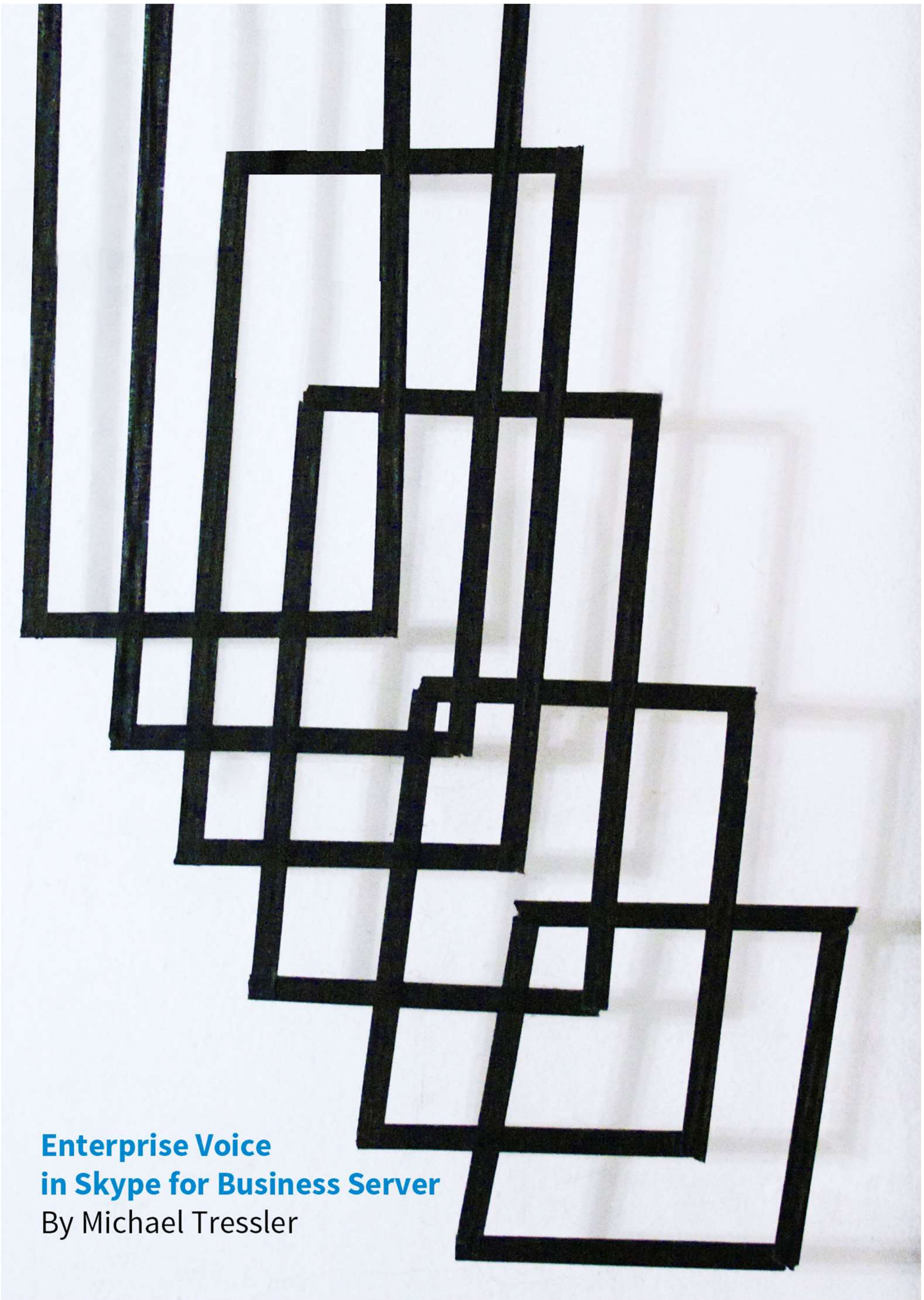


Enterprise Voice in Skype for Business Server

By Michael Tressler

Enterprise Voice in Skype for Business Server 2015

By Michael Tressler

www.evs4b.com

flinchbot@outlook.com

Enterprise Voice in Skype for Business Server 2015

by

Michael Tressler

www.evs4b.com

flinchbot@outlook.com

Published by Edgehill Publishing, Nashville, TN.

For more information on this book, including updates, news and new editions, please visit this books web site at www.evs4b.com.

Cover design by Jessica D. Adams

First edition: 2016

Printed by CreateSpace

Printed in the United States of America

ISBN: 978-0-9826459-4-9

Contents at a Glance

Table of Contents

Acknowledgments

I'd like to first and foremost acknowledge the Lync and Skype for Business community. In my opinion, it's the liveliest and most engaged community of any of Microsoft's products. The amount of blog articles, tweets, forum participation, etc. by this group is unmatched. It is because of this community that I've learned so much about Lync and Skype for Business. So much so that I figured I'm qualified to write a book!

Many thanks to Ken Lasko for his technical editing, guidance, and support of this book. It's great to know that someone of such deep technical expertise was able to point out all of my mistakes. This book as drastically better because of his help.

I'd also like to thank Andrew Young at Edgehill Publishing who guided me through the process of publishing this book and helping me get an ISBN number. I owe him yet another beer.

Finally, I'd like to thank Jessica Adams who provided original artwork for the cover and also designed the front and back covers. You can view her art at www.jessicadadams.com.

About the Author

I have been working professionally with Microsoft Windows since Windows NT 3.1 was released. The first Windows NT 3.1 server I installed for a customer ran on a DEC Alpha. That should not impress you. That just lets you know that I am old.

Since 1993 the majority of my career has been focused on the implementation, support, and operations of Microsoft server products. In 2009, my boss at the time told me that I would be responsible for our company's implementation of Office Communications Server 2007 R2. Quite honestly I wasn't terribly excited. I had always avoided the Telephony side of IT as best as I could.

However, as we started enabling Enterprise Voice in OCS, I became seduced by the possibilities of true Unified Communications. By the time Lync was released, I was voraciously reading every tidbit I could find on the Internet. We installed Lync into production in February of 2011, a scant 2 months after it was released to market. Soon thereafter my job title changed to Sr. Telecom Engineer. My avoidance of Telephony in IT had officially come to an end. But I didn't mind as the team with which I worked were rapidly converting users from old school digital handsets to Enterprise Voice enabled Lync users. By the time I left the company, we had converted around half of our users across the USA and Canada to Enterprise Voice.

I was briefly a Lync consultant and then landed at a multinational corporation where one of my primary responsibilities is standardizing the corporation on Enterprise Voice. I have worked either directly or indirectly on projects on every major continent. I doubt I'll ever do much with Antarctica.

The web page for this book can be found at www.evs4b.com. On that site you will find updates and addendums to this book. My personal website can be found at www.flinchbot.com where you can find articles I write about interesting things I've learned about Skype for Business and Lync.

I also run the UC Now website found at www.uc-now.com. This site contains an aggregation of pretty much every English-language Lync blog that is even somewhat active. As of this writing, over 130 feeds are aggregated to spit out the latest articles. For those with Windows Phones or Android devices, look for the "UC Now" app on your devices app store. There is also a version for Windows 8.1 and later.

This book should help you greatly enhance your knowledge of how Enterprise Voice in the Skype for Business world works. If not, thanks for your money and do everyone else a favor and plaster the Internet with negative reviews.

Michael Tressler

flinchbot@outlook.com

https://twitter.com/flinchbot

About This Book

I wrote this book simply because it didn't exist. There are several major tomes in the marketplace that cover Microsoft Skype for Business and Lync. These books are well-written and highly recommended. However, they provide only a cursory introduction to Enterprise Voice.

I wrote a review of one of those two books and said: "I guess someone should write the book on Enterprise Voice as it apparently has yet to be written."

That line kicked around in my head for several days until I finally realized that I should write that book. Why not? I am experienced with Enterprise Voice. I am confident that I have a strong grasp on the topic. Plus, since I am focusing on only a small slice of the whole Skype for Business pie, this won't take up too much of my time.

Famous last words.

Three years later….

After you have read this book, you should have a firm grasp on how Skype for Business and Lync integrate with the Public Switch Telephone Network (PSTN). You will be prepared to meet with various vendors and speak intelligently about the design and implementation of Enterprise Voice at your organization. You will be able to edit and expand the Enterprise Voice configuration that may already be running in your organization. You will be more attractive and the sun will always shine brightly no matter the weather and which direction you face.

In short – you will no longer be scared to handle the Skype for Business Enterprise Voice features. And if you believed that line about being more attractive and the sun always shining brightly, then maybe you should stop here and read something else. I hear Dr. Seuss is a wonderful author.

What This Book Covers

As the title clearly states, this book covers Enterprise Voice within Microsoft Skype for Business and Lync. However, that is a fairly nebulous term. In short, with regards to this book, Enterprise Voice is that piece of Lync that allows you to have voice communications with the Public Switched Telephone Network. Everything in this book applies to both Skype for Business and Lync 2013 except for the Shared Line Appearance feature. Any other exceptions will be noted. As such I will use the term Skype for Business exclusively.

What This Book Does Not Cover

Other books, blogs, and TechNet articles already cover the following pieces which are also often lumped into the category of Enterprise Voice. As such, if this book mentions the following technologies, then it is only in passing:

Call Admission Control

Call via Work

Cloud PBX

Interactive Voice Response with Exchange

Location-based Routing

Monitoring (i.e. Call Detail Records, Quality of Experience, Call Quality Monitoring)

Media Bypass

Response Groups

Unified Messaging with Exchange

For Whom Was This Book Written?

This book is being written for those IT professionals who already have some experience with Skype for Business. While pains are taken to write in a clear, simple, and concise manner, assumptions are made that you already know things such as what Topology is and that you have at least a passing grasp of basic voice technologies such as PRI’s, T1/E1 lines, etc. This book also assumes you have some familiarity with the SIP protocol and other technologies on which Skype for Business relies.

Chapter 1 – About Enterprise Voice

I suppose a good way to start this book is to define what exactly Enterprise Voice is. And it's a pretty simple explanation.

Enterprise Voice is connecting your Skype for Business environment with some other phone system.

That's it. It's that simple. Now, implementing it…that's a bit more challenging than the definition. I mean, I'm writing a whole book on how to implement this stuff.

Part of the challenge in implementing Enterprise Voice is knowing what the steps are to do it correctly. There are so many available features that it can be a bit overwhelming to know where to begin.

Enterprise Voice doesn't just mean being able to order a pizza with your Skype for Business desktop client. It also involves things such as Response Groups, Call Park, Unassigned Numbers, Call via Work, and Group Call Pickup.

It also means being able to connect with not just the public phone system (PSTN) but with an existing on-premise or cloud-based PBX.

So what all do you need to get this set up? Well, at the least, you just need a Standard Edition server with the Enterprise Voice feature enabled along with a trunk that connects to a different system. At its most complex you have multiple servers scattered around several datacenters with a handful of voice gateways.

To be specific, you need the following:

- A Skype for Business server
- A Skype for Business mediation server
- A SIP trunk connecting the mediation server with a third party system

Figure 1-1

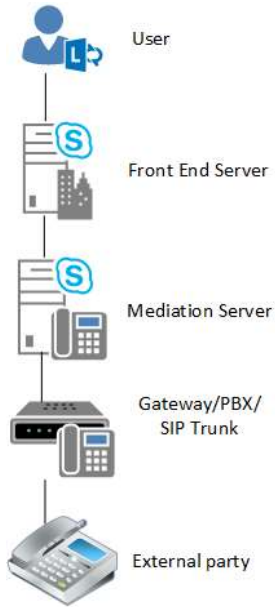


Figure 1-1 gives a basic idea of the call flow (though it's not particularly accurate). A Skype for Business user is connected to a Front End server. That Front End server then connects to the first thing that may be new to you: the Mediation Server.

What is that?

Part of the clue is in the name "Mediation". See the word "media" in there? I'm no etymologist but I don't think that's a total coincidence. A Mediation Server converts media from one format to another.

To get a bit more technical, it translates form one codec to another. In the Skype for Business server world, the Mediation Server generally translates from the native Skype for Business codecs to codecs expected by the gateway, SIP trunk, or PBX. And in pretty much every case, this is the G.711 codec.

So at its simplest, a mediation server converts codecs on one side – such as RTAudio or Silk – to the codec used by the industry standard throughout the phone system (G.711).

The Mediation Server also does a few other things such as making sure the gateways are available or routing calls to a different gateway in event of a failure.

Which brings us to the next piece in Figure 1-1. The Mediation Server somehow has to connect to the external phone system. The easiest way is by connecting to a Skype for Business certified SIP trunk.

A SIP trunk is a Voice over IP (VOIP) service that uses the SIP protocol to connect IP-based phone systems. And a Skype for Business certified SIP trunk is a SIP trunk that natively works with Skype for Business without needing a gateway to connect with the PSTN.

There are other SIP trunks that will connect natively with Skype for Business. Several PBXes can connect directly to Skype for Business so long as you can get the PBX configured correctly. Sometimes that's easier said than done.

A voice gateway is basically the Rosetta Stone of the VOIP world. A proper voice gateway can translate pretty much from anything to anything. It can translate from analog to SIP or digital to SIP....heck even SIP to SIP. Voice gateways are very powerful devices - not just because they can connect Skype for Business with pretty much any phone system on the planet.

If the voice gateway includes Session Border Controller (SBC) features, then the voice gateway can provide firewalling features to help protect your internal environment from the potential dangers of SIP trunks.

Some security conscious admins even insist on putting SBC's between Skype for Business and certified SIP trunks just for the (potential) security benefits.

Microsoft certifies voice gateways that are known to work best with Skype for Business. It is *highly* recommended that if you decide to purchase a voice gateway that you buy a certified one. Pay the extra money and know that it will work.

Architecting a Solution

So how do you know if you should get a SIP trunk or a gateway? What are the rules here?

There aren't any rules, really. Because gateways are so flexible, you can connect Skype to Business to pretty much any telephone system out there in pretty much any location. I've connected to phone systems in small islands in the Caribbean and to small islands in the Indian Ocean. And not one of those places has a Skype for Business certified SIP trunk.

I have experience on my side to help me make the decisions as to how I am going to connect these locations to the PSTN. You may not. As such, I encourage you to find an experienced vendor who knows Skype for Business Enterprise Voice. And don't just take the first guy you talk to. I've seen some vendors out there who don't know the first thing about PowerShell. I'm pretty sure that vendor wasn't the best choice.

Have the vendor assist you in architecting your solution.

There is no simple boilerplate architecture that you can use in every situation. Rather, settle on standards and use those standards when implementing Enterprise Voice in your business.

What kind of standards? Standardize on a gateway vendor. Try to standardize on a PSTN connectivity option such as using only digital trunks or only SIP trunks. Standardize your naming conventions. Standardize where you will do number manipulations (all in Skype for Business or some in Skype for business and some directly on the gateway). Standardize on your monitoring and reporting solution. Standardize on the length of extensions. Standardize on headsets, handsets, and meeting room equipment.

All of this standardization will drive your architecture. So when the time comes that you need to implement Enterprise Voice in a new location - be it across the city or around the world - you will have a pretty good idea of how you will implement it.

Quality of Service

There is one important Enterprise Voice related topic that is not discussed in detail in this book. But before you ever enable your first user, be sure you have decided on how you will implement Quality of Service) on your network.

Unlike most network protocols, SIP is a real-time protocol. You don’t care if an e-mail you sent took 2 seconds or 3 seconds to reach a user. But if it takes 2 or 3 seconds for your words to reach a remote person via VOIP, then that call is going to suck.

If you know your network protocols, you know that TCP can just keep retrying to get its packets through on a busy network. If a packets gets dropped, no worries. TCP will try again. But in a VOIP network, all of these dropped packets and retries will causes stuttering, missed words, or “robot sounding” voices. All kinds of things that will make you declare that VOIP is a terrible technology.

What QoS does is to prioritize the voice packets over all other packets on your network. An e-mail might be delayed specifically to permit your boss to have a clear voice while he fires your slacker coworker from his vacation home in Aruba.

I can’t stress enough that before you get serious with Enterprise Voice that you should get serious with QoS. Make sure every one of your routers and switches supports QoS. You may even have to replace your wireless access points with, yes, Microsoft certified access points.

Wi-Fi is a notorious VOIP killer. All of that shared bandwidth just clobbers the concept of QoS and prioritization. But some really smart people have figured out ways to greatly improve how access points can be successful with VOIP.

You should also make sure that your WAN lines will pass QoS properly. There are stories of vendors stripping the QoS markers in packets as your traffic goes from one of your locations to another.

If you have to, bring in an outside resource to help you get your network infrastructure properly configured. Nothing nukes a VOIP implementation faster than a messed up network. And note that I am not saying Skype for Business here. This applies to any VOIP system from any vendor. A solid, reliable network with QoS properly implemented will do wonders to the reliability of your VOIP rollout.

The reason this book doesn't dive into QoS is because, from the Microsoft side, it's a Windows feature and not a Skype for Business feature. QoS is implemented by Windows (usually via Group Policy objects). I don't have the desire to get into Active Directory and Group Policy.

The other reason is that, depending on who your network vendor is, the implementation of QoS on your routers and switches and firewalls will differ massively between vendors.

Call Flow

So how does a call work in Skype for Business Server? What is this stuff already mentioned about SIP and media codecs?

SIP is an XML-based protocol that focuses on call flow. SIP is used to establish a call, hang up a call, put a call on hold. The SIP protocol is focused on call control.

SIP actually does much more. Every time you send an instant message to a coworker, that whole conversation happens solely using the SIP protocol.

Media is the voice or video portion of the call. The SIP protocol is used to negotiate the type of media to be used. But once that is done, media can go its own way.

Figure 1 – 2

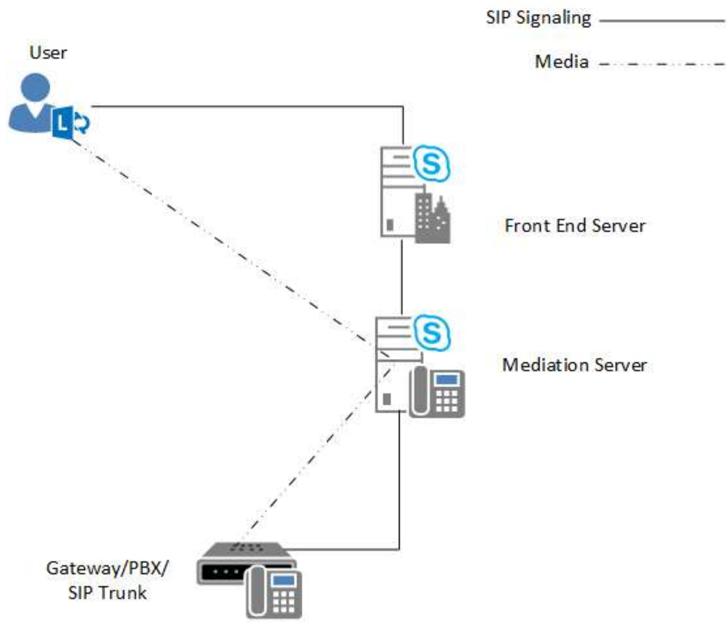


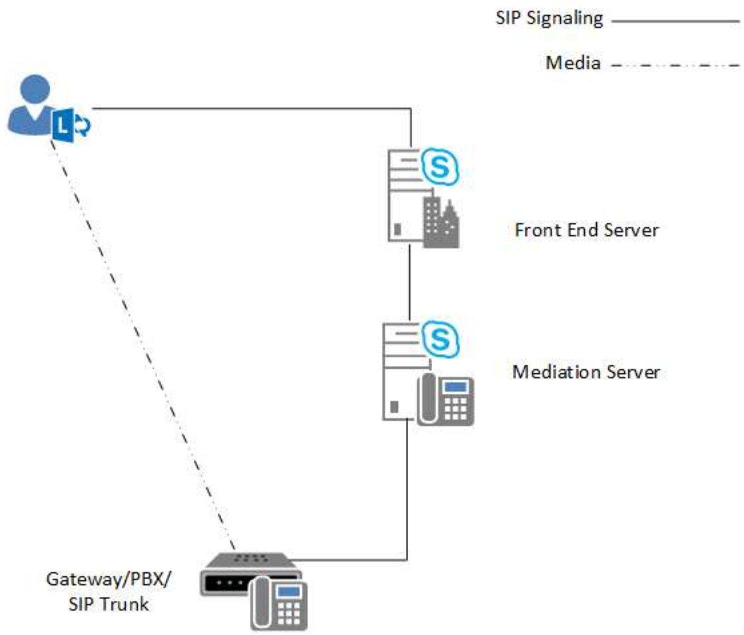
Figure 1-2 is more or less Figure 1-1 but this time with a more accurate flow. The solid line is the SIP signaling. Note that the SIP signaling goes to the Front End Server, then to the Mediation Server, and finally to the gateway.

But the media takes a different path. It goes from the client directly to the Mediation Server and then to the gateway.

This is an important concept to remember: SIP and media can (and usually do) take different paths. This is because SIP needs to control the session. It needs to tell the Front End that you are on a call so others can see your presence. SIP needs to control things like putting a call on hold or forwarding a call. It needs to tell the gateway to hang up the call. All of the components need to know what is happening.

But media - that's a different story. The goal with media is to get it from point A to point B in the shortest path possible. What's not shown in Figure 1-2 is Media Bypass, a feature where the media from the user actually skips the Mediation Server entirely and goes straight to the gateway. Figure 1-3 shows an example of the path taken with Media Bypass.

Figure 1 – 3



Components of Enterprise Voice

After you get your architecture figured out, you'll need to know what the core pieces of Enterprise Voice are so you can go about implementing this stuff. Below is a highlight of the core pieces you need to configure and, not coincidentally, what most of this book is about.

Trunks

A trunk is simply an object that connects a Mediation Server with a gateway. And the first thing you have to do before implementing Enterprise Voice is to configure a trunk in Topology Builder. A trunk defines the path that your call will take once it hits the Mediation Server.

Put another way, a trunk is the exit point from Skype for Business to your external phone system.

Chapters 2 and 7 deal with trunks.

Dial Plans

At their simplest, Dial Plans are used to make sense of whatever your users type in. If a user types four digits, is that something you are expecting to see? If they dial four digits, is that a user's extension? Or is that just someone who accidentally hit dial before putting in the rest of the phone number?

At a more didactic level, dial plans are used to normalize the phone numbers submitted to Skype for Business so that routing decisions can be easily made. If you normalize what your users dial into the E.164 standard your management of Enterprise Voice in Skype for Business will become greatly simplified.

Dial Plans are discussed in chapter 3.

Voice Policies

Voice Policies defines what your users are allowed to do.

On one hand, they control the features available to your users. Are your users allowed to forward calls? Are they permitted to use Call Park?

On the other hand, Voice Policies work with PSTN Usages and Routes to define who users are allowed to call. Can they make International calls? Can they call mobile numbers? Can they make toll calls?

Voice Policies also get into advanced topics like Least Cost Routing.

Chapter 4 is dedicated to Voice Policies.

Routes

I mentioned above that Voice Policies work with Routes (via PSTN Usages) to define who your users are allowed to call. A route basically has 2 pieces to it.

1. A rule defining if a destination is permitted to be called. A destination could be a mobile number or an international number.
2. The trunk that will be used to place the call.

See chapter 5 for more information on Routes.

PSTN Usages

Probably the simplest object in configuring Enterprise Voice is the PSTN Usage. There is very little to it. However, it seems that this is also one of the most confusing aspects when learning Enterprise Voice.

All that a PSTN Usage does is connect a Voice Policy (what a user can do) with a Route (who a user can call). My opinion is that because the PSTN Usage has so few things to configure that administrators never get a proper introduction to what a PSTN Usage actually does.

I hope I've done a good job in chapter 6 of breaking down this deceptively simple concept so that you won't get confused about the humble PSTN Usage.

Summary

This chapter is just a quick introduction to what Enterprise Voice is. There are a bunch of ways to connect Skype for Business with other systems, either directly via SIP trunk or indirectly via a gateway.

No matter how you connect to the PSTN, make sure your network is ready to handle the additional workload of VOIP. Do not underestimate the importance of both optimizing your network (getting rid of bottlenecks, optimizing data paths, etc.) and the importance of QoS. Because even if you have the perfect data network, your VOIP performance could go down the tubes the second someone copies a 2 TB file across the network.

I've heard stories of VOIP working great 23 hours a day and then the performance is terrible for one hour at the exact same time every day. Turns out a large backup was scheduled at that time and it chewed up the whole network. QoS can alleviate some of that; rescheduling your backups to run during off hours can help that too.

Every network admin thinks he has his network perfectly tuned. VOIP will find the flaws in his perfect design. Be gentle when asking the network engineer to make changes. He didn't design the network for the sensitivity of real-time communications and re-architecting the network can be quite a challenge.

Skype for Business tries its best to work on any network. Microsoft didn't just randomly choose the codecs that Skype for Business uses. Some people even think the most valuable thing in the consumer Skype product is the codec used. A codec now also used in Skype for Business.

Beside just using the right codec at the right time, Skype for Business media will try to take the shortest path to its destination.

You've been given a high-level overview of the configurable pieces in Enterprise Voice.

Now let's start implementing this stuff.

Chapter 2 – Gateways

This chapter will briefly explore how to connect Skype for Business to the Public Switched Telephone Network (PSTN) as well as to other equipment such as a legacy PBX. There are a lot of configuration options and not all of them will be covered. The goal here is to introduce you to the major options as well as giving some examples of how to integrate them with Skype for Business.

With regards to connecting Skype for Business with the PSTN, each network is unique and there is no way that all possible configuration options could be covered. As such, this chapter is fairly short and high-level compared with the other chapters. However, don't take that to mean that this material is unimportant. On the contrary, the architectural design of the gateways and Mediation Servers is of paramount importance when deploying Enterprise Voice.

Gateways

A gateway is a hardware device that translates between two protocols. In the case of Skype for Business, a gateway receives SIP and media on one interface and shoots out a different protocol on a different interface. For example, you may connect a digital PRI voice circuit from your telephone company (telco). In this configuration, your telco provider is providing you a line that does not understand SIP or the media (codecs) that Skype for Business supports. To make a call to order a pizza, something needs to convert the Skype for

Business signaling and media to something the telco understands. This is a gateway.

Keep in mind that a gateway works bi-directionally. If the pizza store calls back to verify the order, the gateway needs to convert the signal from the telco to a SIP and media session that Skype for Business understands.

Gateways can be configured to adapt to many scenarios. For example, many telco's send only 4 digits to determine the inbound phone number. For example, if your DID range is 13175551000-13175551999, the telco will often just send a number between 1,000 and 1,999 and leave it up to you to map those 4 numbers to the full phone number. In the case of Skype for Business, you can use the gateway to convert those 4 digits to an E.164 compliant number and then send the full E.164 number on to Skype for Business.

This raises a question: Where should you do your number manipulation?

For consistency's sake, it's best to do as much as possible within Skype for Business so you don't have to scrounge around trying to determine where a manipulation is happening. The more of this that happens in Skype for Business, the more standardized you are and the easier your troubleshooting becomes.

However, there may be advantages in converting inbound numbers to E.164 directly on the gateway. One advantage is that once an inbound number hits Skype for Business, it is already a properly formatted E.164 number. This can be a big advantage if the DID range overlaps with an internal 4-digit dialing range.

In one Skype for Business implementation, we had a remote office that had their own Survivable Branch Appliance (SBA). The office's main number (pilot number) was configured to be answered by an Interactive Voice Response (IVR) provided by the Unified Messaging role in Microsoft Exchange. The twist is that the Exchange server was at another office and inbound calls needed to ride the internal MPLS network to reach the Exchange server.

What happens if the MPLS network should crash? Well, that happened and the MPLS outage was expected to be unacceptably long. Therefore, we re-routed the pilot number on the gateway to an employee at the remote office who then manually answered all inbound calls. This was an easy change for us to make because we configured the gateway to handle the conversion from the 4-digits provided by the telco. We simply edited the rule that converted the pilot number from the number for Exchange to the number of the employee. Once the MPLS line came back, we undid our change and the calls resumed making their way for Exchange to handle.

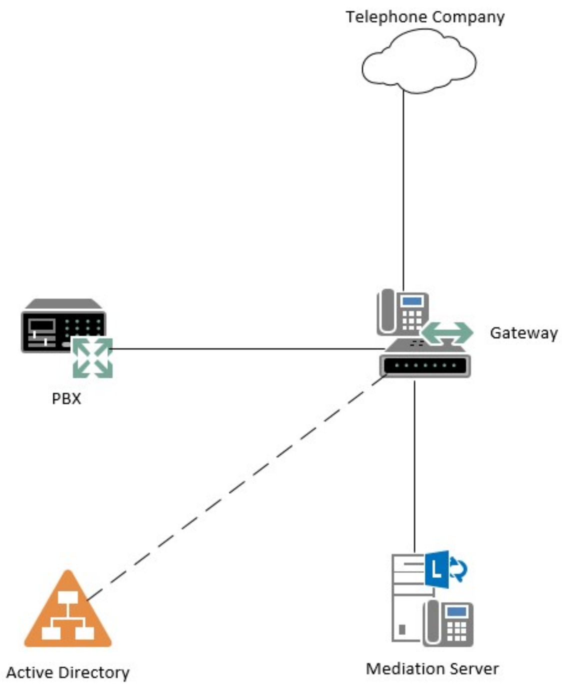
Had we done the number manipulation in Skype for Business we would have been stuck. Since the SBA had lost connectivity to the Central Management Store, any changes we made to Skype for Business wouldn't get replicated to our remote office until the MPLS line came back up.

You might also use a gateway in order to connect Skype for Business with a legacy PBX, perhaps because you need to migrate from the PBX or because you are using Skype for Business in a testing situation and aren't ready to remove the PBX.

Assume you have a PBX that does not have any SIP or IP capabilities. One way to connect this PBX to Skype for Business is via digital trunks. You would put a gateway between Skype for Business and the PBX and use the gateway to perform this translation.

Figure 2-1 gives a visual representation of how this connectivity might look. Note that a gateway can connect to several different end points.

Figure 2 – 1



One neat feature of many gateways is that they are Active Directory-aware. This means that as the gateway makes routing decisions, it can refer to information within Active Directory to determine if a user is configured for Skype for Business.

Assume a call comes in from the telephone company to the gateway. Also assume you have users still using the PBX and you have users configured with Skype for Business Enterprise Voice. As the call comes in to the gateway it must route the call to either the PBX or to Skype for Business. One way to automate the decision making is to have the gateway configured to use Active Directory. Using a simple AD lookup, the gateway can determine if the phone number is for a Skype for Business user or not. If it is, it forwards the call to Skype for Business. If not, it forwards the call to the PBX.

Another benefit of this is that as you migrate users from the PBX to Skype for Business, you don't need to do any manual configuration on the gateway. As Active Directory is updated and the changes replicated, the gateway dynamically learns where it should route the call.

SIP Trunks

Most telco providers are now providing SIP trunks. Some of these telco's provide a SIP trunk that is natively compatible with Skype for Business. What this means is that you don't always need to have a gateway between your network and the telco provider. Since both Skype for Business and the Skyper for Business certified SIP trunk are using the same SIP and media protocols there is no need for a gateway to translate.

Note that with many SIP providers you must still provide a Session Border Controller (SBC) to complete the connection. An SBC is a gateway with specific software that can translate between SIP trunks. While the SIP protocol may be the same, the provider may be using codecs that are not compatible with Skype for Business. It is also possible that one side or the other has certain features enabled that the other doesn't natively understand. An SBC helps to smooth out these issues and ease the deployment of the SIP trunk.

An SBC can also provide firewall capabilities by filtering out unwanted traffic before it gets to your internal Skype for Business servers. Depending on your security requirements, you may want to use an SBC even with a Skype for Business certified SIP trunk.

Skype for Business Certification Program

In order to help you select a vendor that will work correctly with Skype for Business, Microsoft has a certification program called, intuitively enough, the "Skype for Business Certification Program". This is a program that assures customers deploying Skype for Business that products that are part of the program have been certified to work correctly with Skype for Business.

Before selecting a Gateway, SIP trunk provider, or an SBC vendor, it is important to check the certification program website. This site is frequently updated with new vendors and new products.

Beyond gateways, SBC's, and SIP trunks, the site also lists supported IP-PBXes, Survivable Branch Appliances, E911 providers, and more.

You can access the website via this URL:

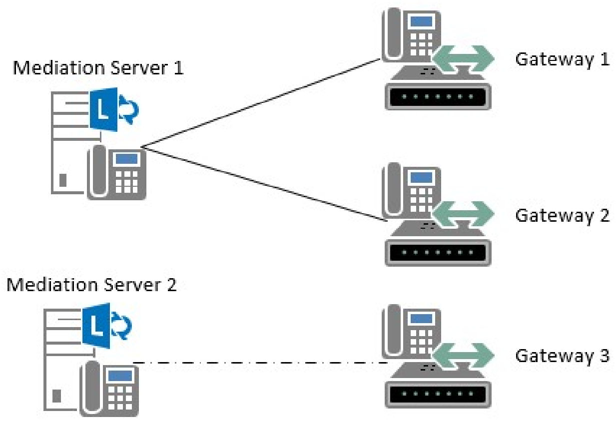
https://technet.microsoft.com/en-us/office/dn947484.aspx

> As each vendors' management interface is different, no guidance can be provided on how to properly configure your gateway or SBC. The one major piece of advice I can offer is that when installing your gateway or SBC, be sure to closely follow the vendors' instructions on how to implement their gateway with Skype for Business.
>
> As part of the certification, each vendor must provide a Quick Start Guide on how to implement their hardware successfully with Skype for Business. Experience has shown that not following these instructions closely can cause for interesting problems long after the gateway has been deployed and put into production.
>
> It's also a great idea to hire the vendor or one of their recommended resellers to help with the implementation of the gateway.

Trunks

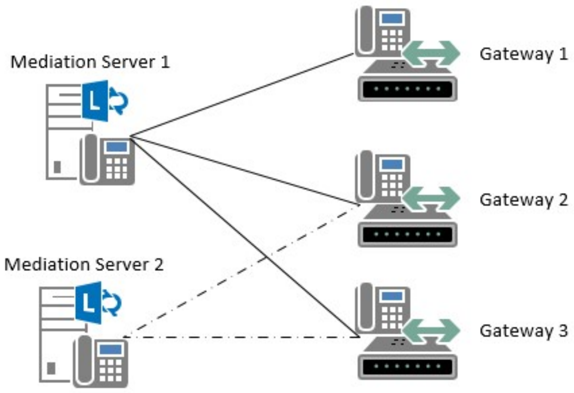
In Skype for Business, a trunk is something that connects a gateway with a Mediation Server. In Lync 2010, a gateway connected directly to a Mediation Server without the use of a trunk. This limited the design flexibility and limited the redundancy options. In Lync 2010, this was called "1:N" routing where a single Mediation Server was able to connect to multiple gateways. The major downside was if your Mediation Server failed then all of your calls would fail too. Some resiliency was possible by using a Mediation Server pool to limit the impact of a single Mediation Server failure.

Figure 2 – 2, 1:N routing



Lync Server 2013 introduced “M:N” routing – multiple mediation servers could now connect to multiple gateways. This concept is also used by Skype for Business Server 2015. Now you can lose an entire mediation server pool and your calls can still be completed by using a secondary mediation server pool.

Figure 2 – 3, M:N routing



As seen in figure 2-3, both Mediation Servers 1 and 2 can route to gateways 2 and 3. If Mediation Server 1 fails, calls can still be completed to gateways 2 and 3 but not to gateway 1. If Mediation Server 2 fails, calls can still be completed to all of the gateways.

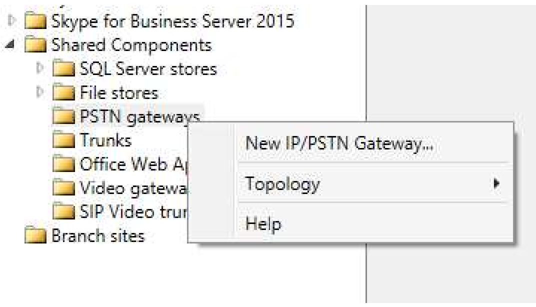
Creating Trunks

Trunks are created in the Skype for Business Topology Builder. Within Topology Builder you define which gateways can speak to which Mediation Server. You will need to have 5 pieces of information before you can add a trunk:

1. The IP address of the gateway or – preferably - the name if it has been added to DNS
2. The listening port for the gateway
3. The SIP transport protocol (either TCP or TLS)
4. The associated Mediation Server
5. The server port on the Mediation Server

With these bits of information, we can now open Topology Builder and add the gateway to the Topology. Open Topology Builder and navigate to the Shared Components folder within your Skype for Business Site. Then expand the "PSTN Gateways" folder. Finally, right click on the "PSTN Gateways" folder and select "New IP/PSTN Gateway…"

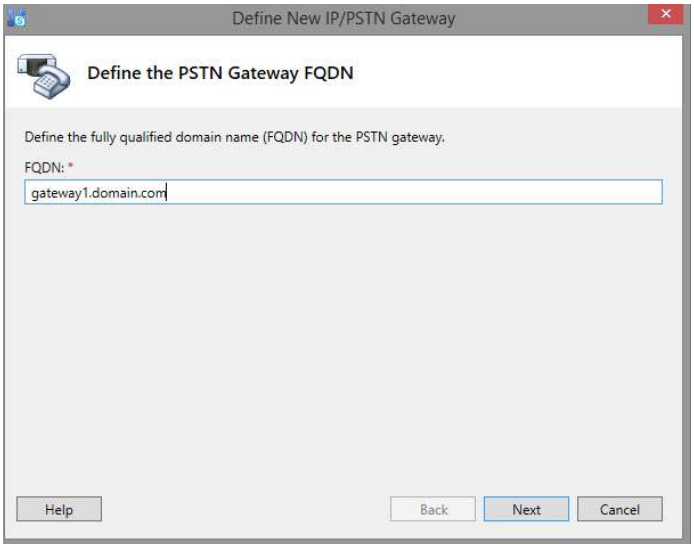
Figure 2 – 4



You can now enter the information you have collected. First add the fully qualified domain name (preferably) or the IP address of the PSTN Gateway. You should use DNS names when possible. This will simplify your future maintenance. For example, if you replace your PSTN gateway with a newer model, you just need to change the IP address in DNS to start pointing traffic to the new gateway. If you add an IP address for this upgraded gateway, you will have to edit all of your routes to reflect the new gateway name (i.e., the new IP address).

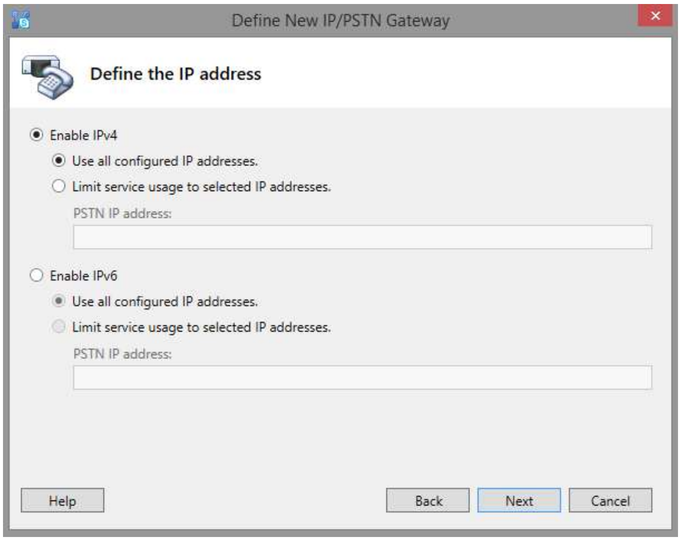
Also, if you choose to use DNS names now and wish to use the insecure TCP protocol, you already have the DNS portion in place. This makes things a touch easier should you choose to upgrade to the secure TLS protocol later.

Figure 2 – 5



After you have entered the gateway name or IP address, click "Next" to configure if you are using IPv4 or IPv6. You can also choose whether to "Use all configured IP addresses" or "Limit Service usage to selected IP addresses". In general, leave this as the default to use all configured IP addresses. Click "Next".

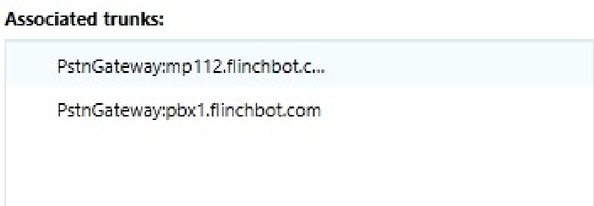
Figure 2 – 6



You are then brought to the screen where you configure the trunk. Your trunk name can be anything but keep it simple. It's also best to use unique names at the beginning of the Trunk name instead of at the end. This is because the Skype for Business Control Panel truncates the names of trunks.

An example can be seen below.

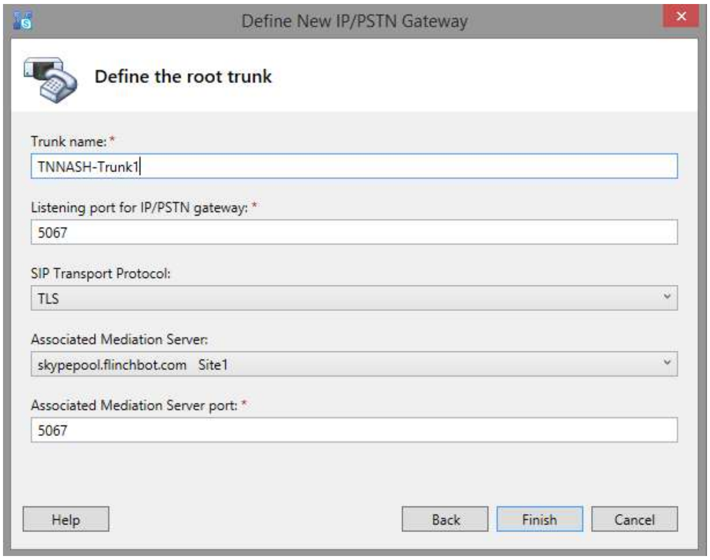
Figure 2 – 7



Note that the first gateway in the list is truncated. It may be named "mp112.flinchbot.com-1" and a second trunk could be named "mp112.flinchbot.com-2". But this view on the Routes page in Control Panel makes it very difficult to tell which specific gateway is being used.

The trunk name can be anything – it does not have to be defined in DNS. The port values should be decided based on the values you configured on the gateway. They can be anything though the defaults for Skype for Business are 5067 for TLS connections and 5060 for TCP.

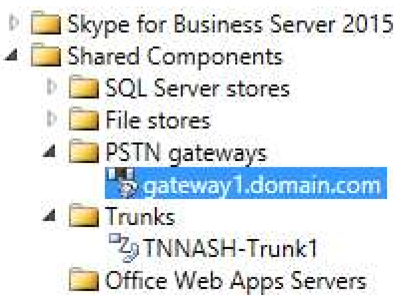
Figure 2 – 8



We also need to select the Mediation Server to which this trunk will connect and the port that the Mediation Server is listening on. Try to keep the Mediation server ports as default. There is rarely a reason to have to change this value.

After clicking finish, you will have a new gateway and trunk defined in Topology Builder.

Figure 2 – 9



If you want to connect this gateway to a second mediation server, create a new trunk using the same information as the first trunk but selecting a different mediation server. This is how we would configure Gateways 2 and 3 in Figure 2-3.

Summary

Connecting Skype for Business with the PSTN is absolutely one of the requirements if you hope to enable Enterprise Voice. This can be done either with gateways, SIP trunks, or both. It is important that you refer to the Skype for Business Certification Program before making a purchase.

When dealing with vendors, make sure that the solution they offer you is scalable enough to handle what you anticipate to be your maximum concurrent calls to and from the PSTN. You also want to make sure that you understand how they handle redundancy in case of a component failure somewhere along the line. Finally, you probably should discuss your options for handling disaster recovery. This conversation should also take place with your telco provider.

The design of your connectivity to the PSTN is an absolutely crucial piece to get correct. Make sure you have detailed discussions with your telco provider and gateway vendor when designing this piece. The gateway vendors are also very experienced in implementing their hardware in a variety of scenarios. Find vendors that you trust and have experience integrating their solutions with Skype for Business.

Finally, if you have not discussed the advantages of SIP trunks with your telco provider, I highly recommend you do. There are many advantages to SIP trunks when compared to traditional digital delivery. One major feature revolves around number redirection and disaster recovery scenarios. If you are at all considering disaster recovery for your Skype for Business implementation, then detailed discussions of SIP trunks is compulsory.

Chapter 3 – Dial Plans

In this chapter you will learn about Dial Plans in Skype for Business. You will learn what a Dial Plan is and how to best use it in your Enterprise Voice deployment.

At its essence, a Dial Plan does nothing more than taking the user input and normalizing it to a common standard. In Skype for Business, that common standard is E.164:

E.164 defines a general format for international telephone numbers. Plan-conforming numbers are limited to a maximum of 15 digits. The presentation of numbers is usually prefixed with the character + (plus sign), indicating that the number includes the international country calling code (country code)…

http://en.wikipedia.org/wiki/E.164

In order to simplify the routing and decision making within Skype for Business, all dialed phone numbers should be normalized to the E.164 standard. It is the Dial Plan that performs this normalization.

An E.164 number looks similar to the following: +13175551212. The + sign is used to represent the digits needed to dial an international call. In the United States, you need to dial 011 before calling internationally. In Germany, you need to dial 00 as the prefix.

Without using E.164, a call from the United States to Germany would look like this:

011 49 40 555555

We can represent that same number in E.164 format as

+49 40 555555

Immediately following the + character is the country code. This code can be any length from 1 to 3 digits and is assigned by the International Telecommunication Union Telecommunication Standardization Sector (ITU-T).

A full list of all country codes can be found here:
http://en.wikipedia.org/wiki/List_of_country_calling_codes

Following the country code is a string of numbers whose exact syntax is defined by a given country. In the United State, the 3 digits following the country code represent the Area Code. In Germany the next 2-5 digits following the country code are the area code. In Colombia, the next digit is a city code.

So with each country having its own set of rules (not to mention cities with their own unique configurations) a standard needs to be set and that standard is E.164. If you are able to normalize (or standardize) all of the numbers in your Skype for Business environment to E.164 format, then you will greatly simplify your Skype for Business roll out.

Note that you don't necessarily have to standardize all of your Skype for Business numbers to E.164 format for Skype for Business to work correctly.

However, file "E.164 normalization" in your mind under the "best practices" heading.

Dial Plan Scope

Like many other parts of Skype for Business, the Dial Plan has a default Global plan. Barring any additional dial plans, all users (and devices) will be assigned to the Global dial plan. It is best to create one or more specific dial plans and not use the Global dial plan at all.

Aside from the Global dial plan, you can create additional dial plans scoped to a Site, a Pool, and a User.

A Site dial plan is a dial plan than can be used by all pools and users within a Skype for Business Site. In this case, it is very analogous to a Global Dial Plan. However, unlike a Global dial plan, if you have multiple sites then a Site Dial Plan can only be applied to a specific Site. A Site dial plan can be used even if your Skype for Business Site covers a geographically dispersed network. These dial plans are not limited to a single building, network, subnet, or server.

As a reminder, a Site is defined in your Skype for Business Topology.

A Pool Dial Plan is limited to a specific Skype for Business pool within a given Site.

And finally, a User Dial Plan can only be assigned to specific users.

If there are multiple Dial Plans, then the User Dial Plan has the highest priority followed by the Pool, Site, and Global Dial Plans.

If the Dial Plan for a user is set to "automatic" then the following table lists which Dial Plan would be assigned to the user.

Dial Plan Scope	Action
Pool	If there is no User Dial Plan, then this gets applied. Unless there is no Pool Dial Plan defined then…
Site	This gets applied as there is no Pool Dial Plan. Unless there is no Site Dial Plan defined then…
Global	This gets applied.

As for any guidance as to which to use, it is generally best to have as few Dial Plans as possible in order to minimize the difficulty in troubleshooting why a certain normalization is or is not being applied.

Site and Pool Dial Plans can minimize the user maintenance required as they can be automatically applied. However, in times of maintenance or failure you may have to move your users to a different pool. At that point a different Dial Plan could get assigned that is incompatible with their local dialing rules.

If you envision moving users around between different pools, then a User Dial Plan is your best option. However, if users will not move between pools often or the dialing rules between different pools is the same, then a Pool or Site Dial Plan may be best.

You would create a Site Dial Plan in order to keep the Global Dial Plan pristine. You will need to add normalizations to the Site (or Global) Dial Plan for "devices" in your Skype for Business environment. Think of a device as anything that makes calls that is not assigned to a real human. An inbound call from a gateway or a SIP trunk is an example of this. Since these are devices, and not users, you cannot assign a User Dial Plan. As such, you have to use (preferably) the Site dial plan or (less preferably) the Global dial plan.

The primary downside to using User Dial Plans is that you will often have duplicate entries in both the User and Site Dial Plans. However, this approach will provide you with maximum flexibility so in most cases it's a decent tradeoff.

Now is also a good time to review how users in your company dial each other. Do you currently have a 3, 4, or 5-digit dial plan? Do some users only dial 4 digits while some dial 5? Are there overlapping number ranges in each office? Does everyone have a Direct-Inward Dial (DID/DDI) number?

If you are bringing in Skype for Business as a new voice platform, it is an excellent time to try to standardize the dialing across your corporation. This is often a large task that involves some political wrangling and general salesmanship. However, because you can centralize all of your dial plans in one place with Skype for Business (as opposed to having each PBX at each of your offices running semi-autonomously) it will be to your benefit to standardize as many of your existing dial plans as possible. Doing some legwork now and deciding how you want to have all of your users dialing will save hours of time in troubleshooting and configuring Skype for Business. For example, if you can get all of your users to be standardized on 4-digit dialing then the implementation and support of your Skype for Business environment will be much easier in the long run.

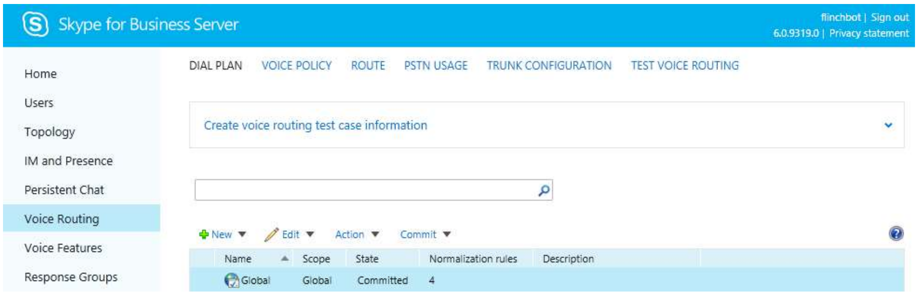
Creating a Dial Plan

While Dial Plans can most certainly be created using the Skype for Business Server Management Shell, it is generally easiest to create and manage Dial Plans using the Skype for Business Server Control Panel.

To create a Dial Plan, follow these steps:

1. Open the Skype for Business Server Control Panel
2. Navigate to the Voice Routing tab
3. You should now see a screen similar to the one show in Figure 3-1.

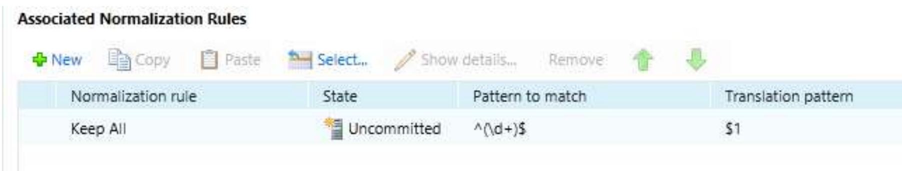
Figure 3 – 1



4. Clicking on the New button will present you with the option to create a Site, Pool, or User dial plan.
   a. If you chose Site or Pool, you must then select the Site or Pool to which this dial plan will be assigned.
5. Provide a name and the "Simple Name" field will be automatically applied. The Name can be anything you want. The Simple Name field has some limitations. The Simple Name cannot contain any spaces. It can contain any letter and is limited to the following range of characters: '-', '.','+', or '_'. A Simple Name helps when using PowerShell by omitting reserved PowerShell characters.

6. The Description field lets you provide a useful comment as to why this Dial Plan exists. Though this field is optional, the experienced administrator will use this field to his or her advantage.
7. You can leave the "Dial-In conferencing region" blank. The field is used with Dial-in conferencing which is discussed in chapter 9. This field has no impact on getting Enterprise Voice set up so for now we can ignore it.
8. The External Access Prefix is only used when dialing in "off-hook" mode when you need to dial a number (or set of numbers) before dialing the number you really want to dial. A typical example on a PBX is to dial a 9 to signify that the call is an external call.
9. As seen in Figure 3-2, all new dial plans automatically get one normalization rule assigned: the "Keep All" rule. If you read Appendix 1 on Regular Expressions, then the "pattern to match" field should be self-explanatory. The pattern ^(\d+)$ means that this normalization will match any numbers (e.g. 18085551212). I will discuss the "Translation Pattern" later.

Figure 3 – 2



Since we are creating a simple Dial Plan right now, we will not add a new normalization rule at this time.

10. The final field – "Dialed number to test" - will also be left blank. This field helps you test your normalization rules.
11. At this point, click the OK button at the top and your new Dial Plan will now appear in the list of dial plans.

Your Dial Plan is not quite saved yet. In order to permanently add this Dial Plan, we will need to commit our changes using the "Commit" menu item. Select the "Commit" pull down and choose "Commit All". A new window appears showing you what you are about to commit. If it all looks OK, click the "Commit" button at the bottom.

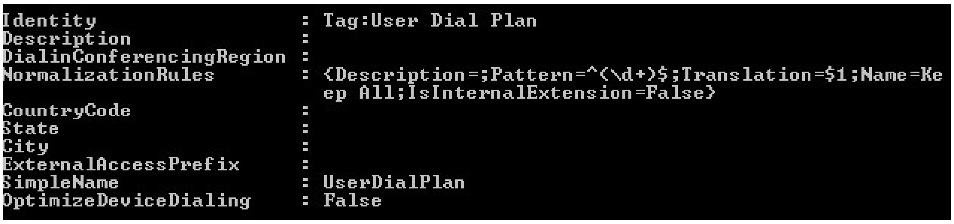
You have now made your first Dial Plan. Woo Hoo!

If you would like to create a Dial Plan via PowerShell, you need to use the `New-CsDialPlan` cmdlet. To create a new User Dial Plan named "User Dial Plan" you would type in the following:

```
New-CsDialPlan –Identity "User Dial Plan" –SimpleName
"UserDialPlan"
```

The output of the New-CsDialPlan cmdlet can be seen below.

Figure 3 – 3



```
Identity                  : Tag:User Dial Plan
Description               :
DialinConferencingRegion  :
NormalizationRules        : {Description=;Pattern=^(\d+)$;Translation=$1;Name=Ke
                            ep All;IsInternalExtension=False}
CountryCode               :
State                     :
City                      :
ExternalAccessPrefix      :
SimpleName                : UserDialPlan
OptimizeDeviceDialing     : False
```

Notice that Skype for Business created the default normalization rule for this new Dial Plan. At the end of the next section you will see how to add your own custom normalization to a Dial Plan via PowerShell.

Creating Normalizations

For this section, we need to make some assumptions. Let's assume the following:

- The main number for our company is 615-555-1212. (Our area code is 615).
- Our company is based in the United States. As such, our country code is 1.
- In order to make a local call, we only need to dial 7 digits.
- Users can dial other employees by dialing a 4-digit extension.

- In order to make a National Long Distance call, we need to dial 11 digits.
- In order to make an International Long Distance call, we need to dial 011 followed by the country code and then the number we are trying to reach.

When creating normalizations, what we are actually doing is answering the following question:

Who did the user intend to call?

In our case, if the user only dials 4 digits we assume the user wants to dial a fellow coworker. If the user dials 11 digits starting with a 1 we assume they want to make a long distance call to someone in North America.

What happens if the user dials 5 digits? Well, if we don't have a normalization built to handle 5 digits, the call will simply drop – and that is a perfectly OK solution. We only want to create normalizations for what we know and anticipate will be called. We do not create normalizations for every possible scenario.

We also don't need to worry about characters that are dialed by the user. The Skype for Business client automatically filters those out for us. If the user dials (317) 555-4646, we only have to worry about the string of digits 3175554646.

> Note: The Skype for Business client will also translate letters to their corresponding T9 digit. So if you dial 1-800-Flowers this will automatically get converted to 18003569377.

Let's go ahead and create our first normalization. We want to make sure that a national long distance number can be successfully dialed. Now, we could train all of our users to add a + sign every time they dial a long distance number. But that isn't a very good approach - at least in North America where most people are oblivious to E.164 and the + sign. So in order to make life easier for them, we will automatically add the + sign any time they dial a long distance number.

Let’s assume a user entered the 11-digit phone number “12425551212” into their Skype for Business client. Since this number is missing the + it is not a properly formatted E.164 number. We need to add the + sign.

To do this, open up Skype for Business Server Control Panel and navigate to the Voice Routing tab on the left. At this point you should see the Global Dial Plan along with a Dial Plan you may have created earlier. To keep things simple, let’s use the Global Dial Plan to add our normalization.

Open the Global dial plan either by double clicking on it or, as seen in figure 4, clicking on it once and selecting “Show details…” from the edit menu.

Figure 3 – 4

You will now be on the details page for the Global Dial Plan.

Scroll down to the “Build a Normalization Rule” section.

You can use this section to build simple normalizations. The default “Keep All” normalization is a simple one and you can follow the logic to see how the dialed number will be manipulated.

Figure 3-5 shows the rule builder for the “Keep All” normalization.

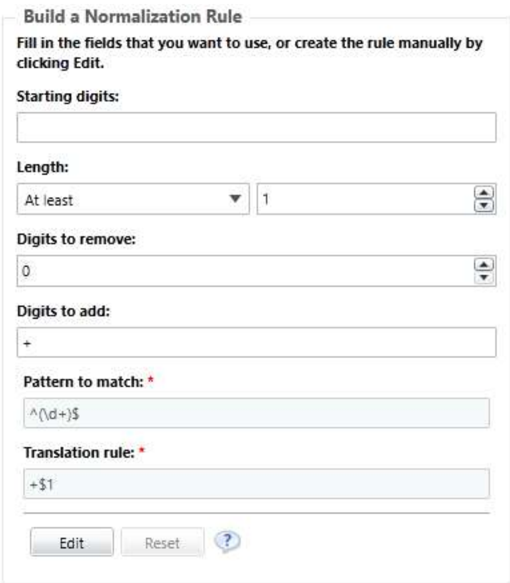
We can see that the length of the dialed number must be at least 1 digit in length. So this will match if the user simply dials “7” or if the user dials “2093587104282157”.

No digits will be removed from the dialed number but we will add a “+” to whatever the user dialed.

The bottom portion of the Normalization Rule builder shows the regular expression (regex) that is generated by the settings defined above. For complex normalizations, you can click the “Edit” button and enter your own regular expression. Skype for Business supports the full range of .Net regular expressions so you are only limited by your imagination as to how you will construct your normalization rules. Well, that and your knowledge of regular expressions.

If you are unfamiliar (or uncomfortable with) regular expressions, now would be a good time to review Appendix 1 which is a tutorial on regular expressions.

Figure 3 – 5



At the bottom of the rule builder is a “Translation rule” field. This field works in concert with the “Pattern to match field”. Notice in the “Pattern to match” field that there are parentheses surrounding “\d+”. In regular expressions, anything surrounded by parentheses becomes a variable. Since this is the first set of parentheses, this output is assigned to a variable called $1. If there were a second set of parentheses, then the digits within those parentheses would be assigned to a variable called $2.

The “Pattern to match” simply states that if any digits were dialed, take those digits and assign them to the regex variable $1.

The “Translation rule” takes the variable and allows us to add additional digits (or characters) to that variable.

We know the user dialed 11 digits to make their national long distance call (i.e. 12425551212) but we need to add a plus. In this case, our translation pattern is very simple and we just add a + to the $1 variable. As can be seen in Figure 3-5, the translation rule is simply *+$1*.

Well that was an easy one! Skype for Business already created that one for us! But let’s see if it actually works. Let’s test it to see what happens if a user dials some digits.

Underneath the “Build a Normalization Rule” section is a field entitled “Dialed number to test:”. In this field, we can simulate what a user dials in their Skype for Business client and see if the normalization rule matches what the user dialed. You can type any digits into the box and, after clicking the Go button, you should get a match.

Figure 6 shows the results of entering some digits into the box and clicking “Go”.

Figure 3 – 6

Note that it shows what the number has changed to – exactly the digits entered but now with a + sign in front.

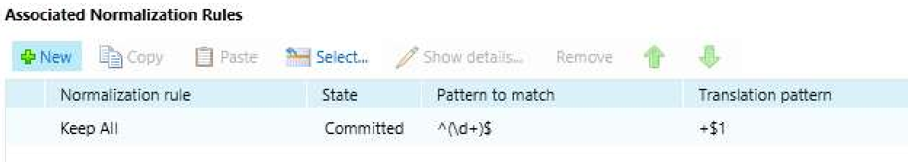
Let’s build a second example.

We only need to dial 7 digits to make a local call. We want to convert the 7 digits to a properly formatted E.164 number. We know we are in the 615 area code. And in order to make it a proper E.164 number, we also need to add a + and the country code.

The user dials 555-1234 with their Skype for Business client.

Make sure you are on the details page for the Global dial plan and click the "New" button in the "Associated Normalization Rules" section. This is seen in Figure 3-7.

Figure 3 – 7



We are now on a screen titled "New Normalization Rule".

The first prompt is for a name. Let's call this rule "7-digit dialing". For "Description", we can add something a bit more descriptive such as "Add +1615 to all 7 digit calls".

We can now build our normalization rule. The first field is called "Starting digits". This field is optional. You would use this field if you wanted this normalization to act on specific starting digits that were dialed. In our example, the user dialed 555-1234. If we wanted this normalization to match on 7 digits and starting with 555, we would enter 555 into the "Starting digits" field. For this example, we'll leave the field blank.

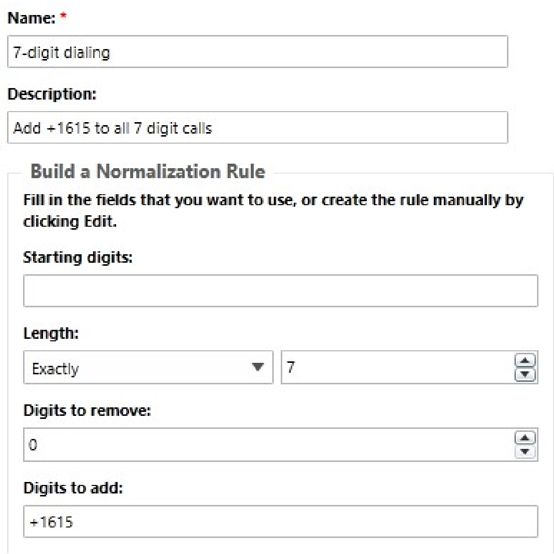
Next it asks us for the length. This is the number of digits we anticipate the user dialing. In this case, we anticipate 7 digits to be dialed. So make sure that the "Length" field exactly matches 7.

The third field asks us if we want to remove any digits. This might be useful in a situation where we expect a user to enter a 9 for reaching an outside line and we want to remove that 9 to get only the actual phone number. This field is optional and we will leave it blank.

Finally, it asks us for digits to add. This field maps to "Translation rule" at the bottom of the Normalization Rules builder. In this case, we want to add +1615 to the 7 digits the user dialed. This will give us our properly formatted E.164 number.

At this point your screen should look like what is seen in Figure 3-8.

Figure 3 – 8



Next we see a section with two greyed out fields called "Pattern to match" and "Translation rule". If you look closely at the Pattern to match you will see that there is a regular expression here that matches what we entered in the fields above. If you look at the "Translation rule" field, you will see that what we added in the fourth field ("Digits to add") has been copied here.

This can be seen in Figure 3-9.

Figure 3 – 9

If you are only creating simple patterns, you can use the first three fields to have regexes automatically created for you. Further, the translation pattern can also easily be created for you. Keep in mind that only simple patterns and translations can be created using the top half of this screen. Anything more complex will require you to manually create your own regular expressions.

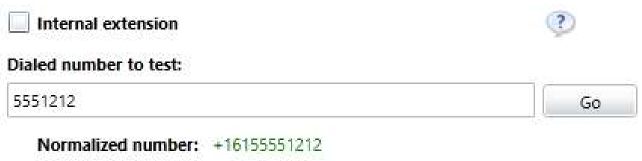
For the sake of this example, we can skip down to the "Internal extension" check box. In general, you should not ever tick this box.

This setting is only relevant when you also use the "External Access Prefix" setting on the Dial Plan. If you tick the "Internal extension" box, this negates the "External Access Prefix" setting on the dial plan. This means that this specific normalization should *not* be evaluated by assuming that the prefix was dialed, e.g. the user isn't expected to dial a 9 first.

For our example, leave "Internal extension" unchecked.

Finally, we can test our new rule. Go ahead and enter any 7 digits to verify that we end up with a properly formatted E.164 number. Figure 3-10 shows that when we enter 7 digits we get a properly formatted E.164 number in return.

Figure 3 – 10



Now that we have verified that this works correctly, go ahead and click the OK button at the top of the screen to return to the main Dial Plan page.

As an exercise, create the normalization rule that would match if a user were to make an International call. As mentioned above, let's assume that an international call is any call that begins with "011" and is at least 10 digits long. The final result should be a dialed number in E.164 format.

Figure 3-11 shows what your screen should look like after completing this challenge. You can see at the bottom of the screen that this rule works as expected.

Figure 3 – 11

Click OK to return to the Global Dial Plan properties page. If you look at the "Associated Normalization Rules" section, you should now see three rules.

Figure 3 – 12

Associated Normalization Rules

New | Copy | Paste | Select... | Show details... | Remove

Normalization rule	State	Pattern to match	Translation pattern
Keep All	Committed	^(\d+)$	+$1
7-digit dialing	Uncommitted	^(\d{7})$	+1615$1
International Call	Uncommitted	^011(\d{6}\d+)$	+$1

The "State" column lets you know if this rule has been committed. In other words, is this rule now in the active configuration? Right now, only the top rule is active.

After committing this change, your users will not immediately begin using the updated Dial Plan. This is because the Dial Plans are sent to the user via in-band provisioning which happens when the user signs in to the Skype for Business client.

In order for the changes to actually take effect, the users will need to sign out of the client and sign right back in.

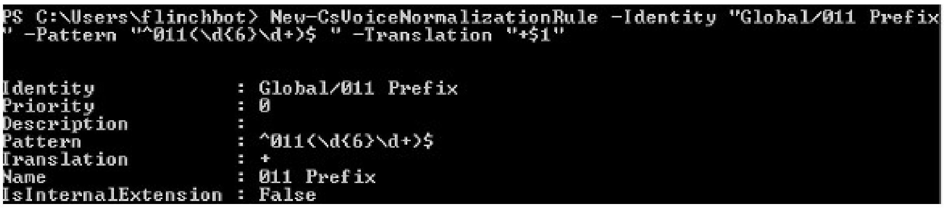
> Note: The Normalizations in Figure 3-12 won't work as we expect them to because of the order in which they are in. Normalization Rule Precedence gets discussed later in this chapter.

If you would like to add a normalization to a dial plan via PowerShell, you need to use the `New-CsVoiceNormalizationRule` cmdlet. The following cmdlet adds the normalization we used in the above example - ^011(\d{6}\d+)$ - to the Global Dial plan. We will name this normalization "011 Prefix".

```
New-CsVoiceNormalizationRule -Identity "Global/011
Prefix" -Pattern "^011(\d{6}\d+)$ " -Translation "+$1"
```

Below is the output of this command.

Figure 3 – 13



```
PS C:\Users\flinchbot> New-CsVoiceNormalizationRule -Identity "Global/011 Prefix
" -Pattern "^011(\d{6}\d+)$ " -Translation "+$1"

Identity            : Global/011 Prefix
Priority            : 0
Description         :
Pattern             : ^011(\d{6}\d+)$
Translation         : +
Name                : 011 Prefix
IsInternalExtension : False
```

If you would like to add the normalization rule in a specific order, use the "Priority" parameter. By default, new rules are added to the bottom of the Dial Plan. If you would like this normalization to be added second in the list, use the following syntax:

```
New-CsVoiceNormalizationRule -Identity "Global/011 Prefix" -
Pattern "^011(\d{6}\d+)$ " -Translation "+$1" –Priority 1
```

Dial plan priorities begin with "0" as the highest priority.

Handling Internal Extensions

Most companies have an internal dialing plan where you only need to dial 3 or 4 digits to be connected to a coworker. It was stated above that in order to simplify your Skype for Business implementation that you should standardize all dialed numbers to E.164 format. That is easily done for calls to the Public Switched Telephone Network (PSTN). But how is that done for calls where the phone number literally only has 4 digits and there is no area code or country code?

In order to handle these calls, the ";ext=" syntax is appended to an E.164 number.

This allows us to add an extension to a phone number. As a real world example, think of when you call a main number and then either ask to be transferred to an extension or you dial the desired extension yourself.

For us to use 3 or 4 digit dialing (or however long your internal extensions are), we append the extension to the end of the pilot number, essentially mimicking what one would have on their business card.

My business card may look like the following:

Figure 3 – 14

In order to map the phone number on the business card to E.164 format, it would look like this:

+16155551212;ext=101

If I had a coworker at extension 102, his E.164 number would be this:

+16155551212;ext=102 and so on for the rest of our company.

By formatting our extensions this way, we can still use 3 digit dialing as well as maintain E.164 standardization. When I dial my coworker on extension 102 that number will get normalized to E.164. Skype for Business will do a reverse number lookup to see if there is a match in the directory to that E.164 number. If a match is found, I will be connected to my coworker.

So now we need to make sure that Skype for Business can find a match for the phone number +16155551212;ext=102.

Let’s make a normalization rule using the example in this section. If a user only dials 3 digits, we will take those 3 digits and append them as an extension to the pilot number (615-555-1212).

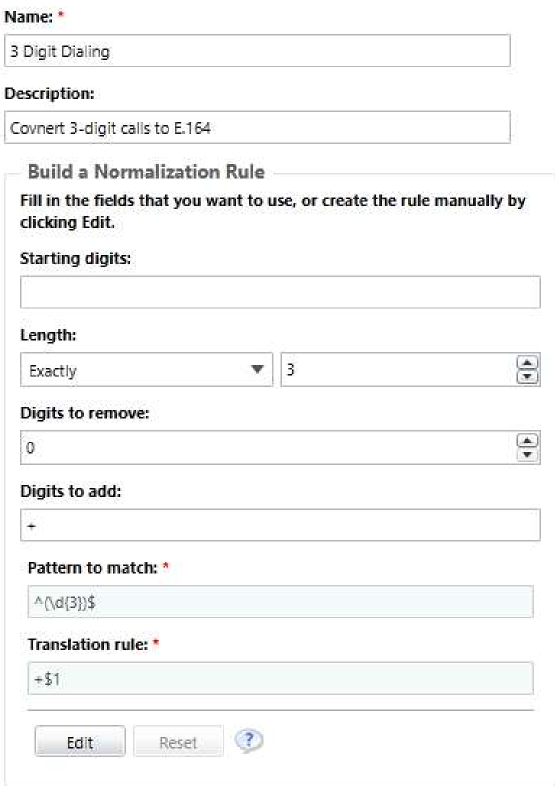
For this example, we are going to have to manually edit the translation rule. The reason for this is because the builder does not support characters such as the semicolon in the “Digits to add:” field.

Create a new Normalization Rule in the Global Dial plan and name it “3 Digit Dialing”. Set the comment to “Convert 3-digit calls to E.164”.

The only value we need to set in the Normalization Rule builder is to set the Length to exactly 3.

This is how it should look:

Figure 3 – 15

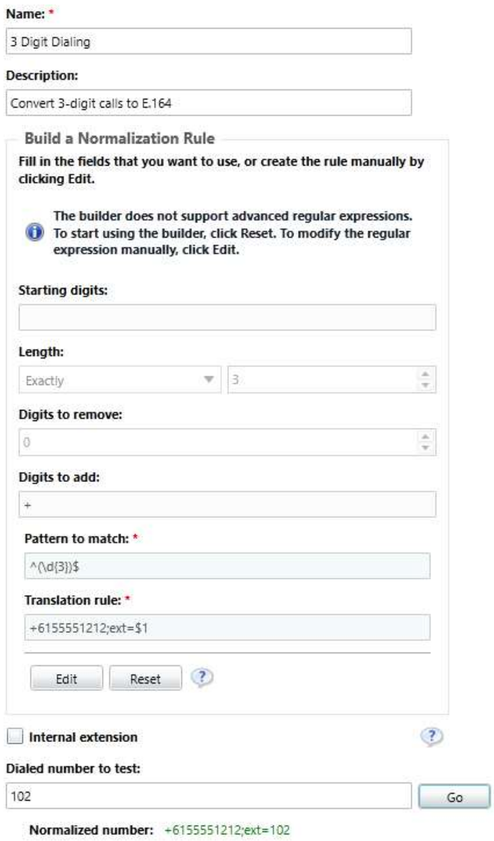


Now we need to edit the "Translation Rule". Click the Edit button at the bottom and a new window pops up. Leave the "Match this pattern" section alone. Edit the "Translation rule" section to look like Figure 3-16.

Figure 3 – 16

After clicking OK, the normalization should look just like it does in Figure 3-17.

Figure 3 – 17



Note that at the bottom of the screen we can see that entering a 3-digit extension does properly convert to a full E.164 number.

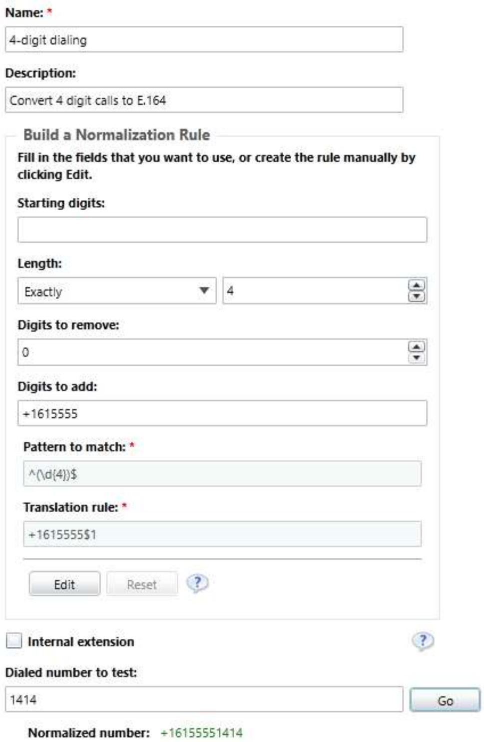
Keep in mind that you will also need to set the LineURI value for the user to be exactly "+16155551212;ext=102". Without this being set the call will not complete as the reverse number lookup will not find a matching Skype for Business user.

You can also implement extension dialing for user that have DID/DDI numbers.

Say my direct dial number is +16155551212 and my coworkers direct inward dial number is +16155551414. We can set up the Dial Plan so that I would only have to dial 1414 to reach my coworker. Most of the time I would pick their name out of the Skype for Business address book and not even bother with the numbers. But if I am using a handset it is much faster to dial my coworker by dialing only 4 digits.

To do this, create a new Normalization Rule in the Global dial plan. Call it "4-digit dialing". Set "Length" to exactly 4. In the "Digits to add" field, enter "+1615555". Now test the number. Figure 3-18 shows the results.

Figure 3 – 18



Advanced Normalizations

As we have seen so far, the Skype for Business Server Control Panel provides us with a straightforward way to create simple dial plans. However, it is very likely that you will need to create normalizations that are not supported by the simple normalization rule builder.

Let's go through an example of an advanced regex and see how we would add it as a normalization rule.

We are replacing our legacy PBXes with Skype for Business. Historically, all of our users had to dial a 9 to make an outside call. With our new Skype for Business infrastructure, there is no need to dial a 9 to get an outside line. However, our users will still be in the habit of dialing a 9 for quite some time. Instead of dropping calls that start with the now-unnecessary 9, let's build a normalization rule that will work regardless of the user dialing the 9 first or not. Let's also assume that this person is dialing an 11-digit long distance number and is using on-hook dialing.

Within the Skype for Business Server Control Panel, go to the Voice Routing tab and open the Global dial plan. From here, click on the "New" button in the "Associated Normalization Rules" section. Give the rule a unique name like "National Long Distance". In the description enter "Allow nationwide long distance with or without a preceding 9".

The regular expression we are going to use is the following: ^9?(\d{11})$

The question mark following the 9 is the regex operator that means the preceding character is optional. Since the Normalization Rule builder doesn't provide anywhere for us to enter this value, we will have to enter the regular expression directly.

Click the edit button near the bottom of this page. Clear out any automatically generated values and type in the regular expression. After you have typed it in, click OK and test the rule. Assuming you didn't make a typo, the test should pass.

Figure 3-19 shows the relevant part of this normalization rule:

Figure 3 – 19

The example given above is a fairly simple yet very useful example of the power of regular expressions with normalization rules.

Normalization Rule Precedence

As you begin building the normalization rules, it is very possible that you may have one or two normalization rules that conflict with one another. Even the best planning might not be able to avoid this situation.

For example, say you have a 4-digit dial plan defined for a range of numbers between 1000 and 1500. Within that range, you have an extension that needs to be handled differently than any other number in that range. You could certainly make multiple normalization rules to work around this or even make an advanced regex to work around this issue. But if you want to keep it as simple as possible, you could create one normalization rule for the entire range and then another normalization rule for the special-case extension.

Assuming you went with the simple approach, what happens if that specific extension is dialed? How does Skype for Business decide which normalization rule to use?

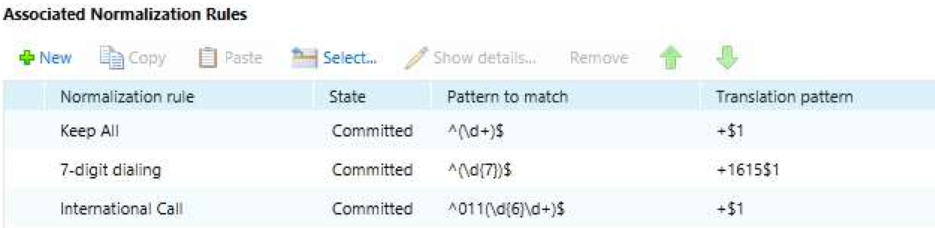
The answer is fairly simple. When looking at the “Associated Normalization Rules” list, the normalization rule that is closest to the top of the list takes precedence over a rule below it.

If you look at Figure 3-20, the "Keep All" rule will always take precedence over the two rules beneath it.

Similarly, the "7 Digit Dialing" rule will take precedence over the "International Dialing" rule.

If you want to change the precedence, simply click either of the green arrows on the upper right and move a selected rule up or down to change its precedence.

Figure 3 – 20


Associated Normalization Rules

New Copy Paste Select... Show details... Remove

Normalization rule	State	Pattern to match	Translation pattern
Keep All	Committed	^(\d+)$	+$1
7-digit dialing	Committed	^(\d{7})$	+1615$1
International Call	Committed	^011(\d{6}\d+)$	+$1

There is one rule that you probably always want to keep at the top and that is the rule for emergency dialing. You never want to accidentally have someone dial an emergency number (e.g. 911 in the United States) and not have that number route out to the PSTN correctly. By keeping that rule at the top you assure that there will never been an issue with precedence handling that call in an unexpected manner.

> For the United States, the following regex should work for you for handling calls to 911 (and also to less emergency-related numbers like 411).
>
> ^9?([3-9]11)$
>
> With a translation pattern of simply: $1
>
> Be sure to test that this dials out correctly. There is plenty to know about E9-1-1 that is out of the scope of this book. Before rolling out Enterprise Voice take time to learn what the requirements are for dialing emergency services in your area.

There is one last thing to consider with Dial Plan normalizations and precedence. Let's say you have three normalizations in your dial plan that do not overlap or conflict with each other in any way. It then doesn't matter which normalization is above the others.

If you are the type of person who likes squeezing out every bit of performance optimization, let me mention this tip: place the most-used normalization of the three highest in the list. Then the second-most used normalization in the middle with the least-used normalization at the bottom. This will help reduce processing time as the most-used normalization will be evaluated before the less-often used normalizations. This reduced processing time can lead to decreased call connectivity wait times. This is especially noticeable in dial plans that have dozens or - the horror! - hundreds of normalizations.

Assigning Dial Plans

Now that we have our dial plans created, how do we use them? If the dial plan was designed for a device, it will automatically be applied at the Site level if a Site Dial Plan exists. Otherwise the Global Dial Plan will be applied.

Users will also receive a default scope just like a device will. However, I encourage you to create User scopes and assign them to users in order to have maximum flexibility. You have to manually assign the dial plan to each user. This can be done with either the Skype for Business Server Control Panel or via PowerShell. To do it via the Control Panel, follow these steps:

Open the Skype for Business Server Control Panel and click on the Users tab.

1. Enter a search term or simply click Find to return the first 200 users.
2. Either double-click on a user or highlight a user and select Edit/Show Details.
3. If necessary, change the Telephony option to Enterprise Voice. Some new options will now become available.
4. Change the Dial Plan Policy to the dial plan you would like to use. This will have to be a User scope Dial Plan. If you do not select a User dial

plan and leave it at automatic, Pool, Site, or Global voice policies are automatically applied based on precedence.

Note that you can assign dial plans or voice policies in bulk via the Control Panel by selecting multiple users then clicking the Action menu. From there select the "Assign Policies…" link. This brings up a screen permitting you to assign policies to all selected users at once.

If you would like to assign the Dial Plan via PowerShell, the `Grant-CsDialPlan` cmdlet will do that for you. For example, the following will assign a dial plan named "User Dial Plan" to a user named Ken Myer:

```
Grant-CsUserDialPlan –Identity "Ken Meyer" –PolicyName "User
Dial Plan"
```

If you want to assign a dial plan to all of your users and the dial plan is named 'User Dial Plan", then you can use the following one-liner.

```
Get-CsUser | Grant-CsDialPlan –PolicyName "User Dial Plan"
```

You can find more information on configuring users in chapter 10.

External Access Prefix; Internal Extensions

I mentioned in step 8 of the "Creating a Dial Plan" section that, generally speaking, you should never set this option. I'll now go into detail about how these two settings work and the advantages and drawbacks provided.

Setting a value for "External Access Prefix" triggers the following logic:

If the number dialed begins with the prefix, then:

1. The Skype for Business Client removes the prefix. It then attempts to find a match among the normalization rules that are for external numbers (i.e., normalizations not marked as Internal Extension)
2. If there is no match, the client keeps the prefix. It then attempts to match all the normalization rules that are marked as Internal Extension.

3. If there is still no match, then the client keeps the prefix. It then attempts to match all the normalization rules that are for external numbers

So here's the thing about the rules above: If you dial an Internal Extension with an External Access Prefix, the call will fail. Let's step through those 3 points. Let's say the External Access Prefix is set to 9 and a valid Internal Extension of one of your users is 1234.

I dial "91234" into my Skype for Business client. What happens?

1. The 9 is removed and there is an attempt to match any normalization rule that is *not* an Internal Extension. Since 1234 will hit an Internal Extensions normalization, there will be no match.
2. The client keeps the prefix and tries to match 91234. The extension is 1234, so there will be no match.
3. The client keeps the prefix and then tries to match "external extensions", i.e., normalizations without Internal Extension set. There will be no match.

So what does this really mean? If you have External Access Prefix set, then any rule where you tick the "Internal Prefix" means that the 9 is not to be dialed by the user. In other words, it negates the 9 as a prefix.

So why tick either box? There is only one good reason…

Off-hook Dialing

Off-hook dialing is the way old-school phones work. You pick up a handset (or turn on the speakerphone) and start punching in numbers. As the numbers are dialed, the call gets processed in real-time.

This is different from on-hook dialing which is what the Skype for Business desktop client uses. It's also how your mobile phone works. You dial the number then hit a button to place the call. So you, the end user, controls when the call is placed.

When dialing off-hook, an inter-digit dialing delay is used to determine when the call should be placed. The idea is that if there is a normalization match and if the user hasn't dialed an additional number within 1.5 seconds, then the user is done entering the number, the matched normalization rule is applied, and the call gets placed.

If there is no matching normalization, the phone waits for more numbers to be entered.

The problem occurs if a user off-hook dials part of a number, then pauses for a few seconds to look up the rest of the number. It's possible the call will be placed based off the first few numbers entered. For example, if the user dialed 4 digits and there is a normalization rule that matches those 4 digits, the call will be placed (assuming there is a match with a 4-digit normalization in the Dial Plan).

To avoid the situation of off-hook calls being placed too quickly, you tick both the "Internal Extensions" box for the 4-digit normalization and you enable "External Access Prefix" in the Dial Plan properties.

Then you instruct your users that if they are dialing an external number to use a 9 and if they are dialing an internal extension to omit the 9. Problem solved.

Except in my experience this situation rarely arises. Like never. I've never received a complaint by some upper manager complaining that he keeps calling the wrong number. I know others have heard this complaint but even to them it's rare.

Quite honestly the only time I've had to deal with this was at work where someone (probably me) ticked the "External Access Prefix" box. A few months later we couldn't figure out why we were having calling issues. A new member on our team asked us why the "Internal Extensions" box was ticked. We unticked it; problem solved.

Another thing to keep in mind is that when testing normalizations Skype for Business does *not* account for the external access prefix in its testing logic - no matter if it is for an "internal" or "external" normalization.

Another disadvantage of the External Access Prefix setting is that this is a client-side parameter. This means that only Skype for Business clients will honor this when normalizing numbers. If a call comes in from an SBC, then server-side rules are in effect. The call from the SBC may not match if it is dialed with a leading External Access Prefix.

So what happens if your users are migrating from a PBX and they are used to dialing a 9 before every external call. Shouldn't you use the External Access Prefix in this scenario?

The easiest and most flexible way is to add "9?" before all of your normalizations. Adding this to your normalizations means the 9 is optional. If they dial it – fine, the call gets normalized. If they don't dial the 9 first, then no problem - the call still gets normalized. Figure 3-19 (in the "Advanced Normalizations" section) shows an example of this. Adding a "9?" into your site-level normalizations will also then apply to calls coming in from a PBX or SBC.

I never use these settings. However, if a situation arises, feel free to use these settings. It's certainly not at all "wrong" to use them. To me it adds complexity that I can work around using other methods.

If you want to enable "External Access Prefix" via PowerShell then you need to set the *IsInternalExtension* parameter. Below I am enabling the setting for an existing normalization rule named "4 Digit" which is in the Global Dial Plan.

```
Set-CsVoiceNormalizationRule -Identity
"Global/4 Digit" -IsInternalExtension $True
```

If I want to set "9" as the "External Access Prefix for the Global Dial Plan, I run the following.

```
Set-CsDialPlan -Identity Global -ExternalAccessPrefix
'9' -OptimizeDeviceDialing $True
```

According to TechNet, you must set *OptimizeDeviceDialing* to true for ExternalAccessPrefix to take effect.

In the description of "External Access Prefix", TechNet says:

The OptimizeDeviceDialing parameter must be set to True for this value to take effect

And for the description of OptimizeDeviceDialing, TechNet says:

Setting this parameter to True will apply the prefix in the ExternalAccessPrefix parameter to calls made outside the organization. This value can be set to True only if a value has been specified for the ExternalAccessPrefix parameter.

https://technet.microsoft.com/en-us/library/gg398644.aspx

If I want to remove the "External Access Prefix" setting for a dial plan named "Nashville", I run the following:

```
Set-CsDialPlan -Identity Nashville -ExternalAccessPrefix $Null
-OptimizeDeviceDialing $False
```

Summary

A lot of information was presented in this chapter. A solid understanding of dial plans is crucial as you design and implement your Enterprise Voice deployment. The main thing to remember is that a Dial Plan is simply a way to standardize the phone numbers dialed by users (and devices). It has no other purpose or role. It is best practice to normalize your numbers to E.164 format.

There are four different scopes:

- User
- Pool
- Site
- Global

The User scope takes precedence, followed by Pool, Site, and Global scopes.

Within a given Dial Plan, the normalization rule closest to the top takes precedence over rules beneath it.

Chapter 4 – Voice Policies

Voice Policies are fairly straightforward. A Voice Policy simply controls the calling features permitted for a user and - along with PSTN Usages and Routes - what numbers the user can dial. Examples of these features are the ability forward or transfer a call as well as not being able to place international calls.

In this chapter, all of the individual settings for a Voice Policy will be described as well as any client settings that a given Voice Policy impacts. This chapter will not discuss how to limit the calls a user is permitted to make. That will be discussed in the next chapter on Routes.

In order to access Voice Policies, open the Skype for Business Server Control Panel, click on the Voice Routing option, and then click on the Voice Policy tab. You should see a screen similar to the one in Figure 4-1.

Figure 4 – 1

Editing a Voice Policy

In order to edit a Voice Policy, you can either double-click on it in the list, or highlight the policy and select Edit from the menu. From within the Edit menu, select the "Show Details…" option.

If you would like to see which Voice Policies you have via PowerShell then you need to use the `Get-CsVoicePolicy` cmdlet.

Run on its own it returns all of the committed voice policies and their settings. If you append the *–identity* parameter you can then see the details for a specific voice policy. The command below shows the settings for a voice policy named "CommonAreaPhone".

```
Get-CsVoicePolicy –identity "CommonAreaPhone"
```

In order to edit a voice policy via PowerShell you use the `Set-CsVoicePolicy` cmdlet. Assuming we have a voice policy named "CommonAreaPhone" and we want to enable the delegation option, we would use the following command:

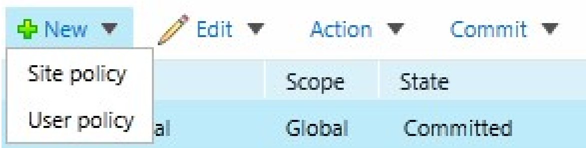
```
Set-CsVoicePolicy –identity
"CommonAreaPhone" -EnableDelegation $True
```

Creating a New Voice Policy

Like Dial Plans, you can create Voice Policies with various scope levels. By default, you will have a Global Voice Policy. However, you can create a Site or User Voice Policy. Also like Dial Plans, user Voice Policies take precedence over any site policies and site policies take precedence over global policies.

In order to create a new Voice Policy, simply click the "New" menu item and select the scope option you would like to use. This can be seen in Figure 4-2.

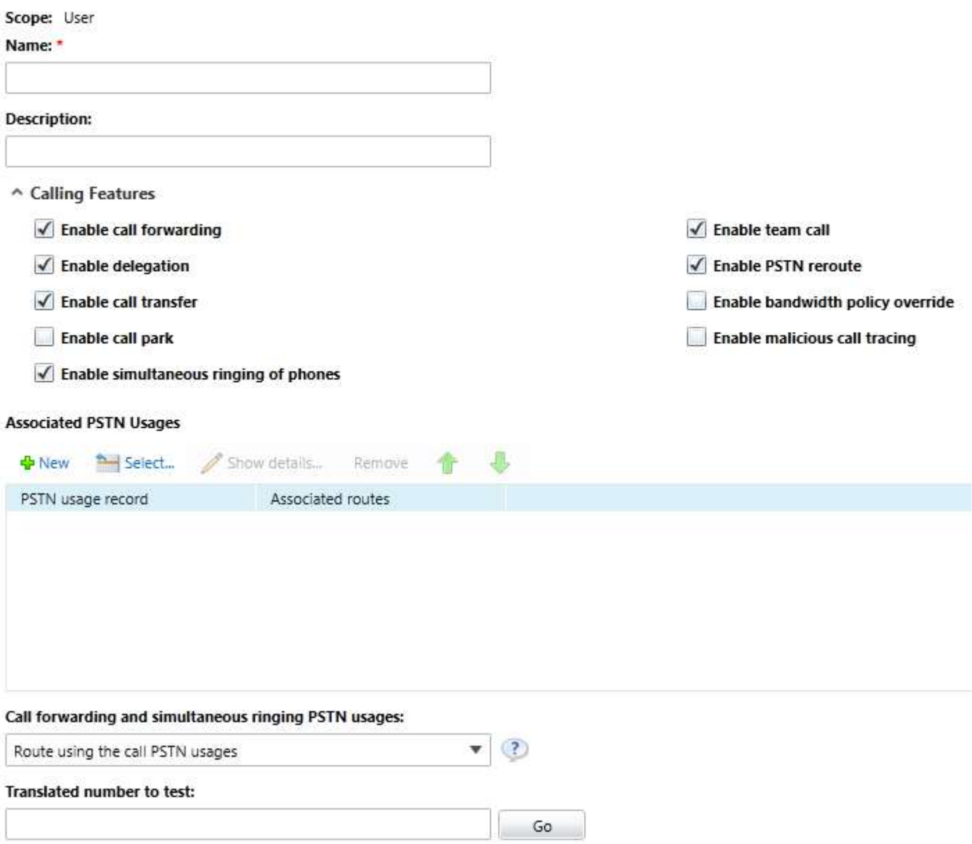
Figure 4 – 2



If you select the Site policy, you will be asked to select a site to which this policy applies. Select a site and click OK. This step is skipped if you select User policy.

Regardless of which scope you selected, you will see the screen shown in Figure 4-3. This screen already has default values selected for you and those settings will be explored in detail.

Figure 4 – 3



You can also create a voice policy using PowerShell and the `New-CsVoicePolicy` cmdlet. Below is a sample command which creates a new Voice Policy called "CommonAreaPhone" and disables call forwarding, simultaneous ring, and call transfer.

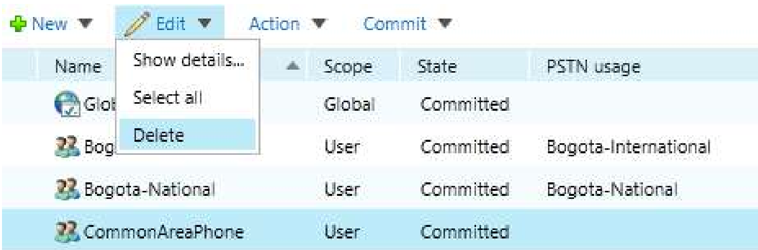
```
New-CsVoicePolicy –Identity
'CommonAreaPhone' -AllowCallForwarding $False -AllowSimulRing
$False -EnableCallTransfer $False
```

> The above Voice Policy won't do much until we attach a PSTN Usage to it. I will discuss PSTN Usages in chapter 6.

Deleting a Voice Policy

In order to delete a Voice Policy, open Skype for Business Server Control Panel and navigate to the Voice Routing link. From there, select the Voice Policy tab. At this point you will see a list of Voice Policies. Highlight one of the policies and select Edit and then Delete from the menu. This is seen in Figure 4-4.

Figure 4 – 4



The state of that Voice Policy will change to say "uncommitted". At this point, select the "Commit" menu item and then select "Commit All". A summary screen appears. Click "Commit" at the bottom of this screen and the Voice Policy will be deleted.

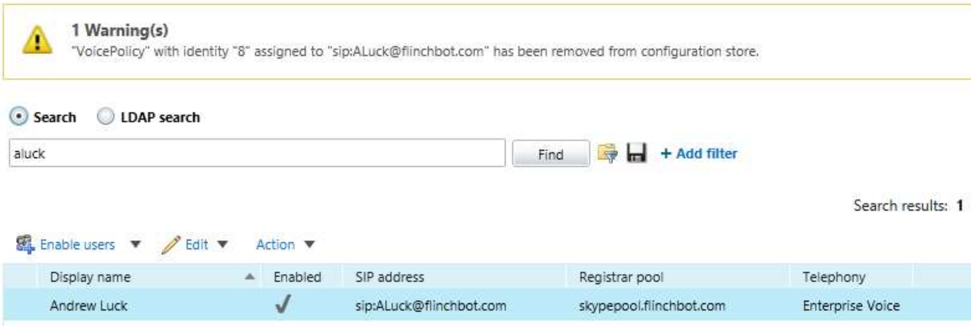
If you would like to use PowerShell to delete a Voice Policy then you need to use the `Remove-CSVoicePolicy` cmdlet. If we have a Voice Policy named "CommonAreaPhone" that we want to remove, we would use the following command:

```
Remove-CsVoicePolicy –identity "CommonAreaPhone"
```

Note that you can delete a User-level voice policy if it is assigned to a user.

If you delete a voice policy while it is assigned to users, those users will be reset to "Automatic" and will take whichever policy is next in the order of precedence (either Site or Global). As seen in Figure 4-5, you will be notified of this at the top of the Skype for Business Server Control Panel when you next search for a user (or users) who had this policy applied to them.

Figure 4 – 5



The only way to get rid of the warning is to assign a different Voice Policy to the user. Simply selecting "Automatic" will not remove the message. If you want the user to have an automatically applied Voice Policy and you do not want to see the warning message, set the Voice Policy to $Null in PowerShell.

```
Grant-CsVoicePolicy -identity aluck -PolicyName $Null
```

Calling Features

Voice policies are used to set the calling features available to a given user and which numbers users are allowed to dial. The following sections will go into detail on the available calling features.

Enable Call Forwarding

This feature allows a user to forward their extension to another device such as their mobile phone.

You may want to disable this feature for a lobby phone.

This feature is enabled by default.

Enable Delegation

Delegation is the ability to have someone else make or receive calls on your behalf. The classic example of this is an executive who has an Administrative Assistant. Quite often the Administrative Assistant will answer a call on behalf of the executive in order to screen the calls the executive receives.

This feature is enabled by default.

Enable Call Transfer

This feature allows a user to forward a call to another user or device. As an example of this feature, think of an operator who receives many calls over the course of the day. Callers ask to be connected to an employee and the operator forwards that call to the requested person.

You may want to disable this feature for a lobby phone.

This feature is enabled by default.

Enable Call Park

You've certainly been to a large store and heard a voice over the loudspeakers say something along the lines of "Call for Plumbing Department on line 115". This is an example of call park. Rather than blindly forwarding the call to the Plumbing Department and hoping someone is standing near the ringing phone, the call is put into a "call orbit". The call is now essentially put on hold and can be retrieved by dialing the extension to which the call was parked.

Call Park will be discussed in detail in Chapter 8.

This feature is disabled by default.

Enable Simultaneous Ringing of Phones

This feature enables the user to have inbound calls ring not only on their primary Skype for Business client or device but also to a secondary device such as a mobile phone. To the "always on" employee, this is a great way to avoid missing calls no matter where that employee is. To the employee who prefers to separate his or her work life from his or her private life, this is a feature he or she could gladly do without.

This feature is enabled by default.

Enable Team Call

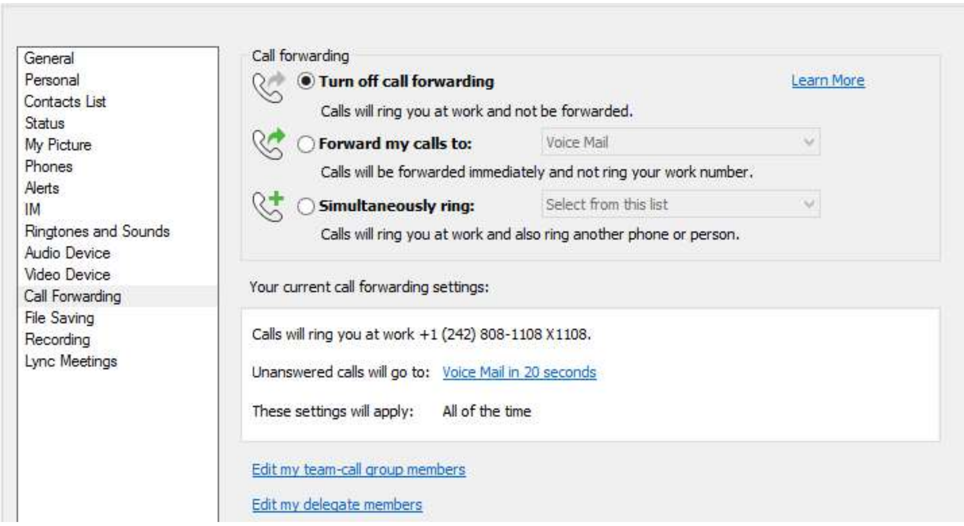
Team Call is the ability of other members in your team to answer your phone. For example, imagine a sales office with 4 salespeople. One of the salespeople just left to get coffee and their phone rings. There is a good chance that this is a customer calling. One of the other salespeople should probably answer the phone in order to provide top-notch customer service. In the classic case, this salesperson would have to stand up from their desk and walk over to the ringing phone and answer it.

However, if Team Call was configured, that salesperson would not have to stand up at all – they could answer the ringing phone directly via their Skype for Business client.

Let's quickly go over the steps required to set up a Team Call. First, this is a client-side setting so it is set up via the Skype for Business client software and not on the server.

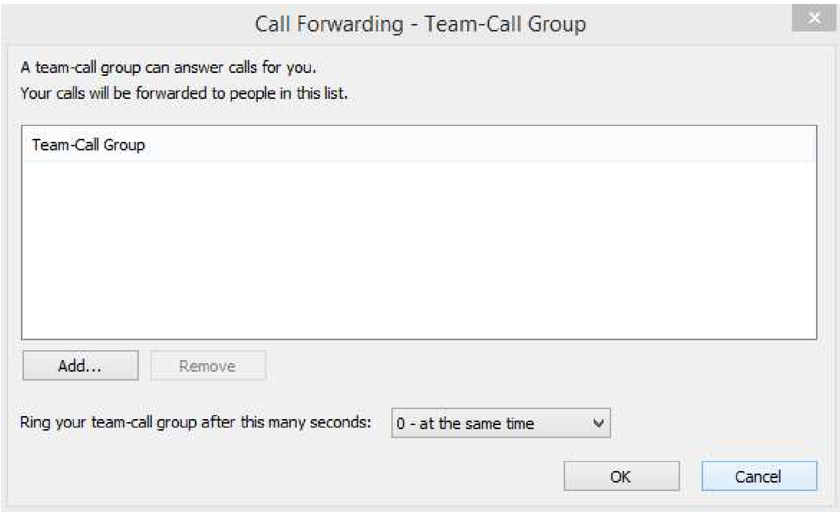
To set up a Team Call, open the Skype for Business Client Options window and select the Call Forwarding Section. This is seen in Figure 4-6.

Figure 4 – 6



At the bottom of Figure 4-6 you will see a link titled "Edit my team-call group members". Clicking on this link brings up the screen seen in Figure 4-7.

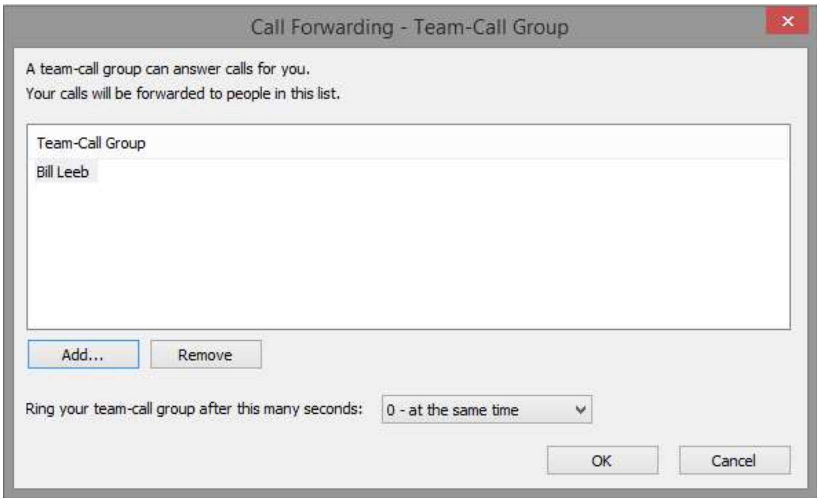
Figure 4 – 7



From this screen you can click the "Add.." button to add the members of your team. The screen in Figure 4-8 also lets you set a delay before members of your team call group hear their clients ring and receive a toast message. For example, if you set this to 5 seconds then you have 5 seconds to answer a call before the rest of your team gets notified of that call.

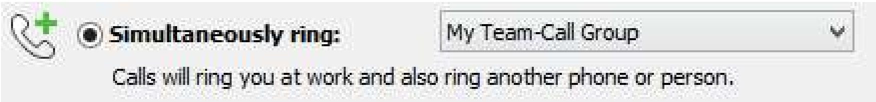
In Figure 4-8, you can see that a user was added to the team call group and the ring delay was set to 0 seconds (i.e. simultaneously ring the Team Call group).

Figure 4 – 8



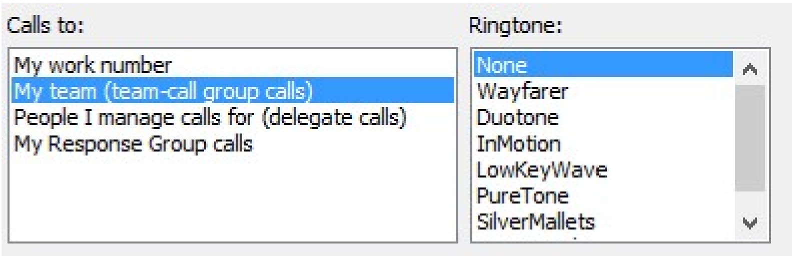
Once this has been configured, be sure to set the Simultaneously Ring setting on the Call Forwarding page to "My Team-Call Group". This is seen in Figure 4-9.

Figure 4 – 9



If you set this up, you may not want to hear multiple desktops ringing every time a call comes in. It could cause quite a cacophonous sound in your office. In order to avoid this, you'll have to set each members client to not play a sound when a team call arrives. In order to do this, open up Skype for Business Options and navigate to the "Ringtones and Sounds" settings. Click on "My team (team-call group calls)" and then select "None". You can see this setting in Figure 4-10.

Figure 4 – 10



Enable PSTN Reroute

This setting allows calls from one enterprise voice enabled user to another enterprise voice enabled user to be re-routed over the public switched telephone network (PSTN). This re-routing would occur in the case of a WAN failure or if the WAN is congested. This setting is only used if Call Admission Control is enabled.

This setting is enabled by default.

Enable Bandwidth Policy Override

Microsoft Skype for Business supports a feature known as "Call Admission Control" (CAC). This is a feature that is used to assure a quality connection across a WAN between two Skype for Business users.

Let's assume CAC is configured and enabled and that a WAN link between two users is congested. If one of the users tries to call the other over this WAN link, Skype for Business will not permit the call to be established. Since there is not enough bandwidth between the two sites, the call would be a pretty miserable experience.

Technically, it is the receiving party who rejects the call not the calling party.

There may be certain users who should always receive inbound calls across a congested WAN link. For those users, you can enable this setting and Skype for Business will try to establish the call every time regardless of how busy the WAN link may be.

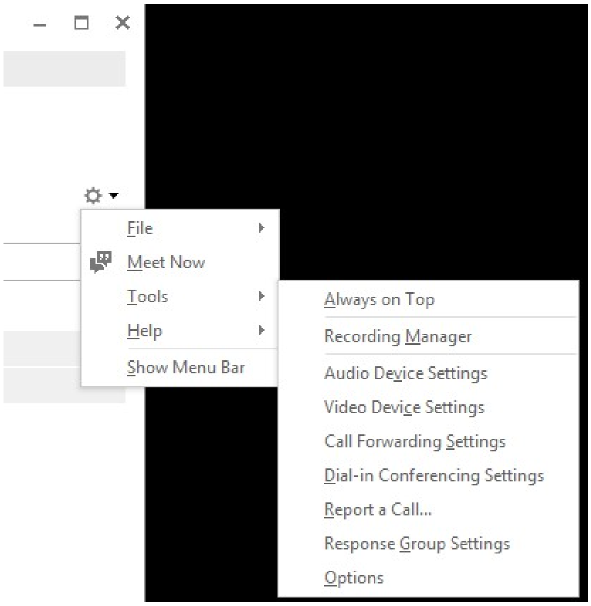
This feature is disabled by default.

Enable Malicious Call Tracing

This feature allows the client to flag a call as malicious (such as a bomb threat) and this call will be tagged in the Call Detail Records (CDR) of this call. Note that if a user flags a call as malicious that no one is notified. It simply sets a value in the CDR database.

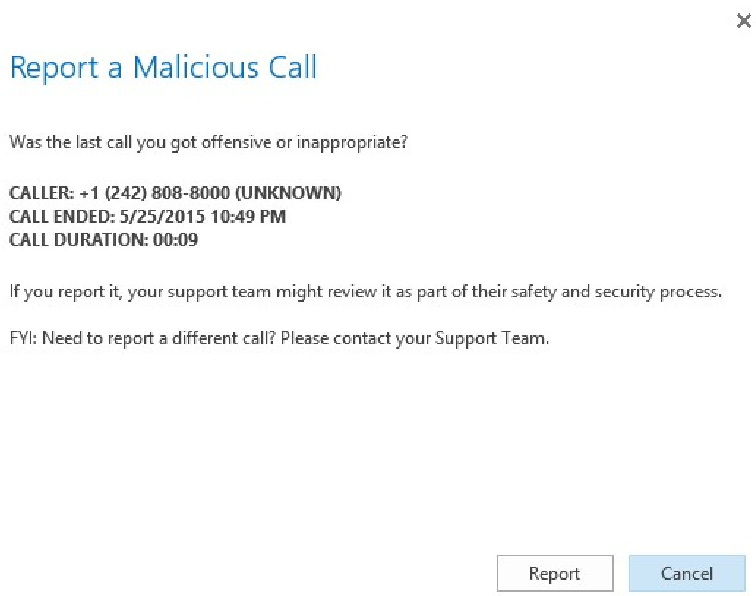
For a client to flag a call as malicious, the user has to terminate the call. Immediately after terminating the call, the user goes to the Skype for Business client and hits the dropdown arrow next to the options icon. From this menu, the user selects Tools and then "Report a Call…". This can be seen in Figure 4-11.

Figure 4 – 11



After selecting "Report a Call…" a new window pops up showing a summary of the previous call. This is seen in Figure 4-12. The user can then click OK to submit the call as a malicious call or click cancel.

Figure 4 – 12



If the user clicks OK, the call is then tagged in the CDR database as a malicious call. In order for the administrator to see calls that have been reported as malicious, he or she needs to run a report against the CDR database.

A video of how all of this this works can be viewed at this link:

http://blogs.technet.com/cfs-file.ashx/__key/communityserver-blogs-components-weblogfiles/00-00-00-84-94/1323.Malicious-Calls-Tracing.wmv

(http://tinyurl.com/CallTraceVid)

The video is for Lync 2010 but it is still valid for Skype for Business Server.

This setting is disabled by default.

Assigning Voice Policies

Now that we have our voice policies created, how do we use them? If you only have a Global voice policy, then it will automatically be assigned to all of your users and nothing more needs to be done.

I encourage you to create User scopes and assign them to users in order to have maximal flexibility by not relying on a static Global Voice Policy. You have to manually assign a User scope to each user. This can be done with either the Skype for Business Server Control Panel or via PowerShell. To do it via the Control Panel, follow these steps:

1. Open the Skype for Business Server Control Panel and click on the Users tab.
2. Enter a search term or simply click Find to return up to 200 users.
3. Either double-click on a user or highlight a user and select Edit/Show Details.
4. If necessary, change the Telephony option to Enterprise Voice. Some new options will now become available.
5. Change the “Voice policy” setting to the Voice Policy you would like to use. This will have to be a User scope Voice Policy. If you do not select a User voice policy and leave it at automatic, Site or Global voice policies are automatically applied based on precedence.

If you would like to set the Voice Policy via PowerShell, the `Grant-CsVoicePolicy` cmdlet will do that for you. For example, the following will assign a voice policy name "User Voice Policy" to a user named Ken Myer:

```
Grant-CsVoicePolicy -Identity "Ken Myer" -PolicyName "User
Voice Policy"
```

If you want to assign a Voice Policy to several users, you can string a few PowerShell cmdlets together. In this example, we want to apply a Voice Policy named "User Voice Policy" but only to users who are already enabled for Enterprise Voice:

```
Get-CsUser -filter {EnterpriseVoiceEnabled -eq $True} | Grant-
CsVoicePolicy -PolicyName "User Voice Policy"
```

Summary

Once you understand what each calling feature means, Voice Policies are fairly simple and straightforward.

There are three different scopes:

- Global
- Site
- User

The User scope takes precedence, followed by Site and then Global scopes.

Voice Policies are attached to a PSTN Usage and we did not cover that in this chapter. However, they are covered in chapter 6.

Chapter 5 – Routes

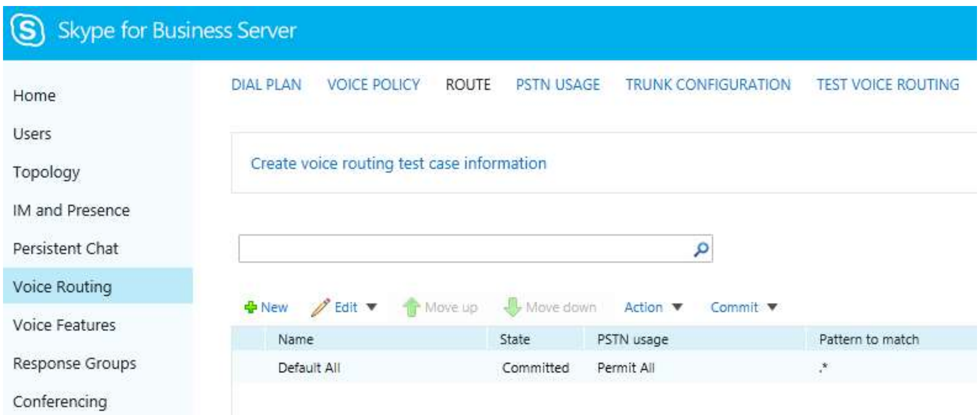
Routes are the portion of Skype for Business that connects dialed numbers with an associated trunk (or trunks).

By limiting the calls allowed to reach a gateway you can ultimately control if users can make international calls or are limited to only local calls. Routes are also used with Least Cost Routing.

In this chapter, all of the settings for creating a route are defined in detail.

In order to access Routes, open the Skype for Business Server Control Panel, click on the Voice Routing option, and then click on the Routes tab. You should see a screen similar to the one in Figure 5-1.

Figure 5 – 1



Creating a New Route

Unlike Dial Plans and Voice Policies, there are no scope levels with routes. You can consider that all routes are “global” by nature.

In order to create a new route using the Skype for Business Control Panel, simply click the “New” menu item and you will be brought to the “New Voice Route” page

You will see the screen seen in Figure 5-2. Begin by giving your route a sensible name such as “Local Calls”, “International Calls”, or “DE-Berlin-Local”. You should provide a good description of why this Route was added in the “Description” box. While this is technically optional, be a good administrator and add some text here.

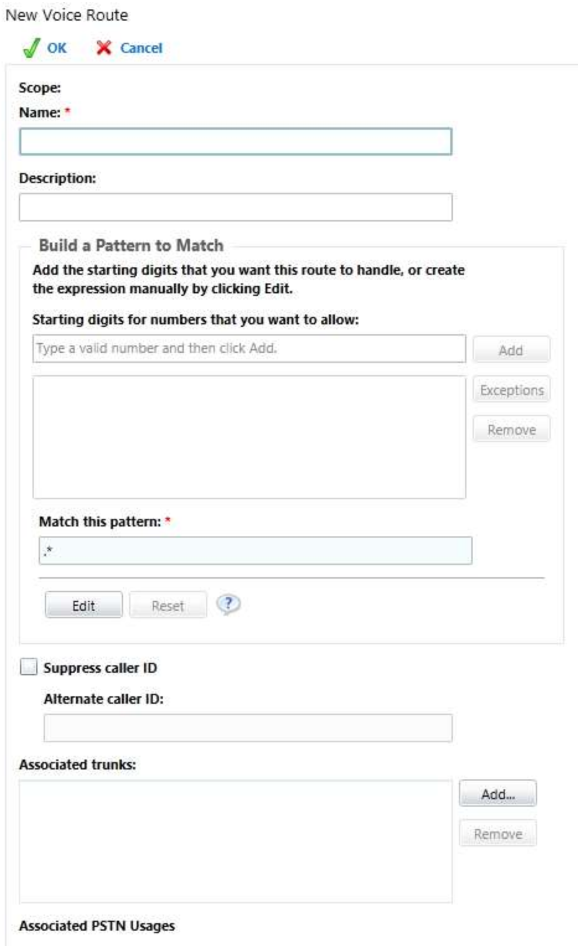
Route Patterns

The “Build a Pattern to Match” section should look familiar. It’s the same basic thing seen when creating Dial Plans. However, this one has only one field to assist in creating regular expressions. If you are not familiar with regular expressions, please review Appendix 1.

It is this “Build a Pattern to Match” section that provides you a lot of power in call handling. Here is where we can limit calls to local calls only or limit the ability to dial International calls. You can also use it to define which number should be routed to a PBX.

The default pattern is ”.*” which means that anything will be permitted. ”.*” is the wildcard in regular expressions. However, you can add any valid regular expression in this field. Let’s go through some examples.

Figure 5 – 2



Assume that you have an office in Nashville, Tennessee in the United States. The country code for the United States (and most of North America) is 1. The area code is 615.

If you set up your dial plans correctly, when a user dials 7 digits, the call is converted to a full E.164 number. So if the user dials “5551212” the Dial Plan should convert this to +16155551212. It is here where standardizing calls to E.164 really pays off as all of the called numbers sent to the Routes section are all normalized.

If we want to limit calls to only the Nashville area, we can create a regular expression that matches only calls starting with +1615. To add this pattern, just type "+1615" in the "Starting digits for numbers that you want to allow" field.

Figure 5 – 3

After adding "+1615" it is added to the list of permitted numbers. Also note that the "Match this pattern" field is automatically updated with a regular expression reflecting the addition of "+1615".

Figure 5 – 4

If there are several area codes that are considered local, you can add those. You should also add support for emergency numbers. Figure 5-5 shows the results of adding a few more entries.

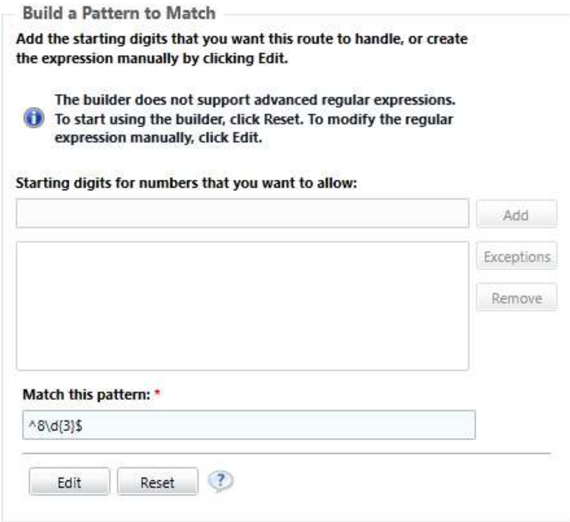
Figure 5 – 5

If you have more advanced needs, you can enter a valid regular expression directly into the "Match this pattern" field. Suppose you want to create a route to a PBX. If you only add an "8" to the "Starting digits for numbers that you want to allow" field, you may inadvertently allow calls to Cambodia, Laos, or North Korea whose country codes all begin with 8.

As such, we want to allow calls that start with an 8 but are only 4 digits long. Therefore, we need to add a custom regular expression into the "Match this pattern" field. To limit calls to 4 digits beginning with 8, we need to add this Regular expression: ^8\d{3}$

Click the Edit button under the "Match this pattern" field and enter the number. Below is the example for limiting calls to the PBX.

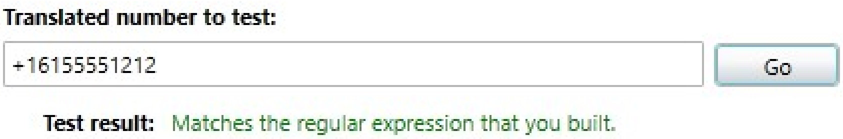
Figure 5 – 6



In order to test if the regular expression is permitting or blocking calls correctly, you can run a quick test. Scroll down to the bottom of the "New Voice Route" page. You will see a new section entitled "Translated number to test". In here, enter the phone number that you expect to send to the route. Keep in mind that this is usually an E.164 formatted number so be sure to add the plus.

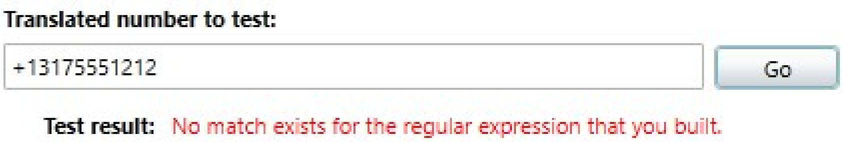
Going back to our example of limiting calls to only Nashville, Tennessee, we can see that the created regular expression works correctly.

Figure 5 – 7



Be sure to also test a number that you expect to fail.

Figure 5 – 8



Overwriting Caller ID

By default, the LineURI associated with your users will be the caller ID value. However, if you instead wish to have every caller ID show your company's main number, you can change that value here. Tick the "Suppress caller ID" value and enter a new value here.

Figure 5 – 9

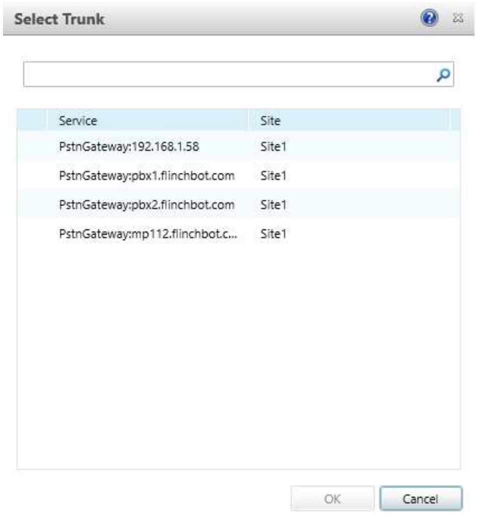
You can add the + sign if you want but some phones don't support displaying the + sign. Note that you can only change the displayed Caller ID number. You cannot change the name associated with the caller ID value in Skype for Business Server. You will have to work with your telco to customize the caller ID name.

Associating Trunks to Routes

As a call is forwarded to a route, it gets evaluated against the regular expression defined in the "Match this Pattern" field. If the called number matches the pattern, it can then be forwarded to the appropriate trunk. You select which trunk(s) will be used by adding them to the "Associated trunks" section. To do this, click the "Add" button and select the Trunk from the list. Note that you have to add Trunks via Topology Builder before they will show up in this list.

After clicking the "Add" button, a list pops up. You can select one or several Trunks. After selecting the Trunks, click the OK button and they are added to the list.

Figure 5 – 10



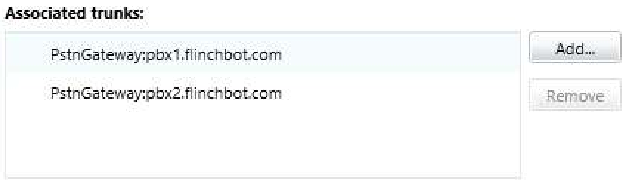
If you look at Figure 5-11, you will see that two trunks have been added.

There is something special to note here. When adding two trunks, this is **not** failover routing. Skype for Business will begin using both trunks in a round robin fashion. So the first call goes to PBX1, the second call goes to PBX2, the third call goes to PBX1, etc. If Skype for Business detects that one of the PBXes is down, it will forward all calls to only one Trunk until the other Trunk (PBX) is returned to service. Once Skype for Business determines that the other Trunk is working, it will then return to sending the calls to both Trunks in a round robin fashion.

If you want failover routing, where the calls get routed to another location in the event of a site failure, then that is configured with PSTN Usages. PSTN Usages are covered in depth in chapter 6.

In general, do not add a second trunk from a remote location to a route. Every time that second trunk gets used the call will originate from that remote location which could incur additional toll charges.

Figure 5 – 11



Associated PSTN Usages

The “Associated PSTN Usages” section is where you can select which PSTN Usage to use. While PSTN Usages are covered in depth in the next chapter, here is a quick description of a PSTN Usage.

A PSTN Usage connects a Voice Policy with a Route (or Routes). How do you then limit a user to only make local calls? You create a mapping between the user-assigned Voice Policy via a PSTN Usage to the Route that limits local calls.

For detailed information, read chapter 6 on PSTN Usages.

> If you look forward to Figure 5-14, you will notice “Move up” and “Move down” arrows. In general, these don’t do anything and can be ignored. The only time the priority of a route matters is if you have a PSTN Usage with two routes that both evaluate to the same Regular expression – i.e., there is a tie on which route to choose. In only that specific scenario does priority of the route matter. The rule of thumb is to not have two routes that evaluate equally in the same PSTN Usage. It is best to create a second, separate PSTN Usage.

Creating a Route in PowerShell

You can also create a Route using PowerShell and the `New-CsVoiceRoute` command. Below is a sample command which creates a new Route called “US-TN-Nashville-Local Calls” which permits only calls that start with “+1615” and uses the trunk “mp112.flinchbot.com”

```
New-CsVoiceRoute -Identity "US-TN-Nashville-Local
Calls" -NumberPattern "+1615" -PstnGatewayList
"mp112.flinchbot.com"
```

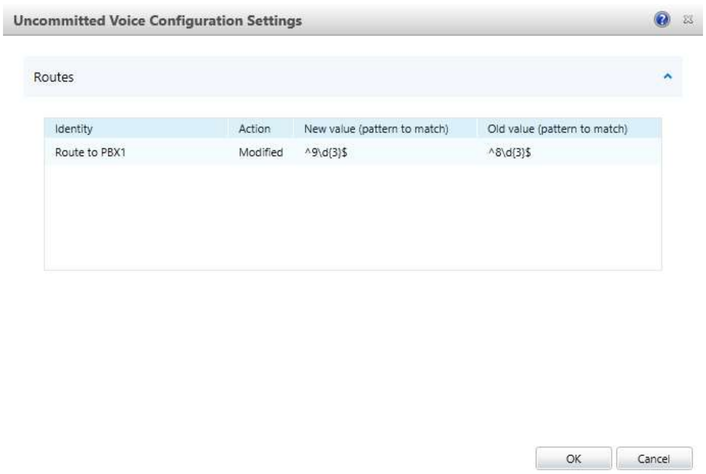
The above voice policy won't do much until we attach a PSTN Usage to it. I will discuss PSTN Usages in the next chapter.

Committing a Route

After creating or editing a Route, the settings will not become active until you commit the changes.

To commit your changes, select the commit menu item from the main Routes page. You are then given 4 choices. You can either choose to view your unsaved changes or commit them all. Both of these options are essentially the same as both will allow you to view the changes though only the commit option lets you save changes.

Figure 5 – 12



If you decide to discard all of your changes, you will not be shown a summary of which changes you are discarding.

Editing a Route

In order to edit a Route, you can either double-click on it in the list, or highlight the route and select Edit from the menu. From within the Edit menu, select the "Show Details…" option. You will then be presented a screen similar to the one seen in Figure 5-13.

Figure 5 – 13

If you would like to see your Routes via PowerShell then you need to use the `Get-CsVoiceRoute` cmdlet.

Run on its own it returns all of the committed routes and their settings. If you append the *-identity* parameter, you can then see the details for a specific Route. The command below shows the settings for a Route named "Route to PBX1".

```
Get-CsVoiceRoute –identity "Route to PBX1"
```

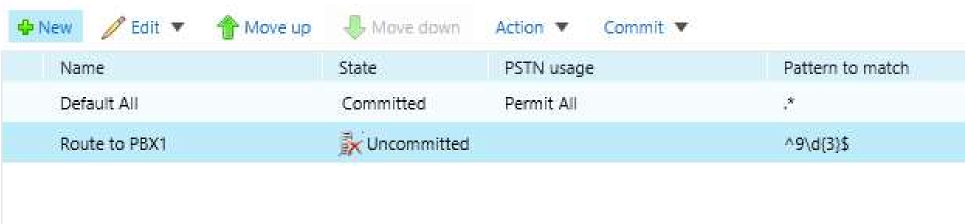
In order to edit a Route via PowerShell you use the `Set-CsVoiceRoute` cmdlet. Assuming we have a Route named "Route to PBX1" and we want to edit the "Match this pattern" value, we would use the following command:

```
Set-CsVoiceRoute –identity “Route to PBX1” -NumberPattern
“^8\d{3}”
```

Deleting a Route

In order to delete a Route, open Skype for Business Server Control Panel and navigate to Voice Routing. From there, select the Routes tab. At this point you will see a list of Routes. Highlight one of the Routes and select "Edit" and then "Delete" from the menu.

Figure 5 – 14



The state of that Route will change to say "Uncommitted". At this point, select the "Commit" menu item and then select "Commit All". A summary screen appears. Click "Commit" at the bottom of this screen and your Route will be deleted.

If you would like to use PowerShell to delete a route, then you need to use the `Remove-CSVoiceRoute` cmdlet. If we have a route named "`US-TN-Nashville-Local Calls`" that we want to remove, we would use the following command:

```
Remove-CsVoiceRoute "US-TN-Nashville-Local Calls"
```

Summary

Routes are used to send calls to the proper trunk(s). They can be used to limit which calls are sent to a trunk or they can simply permit every call. You can set a company-wide caller ID value for each route too.

If you assign two Trunks to a Route, then those Trunks will be used in a round-robin fashion. Routes are connected to Voice Policies via PSTN Usages.

Chapter 6 – PSTN Usages

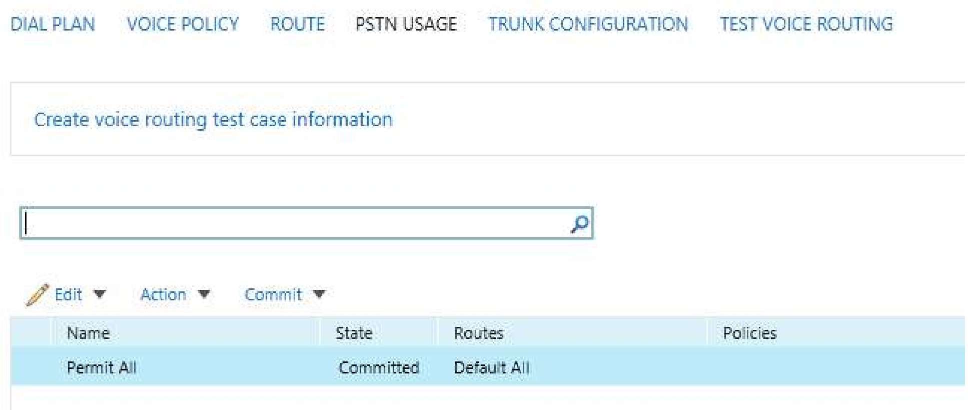
PSTN Usages are a simple concept. A PSTN Usage simply connects a Voice Policy with a Route. A Voice Policy limits the features a user can utilize (call forwarding, call park, etc.) and a Route limits what numbers a user can dial (local, long distance, etc.). A PSTN Usage is the bridge between a Voice Policy and a Route in order to provide a single set of policies and rules for a given user.

So why not connect a Voice Policy directly to a route? PSTN Usages, much like the concept of Trunks, provides much more flexibility in design than if there were a 1:1 mapping of Voice Policy to Route.

In this chapter, PSTN Usages will be explained as well as how to handle Failover and Least Cost routing.

In order to access PSTN Usages, open the Skype for Business Server Control Panel, click on “Voice Routing”, and then click on the PSTN Usage tab. You should see a screen similar to Figure 6-1.

Figure 6 – 1



Creating a New PSTN Usage

All PSTN Usages are Global in scope. There is no sophisticated scoping as in Dial Plans or Voice Policies.

In order to create a new PSTN Usage, you need to go to the Voice Policy section of the Voice Routing tab. You cannot create a new PSTN Usage by navigating to the PSTN Usage tab. The PSTN Usage tab is a read-only view of your PSTN Usages. As such, you will probably not use the tab too often other than as a convenient reference point.

To create a new PSTN Usage, navigate to the Voice Policy section. Within the Voice Policy section, click the "New" button in the "Associated PSTN Usages" section.

A new screen appears - the "PSTN Usage Record" screen. All you need to do is provide the name of the PSTN Usage. In my experience this usually mirrors the name of the Voice Policy as the two items should be logically similar.

After entering a name, you can then either select an existing voice Route (Figure 6-5) or you can create a new one (See Chapter 5 for details).

Figure 6 – 2

Once you have assigned a name and added at least one Route, you click the OK button to return to the "Edit Voice Policy" page. Click OK again to return to the main Voice Policies page, then commit the changes.

You can also create a PSTN Usage using PowerShell and the `Set-CsPstnUsage` command. Why not `New-CsPstnUsage`? Well that's not a command! PSTN Usages are all stored in a Global collection and you cannot create a different collection to store PSTN Usages. As such, you can't create a new PSTN Usage collection but you can add and remove PSTN Usages from the Global collection.

To add a new PSTN Usage called "Toll Calls" to the Global PSTN Usage collection, use the following `Set-CsPstnUsage` command.

```
Set-CsPstnUsage -Identity global -Usage @{add="Toll Calls"}
```

Editing a PSTN Usage

To try to edit a PSTN Usage, you can either double-click on it in the list or highlight the PSTN Usage and select Edit from the menu. From within the Edit menu, select the "Show Details..." option. You will then be presented a screen similar to the one seen in Figure 2.

Except this won't let you edit the PSTN Usage. It will only show you the details and not allow you to change them.

So how do you go about editing a PSTN Usage? Remember that a PSTN Usage is the glue between a Voice Policy and a Route. So to actually edit a PSTN Usage you have to do it via the Voice Policy tab. For more information on Voice Policies see chapter 4.

So, to edit a PSTN Usage, navigate to the Voice Policy tab. You can either double-click a Voice Policy or highlight the policy and select "Edit" from the menu. From within the Edit menu, select the "Show Details…" option. You will then be presented a screen similar to the one seen in Figure 6-3.

Once you have a Voice Policy open, scroll down to the "Associated PSTN Usages" section. From here you can either double click on a PSTN Usage or you can highlight it in the list and click the "Show details…" button.

After opening a PSTN Usage this way, you will see a screen similar to Figure 4.

Figure 6 – 3

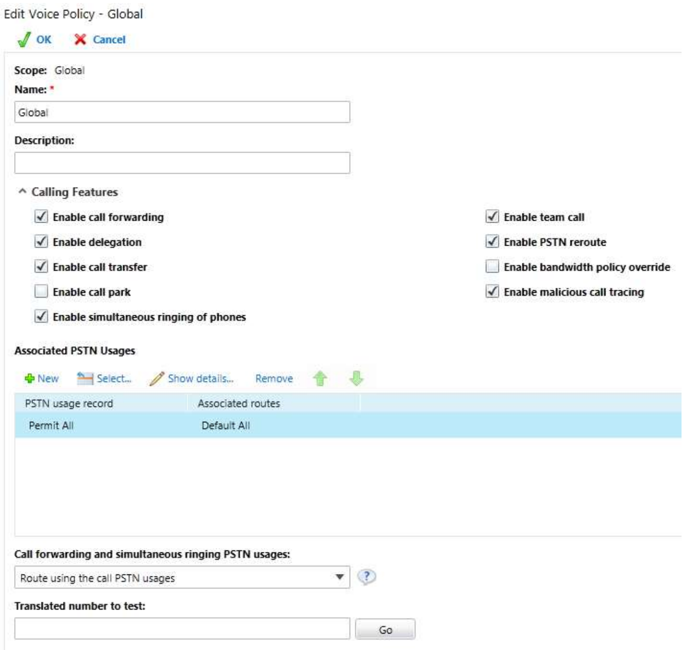
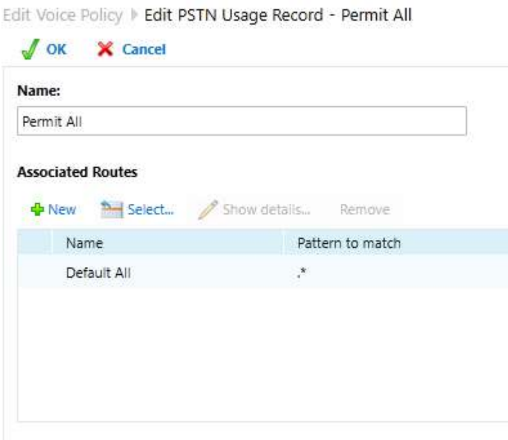


Figure 6 – 4



You can now create a new PSTN Usage or edit it. The only real editing options are to remove a Route from the PSTN Usage or select an existing Route to assign to the PSTN Usage. Removing a Route from a PSTN Usage is as easy as highlighting a route and clicking the "Remove" button. Click OK twice and then commit the change. The route is then removed from the PSTN Usage.

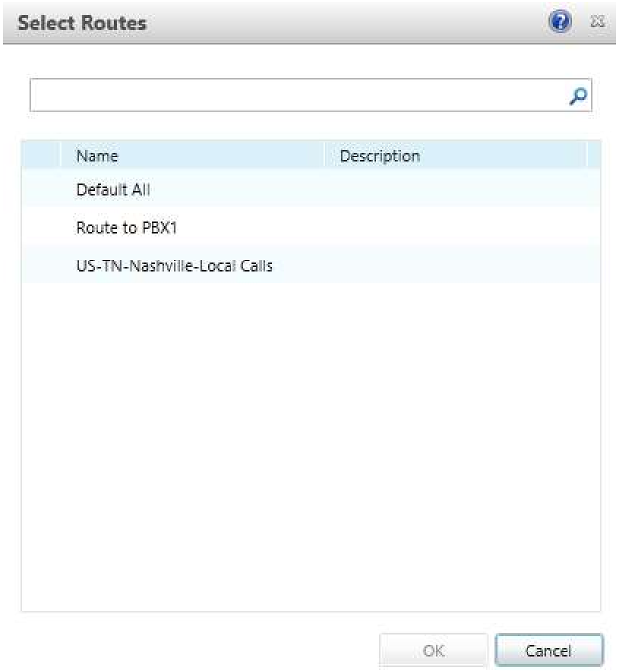
To add a route to a PSTN Usage, click the "Select..." option. This bring up a window as seen in Figure 6-5.

Highlighting a route and clicking OK will add that route to the PSTN Usage.

You can also create a new Route from here. Clicking the "New" button will take you to the "New Voice Route" page. For additional details on this page, please see the previous chapter on Routes.

After adding a Route (or Routes) to the PSTN Usage, click OK twice and then commit the change.

Figure 6 – 5



If you would like to see which PSTN Usages you have via PowerShell then you need to use the `Get-CsPstnUsage` command.

Run on its own it returns all of the committed PSTN Usages. And that is all that the command returns. You cannot see exactly what the PSTN Usage is being used for. Such a useful command!

Figure 6-6 shows the output of the `Get-CsPstnUsage` cmdlet.

Figure 6 – 6

```
PS C:\Users\flinchbot> Get-CsPstnUsage

Identity : Global
Usage    : {Nashville-National, Nashville-International,
           Indianapolis-National, Indianapolis-International...}
```

In order to actually see which Voice Policies and Trunks (aka "PstngatewayList") are used with a PSTN Usage, you have to run both the `Get-CsVoicePolicy` command and the `Get-CsVoiceRoute` command. To see which Voice Policies are using a PSTN Usage named "Local Calls", run the following:

```
Get-CsVoicePolicy| Where-Object {$_.PstnUsages -eq "Local
Calls"}
```

If you look at the Figure 6, you can see that the second value in the list is the PSTN Usage.

Figure 6 – 7

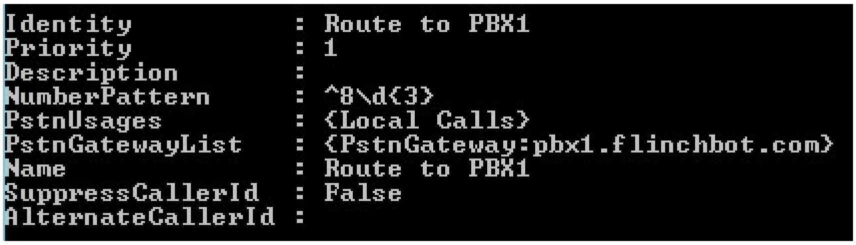
```
Identity                              : Tag:Local Calls
PstnUsages                            : {Local Calls}
CustomCallForwardingSimulRingUsages   : {}
Description                           :
AllowSimulRing                        : True
AllowCallForwarding                   : True
AllowPSTNReRouting                    : True
Name                                  : DefaultPolicy
EnableDelegation                      : True
EnableTeamCall                        : True
EnableCallTransfer                    : True
EnableCallPark                        : False
EnableMaliciousCallTracing            : False
EnableBWPolicyOverride                : False
PreventPSTNTollBypass                 : False
CallForwardingSimulRingUsageType      : VoicePolicyUsage
EnableVoicemailEscapeTimer            : False
PSTNVoicemailEscapeTimer              : 4000
```

And to see which voice routes are using a PSTN Usage named "Local Calls", run the following.

```
Get-CsVoiceRoute | Where-Object {$_.PstnUsages -eq "Local
Calls"}
```

Figure 6-9 shows the output of this command. You will see the "Local Calls" PSTN Usage listed about half way down, appropriately labeled "PstnUsages"

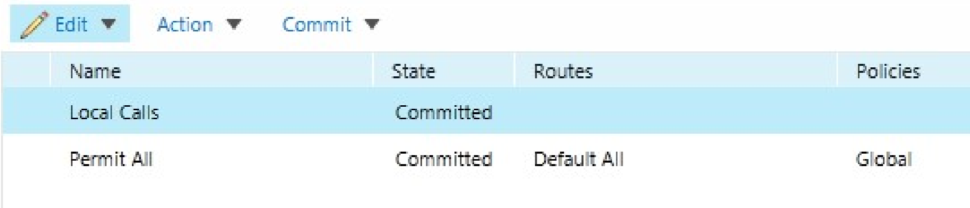
Figure 6 – 8



Deleting a PSTN Usage

In order to delete a PSTN Usage, open Skype for Business Server Control Panel and navigate to the Voice Routing link. From there, select the PSTN Usage tab. At this point you will see a list of PSTN Usages. Highlight one of the usages and select Edit and then Delete from the menu.

Figure 6 – 9



You must remove the PSTN Usage from any associated Voice Policies and Routes before being able to delete it. In Figure 6-10, only the “Local Calls” usage can be removed.

The state of the PSTN Usage will change to say “uncommitted”. At this point, select the “Commit” menu item and then select “Commit All”. A summary screen appears. Click “Commit” at the bottom of this screen and your PSTN Usage will be deleted.

If you would like to use PowerShell to delete a PSTN Usage then, just like adding a PSTN Usage, you need to use the `Set-CsPstnUsage` cmdlet. All that we are doing is removing a PSTN Usage from the Global collection of PSTN

Usages. To remove a PSTN Usage named "Local Calls" you would run the following command.

```
Set-CsPstnUsage -Identity global -Usage @{remove="Local"}
```

Note that you can remove a PSTN Usage in PowerShell that is still attached to a Voice Policy or Route. Returning to Control Panel will not display any error either. So, with regards to the `Set-CsPstnUsage` cmdlet, it is best to remove PSTN Usages via Control Panel since there is a sanity check that happens before you can delete the PSTN Usage.

Failover Routing

In Chapter 5 – Routes - it was mentioned that adding a second gateway to a Route does not provide failover support. Rather that configuration enables round-robin load balancing between gateways. If you want to have a backup Route should a primary location fail, you would then use failover routing. Failover routing is implemented via PSTN Usages.

Envision the following example:

You have two locations – one in Nashville, Tennessee and one in Indianapolis, Indiana. Both locations have gateways that connect them to the public telephone system. Both locations do not put any restrictions on their callers, so callers in Nashville can make local, long distance, and international calls that go out through the Nashville gateway. Users in Indianapolis can make local, long distance, and international calls that go out through the Indianapolis gateway.

You would like to configure Skype for Business so that if the gateway in Nashville fails, calls will go out via the gateway in Indianapolis. Similarly, if the gateway in Indianapolis fails, calls should be routed out via the Nashville gateway.

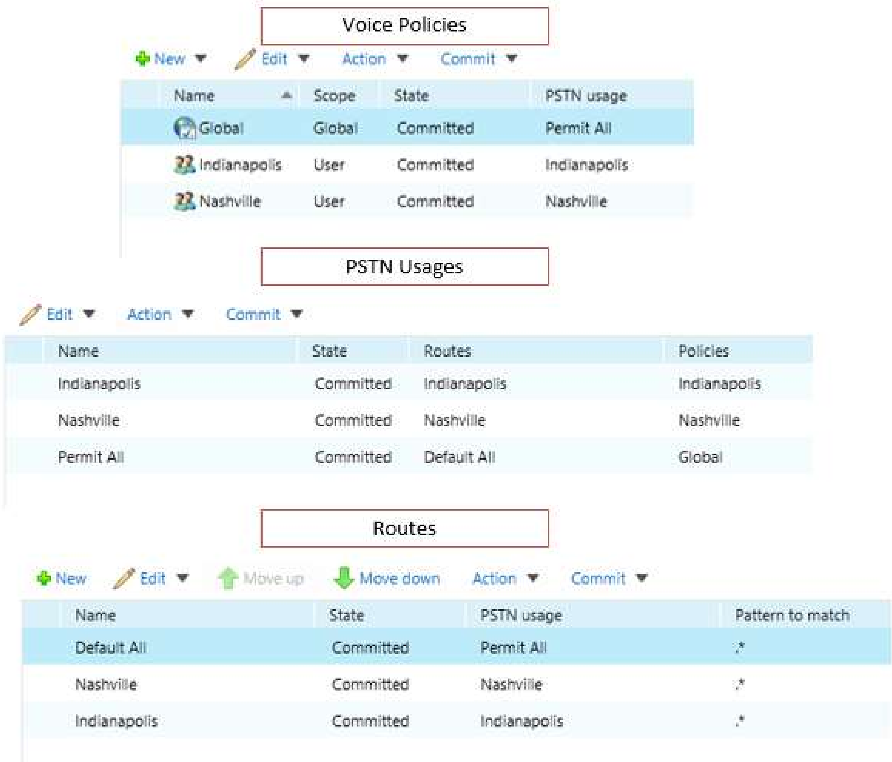
To do this, you will need to have two Voice Policies, one named Nashville and the other named Indianapolis. Make them User-level policies. You can leave all of the Voice Policy settings at their default values.

You will also need 2 routes – 1 named Nashville and the other named Indianapolis. You can leave these routes with their default settings though you will need to assign the appropriate Trunk to each gateway.

> As a reminder, Trunks are created via Topology Builder. For more information on creating Trunks, see Chapter 2.

Figure 6-10 is an overview of how everything should look.

Figure 6 – 10



Notice that the "Pattern to match" for the Nashville and Indianapolis Routes is set to the regular expression wildcard value of ".*". This means that any call can be made for the user who is assigned either of these Voice Policies.

A user is assigned one of these Voice Policies. At a very high level, after the user dials the call, it is then evaluated to make sure their assigned Voice Policy permits that call. As part of this evaluation, the Route attached to the

Voice Policy via a PSTN Usage is also evaluated. If the dialed number does not match the “Pattern to match” of the Route then the call will be denied.

In this example, all calls will be permitted to pass from the Voice Policy to the Route via the PSTN Usage. However, if the Mediation Server determines that the gateway (or SIP trunk) is down or otherwise unavailable, the call will drop.

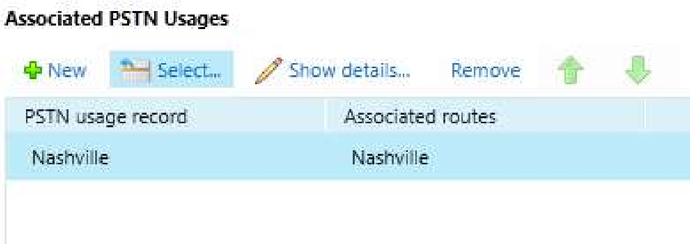
So the path from Voice Policy -> PSTN Usage -> Route -> Trunk is evaluated for every call. If any link in that chain rejects the call, the call will drop.

So what happens with the configuration in Figure 6-10 if the Mediation Server in Nashville determines that the gateway in Nashville is down? Will the calls route to Indianapolis?

No, they will not because we haven’t told Skype for Business that we want Indianapolis to be the failover route for Nashville. In order to make Indianapolis the failover route for Nashville, we need to edit the Nashville Voice Policy.

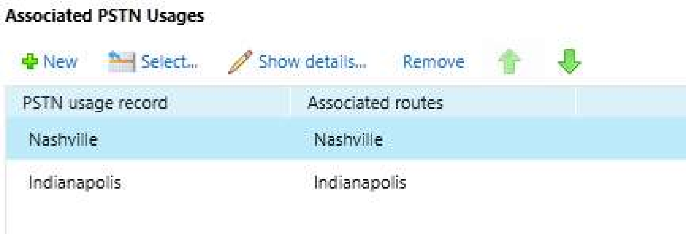
To fail calls over to Indianapolis if the Nashville gateway is unavailable, we simply add the Indianapolis PSTN Usage to the Nashville “Associated Routes” list. This is done by clicking “Select” in the “Associated PSTN Usages” section of the “Edit Voice Policy” screen for the Nashville PSTN Usage.

Figure 6 – 11



This brings up the “Select PSTN Usage Record” screen. Select “Indianapolis” off the list and click OK. The screen should now look like this.

Figure 6 – 12



Click OK and then commit the change to make it active.

Skype for Business evaluates PSTN Usages in a top-down format. When a call is dialed, the first PSTN Usage is evaluated. If the call is not a match for that PSTN Usage, the second usage is then evaluated. If the second is not a match and there is a third usage, then that one gets evaluated next, etc.

Notice the green up and down arrows in Figure 6-12. You use these to move a PSTN Usage up or down in priority.

In this case, the Nashville PSTN Usage will be evaluated first. If the Trunk in the Nashville PSTN Usage is unavailable, Skype for Business will then evaluate the Indianapolis PSTN Usage. If that PSTN Usage is valid then the call will be sent out via the Indianapolis gateway.

Remember that a trunk is a collection of a Mediation Server and a Gateway (or SIP trunk). The call will fail to match a PSTN Usage if either the gateway or the associated Mediation Server (or Mediation Server pool) is unavailable or the call doesn't match the regular expression in the Route.

Once Skype for Business determines that the Nashville trunk is healthy again, it will begin routing the calls back out the Nashville PSTN Usage. In other words, Skype for Business will automatically recover once a failure has been corrected.

Least Cost Routing

Least cost routing is similar to failover routing though it serves a different purpose. Least cost routing is essentially used to avoid paying toll charges. If a user in Nashville is calling a customer in Indianapolis, why not send the call out via the Indianapolis gateway? This way you avoid paying the toll charges associated with calls from Nashville to Indianapolis.

Envision the following example:

You have two locations – one in Nashville, Tennessee and one in Indianapolis, Indiana. Both locations have gateways that connect them to the public telephone system. Both locations do not put any restrictions on their callers, so callers in Nashville can make local, long distance, and international calls that go out through the Nashville gateway. Users in Indianapolis can make local, long distance, and international calls that go out through the Indianapolis gateway.

You would like to configure Skype for Business so that if a Nashville user places a call to a customer in Indianapolis, then that call will go out via the Indianapolis gateway. Similarly, any calls from Indianapolis users to customers in Nashville should go out via the Nashville gateway.

In order to do least cost routing, we need to be able to define what the parameters are for local calls in Nashville and what the parameters are for local calls in Indianapolis. The E.164 standard will help with this.

Both Nashville and Indianapolis are in North America so the "country code" is 1.

Within the North American Numbering Plan, the next 3 numbers following the country code is the area code. The area code for Nashville is 615 and for Indianapolis the area code is 317. Put together with the country code, we can now create a unique pattern defining calls for each city. (Technically there are some "overlay" area codes for Nashville and Indianapolis. But for sake of simplicity, I will pretend those don't exist.)

> The "1" country code is used by much of North America. This includes Canada, the United States (and its territories inside and outside of North America), and most Caribbean nations and territories. The 25 countries that share the "country code" of 1 are all part of what is called the North America Numbering Plan (NANP).

One other bit of information about the North America Numbering plan is that the last 7 digits of the phone number make up the "local" phone number.

From here it is easy to build a Regular expression to determine if a call is destined for Nashville or for Indianapolis.

City	E.164	Regular expression
Indianapolis	+1317	^\+1317\d{7}$
Nashville	+1615	^\+1615\d{7}$

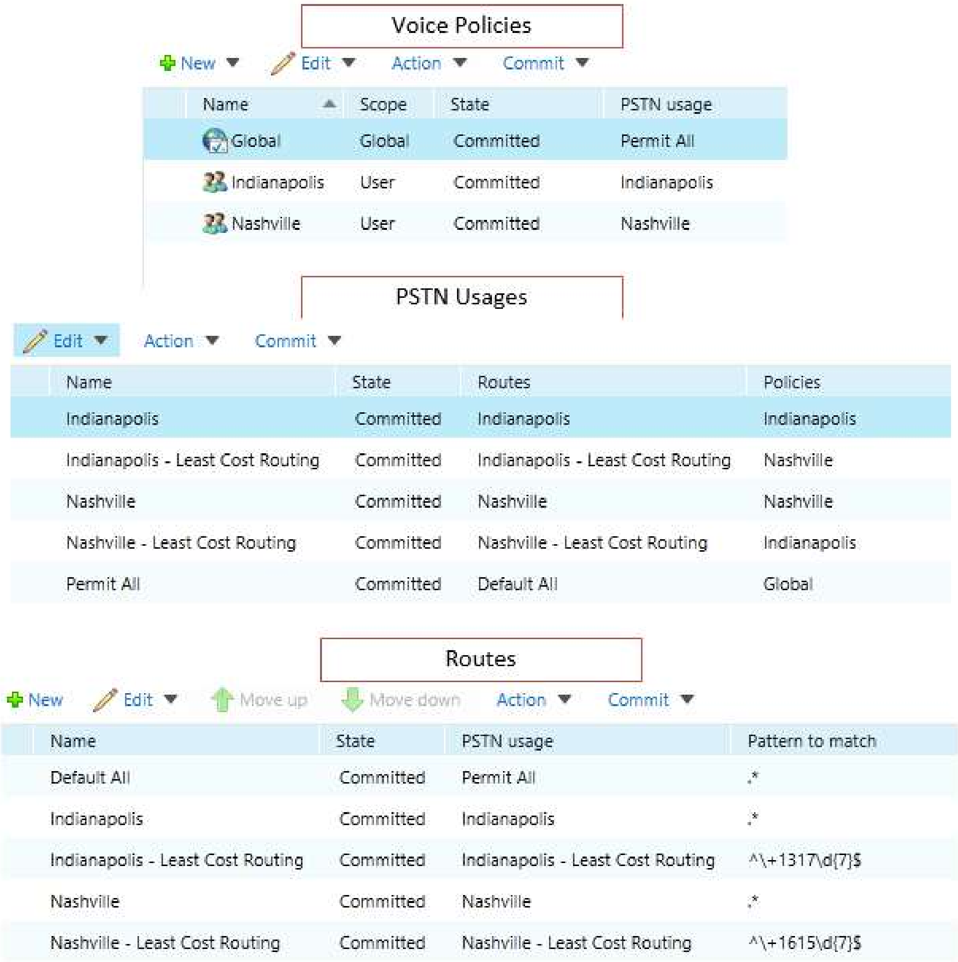
The above regular expressions look for the unique country and area code followed by any 7 digits.

If we were to apply these regular expressions to a Route, we could then limit which calls are permitted to a specific Route and, by association, a specific gateway in either city.

Building on the example shown in Figure 6-10, we cannot use the existing Nashville and Indianapolis routes as they will still be used to permit calls to numbers outside of Nashville and Indianapolis. We need to create two new routes that contain the regular expressions above. We then add those routes to new PSTN Usages. Finally, we add the new PSTN Usages to the Voice Policies for each location.

Figure 6-13 shows these additions and edits to the existing Voice Policy.

Figure 6 – 13



Voice Policies

New ▼ Edit ▼ Action ▼ Commit ▼

Name	Scope	State	PSTN usage
Global	Global	Committed	Permit All
Indianapolis	User	Committed	Indianapolis
Nashville	User	Committed	Nashville

PSTN Usages

Edit ▼ Action ▼ Commit ▼

Name	State	Routes	Policies
Indianapolis	Committed	Indianapolis	Indianapolis
Indianapolis - Least Cost Routing	Committed	Indianapolis - Least Cost Routing	Nashville
Nashville	Committed	Nashville	Nashville
Nashville - Least Cost Routing	Committed	Nashville - Least Cost Routing	Indianapolis
Permit All	Committed	Default All	Global

Routes

New Edit ▼ Move up Move down Action ▼ Commit ▼

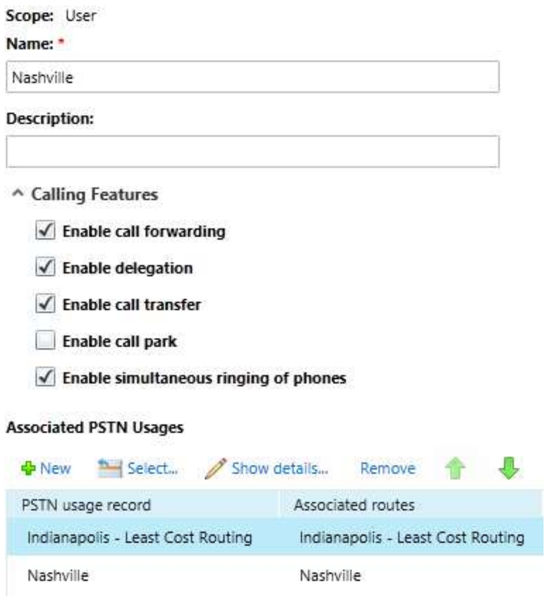
Name	State	PSTN usage	Pattern to match
Default All	Committed	Permit All	.*
Indianapolis	Committed	Indianapolis	.*
Indianapolis - Least Cost Routing	Committed	Indianapolis - Least Cost Routing	^\+1317\d{7}$
Nashville	Committed	Nashville	.*
Nashville - Least Cost Routing	Committed	Nashville - Least Cost Routing	^\+1615\d{7}$

This is a similar table to Figure 6-10 but with the additions of two additional PSTN Usages and two additional routes. The “Indianapolis – Least Cost Routing” PSTN Usage contains the “Indianapolis – Least Cost Routing” Route. Similarly, the “Nashville – Least Cost Routing” PSTN Usage contains the “Nashville – Least Cost Routing” Route.

Simply creating these will not make least cost routing work. To make it work we need to make sure that the PSTN Usages are on the right Voice Policies and in the right order.

The "Indianapolis – Least Cost Routing" PSTN Usage gets added to the Nashville Voice Policy and the "Nashville – Least Cost Routing" PSTN Usage gets added to the Indianapolis Voice Policy. Figure 6-14 shows this for the Nashville Voice Policy.

Figure 6 – 14

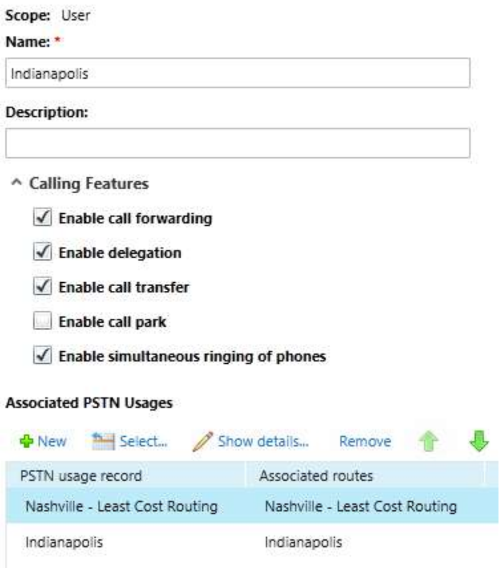


There is one crucial thing to point out. The "Indianapolis – Least Cost Routing" PSTN Usage **must** be higher in the list than the "Nashville" PSTN Usage. Remember, PSTN Usages are evaluated in a top-down manner. So the top PSTN Usage has to be the most specific PSTN Usage. The PSTN Usage at the bottom must be the least-specific usage.

In this case, the top PSTN Usage is very specific. Calls must begin with +1317 and be followed by 7 digits to match the Route assigned to that PSTN Usage. If it does, the call is then routed to the Trunk with the Indianapolis gateway. If the call does not specifically match +1317 (followed by 7 digits) then the call falls through to the Nashville PSTN Usage. Since this PSTN Usage permits all calls - regex of ".*" - the call will then be routed out via the Nashville Trunk and gateway.

Figure 6-15 shows the Indianapolis Voice Policy.

Figure 6 – 15



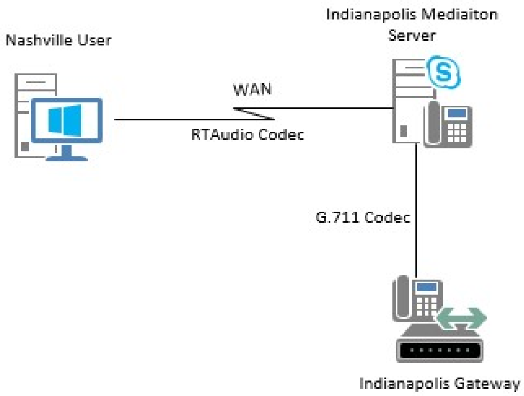
You can use both failover routing and least cost routing together. To do this, you would add the "Nashville" PSTN Usage to the bottom of the "Indianapolis" Voice Policy "Associated PSTN Usages" seen in Figure 6-15. A call would then be evaluated as such:

1. If the call starts with +1615 followed by 7 digits, then the "Nashville Least Cost Routing" PSTN Usage is applied.
2. If the call does not match the first case, send it out via the "Indianapolis" PSTN Usage. Unless the Route and its Trunk(s) is unavailable then...
3. Send the call out via the "Nashville" PSTN Usage and its assigned Route and Trunk(s).

Using PSTN Usages and routes, you can build very complex least cost routing scenarios. You could expand the regex on the Indianapolis route to include the area codes for not just Central Indiana but other locations in the Midwest such as Chicago and Detroit. Similarly, the Nashville regular expression could be expanded to include not only Middle Tennessee but the Southeast United States.

Note that your wide area network (WAN) must be able to support this additional call traffic between your locations. Not only must there be suitable, reliable bandwidth but you should also have Quality of Service properly configured. If your WAN is unreliable then the call quality with least cost routing may be unacceptable to your users. There is no point in using least cost routing if the call quality is so poor that no one can hear what the person on the other end is saying.

Figure 6 – 16



Summary

At the surface, PSTN Usages seem to be very simple. All that they do is connect a Voice Policy to a Route. However, once you start using some of the advanced configurations a PSTN Usage offers, you can gain fine-tuned control over how calls to the PSTN get routed throughout your Skype for Business environment.

Failover routing can be configured to use a secondary gateway if the primary gateway fails. Least Cost Routing can be used to route a call to a remote location to save on toll charges.

Chapter 7 – Trunk Configurations

Occasionally you will have to do some fine tuning of the Trunk. When defining a Trunk in Topology Builder, you only define which gateway connects to which Mediation Server and on which port/protocol. If you want to do something more advanced, such as changing the caller-ID for a specific group of users, you need to create a Trunk Configuration.

Generally speaking, Trunk Configurations are optional and Enterprise Voice can very successfully work without them. However, a Trunk Configuration can help centralize your outbound manipulations within Skype for Business as opposed to doing outbound manipulations on a gateway. It also gives you the ability to control some of the Trunk parameters to support connectivity with SIP trunks and gateways.

Figure 7 – 1

Trunk Configuration Scope

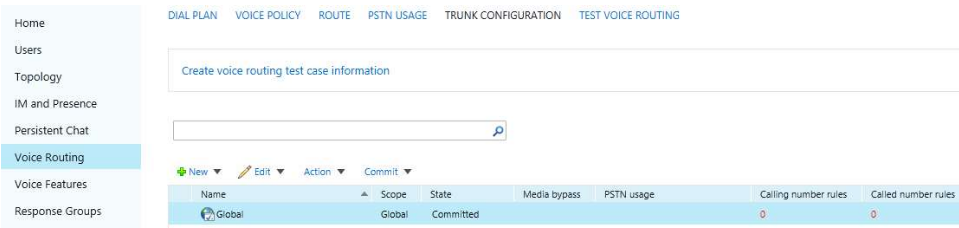
A Trunk can have one of three possible scopes: Global, Site and Pool. By default there is a Global Trunk Configuration. As per best practices, you should avoid editing any of the default Global settings, even in a small Skype for Business environment. Editing the Global configuration takes away a lot of the flexibility provided by Site and Pool-level scopes.

A Site scope applies to all trunks defined within a Skype for Business Site. A Pool scope applies to a single, specific trunk as opposed to the collection of all trunks in a Site.

Any Pool Trunk Configuration will take precedence over the Site Trunk Configuration, much like User Dial Plans take precedence over Site Dial Plans. A Site Trunk Configuration takes precedence over the Global Trunk Configuration.

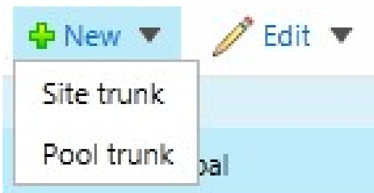
Creating a New Trunk Configuration

In order to create a new Trunk Configuration, open the Skype for Business Control Panel and navigate to the Voice Routing Section. From here, click on the Trunk Configuration tab.



Next, simply click the New menu item and select the scope you would like to use. This can be seen in Figure 7-2.

Figure 7 – 2

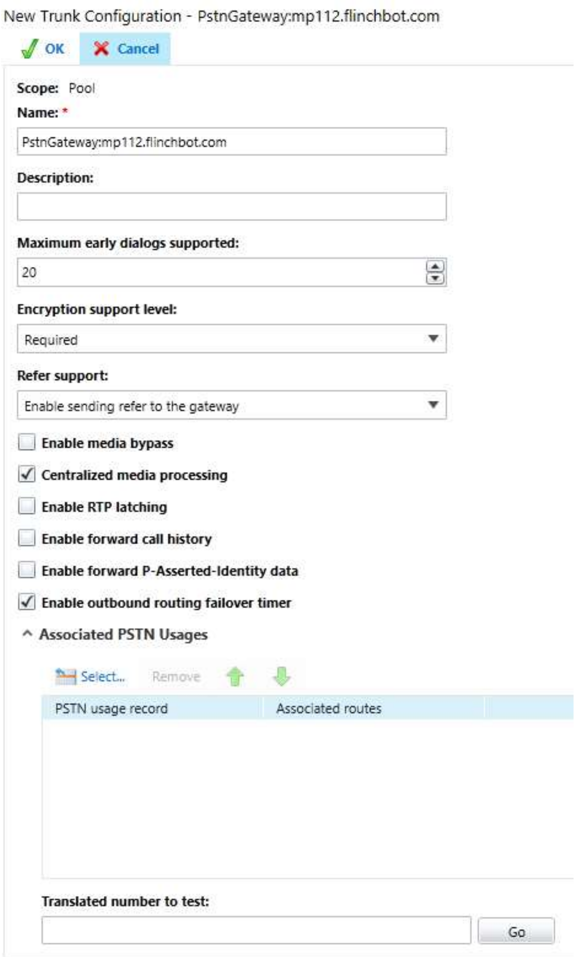


If you select the Site trunk scope you will be asked to select a site to which this policy applies. Select a site and click OK.

If you select the Pool trunk option, you will be presented with a list of all trunks defined via Topology Builder.

Regardless of which scope you selected, you will see the screen seen in Figure 7-3. This screen already has default values selected for you and those settings will be explored in detail throughout this chapter.

Figure 7 – 3



Note that Skype for Business refers to trunks with "PstnGateway" instead of "Trunks". This is a case where the nomenclature is just plain wrong. In all cases when creating a pool-level Trunk, you are always dealing with the Trunk object created in Topology Builder and *not* the PSTN Gateway also defined in Topology Builder. This is legacy naming from Lync 2010

which did not have the concept of Trunks and is just "one of those things" you will have to remember.

You can also create a Trunk Configuration using PowerShell and the `New-CsTrunkConfiguration` command. Below is a sample command which creates a pool-level Trunk Configuration for a Trunk named "PstnGateway:mp112.flinchbot.com".

```
New-CsTrunkConfiguration –Identity
"PstnGateway:mp112.flinchbot.com"
```

Committing a Trunk Configuration

After creating or editing a Trunk Configuration, the settings will not become active until you commit the changes.

To commit your changes, select the Commit menu item from the main Trunk Configuration page. You are then given 4 choices. You can either choose to view your unsaved changes or commit them all. Both of these options are essentially the same as both will allow you to view the changes though only the commit option lets you save changes.

Be careful if you choose the third option – "Cancel Selected Changes". This will remove your changes without giving you a prompt. For this choice to be available, you have to have a scope selected that has uncommitted changes. If you accidentally click this, you will only lose the settings of the selected scope.

If you decide to discard all of your changes, you will not be shown a summary of which changes you are discarding.

Deleting a Trunk Configuration

In order to delete a Trunk Configuration, open Skype for Business Server Control Panel and navigate to the Voice Routing link. From there, select the Trunk Configuration tab. At this point you will see a list of Trunk Configurations. Highlight one of the configurations and select Edit and then Delete from the menu. This is seen in Figure 7-4.

Figure 7 – 4

The state of that Trunk Configuration will change to say "uncommitted". At this point, select the "Commit" menu item and then select "Commit All". A summary screen appears. Click "Commit" at the bottom of this screen and your Trunk Configuration will be deleted.

If you would like to use PowerShell to delete a Trunk Configuration then you need to use the `Remove-CSTrunkConfiguration` cmdlet. If we have a Trunk Configuration named "PstnGateway:mp112.flinchbot.com" that we want to remove, we would use the following command:

```
Remove-CsTrunkConfiguration –identity
"PstnGateway:mp112.flinchbot.com"
```

Trunk Configuration Features

The following sections will go through every feature available via the Trunk Configuration settings.

Maximum Early Dialogs Supported

This feature is used along with Media Bypass (discussed below). This setting should match the value on your gateway. This setting defines the maximum number of early dialogs that the gateway can accept.

Generally, this value is not changed from 20. Check with your gateway vendor if this value should be changed.

Encryption Support Level

This value is used to determine the level of encryption used between the mediation server and the gateway. Allowed values are:

Required – SRTP encryption must be used

Optional – SRTP will be used if possible

Not Supported – SRTP encryption is not used.

Note that, according to TechNet, SRTP mode is only used if the gateway is configured for TLS. If the connection to the gateway is TCP, then Skype for Business internally sets the value to "Not Supported". In other words, you probably don't have to change this value as Skype for Business takes care of it in the background.

https://technet.microsoft.com/en-us/library/jj688104.aspx

The default value is "Required".

Refer Support

Refer support is often used to handle things like transferring a call from one user to another. In general, you don't need to change this value if you are using a supported gateway. However, if you experience problems transferring calls with an unsupported gateway or PBX, then changing this value may help.

The available options are:

"None" – No support for handling refer requests.

"Enable Sending refer to the gateway" – Used if the trunk supports refer requests from the mediation server

"Enable refer using third-party call control" – This indicates that the 3pcc protocol should be used to transfer calls.

This feature is set to "Enable sending refer to the gateway" by default. Change this value only after referring with your gateway or PBX vendor, especially if call forwarding is not working.

Enable Media Bypass

In most scenarios, calls go from the client to the Mediation Server. The Mediation Server then transcodes the audio and passes it to the gateway. In a Media Bypass configuration, the client connects directly to the gateway and thus bypasses the transcoding performed by the Mediation Server. Keep in mind that the signaling will continue to pass through the Mediation Server.

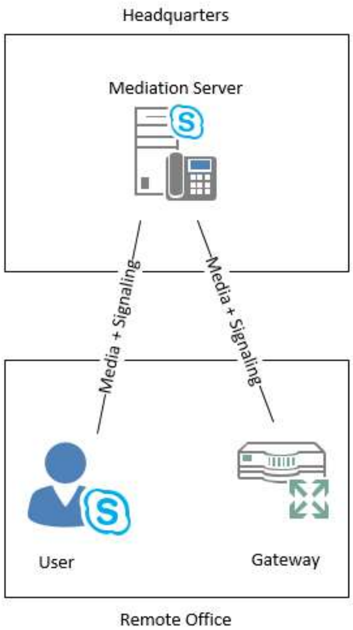
Media Bypass can help minimize poor call quality by reducing latency, needless transcoding, and packet loss.

Media Bypass can decrease the processing that a Mediation Server has to do. This can be useful when the mediation role is collocated with Front End servers.

The Mediation Server is required to convert the audio from the client – the RTAudio Narrowband codec – to the G.711 codec supported by the gateway. But in a Media Bypass configuration, the client will natively use the G.711 codec (instead of RTAudio Narrowband) when connecting directly to the gateway.

Media Bypass is also useful where a remote office has a gateway but does not also have a local Mediation server such as what is provided with a Survivable Branch Appliance. Without Media Bypass, the media from the user at the remote office has to connect to the Mediation Server across a WAN. The media then makes a second pass back across the WAN in order to be delivered to the gateway. An example of this can be seen in in figure 7-5.

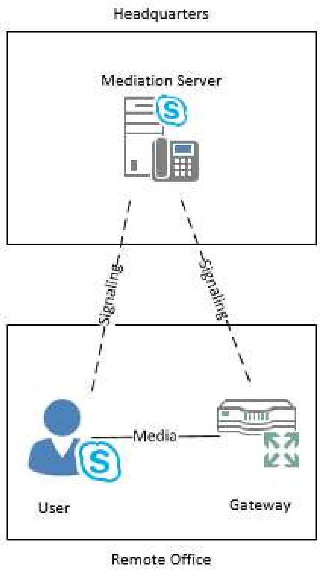
Figure 7 – 5



However, with Media Bypass enabled, there is much less network traffic used. This can be seen in Figure 7-6. The user's media connects directly to the gateway while the signaling traffic continues to traverse the WAN to the Mediation Server.

Keep in mind that this does not provide fault tolerance should the WAN fail. All calls still need to communicate with the Mediation Server which will require an active WAN connection.

Figure 7 – 6



Media Bypass may not be a good solution in a congested LAN or where there is poor Wi-Fi networking. This is because the RTAudio Narrowband codec is much more efficient than the G.711 codec. This will minimize the bandwidth used between the Wi-Fi network and the Mediation server and likely lead to improved call quality compared to Media Bypass.

There are several requirements in order to enable Media Bypass

1. A gateway that supports Media Bypass. Support for Media Bypass is a requirement for Skype for Business certified gateways. Specifically, it must support “early dialogs”.
2. Configuring the Skype for Business network regions. Configuring these is outside the scope for this book. If you only have one central location, you can use Media Bypass without having to configure any regions. Configuring Regions is outside the scope of this book but TechNet does a very good job in explaining how this is done. https://technet.microsoft.com/en-us/library/gg398255.aspx
3. The gateway must be able to receive traffic directly from the Skype for Business client. Some gateways may have firewall or routing rules limiting the traffic to or from the Mediation Server.

4. The client and the Mediation server must be well connected, i.e. in the same Region.

Media Bypass works by giving each region a specific "Bypass ID". When a PSTN call is made, the Mediation Server compares the Bypass ID of the Mediation Server with the Bypass ID of the gateway. If both Bypass ID's match, then Media Bypass will be used for the call. If the Bypass ID's do not match Skype for Business will pass the media through the Mediation Server.

For inbound calls, the Bypass ID's of the gateway and Mediation Server are also compared. If the Bypass ID matches, the media for the call is routed directly to the client.

For the example in figure 7-6, you could use the Global configuration (i.e., no configuration of network Regions).

There are two Media Bypass modes.

1. Always Bypass. This mode is used when you do not intend to implement Call Admission Control (CAC) and you don't need to specify any regions. This is the mode that would be used in Figure 7-6. With this mode enabled, there is a single, global Bypass ID for all Mediation Servers and gateways.
2. Use Site and Region Information. With this mode, a different Bypass ID is computed for each network region. Regions must be pre-defined within Skype for Business.
    a. A unique Bypass ID is created for each Region
    b. Any Site connected to that region without bandwidth constraints (i.e. CAC) will inherit the same Bypass ID for their defined Region
    c. A Site associated with a Region that has bandwidth constraints will be assigned a unique Bypass ID
    d. Subnets defined within a Site inherit the Bypass ID for that Site

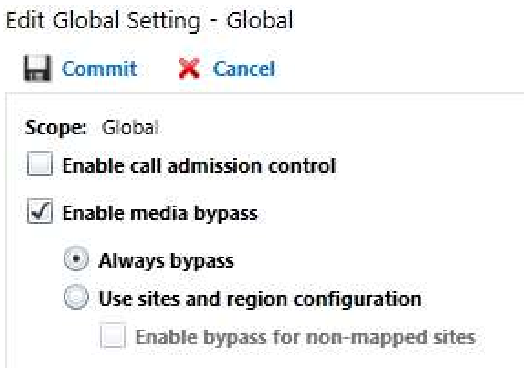
Media Bypass and Call Admission control can work hand in hand with one another. However, they are both mutually exclusive. As such, care must be taken when configuring Media Bypass when CAC is also desired.

1. Media Bypass set to "Use Site and Regions Information" and CAC is enabled. The Site and Region information will be the same for CAC and Media Bypass.
2. Media Bypass set to "Always Bypass" and CAC is not enabled. In this scenario, all Mediation servers and gateways share the same Bypass ID.
3. Media Bypass set to "Use Site and Region Information" and CAC is not enabled. For Regions that have bandwidth constraints, a unique Bypass ID is created for each Region. For Regions that do not have bandwidth constraints, the same Bypass ID is shared.
4. Media Bypass is disabled and CAC is enabled. This would be used where all gateways have poor network connectivity to the Mediation Servers or the gateways do not meet the requirements to support Media Bypass

If the gateways, Mediation Server, and – if necessary – Regions are properly defined, then enabling Media Bypass is simple.

1. Set the Media Bypass Mode. This is done in the Network Configuration section of Skype for Business Server Control Panel. Edit the settings for a defined Region (or edit the Global Region). Tick the box for "Enable media bypass" and then select the bypass mode.

Figure 7 – 7



A quick note about the "Enable bypass for non-mapped sites". This allows sites not explicitly configured to use Media Bypass. If you have a lot of subnets ticking this box can help save time configuring and maintaining subnets.

2. Check the “Enable Media Bypass” setting on the Trunks for which you want Media Bypass used.

Note that Media Bypass requires a Lync 2010 client or later.

This feature is disabled by default.

Centralized Media Processing

This is rarely disabled. It is here to support instances where the media termination point has a different IP address than the signaling termination. In most cases, the gateway uses the same IP address for both signaling and media. Check with your gateway vendor if this setting needs to be changed.

This feature is enabled by default.

Enable RTP Latching

RTP Latching is used when the media between the client and the gateway must pass through a NAT in a Media Bypass scenario. Don’t configure your network this way. But if you have a unique network configuration, RTP Latching might help you. RTP Latching is defined in RFC 6314.

This setting is disabled by default.

Enable Forward Call History

This setting defines whether call history information will be forwarded through the Trunk. This setting is used when connecting to a SIP trunk for PSTN connectivity. The Forward Call History is an entry that the Mediation Server builds and forwards on to the SIP trunk. This helps SIP trunk providers more effectively bill for their services for calls forwarded or transferred to another PSTN number. It also provides some level of fraud prevention. According to RFC 4244, Call History information “provides a standard mechanism for capturing the request history information to enable a wide variety of services for network and end users”.

Check with your PSTN SIP trunk provider if this header needs to be enabled.

This setting is disabled by default.

Enable Forward P-Asserted-Identity Data

This is similar to "Enable forward call history" in that it is only used with SIP trunks. PAI is used to verify the identity of a caller. SIP trunk providers may use this for billing purposes.

This feature is disabled by default.

Enable Outbound Routing and Failover Timer

When an outbound call is made, Skype for Business starts a 10 second timer. If the call is not connected by the gateway, the call will then be routed to the next available trunk. If there are no available trunks, the call will be dropped. If you have poor or slow connectivity between your Mediation Servers and gateways, this setting could cause calls to be dropped.

This setting can cause some issues. For example, if you place an international call, it may take over 10 seconds for the phone company to set up the call and hand it over to the gateway. In this case, the call will be dropped even though the Mediation Server and gateway are perfectly configured. In most cases, 10 seconds is long enough for a call to be set up by the gateway and handed over to the Mediation server. However, I have run into production situations where 10 seconds is not long enough.

There is no "official" way to change this value. However, it can be done by editing a file named "OutboundRouting.exe.Config" on each of your Mediation Servers experiencing this issue. This file can be found in the "C:\Program Files\Skype for Business Server 2015\Server\Core" directory. Open this file in Notepad and look for the following value:

```
<add key="FailuresForGatewayDown" value="10"/>
```

Change the "10" to something larger. In my experience, changing it to "20" has fixed the issue. Save the file and restart the Mediation service on each server where you need to make the change.

Note that you should verify this setting each time after you apply a new Cumulative Update. It is possible that the update will overwrite this file.

This setting is enabled by default.

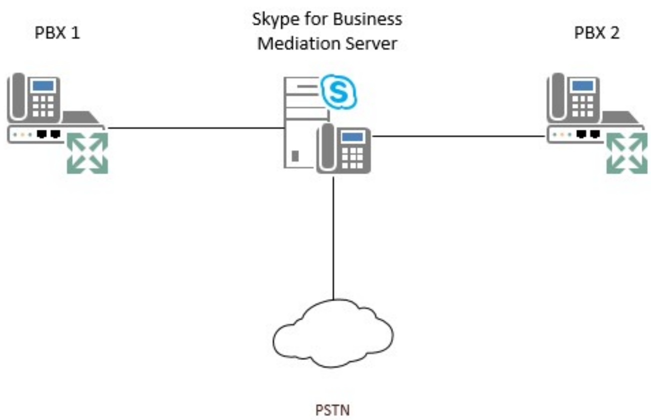
Associated PSTN Usages (Inter-Trunk Routing)

Don't put anything here. Oh you will certainly be tempted to add PSTN Usages into this field. You'll be stuck troubleshooting some super-difficult issue. You'll see that this field is empty and you'll think "*This must be the problem. There aren't any PSTN Usages assigned to this trunk!*" You will refer to TechNet and TechNet will give you misguided information saying "Collection of PSTN usages assigned to the trunk."

But you, along with TechNet, will be wrong*. PSTN Usages for Trunks don't get defined here. Instead, you will have enabled a feature called Inter-trunk routing

What is Inter-Trunk Routing? Inter-Trunk Routing allows Skype for Business to route calls from one system to another system when the calls aren't destined for Skype for Business. Confused? Maybe this picture will help.

Figure 7 – 8



Imagine that a user on PBX1 wants to call a user on PBX2. The call is not destined for a Skype for Business user. With Inter-Trunk Routing, Skype for Business will forward that call from one Trunk to another Trunk, essentially bridging the Trunks together. Depending on your requirements, you can also have calls from PBX2 route to PBX1 and calls from both PBXes to (and from) the PSTN. In order to define this routing, you first have to have PSTN Usages defined for each PBX.

What follows is an example of configuring Inter-Trunk Routing between PBX1 and the PSTN.

First, you will need to have a Trunk configured in Topology Builder to the PBX and the PSTN gateway. Verify that calling between Skype for Business and the PBX works. Again, make sure calling between Skype for Business and the PSTN is working. If these calls don't work, then Inter-Trunk Routing won't work either.

In the "Voice Policy" section of "Voice Routing" open the Global Voice Policy. I know I said that you should never edit the Global policies in Skype for Business. But also remember that you can only create a PSTN Usage from within the Voice Policy section. As such, we are only opening the Global Voice Policy in order to create a PSTN Usage.

Within the Global Voice Policy, click the "New" button in the "Associated PSTN Usages" section.

This brings you to the "New PSTN Usage Record" screen. From here provide a name. In this case, call it "Route to PBX1". Click the "New button" under "Associated Routes".

You will now be on the "New Voice Route" screen. Provide a name, such as "Route to PBX1". We need to define which calls will be routed to this trunk. Let's assume that everyone on PBX1 has a four-digit extension between 8000 and 8999. Click "Edit" by the "Match this Pattern" field and add the following Regular expression:

^8\d{3}$

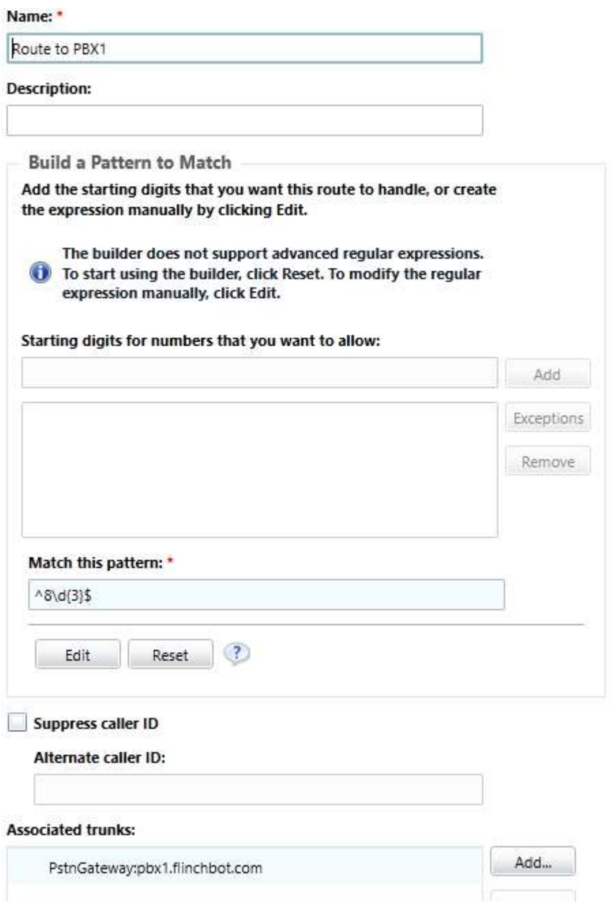
Throughout this book I've repeatedly mentioned to always use E.164. In this example, I am not using E.164. Why? Two reasons:

1. The example is easier to follow if the regular expressions remain simple and I can avoid adding things such as Trunk Configuration called number translation rules.
2. This also shows that Skype for Business will work with non-E.164 formatted numbers.

In a production environment I would normalize these numbers to E.164 and not keep them as 4-digit numbers.

Jump down to the "Associated Trunks" section. Click Add... and select the PBX1 gateway.

Figure 7 – 9

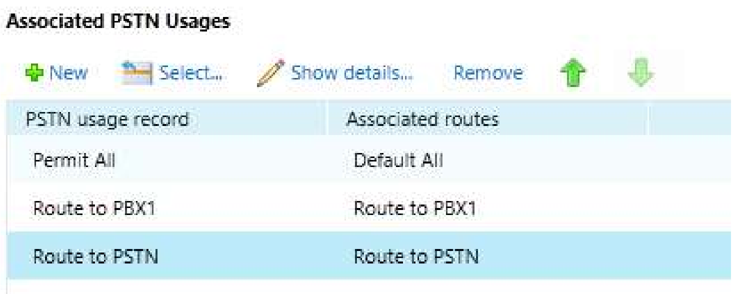


Click OK twice to return to the "Edit Voice Policy - Global" screen. We now need to create the PSTN Usage for the route to the PSTN.

1. Click the New button under Associated PSTN Usages
2. On the "New PSTN Usage Record" screen, name the PSTN Usage "Route to PSTN"
3. Click the New button under Associated Routes
4. On the "New Voice Route" screen, name the route "Route to PSTN".
5. Leave the "Match this pattern" value blank as the PSTN is our default route for any call.
6. In the Associated Trunks section, add the PSTN gateway.
7. Click OK twice to return to the Edit Voice Policy – Global screen.

You should now see three PSTN Usages associated with the Global Voice Policy.

Figure 7 – 10

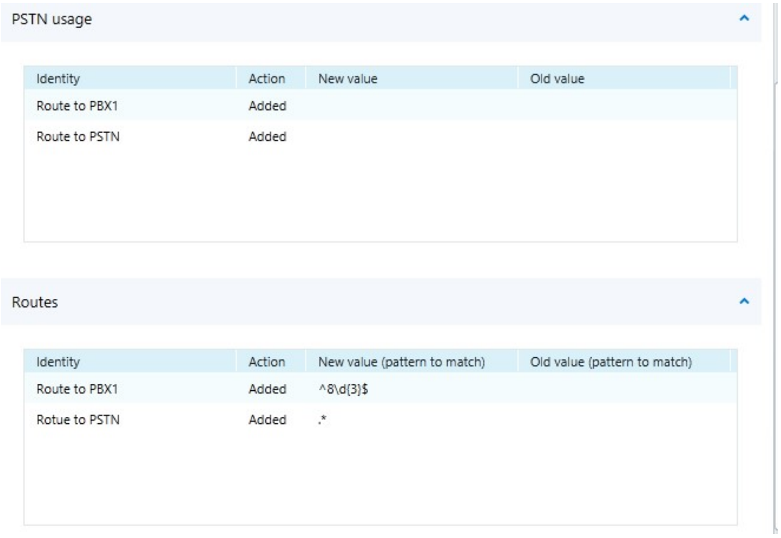


We really don't want these PSTN Usages assigned to the Global Voice Policy. We only needed to go here to create the PSTN Usages. So highlight the "Route to PBX1" and "Route to PSTN" PSTN Usages and click the Remove button. You should now only have the "Permit All" PSTN Usage listed.

Note that this does not delete these two PSTN Usages. It just removes them from the Global Voice Policy. This may all seem a rather confusing way to create PSTN Usages. But remember that there is no way to create a standalone PSTN Usage via Control Panel. You must create them as part of a Voice Policy. You can do this via PowerShell, however.

Click OK to return to the main Voice Policy page. You should see that there are uncommitted changes to the Global Voice Policy. We really didn't make any changes to the Global Voice Policy but we do still need to commit our changes so that the PSTN Usages will be saved. If you click the "Commit" menu item and then select "Commit All" you will see a summary screen showing that 2 PSTN Usages and 2 routes have been added.

Figure 7 – 11



PSTN usage

Identity	Action	New value	Old value
Route to PBX1	Added		
Route to PSTN	Added		

Routes

Identity	Action	New value (pattern to match)	Old value (pattern to match)
Route to PBX1	Added	^8\d{3}$	
Rotue to PSTN	Added	.*	

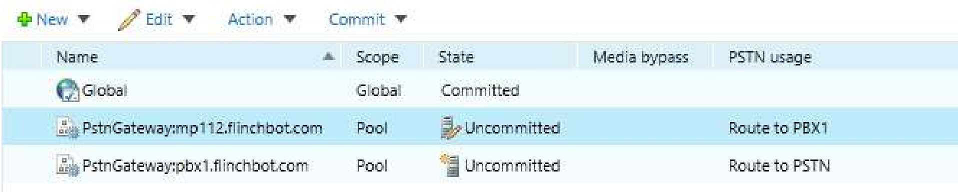
After the change has been committed, you can now add these PSTN Usages to the proper Trunk Configuration. Create a pool-level Trunk Configuration for PBX1.

Leave everything default. Scroll down to the Associated PSTN Usages section and click Select. From the "Select PSTN Usage Record" dialog, highlight "Route to PSTN" and click OK.

Repeat the steps but create a PSTN Trunk Configuration for your PSTN Gateway. Add the "Route to PBX1" PSTN Usage to the Associated PSTN Usages and click OK.

You should now see PSTN Usages values in the main Trunk Configuration Screen for the two newly created Trunk Configurations.

Figure 7 – 12



Finally, commit your changes for them to take effect. Now your users on PBX1 should be able to make calls to the PSTN. Callers from the PSTN should be able to reach users on PBX1.

So now that you know what the Associated PSTN Usages field actually does, don't randomly throw PSTN Usages in here when troubleshooting unrelated issues. You may actually cause bigger problems!

**Technically, TechNet isn't wrong. This is a list of PSTN Usages assigned to the trunk for* Inter-Trunk Routing only. *It would be nice if they added that text to their documentation.*

Associated Translation Rules

The final section in Trunk Configuration settings allows you to manipulate the outbound Caller-ID and manipulate outbound dial strings.

Calling Number Translation Rules

If you read the Routes sections of this book, it was explained that the outbound Caller-ID can be set by clicking the "Suppress Caller ID" option and then entering a value in the "Alternate Caller ID" field. This is where you set the *global* caller ID value used by the Route. However, if you want to override this value for a user or set of users, you perform that here in the Trunk Configuration setting.

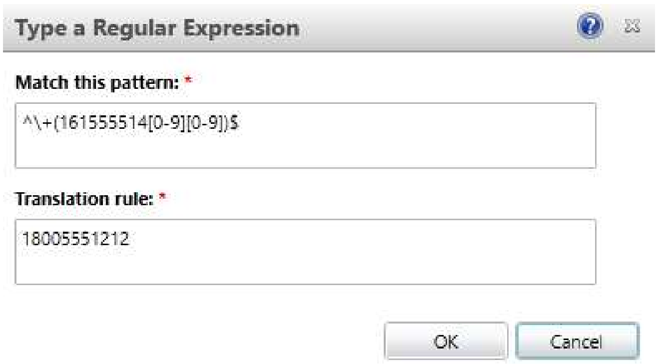
"Calling number translation rules" is the section where you can define custom caller-ID values. Note that this is based on the users' LineURI value. If the user gets a new number or extension, you may need to update this section to reflect that change.

Let's assume that when a user in our Marketing department makes a call, we want to send out the Caller-ID value of 18005551212. Let's also assume that we were forward-thinking in assigning our LineURI's so that all users in Marketing have a number between +16155551400 and +16155551499. In that case, we can create a "Pattern to match" and a "Translation rule" to reflect this.

In the "Calling number translation rules" section click "New" to create a new translation.

This brings up the "New Calling Number Translation Rule" screen. We first enter a name, such as "Marketing Department". Since we are using an advanced translation, we need to click the Edit button to enter the "Pattern to match" and the "Translation rule". Enter the appropriate regular expression and the number you want to use for the Caller ID.

Figure 7 – 13



Clicking OK brings you back to the main Trunk Configuration page. You can easily test if this is working by using the "Phone number to test" section at the bottom of the Trunk Configuration page. Enter the LineURI of a user in the Marketing department. If it matches the regex you defined, you should see the new Caller-ID reflected in the "Translation Rule" output.

Figure 7 – 14

Note that it also tells you which rule was matched which is handy when you have several rules defined. Also notice that the radio button is set to “Calling Number”. This means that you are testing the number of the LineURI making the call. In other words, the user is calling someone and this will be the Caller ID.

If I wanted to create this Calling Number Translation via PowerShell, I would have used the following command.

```
New-CsOutboundCallingNumberTranslationRule -Identity
"Service:PstnGateway:NAS-GW.flinchbot.com/Marketing
Department" -Pattern '^\+161555514[0-9][0-9]$' -Translation
'+18005551212'
```

This also assumes that the trunk getting this configuration is NAS-GW.flinchbot.com

Called Number Translation Rules

Called Number Translation Rules are used to manipulate the outbound dialed number.

Why might you use a “Called Number Translation Rule”? One of the main reasons is to remove the “+” from your E.164 normalized numbers before passing it on to the gateway and ultimately the PSTN. In the overwhelming majority of cases, I have had to remove the “+” as the telephone company was unable to properly handle the called number if the “+” was in the dial string. Some telephone companies also want nothing to do with E.164 extensions, i.e. “;ext=”.

Using the example from earlier in Inter-Trunk Routing, the users on the PBX were only reachable via a 4 digit extension. How would a PSTN call get to them as 4 digits is far too short for a public phone number?

We would remove all but the final 4 digits of the called (incoming) number before passing it to PBX1. This would happen on the "Route to PBX1" trunk. Using the "Called number translation", PBX1 would only receive 4 digits instead of the full public phone number.

Below is an example of how to remove the "+" and ";ext=" from an outbound phone number.

In the "Called number translation rules" section of Trunk Configuration, click the "New" button.

This brings up the "New Called Number Translation Rule" screen. Give it a name such as "Strip + and ext".

By default, the rule that pops up will remove the "+". The "Starting digits" field is set to "+" and the "Digits to remove" is set to "1". So if this is all you wanted to remove, you could simply click OK. However we want to remove the ";ext=" extension if it exists.

This will require an advanced Regular expression. Click the "Edit" button at the bottom of the "Build a Translation Rule" area. This brings up the "Type a Regular expression" window. In the "Match this pattern section" enter the following:

```
^\+(\d+)(;ext=\d+)?$
```

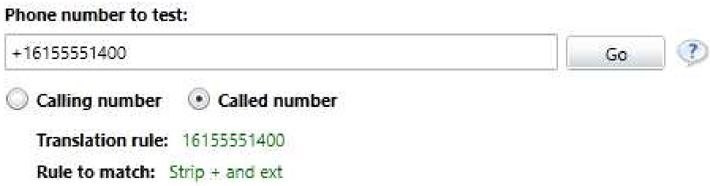
For Translation Rule, enter the following:

```
$1
```

Click OK twice and you will be returned to the Trunk Configuration screen. You can now test the regular expression in the "Phone number to test" section at the bottom.

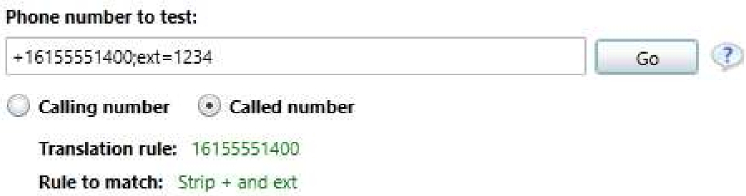
Be sure to select the radio button for "Called Number" as we will be testing a "Called number translation rule".

Figure 7 – 15



In Figure 7-15 we see that the regular expression works for numbers that start with a "+". The next example shows the rule working for calls with a "+" as the first character and a ";ext=" at the end.

Figure 7 – 16



In both cases we see that the "Translation rule" returns the expected number – the "+" and the ";ext=" have been stripped.

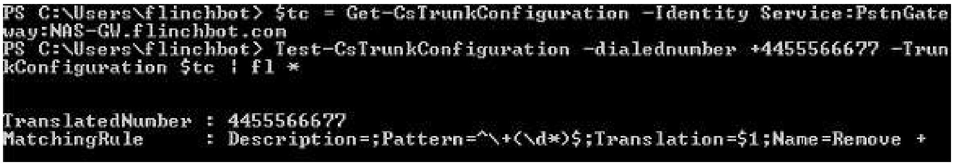
To create this in PowerShell, run the following command:

```
New-CsOutboundTranslationRule -Identity
"Service:PstnGateway:NAS-GW.flinchbot.com/Strip + and
ext" -Pattern '^\+(\d*)(;ext=\d+)?$' -Translation '$1'
```

You can also test trunk normalizations using PowerShell using the `Test-CsTrunkConfiguration` cmdlet.

Below is an example testing if the number +4455566677 matches the Called number translation assigned to the NAS-GW.flinchbot.com trunk.

Figure 7 – 17

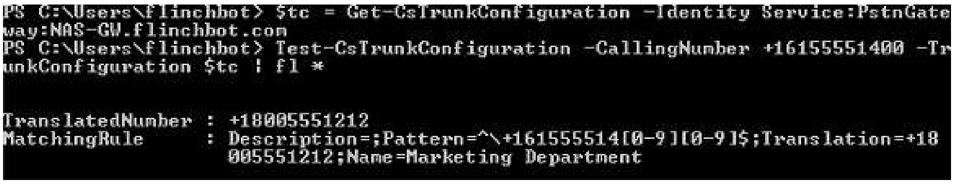


```
PS C:\Users\flinchbot> $tc = Get-CsTrunkConfiguration -Identity Service:PstnGate
way:NAS-GW.flinchbot.com
PS C:\Users\flinchbot> Test-CsTrunkConfiguration -dialednumber +4455566677 -Trun
kConfiguration $tc | fl *

TranslatedNumber : 4455566677
MatchingRule     : Description=;Pattern=^\+(\d*)$;Translation=$1;Name=Remove +
```

The first step is to assign the trunk to a variable ($tc) using the `Get-CsTrunkConfiguration` cmdlet. I then pass the number I want to test along with the trunk variable to the `Test-CsTrunkConfiguration` cmdlet.

If you want to test a Caller ID override (i.e., a calling number translation) you can use the `–CallingNumber` parameter. This is seen below.

Figure 7 – 18



```
PS C:\Users\flinchbot> $tc = Get-CsTrunkConfiguration -Identity Service:PstnGate
way:NAS-GW.flinchbot.com
PS C:\Users\flinchbot> Test-CsTrunkConfiguration -CallingNumber +16155551400 -Tr
unkConfiguration $tc | fl *

TranslatedNumber : +18005551212
MatchingRule     : Description=;Pattern=^\+161555514[0-9][0-9]$;Translation=+18
                   005551212;Name=Marketing Department
```

Summary

Enterprise Voice can work very successfully with or without a Trunk Configuration defined. However, as this chapter has shown, there are some powerful features that Trunk Configurations provide.

In my experience, the primary use of Trunk Configurations is the usage of the “called number translation rules” feature. This is so that number manipulations happen within Skype for Business instead of on a gateway. This helps provide a central location for these changes. In turn, this makes troubleshooting much easier.

Chapter 8 – Voice Features

It's finally time to leave the Voice Routing tab in Control Panel and move to the Voice Features tab. This tab is used to set up Call Park and Unassigned Numbers.

Call Park is used to park a call. OK great. What does that actually mean? We've all probably been to a grocery store and heard something over the in-store paging system saying something like "Produce Line 4". That means that someone from the produce department has been called and they should connect to line 4 to answer the call. The operator at the grocery store received the call, talked to the customer and then, instead of just putting the caller on hold, parked the call on line 4. The produce employee goes to his phone, selects line 4, and is then connected to the customer. Call Park is Skype for Business' version of this feature.

Unassigned Numbers is a feature that you can use to redirect callers who dialed an invalid number. They called a number that isn't assigned to anyone. Instead of the caller just receiving a fast busy signal, they can hear a custom message and then be routed to someone who can direct them to a valid number.

This chapter will explain how these two features can be configured so you can implement these features should the need arise.

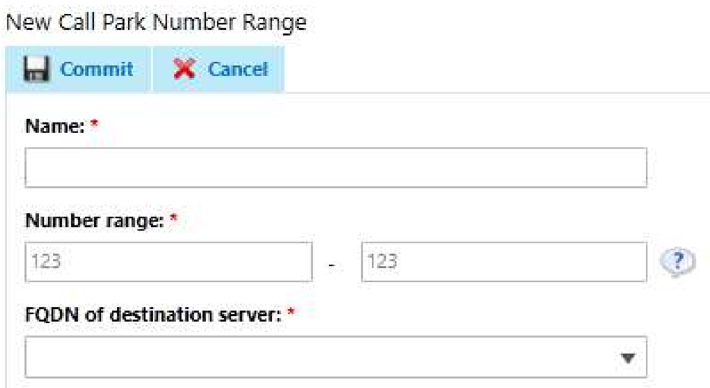
Call Park

This section will dive into the Call Park feature and uncover the hidden Call Pickup Group feature.

Creating a Call Park

To create a Call Park, open the Control Panel and click on the "Voice Features" section. This will drop you directly into the Call Park feature. From here you can click the "New" button and start configuring your first Call Park.

Figure 8 – 1



As you can see, there isn't much to creating a new Call Park. As an example, let's say that our manufacturing facility in Indianapolis needs this feature. Not every factory worker will have a phone number assigned to them. If a call comes in, they can be paged and told which Call Park number to access. After punching a Call Park number into a common area phone they will then be connected to the caller.

The first thing to do is to name the Call Park Number Range. This is pretty much self-explanatory.

Next, you need to provide a range of numbers to use for parking calls. There are a few requirements you need to keep in mind when setting this range.

1. The range can begin with a *, #, or digit
2. If you start the range with a * or # then the range must start with the number 100 or larger.
3. The maximum number supported is 9 digits long, including the * or #

4. The start number must contain the same number of digits as the end number
    a. 100-199 is supported
    b. 100-9999 is not supported as 100 is 3 digits long and 9999 is four digits long
5. The start number must be less than the ending number
    a. 100-199 is supported
    b. 199-100 is not supported
6. A pool can only have 50,000 or fewer numbers in the Call Park range
7. You cannot use DID/DDI numbers as Call Park Numbers

The range of numbers you select is called a "Call Park Orbit". Don't ask me why Microsoft chose the word Orbit. You'll just have to learn to use it.

In practice, most implementations will prefix their Call Park Orbit with a # or a *. This lets the user know that this is a special number that they are dialing.

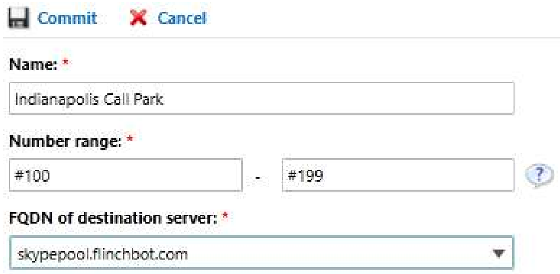
In our sample case, I'll assign the Call Park Orbit of #100 to #199 to the Nashville pool, of which the Indianapolis SBA is a member. No, you cannot define a call park directly to an SBA. You must assign it to the SBA's parent pool.

You, as an administrator, will have to set standards for each of your locations and then train your users to use their orbit.

This also means that a call could come in to the operator in Nashville and they could then park the call in the Indianapolis orbit as both locations are part of the same pool.

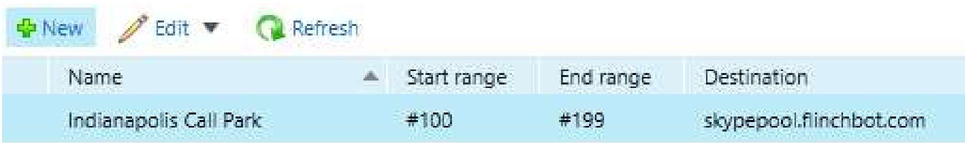
Here is what the new Call Park Orbit looks like now that I have it configured. Remember that "Indianapolis" is just a name and has no bearing on the Indianapolis SBA as this Call Park is actually happening on the Nashville Pool (skypepool.flinchbot.com).

Figure 8 – 2



I now click "commit" and the Call Park has been created.

Figure 8 – 3



To create a new Call Park Orbit using PowerShell, use the New-CsCallParkOrbit cmdlet.

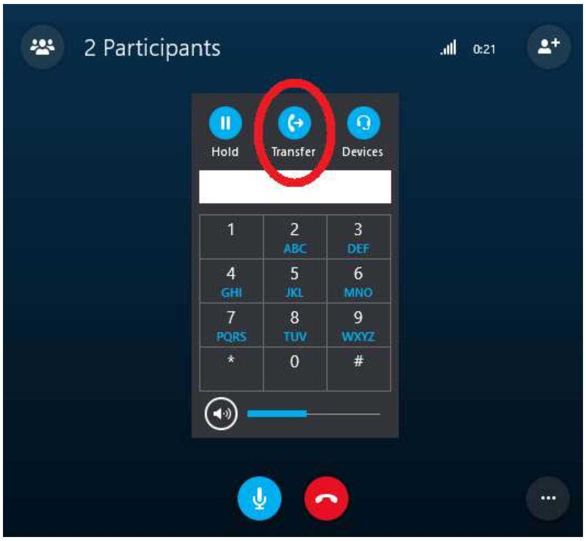
```
New-CsCallParkOrbit -Identity "Indianapolis Call
Park" -NumberRangeStart '#100' -NumberRangeEnd
'#199' -CallParkService
"ApplicationServer:skypepool.flinchbot.com"
```

If this is your first call park, this feature probably won't work immediately. This is because the Call Park service on your servers is probably not running. If you wait a few minutes, the "Skype for Business Call Park" service should start automatically. If not, go to each Front End and manually start the service.

Using Call Park

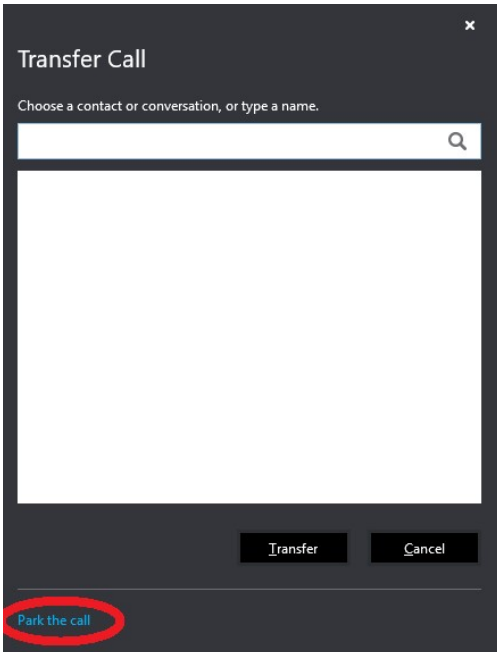
To test this, call in to a user. Answer the call, then click on the "transfer" button in the client.

Figure 8 – 4



After clicking transfer you are brought to the "Transfer Call" screen. If you look at the very bottom you will see a link named "Park the call" It's not obvious but it's there.

Figure 8 – 5

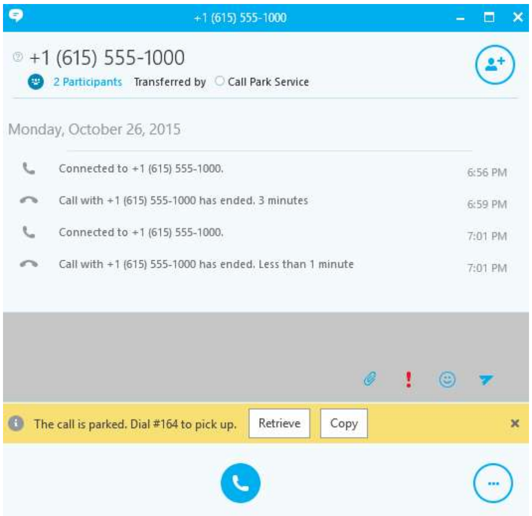


After clicking this link, two things happen.

1. The caller is put on hold and they get to hear some music.
2. The Skype for Business client will pop up and tell you on which number the call was parked.

As can be seen in the image below, the call has been parked at #164.

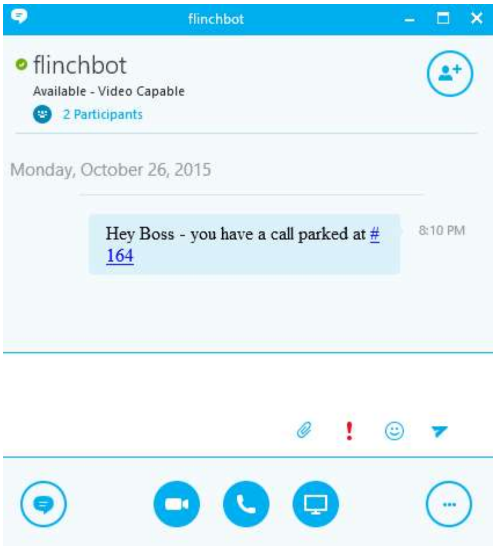
Figure 8 – 6



At this point you can send out a page saying something like "Hey you slackers in manufacturing, somebody wants to call you. Pick it up on #164". If you don't have a paging system, you could also send an IM to someone to let them know about the parked call.

If you click the "Copy" button, a direct link to the parked call will be placed in your Windows clipboard. So just send that as part of the IM and the user for whom the parked call is destined can pick up the call with one click.

Figure 8 – 7



For a user with a handset, they can go to their phone, punch in #164 and they are then connected to the caller.

So what happens if those slackers in manufacturing are super slow to get to the call. Is the caller stuck in limbo?

No.

Two things can happen. First, the person who originally parked the call can always click the "Retrieve" button seen in Figure 8-6 to return to the caller.

The second thing that happens is that there is a timer on a parked call. By default, if they are parked for more than 90 seconds then the call returns to whoever parked the call. Their phone rings and they are expected to answer that call. This is a nice customer service feature as no one wants to sit on hold endlessly wondering if someone is ever going to answer.

The person who parked the call can then either take a message or forward the call to someone else….or they could simply park the call again. The only downside to parking the call a second time is that there is no guarantee that the call will be parked back at the same number. So when the call is parked the second time, you'll have to send out another page (or IM) with the updated call park number.

Customizing Call Park

So what happens if you want the call park to wait 3 minutes instead of 90 seconds before it returns to the call parker. Can you do that? You sure can.

You'll have to use PowerShell to change this. To see which Call Park Configurations you already have, run the `Get-CsCpsConfiguration` cmdlet.

Figure 8 – 8

```
PS C:\Windows\system32> Get-CsCpsConfiguration

Identity                    : Global
OnTimeoutURI                :
MaxCallPickupAttempts       : 1
CallPickupTimeoutThreshold  : 00:01:30
EnableMusicOnHold           : True
```

You will have one Global configuration already. Like anything else, leave this one alone and create a new one. Don't ever edit a Global object unless you absolutely have to edit it.

There are only a few things you can change.

- OnTimeoutURI
    - You can enter the SIP address of a user or response group to which unanswered calls should be routed. After the number of ringbacks defined by the "MaxCallPickupAttempts" parameter is met the call will be forwarded to this SIP address. If "MaxCallPickupAttempts" is empty, then the OnTimeoutURI will be ignored.
- MaxCallPickupAttempts
    - The number of times the parked call will ring back before forwarding the call to the OnTimeoutURI SIP address.
- CallPickupTimeoutThreshold
    - The amount of time that a call will remain parked before it rings back to the user that parked the call.
        - The minimum value is 10 seconds; the maximum value is 10 minutes.
        - Enter the time in the format hh:mm:ss where hh=hours, mm=minutes, and ss=seconds.

- EnableMusicOnHold
    - You want to play music on hold so leave this True
    - If you want to change the music played, you can do this with the `Set-CsCallParkServiceMusicOnHoldFile` cmdlet.

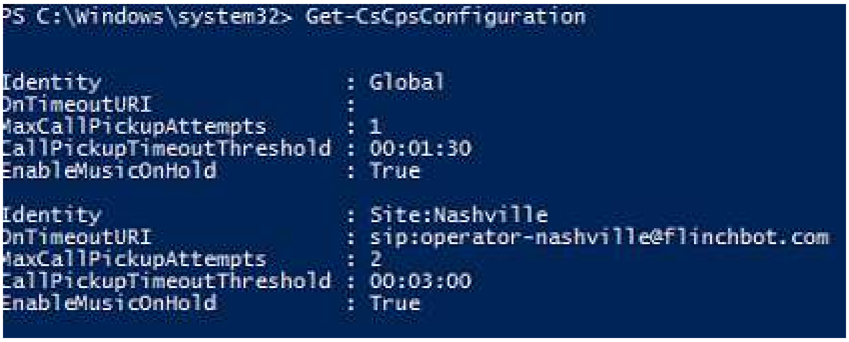
Call Park Configurations are either scoped at the Global level or at the Site level. I'm going to create a new site-level scope to change all of these values except for the Music on Hold value.

```
New-CsCpsConfiguration -Identity site:Nashville -OnTimeoutURI
sip:operator-nashville@flinchbot.com -MaxCallPickupAttempts 2
-CallPickupTimeoutThreshold 00:3:00
```

Now any Call Park in the Nashville site will be forwarded to the Nashville Operator after 2 ring backs. The ring backs will now happen after 3 minutes instead of 90 seconds.

I can verify these settings by running Get-CsCpsConfiguration.

Figure 8 – 9



While I'm at it, I'm going to change the music on hold file. I think callers want to hear some hardcore death metal. So I'll run this command to point to a different file.

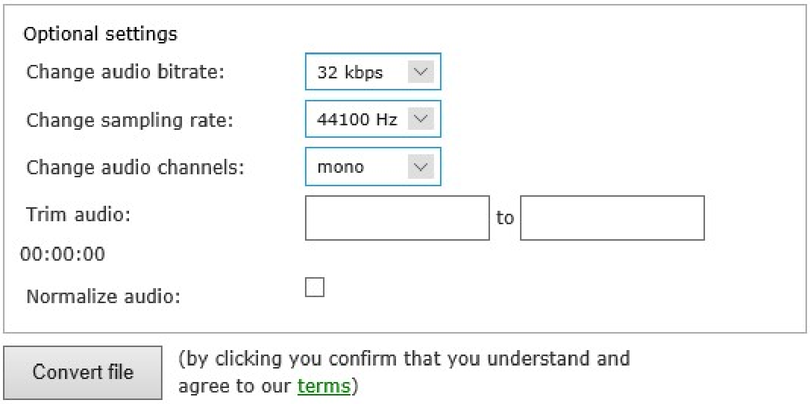
```
$a = Get-Content -ReadCount 0 -Encoding byte
"C:\Temp\ManicDarkLord.wma"


Set-CsCallParkServiceMusicOnHoldFile -Service
ApplicationServer:skypepool.flinchbot.com -Content $a
```

Note that audio files must be in the following format: Windows Media Audio 9, 44 kHz, 16 bits, Mono, CBR, or 32 kbps

If you aren't familiar with converting audio file formats, you don't have to be. There is a website out there which I use when converting files to a specific format – http://online-convert.com

After going to that website, select the "Audio converter" option for WMA. Browse to the file you want to upload and then set the options as seen below.

Figure 8 – 10



Click "Convert File" and you'll end up with a file you can use for your Call Park hold music.

This change won't take effect until you restart the "Skype for Business Server Call Park" service. A quick way to do this in PowerShell is to run `Restart-Service rtccps.`

Editing a Call Park Configuration

After making these changes, my boss wasn't happy at all. He got a severe talking-to by senior management. Not only are they supremely unhappy with my choice of hold music but they think that the hold time should only be two minutes instead of three.

I can change the timeout value using the `Set-CsCpsConfiguration` cmdlet.

To change the timeout to 2 minutes I just run the following:

```
Set-CsCpsConfiguration –Identity
Site:Nashville -CallPickupTimeoutThreshold 00:2:00
```

Now how do I change the hold music back to the default? Run the following cmdlet which reloads the default file that ships with Skype for Business Server.

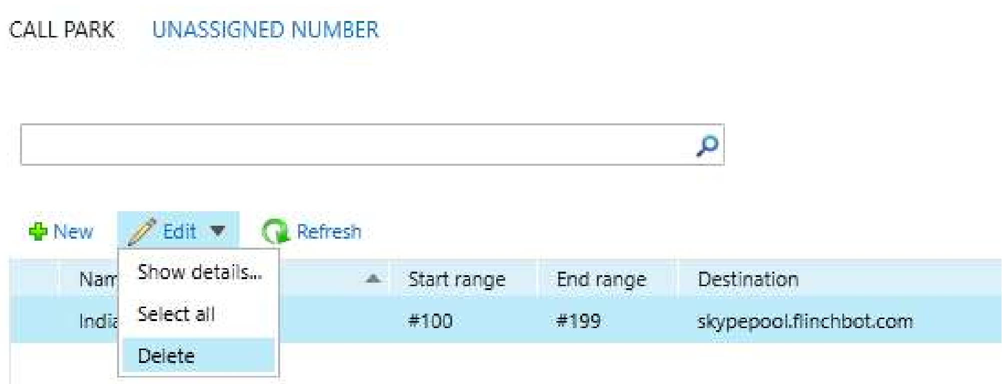
```
$a = Get-Content -ReadCount 0 -Encoding byte "C:\Program
Files\Skype for Business Server 2015\Application
Host\Applications\Call Park\Media\cpsmoh.wma"
```

```
Set-CsCallParkServiceMusicOnHoldFile -Service
ApplicationServer:skypepool.flinchbot.com -Content $a
```

Removing a Call Park Orbit and Configuration

If you decide you want to delete a Call Park Orbit, you can open Control Panel, highlight the Orbit you no longer want, and click "Delete". There is no need to commit this change for the delete to happen.

Figure 8 – 11



To delete a Call Park Orbit via PowerShell, use the `Remove-CsCallParkOrbit` cmdlet.

```
Remove-CsCallParkOrbit –Identity 'Indianapolis Call Park'
```

Group Call Pickup

Group Call Pickup is a feature which lets you answer a coworker's phone. Imagine a sales department where all of the salespeople sit close to one another. One of the salesman is out visiting a customer. You hear their phone ring. Traditionally you would have to run over to their desk to answer the phone…just to get there too late as the call transfers to voice mail.

With Group Call Pickup, you enter a code (just like a Call Park Orbit) and that call will then ring on your phone or Skype for Business client.

The rules for the number range for a Group Call Pickup are the exact same as for a regular Call Park Orbit.

To create a Group Call Pickup, you have to use the `New-CsCallParkOrbit` PowerShell command. There is an additional parameter named “`-Type GroupPickup`” which is not exposed in Control Panel.

I’m going to create a Group Call Pickup with an orbit range from #200 to #249.

```
New-CsCallParkOrbit -Identity "Indianapolis Group Call Pickup"
-NumberRangeStart '#200' -NumberRangeEnd
'#249' -CallParkService skypepool.flinchbot.com -Type
GroupPickup
```

Note the “`-Type GroupPickup`” added on the end.

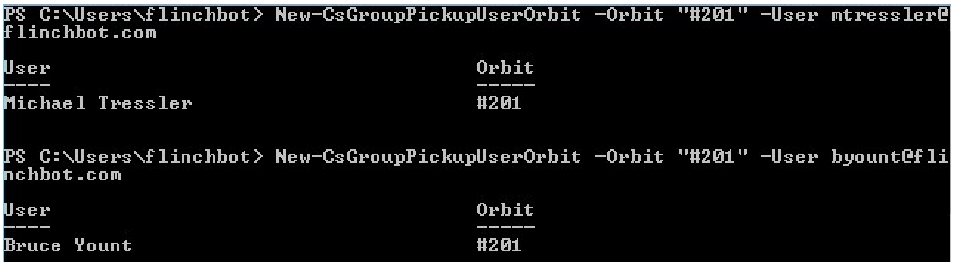
Adding Users to Group Call Pickup

I now have to assign a single number in the defined range for this Group Call Pickup. Every user assigned this single number is in the same group. This group assignment is done via the `New-CsGroupPickupUserOrbit` cmdlet. I am going to assign two users to the group “#201”.

> The following cmdlets to add and remove users from a Group Call Pickup were added in the November 2015 Cumulative Update for Skype for Business Server. Be sure you have this update installed.
>
> Also be sure to run the Skype for Business Management Shell as Administrator for these cmdlets to work.
>
> If you are using Lync Server 2013, you need to use the SEFAUtil tool to add and remove users from a Group Call Pickup. See Appendix 3 for details on using SEFAUtil.

```
New-CsGroupPickupUserOrbit –Orbit “#201” –User
mtressler@flinchbot.com

New-CsGroupPickupUserOrbit –Orbit “#201” –User
byount@flinchbot.com
```

Figure 8 – 12



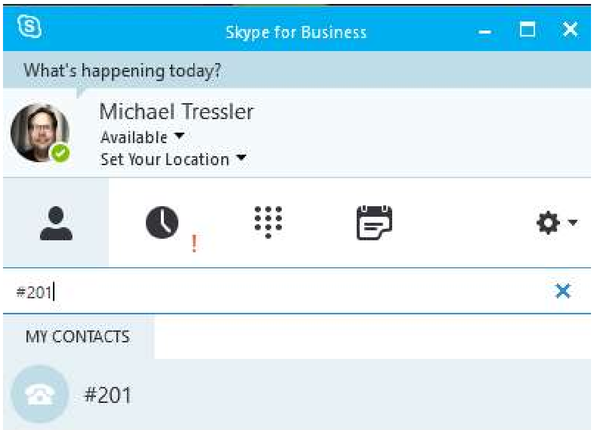
```
PS C:\Users\flinchbot> New-CsGroupPickupUserOrbit -Orbit "#201" -User mtressler@
flinchbot.com

User                                    Orbit
----                                    -----
Michael Tressler                        #201

PS C:\Users\flinchbot> New-CsGroupPickupUserOrbit -Orbit "#201" -User byount@fli
nchbot.com

User                                    Orbit
----                                    -----
Bruce Yount                             #201
```

After running these commands wait a few minutes for this change to replicate throughout your environment. Then have each user sign out of their Skype for Business client and sign back in.

Validating Group Call Pickup

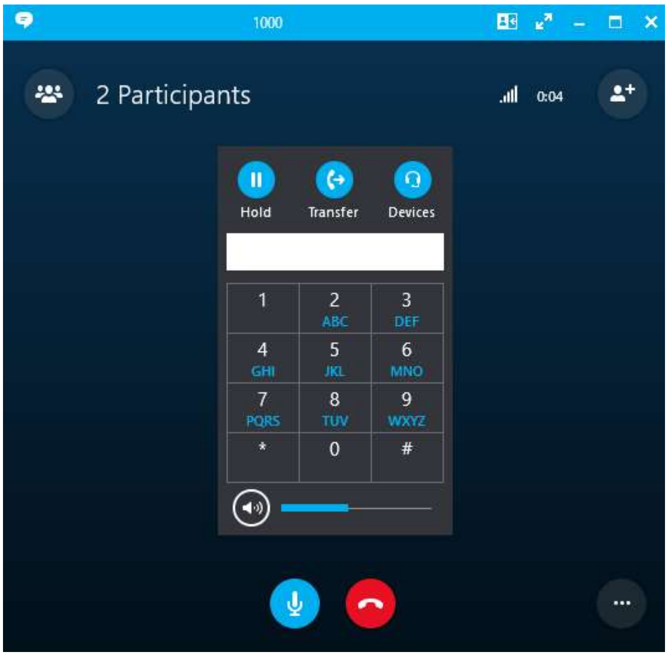
To test this, I called the user byount@flinchbot.com from the PSTN. As his client started ringing I went to the mtressler@flinchbot.com client and typed #201 in the search field in the client.

Figure 8 – 13



As soon as I hit enter, the call was transferred to me and I could talk to the PSTN caller. The "1000" at the top of the image below is the caller ID of the PSTN User.

Figure 8 – 14

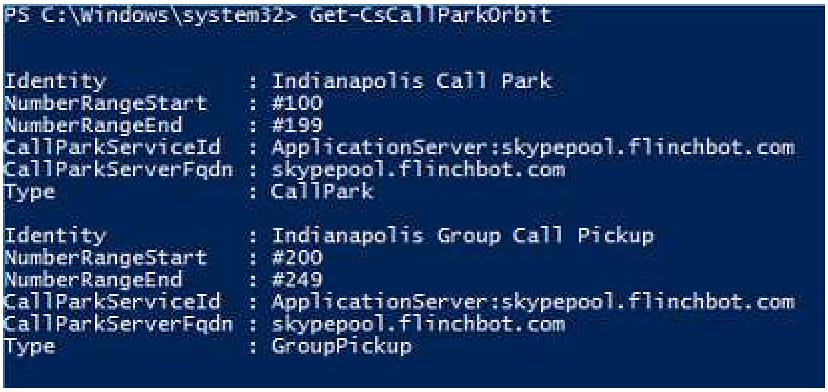


There are a few things to be aware of with a Group Call Pickup. For one – it's "hidden". If you go to Control Panel, you won't see this Group Call Pickup Orbit. Look at the Call Park page below. The Group Call Pickup isn't listed.

Figure 8 – 15

But if you go to PowerShell it magically appears.

Figure 8 – 16



Viewing Enabled Users

There is also no way to know which of your users are configured for a Call Pickup Group or to which group they are assigned unless you use the `Get-CsGroupPickupUserOrbit` cmdlet.

To use this, you need to know which of your users are enabled for Group Call Pickup. Since you may not know which of your users are enabled for Group Call Pickup, you can use this one-liner to get a list of all your enabled users.

```
Get-CsUser | Get-CsGroupPickupUserOrbit -ErrorAction
'SilentlyContinue'
```

You need to add the `–ErrorAction` parameter. Otherwise you will receive a bunch of errors for every user that is not enabled for Group Call Pickup. Using this parameter skips the error output letting you only see which of your users are enabled.

Figure 8 – 17

```
PS C:\Users\flinchbot> Get-CsUser | Get-CsGroupPickupUserOrbit -ErrorAction 'Sil
entlyContinue'

User                                    Orbit
----                                    -----
Bruce Yount                             #201
flinchbot                               #333
Michael Tressler                        #201
```

Removing a User from a Group

In order to remove a user from a Group Call Pickup, use the `Remove-CsGroupPickupUserOrbit` cmdlet.

To remove the user mtressler@flinchbot.com from Group Call Pickup, simply run the following command:

```
Remove-CsGroupPickupUserOrbit -User mtressler@flinchbot.com
```

There is no need to list the Orbit to which the user is assigned. A user can only be assigned to one Group Call Pickup.

Moving a User to a New Group

While you could remove a user from a group and then add them to a new group, there is a simpler way to do this:

Using the `Set-CsGroupPickupUserOrbit` cmdlet.

If I want to move user byount@flinchbot.com to the '#333' group, I would run the following command:

```
Set-CsGroupPickupUserOrbit –Orbit "#333" –User
byount@flinchbot.com
```

Figure 8 – 18

```
PS C:\Users\flinchbot> Set-CsGroupPickupUserOrbit -Orbit '#333' -User byount@fli
nchbot.com

User                                    Orbit
----                                    -----
Bruce Yount                             #333
```

Group Call Pickup Summary

There are a few caveats to be aware of with regards to Group Call Pickup

- A user can be a member of only one call pickup group
- Any user in the Skype for Business Server deployment can answer a call to a call pickup group member if the person answering the call knows the correct call pickup group number to dial
- If a user dials a call pickup group number to answer a call when multiple phones in the group are ringing, the user answers the call that has been ringing the longest
- Group Call Pickup cannot be used to answer the following types of calls:
    - Calls to a private line
    - Calls from a contact who has been assigned the Friends and Family privacy relationship
    - Calls routed to a delegate
    - Calls routed to a response group

An alternative to Group Call Pickup is the Team Call feature. This was shown in Chapter 4 in the “Enable Team Call” section. It is configured on the client-side and is much easier to set up.

Shared Line Appearance

Shared Line Appearance is available only in Skype for Business.You have to have the November 2015 Cumulative Update installed.

Shared Line Appearance is similar to Group Call Pickup with a few key differences.

1. It only works with phones such as Polycom VVX phones that have a firmware later than 5.4.0A
2. You don’t need to enter a code to answer the call. You just select the ringing line on your phone
3. Everyone shares the same SLA number but their personal LineURI will also still ring
4. You must configure it using PowerShell

Below is what a phone looks like when a user is logged into the phone who is also a member of an SLA workgroup.

Figure 8 – 19

You can see at the top left that the user is logged in and configured to take calls on their personal number (LineURI). Below that are five lines configured for "Operator-Nashville". This means that up to five people can call the Nashville Operator at the same time.

As a call comes in, the display updates to show which of the 5 lines currently has a call. In the image below, there is a green bar to the right of the first line (01-Operator-Nashville). This lets you know that there is currently a call on that line.

You can also see that our user Bruce Yount has answered that call. His presence changes to "In a call".

Figure 8 – 20

From here, Bruce can put the call on hold or transfer the call to another user. If he puts the call on hold, another member of his SLA workgroup could pick up that line.

A real world example of this is in a warehouse. A call comes in to an SLA. Someone answers it and is asked if a certain item is in stock. That person puts the call on hold and walks back to the stock room to look for the item. When he has his answer he doesn't need to walk all the way back to his phone. He could just pick up that line from a phone in the stock room and provide the answer (so long as the phone in the stock room is a member of his SLA workgroup).

Creating a Shared Line Appearance

Creating a new SLA requires a little bit of work. You first need to register the Shared Line Appearance application with your pool.

You register the SLA application using the `New-CsServerApplication` cmdlet. Below is the command you need to run. Replace *skypepool.flinchbot.com* with either your pool name or the name of your Standard Edition server.

```
New-CsServerApplication -Identity
'Service:Registrar:skypepool.flinchbot.com/SharedLineAppearanc
e' -Uri http://www.microsoft.com/LCS/SHAREDLINEAppearance -Critical
```

```
$false -Enabled $true -Priority (GetCsServerApplication
Identity
'Service:Registrar:skypepool.flinchbot.com/UserServices').Prio
rity
```

After running this command, you should see output similar to the below image.

Figure 8 – 21

Once you have registered the SLA application, you will need to restart the Front End service on each member of your pool.

After that has been done, run the `Update-CsAdmin` cmdlet. This will update the Role Based Access Control roles by adding the SLA-specific cmdlets to the correct roles. Run that cmdlet even if you aren't using RBAC in your organization.

Note that before running these SLA cmdlets, be sure to have opened the management shell as Administrator. Otherwise you might see an error similar to the one below:

Figure 8 – 22

Now we can finally get around to creating a new SLA workgroup. The term "workgroup" is a tad misleading as you don't actually create a new group. Rather, you take one of your existing users and configure their phone number as a shared line. Once that is done, you delegate users to that shared line.

It is interesting to note that there is no `New-` cmdlet to create a new shared line. Rather you use the `Set-CsSlaConfiguration cmdlet.` As such, this is a global object and cannot be scoped to a site or a pool.

At its simplest, you only need to define the user account that will become a shared line, the number of shared lines to create, and the action to happen when all lines are busy.

In my case, I'm going to take the operator in my Nashville office and convert that to a shared line. To create a new shared line, you use the `Set-CsSlaConfiguration` cmdlet.

```
Set-CsSlaConfiguration -Identity
operatornashville@flinchbot.com -MaxNumberOfCalls
5 -BusyOption BusyOnBusy
```

In the above cmdlet, I set the SLA to use the phone number for the operator-nashville@flinchbot.com account. I decided that 5 lines would be enough though I could have configured up to the maximum allowed, which is 25. I then configured the overflow action to be "BusyOnBusy".

Handling Overflow

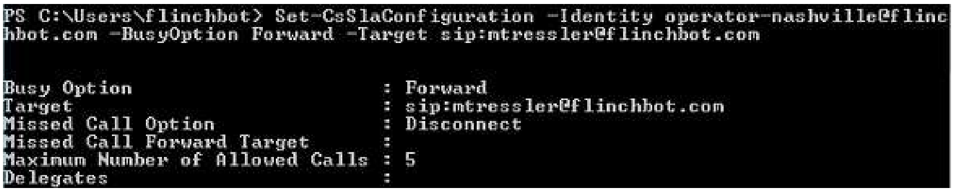
If all five lines are busy and a sixth call comes in, BusyOnBusy will give that sixth caller a busy signal. That's OK in some circumstances. But you probably want that call to be routed somewhere if all of the configured lines are busy.

You have two options to forward calls: Another extension or voice mail.

To forward calls to another extension, you set the `BusyOption` to *forward*. Below I send any overflow calls to the user mtressler@flinchbot.com.

```
Set-CsSlaConfiguration -Identity
operator-nashville@flinchbot.com -BusyOption Forward -Target
sip:mtressler@flinchbot.com
```

Figure 8 – 23



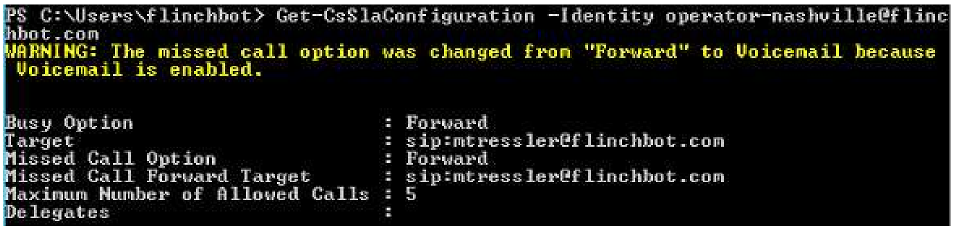
```
PS C:\Users\flinchbot> Set-CsSlaConfiguration -Identity operator-nashville@flinc
hbot.com -BusyOption Forward -Target sip:mtressler@flinchbot.com

Busy Option                   : Forward
Target                        : sip:mtressler@flinchbot.com
Missed Call Option            : Disconnect
Missed Call Forward Target    :
Maximum Number of Allowed Calls : 5
Delegates                     :
```

If you would rather forward overflow calls to a voicemail box, then you would use the following command:

```
Set-CsSlaConfiguration -Identity
operatornashville@flinchbot.com -BusyOption Voicemail
```

Note that if the user in the `–Identity` parameter is enabled for voicemail then Forward is not an available option and voicemail becomes the default option. Look at the image below. The operator-nashville@flinchbot.com account was not initially configured for voice mail. I then successfully set the Busy Option to Forward. However, once I enabled the account for voicemail, the following message began popping up whenever I looked at the configuration or made a change to the SLA:

Figure 8 – 24



```
PS C:\Users\flinchbot> Get-CsSlaConfiguration -Identity operator-nashville@flinc
hbot.com
WARNING: The missed call option was changed from "Forward" to Voicemail because
 Voicemail is enabled.

Busy Option                   : Forward
Target                        : sip:mtressler@flinchbot.com
Missed Call Option            : Forward
Missed Call Forward Target    : sip:mtressler@flinchbot.com
Maximum Number of Allowed Calls : 5
Delegates                     :
```

The Busy Option automatically becomes Voicemail instead of Forward. Interestingly, the BusyOption doesn't change to VoiceMail. It will continue to say Forward.

Handling Missed Calls

So what happens to a SLA call if no one answers the call at all? That is handled with the `–MissedCallOption` parameter. The syntax is very similar to the `–BusyOption` parameter.

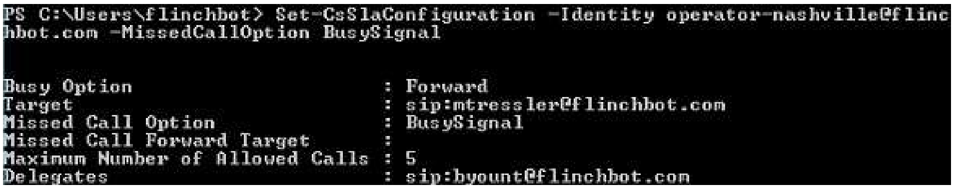
You have three options available to set for a missed call: Disconnect, forward, or busy signal.

The default option is Disconnect. In my testing, after four rings, the call drops and I got a busy signal.

Changing the MissedCallOption to "Busy Signal" does the same thing: After four rings, I get a busy tone. For the sake of completeness, here is how you change the behavior to Busy Signal.

```
Set-CsSlaConfiguration -Identity
operatornashville@flinchbot.com -MissedCallOption BusySignal
```

Figure 8 – 25

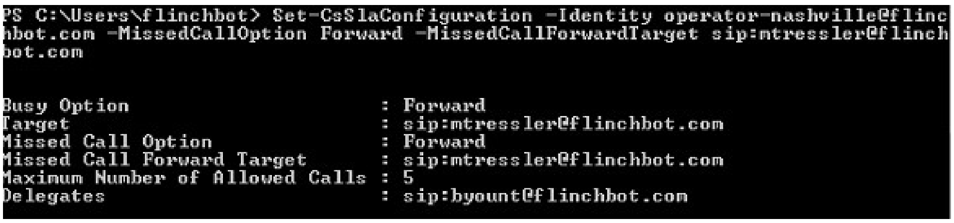


```
PS C:\Users\flinchbot> Set-CsSlaConfiguration -Identity operator-nashville@flinc
hbot.com -MissedCallOption BusySignal

Busy Option                    : Forward
Target                         : sip:mtressler@flinchbot.com
Missed Call Option             : BusySignal
Missed Call Forward Target     :
Maximum Number of Allowed Calls : 5
Delegates                      : sip:byount@flinchbot.com
```

The only one you'll really want to use is the Forward option. You can't forward directly to a voicemail box but you could forward to a response group. In the below example, I forward missed calls to the user mtressler@flinchbot.com. I define the target of the forward using the `-MissedCallForwardTarget` parameter.

```
Set-CsSlaConfiguration -Identity
operatornashville@flinchbot.com -MissedCallOption
Forward -MissedCallForwardTarget sip:mtressler@flinchbot.com
```

Figure 8 – 26



```
PS C:\Users\flinchbot> Set-CsSlaConfiguration -Identity operator-nashville@flinc
hbot.com -MissedCallOption Forward -MissedCallForwardTarget sip:mtressler@flinch
bot.com

Busy Option                    : Forward
Target                         : sip:mtressler@flinchbot.com
Missed Call Option             : Forward
Missed Call Forward Target     : sip:mtressler@flinchbot.com
Maximum Number of Allowed Calls : 5
Delegates                      : sip:byount@flinchbot.com
```

If you think you can get crafty and forward the missed call back to the SLA workgroup…well you can't. I mean, you can define the SLA identity as the `-MissedCallForwardTarget`. But after the call is missed, the caller's phone will continue to ring and ring. But any phone assigned as part of the SLA won't see the call anymore - the call disappears into Neverland.

Note that if the user in the –Identity parameter is enabled for voicemail then Transfer is not an available option and voicemail becomes the default option. Look at the image below. The operator-nashville@flinchbot.com account was not initially configured for voice mail. I therefore was able to successfully set the Missed Call Option to Forward. But once I enabled the account for voicemail, the following message began popping up whenever I looked at the configuration or made a change to the SLA:

Figure 8 – 27

```
PS C:\Users\flinchbot> Get-CsSlaConfiguration -Identity operator-nashville@flinc
hbot.com
WARNING: The missed call option was changed from "Forward" to Voicemail because
 Voicemail is enabled.

Busy Option                       : Forward
Target                            : sip:mtressler@flinchbot.com
Missed Call Option                : Forward
Missed Call Forward Target        : sip:mtressler@flinchbot.com
Maximum Number of Allowed Calls   : 5
Delegates                         :
```

The system will automatically set it to Voicemail and you can't change it to Forward without disabling Unified Messaging on the SLA account. You also can't get rid of the warning message because Voicemail is not an accepted value for the `–MissedCallOption` parameter.

You also can't set it to "BusySignal" or "Disconnect". So basically, once you've enabled Unified Messaging on the account, all missed calls go to that accounts voicemail.

Delegating Users

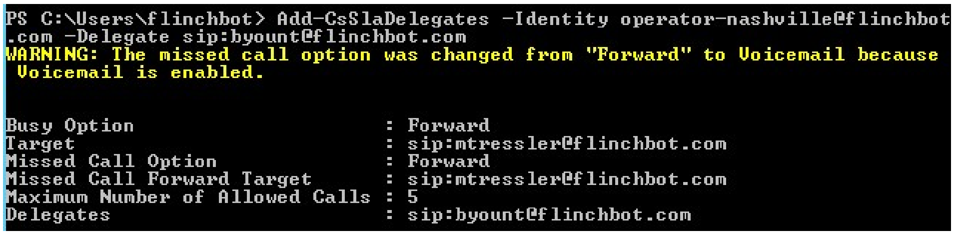
Once you have created the SLA, you probably want to add a few users to be part of the SLA. When a call comes in to the shared line, the phone will ring for those users. Users assigned to a SLA workgroup are called delegates.

You add delegates to a SLA workgroup using the `Add-CsSlaDelegates` PowerShell cmdlet.

If I want to add user byount to the Operator-Nashville@flinchbot.com workgroup, I run the following command:

```
Add-CsSlaDelegates -Identity operator-nashville@flinchbot.com
-Delegate sip:byount@flinchbot.com
```

Figure 8 – 28



```
PS C:\Users\flinchbot> Add-CsSlaDelegates -Identity operator-nashville@flinchbot
.com -Delegate sip:byount@flinchbot.com
WARNING: The missed call option was changed from "Forward" to Voicemail because
 Voicemail is enabled.

Busy Option                   : Forward
Target                        : sip:mtressler@flinchbot.com
Missed Call Option            : Forward
Missed Call Forward Target    : sip:mtressler@flinchbot.com
Maximum Number of Allowed Calls : 5
Delegates                     : sip:byount@flinchbot.com
```

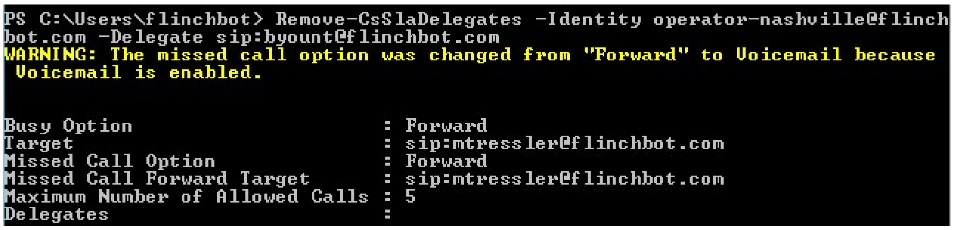
Note that you have to add the “sip:” in front of the delegates SIP address.

Removing a delegate is almost exactly the same as adding a delegate. The only difference is that you use the `Remove-CsSlaDelegates` cmdlet.

If I want to remove user byount from the Operator-Nashville@flinchbot.com workgroup, I run the following command:

```
Remove-CsSlaDelegates -Identity
operator-nashville@flinchbot.com -Delegate
sip:byount@flinchbot.com
```

Figure 8 – 29



```
PS C:\Users\flinchbot> Remove-CsSlaDelegates -Identity operator-nashville@flinch
bot.com -Delegate sip:byount@flinchbot.com
WARNING: The missed call option was changed from "Forward" to Voicemail because
 Voicemail is enabled.

Busy Option                   : Forward
Target                        : sip:mtressler@flinchbot.com
Missed Call Option            : Forward
Missed Call Forward Target    : sip:mtressler@flinchbot.com
Maximum Number of Allowed Calls : 5
Delegates                     :
```

Unassigned Numbers

This feature is used to handle calls to an unassigned number. An unassigned number is a phone number that hasn’t been assigned at all – neither to a user nor a response group nor an AutoAttendant, etc. It’s possible that an external user misdials a number or calls the number of an associate no longer with the company. Instead of just dropping the call (giving the caller a busy signal) we can route that call to an Operator who can then assist the caller in reaching the correct person.

You can build a really big, complicated list to handle your unassigned numbers. You would list each number that is currently unassigned. As a number gets assigned you remove it from the list. You add it back once it gets unassigned. This is a bunch of manual work.

But there is a trick that makes this list super simple to maintain:

Add your entire range of numbers all at once.

In essence, you are saying that all of your numbers are unassigned. Skype for Business is smart enough to first see if a number is actually assigned. If so, it routes it to the user or other endpoint. Only if it can't find a match for the number will it then check to see if the number is part of the unassigned number range. If it is, it then processes based on how you have your Unassigned Number Range and Announcements configured.

For more detail on this, see Appendix 2 – Call Routing.

Creating an Announcement

The first step in creating an unassigned number range is to create an Announcement.

Say what?

When a call gets classified as an unassigned number, a message will get played to the caller. This message can say anything you want. Generally, it will say something like "We're sorry but the number you have dialed is unavailable. Transferring you to the Operator".

Note that you can transfer a call right away without playing an announcement first.

Announcements also control the transfer settings.

We first have to create the announcement so we can then assign it to the number range. The easiest announcement to create is a Text-to-Speech announcement. A Text-to-Speech announcement will have a computer-generated voice read some text that you have defined.

You can only create Announcements using PowerShell.

The following shows the creation of an Announcement using the `New-CsAnnouncement` PowerShell cmdlet.

```
New-CsAnnouncement -Identity
ApplicationServer:skypepool.flinchbot.com -Name "Unassigned
Number-English" -TextToSpeechPrompt "We're sorry but the
number you have dialed is unavailable. We will now transfer
you to our Operator" -Language "en-US"
```

Note that last parameter. You can use text-to-speech in several languages. Here is one we could use for our office in Germany.

```
New-CsAnnouncement -Identity
ApplicationServer:skypepool.flinchbot.com -Name "Unassigned
Number-German" -TextToSpeechPrompt "Hau bloß ab du Drecksau."
-Language "de-DE"
```

Depending on where our range of unassigned numbers is located geographically, we can assign the correct language for the announcement.

Here is the list of supported languages.

Language Code	**Language**
ca-ES	Catalan, Spain
da-DK	Danish, Denmark
de-DE	German, Germany
en-AU	English, Australia
en-CA	English, Canada
en-GB	English, United Kingdom
en-IN	English, India
en-US	English, United States
es-ES	Spanish, Spain
es-MX	Spanish, Mexico
fi-FI	Finnish, Finland
fr-CA	French, Canada
fr-FR	French, France
it-IT	Italian, Italy
ja-JP	Japanese, Japan
ko-KR	Korean, Korea
nb-NO	Norwegian, Bokmal, Norway
nl-NL	Dutch, Netherlands
pl-PL	Polish, Poland
pt-BR	Portuguese, Brazil
pt-PT	Portuguese, Portugal
ru-RU	Russian, Russia
sv-SE	Swedish, Sweden
zh-CN	Chinese, People’s Republic of China
zh-HK	Chinese, Hong Kong SAR
zh-TW	Chinese, Taiwan

Importing an Announcement

While text-to-speech is pretty neat, it doesn't provide the most professional sound to your customers. It would be better if you could play an audio recording that sounds less robotic.

To do this, you need to record a professional audio file. You can do this yourself or hire a professional company like Worldly Voices (http://www.worldlyvoices.com/) to help you – especially useful if you need recordings in multiple languages.

There is no documentation detailing what specific audio format the file should be in other than "wav or wma". There is no guidance as to bitrate, mono or stereo, etc. As such, try using whatever .wav or .wma file you've created. If it doesn't work, go to http://online-convert.com and convert the file to something else. The following settings will work though may not provide the best audio resolution:

Windows Media Audio 9, 44 kHz, 16 bits, Mono, CBR, or 32 kbps

Once you have your audio file, you can copy it up to Skype for Business using the `Import-CsAnnouncementFile` cmdlet.

```
$a = Get-Content "c:\temp\Announcement-English.wma" -ReadCount
0 -Encoding Byte

Import-CsAnnouncementFile -Parent
ApplicationServer:skypepool.flinchbot.com -FileName
"English-Announcement.wma" –Content $a
```

Once the file has been copied in to Skype for Business, we can then create a new announcement using that audio file.

```
New-CsAnnouncement -Identity
ApplicationServer:Skypepool.flinchbot.com -Name "Unassigned
Number-English Recording" -AudioFilePrompt "English-
Announcement.wma"
```

Advanced Announcement Features

If you don't want to play an announcement and just want to send the caller immediately to the operator, you can do that by using the *–TargetUri* parameter.

```
New-CsAnnouncement -Identity
ApplicationServer:skypepool.flinchbot.com –Name "No
Announcement" -TargetURI sip:operator-nashville@flinchbot.com
```

If you want to play an audio file first and then transfer the call to the Operator, you would use this command:

```
New-CsAnnouncement -Identity
ApplicationServer:Skypepool.flinchbot.com -Name "Unassigned
Number-English Recording" -AudioFilePrompt
"English-Announcement.wma" –TargetURI
sip:operator-nashville@flinchbot.com
```

If you want to play a text-to-speech announcement in German and then forward the call to a specific phone number, you would use this command:

```
New-CsAnnouncement -Identity
ApplicationServer:skypepool.flinchbot.com -Name "Unassigned
Number-German-Operator" -TextToSpeechPrompt "Die von Ihnen
gewählte Nummer wurde nicht vergeben. Sie werden an den
Empfang weitergeleitet." -Language "de-DE" -TargetUri
sip:+12425550101@flinchbot.com;user=phone
```

I've created quite a few announcements now. To see what announcements you have you can run the `Get-CsAnnouncement` cmdlet.

Figure 8 – 30

```
PS C:\> Get-CsAnnouncement

Identity          : Service:ApplicationServer:skypepool.flinchbot.com/a0097a19-4161-4211-9e1a-4b39abe16f6a
Name              : Unassigned Number-English
AudioFilePrompt   :
TextToSpeechPrompt : We're sorry but the number you have dialed is unavailable. We will now transfer you to our
                    Operator
Language          : en-US
TargetUri         :
AnnouncementId    : a0097a19-4161-4211-9e1a-4b39abe16f6a

Identity          : Service:ApplicationServer:skypepool.flinchbot.com/0ad92259-2143-469f-a886-8b835a85f50c
Name              : Unassigned Number-German
AudioFilePrompt   :
TextToSpeechPrompt : Hau bloß ab du Drecksau.
Language          : de-DE
TargetUri         :
AnnouncementId    : 0ad92259-2143-469f-a886-8b835a85f50c

Identity          : Service:ApplicationServer:skypepool.flinchbot.com/33000289-c946-47eb-8b35-8f81566e9c09
Name              : Unassigned Number-English Recording
AudioFilePrompt   : English-Announcement.wma
TextToSpeechPrompt :
Language          :
TargetUri         :
AnnouncementId    : 33000289-c946-47eb-8b35-8f81566e9c09

Identity          : Service:ApplicationServer:skypepool.flinchbot.com/c64097f5-9bf2-424a-9cd7-a60660d62429
Name              : No Announcement
AudioFilePrompt   :
TextToSpeechPrompt :
Language          :
TargetUri         : sip:operator-nashville@flinchbot.com
AnnouncementId    : c64097f5-9bf2-424a-9cd7-a60660d62429

Identity          : Service:ApplicationServer:skypepool.flinchbot.com/7cadac50-81e4-447a-929f-99904b1b1a34
Name              : Unassigned Number-German-Operator
AudioFilePrompt   :
TextToSpeechPrompt : Die von Ihnen gewählte Nummer wurde nicht vergeben. Sie werden an den Empfang weitergeleitet.
Language          : de-DE
TargetUri         : sip:+12425550101@flinchbot.com;user=phone
AnnouncementId    : 7cadac50-81e4-447a-929f-99904b1b1a34
```

Editing an Announcement

To edit an announcement, use the `Set-CsAnnouncement` cmdlet. My boss got a very unhappy phone call from senior management in Germany. They want the German text-to-speech announcement changed immediately. To do this, I run the following command to change the text-to-speech text.

```
Set-CsAnnouncement –Identity
Service:ApplicationServer:skypepool.flinchbot.com/0ad92259-
2143-469f-a886-8b835a85f50c –TextToSpeechPrompt “Die von Ihnen
gewählte Nummer wurde nicht vergeben. Sie werden an den
Empfang weitergeleitet.“
```

If you’re feeling like having fun, write the text-to-speech in English but set the language to de-DE or zH-HK or some other one. Now see if you can understand the message when read back in a German, Spanish, or Chinese voice.

Removing an Announcement

To delete an announcement, use the `Remove-CsAnnouncement` cmdlet. If I want to remove the "Unassigned Number-German" Announcement I would run the following:

```
Remove-CsAnnouncement -Identity
Service:ApplicationServer:skypepool.flinchbot.com/0ad92259-
2143-469f-a886-8b835a85f50c
```

There is one additional thing I want to bring up here. While you can import an audio file using `Import-CsAnnouncementFile`, there is no way to either export or remove the announcement file.

So don't ever make a mistake or you'll have some "dead" files floating around.

Well what happens if you did make a mistake with `Import-CsAnnouncementFile` and you are super bummed about it. You just won't be able to sleep until you remove that file.

While there is no supported way to fix this, there is an unsupported way. The announcement files will get copied to your Skype for Business file share. If you look there, you will find the file and then you can delete it.

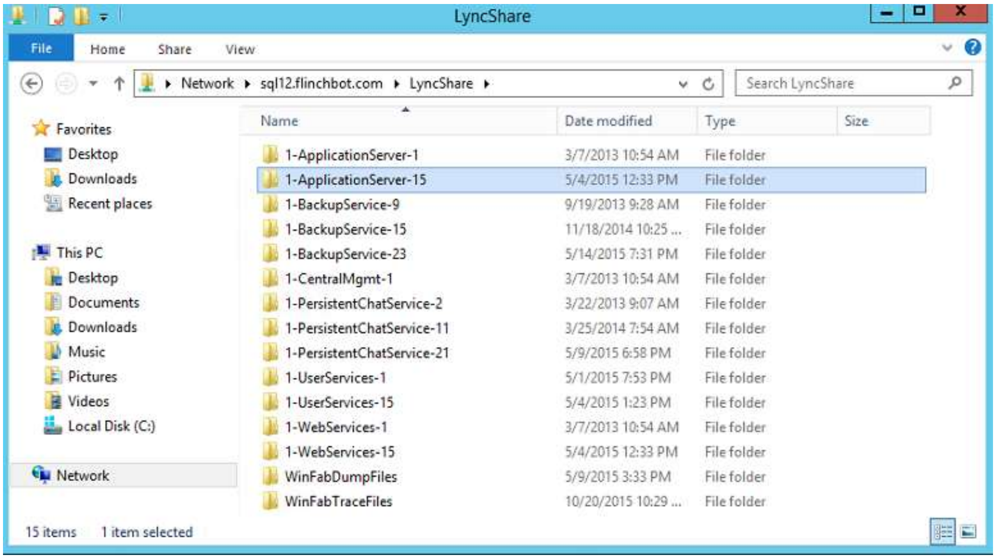
If you don't know what your file share is, you can either open Topology or run the following PowerShell.

```
Get-CsService -Filestore | where {$_.SiteID -eq
"Site:Nashville"} | Select UncPath
```

Figure 8 – 31

This tells me that the path to my file share is \\sql12.flinchbot.com\LyncShare. OK, so let's browse there with Windows Explorer.

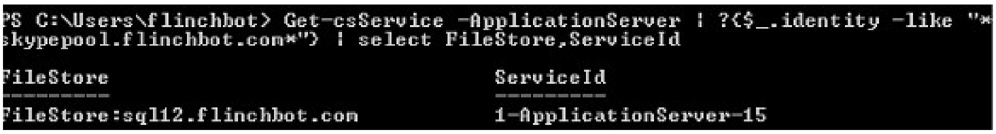
Figure 8 – 32



OK now what. How do I know which directory might contain my bad announcement file? Time to run a second PowerShell cmdlet.

```
Get-CsService -ApplicationServer | ?{$_.identity -like
"*skypepool.flinchbot.com*"} | select ServiceId
```

Figure 8 – 33



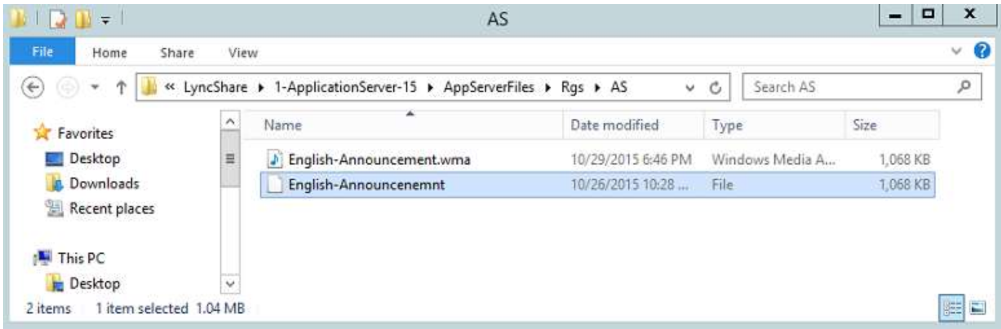
This tells me that the server hosting my file share is sql12.flinchbot.com and my “ServiceId” is “1-ApplicationServer-15”. The “ServiceId” just lets you know the name of the root folder in the share where Skype for Business is dumping files for the Nashville site.

I can now navigate to \\sql12.flinchbot.com\LyncShare\1-ApplicationServer-15 and begin finding the bad announcement file.

The files will get placed into the following path: \\sql12.flinchbot.com\LyncShare\1-ApplicationServer-15\AppServerFiles\Rgs\AS

And sure enough, I can see the bad announcement file and delete it. I forgot to give it a file extension and I butchered the spelling of “announcement”.

Figure 8 – 34



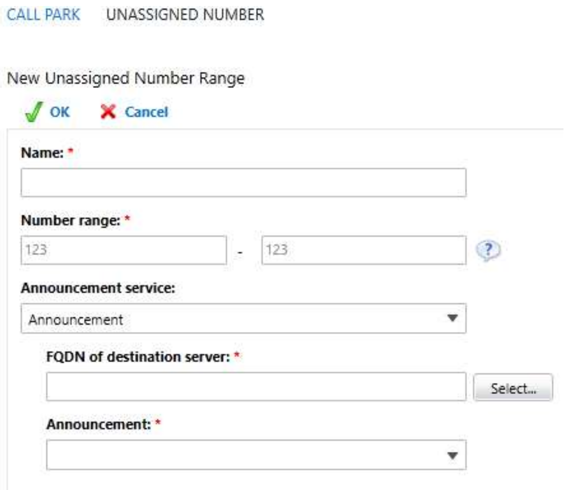
Keep in mind that this is an unsupported action. While I highly doubt this will ever cause a problem with your Skype for Business installation, it’s still up to you to decide if you want to go this route or just live with having an extra announcement file or two sitting around on your network.

Creating an Unassigned Number Range

Finally, we get to create an Unassigned Number Range. To create an Unassigned Number Range, open Control Panel, click on “Voice Features” and then click on “Unassigned Numbers” at the top of Control Panel.

Click the “New” button which brings up the “New Unassigned Number Range” screen.

Figure 8 – 35



The first thing to do is to give this Unassigned Number Range a name.

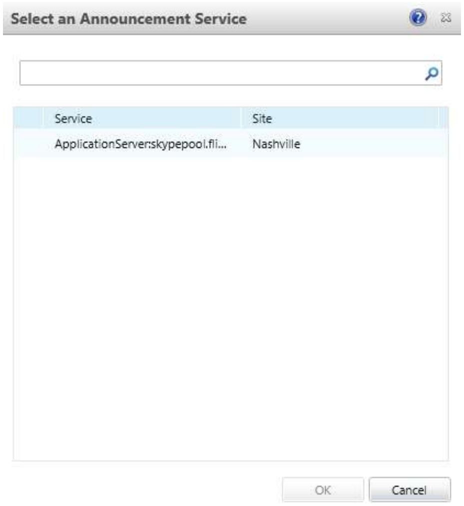
Next you assign the number range. The format is a little specific. Start the number you want to add with tel:+ and then the first number in the range. And yes, I insist it starts with tel:+ and not just tel: because all of our numbers are always in E.164 format so all numbers will start with a +.

For example purposes, let's say the Telephone Company has assigned us a range of DID/DDI's that start at 1-615-555-1000 and ends at 1-615-555-1999. So the first entry I add is tel:+16155551000. In the second field I enter tel:+16155551999.

You then have the choice of using an Announcement or sending calls directly to an Exchange UM AutoAttendant. For now we'll keep the default setting of Announcement.

For the FQDN of the destination server I pick which pool will handle a call to this unassigned number range. If you click the "Select..." button and you don't see anything, then you have most likely not created an Announcement yet.

Figure 8 – 36

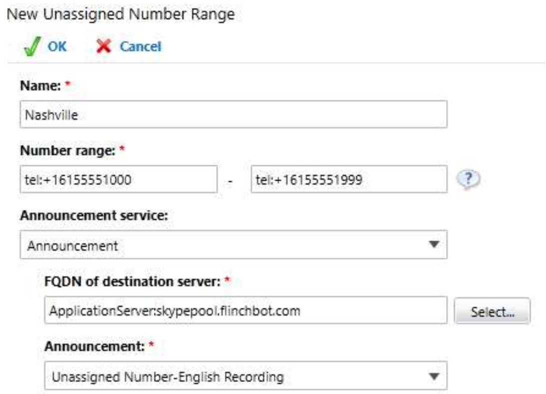


After selecting a server, the "Announcement" pull down will get populated with all of the announcements defined on that server (or more likely, pool).

Figure 8 – 37

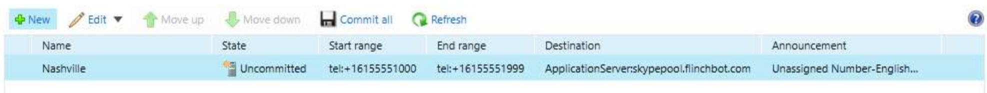
After selecting one of the Announcements, the New Unassigned Number Range now looks like this.

Figure 8 – 38



You can then click OK to return to the main Unassigned Number page. Click "Commit All" to save the Unassigned Number configuration.

Figure 8 – 39



To create this same Unassigned Number range using PowerShell you would use the following command.

```
New-CsUnassignedNumber -Identity "Nashville" -NumberRangeStart
"tel:+16155551000" -NumberRangeEnd
"+16155551999" -AnnouncementService
ApplicationServer:skypepool.flinchbot.com -AnnouncementName
"Unassigned Number-English Recording"
```

Forwarding to an AutoAttendant

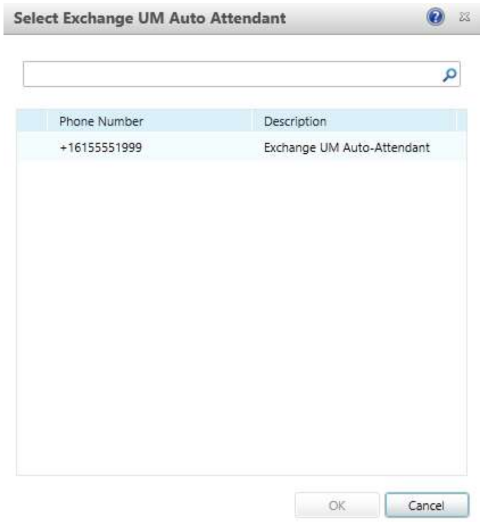
Returning to the "New Unassigned Number Range" screen, you could select Exchange UM instead of the default "Announcement".

If so, you can then forward a call directly to an Exchange UM AutoAttendent instead of sending the call to the Announcement service. After changing the drop down to "Exchange UM" you will see the following screen.

Figure 8 – 40

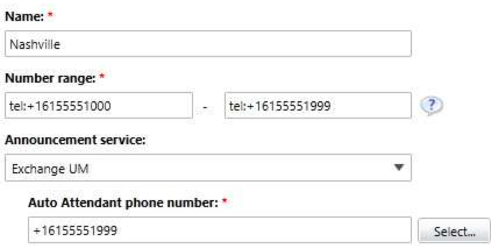
You now have to hit the "Select..." button to bring up a screen which will allow you to choose to which AutoAttendant you want to send the call.

Figure 8 – 41



After selecting my only AutoAttendant, here is how the New Unassigned Number screen looks.

Figure 8 – 42



I click OK and commit the change. Now when a call comes in to a number in the unassigned range, the calls gets immediately forwarded to the defined AutoAttendant.

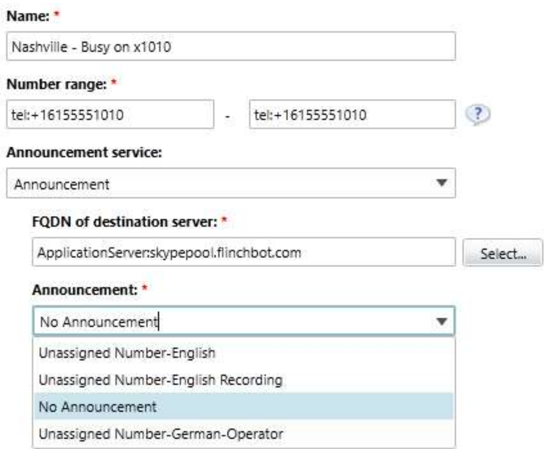
If you want to assign Exchange UM to your unassigned numbers via PowerShell, then here is what you need.

```
New-CsUnassignedNumber -Identity "Nashville" -NumberRangeStart
"tel:+16155551000" -NumberRangeEnd "+16155551999" -
ExUmAutoAttendantPhoneNumber "+16155551999"
```

Ordering

Just for fun, I'm going to create an unassigned number for one specific extension - +16155551010. If anyone calls this number, they will get a busy signal.

Figure 8 – 43



If you prefer PowerShell:

```
New-CsUnassignedNumber -Identity "Nashville - Busy on
1010" -NumberRangeStart "tel:+16155551010" -NumberRangeEnd
"+16155551010" -AnnouncementService
ApplicationServer:skypepool.flinchbot.com -AnnouncementName
"No Announcement"
```

I now have 2 Unassigned Number ranges.

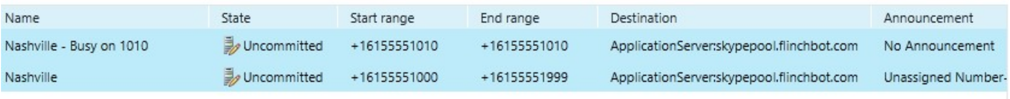
Figure 8 – 44

Name	State	Start range	End range	Destination	Announcement
Nashville	Committed	+16155551000	+16155551999	ApplicationServer:skypepool.flinchbot.com	Unassigned Number-
Nashville - Busy on 1010	Committed	+16155551010	+16155551010	ApplicationServer:skypepool.flinchbot.com	No Announcement

The only problem is that the second Unassigned Number Range will never be used. This is because the Unassigned Number ranges are analyzed in order from top to bottom. And since the top Unassigned Number Range includes +16155551010, the second Unassigned Number will never be analyzed as the first one will match.

Using the up and down arrows, I can rearrange these two so that the "Nashville – Busy on 1010" configuration will be above the "Nashville" range.

Figure 8 – 45



Name	State	Start range	End range	Destination	Announcement
Nashville - Busy on 1010	Uncommitted	+16155551010	+16155551010	ApplicationServer:skypepool.flinchbot.com	No Announcement
Nashville	Uncommitted	+16155551000	+16155551999	ApplicationServer:skypepool.flinchbot.com	Unassigned Number-

Now they are in the correct order. If you want to do this with PowerShell, use the `Set-CsUnassignedNumber` cmdlet with the *–Priority* parameter.

```
Set-CsUnassignedNumber -Identity "Nashville - Busy on
1010" -Priority 0
```

Summary

These voice features are pretty powerful tools that can certainly improve your call intake process as well as improve your customer relations. Call Park provides an easy way to route a call to a destination where people may be sharing a phone such as in a manufacturing facility or a grocery store.

Building on Call Park, Group Call Pickup lets coworkers answer a ringing phone without physically having to move. This doesn't encourage laziness. Rather it encourages customer calls being answered by a person instead of a machine.

Unassigned Numbers Ranges help incoming calls get routed to somewhere useful instead of just giving the caller a busy signal. You can have a custom friendly message played and then forward the call to a real person such as an Operator. Or you can send the call to an Exchange UM AutoAttendant so that the caller can then route him or herself to the correct resource. No matter how it's configured, Unassigned Numbers help callers get to the resource they want to get to in a simple manner.

Chapter 9 – Dial-In Conferencing

First off, what exactly is Dial-In Conferencing? This is the ability to join a Skype for Business meeting using a telephone. Let's say you are hosting a Skype for Business conference and you need a customer to join. While the customer could join using the Web App, more often than not they will want to join the meeting using their phone.

Dial-In Conferencing is the feature used to connect their phone call to your Skype for Business conference.

Creating Regions

In Chapter 3, I made reference to the "Dial in conferencing region" field and told you not to worry about it and leave it blank. Well, now it's time to worry about it.

What is a Dial-In Conferencing region? A region is anything you want it to be. It's just some text. A region is independent of an Active Directory site or a subnet or anything similar. A region could be "Earth". A region could be "Germany". A region could be "My bedroom in the basement of mom's house".

In practicality, a region is used to define where a dial-in conferencing phone number resides. So if I have a dial-in number in Indianapolis and one in Tokyo, the regions could be "Indianapolis" and "Tokyo" or they could be "United States" and "Japan" or they could be "North America" and "Asia". The regions are just a way to let people know where the dial-in conferencing phone number is located.

So where do you configure these region names? If you answer: "In the same place where you would configure the dial-in conferencing number" then you would be wrong. You actually configure them in Dial Plans. There is a reason for this that I will get to later.

In order to set a region, open any Dial Plan and in the "Dial-in conferencing region" box, type in anything you want. In the example below, I entered "United States". Also note that this is being set on the "Site:Nashville" dial plan.

Figure 9 – 1

You can add a region using PowerShell. To do this, use the Set-CsDialPlan cmdlet with the `DialinConferencingRegion` parameter.

```
Set-CsDialPlan -Identity
Site:Nashville -DialinConferencingRegion "United States"
```

If I want to see the existing dial-in conferencing regions, I can run this PowerShell command:

```
Get-CsDialPlan | Select-Object DialInConferencingRegion
```

After setting the region in the Site:Nashville dial plan, I run that command and see the following:

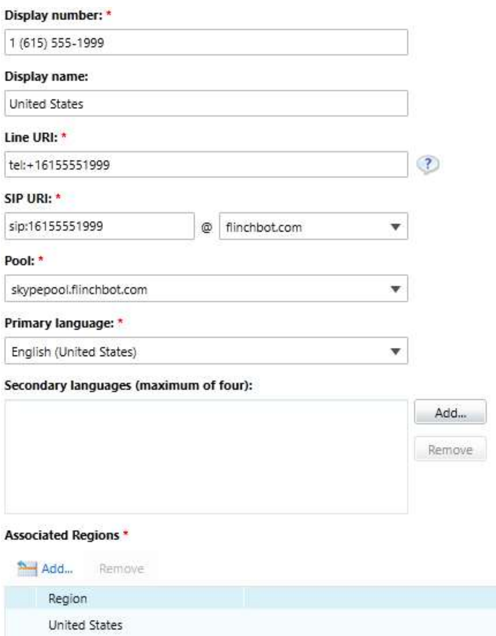
Figure 9 – 2

OK this is great. I have a region. Now how do I assign it to a Dial-In Conference number?

Creating Dial-In Conferencing Numbers

In Control Panel, navigate to the "Conferencing" section and then the "Dial-in Access Number" tab. Click "New".

Figure 9 – 3



Look at the above image. The "Display Number" can be anything you want. It can be in any format you want. In this case I formatted it to the standard way phone numbers in the North America Numbering Plan tend to be formatted. This Display number is what people will see when they get a conferencing invitation or look at the dial-in web page.

The Display Name field is essentially a comment so you know why this number was added.

Line URI must be the exact, normalized phone number that Lync will use to answer the call. This number needs to be exactly right. Note that you need to put “tel:” before the phone number.

The SIP URI can be anything you want. I often make it match the Line URI but you can use something else. This must follow the naming requirements for a proper SIP address so don’t use spaces or any weird characters.

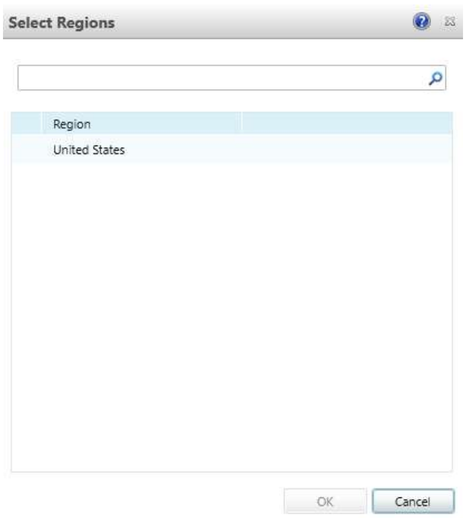
The Pool is used to tell Skype for Business which of your pools will receive the inbound call.

Finally, you can select a Primary language and up to four secondary languages for this dial-in access number.

And where do we define the region? All the way down at the bottom.

After clicking the Add button you are given a list in which to select your region. In this case, we only have one available region.

Figure 9 – 4



Once you select that region, you are returned to the main page for adding a Dial-In Access number. Click "Commit" and you now have a dial-in access number.

For those excited by PowerShell, you can perform all of the above using the following PowerShell command:

```
New-CsDialInConferencingAccessNumber -PrimaryUri
"sip:16155551999@flinchbot.com" -DisplayNumber "1 (615)
555-1999" -DisplayName "United States" -LineUri
"tel:+16155551999" -Pool
skypepool.flinchbot.com -PrimaryLanguage "en-US" -Regions
"United States"
```

So how can we see what this now looks like? Point a web browser to your dial-in simple URL. In my case, that is https://dialin.flinchbot.com and I see the following:

Figure 9 – 5

If you don't know your dial-in Simple URL, you can run the following PowerShell command to find it:

```
Get-CsSimpleUrlConfiguration
```

You should be able to see it on the first line of the `SimpleURL` parameter.

Figure 9 – 6

You can get a list of your defined dial-in access number by running the `Get-CsDialInConferencingAccessNumber` cmdlet.

Figure 9 – 7

```
PS C:\Users\flinchbot> Get-CsDialInConferencingAccessNumber

Identity              : CN={c3346c16-e8eb-4ce2-8d72-217145fdf977},CN=Application
                         Contacts,CN=RTC
                        Service,CN=Services,CN=Configuration,DC=flinchbot,DC=com
PrimaryUri            : sip:16155551999@flinchbot.com
DisplayName           : United States
DisplayNumber         : 1 (615) 555-1999
LineUri               : tel:+16155551999
PrimaryLanguage       : en-US
SecondaryLanguages    : {}
Pool                  : skypepool.flinchbot.com
HostingProvider       :
Regions               : {United States}
ExternalAccessPolicy  :
```

Default Dial-In Number

More has happened here beyond just having created a new dial-in number for the Nashville site.

One important thing to note is that there is precedence in conference numbers. At this point, all of my users in the Nashville site now have this set as their default conferencing number. Note that this isn't limited to the users assigned the Site:Nashvile dial plan. Rather, it's *all users in the site, regardless of which dial plan they have*.

The user *mtressler* is assigned the Indianapolis dial plan. I have not yet created a region for the Indianapolis dial plan nor have I created a dial in conferencing number for Indianapolis.

So does *mtressler* have a default dial-in conferencing number? He sure does. Because even though he works in Indianapolis and has the Indianapolis dial plan assigned and he's registered to the SBA in Indianapolis, he is still a member of the Nashville Site.

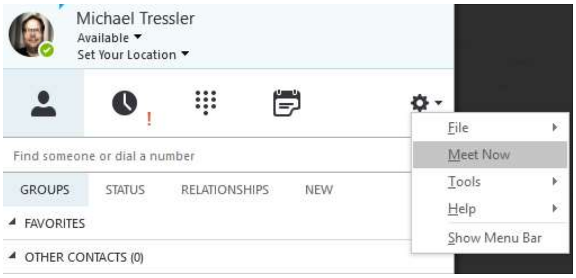
The below table highlights the Pool configuration in my Topology. The Indianapolis SBA is a sub-site of the Nashville site. So anyone in Indianapolis is a member of both the Nashville site and the Indianapolis site.

Site	Location	Server	Dial plan
Nashville	Nashville	Enterprise Edition Pool	Site:Nashville
	Indianapolis	Survivable Branch Appliance	Indianapolis

How can a user see what their current dial-in conferencing number is? The easiest way is to spin up an ad-hoc meeting via the Skype for Business client.

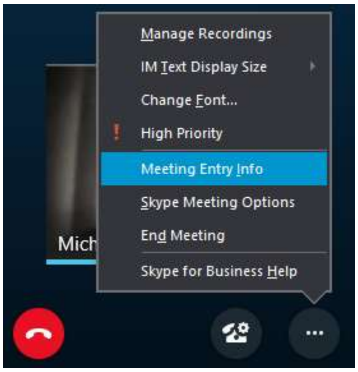
To do this, click on "Meet Now" in the options pull down (click the down arrow).

Figure 9 – 8



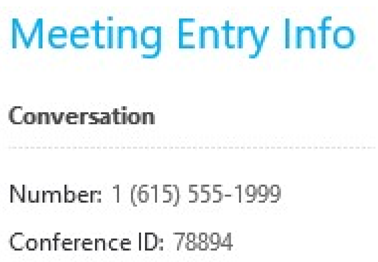
This will fire up a new meeting. From here, click on the three dots in the bottom right and select "Meeting Entry Info".

Figure 9 – 9



This pops up a window showing the dial in number and the conference ID.

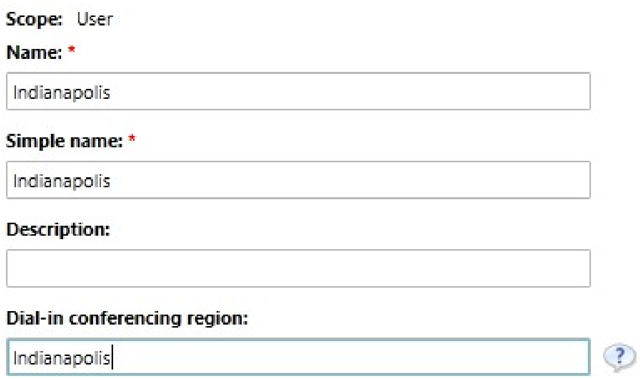
Figure 9 – 10



So even though my user has the Indianapolis dial plan assigned, the number from the Site:Nashville dial plan rolled down to him.

So what happens if I create a new Dial Plan. Does the Site:Nashville region still define the “Dial in conferencing region” for users assigned to a user-level dial plan? Let me edit the Indianapolis dial plan and set a region.

Figure 9 – 11

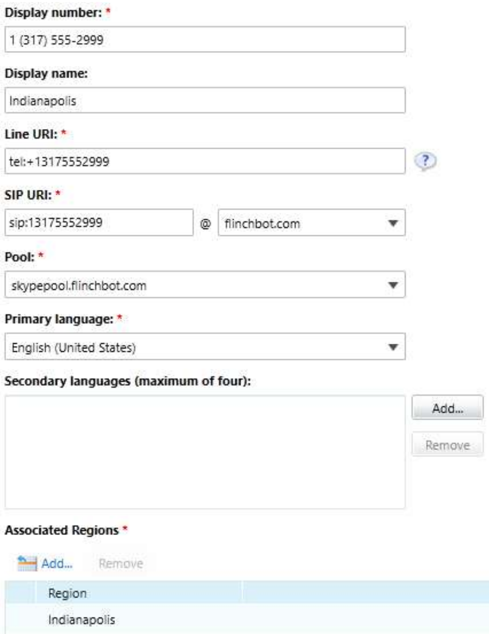


I can also do this via PowerShell –

```
Set-CsDialPlan -Identity
Indianapolis -DialinConferencingRegion "Indianapolis"
```

I then create a new dial in access number.

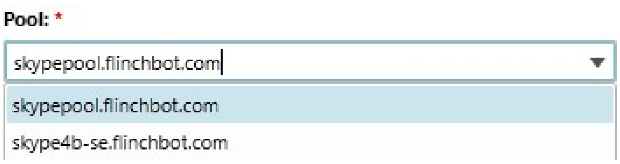
Figure 9 – 12



There is one thing I want to point out here. Note that the "Pool" value is set to skypepool.flinchbot.com and not to the Survivable Branch Appliance in Indianapolis (IND-SBA.flinchbot.com). This is because SBA's do not run any conferencing services. As such, the dial-in access number cannot be hosted in Indianapolis. It must be hosted by a Standard or Enterprise Edition server in the parent pool.

As can be seen below, none of my defined SBA's are an available selection.

Figure 9 – 13



Here is the PowerShell to create the Indianapolis dial-in number:

```
New-CsDialInConferencingAccessNumber -PrimaryUri
"sip:13175552999@flinchbot.com" -DisplayNumber "1 (317)
555-2999" -DisplayName "Indianapolis" -LineUri
```

```
"tel:+13175552999" -Pool
skypepool.flinchbot.com  -PrimaryLanguage "en-US" -Regions
"Indianapolis"
```

After waiting about 5 minutes I reload the https://dialin.flinchbot.com page and I see that I now have a new entry in my list.

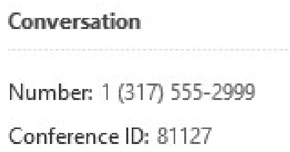
Figure 9 – 14


Conference Dial-in Numbers

Region	Number	Available Languages
Indianapolis	1 (317) 555-2999	English (United States)
United States	1 (615) 555-1999	English (United States)

After signing out and back in to the Skype for Business client, I create a “Meet Now” meeting. Checking the Meeting Entry information, I can see it is now updated to the Indianapolis number.

Figure 9 – 15



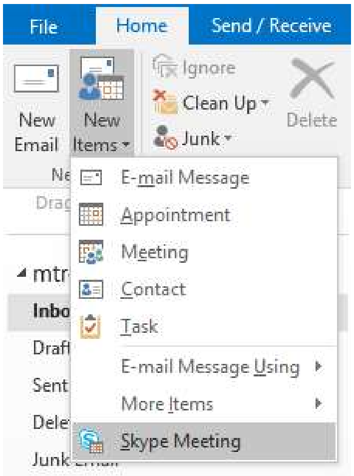
Now any user in the Nashville site that has the Indianapolis Dial Plan will get the Indianapolis number. Anyone else in the Nashville Site gets the Nashville (“United States”) dial in number.

Overriding the Default Dial-In Number

Users have the option to set a different default dial-in number than what is assigned to them via their Dial Plan. In order to do this, you need to use the Skype for Business meeting plugin in the Outlook client.

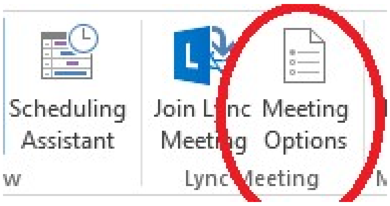
Start by creating a new Skype Meeting.

Figure 9 – 16



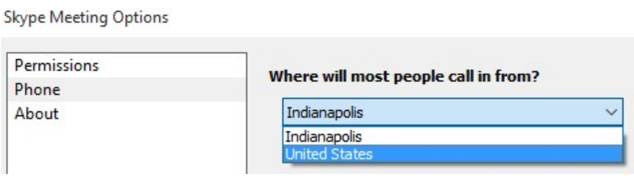
This open a new meeting window with your meeting information automatically populated into the meeting invite. From here, click on the "Meeting Options" icon in the Outlook toolbar.

Figure 9 – 17



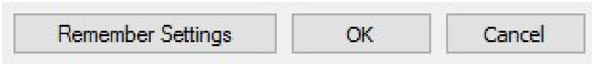
Next, click on the Phone option on the left. You can then change the phone number by changing the value in the "Where will most people call in from" pulldown. Below I can flip between "Indianapolis" and "United States".

Figure 9 – 18



At the bottom of this screen, I can either click “OK” or “Remember Settings”.

Figure 9 – 19



Clicking “OK” changes the dial in number for just this meeting invite. Clicking “Remember Settings” will change my default to whatever I have selected. So if my default is “Indianapolis”, I can change this to “United States” and click “Remember Settings”. From now on, all of my meetings will default to the “United States” number instead of “Indianapolis”.

Adding Multiple Dial-In Numbers

If you have an office (or, more likely, a region) with 3 dial plans, you have to make sure that the region on all 3 dial plans says the same thing. It’s a bit redundant to have to manually type in the same region value into each dial plan.

So use PowerShell!

Below are three dial plans for locations near Indianapolis. Without specifically setting “Indianapolis” as the dial-in conferencing region, these locations would all default to the “Nashville” region as defined in the site-dial plan Nashville.

```
Set-CsDialPlan -Identity
"Bloomington" -DialInConferencingRegion "Indianapolis"

Set-CsDialPlan -Identity
"Plainfield" -DialInConferencingRegion "Indianapolis"

Set-CsDialPlan -Identity
"Noblesville" -DialInConferencingRegion "Indianapolis"
```

Enabling Users Without Enterprise Voice

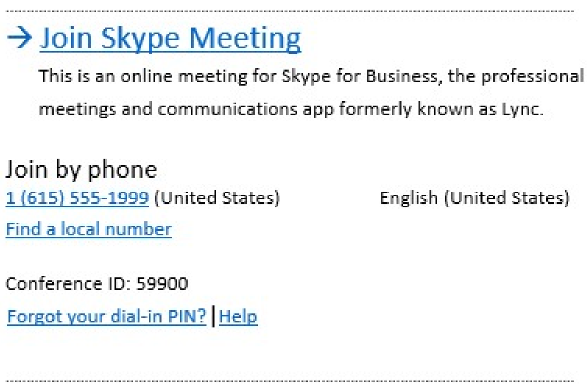
Can you assign a dial-in conferencing number to users who are not enabled for Enterprise Voice? Yes, you can. And it is really easy. Just assign them a Dial Plan.

To demonstrate this, I took the *mtressler* account and changed the Telephony setting from "Enterprise Voice" to the "PC-to-PC only" setting. I then signed out of Skype for Business and closed Outlook. I then signed back into Skype for Business and started Outlook again.

Finally, I created a new Skype Meeting.

Does the image below match what you were expecting to see?

Figure 9 – 20



My user is *not* configured for Enterprise Voice yet a phone number is listed in the invite. This is because the user is configured for the "default" dial plan which, in this case, is the "Site:Nashville" Dial Plan.

And if I wanted to give this user a different default phone number? I would assign him a different Dial Plan. So the Dial Plan is used for dial-in conferencing purposes even if the user is not enabled for Enterprise Voice.

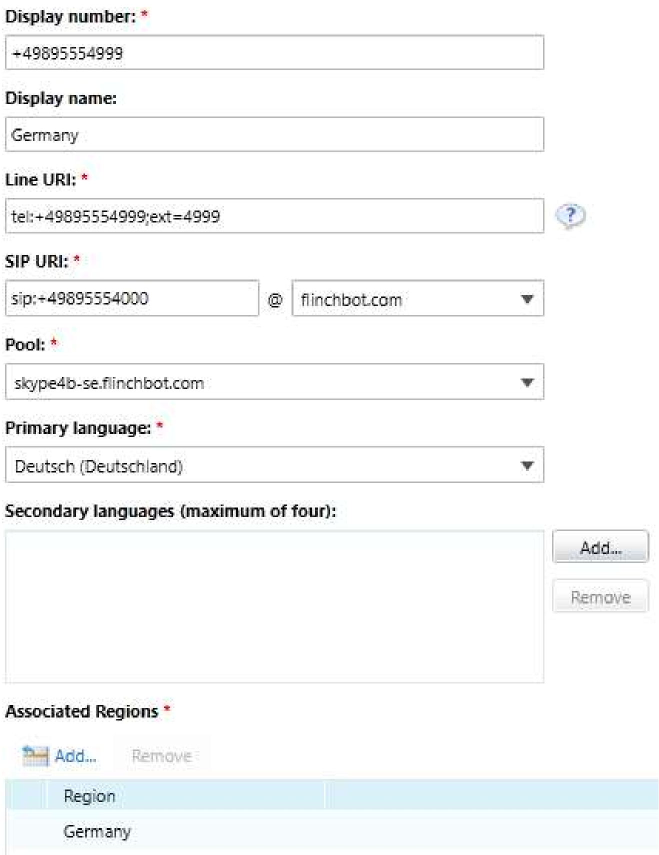
Multilingual Dial-In Numbers

If you remember when we created the dial in access numbers, there was an option to select a language. So far I have only used English. Can we create a dial in conferencing number in German? And what happens if someone calls in who doesn't speak German?

Setting the language of the dial-in access number defines the default language of the recording played when you call the number. This is the message that (in English) starts with "Welcome to the audio conferencing center".

I created a new region and dial-in number for an office in Germany and set the language to Deutsch (German).

Figure 9 – 21



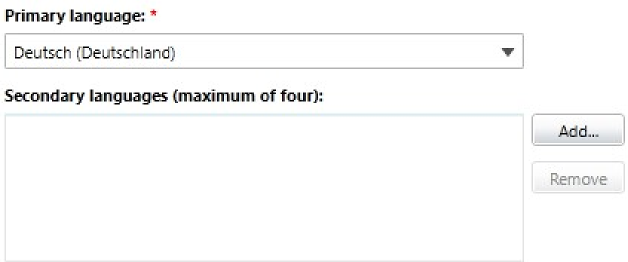
Note that I added the ";ext=" to the end of the LineURI. This is supported, just be aware that you may need to add custom entries to other dial plans to support this. For example, the Nashville dial plan appends +1615555 to 4 digit numbers. So if a Nashville user dials 4999, they will actually call +16155554999. Since this doesn't match the LineURI for the German dial in access number, they will not get connected to the conference. Rather, you will need to add a normalization to the Nashville dial plan so that 4999 gets converted to +49895554999.

When I call the German conference number I now hear "Wilkommen beim Center für audiokonferenzen". So that's an easy way to set the language when dialing in.

But what about setting a secondary language? If someone doesn't speak German, how can they understand the audio prompts?

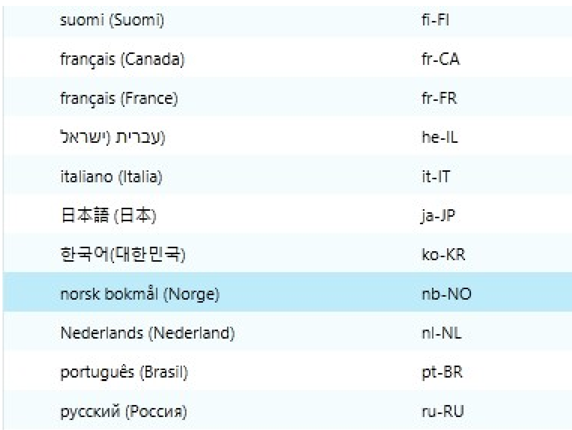
I added a secondary language by editing the dial in access number and then clicking the "Add..." button next to the "Secondary languages" box.

Figure 9 – 22



Clicking this box brings up an impressive list of languages. Below is a partial list of available languages.

Figure 9 – 23



suomi (Suomi)	fi-FI
français (Canada)	fr-CA
français (France)	fr-FR
עברית (ישראל)	he-IL
italiano (Italia)	it-IT
日本語 (日本)	ja-JP
한국어(대한민국)	ko-KR
norsk bokmål (Norge)	nb-NO
Nederlands (Nederland)	nl-NL
português (Brasil)	pt-BR
русский (Россия)	ru-RU

I went ahead and added English, Spanish, and French to the list of secondary languages.

Figure 9 – 24

These languages can also be set via PowerShell by using the `Set-CsDialInConferencingAccessNumber`. The only issue is getting the `-Identity` value. To get this value, first run the `Get-CsDialInConferencingAccessNumber` cmdlet. As you can see below, the Identity is pretty random.

Figure 9 – 25

With the Identity found, you can now run the following PowerShell to set the additional secondary languages. I'm trying to avoid having to deal with that super-long Identity by getting a little advanced with the PowerShell. This shouldn't be too crazy for you. You're smart!

```
$a = (Get-CsDialInConferencingAccessNumber | Where
{$_.DisplayName -eq "Germany"}).identity

Set-CsDialInConferencingAccessNumber -Identity
$a -SecondaryLanguages "en-GB","es-ES","fr-FR"
```

An easier way is to use the PrimaryUri as the Identity. All of the above PowerShell can be collapsed to this:

```
Set-CsDialInConferencingAccessNumber -Identity
'sip:+49895554999@flinchbot.com' -SecondaryLanguages
"en-GB","es-ES","fr-FR"
```

Now that I have all of these languages defined, how does this all work? Well, when you call into the conferencing number, the default language gets played (in this case – German).

Now you just wait through the first message in German. If you wait, eventually you will hear a menu played.

It will say things like "*Um auf Deutsch fortzufahren, drucken sie die Eins; pour les français, appuyez sur 3*; *para español, presione 3; for English, press 4*."

(*Apologies if there are language mistakes there. I only speak English well and I'm not as good at German as I would like to be. My Spanish is terrible and my French is non-existent.*)

If you speak one of those languages, just wait until you hear something you understand and then press the correct button. In my case, I would press 4 to continue to join the conference with English prompts (with a British accent).

Unfortunately, there is no shortcut to quickly skip to your language. You'll just have to wait.

Scoping Dial-In Numbers

By default, every dial-in access number gets added to the global list of available numbers. This means that users at any Skype for Business site will see all of the available dial in numbers when accessing the dial-in web page.

But what happens, for example, if you have a toll-free number that you only want to make available to one country? Let's take the United States for example.

I have one site defined in Nashville, Tennessee. I only want to advertise the 800 number in the United States. I don't want users of my other site in Europe to see this number and dial in using that 800 number.

Fortunately, there is a way to do this, but you have to do this in PowerShell. The `New-CsDialInConferencingAccessNumber` has a parameter called `ScopeToSite`. If you define a dial in access number with the `ScopeToSite` parameter, then this number is only visible on the dial-in page for that site.

To create a new dial in access number that is only scoped to my site in the United States, I could run the following:

```
New-CsDialInConferencingAccessNumber -PrimaryUri
'sip:18005552999@flinchbot.com' -LineUri
'tel:+18005552999'  -Pool 'skypepool.flinchbot.com' -
DisplayNumber '1 (800) 555 2999' -PrimaryLanguage 'en-US'  -
Regions "United States" -ScopeToSite
```

Any user in the Nashville Site, that is, the site that houses skypepool.flinchbot.com, will see that 800 number. Users from a different site will not see that number.

If I look at the dial-in web page for the Nashville site, I see the 800 number for the US.

Figure 9 – 26

Conference Dial-in Numbers

Region	Number	Available Languages
United States	1 (800) 555 2999	English (United States)

The Munich Site has a number for Germany but does not show the 800 number for the United States.

Figure 9 – 27

Conference Dial-in Numbers

Region	Number	
Germany	+49895554999	Deutsch (Deutschland)
Indianapolis	1 (317) 555 2999	English (United States)
United States	1 (615) 555-1999	English (United States)

Note that the Munich Site has a number for Germany not seen on the US site above. It also has a number for Indianapolis which is in the United States.

Why does the Munich site show a number in the United States that the Nashville site does not?

Once you add a site-scoped dial in number to a site, it overrides all of the global dial in numbers. So be careful with the –ScopeToSite parameter as it not only hides a number from a separate site, but it will remove all numbers that aren't scoped to the site.

What if I want the Indianapolis number to show up in the Nashville site but not in Munich?

You guessed it. Set the `ScopeToSite` parameter. (*And just for fun, I'm going to use the full Identity this time just to show different ways to do it*).

```
Set-CsDialInConferencingAccessNumber -Identity "CN={1b3e5662-a552-460c-9c58-4bb9f9e655fe},CN=Application Contacts,CN=RTC Service,CN=Services,CN=Configuration,DC=flinchbot,DC=com"
ScopeToSite
```

The Indianapolis number now shows up on the Nashville site's dial-in page.

Figure 9 – 28

Conference Dial-in Numbers

Region	Number	Available Languages
Indianapolis	1 (317) 555 2999	English (United States)
United States	1 (800) 555 2999	English (United States)

Similarly, It's been removed from the Munich site.

Figure 9 – 29

Conference Dial-in Numbers

Region	Number	
Germany	+49895554999	Deutsch (Deutschland)
United States	1 (615) 555-1999	English (United States)

Notice that there is still a United States number in Munich. This is because I have not scoped that number to a site so it will show up on the global list along with the number in Germany.

So how do you get rid of a site scoping? How can you return a number to be a global dial in access number?

Simply replace the `ScopeToSite` parameter with `ScopeToGlobal`. One other thing to note is that you can't see in either Control Panel or in PowerShell if a number is scoped to a site. The only way to tell is to look at your dial-in pages and figuring it out on your own.

Re-Ordering Dial-In Numbers

By default, the dial-in page is sorted by the region value. While we can't change the sorting for the region, we can change the ordering of how the individual numbers within a region are displayed.

I've removed all of the site-level scoping and returned all of my dial-in number to a global scope. My dial-in page now looks like this:

Figure 9 – 30

Conference Dial-in Numbers

Region	Number	
Germany	+49895554999	Deutsch (Deutschland),
Indianapolis	1 (317) 555 2999	English (United States)
United States	1 (615) 555-1999	English (United States)
	1 (800) 555 2999	English (United States)

I want to move the 800 toll free number in the United States Region to be above the "1 (615) 555-1999" number. I can do this by using the `Priority` and `ReorderedRegion` parameters.

The `Priority` is used to set where you want the number in the list, with 0 being at the top of the list, 1 being second in the list, etc.

The `ReorderedRegion` parameter is used to specify which region you want to reorder.

So in the case of moving the 800 toll free United States number to the top of the list, I run the following command.

```
Set-CsDialInConferencingAccessNumber -Identity
"sip:18005552999@flinchbot.com" -Priority 0 -ReorderedRegion
'United States'
```

> This is how reordering is supposed to work. However, as of this writing, there is a bug in Skype for Business Server 2015. Whenever you try to reorder a dial-in number, you get an error. However, the same command works in Lync 2013. So until this bug gets fixed, the only way to re-order these numbers is via a Lync 2013 pool you hopefully still have laying around.

Moving Dial-In Numbers

What do you do if you want to move a dial in conferencing number from one pool to another pool? For this scenario, you can use the `Move-CsApplicationEndpoint` cmdlet.

To move a Dial-In Conferencing access number, you need to know the SIP Address assigned to the number and the name of the pool to which you want to move the number.

You can get the SIP address for your dial in numbers by running:

```
Get-CsDialInConferencingAccessNumber | Select-Object
Displayname,Pool,PrimaryUri | Out-GridView
```

Figure 9 – 31

DisplayName	Pool	PrimaryUri
United States	skypepool.flinchbot.com	sip:16155551999@flinchbot.com
Indianapolis	skypepool.flinchbot.com	sip:13175552999@flinchbot.com
Munich	skype4b-se.flinchbot.com	sip:+49895554999@flinchbot.com
	skypepool.flinchbot.com	sip:18005552999@flinchbot.com

Now that you know the SIP Address for the number you want to move, you can run the `Move-CsApplicationEndpoint` cmdlet.

In my case, I want to move the Indianapolis number from the skypepool.flinchbot.com pool to the skype4b-se.flinchbot.com pool.

```
Move-CsApplicationEndpoint -Identity
sip:13175552999@flinchbot.com -TargetApplicationPool
skype4bse.flinchbot.com
```

DTMF Commands

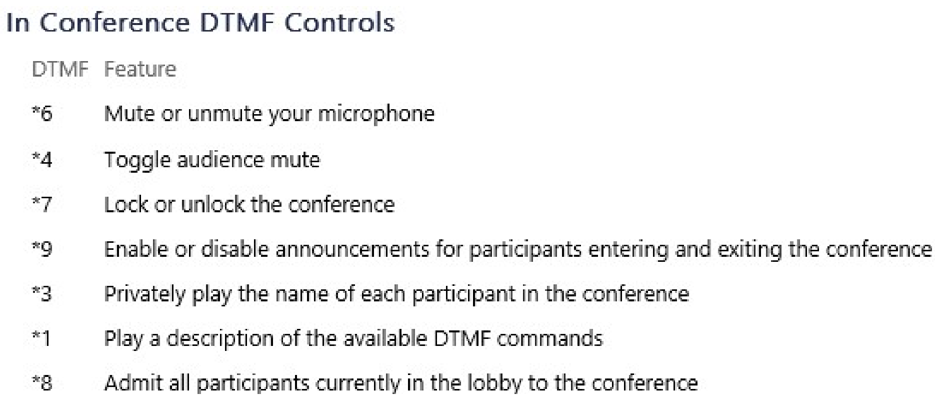
When joining a conference by phone, it would be nice to be able to control the meeting a little. Fortunately, Skype for Business conferencing supports the use of DTMF tones to do some management.

Below is a table showing the default DTMF commands and what they do.

*1	Play a description of the available DTMF commands
*3	Privately play the name of each participant in the conference
*4	Toggle audience mute (leaders only)
*6	Mute or unmute your microphone
*7	Lock or unlock the conference (leaders only)
*8	Admit all participants currently in the lobby to the conference (leaders only)
*9	Enable or disable announcements for participants entering and exiting the conference (leaders only)

You can look up this list by going to your dial-in page and scrolling down past the dial in numbers.

Figure 9 – 32



In Conference DTMF Controls

DTMF	Feature
*6	Mute or unmute your microphone
*4	Toggle audience mute
*7	Lock or unlock the conference
*9	Enable or disable announcements for participants entering and exiting the conference
*3	Privately play the name of each participant in the conference
*1	Play a description of the available DTMF commands
*8	Admit all participants currently in the lobby to the conference

You can also see these values by running the `Get-CsDialinConferencingDtmfConfiguration` cmdlet.

Figure 9 – 33

```
PS C:\Users\flinchbot> Get-CsDialinConferencingDtmfConfiguration

Identity                              : Global
CommandCharacter                      : *
MuteUnmuteCommand                     : 6
AudienceMuteCommand                   : 4
LockUnlockConferenceCommand           : 7
HelpCommand                           : 1
PrivateRollCallCommand                : 3
EnableDisableAnnouncementsCommand     : 9
AdmitAll                              : 8
OperatorLineUri                       :
```

You can change these mappings using the PowerShell cmdlet `Set-CsDialInConferencingDtmfConfiguration`. You might want to change these defaults if you are migrating from a different conferencing system that used different mappings. Your users are used to the DTMF mapping from your prior conferencing system and you want to make it easier for them to learn the new system.

I'll give a quick example of how to do this. Let's say my Munich office just migrated to Skype for Business conferencing. In their old system, they used *2 to mute or unmute their microphone. As seen above, Skype for Business uses *5.

To create a new mapping for the Munich site, I run the following command:

```
New-CsDialInConferencingDtmfConfiguration -Identity
"Site:Munich"
Set-CsDialInConferencingDtmfConfiguration -Identity
"Site:Munich" -MuteUnmuteCommand 2
```

The first command is used to create a new DTMF configuration for the Munich site. The second command enables *2 to be used for muting and unmuting the microphone. Note that the "*" is assumed.

Now I will review the settings for the Munich DTMF Configuration by running the `Get-CsDialInConferencingDtmfConfiguration` cmdlet.

Figure 9 – 34

```
PS C:\Users\flinchbot> Get-CsDialInConferencingDtmfConfiguration -Identity "Site
:Munich"

Identity                               : Site:Munich
CommandCharacter                       : *
MuteUnmuteCommand                      : 2
AudienceMuteCommand                    : 4
LockUnlockConferenceCommand            : 7
HelpCommand                            : 1
PrivateRollCallCommand                 : 3
EnableDisableAnnouncementsCommand      : 9
AdmitAll                               : 8
OperatorLineUri                        :
```

If I want to disable the use of a DTMF command, I set that attribute to $Null. For example, if I want to disable the ability to play the automated help – because I hate my users, apparently – I would run the following command:

```
Set-CsDialInConferencingDtmfConfiguration -Identity
"Site:Munich" -HelpCommand $Null
```

Figure 9 – 35

```
Identity                               : Site:Munich
CommandCharacter                       : *
MuteUnmuteCommand                      : 2
AudienceMuteCommand                    : 4
LockUnlockConferenceCommand            : 7
HelpCommand                            :
PrivateRollCallCommand                 : 3
EnableDisableAnnouncementsCommand      : 9
AdmitAll                               : 8
OperatorLineUri                        :
```

Now that I have shown you this, let me advise you against ever using it. While it's nice to make the transition easier for users, it also makes them used to "nonstandard" DTMF mappings, at least as far as Skype for Business Server is concerned.

The issue is this: If a user from Munich now joins a meeting hosted in the Nashville site, the DTMF tones will be at the default values and not at the custom values set for the Munich users. Taken even further, if my Munich users join a Skype for Business conference hosted by a different company, it's highly likely that the other company will also be using the default DTMF mappings.

In short, keep things default if you can. It's nice to know you can change the mappings but I've never actually done it in a production environment.

PinAuthType

New to Skype for Business Server is the PinAuthType setting.

PinAuthType can be set to two values:

- "OrganizerOnly" - The system will no longer prompt the users to enter a leader PIN if an authenticated user has already activated the meeting.
- "Everyone" - The system will prompt the users to enter a leader PIN even if an authenticated user has already activated the meeting.

A Skype for Business meeting does not become active until an authenticated user joins the meeting. Until an authenticated user joins, callers sit in limbo - which isn't necessarily the Lobby if "bypass the lobby" is enabled for PSTN callers. This is to prevent external users from just dialing in to a meeting and starting to talk which could help prevent meeting fraud.

An authenticated user is anyone who belongs to your organization and logs in with their Skype for Business client or someone who dials in and enters their extension and PIN via the phone. In other words, a user who is configured in your Skype for Business environment must join a meeting to activate it. What PinAuthType does is to remove the prompt for leader credentials after one authenticated user has joined the meeting.

To clarify this setting further, consider the following example where "OrganizerOnly" is the PinAuthType.

- Alice is the first person to join the meeting. She dials in from her mobile phone and provides the conference ID. She gets prompted "If you are the leader, please press * now". But since she is not the organizer she just waits on the line.
- Bob joins next by dialing in. He enters the Conference ID and then also provides his extension and PIN when prompted. He is now an authenticated user in the meeting. As soon as he joins, Alice is admitted into the meeting as well. Alice and Bob can now talk to each other.

- Charlie is the last person to join the meeting. He dials in and provides his Conference ID. He does not hear the "If you are a leader..." message due to the PinAuthType setting.

Setting PinAuthType to OrganizerOnly enables this feature. Setting PinAuthType to Everyone keeps the default setting where every caller is asked if they are a leader.

In order to set the PinAuthType value, you use either the `New-CsDialInConferencingConfiguration` cmdlet or the `Set-CsDialInConferencingConfiguration` cmdlet. You cannot set this value via the Control Panel.

Also note that this setting is a site-wide setting. You cannot enable or disable this feature for specific dial-in access numbers within a site.

In the below example, I create a new dial-in configuration for the Nashville Site and set the `PinAuthType` value to `OrganizerOnly`.

```
New-CsDialInConferencingConfiguration -Identity Site:Nashville
-PinAuthType "OrganizerOnly"
```

Figure 9 – 36

```
PS C:\> New-CsDialInConferencingConfiguration -Identity Site:Nashville -PinAuthT
ype "OrganizerOnly"

Identity                                : Site:Nashville
EntryExitAnnouncementsType              : UseNames
EnableNameRecording                     : True
EntryExitAnnouncementsEnabledByDefault  : False
PinAuthType                             : OrganizerOnly
```

If I want to change the value back to the default setting, I use the `Set-CsDialInConferencingConfiguration` cmdlet and set the `PinAuthType` parameter to “Everyone”.

```
Set-CsDialInConferencingConfiguration -Identity Site:Nashville
-PinAuthType "OrganizerOnly"
```

Chapter 10 – End User Configuration

Now that you've configured all of these dial plans and routes and trunk configurations, it would be nice to be able to place a call, right? I mean, the whole point of all of this is so that your users can make and receive phone calls. So it's about time we get around to configuring some phone numbers for the users, right?

This chapter will dig into how to assign phone numbers to your users and – perhaps more importantly – the correct format to use for those numbers. There will also be a brief discussion over how to configure physical phones. I won't explain how to provision the phones. Work with your reseller or with your vendor for that. Rather I will only deal with the Skype for Business-specific settings needed for common area phones.

Enabling users for Enterprise Voice

Once you've gotten everything set up, enabling users for Enterprise Voice is pretty easy. Other than having dial plans and voice policies already created, the only other pre-requisite is knowing which phone number you want to assign to a user.

You should *always* assign the phone number using E.164 format. What is E.164? It's a standard for telephone numbers. It's the standard that the world has agreed upon to assign phone numbers to your mobile phone or to your hair dresser. As such, do the smart thing and follow the global standard when assigning numbers. You won't be doing yourself any favors by coming up with your own standard or "taking the easy way". In the long run, the "easy way" will end up being the much harder way.

The E.164 standard has the following characteristics:

1. A telephone number can have a maximum of 15 digits
2. The first one to three digits are the country code
3. The second part of the number is the optional "national destination code"
4. The last part of the telephone number is the "subscriber number"

So let's look at these components starting with the second one in the list.

The country code can be between 1 and 3 digits long. In North America, the country code is simply the number "1". There is only 1 other single digit country code and that is "7" which is shared by Russia and Kazakhstan. There are numerous countries with 2 digit codes such as Germany (49), Japan (81), and South Africa (27). And there are also plenty of countries with 3 digit country codes, such as Bolivia (591) Ireland (353), and Tajikistan (992). Wikipedia has the full list that can be found at this URL:
https://en.wikipedia.org/wiki/List_of_country_calling_codes

The second part of the number – the national destination code – is called the Area Code in North America. In other countries it is often called the City Code. While the first part of a telephone number defines the whole country, the second part of the number defines a region within that country. It could also be used to define the class of the call, such as mobile numbers, toll free numbers, etc.

And the last part of the number is the users' specific, unique phone number. This is the part that makes your phone number different than your neighbors.

So if you connect all three sections together and slap a "+" in front of it, you have a properly formed E.164 number.

Wait. Where did that plus come from? The plus is there to designate that the number includes the international country calling code. For our purposes, always add the + in front of your numbers even if no telephone company you deal with ever seems to want to see the + in the dial string. The telco's inability to natively handle properly formatted E.164 numbers is not our problem.

Add the + even if you never expect to have Skype for Business users outside your country.

There's one more thing to discuss with E.164 numbers, and that is *extensions*. What happens if you only have one phone number for your office but want each of your users to have a unique extension?

E.164 covers that too. While everyone will have the same base phone number, you can give everyone a unique extension by adding ";ext=" to the end of the number and then adding their extension. For example, if the E.164 number for our office is +12223334444, we just add this to give a user a unique extension: +12223334444;ext=5555 (assuming we are using 4 digit extensions. You can have any extension length up to 15 digits).

Now, as a rule, *every number* should have an extension, *even if that user doesn't need one.* Say what?

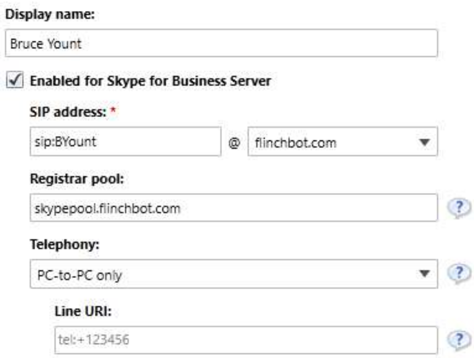
If you have a direct inward dial number (DID/DDI) there seems to be no practical reason to add an extension as that user already has a unique number. What sense does it make to tell Skype for Business that the number is +12223334444;ext=4444 instead of just +12223334444?

There are a few occasions where your users could use their extension instead of using their full number. So this provides a "short cut" for your users. For example, another internal user can now call you by just dialing "4444" instead of your full phone number. Another example is if you dial in to a Skype for Business conference call as a leader of the call. You can now just punch in your extension instead of the full number. This makes for a quicker meeting join and makes it less likely that you will accidentally dial a wrong digit while punching in your full telephone number.

For users who have a DID number and an extension defined, you don't have to add specific normalizations in your Dial Plans for these users. Skype for Business will first do a Reverse Number Lookup by ignoring the ;ext= portion of the users phone number. If there is a match, the call gets routed to the user.

Now that you have the telephone number you want to assign, go to the Skype for Business Control Panel and click on the Users section. Find the user you are looking for and open up the properties for that user.

Figure 10 – 1



Until you enable Enterprise Voice for a user, all of your users will show "PC-to-PC only" under Telephony and their Line URI will be blank.

The first thing to do is to change the Telephony setting from "PC-to-PC Only" to "Enterprise Voice". After you make this change, you'll notice that a few extra fields appear. One is for setting a Dial Plan and the other is for setting a Voice Policy.

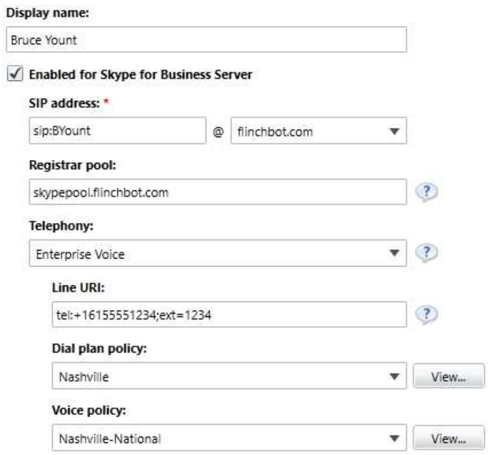
First, enter the phone number for the user into the LineURI field. Enter the full E.164 number (with the + sign!). But just to make things tricky, Skype for Business insists that you prepend "tel:" before the E.164 number. So what you will really enter into the box is something like this:

tel:+12223334444;ext=4444

As for the Dial Plan and Voice Policy fields, you can leave those blank if you have a site-level or a (*not recommended*) global-level Dial Plan and Voice

Policy defined. If you have user Dial Plans defined, pick them off the list. In the image below, I've selected the dial plan "Nashville" and the Voice Policy "Nashville-National".

Figure 10 – 2



After you have made this change, click the "Commit" button. Your user is now Enterprise Voice enabled.

If you want to do this in PowerShell, use the following cmdlet to enable the user for Enterprise Voice:

```
Set-CsUser -Identity byount -EnterpriseVoiceEnabled
$True -LineURI 'tel:+12428081150;ext=1150'
```

Next you need to grant the Dial Plan and Voice Policy to the user.

```
Grant-CsDialPlan -Identity byount -PolicyName 'Nashville'
```

```
Grant-CsVoicePolicy -Identity byount -PolicyName 'Nashville-
National'
```

To verify that the change took place, run `Get-CsUser`. The EnterpriseVoice, LineURI, Dial Plan, and Voice Policy fields should now have values. If not, give it a few minutes for Active Directory replication to happen and then check again.

```
Get-CsUser -Identity byount
```

Figure 10 – 3

```
Identity                          : CN=Bruce Yount,OU=User
                                    Accounts,DC=flinchbot,DC=com
VoicePolicy                       : Nashville-National
VoiceRoutingPolicy                :
ConferencingPolicy                :
PresencePolicy                    :
DialPlan                          : Nashville
LocationPolicy                    :
ClientPolicy                      :
ClientVersionPolicy               :
ArchivingPolicy                   :
ExchangeArchivingPolicy           : Uninitialized
PinPolicy                         :
ExternalAccessPolicy              :
MobilityPolicy                    :
PersistentChatPolicy              :
UserServicesPolicy                :
CallViaWorkPolicy                 :
ThirdPartyVideoSystemPolicy       :
HostedVoiceMail                   :
HostedVoicemailPolicy             :
HostingProvider                   : SRV:
RegistrarPool                     : skypepool.flinchbot.com
Enabled                           : True
SipAddress                        : sip:BYount@flinchbot.com
LineURI                           : tel:+12428081150;ext=1150
EnterpriseVoiceEnabled            : True
```

Private Lines

One of the limitations to Lync is that you can't assign two phone numbers to a user. Well…that's not entirely correct. You can actually add a second number to a user using the Private Line feature.

A Private Line gives a user a second inbound number. In general, this is meant for an executive who has a public phone number shared with the world – the LineURI – and a private phone number only shared with a spouse or a few close associates – the Private Line.

There are a few distinct attributes to a Private Line.

- Like a LineURI, only one private line per user
- A Private Line uses the same voice mail that the LineURI does
- A user with a Private Line does not have a second SIP address or separate presence status
- The Private Line does not appear in any Skype for Business address books.
- Private Lines do not support the following:
    - Call Forwarding
    - Team Call
    - Delegation
    - Team Ring

  - Group Call Pickup
  - Response Groups
- A user can redirect an incoming Private Line call to a configured forwarding device, such as their mobile phone.
- Call Detail Records are the same as records calling the users LineURI but there is an indicator in the CDR that this call was sent to a Private Line.
- The user must already be configured for Enterprise Voice.

A call to a Private Line number will have a different ringtone than the call to the users LineURI to help the user differentiate which number is being called. Also any outbound call will always show the LineURI as the From: number.

So, now that you understand what a Private Line is and how it differs from a user's LineURI, how do you actually assign one to a user? Well you can't use Control Panel for this one. Configuring a Private Line is only available via PowerShell.

Let's say we need to give our user BYount a Private Line. The following cmdlet sets the Private Line for BYount to +13175551212:

```
Set-CsUser –Identity BYount –PrivateLine
'tel:+13175551212;ext=1212'
```

From personal experience, there has been a time when I assigned every user in a location a Private Line. We were migrating from a PBX to Lync and also migrating from a PRI to a SIP trunk. Everyone was going to get a new telephone number that was different than what they had on the PBX. This was also going to be a "hard cutover" meaning that one evening we were going to enable everyone for Enterprise Voice and turn off the PBX.

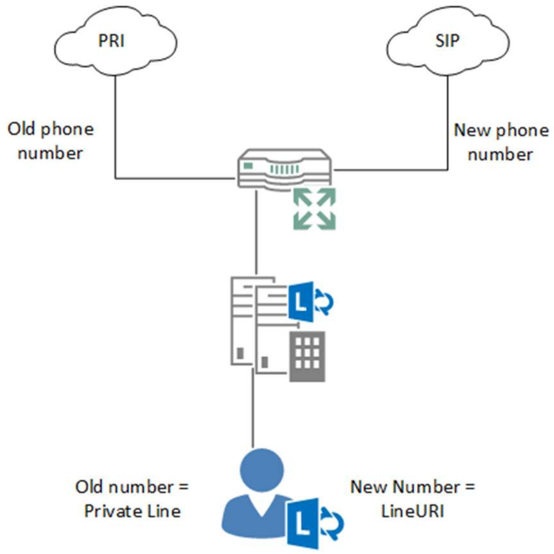
In order to ease the migration to the new numbers on the SIP trunk, we took the old PRI and plugged it into the same gateway terminating the new SIP trunk. We then assigned everyone their old PBX phone number as a Private Line number.

The next morning, if a call came in to their old PBX phone number, the call would go to the gateway via the PRI. The gateway would forward the call to their Lync server and, because of the Private Line, their old number would ring on their Lync desktop.

Similarly, a call to their new phone number on the SIP trunk would route to the gateway. The gateway would forward it to Lync and Lync would ring the users' client. This worked fabulously.

And as the users started making outbound calls their new number would be used for the Caller ID. (Remember, Private Line numbers are never used as the From: number). Using this "trick" we had a much smoother migration and made sure important calls to the old numbers still got delivered to the user.

Figure 10 – 4



Viewing Private Line Numbers

If you want to see if a user is assigned a Private Line number, the obvious thing to do would be to type `Get-CsUser` into PowerShell. And you would be mostly correct. However, just typing `Get-CsUser` will not show the Private Line. Instead, you need to view the "hidden" settings for your users. To do this, you can run either of the following commands:

```
Get-CsUser byount | format-list *
```

The following command will only show the *–privateline* attribute so you don't have to search through the list looking for it.

```
Get-CsUser byount | format-table –property privateline
```

Removing Private Line Numbers

If you want to get rid of a user's Private Line, use the `Set-CsUser` cmdlet and just set the *–privateline* parameter to $Null.

```
Set-CsUser byount –privateline $Null
```

Common Area Phones

A Common Area Phone (CAP) is a physical phone that does not belong to an actual user. An example of this would be a phone in a lobby or in a meeting room. None of your actual user's log in to that phone. However, due to the nature of Skype for Business, someone (or something) needs to be logged in to the phone. So how does a phone log in? It logs in using a Common Area Phone account which is just an Active Directory contact object.

To create a CAP account, you use the `New-CsCommonAreaPhone` cmdlet. You cannot create CAP accounts via the Skype for Business Control Panel.

For this example, let's say we want to create a common area phone for the lobby in our office. There are a few things we will need to know:

- The Telephone number (Line URI) we want to assign to the phone *(12428081199)*
- The RegistrarPool against which the phone will log in *(skypepool.flinchbot.com)*
- The Display Name for the phone *(Lobby Phone)*
- The Active Directory OU where the Contact Object will be created *(Users OU)*
- If we aren't using Site or Global-level Dial Plans or Voice Policies, we'll need to know that too *(Nashville* and *Nashville-National)*

With this information, we can open the Skype for Business Management Shell and type in the following cmdlet:

```
New-CsCommonAreaPhone -LineUri
'tel:+12428081199;ext=1199' -Registrarpool
'skypepool.flinchbot.com' -DisplayName "Lobby Phone" -OU
'CN=Users,DC=flinchbot,DC=com'
```

Just like with a user, we need to grant the Dial Plan and the Voice Policy to the CAP:

```
Grant-CsDialPlan -Identity "Lobby Phone" -PolicyName
'Nashville'
```

```
Grant-CsVoicePolicy -Identity 'Lobby Phone' -PolicyName
'Nashville-National'
```

Finally, we can use the `Get-CsCommonAreaPhone` cmdlet to verify that the contact object has been created and the Dial Plan and Voice Policy have been successfully assigned:

Figure 10 – 5

```
PS C:\Users\flinchbot> Get-CsCommonAreaPhone -Identity 'CN=Lobby Phone,CN=Users,
DC=flinchbot,DC=com'

Identity               : CN=Lobby Phone,CN=Users,DC=flinchbot,DC=com
RegistrarPool          : skypepool.flinchbot.com
Enabled                : True
SipAddress             : sip:2e236ac3-2044-440a-a3c0-51dbedca472f@flinchbot.com
ClientPolicy           :
PinPolicy              :
VoicePolicy            : Nashville-National
VoiceRoutingPolicy     :
MobilityPolicy         :
PersistentChatPolicy   :
ConferencingPolicy     :
LineURI                : tel:+12428081199;ext=1199
DisplayNumber          :
DisplayName            : Lobby Phone
Description            :
ExUmEnabled            : False
```

There is one last thing we need to do before we can sign in to a phone using this CAP account. We need to assign a PIN to the CAP contact. In order to log in a phone, you need the phone number (LineURI) and a PIN.

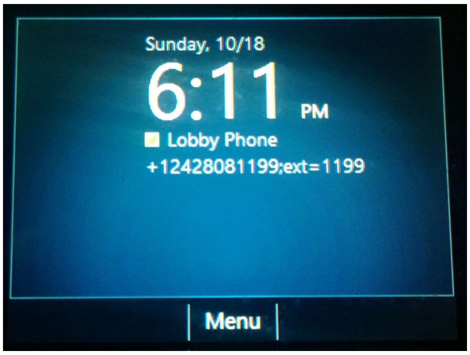
To assign a PIN to your phone, use the `Set-CsClientPin` cmdlet. You cannot do this via the Control Panel.

```
Set-CsClientPin -Identity 'Lobby Phone' -Pin '126578'
```

Now that this is set, you should be able to log in to your phone using this CAP account. If not, the problem is most likely with how your phone is provisioned and you should work with the phone vendor (or your reseller) to troubleshoot the issue further.

As you can see in the below figure this all works! I logged into the phone using only the extension 1199 and then punched in the PIN I set (126578). The phone thought for a few seconds and then logged me in. I love it when things work on the first try.

Figure 10 – 6



Hot Desking

In general, you won't want anyone to be able to log into the Common Area Phone other than the phone itself via the associated contact object.

However, if you put the phone into a meeting room, you may want to permit users to log in to the phone so that they can easily join a meeting via the calendar on the phone. In this case, you would want to enable Hot Desking.

Hot Desking is the concept that permits (or denies) the ability of a user to log in to a common area phone.

To enable or disable Hot Desking, you need to change a value on the client policy assigned to both the CAP contact object and to the user who wishes to log in to the phone. You also can set how long a user is permitted to be logged into the phone. If you set this to one hour, the user will be automatically logged off the phone one hour after they logged in.

To enable Hot Desking for a client policy named "Nashville Client Policy" and to set the Hot Desking timeout to 1 hour, run the following cmdlet:

```
Set-CsClientPolicy –Identity 'Nashville Client
Policy' -EnableHotDesking $True –HotDeskTimeOut "01:00:00"
```

You cannot create or edit client policies using the Control Panel.

Summary

Enabling users is pretty straightforward. In general, you just need to flip a switch and assign an E.164 phone number to the user. If you need to assign specific Dial Plans and Voice Policies, those can easily be granted to your users. Due to the flexibility of PowerShell, you can write scripts to enable users in bulk by importing phone numbers from a .csv file. There are several examples of this that you should be able to easily find on the Internet.

Common Area Phones appear in almost every Enterprise Voice rollout. Knowing how to configure them is a useful thing to know. It is up to you to decide if you wish to enable or disable Hot Desking.

Chapter 11 – Survivability

Once you have all of your Dial Plans and PSTN Usages and Routes configured and working…what happens if something goes wrong? What happens if a Mediation Server crashes? Or a construction crew cuts the cable connecting you to the telephone company?

In the Skype for Business world, surviving these kinds of outages is called 'survivability". This chapter will cover the many ways that you can architect your Skype for Business environment. In general, it involves adding additional servers but which servers to add – and where to add them – is key to understanding Skype for Business survivability.

As this is a book on Enterprise Voice, this chapter will focus only on voice survivability and not cover other things such as redundant Edge servers or SQL Servers.

Standard Edition

Skype for Business Standard Edition is an all-in-one implementation. This is perfect for small or mid-sized business that wants to have a simple, straightforward configuration. All of the Skype for Business roles run on a single server, keeping costs low without giving up any features.

However, if a single server fails – or a single service on that server fails – you will experience some level of outage. So how are you to keep your voice services running when a Standard Edition server fails?

Install a second server!

Though this seems obvious, there is a bit more to it than just installing a second server.

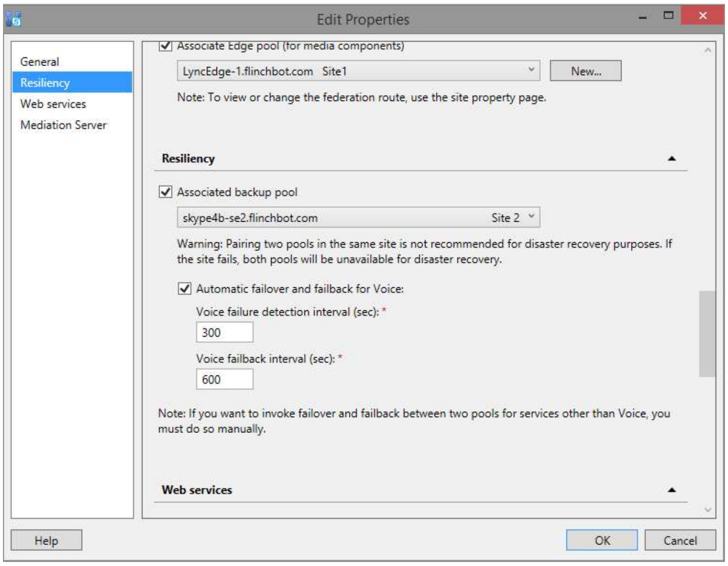
First, you need to add a second standard edition server to your Topology. It does not need to be in the same Skype for Business Site but it does need to be configured for the same services as your first Standard Edition server. This is because Skype for Business will fail over all of the roles running on one server to the other. If the one server doesn't support the same roles as the other, then the failover will not be successful. To prevent this scenario, Topology Builder does a check to make sure that both servers are consistent with the roles provided.

In Figure 11-1, you can see one Standard Edition server backing up another. Note that ticking only the "Associated Backup Pool" value will not provide voice resiliency.

Without getting into the depths of Skype for Business failover, the first option ("Associated backup pool") is used to configure LYSS (The Lync Storage Service) to configure a backup routine between two pools.

The "Automatic failover and failback" for voice is a bit misnamed. What it actually does is enable the backup pool to become a backup registrar. So if one Standard Edition server fails, then your users will automatically log in to the backup Standard edition server. This doesn't inherently mean that your voice will work. However, if your users can't get logged in against a Skype for Business server, then there is no chance at all of Enterprise Voice working.

Figure 11 – 1



There are two values to set for automatic voice failover – "Voice Failure Detection" and "Voice Failback interval".

These values control how soon one of your Standard Edition servers will become the registrar for its partner pool and how soon it will revert control once the failed server is returned to service. By default, these values are 5 minutes (300 seconds) to failover and 10 minutes (600 seconds) to fail back. Feel free to lower these numbers. Do some testing to make sure there aren't any "false positive" failovers so your users aren't unnecessarily moved.

Once you have your two Standard Edition servers set as backups for one another, you are in pretty good shape – at least for your users to be able to log in during a server failure and for *inbound* calls to arrive.

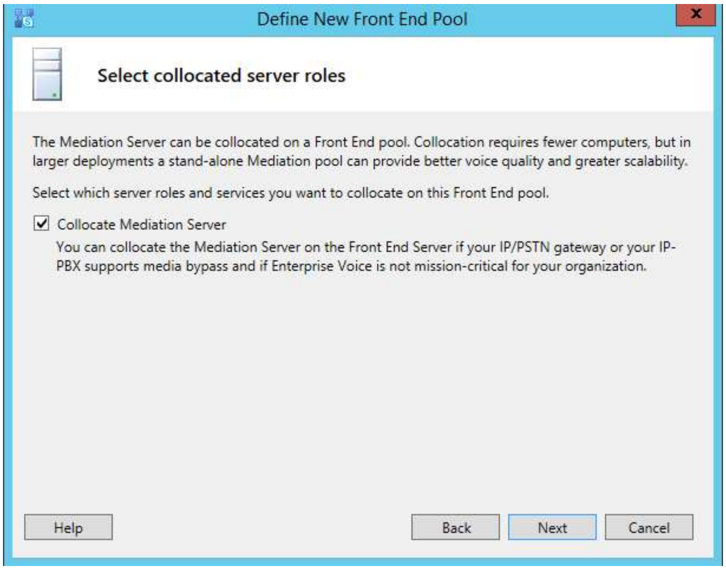
Inbound routing is usually defined by your gateway. Without getting in to the complexities of gateway routing, be sure to add your second Standard Edition server as a fallback host should the gateway be unable to contact the primary Standard Edition server.

Mediation Pools

When adding your servers in Topology Builder, there will be one question asking you if you wish to collocate the Mediation role or not. If you are doing a simple build of a Standard edition server, then keep the default settings.

The default setting tells Skype for Business to run the Mediation services on the same server as everything else. This limits the amount of servers you need in your environment in exchange for higher server utilization. Remember that – barring Media Bypass scenarios and certain SIP trunk providers– the Mediation server will real-time transcode every PSTN phone call to/from the G.711 codec. This can potentially create a heavy burden on the server.

Figure 11 – 2



There are a few things to note in the wording of the message above.

The verbiage following the checkbox is misleading. You can collocate the Mediation service on the Front End server if even if you do *not* support Media Bypass and if Enterprise Voice is mission critical. Collocated mediation will work just fine without Media Bypass so long as you scale your servers correctly to handle the additional processing load. Yes, Media Bypass will reduce the processing required by your servers but collocation will work 100% successfully with or without Media Bypass.

If you have paired Standard Edition pools you have redundancy for your voice, addressing the concern in the message about not collocating in “mission critical” scenarios. And in an Enterprise Edition pool you can have up to 12 Front End servers in one pool. If you collocate Mediation and one of your Front End server dies then.....you still have 11 servers providing Mediation.

Tick this box if you are looking for a simple Standard Edition deployment (on properly sized servers) or if you want a relatively simple Enterprise Edition deployment. Do not uncheck it just because of that poorly worded description.

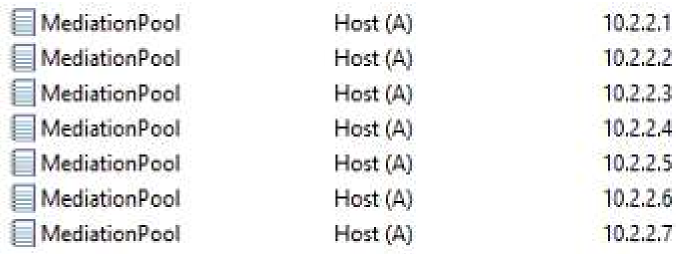
Now, if you want to go truly “enterprise class” with your mediation services, then by all means create a separate pool and throw in a healthy heaping of servers. That will offload the Front End servers from transcoding and can provide better performance and resilience to your Enterprise Voice implementation.

The Mediation Service is responsible for the following activities:

- Encrypting and decrypting SRTP
- Translating SIP over TCP from gateways to encrypted SIP over MTLS
- Transcoding media between Skype for Business and gateways
- Connecting clients that are outside the network to internal ICE components, which enable media traversal of NAT and firewalls.
- Acting as an intermediary for call flows that a gateway does not support, such as calls from remote workers on an Enterprise Voice client.
- In deployments that include SIP trunking, connecting directly with a Skype for Business-certified SIP trunk to provide PSTN support without the need for a PSTN gateway.

Mediation Pools are load balanced via DNS. So you will need to decide on a DNS name for your mediation pool. Then in Topology builder you list out the servers with their actual server names that will be members of this Mediation pool. Figure 11-3 shows a snippet of the DNS entries required to support a Mediation pool.

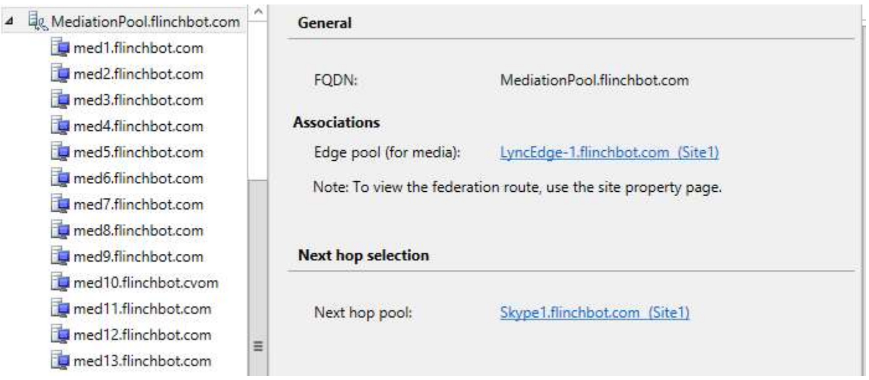
Figure 11 – 3



Each of the IP's represents the physical IP address of one of the Mediation Servers in the pool. Using DNS round robin (built in to DNS) a different server will be selected for each call that is destined to/from the PSTN - assuming your gateways support DNS round-robin which they should.

Below is what Topology builder looks like. Each of the servers listed on the left should correspond to one of the IP addresses listed in DNS.

Figure 11 – 4



For heavy voice implementations, Microsoft recommends enabling receive-side scaling (RSS) on your network adapters. Enabling RSS will improve the media performance by handling incoming packets in parallel by multiple processors on the server. For information on how to enable this feature, check your network adapter documentation.

Enterprise Edition Survivability

I can be really lazy here. Did you just read the last 2 sections on "Standard Edition" and "Mediation Pools"? Because that pretty much covers it! Anywhere you see "Standard Edition" just replace it with "Enterprise Edition".

The one main difference is that collocating Mediation on your Enterprise Edition Front End servers inherently provides resiliency above just failing to the paired pool should the Front End service on a Standard Edition server fail. If you go with an Enterprise Edition pool, you have at least 3 servers in your pool. (*You do actually have at least 3 servers right? Because if not, you're doing it wrong*). You can lose one server and still have both the pool running and mediation running. Should you lose a second server in your Enterprise Edition pool, you can then fail to your backup pool.

If you have a separate Mediation Pool, then you have even more redundancy.

Topology	Resiliency
Collocated Mediation on Standard Edition	Standard Edition server fails, failover to backup Standard Edition server*
Collocated Mediation on Enterprise Pool	Lose at least 1 Front End without losing Enterprise Voice
Separate Mediation Pool with Standard Edition	Lose at least one mediation and still have enterprise voice. If the Standard Edition fails, failover to backup Standard Edition server (Continue to use original Mediation pool)
Separate Mediation Pool with Enterprise Edition	Lose at least one Mediation server and still have Enterprise Voice. Lose at least 1 Front End server and still have Enterprise Voice. If Front End pool fails, failover to paired pool (continue to use original Mediation pool)

* You can pair an Enterprise Edition pool with a Standard Edition server. So you could have all of your users on an Enterprise Pool and then only use the Standard Edition server if you need to failover.

Outbound Failover Routing

Chapter 6, which covered PSTN Usage, dove deeply into failover routing. However, it is important to reiterate some of the points from that chapter here.

Just creating a second Mediation pool – either with Standard or Enterprise Edition – is not enough to handle outbound calling in the event of a Mediation failure.

You must be sure to create new routes and PSTN Usages in order to properly handle a Mediation outage.

If both Mediation pools are in the same physical location, then you can get away with creating a trunk from each Mediation pool to your gateway and then add both Trunks to the same Route. Technically this is a round-robin scenario where each outbound call will go through a different mediation server each time. If one mediation pool should fail, then all of the calls will be routed through the surviving mediation pool. This will continue until Skype for Business detects that the other Mediation pool is back and stable at which point round-robin calling will resume.

However, if you have your Mediation pools in separate locations (or even separate continents) then this isn't the preferred method as you are now (at best) passing calls across a WAN and (at worst) racking up unnecessary toll charges.

Rather, the better approach is to create one Route for each location using the local Trunk. Then create a PSTN Usage for each Route. Finally, assign the PSTN Usages to relevant Voice Policies, ordered in such a way that the "local" route is used first. If that Route should fail, Skype for Business will then go to the next PSTN Usage/Route which will then send calls to the remote location.

While reviewing the PSTN Usage and Route chapters will give you full details on this configuration, it won't hurt to throw in a quick example.

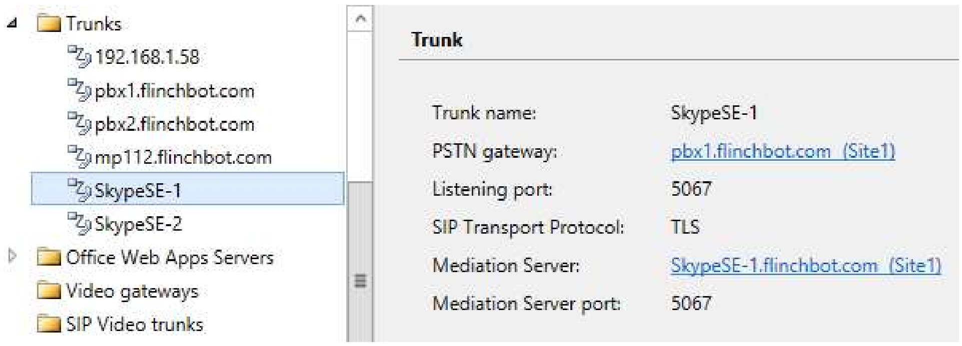
Let's say you have an office in Amsterdam and an Office in Munich. Each office has one Standard Edition server with collocated Mediation. You want to configure Skype for Business so that all calls from users in Amsterdam go out through the local gateway in Amsterdam. All calls from users in Munich should go out through the gateway in Munich. However, if the Mediation service fails* at one location, calls should go out through the other location.

* I should point out here that a Mediation service failure could be due to a crashed gateway. In this scenario, Skype for Business will mark the gateway as down. With the gateway down, there is no route for a call. If there is a second PSTN Usage configured, the call will then be sent via the route attached to that PSTN Usage.

The name of the server in Amsterdam is "SkypeSE-1" and the name of the server in Munich is "SkypeSE-2". After adding the servers to Topology, you need to create a Trunk for each location.

Below is a screenshot showing how this might look in Topology Builder. In this scenario, Skype for Business is configured to reach the PSTN via a local PBX. So the Amsterdam Trunk connects SkypeSE-1 to PBX1 and the Munich trunk connects SkypeSE-2 with PBX2.

Figure 11 – 5



Once this Topology has been published, you can create your Voice Policies, PSTN Usages, and Routes. The table below shows how this would look.

You would create two Voice Policies – one to be assigned to your Amsterdam users and one to be assigned to the Munich users. Within these Voice Policies you then need to create 2 PSTN Usages, one for each city. Within these Usages, create a Route for each city. Within each Route, assign the appropriate Trunk you created in Topology Builder.

Voice Policy	PSTN Usages	Routes	Trunks
Amsterdam	Amsterdam Usage	Amsterdam Route	SkypeSE-1
	Munich Usage	Munich Route	SkypeSE-2
Munich	Munich Usage	Munich Route	SkypeSE-2
	Amsterdam Usage	Amsterdam Route	SkypeSE-1

Once everything has been created, you assign the PSTN Usages in the correct order with the local PSTN Usage listed first and the remote PSTN Usage listed second. As a reminder, PSTN Usages are evaluated in a top down manner. So long as the first PSTN Usage results in a successful route then that will always be chosen.

If the Route is unsuccessful, such as when a Mediation service fails, the second PSTN Usage in the list gets evaluated. Assuming everything is OK, calls will now fail over to the backup city.

Inbound Failover Routing

Inbound failover routing relies on more than simply what Skype for Business can do. Somewhere along the line, either you or your telephony provider have an SBC or a gateway configured. Within that gateway configuration, you can list more than one Skype for Business Mediation Server as the "next hop" for the gateway. If one Mediation Server is down, then the gateway can forward inbound calls to the next Mediation Server on the list.

And that's about it as far as Skype for Business is concerned with inbound failover routing. The real onus of survivability on inbound calls lies with both the gateway and your telephony provider.

What happens if your gateway fails? Well, some models of gateways provide redundant components within a single chassis, such as redundant power supplies and redundant PRI modules. Some gateways can even go so far as keeping two separate gateways synchronized together so if one entire gateway fails then the second gateway immediately handles the calls.

But in order to support that, you will have to work with your telephony provider to be able to continue to provide inbound (and outbound) calling without dropping calls when a gateway fails.

In general, SIP trunks provide much more resilience for both inbound and outbound calling than a PRI or other legacy technology can provide. Since a SIP trunk is just an IP connection, it can easily be re-routed to a different gateway (or an entirely different location) quickly.

In short, if you have strict uptime requirements for inbound routing (and to a lesser extent, outbound routing) then you need to talk with your telephony provider to see what options they provide. Then match those options up with your gateway vendor.

Branch Site Survivability

In many corporations, there is a main office (or a few main offices) and several secondary – or branch – offices. These offices don't require the high availability necessary at the main offices. However, you do want to assure that should the WAN connection to the main data center fail that these users can still make and receive phone calls.

For this scenario, Microsoft has created the concept of the Survivable Branch Appliance (SBA) and the Survivable Branch Server (SBS).

From a technical standpoint, there is no difference between an SBA and an SBS. The only difference is if the Skype for Business services will be running on a gateway then it is an SBA. If you will be providing your own hardware (or virtual machine) to run this role, then it is an SBS.

To go a touch deeper, the operating system and the Skype for Business installation on an SBA is supported by the gateway manufacturer. On an SBS, Microsoft provides the support for Windows and the Skype for Business components.

I know that make little sense, but it's a licensing distinction. On an SBA you buy the Windows license from the gateway provider so they are on the hook for providing support. With an SBS you are providing the license - probably through a volume license agreement with Microsoft. In that case, you call Microsoft directly for support.

Note that with an SBS you will probably still need to purchase a gateway or an SBC to connect the SBS with your telephony provider. With an SBA, the gateway piece is included with the server in one rack-mountable unit.

One other difference is that if your branch office has more than 1,000 users, then the recommendation is to go with an SBS. The size of the server you can embed into a gateway is limited.

An SBA/SBS only has a minimal subset of a full Skype for Business installation:

- A Registrar for user authentication, registration, and call routing
- A Mediation Server for handling signaling between the Registrar and a PSTN gateway
- SQL Server Express for local user data storage
- A PSTN gateway for routing calls to the PSTN (Required for an SBA, optional but recommended for an SBS)

There are a few things of which you should be aware. Currently, you can buy an SBA with as little as 2GB RAM and as much as 8GB. Do not buy the 2GB model. I repeat: *Do not buy a 2GB SBA*. First, if you're any kind of an IT veteran, the thought of running Windows, SQL Express, and Skype for Business with only 2GB of RAM should make you shake your head and ask "Why did Microsoft even certify this configuration"?

I suppose to keep costs down is the reason. So let's say you have a super small branch office of like 8 users. Won't the 2GB SBA work? Technically, yes.

But keep the following things in mind:

With only 2GB of RAM, how much more RAM will be used by monitoring agents such as System Center Operations Manager, System Center Configuration Manager, or an antivirus product? Is 4GB RAM going to be that much better? (Answer: Yes!)

Inevitably, you will need to do some troubleshooting. Will the performance on the SBA be good enough to run logging tools and review those logs in Snooper? Probably not. You'll need to copy those logs to your PC to view them in Snooper.

The real killer to a 2GB (or even 4GB) SBA is the SQL Express instance. The SQL database on an SBA does not only contain records for the 8 users in your branch office – it contains a record for every Skype for Business user in your organization. If you only have 500 users, you can probably survive. But if you have tens of thousands or hundreds of thousands of users, a 2GB SBA will not survive. Get used to restarting the Front End service as it crashes due to lack of resources.

My advice: The bare minimum you should buy is an SBA with 4GB of RAM. But for best results (especially in very large implementations) buy nothing less than the 8GB models.

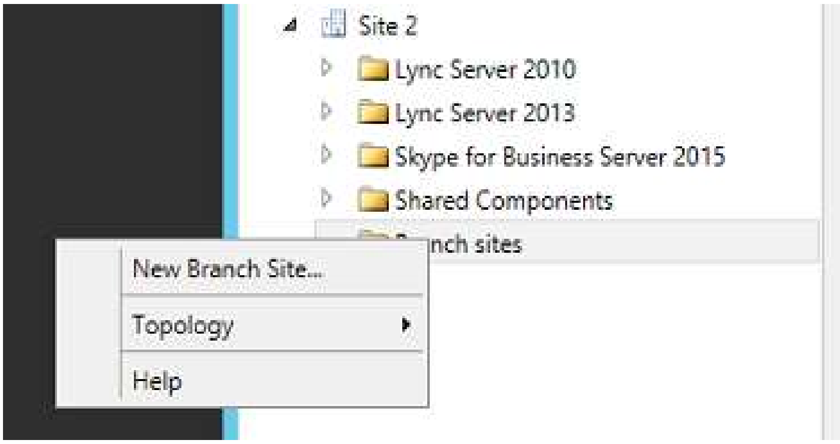
Adding a Branch Site to Topology

When you add an SBA/SBS to Skype for Business Topology, it becomes a sub-site to an existing Site. Further, it is tied in to a specific pool in that existing site (which I like to call the "parent pool").

You will also need to define the gateway that the branch site will use to connect to the PSTN.

To add a new Branch Site to Topology, open Topology Builder and navigate to a Site. From there, navigate to the "Branch sites" container. With "Branch sites" highlighted, select "New Branch Site" from the Action menu. Alternately, right-click on the "Branch sites" container and select "New Branch Site…" from the pop-up menu.

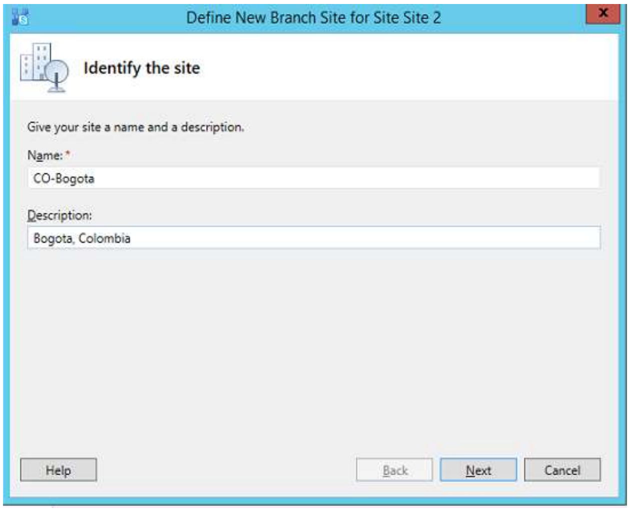
Figure 11 – 6



After selecting "New Branch Site…" the "Define New Branch Site" window opens. In the "Name" window you provide a name for the Site. You can enter anything in here but it's best not to use spaces or any crazy characters. You may have to reference this Site name on occasion via PowerShell so make life easy on yourself and keep the name simple. For example, you could go with "Bogota" or "CO-Bogota" as the name. In our example, we will go with the name "CO-Bogota" where the CO stands for the country Colombia and Bogota is the city where the Branch Site is located.

The "Description" field is where you can go crazy and use whatever characters you want. This is just a field allowing you to note why this site exists. In this case, I'm just going to enter "Bogota, Colombia".

Figure 11 – 7



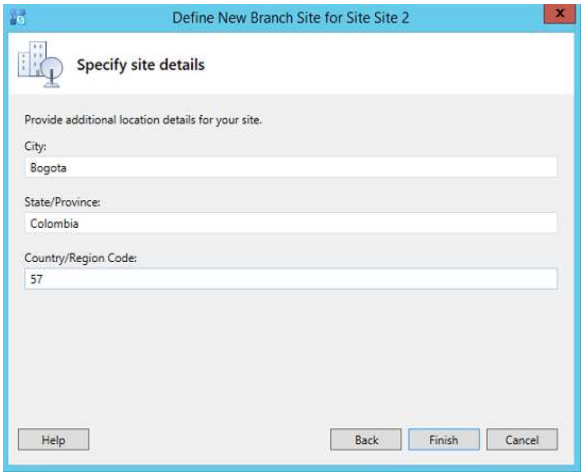
After filling out the fields, click "Next" where you can optionally provide geographic information for this site. You don't need to fill out any of these fields but being a good Systems Administrator means good documentation.

So do yourself and any future administrators a favor and fill out these fields as best possible, especially if your Branch Site name is not very descriptive. While to you it may make perfect sense to name a Site "A17" that may be completely nonsensical to someone else. Filling out these fields (along with a good description in the previous screen) is only the right thing to do.

In the figure below, the first two fields should be self-explanatory (though technically Colombia is neither a State nor a Province, at least not as a state is defined in the United States or how a province is defined in Canada).

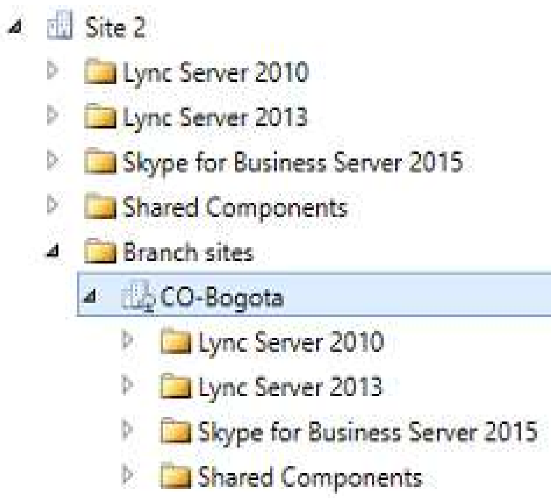
In the third field I entered the ISO dialing code for Colombia. You can enter anything here but adding the country (or region code) is a nice bit of documentation for this site.

Figure 11 – 8



After filling out these fields, click the Finish button to complete the creation of a new Branch Site. You will now see that there is a new branch underneath your original "Parent Site". You will also see several containers that will be used to store the SBA/SBS as well as the gateway(s) you will use in this site.

Figure 11 – 9



Now we need to create the SBA/SBS object within Topology. Note that there is no difference in creating an SBS from an SBA. They are both created the exact same way. The only difference is how you install the software onto your server. If it is an SBA, then there will be a wizard of some sort pre-installed onto the server portion of the gateway. Follow your gateway vendor's instructions on how to install the SBA.

For an SBS, you have to install your own copy of Windows. Then run Setup off the Skype for Business distribution media and install Skype for Business in the exact same way that you would install a new Front End server.

In order to add an SBA/SBS to Topology, open Topology builder and navigate to the Branch Site you have created. Navigate to the "Skype for Business Server 2015" container (or the Lync Server 2013 container if you are still using Lync). With this container highlighted, select "New Survivable Branch Appliance…" from the Action menu. Alternately, right click on the "Skype for Business Server 2015" container and select "New Survivable Branch Appliance…" from the pop-up menu.

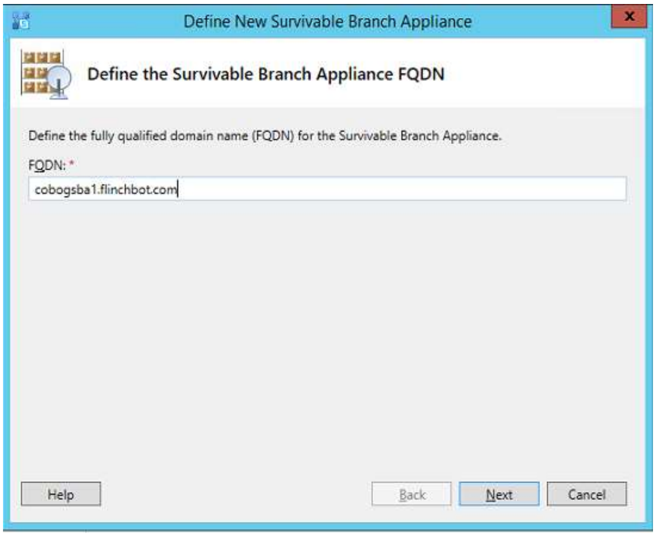
Figure 11 – 10

This open the "Define new Survivable Branch Appliance" window. Again – this is also used to define a new Survivable Branch Server. Microsoft is just using the name "Survivable Branch Appliance" here because putting both names in the title bar would be way too long.

The first thing to add here is the Fully Qualified Domain Name (FQDN) for the server. Try to pick a descriptive name for the server but I understand that there are IT departments out there that prefer to give cryptic names to all of their servers. So use something insane like:

"A143E6DP.317.swnesenw.corpnet17.org" for the server name if you must, but I'm going with "cobogsba1.flinchbot.com"

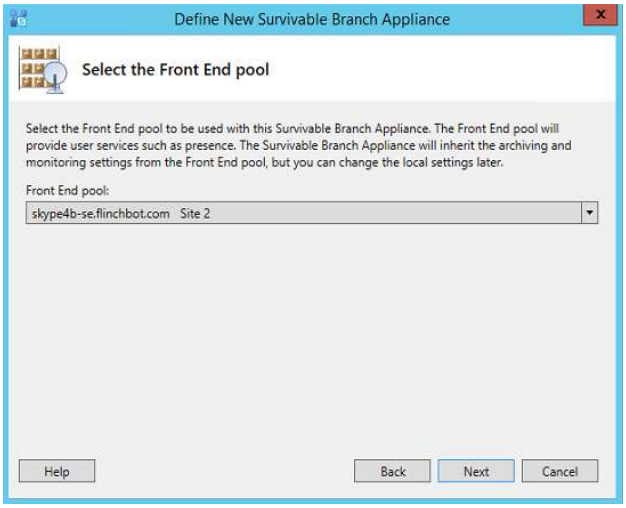
Figure 11 – 11



After you click Next, you will be asked to select the Front End pool ("Parent Pool" in my little world) to which this SBA/SBS will be connected. You can only select a pool that is in the parent Site to your new Branch Site. In this example, I will be using a Standard Edition server named "skype4b-se.flinchbot.com" as the Front End pool for this branch server.

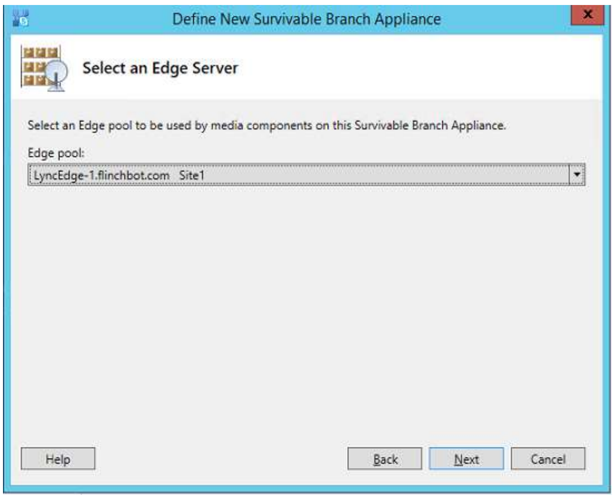
After selecting the pool, click "Next" to select an optional Edge pool for this branch server. If you do not select an option here, your users will not be able to take advantage of any Edge-related features such as using the mobile client. You can pick any Edge server here, not just ones from the parent site.

Figure 11 – 12



In the example below, I've select an Edge server from a different site ("Site 1") than where the Parent Site for my branch site sits (which is in "Site 2").

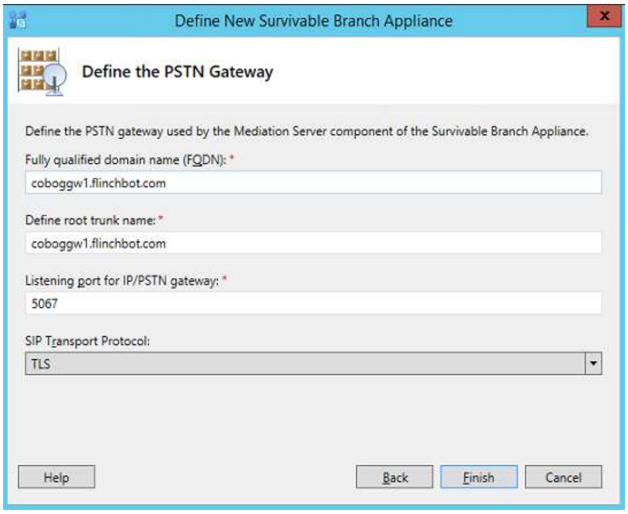
Figure 11 – 13



After clicking Next, you are then taken to a screen to define the PSTN gateway. I won't go into the details here as they are covered in the "Gateways and Trunks" chapter. Just be sure to have a gateway name and trunk name decided on before filling this out.

You will also need to know which port and protocol (TLS or TCP) have been configured on the gateway. In this screen, you are entering the values as defined by your gateway configuration.

Figure 11 – 14



After you click Finish, your branch server and gateway will have been entered into Topology. After you publish this updated Topology you can then proceed to install the Skype for Business software onto your SBA or SBS server.

Survivability Features

Regardless of if you have an SBA or an SBS (or both), they really provide only one function: Users at the branch can continue to make and receive calls while the connection to the data center is down. There are a few other things that can happen on an SBA/SBS when the WAN is down, such as users homed on the SBA/SBS can still IM another user also homed on the same SBA/SBS. But primarily it's maintaining connectivity to the PSTN that matters here.

Below is a list that shows the features that are still available to users homed on an SBA/SBS while the link to the main data center (i.e., the Parent Pool) is unavailable:

- Inbound and outbound public switched telephone network (PSTN) calls
- Enterprise calls between users at both the same site and between two different sites, via the PSTN
- Basic call handling, including call hold, retrieval, and transfer
- Two-party instant messaging
- Call forwarding, simultaneous ringing of endpoints, call delegation, and team call services, but only if the delegator and delegate (for example, a manager and the manager's administrator), or all team members, are configured at the same site
- Call detail records (CDRs)
- Voice mail capabilities, if you configure voice mail rerouting settings.
- User authentication and authorization (Registrar)

Perhaps just as important as knowing which features will continue to work is to know which features will no longer work when a branch site is in "survivability mode":

- IM, web, and A/V conferencing
- Presence and Do Not Disturb (DND)-based routing (where calls are prevented from ringing on extensions that have DND activated)
- Updating call forwarding settings
- Response Group application and Call Park application
- Provisioning new phones and clients, but only if Active Directory Domain Services is present at the branch site.
- Enhanced 9-1-1 (E9-1-1)

The user contact list will also be unavailable. In order to search for a user, they will have to enter the full SIP address for that user.

So it's OK to use an SBA/SBS when you only need to maintain inbound and outbound calling to a single branch while the connectivity to the Parent Pool is unavailable. However, if your branch site needs to have things such as conferencing working, then you should consider installing a Standard Edition server at the branch site.

It should also be noted that when a parent pool fails for an SBA, the SBA is not moved to the backup pool. The users on the SBA will be in reduced-functionality mode until the parent pool returns to functionality.

Let me re-iterate this here – conferencing happens only on the parent pool. SBA/SBS have no conferencing components installed. As such, all conferencing traffic is passed up the WAN to the conferencing servers in the parent pool. This can add a burden on your WAN if it is already nearly saturated. And if this WAN is down, your Branch Site users will only be able to join a meeting in one of 2 ways:

1. Dial-in Conferencing to a number at a different location
2. Connecting to the meeting via the Web client from an external location. They won't be able to log in with their full Skype for Business client as their home pool (aka "Registrar") is not accessible from the main data center during the WAN outage. More on this scenario will be covered later in this chapter.

Standard Edition Servers

If you cannot live with the limitations of an SBA/SBS, then the best advice is to install a Standard Edition server at the remote office. This will give you full resilience while the WAN is down. Conferences hosted on the Standard Edition server will continue to work. Users can update their buddy lists; presence will work – at least for users in the local office until the WAN returns.

So if a Standard Edition gives you full resilience, why even bother with an SBA/SBS solution?

One of the primary reason has to do with the management of your network and server infrastructure. Many remote offices don't have a server room. Or they share a closet with the networking gear and the office cleaning supplies. By putting everything into a single unit (an SBA) you minimize the amount of hardware that you need to install and maintain at a remote site.

The other issue has to do with resiliency. If the local Standard Edition server fails, what happens to your users? With an SBA, the SBA-homed users automatically migrate to the parent pool and continue to work just fine –

barring additional stress on the WAN. But if a Standard Edition server goes down, there is no automatic failover to the parent pool.

With Standard Edition, you need to have a paired-pool for failover. In other words, in a branch-office scenario, you'll either have to install two Standard Edition servers in the branch office, or install a second Standard Edition server in your main HQ. (*Note that you can pair Standard Edition and Enterprise Edition pools too though Microsoft doesn't officially support this*). This could lead to doubling the amount of servers you need. So while Standard Edition provides a full feature set when the WAN is down, it's a bit more of a challenge to architect the solution for when the Standard Edition server in the Branch Site has a failure.

Summary

There was a lot covered in this chapter. It basically boils down to this:

Spend a lot of time architecting your voice resiliency and work with your telephony providers to implement the best solution.

Skype for Business has a lot of options for voice resiliency built right in. It's one of the reasons that it has become a very popular PBX replacement. However, like any complex system, you need to be aware of what your options are and how they best fit into your infrastructure.

Since you are reading this book, you are interested in becoming an expert on Enterprise Voice. However, Enterprise Voice is perhaps the most difficult concept in Skype for Business. Bringing in a veteran consultant to review your architecture and provide feedback will probably prove to be a very worthy investment.

Chapter 12 – Testing and Troubleshooting

Just as with any other technology in IT, the ability to fix a problem when things go wrong is just as important as knowing how to configure the technology in the first place.

For the most part, Skype for Business has some very good testing and troubleshooting tools. Most of the tools are provided natively by Microsoft but where there are gaps third party tools are available to assist.

Regular Expressions

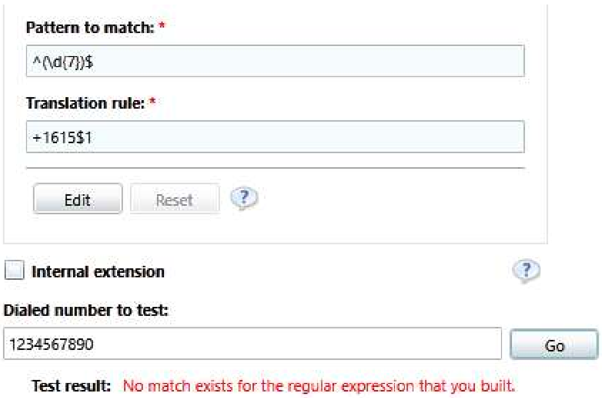
When working with Dial Plans, Voice Routes, and Trunk configurations you will most likely have to deal with regular expressions. If certain calls are not completing, it may be due to an incorrect regular expression. So how do you test if your regular expressions are correct?

Within the Skype for Business Control Panel, Microsoft has provided simple testing within the interface. If you look at the example below, you can see an example of this from the Dial Plan settings page.

Figure 12 – 1

In the “Dialed Number to test” field, you can enter the number that isn’t working. In the example above there is a match which is easy to see as the normalized number is displayed in green. Meanwhile if there is not a match, Control Panel will point this out using red text.

Figure 12 – 2



No matter if you are testing a Dial Plan, a Voice Route, or a Trunk Configuration the display is the same. You enter the number that you expect to work and Control Panel returns a result. This is a very welcome feature but it doesn’t tell you *why* your Regular expression doesn’t match.

For more advanced cases you will need to use a 3rd party solution which pretty much means going to a website dedicated to analyzing regular expressions.

There are several of them that all work relatively well. Here are two websites you can use to help test your regular expressions.

Debuggex - http://www.debuggex.com

Regex Hero – http://regexhero.net/tester (Use the "replace" tab when testing)

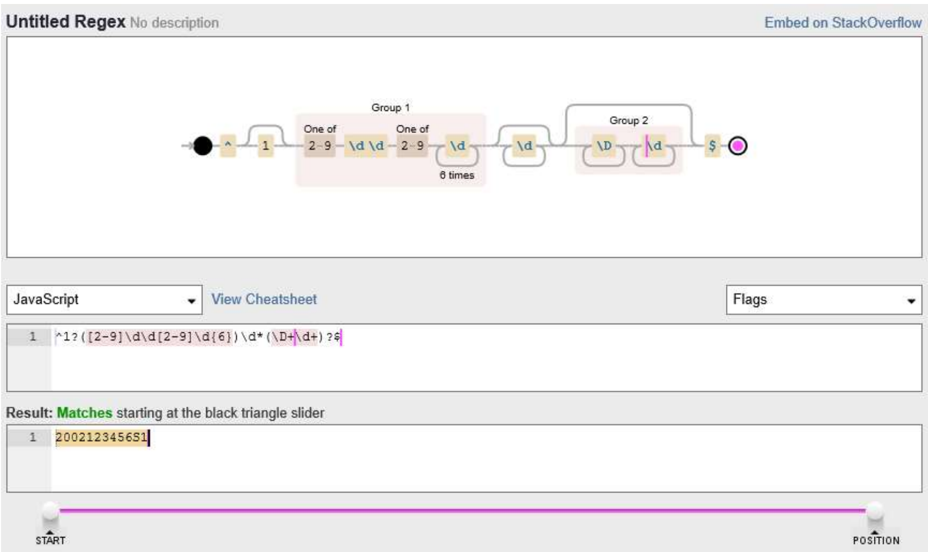
I use debuggex.com but I know that many other Skype for Business administrators prefer Regex Hero. Both of them do the same thing so try both out and see which one you like better. Regex Hero is closest in concept to how Control Panel is designed but Debuggex has nifty graphics which help visualize the regex.

After you navigate to http://www.debuggex.com, enter your Regular expression into the middle field. In this case, I've enter the following regex:

```
^1?([2-9]\d\d[2-9]\d{6})\d*(\D+\d+)?$
```

This may seem like a bunch of gibberish but this is where Debuggex shines. As can be seen below, it breaks down the Regular expression visually.

Figure 12 – 3



In the bottom field you type in the numbers that you think should match the Regular expression. As you type in the values, a magenta colored pointer travels from left to right showing which part of the regex is currently being analyzed. In the example above, the marker is in front of the last \d. The circle at the end is also colored magenta showing that you have matched the Regular expression successfully.

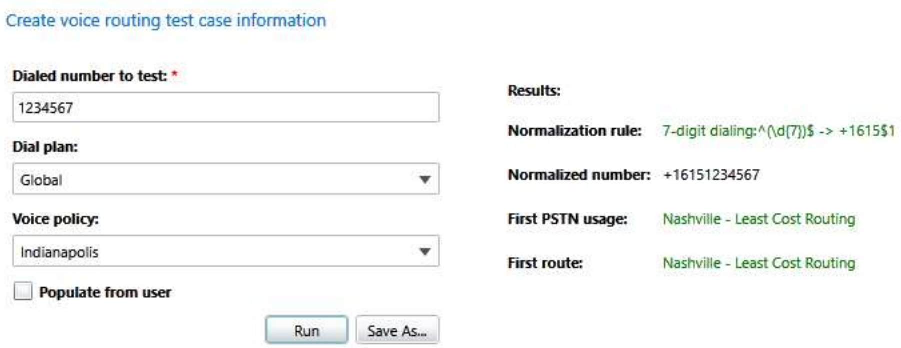
I've found that this visual display has helped me greatly when troubleshooting why a given regular expression isn't working.

Voice Routing Test Case

At the top of every tab in the Voice Routing section of Control Panel is a link named "Create voice routing test case information".

When you click this link, it slides the screen down and pops in some fields. This tool is a quick way to test if a user is assigned the correct Dial Plan and Voice Policy. To use this feature, enter the digits that the user dialed (before Dial Plan normalization) and then select the users assigned Dial Plan and Voice Policy.

Figure 12 – 4



After setting those values, click the "Run" button. The dialed number will be analyzed against the Dial Plan and Voice Policy and the results will be shown to the right. In the example above, all of the fields are green which lets you know that this is a valid number that will be normalized and routed through the listed PSTN Usage and Route.

Sometimes you won't know which Dial Plan or Voice Policy is assigned to the user you are testing. To save time in looking these values up you can tick the "Populate from user" check box. After ticking this box, you are given the option to type in the users SIP address. If you don't know it, then you can click the "Browse…" button to search for the user.

After you find the user in the browse window, the Dial Plan and Voice Policy fields will be automatically updated to match the settings of the user. With these now set correctly for your user, you can click the "Run" button to see if the dialed number is valid for this user.

This is a quick way to test if a user can't make a call due to being limited to only making local calls instead of tolls calls via a Voice Policy. Note that the call could also fail due to being routed to the wrong gateway or if a Trunk Configuration isn't being applied. So this tool can help troubleshoot some of the reasons for a failed call but not all of them.

Clicking the "Save As…" button lets you save this test if it's something you think you will need to use often.

Test Voice Routing

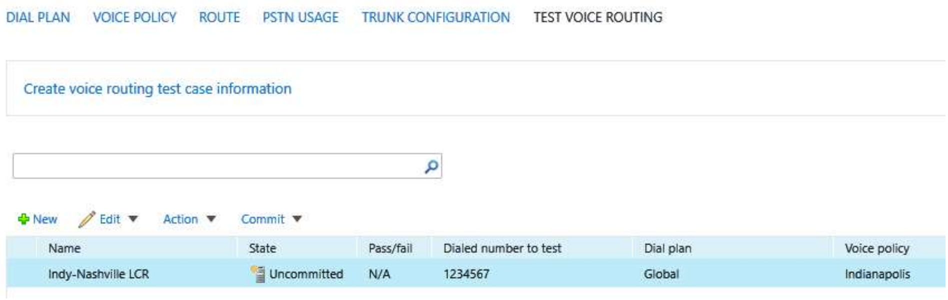
Within the Voice Routing section of Control Panel there is a tab named "Test Voice Routing". Using this tool, you set all of the settings that you expect a call to take – which Dial Plan, Voice Policy, etc. You then run the test. If you're desired settings match with how Skype for Business is configured, you win a prize! If not, you get a bunch of red errors.

I'm honestly not sure what the real value of this section is as it just lets you verify what you *think* you have configured with what you actually have configured.

To navigate to this testing tool, open Control Panel, click on Voice Routing, and then select the last tab on the right named "Test Voice Routing".

Initially this will show no entries and will be empty…unless you've saved a Voice Routing test case from the previous section. In that case, the saved test will be found here.

Figure 12 – 5



To create a new test case, click the "New" button. And a window will appear where you can fill out how you think you have configured a given voice call path. Start off by providing a name for this test and then entering the number that you want to call.

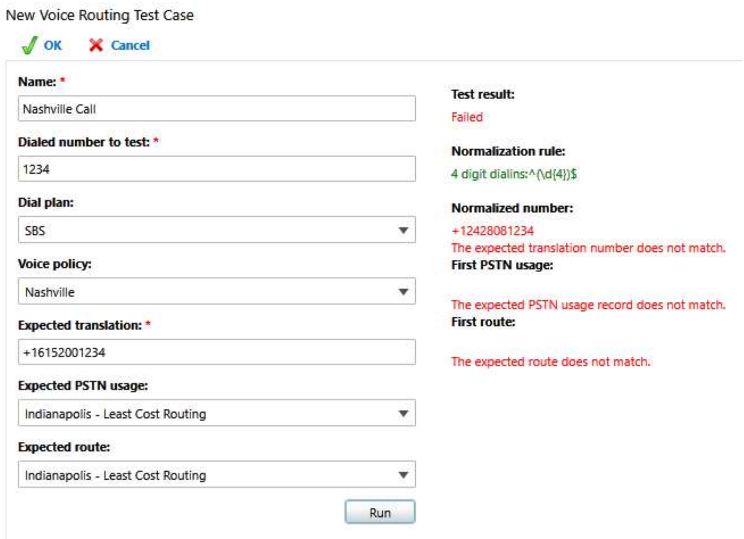
Next, select the Dial Plan you want to use to normalize the number. Then select a Dial Plan and Voice Policy that you have assigned to the user you are testing. This is all very similar to the "Voice Routing Test Case" above but without the convenience of looking up a user and automatically populating these fields.

Then you need to type in what you expect the dialed number to be normalized as. For example, if your test number is a 4-digit extension then add the required digits that would make this a properly formatted E.164 number.

Then select the PSTN Usage and route that you expect this call to take. Once all of this is filled out click the "Run" button and see if you win!

If everything is green, go to the kitchen and eat a cookie as your prize. If you're wrong, go stand against the wall for one minute as punishment.

Figure 12 – 6



Looks like I need to go stand against the wall for one minute.

The reason I'm not such a big fan of this test is that it doesn't provide any help in what went wrong. It looks like my expected normalized number is wrong but at least it tells me how Skype for Business will translate my 4 digits using the Dial plan I selected. However, it provides no help as to which PSTN Usage or Route the call would actually need to take to be a successful call. It's up to me to change the fields until I luck in to a success.

I haven't used this section in well over a year and I don't know why I'd use it in the future. However, if you are new to Enterprise Voice, this is a pretty good tool to help you visualize all of the required steps needed to handle a call successfully. When I was first learning all of this I used this section a fair amount. But as I've started mastering all of this, I've found it adds little to no value to my ability to test and troubleshoot Enterprise Voice.

Debugging Tools

There are no logging tools shipped with Skype for Business Server. Rather you need to download them from Microsoft's website. Just search the Internet for "Skype for Business Debugging Tools" and the first link should be the right one. Download the installer, copy it to all of your Skype for Business servers and install them.

CLSLogger

All logging in Skype for Business is handled by a subsystem called the Centralized Logging Service (CLS). When configured correctly, CLS will provide you with the relevant logs you will need to troubleshoot issues within Skype for Business.

The primary tool provided by the Debugging Tools to control the CLS is the CLSLogger utility. There is nothing that the CLSLogger can do that you can't do via PowerShell. However, most people will find using the CLSLogger graphical user interface to be a much simpler way to log what is happening within Skype for Business than using PowerShell.

CLSLogger does not ship with Skype for Business Server. To get it, search the Internet for "Skype for Business Server debugging tools". The first returned link should be the right one. Download it and install it on each of your servers.

Before you even fire up the tool, run `Get-CsClsConfiguration` from the Skype for Business Management Shell. This will let you know how CLS is currently configured in your environment.

For the most part, ignore everything returned except for the "ETLFileFolder" and "CacheFileLocalFolders" values. By default, this is set to `%TEMP%\Tracing`. And where exactly is that directory?

```
%WINDIR%\ServiceProfiles\NetworkService\AppData\Local
```

Or most likely, here:

```
c:\windows\ServiceProfiles\NetworkService\AppData\Local
```

These directories are important because this is where the log files are going to be created. You may want to edit this to get them off of your C: drive so you don't accidentally fill it up.

And since I never like editing the Global configuration of anything (if it can be avoided), create a new CLS Configuration for your Skype for Business Site(s). This is done with the `New-CsClsConfiguration` cmdlet. Below is an example creating a new configuration for a Site named "Site 2" and changing the logging folders to `D:\logging`:

```
New-CsClsConfiguration -Identity "site:Site 2" -ETLFileFolder
"d:\logging" -CacheFileLocalFolders "d:\logging"
```

Even if you only have a C: drive, you may want to change the directories to something you can find more easily than
`%WINDIR%\ServiceProfiles\NetworkService\AppData\Local`

CLS will automatically create the directory you define so that saves time connecting to every server in your site to create a directory. Just make sure that if you set the logging to a drive other than C: that every server in your site has that drive. Otherwise your logs on servers without an D: drive will be stored in the default location.

So, now that the directories have been defined, it's time to launch CLSLogger, right? Not quite.

CLSLogger is really a super elaborate PowerShell wrapper. And if there's one thing you've learned about running PowerShell scripts it should be this: You need to set the Execution Policy before you can run somebody else's script, even Microsoft scripts.

So open a PowerShell window with "Run as Administrator" and type the following to permit this to run:

```
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy
Unrestricted
```

If you are super security conscious, you may want to use something other than Unrestricted. But that's up to you to figure out the magic voodoo. I always use Unrestricted but then I don't entirely know what I am doing with this – I'm not a PowerShell expert. I just set it to Unrestricted because it tells PowerShell to never bother me again with this security nonsense but you should be sure to follow your organizations security requirements on this.

So now, assuming you have installed the Debugging Tools already, you can finally go ahead and launch CLSLogger.

Navigate to `C:\Program Files\Skype for Business Server 2015\Debugging Tools` and launch ClsLogger.exe

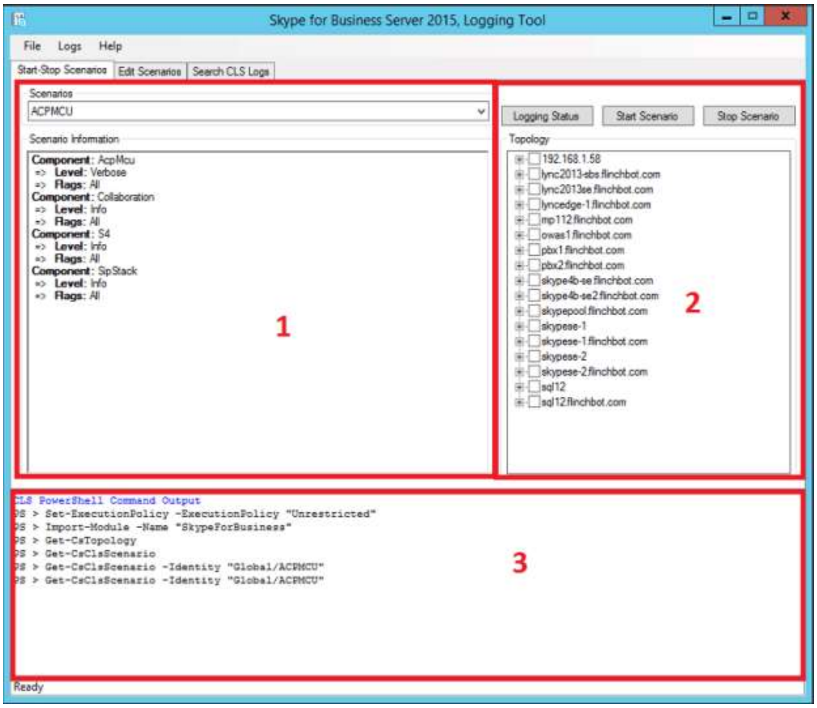
Now, even with the execution policy in PowerShell set correctly, you still get:

Error: Access to the registry key 'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\PowerShell\1\ShellIds\Microsoft.PowerShell' is denied. To change the execution policy for the default (LocalMachine) scope, start Windows PowerShell with the "Run as administrator" option. To change the execution policy for the current user, run "Set-ExecutionPolicy -Scope CurrentUser".

Oh wait…Run ClsLogger as Administrator. Otherwise it won't have rights to read the keys from the registry.

When it starts, the Start/Stop Scenarios tab opens and there are three main sections.

Figure 12 – 7



Section 1 is where you can select a scenario.

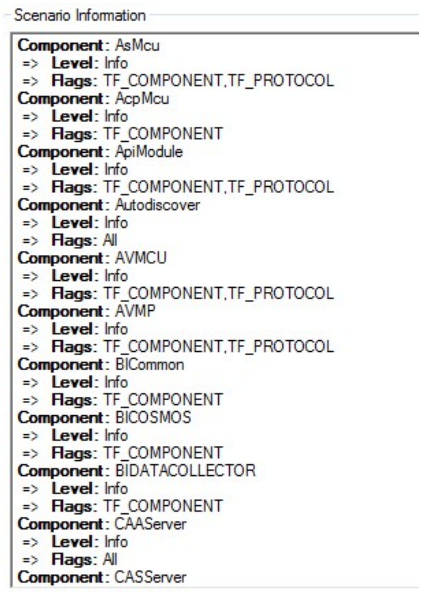
Scenarios take the guesswork out of your hands. Microsoft has created a whole pile of scenarios for you with all of the correct options already selected. So for example if you aren't tracking an Enterprise Voice issue, S4 will be unselected for you thus minimizing the amount of "noise" in your logs

To select a scenario, just hit the pull down and pick the desired scenario from the sorted list.

If you have multiple sites, you will also notice that the list contains duplicates…possibly a lot of them. If you type `Get-CsClsScenario | select Identity` in a PowerShell window you will see that the list in PowerShell matches the list in CLSLogger. Except the PowerShell version prepends the scenarios with the site name making the duplication easy to understand.

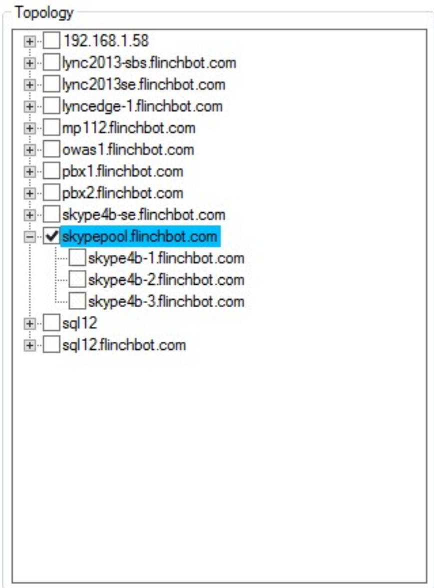
Fortunately, all of the out-of-the-box scenarios are the same regardless of Site so you can pick any one of them. In this example, I selected AlwaysOn because I am lazy and that is the default option. After selecting a scenario, the Scenario Information screen will list all of the logging options, logging levels, and logging Flags that are part of the scenario.

Figure 12 – 8



After selecting the scenario, you have to tell the logger against which pools and servers (and trunks?) you want to run this scenario against. (Never select a trunk. It won't give you any logging). So pick a pool or two off the list. You can also drill down to pick a specific server too in case you suspect a subset of your pool is having issues.

Figure 12 – 9



The 3rd section, at the bottom of Figure 12-7, is a PowerShell window showing you which PowerShell cmdlets the tool is running.

There are three buttons above the Topology window (Figure 12-7, Section 2). Before running a new scenario, click the "Logging Status" button to see if you are already doing logging against the pool/server(s) you selected in the Topology window.

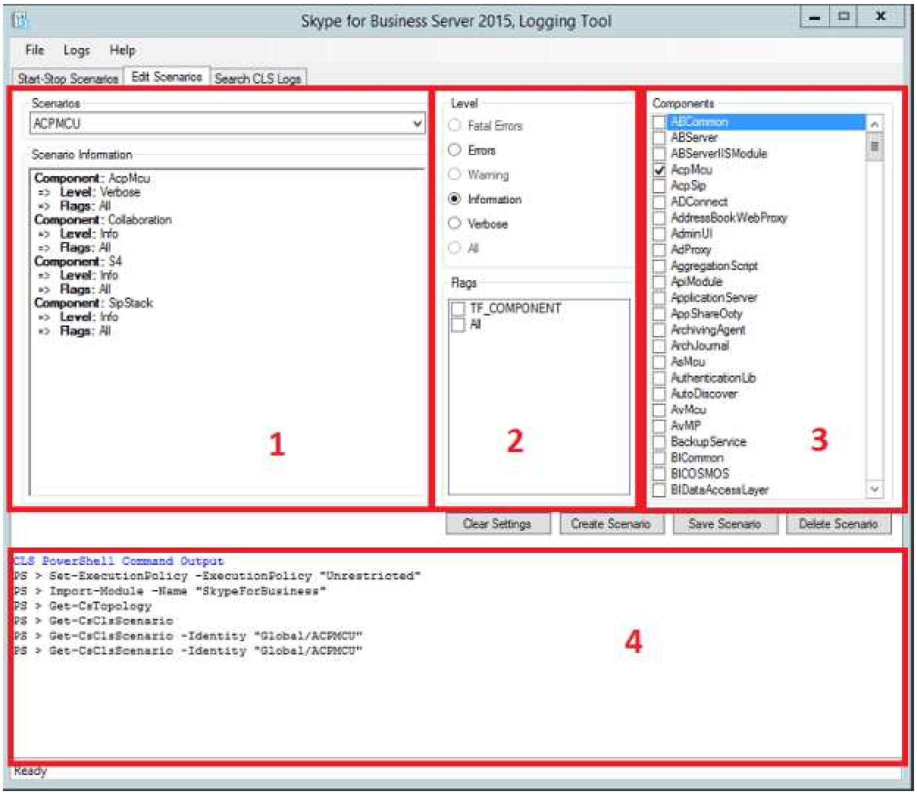
If everything looks ready, click the "Start Scenario" button.

Replicate your issue as fast as you can and then click the "Stop Scenario" button. No one wants to deal with a 37GB log file because you couldn't be bothered to stop the logging until hours after the issue has been tested.

Creating a Scenario

If none of the pre-configured scenarios match what you want to log, you can create your own custom scenario. To do this, click on the "Edit Scenario" tab and the following screen appears.

Figure 12 – 10

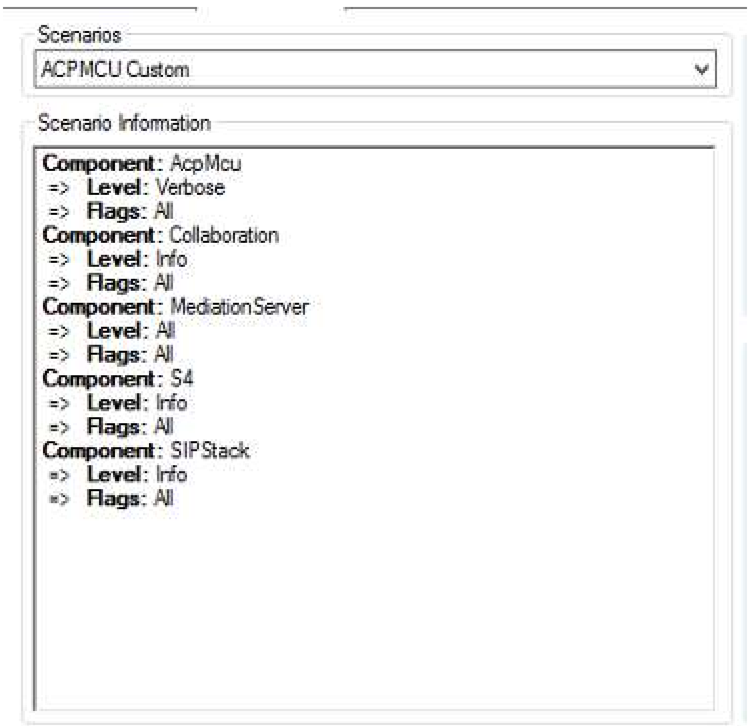


In general, you will be editing one of the existing scenarios because you need an additional component to be logged or you need more detail than what is provided. In the image above, the ACPMCU scenario is selected. You can see in section 1 that this scenario logs AcpMcu as verbose and Collaboration, S4, and SipStack as Information-level. Let's say we want to add logging of the Mediation service to this scenario.

Scroll down in section 3 to find the "MediationServer" Component. Section 2 will adjust to "Information" and Flags will go to "All". Just for fun, let's change the Level value from "Information" to "All".

Now you click on the “Save Scenario” button underneath section 3. You will be prompted to verify the change. After saving the change, Section 1 will be updated. Note that MediationServer is now added to the list.

Figure 12 – 11



The Scenario name now also has a new name I gave it which is admittedly not terribly descriptive. I just added “Custom” to the default name.

To create a new scenario, click the “Create Scenario” button found underneath sections 2 and 3 – right above the 4 in Figure 12-10. You’ll be prompted to provide a name so give it something descriptive beyond just “Test Scenario” or “Flinchböt’s Scenario 1”. A better name would be something like “Inbound calls from SIP trunk”.

Now if you flip back to the Start-Stop Scenarios tab, you should see your new scenario in the Scenarios list. Select this and start your logging.

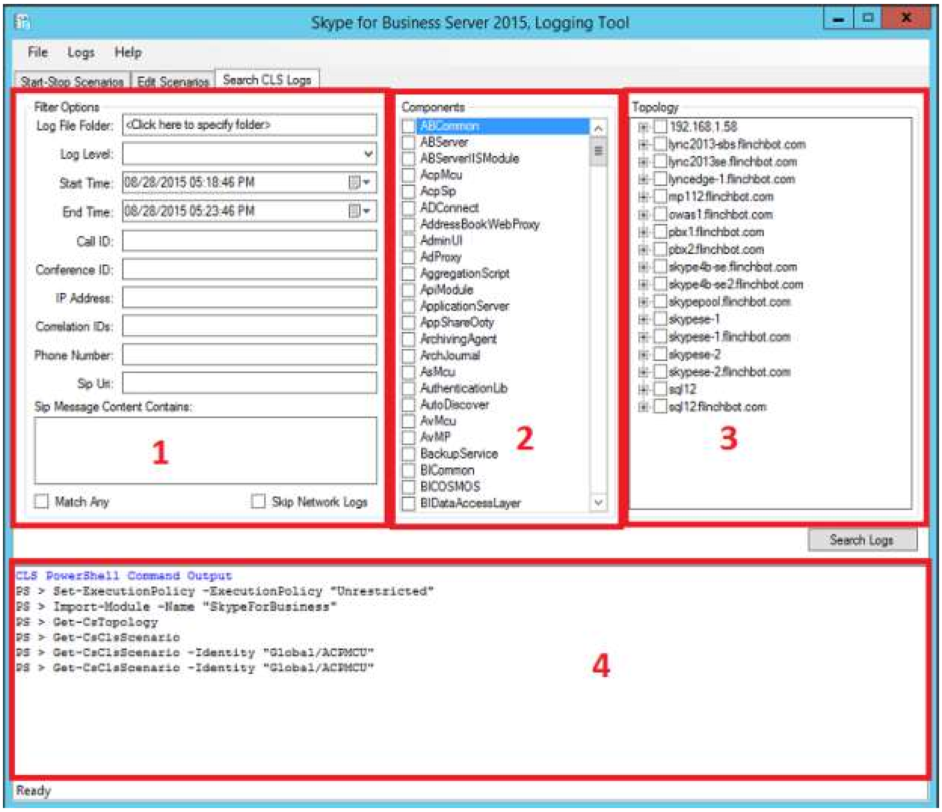
Searching CLS Logs

So the next step: looking at all of this wonderful data. That's where the "Search CLS Logs" tab comes into use.

Unfortunately, nothing is automated here.

From my point of view, the point of this tab is to get a unified log file that you can open in the Snooper tool. As such, I leave most of this stuff empty.

Figure 12 – 12



The Snooper tool takes all of the data collected by CLSLogger and presents it in a very nice format making it easier to view the logs. The Snooper tool is installed as part of the Skype for Business Debugging Tools.

Make sure the Start and End Time fields are correct. To me it feels like this is a semi-random guess by CLSLogger to what window of time you want to review. So don't trust it to be correct or your logs may be missing the data you are looking for.

The only setting in Section 1 you have to set is the Log File Folder setting. You can't type anything here and have to navigate to a folder. If you aren't running this on a server then manually create the folder on your PC in the exact same location it is on your server. I hope your PC has a D: drive like your server does. If not, run this from the server.

Remember when we set the logging directory at the very beginning? Navigate to there and click OK.

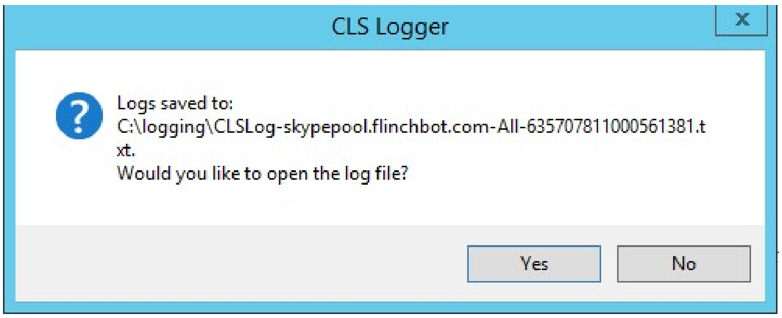
You do not have to pick anything in section 2. So unless you really want to narrow down your logs, leave this blank. If you are using the AlwaysOn scenario, then it's a good idea to check the relevant boxes here. How do you know which boxes to select? Go back to the "Start/Stop Scenario" tab and pick a specific scenario off the list and manually replicate what's pre-configured there into Section 2.

In Section 3, you have to select the pool you logged against.

Once you have at least a Log File Folder, a valid time range, and a Server/Pool selected, click "Search Logs". This will go out to the servers, grab the logs from each of them, and put them into one big file.

Curiously, once this is done it asks you if you want to open the log file. Pick "No". This just launches Notepad and opening really large log files in Notepad is painful. The whole point of Snooper is so that you don't have to use Notepad. But do take notice of the file name for the log.

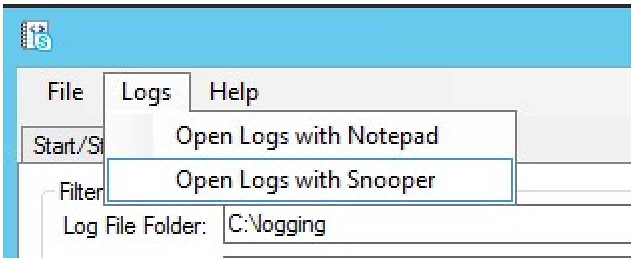
Figure 12 – 13



You don't have to memorize the log file path in the message. It is conveniently listed in the PowerShell monitor.

So how do you get this log file opened in Snooper? Go to the menu at the top of the Logging Tool and select the "Logs" pull down. Then select "Open Logs with Snooper". Like magic, Snooper opens and pulls in the log file automatically.

Figure 12 – 14



Snooper

Snooper is the tool within which you will do most of your troubleshooting. Snooper takes the logs created by the Centralized Logging Service and it presents them in a legible format. Instead of just seeing line after line of text, Snooper breaks everything out by individual lines with color coding and built-in intelligence. It knows which lines in the logs are related - allowing you to quickly filter out all of the noise and concentrate on the main "conversation" of interest to you.

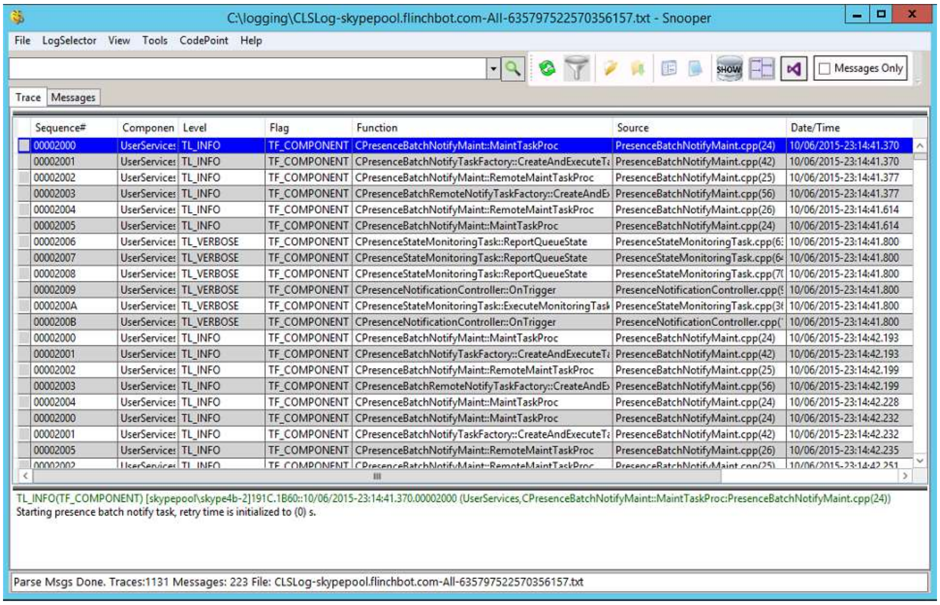
Snooper is not installed as part of the Skype for Business installation. It is installed as part of the debugging tools. If you haven't installed those yet, you can get them by doing an Internet search for "Skype for Business Debugging Tools".

After installing the debugging tools, Snooper is found in the `C:\Program Files\Skype for Business Server 2015\Debugging Tools` directory. Double click on Snooper.exe to launch it. But generally speaking, you won't be launching Snooper manually. Instead, you'll launch it after performing a logging session using the CLS tool described earlier in this chapter.

After doing some logging, and then searching the logs, you launch Snooper by selecting “Open Logs with Snooper” from the Logs menu. (See Figure 12-14)

This will launch Snooper and automatically load the logs that you have just collected. Below is a graphic showing the initial screen following the launch of Snooper.

Figure 12 – 15



The initial screen is set to the “Traces” view. This view is the raw dump of what was returned by your Centralized Logging session. In general, you can ignore the Traces view as the Messages view is a much more user-friendly view of the data. However, there are messages in the Traces view that do not appear in the Messages view. On rare occasion, you will need to scour through the Traces view looking for useful log entries.

To ease the searching of both Traces and Messages, there is a search box right above the Traces and Messages tabs. In this field you can type any text to search the logs. For example, if you are troubleshooting an inbound call that is failing, you could type in the phone number you are trying to call. This search will skip straight to the first occurrence of the phone number in the log file.

Search terms surrounded by quotes will search that specific term. So searching for “invite sip:” will search the logs for that specific phrase. If you do not use the quotes, the search will become an “or” search meaning search for *invite* or *sip:*.

For even more advanced searching, you can use searches with keywords.

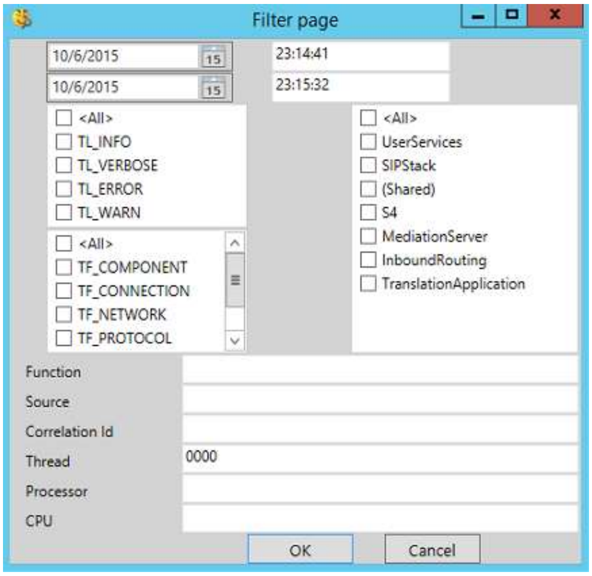
Keyword	Description	Examples
callid	Search only the dialog ID (SIP call ID)	callid: 1202271547 callid:#current# (match the ID of the currently selected message)
from	Search only the From column	From:elvis@flinchbot.com
direction	Search only the Direction column	Direction:in Direction:none (messages having no direction)
flag	Search only the flag/icon column	Flag:# Flag:mark Error =! Mark = # Any = errors and/or marked
logtype	Matches only messages from the specified type of trace	Logtype:stack Logtype:S4 Stack=SIPStack Uccp = Client Logs Focus = UserServices Mcu = MCUInfra S4 = S4 Ldm = LDM Medsrv = Mediation Server
message	Search the entire message	“invite sip:” Message:”Invite sip:” These two are the same as Snooper assumes it’s a message search if no type is given.
startline	Search only the Startline column	Startline:INFO Startline:flinchbot@flinchbot.com
time	Search the Time column	Time:12 Time:”12:01:39”
to	Search only the To column	To:elvis@flinchbot.com

If a search is turning up too many entries, you can apply a filter the log. To the right of the search box is a row of icons. The second icon is the filter icon.

Figure 12 – 16, The Filter Icon

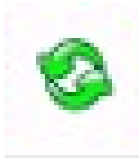
After clicking on the filter icon, you are presented with a filter screen. From here you can filter the log on such attributes as a specific date and time range, log file level (Warning, error, etc.), or log source *S4, SipStack, etc.). After setting your filter, clicking OK will apply the filter.

Figure 12 – 17



To remove a filter or to clear a search, press the green “refresh” icon to the right of the search box. To me, this icon means to “refresh” the log – such as loading in new entries. However, in this case it has a slightly different meaning in that it will refresh the log by removing any searches or filters.

Figure 12 – 18, Clearing searches and filters



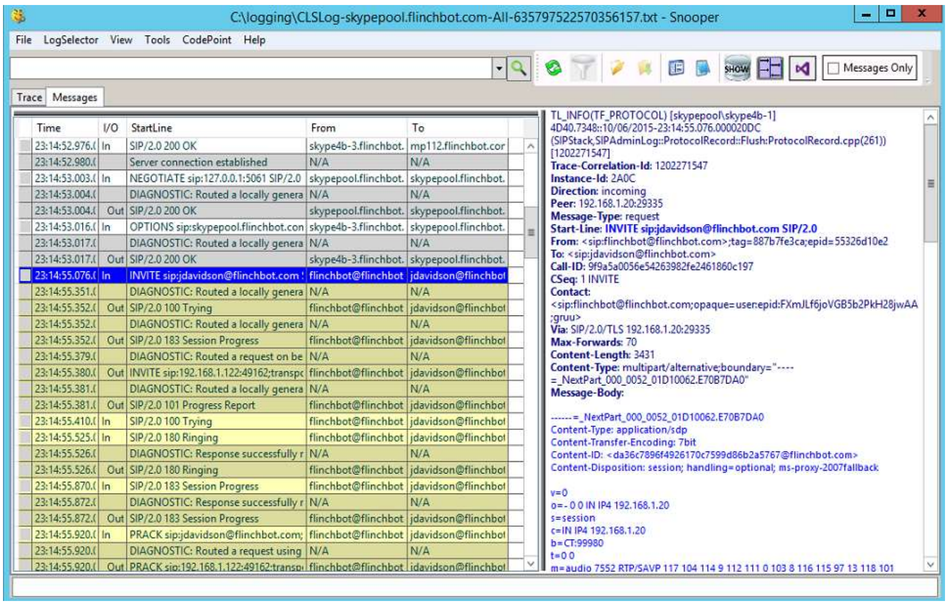
As mentioned before, most of the work you'll be doing in Snooper is in the Messages view. To get to this view, simply click the Messages tab.

The messages tab is broken into two panes. On the left is a list of all of the log messages that Snooper can display as a "Message". The messages that can be displayed are SIP, CCCP, S4, MCUInfr, Client UCCP, Focus C3P, PSOM/LDM, and Mediation messages.

The right pane shows the detail of a message selected in the left pane.

As seen below, there is a log message that was selected. A selected log message changes to a blue background. To the right is the detail of this log message.

Figure 12 – 19



The messages in yellow (bottom half of figure 12-19, on the left) are automatically highlighted to reflect that they are part of the SIP conversation with the blue highlighted message. This lets you quickly know which messages in the log are related to one another. For example, the messages in the log above the blue message are not related as they have either a gray or white background.

If you want to quickly filter the log to only show the related messages (i.e., the ones in yellow) you can right click on any of the yellow messages. A context menu pops up giving you a few options.

If you want to “mark” the message so you can easily find it later, select the Mark option. This will set the selected message to be highlighted in Green. This will help you more easily find a few log entries that interest you by just looking for the green highlighted messages.

The most useful option in this menu is the “Find Related” option. This will automatically filter the log and only show messages that are related to the selected line.

How does Snooper know which messages are related? Well, each message in a SIP conversation is tagged with a conversation ID. If you look closely at Figure 12-19, on the right hand side you will see this number “1202271547”. This number will appear in every one of the yellow highlighted messages. So all that “Find Related” does is apply a filter based on the unique conversation ID found in the SIP messages.

The Copy option takes the selected message and copies it to your Windows clipboard. This is handy so you can paste specific log messages directly into your notes.

Figure 12 – 20

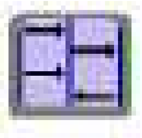
Clicking "Go to Nearest Entry in Trace Viewer" will change the view back to the Traces view and highlight the message selected. This is useful if you've found part of what you want but feel like something in the logs is missing. This will send you back into the Traces view so you can see if there were some log entries that could not be displayed in the Messages view.

"Clear Search and Select this message" is used after doing a search. Whenever you do a search, you are really filtering the log to return only those messages that contain the search term. If you have found a message you are interested in and also want to see the rest of the messages too, then select this option. Your selected message will be highlighted and the rest of the messages will also reappear.

After you've found the SIP conversation you are trying to troubleshoot, it might be useful to see a graphical representation of the messages in the conversation. This is where the "Call Flow Window" comes in very handy.

To open a call flow window, highlight a SIP message and click the Call Flow Window button. This button can be found near the end of the row of icons to the right of the search window.

Figure 12 – 21, Call Flow Window icon



After clicking this button, a call flow window will open. This visual representation is often also called a “ladder diagram” – which is the name I usually use for it.

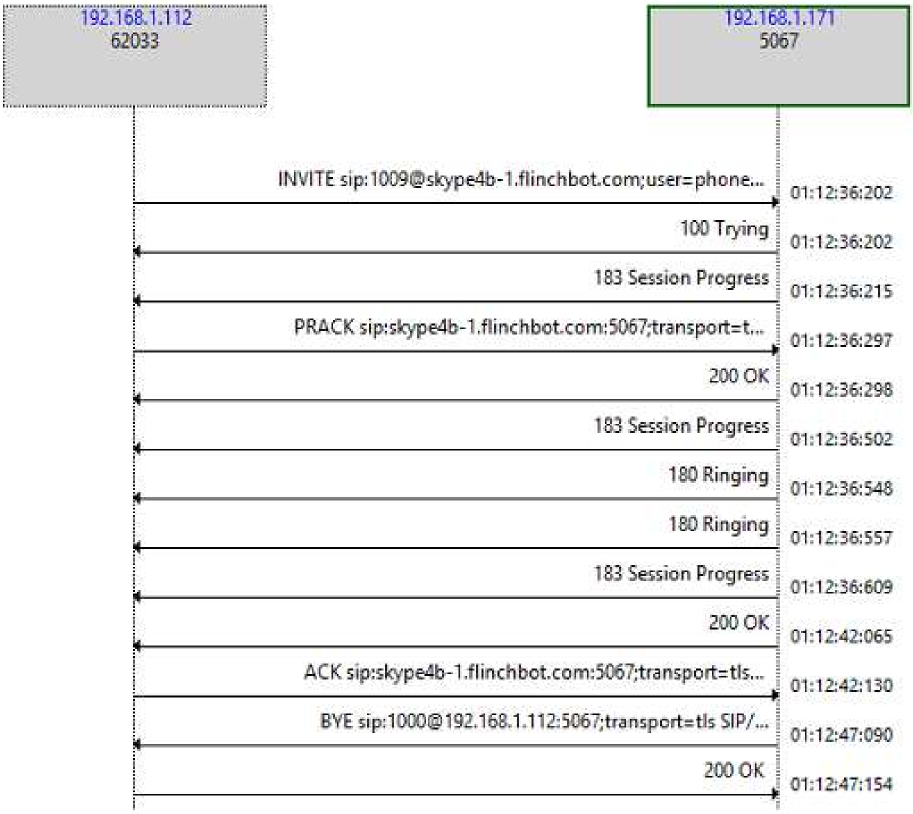
This ladder diagram shows all of the nodes that are involved in the conversation. This will always involve at least one Skype for Business Server and at least one end point which could be a user, a PSTN gateway, or another Skype for Business Server such as an Edge server or Mediation server.

Figure 12-22 shows a call from the PSTN to a Skype for Business user. The PSTN gateway is on the left with the IP address of 192.168.1.112. The device on the right is the IP address of a Front End server with a collocated Mediation service.

You can click on any of the messages in the diagram and the Snooper window will jump to the selected message.

The Call Flow diagram view is a very useful view that you can use to quickly see which devices are part of the call flow. Once you start using this view you’ll have a better understanding of not just the issue you are trying to troubleshoot but how conversations flow within the Skype for Business environment.

Figure 12 – 22



Debugging a Call

Let's take a look at the ladder diagram in Figure 12-22. This is a capture of an inbound call from the PSTN to a Skype for Business user. On the left is the gateway (192.168.1.112) and on the right is a Skype for Business server.

The first thing that happens is the SIP Invite message. This is the first step of any conversation. The gateway sends an invite to the Mediation Server. Below is the actual SIP Invite header.

Figure 12 – 23

```
INVITE sip:1009@skype4b-1.flinchbot.com;user=phone SIP/2.0
FROM: <sip:1000@nas-gw.flinchbot.com>;tag=1c391760207
TO: <sip:1009@skype4b-1.flinchbot.com;user=phone>
CSEQ: 1 INVITE
CALL-ID: 391759601511201511 12@192.168.1.112
MAX-FORWARDS: 70
VIA: SIP/2.0/TLS 192.168.1.112:5067;branch=z9hG4bKac391775866;alias
CONTACT: <sip:1000@192.168.1.112:5067;transport=tls>
CONTENT-LENGTH: 449
SUPPORTED: em,100rel,timer,replaces,path,resource-priority,sdp-anat
USER-AGENT: MP-112 FXS/v.6.60A.294.004
CONTENT-TYPE: application/sdp
ALLOW: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
```

At the very top you can see that the inbound phone number is simply "1009". This is a classic example of where the telephone company only sends the last four digits of the dialed phone number. It will be up to us to add the rest to make this a proper E.164 phone number.

This is my lab, so the FROM: number is a little unrealistic in that it is only "1000". But we really don't care about the FROM: number here so it doesn't matter.

As mentioned, the TO: field is the number being called.

CSEQ is the call sequence. It starts with 1. As the call progresses this will increase in each further step of the process.

CALL-ID is the unique ID of this call. Remember where I said in Snooper that you can use the "Find Related" feature to show only the relevant entries in the log for a given call? That feature is based on the CALL-ID field.

In order to prevent endless call looping, the MAX-FORWARDS field will decrement each time the call is forwarded to another server. When this counter hits 0, the call can no longer be forwarded.

In the "via" header you can see that this came in from the gateway. I know it's the gateway because of the IP address – 192.168.1.112:5067. Notice the 5067 which is the TLS port for my Mediation server. As a reminder, this port gets defined in Topology Builder.

The CONTENT-LENGTH field is just used as a kind of checksum. If you were to do the math, the message-body is 449 bytes long.

The SUPPORTED header field contains a list of option tags. No one knows what these mean. OK, they are defined in an RFC somewhere but vendor support would be the only people who really care.

The USER-AGENT field is also interesting. Here the SIP header is stamped with the name of the gateway, which in my lab is an AudioCodes MP112.

The media type is defined in the CONTENT-TYPE header. In this case it is SDP (Session Description Protocol).

Finally, we get to the ALLOW field which lists all of the SIP methods supported by the gateway.

The body of the invite contains the media settings that are supported by the gateway. I won't go into full detail of this but I will point out the interesting bits.

Figure 12 – 24

```
v=0
o=AudiocodesGW 391731739 391731738 IN IP4 192.168.1.112
s=Phone-Call
c=IN IP4 192.168.1.112
t=0 0
m=audio 6000 RTP/AVP 8 0 13 101
a=ptime:20
a=sendrecv
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:ZB4kXTU+F488SsDA09m8K4muZnUesksObhekXWKV|2^31|95:1
a=crypto:2 AES_CM_128_HMAC_SHA1_32 inline:bU/7atIzWTCcWbapxF3jsR5HuK7vgmMJgVZWyWOx|2^31|238:1
```

The "v" is the protocol version number which is always 0. "s" is the session name which in this case uses the easily understood name of "Phone-Call".

Connection information is defined by the "c" field. This contains the IP address of the gateway.

The time that the session is active is defined by "t" and can pretty much always be ignored.

It starts getting interesting with the "m" field. This field describes the supported media name and protocols. Some of this is easy to understand. The media type is audio and the protocol is the Real-time Transport Protocol

(RTP) Audio-Video Profile (AVP) The next set of numbers starts describing the media payload types. The full list of payload types can be found here: https://en.wikipedia.org/wiki/RTP_audio_video_profile

What this is telling us is that, in order, the gateway prefers the following media.

8	G.711 PCMA (A-Law)
0	G.711 PCMU (µ-law)
13	Comfort Noise
101	DTMF (Dial Tones)

So our Mediation Server has to support at least one of the first two if we want to have any audio. Fortunately, the Mediation Server supports the G.711 protocol and its variants.

The “a” values are used to set additional attributes for the media. This section is very useful if you are troubleshooting issues where the phone rings but you don’t get audio. It’s possible that there is a media mismatch – one end is offering codecs that the other side doesn’t support. Or the values are off. For example, one end only supports G.711 A Law while the other end only support G.711 µ-law.

Back to Figure 12-22…

The next step in the conversation is the “100 Trying”. Here the Mediation Server is responding to the gateway and telling it that it has received the request and is processing it.

Next, the Mediation Server sends back a “183 Session Progress” message. This is used to start the early media process. The basic idea of early media is to allow endpoints to exchange media (RTP) packets before the SIP handshake to establish the call is completed. This helps the media get to the user faster so the call feels more natural by removing “dead air” after answering the call.

A quick peek at the body shows which media Skype for Business has decided it wants to use for this call.

Figure 12 – 25

```
v=0
o=- 20 1 IN IP4 192.168.1.171
s=session
c=IN IP4 192.168.1.171
b=CT:1000
t=0 0
m=audio 55994 RTP/SAVP 8 13 101
c=IN IP4 192.168.1.171
a=rtcp:55995
a=label:Audio
a=crypto:1 AES_CM_128_HMAC_SHA1_80 inline:yh17bRuU4YokfWev6chP+cbGhCgTjVmxFtmjWy5H|2^31|1:1
a=sendrecv
a=rtpmap:8 PCMA/8000
a=rtpmap:13 CN/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
```

Look at the "m" entry. Skype for Business has chosen to use G.711 A-Law for this call and will also support comfort noise and DTMF tones. G.711 µ-law has been dropped.

In Figure 12-22 we next see that the gateway responds with a Provisional Acknowledgement (PRACK). There isn't anything to special with this. It is simply acknowledging the 183 Session Progress that was sent from the Mediation Server. It also means that the gateway is good to go with the media selected by the Mediation server.

Upon receipt of the PRACK, Skype for Business kicks back a "200 OK" that basically means that Skype for Business is ready for some media.

Now I'm going to point out a log entry that isn't in the ladder diagram. Remember that the telephone company is only sending the last four digits (1009) and I need to convert it to E.164. This is done via a Site dial plan.

In the log file is a "101 Progress Report" entry. Look at this snippet from that log entry:

Figure 12 – 26

```
ms-diagnostics: 14011;reason="Called Number
translated";source="SKYPE4B-1.FLINCHBOT.COM";RuleName="4-Digit
Extension";CalledNumber="1009";TranslatedNumber="+16155551009";appName="TranslationService"
```

We can clearly see that 1009 hit the "4-digit Extension" rule in the Site dial plan and translated the number to +16155551009. I bet that Reverse Number Lookup will now be able to find the user with that phone number.

Back to the ladder diagram (Figure 12-22). The Mediation Server is sending back another "183 Session Progress" to the gateway. This is because Skype for Business is now getting serious. It has a translated number. Now it needs to see if the user is online and available to take a call.

Looking back in the log I can see that there is another INVITE message being sent. This log entry is really long and can get complicated, but let me highlight some of the key things.

Figure 12 – 27

REFERRED-BY: <sip:MTressler@flinchbot.com>;

This is the first occurrence of the SIP address of the target user. I can verify that the phone number +16155551009 is the LineURI value for the user mtressler@flinchbot.com. Reverse Number Lookup was indeed able to take the normalized phone number (+16155551009) and find the user to whom it is assigned (sip:mtressler@flinchbot.com).

By the way, this Invite is from the Mediation Server to one of the Front End servers. Remember SIP traffic does not go from the Mediation Server to the client. Rather it goes from the Mediation Server to the Front End server on which the user is homed.

Back to the ladder diagram (Figure 12-22). Skype for Business sends two ringing messages to the gateway. This tells the gateway that we have found a user and to play ring back to the caller. At this point, the PSTN caller will start hearing the ring tone coming through his handset.

After the 180 Ringing is sent, something else of interest happens in the log that is not seen in the ladder diagram. Skype for Business now starts negotiating media with the users Skype for Business client. Here is part the body of that message.

Figure 12 – 28

```
v=0
o=- 0 0 IN IP4 192.168.1.20
s=session
c=IN IP4 192.168.1.20
b=CT:99980
t=0 0
a=x-devicecaps:audio:send,recv;video:send,recv
m=audio 8814 RTP/SAVP 0 8 115 97 13 118 101
```

Look at the "o" entry. The 192.168.1.20 is the IP address of the PC running the Skype for Business client of mtressler@flinchbot.com.

What are the protocols offered in the "m" line?

0	G.711 PCMA (µ -Law)
8	G.711 PCMU (A-law)
115	RTAudio
97	Forward Error Correction
13	Comfort Noise
118	Comfort Noise
101	DTMF (Dial Tones)

Going back to the ladder diagram in Figure 12-22, note the timing on the right hand side. After this last 183 Session Progress, there is a 6 seconds gap until the 200 OK shows up. This is because the call was now going on. It was a short conversation between me and myself. I didn't have much to tell myself.

During the call, there isn't much to see in the log until you get to the 200 OK entry except for one specific entry. The entry showing me hanging up on myself.

There is a CANCEL entry in the log which means that the call was canceled. Looking at the detail of that log entry you can see the following:

Figure 12 – 29

```
Reason: SIP;cause=200;text="Call completed elsewhere";ms-acceptedby="sip:mtressler@flinchbot.com"
ms-diagnostics-public: 5025;reason="Cancel sent by application for INVITE client
transaction.";AppUri="http%3A%2F%2Fwww.microsoft.com%2FLCS%2FDefaultRouting"
```

This is me clicking the red hang-up icon in the Skype for Business client.

The 200 OK in the ladder diagram is the Mediation server warning the gateway that the call is about to end. We can see this because the ALLOW field in the 200 OK is pretty limited in what it wants to hear back from the gateway.

Figure 12 – 1

```
ALLOW: ACK
ALLOW: CANCEL,BYE,INVITE,PRACK,UPDATE
```

The gateway sends an ACK to acknowledge that it received the 200 OK. Then the Mediation Server send the BYE to end the call. Finally, the gateway acknowledges that it dropped the call.

Synthetic Transactions

Skype for Business contains some built-in test cases that you can manually run. These can also be run automatically and Systems Center Operations Manager(SCOM) takes advantage of these synthetic transaction to better monitor your Skype for Business environment.

As SCOM is out of focus for this book, I'll instead point out a few Synthetic Transactions that you can manually run.

The first one to be aware of is `Test-CsPstnOutboundCall`. This will automatically place an outbound call, wait until it is answered, and then send four DTMF tones (#777). It will then let you know if everything worked correctly or not.

Below is a sample run of this cmdlet. Note that you first need to provide a valid Enterprise Voice-enabled user account and the credentials of this account. This is so that the transaction can log in as that user. Because of this, the correct Dial Plan and Voice Policy for that user will be used.

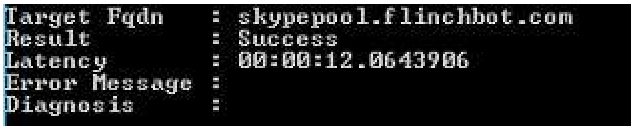
```
$cred1 = Get-Credential "flinchbot\mtressler"
```

After you are prompted to provide the password for the account, you then run the following command. You define the SIP Address of the user associated with the account you logged in with. Next you must provide the name of the pool you wish to test. Finally, you provide the phone number that you are going to call. Note that the phone call must get answered for the test to succeed.

```
Test-CsPstnOutboundCall –UserSipAddress
"sip:mtressler@flinchbot.com" -TargetFqdn
skypepool.flinchbot.com -TargetPstnPhoneNumber "+16135551234"
-UserCredential $cred1
```

After running this, I got a success. Below is the output of a successful call.

Figure 12 – 31



You can also get HTML-formatted output from this transaction. Add the *-OutLoggerVariable* to the end of the command along with a specified variable. Below is an example showing the previous command but with the additional parameter.

```
Test-CsPstnOutboundCall –UserSipAddress
"sip:mtressler@flinchbot.com" -TargetFqdn
skypepool.flinchbot.com -TargetPstnPhoneNumber "+16135551234"
-UserCredential $cred1 –OutLoggerVariable PSTNTest
```

Then you process the variable (PSTNTest) in PowerShell to spit out an HTML-formatted log file.

```
$PSTNTest.ToHTML() > C:\Logs\TestOutput.html
```

Now if you navigate to the C:\Logs directory, you will see the file. Below is a sample output from opening this file.

Figure 12 – 32

Synthetic Transactions Execution Result

Command Executed: Test-CsPstnOutboundCall -TargetFqdn skypepool.flinchbot.com -UserSipAddress sip:mtressler@flinchbot.com -TargetPstnPhoneNumber +16135551234 -UserCredential System.Management.Automation.PSCredential -OutLoggerVariable TestOutput

Activity	Result	Start Time	End Time	Duration (seconds)
▼STOcsPSTNWorkflow	Success	2015-11-01T20:03:57.4610194Z	2015-11-01T20:03:59.5759071Z	2.11
►InputParameters				
►Traces				
▼Query Authentication Mechanism and Register	Success	2015-11-01T20:03:57.4610194Z	2015-11-01T20:03:57.7048597Z	0.24
▼InputParameters				
Name	Value			
SupportDataCollaboration	False			
RegistrarPortNumber	5061			
TargetFqdn	skypepool.flinchbot.com			
User	sip:mtressler@flinchbot.com			
Authentication	Negotiate			
UserResources	NULL			
IsInternalAccess	False			
faultHandlerActivity1_Fault1	NULL			
►Traces				
►Register Endpoint 'sip:mtressler@flinchbot.com'	Success	2015-11-01T20:03:57.4818591Z	2015-11-01T20:03:57.7038593Z	0.22
►InviteOcsPSTNActivity	Success	2015-11-01T20:03:57.7048597Z	2015-11-01T20:03:59.3919031Z	1.69
▼Send DTMF Tones	Success	2015-11-01T20:03:59.3919031Z	2015-11-01T20:03:59.3969017Z	0.00
▼InputParameters				
Name	Value			
OutboundAudioVideoFlow	Microsoft.Rtc.Collaboration.AudioVideo.AudioVideoFlow			
►Traces				
►Unregister Endpoint	Success	2015-11-01T20:03:59.3969017Z	2015-11-01T20:03:59.5759071Z	0.18

Below is what happens if you let the phone ring endlessly and it finally gives up. I actually unplugged the phone to put a stop to the endless ringing because it really didn't want to give up easily. I almost got a headache!

Figure 12 – 33

```
Target Fqdn   : skypepool.flinchbot.com
Result        : Failure
Latency       : 00:00:00.2103607
Error Message : This operation has timed out.

Diagnosis     :
```

Summary

No one has ever implemented something as complex as Skype for Business Enterprise Voice without needing to do at least a little troubleshooting. I have a lab environment and even I had to troubleshoot things in order to make sure this book is accurate.

Using the tools listed in this chapter, you should be able to get a grasp on the issues you are facing. Those issues could be anything from a malformed regular expression to a codec mismatch that you could only determine by reviewing the Snooper logs.

Over time, you will become more comfortable with these tools and your success as a Skype for Business Enterprise Voice administrator will be greatly enhanced. For it could be argued (probably unsuccessfully) that the true measure of a network engineer is not in how well she designs or implements a solution but in how quickly she can fix it when the inevitable issues arise.

Chapter 13 – Putting It All Together

This final chapter is designed to walk you through a complete and fairly complex roll out of Skype for Business Enterprise Voice. You will be walked through enabling and configuring many of the concepts discussed in this book – from adding PSTN Gateways to configuring Least Cost Routing.

You can use this chapter in several ways

- Use it as a recap of all that you have learned
- Use it as a reference design when rolling out your own Enterprise Voice deployment
- Quiz yourself before reading each section to see if you recall the information

The first part of this chapter describes the requirements of our fictional Enterprise Voice roll out. Thereafter, each section will focus on a specific piece of the rollout, such as adding gateways to Topology or configuring Common Area Phones

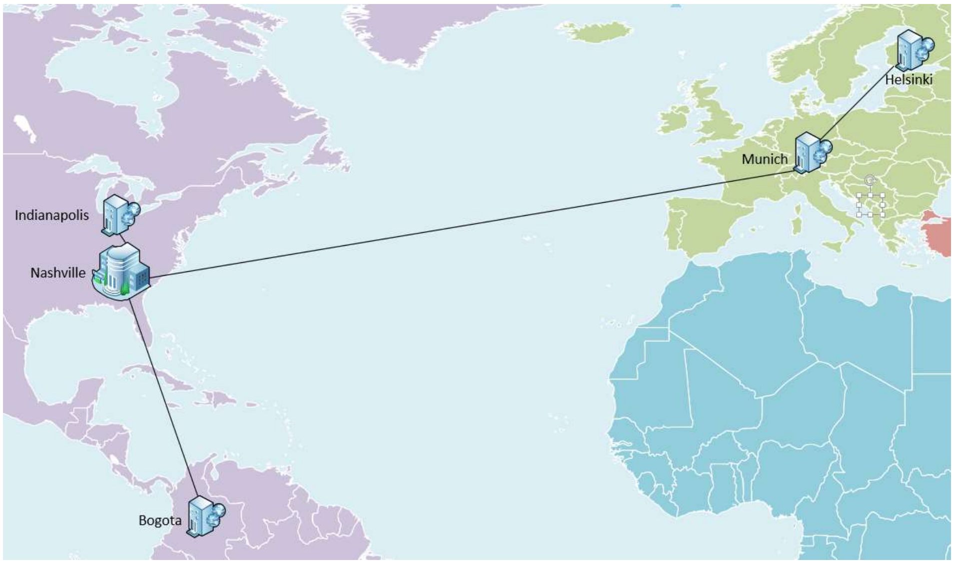
This is an ambitious chapter but hopefully you will get a lot of value out of this.

Scenario

You have just been assigned the task of rolling out Skype for Business Enterprise Voice at Flinchbot, Inc. an international leader in the exciting field of manufacturing the world's best mouse wheels. Just about every computer mouse has a wheel on it and we make the best ones ever made. Because of our success we have operations located around the world.

Here is the obligatory Visio diagram of our global network.

Figure 13 – 1



Our corporate headquarters are located in Nashville, Tennessee in the United States. This is where our executives work along with human resources, research and development, sales, and tech support. We have 3 manufacturing plants located in Indianapolis, Indiana, USA; Bogota, Colombia; and Helsinki, Finland. Our European operations are based out of Munich, Germany where we have more sales, tech support, and research and development departments.

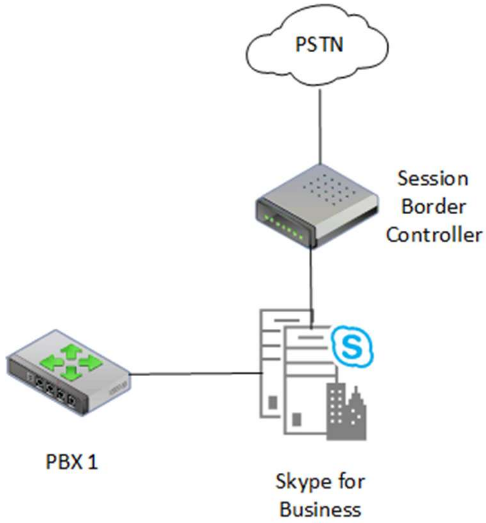
There are 2 Sites in our Topology. One is named Nashville and the other is named Munich. Within the Nashville site will be two Branch Sites – one for Bogota and the other for Indianapolis. The Munich site will have a branch site for Helsinki.

Location	Server Edition	Site
Bogota	Survivable Branch Appliance	Nashville
Helsinki	Survivable Branch Appliance	Munich
Indianapolis	Survivable Branch Appliance	Nashville
Munich	Standard Edition Server	Munich
Nashville	Enterprise Edition	Nashville

We have an Enterprise Edition of Skype for Business located in our headquarters in Nashville. In Munich we have a Standard Edition server installed. In Bogota, Helsinki, and Indianapolis we have Survivable Branch Appliances. The SBA's in Bogota and Indianapolis are connected to the Skype for Business pool in Nashville while the SBA in Helsinki is attached to the Standard Edition server in Munich.

Within the Headquarters location in Nashville we have a legacy PBX that we want to replace with Skype for Business Enterprise Voice. We would like to have the following configuration in order to begin our migration from the PBX to Skype for Business Server.

Figure 13 – 2



We want to place our Skype for Business pool between the PSTN and the PBX then use the Skype for Business pool as a router to pass calls between the PBX and the PSTN. Eventually as we decommission the PBX we won't have to do any additional architecting – we just remove the routes to that PBX and we're done with that decommission.

We also need to have two voice policies for each location –one which permits national calling (including local calling), and one which permits international calling (including local and national calling).

We also want Least Cost Routing for the Nashville location. Users in Nashville should be configured so if they call any number in Colombia, Germany, or Finland, that call should go out the gateway already in each country.

Call Park is required in the Munich office. The Call Park Orbit should be the numbers *450-*460. The Music on Hold for call park should be a custom recording.

We also want an Unassigned Number range in Munich. There should be a text-to-speech announcement played. After the announcement is played the call should be forwarded to the Operator in Nashville (+16155551000).

Dial In Conferencing should be a real cost saver for us. So let's enable dial in numbers for Indianapolis, Nashville, and Munich.

Adding the SBA's

We already have our Skype for Business pools up and running in Nashville and Munich. We need to add the three SBA's in Bogota, Helsinki, and Indianapolis. In order to do this, we need to open Topology Builder to add them.

Creating Sites

We will first add the SBA Branch Sites for Bogota and Indianapolis. To do this I open Topology Builder and navigate to Nashville and right click on "Branch Sites". This brings up a menu and I select "New branch site…". I then fill in the answers and click Next on each page of the wizard.

Figure 13 – 3

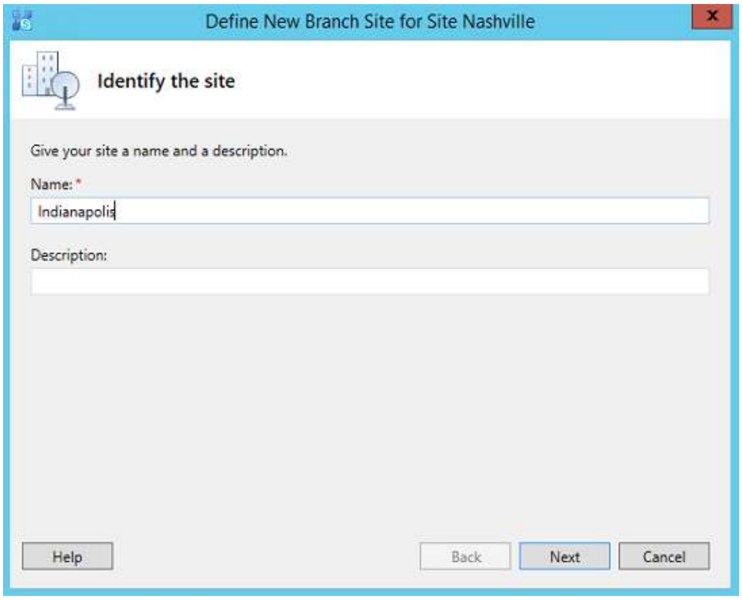
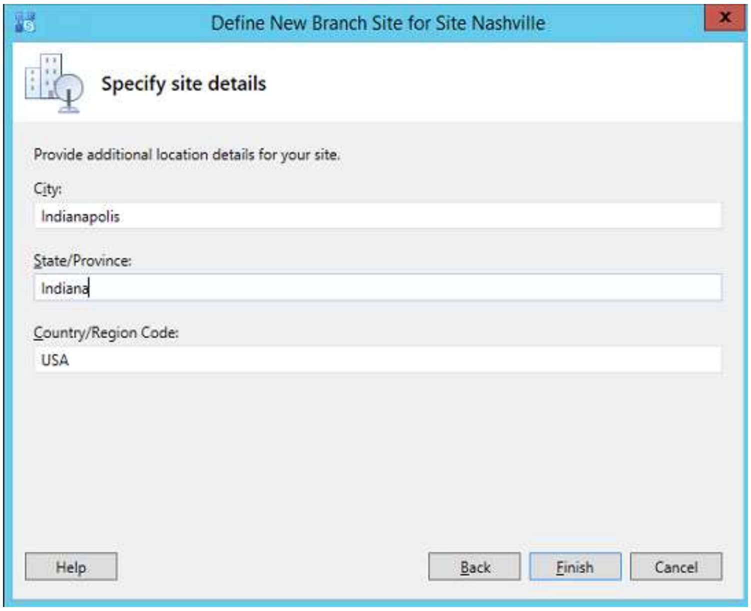
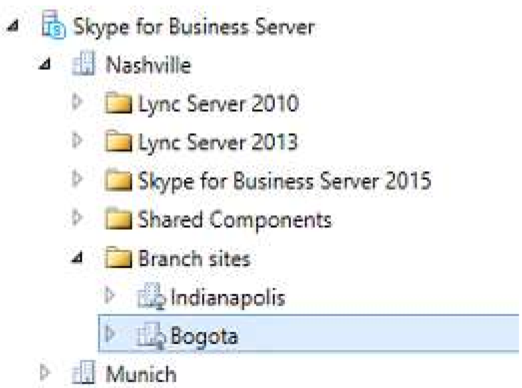


Figure 13 – 4



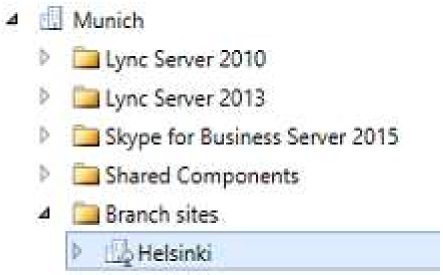
I then follow the same steps to create the Branch site for Bogota. When I get done I now have the branch sites seen in Figure 13-5.

Figure 13 – 5



Following the same steps, I proceed to add the Branch Site for Helsinki underneath the parent site Munich.

Figure 13 – 6



Creating SBA's and Gateways

Now that the Branch Sites have been created I can add the Survivable Branch Appliances for each location. I do this by expanding the Indianapolis Site then right-clicking on the "Skype for Business Server 2015" folder. This brings up a menu and I select "New" and then "Survivable Branch Appliance".

Figure 13 – 7

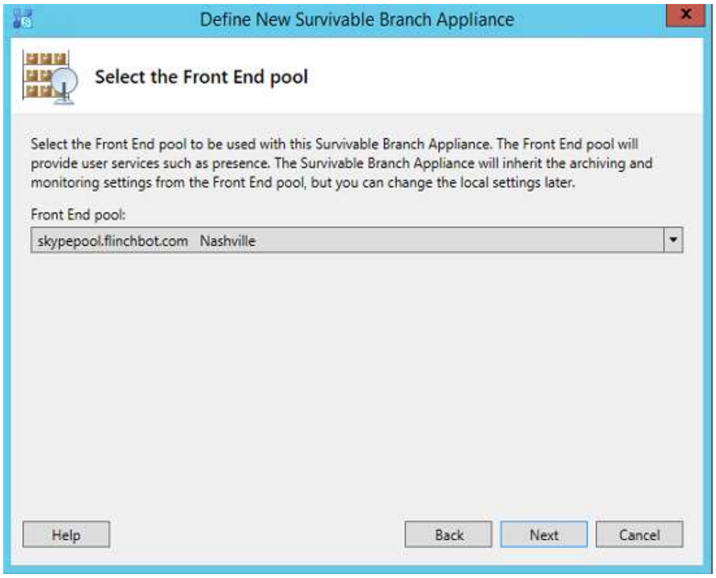


Figure 13 – 8

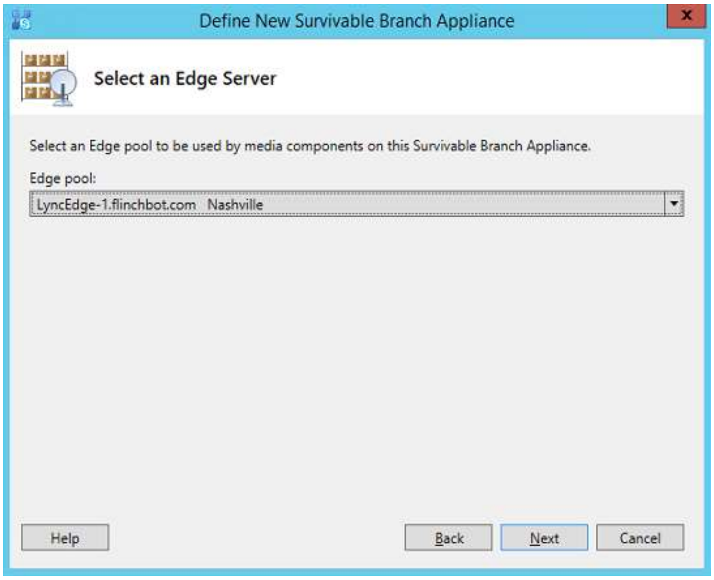
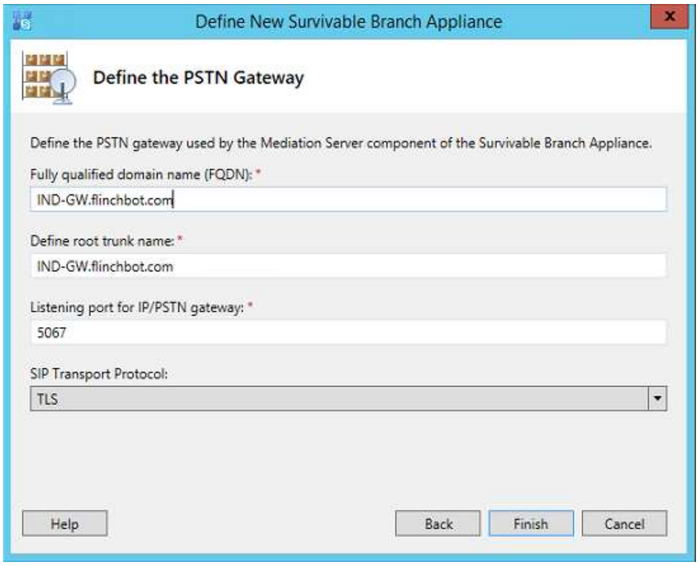


Figure 13 – 9

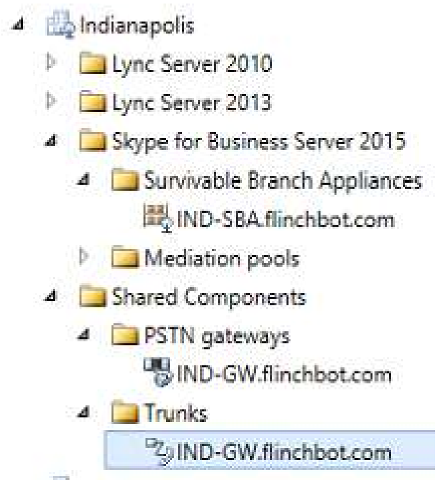


I should say a few words about Figure 13-9. How do you know the listening port and transport protocol for your gateway?

In general, you can use the defaults listed. However, the real answer is that it depends solely on how your gateway is configured. If your gateway is configured to listen on port 5060 using TCP, then you'll have to set those values here instead of using the defaults. In most cases you should be working with the vendor to assist you in installing the gateway.

And I *highly* recommend purchasing installation services from your gateway vendor or reseller for your first few gateways. The PSTN gateways are a whole different world with their own vocabulary and configuration steps. You can save a lot of time by working with the vendor or with your reseller.

Figure 13 – 10



After finishing the SBA wizard, you will see that the SBA and the gateway have been added to Topology. Note that a Trunk is also automatically created for you.

Now that the first SBA has been added I'll proceed to add the SBA's for Bogota and Helsinki.

Figure 13 – 11

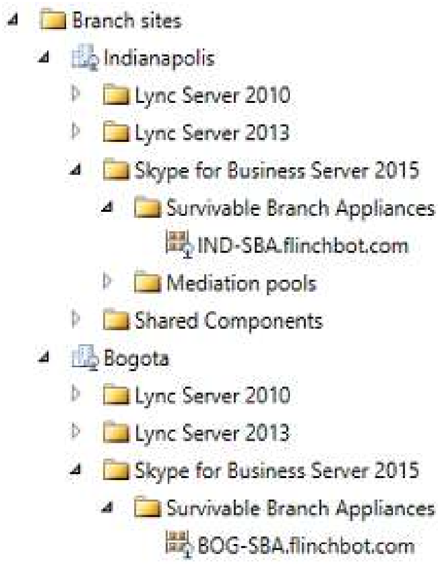
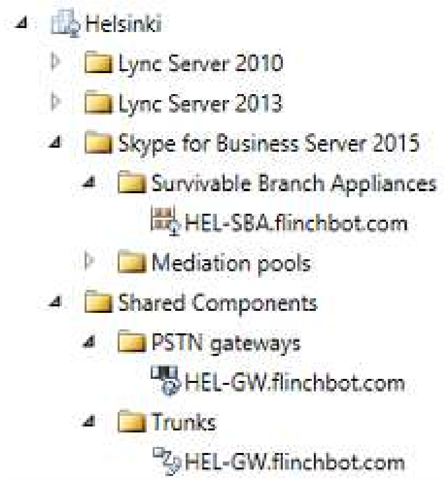


Figure 13 – 12



Now before you publish the Topology, SBA's have a quirk that you need to be aware of. You need to manually create the Active Directory computer object for the SBA before you can publish the Topology. This is because Topology Builder will add some Active Directory attributes to the SBA's computer object. But if that Computer Object doesn't exist, Topology publishing will fail.

Active Directory Permissions

So now I skip over to the Active Directory Users and Computers tool and add the Computer Objects to an OU where I have Full Control security rights. The Full Control for that OU should be delegated to the RTCUniversalServerAdmins group.

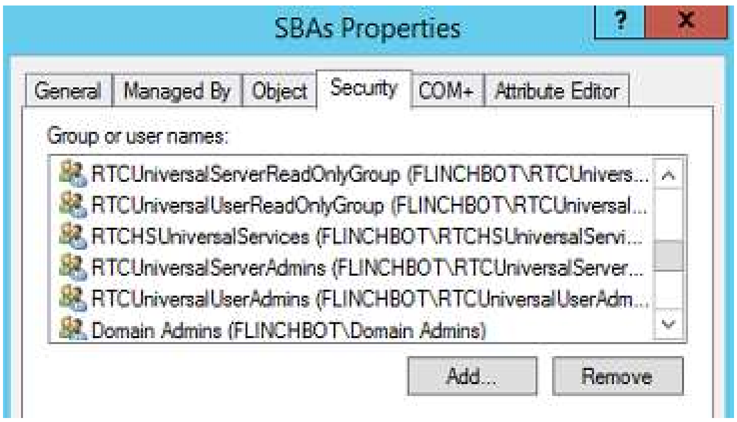
There are two PowerShell cmdlets that can configure these permissions for you. You only really need to do this if you are the Skype for Business administrator but do not have Domain or Enterprise Admin rights. In that case, you will need to get a Domain or Enterprise Admin to run the `Grant-CsSetupPermission` and `Grant-CsOUPermission` cmdlets for you.

In our case, I have created an OU named "SBAs". Let's assume I am not a Domain or Enterprise Admin. As such, one of my domain or enterprise admins will need to run the following two cmdlet's from the Skype for Business Management Shell:

```
Grant-CsSetupPermission -
ComputerOU ‘OU=SBAs,DC=flinchbot,DC=com’

Grant-CsOUPermission -OU ‘OU=SBAs,DC=flinchbot,DC=com’
ObjectType- ‘user’
```

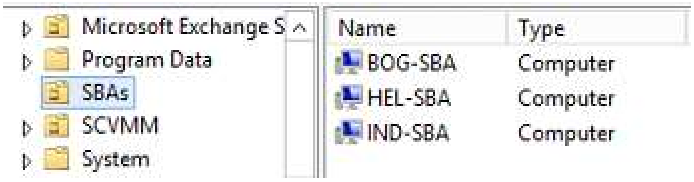
After running those 2 cmdlets you can look into the Security Tab of the OU properties to see that a bunch of new permissions have been added.

Figure 13 – 13



Now I will add 3 computer objects to this OU – one for each of the three SBA's I've just created in Topology Builder.

Figure 13 – 14



Service Principal Names

Note: If you are building an SBS, you can skip this step as you can join the computer to the domain before running Topology builder.

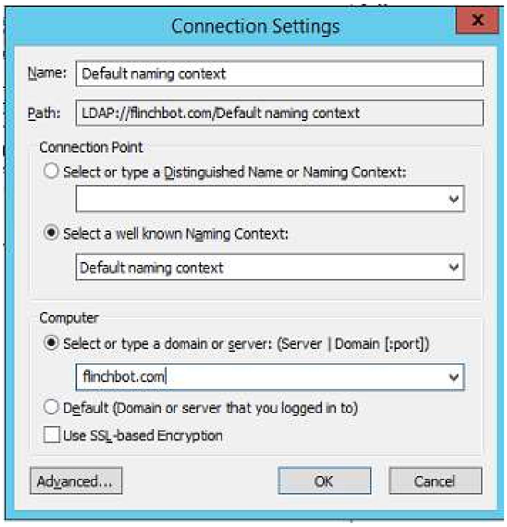
We can't add an SBA to Active Directory the way you would any regular server. You are supposed to use the tools provided by the SBA vendor which involves navigating a website with built-in wizards. As such, the full computer object won't be in Active Directory yet when you publish Topology. Because of this, you need to do the below steps to help Topology builder find the computer object you manually created.

Topology Builder locates these Computer Objects by searching Active Directory for a specific Service Principal Name (SPN). If the SPN's don't exist, Topology Builder will fail to find the computer objects we just created and Topology Builder will fail to publish.

For this step I'm going to use ADSIEdit but you can also do this from the Attributes tab in Active Directory. I'm just using ADSIEdit to demonstrate using a different tool.

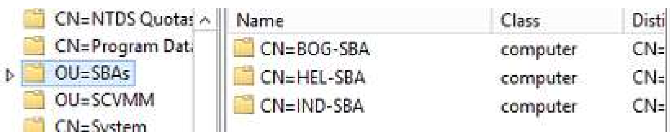
Start ADSIEdit. If it did not automatically connect me to my domain, I will have to configure ADSIEdit manually to connect to the flinchbot.com domain. To do this, right click on ADSI Edit at the root of the left pane and click "Connect to…". From here, click the radio button before the word Computer and type in the name of your domain and then click OK.

Figure 13 – 15



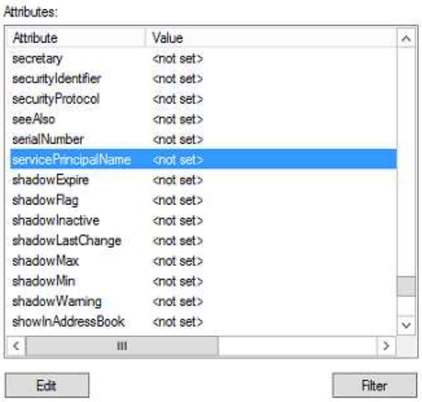
Double click on "Default Naming Context and then double click on "DC=Flinchbot, DC=com". From here you will see a view that mirrors what you see in Active Directory Users and Computers. Clicking on the SBAs OU shows the three computer objects on the right.

Figure 13 – 16



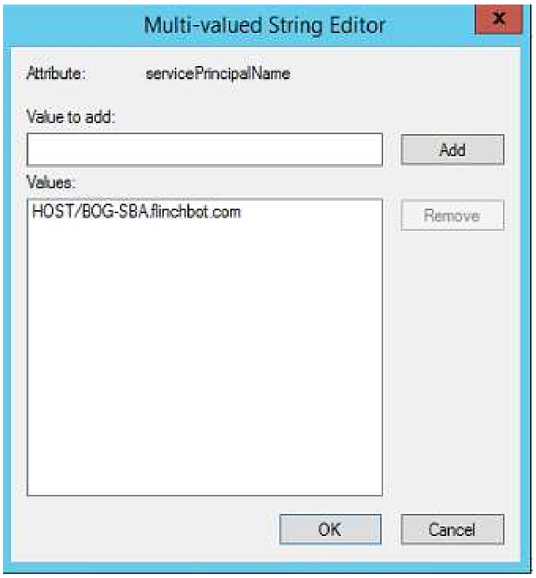
Now I need to right click on the first server (CN=BOG-SBA) and click "Properties" from the list. Next I scroll all the way down the list until I get to "Service Principal Name" and then click the Edit button.

Figure 13 – 17



This brings up the "Multi-valued String Editor" window. From here you just need to add the term "Host/<fqdn>" to the list. In my case, the FQDN of the Bogota SBA is BOG-SBA.flinchbot.com. So I type "Host/BOG-SBA.flinchbot.com" and click the Add button. Once added I can then click OK.

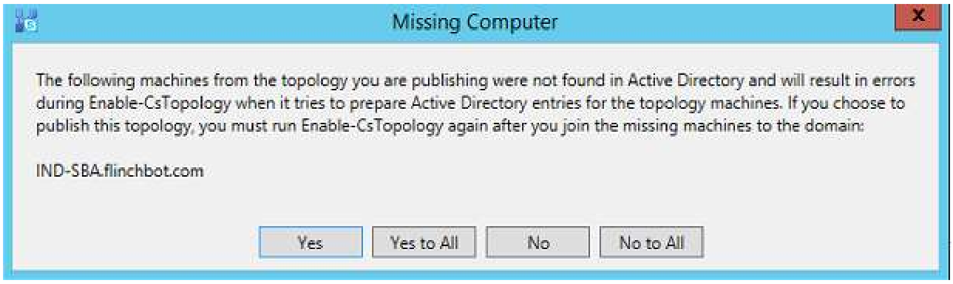
Figure 13 – 18



Click OK one more time to commit the change. I then need to repeat these steps for the other two SBA's. Wait a few minutes for Active Directory to replicate this change to all of your Domain Controllers. Once you are confident that these changes have been replicated by AD, return to Topology Builder and publish your changes.

If you don't wait long enough, you will see the following message:

Figure 13 – 19



At this point you may as well click "No to All" and wait a while longer for Active Directory to replicate the changes. You may also want to make sure there are no typos in the SPN's that you added.

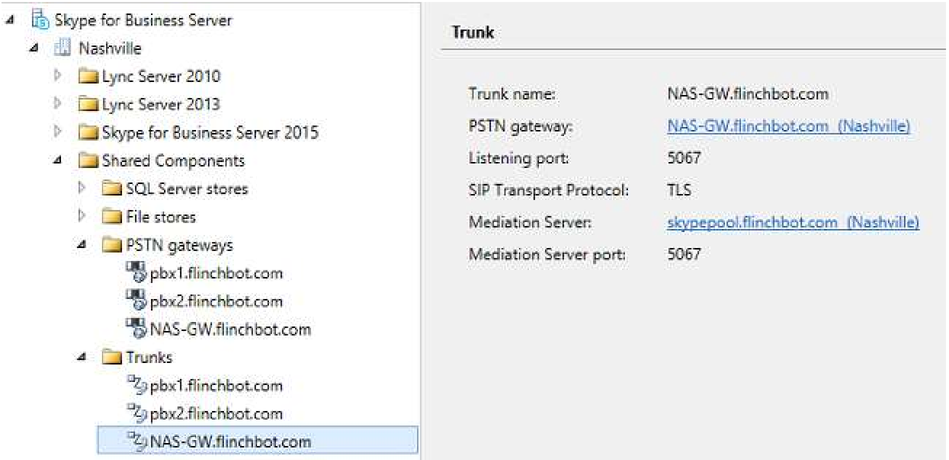
After the SBA has been added to Topology successfully, you can then proceed to follow the vendor's instructions to complete the installation of the SBA which includes joining it to the domain and installing the Skype for Business components.

For this example environment, I will repeat these steps 2 more times in order to add the SBA's for Indianapolis and Helsinki.

Adding a Gateway

You can add gateways that are not part of an SBA. This is a straightforward operation that was discussed in detail in chapter 2. As such, I won't go into the step-by-step detail here. However, below is a screenshot of Topology Builder showing a gateway and trunk that were added to support the Enterprise Edition pool in Nashville.

Figure 13 – 20



With all of the SBA's and gateways defined in Topology, it's time to start configuring Enterprise Voice!

Creating Dial Plans

The first thing we have to do when creating a Dial Plan is to figure out our end user behavior. Are they used to dialing a "9" before placing a PSTN call? Do they want to dial extensions? Do they have to add a code before dialing a long distance number?

For our example scenario we're going to keep it fairly simple and straightforward. Below are tables for each location which provide the high-level settings for our dial plans. I am setting it up so a user can dial an in-office operator by dialing "0". Dialing 4-digit extensions is also supported as is making national and International calls.

The dial plans do not support 3-digit emergency calling nor do they block calls to toll numbers or do alternate routing for mobile numbers.

Note that these are just example Dial Plans and probably won't work for the countries used in this example. In the real world, you would work with either your current PBX vendor to translate the existing rules on the PBX to Skype for Business or work with your telco to find out the dial plans.

Alternately, you can use the website www.lyncoptimizer.com which will create the correct PowerShell commands for every country in the world. Every single one. Except North Korea. And Kosovo.

Nashville	User Dials	Translation
Operator	0	+16155551000
Internal Extension	4 digits	+1615555<4digits>
Local calls	7 digits	+1615<7 digits>
National Calls	A number in North America	+1<dialed number>
International Calls	Any number starting with 011	+<digits dialed by user (without 011)>

Indianapolis	User Dials	Translation
Operator	0	+13175552000
Internal Extension	4 digits	+1317555<4digits>
Local calls	7 digits	+1317<7 digits>
National Calls	A number in North America	+1<dialed number>
International Calls	Any number starting with 011	+<digits dialed by user (without 011)>

Bogota	User Dials	Translation
Operator	0	+5715553000
Internal Extension	4 digits	+571555<4digits>
Local calls	1 followed by 7 digits	+57<7 digits>
National Calls	A number in Colombia	+57<dialed number>
International Calls	Any number starting with 00	+<digits dialed by user (without 00)>

Munich	User Dials	Translation
Operator	0	+49895554000
Internal Extension	4 digits	+4989555<4digits>
Local calls	89 followed by 7 digits	+4989<7 digits>
National Calls	A number in Germany	+49<dialed number>
International Calls	Any number starting with 00	+<digits dialed by user (without 00)>

Helsinki	User Dials	Translation
Operator	0	+35895555000
Internal Extension	4 digits	+3589555<4digits>
Local calls	9 followed by 7 digits	+3589<7 digits>
National Calls	A number in Finland	+358<dialed number>
International Calls	Any number starting with 00	+<digits dialed by user (without 00)>

Now that we know the Dial Plan, we can start implementing it. The first step is to figure out the regular expressions necessary to create the dial plan normalizations. You can either figure these out on your own, steal samples off the Internet, or look to www.lyncoptimizer.com for assistance.

For this example, I am going to create user-level Dial Plans because when I add the Dial Plans to my users it will be easiest to explain in this book. It is perfectly fine if you want to use Pool or Site-level Dial Plans instead. Just don't ever use the Global Dial Plan. Like never ever never. If at some point you think you have to use the Global dial, then you're probably wrong.

Creating a Dial Plan was discussed in-depth in chapter 3 so I won't go into detail here. However, I will repeat the tables above but this time showing the regular expressions that will be used within each normalization in each dial plan.

Nashville	Pattern to Match	Translation
Operator	^(0)$	+16155551000
Internal Extension	^(\d{4})$	+1615555$1
Local calls	^(\d{7})$	+1615$1
National Calls	^1?([2-9]\d\d[2-9]\d{6})\d*(\D+\d+)?$	+1$1
International Calls	^(?:011)([2-9]\d{6,14})(\D+\d+)?$	+$1

Indianapolis	Pattern to Match	Translation
Operator	^(0)$	+13175552000
Internal Extension	^(\d{4})$	+1317555$1
Local calls	^(\d{7})$	+1317$1
National Calls	^1?([2-9]\d\d[2-9]\d{6})\d*(\D+\d+)?$	+1$1
International Calls	^(?:011)([2-9]\d{6,14})(\D+\d+)?$	+$1

Bogota	Pattern to Match	Translation
Operator	^(0)$	+5715553000
Internal Extension	^(\d{4})$	+571555$1
Local calls	^(1\d{7})$	+57$1
National Calls	^\+57[124-8][2-8]\d{6}$	+57$1
International Calls	^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-9]\d{6,14}))(\D+\d+)?$	+$1

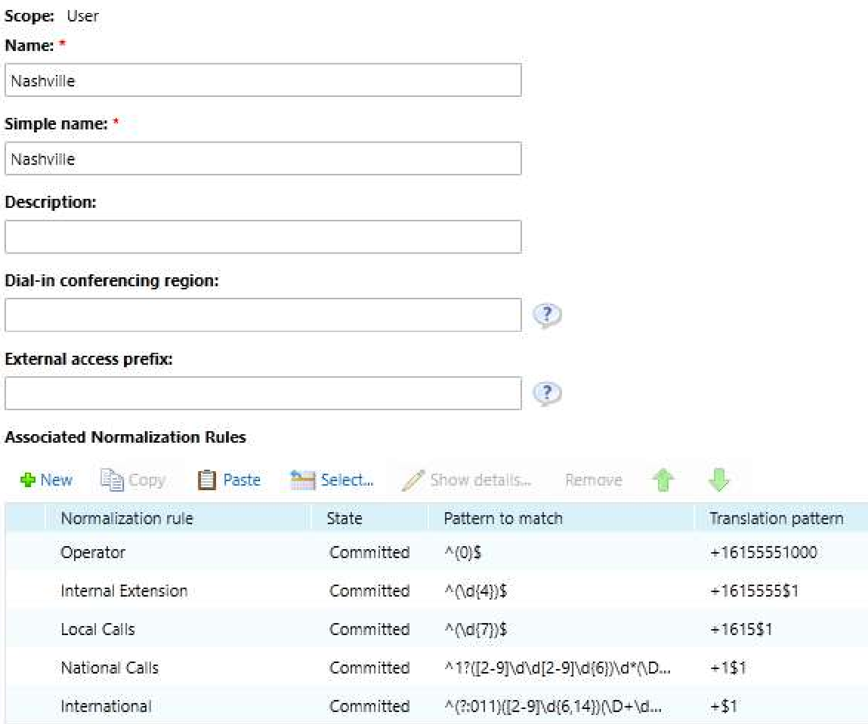
Munich	Pattern to Match	Translation
Operator	^(0)$	+49895554000
Internal Extension	^(\d{4})$	+4989555$1
Local calls	^(89\d{7})$	+4989$1
National Calls	^0((180\d{5,7}|[2-7]\d{5,}|[89][1-9]\d{5,10}|90[1-9]\d{4,8}))\d*(\D+\d+)?$	+49$1
International Calls	^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-9]\d{6,14}))(\D+\d+)?$	+$1

Helsinki	Pattern to Match	Translation
Operator	0	+35895555000
Internal Extension	^(\d{4})$	+3589555$1
Local calls	^(9\d{7})$	+3589$1
National Calls	^0(((1[3-9]|[235689])[1-8]\d{3,9}|[123]0[^0]\d{2,7}|7([13]\d{7,8}|5[3-9]\d{2,7})))\d*(\D+\d+)?$	+358$1
International Calls	^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-9]\d{6,14}))(\D+\d+)?$	+$1

Those regular expressions for National and International sure look complicated. How did I come up with those? I didn't! I swiped them from www.lyncoptimizer.com. They reflect the exact rules that define a National call for a given country and an International call globally.

Below is a screen shot of the Nashville dial plan.

Figure 13 – 21



While using Control Panel is a fine way to add these dial plans and normalizations it might be faster to use PowerShell to load up the rest of these dial plans. As such, below is the PowerShell required to create the rest of the Dial Plans and Normalizations.

```
New-CsDialPlan -Identity 'Indianapolis'
New-CsVoiceNormalizationRule -Name 'Operator' -Parent
'Indianapolis' -Pattern '^(0)$' -Translation '+13175552000'
```

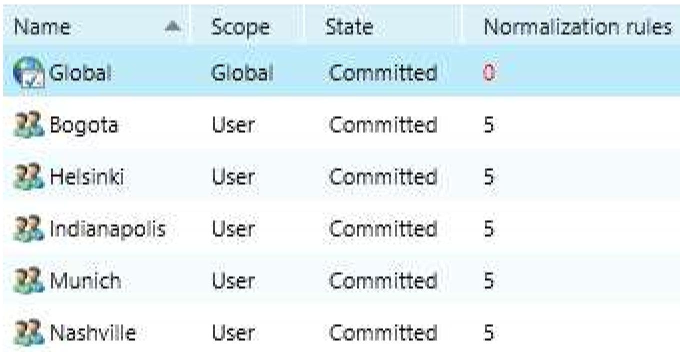
```
New-CsVoiceNormalizationRule -Name 'Internal
Extension' -Parent 'Indianapolis' -Pattern '^(\d{4})$' -
Translation '+1317555$1'
New-CsVoiceNormalizationRule -Name 'Local Calls' -Parent
'Indianapolis' -Pattern '^(\d{7})$' -Translation '+1317$1'
New-CsVoiceNormalizationRule -Name 'National Calls' -Parent
'Indianapolis' -Pattern '^1?([2-9]\d\d[2-
9]\d{6})\d*(\D+\d+)?$' -Translation '+1$1'
New-CsVoiceNormalizationRule -Name 'International' -Parent
'Indianapolis' -Pattern '^(?:011)([2-
9]\d{6,14})(\D+\d+)?$' -Translation '+$1'
Remove-CsVoiceNormalizationRule -Identity 'Indianapolis/Keep
All'

New-CsDialPlan -Identity 'Bogota'
New-CsVoiceNormalizationRule -Name 'Operator' -Parent 'Bogota'
-Pattern '^(0)$' -Translation '+5715553000'
New-CsVoiceNormalizationRule -Name 'Internal
Extension' -Parent 'Bogota' -Pattern '^(\d{4})$' -Translation
'+571555$1'
New-CsVoiceNormalizationRule -Name 'Local Calls' -Parent
'Bogota' -Pattern '^(1\d{7})$' -Translation '+57$1'
New-CsVoiceNormalizationRule -Name 'National Calls' -Parent
'Bogota' -Pattern '^\+57[124-8][2-8]\d{6}$' -Translation
'+57$1'
New-CsVoiceNormalizationRule -Name 'International' -Parent
'Bogota' -Pattern '^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-
9]\d{6,14}))(\D+\d+)?$' -Translation '+$1'
Remove-CsVoiceNormalizationRule -Identity 'Bogota/Keep All'

New-CsDialPlan -Identity 'Munich'
New-CsVoiceNormalizationRule -Name 'Operator' -Parent 'Munich'
-Pattern '^(0)$' -Translation '+49895554000'
New-CsVoiceNormalizationRule -Name 'Internal
Extension' -Parent 'Munich' -Pattern '^(\d{4})$' -Translation
'+4989555$1'
New-CsVoiceNormalizationRule -Name 'Local Calls' -Parent
'Munich' -Pattern '^(89\d{7})$' -Translation '+4989$1'
New-CsVoiceNormalizationRule -Name 'National Calls' -Parent
'Munich' -Pattern '^0((180\d{5,7}|[2-7]\d{5,}|[89][1-
9]\d{5,10}|90[1-9]\d{4,8}))\d*(\D+\d+)?$' -Translation '+49$1'
New-CsVoiceNormalizationRule -Name 'International' -Parent
'Munich' -Pattern '^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-
9]\d{6,14}))(\D+\d+)?$' -Translation '+$1'
Remove-CsVoiceNormalizationRule -Identity 'Munich/Keep All'
```

```
New-CsDialPlan -Identity 'Helsinki'
New-CsVoiceNormalizationRule -Name 'Operator' -Parent
'Helsinki' -Pattern '^(0)$' -Translation '+35895555000'
New-CsVoiceNormalizationRule -Name 'Internal
Extension' -Parent 'Helsinki' -Pattern '^(\d{4})$' -
Translation '+3589555$1'
New-CsVoiceNormalizationRule -Name 'Local Calls' -Parent
'Helsinki' -Pattern '^(9\d{7})$' -Translation '+3589$1'
New-CsVoiceNormalizationRule -Name 'National Calls' -Parent
'Helsinki' -Pattern '^0(((1[3-9]|[235689])[1-
8]\d{3,9}|[123]0[^0]\d{2,7}|7([13]\d{7,8}|5[3-
9]\d{2,7})))\d*(\D+\d+)?$' -Translation '+358$1'
New-CsVoiceNormalizationRule -Name 'International' -Parent
'Helsinki' -Pattern '^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-
9]\d{6,14}))(\D+\d+)?$' -Translation '+$1'
Remove-CsVoiceNormalizationRule -Identity 'Helsinki/Keep All'
```

Notice the `Remove-CsVoiceNormalizationRule` cmdlet. When you create a new dial plan, a default normalization named "Keep All" is automatically created. We don't want that normalization so we remove it from each of our dial plans.

Now if I refresh Control Panel, I see the following beautiful entries.

Figure 13 – 22



Name	Scope	State	Normalization rules
Global	Global	Committed	0
Bogota	User	Committed	5
Helsinki	User	Committed	5
Indianapolis	User	Committed	5
Munich	User	Committed	5
Nashville	User	Committed	5

Routes

Based on this scenario, we need to create two voice policies in each location – one which permits only National dialing and one which permits all dialing. This will be the same as National but with International dialing added.

If you remember back to the Voice Policy and Routes chapters, it is a Route that controls which PSTN calls a user is allowed to make. If the Route assigned to the users Voice Policy permits the call, then the call gets placed. This is controlled by a regular expression on the assigned Route. As such, I will first create the necessary Routes. After that I'll create the Voice Policy and the PSTN Usage to connect the two together.

Note that you can create the Voice Policy first. The Control Panel more or less walks you through from creating a Voice Policy to creating a PSTN Usage and then creating the Route. That is a perfectly acceptable way to do it and odds are you will probably do most of your work this way. However, I am doing it differently primarily to show you that there are multiple ways to accomplish the same goals.

Just like with the Dial Plans, I am going to build a chart showing which Routes I need to create for each location. A user allowed to make a national call also gets to call the Operator, 4 digit extensions, local calls, and national calls.

We only need to create routes for local calls, national, and international. Why not the other two? Because calls to the operator and to a coworker's 4-digit extension will never be routed to the PSTN. In the Dial Plan we are normalizing those calls to be proper E.164 phone numbers. When that E.164 call is submitted to Skype for Business, a Reverse Number Lookup will occur. Since there should be a match for the Operator or for the coworker's LineURI (or private number) that call will be routed as a peer-to-peer call.

So we can skip those two.

Which leaves us three routes to create for each location. And when you look at the regular expressions that will be assigned to these routes, they will be different than what was done in the Dial Plans.

By the time the call gets to the route to be evaluated, the call will be in E.164 format. So we can't just say "if there are 7 digits it's a local number" because we will never see just 7 digits. Referring back to the dial plan for Nashville, if the user dials 7 digits, we convert it to +1615<7 digits>. This means that the route will see +1615<7 digits> and so we will need some different regular expressions to handle this.

Below are the tables for each location and the Regular expressions used for each route.

Nashville	Match this pattern
Local calls	+1615
National Calls	^\+1[2-9]\d\d[2-9]\d{6}$
International Calls	^\+[2-9]\d{6,14}$

Indianapolis	Match this pattern
Local calls	+1317
National Calls	^\+1[2-9]\d\d[2-9]\d{6}$
International Calls	^\+[2-9]\d{6,14}$

Bogota	Match this pattern
Local calls	\+571
National Calls	^\+57[124-8][2-8]\d{6}$
International Calls	\+((1[2-9]\d\d[2-9]\d{6})|(?(?!(57))([2-9]\d{6,14})))$

Munich	Match this pattern
Local calls	^\+4989
National Calls	^\+49(180\d{5,7}|[2-7]\d{5,}|[89][1-9]\d{5,10}|90[1-9]\d{4,8})$
International Calls	^\+((1[2-9]\d\d[2-9]\d{6})|(?(?!(49))([2-9]\d{6,14})))$

Helsinki	Match this pattern
Local calls	^\+3589
National Calls	^\+358((1[3-9]|[235689])[1-8]\d{3,9}|[123]0[^0]\d{2,7}|7([13]\d{7,8}|5[3-9]\d{2,7}))$
International Calls	^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-9]\d{6,14}))(\D+\d+)?$

I will create the routes for the headquarters in Nashville using Control Panel. I will then create the other routes using PowerShell.

1. I name the first route “Nashville-Local”.

2. Next I click in the “Starting digits for numbers that you want to allow” and I simply enter ”+1615” as any E.164 normalized call within the city of Nashville will begin with ”+1615”. I could create a regular expression here to make sure that only 7 digits follow the ”+1615” but I don’t want to complicate things here.

   Now if you’re clever, you may think “what about someone who has a four-digit extension of 1615”? That call won’t get here as we take any four digit extension and normalize it to a full number such as +16155551615;ext=1615. Skype for Business will then do the Reverse Number Lookup and route that call as a peer-to-peer call.

3. I want the caller ID for all callers to be the main line for the Operator. So in here I will enter the operators full phone number for Nashville.

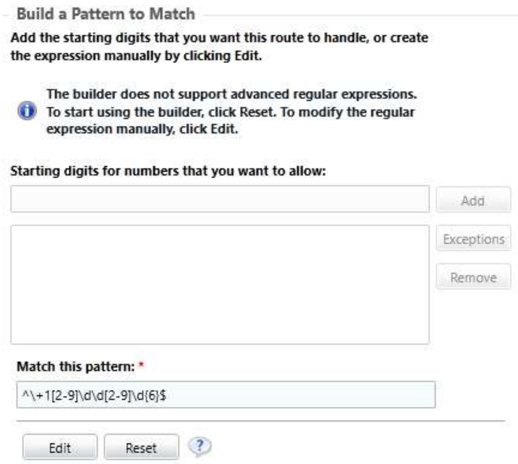
4. Finally, I assign one of the Trunks. Since this is the Nashville office, I will assign the “NAS-GW.flinchbot.com” trunk.

If you are following along, you’ll notice that there is no entry in the “Assigned PSTN Usages” section at the very bottom of the Routes screen. This is OK for now since we have not yet created a PSTN Usage.

Figure 13 – 23

The National site is created the same way, except we have to use a more complicated way to analyze if the dial string is valid for the national route. As such, step two will be slightly different. This time I click on the Edit button under "Match this pattern" and enter the regular expression.

Figure 13 – 24



Otherwise everything else is created the same way for the National and International routes.

After creating the National and International Routes, the Routes screen now looks like this.

Figure 13 – 25

Name	State	PSTN usage	Pattern to match
Nashville-Local	Committed		^\+1615
Nashville-National	Committed		^\+1[2-9]\d\d[2-9]\d{6}$
Nashville-International	Committed		^\+[2-9]\d{6,14}$

Below is the PowerShell needed to create the routes for the remaining locations.

```
New-CsVoiceRoute -Name 'Indianapolis-Local' -PstnGatewayList
@{add='IND-GW.flinchbot.com'} -NumberPattern '^\+1317'
New-CsVoiceRoute -Name 'Indianapolis-
National' -PstnGatewayList @{add='IND-GW.flinchbot.com'} -
NumberPattern '^\+1[2-9]\d\d[2-9]\d{6}$'
New-CsVoiceRoute -Name 'Indianapolis-
International' -PstnGatewayList @{add='IND-GW.flinchbot.com'}
-NumberPattern '^\+[2-9]\d{6,14}$'

New-CsVoiceRoute -Name 'Bogota-Local' -PstnGatewayList
@{add='BOG-GW.flinchbot.com'} -NumberPattern '^\+571'
```

```
New-CsVoiceRoute -Name 'Bogota-National' -PstnGatewayList
@{add='BOG-GW.flinchbot.com'} -NumberPattern '^\+57[124-8][2-
8]\d{6}$'
New-CsVoiceRoute -Name 'Bogota-International' -PstnGatewayList
@{add='BOG-GW.flinchbot.com'} -NumberPattern '\+((1[2-
9]\d\d[2-9]\d{6})|(?(?!(57))([2-9]\d{6,14})))$'

New-CsVoiceRoute -Name 'Munich-Local' -PstnGatewayList
@{add='MUN-GW.flinchbot.com'} -NumberPattern '^\+4989'
New-CsVoiceRoute -Name 'Munich-National' -PstnGatewayList
@{add='MUN-GW.flinchbot.com'} -NumberPattern
'^\+49(180\d{5,7}|[2-7]\d{5,}|[89][1-9]\d{5,10}|90[1-
9]\d{4,8})$'
New-CsVoiceRoute -Name 'Munich-International' -PstnGatewayList
@{add='MUN-GW.flinchbot.com'} -NumberPattern '^\+((1[2-
9]\d\d[2-9]\d{6})|(?(?!(49))([2-9]\d{6,14})))$'

New-CsVoiceRoute -Name 'Helsinki-Local' -PstnGatewayList
@{add='HEL-GW.flinchbot.com'} -NumberPattern '^\+3589'
New-CsVoiceRoute -Name 'Helsinki-National' -PstnGatewayList
@{add='HEL-GW.flinchbot.com'} -NumberPattern '^^\+358((1[3-
9]|[235689])[1-8]\d{3,9}|[123]0[^0]\d{2,7}|7([13]\d{7,8}|5[3-
9]\d{2,7}))$'
New-CsVoiceRoute -Name 'Helsinki-
International' -PstnGatewayList @{add='HEL-GW.flinchbot.com'}
-NumberPattern '^(?:00)((1[2-9]\d\d[2-9]\d{6})|([2-
9]\d{6,14}))(\D+\d+)?$'
```

There are a few more things left to be said here.

1. Since I haven't created my PSTN Usages yet, I receive a warning every time I run one of those commands: *WARNING: No PSTN usage specified. Users granted this voice policy will not be able to make outbound PSTN calls*. That OK as I will be creating those soon
2. No call can be made to Emergency Services or any other 3-digit service number. Remember that this is just a sample scenario and I'm trying to keep it simple. However, you probably want to permit these. If so, create a new route for these Emergency/Service numbers.
3. You don't technically need to create a new route for each call type. You can concatenate several of these regular expressions into one large regular expression. For example, if I want to add Emergency call

support to the Nashville-Local route, I could use this regular expression: (^\+?([2-9]11)$)|(\+1615)

It's up to you to decide how simple or complicated you want to make your routes.

Voice Policies

Voice Policies are used (directly) to permit users to use certain features and (indirectly) to control which calls they are permitted to make (Local, National, etc.).

I will be creating two Voice Policies for each location. One Voice Policy will be used along with the associated Route to limit calling to only numbers within a country (National) and the other Voice Policy will be created to permit International calling.

For demonstration purposes I will only create user-level dial plans but in your environment you may find it best to create site-level dial plans.

I'm also going to permit Call Park in the Nashville and Munich locations.

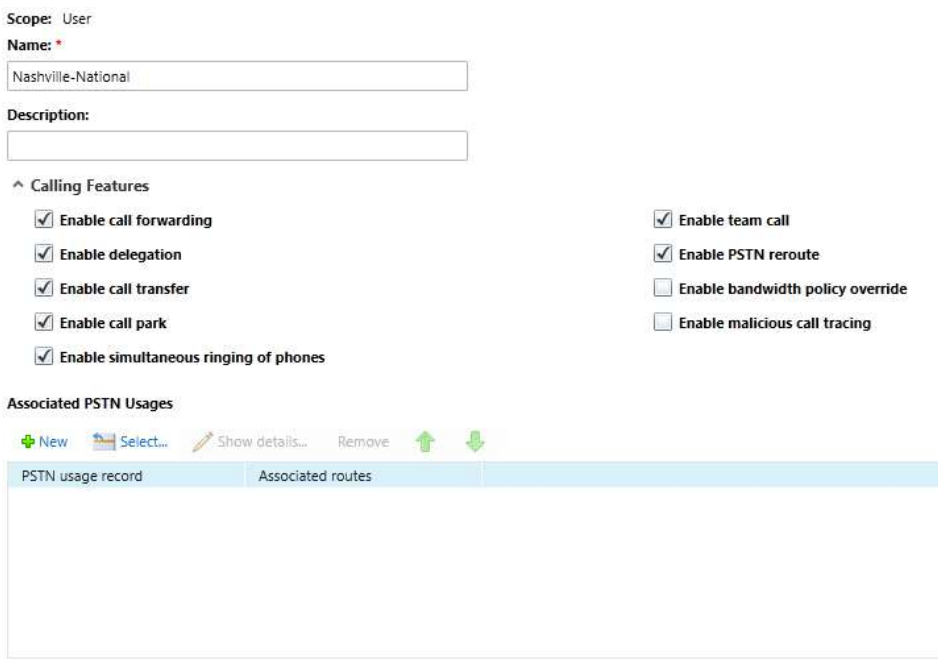
Creating Voice Policies is really simple. At its simplest, give it a name and you're done. At its most complex, you assign a name, select the features you want, and then create or select the PSTN Usages you need to apply to the Voice Policy. As this is just a demonstration I'm going to create the Voice Policies without creating the necessary PSTN Usages. In reality you would go on to the next step and finish things up.

But I want to make it clear that Voice Policies are independent of PSTN Usages which are independent of Routes. Ultimately you connect them together but I'm using this approach in the book to hopefully help you get these concepts straight in your head.

I'm going to create the first Voice Policy for Nashville using Control Panel and the rest using PowerShell.

First I'll create the Nashville-National Voice Policy. After assigning a name, I also tick the box to permit Call Park. I want to keep all of the other features as is.

Figure 13 – 26

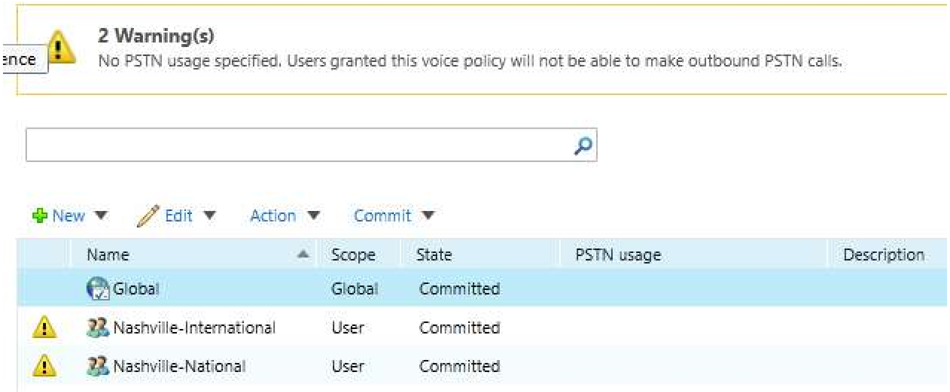


After creating the "Nashville-International" Voice Policy, I commit the changes in Control Panel. Immediately I get a pop-up warning.

Figure 13 – 27

This warning is being presented because these Voice Policies do not have a PSTN Usage assigned to them. As such, these policies are pretty much worthless!

Figure 13 – 28



I'll fix the PSTN Usage issue in the next section. But for now I'm going to proceed and create the rest of the Voice Policies using PowerShell. The below cmdlets are all pretty much the same except that I'm enabling the Call Park feature in Munich.

```
New-CsVoicePolicy -Identity "Indianapolis-National"
New-CsVoicePolicy -Identity "Indianapolis-International"

New-CsVoicePolicy -Identity "Bogota-National"
New-CsVoicePolicy -Identity "Bogota-International"

New-CsVoicePolicy -Identity "Munich-National" -EnableCallPark
$True
New-CsVoicePolicy -Identity "Munich-
International" -EnableCallPark $True

New-CsVoicePolicy -Identity "Helsinki-National"
New-CsVoicePolicy -Identity "Helsinki-International"
```

Just like Control Panel, PowerShell also shows a warning after creating these Voice Policies.

WARNING: No PSTN usage specified. Users granted this voice policy will not be able to make outbound PSTN calls.

This is an OK error for me as I haven't created those PSTN Usages yet. But I'm about to!

PSTN Usages

As discussed in the chapter on PSTN Usages, a PSTN Usage connects Voice Policies to Routes. In the previous 2 sections I received warnings because I created Voice Policies and Routes without assigning a PSTN Usage. Hopefully the relationship between these objects is clear in your head by now. If not, this section should certainly help.

As part of this scenario, we created two Voice Policies for each location – one which permits only local and national calling and one that permits local, national, and international calling. We've already created those Voice Policies so all we need to do now is create PSTN Usages and link those PSTN Usages to the Voice Polices and Routes.

Below is a table showing all of the existing Voice Policies and Routes along with the new PSTN Usages that will connect the two.

Nashville Voice Policies	PSTN Usages	Routes
Nashville-National	Nashville-National	Nashville-Local Nashville-National
Nashville-International	Nashville-International	Nashville-Local Nashville-National Nashville-International

Indianapolis Voice Policies	PSTN Usages	Routes
Indianapolis-National	Indianapolis-National	Indianapolis-Local Indianapolis-National
Indianapolis-International	Indianapolis-International	Indianapolis-Local Indianapolis-National Indianapolis-International

Bogota Voice Policies	PSTN Usages	Routes
Bogota-National	Bogota-National	Bogota-Local Bogota-National
Bogota-International	Bogota-International	Bogota-Local Bogota-National Bogota-International

Munich Voice Policies	PSTN Usages	Routes
Munich-National	Munich-National	Munich-Local Munich-National
Munich-International	Munich-International	Munich-Local Munich-National Munich-International

Helsinki Voice Policies	PSTN Usages	Routes
Helsinki-National	Helsinki-National	Helsinki-Local Helsinki -National
Helsinki-International	Helsinki-International	Helsinki-Local Helsinki-National Helsinki-International

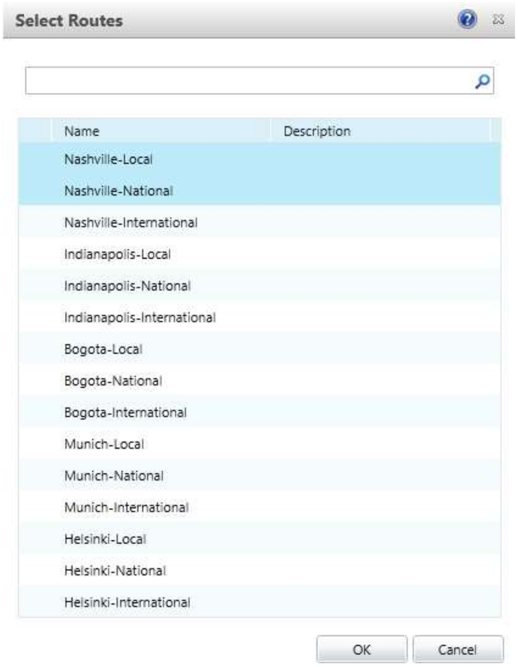
As a reminder, you cannot go to the PSTN Usages tab in Control Panel to create a PSTN Usage – which is a bit weird. Instead, you have to create a new PSTN Usage from within the Voice Policies tab in Control Panel. So to create the “Nashville-National” PSTN Usage, I open the “Nashville-National” Voice Policy.

From here I click “New” in the “Associated PSTN Usages” section.

This opens the “New PSTN Usage Record” screen. In the “Name:” field I type in “Nashville-National”.

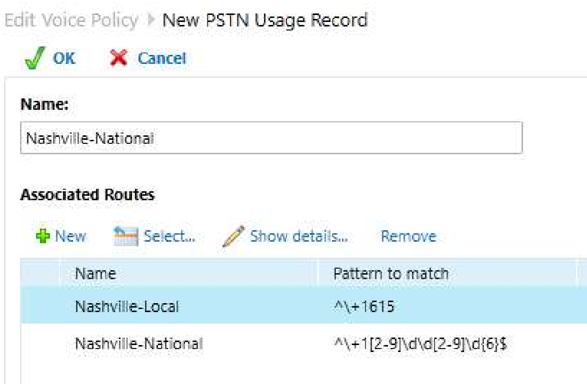
I then have to assign my routes. Since they are already created, I click the “Select” button in the “Associated Routes” section. This open the “Select Routes” screen. On this screen I click on both the “Nashville-Local” and “Nashville-National” Routes and click OK.

Figure 13 – 29



With those two routes added the "New PSTN Usage Record" now looks like the image below. After reviewing that the routes look correct, I click OK to create the new PSTN Usage.

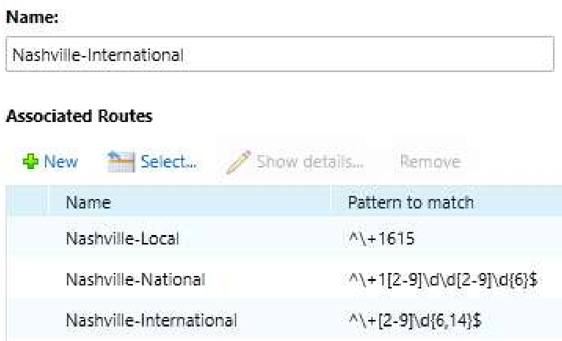
Figure 13 – 30



I will now proceed and create the PSTN Usage for the "Nashville-International" Voice Policy. It is the exact same in that I will also select the "Nashville-Local" and "Nashville-National" routes. The one difference is that I will also add the “Nashville-International” route to the PSTN Usage that I am creating.

Below is an image of the “Nashville-International” PSTN Usage. Note that it has the additional route to support international calls.

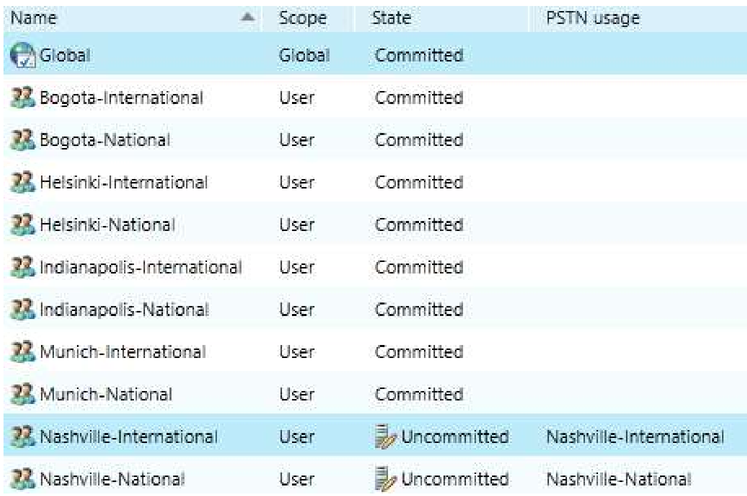
Figure 13 – 31



Name	Pattern to match
Nashville-Local	^\+1615
Nashville-National	^\+1[2-9]\d\d[2-9]\d{6}$
Nashville-International	^\+[2-9]\d{6,14}$

Before I commit these changes, I want you to see what the Voice Policies page now looks like. Notice that the two Nashville Policies at the bottom now have PSTN Usages that I need to commit.

Figure 13 – 32



Name	Scope	State	PSTN usage
Global	Global	Committed	
Bogota-International	User	Committed	
Bogota-National	User	Committed	
Helsinki-International	User	Committed	
Helsinki-National	User	Committed	
Indianapolis-International	User	Committed	
Indianapolis-National	User	Committed	
Munich-International	User	Committed	
Munich-National	User	Committed	
Nashville-International	User	Uncommitted	Nashville-International
Nashville-National	User	Uncommitted	Nashville-National

Finally, I commit the changes and the PSTN Usages are now active within my Skype for Business environment.

If I take a look now at the Routes tab, I can see that these PSTN Usages have been added to the routes.

Figure 13 – 33

Name	State	PSTN usage
Nashville-Local	Committed	Nashville-National, Nashville-International
Nashville-National	Committed	Nashville-National, Nashville-International
Nashville-International	Committed	Nashville-International
Indianapolis-Local	Committed	
Indianapolis-National	Committed	
Indianapolis-International	Committed	

I will create the rest of the PSTN Usages and assign the associated routes using PowerShell. There is no "New-CsPstnUsage" cmdlet. Which is a bit weird until you understand that all PSTN Usages are shared in the Global context. In other words, you can't scope a PSTN Usage to a Site or Pool level. As such, you use the Set-CsPstnUsage cmdlet to add a new PSTN Usage to the existing global collection.

Once the PSTN Usage has been added to the global collection, I then need to update the Voice Policies and Routes to assign the PSTN Usages to them.

```
Set-CsPstnUsage -Identity global -Usage @{Add="Indianapolis-
National", "Indianapolis-International"}
Set-CsVoicePolicy -Identity 'Indianapolis-
National' -PstnUsages @{Add="Indianapolis-National"}
Set-CsVoicePolicy -Identity 'Indianapolis-
International' -PstnUsages @{Add="Indianapolis-International"}
Set-CsVoiceRoute -Identity 'Indianapolis-Local' -PstnUsages
@{Add="Indianapolis-National","Indianapolis-International"}
Set-CsVoiceRoute -Identity 'Indianapolis-National' -PstnUsages
@{Add="Indianapolis-National","Indianapolis-International"}
Set-CsVoiceRoute -Identity 'Indianapolis-
International' -PstnUsages @{Add="Indianapolis-International"}

Set-CsPstnUsage -Identity global -Usage @{Add="Bogota-
National", "Bogota-International"}
Set-CsVoicePolicy -Identity 'Bogota-National' -PstnUsages
@{Add="Bogota-National"}
Set-CsVoicePolicy -Identity 'Bogota-International' -PstnUsages
@{Add="Bogota-International"}
```

```
Set-CsVoiceRoute -Identity 'Bogota-Local' -PstnUsages
@{Add="Bogota-National","Bogota-International"}
Set-CsVoiceRoute -Identity 'Bogota-National' -PstnUsages
@{Add="Bogota-National","Bogota-International"}
Set-CsVoiceRoute -Identity 'Bogota-International' -PstnUsages
@{Add="Bogota-International"}

Set-CsPstnUsage -Identity global -Usage @{Add="Munich-
National", "Munich-International"}
Set-CsVoicePolicy -Identity 'Munich-National' -PstnUsages
@{Add="Munich-National"}
Set-CsVoicePolicy -Identity 'Munich-International' -PstnUsages
@{Add="Munich-International"}
Set-CsVoiceRoute -Identity 'Munich-Local' -PstnUsages
@{Add="Munich-National","Munich-International"}
Set-CsVoiceRoute -Identity 'Munich-National' -PstnUsages
@{Add="Munich-National","Munich-International"}
Set-CsVoiceRoute -Identity 'Munich-International' -PstnUsages
@{Add="Munich-International"}

Set-CsPstnUsage -Identity global -Usage @{Add="Helsinki-
National", "Helsinki-International"}
Set-CsVoicePolicy -Identity 'Helsinki-National' -PstnUsages
@{Add="Helsinki-National"}
Set-CsVoicePolicy -Identity 'Helsinki-
International' -PstnUsages @{Add="Helsinki-International"}
Set-CsVoiceRoute -Identity 'Helsinki-Local' -PstnUsages
@{Add="Helsinki-National","Helsinki-International"}
Set-CsVoiceRoute -Identity 'Helsinki-National' -PstnUsages
@{Add="Helsinki-National","Helsinki-International"}
Set-CsVoiceRoute -Identity 'Helsinki-
International' -PstnUsages @{Add=”Helsinki-International”}
```

With all of that completed, below is an updated view of the Voice Policy tab

Figure 13 – 34

Name	Scope	State	PSTN usage
Global	Global	Committed	
Bogota-International	User	Committed	Bogota-International
Bogota-National	User	Committed	Bogota-National
Helsinki-International	User	Committed	Helsinki-International
Helsinki-National	User	Committed	Helsinki-National
Indianapolis-International	User	Committed	Indianapolis-International
Indianapolis-National	User	Committed	Indianapolis-National
Munich-International	User	Committed	Munich-International
Munich-National	User	Committed	Munich-National
Nashville-International	User	Committed	Nashville-International
Nashville-National	User	Committed	Nashville-National

I can also see that the routes in the Routes tab have also been assigned the correct PSTN Usages.

Figure 13 – 35

Name	State	PSTN usage	Pattern to match
Nashville-Local	Committed	Nashville-National, Nashville-International	^\+1615
Nashville-National	Committed	Nashville-National, Nashville-International	^\+1[2-9]\d\d[2-9]\d{6}$
Nashville-International	Committed	Nashville-International	^\+[2-9]\d{6,14}$
Indianapolis-Local	Committed	Indianapolis-National, Indianapolis-International	^\+1317
Indianapolis-National	Committed	Indianapolis-National, Indianapolis-International	^\+1[2-9]\d\d[2-9]\d{6}$
Indianapolis-International	Committed	Indianapolis-International	^\+[2-9]\d{6,14}$
Bogota-Local	Committed	Bogota-National, Bogota-International	^\+571
Bogota-National	Committed	Bogota-National, Bogota-International	^\+57[124-8][2-8]\d{6}$
Bogota-International	Committed	Bogota-International	\+((1[2-9]\d\d[2-9]\d{6})|(?(?!...
Munich-Local	Committed	Munich-National, Munich-International	^\+4989
Munich-National	Committed	Munich-National, Munich-International	^\+49(180\d{5,7}|[2-7]\d{5,}|...
Munich-International	Committed	Munich-International	^\+((1[2-9]\d\d[2-9]\d{6})|(?(?!...
Helsinki-Local	Committed	Helsinki-National, Helsinki-International	^\+3589
Helsinki-National	Committed	Helsinki-National, Helsinki-International	^^\+358((1[3-9]|[235689])...
Helsinki-International	Committed	Helsinki-International	^(?:00)((1[2-9]\d\d[2-9]\d{6})|...

Trunk Configuration

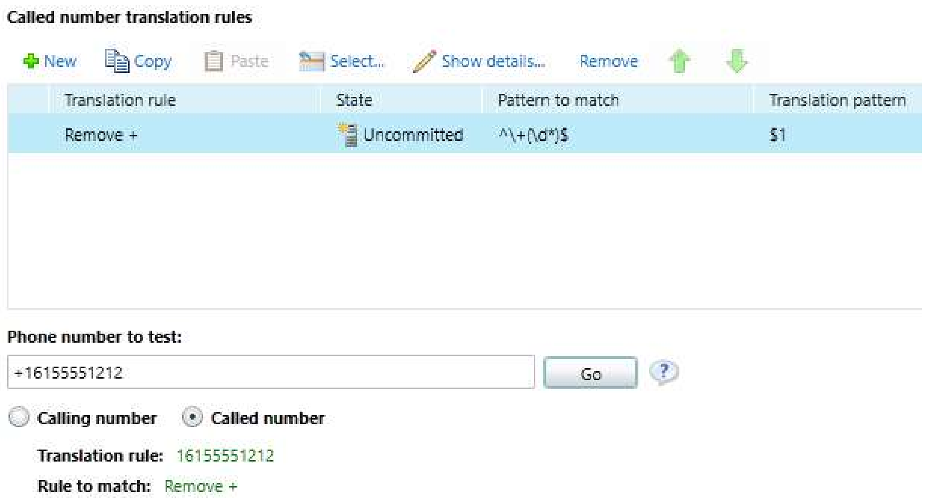
Trunk configurations are optional. You don't need to configure them as it's not a requirement to send calls to a Trunk. Much of what a Trunk Configuration does can be configured on your PSTN gateway.

For this scenario, I'll assume that the SIP trunk providers in Nashville and Indianapolis will not accept calls that begin with a "+". For example, they will accept the call "16155551212" but they will drop the call if it is "+16155551212". *(This is actually a very common scenario.)*

I create a Pool-level trunk configuration and add a "Called number translation rule" that removes the "+". I then go ahead and test the rule to make sure it works as I expect.

For the Indianapolis Trunk, I will configure it using PowerShell.

Figure 13 – 36



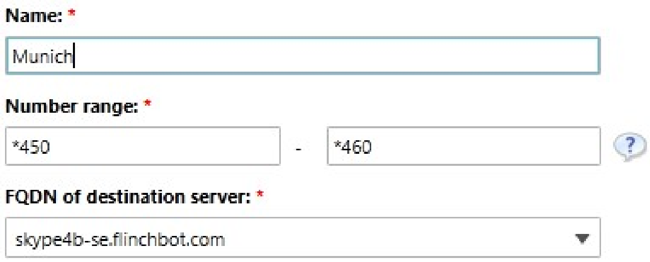

```
New-CsTrunkConfiguration -Identity PstnGateway:IND-
GW.flinchbot.com
New-CsOutboundTranslationRule -Identity ‘PstnGateway:IND-
GW.flinchbot.com/Remove +’ -Pattern ‘^\+(\d*)$’ -Translation
‘$1’
```

After adding these Trunk Configurations, any calls I make in Nashville or Indianapolis will be sent on to the gateway without the + sign.

Call Park

For Call Park, we want to create a range that is reserved for our users in Munich. The Call Park orbit range should be between *450 and *460.

Figure 13 – 37



In this case, we selected the Standard Edition server as that is the server located in the Munich office. The equivalent PowerShell is below.

```
New-CsCallParkOrbit -Identity “Munich” -NumberRangeStart
‘*450’ -NumberRangeEnd ‘*460’ -CallParkService
“ApplicationServer:skype4b-se.flinchbot.com”
```

The next requirement is to set a custom music on hold recording for Call Park. These steps must be configured via PowerShell.

The first thing to do is to see if I already have a configuration for Munich. This is done by running `Get-CsCpsConfiguration`.

Figure 13 – 38

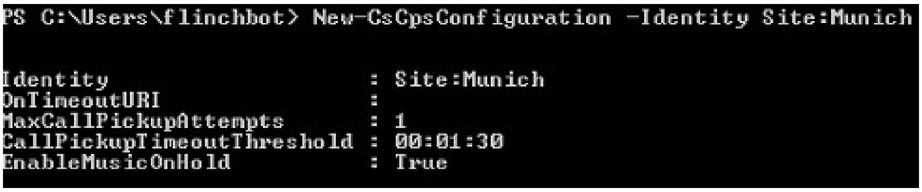
I only have a Global configuration. Since I never edit Global configurations, I will have to create a new configuration for the Munich Site. Note that I could also just use the Global configuration if the settings are correct and only create a new Call Park configuration if I need to make a change sometime later. However, I am assuming I will need to make a change at some point to the default Global settings. So I'll make a new one now.

I run the following to create a new Call Park configuration for the Munich site.

```
New-CsCpsConfiguration -Identity Site:Munich
```

Now I check the settings of this configuration to see if music on hold is enabled.

Figure 13 – 39



These settings look OK to me. So now I will proceed to import the custom music on hold audio file.

```
$a = Get-Content -ReadCount 0 -Encoding byte "C:\Temp\Call-Park-Announcement-Munich.wma"
Set-CsCallParkServiceMusicOnHoldFile -Service ApplicationServer:skype4b-se.flinchbot.com -Content $a
```

Unassigned Numbers

Next we get to create the Unassigned Number range.

The first thing we need to do is to create an announcement. This can only be done via PowerShell.

The below PowerShell will create the announcement on the Standard Edition server in Munich. It will be named "Unassigned Number-Munich" and a text-to-speech announcement will be played in German. Should this Unassigned Number need to be forwarded, it will go to the Operator in Nashville.

```
New-CsAnnouncement -Identity ApplicationServer:skype4b-
se.flinchbot.com -Name "Unassigned Number-
Munich" -TextToSpeechPrompt "Die von Ihnen gewählte Nummer
wurde nicht vergeben. Sie werden an den Empfang
weitergeleitet." -Language „de-DE" -TargetURI sip:operator-
nashville@flinchbot.com
```

Now that I have an announcement created I am going to create the Unassigned Number range. It wasn't really defined in the scenario, so let's just say that the number range in Munich is +49895554000 to +49895554999.

Figure 13 – 40

Of course, I can also create this using PowerShell.

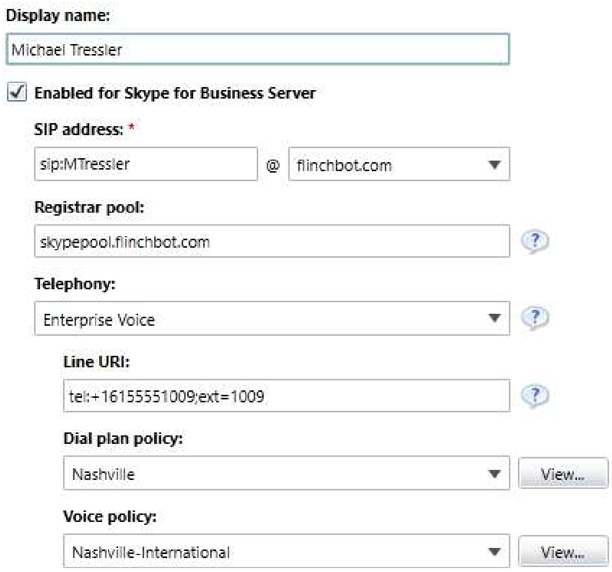
```
New-CsUnassignedNumber -Identity "Munich" -NumberRangeStart
"tel:+49895554000" -NumberRangeEnd
"+49895554999" -AnnouncementService ApplicationServer:skype4b-
se.flinchbot.com -AnnouncementName "Unassigned Number-Munich"
```

Configuring Users

Now that I have created Dial Plans and Voice Policies, I need to assign those to my users. Since this was discussed in detail in chapter 10, I'll just go over the highlights here.

I start by configuring myself. I'm in the Nashville office so I set myself with these values.

Figure 13 – 41



Changes to users are passed to the client via in-line configuration. This just means that for this configuration to take effect, I need to sign out and sign back in to my Skype for Business client.

Making Test Calls

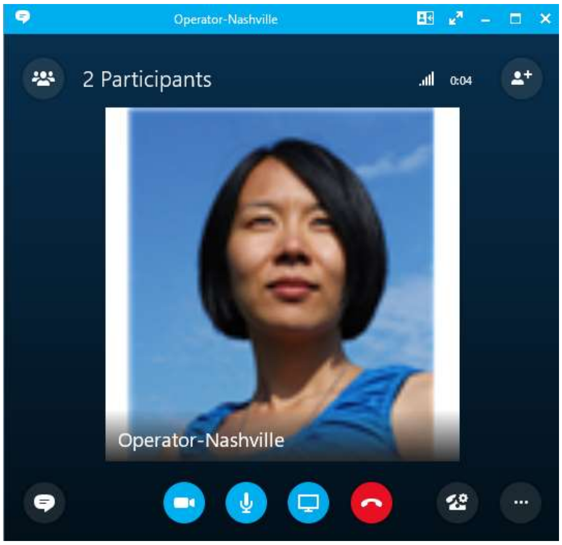
After signing in, I see that I now have a dial pad within my client. For testing, I start going through all of the configurations in the Dial Plan. First I dial "0" to see if I can connect to the Operator. *(Of course, the operator also needs to be configured for Enterprise Voice before this call can complete.)*

Sure enough, the Operator's full phone number appears in the client when I dial "0", just as I defined it in the Dial Plan.

Figure 13 – 42

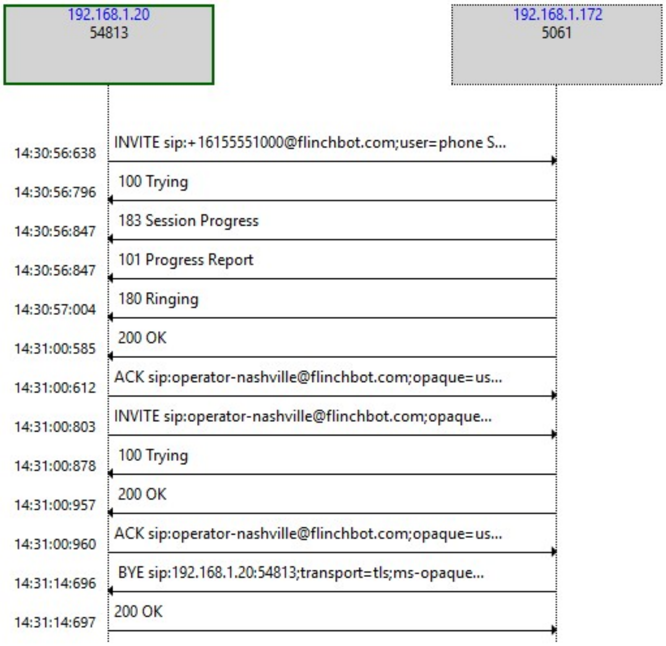
Now if I initiate the call, the Operator should get a call and we should be able to talk to each other. Sure enough, the call completed just fine.

Figure 13 – 43



This should be a peer-to-peer call and not go out via the PSTN. I can verify this by checking out my client log in Snooper. I find the call and look at the call flow diagram. I can immediately tell that it is a peer-to-peer call since the top line of the diagram shows the dialed phone number but towards the middle you see that I connected to “sip:operator-nashville@flinchbot.com”.

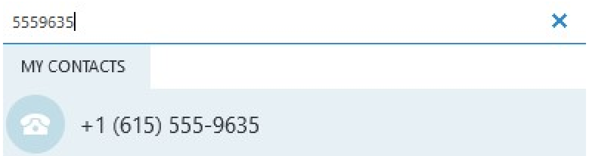
Figure 13 – 44



If this call were actually going out via the PSTN, I would not have seen the Operator's SIP address and instead would have seen the phone number throughout the entire call flow diagram.

Next I try the local call by making a 7-digit call to order a pizza from the local pizzeria. The phone number for the pizzeria is 555-9635. As I dial that into the client, I can see that it has been normalized to a full E.164 number.

Figure 13 – 45



I go ahead and submit the call and…the pizza will be here in 28 minutes!

Next I try to call a National number. Hey – why don't I order a pizza for my coworkers in Indianapolis? I think the phone number for their favorite pizzeria is 317-555-9845.

Figure 13 – 46

Not only has it been normalized just as defined in the dial plan but notice that Skype for Business automatically removes the dashes for me. After placing the call…success! My coworkers in Indianapolis will get their pizza shortly.

Finally, I need to try an international call. It's too late to order a pizza so I'll just call the weather service in Australia. That number is 61 1900 926 113. I am adding the 011 so that the dial plan knows that this is an international call.

Figure 13 – 47

Looks like it will be another sunny day in Australia.

Inter-Trunk Routing

As part of this scenario, I want to connect my legacy PBX in Nashville with my Skype for Business Server. I really want to make my Skype for Business pool the only path to and from the PSTN for the PBX. This is what Inter-Trunk Routing does.

The first thing to do is to add the PBX as a gateway and then build a trunk in Topology Builder.

Once that is done, you should try making a call from the PBX to a Skype for Business user to make sure this link is working. At its simplest, dial the full E.164 phone number of the Skype for Business user. On the PBX side, I would dial +16155551009. And hooray! My Skype for Business client rings.

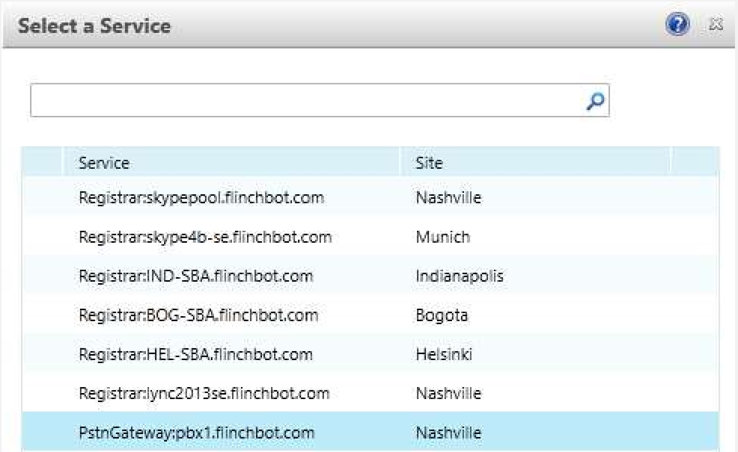
Site Dial Plan

But that isn't the most user-friendly way for the PBX users to place calls to their coworkers. I want to allow them to dial just the 4-digit extension of my Skype for Business Users. In order to support this, I will need to create a Site-level Dial Plan for the Nashville location.

Why a site-level dial plan? When calls come in from a non-authenticated source – such as a FAX machine or a PBX – I can't use a user-level Dial Plan because well…the PBX isn't a user. I prefer to use Site-level dial plans as these apply to all the Skype for Business servers in my site which can help in potential failover scenarios.

Note that you could also create a Pool-level Dial Plan and then select the PBX as the "pool". This is a useful option if you want to limit the devices (gateways, PBX, etc.) that can send 4-digit calls into the environment. Using a site-level dial plan allows any device in the Nashville Site to submit a 4-digit call. Using a Pool-level Dial Plan limits inbound 4-digit calls to only PBX1.

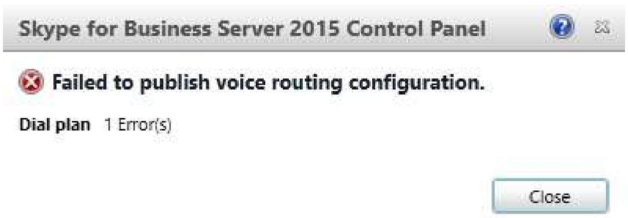
When creating a Pool-level Dial Plan you must provide the pool to which this Dial Plan applies. Below is an image showing that I can also select a Trunk (listed with the legacy naming of PstnGateway).



So off I go to create a new Site-level dial plan.

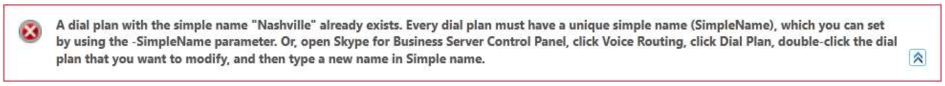
I created a new Site-level Dial Plan and picked "Nashville" as the site. When I click OK to create the Dial Plan I get a big nasty error:

Figure 13 – 48



Clicking "Close" let's me see more detail on this error at the top of Control Panel.

Figure 13 – 49



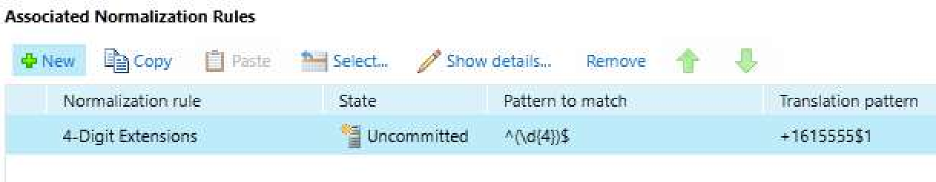
When I created the user-level dial plan for Nashville, I accepted the default for the "Simple Name" field which was "Nashville". Control Panel isn't smart enough to see that this Simple Name was already used so it tried to use it again. That's not allowed so I got the error. This is easily fixed by changing the Simple Name field for the Site Dial Plan I'm trying to create. I'm just going to change it from "Nashville" to "Nashville-Site".

Figure 13 – 50

Now I no longer receive an error message when trying to save and then commit this Dial Plan. I still need to add a normalization to this Dial Plan so that it will convert a 4-digit call into a full E.164 number.

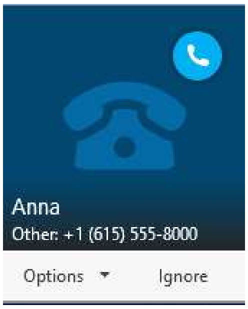
First I remove the "Keep All" default normalization and then I add the one that I want.

Figure 13 – 51



After committing this change I can try and make a phone call. Check this out! I just got a toast notification from Anna, one of my PBX users.

Figure 13 – 52



After accepting the call, I was able to talk with Anna.

The PowerShell commands for creating the Site Dial Plan are below.

```
New-CsDialPlan -Identity 'Site:Nashville' -SimpleName
'Nashville-Site'
New-CsVoiceNormalizationRule -Name '4-Digit Extension' -Parent
"Site:Nashville" -Pattern '^(\d{4})$' -Translation
'+1615555$1'
Remove-CsVoiceNormalizationRule -Identity 'Site:Nashville/Keep
All'
```

Calling from Skype for Business to the PBX is a little more involved as I'll need to update the Dial Plan, create a new Route, and assign that Route to an existing PSTN Usage.

All of the users on the PBX fall into the following phone number range: 1-615-555-8000 to 1-615-555-8999. Users on the PBX are only defined with their four-digit extension. So PBX user Anna can only be reached if you call the PBX with the following string: 8000

Pretty simple yet somewhat abhorrent to the E.164 standards I'm trying to encourage you to use in the Skype for Business world. But whatever…we can work with this.

A Skype for Business user can dial the PBX user by a four-digit extension. Instead of doing this the easy way and just passing those same four digits from the Dial Plan through to the Route, I'm going to do it the hard way. Because I'm hardcore.

I'm going to convert those four digits to a full E.164 number first. Why? At some point I intend to migrate these users to Skype for Business. So when Anna gets moved from the PBX to Skype for Business I won't have to do anything in the future to handle this. After she is moved if a coworker dials her 8000 extension, it will be converted to E.164, she will be found as a Skype for Business user via Reverse Number Lookup, and a peer-to-peer session will get established.

If I did it the "easy way", after I moved Anna to Skype for Business I'd have to create a specific normalization for Anna. When someone dials 8000 it gets converted to E.164 but keep only four digits for any other number in the 8000-range of the PBX. I'd have to do a lot more work in the future as I am migrating users.

If this doesn't make a lot of sense to you right now, don't worry. Just remember this: always convert everything to E.164.

Fortunately, the existing normalization in the Nashville Dial Plan is correct. A user on the PBX falls within the regex where I add +1615555 to any four-digit number. So I don't actually need to change the Dial Plan.

If the numbers are all the same (E.164 formatted) for Skype for Business and PBX users, how will Skype for Business know to route calls to the PBX instead of to Skype for Business?

Excellent question!

When the PBX extension gets normalized to an E.164 number, Skype for Business will perform a Reverse Number Lookup to see if that phone number belongs to a Skype for Business user. If there is no match, it will then try to route the call out to the proper location. We can control this routing via…Routes!

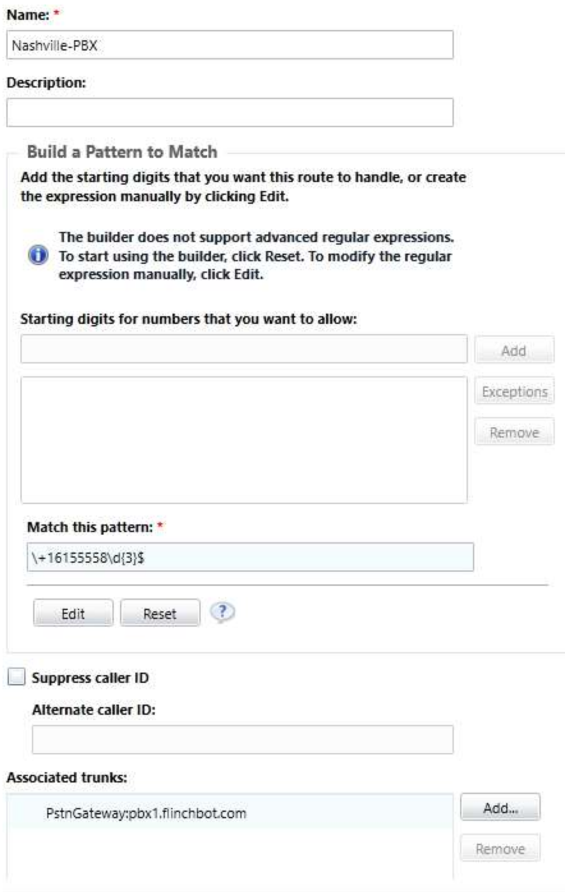
Route to PBX

I am going to create a Route that says the following:

If the last 4 digits of a phone number that starts with +1615555 falls in the range between 8000 and 8999, send that call to the PBX.

This is the route I created using Control Panel.

Figure 13 – 53



Unlike earlier examples, the PSTN Usages are already created. So I'll add those PSTN Usages directly into the "Associated PSTN Usages" section.

Figure 13 – 54

By now you should understand what this all means so I'll just move forward and show the PowerShell that can also be used to create the above.

```
New-CsVoiceRoute -Name 'Nashville-PBX' -NumberPattern
'\+16155558\d{3}$' -PstnGatewayList
@{add='pbx1.flinchbot.com'} -PstnUsages @{add='Nashville-
National','Nashville-International'}
```

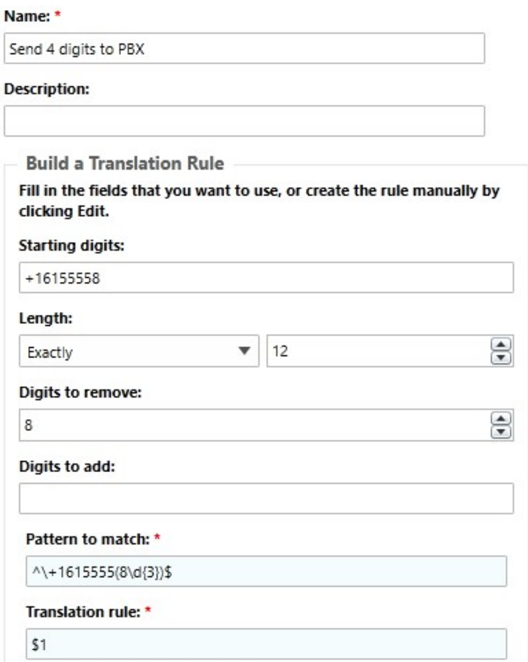
Now that the Route and the PSTN Usage have been created and updated, I need to do one more configuration change. I need to remove the "+1615555" part of the number so I only send the last 4 digits to the PBX.

PBX Trunk Configuration

To do this, I need to create a Trunk Configuration for pbx1.flinchbot.com and add a normalization.

This should be easy for you by now but I'll show you how I created the called number normalization.

Figure 13 – 55

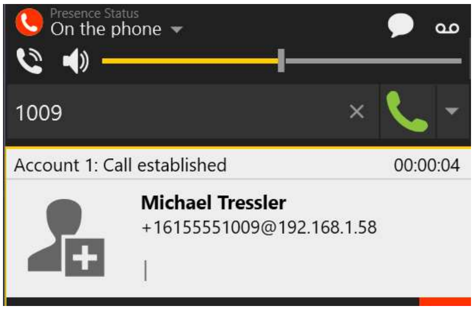


Here is the PowerShell that can be used to create the Trunk Configuration and assign the normalization.

```
New-CsOutboundTranslationRule -Identity
'PstnGateway:pbx1.flinchbot.com/Send 4 digits to PBX' -Pattern
'^\+1615555(8\d{3})$' -Translation '$1'
```

After making those changes, I can then make a successful call to a user on the PBX.

Figure 13 – 56



Now that I have calling working in both directions between Skype for Business and the PBX, I now need to configure Inter-Trunk Routing so the PBX users can call the PSTN.

Inter-Trunk Routing PSTN Usages

I am going to make two new PSTN Usages – one that only has the PBX and one that only has the Nashville Gateway. This will simplify troubleshooting as the other PSTN Usages have both gateways in their list of routes.

Creating just PSTN Usages can be done via Control Panel but it's painful. This is a step best done via PowerShell.

```
Set-CsPstnUsage -Identity global -Usage @{Add="Nashville-PBX",
"Nashville-PSTN"}
```

I open the Trunk configuration for pbx1.flinchbot.com and add the "Nashville-Local, Nashville-National, and Nashville-International" PSTN Usages. This gives PBX users the right to make all the configured call types. I could only assign the "Nashville-Local" and "Nashville-National" PSTN Usages if I wanted to prevent PBX callers from making International calls.

Next I edit the NAS-GW.flinchbot.com trunk and add the exact same PSTN Usages to the “Associated PSTN Usages” section.

Here is the PowerShell for how to set this up.

```
Set-CsTrunkConfiguration -Identity ‘pstngateway:NAS-
GW.flinchbot.com’ -pstnusage @{Add=”Nashville-PBX”}
Set-CsTrunkConfiguration -Identity
‘pstngateway:pbx1.flinchbot.com’ -pstnusage @{Add=”Nashville-
PSTN”}
```

Route Updates

I also need to assign this usage to the routes configured in Skype for Business so that calls from Skype for Business users to PBX users can be routed correctly. This can easily be done in Control Panel. Below is the PowerShell to accomplish this.

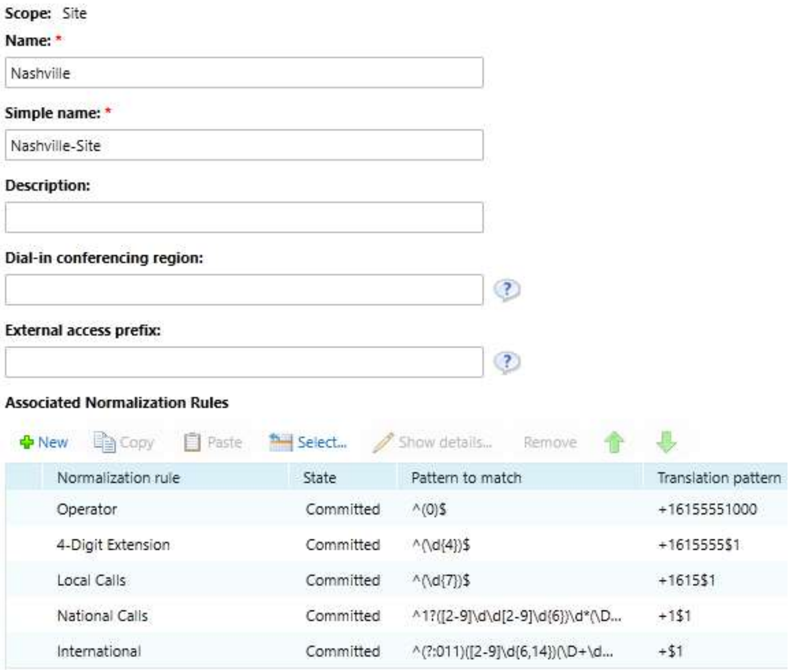
```
Set-CsVoiceRoute -Identity Nashville-Local -PstnUsages
@{Add=”Nashville-PSTN”}
Set-CsVoiceRoute -Identity Nashville-National -PstnUsages
@{Add=”Nashville-PSTN”}
Set-CsVoiceRoute -Identity Nashville-International -PstnUsages
@{Add=”Nashville-PSTN”}
```

Update Site Dial Plan

Now I am not quite done yet. I need to update the Nashville Site-level Dial Plan. Why? Remember, the Site-level Dial Plans handle inbound calls from unauthenticated users. If a national call comes into the Dial Plan and the Dial Plan can’t make a match, it will drop the call – simple as that.

I need to copy the user-level Dial Plan normalizations from the Nashville user Dial Plan into the Site Dial Plan. After I do this, users on the PBX will be permitted to make the same calls that Skype for Business users who are assigned the Nashville user dial plan can make.

Figure 13 – 57



I can also do all of this using PowerShell.

```
New-CsVoiceNormalizationRule -Name ‘Operator’ -Parent
‘Site:Nashville’ -Pattern ‘^(0)$’ -Translation ‘+16155551000’
-Priority 0
New-CsVoiceNormalizationRule -Name ‘Local Calls’ -Parent
‘Site:Nashville’ -Pattern ‘^(\d{7})$’ -Translation ‘+1615$1’
New-CsVoiceNormalizationRule -Name ‘National Calls’ -Parent
‘Site:Nashville’ -Pattern ‘^1?([2-9]\d\d[2-
9]\d{6})\d*(\D+\d+)?$’ -Translation ‘+1$1’
New-CsVoiceNormalizationRule -Name ‘International’ -Parent
‘Site:Nashville’ -Pattern ‘^(?:011)([2-9]\d{6,14})(\D+\d+)?$’
-Translation ‘+$1’
```

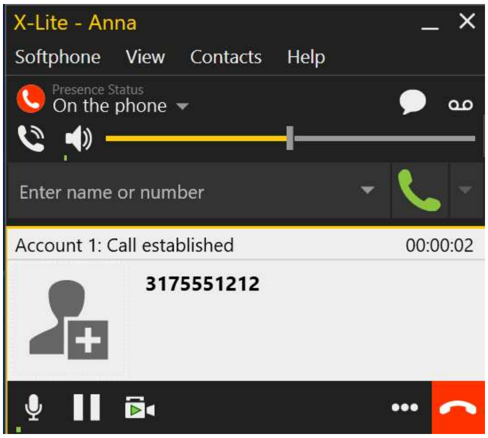
I want to point out a parameter in the first PowerShell cmdlet. At the very end I added “-Priority 0”. Dial Plans are analyzed from the top down. I want the Operator to be the first normalization in the list. In order to do this, I set the priority to zero. Any existing normalizations go down one spot in the list. I added the “4-Digit Extension” earlier in this chapter so it was already in the list.

I do not need to add any special priority to the rest of the normalizations as new normalizations are always added to the bottom of the list which is where I want these.

Now I think I have everything in place so it's time to make a call. First I call the local pizzeria form the PBX and order another pizza. That Works!

Again I call the pizzeria in Indianapolis from the PBX . The PBX client view of this call can be seen in Figure 13-58. The weather service in Australia can also be called. Success!

Figure 13 – 58



Least Cost Routing

As part of our scenario, we want to save on toll charges by using Least Cost Routing. For users in Nashville, if they call any number in Colombia, Germany, or Finland, that call should go out the gateway in each country.

The first thing we need to do is figure out how to know if a call is destined for Colombia, Germany or Finland versus any other phone call. If you are making sure you are following the E.164 standard then this is easy.

In North America, all E.164 formatted calls start with "+1" as in +16155551212. All calls in Colombia start with +57 as in +5715551212. Similarly, all calls in Germany start with +49 and all calls in Finland begin with +358.

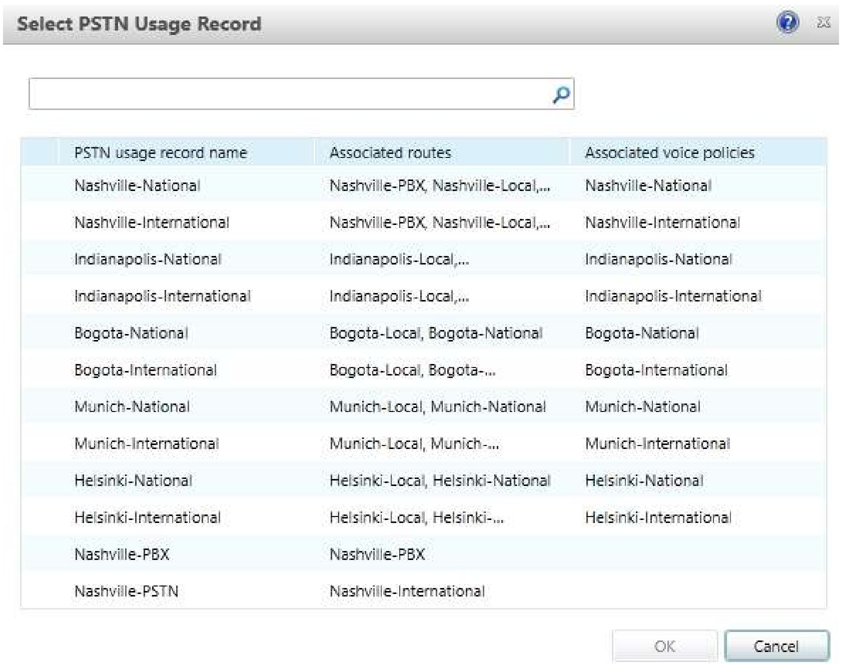
Knowing this, we can then use PSTN Usages and Routes to send the calls to the correct trunks in each location. Once it reaches the trunk, the call will get placed via the in-country gateway.

In our scenario this is going to be really easy to implement. We already have PSTN Usages that are limited to National calling in each country. For example, the "Munich-National" PSTN Usage only permits calls within Munich or within Germany. The same goes for the "Bogota-National" and "Helsinki-National" PSTN Usages. So we already have definitions of what calls are permitted in each country.

To set this up, edit the "Nashville-International" voice policy. Scroll down to the "Associated PSTN Usages" section and click the "Select..." button. This pops up the "Select PSTN Usage Record Screen" as seen in Figure 13-59. From here, highlight the "Bogota-National", "Munich-National", and "Helsinki-National" PSTN Usages.

Click OK.

Figure 13 – 59



PSTN usage record name	Associated routes	Associated voice policies
Nashville-National	Nashville-PBX, Nashville-Local,...	Nashville-National
Nashville-International	Nashville-PBX, Nashville-Local,...	Nashville-International
Indianapolis-National	Indianapolis-Local,...	Indianapolis-National
Indianapolis-International	Indianapolis-Local,...	Indianapolis-International
Bogota-National	Bogota-Local, Bogota-National	Bogota-National
Bogota-International	Bogota-Local, Bogota-...	Bogota-International
Munich-National	Munich-Local, Munich-National	Munich-National
Munich-International	Munich-Local, Munich-...	Munich-International
Helsinki-National	Helsinki-Local, Helsinki-National	Helsinki-National
Helsinki-International	Helsinki-Local, Helsinki-...	Helsinki-International
Nashville-PBX	Nashville-PBX	
Nashville-PSTN	Nashville-International	

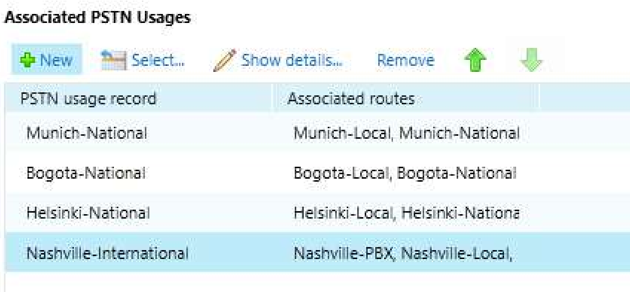
Click OK.

Now, the key to all of this is to make sure that the PSTN Usages are evaluated in order from most-specific to least-specific. In this case, it doesn't matter which country is listed in front of the other. All that matters is that all three countries are listed before the "Nashville-International" PSTN Usage.

This is because the three country PSTN Usages are very specific – they only allow calls to their country. But the "Nashville-International" usage permits calls throughout North America and all other countries. It is the least-specific PSTN Usage.

I click the arrows in the "Associated PSTN Usages" section to rearrange the PSTN Usage so that "Nashville-International" ends at the bottom. Below is how mine is ordered.

Figure 13 – 60



> If you really want to optimize the dialing experience, add the Nashville-National PSTN Usage to the top of the list. A good portion of calls from Nashville users will be national calls. As such, adding it to the top of the list means that the Nashville-National PSTN Usage will be evaluated first instead of last as part of the Nashville-International PSTN Usage.
>
> Not doing this means all national calls from Nashville users must first be evaluated against the Munich, Bogota, and Helsinki PSTN Usages every time.

If you want to do this all in PowerShell, here is how you do it.

We can’t re-order the PSTN Usages using PowerShell. Instead, we have to rebuild the existing PSTN Usages and then add them back in the preferred order.

The first cmdlet removes the PSTN Usage “Nashville-International” from the Voice Policy “Nashville-International”

```
Set-CsVoicePolicy “Nashville-International” -PstnUsages
@{remove=”Nashville-International”}
WARNING: No PSTN usage specified. Users granted this voice
policy will not be
able to make outbound PSTN calls.
```

Note the Warning. We can ignore this since we are about to add PSTN Usages back in to the Voice Policy.

```
Set-CsVoicePolicy “Nashville-International” -PstnUsages
@{add=”Munich-National”,”Bogota-National”,”Helsinki-
National”,”Nashville-International”}
```

A more elegant way to do this is to create an array and assign the desired PSTNUsages to that variable.

```
$Usages=@(“Munich-National”,”Bogota-
National”,”Helsinki-National”,”Nashville-International”)
```

Next, run the Set-CsVoicePolicy cmdlet and replace the existing *-PstnUsages* value with the value defined in the $Usages array.

```
Set-CsVoicePolicy “Nashville-International” -PstnUsages
@{replace= $Usages}
```

At this point, any calls I make to Germany will get routed to the Trunk in Germany and I will successfully bypass the transatlantic toll charges. Pretty neat, huh?

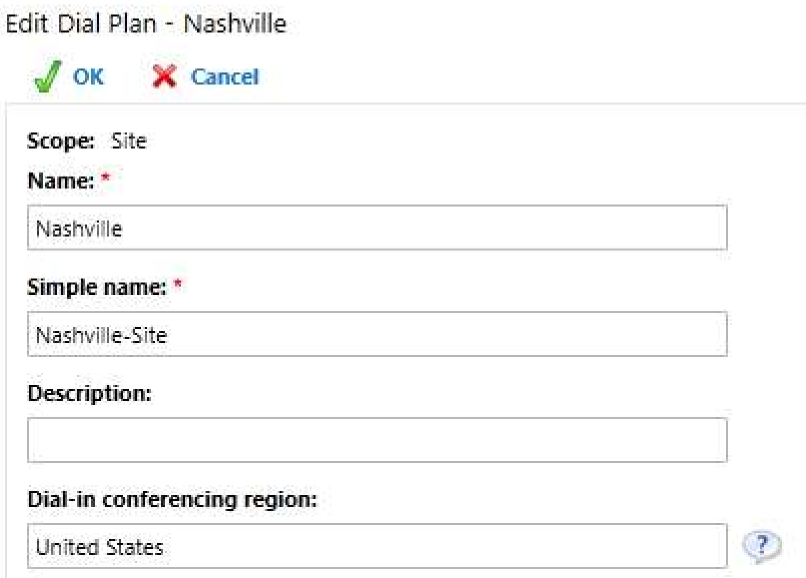
Dial-In Conferencing

For dial-in conferencing, here are the phone numbers I will assign to the Nashville, Indianapolis, and Munich offices. I'm also going to use continents for the region names.

Office	Region	Phone Number
Nashville	United States	+16155551999
Indianapolis	United States	+13175552999
Munich	Europe	+49895554999

The first thing I need to do is to create regions for each office. This is done by adding the region name to the "Dial-in conferencing region" setting in the Dial Plan defined for each office.

Figure 13 – 61



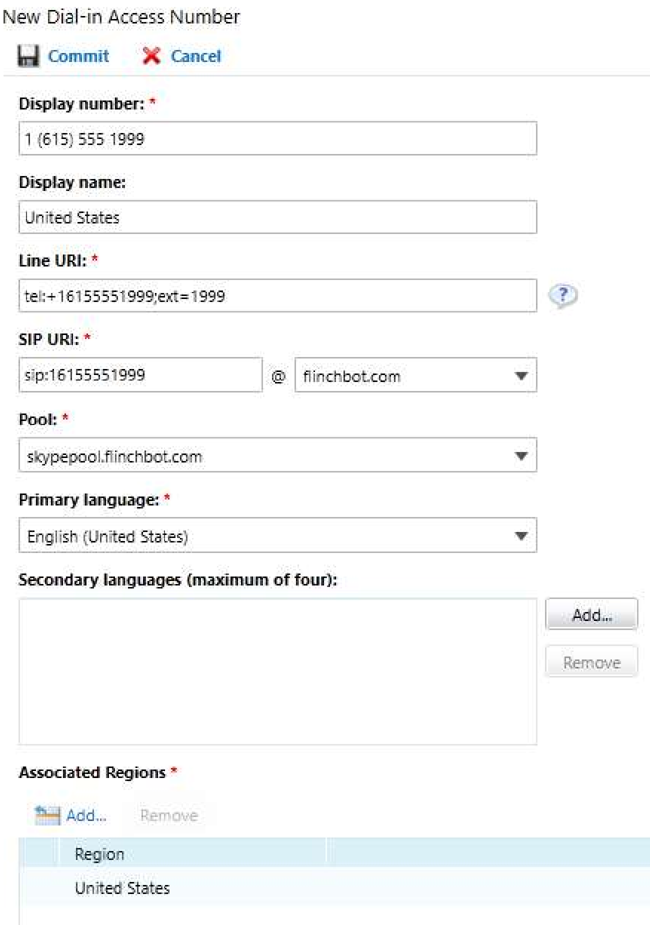
I can quickly do all three using PowerShell.

```
Set-CsDialPlan -Identity ‘Nashville’ -DialinConferencingRegion
'United States'
Set-CsDialPlan -Identity 'Indianapolis'
DialinConferencingRegion 'United States'
```

```
Set-CsDialPlan -Identity 'Munich' -DialinConferencingRegion
'Europe'
```

Now that those have been created, I can create the Dial-in Access Numbers. In Control Panel, I click on Conferencing and then Dial-In Access Number. Clicking the New icon lets me add a new number.

Figure 13 – 62



I will add the next two using PowerShell.

```
New-CsDialInConferencingAccessNumber -PrimaryUri
'sip:13175552999@flinchbot.com' -DisplayNumber '1 (317) 555-
2999' -DisplayName 'United States' -LineUri
'tel:+13175552999;ext=2999' -Pool
```

```
'skypepool.flinchbot.com' -PrimaryLanguage 'en-us' -Regions
'United States'
New-CsDialInConferencingAccessNumber -PrimaryUri
'sip:49895554999@flinchbot.com' -DisplayNumber '+49 895554999'
-DisplayName 'Europe' -LineUri
'tel:+49895554999;ext=4999' -Pool 'skype4b-se.flinchbot.com' -
PrimaryLanguage 'de-de' -secondaryLanguages 'en-gb' -Regions
'Europe'
```

Note that I added British English as a secondary language to the dial in access number in Europe.

After waiting a few minutes, I load the dial-in web page (https://dialin.flinchbot.com) to view the numbers I just added.

Figure 13 – 63

Conference Dial-in Numbers

Region	Number	Available Languages
Europe	+49 895554999	Deutsch (Deutschland), English (United Kingdom)
United States	1 (615) 555 1999tel:+16155551999;ext=1999	English (United States)
	1 (317) 555-2999	English (United States)

Summary

I hope that walking you through a complete configuration proved valuable. You can use each one of these steps as a template for your own implementations so you can see which steps are needed to get each feature working. Certainly your actual implementations will be more complex than what I presented here.

At this point. You are done with this book!

Thank you for reading. Now go out there and make the world a better place, one Skype for Business Enterprise Voice installation at a time.

Appendix 1 – Regular Expressions

In this section, you will receive a basic introduction to regular expressions. This section will not be a complete or exhaustive presentation on regular expressions. For that, there are literally dozens of books for you to reference. Rather, this section is meant to provide you with enough information so that you can get comfortable with regular expressions with Skype for Business.

A regular expression (regex) is a way to parse and process a string of characters. Depending on how the regex is written, it can store the results of its parsing into variables which can then be passed on to some other action.

The table below lists the most common regex operators that you will use when working with Skype for Business.

Table 1

Operator	Definition
^	Match the beginning of a string
$	Match the end of a string
\d	Match a single digit
()	Grouping. Anything within a group is assigned to a variable
{x}	Range of characters, x characters long
[]	Group specific characters or a range of characters
?	An optional character
|	A logical OR
,	At least
*	Match zero of more of the preceding character

Let's go over examples of each of the operators shown above. Since this is specific to Skype for Business, we will only focus on digits that might be used when calling someone.

The ^ Operator

With regards to the ^ (caret or hat) operator and its usage in Skype for Business, you only need to know that you always need to use this! It's that simple. In essence, it means "start parsing the characters at the beginning of what the user submitted".

The $ Operator

Similarly, all you need to know about the $ operator is that you need to add it to the end of all of your regexes. For our purposes it means "continue parsing the regex all the way to the end of what the user submitted".

More specifically, the $ means to stop processing. Without a $ sign, your regular expression could go on infinitely long because there is no defined end.

For the purposes of Skype for Business, consider these two operators as mandatory in any regular expression you build.

The \d Operator

\d is used to match a specific digit. For example, suppose you have 2-digit internal dialing and you need to know if someone dialed a 2-digit extension. Your regular expression would look like similar to this:

```
^\d\d$
```

By the way, if you omit the $, this means to match at least 2 digits. So the match would be any two digits followed by an endless amount of other digits.

The () Operator

In the world of Skype for Business, the regex ^\d\d$ means to match 2 digits. If you want to do something later with these digits, you need to assign them to a variable. This is done very easily by wrapping the relevant part of the regex between a pair of parentheses.

```
^(\d\d)$
```

By adding the parentheses, we are assigning the value returned by the regex to a variable called $1. Let's say we want to split up our two dialed digits into two separate variables. That would look like this:

```
^(\d)(\d)$
```

The first number is assigned to the variable $1 and the second number is assigned to the variable $2. So if the dialed 2-digit extension is 75, $1 would equal the number 7 and $2 would equal the number 5.

Using these together, Part 1

It may not seem like much, but I've already presented some pretty powerful operators. Let's say you wanted to match on the number 12. To do this, you would write the following regex:

```
^(12)$
```

Now, if you wanted to match on any number between 10 and 19, you would do the following:

```
^(1\d)$
```

The first digit must be a 1 but the second can be any digit as defined by the "\d" operator.

And both of these would be assigned to the $1 variable.

The \ Operator

The \ (backslash) operator is used to "escape" a character. If you look the "\d" operator, you can see this in use. The \ tells regex that it should not evaluate the "d" as the fourth letter of the alphabet but rather evaluate it as a special operator.

What is seen all the time in Skype for Business is escaping the "+". By default, regex thinks the "+" is an operator that matches one-or-more characters. For example, `ca+r` matches "car" and "caaaar" but not "cr".

But in our cases, we need the + to be used as a character and not as an operator. As such, we have to escape the + so it will match our E.164 phone numbers.

So If I wanted to match the phone number +13175551212 I would use the following regex:

^\+13175551212$

The {} Operator

The {} (curly braces) operator is used when you want to use a range of characters. In the examples above, we have simply repeated the \d operator twice to represent two characters. In a short regex, that isn't a problem. But try easily parsing the following regex:

```
^(\d\d\d\d\d\d\d\d\d\d)$
```

That requires you to do a lot of typing as well as being difficult to read that the regex is looking for 10 digits. A much more concise and easier way to write that regex is as follows:

```
^(\d{10})$
```

That says the exact same thing but in a much easier way. So in keeping with our 2-digit example, here is how that would look:

```
^(\d{2})$
```

This regex matches on any two digits.

The [] Operator

The [] (square brackets) are used to denote a specific range of digits. In the following (generally pointless) example, we will write a regex that matches when the user dials a 12.

```
^([1][2])$
```

As you can see, this does the exact same thing as the regex we wrote earlier: `^(12)$`. Now let's see how we can write a regex where we want to match any number between 10 and 19:

```
^(1[0-9])$
```

The first part of this regex should be easy for you to understand by now. Once we reach the brackets, we are asking to match on any single digit between 0 and 9. Essentially, this mirrors what the /d operator does and it is really up to you as to which operator you want to use to match the full range of digits between 0 and 9. But let's say you need to match a limited range of digits like 2-7. In the following example, we will match on any number between 12 and 17:

```
^(1[2-7])$
```

And of course, we can use this slightly more complicated approach where we match the range of numbers between 12 and 17 and between 22 and 27:

```
^([1-2][2-7])$
```

Using these together, Part 2

At this point, let me throw together what we've seen so far into a single regex. Given the regex below, which range of numbers could be assigned to the variable $1? And just as importantly, what does the user need to type in to Skype for Business in order to match this regular expression?

```
^\+1([3-5][0-7]\d{2})$
```

The regular expressions that could be assigned to the $1 variable are seen in table 2.

Table 2

3000-3799
4000-4799
5000-5799

In order for this regex to fire, the user would have to enter any number in the range above preceded by a + and the number 1. So the regex will match if the user types in +14564 but not if they type 4564. Note the +1 at the very beginning, just after the ^. These character are just as important as any other character entered by the user. Just because they are not passed into the $1 variable doesn't mean they are unimportant.

After the +1 we get to a left parenthesis, so we know that we are grouping things to send to the first variable - $1. Inside the square brackets we see 3-5. This means that the second number that the user enters must be a 3, 4, or 5.

We then see a second set of square brackets. This tells us that the third number entered must be a 0, 1, 2, 3, 4, 5, 6, or 7.

Next we see the \d operator followed by a curly brace. This tells us that there must be 2 more digits with no limitation as to what the digits are. Putting this all together gives us Table 2.

When you parsed the regex, did you know what to do with the +1 at the beginning? Here is a real world example of where we would require the +1 but then not send it to the $1 variable.

In North America, the situation exists where you have to use a +1 to dial a long distance 10-digit number. However, there are some areas where a 10-digit number is considered to be a local call and you should not add the +1 at the beginning when dialing.

In that case, you would build a regular expression and then "drop" the +1 by not assigning it to a variable. Let's say that you need to add a +1 to all 10 digit calls except when dialing the 317 area code. In that case, your regular expression could be the following:

```
^\+1(317\d{7})$
```

So if the user enters +13175551212, this will strip the leading +1 and pass on the rest of the dialed number as $1, assuming that the second, third, and fourth numbers are 317 followed by seven digits. Note that in our \d we use a 7 because we're statically matching on the first three of the ten numbers and we don't care what the final seven are.

One final example.

As a comparison of what has been presented so far, there are several ways the following regex could be written:

```
^\+1([3-5][0-7]\d{2})$
```

Several possible ways this could be written are below:

```
^\+1([3-5][0-7][0-9][0-9])$
^\+1([3-5][0-7][0-9]\d)$
^\+1([3-5][0-7]\d\d)$
```

All of these have the same end result and there is no specific reason to use one over the other. It is up to you which you prefer. Some may be easier to read while some are more concise.

The ? Operator

The ? is used when a number is optional. In the area code example above, we assume the user will enter a 1 followed by 10 digits. But what if we wanted that regex to match if the user enters a 1 followed by 10 digits *or* if they simply entered 10 digits. In that case, we would write a regex similar to this:

```
^1?(317\d{7})$
```

In this case, the regex will match if the user dials 13175551212 or 3175551212.

The | operator

The I (pipe) is used when you want to use an "or" operation. A simple example would be to match on a 12 or a 24. The following regex shows how that would be written:

```
^12|24$
```

In companies that have complicated dial plans, you could easily see something like the following:

```
^(12|14|57|99|123|45/d/d)$
```

This would match on any of the following numbers: 12, 14, 57, 99, 123, or 4500-4599. Note that the matched value isn't limited to 2 digits. It could be any length.

Using these together, Part 3

Now let's see how complicated this can get:

```
^\+1?([3-578][0-7|9]\d{2})$
```

Take a second to see if you can figure out what the correct range of numbers is that could be assigned to the $1 variable.

Table 3 shows the range of the numbers that could be assigned to $1.

Table 3

+3000-3799
+3900-3999
+4000-4799
+4900-4999
+5000-5799
+5900-5999
+7000-7799
+7900-7999
+8000-8799
+8900-8999

And it doesn't matter if the user prepends a 1 or not to the dialed number. We use a '?' after the 1 to make that optional. However, the + is required.

Now, on to the final three operators which you won't see often but could also prove very useful when working with Skype for Business.

The , Operator

The , (comma) operator is used when you need your matched number length to be either a specific range in length or at least a given length. That probably isn't the best constructed sentence so a few examples will help explain what that means.

Say that where you live considers both 7 and 8 digit numbers to be local numbers. Instead of creating 2 regexes for each condition - which is perfectly fine - you could create a single regex that encompasses both number ranges.

The following regex will match if a user dials either 7 or 8 digits:

^\d{7,8}$

If the user dials either 1234567 or 12345678, the regex will be a match.

Now let's add this operation to one of our existing regular expressions to see how it could fit in:

^([3-5][0-7]\d{2,3})$

In this example, we will match on either a 4 or 5-digit number so long as that number is in the following acceptable range.

Table 4

3000-3799
30000-37999
4000-4799
40000-47999
5000-5799
50000-57999

The * Operator

The * (asterisk) operator is used for when you want to match on zero or more of the preceding character. Like the comma operator, this one isn't seen very often in the Skype for Business world. It is usually used when removing invalid entries in the dial string.

Let's take a completely invalid example of calling a mobile phone in England. The following is how the number may be written on a business card (incorrect as it may be):

44 (0) 12 3456 7890

The 44 is the country code for England.

The (0) means that the number needs to be dialed if in England, but is not to be dialed outside of England. (See here for more information: http://revk.www.me.uk/2009/09/it-is-not-44-0207-123-4567.html)

12 3456 7890 is the local phone number.

Now if someone outside of England wants to dial that number, we need to strip out the 0. But we also want our regex to fire if someone is calling England and omitted the entire 0 mess entirely (i.e., they dialed 441234567890)

The following regex will handle this situation:

^44[0]*(\d{9,10})$

Let us break this down. Starting from the beginning (^) we need to see a 44. The next section in the square brackets has a 0 followed by the asterisk. This means that there is a match if *zero or more items in the square brackets exist*. If this is the case, the comparison marches on and verifies that there are either 9 or 10 digits. If this is all correct, it then assigns either 9 or 10 digits to the $1 variable.

Since the * can match on zero or more values, this regex will fire if the user entered the 0 or not. But if we don't have this added to our regex, it will never fire if the user entered the 0.

The ?: Operator

This operator has appeared a few times in the book and it stands for "non-capturing group". Specific to this book, it's been used with the International calling regex:

```
^(?:011)([2-9]\d{6,14})(\D+\d+)?$
```

Note that the ?:011 is wrapped in parentheses. This should mean that this gets assigned to the $1 variable. But by adding the ?: it means to not assign this grouping to a variable. So the second pair of parentheses is assigned to the $1 variable instead.

The ?! (Negative Lookahead) Operator

Sometimes you want to create a regex that allows everything **except** something specific. Below is an example where we allow any number except for those that begin with 900 or 976.

```
^(?!(900|976))(\d+)$
```

So this number will return a match:

9011234567

But this number will fail to match:

9001234567

This is called a negative lookahead as the regular expression has to look ahead to see if the rest of the regular expression is a match before deciding if it is a "negative match" (rejected) or a "positive match" accepted.

Summary

That's it! If you've understood this section, then pat yourself on the back. When looking at regexes your eyes can easily glaze over in confusion and possibly horror. But now when you see a regex, or at least one in the context of Skype for Business, you should have a pretty good understanding of what is happening.

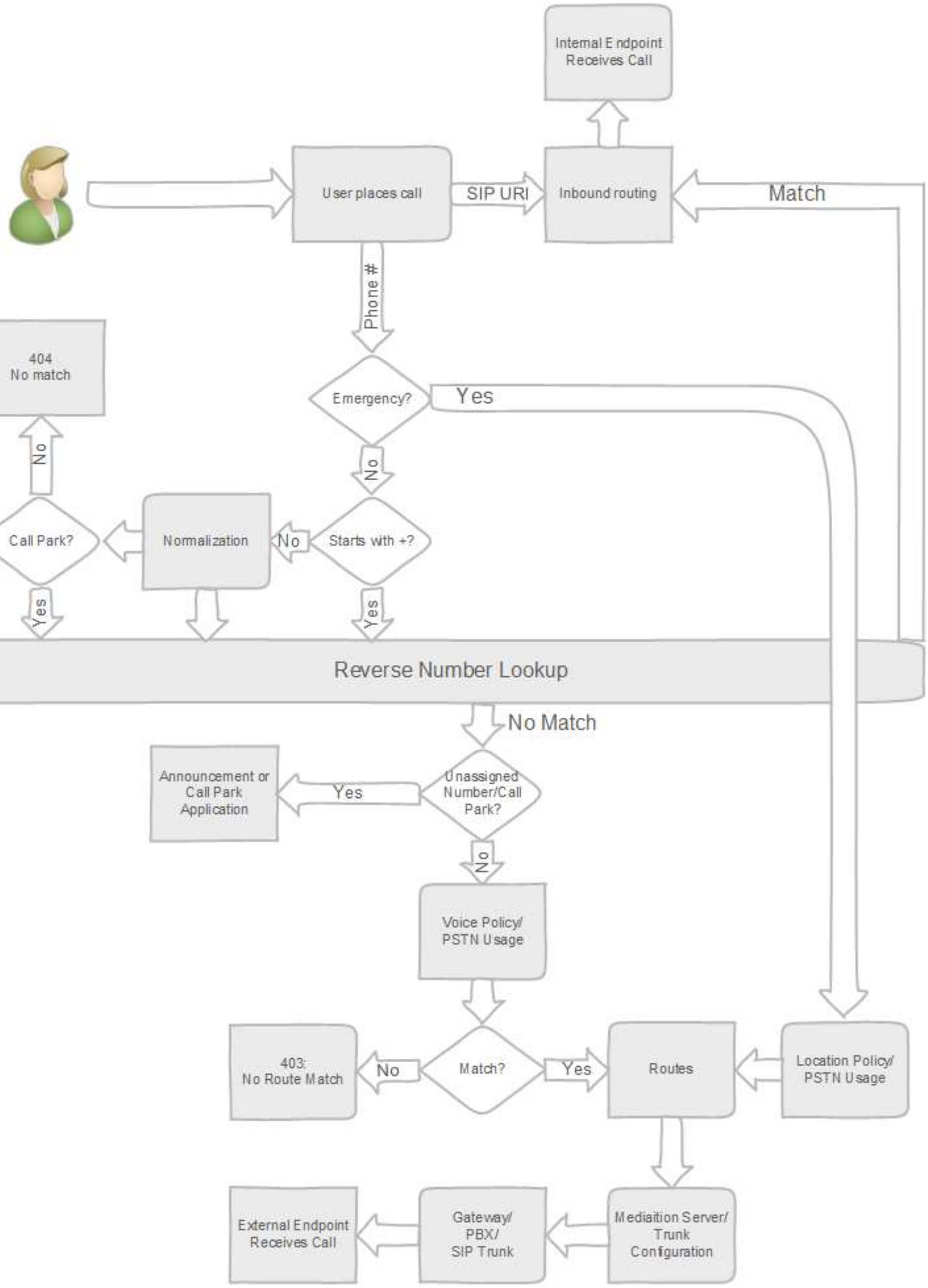
Appendix 2 – Call Routing

Skype for Business follows a call flow in order to decide when it should apply certain processing or routing to a call.

On the next page is a flow chart that shows all of the steps that happen after a call is placed. By following the flow chart, you can see which path any given call takes which can help you both with understanding Skype for Business Enterprise Voice better as well as assisting in troubleshooting.

There is a lot to this so I'll explain the whole process in the next few pages. You don't have to know any of this to make Skype for Business Enterprise Voice work. However, there are times when things don't seem to work the way you might expect them to. At those times, referring to this chart might help you figure out what is going on.

Figure A – 1


Internal Endpoint Receives Call
User places call
SIP URI
Inbound routing
Match
Phone #
404 No match
Emergency?
Yes
No
No
Call Park?
Normalization
No
Starts with +?
Yes
Yes
Reverse Number Lookup
No Match
Announcement or Call Park Application
Yes
Unassigned Number/Call Park?
No
Voice Policy/ PSTN Usage
403: No Route Match
No
Match?
Yes
Routes
Location Policy/ PSTN Usage
External Endpoint Receives Call
Gateway/ PBX/ SIP Trunk
Mediaition Server/ Trunk Configuration

The first thing that happens, obviously, is that the user places a call. If the call contains a SIP URI, then things are easy. Skype for Business finds the user who owns that SIP URI and tries to connect the call. Job Done!

Things get complicated if the user dials a phone number.

The first thing that happens is to determine if this is an Emergency Call. An Emergency Call is defined by the Location Profile assigned to the user and the emergency number defined in that profile. Location Profiles are outside the scope of this book. So just be aware that naming a normalization in a Dial Plan "Emergency" has no bearing on this routing step. However, should you have a Location Policy with a defined Emergency Number, and that number gets called, a few steps get skipped and the call is immediately handed off to a Voice Policy which starts the whole call routing flow.

If it's not an emergency number, the next step is to see if the call starts with a + in the first position, such as +1234567. Note that Skype for Business doesn't check for the validity of the number (though the assigned Dial Plan on the Skype for Business client could reject it before it ever gets sent into this call flow). It need not be a proper E.164 phone number; it only needs to start with a +.

If the call does not start with a +, then the Dial Plan normalizations get processed. Ostensibly this will convert the call to a properly formatted E.164 number.

If the normalization rules fail to match the dialed number, one last check is made to see if the dialed number falls within a defined Call park range. If so, the processing carries on. If the call cannot be normalized and it is not a number in the call park range, then the call gets dropped.

Next, Reverse Number Lookup happens. This is the process where Skype for Business checks to see if the dialed number belongs to one of your Skype for Business users. This is done in order to prevent calls from needlessly going to the PSTN and looping back into your office. The goal is to cut down on using voice channels on your PSTN line as well as to possibly remove toll charges should the user you called be in a remote office.

If Reverse Number Lookup finds a match, the call gets routed to the Inbound Routing mechanism and the call gets delivered to the internal user – or at least it tries to deliver it. The call may get redirected to their voice mail or forwarded to their mobile phone if they have simultaneous ringing enabled.

If Reverse Number Lookup fails to find a match, it then checks to see if the call falls into a Call Park orbit or an Unassigned Number Range. If you've read the Voice Features chapter and the section on Unassigned Numbers, I say it's easiest to just define all of your public numbers as an Unassigned Number range. If you look at the chart, if the Reverse Number Lookup matches a user, that call gets delivered before Unassigned Numbers is even evaluated.

So if the call is an Unassigned Number or a Call Park orbit, then the call gets routed there. Failing that, the call is then assumed to be headed toward the PSTN.

First a check is made to see if the call is permitted via Voice Policy and related PSTN Usages. If the call is not permitted, then the call is dropped.

However, if there is a Voice Policy match, the call is then forwarded to the correct route. At this point, the Route selects a Trunk and any Trunk Configurations are applied.

Finally, the call is handed off to the PSTN Gateway (or an SBC, PBX, or SIP trunk) and sent out to the PSTN.

Summary

This flowchart is very handy to be familiar with. This is the core engine that defines how call routing within Skype for Business works. It also lets you understand how certain "tricks" work like using your whole DID/DDI range as an Unassigned Number range or why calls that start with a + might not get normalized.

Appendix 3 – SEFAUtil

The Secondary Extension Feature Activation Utility (SEFAUtil) is a support tool which originally appeared in the Lync 2010 Resource Kit. It is most often used to administratively set delegate-ringing and call-forwarding settings on behalf of a Skype for Business user. As an example, without this tool, you would have to log in as the user in order to change their call forwarding settings. Using SEAUtil, you can change the call forwarding settings from a server instead.

SEFAUtil is also required if you are using Lync Server 2013 and want to assign delegates to a Group Call Pickup group.

Install SEFAUtil

If you have not yet configured SEFAUtil in your environment, here are the high level steps:

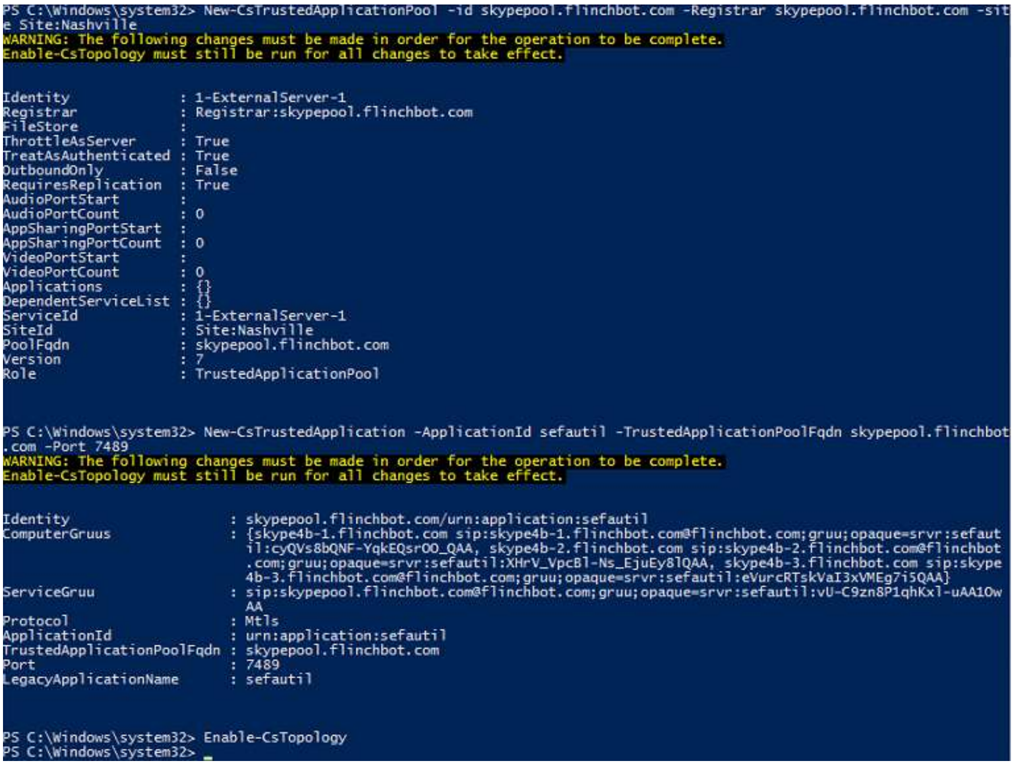
1. Use the `New-CsTrustedApplicationPool` cmdlet to create a new trusted application pool.
2. Use the `New-CsTrustedApplication` cmdlet to specify the SEFAUtil tool as trusted application.
3. Run the `Enable-CsTopology` cmdlet to enable the topology.
4. If you don't already have it, download the Skype for Business Server version of the SEFAUtil tool and install it on the trusted application pool you created in step 1.
    a. Link to SEFAUtil: https://www.microsoft.com/en-us/download/details.aspx?id=47704

5. Verify that SEFAUtil is running correctly by running it to display the call forwarding settings of a user in the deployment.

Here are the commands I needed to run to get his configured in my environment.

```
New-CsTrustedApplicationPool -Identity skypepool.flinchbot.com
-Registrar skypepool.flinchbot.com -site Site:Nashville
New-CsTrustedApplication -ApplicationId
sefautil -TrustedApplicationPoolFqdn skypepool.flinchbot.com -
Port 7489

Enable-CsTopology
```

Figure A – 1



Validating SEFAUtil

To verify that SEFAUtil works, I run the following command against one of my users.

```
SEFAUtil /server:skypepool.flinchbot.com
flinchbot@flinchbot.com
```

Figure A – 2

The above command shows how you can view the settings for a given user. The output lets you know if the user is enabled for Unified Messaging, if they have Simultaneous Ring enabled, if they have Call Forwarding enabled, and if they are a member of a Group Call Pickup orbit.

Simultaneous Ring

If you want to set the Simultaneous Ring settings for one of your users, you can use SEFAUtil to accomplish this. The below example enables simultaneous ring for the user mtressler@flinchbot.com and sets the simultaneous ring number to +16155550088.

```
SEFAUtil /server:skypepool.flinchbot.com
mtressler@flinchbot.com /setsimulringdestination:+16155550088
/enablesimulring
```

Figure A – 3

Notice that the output of the command gets updated to show that simultaneous ring is enabled and to where it is set. These changes also take effect immediately. Unlike some changes to the user, there is no need to sign out and sign back in to the client.

To change the simultaneous ring number, just run the same command above but you can leave off the `/enablesimulring` parameter.

To remove simultaneous ringing from a user, use the `/disablesimulring` parameter.

```
SEFAUtil /server:skypepool.flinchbot.com
mtressler@flinchbot.com /disablesimulring
```

Call Forwarding

You can edit the call forwarding settings of your users using the SEFAUTil tool. To do this, you need to use a few switches:

`/enablefwdnoanswer` – enable call forwarding

`/callanswerwaittime` – how long to wait before forwarding the call

`/setfwddestination` – the phone number to forward to

In the below example, I enabled call forwarding. If there is no answer after 15 seconds, the call gets forwarded to the phone number +16155550033. Note the phone number format in the `/setfwddestination` parameter in the example below. You can technically leave off everything after the phone number. The rest will get automatically added by SEFAUtil. However, if you are in an environment with multiple SIP domains, you should specify everything seen in the `/setfwddestination` parameter.

```
SEFAUtil.exe /server:skypepool.flinchbot.com
sip:mtressler@flinchbot.com /enablefwdnoanswer
/callanswerwaittime:30
/setfwddestination:+16155550033@flinchbot.com;user=phone
```

Figure A – 4

```
C:\>SEFAUtil.exe /server:skypepool.flinchbot.com sip:mtressler@flinchbot.com /en
ablefwdnoanswer /callanswerwaittime:30 /setfwddestination:+16155550033@flinchbot
.com;user=phone
User Aor: sip:mtressler@flinchbot.com
Display Name: Michael Tressler
UM Enabled: False
Simulring enabled: False
User Ring time: 00:00:30
Call Forward No Answer to: sip:+16155550033@flinchbot.com;user=phone
```

To change one of these attributes, you can run the above command but leave off the `/enablefwdnoanswer` parameter. For example, if I wanted to change the number to which I am forwarding, I would run the following:

```
SEFAUtil.exe /server:skypepool.flinchbot.com
sip:mtressler@flinchbot.com
/setfwddestination:+13175559900@flinchbot.com;user=phone
```

If you want the call to forward immediately, you would use the `/enablefwdimmediate` parameter. You can also forward a call to another user. The below example will immediately forward a call destined for mtressler@flinchbot.com to byount@flinchbot.com:

```
SEFAUtil.exe /server:skypepool.flinchbot.com
sip:mtressler@flinchbot.com /enablefwdimmediate
/setfwddestination:byount@flinchbot.com
```

Figure A – 5

To disable call forwarding, use either the `/disablefwdimmediate` or the `/disablefwdnoanswer` depending on if the call forward is immediate or has a delay.

Below I will disable the immediate call forwarding I set up above.

```
SEFAUtil.exe /server:skypepool.flinchbot.com
sip:mtressler@flinchbot.com /disablefwdimmediate
```

Group Call Pickup

If you are using Lync Server 2013, you need to use SEFAUtil to add delegates to a Group Call Pickup orbit. If you are using Skype for Business Server 2015 with the November 2015 Cumulative Update installed, you can use the cmdlets shown in chapter 8.

To enable a user to be a Group Call Pickup delegate, use SEFAUtil with the /enablegrouppickup parameter. In the below example, I am adding the user mtressler@flinchbot.com to the Group Call Pickup '#201'.

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com /enablegrouppickup:#201
```

Figure A – 6

To remove the Group Call Pickup membership for a user, you need to use the `/disablegrouppickup` parameter. In the below example, I disable Group Call Pickup for the user mtressler@flinchbot.com.

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com /disablegrouppickup
```

Team Call

Team call was discussed in chapter 4. Team Call is a feature of the client where you can configure your phone number to ring on not only your client but also on the client of your coworkers.

You can use SEFAUtil to administratively make changes to a user's Team Call configuration.

Enable Team Call

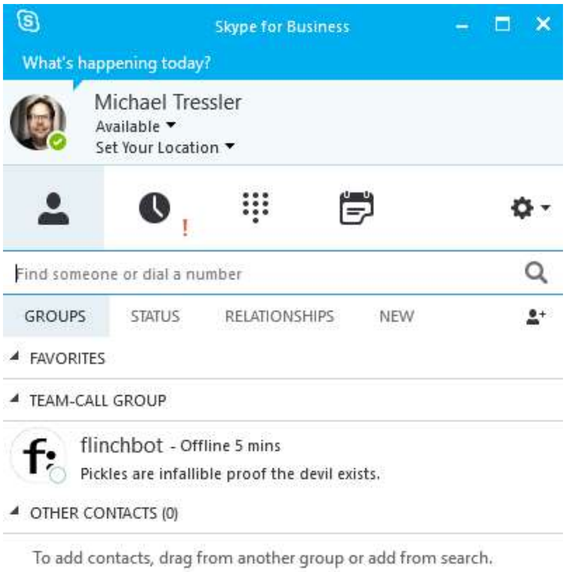
To enable Team Call for the user mtressler@flinchbot.com and assign the user flinchbot@flinchbot.com to the Team Call group, run the following command:

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com
/addteammember:flinchbot@flinchbot.com /simulringteam
```

Figure A – 7

After enabling Simultaneous Ring for the Team Call group and adding the user flinchbot@flinchbot.com, the client updates to reflect these changes. A new "Team-Call Group" group has been added to the client.

Figure A – 8



Note that if you only use the `/simulringteam` parameter and the Team Call group has no members, then Team Call will not be enabled. As such, if you are enabling a Team Call group for the first time, be sure to also use the `/addteammember` parameter to add a member to the Team Call group.

Disable Team Call

If you want to disable Team Call while still leaving all of the members in the group, you can use the `/disableteamcall` parameter.

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com /disableteamcall
```

Figure A – 9

```
C:\>SEFAUtil mtressler@flinchbot.com /server:skypepool.flinchbot.com /disableteamcall
User Aor: sip:mtressler@flinchbot.com
Display Name: Michael Tressler
UM Enabled: False
Simulring enabled: False
User Ring time: 00:00:30
Call Forward No Answer to: sip:byount@flinchbot.com
```

Edit Team Call Members

If you want to add a new member to mtressler@flinchbot.com's Team Call group, use the `/addteammember` parameter.

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com
/addteammember:byount@flinchbot.com
```

Figure A – 10

```
C:\>SEFAUtil mtressler@flinchbot.com /server:skypepool.flinchbot.com /addteammember:byount@flinchbot.com
User Aor: sip:mtressler@flinchbot.com
Display Name: Michael Tressler
UM Enabled: False
Simulring enabled: False
Team ringing enabled. Team: sip:flinchbot@flinchbot.com sip:byount@flinchbot.com
```

The client will update to show this new member of the Team Call group.

If you want to remove a member from the Team Call group, use the `/removeteammember` parameter. Note that if you remove the last member of a Team Call Group, then simultaneously ringing the Team Call group will automatically be disabled.

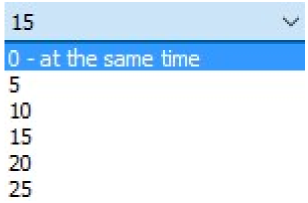
```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com
/removeteammember:byount@flinchbot.com
```

Delayed Team Call

You can configure Team Call so that it doesn't immediately (simultaneously) ring the members of the Team Call Group, Instead, you can have it ring the user first and then after a few seconds ring the members of the group.

When doing this within the client, you can only select the times seen in the image below.

Figure A – 11



Using SEFAUtil, you aren't limited to the times listed above. So if you want an 8 second delay, you can set that value using SEFAUtil. In the below example, I will set the ring delay to 12 seconds by using the `/delayringteam` parameter.

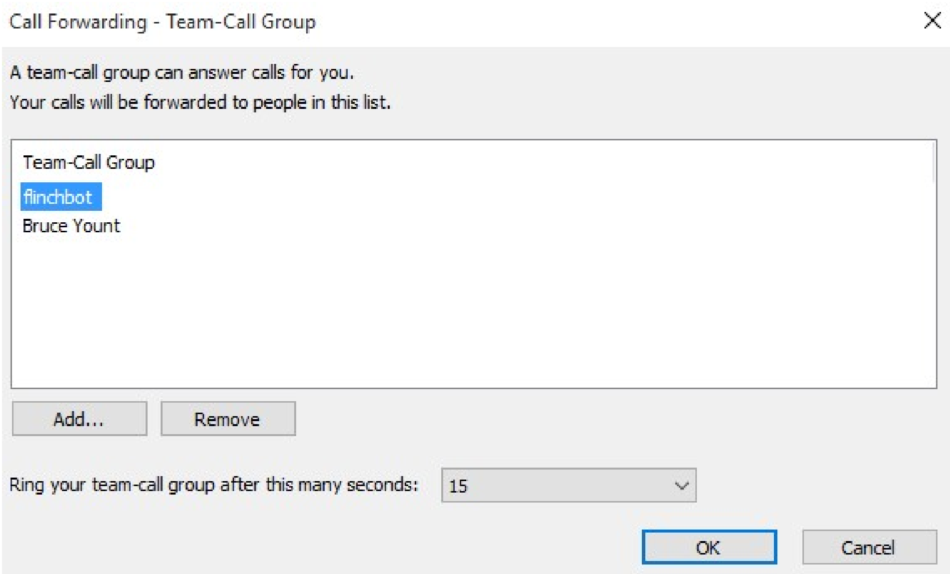
```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com /delayringteam:12
```

Figure A – 12

So what does the client show when you have this set to 12 seconds? It shows an incorrect value! Since the GUI is limited to what it will display, you can't always trust what you see in the client.

The client shows "15" when in reality the delay is 12 seconds.

Figure A – 13



Delegates

Setting delegates is similar to a Team Call group except a delegate has more features than a Team Call group. The primary additional feature is that a delegate can call on behalf of another user. So when the delegate places the call, it looks like it is coming from a different user than their own. You can also disable a delegate from receiving the inbound call and only be enabled for making an outbound call.

This is often referred to as the "Boss/Admin" feature.

Note that you can configure this via the Skype for Business client. SEFAUtil only needs to be used if you would like to edit these settings administratively.

Enabling a Delegate

To enable the delegate feature, use the `/simulringdelegates` parameter. Much like enabling Team Call, if you don't already have a delegate defined, then using `/simulringdelegates` has no effect. As such, when enabling delegates for the first time, be sure to add a delegate.

In the below command, I add flinchbot@flinchbot.com as a delegate to mtressler@flinchbot.com. I then enable the Delegates feature.

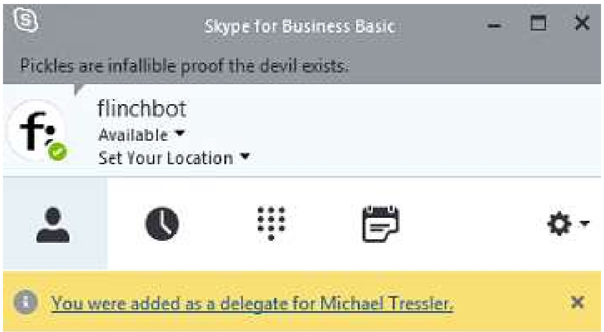
```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com
/adddelegate:flinchbot@flinchbot.com /simulringdelegates
```

Figure A – 14

```
C:\>SEFAUtil mtressler@flinchbot.com /server:skypepool.flinchbot.com /adddelegat
e:flinchbot@flinchbot.com /simulringdelegates
User Aor: sip:mtressler@flinchbot.com
Display Name: Michael Tressler
UM Enabled: False
Simulring enabled: False
Simultaneously Ringing Delegates: sip:flinchbot@flinchbot.com
```

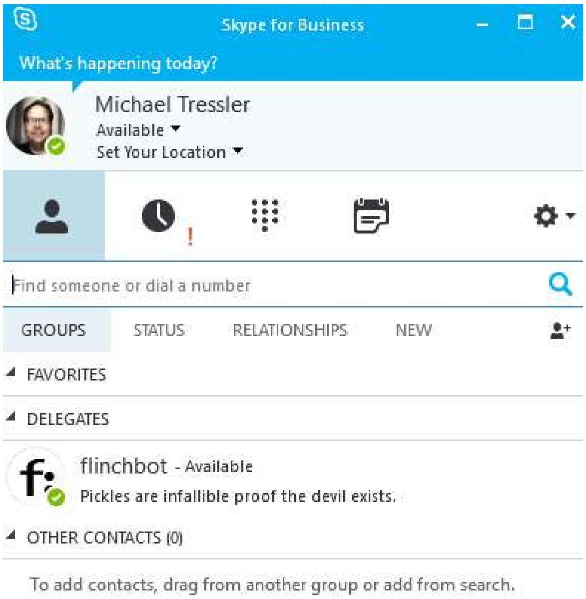
After this has been run, the user flinchbot@flinchbot.com will be notified that they have been configured as a delegate.

Figure A – 15



The client for mtressler@flinchbot.com also gets updated with a new Delegates group which lists all of the defined Delegates.

Figure A – 16



Adding Delegates

Once delegation has been enabled, you can add additional delegates by using the `/adddelegate` parameter. In the below example, I add byount@flinchbot.com as a delegate for mtressler@flinchbot.com.

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com
/adddelegate:byount@flinchbot.com
```

Figure A – 17

Removing Delegates

To remove a delegate, you use the `/removedelegate` parameter. In the example below, I remove flinchbot@flinchbot.com as a delegate for mtressler@flinchbot.com.

```
SEFAUtil mtressler@flinchbot.com
/server:skypepool.flinchbot.com
/removedelegate:flinchbot@flinchbot.com
```

Figure A – 18

```
C:\>SEFAUtil mtressler@flinchbot.com /server:skypepool.flinchbot.com /removedele
gate:flinchbot@flinchbot.com
User Aor: sip:mtressler@flinchbot.com
Display Name: Michael Tressler
UM Enabled: False
Simulring enabled: False
Simultaneously Ringing Delegates: sip:BYount@flinchbot.com
```

Index

This is the most comprehensive and useful guide to Enterprise Voice in Skype for Business Server 2015 ever written.

It's also the only one.

If you are responsible for the roll out or support of Enterprise Voice in Skype for Business Server 2015 (or Lync Server 2013), then this is the book for you. This book covers every core feature in detail, including hundreds of screenshots and PowerShell examples. A variety of topics are covered, which include adding gateways to your Topology and enabling brand new features like Shared Line Appearance.

Each chapter walks you through all of the background, detail, and steps required to fully understand Enterprise Voice. This book culminates in a chapter that walks you through a complete, real-world Enterprise Voice scenario which will help you apply the concepts in this book to your own Skype for Business environment.

This book provides detailed information on:

- Configuring Dial Plans
- Defining Voice Policies
- Deploying gateways and trunks
- Adding Routes
- Describing what the heck a PSTN Usage is
- Enabling voice features such as Call Park, Group Pickup, and Unassigned Numbers
- Creating Dial-in Conferencing numbers
- Configuring your end users
- Designing and planning for survivability
- How to test and troubleshoot Enterprise Voice
- A primer on Regular Expressions
- Detailed breakdown of call flow within Skype for Business

Note that this book is also valid if you are running Lync 2013.

If you are running anything older, you're hosed. But you probably already knew that.

Michael Tressler works for a major multinational corporation. His responsibilities include deploying Enterprise Voice globally and supporting and maintaining the related infrastructure.

He runs an active blog at www.flinchbot.com and should not be followed on Twitter at @flinchbot. Most of what he says there isn't related to any of this.

User Level: Intermediate to Advanced
ISBN: 978-0-1234567-0-1
$29.99 US