Pexip Infinity
Hardware Resource Allocation Guide

Contents

Introduction

This guide describes how hardware resources are allocated in a Pexip Infinity deployment. It includes the following sections:

- Hardware resource allocation rules: how hardware resources are allocated and consumed in a Pexip Infinity deployment.
- Resource allocation examples: how resources are used in a number of specific scenarios, such as distributed and non-distributed conferences, gateway calls, calls using standards-based endpoints, Skype for Business / Lync clients, or a combination of clients.
- Resource allocation case study: a walk through of a fictitious deployment that at first resulted in lower than expected capacity, and how this was resolved.

Hardware resource allocation rules

A number of different types of connections to Transcoding Conferencing Nodes are required for a conference to take place, or for a gateway call to be made.

A connection can be a call or presentation from an endpoint to a Virtual Meeting Room or Virtual Auditorium, a backplane between Transcoding Conferencing Nodes, or a call into or out of the Pexip Distributed Gateway. In this context, a **connection** is analogous to a **port**.

When a connection is proxied via a Proxying Edge Node, the proxying node also consumes connection resources in order to forward the media streams on to a Transcoding Conferencing Node. A transcoding node always consumes the same amount of connection resources regardless of whether it has a direct connection to an endpoint, or it is receiving the media streams via a proxying node.

Each of these connections requires a certain amount of resource, or capacity. In general, the amount of resource required for each connection to a Transcoding Conferencing Node can be categorized as:

- audio-only
- standard definition (SD)
- high definition (HD - 720p), or
- Full HD (1080p).

The following rules determine how hardware resources are allocated and consumed by conference and gateway calls in a Pexip Infinity deployment:

- Each HD participant uses one HD connection.
- **If** these are Skype for Business / Lync participants, they each require one additional HD connection when sending or receiving presentation.
- **If** these are WebRTC participants:
  - the presenter requires one additional HD connection when sending presentation
  - the participant requires one additional HD connection when receiving full motion presentation.
- Standards-based endpoints (SIP or H.323) do not require an additional connection when sending or receiving presentation.
- Each conference instance on each Transcoding Conferencing Node reserves 1 HD connection for a backplane, to allow the conference to become geographically distributed if required. The exceptions to this are:
  - Deployments with a single Conferencing Node. In such cases, no backplanes will ever be required, so capacity is not reserved.
  - Conferences that are audio-only (in other words, where the conference has its **Conference capabilities** set to ***Audio-only***). In such cases, capacity equivalent to one audio connection is reserved for the backplane.
  - Deployments with 1080p enabled. In such cases, backplanes reserve 1 Full HD connection of capacity, approximately double that of an HD connection.
- Only one backplane connection is used for each conference on each Transcoding Conferencing Node, regardless of the number of other transcoding nodes that are involved in the conference.
- Pexip Infinity always tries to optimize gateway calls:
  - a gateway call does not reserve resource for a backplane, but will use one if required (for example, if the participants are connected via different Transcoding Conferencing Nodes)
  - if the gateway call participant is a Skype for Business / Lync client, Pexip Infinity reserves 1 HD connection in case the SfB/Lync client needs to send or receive presentation.
- SD quality can be enforced by reducing the bandwidth of a service or gateway call to 960 kbps or less. Pexip Infinity will consume less resource per-call, however, it will still reserve an HD connection for the backplane.
- Note that if an API participant is the first participant to join a conference, it will reserve a backplane for the conference.

Proxying Edge Node resource requirements

When a connection is proxied via a Proxying Edge Node, the proxying node also consumes connection resources in order to forward the media streams on to a Transcoding Conferencing Node.

A proxying node uses approximately the equivalent of 3 audio-only resources to proxy a video call (of any resolution), and 1 audio-only resource to proxy an audio call.

Extra information

See Pexip Infinity license installation and usage for full information about how call licenses are consumed.

Resource allocation examples

The examples below are designed to give you an idea of how the Hardware resource allocation rules apply as the capabilities of the Infinity Connect platform are extended in the following scenarios:

Note that in each of these cases, we describe the hardware capacity requirements in terms of HD **connections**. A connection can be a call or presentation from an endpoint to a Virtual Meeting Room or Virtual Auditorium, a backplane between Transcoding Conferencing Nodes, or a call into or out of the Pexip Distributed Gateway. In this context, a **connection** is analogous to a **port**.

The examples also assume that endpoints are connecting directly to Transcoding Conferencing Nodes. (When a connection is proxied via a Proxying Edge Node, the proxying node also consumes connection resources in order to forward the media streams on to a Transcoding Conferencing Node. A transcoding node always consumes the same amount of connection resources regardless of whether it has a direct connection to an endpoint, or it is receiving the media streams via a proxying node.)

Example 1: non-distributed VMR (standards-based endpoints)

- There are multiple Conferencing Nodes in a single Pexip Infinity system location, but the conference is hosted on a single node.
- We have 3 standards-based endpoints connecting directly to the node.
- Each endpoint uses HD video and audio.

Pexip Infinity will **require 4 HD connections** of capacity. 3 HD connections are required by the endpoints, and 1 HD connection is reserved by the backplane.

- A presentation is then sent from a standards-based endpoint.

Pexip Infinity will **require no additional connections**.

Pexip Infinity will require 3 concurrent call licenses.

Example 2: non-distributed gateway call (standards-based endpoint to SfB/Lync client)

- There are multiple Conferencing Nodes in a single Pexip Infinity system location, but the gateway call is located on a single node.
- We have one standards-based endpoint connecting via the node to a SfB/Lync client.
- Each endpoint uses HD video and audio.

Pexip Infinity will **require 2 HD connections** of capacity, one for the inbound leg (endpoint to Pexip Infinity) and one for the outbound leg (Pexip Infinity to recipient). It also **reserves 1 HD connection** in case the SfB/Lync client needs to send or receive presentation, so a **total of 3 HD connections** will be used.

- If a presentation is sent from the standards-based endpoint, Pexip Infinity will use the 1 reserved HD connection of capacity for the SfB/Lync client to receive presentation.
- If a presentation is sent from the SfB/Lync client, Pexip Infinity will use the 1 reserved HD connection of capacity for the SfB/Lync client to send presentation. However, if the incoming RDP presentation call from the SfB/Lync client lands on a different Conferencing Node to the video call, then the call will require an additional 2 HD connections (as a backplane is required in each direction between the node handling the video call and the node handling the RDP call, making 5 HD connections in total — 2 video + 2 backplane + 1 RDP presentation).

Pexip Infinity will require 2 concurrent call licenses in both cases.

Example 3: non-distributed Pexip/SfB call 1 (single standards-based endpoint to SfB/Lync meeting)

- There are multiple Conferencing Nodes in a single Pexip Infinity system location, but the SfB/Lync call is located on a single node.
- We have one standards-based endpoint connecting via the node.
- A user in a SfB/Lync meeting drags the standards-based endpoint into the SfB/Lync meeting.
- Each endpoint uses HD video and audio.

Pexip Infinity will **require 3 HD connections** of capacity. The SfB/Lync connection to the SfB/Lync AVMCU uses the equivalent of 2 HD connection resources (as we can simultaneously request a stream from the main speaker and up to 5 thumbnail streams from other participants), therefore we require an additional HD connection of capacity in comparison to a normal gateway call.

- A presentation is then sent from the standards-based endpoint.

Pexip Infinity will **require 1 additional HD connection** of capacity for the SfB/Lync meeting to receive presentation, so a total of 4 HD ports will be used. The same is true if a SfB/Lync user were to send a presentation within the SfB/Lync meeting.

Pexip Infinity will require 2 concurrent call licenses in both cases.

Example 4: non-distributed Pexip/SfB call 2 (VMR to SfB/Lync meeting)

- There are multiple Conferencing Nodes in a single Pexip Infinity system location, but the SfB/Lync call is located on a single node.
- We have 5 standards-based endpoints connecting directly to a VMR on the node.
- A user in a SfB/Lync meeting drags the VMR into the SfB/Lync meeting.
- Each endpoint uses HD video and audio.

Pexip Infinity will **require 8 HD connections** of capacity. 5 HD connections are used for the connected standards-based endpoints into the VMR, plus 2 HD connections for the SfB/Lync meeting (as we can simultaneously request a stream from the main speaker and up to 5 thumbnail streams from other participants). An additional HD connection is reserved for a backplane. This gives us a total of 8 HD connections of capacity.

- A presentation is then sent from a standards-based endpoint.

Pexip Infinity will **require 1 additional HD connection** of capacity for the SfB/Lync meeting to receive presentation, so **a total of 9 HD connections** will be used. The same is true if a SfB/Lync user were to send a presentation within the SfB/Lync meeting.

Pexip Infinity will require 6 concurrent call licenses in both cases (five for the connected endpoints to the VMR, and one to connect the VMR to the SfB/Lync meeting).

Example 5: distributed VMR (standards-based endpoints)

- There are two Conferencing Nodes in two different Pexip Infinity system locations (let's call them LAN and DMZ).
- One standards-based endpoint connects to the VMR in the DMZ node.
- Two standards-based endpoints connect to the VMR in the LAN node.
- Each endpoint uses HD video and audio.

Pexip Infinity will **reserve 3 HD connections** of capacity in the LAN node (two for the connected participants, and one for the backplane), **and 2 HD connections** of capacity in the DMZ node (one for the connected participant, and one for the backplane).

- A presentation is then sent from a standards-based endpoint.

Pexip Infinity will **require no additional connections**.

Pexip Infinity will require 3 concurrent call licenses.

Example 6: distributed VMR (standards-based and SfB/Lync endpoints)

- There are two Conferencing Nodes in two different Pexip Infinity system locations (let's call them LAN and DMZ).
- One SfB/Lync endpoint connects to the VMR in the DMZ node.
- Two standards-based endpoints connect to the VMR in the LAN node.
- Each endpoint uses HD video and audio.

Pexip Infinity will **require 3 HD connections** of capacity in the LAN node (two for the connected participants, and one for the backplane), **and 2 HD connections** of capacity in the DMZ node, (one for the connected participant, and one for the backplane).

- A presentation is then sent from a standards-based endpoint.

Pexip Infinity will **require 1 additional connection** of capacity in the DMZ node, so the **total hardware capacity in the DMZ node will be 3 HD connections**. The same is true if the SfB/Lync client were to send presentation.

Pexip Infinity will require 3 concurrent call licenses.

Example 7: single distributed gateway call

- There are two Conferencing Nodes in two different Pexip Infinity system locations (let's call them LAN and DMZ).
- We are looking to place a gateway call from a SfB/Lync client landing on the DMZ node, to a WebRTC based Pexip Infinity Connect client registered to the LAN node.
- Each endpoint uses HD video and audio.

Call flow:

Caller inbound leg (HD connection) > DMZ node backplane (HD connection) > LAN node backplane (HD connection) > Callee outbound leg (HD connection).

Pexip Infinity will **require 4 HD connections** of capacity: **2 HD connections** on the DMZ node (one for the connected participant, and one for the backplane), **and 2 HD connections** on the LAN node (also one for the connected participant, and one for the backplane).

- Now the WebRTC Pexip Connect client will present to the SfB/Lync client as well as using HD video and audio.

Pexip Infinity will **require 1 additional HD connection** for the DMZ node for the SfB/Lync client to consume presentation (a **total of 3 HD connections in the DMZ node**), and **1 additional HD connection** for the LAN node for the Pexip Connect client to send presentation (a total of **3 HD connections in the LAN node**).

If the SfB/Lync client was to send presentation rather than receive, we would still require the same 3 HD connections of capacity on the DMZ node, but as the WebRTC client does not consume a HD connection to receive content, we would only require 2 HD connections of capacity on the DMZ node.

Pexip Infinity will require 2 concurrent call licenses.

Example 8: multiple distributed gateway calls

- There are two Conferencing Nodes in two different Pexip Infinity system locations (let's call them LAN and DMZ).
- We are looking to place 5 concurrent (and separate) gateway calls from SfB/Lync clients landing on the DMZ node, to WebRTC based Pexip Connect clients registered to the LAN node.
- Each of the gateway calls will start by using HD video and audio.

Call flow:

Caller inbound leg (HD connection) > DMZ node backplane (HD connection) > LAN node backplane (HD connection) > Callee outbound leg (HD connection).

As can be seen from the example above, a distributed gateway will use 2 HD connections per node; one for the connected participant, and one for the backplane.

Therefore in this scenario, Pexip Infinity will **require 10 HD connections** of capacity on the LAN node **and 10 HD connections** of capacity on the DMZ node, so **20 connections in total**.

- Now the WebRTC Pexip Connect client in each call will present to the SfB/Lync client as well as using HD video and audio.

Pexip Infinity will require **1 additional HD connection** for the Pexip Connect client to send presentation, **and 1 additional HD connection** for the SfB/Lync client to consume presentation.

That's an **additional 5 connections per node**, so we are now using a **total of 15 HD connections per node** each of the LAN and DMZ nodes.

Pexip Infinity will require 10 concurrent call licenses.

Example 9: Pexip/SfB distributed call (VMR to SfB/Lync meeting)

- There are two Conferencing Nodes in two different Pexip Infinity system locations (let's call them LAN and DMZ).
- We have 2 standards-based endpoints connecting directly to a VMR on the DMZ node.
- We have 3 standards-based endpoints connecting directly to the same VMR on the LAN node.
- A user in a SfB/Lync meeting drags the VMR into the SfB/Lync meeting (assuming an on-prem Pexip Infinity <-> SfB/Lync integration).
- Each endpoint uses HD video and audio.

Pexip Infinity will **require a total of 3 HD connections of capacity in the DMZ node**: 2 HD connections for the connected standards-based endpoints into the VMR on the DMZ node, and 1 HD connection for the backplane.

In addition, Pexip Infinity will **require a total of 6 HD connections of hardware capacity in the LAN node**: 3 HD connections are required for the connected standards-based endpoints into the VMR, 1 HD connection for the backplane, plus 2 HD connections for the SfB/Lync meeting (as we can simultaneously request a stream from the main speaker and up to 5 thumbnail streams from other participants).

- A presentation is then sent from the standards-based endpoint.

Pexip Infinity will **require one additional HD connection** of capacity for the SfB/Lync meeting to receive presentation, so a **total of 7 HD connections will be used on the LAN node**. The same is true if a SfB/Lync user were to send a presentation within the SfB/Lync meeting.

Pexip Infinity will require 6 concurrent call licenses in both cases (five for the connected endpoints to the VMR, and one to connect the VMR to the SfB/Lync meeting).

Resource allocation case study

This is a fictitious example of a deployment that did not meet initial expectations, the reasons why, and the steps that were taken to rectify the issues that were encountered. This case study takes you through the following topics:

Requirements and initial server/virtualization specification

- Example Corp (our fictitious company) wanted to deploy Pexip Infinity as a proof of concept, with a requirement it would handle **either** a single video conference for 20 users, or up to 150 simultaneous, audio-only calls across 10 different conferences.
- They wanted to use an off the shelf, dual-CPU server - Intel E5-2660 v3 – which has 10 physical cores per socket, and a Processor Base Frequency of 2.6 GHz and 32 GB RAM (consisting of 2 x 16 GB DIMMs).
- As Pexip Infinity is a virtualized platform, Example Corp wanted to run multiple Virtual Machines (VMs) on the same host server, as they do for other VMs in their datacenter. They currently use VMware 5.5 as their virtualization platform.

Pre-deployment

Memory configuration

The memory configuration is not ideal for this server and CPU combination. We say as part of our **server design guidelines** that a Conferencing Node VM should be deployed with 1 vCPU and 1 GB vRAM per physical core, and we also state that the physical RAM

should be deployed across all channels accessed by the physical CPU. From the Intel Ark database we see that the specified E5-2660 v3 CPU has 4 memory channels, so for a dual CPU server, you should populate 8 DIMM slots (consult motherboard/server documentation as to which 8 channels should be populated if more exist), rather than the 2 slots currently occupied.

Pexip Infinity is an application that requires a high amount of compute power, with intensive calculations on data, so requires fast memory access. The more memory channels that a CPU can use, the better overall efficiency of the application.

In this case, assuming that Pexip Conferencing Nodes could utilize all available 20 cores (2 x 10-core CPUs), we would need a minimum of 20GB of physical RAM for the server. Given that DIMM modules usually are sized in 2, 4, 8, 16 etc. GB sticks, and that we need 8 modules (and assuming the same specification module is used in each slot), the ideal memory allocation is 8 x 4GB = 32 GB.

Hardware over-committing

Pexip Infinity does not support over-committing of RAM and CPU resources within a host server, i.e. it does not co-exist well with other VMs running on the same host that may use the resources allocated to Pexip Infinity VMs. Running additional VMs on host cores that are in use by Pexip Infinity results in the hypervisor time-slicing the physical host resources between the VMs. For normal applications (e.g. email, database applications, directory services, file server etc.) that may exist in a virtual environment, the time-slicing by the hypervisor makes no perceivable difference to the running application. However, because Pexip Infinity is a real-time application, this will result in poor performance, such as stuttering video and audio. Pexip Infinity is a very CPU-intensive platform, and the reason for virtualization in our case is more related to hardware abstraction rather than the traditional sharing of resources. Further information can be found in **Configuring VMware** and **Advanced VMware ESXi administration**. In particular, if you use vMotion with EVC, this can also cause issues and is also covered in this documentation.

The specified host server contains 20 cores. We must ensure that all the VMs that are running on this one host do not consume more than the 20 cores available. The specified CPU supports hyperthreading (effectively allowing 2 vCPUs per physical core); however, for Pexip Infinity to make use of the hyperthreading capabilities we need to adjust how the hypervisor works, by ensuring that we lock Conferencing Nodes to exact physical sockets. This is known as NUMA pinning, and is discussed further later on.

It is also important to note that the overall CPU utilization figure of the host server, as reported by the hypervisor, may still seem to be within reasonable tolerance of the entire CPU capacity, even if many VMs are running on the host and they consume more cores than are physically available. However, if this is true, the hypervisor will time-slice these resources, and you will still notice poor performance on Pexip Infinity. As such, if you see a low overall host CPU utilization level, it does not necessarily mean that you are NOT over-committing.

For all of these reasons, we would generally recommend specifying dedicated hosts to run Pexip Conferencing Nodes.

Rule of thumb capacity calculation

Examples of the number of connections (also referred to as ports) you may expect to achieve for various CPU types is given in **Example Conferencing Node server configurations**. As a rule of thumb, for a normal deployment (not NUMA pinned), you may expect to see the following:

- Utilize approximately 1.4 to 1.5 GHz CPU capacity per HD (720p) connection
- 1 HD connection = approximately 2 SD connections
- 1 HD connection = approximately 12 audio connections
- 1 Full HD (1080p) connection = approximately 2 HD (720p) connection (assuming Full HD is enabled on the deployment)

So, looking at the Intel Ark database for the CPU specified for this server, we can see the core speed of the processor is 2.6 GHz, each CPU contains 10 cores, and there are 2 physical CPU sockets. The calculation we can use to work out an approximate connection (port) capacity is:

$$\left(\frac{\text{Processor base frequency} \times \text{Number of cores}}{\text{CPU capacity per HD connection}}\right) \times \text{Number of CPU sockets}$$

So,

$$\left(\frac{2.6 \times 10}{1.4}\right) \times 2 = 37 \text{ HD connections (rounded down)}$$

Different hardware specifications may result in slightly lower numbers, hence, in our example documentation we have specified 35 HD connections for this CPU. This also assumes Pexip Infinity will consume all cores on the host.

Deployment 1 – making Conferencing Nodes too big

Additional deployment scope

- Example Corp is using the host server (dual CPU, 10 physical cores per socket) to run multiple VMs, although they have read our guidance and understand that they must not over-commit resources.
- They already have 2 standard IT infrastructure VMs consuming 4 cores of host resource each (8 cores in total) running on the host.
- They want to use the remaining 12 cores for Pexip Infinity. Thus, they deployed a single Conferencing Node VM with 12 vCPUs and 10 GB of vRAM.
- On boot, the administrator noted a very low HD and audio connection count (HD = 4, audio = 32).
- In the administrator log, the administrator noted the following entry:

```
Message="Multiple numa nodes detected during sampling" Detail="We strongly recommend that a Pexip Infinity Conferencing Node is deployed on a single NUMA node”
```

Understanding the deployment

Example Corp have used our calculation (with some mathematical transposition) to show that 12 of the 20 available cores could achieve at least the 20 HD connections and 150 audio connections required for the PoC (with a small amount of additional capacity).

$$\left(\frac{2.6 \times 12}{1.4}\right) = \textbf{22 HD connections (rounded down)}$$

And as they have seen that approximately 1 HD connection = 8 audio connections they have assumed they should get:

22 x 8 = 176 audio connections

So why is the connection count so low?

At first glance from the deployment notes above, it may seem that Example Corp has simply failed to follow our memory guidelines and has not allocated the full 12 GB of RAM to a Conferencing Node totaling 12 vCPUs. However, there is more to it than that.

The Pexip Conferencing Node is now using 10 cores on one socket, and 2 cores on the other. It may seem logical to simply increase the number of vCPUs assigned to a Conferencing Node in order to achieve more computational power and thus higher connection capacity. However, when the number of vCPUs on a node increases beyond the number of physical cores in a socket (for a normal deployment), the Conferencing Node VM is then hosted on two different physical CPUs and requires access to different banks of memory. This actually makes things quite inefficient, and results in poor connection capacity. NUMA nodes are described in more detail in Achieving high density deployments with NUMA. The warning log entry in the administrator log shows that the Conferencing Node has spanned NUMA nodes; this must be rectified, and the following examples show how this was done.

Deployment 2 – making Conferencing Nodes too small

Additional deployment scope

- Example Corp is using the host server (dual CPU, 10 physical cores per socket) to run multiple VMs, although they have read our guidance and understand that they must not over-commit resources.
- They are now only utilizing 5 cores on the host with their standard IT infrastructure VMs.
- Not wanting to make the same mistake as previously by spanning NUMA nodes, they decided to create smaller Conferencing Nodes to utilize the remaining 15 cores available on the host, so they deployed 3 Conferencing Node VMs with 5 vCPUs and 5 GB of vRAM each.
- On boot, the administrator saw reasonable HD and audio connection counts on each of the nodes:
  - Node 1: HD = 8, audio = 68
  - Node 2: HD = 7, audio = 65
  - Node 3: HD = 7, audio = 65
  - Total capacity: 22 HD, 198 audio

Understanding the deployment

The capacity figures are reasonable, if just a little lower than Example Corp hoped for. In this case, the Example Corp administrator had not tuned the BIOS settings on the server, but left them as the defaults, and these were configured by the manufacturer with energy saving in mind. Changing the relevant power setting to Maximum Performance and no power saving should further enhance the Pexip Infinity connection count.

However, given that the minimum requirements were met, Example Corp continued with their testing. Still, they were disappointed to find that they could not get all 20 users with HD video into a single videoconference. When the 20th user attempted to join the conference, they were disconnected with an announcement saying that capacity had been exceeded, and the Example Corp administrator saw the following entry in the administrator log:

```
Message="Alarm raised" Node="192.168.0.1" Name="capacity_exhausted" Time="1453719569.94" Details=""
```

```
Message="Participant failed to join conference." Conference="DB441"
Participant="sip:+12345@10.0.0.1;user=phone" Protocol="SIP" Participant-id="c09ee791-cd60-435e-b1f6-
36c24e3dc9fc" Reason="Out of resource"
```

So, what had gone wrong?

To answer this, we need to look back at the Hardware resource allocation rules. For a multi-node deployment, each video-based VMR instance hosted on each Conferencing Node will reserve 1 HD connection of capacity for a backplane. Because Example Corp had used 3 separate nodes, no single node had the capacity to host the entire conference, and so the conference was distributed among the nodes. Because a VMR instance was initiated on each node, Pexip Infinity immediately reserved 1 HD connection for the backplane for each instance. So, with 1 VMR instance running on each node, we can say:

Total node HD connection count – 1 HD backplane per VMR = Available node HD endpoint connections

Node 1: **8 – 1 = 7**
Node 2: **7 – 1 = 6**
Node 3: **7 – 1 = 6**

Total available endpoint connections = 7 + 6 + 6 = 19

While this was not ideal, Example Corp continued their testing, but this time for the audio conferences. However, they were somewhat puzzled in that they only reached a little over 50% of their requirement of 150 concurrent audio calls across 10 conferences, even though the raw calculation showed that the nodes could handle up to 196 audio calls.

Given that there are multiple calls and conferences occurring at the same time, it is useful to look at the data in a different view. Within Pexip support, we can create a single pivot table that shows the call load across all nodes at any single point in time. The data shows the conferences that are in operation, the number of participants, and the call media type (video, audio or presentation), across all the nodes. In the case of Example Corp and the example data shown below, these calls were all audio, so only audio call types are displayed for each node:

Count of Stream Type			
	Nodes and Media Types		
Conference (capability)	**192.168.0.1 audio**	**192.168.0.2 audio**	**192.168.0.3 audio**
IVR (audio-only)			1
VMR 1 (audio-only)	3		
VMR 2 (main video plus presentation)		6	
VMR 3 (main video plus presentation)			31
VMR 4 (main video plus presentation)	1		
VMR 5 (main video plus presentation)			9
VMR 6 (main video plus presentation)		4	

Count of Stream Type			
	Nodes and Media Types		
Conference (capability)	**192.168.0.1 audio**	**192.168.0.2 audio**	**192.168.0.3 audio**
VMR 7 (main video plus presentation)			2
VMR 8 (main video plus presentation)		4	
VMR 9 (main video plus presentation)	10	3	
VMR 10 (main video plus presentation)	12	1	
Total	**26**	**18**	**43**

At first glance, we can see from the totals that only 26 audio calls were running on Node 1, 18 on Node 2 and 43 on Node 3, for a total of 87 concurrent calls. The audio participants are first connected to a Virtual Reception (IVR); from there they enter the conference ID they wish to join, and Pexip Infinity transfers them to the correct VMR. The user currently in the IVR wished to join VMR 4, however, after they entered the conference ID, they heard the capacity exceeded warning and the administrator saw a similar entry seen previously in the Administration Log.

Why did Example Corp achieve such a poor result?

The pivot table shows some very useful information, but does NOT show all the necessary connections reserved for backplanes for each VMR on each node, so we will need to account for these as well in our capacity calculations. In the Hardware resource allocation rules we have seen that each VMR in a multi-node deployment will reserve a connection for use by the backplane. However, we also noted that a Conference Capability (call type) can be defined on each VMR, so we can set a VMR to be audio-only. In this case the backplane will only reserve a single audio connection, rather than an HD connection as is the case with a video VMR.

We have added a label beside each VMR listed in the pivot table above to show how the Example Corp administrator had configured each of the Conference Capabilities for that VMR. You can see that apart from the IVR and VMR 1, all other VMRs have been left with the default Conference Capability of "main video plus presentation". As such, each of these VMRs will reserve an HD video connection, even if all the calls within it are audio-only.

So, in summary two mistakes were made here:

1. The Conferencing Nodes were deployed with a vCPU count that was too small.
2. The VMR Conference Capability type for audio-only VMRs was incorrectly set (i.e. left as the default rather than set to audio-only).

The necessity for reservation of the backplane connection is the reason why we recommend processors with high core count: so that we can deploy Conferencing Nodes with high number of vCPUs (as long as they do NOT span NUMA nodes), rather than a larger number of nodes with a smaller vCPU count. In this way, we can achieve higher concentration of capacity with the same resources. For further information, see our **server design guidelines** and **Handling of media and signaling**.

If Example Corp had deployed 2 Conferencing Nodes, one with 5 vCPUs and one with 10 vCPUs, they would have been able to achieve their requirement of creating a single videoconference with 20 HD video participants. In addition, if they had set the Conference Capability of the VMRs that were assigned to be used for audio-only conferences, they would have been able to achieve the audio capacity listed in their requirements.

Deployment 3 – correct deployment

Additional deployment scope

- Example Corp have decided to follow our recommended best practice (as per the **server design guidelines**) and use all the resources of this host server (dual CPU, 10 physical cores per socket) for the deployment of Conferencing Node VMs.
- They continue to utilize 8 x 4 GB RAM DIMMs (total of 32 GB RAM), thus have populated all 4 memory channels for each of the NUMA nodes (sockets).

- The administrator has set the BIOS performance levels to Maximum and switched off any Energy Saving settings.
- They deploy two Conferencing Nodes, each utilizing 10 vCPUs and 10 GB vRAM. In this way, both Conferencing Nodes will consume the resources of each NUMA node (socket) without spanning.
- On boot, the administrator saw good HD and audio connection counts on each of the nodes:
  - Node 1: HD = 18, audio = 144
  - Node 2: HD = 17, audio = 136
  - Total capacity: 35 HD, 280 audio
- They ensured that the IVR and VMRs that are specifically used for audio-only calls are configured with "Conference Capability" set to audio-only.

Understanding the deployment

The connection capacities are calculated during the boot phase of each Conferencing Node, when they simulate call loads. It is not uncommon for the nodes to give a subtle variance, even when they are running on the same host and configured in the same way.

Example Corp re-ran their tests and confirmed that the hardware could meet their initial requirements, i.e. they could host either:

- a single 20-user videoconference, where each participant was connected using HD video, or
- 150 concurrent audio calls spread across 10 simultaneous conferences.

They noted that the videoconferences were initiated on Node 1. This is because Node 1 had calculated a slightly higher capacity during its boot phase, and media is allocated according to which node has the least load. When additional participants are added to the same conference, Pexip Infinity attempts to keep all media on the same node. Example Corp then noted that as the load on Node 1 increased to its maximum, media for additional participants was allocated to Node 2. This is as per the distributed system design of Pexip Infinity (for more information, see **Handling of media and signaling**). The audio conferences appeared to be more evenly distributed across the two nodes. This is due to the variation in load on the nodes as new conferences are initiated, remembering that a node with the least load will be used to handle a new conference.

Example Corp were now satisfied that the hardware was sufficient and Pexip Infinity would perform well in their environment. But they wondered if there was even more they could do. After all, they had read in the Hardware resource allocation rules that Skype for Business / Lync calls consume additional HD connections if they are either sending or receiving content. In addition, they wondered if the hardware could support both elements of their original requirements concurrently, i.e. host a single 20-person HD videoconference, and the 150 audio calls across 10 separate audio conferences, all at the same time.

Deployment 4 – NUMA pinning to increase capacity

Additional deployment scope

- Example Corp followed our recommended best practice (as per the **server design guidelines**) and used all the resources of this host server (dual CPU, 10 physical cores per socket) for the deployment of Conferencing Node VMs.
- In addition, they made use of the hyperthreading capability of the Intel E5-2660 v3 CPU, so followed our additional guidance regarding **Achieving additional performance with VMware NUMA affinity and hyperthreading**.
- They changed the physical memory to utilize 8 x 8 GB RAM DIMMs (total of 64 GB RAM), and thus have populated all 4 memory channels for each of the NUMA nodes (sockets).
- The administrator set the BIOS performance levels to Maximum and switched off any Energy Saving settings, and ensured that hyperthreading is enabled.
- They deployed two Conferencing Nodes, each utilizing 20 vCPUs and 20 GB vRAM. In this way, both Conferencing Nodes will consume the resources of each NUMA node (socket), and because the Conferencing Nodes are now pinned to a NUMA node and make use of hyperthreading, they still do not span NUMA nodes.
- On boot, the administrator saw good HD and audio connection counts on each of the nodes:
  - Node 1: HD = 28, audio = 224
  - Node 2: HD = 27, audio = 216
  - Total capacity: 55 HD, 440 audio
- They ensured that the IVR and VMRs that are specifically used for audio-only calls are configured with "Conference capability" set to audio-only.

Example Corp ran their test again. They now find that they can run both scenarios set out in the initial requirements simultaneously.

However, if all 20 video participants in the VMR were Skype for Business / Lync endpoints (a potentially unlikely scenario), and they consumed both HD video and RDP presentation (perhaps in a multi-screen user environment), this in itself would consume at least 40 HD connections, and the conference would be split across both nodes, hence the VMR instance on each node would also require an HD connection. This would leave approximately 13 HD connections available on Node 2, and based on 1 HD connection = 8 audio connections, we would only have available 104 audio connections. This would not be enough to also support the 150 concurrent audio calls across 10 simultaneous audio conferences. At this point Example Corp would need to add an additional host and nodes to their Pexip Infinity deployment.