HEISON CHAK

voice over IP with Asterisk

Heison Chak is a system and network administrator at SOMA Networks. He focuses on network management and performance analysis of data and voice networks. Heison has been an active member of the Asterisk community since 2003 and will be delivering a tutorial on VoIP Principles and Practice at the USENIX '05 Annual Technical Conference.

heison@chak.ca

ASTERISK IS A SOFTWARE-BASED PBX that runs under Linux, *BSDs, and Mac OS X. It can also be used as a VoIP gateway and can provide a bridge to the PSTN (Public Switched Telephone Network, a.k.a. Plain Old Telephone Service). IAX (Inter-Asterisk eXchange), pronounced "eeks," is a protocol used by Asterisk as an alternative to SIP, H.323, etc., when connecting to other VoIP devices that support IAX.

This article describes some of the interesting things I have done using Asterisk and VoIP, told as the story of a business trip from my home in Toronto.

It was 12:15 in the morning and I had just gotten home from work. Realizing that I had not yet packed for my trip to San Francisco, I was a little worried about missing my early morning flight. I knew that snooze alarms no longer worked for me; I would often keep hitting the snooze button when the alarm went off, or I would just be so used to the sound that I wouldn't even hear it at all. Either way, I could wind up being late. The AGI (Asterisk Gateway Interface) script I had installed on my Asterisk box that serves wake-up calls solved both my problems with snooze alarms.

Wake-Up Calls

Before going to bed, I dial ext. 100 from any phone in the house to request a wake-up call, and the voice of Allison Smith (Allison is the voice of Asterisk) prompts me for the desired time. At 6:30 a.m. that morning, Asterisk called the auto-answer extension of my IP phone in the bedroom. Besides telling me that it was a wake-up call and announcing the current weather in Toronto, Allison also challenged me to repeat a four-digit number after her. If I failed to respond or if I hung up after three tries, Asterisk would call for help, playing back my own recorded voice begging her to wake me up on that same auto-answer extension. The last resort would usually get me the unpleasant voice of my mother.

Dynamic Contents

That morning, I managed to get out of bed before Asterisk escalated the call, yet I missed the weather report when Allison did the announcement. I wanted to double-check the weather in both Toronto and San

Francisco, as well as make sure my flight was on time before heading out to the airport, but my laptop was already packed away.

With the help of a PHP script on my Web server, I was able to look at reformatted contents from theweathernetwork.com and aircanada.com on my IP phone as I hit the service button. Calling ext. 102 invoked weather.agi, allowing me to get a more detailed weather report based on the three-digit airport code (e.g., 736 for SFO and 999 for YYZ). A TTS (Text-to-Speech) engine from Cepstral was used to announce the weather forecast of the requested city. After hearing that Toronto's temperature was in the minus 40s, I was happy to find out San Francisco would be in the mid-60s throughout the week.

IAXy

Given it was a Monday morning, the check-in line-up wasn't so bad. However, I was not making friends with airport security, especially when they found this little blue thingie in my luggage and wanted me to explain what it was: "Okay, this blue thingie is called IAXy (pronounced 'eek-see'); it's an analog telephone adapter that allows telephone calls to be made over the Internet with a regular telephone, and the 'y' in IAXy doesn't stand for anything, it's just a product from Digium (*http://www.digium.com*) with a cool name."

Then, I heard a very familiar voice saying, "Yeah, this is similar to what Vonage and other voice-over-broadband providers are offering, but he is doing it with some open source software." I turned around and there was my boss, who was traveling with me. I was glad he was there, or there would have been a good chance I'd have missed my flight.

Toll-Free Service

I promised to call home when I landed, but my cell phone had very poor reception at the baggage claim. So I went to a pay phone and dialed my home number without inserting any money…and it worked! That was my first time calling my toll-free number—my VoIP provider was charging $0.03/min. While most VoIP providers charged a monthly access fee plus usage, I chose one that supported IAX on a prepaid account with no monthly fee. Calls made within US48 were $0.03/min, while calls made within Canada were only $0.02/min. My VoIP provider also provided a Web page for customers to monitor their usage.

After the call had terminated on the PSTN in Chicago, it was delivered to me via IP over the Internet to my Asterisk server behind a firewall. The voice on IVR (Interactive Voice Response) sounded very much the way it does on a landline.

I entered my wife's extension number, and the IVR responded with, "Please wait while I connect your call." My wife and I talked for a few minutes before hanging up. Next, I tested the calling card extension and entered my boss's mobile number. It freaked him out when he saw my home number showing up on his mobile when I was standing right next to him.

Telemarketers and Junk Faxes

As soon as I had network connectivity for my laptop, I wanted to check my email and there were a few .wav voicemail attachments from the PBX. Upon receiving voice messages, Asterisk would send a notification to email with an attachment of the voicemail file. Clicking on the attachments allowed me to listen to the recording without dialing into the PBX. I was also interested in other calls that I missed during the flight, and that's when my PHP-based CDR (Call Detail Record) report came in handy. Through Mozilla, I queried against my

PostgreSQL-based CDR and saw multiple entries of the same toll-free number. Taking a closer look at the CDR, I suspected it was a telemarketing firm who had made many attempts in the past few days but kept getting trapped in the IVR menu until the telemarketing agent finally hung up.

Blacklisting such a telephone number would result in a fast busy signal by Asterisk upon receiving CID (Caller ID) matched calls. In most cases, my number was removed from their database within two more attempts. Incoming faxes could also be blocked with this technique, after identifying the CID from which the junk originated.

Free Long Distance

Most hotels in the Bay Area offered complimentary high-speed Internet connectivity, and I was looking forward to making free long distance calls over IP. I connected my IAXy to the RJ45 outlet and moved the telephone in my room over to the IAXy from the wall jack. Unfortunately, there was no dial tone when I picked up the handset.

To verify that my problem was not related to the IAXy, I connected my laptop to the network and soon found out that I needed to register to the hotel's proxy before using the high-speed connection. Knowing that I could not use the IAXy, I started up iaxcomm (an IAX-based soft IP phone, http://iaxclient.sourceforge.net/iaxcomm/) in Linux and used the microphone and speaker on my laptop to chat with my wife. Unlike the toll-free calls I made earlier that cost me $0.03/min, calls made with iaxcomm were completely free, even if I were to call someone in Toronto with iaxcomm, since Asterisk was providing a bridge to the PSTN. During the conversation, we noticed some acoustic echo issues, but these were resolved when I switched over to a headset with a built-in microphone.

Dialplan

To demonstrate how Asterisk can be used to filter unwanted numbers (a.k.a. "the 'ex-girlfriend' feature"), let's examine the dialplan, stored in the file extensions.conf. The Asterisk dialplan file consists of a collection of contexts. Each context consists of a collection of extensions. The priority of an extension specifies the order in which it is executed by Asterisk.

```
[context]
exten => <extension>,<priority>,<application>(<args>)
```

In the following example there are two contexts, [incoming] and [real_extensions]. Extensions 100, 101, and 102 have been included by the [incoming] context using the include command.

When the [incoming] context answers a call with extension "s," it executes each line of exten => s, according to the priority. It performs Caller ID number matching. If there is a match, it plays a fast busy tone. If all matches fail, it moves on to priority 30, plays the IVR greeting, and waits for a number to be entered. If a caller enters 101, it will then ring my IP phone for 20 seconds before passing the caller to the voicemail application with the unavailable flag, "u."

```
[incoming]
exten => s,1,Wait,1                  ; Wait a little for CallerID
exten => s,2,Answer                  ; Asterisk answers the line
exten => s,3,DigitTimeout,5          ; Set Digit Timeout to 5sec.
exten => s,4,ResponseTimeout,10      ; Set Response Timeout to 5sec.
;
; Black listing numbers
```

```
;
; Jump to priority 20 if CID matches, else goto priority 6
exten => s,5,GotoIf($[${CALLERIDNUM} =  18668493243]?20:6)
; Jump to priority 20 if first 6 digits of CID match, else goto priority 30
exten => s,6,GotoIf($[${CALLERIDNUM:0:6} =  647722]?20:30)
;
; Banned numbers
;
exten => s,20,Congestion    ; Plays the fast busy tone
;
; Auto attendant says,
; "Thank you for calling, if you know the 4 digit extension . . . "
;
exten => s,30,BackGround(ivr-greeting)
include => real_extensions
[real_extensions]
exten => 100,1,AGI,wakeup.agi
exten => 101,1,Dial(SIP/cisco1,20,Tr)
exten => 101,2,Voicemail(u101)
exten => 102,1,AGI,weather-station.agi
```

Conclusion

I've described a few of the interesting ideas that can be accomplished with Asterisk and some programming. Most of these ideas are based on the Asterisk dialplan and AGI (Asterisk Gateway Interface). While the dialplan handles routing of calls through Asterisk, the AGI provides hinges for the logic to extend beyond the dialplan with other programming languages. For rapid development/deployment, most people use Perl and PHP to write AGIs. However, if you are concerned about performance, you may want to execute only AGIs that are written in C.