this is blank, Guests are not required to authenticate.
host_identity_provider_group	No	String	The set of Identity Providers to be offered to Hosts to authenticate with, in order to use the service. If this is blank, Hosts are not required to authenticate.
host_view (applies to service_type of "lecture" only)	No	String	The layout seen by Hosts: <ul style="list-style-type: none"> "one_main_zero_pips": full-screen main speaker only "one_main_seven_pips": large main speaker and up to 7 other participants "one_main_twentyone_pips": main speaker and up to 21 other participants "two_mains_twentyone_pips": 2 main speakers and up to 21 other participants "one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants "four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers "nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers "sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers "twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers "five_mains_seven_pips": Adaptive Composition layout "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants "teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) Default: "one_main_seven_pips"

Field name	Required	Type	Description
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.
locked	No	Boolean	Whether to lock the conference on creation: <ul style="list-style-type: none"> • true: the conference will be locked on creation • false: the conference will not be locked Note that this field has no effect on the conference if it is already running. Default: false
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"> • <null>: use the global maximum call quality setting. • "sd": each participant is limited to SD quality. • "hd": each participant is limited to HD (720p) quality. • "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)
mute_all_guests (applies to service_type of "lecture" only)	No	Boolean	Controls whether to mute guests when they first join the conference: <ul style="list-style-type: none"> • true: mute Guests when they first join the conference • false: do not mute Guests when they first join the conference Default: false
non_idp_participants	No	String	Determines whether participants joining a SSO-protected service from devices other than the Connect web app (for example SIP or H.323 endpoints) are allowed to dial in to the service. <ul style="list-style-type: none"> • "disallow_all": these devices are placed in a waiting room where they must wait to be admitted by a Host. • "allow_if_trusted": these devices may join the service if they are locally registered. They must still enter a Host PIN or Guest PIN if either is required. All other devices are placed in a waiting room where they must wait to be admitted by a Host. Default: "disallow_all"
participant_limit	No	Integer	The maximum number of participants allowed to join the service.
pin	No	String	Host PIN — the secure access code for Host participants.

Field name	Required	Type	Description
prefer_ipv6	No	String	Whether to use IPv6 for SIP media when dialing out to any ADPs: <ul style="list-style-type: none"> "default": use default Pexip Infinity behavior, which is to prefer IPv4 addresses for SIP media "yes": use IPv6 addresses for SIP media "no": do not use IPv6 addresses for SIP media Default: "default"
primary_owner_email_address	No	String	The email address of the owner of the VMR.
view (applies to service_type of "conference" only)	No	String	The layout seen by all participants: <ul style="list-style-type: none"> "one_main_zero_pips": full-screen main speaker only "one_main_seven_pips": large main speaker and up to 7 other participants "one_main_twentyone_pips": main speaker and up to 21 other participants "two_mains_twentyone_pips": 2 main speakers and up to 21 other participants "one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants "four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers "nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers "sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers "twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers "five_mains_seven_pips": Adaptive Composition layout "one_main_twelve_around": large main speaker and up to 12 other participants "two_mains_eight_around": two main speakers and up to 8 other participants "teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) Default: "one_main_seven_pips"

Breakout room fields

If you want the configuration to represent a breakout room, then the following fields should also be supplied:

Field name	Required	Type	Description
breakout	Yes	Boolean	Indicates this is a breakout room. If <code>breakout</code> is true, then <code>breakout_rooms</code> should be false (you cannot create breakout rooms from within a breakout room). Also, note that <code>name</code> should be the main VMR room name (of which this is a breakout).
breakout_uuid	Yes	String	A uuid string that represents this breakout room, in the format "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX".
breakout_name	Yes	String	Name of the breakout room.

Field name	Required	Type	Description
breakout_uuids	No	List	<p>When returning the main room configuration back to a Host you can include a list of uuid strings that represents multiple breakout rooms to be created.</p> <p>This causes the Host to attempt to join each of these breakout_uuid rooms as a control only participant. This will trigger another policy lookup that includes the following fields (either as external policy parameters, or as part of the call_info structure for internal policy):</p> <pre>"service_name": "<main_room_name>_breakout_<breakout_uuid>" "unique_service_name": same as service_name "breakout_uuid": breakout_uuid of breakout room being created</pre> <p>The response to these policy lookups should be a configuration block which represents a breakout room with the fields configured appropriately for each room.</p>
breakout_description	No	String	Long name / description of the breakout room.
end_action	No	String	<p>What to do with participants when the timer expires, or the breakout is closed:</p> <ul style="list-style-type: none"> "disconnect": disconnect them all "transfer": transfer them back to the main room <p>Default: "transfer"</p>
end_time	No	Integer	<p>Time when breakout ends, in seconds since epoch. 0 = no automatic end.</p> <p>Default: 0</p>

See [Route incoming calls directly into a breakout room](#) for an example local policy script that shows how you could use these fields.

Infinity Gateway service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "gateway":

Field name	Required	Type	Description
local_alias	Yes	String	The calling or "from" alias. This is the alias that the recipient would use to return the call.
name	Yes	String	<p>The name of the service. Ensure that all gateway service instances have a unique name to avoid conflicts between calls.</p> <p>Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that call.</p>
outgoing_protocol	Yes	String	<p>The protocol to use to place the outgoing call:</p> <ul style="list-style-type: none"> "h323": an H.323 call; you can also optionally nominate an <code>h323_gatekeeper_name</code> "sip": a SIP call; you can also optionally nominate a <code>sip_proxy_name</code> "mssip": a Microsoft Skype for Business / Lync call; you can also optionally nominate an <code>mssip_proxy_name</code> and a <code>turn_server_name</code> "rtmp": uses the RTMP protocol; typically this is used for content streaming "gms": for calls to the Google Meet service. "teams": for calls to the Microsoft Teams service.

Field name	Required	Type	Description
remote_alias	Yes	String	<p>The alias of the endpoint to call.</p> <p>The alias can include scheme information, for example <code>sip:alice@example.com</code>, but the call will always be made using the specified <code>outgoing_protocol</code>.</p>
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	<p>The type of service, in this case:</p> <ul style="list-style-type: none"> "gateway": an Infinity Gateway call
bypass_proxy	No	Boolean	<p>Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node.</p> <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. <p>Default: false</p>
call_type	No	String	<p>The call capability of the outbound call:</p> <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only "auto": match the capability of the outbound call to that of the inbound call <p>Default: "video"</p>
crypto_mode	No	String	<p>Controls the media encryption requirements for participants connecting to this service.</p> <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) <p>Default: <null> (use global setting)</p>

Field name	Required	Type	Description
called_device_type	No	String	<p>The device or system to which to route the call:</p> <ul style="list-style-type: none"> "external": route the call to a matching registered device if it is currently registered, otherwise attempt to route the call via an external system "registration": route the call to a matching registered device only (providing it is currently registered) "mssip_conference_id": route the call via a Skype for Business / Lync server to a Sfb/Lync meeting where the remote_alias is a Sfb/Lync meeting Conference ID "mssip_server": route the call via a Skype for Business / Lync server to a Sfb/Lync client or meeting "gms_conference": for calls to the Google Meet service. "teams_conference": for calls to the Microsoft Teams service. "teams_user": for direct calls to a Microsoft Teams Room. "telehealth_profile": for calls to an Epic telehealth system. <p>Default: "external"</p>
denoise_audio	No	Boolean	<p>Applies to Google Meet integrations only. Controls whether to remove background noise from audio streams as they pass through the infrastructure.</p> <p>Default: true</p>
description	No	String	A description of the service.
enable_active_speaker_indication	No	Boolean	<p>When active speaker display is enabled, the display name or alias of the current speaker is shown across the bottom of their video image. This option is not available in every layout.</p> <ul style="list-style-type: none"> true: active speaker is indicated false: active speaker is not indicated <p>Default: false</p>
enable_overlay_text	No	Boolean	<p>If participant name overlays are enabled, the display names or aliases of all participants are shown in a text overlay along the bottom of their video image.</p> <ul style="list-style-type: none"> true: participant names are shown false: participant names are not shown <p>Default: false</p>
external_participant_avatar_lookup	No	String	<p>Applies to Microsoft Teams integrations only. This determines whether or not the Teams Connector requests from Exchange Online an avatar for each participant in the Teams conference.</p> <ul style="list-style-type: none"> "default": use the global external participant avatar lookup setting. "yes": request the participant avatar from Exchange Online via the Teams Connector. "no": use default avatar behavior. <p>Default: "default" (use global setting)</p>
gms_access_token_name	No	String	<p>The name of the access token to use to resolve Google Meet IDs. You should select either a trusted or untrusted type of token, depending on whether you want to enable the device to be automatically admitted into the Google Meet conference (subject to also being a trusted endpoint from Pexip Infinity's perspective i.e. if the rule also has <code>treat_as_trusted</code> enabled).</p>

Field name	Required	Type	Description
h323_gatekeeper_name	No	String	The name* of the H.323 gatekeeper to use to place the outgoing call. DNS is used if no gatekeeper is specified.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)
mssip_proxy_name	No	String	The name* of the Skype for Business / Lync server to use to place the outgoing call. DNS is used if no server is specified.
outgoing_location_name	No	String	The name* of the location of a Conferencing Node from which to place the call.
prefer_ipv6	No	String	Whether to use IPv6 for SIP media for the outgoing call: <ul style="list-style-type: none"> "default": use default Pexip Infinity behavior, which is to prefer IPv4 addresses for SIP media "yes": use IPv6 addresses for SIP media "no": do not use IPv6 addresses for SIP media Default: "default"
sip_proxy_name	No	String	The name* of the SIP proxy to use to place the outgoing call. DNS is used if no proxy is specified.
stun_server_name	No	String	The name* of the STUN server to use when placing calls to external SIP and WebRTC endpoints that are ICE-enabled (such as Skype for Business / Lync clients and Connect app WebRTC clients).
teams_proxy_name	No	String	The name of the Teams Connector to handle the call to Microsoft Teams. If you do not specify anything, the Teams Connector associated with the outgoing location is used.
treat_as_trusted	No	Boolean	This indicates that the target of this Call Routing Rule may treat the caller as part of the target organization for trust purposes.
turn_server_name	No	String	The name* of the TURN server to offer to external SIP and WebRTC endpoints that are ICE-enabled (such as Skype for Business / Lync clients and Connect app WebRTC clients).

Field name	Required	Type	Description
view	No	String	<p>The layout seen by Pexip participants:</p> <ul style="list-style-type: none">"one_main_zero_pips": full-screen main speaker only"one_main_seven_pips": large main speaker and up to 7 other participants"one_main_twentyone_pips": main speaker and up to 21 other participants"two_mains_twentyone_pips": 2 main speakers and up to 21 other participants"one_main_thirtythree_pips": 1 small main speaker and up to 33 other participants"four_mains_zero_pips": 2 x 2 layout, up to a maximum of 4 speakers"nine_mains_zero_pips": 3 x 3 layout, up to a maximum of 9 speakers"sixteen_mains_zero_pips": 4 x 4 layout, up to a maximum of 16 speakers"twentyfive_mains_zero_pips": 5 x 5 layout, up to a maximum of 25 speakers"five_mains_seven_pips": Adaptive Composition layout"one_main_twelve_around": large main speaker and up to 12 other participants"two_mains_eight_around": two main speakers and up to 8 other participants"teams": Teams-like layout (this is only suitable for use with Microsoft Teams gateway calls) <p>Default: "one_main_seven_pips"</p>

* The returned "name" must match the name of the location, H.323 gatekeeper, SIP proxy etc. configured within Pexip Infinity.

Virtual Reception service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "two_stage_dialing" (Virtual Reception):

Field name	Required	Type	Description
name	Yes	String	<p>The name of the service. You could, for example, use the <code>local_alias</code> received in the configuration request.</p> <p>Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that service.</p>
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	<p>The type of service, in this case:</p> <ul style="list-style-type: none"> "two_stage_dialing": a Virtual Reception
bypass_proxy	No	Boolean	<p>Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node.</p> <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. <p>Default: false</p>
call_type	No	String	<p>The call capability of the reception. It can be limited to:</p> <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only <p>Default: "video"</p>
crypto_mode	No	String	<p>Controls the media encryption requirements for participants connecting to this service.</p> <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) <p>Default: <null> (use global setting)</p>
description	No	String	A description of the service.
gms_access_token_name	No	String	The name of the access token to use to resolve Google Meet IDs. When configuring a Virtual Reception it does not matter if you use a trusted or untrusted access token.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.

Field name	Required	Type	Description
match_string	No	String	<p>An optional regular expression used to match against the alias entered by the caller into the Virtual Reception. If the entered alias does not match the expression, the Virtual Reception will not route the call.</p> <p>If this field is left blank, any entered alias is permitted.</p>
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	<p>Controls the maximum call quality for participants connecting to this service:</p> <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. <p>Default: <null> (use global setting)</p>
mssip_proxy_name	No	String	The name of the Skype for Business / Lync server to use to resolve the SfB/Lync Conference ID entered by the user in the Virtual Reception.
post_match_string	No	String	<p>An optional regular expression used to match against the meeting code entered by the caller into the Virtual Reception. This is typically used in conjunction with the post_replace_string to transform the meeting code into a distinct alias pattern that will match a Call Routing Rule configured to route calls into external conferences.</p> <p>For example, you would typically set the regex match to <code>(.*)</code> (to match everything) and the replace string pattern to something like <code>meet.\1</code> which would prefix the meeting code entered into the Virtual Reception with "meet.". You would then configure an associated Call Routing Rule to match calls placed to aliases prefixed with <code>meet.</code> which then strips off that prefix (to leave just the meeting code again) before directing the call to the external conference.</p>
post_replace_string	No	String	An optional regular expression used in conjunction with the post_match_string field to transform the meeting code into a distinct alias pattern that will match a Call Routing Rule configured to route calls into external conferences. (Only applies if the post-lookup regex match string is also configured and the entered code matches that regex.)
replace_string	No	String	<p>An optional regular expression used to transform the alias entered by the caller into the Virtual Reception. (Only applies if a regex match string is also configured and the entered alias matches that regex.)</p> <p>Leave this field blank if you do not want to change the alias entered by the caller.</p>
system_location_name	No	String	This is an optional field used in conjunction with the two_stage_dial_type setting, when a type other than <i>regular</i> is selected. If specified, a Conferencing Node in this system location will perform the SfB/Lync Conference ID lookup on the SfB/Lync server, or the Microsoft Teams or Google Meet code lookup, as appropriate. We recommend that a location is specified here, otherwise the transcoding node hosting the Virtual Reception will perform the lookup (which may lead to routability issues).

Field name	Required	Type	Description
teams_proxy_name	No	String	The name of the Teams Connector to use to resolve Microsoft Teams meeting codes entered in a Virtual Reception. If you do not specify anything, the Teams Connector associated with the outgoing location is used.
two_stage_dial_type	No	String	<p>The type of Virtual Reception:</p> <ul style="list-style-type: none">"regular": the default type of Virtual Reception, used for routing calls to VMRs, or to other devices and call control systems via the Infinity Gateway."mssip": a special type of Virtual Reception, used when you want to provide an IVR gateway to scheduled and ad hoc Skype for Business / Lync meetings."gms": a special type of Virtual Reception, used when you want to provide an IVR gateway to scheduled and ad hoc Google Meet meetings."teams": a special type of Virtual Reception, used when you want to provide an IVR gateway to scheduled and ad hoc Microsoft Teams meetings. <p>Default: "regular"</p>

Media Playback Service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "media_playback" (Media Playback Service):

Field name	Required	Type	Description
name	Yes	String	The name of the service. You could, for example, use the <code>local_alias</code> received in the configuration request. Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that service.
media_playlist_name	Yes	String	The name of the media playlist.
service_type	Yes	String	The type of service, in this case: <ul style="list-style-type: none"> "media_playback": a Media Playback Service
service_tag	Yes	String	A unique identifier used to track usage of this service.
allow_guests	No	Boolean	Whether to distinguish between Host and Guest participants: <ul style="list-style-type: none"> true: the conference has two types of participants: Hosts and Guests. The <code>pin</code> to be used by Hosts must be specified. A <code>guest_pin</code> can optionally be specified; if a <code>guest_pin</code> is not specified, Guests can join without a PIN. false: all participants have Host privileges Default: false
description	No	String	A description of the service.
guest_pin	No	String	Guest PIN — the secure access code for Guest participants.
guest_identity_provider_group	No	String	The set of Identity Providers to be offered to Guests to authenticate with, in order to use the service. If this is blank, Guests are not required to authenticate.
host_identity_provider_group	No	String	The set of Identity Providers to be offered to Hosts to authenticate with, in order to use the service. If this is blank, Hosts are not required to authenticate.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_callrate_out	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will send to each individual participant dialed in to the service. Range 128000 to 8192000 bps.
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"> <null>: use the global maximum call quality setting. "sd": each participant is limited to SD quality. "hd": each participant is limited to HD (720p) quality. "fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)

Field name	Required	Type	Description
non_idp_participants	No	String	<p>Determines whether participants joining a SSO-protected service from devices other than the Connect web app (for example SIP or H.323 endpoints) are allowed to dial in to the service.</p> <ul style="list-style-type: none">"disallow_all": these devices are placed in a waiting room where they must wait to be admitted by a Host."allow_if_trusted": these devices may join the service if they are locally registered. They must still enter a Host PIN or Guest PIN if either is required. All other devices are placed in a waiting room where they must wait to be admitted by a Host. <p>Default: "disallow_all"</p>
on_completion	No	String	<p>Action to perform on playlist completion. This is defined as a JSON object.</p> <p>To disconnect the user, use the syntax:</p> <pre>"on_completion": { "disconnect": true }</pre> <p>To perform a transfer, use the syntax:</p> <pre>"on_completion": { "transfer": { "conference": "<alias>" }}</pre> <p>or (if you also want to specify the role):</p> <pre>"on_completion": { "transfer": { "conference": "<alias>", "role": "<role>" }}</pre>
pin	No	String	Host PIN — the secure access code for Host participants.

Test Call Service type response fields

Pexip Infinity expects the following fields to be returned for a `service_type` of "test_call" (Test Call Service):

Field name	Required	Type	Description
name	Yes	String	<p>The name of the service. You could, for example, use the <code>local_alias</code> received in the configuration request.</p> <p>Pexip Infinity will then use this name — as the <code>service_name</code> parameter — in subsequent requests to identify that service.</p>
service_tag	Yes	String	A unique identifier used to track usage of this service.
service_type	Yes	String	<p>The type of service, in this case:</p> <ul style="list-style-type: none"> "test_call": a Test Call Service
bypass_proxy	No	Boolean	<p>Whether to bypass a Proxying Edge Node and send media directly to a Transcoding Conferencing Node.</p> <ul style="list-style-type: none"> true: if the call signaling is received by a Proxying Edge Node, that proxying node will determine which Transcoding Conferencing Node should handle the call media (as per standard media allocation rules), but it then instructs the client to send its media directly to that nominated transcoding node, thus bypassing the proxying node. The proxying node will continue to handle the call signaling. false: standard behavior — normal proxying and media allocation rules apply. <p>Default: false</p>
call_type	No	String	<p>The call capability of the service. It can be limited to:</p> <ul style="list-style-type: none"> "video": main video plus presentation "video-only": main video only "audio": audio-only <p>Default: "video"</p>
crypto_mode	No	String	<p>Controls the media encryption requirements for participants connecting to this service.</p> <ul style="list-style-type: none"> <null>: use the global media encryption setting. "besteffort": each participant will use media encryption if their device supports it, otherwise the connection will be unencrypted. "on": all participants (including RTMP participants) must use media encryption. "off": all H.323, SIP and MS-SIP participants must use unencrypted media. (RTMP participants will use encryption if their device supports it, otherwise the connection will be unencrypted.) <p>Default: <null> (use global setting)</p>
description	No	String	A description of the service.
ivr_theme_name	No	String	The name of the theme to use with the service. If no theme is specified, the default Pexip theme is used.
local_display_name	No	String	The display name of the calling alias.
max_callrate_in	No	Integer	The maximum media bandwidth in bps that Pexip Infinity will receive from each individual participant dialed in to the service. Range 128000 to 8192000 bps.

Field name	Required	Type	Description
max_pixels_per_second	No	String	Controls the maximum call quality for participants connecting to this service: <ul style="list-style-type: none"><null>: use the global maximum call quality setting."sd": each participant is limited to SD quality."hd": each participant is limited to HD (720p) quality."fullhd": allows any endpoint capable of Full HD to send and receive its main video at 1080p. Default: <null> (use global setting)

Automatically dialed participants (ADP) fields

The `automatic_participants` list field, which may be included in the service configuration response for a `service_type` of "conference" (VMR) or "lecture" (Virtual Auditorium), contains the following fields per participant:

Field name	Required	Type	Description
local_alias	Yes	String	The calling or "from" alias. This is the alias that the recipient would use to return the call.
protocol	Yes	String	The protocol to use to place the outgoing call: <ul style="list-style-type: none">"h323": an H.323 call"sip": a SIP call"mssip": a Microsoft Skype for Business / Lync call"rtmp": uses the RTMP protocol; typically this is used for content streaming Calls are routed to externally-located participants based on the configuration of the system location used to place the call.
remote_alias	Yes	String	The alias of the endpoint to call.
role	Yes	String	The level of privileges the participant has in the conference: <ul style="list-style-type: none">"chair": the participant has Host privileges"guest": the participant has Guest privileges
call_type	No	String	The call capability of the participant. It can be limited to: <ul style="list-style-type: none">"video": main video plus presentation"video-only": main video only"audio": audio-only Default: "video"
description	No	String	An optional description of the Automatically Dialed Participant. Note that the description is for information purposes only. It has no subsequent effect in call processing.
dtmf_sequence	No	String	An optional DTMF sequence to transmit after the call to the dialed participant starts.

Field name	Required	Type	Description
keep_conference_alive	No	String	<p>Determines whether the conference continues when all other non-ADP participants have disconnected:</p> <ul style="list-style-type: none"> "keep_conference_alive": the conference continues to run until this participant disconnects (applies to Hosts only). "keep_conference_alive_if_multiple": the conference continues to run as long as there are two or more "keep_conference_alive_if_multiple" participants and at least one of them is a Host. "keep_conference_alive_never": the conference terminates automatically if this is the only remaining participant. <p>Default: "keep_conference_alive_if_multiple"</p> <p>For more information, see https://docs.pexip.com/admin/automatically_terminate.htm.</p>
local_display_name	No	String	The display name of the calling or "from" alias.
presentation_url	No	String	This additional parameter can be specified for RTMP calls to send the presentation stream to a separate RTMP destination.
remote_display_name	No	String	An optional friendly name for this participant. This may be used instead of the participant's alias in participant lists and as a text overlay in some layout configurations.
routing	No	String	<p>Specifies how to route the call:</p> <ul style="list-style-type: none"> "manual": uses the requested protocol and the defaults for the specified <code>system_location_name</code>. "routing_rule": routes the call according to the configured Call Routing Rules. This means that the dialed alias must match an outgoing Call Routing Rule for the call to be placed (using the protocols, outgoing location and call control systems etc. as configured for that rule). <p>Default: "manual"</p>
streaming	No	Boolean	<p>Identifies the dialed participant as a streaming or recording device.</p> <ul style="list-style-type: none"> true: the participant is a streaming or recording device false: the participant is not a streaming or recording device <p>Default: false</p>
system_location_name	No	String	The location of the Conferencing Node from which to place the call.

Example service configuration data responses

Response containing service configuration data

This is an example response of service configuration data for a "conference" service type:

```
{
  "action": "continue",
  "result": {
    "service_type": "conference",
    "name": "Alice Jones",
    "service_tag": "abcd1234",
    "description": "Alice Jones personal VMR",
    "pin": "1234",
    "allow_guests": true,
    "guest_pin": "5678",
    "view": "one_main_zero_pips",
    "enable_overlay_text": true,
    "automatic_participants":
    [
      {
        "remote_alias": "sip:alice@example.com",
        "remote_display_name": "Alice",
        "local_alias": "meet.alice@example.com",
        "local_display_name": "Alice's VMR",
        "protocol": "sip",
        "role": "chair",
        "system_location_name": "London"
      },
      {
        "remote_alias": "rtmp://example.com/live/alice_vmr",
        "local_alias": "meet.alice@example.com",
        "local_display_name": "Alice's VMR",
        "protocol": "rtmp",
        "role": "guest",
        "streaming": true
      }
    ]
  }
}
```

Response instructing Pexip Infinity to reject the call

This is an example response that tells Pexip Infinity to reject a call.

```
{
  "action": "reject"
}
```

Participant configuration responses

The response to a participant properties request takes the basic format:

```
{
  "action": "reject|continue",
  "result": { <participant_configuration_data> }
}
```

where:

- "action" instructs Pexip Infinity how to proceed:
 - "reject" instructs Pexip Infinity to reject the call (i.e. return "conference not found").
 - "continue" use the data supplied in the "result" field.
- "result" is a JSON object of key value pairs that override the participant properties provided in the original participant information. The fields that may be included are listed below. All data fields are optional.

Field name	Required	Description
preauthenticated_role	No	<p>An override for the participant's role. This must be either "chair" or "guest", or null if the originally supplied value was also null.</p> <p>For SIP and H.323 participants the original value will be null as participant policy is applied before any PIN entry (unlike WebRTC). Thus, returning null in the response means the caller will still be prompted for a PIN, whereas chair/guest will behave as if the user had entered the PIN for the specified role.</p>
remote_alias	No	<p>An override for the name of the user or the alias of the endpoint.</p> <p>Note that the original caller alias/name will still be recorded in the Pexip Infinity call logs when the participant first tries to connect to a conference.</p>
remote_display_name	No	<p>An override for the display name of the participant. It may return null if the originally supplied value was also null.</p> <p>Note that the original caller display name will still be recorded in the Pexip Infinity call logs when the participant first tries to connect to a conference.</p>

Example participant properties data responses

Response containing participant properties

This is an example response of participant properties:

```
{
  "action": "continue",
  "result": {
    "preauthenticated_role": "guest",
    "remote_alias": "Witness A alias",
    "remote_display_name": "Witness A"
  }
}
```

Response instructing Pexip Infinity to reject the call

This is an example response that tells Pexip Infinity to reject a call.

```
{
  "action": "reject"
}
```

Media location data responses

The response to a media location request takes the format:

```
{
  "result": { <location_data> }
}
```

where "result" is a JSON object of key value pairs that describes the media location to use. The fields that may be included are:

Field name	Required	Description
location	Yes	The name* of the principal location to use for media allocation. A Conferencing Node assigned to that location will handle the media for the participant.

Field name	Required	Description
overflow_locations	No	<p>A list of one or more system location names* to handle the media if the principal location has reached its capacity. The order of the list defines the order that the locations are used.</p> <p>As of version 23.2, we recommend that you use overflow_locations instead of the primary_overflow_location and secondary_overflow_location fields. However, to ensure backwards compatibility with any previously-configured policy, any overflow locations specified here will be appended to the locations specified in the primary_overflow_location and secondary_overflow_location fields if they are also nominated.</p>
primary_overflow_location	No	The name* of the system location to handle the media if the principal location has reached its capacity. As of version 23.2 we recommend that you use overflow_locations instead.
secondary_overflow_location	No	The name* of the system location to handle the media if both the principal location and the primary overflow location have reached their capacity. As of version 23.2 we recommend that you use overflow_locations instead.

* The returned "name" must match the name of a location configured within Pexip Infinity. If any of the location names included in the response do not match a configured location within Pexip Infinity, the entire response is deemed to have failed and Pexip Infinity will fall back to its own default behavior for that request.

Example media location data response

This is an example response containing media location data:

```
{
  {% if call_info.location == "Oslo" %}
    "result": {
      "location": "Oslo",
      "overflow_locations": ["London", "Paris", "New York"]
    }
  {% else %}
    "result": {
      "location": "New York",
      "overflow_locations": ["London", "Oslo"]
    }
  {% endif %}
}
```

This script tests if the system location handling the incoming call request is "Oslo" and if so sets the location to use for media to "Oslo", and sets "London", then "Paris" and finally "New York" as the overflow locations to use if "Oslo" is out of capacity. For all other system locations handling incoming calls, the location to use for media is "New York", with "London" and then "Oslo" as the overflow locations.

Dialing out from conference

Pexip Infinity makes a service configuration request if it dials out from a conference to invite a participant to join (where the `call_direction` parameter will be `dial_out`). In these cases, the response to the service configuration request must match the existing service data (i.e. the same `name`, `service_type` and so on).

When dialing out, the only configuration you can control via policy is:

- The **prefer_ipv6** setting in the service configuration response.
- The media location — you can do this in the response to the media location request that follows the service configuration request.

Using filters in local policy scripts

Local policy manipulates service configuration and media location data by running a jinja2 script against that data.

To assist you in manipulating this data Pexip Infinity supports a subset of filters from the jinja2 templating language and also provides some custom Pexip filters as described below.

Supported jinja2 filters

Pexip Infinity supports a subset of filters from the jinja2 templating language. Any jinja filters that are not listed below have been disabled in Pexip Infinity. See [List of Builtin Filters](#) for more information about these filters.

abs	float	last	replace	truncate
capitalize	format	length	round	upper
default	int	lower	striptags	
first	join	range	trim	

To use a filter you would typically follow the syntax `{{<source_value>|<filter_name>}}`.

In most cases the `<source_value>` is likely to be a variable, for example `{{ call_info.local_alias|replace("@example.com", "") }}`.

Custom Pexip filters

In addition to the jinja filters, Pexip also provides the following custom filters, which are typically used to manipulate data:

Filter	Description and example usage
pex_base64	Performs Base64 encoding on the input field.
pex_clean_phone_number	This extracts only +0123456789 characters (and removes ()&%#@ '";:, A-Z,a-z etc).
pex_debug_log (message)	<p>The <code>pex_debug_log</code> filter can be used to help debug your script. It writes debug messages to the Pexip Infinity support log. You can include literal text and variables.</p> <p>i To avoid filling the support log and causing it to rotate, remove all <code>pex_debug_log</code> filters from your scripts as soon as they are working correctly.</p> <p>Example usage: <code>{{pex_debug_log("Media location policy ", call_info.location, " protocol", call_info.protocol) }}</code></p> <p>This will generate a support log entry in the style of: Name="support.jinja2" pex_debug_log Detail="Media location policy ,Oslo, protocol,sip"</p>
pex_find_first_match(string_list, 'find_regex')	This extracts from the list the first value that matches the specified regex.
pex_hash	Performs a hash of a field.
pex_head (maxlength)	Returns, at most, the first maxlength characters from the input field.

Filter	Description and example usage
pex_in_subnet	<p>Tests whether a given address is within one or more subnets. It takes as input the address you want to test, and one or more subnet ranges, and returns either True or False.</p> <p>This filter could be useful if, for example, you want to place media in a particular location based upon the network location of the participant. For example:</p> <ul style="list-style-type: none"> <code>{{pex_in_subnet("10.47.0.1", ["10.47.0.0/16", "10.147.0.0/16"])}}</code> returns True <code>{{pex_in_subnet("10.44.0.1", ["10.47.0.0/16", "10.147.0.0/16"])}}</code> returns False <code>{{pex_in_subnet("2001:658:22a:cafe:ffff:ffff:ffff:ffff", ["2001::/16", "2002::/16"])}}</code> returns True <code>{{pex_in_subnet("2000:758:23a:beef:ffff:ffff:ffff:ffff", ["2001::/16", "2002::/16"])}}</code> returns False <p>For practical usage you will most likely want to refer to the calling participant's address (<code>call_info.remote_address</code>) instead of literally specifying the address to be tested, for example: <code>{{pex_in_subnet(call_info.remote_address, ["10.44.0.0/16"])}}</code></p> <p>See Test whether a participant's address is within a particular subnet for a more advanced example that shows how you can make multiple tests against multiple locations with multiple subnets.</p>
pex_md5	Applies an MD5 hash to the input field.
pex_now (timezone)	<p>The <code>pex_now</code> filter takes an optional parameter of a timezone description e.g. 'UTC', 'Asia/Tokyo', or 'US/Eastern' and returns the current date and time for that timezone. UTC is assumed if a timezone is not provided.</p> <p>The resulting available attributes are <code>year</code>, <code>month</code>, <code>day</code>, <code>hour</code>, <code>minute</code>, <code>second</code> and <code>microsecond</code>.</p> <p>Example usage:</p> <pre>{% set now = pex_now("Europe/London") %} {% if now.month == 2 and now.day == 29 %}</pre> <p>See Use a different theme based on time of day for a full example that shows how to make decisions based on the time of day.</p>
pex_random_ pin(length)	Generates a random PIN of the given length. Note that this filter does not take any input.
pex_regex_ replace('find_ regex', 'replace_string')	<p>This performs a regex find and replace.</p> <p>Example usage: <code>{{mail pex_regex_replace('@.+','@otherdomain.com')}}</code></p> <p>This example takes as input an email address contained in the <code>mail</code> variable and changes the domain portion of the address to <code>@otherdomain.com</code>. For example, it will transform <code>user1@domainA.com</code> to <code>user1@otherdomain.com</code>, and <code>user2@domainB.com</code> to <code>user2@otherdomain.com</code> etc.</p>
pex_regex_ search('regex pattern', 'string_to_ search')	<p>This performs a regex search for the first location that matches the pattern and returns the regex groups.</p> <p>Example usage:</p> <pre>{% set groups = pex_regex_search("([a-z0-9.-]+)@([a-z0-9.-]+.com)", "example string with someone@example.com") %} {% if groups %} {{ groups[0] }}@{{ groups[1] }} {% endif %}</pre> <p>This example takes as input a string containing an email address and extracts the email using two regex groups.</p>
pex_require_ min_length (length)	<p>This validates that the input string field has the specified minimum length.</p> <p>Syntax: <code>{{ some_string pex_require_min_length(2) }}</code></p> <p>When used in local policy, if the input string fails the minimum length requirement, the policy is not applied and the original configuration data is retained unchanged.</p>
pex_reverse	This reverses the characters in the input field.

Filter	Description and example usage
pex_strlen	<p>Returns the length of string. The basic usage syntax is:</p> <pre>{% set some_length = "Example" pex_strlen %} {# sets a variable named some_length to the value 7 #}</pre>
pex_tail (maxlength)	<p>Returns, at most, the last maxlength characters from the input field.</p>
pex_to_json	<p>Converts a Python dictionary variable into JSON format.</p> <p>It can be used to convert the data in the <code>service_config</code> and <code>suggested_media_overflow_locations</code> variables (which are Python dictionaries) into JSON format (which is the required data structure for the configuration data returned to Pexip Infinity) when you want to use those variables in a response <code>result</code> object.</p> <p>Example usage: <code>{{suggested_media_overflow_locations pex_to_json}}</code></p> <p>The example above converts the <code>suggested_media_overflow_locations</code> variable into JSON format.</p> <p>Example usage: <code>{{service_config pex_update({"participant_limit":10, "ivr_theme_name":"funky"}) pex_to_json}}</code></p> <p>This second example combines the <code>pex_update</code> filter with the <code>pex_to_json</code> filter. It makes updates to the <code>service_config</code> variable and then converts it to JSON.</p> <p>Note that you do not need to use the <code>pex_to_json</code> filter if you are constructing the response <code>result</code> object directly in JSON format (e.g. as shown in the Example service configuration data response).</p>
pex_to_uuid	<p>Converts a base64 string to a UUID.</p>
pex_update	<p>Updates Python dictionary variables.</p> <p>It can be used to update the <code>service_config</code> (service configuration data) and <code>suggested_media_overflow_locations</code> (media location data) variables.</p> <p>Example usage: <code>{{service_config pex_update({"pin":"1234", "guest_pin":"", "allow_guests" : false})}}</code></p> <p>This updates 3 of the data elements in the <code>service_config</code> variable. It sets the <code>pin</code> to 1234, the <code>guest_pin</code> to null and the <code>allow_guests</code> flag to false. (See Service configuration data responses for a list of all of the elements in the <code>service_config</code> variable, and Media location data responses for a list of all of the elements in the <code>suggested_media_overflow_locations</code> variable.)</p>
pex_url_decode	<p>This filter decodes a previously URL-encoded string. For example, it can be used with OTJ custom rules to simplify the regular expressions required.</p>
pex_url_encode	<p>This filter creates URL parameters that are safely URL-encoded.</p>
pex_urldefense_decode	<p>This filter converts a string that has been rewritten by Proofpoint's URL Defense into the original URL.</p>
pex_uuid4()	<p>This generates a uuid (universally unique identifier). Note that this filter does not take any input.</p>


Testing local policy scripts

To help you construct and test your scripts, Pexip Infinity has a built-in script testing facility. It allows you to provide some test configuration data, run your script against that test data and check the output produced by your script.

To test your scripts:

1. Go to **Call Control > Policy Profiles** and select the policy profile containing the script you want to test.
(Note that you must select an existing profile.)
2. Select **Test local service configuration policy**, **Test local participant policy** or **Test local media location policy** as appropriate (at the bottom of the page).
3. You are taken to a script testing page where you can provide test input data and edit your script:
 - The **Input** field contains some example **call_info** and either some **service_config** data (for service configuration scripts), some **participant** data (for participant scripts), or some **suggested_media_overflow_locations** data (for media location scripts), in JSON format, that you can use as input data to test your script. You can use this data as is, or change the data to suit your needs, by adding or removing data elements. Ensure that you use valid data elements and maintain the correct JSON formatting.
 - The **Script** field is initially populated with your current script from your policy profile. You can edit the script as required.
 - The **Result** field shows the result of running the input data against your script. If there is an error when running the script this field will contain a "reject" action.
 - The **Diagnostics** field contains any additional information (if available) pertaining to errors arising from running the test script against the test input.
4. To test your script, edit the **Input** and **Script** fields as required and then select **Test** option (at the bottom of the page).
5. Check the **Result** and **Diagnostic** fields to see the effect of your script.
6. You can continue to edit the **Input** and **Script** fields and test your script until you are happy that the script is performing as expected.
7. When you have finished testing your script:
 - Select **Save changes and return** to save the current contents of your script to your policy profile.
 - Select **Cancel** to return to your policy profile without saving the changes to your script.

Checking call information (call_info)

-  The contents of the **call_info** variable can vary within your Pexip Infinity system depending on the types of devices and call protocols in use. A good way to identify what information is actually being presented to your script is to include a **pex_debug_log** filter such as:

```
{{pex_debug_log("Call information ", call_info) }}
```

You can then make some test calls and look in the support log to see the actual call information that is being passed through to your script. To see the call information data go to **History & Logs > Support Log** and then search for entries that contain **pex_debug_log**. See [Basic pass-through service configuration script with call_info debug line](#) for a full example script.

Example local policy scripts

Here are some example local policy scripts that illustrate how to structure a script, manipulate variables and format the data responses.

You can use and adapt these scripts as appropriate for your own environment. See [Using filters in local policy scripts](#) for more information about the filters used in these examples.

Minimum basic pass-through service configuration script

This is the smallest "no changes" service configuration script that you can use. It will allow a call to continue if it was going to be allowed anyway — and reject it if it was going to be rejected.

We recommend that you use this as the basis for your own scripts and add extra logic to get your desired functionality.

```
{
  {% if service_config %}
    "action" : "continue",
    "result" : {{service_config|pex_to_json}}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Minimum basic pass-through media location script

This is the smallest "no changes" media location script that you can use. It simply returns the original set of suggested_media_overflow_locations without modification.

```
{
  "result" : {{suggested_media_overflow_locations|pex_to_json}}
}
```

Basic pass-through service configuration script with call_info debug line

This script extends the minimum "no impact" service configuration script above by adding a `{{pex_debug_log("Call information ", call_info) }}` line.

When this script runs it will not change any configuration data but it will record the contents of the `call_info` variable — all of the information relating to the call being processed — to the support log. This allows you to analyze the nature of the call information for different call types and help you construct your script appropriately. To see the call information data go to **History & Logs > Support Log** and then search for entries that contain `pex_debug_log`.

i To avoid filling the support log and causing it to rotate, remove all `pex_debug_log` filters from your scripts as soon as they are working correctly.

```
{
  {{pex_debug_log("Call information ", call_info) }}
  {% if service_config %}
    "action" : "continue",
    "result" : {{service_config|pex_to_json}}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Unconditionally nominate media locations

This media location script sets the location to be used for media to "Oslo", and sets "London" and "New York" as the primary and secondary overflow locations if "Oslo" is out of capacity. These locations will be used for every call / media location request, regardless of where call signaling is received.

```
{
  "result" : {
    "location" : "Oslo",
    "primary_overflow_location" : "London",
    "secondary_overflow_location" : "New York"
  }
}
```

Nominate a media location for RTMP streaming

This media location script explicitly sets the location to use for media to "DMZ" if the call being made is an outbound RTMP call (and also overrides any original primary and secondary overflow locations). All other call types will continue to use the original media locations.

```
{
  {% if
    call_info.protocol == "rtmp" and
    call_info.call_direction == "dial_out"
  %}
  "result" : {
    "location" : "DMZ",
    "primary_overflow_location" : "Location B",
    "secondary_overflow_location" : "Location C"
  }
  {% else %}
  "result" : {{suggested_media_overflow_locations|pex_to_json}}
  {% endif %}
}
```

Remove PIN based on location

This service configuration script removes the PIN for participants in "Location A" or "Location C" (these might, for example, be "internal" locations) but keeps any existing PIN requirements for participants in other locations.

```
{
  {% if service_config %}
  "action" : "continue",
  {% if call_info.location == "Location A" or call_info.location == "Location C" %}
  "result" : {{service_config|pex_update({"pin":"","guest_pin":"","allow_guests" : False})|pex_to_json}}
  {% else %}
  "result" : {{service_config|pex_to_json}}
  {% endif %}
  {% else %}
  "action" : "reject",
  "result" : {}
  {% endif %}
}
```

Remove PIN if device is registered

This service configuration script removes the PIN for participants if their device is registered to a Conferencing Node, but keeps any existing PIN requirements for participants in other locations.

```
{
  {% if service_config %}
  "action" : "continue",
  {% if call_info.registered %}
  "result" : {{service_config|pex_update({"pin":"","guest_pin":"","allow_guests" : False})|pex_to_json}}
  {% else %}
  "result" : {{service_config|pex_to_json}}
  {% endif %}
  {% else %}
  "action" : "reject",
  "result" : {}
  {% endif %}
}
```

Inject an ADP into every conference (with a debug line for service_config)

This service configuration script adds an Automatically Dialed Participant (ADP) into every conference. Note that this script will override any existing ADPs that may be present in the `service_config` variable.

This example script also demonstrates how to use the `pex_debug_log` filter. It writes the original contents of the `service_config` variable to the support log.

i To avoid filling the support log and causing it to rotate, remove all `pex_debug_log` filters from your scripts as soon as they are working correctly.

```
{
  {% if service_config %}
    {{pex_debug_log("service_config=", service_config) }}
    "action" : "continue",
    "result" : {{service_config|pex_update({"automatic_participants" : [{"remote_alias":"participant@domain.com","local_
alias":"policyuser@domain.com","local_display_name":"Local policy ADP","description":"Dial out to an
ADP","protocol":"sip","role":"chair","system_location_name":"Location A" }]}|pex_to_json)}}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Reject specified User Agents

This service configuration script rejects calls from (potentially) malicious User Agents such as "sipvicious" (User Agent "friendly-scanner"). Any calls from User Agents that are in the `suspect_uas` list will be rejected. You can edit the `suspect_uas` as required for your environment if appropriate.

```
{
  {# NOTE: not all of the UAs listed are always malicious - they have legitimate uses so you may wish to adapt the list to your particular
environment/usage #}
  {# NOTE: "cisco" is not used by genuine Cisco equipment - rather by an openH323 based VOIP scanner trying to pass itself off as something
respectable #}
  {% set suspect_uas = [ "cisco", "friendly-scanner", "sipcli", "sipvicious", "sip-scan", "sipsak", "sundayddr", "iWar", "CSipSimple",
"SIVuS", "Gulp", "sipv", "smmap", "friendly-request", "VaxIPUserAgent", "VaxSIPUserAgent", "siparmyknife", "Test Agent", "PortSIP VoIP SDK" ] %}

  {% if service_config %}
    {% if call_info.vendor in suspect_uas %}
      "action" : "reject",
      "result" : {}
    {% else %}
      "action" : "continue",
      "result" : {{service_config|pex_to_json}}
    {% endif %}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Test whether a participant's address is within a particular subnet

You can use the `pex_in_subnet` filter to test whether a given address is within one or more subnets. This filter could be useful if, for example, you want to place media in a particular location based upon the network location of the participant.

This example template structure sets an address map variable (`nl_map`) to contain a range of subnets associated with multiple locations, and then uses that variable in multiple `pex_in_subnet` tests to see if the calling participant's address (`call_info.remote_address`) is within one of those subnets:

```
{% set nl_map = {
  "Oslo" : [ "10.47.0.0/16", "10.147.0.0/16", "10.247.200.0/24" ],
  "New York" : [ "10.1.0.0/16", "10.201.5.0/23"],
  "Sydney" : [ "10.61.0.0/16" ],
  "London" : [ "10.44.0.0/16" ] } %}
{% if pex_in_subnet(call_info.remote_address, nl_map["Oslo"]) %}
  {# Apply Norwegian rules #}
{% elif pex_in_subnet(call_info.remote_address, nl_map["New York"]) %}
  {# Apply American rules #}
{% else %}
  {# Do the default thing #}
{% endif %}
```

Lock a conference when the first participant connects

This example shows a way of locking a conference when the first participant (regardless of Host/Guest) connects. When a conference is locked a Host can manually unlock the conference with *7 or use the Connect app to let someone in.

It works by looking at the `Service tag` property of the VMR and if the tag value starts with "locked" it will lock the conference. If it doesn't start with "locked" it will not lock the conference. This requires that you set in advance the `Service tag` to "locked" of the VMRs that you want to automatically lock when the first participant joins.

```
{
  {% if service_config %}
    "action" : "continue",
    {% if service_config.service_type == "conference" and service_config.service_tag.startswith("locked") %}
      "result" : {{service_config|pex_update({"locked":True })|pex_to_json}}
    {% else %}
      "result" : {{service_config|pex_to_json}}
    {% endif %}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Use a different theme based on time of day

This example changes the theme based on the time of day in the Europe/London timezone. If it is morning (now.hour < 12) the default theme is used. If it is afternoon the "Afternoon theme" is used, otherwise the "Evening theme" is used.

```
{
  {% set now = pex_now("Europe/London") %}
  {% if service_config %}
    "action" : "continue",
    {% if now.hour < 12 %}
      "result" : {{service_config|pex_to_json}}
    {% elif now.hour < 18 %}
      "result" : {{service_config|pex_update({"ivr_theme_name":"Afternoon theme"})|pex_to_json}}
    {% else %}
      "result" : {{service_config|pex_update({"ivr_theme_name":"Evening theme"})|pex_to_json}}
    {% endif %}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Limit VMR access to registered devices only

This example allows only registered devices to call into VMRs.

```
{
  {% if service_config %}
    {% if service_config.service_type == "conference" %}
      {% if call_info.registered %}
        "action" : "continue",
        "result" : {{service_config|pex_to_json}}
      {% else %}
        "action" : "reject",
        "result" : {}
      {% endif %}
    {% else %}
      "action" : "continue",
      "result" : {{service_config|pex_to_json}}
    {% endif %}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

If you want to apply the registered-devices-only limitation to calls coming through a specific location only, then you can change

```
{% if service_config.service_type == "conference" %}
to
{% if service_config.service_type == "conference" and call_info.location == "location name" %}
```

Note that you have to apply policy profiles to specific locations anyway, but if you have additional policy statements that control other aspects of the call, this allows you to use the same policy profile in all of your locations (but only applies the registration limitation to the nominated location).

Disable encryption when calling a specific device

In some scenarios it may be necessary to not offer media encryption when dialing out to a specific device, such as older PBXs that do not support SRTP.

This example disables media encryption when calling out to a device `sip:user@example.com`.

```
{
  {% if service_config %}
    {% if call_info.call_direction == 'dial_out' %}
      {% if 'sip:user@example.com' in call_info.remote_alias %}
        "action" : "continue",
        "result" : {{service_config|pex_update({"crypto_mode":"off"})|pex_to_json}}
      {% else %}
        "action" : "continue",
        "result" : {{service_config|pex_to_json}}
      {% endif %}
    {% else %}
      "action" : "continue",
      "result" : {{service_config|pex_to_json}}
    {% endif %}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Change layout for Microsoft Teams gateway calls

This example local policy script enables the display of participant names and selects Adaptive Composition as the initial layout for all Microsoft Teams gateway calls:

```
{
  {% if service_config and service_config.service_type == "gateway" and service_config.called_device_type == "teams_conference" %}
    "action" : "continue",
    "result" : {{service_config|pex_update({"enable_overlay_text": true, "view":"five_mains_seven_pips"})|pex_to_json}}
  {% elif service_config %}
    "action" : "continue",
    "result" : {{service_config|pex_to_json}}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

If you want to apply the policy to a specific Call Routing Rule you can test against `service_config.name` (the Name of the rule), for example: `service_config.name == "name of routing rule"`

To use the 4 + 0 layout with Teams you would use: `"view":"four_mains_zero_pips"`.

Teams-like layout

To use the Teams-like layout you must use `"view":"teams"` and also enable the overlay text and active speaker indicators, as shown in the following script:

```
{
  {% if service_config and service_config.service_type == "gateway" and service_config.called_device_type == "teams_conference" %}
    "action" : "continue",
    "result" : {{service_config|pex_update({"enable_overlay_text": true, "view":"teams", "enable_active_speaker_indication":"true"})|pex_to_json}}
  {% elif service_config %}
    "action" : "continue",
    "result" : {{service_config|pex_to_json}}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

Note that:

- The layout cannot be customized via themes.
- It has the same hardware resource usage requirements as the Adaptive Composition layout.
- This layout is only suitable for use with Teams gateway calls.

Change layout for Google Meet gateway calls

This example local policy script enables the display of participant names and selects the 2 x 2 layout for all Google Meet gateway calls.

```
{
  {% if service_config and service_config.service_type == "gateway" and service_config.called_device_type == "gms_conference" %}
    "action" : "continue",
    "result" : {{service_config|pex_update({"enable_overlay_text": true, "view":"four_mains_zero_pips"})|pex_to_json}}
  {% elif service_config %}
    "action" : "continue",
    "result" : {{service_config|pex_to_json}}
  {% else %}
    "action" : "reject",
    "result" : {}
  {% endif %}
}
```

If you want to apply the policy to a specific Call Routing Rule you can test against `service_config.name` (the Name of the rule), for example: `service_config.name == "name of routing rule"`

Route incoming calls directly into a breakout room

This example shows how you could route incoming calls directly into a breakout room.

- It assumes that the main "gamesroom" VMR is already configured in Pexip Infinity, with a Name of "gamesroom" (which must have the same value as the `name` field in the `result` dictionary).
- It enables two "team" aliases that participants can dial: `teama@example.com` (to directly join the Team A breakout room) and `teamb@example.com` (to directly join the Team B breakout room). Note that these should **not** be configured as additional aliases for the main "gamesroom" VMR.

```
{
  {% if service_config %}
    "action" : "continue",
    "result" : {{service_config|pex_to_json}}
  {% else %}
    {% if call_info.local_alias == "teama@example.com" %}
      "action" : "continue",
      "result" : {"name":"gamesroom", "breakout_uuid":"00000000-0000-0000-0000-000000000001", "breakout":true, "breakout_rooms":false,
        "breakout_name" : "Team A", "breakout_description": "Team A room", "end_action":"transfer", "end_time":0, "service_tag":"gamesroom_teama",
        "service_type":"conference", "allow_guests": true, "pin": "4567"}
    {% elif call_info.local_alias == "teamb@example.com" %}
      "action" : "continue",
      "result" : {"name":"gamesroom", "breakout_uuid":"00000000-0000-0000-0000-000000000002", "breakout":true, "breakout_rooms":false,
        "breakout_name" : "Team B", "breakout_description": "Team B room", "end_action":"transfer", "end_time":0, "service_tag":"gamesroom_teamb",
        "service_type":"conference", "allow_guests": true, "pin": "4567"}
    {% else %}
      "action" : "reject",
      "result" : {}
    {% endif %}
  {% endif %}
}
```