

Camal Şahverdiyev

**FreeSWITCH - IP üzərindən səsin qeyri
məhdud imkanları**

Müəllif: Camal Şahverdiyev

Oxucuya müraciət:

Bu sahə üzrə Azərbaycan dilində kitab ilk dəfə nəşr olunduğundan istifadə edilən termin və sözlər məlumatın daha anlaşıla bilən olması üçün tətbiq edilmişdir. Kitabın daxilində səhv aşkar etsəniz, xahiş edirik, sərt şəkildə tənqid etməyəsiniz. Yalnız söz və ya sintaksis səhvini gördüyüiniz halda, bookcorrector@gmail.com mail ünvanına yazmağınız xahiş olunur. Bununla növbəti kitabların daha mükəmməl edilməsinə yardımçı olarsınız.

Bütün müəllif hüquqları qorunur. Kitabın daxilində eks olunan məlumatların yayılmasına, çapı, surətinin çıxarılması və ya digər bir şəkildə istifadə olunması yalnız müəllifdən razılıq alındıqdan sonra mümkündür. Məlumat qeyd olunan məqamlar nəzərə alınmadan istifadə edilərsə, müvafiq qanunvericilik üzrə tədbirlər tətbiq olunacaq.

ISBN: 978-9952-8302-5-5

Kitabdan istifadə qaydaları

Aşağıdakı açıqlamalar kitabın mütaliəsində oxucuya yardımcı olacaq:

Əsas başlıq - **Bold və böyük hərfələr**

Əsas başlıq 1-ci dərəcəli alt başlıq - **Nisbətən kiçik həcmli bold font**

Əsas başlıq 2-ci dərəcəli alt başlıq - **1-ci dərəcəli alt başlıq nisbətən kiçik və bold olmaqla**

Əmrlər bold qeyd olunub. Əgər hansısa faylin içərisində olan sintaksisdən danışılırsa, öncədən faylin adı və tərkibinə əlavə ediləcək sətirlər bildirilir.

Qeydlər altdan xətt və bold edilmişdir - **Qeyd**:

```
# - İstənilən UNIX/Linux əməliyyat sistemində faylların içində şərh üçün istifadə edilir.  
Simvoldan sonraki sözlər oxunmur.  
/* şərh */ - DNS BIND-da və PHP programlaşdırma dilində yazılmış kodlarda göstərilən simvolların daxilində olan istənilən yazı şərhdür.  
// - DNS BIND-da və PHP programlaşdırma dilində yazılmış kodlarda göstərilən simvollardan  
sonra olan ixtiyari yazı şərhdür.  
; - DNS BIND-da sətirin sonu deməkdir.  
<!-- XML daxili şərh -->
```

Kitab üçün size nələr tələb edilir.

FreeSWITCH yüklənməsi üçün 1 ədəd FreeSWITCH, 1 ədəd Asterisk və bir ədəd CUCM(Cisco Unified Communications Manager) server ya da PC. Üç ədəd SIP telefon bu ya fiziki ya da soft ola bilər. Həmçinin elə də sərt tələb yoxdur ki, hansısa SIP providerdən hesab alasınız.

Kitab kimin üçündür

Bu kitab FreeSWITCH inzibatçılar üçündür hansı ki, pulsuz VoIP program təminatı haqqında daha ətraflı öyrənmək istəyirlər. Əgər siz artıq FreeSWITCH istifadə etmək istəyirsinizsə, kitabdan daha da ətraflı praktik məlumatlar əldə edə bilərsiniz ki, real həyatda istifadə edəsiniz.

Oxucu tərəfindən kitabın başa düşülməsi üçün tələb edilən biliklər:

1. UNİX/Linux əməliyyat sistemlərində dərin biliklərə sahib olmalı
2. CCNA şəbəkə səviyyəsinə sahib olmalıdır
3. VoIP-dən orta biliklərə sahib olmalıdır

7 FreeSWITCH-in arxitekturası

- 8 Telefon dünyasında çevriliş
- 9 FreeSWITCH-in üstünlüyü
- 11 Vacib modullar - Endpoint və Dialplan
- 14 Çətin programlar daha da asanlaşdırır

22 Kompilyasiyası və yüklənməsi

- 23 FreeSWITCH mühitinin qurulması
- 26 FreeSWITCH-in UNIX/Linux məşinlərdə yüklənməsi və qurulması
- 34 FreeSWITCH-in Windows 2012 serverdə kompilyasiyası və yüklənməsi

49 Quraşdırma nüsxələrində sınaqların edilməsi

- 50 VoIP və FreeSWITCH-in vacib konsepsiyası
- 54 FreeSWITCH command line interfeysin istifadə edilməsi(fs_cli)
- 57 FreeSWITCH-ə qoşulması üçün telefonun quraşdırılması
- 65 CUCM SIP trunk-da istifadə etməyimizdə tələb ediləcək Cisco İP communicator quraşdırılması
- 71 Dialplan nüsxəsinin sınaqdan keçirilməsi

75 SIP və istifadəçi qovluğu

- 76 FreeSWITCH istifadəçi qovluğunun başa düşülməsi.
- 78 İlk dəfə üçün FreeSWITCH istifadəçi qovluğunun quraşdırılması və açıqlanması.
- 85 FreeSWITCH-in xidmət təcizatçılarına qoşulması(FreeSWITCH – Aserisk SIP trunk, FreeSWITCH – CUCM SIP trunk).
- 115 Gateway olmadan zənglərin edilməsi
- 116 SIP profillərin və istifadəçi agentlərin qısaca müzakirə edilməsi.

118 XML Dialplan-ın başa düşülməsi

- 119 FreeSWITCH XML Dialplan elementləri
- 123 Zəng ayaqları və kanal dəyişənləri
- 133 Yeni extension(genişlənmə)-in yaradılması
- 136 Vacib Dialplan programları
- 141 Dialstring formatları

144 XML IVR və Phrase macroslarının istifadə edilməsi

- 145 IVR motoruna qısa baxış
- 145 IVR XML quraşdırma faylı
- 147 IVR menyünün təyin edilməsi
- 150 IVR menyusunun təyinat yerləri
- 153 Zənglərin sizin IVR-lara yönlendirilməsi

153 IVR-a əlavələr
154 IVR vasitəsilə phrase-ların istifadə edilməsi
159 Təkmilləşdirilmiş yönləndirilmə

161 LUA vasitəsilə Dialplan Script yazma

162 Lua ilə başlayaq
164 Səs proqramlarının qurulması
175 Irəliləmiş IVR konsepsiyaları
187 Script tipləri

189 Irəliləmiş Dialplan konsepsiyaları

190 Dialplana yenidən baxış
193 Ümumi Dialplan konsepsiyaları
197 XML Dialplan moduluna analiz
208 Tələlərdən qacış
209 XML Dialplan proqramları
214 İstifadə edilən dəyişənlər
222 Dəyişənlərin zəng başlıqları ilə ötürülməsi

228 Statik XML quraşdırılmaları sərhədinin aşılması

229 mod_xml_curl
237 Dillərin əlaqələndirilməsi vasitəsilə quraşdırılmaların dinamik generasiyası
238 Zənglərin API-la CLI-dan edilməsi
240 Əmrələrin yerinə yetirilməsi üçün ESL-in istifadə edilməsi

243 FreeSWITCH-in kənardan idarə edilməsi

244 Event sisteminə ümumi baxış
245 Event sistem arxitekturası
246 Event socket quraşdırılmaların edilməsi
258 Event socket kitabxanası
263 Təcrübədə olan eventlər

268 mod_httapi vasitəsilə WEB bazalı zəngin idarə edilməsi

269 HTTPAPI sintaksisi
278 mod_httapi quraşdırma faylı
284 HTTAPI-da demo IVR

289 NAT emal edilməsi

290 NAT-a qısa giriş
292 NAT-ın dörd tələsi
293 NAT-ı aşmağa kömək edən FreeSWITCH quraşdırmları

299 NAT vəziyyətlərində FreeSWITCH-in daha da yaradıcı istifadə edilməsi

302 **VoIP Təhlükəsizliyi**

303 Şəbəkə səviyyəsinin qorunması
312 SIP siqnal səviyyəsinin qorunması
316 Səsin qorunması
319 Şifrlərin qorunması

322 **Qabaqcıl imkanlar və əlavə oxumalar**

323 Çox istifadəçili konfranslar(mod_conference)
329 Real-time billing (mod_nibblebill)
344 Alternativ son nöqtələr
348 Web GUI-lər və digər proektlər

352 **Sınaqdan keçirilmiş əlavələr**

353 FreeBSD əməliyat sistemində BlueBox yüklənməsi və qurulması
366 BlueBox üzərində FreeSWITCH üçün Multi Tenant Domain qurulması
373 FreeBSD əməliyyat sistemində SIPvicious qurulması və sınaqdan keçirilməsi
374 FreeSWITCH üzərində səslərin yazılması
375 FreeSWITCH recover və Linux HA vasitəsilə Clusterin qurulması
381 Mod_Event_Socket dili vasitəsilə PHP və Python-da misallar
383 FreeSWITCH SIP istifadəçiləri MySQL-də [Mox_XML_CURL]
387 FreeSWITCH vasitəsilə Mod_callcenter üzərində növbələmə
396 API vasitəsilə FreeSWITCH serverdən statusların alınması

400 **FreeSWITCH Onlayn cəmiyyəti**

401 FreeSWITCH mail siyahıları
402 IRC vasitəsilə real vaxtda əlaqə
404 FreeSWITCH əsas WEB səhifə və WIKI
405 Hər il keçirilən ClueCon açıq qaynaqlı yaradıcı konfransı

406 **Asterisk-dən FreeSWITCH-ə miqrasiya edilməsi**

407 FreeSWITCH və Asteriskin işə salınması və dayandırılması
407 Jurnalların ətraflı araşdırılması səviyyələri

1-ci başlıq

FreeSWITCH-in arxitekturası

FreeSWITCH telefon təkmilləşməsi müddətində yaradılmış program təminatıdır. Bu güclü program təminatının arxitekturasına baxmazdan önce, dünya səs əlaqələrinin quruluşunu aşdırmanız daha yaxşı olar.

Başlıqda biz aşağıdakılari aşadıracayıq:

- Telefon dünyasında çevriliş
- FreeSWITCH-in üstünlüyü
- Vacib modular - Endpoint və Dialplan
- Çətin programlar daha da asanlaşdı

Telefon dünyasında çevriliş

Adətən telefonun necə işləməsi hər kəs üçün sırr olaraq qalır. Biz sadəcə telefonumuzu divardakı şunura qoşuruq və o işləməyə başlayır. Telefonda olan çevriliş ona görə başladı ki, özündə olan gizlilikləri kənara çıxarsın. Artıq istənilən insan gündəlik istifadə etdiyi telefon sistemini özü qura və hətta öncəkində olan funksionallıqdan qat-qat artığına havayı sahib ola bilər. Bəzi insanlar hətta FreeSWITCH-dən səxsi qazancı olaraq da istifadə etməyə başlayır. FreeSWITCH bütün bu işləri asanlaşdırmaq üçün yaradılmışdır və biz ardıcılıqla onun arxitekturasını araşdıracaqıq.

Burda gördüğünü konsepsiyalardan sizə mənasız görünənləri olarsa narahat olmayıñ və bu başlığı təkrar-təkrar oxuyun çünki VoIP həddən ziyadə çətin bir mövzudur və başa düşülməsi üçün tam araşdırılmalıdır. Artıq 5-ci başlıqdan sonra XML dialplani başa düşəcəksiniz. Anlamağa başlayacaqsınızki, sizin VoIP və FreeSWITCH-lə bağlı bilikləriniz necə təkmilləşməyə başlayıb. Sonra 10-cu başlığı bitirib yenidən bura qaydırıb oxusanız, FreeSWITCH-in kənardan idarə edilməsini tam anlayacaqsınız. Bu halda sizin FreeSWITCH və VoIP haqqında köklü bilikləriniz olacaq. Bu konsepsiyaların anlaşılması üçün özünüzə kifayət qədər vaxt verin və müəyyən vaxtda sonra görəcəksiniz ki, siz artıq yetişmiş inzibatçısınız.

Telefonlar və telefon sistemləri (misal üçün telefon komutatorları və ATŞ "Avtomatik Telefon Stansiyası"-lər) çox çətindir və uzun illər ərzində fərqli usullarla təkmilləşdirilmişdir. Ingiltərə və USA-də telefonun daha təkmilləşdirilmiş növü ən-ənəvi analoq telefonlarıdır hansı ki, biz POTS (Plain Old Telephone Service) adlandırırıq. Ən-ənəvi Ma-Bell telefonlarından tutmuş uzun məsafəli naqilsiz telefonlaradək (hər birimizdə olan gündəlik telefonlar) eyni baza texnologiyalarından istifadə edir. Son 10-15 il ərzində kompyuterlərlə telefonlar arasında bir əlaqə yarandı və nəticədə baha olmayan POTS xətlərində işləyən mobil telefonlarla, VoIP telefonlar (Həmçinin internet telefonlar adlanır) istehsal edilməyə başlandı.

FreeSWITCH bu qarışiq telefon texnologiyalarının siyahısına aiddir. Fərqli texnologiyalara sahib olmalarına baxmayaraq, onları əlaqələndirmək imkanına sahibdir. FreeSWITCH həmçinin kompyuter programları vasitəsilə edilən zənglər arasında körpü işini görə bilər və hətta sizə şərait yaradır ki, daxili imkanlarından yararlanaraq bu müştəri programlarını siz özünüz yazasınız. FreeSWITCH program təminatıdır hansı ki, Windows və istənilən UNIX/Linux (istənilən BSD, MacOS və hətta Solaris) tipli əməliyyat sistemində işləyə bilir. Bu o deməkdir ki, siz FreeSWITCH-i öz evdəki PC-niz və ya tələbdən asılı olaraq zəng sayına görə hansısa serverə yükləyə bilərsiniz. FreeSWITCH-in yüklenməsi daha ətraflı 2-ci başlıqda açıqlanır (Bu işi baza arxitekturasını öyrəndikdən sonra edəcəyik).

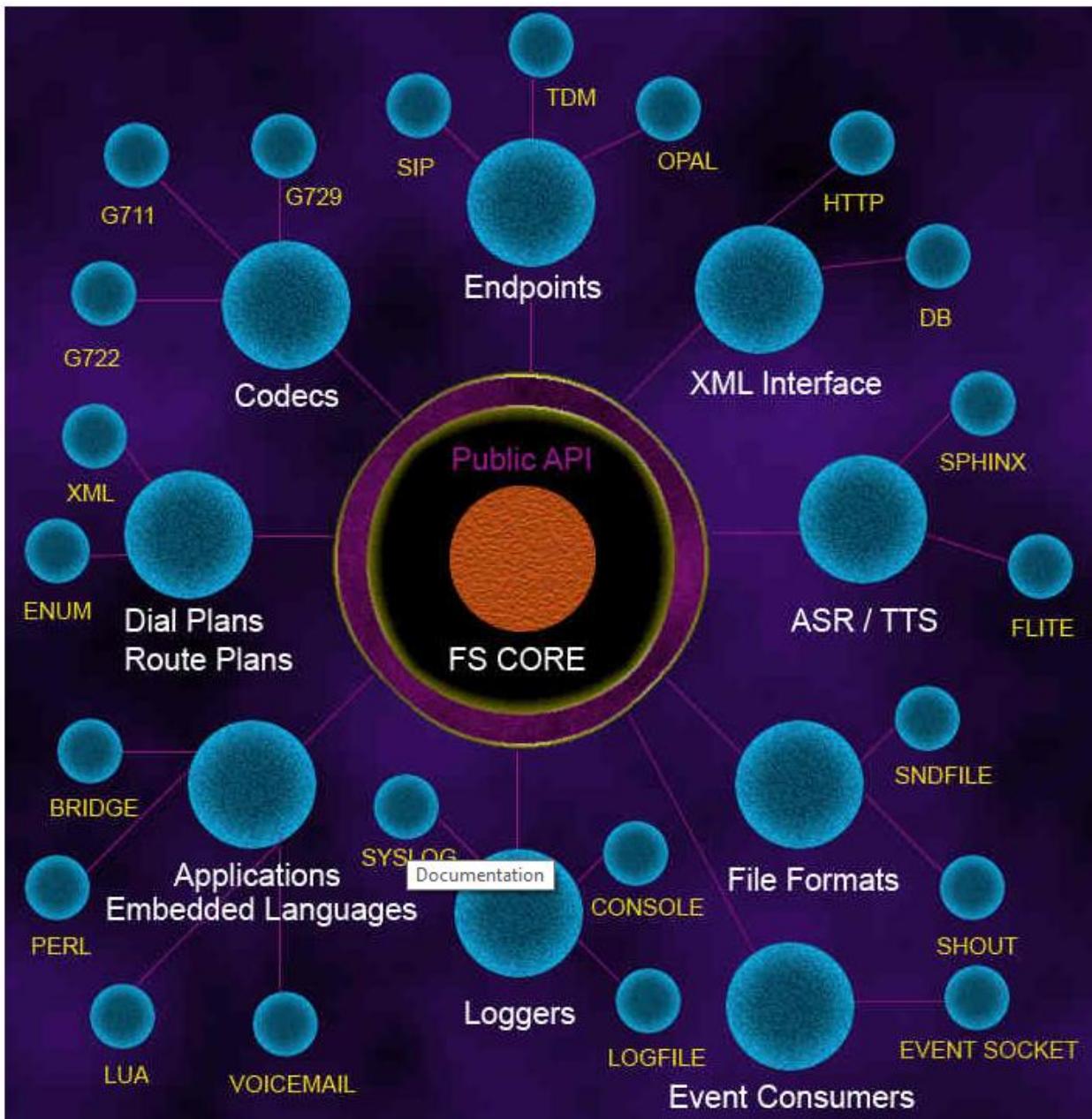
FreeSWITCH-in üstünlüyü

FreeSWITCH-in konstruksiyası modul yaratmaq üçün, genişlənilmək üçün dayanıqlı bir komutasiya özəyinin üzərində qurulmuşdur. O həmcinin programçılar üçün etibarlı bir interfeysə sahibdir hansı ki, sistemə əlavə etmək və idarə etmək üçün çox mükəmməldir. FreeSWITCH-in fərqli elementləri bir-birindən asılı olmayaraq işləyirlər və bütün bu elementlərin necə işləmələrinin haqqında məlumatınız olmasa da belə ümumi sistemi idarə edə biləcəksiniz. Siz həmcinin yüklənilə bilən modulların sayəsində FreeSWITCH imkanlarını genişləndirə bilərsiniz hansı ki, kənar texnologiyaları ozekle əlaqələndirə bilir.

FreeSWITCH öz özəyinin ətrafında çoxlu tip fərqli modullara sahibdir hansı ki, quruluşca yer kürəsinin ətrafında olan sputniklər kimi görünüşə sahibdir. Siyahı aşağıdakı kimidir:

Module tipi	Məqsədi
Endpoint	SIP/H.323 və POTS liniyaları kimi, telefon protokolları
Application	Musiqinin oxunulması və ya datanın təyin edilməsinə uyğun olaraq işlər görür
Application	Programlaşdırma interfeysi (API) funksiyani eksport edir hansı ki, tekst daxil edilməsini alır və tekstin çıxışını qaytarır. Bu fərqli modullar ya da kənar qoşulmalar üçün istifadə edilə bilər.
Automated Speech Recognition (ASR)	Səs təyinatı sistemləri ilə interfeys
Chat	Əksər chat protokolları arasında körpü yaradır və mübadilə edir
Codec	Audio formatlar arasında tərcümə işini görür
Dialplan	Zəngin məlumatını analiz edir və zəngin hara yönləndirilməsinə qərar verir
Directory	Məlumat xidmətləri qovluğuna qoşulur (Misal olaraq ümumi özək API sahib olan LDAP)
Event handlers	FreeSWITCH-i idarə etmək üçün kənar programlara izin verir
File	Fərqli səsli fayl formatlarında olan səslərin açılması və işə salılması üçün interfeys verir
Formats	Fərqli formatlarda olan audio faylların işə salılması
Languages	Zənglərin idarə edilməsi üçün, istifadə edilən programlaşdırma dili interfeysi
Loggers	Sistem jurnalları ya da jurnal fayllarının konsola jurnallamasını idarə edir
Say	Bəzi dillərdə olan audio fayllarla birgə olan sətirlər geriyə qayıdışı təmin etmək üçün istifadə edilir ki, geriyə telefon nömrəsi, günün vaxtı və ya hansısa digər məlumat bildirilsin.
Text-To-Speech (TTS)	Tekst-dən-səsə motoru ilə olan interfeyslər
Timers	Programlarda olan POSIX ya da Linux kernel vaxtlaması
XML	Call Detail Record(CDRs)-lar üçün XML istifadə edilən interfeyslər, RADIUS, CURL, LDAP, RPC və ya SCGI

Aşağıdakı şəkil FreeSWITCH arxitekturasının hansı quruluşa və FreeSWITCH core-unun orbitində nə qədər modullara sahib olduğunu göstərir:



Fərqli modul interfeyslərinin birləşdirilməsi nəticəsində FreeSWITCH-i IP telefonların qoşulması, telefon liniyalarının və IP-yə əsaslanan telefon xidmətlərini quraşdırıa bilərik. O həmcinin system istifadəçi menyusundan audio formatları və interfeysləri tərcümə edə bilər hansı ki, siz özünüz yarada bilərsiniz. Siz həmcinin işlək FreeSWITCH serveri digər maşından idarə edə bilərsiniz. Gəlin bəzi vacib modullara ciddi baxış keçirək.

Vacib modular - Endpoint və Dialplan

Endpoint modulların vacibliyi kiritikdir və bəzi açar imkanların əlavə edilməsi bizim günlərimizdə FreeSWITCH platformasını güclü edir. Bu modulların əsas rolu həmişəki ümumi istifadə edilən komunikasiya texnologiyasını götürmək və ümumi adlandırdığımız **session** içində normallaşdırmaqdır. FreeSWITCH ilə müəyyən bir protocol arasında qoşulmaya **Session** deyilir. FreeSWITCH üzərində çoxlu Endpoint modulları gəlmişdir hansı ki, müxtəlif protokolları realizasiya edir(**SIP, H.323, Jingle "Google Talk"**). Bu modullardan ən məhşuruna **mod_sofia**-ya biz müəyyən bir vaxt ayıracayıq.

Sofia-SIP(<http://sofia-sip.sourceforge.net/>) açıq kodlu proektdir və Nokia tərəfindən maliyyələşdirilir hansı ki, **Session Initiation Protocol(SIP)** üçün programlaşdırma interfeysi verir. Biz bu kitabxananı FreeSWITCH-də **mod_sofia** adlanan modulda istifadə edirik. Bu modul bütün müraciət edənləri FreeSWITCH-də qeydiyyatdan keçirir, doğma FreeSWITCH konstruksiyasını SIP konstruksiyasına tərcümə edir və geriyə eyni qaydada tərcümə edir. Quraşdırma informasiyaları mərkəzi FreeSWITCH quraşdırma fayllarından alınır və **mod_sofia**-ya izin verir ki, təyin edilmiş istifadəçi parametrləri ilə qoşulma parametrləri detallarını yüklesin. Bu FreeSWITCH-ə izin verir ki, SIP telefonlar və alətlərdən gələn qeydiyyatı qəbul etsin, digər SIP Endpoint-lərdə qeydiyyatdan keçsin(Service Providerlər kimi), notifikasiyalar yollasın və telefonlara xidmətləri(VoiceMail kimi) təqdim edə bilsin.

Qeyd: SIP protokolu RFC(request for comment)-də qeydə alınmışdır. Əsas RFC <http://www.ietf.org/rfc/rfc3261.txt> linkindən əldə edilə bilər.

SIP qoşulması FreeSWITCH ilə digər SIP alət arasında uğurla yaradıldıqdan sonra, o FreeSWITCH-də aktiv sessiya kimi görünəcək. Əgər zəng daxilidirsə o, **interactive voice response(IVR)** menyusuna yönləndirilə ya da körpülənə, sixib musiqi dinləmə, bir və ya bir neçə genişlənmə və.s ola bilər. Gəlin misal olaraq deyək ki, SIP telefon qeydiyyatdan keçmişdir və ümidi 2000 rəqəmli telefon 2001 telefonuna cavab alacağını gözləyərək zəng edir.

Öncə şəbəkə üzərindən SIP telefonu zəng qurulması mesajını **mod_sofia**-ya ötürür(**mod_sofia** uyğun mesajlar üçün qulaq asır). Mesajın alınmasından sonra, **mod_sofia** öz növbəsinə verilən məlumatları parçalayır və zəngi core statuslu FreeSWITCH maşına ötürür. Sonra statuslu(FreeSWITCH core) maşın zəngi **ROUTING** statusuna ötürür.

Növbəti addım zəng edən Endpoint üçün quraşdırma datasına əsaslanan, Dialplan modulun ünvanını tapmaqdır. Əksər istifadə edilən Dialplan modulu XML Dialplan moduldür. Bu modul FreeSWITCH-lə mərkəzi XML registrində olan instruksiya siyahısına baxış üçün nəzərdə tutulmuşdur. XML Dialplan modulu XML genişlənmələri obyektlərini axtarış şablonlarına əsaslanaraq **regex**-lə parçalayacaq.

2001 nömrəsinə zəng etməyə cəhd etdiyimizə görə, ümid edirik ki, XML test etdiyimiz 2001 rəqəmini destination_number sütununda tapıb ona uyğun olan route işini edəcəyik. Dialplan bir genişlənmə ilə məhdudlaşdırılmış. Əslində siz 5-ci başlıqda XML Dialplan-ın başa düşülməsi üçün extension terminini həddən artıq çox oxuyacaqsınız. XML Dialplan modulu öz növbəsində zəng üçün iş siyahısını düzəldir. Hər üst-üstə düşən genişlənmənin, zəng siyahilanmasında olan uyğun işi yerinə yetiriləcək.

Təsəvvür etsək ki, FreeSWITCH ən azı bir genişlənmə tapacaq, XML Dialplan-ı sessiya obyektinə 2001-ə zəng etmək üçün tələb edilən infromasiyanı əlavə edəcək. Artıq bu qaydalar birlikdə olduqdan sonra, zəng sessiyasının status **ROUTING**-dən dəyişib **EXECUTE** olur harda ki, növbəti emaledici **ROUTING** statusu zamanı əldə etdiyi instruksiyalarla işini görərək yerinə yetirir. Bu həmin yerdir hansı ki, program interfeysi şəklə daxil olur.

Sessiyaya əlavə edilən hər bir instruksiya programın adı, argument verilənləri şəklində əlavə ediləcək hansı ki, o da həmin programma ötürülləcək. Misalımızda istifadə edəcəyimiz bridge programı olacaq. Bu programın məqsədi çıxış qoşulması ilə digər sessiyanın yaradılması və sonra iki sessiya arasında birbaşa audio mübadiləsinin aparılmasıdır. Bridge üçün verdiyimiz argument **user/2001** olacaq hansı ki, 2001 genişlənməsi üçün zəng generasiya edilməsində ən asan yol olacaq. **2001** üçün Dialplan yazısı aşağıdakı şəkildə ola bilər:

```
<extension name="example">
    <condition field="destination_number" expression="^2001$">
        <action application="bridge" data="user/2001"/>
    </condition>
</extension>
```

Genişlənmə example adla adlandırılmışdır və o bir ədəd uyğunluq şərtinə malikdir. Əgər şərt yerinə yetirilərsə, yerinə yetirilməsi üçün bir program təminatı olacaq. Adı dildə, adı çəkilən genişlənmə belə göstərilə bilər: Əgər zəng edən 2001 yaşarsa, o əlaqəni zəng edənlə son nöqtə(Yəni telefon) 2001 arasında qurur. Necə baş verməsinə baxaq.

Artıq instruksiyanı sessiyanın daxilinə yeritdikdən sonra, sessiyanın statusu dəyişib **EXECUTE** olacaq və FreeSWITCH core-u tələb edilən iş üçün yığılmış məlumatları istifadə edəcək. İlk olaraq susmaya görə olan status emaledicisi **user/2001**-də bridge işə salmaq üçün əmri parçalayacaq və sonra bridge programına istiqamətlənəcək ki, onun içindən **user/2001** datanı ötürsün. Bu gətirib ona çıxaracaq ki, FreeSWITCH core-u tələb edilən tip üçün yeni çıkış sessiyası yaradacaq. İstifadəçi 2001 həmçinin SIP telefondur ona görə də, **user/2001** çevriləcək SIP zəng sətirinə hansı ki, **mod_sofia**-ya yeni çıkış seansının yaradılması üçün ötürülləcək.

Əgər bu yeni sessiya üçün quruluş uğurlu olarsa, artıq FreeSWITCH core-unda iki sessiya olacaq. Bridge programı bu yeni sessiyani və original sessiyani(zəng edənin telefonu) götürəcək və bununla bridge funksiyasına zəng edəcək. Bu artıq 2001 telefonunda real insan tərəfindən cavablandırıldığı hal olan kimi, audio axının hər iki tərəfdə gedisiyi üçün şərait yaradır. Əgər bu istifadəciyə çatmaq mümkün olmazsa ya da məşğul olarsa, timeout (yəni imtina) baş verəcək və geriyə zəng edənin telefonuna yerinə uyğun olan ismaric qaytarılacaq. Əgər zəng cavabsızdırsa ya da məşğuldursa, o halda çoxlu yönləndirmə imkanları mövcuddur(zəng yönləndirilməsi ya da voicemail daxil etməklə).

Bütün bu ardıcılıq adı telefon dəstəyinin qaldırılması və **2 0 0 1** rəqəminin yığılması ilə baş verir. FreeSWITCH bütün SIP çətinliyini öz üzərinə alır və onu ümumi məxrəcədək asanlaşdırır. Burdan da o işimizi elə asanlaşdırır ki, bir instruksiya ilə 2000 telefonundan 2001 telefonuna qoşulmayı quraşdırıra

bilirik. Əgər biz 2001 telefonundan 2000 telefonuna geriyə zəng edilməsi üçün şərait yaratmaq istəsək, biz digər veriləni Dialplan-a aşağıdakı qaydada əlavə edə bilərik:

```
<extension name="example 2">
    <condition field="destination_number" expression="^2000$">
        <action application="bridge" data="user/2000"/>
    </condition>
</extension>
```

Bu ssenaridə Endpoint modulu SIP-i FreeSWITCH sessiyasına çevirdi və Dialplan modulu XML-i genişlənməyə çevirdi. Bridge programını çıkış zəngi yaranan və adı application/data cütlüyünün içində audio-ya qoşulan çətin bir koda çevirdik. Hər iki Dialplan və application modul interfeysi FreeSWITCH-in növbəli sessiyaları ətrafi üçün yaradılmışdır. Ona görə də, bu quruluş nəinki təkcə istifadəçi səviyyəsində bizim işimizi asanlaşdırır həmçinin Dialplan işinin araşdırılmasını son zəng üçün asan edir. Bu oxşarlıqla əsasən sabahısı gün yeni bir Endpoint modulu tərtib elədikdə(Misal olaraq Skype modulu) biz öncə təyin elədiyimiz programlar və Dialplan modullarını yenidən istifadə edə bilərik. Eyni məntiq **Say**, **Automatic Speech Recognition(ASR)**, **Text-to-Speech(TTS)** modulları və digərlərinə aiddir.

Ola bilər ki, siz Endpoint-in özünün təqdim elədiyi spesifik data ilə işləmək istəyəsiniz. Misal üçün SIP-də bir neçə təsadüfi başlıqlar və həmçinin SIP paketlərdə bir neçə bit maraqlı data mövcuddur. Bu problemi kanala dəyişən əlavə etməklə həll edirik. Kanal dəyişənlərini istifadə edərək, mod_sofia bu təsadüfi mənaları kanalın SIP datasında görünən kimi, təyin edə bilir hansı ki, siz onları adları ilə öz Dialplan kanalı ya da programınızdan əldə edə bilərsiniz. Beləliklə biz öz biliklərimizi bu spesifik dəyişənlərlə SIP Endpoint-də paylaşırıq. Buna baxmayaraq FreeSWITCH core onları təsadüfi kanal dəyişənləri kimi görür və onlara məhəl qoyulmaya bilər. Hətta bir neçə rezerv edilmiş spesifik dəyişənlərdə mövcuddur ki, FreeSWITCH kanalının özünü aparmasına bir neçə maraqlı istiqamətdə təsir edə bilir. Əgər siz nə zamansa, dəyişən üçün hansısa script dili ya da quraşdırma motoru istifadə etmisinizsə(Bəzi hallarda Attribute Value Pairs AVP adlanır), xeyrinizədir ona görə ki, **channel**(kanal) dəyişənləri eyni konsepsiyyaya sahibdirlər. Sadəcə dəyişən adı və mənası kanala təyin edilir və data təyin olunur.

Bunun hətta program interfeysi(programlar yığımı) mövcuddur hansı ki, sizə Dialplan-dan öz dəyişənlərinizi təyin etməyə şərait yaradır:

```
<extension name="example 3">
    <condition field="destination_number" expression="^2000$">
        <action application="set" data="foo=bar"/>
        <action application="bridge" data="user/2000"/>
    </condition>
</extension>
```

Bu misal demək olar ki, öncəki ilə eynidir ancaq sadəcə zəngin yerləşdirilməsi əvəzinə biz sadəcə **foo** dəyişənin mənasını **bar** edirik. Bu dəyişən tam zəng müddətində olacaq və hətta zəngin sonunda təyin edilə bilər(jurnalarda tam göstərilir).

Qurmaq istədiyimiz daha kiçik şeylər daha da baza resurslarının təkrar istifadəsinə gətirib çıxarırlar və bu da sistemin istifadəsini asanlaşdırır.

Misal üçün codec interfeysi core-dan, audio paketlərin izolyasiya edilmiş coding və decoding-dən başqa heçnə bilmir. Artıq bu modul yazılıqlardan sonra, onun axınında olan audio kodek üçün istənilən Endpoint interfeysə yararlı olur. Bu o deməkdir ki, əgər biz işləyən Text-To-Speech modulu götürsək, sintez olunmuş danışışı FreeSWITCH dəstəkləyən istənilən bütün Endpointslərdə generasiya edə bilərik. Hansının birinci gəlməsinin önəmi yoxdur ona görə ki, onların bir-birləri ilə heç bir əlaqələri yoxdur. Buna baxmayaraq, birinin əlavə edilməsi digəri üçün yeni funksional imkanlar yaradır. TTS modulu daha istifadə ediləndir ona görə ki, çoxlu codec-lər istifadə edə bilir. Codec-lər daha funksionaldır ona görə ki, onlardan istifadə etmək üçün yeni funksional yaradılmışdır. Eyni fikir programlara da aiddir. Əgər biz yeni program modulu yazsaq, mövcud olan endpoint-lər dərhal həmin programı işə salıb istifadə edə biləcək.

Çətin programlar daha da asanlaşdırı

FreeSWITCH əksər programlarda olan böyük çətinliyi yığışdırır. Gəlin iki ədəd daha böyük olan programın misalına baxaq.

Voicemail

Müzakirə edəcəyimiz ilk program voicemail-dir. Bu programın əsas üstünlüyü ondan ibarətdir ki, çox asan tətbiq eləmək olur. O səs xidmətini təmin edir. Bu programı zəng bitməyən hallar olduqda ikinci seçim olaraq, bridge tətbiqindən həmən sonra əlavə etmək düzgündür. Biz öncə müzakirə elədiyimiz xüsusi dəyişənlər vasitəsilə bunu tətbiq edə bilərik. Gəlin son genişlənməmizin yeni versiyasına baxaq hansı ki, bizə səsli mesaj qoymağə şərait yaradır.

```
<extension name="example 4">
    <condition field="destination_number" expression="^2000$">
        <action application="set" data="hangup_after_bridge=true"/>
        <action application="bridge" data="user/2000"/>
        <action application="voicemail" data="default ${domain} 2000"/>
    </condition>
</extension>
```

Biz burda kanal dəyişənlərindən ikisinin istifadə edildiğini görürük. İlk olaraq biz **hangup_after_bridge=true** yazımaqla sistemə deyirik ki, əgər heç olmazsa digər telefona gedən bir ədəd uğurlu bridge zəngi alırsa təyin et və digər instruksiyalara məhəl qoyma. Biz həmçinin dollar işarəsi ilə **{}\$** fiqurlu mötərizə daxilində olan domain dəyişən istifadə edirik. Bu domain adının avtomatik quraşdırılması üçün spesifik dəyişəndir hansı ki, bütün telefonlar üçün quraşdırımdan istifadə edilir.

Bu misalda biz 2000-ə istənilən şəxsdən gedən zəngi yoxlayırıq. Sonra gələn zəngi 2000 genişlənməsi ilə qeydə alınan telefonla bridge etməyə çalışırıq.

Əgər zəngdə problem olarsa, ya da cavab olmazsa növbəti instruksiyaya keçid edirik hansı ki, voicemail programını işə salacaq. Biz programın bilməli olduğu məlumatları ona ötürürük və voicemail üçün hansı genişlənmənin olduğunu da bildiririk ki, bu situasiyada emal etmək üçün bütün məlumatlara sahib olsun. Ardınca voicemail programı öncədən qeydə alınmış səs yazısını işə salır ya da önce müzakirə elədiyimiz **Say** modul interfeys sayesində sizin üçün yenisiyi generasiya edir. Bu sətirlər birlikdə səs fayllarıdır hansı ki, "The person at extension 2000 is not available, please leave a message" sözləri ilə olan səsi işə salır. Sonra **mod_voicemail** sizə təklif edəcək ki, mesajı yarasınız və artıq siz zəng etdiyiniz şəxsin səs daxil olan qutusunda öz mesajınızı səslə deyə bilərsiniz. Əlavə funksiya olaraq əgər, siz yerləşdiriyiniz mesajla razi olmasanız bunu istədiyiniz sayda edə bilərsiniz. Artıq sonda, işinizi bitirdikdə FreeSWITCH-in **MESSAGE_WAITING** xəbərdarlığı mərkəzi xəbərdarlıq sisteminə ötürülür hansı ki, **mod_sofia** tərəfindən istifadəçi xəbərdalığı olaraq götürülür harda ki, xəbərdarlıq informasiyası SIP-ə tərcümə edilir və bu halda **SIP_NOTIFY** mesajı SIP telefonə mesajın gözlənilməsi barədə məlumat verir. Əgər bütün işlər planladığımız kimi gedərsə, 2000 nömrəsi ilə qeydiyyatdan keçən SIP telefonun indikatoru ona mesajın gəlməsi barədə yanib sönərək xəbərdarlıq edəcək.

Bu misalda biz ancaq salamlayıcının necə işləməsini, mesajın yazılması və istifadəçiyə yollanılmasını deyil həmçinin FreeSWITCH core event sisteminin işləməsini aşkarla çıxardırıq. FreeSWITCH event sistemi digər misallarda göstərdiyimiz kimi, modul interfeys deyil bu mərkəzi motordur hansı ki, siz xəbərdarlıq adına birləşdirə bilərsiniz ki, hadisə baş verən kimi dərhal reaksiya verməsin. Digər sözlə desək, FreeSWITCH core-dan gedən və qayıdan xəbərdarlıqların axını keçir. Modullar fərqli hadisələrə birləşdirilə(yəni qulaq asa) bilər. Onlar həmçinin hadisələri birbaşa event motorunun içine generasiya edə bilərlər və digər modullar bu hadisələr üçün qulaq asa bilərlər. Öncə müzakirə elədiyimiz kimi, **Sofia** SIP modulu **MESSAGE_WAITING** məlumatı üçün bu eventlə əlaqələnir. Bu bizim **mod_voicemail** moduluna şərait yaradır ki, sistemdə olan digər istənilən bilgi olmadan **mod_sofia** ilə əlaqəyə girə bilsin. **mod_sofia**-dan gələn xəbərdarlıq **mod_voicemail** tərəfindən generasiya edilir və SIP mesajın içine ötürülür.

Bu tip çətin interaktiv sistemin qurulmasında bütün mümkün ola bilən dillərdə bir necə problem ola bilər ki, dəstəyə ehtiyac olsun çünki, fayllar avtomatik mesajlarda bir-biri ilə əlaqələnib işə salınır. **Say** modulu sətir fayllarının birlikdə işlədilməsi üçün yaxşı xidmət təqdim edir ancaq, o konkret spesifik məhdudiyyətlərə sahibdir(sözün hərflərlə deyilməsi, nəyinsə saygacı ya da hansısa tarixin deyilməsi). **Say** modulunun ən üst seviyyədə daha çətin təyin edilən modulu **Phrase Macros**-dur. Phrase Macros-ları XML ifadələr yığımıdır hansı ki, müntəzəm ifadəyə(**RegEX**) uyğun olaraq və komanda sətirini yerinə yetirərək parametrlərin siyahısını çıxardırlar. Bu XML Dialanın işləməsinə çox oxşayır ancaq, individual sifarişlərdə interaktiv IVR ssenarilərinin səsli cavablandırılması üçündür. Misal üçün, sətir fayllarında onun deyilməsini məcbur etmək üçün kodlaşdırmaq əvəzinə, **mod_voicemail** sizdən mesajınızın yazılmasını soruşur və code yalnız **voicemail_record_message** adlı **Phrase Macro** çağırır. Bu sərbəst sətir quraşdırımda **mod_voicemail** və **Phrase Macro** bölməsiylə birgə istifadə olunmaqla bizə, istifadəçilərə icazə verir ki, qeyri-adi programlaşdırmanı etmədən faylı redaktə edək.

```
<macro name="voicemail_record_message">
<input pattern="^(.*)$">
```

```

<match>
    <action function= "play-file" data="voicemail/vm-record_message.wav"/>
</match>
</input>
</macro>

```

mod_voicemail öz növbəsində **voicemail_record_message** macro-nu işə saldıqda o, önce nümunəni uyğunlaşdırır hansı ki, bu təyin edilmiş makrosda heç bir daxiletmə olmadığına görə hər şeyi uyğunlaşdırır. Əgər makrosda həqiqətən daxil etmə olsaydı, müxtəlif daxil etməyə əsaslanan müxtəlif hərəkətləri yerinə yetirmək üçün, nümunəylə tutuşdurma istifadə oluna bilərdi. Uyğunluq olan kimi, uyğunluğun tag-ı bizim Dialplan-in nümunəsində olduğu qaydada təsirin tag-ları üçün XML-ə təhlil edilir. Bu macro yalnız **vm-record_message.wav** faylini oxudur ancaq, daha çətin macro-lar vardır hansı ki, biri sizin yazdığınız səsləri yoxlayır ya da sizə deyir ki, öz inbox-nuzda nə qədər mesaj var. **Say** modulu və audio faylların işə salınması üçün fərqli kombinasiyalardan istifadə edə bilərsiniz. Phrase Macros-ları detallı şəkildə 6-ci başlıqda "XML IVR-ların və Phrase Macrosların" və ekstensiv olaraq 7-ci başlıqda "LUA ilə Dialplan script yazma" bölmələrində istifadə edilir.

Bir daha, bizim core tərəfindən yüklənilmiş phrase system, audio fayl və Say modulu ilə əlaqəmiz var hansı ki, bize böyük funksionallıq verir. Say modulu təyin edilmiş dil və ya həmin dilin məhdudiyyətləri çərçivəsi üçün xüsusi yazılmışdır. Biz daxil edilən dəyişənlərə əsaslanaraq, təyin elədiyimiz vaxtda say moduluna müraciət edərək deyə bilərik ki, uyğun olan **Say** moduluna tərcümə edilsin. Phrase Macro sistemi sizin code-nuzda işləməsi üçün böyük dəyişənlər konsepsiaya sahibdir hansı ki, sonra gündəlik istifadəçilərinizdə asanlıqla quraşdırılacaq. Misal üçün əgər kiçik bir **IVR** qurmaq istəsək hansı ki, bizdən 4 rəqəmli bir yiğim etməmizi, onun geriye oxunulmasını və dayanmamızı istəsə, biz **myapp_ask_for_digits** və **myapp_read_digits** adlı bir macro qura bilərik. Öz code-umuzda macros-un adla yerinə yetirilməsini istərdik hansı ki, daxil etdiyimiz mənaları ötürərək önce rəqəmləri soruşacaq sonra oxuyacaq. Artıq bu yerə çatan kimi, daha az təcrübəyə sahib olan inzibatçı XML fayllar yarada bilər ki, uyğun səsləri işə salsın. İstənilən şəxs Say modulunu müxtəlif dillərdə geriye oxumaq üçün heç bir code-lama etmədən istifadə edə bilər. Voicemail programı FreeSWITCH-də istifadə edilə biləcək Server programlarından biridir. FreeSWITCH üzərində saysız imkanlar mövcuddur ki, telefon zənglərini kompyüterlərlə birləşdirə biləsiniz.

Multi-party konfrans

FreeSWITCH-in konfrans üçün **mod_conference** adlı digər məhşur utiliti vardır. **mod_conference** modulu dinamik konfrans zallar təmin edir hansı ki, bir neçə səs kanalını birləşdə doldurma imkanına sahibdir. Bu görüş keçirilməsi üçün istifadə edilə bilər harda ki, bir neçə zəng edən tərəf var və onlar həmin zəngdə iştirak etmək istəyirlər. Konfransa qoşulan hər bir şəxs digərlərinə qoşulur və həmin anda da digər iştirakçılarla danışmaq imkanına sahib olur. Digər telefona bridge-lə qoşulmaq üçün önce yazdığımız Dialplan misalını istifadə edərək genişlənmə yarada bilərik ki, konfrans zalına qoşulaq:

```

<extension name="example 4">
    <condition field="destination_number" expression="^3000$">

```

```

<action application="conference" data="3000@default"/>
</condition>
</extension>
```

Bu elə körpü(bridge) yaradılması kimi çox sadədir ancaq bu genişlənmənin üstün cəhəti odur ki, çoxlu zəng edən **3000** nömrəsinə yığıb eyni konfrans otağına daxil ola bilərlər. Əgər bu konfransa 3 şəxs qoşuludursa və 1-ci çıxarsa, onda digər iki şəxs heç bir problem hiss etmədən danışığına davam edəcək.

Konfrans modulunun həmçinin digər üstünlükləridə var hansı ki, bütün konfrans iştirakçılarına və ya konkret iştirakçıya səs faylini oxuda ya da Text-To-Speech işə sala bilər. Artıq bildiyiniz kimi, biz bu işi TTS və səs faylı interfeyslərinin modullarını istifadə edərək edə bilərik. Gördüyünü kimi funksionallığı sistemdə olan digər komponentlərin haqqında dərindən araşdırmaqla yox, çox az hissələrin birgə istifadə edilməsi ilə artırmaq olur. Konfrans modulu həmçinin event sistemdən xüsusi üsulla istifadə edir ki, istifadəçi hadisələrini təyin edə bilsin. O ilk dəfə yüklenəndə **mod_conference** spesifik ad hadisəsini subclass adı ilə bron edə bilər. Hansısa maraqlı bir hadisə misal olaraq istifadəçinin konfransa qoşulması tərk etməsi, danışığa başlaması və ya dayanması baş verən kimi o, bu hadisələri core-da olan **CUSTOM** event kanalına ötürür. Əgər biz bu hadisələrin alınmasında maraqlı olduğumuz halda, CUSTOM eventinə üzv olaraq marağında olduğumuz eventi təyin edən əlavə subclass sətiri ötürürük. Bu halda o, **conference::maintenance** olacaq. Konfrans imkanları data ətraflı 14-cü başlıqdə, **Genişlənmiş funksiyalar və əlavə ədəbiyyat** başlığında müzakirə ediləcək.

FreeSWITCH API (FSAPI)

FreeSWITCH-də digər çox güclü olan **FSAPI** adlı bir modul mövcuddur. Bu interfeysin iş prinsipi çox sadədir. O spesifik işin yerinə yetirilməsi üçün adı tekst sözü öz girişində alır hansı ki, parçalaya və ya parçalamaya bilər. Qayıdan cavab həmçinin sətir olacaq və çağırılan funksiyadan asılı olaraq, bir simvoldan tutmuş, bir neçə səhifə tekstdək tərkibə malik olan həcmə sahib ola bilər. FSAPI funksiyasının əsas üstün cəhəti ondan ibarətdir ki, bu modul digər modullarda olan zəngin gündəlik işlərini kodla birbaşa əlaqələnmədən çağrıra bilər. FreeSWITCH-in command line interfeysi FSAPI istifadə edir ki, FreeSWITCH API əmrlərinə əməliyyat sistemi səviyyəsindən keçid edə bilsin.

Aşağıdakı status adlı FSAPI əmrini FreeSWITCH CLI-dan işə salaraq kiçik bir misal göstəririk:

```
freeswitch@internal> status
UP 0 years, 7 days, 0 hours, 17 minutes, 14 seconds, 478 milliseconds, 390 microseconds
FreeSWITCH (Version 1.5.final git 6a2fc5e 2015-05-28 17:35:17Z 64bit) is ready
206 session(s) since startup
0 session(s) - peak 1, last 5min 0
0 session(s) per Sec out of max 30, peak 1, last 5min 0
1000 session(s) max
min idle cpu 0.00/100.00
Current Stack Size/Max 524288K/524288K
```

Biz burda **status** əmrini daxil edib **ENTER** sıxan kimi, "**status**" sözü FSAPI **status** funksiyasını tətbiq edilən modulda axtarış edir. Sonra, baza funksiya çağırılır və özəyə statusla bağlı müraciət gedir. Status verilənləri alınan kimi, çıxarış qayıdan axına yazılır və əmrin nəticəsi kimi çap edilir.

Biz artıq öyrəndik ki, bu modulla FSAPI funksiyalarını istənilən yerdən yerinə yetirərək yarada və çıxara bilərik(Misal olaraq CLI). Ancaq daha da geniş imkanlar var. Modullar həmçinin yazılıa bilərlər ki, əmrləri FSAPI interfeysə yollasın və nəticəsini spesifik protokolla ötürsün. FreeSWITCH-də bu işi görən iki modul mövcuddur - **mod_xml_rpc** və **mod_event_socket**(9-cu başlıqda açıqlanır, Quraşdırma Static XML-dən kənara yerləşdirilir və 10-cu başlıqda FreeSWITCH-in kənardan idarə edilməsi). Bu modul FreeSWITCH-in modulu olaraq standartlaşmış **XML-RPC** protocol tətbiq edir. Müştərilər XMLRPC interfeys istifadə edərək, FreeSWITCH-ə qoşula və FSAPI-da olan istənilən secdikləri əmri yerinə yetirə bilərlər. Beləliklə uzaq müştəri biraz önceki misalda göstərdiyimiz nəticəni **RPC** müraciətlə **status** əmrini yerinə yetirsə əldə edəcək. Bu modul həmçinin FreeSWITCH-i ümumi web server ilə təmin edir hansı ki, FSAPI əmrlərini birbaşa URL-dən əlcətan edir. Misal üçün WWW üzərindən <http://meselen.freeswitch.lan:8080/api/status> linkinə daxil olmaqla siz **status** əmrinin nəticəsini əldə edə bilərsiniz. Bu üsulu istifadə edərək, FSAPI-da əmrlərini yarada bilərik ki, CGI kimi işləsin hansı ki, dinamik web imkanları ilə birbaşa FreeSWITCH daxilinə yetkini təmin edir.

Gördüyüümüz kimi FSAPI çox elastikdir. Artıq biz bilərik ki, bu bize modulların hər birinin funksiyalarını çağırmaq üçün, *WWW* ya da *XML-RPC* funksiyalarına export etmək üçün, modullara çıxan yolu CLI interfeyslə təmin edə bilər. FSAPI üçün açıqlamadığımız başqa bir funksiya var. Biz kanalda olan dəyişən konsepsiyasına qısa da olsa öncə toxunduq və qeyd edək ki, təyin edilən dəyişən mənasını almaq üçün biz **\${myvariable}** formatında regex istifadə edə bilərik. FSAPI funksiyalarına həmçinin **\${myfunction()}** formatında çatmaq olar. Bu o deməkdir ki, FSAPI əmri **myfunction** çağırılmalıdır və **regex** bu funksiyanın çağırışının çıxarışı ilə əvəz edilməlidir. Ona görə də dəyişən genişlənərsə, biz statusu almaq üçün istənilən yerde **\${status()}** istifadə edə bilərik. Misal üçün:

```
<action application="set" data="my_status=${status()}" />
```

my_status dəyişəninə mənimsədilən məna **status** əmrinin çıxışı olacaq.

Tək modul interfesin yeganə çatışmayan cəhəti ondan ibarətdir ki, biz bütün hədəflərə çatmaq üçün onun bütün bacarıqlarını dərinən öyrənməliyik. Bu o deməkdir ki, müzakirə elədiyimiz kimi bütün üsullarla bir FSAPI əmrinə çatmaq mümkündür. Bundan başqa elə spesifik funksiyalar mövcuddur ki, onlar məhz giriş hüquqları metodu üçün nəzərdə tutulmuşdur. Məsələn, əgər biz WEB browserlə giriş hüququn alınması üçün, FSAPI komandasını işə salaraq HTML yaratsaq, ona dəyişən kimi yönəlmək və CLI-dan yetkili olmamızı istəməsək!!! Həmçinin əgər biz Dialplan istifadəsi üçün hansısa zəng detallarına əsaslanan hesablama işini FSAPI funksiyası ilə görsək bu, WEB ya da CLI-dan yararlı olmayıacaq. Böyük imkanlar böyük məsuliyyət deməkdir və buna görə də biz tam düzgün qərar verməliyik ki, harda və necə FSAPI funksiyalarından istifadə etməliyik ki, onlarda maksimal yarar əldə edə bilək.

XML reyestri

Biz artıq FreeSWITCH-in əksər fundamental komponentləri və onların necə bir-birləri ilə əlaqəyə girmələri haqqında müzakirə apardıq. Biz event sisteminin core üzərindən informasiyanın necə aparmasını və verilənlər üçün XML Dialplan-in XML reyestrə necə müraciət etməsini gördük. XML reyestrinin daha ətraflı araşdırılmasının əsl vaxtıdır. XML reyestri mərkəzi idarəedilən XML sənəddir hansı ki, FreeSWITCH-in düzgün şəkildə idarə edəcəyi bütün kritik məlumatları özündə saxlayır. Başlanğıc sənəd sizin sərt diskdən yüklənir və spesifik preprosesora ötürülür. Bu preprocessor digər XML sənədləri və spesifik əməliyyatları əlavə edə bilər. Misal üçün qlobal dəyişənlərin elan edilməsi hansı ki, preprocessorla sənədin sonuna gedərək həll edilə bilər.

Mövcud sənəd və ona əlavə edilən bütün fayllar parçalandıqdan, dəyişdirildikdən və statik XML sənədə generasiya edildikdən sonra bu sənəd RAM-a yüklənir. XML reyestri bir neçə quraşdırma seksiyalarına bölünür - *odbc*, *dialplan*, *directory*, *location*, *chatplan*, *languages* və *phrases*. Core və modulları öz quraşdırırmalarını quraşdırma seksiyasından alır. XML Dialplan modulu öz Dialplan datasını dialplan seksiyasından alaraq çəkir. SIP autehtifikasiyası, istifadəçi axtarışı və voicemail modulu öz hesab informasiyalarını directory seksiyasından alır. Phrase Macros isə öz quraşdırırmalarını phrases seksiyasından alır. Əgər biz diskdə olan sənədlərin istənilən birində dəyişiklik eləsək, RAM-da olan quraşdırmanın dəyişikliyi özünə götürməsi üçün, CLI-dan **reloadxml** əmrini daxil etməliyik(Bu FSAPI interfeysin FreeSWITCH core-a qoşulmasının misali idi).

Dil modulları

Fayllar və son nöqtələr kimi, FreeSWITCH-lə birbaşa interfeysə sahib olmayan ancaq yenə də mövcud texnologiyalarla sıx əlaqə imkanları təqdim edən bir xüsusi dil modul tipi var. Dil modulları *LUA*, *JavaScript*, *Perl* və hətta *C#* kimi dilləri(**mod_managed** istifadə edərək) FreeSWITCH-ə uyğunlaşdırır və funksionallığı işləmə müddətində core ilə proqramlaşdırma dilləri arasında yayımlayır. Bu IVR kimi proqramların FreeSWITCH-ə geriye çətin qayıdış üçün adı interfeyslə daxili dildə yazılımasına imkan yaradır. Dil modulları adətən application interfeys, FSAPI interfeys vasitəsilə core-da qeydiyyatdan keçir və Dialplan-dan yerinə yetirilir. Dil modulları çox böyük imkanlar yaradır və çox böyük gücə sahibdir. Dil modullarını istifadə edərək, standart proqramlaşdırma dil ilə güclü səs proqramları yazmaq olar. Başqa dillə desək, proqramlaşdırma dili vasitəsilə telefonu idarə etmək olar.

Göstərmək üçün quraşdırma

Bütün bu konsepsiyanın başa düşülməsi həqiqətdən də başlanğıcda çətin gəlir. Xüsusəndə düşünsək ki, bu imkanların hamisini click-lə əldə etmək olar. Hər bir yeni səviyyənin core-un əvvəline qoyulmasına görə bütün işlər və oxumaq asanlaşır. FreeSWITCH quraşdırmasının demonstrasiyası - program təminatı ilə istifadəçi arasında olan son sətir müdafiədir çünki, telefon mühitində həm yaxşı və həmdə pis isifadəçilər ola bilər. Biz normal istifadəçiləri belə şəylərdən çox çətinliklə xilas edə bilərik.

Quraşdırmanın əsas məqsədi ondan ibarətdir ki, FreeSWITCH üzərində olan yüzlərlə parametrin işləməsini göstərməkdir. Biz sizə işlək biq quraşdırma veririk hansı ki, toxunmadan istifadə etsəniz hər şey işləyəcək amma dəyişiklik üçün əlinizdə hazır nüsxələriniz olacaq. FreeSWITCH haqqında lego kimi düşünün. FreeSWITCH və onun böyük olmayan hissələri lego-nu xatırladır hansı ki, təsəvvürümüzə gələn hər şeyi üzərində edə bilərik. Demonstrasiya tərkibində addım-adım olan işlək bir nüsxənin qurulmasını təşkil edir. Artıq müəyyən bir təcrübəyə sahib olduqdan sonra siz özünüz bu quraşdırmaları dəyişə və öz fantaziyanıza uyğun olan dəyişiklikləri edə bilərsiniz. Əgər quraşdırma hər hansıa bir dəyişikliklə səhv etsəniz sadəcə susmaya görə olan vəziyyətə yenidən yüksəsəniz yetər və yenidən kompilyasiyaya ehtiyac qalmır. Quraşdırılmanın göstərilməsi 3-cü başlıqda "Misal nüsxəsinin sınaqdan keçirilməsi"-ində olacaq.

Artıq FreeSWITCH sizin sistemə yüklenikdən sonra, heç bir quraşdırma sətirində dəyişiklik etmədən onu işə sala bilərsiniz. Artıq siz öz SIP telefonunuz yada SIP soft telefonunuzda qurdugunuz serverin ünvanını yazıb qoşulmaqla test zəng edə bilərsiniz. Əgər analoq telefonun SIP-ə çevrilməsini istəyirsinizsə "**ATA-analog telephone adapter**" adlı bir avadanlıq kartı alıb serverinizə qoşmalısınız. Əgər sizin telefonun sayı 1-dən çoxdurسا onda, **1000-1019**(bu aralıq susmaya görə sınaq məqsədləri üçün öncədən quraşdırılmışdır və bu aralıqda olan hər bir genişlənmə üçün şifrə **1234**-dur) aralığında olan nömrələrdən istifadə edərək onların hər birini quraşdırı bilərsiniz. Bu aralıqda olan nömrələrdə bir neçə qeydiyyatdan keçən telefon qoşulan kimi, siz onlar arasında zəng edə və ya **3000-3399**(Susmaya görə quraşdırılmış konfrans zalları) arasında olan nömrələrə zəng edərək konfrans zallarında görüşə bilərsiniz. Əgər siz qeydiyyatdan keçməyən digər bir nömrəyə zəng etsəniz ya da digər nömrənin çağırışını həddən artıq gözləsəniz o halda **phrase** sistemi sizə cavab verəcək ki, qarşı tərəfə çatmaq mümkün olmadı və sizə voicemail-in saxlanılmasını soruşacaq. Əgər siz **5000** rəqəminə yüksənəz görəcəksiniz ki, səliqə ilə yüksəlmış bir neçə menyuya sahib olan IVR sisteminin nüsxəsi işləyir. Təkmilləşdirilməsi üçün çox kiçik əlavə və dəyişikliklər var ki, demonstrasiya quraşdırmasında edilsin. Misal üçün öncə danışdığımız preprocessor direktivlərinin istifadə edilməsində demonstrasiya quraşdırmları faylların siyahısını XML reyestrinə müxtəlif ünvanlardan yükleyir(Bu o deməkdir ki, fərqli qovluqlarda olan hər bir faylı son XML quraşdırılma sənədində birləşdiriləcək). Özünə yer alan iki vacib ünvan hansı ki, istifadəçi hesablari və genişlənmələr Dialplan-da saxlanılır. Susmaya görə öncədən hazır olan 20 genişlənmənin hər biri üçün quraşdırma, özünə aid olan faylda saxlanılır. Biz asanlıqla yeni bir istifadəçi üçün bu qovluğun içində yeni faylı yaradıb və hazır nüsxələrdə olan sintaksisi fayla nüsxələyib lazımı genişlənmə dəyişikliyi edərək yeni nömrə əlavə edə bilərik. Sadəcə sonda FreeSWITCH CLI-da **reloadxml** əmrini daxil etmək lazımdır. Tamamilə eyni fikir Dialplan nüsxəsinə də aid edilir. Biz adı genişlənməni onun faylına əlavə edə və istədiyimiz ünvandan işə sala bilərik.

Notice

FreeSWITCH hərəkətli hissələrdən ibarət olan çətin bir sistemdir hansı ki, elastikliyə və genişlənmə imkanına sahib olaraq, möhkəm və dayanıqlıdır. Core modullar üçün öz interfeysi genişləndirir. Bu modullar funksionallığı daha da asanlaşdıracaq və istifadəçi səviyyəsinədək sadələşdirəcək. Bu modullar sayəsinə FreeSWITCH-də olan funksionallığı fərqli əlaqə protokolları ilə tanınmış formatda kənara ötürə bilərik. Biz fərqli tip modullara, onların nəcə core ətrafında birləşməsinə və daha böyük funksionallığın əldə edilməsi üçün bir-birlərile necə əlaqəyə girmələrinə baxdıq. Biz FreeSWITCH-in bəzi məhşur programları olan konfrans və voicemail programlarına baxdıq hansı ki, işləmələri üçün digər sistem daxilində olan modullarala əlaqələnir və onların mövcudluğundan heç xəbərləri belə olmur. Bu event sistem vasitəsilə həyata keçirilir. Biz həmçinin baxış üçün bir neçə nüsxəni analiz elədik. Ancaq oxuduqlarınızı mütləq sinaqdan keçirin. Mənbə kodlarını internetdən çəkin və hansısa bir maşına quraşdırıb işə salın. Test üçün bir neçə SIP fiziki telefon və ya program SIP telefonu əldə etməlisiniz. Başlanğıc yoxlanışları etdikdən sonra biz daha dərinliklərə gedəcəyik. Misal üçün IVR-da IVR menyu və ya bir neçə yeni genişlənmə yaradılması.

Növbəti başlıqda biz FreeSWITCH sistemin qaldırılması və işə salınmasının ilk addımını edəcəyik.

2-ci başlıq

Kompilyasiyası və yüklənməsi

FreeSWITCH açıq kodlu program təminatıdır. Bu o deməkdir ki, istənilən şəxs açıq kodu oxuya, üzərində istənilən dəyişikliyi edə, düzəldə, ya da sıradan çıxara bilər. İlk başlayanlar üçün mənbə kodlarına baxış həddən artıq çətin görünə bilər ancaq, FreeSWITCH developerlər kodların oxuyucu üçün həddən ziyadə asan olması üçün hər şey edirlər. Hal-hazırda bir neçə Linux distributiv üçün hazır binar fayllar olsa da, hamısı üçün mövcud deyil. Ancaq FreeSWITCH komandası gələcək üçün bütün UNIX/Linux üçün kompilyasiya edilmiş hazır paketlərin öz anbarlarında saxlanılmalarına söz verirlər. Biz yüklənməni əlimizlə mənbə kodlardan edəcəyik. Başlığımızda bütün UNIX və Linux dedikdə həmçinin, FreeBSD və MacOS-da nəzərdə tutulur.

Bu başlıqda biz FreeSWITCH-in UNIX tipli sistemlərə mənbə kodlarının endirilməsi və yüklənməsini və həmçinin Windows maşında yüklənməsini açıqlayacaqıq. Hər bir sistemin özünəməxsus problemlərini açıqlayacaqıq və sonda FreeSWITCH-i arxa fonda işə salacaqıq.

Bu başlıqda biz aşağıdakı başlıqları açıqlayacaqıq:

- FreeSWITCH mühitinin qurulması
- FreeSWITCH-in UNIX/Linux maşınlarda yüklənməsi və qurulması
- FreeSWITCH-in Windows 2012 serverdə kompilyasiyası və yüklənməsi

FreeSWITCH mühitinin qurulması

FreeSWITCH-də digər program təminatları kimi, qurulması üçün müəyyən bir mühit tələb edir. Əsasən nəzərdə tutulur ki, öz avadanlığınız üçün əməliyyat sistemini seçə və sonra onu LAN/WAN-la fiziki mühitə qoşasınız.

Əməliyyat sistemi

İlk sual yaranır ki, hansı əməliyyat sistemi seçilməlidir? Ümumiyyətlə əməliyyat sistemini sizin həmin sistem üzərində olan biliyinizi görə seçmək lazımdır çünki, problem çıxdığı halda siz o sistem üzərində kifayət qədər biliyə sahib olmalısınız ki, üst səviyyə problemləri tezliklə həll edə biləsiniz. Bəzi istifadəçilər gileylənlərlə rastlaşıq və buna görə də FreeSWITCH qəti şəkildə məsləhət görür ki, 64 bitlik avadanlığa 32 bitlik platformanı yazdıqda problemlərlə rastlaşıq və buna görə də FreeSWITCH qəti şəkildə məsləhət görür ki, 64 bitlik avadanlığa 64 bitlik OS yazılısın və sonra FreeSWITCH yüklənsin.

Kim Windows maşına üstünlük verir o XP, Vista, Windows 7, Server 2003, Server 2008 R2 ya da Server 2012 yükleyə biler. Xeyli istifadəçi istismarda 2008 server üzərində problem yaşamadan istifadə etdiklərini bildiriblər.

Digər tərəfdən UNIX tipli sistemlərdə müxtəlif çeşidlilik var və onlardan birini seçmək daha rahat olar. Təklif edilən Linux/UNIX əməliyyat sistemlərinən CentOS, Debian, Ubuntu, BSD, MacOS və Solaris seçə bilərsiniz. FreeSWITCH developerləri hansısa bir spesifik OS-a fərq qoymurlar.

Hər kəs FreeSWITCH-in hansı distrubutiv üzərində yaxşı işləməsini soruşur. Hansısa əməliyyat sisteminin seçilməsi çoxlu faktorlara əsaslanır və məhz bu səbəbdən hansının dəqiq seçilməsi düzgün cavab deyil. Ancaq sisteminizin etibarlı və dayanıqlı olmasını istəyirsinizsə, dünya praktikası CentOS5 və Debian6-nı göstərir. Yalnız FreeSWITCH debian-a həddən artıq dəstək verir və şəxsi təcrubəmdə Debian üzərində işlərin daha da stabil olmasını gördüm.

Qeyd: Hal-hazırda stabil 1.4 və 1.6 versiyaları var və 1.7 üzərində işlər gedir. Amma gələcəkdə 1.4 tamam durdurulacaq.

Nəzərinizdə saxlayın ki, telefon sistemləri üçün ən son versiya heç də ən yaxşısı demək deyil.

Əməliyyat sistemləri tələbləri

Hər bir əməliyyat sisteminin özünə məxsus tələbləri mövcuddur. Bu başlıqdə biz onların hər birinə uyğun olan yüklənməni edəcəyik.

Linux/UNIX

Aşağıdakı paketlər sizin sistemdə artıq yüklənmiş olmalıdır. Qeyd edin ki, git client tələb edilmir:

- **Git:** Git client həmçinin size kodlar yerləşən anbara yetki verir (Əsasən programçılar üçün yararlıdır ki, son kodu görsünlər)
- **GNUMAKE:** Make-in GNU versiyası
- **AUTOCONF:** 2.60 ya da daha yuxarı versiya

- **AUTOMAKE:** 1.9 versiyası və ya daha yüksək
- **LIBTOOL:** 1.5.14 versiyası və ya daha yüksək
- **GCC:** 3.3 və ya daha yüksək
- **WGET:** İstənilən versiya
- **LIBNCURSES:** İstənilən versiya
- **BZIP2:** İstənilən versiya

MacOS X

Təkidlə məsləhət görülür ki, MacOS X istifadəçilərinə ən azı 10.4 versiyası olsun. FreeSWITCH-in OS X-da kompilyasiya edilməsi üçün Apple XCode Developer Tools-u tələb edilir. Siz bunu apple-in rəsmi saytından <https://developer.apple.com/xcode/> əldə edə bilərsiniz ancaq öncədən qeydiyyatdan keçməlisiniz.

Qeyd: Apple-in OS X alətlərinə bəzi dəyişikliklər edilmişdir. Yükləmə və quraşdırma üçün <https://freeswitch.org/confluence/display/FREESWITCH/Installation+and+Setup+on+OS+X> linkini oxumanız yetər.

Windows

Windows mühitində FreeSWITCH-in iki əsas tələbi mövcuddur. Onlar aşağıdakılardır:

1. Microsoft Visual C++ 2008 ya da 2010(ya da 2008 və ya 2010 Express edition)
2. Sixilmış faylı açmaq üçün istənilən alət.

FreeSWITCH-i Windows-da kompilyasiya etmək və yükləmək üçün Microsoft Visual C++(MSVC) ya da Visual C++ Express Edition(MSVCEE) yazılır. Express versiyasını siz asanlıqla <http://www.microsoft.com/Express/VC> linkindən əldə edə bilərsiniz. Ancaq mən VS2010 versiyasını internetdən çəkdirim. Hər bir halda yükləməyə başlamazdan önce **7zip** <http://www.7-zip.org/> ya **Winrar**-ı <http://www.rarlab.com/> internetdən endirib yükləmək lazımdır. Həmçinin mənbə kodları GIT-dən endirə bilmək üçün <https://github.com/msysgit/msysgit/releases/download/Git-1.9.5-preview20150319/Git-1.9.5-preview20150319.exe> programını endirib serveriniziə yükləmək lazımdır.

Qeyd: Visual C++ üçün 64 bitlik platforma dəstəyi mövcud deyildir və əgər siz öz serveriniziə FreeSWITCH 64 bitlik platformanı kompilyasiya etmək istəyirsizsinizsə, VS2010 Professional Edition yükləmək lazımdır. Mən öz testimdə də 2010 professional yükləmişdim.

Mətn redaktoru və XML

FreeSWITCH-lə işləmək o deməkdir ki, sizin artıq özünüzə rahat olan çox gözəl bir mətn redaktorunuz var. Mətn redaktoru istifadə etdikdə, mütləq

məsləhətlidir ki, o XML-də dəyişiklik etməyi dəstəkləsin çünki, dəyişiklik zamanı sizin üçün görünüş çox asan olacaq və gözünüzə rahat olacaq.

Əgər düzgün seçimiz yoxdursa sizə məsləhət verə bilərik. Linux/UNIX mühiti üçün mütləq Grafik istifadəçi interfeysi olmalıdır. Aşağıdakı editorları sadalayıraq:

- **Emacs:** Yalnız UNIX tipli platformalarda işləyir(MacOS daxil olmaqla). rahatlıqla mənbə kodları XML, HTML-də rahat işləyə bilir. FreeSWITCH-in development komandası elə bu redaktorla ilə işləyirlər(Həmçinin GUI versiyası da mövcuddur).
- **Vi/Vim:** Mətn redaktorudur hansı ki, bütün UNIX/Linux məşinlarda işləyir(VIM üçün həmçinin GUI versiya da mövcuddur) .
- **Notepad++:** Windows mühiti üçün qrafik redaktordur. Əksər program dillərinde işləmək üçün çox rahat şərait yaradır. Havayı mətn redaktoru kimi çox rahat və bol funksionallığı sahibdir.
- **Microsoft Visual Studio/Visual C++ Express:** Bu Integrated Development Environment(IDE)-dir hansı ki, öz mətn redaktoruna sahibdir və XML faylların bu mühitdə açılması sadəcə böyük bir komfortdur. Mühit həmçinin XML tag-larda avtodoldurulmayı dəstəkləyir və həmçinin XML tag-larda olan düzgün olmayan bağlanmalar ya da elementlərdə olan səhvləri altdan qırmızı xətlə göstərəcək.

Mənbə kodlarının endirilməsi

Əksər open source proektləri öz mənbə kodlarını iki kateqoriyaya ayırır: **stable**(stabil) və **latest**(ən yeni və tam stabil olmayan). FreeSWITCH developerləri iki versiya üzərində işləyirlər 1.4.20(Stable), 1.6.9(Stable) və 1.7.0(tam stabil olmayan). Ancaq 1.7.0 üzərində video konfrans üçün çox yeni təkmilləşdirilmələr edilib. Ən yeni mənbə kodlarını birbaşa GIT-lə endirə bilərsiniz(Bu başlıqda ən yeni mənbə kodlarının git vasitəsilə kompilyasiyasına baxacayıq). Bunlardan başqa yaddan çıxarmalı deyilik ki, FreeSWITCH-in hazır binar fayllarda var. Binar faylların yüklənməsi mənbə kodlarının endirilməsi və kompilyasiyadan çox asandır çünki onlar artıq kompilyasiya edilmiş və yüklənməyə hazır vəziyyətdə olurlar.

Əmin olun ki, sizin serverinizin Internetə çıxışı mövcuddur çünki, kompilyasiya müddətində o internetdən əlavə faylları endirəcək.

Bütün mənbə kodlarını aşağıdakı ünvanlardan əldə edə bilərsiniz:

<http://files.freeswitch.org/>

<http://files.freeswitch.org/freeswitch-releases/>

Ən son stabil mənbə kodunu internetdən endirək öz daxili qovluğumuza və açaq. İstənilən Linux-da aşağıdakı göstərilən əmrlər uyğun olaraq işləyəcək:

```
[root@cosfs src]# cd /usr/src
[root@cosfs src]# wget http://files.freeswitch.org/freeswitch-
releases/freeswitch-1.4.20.tar.bz2
[root@cosfs src]# tar jxvf freeswitch-1.4.20.tar.bz2
```

Sonda endirdiyimiz versiyaya uyğun olaraq özündə mənbə kodlarını saxlayan bir qovluq yaranacaq.

Ən yeni mənbə kodlarının kompilyasiyası

Əgər siz FreeSWITCH-in ən son versiyasını daha üstün tutursunuzsa, onda GIT-dən istifadə etməlisiniz. Git-i yükləmək üçün üstün tutduğunuz Linux/UNIX server-də **apt**, **yum** və ya FreeBSD **pkg**(paketlərdən) yükləməlisiniz.

Ubuntu üçün aşağıdakı əmrlə yükleyirik:

```
# apt-get install git
```

CentOS üçün aşağıdakı əmrlə yükleyirik:

```
# yum install git
```

FreeBSD üçün aşağıdakı əmrlə yükleyirik:

```
# pkg install git
```

FreeSWITCH-in UNIX/Linux maşınlarda yüklenməsi və qurulması

Məqsədimiz FreeSWITCH telefon sisteminin hansısa bir UNIX və Linux машında kompilyasiya edilməsi və yüklenməsini etməkdir. Hal-hazırda CentOS 6.7 və FreeBSD10.1-dən istifadə edilir.

FreeSWITCH-in CentOS 6.7 üçün yüklenməsi və quraşdırılması

RPM program anbarını əməliyyat sistemimizə əlavə edək:

```
# rpm -ivh http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el6.rf.x86_64.rpm
```

Həmçinin avropa anbarından da istifadə edə bilərsiniz:

```
# rpm -ivh http://mirror.cedia.org.ec/fedora-epel/6/x86_64/epel-release-6-8.noarch.rpm
```

Mütləq tələb edilən komponentləri yükleyirik:

```
# yum install git gcc-c++ autoconf automake libtool wget python ncurses-devel zlib-devel libjpeg-devel openssl-devel e2fsprogs-devel sqlite-devel libcurl-devel pcre-devel speex-devel ldns-devel libedit-devel lua-devel opus-devel libshout-devel lame-devel mpg123.x86_64 mpg123-devel.x86_64 libsndfile-devel libyuv-devel libvpx-devel unixODBC-devel.x86_64 arping2.x86_64
```

Standart mənbə kodlarının ünvanına daxil oluruq:

```
[root@cosfs src]# cd /usr/src/
```

Əsas mənbə kodlarından yükləmək üçün aşağıdakı əmrən (Bu stabil deyil):

```
[root@cosfs src]# git clone
https://freeswitch.org/stash/scm/fs/freeswitch.git
```

Ya da hal-hazırkı, stabil buraxılışdan götürə bilərsiniz (iş müddətində bu versiyadan istifadə edilmişdir):

```
git clone -b v1.4 https://freeswitch.org/stash/scm/fs/freeswitch.git
```

Klonladığımız mənbə kodlarının qovluğuna daxil oluruz:

```
[root@cosfs freeswitch]# cd /usr/src/freeswitch
```

Yığım prosesinin daha sürətli getməsi üçün, **-j** argumenti yığımı fərqli axınlarda işə salır:

```
[root@cosfs freeswitch]# ./bootstrap.sh -j
```

1-ci addım - modules.conf faylında dəyişiklik

Əgər hansısa modulun aktiv ya da deaktiv olmasını istəyirsinizsə, yerləşdiyiniz qovluğun içində olan **modules.conf** faylında istədiyiniz mətn redaktoru vasitəsilə tələb edilən modulun qarşısından "#" simvolunu silib ya da əlavə edib, faylı yadda saxlayaraq çıxmanız yetər. **mod_flite** modulu FreeSWITCH-ə izin verir ki, **Festival Lite text-to-speech (TTS)** motorunu istifadə edə bilsin (Filter TTS motoru danışığın keyfiyyətli sintezini generasiya eləmir ancaq yenədə TTS-in sınaqdan keçirilməsi üçün bu çox yaxşı alətdir).

Qeyd: Festival Lite haqqında daha ətraflı məlumat əldə etmək istəyirsinizsə, <http://www.speech.cs.cmu.edu/flite/> linkinə müraciət edə bilərsiniz.

Misal üçün bu kompilyasiyada aşağıdakı modulların qarşısından şərh silinmişdi:

```
asr_tts/mod_flite
xml_int/mod_xml_curl
xml_int/mod_xml_ldap
xml_int/mod_xml_radius
```

2-ci addım - configure skriptinin işə salınması

Əksər açıq kodlu proektlərdə olduğu kimi, UNIX tipli mühitlərdə FreeSWITCH öz **configure** skriptindən istifadə edir. Bunun üçün mənbə kodlarının yerləşdiyi qovluğun içindən scripti "**-C**" opsiyası ilə işə salırıq. Configure skripti çoxlu işləri yerinə yetirir və həmçinin öncəki işlərin uğurlu yerinə yetirilməsini də yoxlayır. Əgər öncəki işlər qaydasında yerinə yetirilməyibsə, configure skripti çıxacaq və sizə asılıqlar barəsində məlumatı çap edəcək. Bu baş verdiyi halda siz problemi həll edib yenidən configure skriptini işə salmalısınız. Siz configure skriptini yenidən işə salmazdan önce əmin olmalısınız ki, bütün problemləri artıq həll etmisiniz. **-C** opsiyası quraşdırma moduluna bildirir ki, **config.cache** faylı yaratsın hansı ki, gələcək quraşdırma skriptlərində fərqli kitabxanalar və mənbə kodları strukturunda istifadə ediləcək.

Qeyd: `configure` skripti əksər UNIX/Linux sistemləri üçün açıq kodlu programların yığılması üçün alətdir. `./configure -help` əmrini daxil etməklə bütün köməkçi siyahısını çap edə bilərsiniz.

Qeyd: Əgər FreeSWITCH-in core səviyyəsində **ODBC** (verilənlər bazasına qoşulmaq üçün connector) dəstəklənməsini istəyirsinizsə onda `./configure --enable-core-odbc-support` etməlisiniz.

Aşağıdakı əmrlə quraşdırırıq:
`[root@cosfs freeswitch]# ./configure -C`

Quraşdırılma müddətində siz görəcəksiniz ki, `configure` skripti bir neçə dəfə işə salınır. FreeSWITCH izin verir ki, Apache Portable Runtime(**APR**) və Perl Compatible Regular Expressions(**PCRE**) kimi çoxlu kitabxanalardan istifadə edək. Bu elementlərin hər birinin özüne və tələblərinə aid quraşdırma skripti var. Müəyyən vaxtdan sonra `configure` skripti öz işini bitirəcək və sizə sistem çıxışına məlumat verəcək. Əgər ekrana çıxan mesajların içində heç bir səhv görməsəniz, o halda kompilyasiya prosesinə keçid edə bilərsiniz.

Ekrana çıxan nəticə aşağıdakı kimi olacaq:

```
----- FreeSWITCH configuration -----
Locations:
FHS enabled:      no
prefix:            /usr/local/freeswitch
exec_prefix:       ${prefix}
bindir:            ${exec_prefix}/bin
sysconfdir:        /usr/local/freeswitch/conf
libdir:            ${exec_prefix}/lib

certsdir:          /usr/local/freeswitch/certs
dbdir:             /usr/local/freeswitch/db
grammardir:        /usr/local/freeswitch/grammar
htdocsdir:         /usr/local/freeswitch/htdocs
logfiledir:        /usr/local/freeswitch/log
modulesdir:        /usr/local/freeswitch/mod
pkgconfigdir:      ${exec_prefix}/lib/pkgconfig
recordingsdir:     /usr/local/freeswitch/recordings
runtimedir:        /usr/local/freeswitch/run
scriptdir:         /usr/local/freeswitch/scripts
soundsdir:         /usr/local/freeswitch/sounds
storagedir:        /usr/local/freeswitch/storage
cachedir:          /usr/local/freeswitch/cache
-----
```

3-cü addım - `make` və `make install` alətlərini işə salaq

Öncəki addımdan sonra, FreeSWITCH üçün mənbə kodları olan qovluqda **Makefile** adlı bir fayl yaradılacaq. FreeSWITCH-in kompilyasiyası və yüklənməsi `make` utiliti sayəsində yerinə yetirilir. Önce `make` və sonra `make install` əmrini işə salın. Əksər istifadəçilər onların ikisini də bir sətirdə işə salır. “**&&**” simvolları əslində çox rahatdır və bu simvollar bildirir ki, əgər ilk əmin nəticəsi uğurla bitərsə, ikinci əmri yerinə yetir:

`[root@cosfs freeswitch]# make && make install`

configure skriptində olduğu kimi, **make** həmçinin iş gördüyü müddət uzun olacaq və kompilyasiya müddətində hansısa səhv baş verərsə öz işini dayandıracaq. Siz bu səhvi aradan qaldırmalı və yenidən kompilyasiya etməlisiniz. Adətən kompilyasiya və yüklənmə uğurlu olarsa, nəticədə aşağıdakı kimi çap ediləcək:

```
+----- Freeswitch install Complete -----+
+ FreeSWITCH has been successfully installed. +
+
+     Install sounds: +
+         (uhd-sounds includes hd-sounds, sounds) +
+             (hd-sounds includes sounds) +
+----- +
+                 make cd-sounds-install +
+                 make cd-moh-install +
+----- +
+                 make uhd-sounds-install +
+                 make uhd-moh-install +
+----- +
+                 make hd-sounds-install +
+                 make hd-moh-install +
+----- +
+                 make sounds-install +
+                 make moh-install +
+----- +
+     Install non english sounds: +
+         replace XX with language +
+             (ru : Russian) +
+             (fr : French) +
+----- +
+                 make cd-sounds-XX-install +
+                 make uhd-sounds-XX-install +
+                 make hd-sounds-XX-install +
+                 make sounds-XX-install +
+----- +
+     Upgrade to latest: +
+----- +
+                 make current +
+----- +
+     Rebuild all: +
+----- +
+                 make sure +
+----- +
+     Install/Re-install default config: +
+----- +
+                 make samples +
+----- +
+     Additional resources: +
+----- +
+         https://www.freeswitch.org +
+         https://freeswitch.org/confluence +
+         https://freeswitch.org/jira +
+         http://lists.freeswitch.org +
+
```

```
+      irc.freenode.net / #freeswitch
+
+      Register For ClueCon:
+      -----
+      https://www.cluecon.com
+
-----+
```

Əgər siz öncəki mesajı öz ekranınızda görürsünüzsə, demək FreeSWITCH uğurla kompilyasiya edilmiş və yüklənmişdir. Əgər çıxan səhv tanış deyilsə, siz mütləq FreeSWITCH-in əlaqəli cəmiyyəti <https://freeswitch.org/confluence/display/FREESWITCH/Community> ilə birbaşa əlaqə saxlamalısınız.

4-cü addım - modules.conf.xml faylında dəyişiklik

modules.conf.xml faylında modulların siyahısı olur hansı ki, FreeSWITCH işə düşən zaman onların hamisini yükleyir. Susmaya görə olan **modules.conf.xml** faylı **modules.conf** faylı ilə əlaqələnir. Kompilyasiya prosesində **modules.conf** faylında aktivləşdirdiyimiz modullar həmçinin **modules.conf.xml** faylında da olur ki, FreeSWITCH daemon işə düşəndə inzibatçı tərəfindən idarə edilə bilən olsun. Biz **mod_flite** aktivləşdirdiyimizə görə də, **modules.conf.xml** faylında da onu işə salmalıyıq.

modules.conf.xml faylı **conf/autoload_configs** alt qovluğunun içində yerləşir. Susmaya görə olan ünvan **/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml**-dir. Faylı mətn redaktoru ilə açın və aşağıdakı sətiri tapıb şərhləri silərək(yəni **<!-- -->**):

```
<!-- <load module="mod_flite"/> -->
```

Növbəti sətir kimi edirik:

```
<load module="mod_flite"/>
```

Sonra faylı yadda saxlayaraq çıxın.

Qeyd: *modules.conf* faylı ilə *modules.conf.xml* faylların fərqi nədir?? Fərq ondan ibarətdir ki, kompilyasiya vaxtı **modules.conf** faylı **make** əmri tərəfindən istifadə edilir ki, "#" simvolu ilə lazımi modulların qarşısından şərh silinsin və hazır olan paketin içində bu modulun imkanı olsun. Məhz kompilyasiyadan sonra modulun idarə edilməsi **conf/autoload_configs/modules.conf.xml** faylinə verilir.
modules.conf.xml faylı ümumi XML faylların bir hissəsidir və şərh olaraq **<!-- -->** simvollarından istifadə edir.

5-ci addım - səs və musiqi fayllarını yükləyək

Səs və musiqi faylları mütləq tələb edilmir ancaq, onlar məsləhətdir. Onlarsız sizin telefon sisteminizdə gözləmədə olan musiqi, voicemail və IVR nüsxəsi funksionallığı olmayıacaq. FreeSWITCH-də fərqli nüsxə səviyyələrinə görə səs və musiqi fayllarının müxtəlif versiyaları var. Məsləhətdir ki, onların hamisini istifadə edəsiniz ona görə ki, yüksək keçiricilik

qabiliyyəti olan şəbəkələrdə daha da keyfiyyətli səslərdən istifadə edə bilərsiniz.

Səs fayllarının yüklənilməsi üçün sadəcə FreeSWITCH mənbə kodlarının yerləşdiyi ünvana(yəni **/usr/src/freeswitch**) daxil olub aşağıdakı əmri yerinə yetirmək yetər.

```
[root@cosfs freeswitch]# make cd-sounds-install
```

Musiqi fayllarını yüklemək üçün isə aşağıdakı əmri yerinə yetirmək lazımdır:

```
[root@cosfs freeswitch]# make cd-moh-install
```

Bu əmr 8kHz, 16Khz, 32Khz səs səviyyələrinə görə musiqi və səs nüsxələrini internetdən endirəcək və yükleyəcək. FreeSWITCH uyğun olan səs və musiqi nüsxələrini musiqinin işə salınması ya da zəng edən üçün musiqi faylı kimi istifadə edəcək. Artıq FreeSWITCH-i işə sala bilərsiniz.

Üzvlük və yetkilərin təyin edilməsi

freeswitch adlı istifadəçi yaradaq və həmin istifadəçini **daemon** adlı qrupa əlavə edək. Yüklədiyimiz FreeSWITCH qovluğunun istifadəçi və qrup üzvlüyünü dəyişib **freeswitch** və **daemon** edək.

```
[root@cosfs freeswitch]# cd /usr/local/
[root@cosfs local]# useradd --system --home-dir /usr/local/freeswitch -G
daemon freeswitch
```

-l opsiyası sayesində **freeswitch** istifadəçisini bloklayırıq və bu opsiya yalnız root istifadəcisi tərəfindən yerinə yetirilə bilər:

```
[root@cosfs local]# passwd -l freeswitch
```

Locking password for user freeswitch.

```
passwd: Success
```

```
[root@cosfs local]# chown -R freeswitch:daemon /usr/local/freeswitch/
[root@cosfs local]# chmod -R 770 /usr/local/freeswitch/
[root@cosfs local]# chmod -R 750 /usr/local/freeswitch/bin/*
[root@cosfs local]# mkdir /var/run/freeswitch
[root@cosfs local]# chown -R freeswitch:daemon /var/run/freeswitch
```

/etc/init.d/freeswitch tərəfindən tələb edilir:

```
[root@cosfs local]# ln -s /usr/local/freeswitch/bin/freeswitch /usr/bin/
```

FreeSWITCH-in işə salınması və StartUP-a əlavə edilməsi

İlk dəfə sınaq üçün freeswitch-i işə saldıqdan çoxlu məlumatlar çıxışa çap ediləcək və nəticədə **FreeSWITCH** böyük hərflərlə olan yarıqrafik şəkil çap ediləcək. Bu uğurlu start deməkdir. Aşağıdakı əmrlə işə sala bilərsiniz:

```
[root@cosfs local]# /usr/local/freeswitch/bin/freeswitch
```

Son yoxlanış üçün aşağıdakı əmrlə sofia SIP-in vəziyyətinə baxa bilərsiniz(**help** əmri ilə bütün əmrlərin siyahısına baxa bilərsiniz):
freeswitch@cosfs.opensource.az> sofia status

Sistem qalxdıqda avtomatik işə düşsün

FreeSWITCH-in sistem qalxdıqda avtomatik işə düşməsi üçün sadəcə **init** scriptini **/etc/init.d** qovluğuna nüsxələmək lazımdır. Nüsxə faylı git repository-dən əldə etdiyimiz mənbə kodların içində build qovluğunda **freeswitch.init.redhat** adındadır. Əgər siz binar faylları fərqli ünvanda kompilyasiya etmişinizsə, bu skriptdə həmin ünvanları təyin edərək dəyişiklik etməlisiniz(bizim halda hər şey susmaya görə olduğundan lazım deyil). Aşağıdakı qaydada bütün ardıcılılığı edərək işə salırıq və sistemi yenidənyüklənmə edərək avtomatik işləməsini yoxlayırıq:

```
[root@cosfs ~]# cp /usr/src/freeswitch/build/freeswitch.init.redhat
/etc/init.d/freeswitch
[root@cosfs ~]# chmod 750 /etc/init.d/freeswitch
[root@cosfs ~]# chown freeswitch:daemon /etc/init.d/freeswitch
[root@cosfs ~]# chkconfig --add freeswitch && chkconfig --levels 2345
freeswitch on
[root@cosfs ~]# reboot
```

FreeSWITCH-in FreeBSD10.1 üçün yüklenməsi və quraşdırılması

FreeBSD 10.1 serverində FreeSWITCH-in yüklenməsinə başlayırıq. Öncə kompilyasiya mühiti yaratmalıyıq. Məhz bunun üçün də tələb edilən paketləri yükləyirik.

```
pkg install autoconf automake curl git gmake jpeg ldns libedit libtool
openssl pcre pkgconf speex sqlite3 wget sudo
```

```
mkdir ~/src          # Mənbə kodları yükleyəcəyimiz ünvanı yaradırıq
cd ~/src           # Mənbə kodların ünvanına daxil oluruq
```

Oeyd: GIT haqqında <https://git-scm.com/> rəsmi səhifəsindən oxuya bilərsiniz.

Mənbə kodları daxili qovluğumuza sinxronizasiya edirik:

```
git clone -b v1.5.final https://stash.freeswitch.org/scm/fs/freeswitch.git
```

```
cd freeswitch      # clone edilən qovluğa daxil oluruq
./bootstrap.sh -j # Kompilyasiya mühitini hazırlayırıq
./configure        # Config edirik(Bitdikdən sonra aşağıdakı sətirləri
                     # görməliyik)
----- FreeSWITCH configuration -----
Locations:
FHS enabled:      no

prefix:            /usr/local/freeswitch
exec_prefix:       ${prefix}
bindir:            ${exec_prefix}/bin
sysconfdir:        /usr/local/freeswitch/conf
libdir:            ${exec_prefix}/lib

certsdir:          /usr/local/freeswitch/certs
dbdir:             /usr/local/freeswitch/db
```

```

grammardir:      /usr/local/freeswitch/grammar
htdocsdir:       /usr/local/freeswitch/htdocs
logfiledir:      /usr/local/freeswitch/log
modulesdir:      /usr/local/freeswitch/mod
pkgconfigdir:    ${exec_prefix}/lib/pkgconfig
recordingsdir:   /usr/local/freeswitch/recordings
runtimedir:      /usr/local/freeswitch/run
scriptdir:       /usr/local/freeswitch/scripts
soundsdir:       /usr/local/freeswitch/sounds
storagedir:      /usr/local/freeswitch/storage
cachedir:        /usr/local/freeswitch/cache
-----
```

```

gmake          # Kompilyasiyaya başlayırıq

sudo gmake install cd-sounds-install cd-moh-install  # səsləri və imkanları
                                                               # yükləyirik
```

/usr/local/etc/rc.d/freeswitch faylı yaradırıq və tərkibinə aşağıdakı sətirləri əlavə edirik. Bu fayl freeswitch üçün startup scriptdir hansı ki, sistem yenidənyüklənməsindən sonra işə salınması üçündür.

```

#!/bin/sh
#
# PROVIDE: freeswitch
# REQUIRE: LOGIN cleanvar
# KEYWORD: shutdown
#
# Add the following lines to /etc/rc.conf to enable freeswitch:
# freeswitch_enable:      Set it to "YES" to enable freeswitch.
#                         Default is "NO".
# freeswitch_flags:        Flags passed to freeswitch-script on startup.
#                         Default is "".
#
. /etc/rc.subr
name="freeswitch"
rcvar=${name}_enable
load_rc_config $name
: ${freeswitch_enable="NO"}
: ${freeswitch_pidfile="/usr/local/freeswitch/run/freeswitch.pid"}
start_cmd=${name}_start
stop_cmd=${name}_stop
pidfile=${freeswitch_pidfile}
freeswitch_start() {
    /usr/local/freeswitch/bin/freeswitch ${freeswitch_flags}
    echo -n "Starting FreeSWITCH: "
}
freeswitch_stop() {
    /usr/local/freeswitch/bin/freeswitch -stop
}
run_rc_command "$1"
```

```
chmod u-w,ugo+x /usr/local/etc/rc.d/freeswitch - Scripti yerinə
yetirən edirik

/etc/rc.conf faylina tələb edilən sətirləri əlavə edirik ki, reboot-dan sonra
freeswitch-i işə salsın:
freeswitch_enable="YES"
freeswitch_flags="-nc"

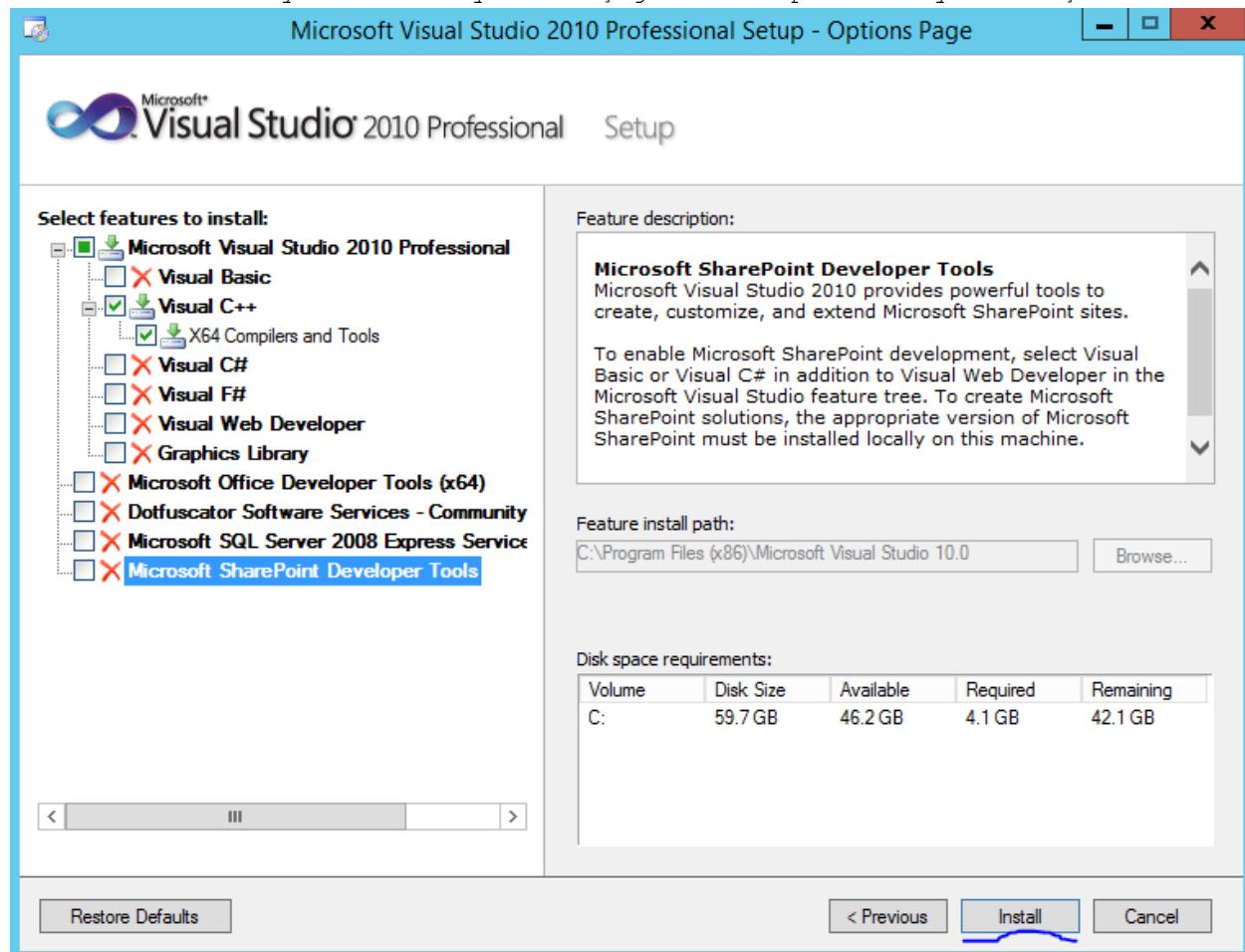
Hal-hazırda root adlı istifadəçimizin SHELL mühiti CSH olduğuna görə,
/root/.cshrc faylında path dəyişənini aşağıdakı formaya gətiririk(freeswitch-
in binar faylları /usr/local/freeswitch/bin ünvanında yerləşir):
set path = (/sbin /bin /usr/sbin /usr/bin /usr/local/freeswitch/bin
/usr/local/sbin /usr/local/bin $HOME/bin)

-nc - no console deməkdir ancaq, siz sonra fs_cli əmri ilə console-a daxil
ola biləcəksiniz
-nonat - Əgər sizin freeswitch-in PUBLIC IP ünvanı varsa və o NAT arxasında
işləmirsə, bu parametr istifadə edilir(Bu freeswitch üçün NAT
traversal parametrini söndürür).
```

FreeSWITCH-in Windows 2012 serverdə kompilyasiyası və yüklənməsi
Bu kitab yazılıan müddət müəllif **1.2** versiyasından istifadə etdiyinə görə
həmin vaxt üçün də, Visual Studio 2010 istifadə edilirdi. Hal-hazırda en yeni
versiyani kompilyasiya edib yükləmək üçün **VS2013** professional ya da ultimate
tələb edilir. Məhz bu səbəbdən həm visual studio 2010 və həmdə 2013 üzərində
edilən işlər ardıcılıqla qeyd edilmişdir.

Windows 2012 serverdə Visual Studio 2010 professional vasitəsilə
FreeSWITCH-in kompilyasiyası və yüklənməsi
Programçı olmadığınıza görə sizin əslində kompilyasiya etməyə itirəcəyiniz
vaxtin getməsinə ehtiyac yoxdur. Sadəcə
<http://files.freeswitch.org/windows/installer/> ünvanından **x64** və ya **x86**
platforma üçün **freeswitch.msi** paketi endirib yükləməniz yetər. Binar fayllar
haqqında daha da ətraflı məlumat əldə etmək istəyirsinizsə,
https://freeswitch.org/confluence/display/FREESWITCH/Windows#Precompiled_Binaries
linkinə baxa bilərsiniz. Öncədən məlumat veririk: Çalışmayıñ ki, 2010-a
aid olan Visual Studio faylı isual Studio 2012-də yerinə yetirəsiniz. Hal-
hazırda Windows 2012 serverimizdə VS2010-Professional yüklənmiş
vəziyyətdədir(Mənim yüklədiyim versiya

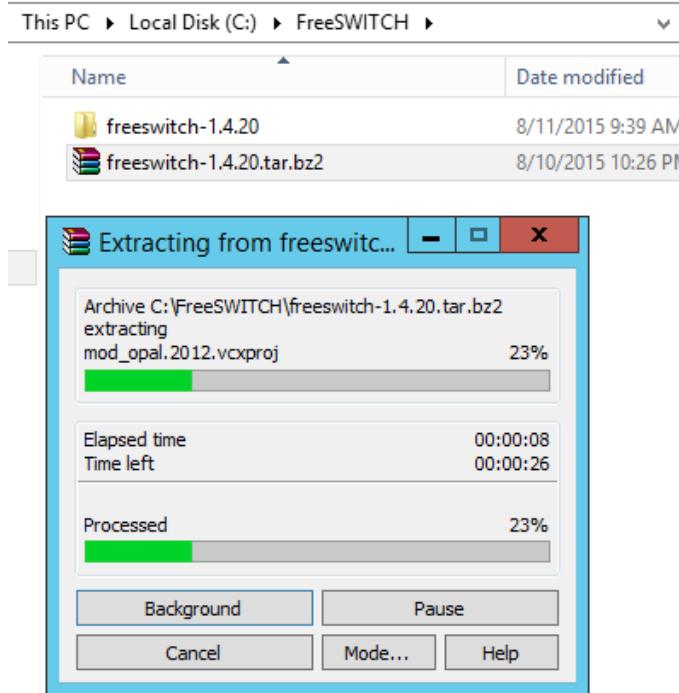
SW_DVD9_Visual_Studio_Pro_2010_English_Core_MLF_X16-76715). Microsoft Visual C++ Professional yüklenindikdə yalnız aşağıdakı komponentlər yüklənmişdir:



Yükləməyə və kompilyasiya etməyə başlayaq:

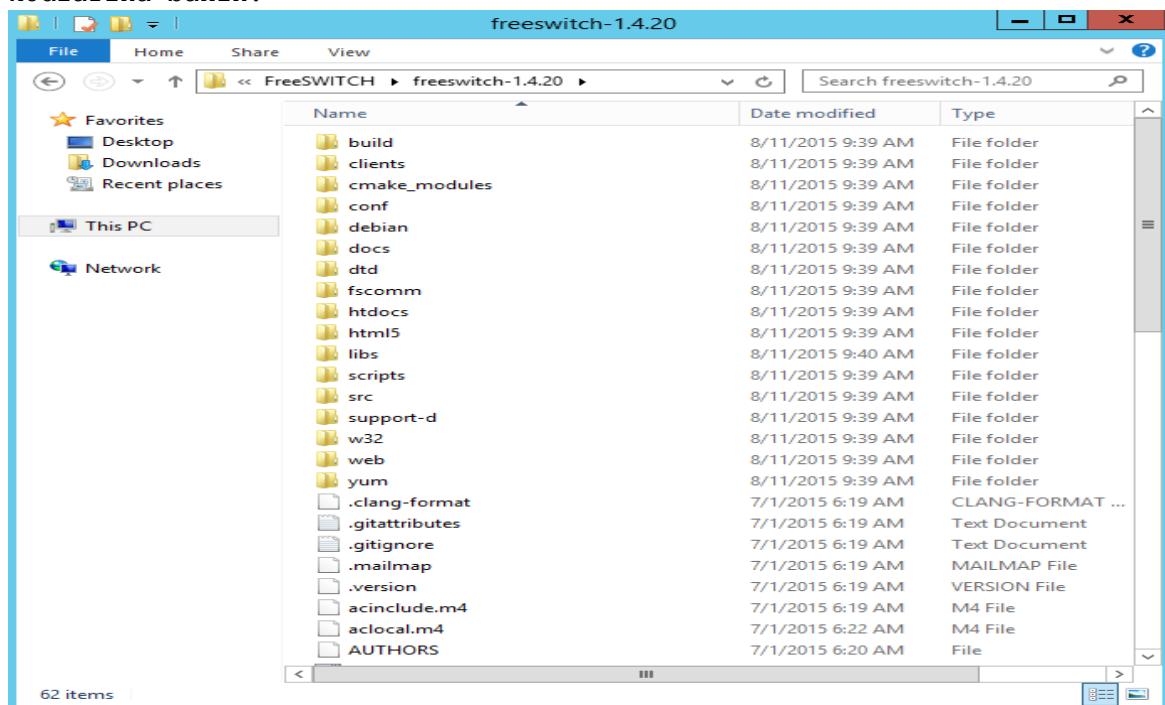
1. C: diskimizdə **FreeSWITCH** adlı bir qovluq yaradaq və <http://files.freeswitch.org/releases/freeswitch/freeswitch-1.4.20.tar.bz2> URL-indən ən son mənbə kodlarını həmin qovluğa endirib WinRAR vasitəsilə açaq. Ünvan bu şəkildə olacaq:
C:\FreeSWITCH\freeswitch-1.4.20.tar.bz2

2. C:\FreeSWITCH qovluğuna daxil olub **freeswitch-1.4.20.tar.bz2** faylinin üstündə sağ düyməni sıxaraq **WinRAR** vasitəsilə **Extract Here** deyirik:

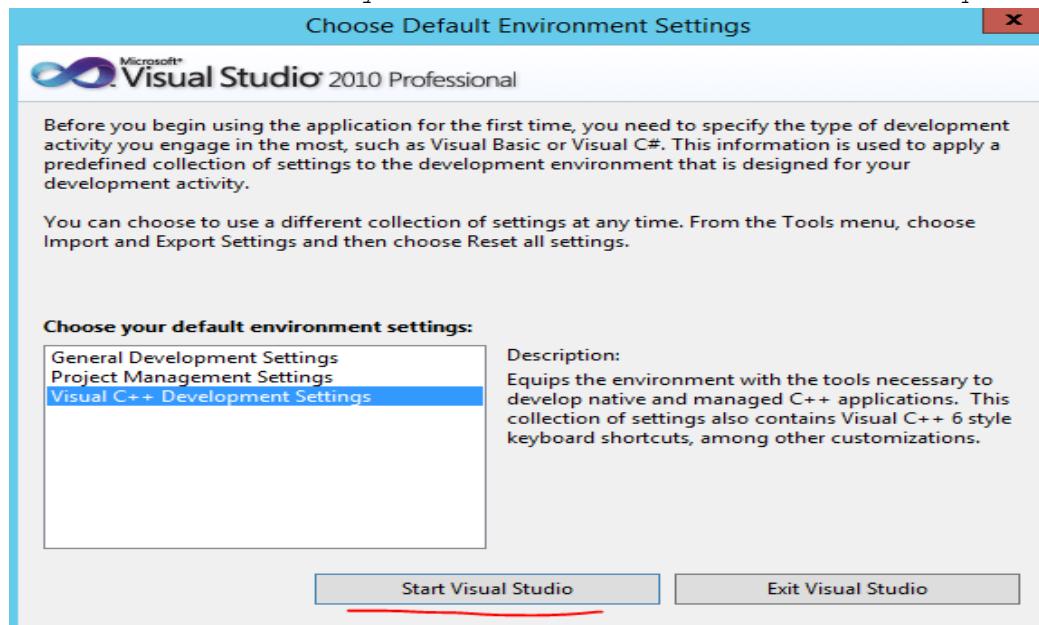


Qeyd: Winrar eyni zamanda həm BZ2-ni açır və arxivdən çıxarır.

3. Gördüyüümüz kimi açılmadan sonra, bizim **freeswitch-1.4.20** adlı qovluğunuz olacaq. Qovluğun üstündə iki dəfə sıxıb daxil olun və mənbə kodlarına baxın:

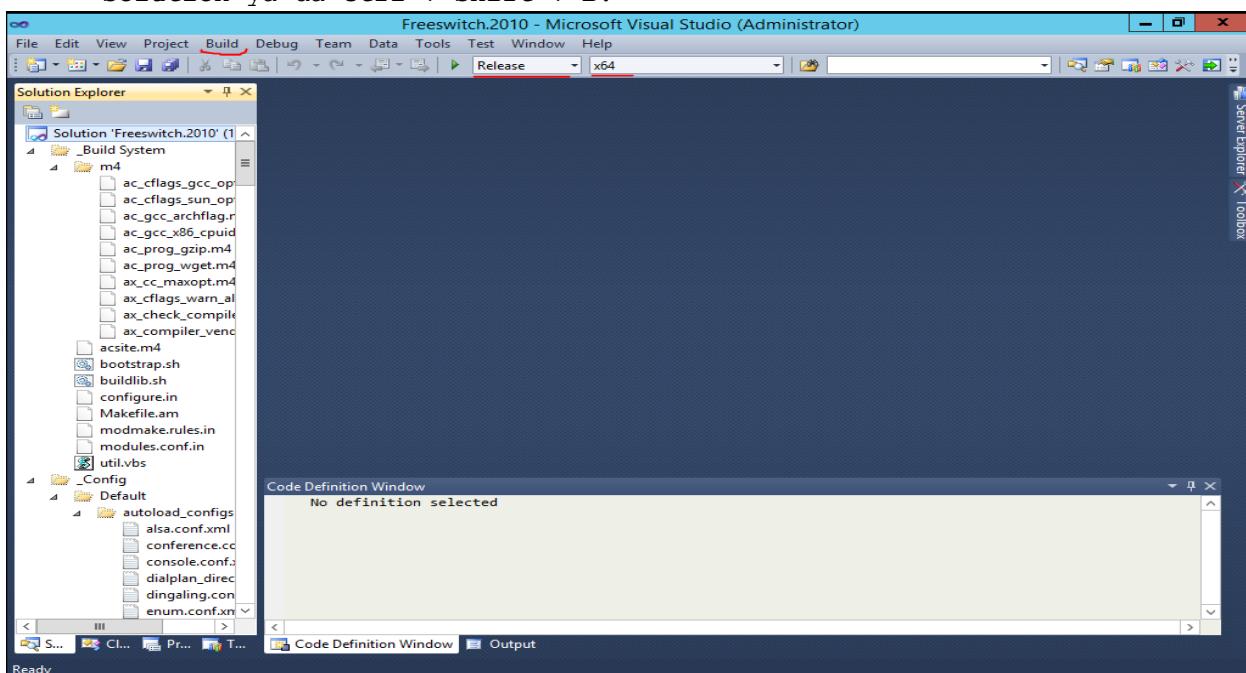


4. Mənbə kodları yerləşən ünvanda biz iki ədəd vacib fayl görəcəyik. Onlardan biri **MSVC** üçün **FreeSWITCH.2010.sln** və **MSVCEE** üçün isə **Freeswitch2010.express.sln** olacaq. Yüklədiyimiz MSCV-ə aid olan **FreeSWITCH.2010.sln** faylin üstündə mouse-la iki dəfə sıxırıq:

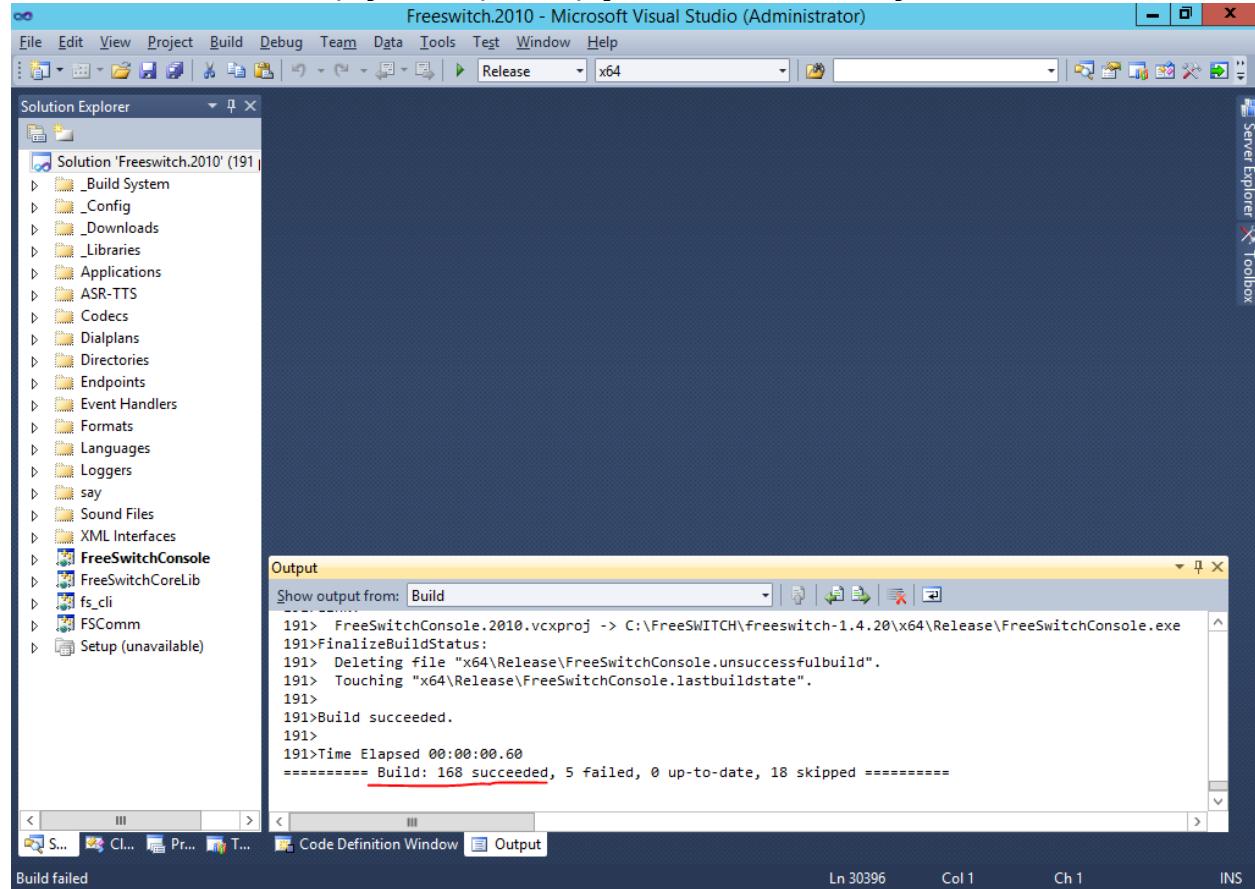


Yüklənmə müddətində **mod_managed** və **.vcproj** ilə bağlı çıxan səhvə fikir verməyin.

5. Bütün mənbə kodları VC2010-a yükləndikdən sonra, Panel-də **Debug** əvəzinə **Release** seçirik və sonra menyuda **Build -> Build Solution** ya da **F7**. Əgər siz Visual Studio 2010 IDE istifadə edirsinizsə, onda **Build -> Build Solution** ya da **Ctrl + Shift + B**.



Artıq bundan sonra kompilyasiya başlayacaq və ekrana çoxlu mesajlar çap ediləcək. Nəticədə aşağıdakı şəkil çap ediləcək və bu uğurlu nəticə deməkdir:

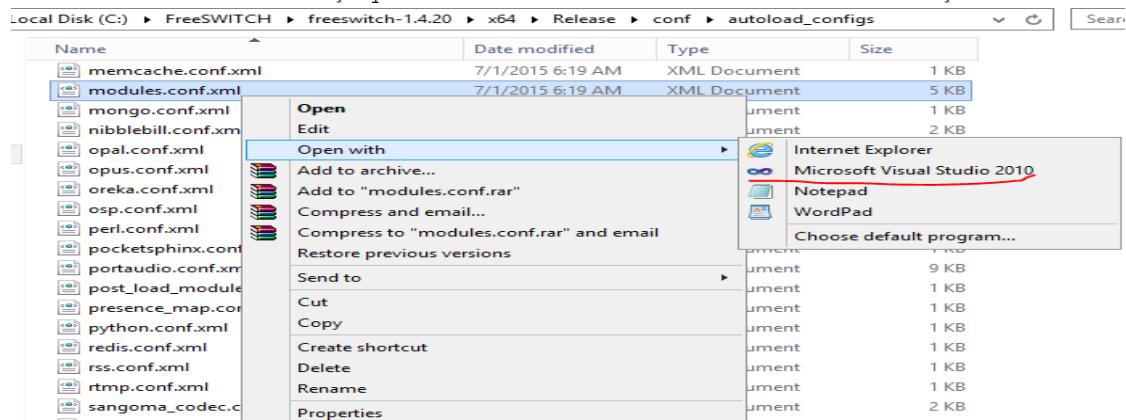


Qeyd: Adətən **UNIX/Linux** OS-lar üzərində əlimizlə gördüyüümüz işləri

MSVC/MSVCEE özü avtomatik olaraq edəcək. Bura həmçinin səs, musiqi faylları və Flite(text-to-speech), PocketSphinx(speech recognition) kimi kompilyasiya da daxildir. Ancaq, bu istəkdən asılı olan modulların FreeSWITCH-in işə salınması müddətində avtomatik olaraq işə düşməsini istəyirsizsə, **modules.conf.xml** quraşdırma faylında aktivləşdirilməlisiniz. Əgər siz PocketSphinx haqqında daha da ətraflı oxumaq istəsəniz, <http://cmusphinx.sourceforge.net/wiki/start> linkinə müraciət edə bilərsiniz. Yenidən kompilyasiy qovluğunda Windows Explorer vasitəsilə qayidian və x64 platformlu kompilyasiya seçdiyinizə görə **x64\Release** qovluğunun yarandığını görəcəksiniz(Tam ünvan: **C:\FreeSWITCH\freeswitch-1.4.20\x64\Release**). Bu FreeSWITCH-in yüklenmə qovluğudur. FreeSWITCH-in işə salınmasından once son görəcəyimiz iş **modules.conf.xml** faylında səliqə ilə **mod_flite** işə salmaq lazımdır ki, FreeSWITCH-i işə saldıqda avtomatik olaraq o da işə düşsün. Biz **mod_flite**-i text-to-speech(TTS) motoru üçün müxtəlif misallarda bu kitabda istifadə edəcəyik.

6. VS2010-Professional GUI-də **conf** qovluğunun altında **autoload_configs** qovluğunu açırıq(**C:\FreeSWITCH\freeswitch-1.4.20\x64\Release\conf\autoload_configs**) və mətn redaktoru ilə

modules.conf.xml faylını açırıq. Açıqdıqda bizdən soruşacaq ki, hansı mətn redaktoru ilə açaq və Microsoft Visual Studio 2010 seçirik:

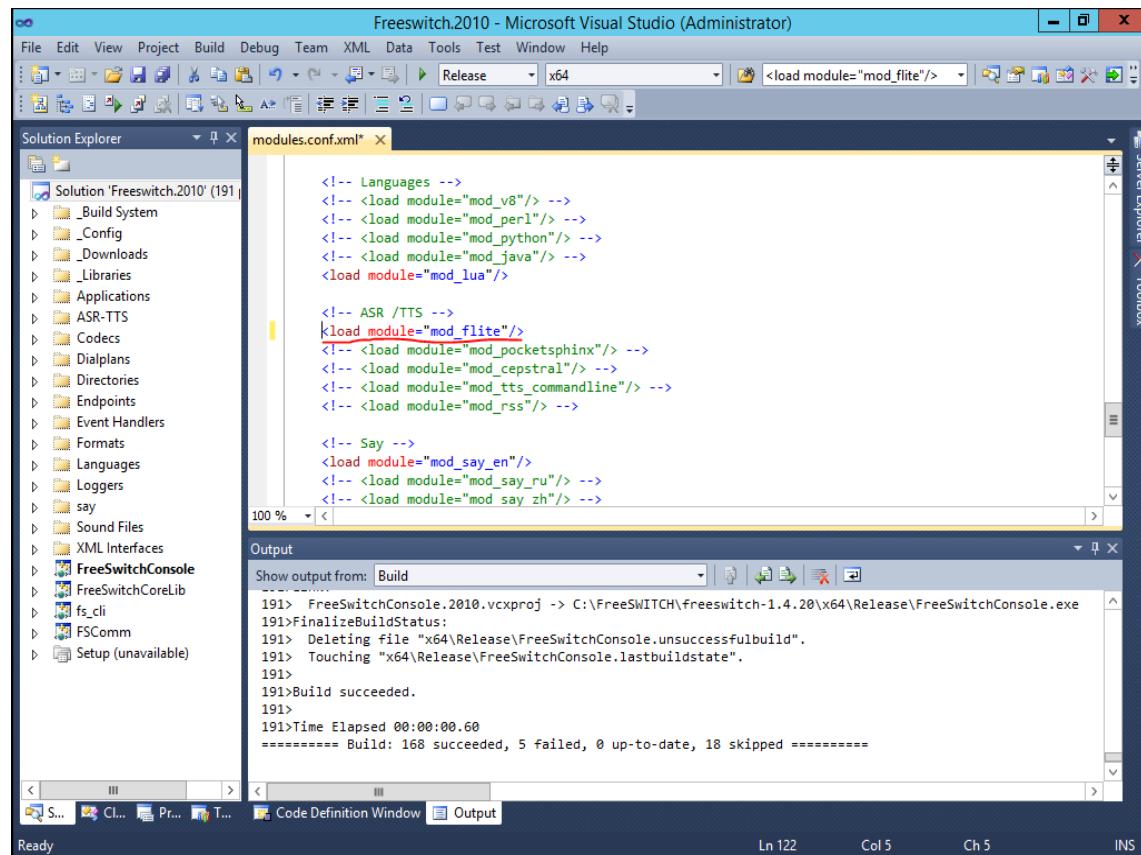


7. Aşağıdakı sətiri tapırıq, əvvəlində `<!--` və sonunda `-->` olan simvolları silirik ki, sətir işə düşsün(Bu simvollar şərhdir):

`<!-- <load module="mod_flite"/> -->`

Sildikdən sonra bu şəkildə olmalıdır:

`<load module="mod_flite"/>`



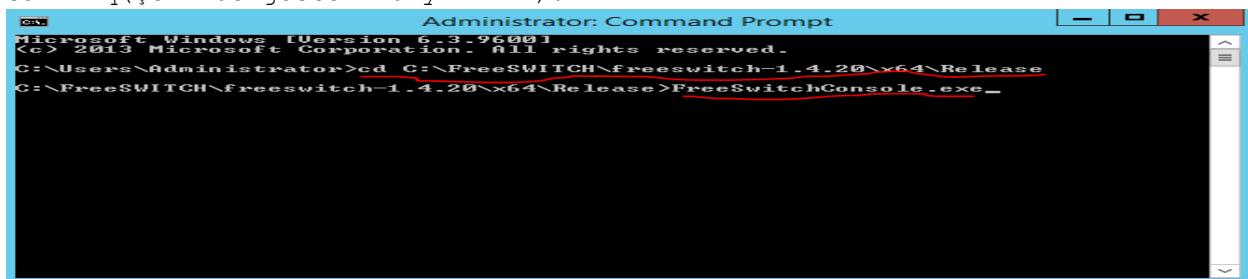
8. Faylı yadda saxlayın və mətn redaktorundan çıxın. Siz artıq FreeSWITCH-i ilk dəfə işə sala bilərsiniz.

FreeSWITCH-in işə salınması

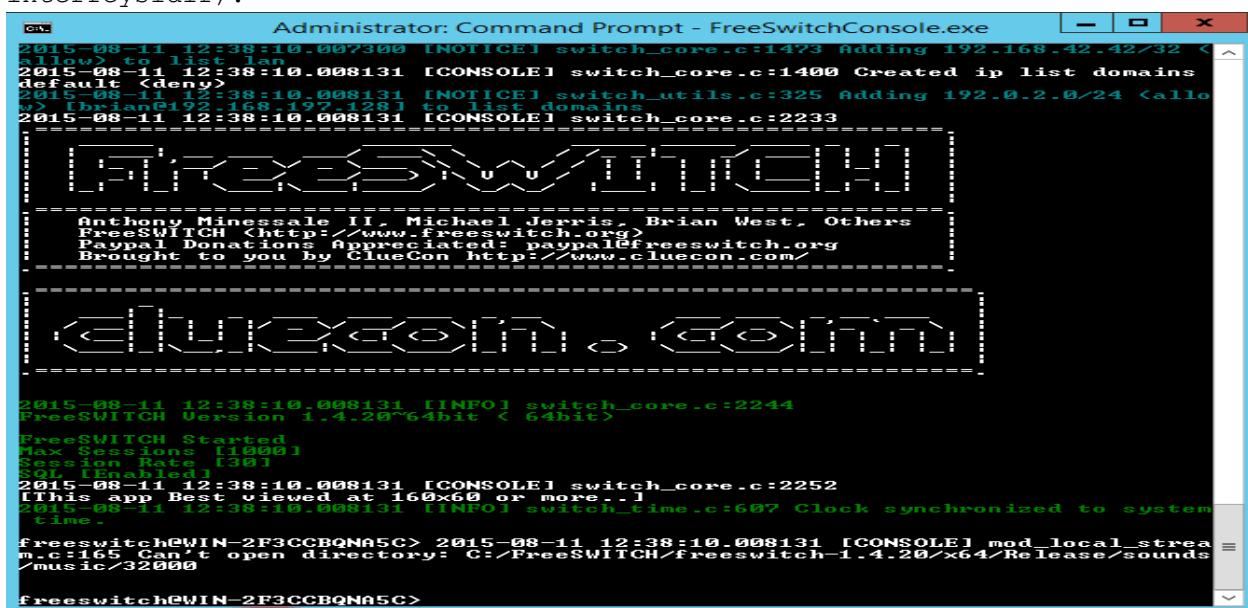
Artıq FreeSWITCH-in kompilyasiya edilməsi və yüklənməsindən sonra siz onu işə sala bilərsiniz:

- Linux/UNIX OS-larda siz yüklənmə ünvani `/usr/local/freeswitch` seçdiyinizi görə programın ünvani `/usr/local/freeswitch/bin/freeswitch`-dir.
- Windows-da işə bu ünvan `freeswitchconsole.exe` faylıdır və biz **x64** üçün kompilyasiya elədiyimizə görə o `C:\FreeSWITCH\freeswitch-1.4.20\x64\Release` ünvanında olacaq.

Windows `cmd` açırıq və `cd C:\FreeSWITCH\freeswitch-1.4.20\x64\Release` əmri ilə ünvana daxil oluruq. Sonra `FreeSwitchConsole.exe` əmri ilə FreeSWITCH-i işə salırıq (Şəkildə göstərildiyi kimi):



Sistem yüklənməyə başlayacaq və ekrana çoxlu mesajlar çap edilecək. Çıxan mesajlardan narahat olmayın sadəcə sonda aşağıdakı console sətirini nəticə olaraq əldə etmiş olmalısınız (Bu artıq FreeSWITCH-in command line interfeysidir):



Gördüyüümüz kimi, yüklənmiş FreeSWITCH versiyası 1.4.20-dir. Ancaq yenə də version əmrini daxil edib versiyaya baxaq və status əmri ilə bəzi statistikalara baxaq:

Administrator: Command Prompt - FreeSwitchConsole.exe

```
freeswitch@WIN-2F3CCBQNA5C> version
FreeSWITCH Version 1.4.20~64bit < 64bit>

freeswitch@WIN-2F3CCBQNA5C>
freeswitch@WIN-2F3CCBQNA5C>
freeswitch@WIN-2F3CCBQNA5C> status
UP 0 years, 0 days, 0 hours, 6 minutes, 16 seconds, 936 milliseconds, 628 microseconds
FreeSWITCH <Version 1.4.20 64bit> is ready
0 session(s) since startup
0 session(s) - peak 0, last 5min 0
0 session(s) per Sec out of max 30, peak 0, last 5min 0
1000 session(s) max
min idle cpu 0.00/99.51

freeswitch@WIN-2F3CCBQNA5C> _
```

FreeSWITCH-in işləməsinə və **5060**-ci portda həqiqətən qulaq asmasına əmin olmaq istəyiriksə, Windows CMD-dən **netstat -na | findstr 5060** əmri ilə aşağıdakı kimi buna baxırıq:

```
C:\Users\Administrator> netstat -na | findstr 5060
TCP 192.168.197.128:5060 0.0.0.0:0 LISTENING
UDP 192.168.197.128:5060 *:*
```

Həmçinin Windows CMD-dən 5060-ci porta **telnet localhost 5060** əmri ilə telnet atırıq və yoxlayırıq ki, qoşulmaları qəbul edir(Əgər cursor yanıb sönürse hər şey qaydasındadır).

Öyrənəcəyiniz çoxlu əmrlər var və onların tam siyahısını əldə etmək üçün **help** əmrini daxil edib **Enter** sıxın. Sonda FreeSWITCH-i dayandırmaq üçün **fsctl shutdown** əmrini daxil edib dayandırmaq lazımdır(Əgər siz **freeswitchconsole.exe** əmrini CLI-dən işə salmışdınızsa, sadəcə pəncərəni bağlamağınız yetər).

Windows машında FreeSWITCH-in startup-a əlavə edilməsi

FreeSWITCH-in startup-a əlavə edilməsi üçün Windows-da bəzi işlər görmək lazımdır. Bunlar aşağıda sadalanır:

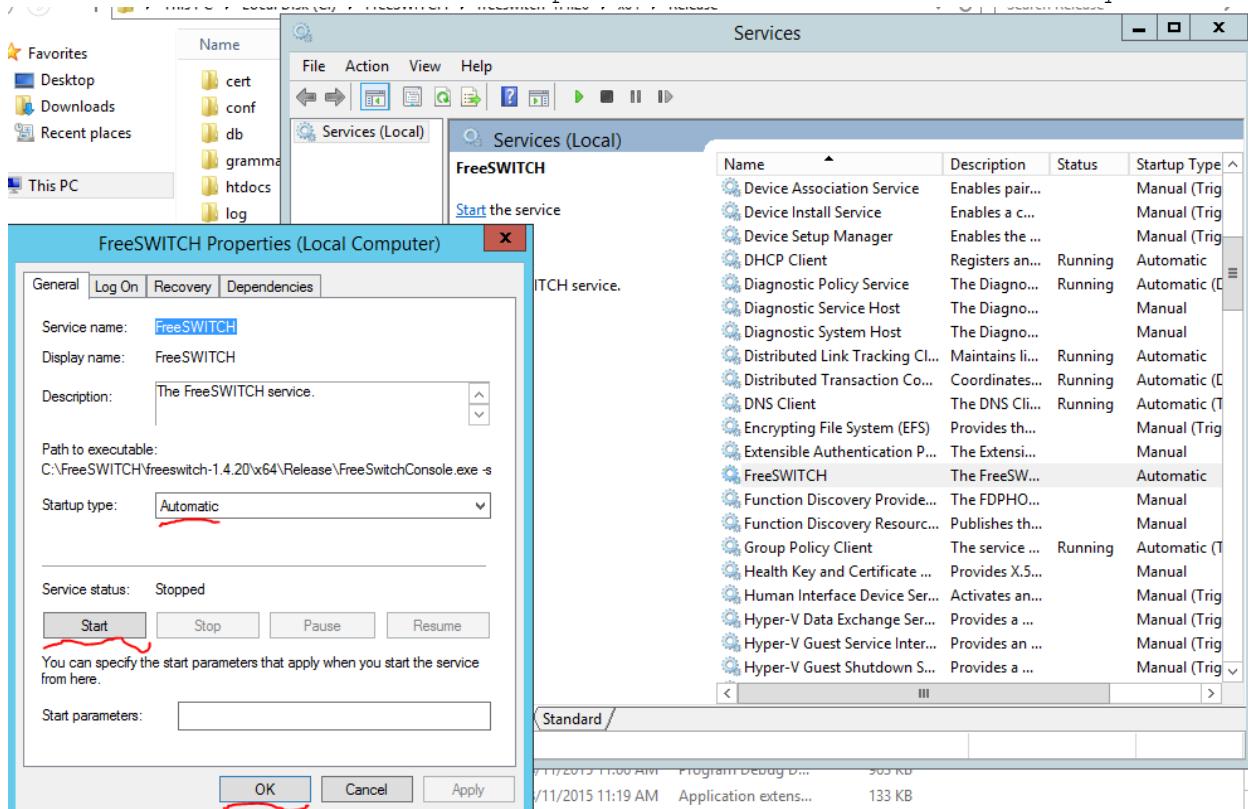
1. Windows command-line-i açırıq(**Windows+R** -> **cmd** əmrini daxil edib Enter sıxırıq).
2. Aşağıdakı əmrlə FreeSWITCH yüklənmiş qovluğa daxil oluruq:
cd C:\FreeSWITCH\freeswitch-1.4.20\x64\Release
3. Aşağıdakı şəkildəki kimi, **freeswitchconsole.exe** əmrini **-install** arqumeti ilə işə salırıq:

Administrator: C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd C:\FreeSWITCH\freeswitch-1.4.20\x64\Release
C:\FreeSWITCH\freeswitch-1.4.20\x64\Release>freeswitchconsole.exe -install
C:\FreeSWITCH\freeswitch-1.4.20\x64\Release>_
```

- Sonra servislərin siyahısına əlavə edirik ki, system yenidənyüklənməsində avtomatik işə düşsün. Bunun üçün **Windows+R** -> və **services.msc** daxil edib **Enter** sıxırıq. Aşağıdakı şəkildəki kimi, FreeSWITCH-i servislərdən tapıb mousla üstündə iki dəfə sıxırıq:



- Startup type**-ni **Automatic** edərək **Start** düyməsinə sıxırıq.
- Nəticəni yoxlamaq üçün işə, windows CLI ilə **C:\FreeSWITCH\freeswitch-1.4.20\x64\Release** qovluğuna daxil oluruq və **fs_cli.exe** əmrini daxil edirik. Xoş gəlmisiniz səhifəsi açılacaq. Sınaq üçün **status** əmrini daxil edib nəticəyə baxın.
- Sonda **/exit** əmrini daxil edərək **fs_cli.exe** programından çıxış edin.

Administrator: Command Prompt

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd C:\FreeSWITCH\freeswitch-1.4.20\x64\Release
C:\FreeSWITCH\freeswitch-1.4.20\x64\Release>fs_cli.exe
```

FS CLI

 Anthony Minassale II, Ken Rice,
 Michael Jerris, Travis Cross
 FreeSWITCH <<http://www.freeswitch.org>>
 Paypal Donations Appreciated: paypal@freeswitch.org
 Brought to you by ClueCon <http://www.cluecon.com/>

ClueCon.COM

Type /help <enter> to see a list of commands

[This app Best viewed at 160x60 or more...]
+OK log level [?]
freeswitch@internal> status
UP 0 years, 0 days, 0 hours, 5 minutes, 38 seconds, 233 milliseconds, 200 microseconds
FreeSWITCH <Version 1.4.20 64bit> is ready
0 session(s) since startup
0 session(s) - peak 0, last 5min 0
0 session(s) per Sec out of max 30, peak 0, last 5min 0
1000 session(s) max
min idle cpu 0.00/98.44
freeswitch@internal> /exit
C:\FreeSWITCH\freeswitch-1.4.20\x64\Release>

Qeyd: Gördüyüümüz bütün işlərin doğruluğunu yoxlamaq üçün sonda sistemə mütləq yenidənyüklənmə edin və **netstat -na|findstr 5060** əmri ilə FreeSWITCH-in işləməsini yoxlayın.

Artıq sizin Windows servislərdə işləyən FreeSWITCH serveriniz var.

Windows 2012 serverdə Visual Studio 2013 ultimate vasitəsilə FreeSWITCH-in kompilyasiyası və yüklənməsi

MS VS2013Ultimate yükleyərkən aşağıdakı opsiyanı seçirik:



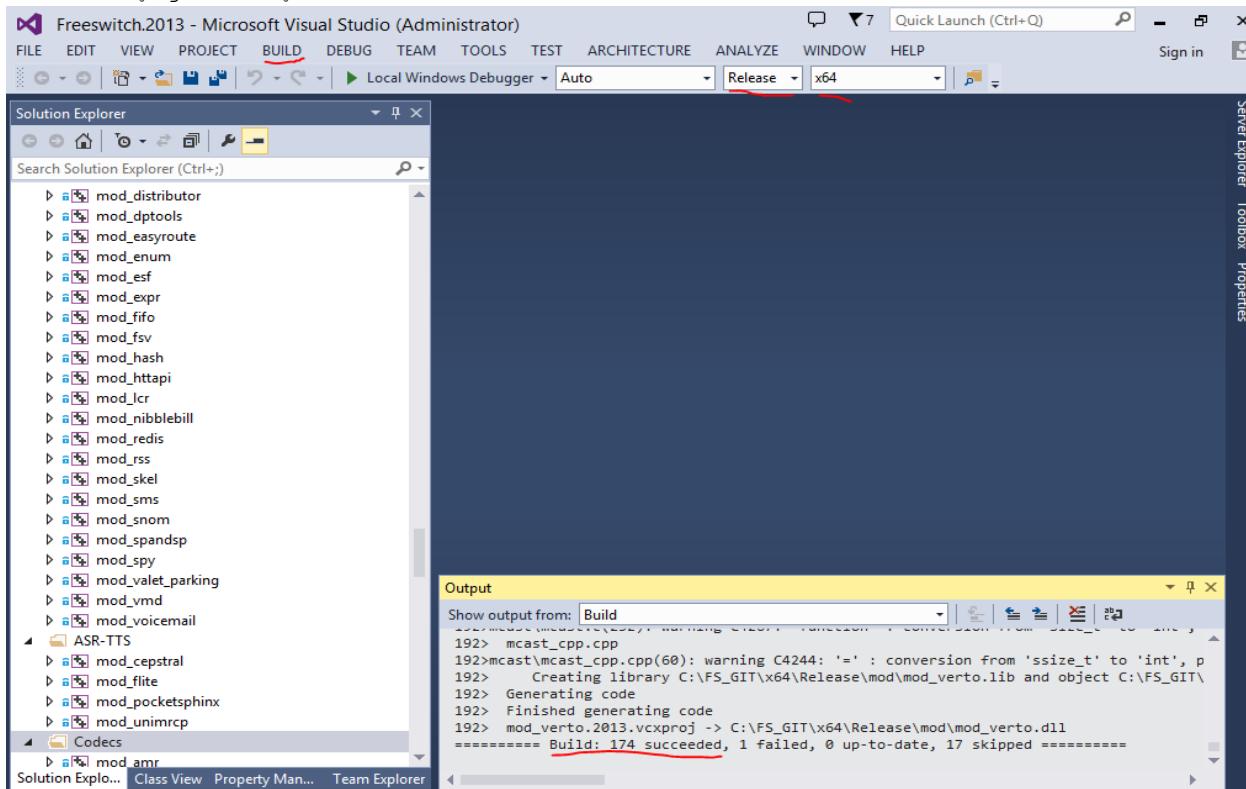
Windows 2012 serverimize GIT yüklenildikdən sonra aşağıdakı əmrlə FreeSWITCH-in ən yeni mənbə kodlarını serverimizə təyin elədiyimiz qovluğa endiririk:

```
C:\Users\Administrator>git clone
https://stash.freeswitch.org/scm/fs/freeswitch.git C:/FS_GIT/
```

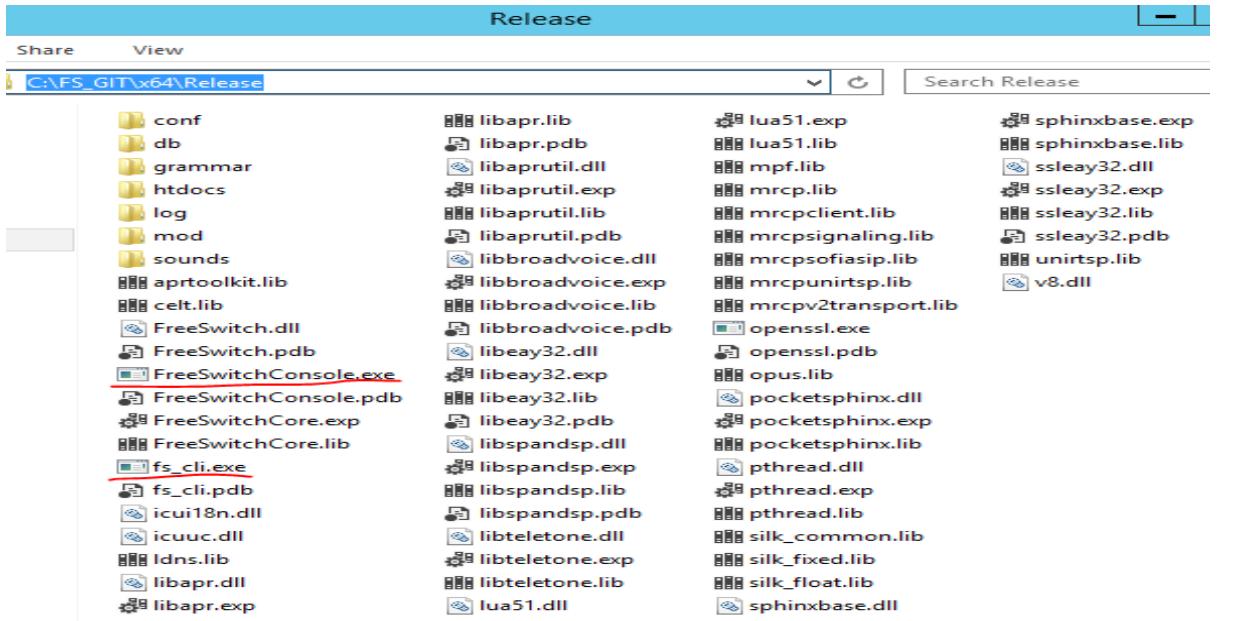
Sonra **C:/FS_GIT/** qovluğunda olan **Freeswitch.2013.sln** faylını **VS2013Ultimate** vasitəsilə açırıq. Menyu panelində **Debug** əvəzinə **Release** və platforma olaraq **x64** seçirik. Ardınca menyuda **Build -> Build Solution** (Ya da **F7**)

Qeyd: Unutmayın ki, kompilyasiya müddəti səs və musiqi faylları internetden endirilir və Windows 2012 serverin internete birbaşa çıxışı olmalıdır.

Nəticə aşağıdakı şəkildəki kimi olmalıdır:



Kompilyasiya bitdikdən sonra, **C:\FS_GIT\x64\Release** qovluğunda **FreeSwitchConsole.exe** adında fayl yaradılmalıdır. Aşağıdakı şəkildəki kimi:

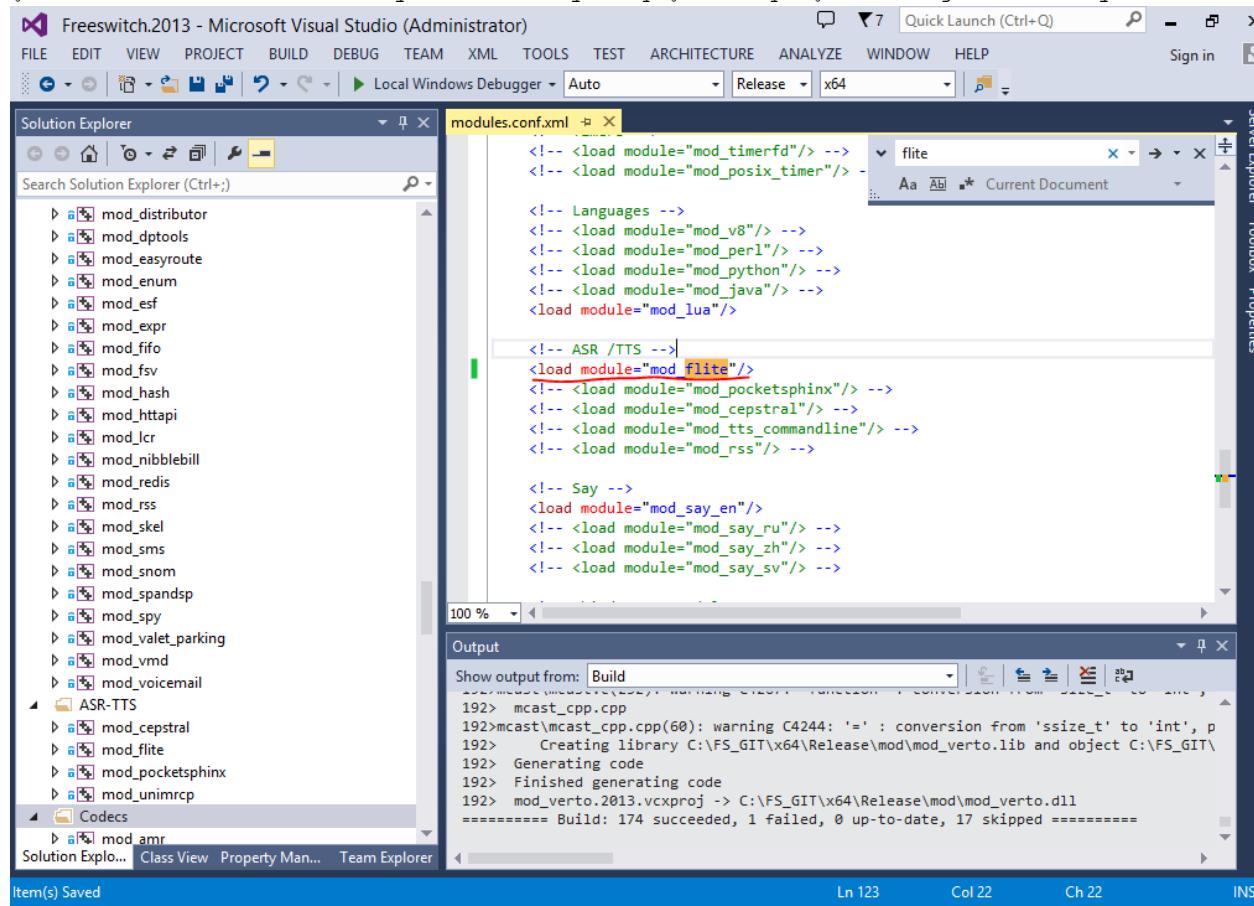


Əmrliə "cd C:\FS_GIT\x64\Release" windows CLI-dan ünvana daxil oluruq və **FreeSwitchConsole.exe** faylını işə salırıq(**version** əmrni daxil edib Enter sıxırıq):

Başqa bir Windows CLI açırıq və 5060-ci portun qulaq asdığını yoxlayırıq:
C:\Users\Administrator> **netstat -na|findstr 5060**

```
TCP      192.168.197.128:5060      0.0.0.0:0      LISTENING
UDP      192.168.197.128:5060      *:*
```

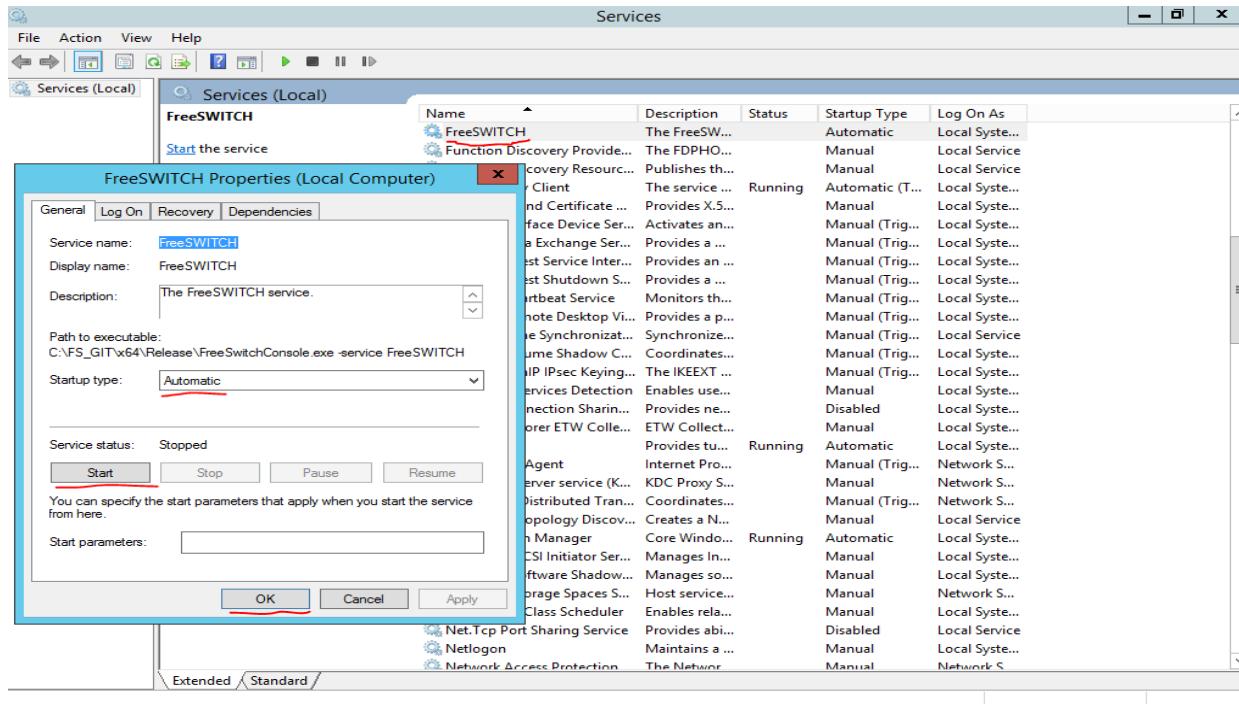
Sonra yənə CLI-dan **C:\FS_GIT\x64\Release\conf\autoload_configs** ünvanına daxil oluruq və **modules.conf.xml** faylinin üstündə iki dəfə sıxırıq. Faylin açılması üçün Microsoft Visual Studio 2013 seçirik. Aşağıdakı şəkildəki kimi, **<!-- <load module="mod_flite"/> -->** sətirini tapırıq və **<!-- -->** simvollarını silirik ki, şərh olmasın və ardınca yadda saxlayaraq çıxırıq. Şəkildə göstərildiyi kimi:



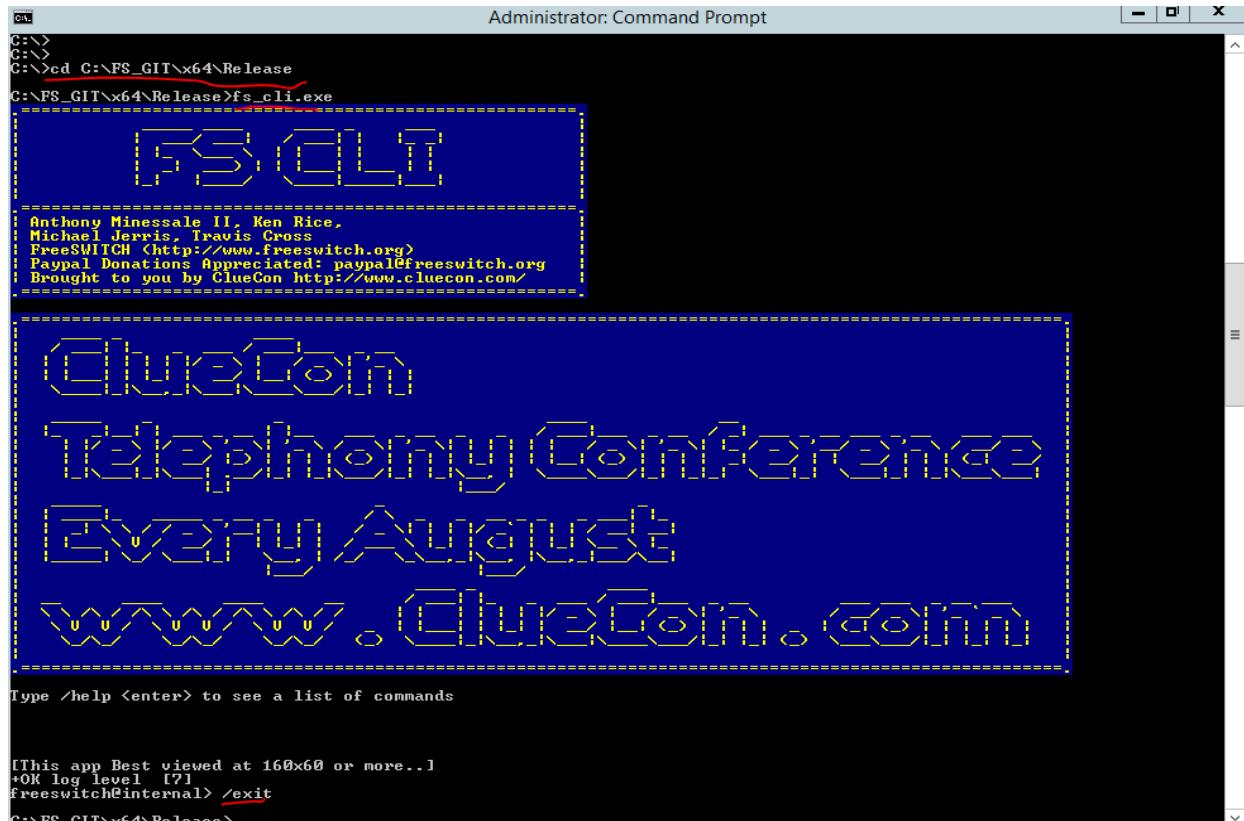
Starup-a əlavə etmək üçün isə, windows CLI-dan **C:\FS_GIT\x64\Release** ünvanına daxil olduqdan sonra, **freeswitchconsole.exe -install** əmrini daxil edirik ki sistemə yüklənsin. Aşağıdakı kimi:

```
C:\Users\Administrator>cd C:\FS_GIT\x64\Release
C:\FS_GIT\x64\Release>freeswitchconsole.exe -install
```

Sistem yenidən yüklenməsindən sonra işləməsi üçün eynilə **Windows+R -> services.msc** və şəkildəki kimi, **FreeSWITCH** seçib mouse-la iki dəfə sıxırıq. Startup type: **Automatic** -> **Start** -> **Ok**



Sonda **C:\FS_GIT\x64\Release** ünvanına daxil olurug və şəkildə göstərildiyi kimi, **fs_cli**-a daxil olurub **/exit** əmri ilə çıxırıq:



```

Administrator: Command Prompt
C:\>
C:\>>cd C:\FS_GIT\x64\Release
C:\FS_GIT\x64\Release>fs_cli.exe
=====
[FS] [CLI]
=====
Anthony Minassale II, Ken Rice,
Michael Jerris, Travis Cross
FreeSWITCH <http://www.freeswitch.org>
Paypal Donations Appreciated: paypal@freeswitch.org
Brought to you by ClueCon http://www.cluecon.com/
=====

ClueCon
TELEPHONY CONFERENCE
ENGINEERING
www.ClueCon.com

Type /help <enter> to see a list of commands

[This app Best viewed at 160x60 or more..]
+OK log level [?]
freeswitch@internal> /exit
C:\FS_GIT\x64\Release>

```

Sistemə yenidənyüklənmə edib, windows CLI-dan aşağıdakı əmri daxil edərək FreeSWITCH servisin işləməsini yoxlayırıq:

```
C:\Users\Administrator> netstat -na | findstr 5060
```

TCP	192.168.197.128:5060	0.0.0.0:0	LISTENING
UDP	192.168.197.128:5060	*:*	

Nəticə

Bu başlıqda biz aşağıdakı addımları açıqladıq:

- FreeSWITCH-i internetdən endirdik və yüklədik
- Kompiylasiya zamanı özümüzə uyğun olaraq modullarda dəyişiklik üçün **modules.conf** faylında işə gördük.
- Dəyişiklik etdiyimiz modulu həmçinin FreeSWITCH-in quraşdırma faylı olan **modules.xml** faylında düzəltdiq. **mod_flite** modulunun FreeSWITCH işə düşdükdə avtomatik işə düşməsi üçün şərhi sildik.
- FreeSWITCH fərqli əməliyyat sistemlərində yüklədik, işə saldıq və sinaq üçün bir neçə əmri işə saldıq.
- FreeSWITCH-i daemon(Linux/UNIX) və servis(Windows) kimi işə saldıq.

Növbəti başlıqda artıq yükləmiş olduğumuz FreeSWITCH-i tətbiqdə quraşdırıb istifadə edəcəyik.

3-cü başlıq

Quraşdırma nüsxələrində sınaqların edilməsi

Artıq yüklenmiş və hazır vəziyyətdə olan FreeSWITCH-in quraşdırma nüsxələrini açıqlamaq vaxtıdır. Quraşdırma nüsxələri öncədən istifadəçilər, Dialplan, təhlükəsizlik və daha da çox məqsədlər üçün hazır qurulmuşdur. Quraşdırma nüsxələri sizin FreeSWITCH-də başlanğıc olaraq nə bacarığa sahib olmasına göstərmək üçün nəzərdə tutulmuşdur.

Bu başlıqda biz aşağıdakıları açıqlayacaqıq:

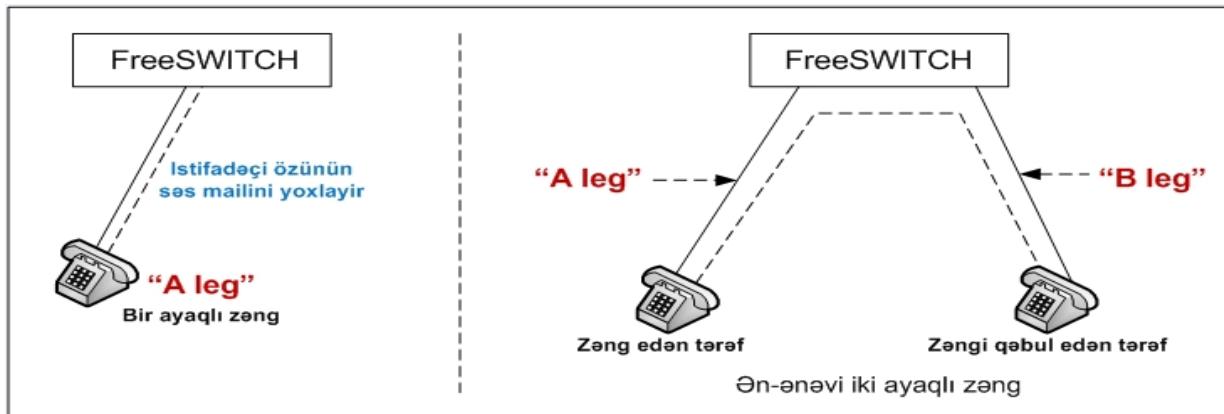
- VoIP və FreeSWITCH-in vacib konsepsiyası
- FreeSWITCH command line interfeysin istifadə edilməsi (**fs_cli**)
- FreeSWITCH-ə qoşulması üçün telefonun quraşdırılması
- CUCM SIP trunk-da istifadə etməyimizdə tələb ediləcək Cisco İP communicator quraşdırılması
- Dialplan nüsxəsinin sınaqdan keçirilməsi

VoIP və FreeSWITCH-in vacib konsepsiyası

FreeSWITCH çox unikal program təminatıdır. Unikallığının səbəblərindən biri də odur ki, telefon sistemləri dünyası həddən artıq dinamikdir. FreeSWITCH program təminatının developerləri qeyd etdiyi kimi, onlar FreeSWITCH-in fərqli situasiyalarda özünü necə aparması üçün seçim etməkdə böyük çətinliklərlə qarşılaşmışlar. FreeSWITCH əksər telefon avadanlıqlarını dəstəkləyir ancaq, elə telefonlar ola bilər ki, açıq şəkildə spesifikasiyanın qaydalarını pozurlar. FreeSWITCH elə nəzərdə tutulub ki, həm eyni zamanda statik olaraq telefonları quraşdırır və eyni zamanda da çoxlu sayda olan telefonu dinamik olaraq hazır quraşdirmalar istifadə edərək quraşdırır bilərsiniz. Yeni başlayan istifadəçi üçün bu çox olsa da narahat olmayın. FreeSWITCH yükləndikdən sonra, sizin yüksəlmış versiyanın içində hazır quraşdirmalar olacaq ki, sadəcə xırda dəyişikliklərdə siz bütün kitab boyunca sınaqlarınızı keçirə biləsiniz.

1-ci başlıqda FreeSWITCH konsepsiyasında danışdığımız kimi, FreeSWITCH mərkəzi core-dan ibarətdir hansı ki, XML reestr-lə sovrulur və orbitində core vasitəsilə bir-biri ilə əlaqələnən çoxlu modullar mövcuddur. Biz XML reyestrində olan quraşdırma nüsxələrini telefonları qeydiyyata salmaq və bəzi sınaq zəngləri eləmək üçün, istifadə etməyə gedirik. Siz zəng edən kimi, SIP modulu müraciəti XML Dialplan-a ötürür harda ki, sizin daxil elədiyiniz rəqəmlər yenidən **regular expressions** (müntəzəm ifadələr) adlanan başlıqlar seriyaları ilə üst-üstə düşür. Uyğunluq tapılan kimi, XML genişləmədən gələn və üst-üstə düşən məlumatlar daxili kanala nüsxələnirlər. Beləliklə instruksiyalar siyahısı var ki, zəngin növbəti addimından bunlar yerinə yetiriləcəkdir. Quraşdirmada olan açar sözlərin seçimində asılı olaraq, mümkündür ki, eyni Dialplan asılılığında birdən çox genişlənmə uyğun olsun. Bu tip kiçik sınaqların aparılmasında siz sadəcə bir ədəd genişlənmədən istifadə edəcəksiniz və sizin şansınız var ki, kanal *ROUTNING* statusunda olanda bütün zəng müddətinin datalarını görə biləsiniz (Kanal statuslarının detalları üçün 8-ci başlıqda *İrəliləmiş Diaplan Konsepsiyalarında* olan *Bütün bunların birləşməsi* seksiyasında baxa bilərsiniz..).

Telefon terminalogiyasında biz iki alət arasında olan qoşulmaya **call leg** (çağırış ayağı) deyirik. "**A Leg**" termini istifadə edilir ki, zəng edən tərəf və FreeSWITCH arasında olan qoşulma ünvanını açıqlasın. "**B leg**" termini isə zəngi qəbul edən tərəf(ayağı) və FreeSWITCH arasında olan qoşulma ünvanını açıqlamaq üçün istifadə edilir. Aşağıdakı quruluşa baxın:

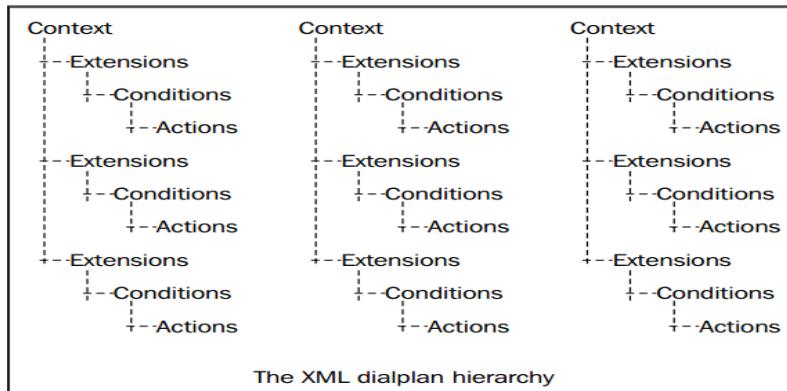


Əgər siz telefonu demo genişlənməyə zəng etmək və qulaq asmaq üçün istifadə edirsizsə, bir leg(ayaq) istifadə edilir və bu sizin telefonla FreeSWITCH

arasında olan əlaqədir. Əgər siz FreeSWITCH-də hal-hazırda qeydiyyatda olan hansısa bir telefonu yığırsınızsa ya da xidmət təcizatçınız tərəfdə olan hansısa bir telefona zəng edirsinizsə sizin iki zəng ayağınız var - zəng edən tərəf "**A Leg**" və zəngi qəbul edən tərəf "**B leg**". Bu FreeSWITCH və digər telefonu ya da xidmət təcizatçısını birləşdirir. Hər zəng ayağının özünəməxsus unikal quruluşu və spesifik tələblə qarşı ayaqla xüsusi əlaqələri var. Bir və ya bir neçə ayaq öz aralarında media paylaşımı etdikdə, buna bridge deyilir. Bridge olunmuş zəngdə, zəng ayağı eyni bridge-in daxilində digər ayağa təyin edilmiş əməliyyatlar yerinə yetirə bilər. Misal üçün, digər ayağı gözləmədə saxlamaq, digər genişlənməyə transfer(yönləndirmə) etmək ya da bu genişləndirmənin 3-cü tərəf ilə əlaqələndirilməsi(3 tərəfli danışıq üçün).

Bəzi zənglərin yalnız bir ayağı var - qoşulma yalnız bir telefonla FreeSWITCH arasında olur və FreeSWITCH birbaşa zəng edən tərəflə əlaqəyə girir. Əksər hallarda bu tip interfeysə IVR ya da Interactive Voice Response menyu deyilir. Bir ayaqlı zənglərin digər misal isə voicemail yada zəng edənin konfrans otağına qoşulması deyilə bilər. IVR çox mükəmməl bir sistemdir və siz sözsüz ki, hansısa bir böyük şirkətə zəng etdikdə sizinlə danişan və müəyyən rəqəmlər vasitəsilə seçim etmənizi təklif edən hansısa bir sistemlə qarşılaşmışınız ki, bu da IVR-dir. Əgər siz vizit kart istifadə etmisinzsə, orda olan PUBLIC nömrə və daxili nömrə IVR-da olan seçimdir. Bəzi IVR-lar hətta uyğun olan vaxtda müəyyən sözləri və səsi təyin edib qərar verə bilər. FreeSWITCH-lə IVR qurmaq çox asandır. Biz bunu 6-ci başlıqdə "XML IVR-ların istifadə edilməsi və Phrase Macros" və "7-ci başlıqdə LUA vasitəsilə Dialplan Scripting"-də keçəcəyik.

XML Dialplan genişlənməli spesifik qruplara ayırır hansı ki, bunlara **contexts** deyilir. Context isə özündə bir neçə genişlənmə elementini tutan XML tagdır. **Extension** isə başlıqlar yığımıdır hansı ki, yığılan rəqəm üçün üst-üstə düşdükdə pozitiv ya da negativ istiqamətdə müəyyən instruksiyaları yerinə yetirir. Aşağıdakı XML Dialplan ierarxiyasına baxa bilərsiniz:



FreeSWITCH core-a daxil olan hər bir zəngin hara yönləndirilməsini təyin etmək üçün öncədən müəyyənləşdirilmiş kontekst yığımı və genişlənmə rəqəmlərinə malik olmalıdır. Biz öz misallarımızda XML Dialplan və default Dialplan konteksti istifadə edəcəyik. Genişlənmə rəqəmləri sizin nə yığdıığınızdan və zəngə nə zaman yerləşdiriyinizdən asılıdır. Siz genişlənməni yığan kimi, SIP Endpoint modulu sizin SIP telefondan decode olaraq alınan bütün zəng datasını əlavə edir, XML-ə Dialplan yükleyir,

context-i **default** edir və zənglərin statusunu ROUTING-ə ötürür. Core-da olan susmaya görə ROUTING logikası XML Dialplan moduluna baxacaq və zəngi alt zəng emaledicisi programına ötürəcək. Bu altprogram XML reyestri ilə əlaqələnir və **default** kontekstinə aid olan kontekst siyahısını axtarır. Kontekst tapılan kimi, o kontekstdə olan hər bir genişlənməni analiz edir və uyğun olanı tapanadək şərt(**condition**) tag-ində olan başlıq nüsxələrini yoxlayır. Eyni şərt tag sərhədlərində olan hər bir iş(**action**) tag-ı proqrama və əlavə data arqumentinə malikdir. Bu programlar bize **application modules**(1-ci başlıqdə "FreeSWITCH-in arxitekturasi"-ında) tərəfindən verilir hansı ki, səliqə ilə sonuncuya çatanadək və ya zəng bitənədək yerinə yetiriləcək.

Programların arqumentləri özlərində **channel variables**(kanal dəyişənləri) saxlaya bilər, ad/məna cütlüyünün spesifik qrupu kanalın özünü aparmasına təsir və həmçinin zəngdə olan vacib məlumatların saxlanması üsulu üçün dizayn edilmişdir. Onlar biraz önce danışdığımız pre-prosesor dəyişənlərinə oxşayırlar ancaq, iki dollar simvolun əvəzinə bir dollar simvolu istifadə edilir. **\${destination_number}** misali bize deyir ki, zəng edən hansı rəqəmlərə yığıb və zəngin yönləndirilməsi üçün bu əsas mənadırmı? Hansı mənanın başlıq uyğunluğunu təyin ələmək üçün, şərt tag-ləri **field** atribut istifadə edir. Əgər bu məna zəng edən tərəfin profilində yadda saxlanılmış vacib dəyişənlərdən biridirsə, sadəlik üçün **\${}** buraxmaq olar.

Spesifik zəng edənin **profile** dəyişənləri aşağıdakı kimidir. Onlardan bəziləri istifadəyə yararsız kimi görünə bilər ancaq, siz FreeSWITCH-i istifadə etdikcə onların harda işə yaradığını görəcəksiniz:

- *username*
- *dialplan*
- *caller_id_name*
- *caller_id_number*
- *callee_id_name*
- *callee_id_number*
- *network_addr*
- *ani*
- *aniii*
- *rdnis*
- *destination_number*
- *source*
- *uuid*
- *context*

Zəng edən profaylı spesifik informasiya kolleksiyasıdır hansı ki, ümumiyyətlə bir ayaqdan digər ayağa edilən hər bir zəngdə olur. Bu informasiyaya digər dəyişənlərdə olduğu kimi, eyni üsulla yetki almaq olar və verilənlərin başlanğıc zəngində məlumatlar təmin edildikdə ancaq **read-only** hesab edilməlidir. Aşağıda default quraşdırmanın real misaldır hansı ki, **1980**-ci illərin məhşur **tetris** oyununu oxutmaq üçün, core-da olan tone generator istifadə edir:

```
<extension name="tone_stream">
<condition field="destination_number" expression="^9198$">
<action application="answer"/>
<action application="playback"
```

```

    data="tone_stream://path=${base_dir}/conf/tetris.ttml;loops=10"/>
</condition>
</extension>
```

Gördüyünüz kimi bu **field="destination_number"** istifadə edir ki, sizin **9198**-ə zəng etməyinizi yoxlaya və əgər siz bunu etmisinizsə zəngə cavab verilir və **tone stream** oxumağa başlayır. Quraşdırma **\${base_dir}** istifadə edilir ki, tonal data-nın tam ünvanını təyin edək(Burda detallı məlumatın olmamasından narahat olmayın çünki, bu haqda detallı məlumatı tam açıqlayacaq). Sizin imkanınız var ki, FreeSWITCH-in susmaya görə olan quraşdırmasında müəyyən genişlənmələr arasında zənglər edərək sınaqlar edəsiniz. Kanal dəyişənləri sizə zəng haqqında kənar informasiyanın integrasiyası zamanı lazımlı ola bilər hansı ki, siz gələcəkdə öz programlarınız vasitəsilə əldə edə və təyin edə bilərsiniz. Bu dəyişənlər özündə zəng edənin hesab rəqəmi kimi məlumatları daşıya bilər hansı ki, əgər kimsə buna zəng etməklə öz hesabını idarə etmək istərsə lazımlı olacaq. Orda dəyişənlərin programa təyin edilməsi üçün interfeys var hansı ki, bu imkan set programı tərəfindən verilmişdir.

Aşağıdakı misalda göstərilir:

```
<action application="set" data="customer_id=1234"/>
```

Bu nöqtədən etibarən, kanal dəyişəni olan **customer_id**-nin mənası **1234** olacaq. Əgər bu məna edilən zəngin sonunadək dəyişdirilməzsə onda, bu məna **Call Detail Record(CDR)** data-da da mövcud olacaq.

XML reyestri həddən artıq böyük və qorxulu olduğuna görə, FreeSWITCH komandası onu quraşdırma qovluğununda məntiqi olaraq parçalayıb bir neçə kiçik fayla çevirdi. Bu o deməkdir ki, bir böyük faylı dayanmadan qazmaq əvəzinə siz daha kiçik və sadə fayllara ünvan təyin edə bilərsiniz hansı ki, sayələrində FreeSWITCH-də fərqli spesifik tipə sahib olan funksionallıqlar işləyəcək.

Qeyd: FreeSWITCH wiki-də onun quraşdırmasının böyük sxemi üçün nüsxələr göstərilir:

http://wiki.freeswitch.org/wiki/Default_config#Overview_Diagram_of_the_Demo_Configuration

FreeSWITCH əsas reyestri(faylin adı **freeswitch.xml**-dir) yükləndikdən sonra, fayl spesifik preprocessor tərəfindən yerinə yetilir ki, faylı spesifik direktivlər üçün analiz edir və faylin tərkibini digər faylların tərkibi ilə əvəz edir. Bəzi hallarda preprocessor hətta global dəyişənləri(**global variables**) təyin edir. Bu deməkdir ki, siz bir dəfə yuxarı səviyyəli quraşdırma faylında vacib dəyişənləri təyin edə və sonra reyestrin daha dərin bölmələrdə onlara istinad edə bilərsiniz. Misal üçün IP ünvan ya da domen adını götürün. Deyək ki, sizin hansısa vacib IP ünvanınız mövcuddur misal olaraq, **74.112.132.98**. Əgər siz bu mənanı öz quraşdirmalarınızda bir neçə dəfə istifadə edirsinizsə, onda aşağıdakı sətiri ilk faylin üst hissələrində təyin edə bilərsiniz ki, başlanğıcda yüklənsin:

```
<x-PRE-PROCESS cmd="set" data="my_ip=74.112.132.98"/>
```

Artıq siz öz quraşdirmalarınızda IP ünvanın görünmək istənilən yerlərində **\$\$ {my_ip}** dəyişənidən istifadə edə bilərsiniz. Bu genişlənmə preprocessor tərəfindən edilir və **\$\$ {my_ip}** dəyişəni olan yerlərin hamisində bu IP ünvanı FreeSWITCH tərəfindən generasiya edilib yüklənən son XML-də elə görünəcək ki, elə bil core kodun içində sərt olaraq yerləşdirilmişdir.

FreeSWITCH-in digər gözəl funksionallıqlarından biridə odur ki, tək sətirdə digər faylları əlavə etmək olur. Aşağıdakı misal Dialplan konteksti üçün təyin edilən **default/** qovluğunda olan bütün *.xml genişlənməli faylları əlavə edir:

```
<!--PRE-PROCESS cmd="include" data="default/*.xml"/>
```

Bu o deməkdir ki, **default** qovluğunda olan və .xml genişlənməsi ilə bitən istənilən fayl öncəki sətirlə təyin edilmişdir. Bu şərait yaradır ki, **default.xml** (fayldə susmaya görə olan Dialplan kontekst olur) fayla korlamadan yeni genişlənmələri onların özlərinə aid olan seçilmiş fayllarda yaradıb öz Dialplan-ımıza əlavə edə bilək.

FreeSWITCH-i işə salaq

Artıq FreeSWITCH-in başlanğıc imkanlarını öyrəndikdən sonra, onu işə salmağın vaxtıdır. Öncə biz FreeSWITCH-in idarə edilməsi üçün aləti (**Command Line Interface** ya da **CLI**) dərindən öyrənəcəyik və sonra bir və ya bir neçə telefonu quraşdıracaq ki, sınaq zəngləri edək.

FreeSWITCH command line interfeysin istifadə edilməsi(fs_cli)

2-ci başlıqda "Kompilyasiyası və Yüklənmə"-də biz kiçik də olsa, **fs_cli** utuiliti haqqında danışdıq. FreeSWITCH ümumiyyətlə Linux/UNIX-de daemon və Windows-da servis kimi işə salındığına görə, biz **fs_cli** aləti ilə artıq tanışıq. Rahatçılıq üçün siz **fs-cli.exe** alətini Windows-da susmaya görə olan qlobal mühit dəyişənlərinə əlavə edə bilərsiniz (Bunu Linux-da "**ln -s**" əmri və FreeBSD-də root istifadəçi profilinin **path** dəyişəninə əlavə etməklə elədik). Əksər Linux/UNIX istifadəçiləri **/usr/local/freeswitch/bin** ünvanını birbaşa qlobal **path** dəyişəninə əlavə etməyi üstün tuturlar. Artıq sadəcə **fs_cli** əmrini sistem CLI-dan daxil etməklə avtomatik olaraq **fs_cli** programını işə salacaq.

Oeyd: Windows машında FreeSWITCH-i command line interfeys əmri **.exe** genişlənməli olduğuna görə, istifadəçilər **fs_cli.exe** əmrindən istifadə etməlidirlər.

FreeBSD машında command-line alətini işə salırıq:
root@frfs:~ # fs_cli

Siz aşağıdakı şəkildə göstərildiyi kimi FS CLI xoş gəlmisiniz mesajını ekranda görəcəksiniz:

Qoşulduqdan sonra **/(slash)** simvolu ilə başlayıb daxil edilən əmrlərdən başqa istənilən əmr FreeSWITCH serverə ötürülcək. Slash simvolu ilə başlayan əmrlər birbaşa **fs_cli** programın idarə edilməsi üçün istifadə edilir. **/help** əmrini daxil edərək **/(slash)** simvolu ilə başlayan bütün əmrlərin siyahısını görə bilərsiniz(ağıkdakı şəkildəki kimi):

```
freeswitch@internal> /help
Command          Description
-----
/help            Help
/exit, /quit, /bye, ...   Exit the program.
/event, /noevents, /nixevent Event commands.
/log, /nolog      Log commands.
/uuid            Filter logs for a single call uuid
/filter          Filter commands.
/logfilter       Filter Log for a single string.
/debug [0-7]      Set debug level.

freeswitch@internal>
```

Nəzərə alın ki, sistemdən çıxış üçün orda çoxlu fərqli slash əmrləri mövcuddur: **/exit**, **/quit** və **/bye**. Həmçinin programdan çıxış üçün qısa keçid **3** nöqtə(...) mövcuddur. Bu dörd əmrlərdən hansısa biri daxil edildikdə, FreeSWITCH sistemdən çıxıb sistem CLI-na qayıdacaq. Bu əmrlərin hamısı bir-birinin ekvivalentidir. Yadda saxlayın ki, əgər FreeSWITCH console-dan işə salınıbsa, yeni daemon və ya servis kimi deyilsə, **3nöqtə**(...) ilə çıkış FreeSWITCH sistemini söndürəcək!

Yadda saxlayacağımız digər slash əmri isə **/log**-dur. Susmaya görə FreeSWITCH tam **debug** jurnallama səviyyəsi ilə işə düşür(Xoş gəlmisiniz ekranı bunu **+OK** **log level [7]** sətiri ilə bildirir). **/log** əmri sizə daxil olduğunuz **fs_cli** sessiyası üçün hansı səviyyə jurnallama olmasının təyin edilməsinə kömək edir. Siz öz sessiyanızın istənilən məqamında istənilən jurnallama səviyyəsi təyin edə bilərsiniz. Siz **fs_cli**-dan çıxıb yenidən daxil olduqda isə **log** səviyyəsi yenidən sıfırlanıb **7**-ci səviyyəyə qayiadacaq(Bu tip idarəetmə **-d** ya da **-debug** command-line parametrlə idarə edilə bilər). Əgər siz çoxlu debug məlumatı görmək istəmirsinizsə, jurnallama səviyyəsi aşağıda göstərilən qaydada **6** təyin etməniz düzgün olar:

```
freeswitch@internal> /log 6
+OK log level 6 [6]
```

0-7 aralığında olan hər bir rəqəm fərqli **debug** səviyyəsini aşağıdakı cədvəldəki kimi özündə təşkil edir:

Debug səviyyəsi:	Ad:	Tekst göstərilən rəng:
0	Console	Ağ
1	Alert	Qırmızı
2	Critical(Crit)	Qırmızı
3	Error(Err)	Qırmızı
4	Warning	Bənövşəyi
5	Notify	Açıq gøy
6	Information(Info)	Yaşıl
7	Debug	Sarı

Siz jurnallama səviyyəsini rəqəmlərə təyin etdiyiniz kimi, həmçinin adlarla(reqistrə hissyatlıdır) da təyin edə bilərsiniz:

```
freeswitch@internal> /log info
+OK log level info [6]
```

Bütün digər daxil edilən əmrlər FreeSWITCH serverə ötürüləcək. Bəzi əsas əmrlər var ki, onları mütləq bilmək lazımdır və aşağıda sadalanır:

- **help:** Mövcud olan CLI əmrlərin siyahısını çap edir. Bu əmrlərə **FSAPI** əmrlər ya da qısa olaraq **API** əmrlər deyilir.
- **version:** İşləyən FreeSWITCH versiyasını çap edir.
- **status:** Hal-hazırda işləyən FreeSWITCH prosesinin müəyyən statistikalarını çap edir.
- **show channels:** Aktiv olan ayrıca kanalların siyahısını çap edir.
- **show calls:** Bridge edilmiş zənglərin siyahısını çap edir.

Kanal(channel) zəngin bir ayağıdır. Bir ayaqlı zəng üçün misal olaraq istifadəçinin voicemail yoxlanışı ola bilər. Digər tərəfdən isə zəng, birlikdə olan bridge(qoşulu olan) edilmiş iki individual zəng ayağıdır. Əmin olun ki, **show channels** və **show calls** arasında olan fərqi tam başa düşürsünüz.

Növbəti bölməmizdə, FreeSWITCH-ə qoşulmada telefonların quraşdırılmasında bizə kömək olacaq daha geniş əmrləri öyrənəcəyik.

FreeSWITCH-ə qoşulması üçün telefonun quraşdırılması

FreeSWITCH-ə qoşacağımız əksər telefonlar SIP bazalıdır. SIP ya da Session Initiation Protokol telefonların zəngində çox yayılmış siqnal səviyyəsinin protokoludur (SIP təkcə səslə məhdud deyil. O həmçinin chat video və digər sessiya tiplərini emal edə bilir). SIP telefonlar iki versiyada buraxılır: avadanlıq və program təminatı telefonları. **Fiziki telefon** adətən klaviatura, səs qarniturası və adətən ekranla olur. Soft telefon isə program təminatıdır və kompüterin mikrofonu (ya da əlavə qarnitura) ilə dinamiklərini istifadə edir. Biz quraşdirmalarımızı açıq kodlu olan X-Lite soft telefon üzərində edirik ancaq, SIP bazalı fiziki Aastra, Polycom yada Snom telefonlarından istifadə edə bilərsiniz.

SIP quraşdirmaları

Bütün SIP alətlərinin minimal quraşdırma tələbləri var ki, onlar təyin edilməlidir. Bütün çətin protokollar kimi, SIP-in də öz qaranlıq və gizli quraşdırma üsulları mövcuddur. Buna baxmayaraq SIP haqqında ətraflı məlumat bu kitab çərçivəsini aşır. Ona görə də biz öz diskussiyamızı SIP alətin FreeSWITCH-ə qoşulub işləməsinə və standart PBX funksiyalarının edilməsinədək limitləyəcəyik (Zəng etmək, zəngi qəbul etmək, zəngin yönləndirilməsi, zəngin gözləməyə qoyulması və.s.).

Biz öz SIP quraşdirmalarımızda SIP alətə sahib olacaq ki, FreeSWITCH serverimizlə əlaqələndirək. SIP aləti SIP qeydiyyatçıda qeydiyyatdan keçdikdən sonra, qeydiyyatçı (**SIP registrar**) artıq zənglərin SIP alətinə necə yönləndirilməsi məlumatına sahib olur. Bizim üçün FreeSWITCH qeydiyyatçıdır (**SIP registrar**). SIP **digest authentifikasiya** ilə qeydiyyatdan keçmək istəyən son SIP nöqtələrininə şərait yaradır. Həmçinin mümkündür ki, qeydiyyatdan keçməyən SIP istifadəçilərini qeydiyyata salaq amma, bu məsləhət deyil (Bunun düzgün analogiyası isə, açıq SMTP relay serverdir. Əgər siz kiməsə öz sisteminizdən istifadə edərək email yollamağa izin verərsinizsə, pis şeylər baş verəcək. Xahiş olunur bunu etməyəsiniz!). SIP istifadəçiləri elektron məktuba oxşar strukturda qoşulurlar. SIP URI-də həmçinin emaildəki kimi, **user@domain** strukturu olur. Həmçinin istifadəçi adına əlavə olaraq real ad ya da ekran adı və domen mövcuddur. Qeydiyyatdan keçmək üçün istifadəçi adı və şifrə də mövcuddur. Avtorizasiya adı ilə istifadəçi adı eyni olmalı deyil amma əksər hallarda bu belə olur.

FreeSWITCH susmaya görə öncədən quraşdırılmış **20** SIP istifadəçi ilə gəlir (4-cü başlıqda, *SIP və İstifadəçi qovluğunda* biz bu haqda daha ətraflı danışacayıq və hətta əlavə istifadəçilərin necə əlavə edilməsini də belə). Susmaya görə olan istifadəçilər **1000** və **1019** aralığındadır. Siz öz sinaqlarınızda bu aralıqda olan istifadəçi adlarından istədiyinizi istifadə edə bilərsiniz.

Aşağıdakı quraşdirmalar **1000** nömrəsi üçün göstərilir:

- **Username:** 1000
- **Authorization Username:** 1000
- **Password:** 1234
- **Domain:** [FreeSWITCH serverin IP ünvani ya da DNS adı]

Qeyd: FreeSWITCH susmaya görə yükləndikdə, **1000-1019** genişlənmələri üçün

`/usr/local/freeswitch/conf/vars.xml` faylında `<X-PRE-PROCESS cmd="set" data="default_password=1234"/>` sətiri ilə **1234** şifrəsi təyin edir. Siz bu şifrəni dəyişmək istəsəniz sadəcə **1234** rəqəmlərinin əvəzinə istdəyiniz şifrəni yaza bilərsiniz.

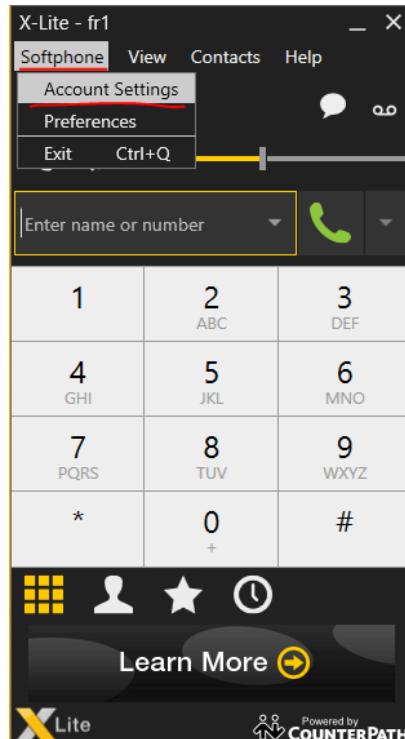
Bu quraşdirmaları özünüzə rahat yerdə saxlayın ki, lazımlı olan halda tez əldə edəsiniz. Gəlin quraşdirmamıza fərqli SIP telefonları üçün baxaq. Sizin istifadə etdiyiniz brand burda göstərilməsə də belə, heç bir problem görmədən eyni qoşulma məntiqini istifadə edərək siz serverə qoşula bilərsiniz. Aşağıda göstərilən hər bir misal üçün, biz öz daxili şəbəkəmizdə işləyən FreeSWITCH serverimizə fərqli telefon qoşacayıq.

X-Lite soft telefon

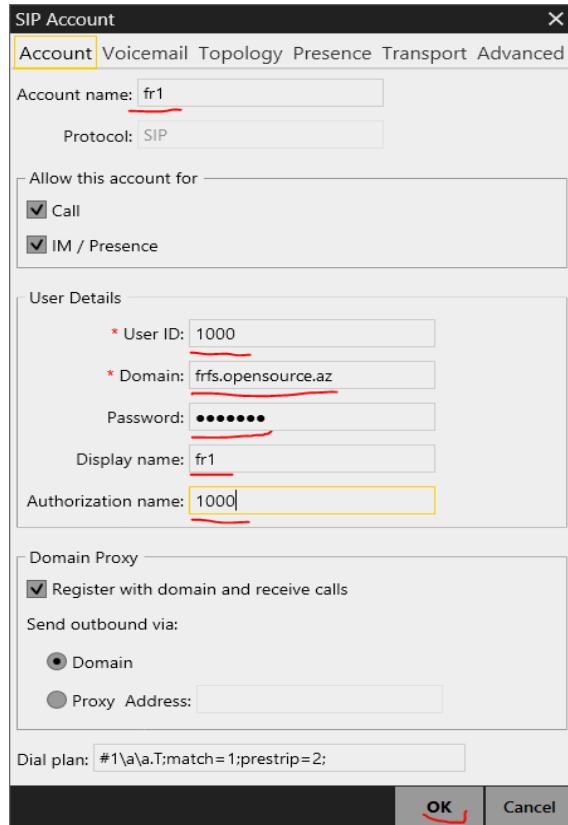
X-Lite (http://counterpath.s3.amazonaws.com/downloads/X-Lite_Win32_4.8.4_76589.exe) pulsuz program telefonudur (Ancaq açıq kodlu deyil). X-Lite programını endirin və FreeSWITCH serverinizin daxili şəbəkəsində olan eyni Windows maşına yükləyin.

Qeyd: Texniki cəhətdən soft telefonu FreeSWITCH serverlə eyni maşında işə salmaq olar amma, bu dəstəklənmir və məsləhət deyil.

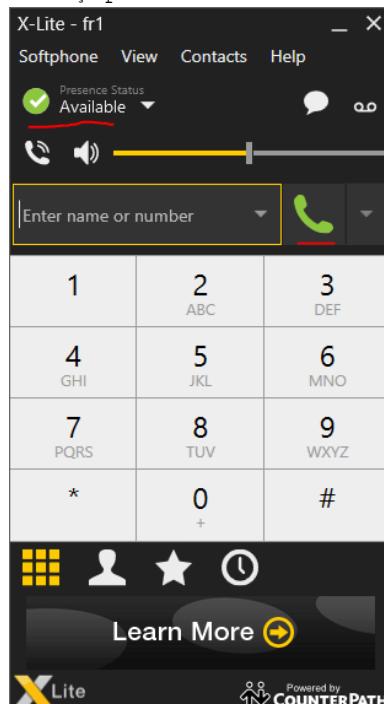
X-Lite programını işə salın. Siz telefonu aşağıdakı şəkildəki kimi görəcəksiniz:



Softphone -> Account Settings düymələrinə sıxın ki, SIP hesabı menysunu açasınız. X-Lite yalnız SIP hesabı dəstəkləyir. Aşağıdakı şəkildə öz hesab verilənlərinizi daxil edin:



Sonda **OK** düyməsini sıxın ki, qeydiyyatdan keçəsiniz. Uğurlu qeydiyyat olan halda aşağıdakı şəkil çap ediləcək:



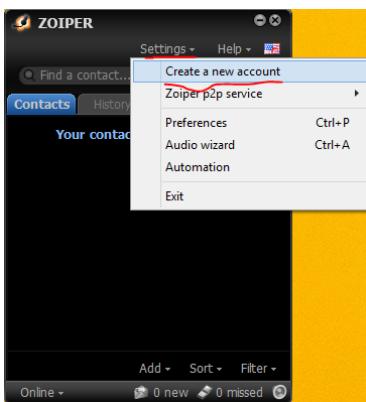
Telefon artıq qeydiyyatdan keçmişdir. Əgər siz ekranınızda hansısa səhv görürsünüzsə, çox guman quraşdırmanızda səhv var. Əksər səhvlər 403, 404 və 408-dir:

- **403 – Forbidden:** Bu o deməkdir ki, qeydiyyat istifadəçi adı və ya şifrə yalnızdır.
- **404 – Not Found:** Bu o deməkdir ki, təyin edilən istifadəçi adı FreeSWITCH server tərəfindən tapılmayıb.
- **408 – Timeout:** Adətən bu o anlama gelir ki, domain adı düzgün deyil ya da şəbəkədə problem var. Əmin olun ki, firewall-nız **5060**-cı portu buraxır.

Artıq sizin telefon qeydiyyatdan keçidkən sonra sınaq zəngləri edə bilərsiniz. **Dialplan nüsxəsinin sınaqdan keçirilməsi** sekiyasınadək ötürə bilərsiniz.

Soft telefon fərqli platformalarda

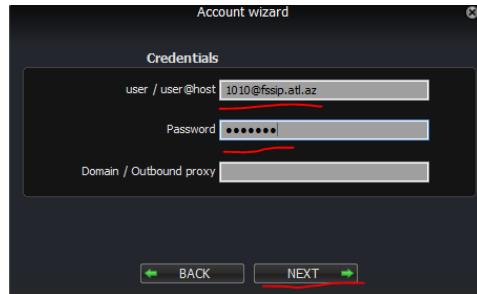
X-Lite telefon Windows, MAC və iPhone, Android(Pulludur **8\$ Bria** version). Ancaq siz Zoiper program təminatından istənilən platformada istifadə edə bilərsiniz. Zoiper rəsmi saytı <http://www.zoiper.com/> -dan daha ətraflı məlumat ala bilərsiniz. Kitab yazılın müddətdə **Zoiper_3.9** son versiya idi və onun **Windows8.1** x64 üzərində quraşdırılmasını edək. Programı açıldıqdan sonra, **Settings -> Create a new account** düyməsinə sixırıq və aşağıdakı şəkil açılacaq:



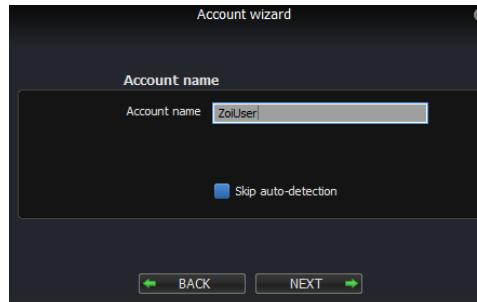
Açılan pəncərədə **SIP** seçirik(Şəkildəki kimi) və **NEXT** düyməsinə sixırıq:



Account wizard olan hissəni aşağıdakı şəkildəki kimi doldurub, **NEXT** düyməsinə sixırıq:



Istədiyimiz hesab adını daxil edib **NEXT** düyməsinə sixırıq:



Sonda **Close** düyməsini sixırıq və sinaq üçün **4000** rəqəminə zəng edirik(Nəticə şəkildəki kimi olarsa uğurlu sayılır):



Qeyd: Həmçinin bilməlisiniz ki, mobildən başqa bütün Linux/UNIX və Windows platformalarda işləyə bilən başqa bir açıq kodlu SIP client program təminatı mövcuddur və bu Jitsi-dir. Siz onun haqqında daha ətraflı rəsmi saytından <https://jitsi.org/Main/Download> əldə edə bilərsiniz.

Fiziki telefonlar

Biz qısa şəkildə fərqli fiziki telefonların quraşdırılmasına baxacayıq. *Aastra*, *Polycom* və *Snom* telefonlarının quraşdırılma nüsxələrinə baxıqdan sonra, siz istənilən SIP telefonun başlanğıc quraşdırılma prinsiplərini biləcəksiniz.

Başlamazdan önce əmin olun ki, telefonunuz üçün ən azı aşağıdakı başlanğıc məlumatlar mövcuddur:

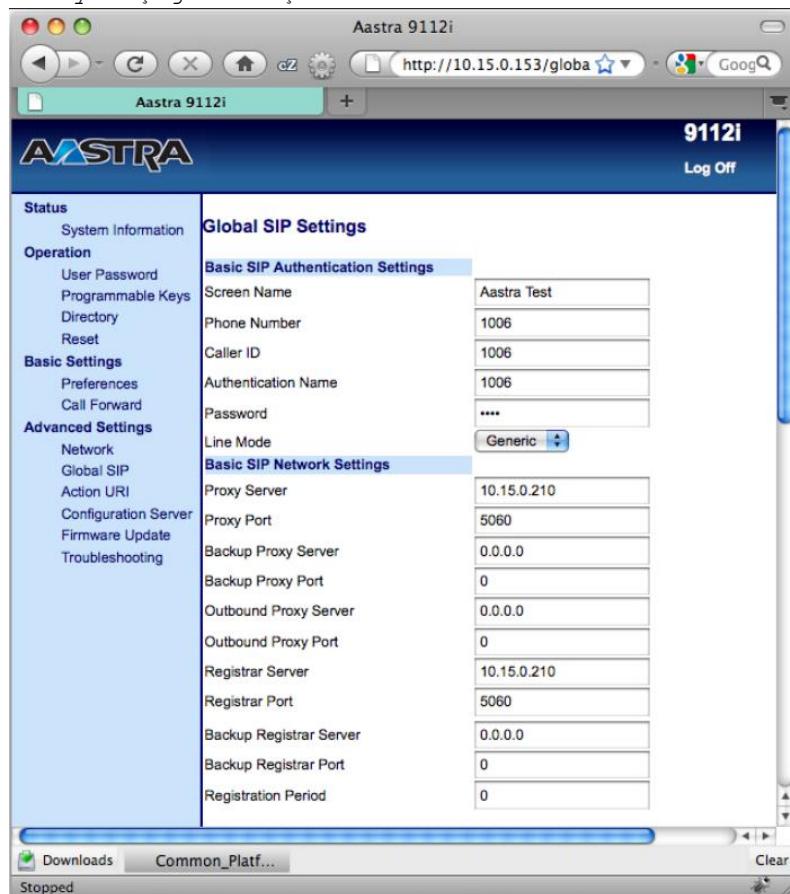
- IP ünvanı:** Siz telefonunuzun WEB interfeysinden istifadə etmək istəyirsinizsə, onun susmaya görə olan IP ünvanını bilməlisiniz. Əksər telefonların quraşdırma üçün kiçik menyusu olur.
- Admin istifadəçi adı:** Əksər telefonların iznibatçı istifadəçi adı olur.
- Admin istifadəçi şifrəsi:** Əksər telefonlar həmçinin inzibatçı şifrəsi tələb edir.

Əgər siz telefonunuzun susmaya görə olan istifadəçi adı və şifrəsini bilmirsinizsə, brendin rəsmi saytından cavabı tapmağa çalışmalısınız.

Qeyd: Əksər telefonlar yenilənilə biləndir və həmişə yeni program təminatlarını rəsmi saytlarında yerləşdirirlər. Mütləq istifadəyə başlamazdan önce rəsmi saytda olan son program təminatını(**firmware**) endirib avadanlığınıza yeniləyin çünki, bu önce ola səhvləri düzəldir və ən yeni yenilikləri əlavə edir.

Aastra telefonları

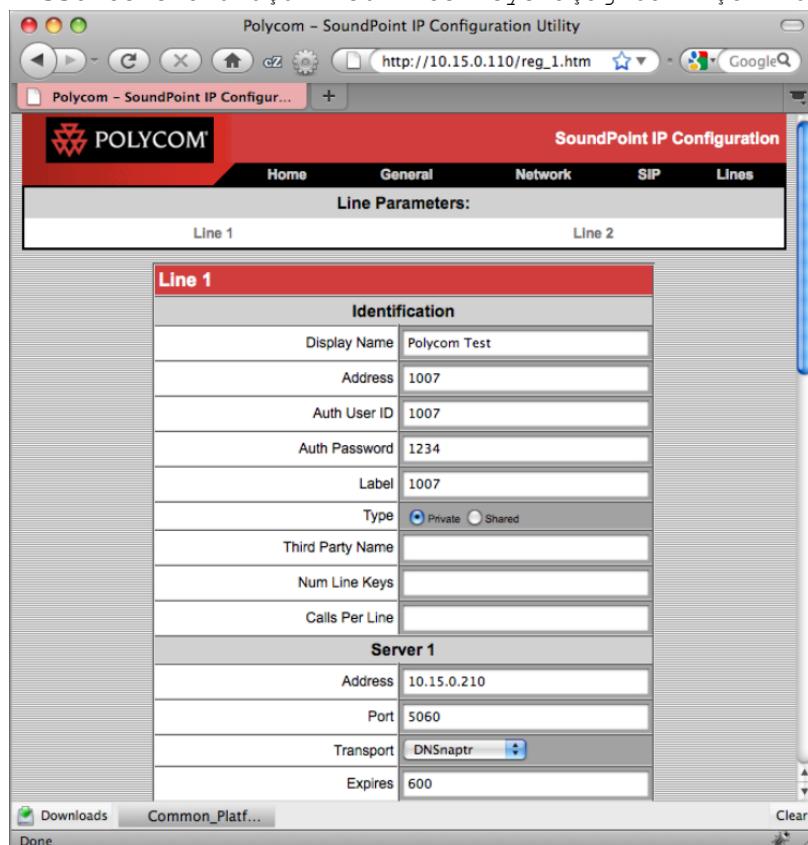
Aastra telefonların standart web interfeysi olur. Avadanlığa qoşulmaq üçün öz web browserinizdə onun IP ünvanını daxil edib və lazımi istifadəçi adı ilə şifrəni yazaraq qoşulun. Panel-də **Global SIP** linkinə sixin. Aastra 9112i üçün web interfeys aşağıdakı şəkildəki kimidir:



Basic SIP Authentication Settings altında olan bütün başlangıç sütunları doldurun. **Basic SIP Network Settings** bölümündə isə, **Proxy Server**, **Proxy Port**, **Registrar Server** və **Registrar Port** sütunlarını doldurun. Sonra aşağı düşüb **Save Settings** düyməsinə sıxın və ardınca idarə edici paneldə **Reset** düyməsinə sıxın ki, telefonunuz yenidən yüklənmə etsin. Telefon yenidən yükləndikdən sonra FreeSWITCH-ə qoşulacaq. Əgər qoşulmanız uğurlu olarsa, ekran adı (**Astra Test**) və telefon nömrəsi (**1006**) ekranda görünəcək. Artıq telefon qeydiyyatdan keçindən sonra, siz sınaq zənglərinizi edə bilərsiniz. **Dialplan nüsxəsinin sınaqdan keçirilməsi** seksiyasında ötürə bilərsiniz.

Polycom telefonları

Polycom telefonların web interfeysi və telefonda menyusu mövcuddur. Siz hansısa birindən istifadə edə bilərsiniz ancaq daha asanı web interfeysdir. WEB browserdə telefonunuzun IP ünvanını daxil edin və daxil olun. **Lines** linkinə sıxın. Əksər SIP telefonlarda olduğu kimi, **Polycom SoundPoint IP 330** telefonunda birdən çox SIP serverdə qeydiyyatdan keçə bilmə imkanı var. Bu misalda biz FreeSWITCH-ə qoşulmaq üçün **Line1** istifadə edəcəyik. **Polycom SoundPoint IP 330** telefonu üçün web interfeys aşağıdakı şəkildəki kimidir:

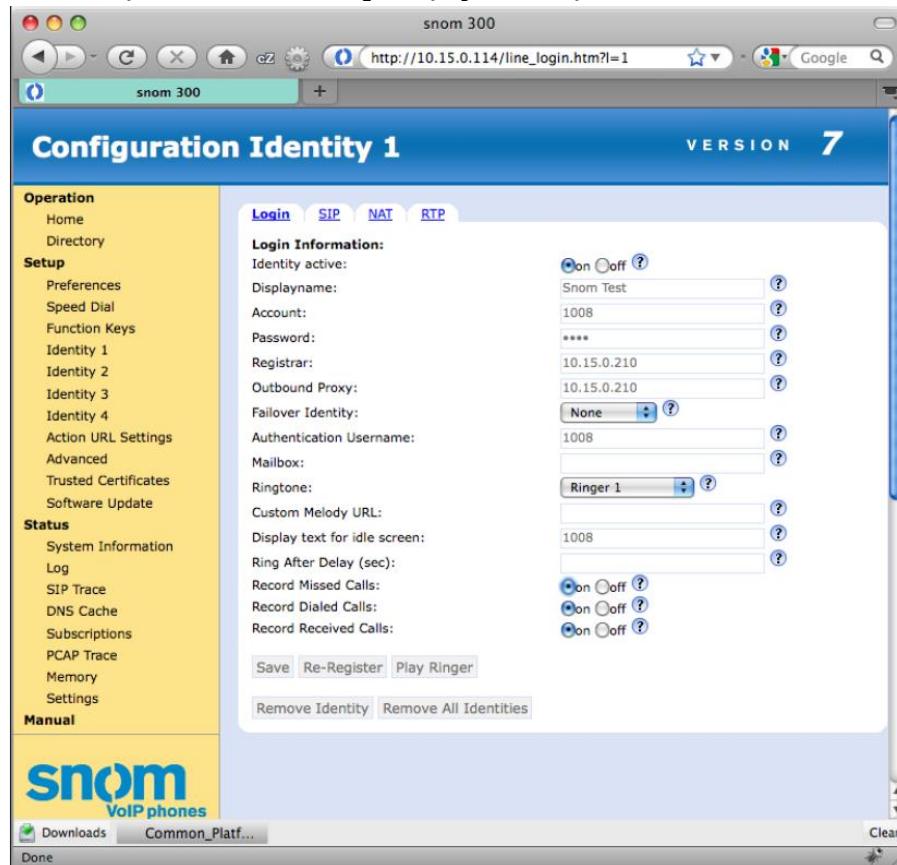


Line1 üçün göstərilən sütunları doldurun: **Display Name**, **Address**, **Auth User ID**, **Auth Password** və **Label**. **Server1**-in altında **Address** və **Port** sütunlarını doldurun. Aşağı düşün və **Submit** düyməsinə sıxın. Telefon **reboot** edəcək və sonra, FreeSWITCH-ə qoşulacaq.

Artıq telefonunuz qeydiyyatdan keçdikdən sonra siz öz sınaq zənglərinizi edə bilərsiniz. **Dialplan nüsxəsinin sınaqdan keçirilməsi** seksiyasınadək ötürə bilərsiniz.

Snom telefonları

Snom telefonların quraşdırma üçün tam imkana sahib olan web interfeysi vardır. Telefonunuzda olan IP ünvanı web browserdə daxil edin ki, quraşdırıcınız. Nəzərə alın ki, Snom telefonların **identities** konsepsiyası mövcuddur hansı ki, sayəsində birdən çox serverə qoşulmağa imkan yaradır. İdarəetmə panelində **Identity1** linkinə sixin və SIP quraşdırma sütunlarını doldurun. **Snom300** üçün web interfeys aşağıdakı şəkildəki kimidir:



Identity1-i üçün **Login Information** altında göstərilən sütunları quraşdırın: **Displayname**, **Account**, **Password**, **Registrar**, **Outbound Proxy**, **Authentication Username** və **Display test for idle screen**. **Save** düyməsinə sixin və sonra, **Re-Register** sixin. Telefon gözləmədən FreeSWITCH serverinize qoşulacaq.

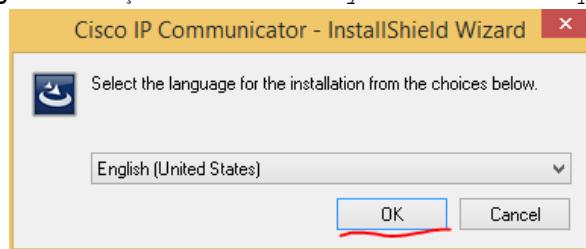
Artıq telefonunuz qoşulduqdan sonra bütün sınaqların keçirilməsi üçün tam hazır vəziyyətdədir. Sınaqlarınız üçün **Dialplan nüsxəsinin sınaqdan keçirilməsi** seksiyasına keçə bilərsiniz.

Qeyd: FreeSWITCH daxili səs kartı və kənar qulaqlıqlı telefonların özünə qoşulmasını dəstəkləyir. PortAudio modulu(**mod_portaudio**) önce 2-ci başlıqdə Kompilyasiya və yüklənmədə **mod_flite**-da göstərildiyi kimi, quraşdırılıla və işə salına bilər. FreeSWITCH üzərində PortAudio modulunun necə kompilyasiya edilməsi və quraşdırılmasını görmək istəyirsinizsə,

http://wiki.freeswitch.org/wiki/Mod_portaudio linkinə müraciət edə bilərsiniz.

CUCM SIP trunk-da istifadə etməyimizdə tələb ediləcək Cisco İP communicator quraşdırılması

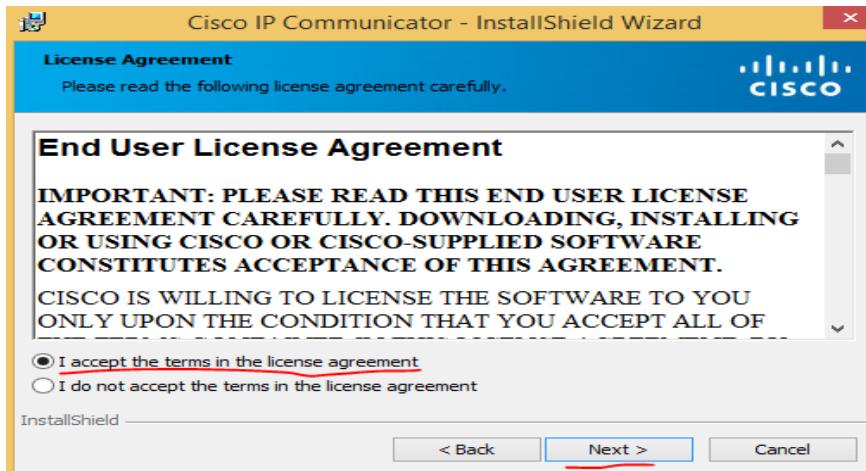
CiscoIPCommunicatorSetup.exe programını **Run as Administrator** adından işə salırıq. **English** seçirik və **OK** düyməsinə sixırıq:



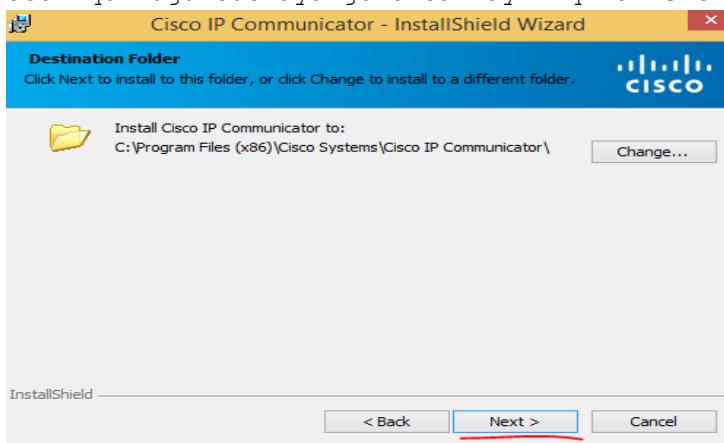
Next düyməsinə sixırıq:



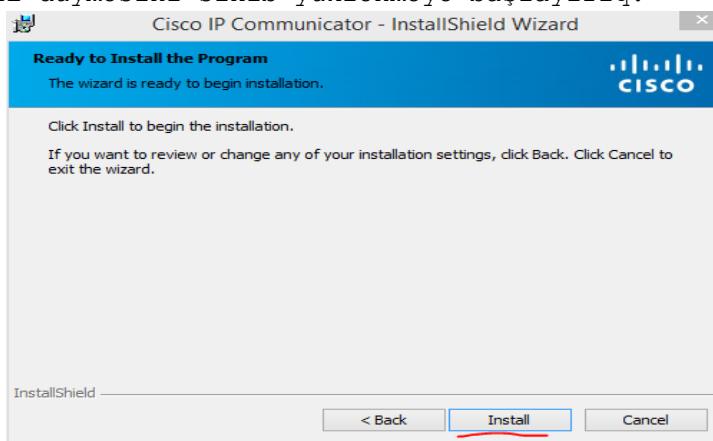
Lisenziya ilə razılışib **Next** düyməsinə sixırıq:



Yüklənəcək qovluğunu susmaya görə saxlayırıq və **Next** düyməsinə sıxırıq:



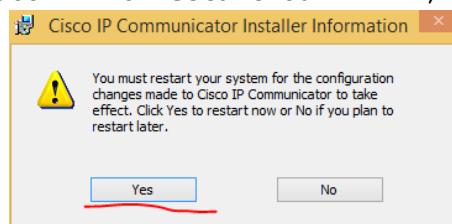
Install düyməsini sıxbı yüklənməyə başlayırıq:



Finish düyməsinə sıxırıq:



Sonda kompyuterimize **restart** edirik ki, programımız sistem reestrinə ötürsün:



IP Communicator quraşdırılması

Məqsədimiz Cisco IP communicator programının quraşdırılmasıdır. Nəzərə alın ki, Cisco **Skinny** protocol istifadə edir və onun quraşdırma fayllarının yüklənməsi üçün **TFTP** server tələb edilir. Bu TFTP server artıq CUCM serverimizdə servislərdə işə salınmış və quraşdırılmışdır.

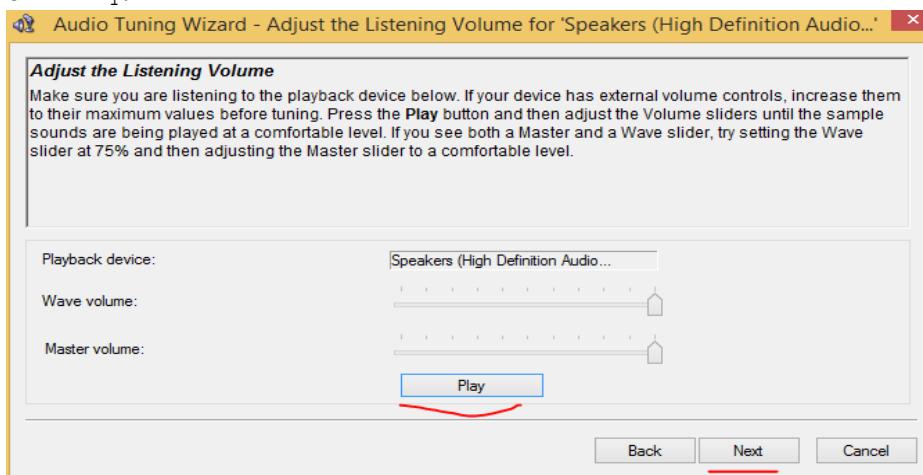
İlk pəncərəmizdə aşağıdakı şəkil çap ediləcək (**Next** düyməsinə sıxırıq):



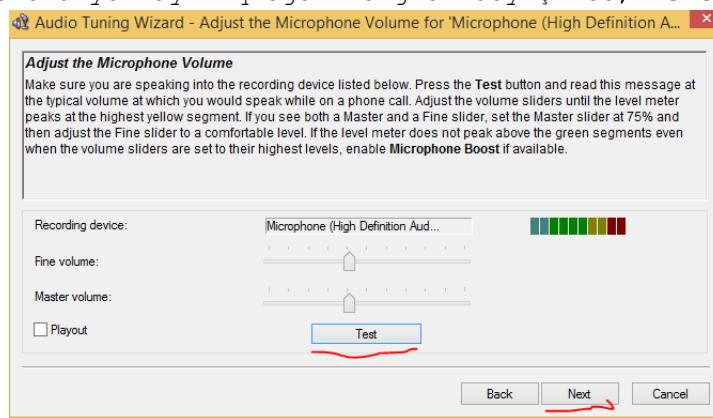
Bütün driverləri susmaya görə saxlayırıq və **Next** düyməsinə sıxırıq:



Play düyməsinə sixaraq səsi yoxlayırıq və səs gəlirsə, **Next** düyməsinə sixırıq:



Sonra mikrofon driverlərinin sinaqdan keçirilməsi üçün **Test** düyməsini sixib, mikrofonu yoxlayırıq əgər rənglər dəyişirse, **Next** düyməsini sixırıq:



Sonda **Finish** düyməsini sixırıq:



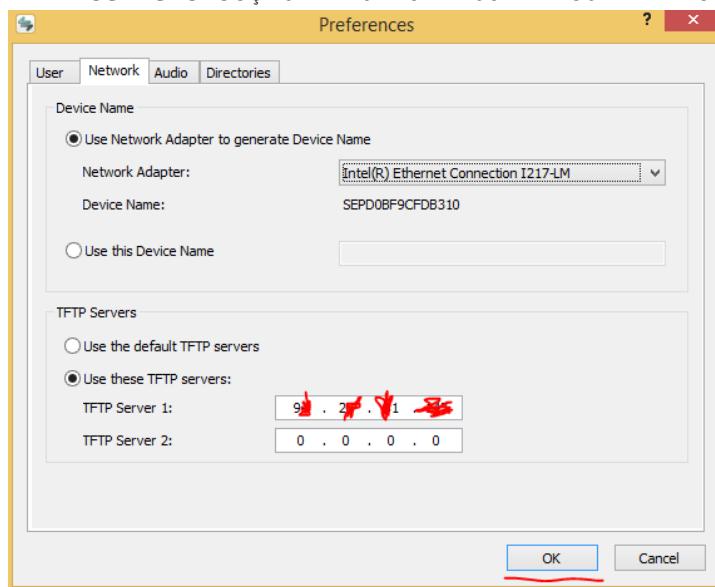
Sonra **Cisco IP Communicator** qısa keçidini **Run As Administrator** ilə açırıq və açılan telefon pəncərəsində qırmızı rənglə olan düyməni sixırıq:



Açılan pəncərədə **Preferences**-ə daxil oluruq:



Görünən pəncərədə **Network** tab-a daxil olurug və **TFTP servers** bölümündə **Use these TFTP servers** seçib IP ünvanı daxil edirik və **OK** düyməsinə sıxırıq:



Uğurlu nəticə aşağıdakı şəkildəki kimi olacaq:



Dialplan nüsxəsinin sınaqdan keçirilməsi

Artıq telefonunuz quraşdırıldıqdan sonra, fərqli sınaqlarınıza başlaya bilərsiniz. Əgər siz iki ədəd telefonu hazır vəziyyətdə qurmusunuzsa, fərqli tip zəgləri edib sınaqdan keçirə biləcəksiniz. Sınaqlarınıza başlamazdan əvvəl əmin olun ki, susmaya görə olan səs və musiqi fayllarını artıq yükləmisiiniz (Windows istifadəçilərində bu susmaya görə olur. UNIX/Linux əməliyyat sistemlərində isə 2-ci başlıqda "Kompilyasiya və yüklənmə"-də 5-ci addımı - musiqi və səs fayllarının yüklənməsində etməlisiniz).

Tek telefon üçün sınaqlar

Aşağıda göstərilən sınaqlar bir istiqamətlidir və FreeSWITCH serverin düzgün işləməsini təyin etmək üçün istifadə edilir. Sadəcə 4 rəqəmli nömrəni sıxıb, **Send** düyməsini sıxaraq zəng etməlisiniz.

Tetris genişlənməsi

9198 nömrəsinə yığın. Siz Tetris temasına oxşar bir temada səs eşidəcəksiniz (Səs tamamilə ton generator sayəsində generasiya edilir. **TMGL** haqqında daha ətraflı məlumat əldə etmək istəsəniz <https://freeswitch.org/confluence/display/FREESWITCH/TGML> linkinə baxa bilərsiniz. FreeSWITCH tərəfindən **Tone generation markup language** istifadə edilir).

Echo test

9196 nömrəsinə yığın. Telefonda danışın və danışdıcılarınız əks səda ilə size qayıdacaq. Bu sınaq göstərir ki, səs hər iki istiqamətdə normal gedib qayıdır (Yadda saxlayın ki, FreeSWITCH və SIP client eyni şəbəkədə olduqları halda əks səda həddən artıq tez qayıdacaq. Bunun üçün **5 saniyə gecikməli** olan **9195** nömrəsinə zəng edə bilərsiniz).

Gözləmədə musiqi (Music on Hold - MOH)

9664 nömrəsinə yığın. Sistem gözləmədə olan musiqini işə salacaq. Əgər siz musiqini eşidirsizsə, demək *music on hold* faylları normal yüklənmişdir və FreeSWITCH səs fayllarını normal işə sala bilir.

Göstərici IVR

5000 nömrəsinə yığın. Göstərmək üçün IVR işə düşəcək. Aşağıdakı göstərilən opsiyalar sizə seçim üçün təklif ediləcək:

- FreeSWITCH public konfransa zəng etmək (**1** rəqəmini sıxmaqla)
- Echo test (**2** rəqəmini sıxmaqla)
- Music on hold (**3** rəqəmini sıxmaqla)
- ClueCon qeydiyyatından keçmək (**4** rəqəmini sıxmaqla)
- Qışqıran meymunlar (**5** – rəqəmini sıxmaqla)
- IVR alt menyu nüsxəsi – (**6** rəqəmini sıxmaqla)

FreeSWITCH public konfrans sadəcə sözdə public konfransdır və istənilən şəxs bu nömrəyə zəng edə bilər. Nəzərə alın ki, sizin FreeSWITCH serverin internetə çıxışı olmalıdır və Firewall-a NAT düzgün quraşdırılmalıdır ki, SIP-lə RTP trafik kənara çıxıb qayida bilsin.

echo test və **music on hold** opsiyaları uyğun olaraq **9196** və **9664** nömrələrində işləyirlər.

ClueCon telefon sisteminin istehsalçılar üçün hər il olan konfransıdır. 4 rəqəmini yığmaqla siz operatora yönləndirilirsiniz hansı ki, sizi qeydiyyatdan keçdiyinizə görə xoş qarşılıyacaq.

Nüsxə **sub-menu** çox sadədir və başa qayıtmək üçün sadəcə * simvolunu yığmaq lazımdır. Nümayiş IVR menyusu **/usr/local/freeswitch/conf/ivr_menus/demo_ivr.xml** faylında tapıla bilər.

İnformasiya programı

9192 nömrəsinə zəng edin. Bu genişlənmə adı misaldır. Zəng etməzdən önce əmin olun ki, **fs_cli** alətini açmışınız artıq. **info** programı bu zəng haqqında çoxlu debug datanı konsolunuza çap edəcək. Zəng etməzdən önce **debug level(INFO)**-i **6**-ya endirin. Siz çıxışın nüsxəsini **3-cü başlıqda** tapa bilərsiniz.

Bu an üçün bütün bunların nə olmasından narahatçılıq keçirməyin. Sadəcə bilin ki, FreeSWITCH aktiv olan hər bir ayaq üçün özündə həddən artıq çoxlu məlumat saxlayır. **info** Dialplan programı özünüzə aid olan Dialplan verilənlərində olan problemlərin araşdırılmasında çox güclü imkandır.

Iki və daha çox zənglər üçün sınaq zəngləri

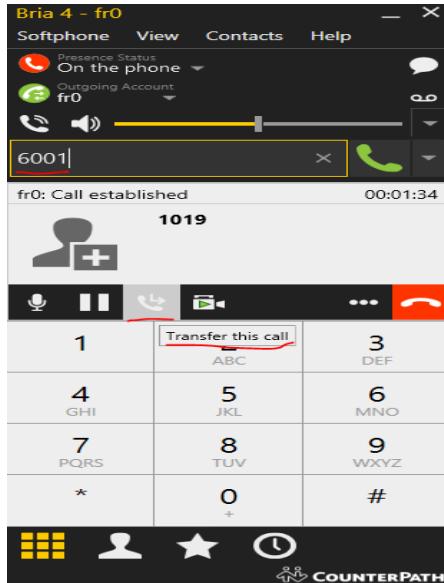
FreeSWITCH-in əsl gücü odur ki, çoxlu son nöqtələrdən gələn zəngləri emal edə bilir. Növbəti sınaqlar sizə FreeSWITCH-də olan bəzi üstünlükleri açıqlayacaq. Bu sınaqların keçirilməsi üçün sizə quraşdırılmış ən azı iki ədəd telefon lazımdır.

Digər telefona zəng edək

1000, 1001 və ardıcılıqla zəng edin. Sadəcə digər telefona zəng edin və o çağırmalıdır. Əksər SIP telefonlar adı analog telefonları kimidir sadəcə telefonu qaldırıb cavab vermək lazımdır. Əgər telefon zəngin cavab verilməsi üçün quraşdırılmayıbsa, siz voicemail-ə yönləndiriləcəksiniz ki, həmin genişlənmə üçün səs mesajını saxlaya biləsiniz.

Zəngin park edilməsi

Digər telefona zeng edin və zəng edilən tərəfdə zəngə cavab verib gözləyin. Zəng edən tərəfdə Transfer(Xlite üçün Transfer This call) düyməsi sıxın və **6001** nömrəsinə yığın. Digər tərəf artıq park edilmişdir və musiqini kanalda eşidəcək. Aşağıda şəkildə göstərilir(X-Lite transfer call dəstəkləmədiyinə görə lisenziyalı Bria-dan istifadə etdim. Təqribən 90\$-adək dəyəri var idi):



Zəngi aşağıdakı usullardan biri ilə əldə edə bilərsiniz:

1. **6001** nömrəsinə yığın. Park edilən zəng avtomatik olaraq parkdan çıxarılaçacaq.
2. **6000** nömrəsinə yığın və sistemin cavab verməsini gözləyin. Sistem sizdən daxili nömrənin yığılmasını istəyəcək.

6001 nömrəsinə zəng edin. Avtomatik olaraq parkdan çıxarılaçacaq.

Konfrans zəng etmə

Bir neçə fərqli telefonlardan **3000** nömrəsinə zəng edin. Hər bir tərəfə eyni konfrans otağına qoşularaq digərləri ilə birlikdə danışa biləcək.

Dialplan nüsxəsinin qısa açıqlanması

Aşağıdakı cədvəldən genişlənmə rəqəmlərini və onların funksiyalarını əldə edə bilərsiniz:

Genişlənmə	Funksiyası
1000-1019	Daxili genişlənmələr
**+genişlənmə rəqəmi	Zəng edən telefonu tutur(zəngin tutulması)
2000	Zəng qrupu nüsxələri: Satış
2001	Zəng qrupu nüsxələri: Texniki Dəstək
2002	Zəng qrupu nüsxələri: Billing
3000-3399	Konfrans otaqları nüsxələri
4000 ya da *98	voicemail-ə zəng edir
5000	Demo IVR
5900	FIFO növbələmə parkı
5901	FIFO növbələnmə əldə edilir
6000	Xidmət parketmə/park-götürmə, əllə
6001-6099	Xidmət parketmə/park-götürmə, avtomatik
7243	RTP multicast səhifəsi
0911	Daxili əlaqə üçün qrup nüsxəsi #1
0912	Daxili əlaqə üçün qrup nüsxəsi #2

0913	Böhran vəziyyətində çıxış konfrans nüsxəsi
9178	Fax qəbulu nüsxəsi
9179	Fax göndəriş nüsxəsi
9180	Çağırış sınağı, çağırış tonu səslənəcək
9181	Çağırış sınağı, U.K. çağırış tonu
9182	Çağırış sınağı, çağırış tonu musiqi olacaq
9183	Cavablayır və sonra U.K səs tonuna yollayır
9184	Cavablayır və sonra musiqiyə yollayır
9191	ClueCon qeydiyyatı
9192	İnformasiya dump-i
9195	Gecikdirilmiş echo test
9196	Echo test
9197	Milliwatt tone (signal keyfiyyəti sınağı)
9198	Tetris musiqisi
9664	Music on hold(Gözləmədə musiqi)

Dialplan yığımının əsas hissəsi

/usr/local/freeswitch/conf/dialplan/default.xml faylında yerləşir.

Notice

Bu başlıqda biz FreeSWITCH-in susmaya görə olan quraşdırmalarını açıqladıq. Başlıqlarla desək aşağıdakı ardıcılılıqda işləri gördük:

- İstənilən zəng edildikdə FreeSWITCH-in özünü hansı qaydada və necə aparmasının vacib konsepsiyalarını açıqladıq.
- FreeSWITCH-in əmrlər sətiri utiliti **fs_cli** istifadəçisinin başlanğıcını açıqladıq.
- FreeSWITCH-ə öncədən təyin edilmiş hesablarla qoşulmaq üçün, SIP alətlərinin necə istifadə edilməsini açıqladıq.
- Fərqli genişlənmə rəqəmlərinə zəng edərək susmaya görə olan Dialplan-ın sınaqdan keçirilməsi.

Artıq biz diqqətimizi FreeSWITCH-də olan digər hissəyə yönəldirəcəyik. Bu istifadəçi qovluğudur. Növbəti başlıqda biz FreeSWITCH istifadəçi qovluğuna tam diqqətimizi ayıracayıq.

4-cü başlıq

SIP və istifadəçi qovluğu

Öncəki başlıqlarda biz SIP(Session Initiation Protocol) haqqında qısaca danışdıq. Telefonun FreeSWITCH-də necə qeydiyyata alınmasından danışdıq. Bu başlıqda istifadəçiləri əlaqələndirmək üçün SIP-in daxildə və bütün dünyada danışması üçün daha detallı danışacayıq. VoIP protokollar içində SIP hər yerdə istifadə ediləndir.

Bu başlıqda biz aşağıdakılari açıqlayacayıq:

- FreeSWITCH istifadəçi qovluğunun başa düşülməsi.
- İlk dəfə üçün FreeSWITCH istifadəçi qovluğunun quraşdırılması və açıqlanması.
- FreeSWITCH-in xidmət təcizatçılarına qoşulması (FreeSWITCH - Aserisk SIP trunk, FreeSWITCH - CUCM SIP trunk).
- Gateway olmadan zənglərin edilməsi
- SIP profillərin və istifadəçi agentlərin qısaca müzakirə edilməsi.

FreeSWITCH istifadəçi qovluğunun başa düşülməsi

FreeSWITCH istifadəçi qovluğu bir və ya bir neçə **<domain>** elementlərindən ibarət olan mərkəzləşmiş XML sənədə əsaslanır. Hər bir **<domain>** elementində **<users>** elementləri ya da **<groups>** elementləri ola bilər. **<groups>** elementində bir və ya bir neçə **<group>** elementi ola bilər hansı ki, onlarında hər birində bir və ya bir neçə **<users>** elementi olur. Öz növbəsində **<users>** elementində də bir və ya bir neçə **<user>** elementi olur. Kiçik nüsxə ilə göstərsək aşağıdakı kimi olacaq:

```
<section name="directory">
  <domain name="example.com">
    <groups>
      <group name= "default">
        <users>
          <user id="1001">
            <params>
              <param name="password" value="1234"/>
            </params>
          </user>
        </users>
      </group>
    </groups>
  </domain>
</section>
```

Qeyd: Kod nüsxələrini rəsmi saytımız <http://opensource.az> ünvanına daxil olub, qeydiyyat ünvanında tələb edilən məlumatlarla kitabın arxasında olan üzvlük açarını yazaraq endirə bilərsiniz.

Bəzi başlanğıc elementlər istifadəçilərin qrupda yiğilmasının tələbini etməyə bilər, buna görə də **<groups>** elementini buraxıb ancaq, **<domain>** elementinin altında bir neçə **<user>** elementi yazmaq olar.

Vacib məqam odur ki, bu qovluqdan götürülən hər bir *user@domain* sistemdə olan bütün komponentlər üçün əlçatılmalıdır - Bu FreeSWITCH istifadəçi informasiyasının saxlanması üçün yeganə mərkəzləşdirilmiş qovluqdur. Əgər siz SIP telefonda istifadəçi kimi qeydiyyatdan keçmək ya da voicemail mesajı yerləşdirmək istəsəniz, FreeSWITCH istifadəçi verilənləri üçün eyni ünvana baxacaq. Bu vacibdir ona görə ki, istifadəçi datasının təkrarlanması limitləyir və buna görə də hər komponentin öz istifadəçilərini ayrı izləməsindən daha effektiv olur.

Sistem bir neçə istifadəçi olan kiçik ofislərdə çox rahat işləyir amma, istifadəçi sayı minlərlə olsa nə olacaq? Müştəri istifadəçi qovluğunu təmin etmək üçün özündə mövcud olan minlərlə istifadəçi saxlanılan bazanı FreeSWITCH-ə qoşmaq istəsə necə olacaq? Birinci başlıq *FreeSWITCH-in arxitekturasında* danışdığınız **mod_xml_curl** istifadə edərək web servis yarada bilərik ki, istifadəçi qovluğunda yazmaq üçün müraciət alır(Demək olar ki, HTML web formanın göndərdiyi nəticə ilə eyni olur). Öz növbəsində, formatından asılı olmayaraq web servis mövcud istifadəçi bazasına sorğu edə bilər və XML yazıları ilə FreeSWITCH resetri formatına yıga bilər.

mod_xml_curl axtarış sorğusu edilən modula verilənləri qaytarır. Bu o

deməkdir ki, heç bir əlavə yüklenmə etmədən mövcud baza ilə integrasiya mövcuddur. Həmçinin yənə də əvvəlki kimi bütün verilənlər mərkəzi qeydiyyat ünvanında qalacaq.

Istifadəçi qovluğuna FreeSWITCH çərçivəsində olan istənilən alt sistemi istifadə hüququ ala bilər. Buna daxildir modullar, scriptlər və FSAPI interfeysi. Bu başlıqda biz Sofia SIP modulunun istifadə edilməsi ilə sizin program SIP telefonu və avadanlıq sip telefonun necə qeydiyyatdan keçməsini öyrənəcəyik. Əgər siz programçisinizsa, bəzi əla imkanların sayəsində istifadəçi kataloqunda **<variables>** elementini **<domain>**, **<group>** ya da **<user>** elementinə əlavə edə bilərsiniz. Bu element sizə çoxlu **<variable>** elementləri təyin etmə imkanı yaradır və sayəsində hər bir qeydiyyatdan keçmiş istifadəçinin zəngi üçün çoxlu kanal dəyişənləri təyin edə bilərsiniz. Bu sizə Dialplan zamanı çox yararlı olacaq ona görə ki, spesifik istifadəçi yönləndirilməsində fərqli dizaynların yerinə yetirilməsində çox rahatdır. Həmçinin CIDR quruluşu istifadə edərək uzaqdan qeydiyyatdan keçən istifadəçilərin IP ünvan aralıqlarını da təyin eləmək mümkündür. Əgər sizin istifadəçiləriniz uzaqdan qoşulduğda həmişə eyni IP ünvan istifadə edirsə, bu onların istifadəçi adı və şifrə daxil etmək tələbini aradan qaldırır (Yəni artıq istifadəçi adı və şifrə daxil etməyə ehtiyac qalmır).

Qeyd: **Authentication** – istifadəçinin təyin edilməsi prosesidir. Authorization – istifadəçinin hansı səviyyə yetkiyə malik olmasını təyin edən prosesdir. Authentication "Bu istifadəçi həqiqətən də özünü təqdim edəndirmi?" sualına cavab verir. Authorization "Bu istifadəçiyə bizim mühitdə nə etməyə izin var?" sualına cavab verir. Siz IP Auth və Digest Auth ifadələrini gördükdə, yadda saxlayın onlar istifadəçinin təyin edilməsi üçün iki əsas üsuldur. **IP authorization** istifadəçilərin IP ünvanlarına əsaslanır. **Digest authentication** isə istifadəçi adı və şifrəyə əsaslanır. SIP(və FreeSWITCH) hər iki metodu istifadə edə bilir. Digest authentifikasiya metodunun işləməsini təyin etmək üçün https://en.wikipedia.org/wiki/Digest_access_authentication linkinə müraciət edə bilərsiniz.

Qovluq təmiz XML-ə əsaslanır. Bu da bizə genişlənmə üstünlüyü verir. XML genişlənmə imkanına sahib olduğuna görə, qovluqda genişlənə bilir. Əgər biz qovluğa yeni element əlavə etmək istəyiriksə, sadəcə mövcud XML strukturuna onu əlavə etmək bəs edir.

Ilk dəfə üçün FreeSWITCH istifadəçi qovluğunun quraşdırılması və açıqlanması

Nüsxə quraşdırmasının qovluqda 20 istifadəçi üçün domain adı var. Istifadəçilər çox asanlıqla əlavə edilə və silinə bilər. Sisteminizdə istifadəçilərin hansısa bir sayı təyin edilməsinə heç bir limit yoxdur(yəni istifadəçilərin sayı yalnız serverinizin gücündən asılıdır). Bütün istifadəçilər birlikdə **directory** kimi adlanır. Istifadəçilər bir və ya bir neçə qrupa aid edilə bilər. Sonda bütün istifadəçilər bir domain altına aid edilirlər. Susmaya görə FreeSWITCH serverin IP ünvani **domain** adlanır.

Qeyd: FreeSWITCH həmçinin çoxqatlı domain dəstəkləyir. Siz bunun haqqında məlumatı göstərilən <http://wiki.freeswitch.org/wiki/Multi-tenant> linkdən əldə edə bilərsiniz.

Növbəti seksiyalarımızda biz aşağıdakı başlıqları açıqlayacaqıq:

- Istifadəçi bacarıqları
- Yeni istifadəçinin əlavə edilməsi
- VoiceMail-in sınaqdan keçirilməsi
- Istifadəçi qrupları

Istifadəçi imkanları

Gəlin istifadəçinin təyin edilməsi üçün XML fayla baxaq. İstənilən mətn redakteedicisi ilə **/usr/local/freeswitch/conf/directory/default/1000.xml** faylını açın. Siz aşağıdakı kontentə oxşarını görməlisiniz:

```
<include>
<user id="1000">
<params>
<param name="password" value="${default_password}"/>
<param name="vm-password" value="1000"/>
</params>
<variables>
<variable name="toll_allow" value="domestic,international,local"/>
<variable name="accountcode" value="1000"/>
<variable name="user_context" value="default"/>
<variable name="effective_caller_id_name" value="Extension 1000"/>
<variable name="effective_caller_id_number" value="1000"/>
<variable name="outbound_caller_id_name" value="${outbound_caller_name}"/>
<variable name="outbound_caller_id_number" value="${outbound_caller_id}"/>
<variable name="callgroup" value="techsupport"/>
</variables>
</user>
</include>
```

XML strukturu çox adıdır. **<include>** taglarının daxilində istifadəçi aşağıdakılara sahib olur:

- **id** atributu ilə olan **user** elementi
- **params** elementi harda ki, parametrlər təyin edilir
- **variables** elementi harda ki, kanal dəyişənləri təyin edilir

Bu faylda olan bütün sintaksisi anlamasaq da ən azınnan deyə bilərik ki, istifadəçi ID-si **1000** və **password** və **vm-password**(voicemail şifrəsi) nə işə yarayır. Göstərilən quraşdirmalarda **password** parametri qoşulma şifrəsini nəzərdə tutur(Biz bunun haqqında 3-cü başlıqda *FreesWITCH-lə işləməsi üçün SIP telefonun quraşdırılması* seksiyasında danışmışdıq). Görünən **\$\$default_password** ifadəsi isə, **/usr/local/freeswitch/conf/vars.xml** faylında təyin edilən global dəyişən olan **default_password**-a yönləndirilir. Yəqin artıq **vm** simvollarından anlamısınız ki, **vm-password** ifadəsi voicemail şifrəsi deməkdir. Bu məna istifadəçinin öz voicemail-ni yoxlamaq istədiyi anda daxil edəcəyi simvollardır çünki, əks halda şifrə olmazsa hər bir şəxs digərinin voicemailini yoxlaya bilər. **value**-də qeyd edilən rəqəm isə eynilə istifadəçi **id**-si təyin edilmişdir.

Əlavə olaraq bu istifadəciyə təyin edilən çoxlu kanal dəyişənləri mövcuddur. Onlardan əksəriyyəti Dialplan-la birbaşa əlaqələnirlər. Növbəti cədvəl hər bir dəyişən və onların nə üçün istifadə edilməsini açıqlayır:

Dəyişən	Təyinatı
toll_allow	Istifadəçi edə biləcəyi zənglərin tiplərini təyin edir
accountcode	CDR məlumatlarında müəyyən edilən sərbəst məna
user_context	Bu şəxs zəng etdikdə istifadə edilən, Dialplan konteksti
effective_caller_id_name	Caller ID name zəng edilən tərəfin ekranında qeydiyyatdan keçmiş istifadəçinin adını göstərir
effective_caller_id_number	Caller ID number zəng edilən tərəfin ekranında qeydiyyatdan keçmiş istifadəçinin nömrəsini göstərir
outbound_caller_id_name	Çıxan zənglər üçün təçizatçıya ötürülən Caller ID ad
outbound_caller_id_number	Çıxan zənglər üçün təçizatçıya ötürülən Caller ID nömrə
callgroup	Dialplan ya da CDR-da istifadə edilə bilən təsadüfi rəqəm

Nəticə olaraq, istifadəçinin susmaya görə olan quraşdirmaları aşağıdakılara sahibdir:

- SIP və qeydiyyat üçün istifadəçi adı
- Voicemail şifrəsi
- İstifadəciyə icensə/məhdudiyyət imkanı
- Zəng edən tərəfin ID emali imkanı
- Tələbdən asılı olaraq istifadə edilə biləcək ya da məhəl qoyulmayacaq bir neçə dəyişən

Gəlin artıq qovluğumuza yeni istifadəçi əlavə edək.

Yeni istifadəçinin əlavə edilməsi

Bir və ya bir neçə istifadəçinin əlavə edilməsi adı iki addımlı işdir və aşağıda göstərilir:

1. Mövcud XML fayldan yeni istifadəçi üçün birini nüsxələyirik.
2. **Local_Extension** dialplan yazısında dəyişiklik edin.

Bu misalda biz Elçin adlı şəxs üçün istifadəçi adı 1100 olan hesab yaradırıq. Aşağıdakı addımları edirik:

1. Terminalı açırıq və **/usr/local/freeswitch/conf/directory/default** qovluğuna daxil oluruq.
2. **1000.xml** faylini **1100.xml** adlı fayla nüsxəleyin. Aşağıdakı kimi:


```
# cd /usr/local/freeswitch/conf/directory/default
# cp 1000.xml 1100.xml
```
3. Mətn redaktorunda **1100.xml** faylini açın və aşağıdakı dəyişiklikləri edin:
 - o İstənilən 1000 olan yerləri 1100 ilə dəyişin
 - o **effective_caller_id_name** olan yerin mənasını Elçin yazın
4. Yeni fayl aşağıdakı kimi görünəcək:


```
<include>
<user id="1100">
  <params>
    <param name="password" value="${default_password}"/>
    <param name="vm-password" value="3337771100"/>
  </params>
  <variables>
    <variable name="toll_allow" value="domestic,international,local"/>
    <variable name="accountcode" value="1100"/>
    <variable name="user_context" value="default"/>
    <variable name="effective_caller_id_name" value="Elcin"/>
    <variable name="effective_caller_id_number" value="1100"/>
    <variable name="outbound_caller_id_name" value="${outbound_caller_name}"/>
    <variable name="outbound_caller_id_number" value="${outbound_caller_id}"/>
    <variable name="callgroup" value="techsupport"/>
  </variables>
</user>
</include>
```
5. Faylı yadda saxlayın. Sonra biz **Local_Extension** üçün Dialplan verilənini dəyişməliyik. Mətn redatorunda **/usr/local/freeswitch/conf/dialplan/default.xml** faylini açın və aşağıdakı sətirləri tapın:


```
<extension name="Local_Extension">
  <condition field="destination_number" expression="^(\d{1}[01]\d{3})$">
```

Bu Dialplan genişlənməsidir və adından göründüyü kimi, zəngləri daxili genişlənmələrə yönəldirir. Misalımızdakı daxili genişlənmə qovluğumuzda olan istifadəciyə qeyd edilmiş telefondur. Xatırladaq ki, qovluqda FreeSWITCH öncədən quraşdırılmış 20 ədəd istifadəçi ilə gəlir və onların rəqəmləri 1000 ilə 1019 arasıdır. Susmaya görə 1000,1001 və 1019-adək olan nömrələrə edilən istənilən zənglər **Local_Extension** adlı Dialplan verilənində emal ediləcək. Biz 1100 nömrəsini müntəzəm ifadələrə əlavə etməliyik. Yuxarıda **/usr/local/freeswitch/conf/dialplan/default.xml** faylında göstərdiyimiz ikinci sətiri aşağıdakı şəklə gətiririk:

```
<condition field="destination_number" expression="^(10[01][0-9] | 1100)$">
```

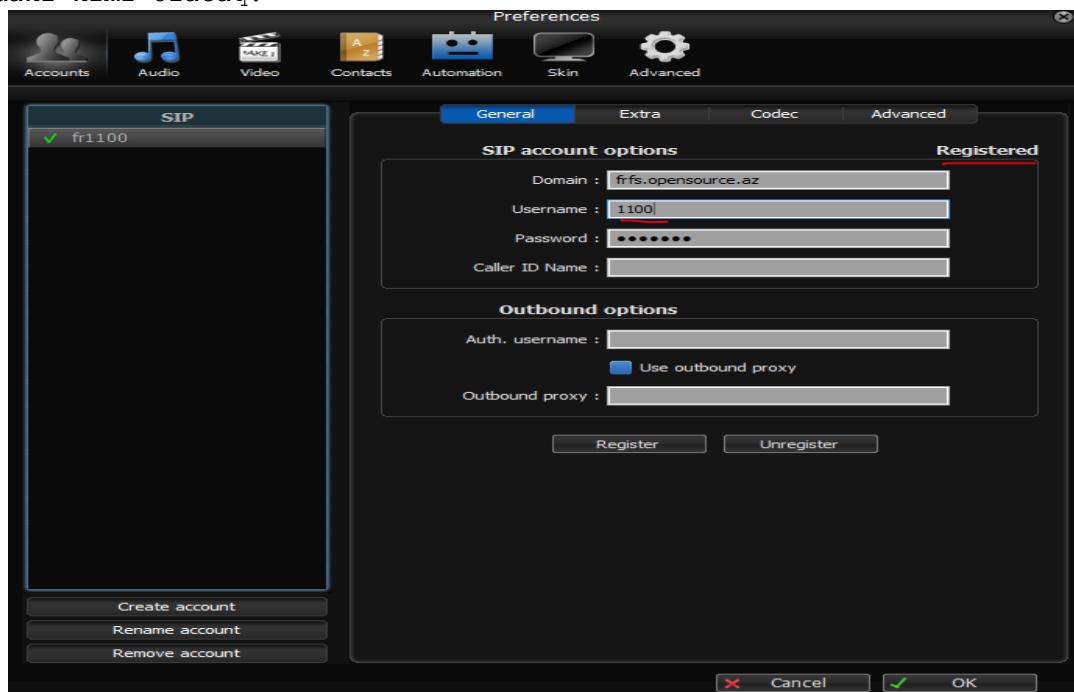
Faylı yadda saxlayırıq və çıxırıq (Müntəzəm ifadələri biz 5-ci başlıqda XML Dialplanın başa düşülməsinin Müntəzəm Ifadələr seksiyasında ətraflı açıqlayacaq).

Son görəcəyimiz iş XML quraşdırmasının yenidənyüklənməsini etməkdir. **fs_cli-a** daxil olun və **reloadxml** əmrini aşağıdakı kimi işə salın:

```
freeswitch@internal> reloadxml
+OK [Success]
2015-08-28 11:34:32.501262 [INFO] mod_enum.c:880 ENUM Reloaded
2015-08-28 11:34:32.501262 [INFO] switch_time.c:1411 Timezone reloaded 1781
definitions
```

Qeyd: Linux/UNIX istifadəçiləri işlərini bir səhifədə görərək sinaqlarını digər pəncərədə fərqli SSH sessiyalarının sayəsində görə bilərlər. Hətta daha da geniş fərqli sessiyalar istifadə etmək istəsəniz açıq kodlu screen program təminatından yararlana bilərsiniz. Rəsmi səhifəsi <http://www.gnu.org/software/screen/> ünvanından ətraflı məlumat əldə edə bilərsiniz.

Bizim yeni genişlənməmiz təyin edilmişdir və artıq SIP telefon vasitəsilə 1100 nömrəsinə qeydiyyatdan keçirə bilməliyik. Qoşulma üsullarını artıq 3-cü başlıqda SIP telefonun FreeSWITCH-lə işləməsi üçün quraşdırılması mövzusunda açıqlamışdıq. Zoiper program SIP telefonunun **1100** nömrəsinə qeydiyyatı aşağıdakı kimi olacaq:



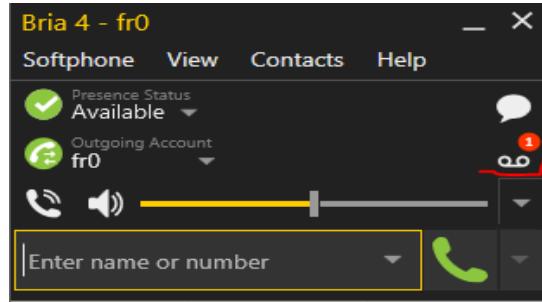
Qeydiyyatdan keçmiş telefon artıq çıkışa zəngləri edə bilər və **1100** nömrəsinə gələn zəngləri qəbul edə bilər.

Qeyd: Qeydiyyatdan keçmiş SIP telefonları görmək üçün FreeSWITCH **fs_cli**-dan **sofia status profile internal reg** əmrini yazıb baxa bilərsiniz.

Artıq uğurla yeni istifadə etdik. Növbəti sinağımızı voicemail-lə edirik.

VoiceMail-in sinaqdan keçirilməsi

Directory(qovluq)-da hər bir istifadəçinin səs(voice) məktub yesiyi(mailbox) mövcuddur ki, digərləri tərəfindən orda səs saxlanıla bilsin. Dialplan nüsxəsinin quraşdırımalarında istifadəçiye edilən zəng **30** saniyə ərzində cavablaşdırılmazsa, yönəldiriləcək həmin istifadəçinin voicemail-inə. İşləməsinə əmin olmaq üçün sinaq zəngi edib yoxlayın. Bir nömrədən digərinə zəng edin və cavab vermədən gözləyin. **30** saniyə zəng getdikdən sonra, voicemail sistemi cavab verəcək və sizə voicemail yerləşdirməyi təklif edəcək. Yerləşdirəcəyiniz mesaj ən azı **3** saniyə olmalıdır(minimal uzunluq). Əgər sizdə bir ədəd telefondursa, hətta özünüz özünüzə zəng edib yoxlaya bilərsiniz. Nəticədə voicemail saxlanılmış istifadəçinin program SIP telefonunda aşağıdakı şəkil görünməlidir:



Nəzərə alın ki, voicemail və qəbul edilməyən zənglərin indikator işarələri fərqlidir. Zənglər üçün olan işarə aşağıdakı şəkildəki kimidir:



Qeyd: Səs mailini tez zamanda yadda saxlayıb çıxmaq üçün "#" simvolunu sixın ki, menyunun ardına qulaq asmayasınız.

Səs mesajının əldə edilməsi də çox asandır. Sadəcə ***98** ya da **4000** nömrəsinə zəng edin. Sistem sizi sistemə daxil olan kimi, yeni səs mesajlarının dinişənilməsi üçün ID yazılımasını təklif edəcək. Adı seans aşağıdakı şəkildə ardıcılıqla olacaq(**pound** bu -> "#" simvoldur):

***98**

"Please enter your ID, followed by pound"

1100#

"Please enter your password, folowed by pound"

3337771100#

"You have one new message."

Əgər istifadəçinin yeni mesajı varsa, sistem avtomatik olaraq onun saxlanılması tarixi və saatını bildirəndən sonra işə salacaq. Susmaya görə olan voicemail menyusu aşağıdakı şəkildə quraşdırılıb:

Əsas menyu:

- 1 - Listen to new messages
- 2 - Listen to saved messages
- 5 - Options menu(yazılmış ad, salamlaşma, şifrə dəyişmə və.s)
- # - Exit voicemail

Mesajı dinlədiyimiz halda:

- 1 - Replay message from the beginning
- 2 - Save message
- 4 - Rewind
- 6 - Fast-forward

Mesajı dinlədiyinizdən sonra:

- 1 - Replay message from the beginning
- 2 - Save message
- 4 - Send to e-mail (requires configuration)
- 7 - Delete message

Çoxlu sınaqlar edin ki, quruluşu tam mənimsəyəsiniz. Öz voicemail-nizə daxil olun və **5** düyməsini sıxaraq salamlaşma menyusuna daxil olun və bir neçə salamlaşma qeydə alın. Susmaya görə 1-dən 9-adək salamlaşma qeydə ala bilərsiniz(Əksər hallarda bütün istifadəçilər 1 salamlaşma yazısı qeydə alırlar). Adətən elə 1 rəqəmli salamlaşma istifadə edilir.

Qeyd: Artıq voicemail-in tam işlək vəziyyətini yoxladıqdan sonra, biz da çox istifadə edilən istifadəçi qruplarına diqqətimizi ayıra bilərik.

Istifadəçi qrupları

Geniş quruluşlu olan şirkətlərdə bir neçə telefona zəng etmək tələbi tez-tez yaranır. Misal üçün şirkətin şöbəsi ola bilər ki, onun bütün işçiləri bu departamentə gələn zəngləri qəbul edə bilər. Eynilə bütün bu işçilərin daxili telefonu olduğuna görə, onlar hər biri gələn zəngi ayrılıqda qəbul edə bilər. FreeSWITCH-in directory(qovluq) imkanı var hansı ki, sayəsində bir neçə istifadəçini eyni qrupa əlavə etmək mümkündür. Bir istifadəçi eyni zamanda bir neçə qrupda ola bilər.

Qeyd: Bəzi PBX sistemlər var ki, ACD(Automatic Call Distribution) vasitəsilə daxil olan zənglərin yönləndirilməsini daha da irəliləmiş formasını istifadə edirlər. Zəng qrupları bu tip programlarda istifadə edilmir. Bu müzakirə bizim mövzumuzu aşsa da, siz **FIFO** müraciətləri haqqında daha ətraflı https://freeswitch.org/confluence/display/FREESWITCH/mod_fifo linkindən oxuya bilərsiniz.

Qruplar **/usr/local/freeswitch/conf/directory/default.xml** faylinda təyin edilir. Faylı açın və **groups** düyüünü tapın. Diqqət yetirin ki, orda 4 ədəd öncədən təyin edilmiş qrup mövcuddur. Onlar aşağıdakılardır:

- Default - Qovluqda olan **bütün** istifadəçilər
- Sales - **1000-1004** aralığı
- Billing - **1005-1009** aralığı
- Support - **1010-1014** aralığı

Son 3 qrup yalnız sinaqlar üçün öncədən təyin edilmişdir və istəsəniz onları rahat şəkildə silə ya da dəyişə bilərsiniz. "**default**" qrupu çox maraqlıdır ona görə ki, o directory(qovluq)-da olan bütün istifadəçiləri özündə təşkil edir(Ehtiyatla istifadə edin!). Gəlin yeni qrup əlavə edək və sonra qrupların necə işləməsini öyrənək. Aşağıdakı addımları yerinə yetirin:

1. **/usr/local/freeswitch/conf/directory/default.xml** faylını açın və aşağıdakı sətirləri **groups** düyüünün içində əlavə edin:


```
<group name="custom">
<users>
  <user id="1000" type="pointer"/>
  <user id="1100" type="pointer"/>
</users>
</group>
```
2. Əgər sizdə artıq qeydiyyatdan keçmiş **2** və daha artıq başqa nömrələr mövcuddursa, onları istifadə edin. Faylı yadda saxlayın.
3. **fs_cli** əmri ilə FreeSWITCH console-a daxil olun. **F6** ya da **reloadxml** əmrini daxil edin ki, quraşdılmalarımız yenidən yüklənsin.

Əmin olun ki, yeni **custom** adlı qrup düzgün əlavə edilmişdir. Bunun üçün **fs_cli**-da **group_call** əmrindən istifadə eləmək lazımdır. İşlək vəziyyətdə aşağıdakına uyğun olan bir sətir əldə etməlisiniz:

```
freeswitch@internal> group_call custom
[^^:sip_invite_domain=100.120.81.154:presence_id=1000@100.120.81.154]so
fia/internal/sip:1000@85.132.57.60:62144;rinstance=96dd923e82b6e81c,[^
:sip_invite_domain=100.120.81.154:presence_id=1100@100.120.81.154]sofia
/internal/sip:1100@85.132.48.165:49477
```

Əgər telefonun hansısa biri qeydiyyatda olmazsa aşağıdakı sətiri əldə edəcəksiniz:

```
freeswitch@internal> group_call custom
[^^:sip_invite_domain=100.120.81.154:presence_id=1000@100.120.81.154]so
fia/internal/sip:1000@85.132.57.60:62144;rinstance=96dd923e82b6e81c,[^
:sip_invite_domain=100.120.81.154:presence_id=1100@100.120.81.154]error
/user_not_registered
```

Göstərilən bu kod nə deməkdir? **group_call** əmri istifadə edilir ki, çoxlu telefona zəng etmək üçün **SIP dialstring** yaradılsın. Bizim nüsxələrdə **1100** telefonunun qeydiyyatdan keçməyən hissəsini göstərdik və bu o deməkdir ki, zəng gəldiyi halda həmin istifadəçi zəngə cavab verməyəcək(**user_not_registered** səhvi həmin mənənəni verir). Ancaq 1000 nömrəsi problemsız qeydiyyatda idi və əgər qrupa zəng gələrsə, qeydiyyatda olmayan istifadəçiye məhəl qoymadan digərlərinə zəng gedəcək. Yeni qrupa zəng etməzdən önce, biz Dialplana aşağıdakı sətirləri əlavə etməliyik:

1. `/usr/local/freeswitch/conf/dialplan/default.xml` faylini açın və `group_dial_billing` olan ərazini tapın:


```
<extension name="group_dial_billing">
  <condition field="destination_number" expression="^2002$">
    <action application="bridge" data="group/billing@${domain_name}"/>
  </condition>
</extension>
```
2. Aşağıdakı yeni sətirləri `group_dial_billing` genişlənməsində `</extension>` tag-dan sonra əlavə edin:


```
<extension name="group_dial_custom">
  <condition field="destination_number" expression="^2003$">
    <action application="bridge" data="group/custom@${domain_name}"/>
  </condition>
</extension>
```
3. Faylı yadda saxlayın.
4. `fs_cli`-a daxil olun və `reloadxml` əmrini işə salın.
5. Yeni qurdugunuz qrupa zəngi **2003** nömrəsinə yiğaraq sınaqdan keçirin. Bundan sonra qrupda olan bütün istifadəçilərə zəng gedəcək.

Qrupda olan bütün telefonların zəngi çaldıqda, ilk cavab verən qalib gəlir və zəngə cavab vermiş olur. Digər telefonların zəngi dayanır. Biz telefonların necə FreeSWITCH serverə qoşulmasını və onların imkanlarının bir hissəsini gördük. Gəlin artıq zənglərin kənarə gedişi haqqında öyrənək.

FreeSWITCH-in xidmət təçizatçılarına qoşulması (FreeSWITCH - Aserisk SIP trunk, FreeSWITCH - CUCM SIP trunk)

Iki ədəd serverin bir-biri ilə qeydiyyatdan keçməsi üçün hər birindən digərinə veriləcək istifadəçi adı və şifrə olmalıdır. Bu işi görmək üçün gateway-lərdən istifadə edilir. Gateway - digər SIP serverlə qeydiyyatdan keçmək üçün çox adı bir üsuldur. Bu **SIP REGISTER** autentifikasiya zəngləri cəhdlerini **INVITE** mesajlarında olduğu kimi əlavə edir. Telefon xidməti təçizatçıları çox böyük serverlərdən istifadə edirlər(bəzi işləyən FreeSWITCH-ləri əlavə etməklə) ki, öz müraciətlçilərinə SIP trunklar verə bilsinlər. FreeSWITCH-lə də biz SIP təçizatımıza qoşulmaq üçün gateway

istifadə edə bilərik. Biz həmçinin gateway-i istifadə edərək digər SIP serverlə(misal üçün FreeSWITCH kimi digər server) əlaqələnə bilərik.

Yeni gateway-in işə salınması

Gateway eynilə SIP telefonun FreeSWITCH serverə qoşulduğu qaydada SIP serverə qoşulur. Eynilə gateway quraşdırması elə SIP telefonun quraşdırmasına müəyyən məqamlarda oxşayır. FreeSWITCH-də qeydiyyatdan keçən SIP telefonda olduğu kimi, gateway-ində minimal tələbləri var. Onlar aşağıdakılardır:

- İstifadəçi adı və şifrə
- Server ünvanı ya da IP və port

Bu mənalar xidmət təcizatçıları tərəfindən verilir. Müəyyən hallarda bəzi digər parametrlərdə olur(Misal üçün proxy server və port). Əgər sizdə artıq SIP təcizatçısı ilə hesab mövcuddursa, onda onu öz gateway-niz üçün istifadə edə bilərsiniz. Bu misalda biz **iptel.org**-dan olan hesabı istifadə edəcəyik.

Qeyd: <http://www.iptel.org/service> ünvanına daxil olun və pulsuz SIP hesabı üçün qeydiyyatdan keçərək istifadə edə bilərsiniz. Ancaq **RYTN**-in qərarına əsasən ölkədən kənara SIP trafikin çıxışı bütün Internet xidməti təcizatçılarında bağlı olduğu üçün, sınaqlar üçün Asterisk və 1 ədəd CUCM quraşdırıldım ☺.

FreeSWITCH ilə Asterisk arasında Digest authentication SIP TRUNK

Məqsədimiz FreeSWITCH ilə Asterisk arasında istifadəçi adı ilə şifrəyə əsaslanan SIP trunkin qaldırılmasıdır. Quraşdırımızda Asterisk 13.5.0 və FreeSWITCH 1.5.final istifadə edilmişdir. Nəzərdə tutulur ki, FreeSWITCH tərəfdə olan genişlənmələr **1XXX** aralığında Asterisk tərəfdə isə **7XXX** aralığındadır.

Asterisk - **asterisk.opensource.az**

FreeSWITCH - **frfs.opensource.az**

1. Öncə Asterisk tərəf quraşdırılmamızıza baxaq

FreeBSD 10.1 x64 üzərində Asterisk13-ü aşağıdakı əmlərlə yükleyirik:

```
# cd /usr/ports/net/asterisk13/      - Port ünvanına daxil oluruq
# make config                         - Lazımı modulları seçirik
```

```
# echo 'asterisk_enable="YES"' >> /etc/rc.conf - StartUP-a əlavə edirik  
# /usr/local/etc/rc.d/asterisk start - Isə salırıq
```

```
/usr/local/etc/asterisk/sip.conf faylinda yalnız sizin aşağıdaki satırlarında  
deyişiklik etmişsiniz:
```

transport=udp,tcp

tcpenable=yes

/usr/local/etc/asterisk/sip.conf faylinin sonuna aşağıdaki sətiri əlavə edib yadda saxlayaraq çıxırıq(Bununla **sip_additional.conf** faylinin da yüklənmə zamanı oxunmasını deyirik):

```
#include sip_additional.conf
```

Eynilə `/usr/local/etc/asterisk/extensions.conf` faylinin sonuna aşağıdakı sətiri əlavə edirik ki, yüklənmədə `extensions_fs.conf` faylı da oxunsun:

```
#include extensions fs.conf
```

```
/usr/local/etc/asterisk/sip_additional.conf faylında iki ədəd genişlənmə (7000  
və 7001) və fsar adlı SIP Trunk quraşdırması olacaq. faylinin tərkibi  
aşağıdakı kimi olacaq:  
[7000]  
defaultuser=7000  
secret=freebsd
```

```

host=dynamic
context=phones
qualify=yes
transport=udp,tcp
insecure=port,invite
canreinvite=no
disallow=all
allow=alaw
type=friend

[7001]
defaultuser=7001
secret=freebsd
host=dynamic
context=phones
qualify=yes
transport=udp,tcp
insecure=port,invite
canreinvite=no
disallow=all
allow=alaw
type=friend

[fsar]
type=friend
host=dynamic
port=5060

bindport=5080

fromuser=fsar

fromdomain=frfs.opensource.az

defaultuser=fsar

secret=12345

context=incoming
transport=udp,tcp
qualify=yes
disallow=all
allow=alaw
canreinvite=no

trustrpid=yes

sendrpid=yes

```

- Qeydiyyat üçün host-da quraşdırılmış hansı porta müraciətlər yollanılacaq

- Asteriskin SIP zənglər üçün qəbul edəcəyi portun rəqəmi
- hansı istifadəçi adından asteriskə qoşulma olacaq
- hansı domain adından asteriskə qoşulma olacaq
- "SIP INVITE" mesajlarında FreeSWITCH-lə qarşılıqlı istifadə edilən istifadəçi adı
- SIP INVITE mesajlarında FreeSWITCH-lə qarşılıqlı istifadə edilən şifrə

- Clientin yenidən INVITE mesajlarının ötürməsinin qarşısını alırıq

- Remote-Party-ID ilə gələn SIP clientlərə inanırıq
- Sip client-ə Remote-Party-Id başlığını ötürürük.

`/usr/local/etc/asterisk/extensions_fs.conf` faylinin tərkibi aşağıdakı kimidir(Eyni IP trunk-da olduğu kimidir yalnız trunk adı **fsar** istifadə edilir):

```
[incoming]
exten => _7XXX,1,Dial(SIP/${EXTEN})
exten => _7XXX,n,Hangup()

[outgoing]
exten => _1XXX,1,Dial(SIP/fsar/${EXTEN})
exten => _1XXX,n,Hangup()

[phones]
include => incoming
include => outgoing
```

`/usr/local/etc/rc.d/asterisk restart` - Asteriski yenidən yükleyirik ki, dəyişikliklər işə düşsün.

Ya da etdiyimiz dəyişikliklərin dərhal işə düşməsi üçün asterisk console-da aşağıdakı əmri daxil etməniz yetər:

```
asterisk*CLI> sip reload
```

Sonda Asterisk console-a verbose rejimdə daxil oluruq və uğurlu SIP qoşulmalarına baxırıq(Göründüyü kimi **7000** nömrəli telefon və **fsar** sip trunk qoşulmuşdur):

```
# asterisk -rvvv
asterisk*CLI> sip show peers
Name/username Host Dyn Forcerport Comedia ACL Port Status Description
7000/7000 217.168.188.211 D Auto(No) No 41735 OK(45 ms)
fsar/fsar 100.120.81.154 D Auto(No) No 5080 OK(1 ms)
3 sip peers [Monitored: 3 online, 0 offline Unmonitored: 0 online, 0 offline]
```

2. FreeSWITCH tərəf quraşdirmalarımıza baxaq:

İlk işimiz `/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml` faylinda olan `<list name="domains" default="deny">` tag-in altında bütün FreeSWITCH-ə qoşulacaq həm SIP trunk və həmdə sip clientlərin PUBLIC IP ünvanlarını yazırıq(Əks halda qoşulmağa izniniz olmayacaq və problemin həllini gec anlayacaqsınız.) Aşağıdakı kimi(Bu siyahıda qoşulan SIP clientlərin fərqli ofisləri, telefonda Mobile Data-da olan SIP clientlərin IP-ləri və asterisk-in IP-si yazılmışdır):

```
<list name="domains" default="deny">
<node type="allow" domain="$$ {domain} "/>
<node type="allow" cidr="94.50.81.142/32"/>
<node type="allow" cidr="94.5.100.151/32"/>
<node type="allow" cidr="85.70.124.60/32"/>
<node type="allow" cidr="85.162.11.165/32"/>
<node type="allow" cidr="85.79.62.235/32"/>
<node type="allow" cidr="10.50.93.0/24"/>
<node type="allow" cidr="217.23.188.247/32"/>
```

```
<node type="allow" cidr="217.66.184.62/32"/>
</list>
```

`/usr/local/freeswitch/conf/dialplan/public/00_asterisk.xml` adlı fayl yaradırıq və tərkibinə aşağıdakı sətirləri əlavə etməklə deyirik ki, **7** rəqəmi ilə başlayan və sonra **3** istənilən rəqəmə zəng getdikdə onu **ast** adlı SIP trunkin üstünə ötür. Bu quraşdirmalari **asterisk** genişlənmə adına mənimsədirik:

```
<include>
  <extension name="asterisk">
    <condition field="destination_number" expression="^(7\d{3})$">
      <action application="bridge" data="sofia/gateway/ast/$1"/>
    </condition>
  </extension>
</include>
```

`/usr/local/freeswitch/conf/dialplan/public/00_asterdengelen.xml` faylinə aşağıdakı sətirləri əlavə etməklə deyirik ki, **ast** SIP trunk-dan **1234** nömrəsinə zəng gələrsə onu FreeSWITCH-də olan **1000** nömrəsinə yönləndir (Əgər **1234** nömrəsinə zəng etdikdə, paralel **1000** və **1001** nömrələrinə zəngin gedilməsini istəsəniz bu sətiri `<action application="bridge" data="user/1000@$ ${domain},user/1001@$ ${domain}" />` istifadə etməniz yetər):

```
<include>
  <extension name="from-ast">
    <condition field="destination_number" expression="^(1234)$">
      <action application="bridge" data="user/1000@$ ${domain}" />
      <action application="hangup" />
    </condition>
  </extension>
</include>
```

Hətta SIP trunk-in istifadəçi adından gələn bütün zənglərin zəng edilən şəxsin nömrəsinə birbaşa ötürülməsi üçün, `/usr/local/freeswitch/conf/dialplan/public/00_asterdengelen.xml` faylinə aşağıdakı sətirləri əlavə etməniz yeter (**\$1** zəng edilən istifadəçinin nömrəsidir ki, parameter kimi alınır):

```
<include>
  <extension name="from-ast">
    <condition field="destination_number" expression="^(fsar)$">
      <action application="transfer" data="$1 XML default" />
      <action application="hangup" />
    </condition>
  </extension>
</include>
```

`/usr/local/freeswitch/conf/sip_profiles/external/asterisk.xml` faylında SIP trunk quraşdirmalarımızı edirik ki, həm asteriskə qoşulaq və həm də asterisk bizə qoşulsun. Göstərilən quraşdirmada vacib bilməli olduğumuz hissələr SIP trunkin adı **ast**, qoşulacağımız host **asterisk.opensource.az**, istifadəçi adı ilə şifrə asteriskdə olduğu kimi eyni, istifadə ediləcək protocol **udp** və ən əsası SIP trunk-da istifadə edilən kontekst **public**-dir. Bu o deməkdir ki, zəng geldikdə public context-ə düşəcək hansı ki,

```
/usr/local/freeswitch/conf/dialplan/public.xml faylini oxuyur və <X-PRE-  
PROCESS cmd="include" data="public/*.xml"/> əlavə faylları oxuyub yerinə  
yetirir. Bu include o deməkdir ki, /usr/local/freeswitch/conf/dialplan/public  
qovluğunda olan bütün xml genişlənməli faylları yerinə yetirilsin. Bizim  
00_asterisk.xml adlı zəng planı faylimizda həmin qovluqda yerləşir.  
<include>  
<gateway name="ast">  
  <param name="username" value="fsar"/>  
  <param name="password" value="12345"/>  
  <param name="realm" value="asterisk.opensource.az"/>  
  <param name="from-user" value="fsar"/>  
  <param name="from-domain" value="asterisk.opensource.az"/>  
  <param name="proxy" value="asterisk.opensource.az"/>  
  <param name="expire-seconds" value="800"/>  
  <param name="register" value="true"/>  
  <param name="register-transport" value="udp"/>  
  <param name="context" value="public"/>  
</gateway>  
</include>
```

Adı halda FreeSWITCH öz üzərinə gələn SIP trunk qoşulmaları üçün **10000ms** intervalında yoxlanış edir onu söndürmək üçün isə,

```
/usr/local/freeswitch/conf/dialplan/default.xml faylında <condition  
field="${default_password}" sekisiyini aşağıdakı şəklə getiririk:  
<condition field="${default_password}" expression="^freebsd$" break="never">  
  <!--  
    <action application="sleep" data="10000"/>  
  -->  
</condition>
```

Etdiyimiz dəyişiklikləri işə düşməsi üçün CLI-dan FS console-a daxil oluruq və quraşdırmalarımızı yenidən yükleyib qeydiyyatdan keçmiş SIP hesablara baxırıq:

```
root@frfs:~ # fs_cli  
freeswitch@internal> sofia profile restart all reloadxml
```

Qeyd: Profaylin **restart** edilməsi mövcud profile üzərində olan bütün aktiv qoşulmaları kəsəcək. Restart edilmədən yeni profile-in işə salınması üçün alternative əmr **sofia profile <profile name> rescan reloadxml** istifadə edilə bilər.

ast adlı SIP trunk statusuna baxırıq:
freeswitch@internal> **sofia status gateway ast**

```
=====  
Name      ast  
Profile   external  
Scheme    Digest  
Realm     asterisk.opensource.az  
Username  fsar  
Password  yes
```

```

From <sip:fsar@asterisk.opensource.az>
Contact <sip:gwt+ast@100.120.81.154:5080;transport=udp;gw=ast>
Exten fsar
To sip:fsar@asterisk.opensource.az
Proxy sip:asterisk.opensource.az
Context public
Expires 800
Freq 800
Ping 0
PingFreq 0
PingTime 0.00
PingState 0/0/0
State REGED
Status UP
Uptime 19s
CallsIN 0
CallsOUT 0
FailedCallsIN 0
FailedCallsOUT 0
=====

```

Bütün SIP statuslara baxırıq:

```

freeswitch@internal> sofia status
Name          Type     Data           State
=====
external-ipv6    profile  sip:mod_sofia@[:1]:5080      RUNNING (0)
external-ipv6::example.com gateway  sip:joeuser@example.com  NOREG
100.120.81.154   alias    internal          ALIASED
external        profile  sip:mod_sofia@100.120.81.154:5080
                  RUNNING (0)
external::example.com gateway  sip:joeuser@example.com  NOREG
external::ast      gateway  sip:fsar@asterisk.opensource.az  REGED
internal-ipv6    profile  sip:mod_sofia@[:1]:5060      RUNNING (0)
internal         profile  sip:mod_sofia@100.120.81.154:5060  RUNNING (0)
=====
```

Əgər qoşulmada probleminiz yaranarsa, aşağıdakı əmrlə debug rejimi işə salaraq bütün **trace** logları analiz edə bilərsiniz:

```
freeswitch@internal> sofia global siptrace on
```

FreeSWITCH ilə Asterisk arasında IP authentication ilə SIP trunk

Nəzərdə tutulur ki, FreeSWITCH-də Nömrələr **1XXX** aralığında (Susmaya görə yükləndikdə **1000-1019** aralığında hazır nömrələr olur) və Asteriskdə isə **7XXX** aralığındadır.

Resurslarımız aşağıdakı kimidir:

frfs.opensource.az - 100.120.81.154
asterisk.opensource - 100.120.81.142

1. FreeSWITCH-ə aid olan quraşdırılmalarımızı edirik:

Siz `/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml` faylında asteriskdən gələn zənglərin giriş hüququ üçün aşağıdakı sətirləri `<network-lists>` sekiyasının daxilinə əlavə etməlisiniz.

```
<list name="asterisks" default="deny">
  <node type="allow" cidr="100.120.81.142/32"/>
</list>
```

Həmçinin `/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml` faylında domain list-ni aşağıdakı şəklə gətiririk:

```
<list name="domains" default="deny">
  <node type="allow" domain="${domain}"/>
  <node type="allow" cidr="100.120.81.142/32"/>
</list>
```

Sonra `/usr/local/freeswitch/conf/sip_profiles/external.xml` faylinin `<settings>` sekiyasına aşağıdakı sətiri əlavə edirik:

```
<param name="apply-inbound-acl" value="asterisks"/>
```

Ardınca `/usr/local/freeswitch/conf/dialplan/default.xml` faylinin `<context name="default">` sekiyasının sonlarına yaxın aşağıdakı sətirləri əlavə edirik:

```
<extension name="ast_extens">
  <condition field="destination_number" expression="^(7\d{3})$">
    <action application="set" data="hangup_after_bridge=true"/>
    <action application="bridge" data="sofia/external/$1@100.120.81.142"/>
    <action application="hangup"/>
  </condition>
</extension>
```

Adı halda FreeSWITCH öz üzərinə gələn SIP trunk qoşulmaları üçün 10000ms intervalında yoxlanış edir onu söndürmək üçün isə,

`/usr/local/freeswitch/conf/dialplan/default.xml` faylinda `<condition field="${default_password}"` sekiyasını aşağıdakı şəklə gətiririk:

```
<condition field="${default_password}" expression="^freebsd$" break="never">
  <!--
    <action application="sleep" data="10000"/>
  -->
</condition>
```

2. Asterisk-ə aid olan quraşdirmalarımızı edirik:

`/usr/local/etc/asterisk/sip.conf` faylinin sonuna aşağıdakı sətiri əlavə edirik, yeni quraşdırma faylimizda əlavə olsun:

```
#include sip_additional.conf
```

Eynilə `/usr/local/etc/asterisk/extensions.conf` faylinin sonuna aşağıdakı sətiri əlavə edirik ki, genişlənmələrimiz üçün yeni quraşdırma faylimız əlavə edilsin:

```
#include extensions_fs.conf
```

/usr/local/etc/asterisk/sip.conf faylında **[general]** seksiyasının altında aşağıdakı kimi, TCP ilə qoşulmaları və ötürülmə üçün hər iki protokolu aktivləşdiririk:

```
tcpenable=yes
transport=udp,tcp
```

/usr/local/etc/asterisk/sip_additional.conf yeni fayl yaradırıq və aşağıdakı mətni içində əlavə edib yadda saxlayaraq çıxırıq. Gördüyüümüz kimi iki ədəd telefon **7000** və **7001** rəqəmləi telefonlar və **fs_trunk** adlı trunk quraşdırılmışıq.

[7000]

```
defaultuser=7000          - Yeni istifadəçi
secret=freebsd            - 7000 istifadəçisinin şifrəsi
host=dynamic              - İstənilən IP ünvandan qoşula bilər
context=phones             - Dialplan phones-a görə zəng edə bilər
qualify=yes                - Alətin qoşulmasını yoxlayırıq
transport=udp,tcp          - Hər iki protokolla qoşulmağa izin veririk
insecure=port,invite        - Qeydiyyatdan keçmək üçün başlanğıc INVITE mesajını tələb etməmək və müraciət gələn porta məhəl qoymamaq
canreinvite=no             - NAT istifadə edildikdə yenidən INVITE mesajların göndərilməsi üçün istifadə edilir. Bizim halda NAT yoxdur.
disallow=all                - Bütün kodeklərin istifadəsinə qadağa qoyulur
allow=alaw                  - ulaw kodekin istifadəsinə izin verilir
type=friend                 - firend həm istifadəçi və həmdə qarşı tərəf üçündür.
                             Bu adətən bütün stolüstü telefonlar üçün və digər alətlərin istifadəsi üçün nəzərdə tutulur. Bu halda asterisk həm istifadəçi və həmdə qarşı tərəf üçün eyni adla iki obyekt yaradacaq,
```

[7001]

```
defaultuser=7001
secret=freebsd
host=dynamic
context=phones
qualify=yes
transport=udp,tcp
insecure=port,invite
canreinvite=no
disallow=all
allow=alaw
type=friend
```

[fs_trunk]

```
type=friend
host=frfs.opensource.az   - Hansı host ilə SIP trunk edəcəyik
context=incoming           - fs_trunk-ımızın incoming konteksti ilə işləyəcəyini deyirik
transport=udp,tcp          - Trunkımızın hər iki protokolu istifadə edilməsinə izin veririk
qualify=yes
insecure=port,invite
disallow=all
allow=alaw
```

```
canreinvite=no
```

```
/usr/local/etc/asterisk/extensions_fs.conf faylina yuxarıda yazdığımız
kontekstləri əlavə edirik:
[incoming]
exten => _7XXX,1,Dial(SIP/${EXTEN}) - 7000-dən gələn zəngləri qəbul edirik
exten => _7XXX,n,Hangup()

[outgoing]
exten => _1XXX,1,Dial(SIP/fs_trnk/${EXTEN}) - 1XXX nömrələrinə gedən
zəngləri fs_trunk-a ötürürük
exten => _1XXX,n,Hangup()

[phones]
include => incoming
include => outgoing
```

Sonda iki telefonu qeydə alıb bir birlərinə zəng edin. Mən **frfs.opensource.az** üzərində **1000** nömrəsini və **asterisk.opensource.az** üzərində **7000** nömrəsini qeydə almışdım.

Asterisk tərəfdə aşağıdakı əmrlə qoşulmalara baxırıq:

```
asterisk*CLI> sip show peers
Name/username Host      Dyn Forcerport Comedia  ACL Port Status Description
7000/7000   (Unspecified) D  Auto(No)    No          0     UNKNOWN
7001/7001   (Unspecified) D  Auto(No)    No          0     UNKNOWN
fs_trnk    100.120.81.154 Auto (No)   No          5060 OK (1 ms)
3 sip peers [Monitored: 1 online, 2 offline Unmonitored: 0 online, 0 offline]
```

FreeSWITCH tərəfdə aşağıdakı əmrlə zəng müddətində real zəngə baxa bilərsiniz:

```
freeswitch@internal> show calls
uuid,direction,created,created_epoch,name,state,cid_name,cid_num,ip_addr,dest
,presence_id,presence_data,callstate,callee_name,callee_num,callee_direction,
call_uuid,hostname,sent_callee_name,sent_callee_num,b_uuid,b_direction,b_crea
ted,b_created_epoch,b_name,b_state,b_cid_name,b_cid_num,b_ip_addr,b_dest,b_pr
esence_id,b_presence_data,b_callstate,b_callee_name,b_callee_num,b_callee_dir
ection,b_sent_callee_name,b_sent_callee_num,call_created_epoch
9139a378-ad57-e511-8f8b-0050568873a8,inbound,2015-09-10
16:17:06,1441883826,sofia/internal/7000@100.120.81.142,CS_EXECUTE,as7000,7000
,100.120.81.142,1000,7000@100.120.81.142,,EARLY,,,,,frfs.opensource.az,,,
#####
05cfa578-ad57-e511-8f8b-0050568873a8,outbound,2015-09-10
16:17:06,1441883826,sofia/internal/1000@85.132.57.60:58885,CS_CONSUME_MEDIA,a
s7000,7000,100.120.81.142,1000,1000@100.120.81.154,,RINGTON,Outbound
Call,1000,,9139a378-ad57-e511-8f8b-
0050568873a8,frfs.opensource.az#####
2 total.
```

FreeSWITCH ilə CUCM 10.1 arasında SIP trunk-in qurulması

Məqsədimiz CUCM olan **7000-7020** aralığında olan nömrələrdən FreeSWITCH üzərinə **1XXX** nömrələrinə zəngin gəlməsi və geriyə qayıtmamasıdır. CUCM 10.1 tərəfində olan Trunk-in qaldırılmasını siz **CUCM 10 - Trunk Configuration.docx** sənədindən oxuya bilərsiniz.

Bu quraşdırımızda aşağıdakı domain adları istifadə edilir:

frfs.opensource.az - FreeSWITCH

cucm.opensource.az - Cisco Unified Communications Manager

FreeSWITCH-tərəfdə isə eynilə Asterisk-də olan IP authentication kimi, aşağıdakı quraşdirmaları edirik.

/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml faylında **<list name="domains" default="deny">** tag-ının içine CUCM IP ünvanını aşağıdakı şəkildə əlavə edirik ki, qoşulmaq izni olsun:

```
<list name="domains" default="deny">
    <node type="allow" domain="${domain}" />
    <node type="allow" cidr="100.120.81.143/32"/>
</list>
```

/usr/local/freeswitch/conf/dialplan/public/00_cucm.xml faylinə aşağıdakı sətirləri əlavə edərək deyirik ki, **7XXX** genişlənməsi ilə gələn bütün nömrələri CUCM-ə ötürürük:

```
<extension name="cucm_extens">
    <condition field="destination_number" expression="^(7\d{3})$">
        <action application="set" data="hangup_after_bridge=true"/>
        <action application="bridge" data="sofia/external/$1@100.120.81.143"/>
        <action application="hangup"/>
    </condition>
</extension>
```

Sonra misal üçün CUCM-dən **1234** nömrəsinə gələn zəng-i daxili **1019** nömrəsinə yönləndirilməsi üçün

/usr/local/freeswitch/conf/dialplan/public/00_cucmdengelen.xml faylinə aşağıdakı sətirləri əlavə edirik:

```
<include>
    <extension name="from-cucm">
        <condition field="destination_number" expression="^(1234)$">
            <action application="bridge" data="user/1019@${domain}"/>
            <action application="hangup"/>
        </condition>
    </extension>
</include>
```

Qeyd: Adı halda əgər siz SIP trunk-ı

/usr/local/freeswitch/conf/dialplan/public/ qovluğun içindən quraşdırırsınızsa, bu halda bütün trafik çöldən gəlmış hesab ediləcək və FreeSWITCH trunk-dan gələn trafiki **default** dialplan əvəzinə **public** dialplan üzərinə atacaq. Bunun üçün lazımlı olacaq ki, qoşulan tərəfə

/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml faylında izin verilsin.

Ancaq siz sip trunk-ı external üzerinde quraşdırırsaz və dialplani **default** təyin etsəniz, o halda bütün səs trafiki stabil olaraq dialplanlar üzərindən keçmiş olacaq. Misal üçün

/usr/local/freeswitch/conf/sip_profiles/external/cucm.xml faylin tərkibi aşağıdakı kimi olacaq:

```
<include>
<gateway name="cucm">
  <param name="username" value="not-used"/>
  <param name="password" value="not-used"/>
  <param name="realm" value="100.120.81.143"/>
  <param name="from-domain" value="100.120.81.143"/>
  <param name="proxy" value="100.120.81.143"/>
  <param name="expire-seconds" value="800"/>
  <param name="caller-id-in-from" value="true"/>
  <param name="register" value="false"/>
  <param name="register-transport" value="udp"/>
  <param name="context" value="default"/>
</gateway>
</include>
```

Sınaq üçün

/usr/local/freeswitch/conf/dialplan/default/02_mobile_hot_outgoing_calls.xml faylında aşağıdakı dialplani yazmışıq və bu problemsız işləyəcək:

```
<include>
<extension name="cucm">
  <condition field="destination_number" expression="^([3-5]\d{6})$">
    <action application="set" data="RECORD_TITLE=Recording
${destination_number} ${caller_id_number} ${strftime(%Y-%m-%d %H:%M)}"/>
    <action application="set" data="RECORD_COPYRIGHT=(c) 2009"/>
    <action application="set" data="RECORD_SOFTWARE=FreeSwitch"/>
    <action application="set" data="RECORD_ARTIST=FreeSwitch"/>
    <action application="set" data="RECORD_COMMENT=FreeSwitch"/>
    <action application="set" data="RECORD_DATE=${strftime(%Y-%m-%d
%H:%M)}"/>
    <action application="set" data="RECORD_STEREO=true"/>
    <action application="record_session"
data="$$ ${base_dir}/recordings/archive/${strftime(%Y-%m-%d-%H-%M-
%S)}_${destination_number}_${caller_id_number}.wav"/>
    <action application="set" data="hangup_after_bridge=true"/>
    <action application="bridge" data="sofia/gateway/cucm/$1"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="cucm_regions">
  <condition field="destination_number" expression="^(0[1236][0-6][2-
5]\d{6}|994[1236][0-6][2-5]\d{6})$">
    <action application="set" data="hangup_after_bridge=true"/>
```

```

<action application="bridge" data="sofia/gateway/cucm/$1"/>
  <action application="hangup"/>
</condition>
</extension>

<extension name="cucm_mobile">
  <condition field="destination_number"
expression="^([0][57][0157]\d{7})$">
    <action application="set" data="hangup_after_bridge=true"/>
    <action application="bridge" data="sofia/gateway/cucm/$1"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="cucm_hot_3digits">
  <condition field="destination_number" expression="^([0-9][0-
9][013456789][[0-9][2-9][0-9]|102)$">
    <!-- <condition field="destination_number" expression="^([1-9]\d{2})$"> --
->
    <action application="set" data="hangup_after_bridge=true"/>
    <action application="bridge" data="sofia/gateway/cucm/$1"/>
    <action application="hangup"/>
  </condition>
</extension>

<extension name="cucm_hot_4digits">
  <condition field="destination_number" expression="^([1-9]\d{3})$">
    <action application="set" data="hangup_after_bridge=true"/>
    <action application="bridge" data="sofia/gateway/cucm/$1"/>
    <action application="hangup"/>
  </condition>
</extension>
</include>

```

CUCM 10.1 tərəf quraşdırma

- İlk işimiz tələb edilən servislərin işə salınmasıdır. CUCM interfeysinə application üçün nəzərdə tutulan istifadəçi adı ilə daxil oluruq:

Cisco Unified CM Console X +

<https://cucm.opensource.az/ccmadmin/index.jsp> Search

Cisco Unified CM Administration
For Cisco Unified Communications Solutions

Navigation Cisco Unified CM Administration Go

Cisco Unified CM Administration

Username: ccmappuser
Password: 

Login Reset

Copyright © 1999 - 2013 Cisco Systems, Inc.
All rights reserved.

This product contains cryptographic features and is subject to United States and local country laws governing import, export, transfer and use. Delivery of Cisco cryptographic products does not imply third-party authority to import, export, distribute or use encryption. Importers, exporters, distributors and users are responsible for compliance with U.S. and local country laws. By using this product you agree to comply with applicable laws and regulations. If you are unable to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at our [Export Compliance Product Report](#) web site.

For information about Cisco Unified Communications Manager please visit our [Unified Communications System Documentation](#) web site.

For Cisco Technical Support please visit our [Technical Support](#) web site.

Açılan pencərədə **Navigation** bölümündə **Cisco Unified Serviceability** seçib **Go** düyməsinə sıxırıq:

Cisco Unified CM Console X +

<https://cucm.opensource.az/ccmadmin/showHome.do> Search

Cisco Unified CM Administration
For Cisco Unified Communications Solutions

Navigation Cisco Unified Serviceability Go

ccmappuser | Search Documentation | About | Logout

System Call Routing Media Resources Advanced Features Device Application User Management Bulk Administration Help

Cisco Unified CM Administration

System version: 10.0.1.10000-24

VMware Installation: 4 vCPU Intel(R) Xeon(R) CPU E5620 @ 2.40GHz, disk 1: 100Gbytes, 8192Mbytes RAM, Partitions aligned

Last Successful Logon: Unavailable

Copyright © 1999 - 2013 Cisco Systems, Inc.
All rights reserved.

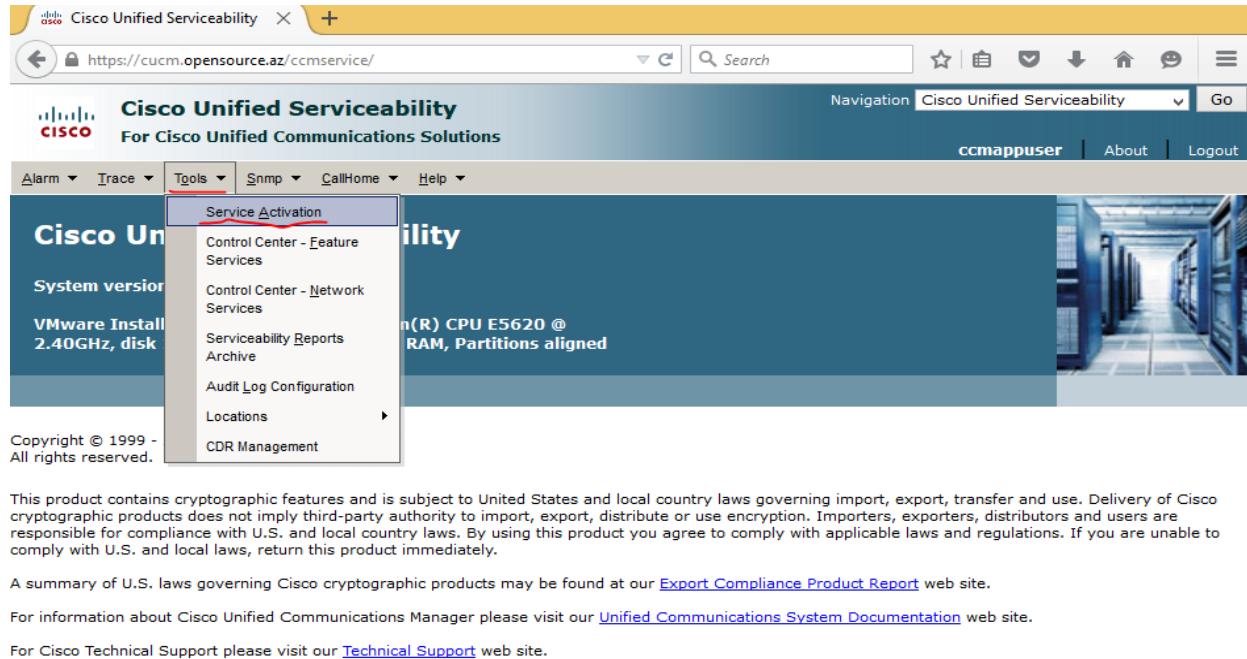
This product contains cryptographic features and is subject to United States and local country laws governing import, export, transfer and use. Delivery of Cisco cryptographic products does not imply third-party authority to import, export, distribute or use encryption. Importers, exporters, distributors and users are responsible for compliance with U.S. and local country laws. By using this product you agree to comply with applicable laws and regulations. If you are unable to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at our [Export Compliance Product Report](#) web site.

For information about Cisco Unified Communications Manager please visit our [Unified Communications System Documentation](#) web site.

For Cisco Technical Support please visit our [Technical Support](#) web site.

Tools -> Service Activation bölümünüə daxil olurug:



This screenshot shows the Cisco Unified Serviceability interface. The top navigation bar includes links for Alarm, Trace, Tools, Snmp, CallHome, Help, Navigation (Cisco Unified Serviceability), Go, and user authentication (ccmappuser, About, Logout). A sub-menu for 'Tools' is open, with 'Service Activation' highlighted. The main content area displays system information: System version 12.5(1)T1, VMware Install 2.40GHz, disk 2.40GHz, and a note about RAM, Partitions aligned. A copyright notice from 1999 is also present. Below the main content, there are links for Export Compliance Product Report, Unified Communications System Documentation, and Technical Support.

CM services-də aşağıdakı kimi seçirik:

CM Services		Activation Status
	Service Name	
<input checked="" type="checkbox"/>	Cisco CallManager	Deactivated
<input type="checkbox"/>	Cisco Unified Mobile Voice Access Service	Deactivated
<input checked="" type="checkbox"/>	Cisco IP Voice Media Streaming App	Deactivated
<input checked="" type="checkbox"/>	Cisco CTIManager	Deactivated
<input type="checkbox"/>	Cisco Extension Mobility	Deactivated
<input type="checkbox"/>	Cisco Extended Functions	Deactivated
<input type="checkbox"/>	Cisco DHCP Monitor Service	Deactivated
<input type="checkbox"/>	Cisco Intercluster Lookup Service	Deactivated
<input checked="" type="checkbox"/>	Cisco Location Bandwidth Manager	Deactivated
<input type="checkbox"/>	Cisco Directory Number Alias Sync	Deactivated
<input type="checkbox"/>	Cisco Directory Number Alias Lookup	Deactivated
<input checked="" type="checkbox"/>	Cisco Dialed Number Analyzer Server	Deactivated
<input checked="" type="checkbox"/>	Cisco Dialed Number Analyzer	Deactivated
<input checked="" type="checkbox"/>	Cisco Tftp	Deactivated

Database and Admin Services-də aşağıdakılari seçirik:

Database and Admin Services		
	Service Name	Activation Status
<input checked="" type="checkbox"/>	Cisco Bulk Provisioning Service	Deactivated
<input checked="" type="checkbox"/>	Cisco AXL Web Service	Activated
<input checked="" type="checkbox"/>	Cisco UXL Web Service	Deactivated
<input checked="" type="checkbox"/>	Cisco TAPS Service	Deactivated

Directory Services-də aşağıdakı olan seçirik:

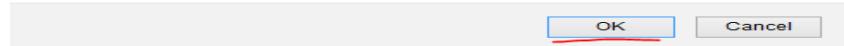
Directory Services		
	Service Name	Activation Status
<input checked="" type="checkbox"/>	Cisco DirSync	Deactivated

Sonda aşağıda sonda olan **Save** düyməsinə sıxırıq:



Serviserin aktivləşməsinin müəyyən vaxt alması haqqında xəbərdarlıq çap ediləcək. **OK** düyməsinə sıxırıq:

Activating/Deactivating services will take a while... Please wait for the page to refresh.



2. İkinci işimiz SIP profile yaradırıq:

Yenidən qayıdırıq **Cisco Unified CM Administration**-a və **Go** düyməsinə sıxırıq:



Sonra gedirik **Device** -> **Device Settings** -> **Sip Profile** bölümünüə:

System ▾ Call Routing ▾ Media Resources ▾ Advanced Features ▾ Device ▾ Application ▾ User Management ▾ Bulk Administration ▾ Help ▾

Navigation Cisco Unified CM Administration ▾ Go
ccmappuser | Search Documentation | About | Logout

Service Activation Related Links: Control Center - Feature Services ▾ Go

System version: 10.0.1.10000-24

VMware Installation: 4 vCPU Intel(R) Xeon(R) CPU E5620 @ 2.40GHz, disk 1: 100Gbytes, 8192Mbytes RAM, Partition 1: 100Gbytes

Last Successful Logon: Unavailable

Copyright © 1999 - 2013 Cisco Systems, Inc.
All rights reserved.

This product contains cryptographic features and is subject to United States and local country laws governing cryptographic products does not imply third-party authority to import, export, distribute or use encryption. Responsible for compliance with U.S. and local country laws. By using this product you agree to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at our [Export Compliance Program](#).

For information about Cisco Unified Communications Manager please visit our [Unified Communications System](#).

For Cisco Technical Support please visit our [Technical Support](#) web site.

- CTI Route Point
- Gatekeeper
- Gateway
- Phone
- Trunk
- Remote Destination
- Device Settings**

- Device Defaults
- Firmware Load Information
- Default Device Profile
- Device Profile
- Phone Button Template
- Softkey Template
- Phone Services
- SIP Profile**
- Common Device Configuration
- Common Phone Profile
- Remote Destination Profile
- Feature Control Policy
- Recording Profile
- SIP Normalization Script
- SDP Transparency Profile
- Network Access Profile
- Wireless LAN Profile
- Wireless LAN Profile Group
- Wi-Fi Hotspot Profile

Açılan pəncərədə **Find** düyməsini sıxırıq və nəticədə çıxan **Standart SIP Profile**-i şəkildə göstərildiyi kimi nüsxələyirik:

SIP Profile (1 - 5 of 5)		Rows per Page 50	
	Name	Description	Copy
<input type="checkbox"/>	Standard SIP Profile	Default SIP Profile	
<input type="checkbox"/>	Standard SIP Profile For Cisco VCS	Default SIP Profile For Cisco Video Communication Server	
<input type="checkbox"/>	Standard SIP Profile For TelePresence Conferencing	Default SIP Profile For Cisco TelePresence Conferencing	
<input type="checkbox"/>	Standard SIP Profile For TelePresence Endpoint	Default SIP Profile For Cisco TelePresence Endpoint	
<input type="checkbox"/>	Standard SIP Profile for Mobile Device	Default SIP Profile for Mobile Device	

Add New Select All Clear All Delete Selected

Açılan pəncərədə **SIP Profile Information** altında yalnız **Name** və **Description**-u şəkildəki kimi qeyd edirik(Bu öz rahatçılığımız üçündür):

SIP Profile Information

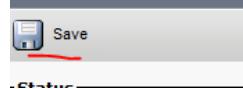
Name*	FreeSWITCH SIP Profile
Description	FreeSWITCH ile CUCM arasında SIP Trunk ucun
Default MTP Telephony Event Payload Type*	101
Early Offer for G.Clear Calls*	Disabled
User-Agent and Server header information*	Send Unified CM Version Information as User-Agent
Version in User Agent and Server Header*	Major And Minor
Dial String Interpretation*	Phone number consists of characters 0-9, *, #, and .
Confidential Access Level Headers*	Disabled
<input type="checkbox"/> Redirect by Application	
<input type="checkbox"/> Disable Early Media on 180	
<input type="checkbox"/> Outgoing T.38 INVITE include audio mline	
<input type="checkbox"/> Use Fully Qualified Domain Name in SIP Requests	
<input type="checkbox"/> Assured Services SIP conformance	
SDP Information	
SDP Session-level Bandwidth Modifier for Early Offer and Re-invites*	TIAS and AS
SDP Transparency Profile	< None >
Accept Audio Codec Preferences in Received Offer.*	Default
<input type="checkbox"/> Require SDP Inactive Exchange for Mid-Call Media Change	

Həmçinin Trunk Specific Configuration altında şəkildəki kimi, "**Early Offer support for voice and video calls (insert MTP if needed)**" seçirik:

Trunk Specific Configuration

Reroute Incoming Request to new Trunk based on*	Never
RSVP Over SIP*	Local RSVP
Resource Priority Namespace List	< None >
<input checked="" type="checkbox"/> Fall back to local RSVP	
SIP Rel1XX Options*	Disabled
Video Call Traffic Class*	Mixed
Calling Line Identification Presentation*	Default
Session Refresh Method*	Invite
<input type="checkbox"/> Enable ANAT	
<input type="checkbox"/> Deliver Conference Bridge Identifier	
<input checked="" type="checkbox"/> Early Offer support for voice and video calls (insert MTP if needed)	
<input type="checkbox"/> Allow Passthrough of Configured Line Device Caller Information	
<input type="checkbox"/> Reject Anonymous Incoming Calls	
<input type="checkbox"/> Reject Anonymous Outgoing Calls	
<input type="checkbox"/> Send ILS Learned Destination Route String	

Sonda solda yuxarıda olan **Save** düyməsinə sixırıq:



3. Üçüncü işimiz SIP trunk security profile yaratmalıyıq
 Bunun üçün **System -> Security -> SIP Trunk Security Profile** bölməsinə daxil oluruq:

The screenshot shows the Cisco Unified CM Administration interface. On the left, there's a navigation tree with 'System' selected. In the main pane, under 'Security', 'SIP Trunk Security Profile' is highlighted. A note at the top right says 'must be restarted before any changes will take affect.' The configuration form shows 'Type' set to '101' and 'Security' set to 'SIP Trunk Security Profile'.

Açılan pəncərədə **Find** düyməsini sixırıq və nəticədə alınan nüsxələrdən **Non Secure SIP Trunk Profile**-i sağda olan **Copy** işarəsi ilə nüsxələyirik:

SIP Trunk Security Profile (1 - 2 of 2)		Rows per Page 50
Find SIP Trunk Security Profile where Name begins with		<input type="button" value="Find"/> <input type="button" value="Clear Filter"/> <input type="button" value="+"/> <input type="button" value="-"/>
<input type="checkbox"/>	Name	Description
<input type="checkbox"/>	Non Secure SIP Conference Bridge	Non Secure SIP Conference Bridge
<input type="checkbox"/>	Non Secure SIP Trunk Profile	Non Secure SIP Trunk Profile authenticated by null String

Göstərilən şəkildəki kimi, qırmızı rəngdə olanları uyğun şəkildə dəyişirik və sonda **Save** düyməsini sixırıq:

SIP Trunk Security Profile Configuration

Related Links: [Back To Find/List](#) [Go](#)

Status

(i) Status: Ready

SIP Trunk Security Profile Information

Name *	FreeSWITCH Non Secure SIP Trunk Profile
Description	FreeSWITCH Trunk access SIP security profile
Device Security Mode	Non Secure
Incoming Transport Type *	TCP+UDP
Outgoing Transport Type	TCP
<input type="checkbox"/> Enable Digest Authentication	
Nonce Validity Time (mins)*	600
X.509 Subject Name	
Incoming Port*	5060
<input type="checkbox"/> Enable Application level authorization	
<input type="checkbox"/> Accept presence subscription	
<input checked="" type="checkbox"/> Accept out-of-dialog refer**	
<input checked="" type="checkbox"/> Accept unsolicited notification	
<input checked="" type="checkbox"/> Accept replaces header	
<input type="checkbox"/> Transmit security status	
<input type="checkbox"/> Allow charging header	
SIP V.150 Outbound SDP Offer Filtering*	Use Default Filter

4. **Media Resource Group** yaradırıq. Resurs qrupun altına **MTP (Media Termination Point)** əlavə edirik. Qrupu isə **Media Resource Group List** altına əlavə edirik. Məqsədimiz daxil olan səslərin içindən DTMF rəqəmləri çıxarmaqdır:

Cisco Unified CM Administration

Navigation Cisco Unified CM Administration [Go](#)

ccmappuser | Search Documentation | About | Logout

System ▾ Call Routing ▾ **Media Resources** ▾ Advanced Features ▾ Device ▾ Application ▾ User Management ▾ Bulk Administration ▾ Help ▾

Find and List Media Resources

Status

(i) 0 records found

Media Resource Groups

Find Media Resource Groups

Announcer

Conference Bridge

Media Termination Point

Music On Hold Audio Source

Fixed MOH Audio Source

Music On Hold Server

Video On Hold Server

Transcoder

Media Resource Group

Media Resource Group List

MOH Audio File Management

Mobile Voice Access

Announcement

Rows per Page 50 ▾

Please enter your search criteria using the options above.

Açılan pəncərədə **Add New** düyməsinə sıxırıq:

Status

 0 records found

Media Resource Group

Find Media Resource Group where Name begins with Find Clear Filter

No active query. Please enter your search criteria using the options above.



Resource Group için quraşdırımıları aşağıdaki kimi edirik ve sonda **Save** düyməsinə sıxırıq:

Media Resource Group Configuration

 Save Related Links: Back To Find/List Go

Status

 Status: Ready

Media Resource Group Status

Media Resource Group: New

Media Resource Group Information

Name* FS_MRES_GR Description FreeSWITCH Media Resource Group

Devices for this Group

Available Media Resources** ANN_2 CFB_2

Selected Media Resources* (MTP_2 MOH_2)

Use Multi-cast for MOH Audio (If at least one multi-cast MOH resource is available)



Ardınca **Media Resource Group List** əlavə edirik:

System Call Routing Media Resources Advanced Features Device Application User Management Bulk Administration Help

Media Resource Group

 Save  Delete

Status

 Add successful  Devices associated call processing.

Media Resource Group

Media Resource Group:

Media Resource Group

Name* FS_MRES_G Description FreeSWITCH

Media Resource Group List

MOH Audio File Management Mobile Voice Access Announcement

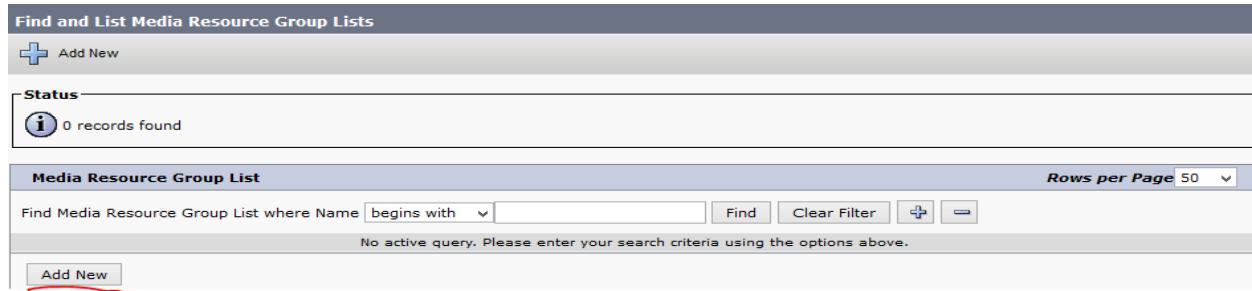
Devices for this Group

Available Media Resources** ANN_2 CFB_2

Selected Media Resources* MOH_2 (MOH) MTP_2 (MTP)

Use Multi-cast for MOH Audio (If at least one multi-cast MOH resource is available)

Sonra **Add New** düyməsinə sıxırıq ki yenisini əlavə eləsin:



Find and List Media Resource Group Lists

Status
(i) 0 records found

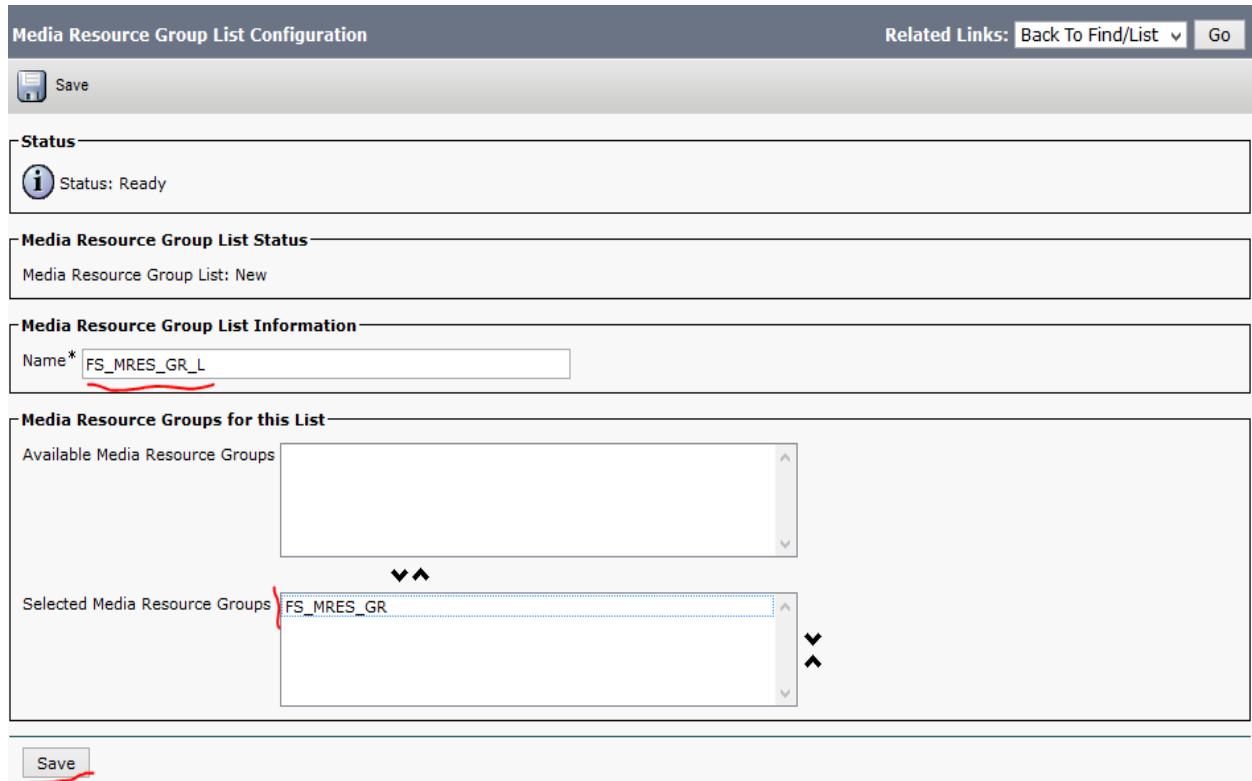
Media Resource Group List

Find Media Resource Group List where Name begins with Find Clear Filter

No active query. Please enter your search criteria using the options above.

Add New

Şəkildəki kimi, Resurs adını əlavə edirik və **Save** düyməsinə sıxırıq:



Media Resource Group List Configuration

Status
(i) Status: Ready

Media Resource Group List Status
 Media Resource Group List: New

Media Resource Group List Information
 Name* FS_MRES_GR_L

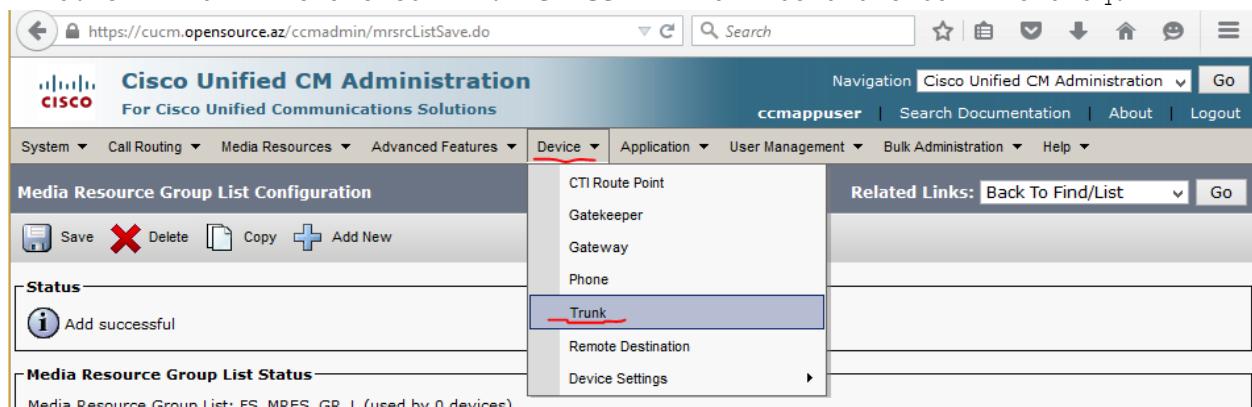
Media Resource Groups for this List

Available Media Resource Groups

Selected Media Resource Groups FS_MRES_GR

Save

5. SIP Trunk-i əlavə edirik. **Device -> Trunk** bölümünə daxil oluruq:



Cisco Unified CM Administration
 For Cisco Unified Communications Solutions

Navigation Cisco Unified CM Administration Go
 ccmappuser | Search Documentation | About | Logout

System Call Routing Media Resources Advanced Features Device Application User Management Bulk Administration Help

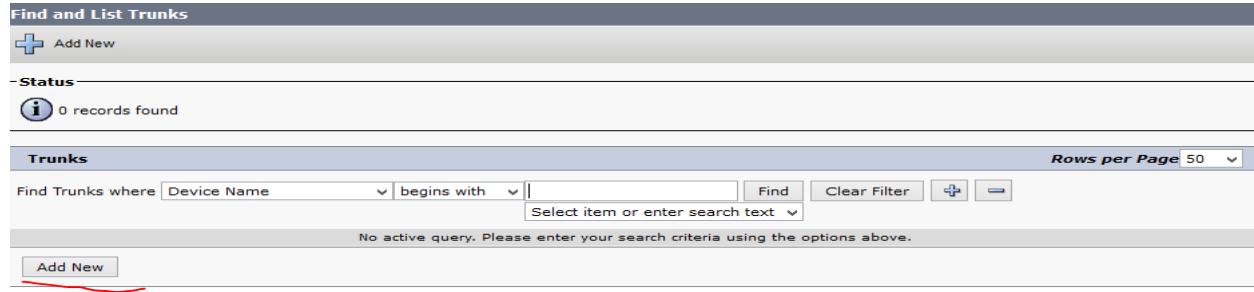
Media Resource Group List Configuration

Status
(i) Add successful

Media Resource Group List Status
 Media Resource Group List: FS_MRES_GR_1 (used by 0 devices)

CTI Route Point
 Gatekeeper
 Gateway
 Phone
Trunk
 Remote Destination
 Device Settings

Açılan pəncərədə **Add New** düyməsinə sixirinq:



Find and List Trunks

Add New

Status

(i) 0 records found

Trunks

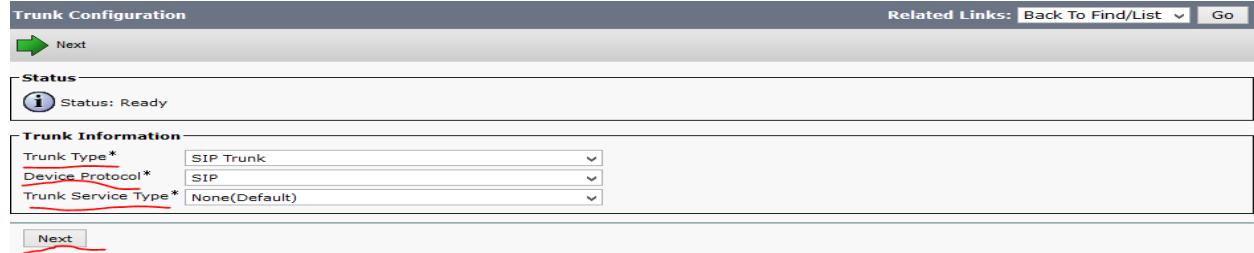
Find Trunks where Device Name begins with | Find Clear Filter + -

Select item or enter search text

No active query. Please enter your search criteria using the options above.

Add New

Trunk Type: **SIP Trunk**, Device Protocol: **SIP**, Trunk Service Type: **None** seçirik(Sonda **Next** düyməsinə sixirinq):



Trunk Configuration

Related Links: Back To Find>List Go

Next

Status

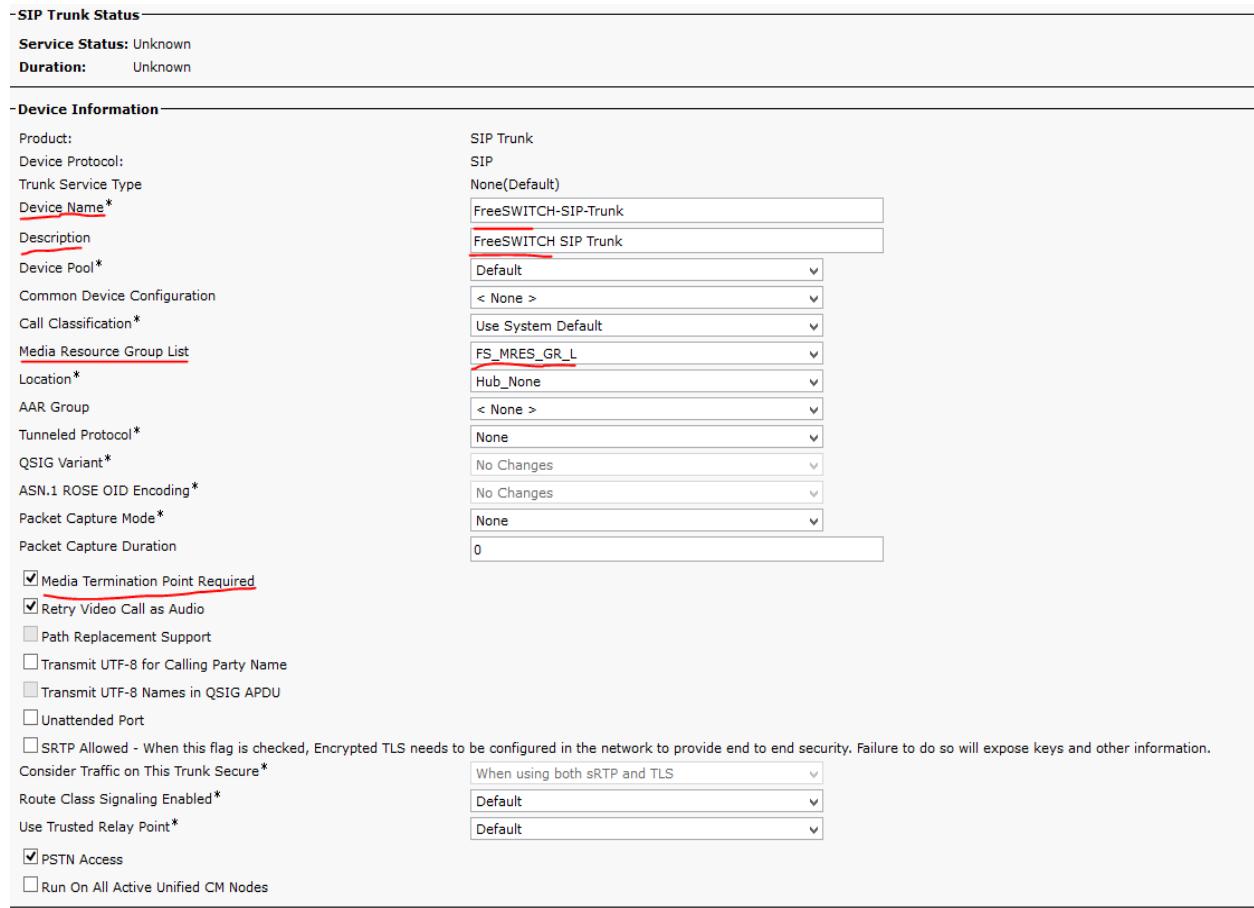
(i) Status: Ready

Trunk Information

Trunk Type* SIP Trunk
Device Protocol* SIP
Trunk Service Type* None(Default)

Next

Device Information bölümündə aşağıda şəkildəki kimi qırmızı işaretlənləri seçirik:



-SIP Trunk Status

Service Status: Unknown
Duration: Unknown

-Device Information

Product: SIP Trunk
Device Protocol: SIP
Trunk Service Type: None(Default)

Device Name* FreeSWITCH-SIP-Trunk
Description FreeSWITCH SIP Trunk
Device Pool* Default
Common Device Configuration
Call Classification*
Media Resource Group List
Location* FS_MRES_GR_L
AAR Group
Tunnled Protocol*
QSIG Variant*
ASN.1 ROSE OID Encoding*
Packet Capture Mode*
Packet Capture Duration: 0
 Media Termination Point Required
 Retry Video Call as Audio
 Path Replacement Support
 Transmit UTF-8 for Calling Party Name
 Transmit UTF-8 Names in QSIG APDU
 Unattended Port
 SRTP Allowed - When this flag is checked, Encrypted TLS needs to be configured in the network to provide end to end security. Failure to do so will expose keys and other information.
Consider Traffic on This Trunk Secure* When using both s RTP and TLS
Route Class Signaling Enabled* Default
Use Trusted Relay Point* Default
 PSTN Access
 Run On All Active Unified CM Nodes

SIP Information bölümündə **Destination Address**-də FreeSWITCH serverimizin IP ünvanı yada DNS adı, **Destination Port**-da FreeSWITCH serverimizin port rəqəmi, **SIP Trunk Security Profile**-da yaratdığımız **FreeSWITCH Non Secure profile**, **SIP Profile**-da isə **FreeSWITCH SIP Profile** seçirik(Şəkildə göstərildiyi kimi):

SIP Information

Destination

Destination Address is an SRV

Destination Address	Destination Address IPv6	Destination Port	Status	Status Reason	Duration
1* frfs.opensource.az		5080	N/A	N/A	N/A

MTP Preferred Originating Codec* 711ulaw

BLF Presence Group* Standard Presence group

SIP Trunk Security Profile* **FreeSWITCH Non Secure SIP Trunk Profile**

Rerouting Calling Search Space < None >

Out-Of-Dialog Refer Calling Search Space < None >

SUBSCRIBE Calling Search Space < None >

SIP Profile* **FreeSWITCH SIP Profile** [View Details](#)

DTMF Signaling Method* No Preference

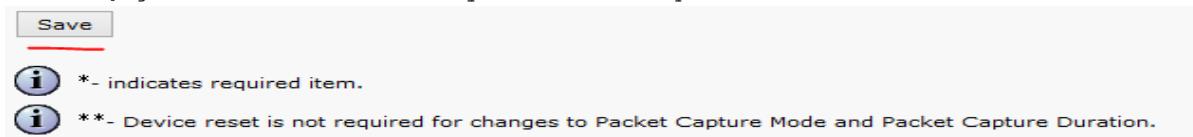
Normalization Script

Normalization Script < None >

Enable Trace

Parameter Name	Parameter Value
1	

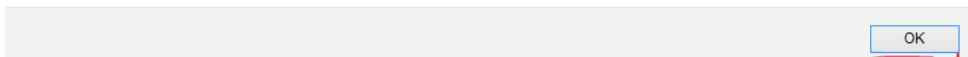
Sonda aşağıda solda olan **Save** düyməsinə sıxırıq:



Çıxan pəncərədə **OK** düyməsinə sıxırıq.

Qeyd: SIP Trunk-da hər hansıa bir dəyişiklik etdiğdə, onun işləməsi üçün mütləq **reset** eləmək lazımdır:

The configuration changes will not take effect on the trunk until a reset is performed. Use the Reset button or Job Scheduler to execute the reset.



Mütləq hər dəyişiklikdə **Reset** edirik:

Trunk Configuration

Save Delete Reset Add New

Status

Update successful

SIP Trunk Status

Service Status: Unknown
Duration: Unknown

Device Information

Product:	SIP Trunk
Device Protocol:	SIP
Trunk Service Type	None(Default)
Device Name*	FreeSWITCH-SIP-Trunk
Description	FreeSWITCH SIP Trunk

Reset və Close

<https://cucm.opensource.az/ccmadmin/reset.do?pkid=c6f82027-0d52-bdc5-a933-5bc183c8b67e&type=undefined>

Device Reset

Reset Restart

Status

Status: Ready

Reset Information

Selected Device: FreeSWITCH-SIP-Trunk (FreeSWITCH SIP Trunk; SIP Trunk)
 If a device is not registered with Cisco Unified Communications Manager, you cannot reset or restart it. If a device is registered, to restart a device without shutting it down, click the **Restart** button. To shut down a device and bring it back up, click the **Reset** button. To return to the previous window without resetting/restarting the device, click **Close**.

Note:
 Resetting a gateway/trunk/media devices **drops** any calls in progress that are using that gateway/trunk/media devices. Restarting a gateway/media devices tries to preserve the calls in progress that are using that gateway/media devices, if possible. Other devices wait until calls are complete before restarting or resetting. Resetting/restarting a H323 device does not physically reset/restart the hardware; it only reinitializes the configuration loaded by Cisco Unified Communications Manager.

6. CUCM-dən FreeSWITCH-ə DialPlan-in hazırlanması. Məqsədimiz odur ki, **1***** rəqəmlə başlayan nömrələr geriyə FreeSWITCH-ə CUCM-dən zəng edə bilsin. Bunun üçün **Call Routing -> Route/Hunt -> Route Pattern** bölümünə daxil oluruq:

Cisco Unified CM Administration
For Cisco Unified Communications Solutions

System ▾ Call Routing ▾ Media Resources ▾ Advanced Features ▾ Device ▾ Application ▾ User Management ▾ Bulk Administration ▾ Help ▾

Find and

- AAR Group
- Dial Rules
- Route Filter
- Route/Hunt**
 - Route Group
 - Local Route Group Names
 - Route List
 - Route Pattern**
 - Line Group
 - Hunt List
 - Hunt Pilot
- SIP Route Pattern
- Intercom
- Class of Control
- Client Matter Codes
- Forced Authorization Codes
- Translation Pattern
- Call Park
- Directed Call Park
- Call Pickup Group
- Directory Number
- Dial Plan Installer
- Meet-Me Number/Pattern
- Route Plan Report
- Transformation
- Mobility
- Logical Partition Policy Configuration
- External Call Control Profile
- HTTP Profile
- Call Control Discovery
- Global Dial Plan Replication

Status

Trunks

Add New

Delete Selected **Reset Selected**

Route Pattern

Search text Find Clear Filter

Calling Search Space Device Pool **Default**

Selected Reset Selected

Add New düyməsini sıxaraq yenisini yaradırıq:

Find and List Route Patterns

Add New

Route Patterns

Find Route Patterns where Pattern begins with Find Clear Filter

No results

Add New

Sonra Route Pattern RegEX daxil edirik(Yəni **1XXX**).Bu CUCM-dən FreeSWITCH-ə gedən zənglər üçündür. **Gateway/Route List-i FreeSWITCH-SIP-Trunk** seçirik və **Save** düyməsinə sıxırıq:

Route Pattern Configuration

Save Delete Copy

Status: Status: Ready

Pattern Definition

Route Pattern*	1XXX
Route Partition	< None >
Description	
Numbering Plan	-- Not Selected --
Route Filter	< None >
MLPP Precedence*	Default
<input type="checkbox"/> Apply Call Blocking Percentage	
Resource Priority Namespace Network Domain	< None >
Route Class*	Default
Gateway/Route List*	FreeSWITCH-SIP-Trunk
Route Option	<input checked="" type="radio"/> Route this pattern <input type="radio"/> Block this pattern <input type="checkbox"/> No Error
Call Classification*	OffNet
External Call Control Profile	< None >
<input type="checkbox"/> Allow Device Override <input checked="" type="checkbox"/> Provide Outside Dial Tone <input type="checkbox"/> Allow Overlap Sending <input type="checkbox"/> Urgent Priority	
<input type="checkbox"/> Require Forced Authorization Code	
Authorization Level*	0
<input type="checkbox"/> Require Client Matter Code	

Calling Party Transformations

<input type="checkbox"/> Use Calling Party's External Phone Number Mask	
Calling Party Transform Mask	
Prefix Digits (Outgoing Calls)	
Calling Line ID Presentation*	Default
Calling Name Presentation*	Default
Calling Party Number Type*	Cisco CallManager
Calling Party Numbering Plan*	Cisco CallManager

Connected Party Transformations

Connected Line ID Presentation*	Default
Connected Name Presentation*	Default

Called Party Transformations

Discard Digits	< None >
Called Party Transform Mask	
Prefix Digits (Outgoing Calls)	
Called Party Number Type*	Cisco CallManager
Called Party Numbering Plan*	Cisco CallManager

ISDN Network-Specific Facilities Information Element

Network Service Protocol	-- Not Selected --	
Carrier Identification Code		
Network Service	Service Parameter Name	Service Parameter Value
-- Not Selected --	< Not Exist >	

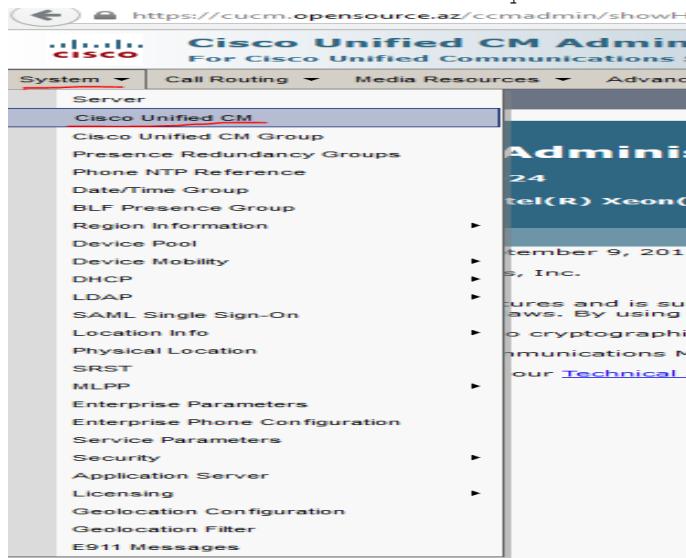
Save Delete Add New

Növbəti gələn hər iki pəncərədə **OK** düyməsinə sixirinq:

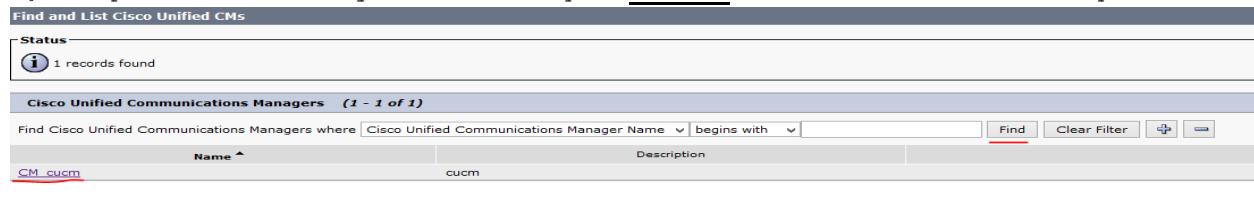
The Authorization Code will not be activated.
 Press OK if you want to proceed and activate it at a later time.
 Press Cancel and check the Force Authorization Code checkbox if you want to activate it now.

Any update to this Route Pattern automatically resets the associated gateway or Route List
 Prevent this page from creating additional dialogs

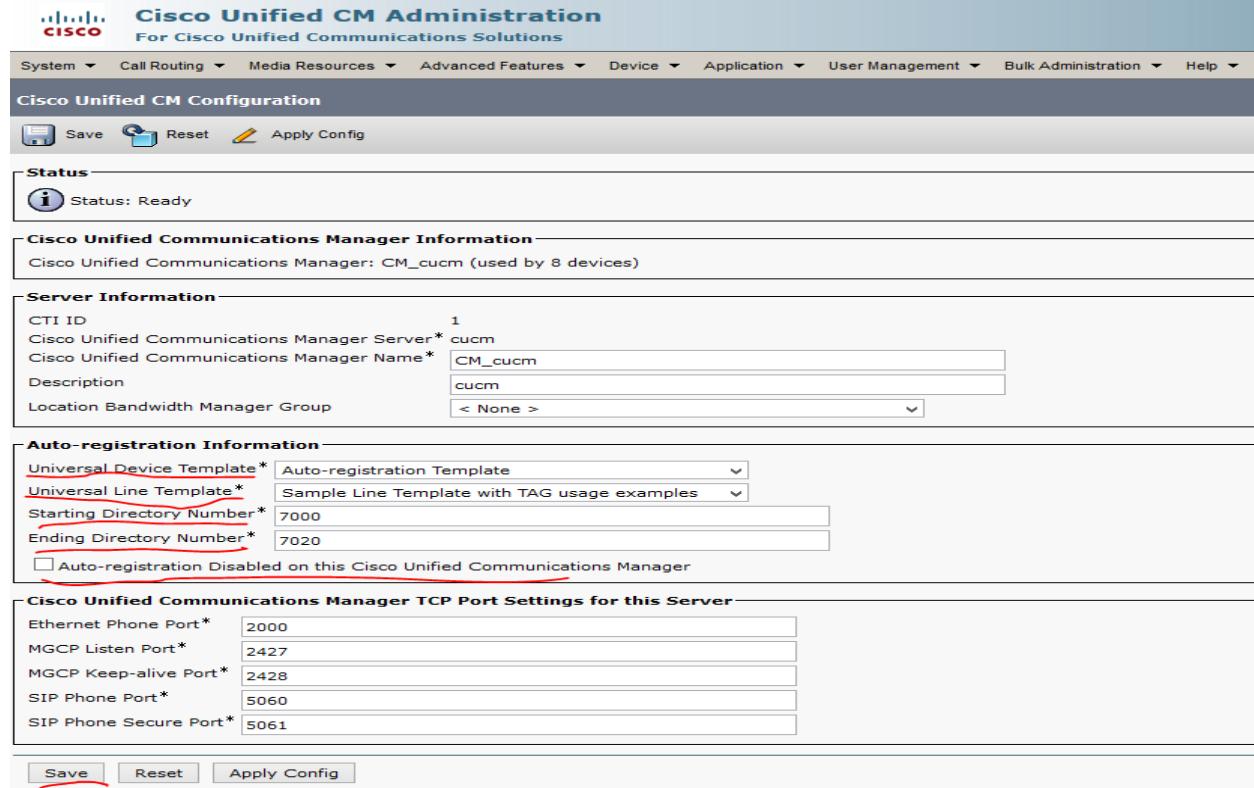
7. Telefonların avtomatik qeydiyyatı üçün lazımi işlər görürük. Bunun üçün **System**
 -> **Cisco Unified CM** bölməsinə daxil oluruz:



Açılan pəncərədə **Find** düyməsini sıxırıq və **CM_cucm** mövcud olana daxil oluruz:



Avtomatik qosulan IP communicatorlara bu **7000-7020** aralıqdan nömrələr verilir.



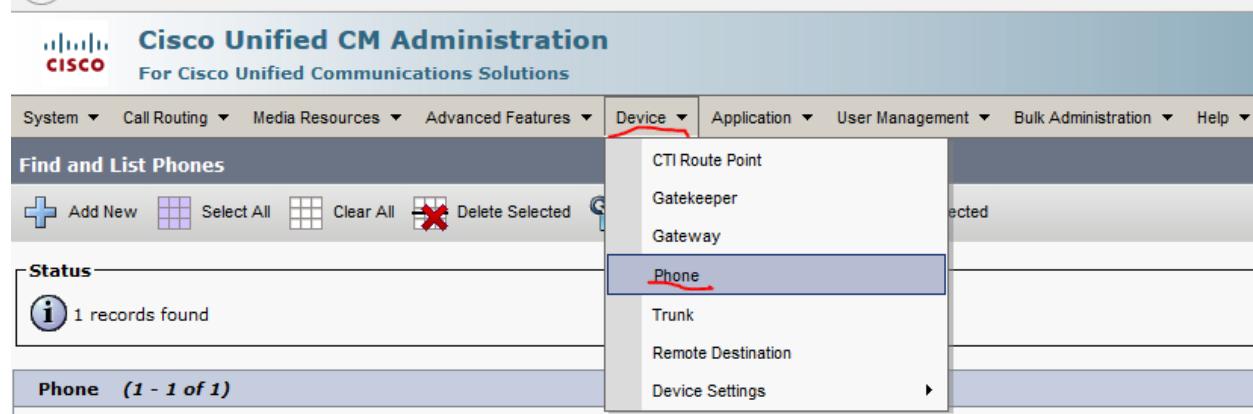
Qeyd: Telefonlar qeydiyyatdan keçikdən sonra, mütləq aşağıdakı şəkildəki kimi Auto-Registration-dan seçimi silirik.



Auto-registration Disabled on this Cisco Unified Communications Manager

Artıq IP Communicator-u qeydiyyata aldıqdan sonra, CUCM interfeysində **Device -> Phone** bölümünə daxil oluruq və qeydiyyatdan keçən telefona baxırıq:

(  <https://cucm.opensource.az/ccmadmin/phoneFindList.do?lookup=false&multiple=true&recCnt=1&colCnt=17>)



Cisco Unified CM Administration
For Cisco Unified Communications Solutions

System ▾ Call Routing ▾ Media Resources ▾ Advanced Features ▾ Device ▾ Application ▾ User Management ▾ Bulk Administration ▾ Help ▾

Find and List Phones

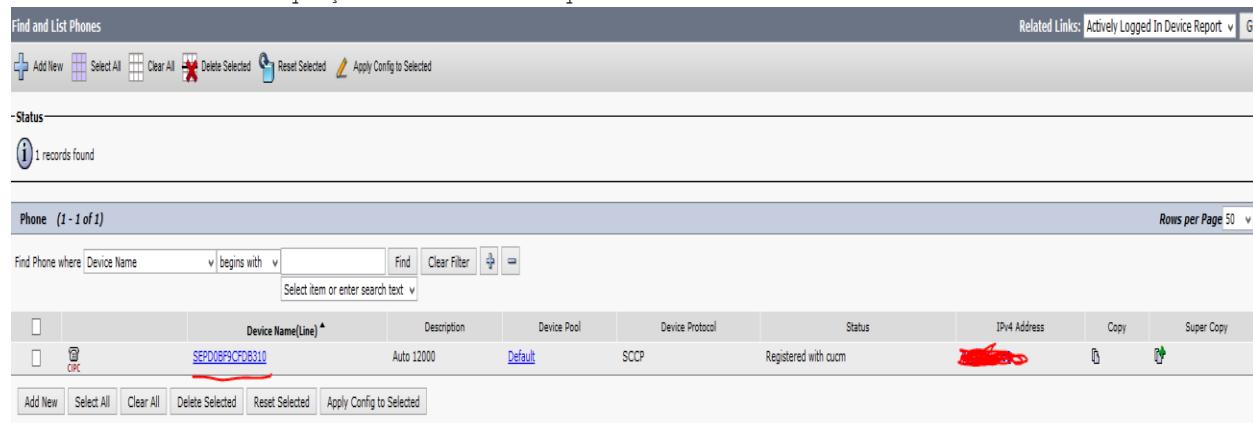
Add New Select All Clear All Delete Selected

Status
1 records found

Phone (1 - 1 of 1)

CTI Route Point
Gatekeeper
Gateway
Phone
Trunk
Remote Destination
Device Settings

Açılan pəncərədə aşağıdakı kimi qeydiyyata alınmış telefonu görürük və telefonun device adına sixaraq içini daxil oluruq:



Find and List Phones

Related Links: Actively Logged In Device Report Go

Add New Select All Clear All Delete Selected Reset Selected Apply Config to Selected

Status
1 records found

Phone (1 - 1 of 1)

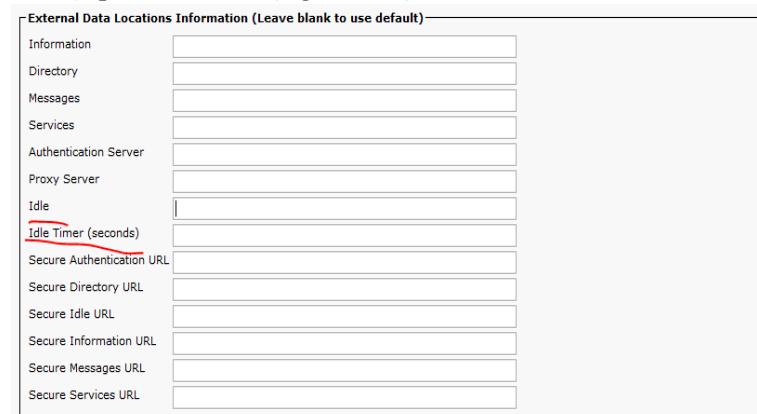
Find Phone where Device Name begins with Find Clear Filter

Select item or enter search text

	Device Name(Line)*	Description	Device Pool	Device Protocol	Status	IPv4 Address	Copy	Super Copy
	 SEP00BF9CFD8310	Auto 12000	Default	SCCP	Registered with ccm			

Add New Select All Clear All Delete Selected Reset Selected Apply Config to Selected

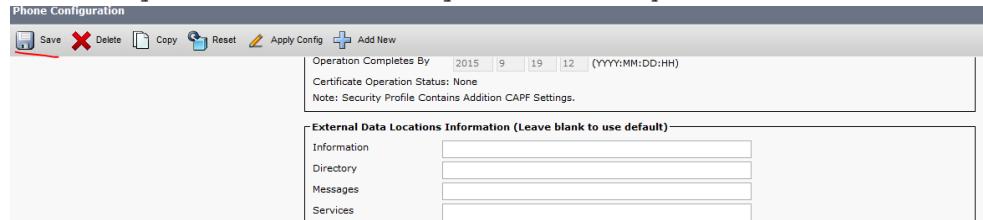
Ardınca **External Data Locations Information (Leave blank to use default)** bölümünü seçib hər şeyi silirik(asağıdakı şəkildəki kimi):



External Data Locations Information (Leave blank to use default)

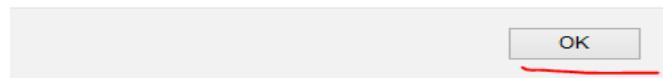
Information	<input type="text"/>
Directory	<input type="text"/>
Messages	<input type="text"/>
Services	<input type="text"/>
Authentication Server	<input type="text"/>
Proxy Server	<input type="text"/>
Idle	<input type="text"/>
Idle Timer (seconds)	<input type="text"/>
Secure Authentication URL	<input type="text"/>
Secure Directory URL	<input type="text"/>
Secure Idle URL	<input type="text"/>
Secure Information URL	<input type="text"/>
Secure Messages URL	<input type="text"/>
Secure Services URL	<input type="text"/>

Solda ən yuxarıda olan **Save** düyməsinə sixiriq.

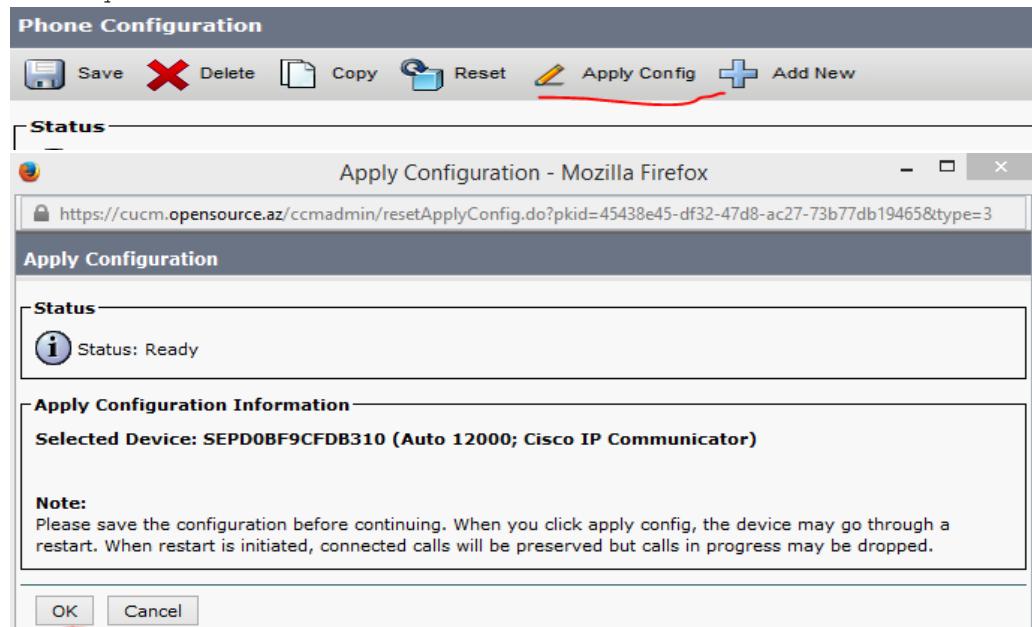


Çıxan Warning-i OK sixaraq qəbul edirik:

Click on the **Apply Config** button to have the changes take effect.



Və sonda, **Apply Config** düyməsinə sixiriq və yeni açılan pəncərədə də OK düyməsinə sixiriq:



Gateway olmadan zənglərin edilməsi

Bəzi hallarda gateway-in istifadə edilməsi tələb edilmir. Misal olaraq deyə bilərik ki, bütün servislərin digest qeydiyyata ehtiyacı olmur. Bunun misali FreeSWITCH dünya konfrans serveridir. Əslində dialplan konfransa zəng üçün özündə **9888** nömrəsini susmaya görə saxlayır(FreeSWITCH conference serverdə fərqli konfrans otaqları mövcuddur). Gəlin genişlənməyə baxaq.

/usr/local/freeswitch/conf/dialplan/default.xml faylını açın və **freeswitch_public_conf_via_sip** sətirini tapın. Bridge olan sətirə diqqət yetirin:

```
<action application="bridge"
data="sofia/${use_profile}/$1@conference.freeswitch.org"/>
```

Öncəki misalda **\${use_profile}** mənasi daxildə təyin edilir(**/usr/local/freeswitch/conf/vars.xml** faylında təyin edilir). İstifadəçi **9888** nömrəsinə zəng etdikdə, o kənara adətən aşağıdakı qaydada çıxacaq:

sofia/external/888@conference.freeswitch.org

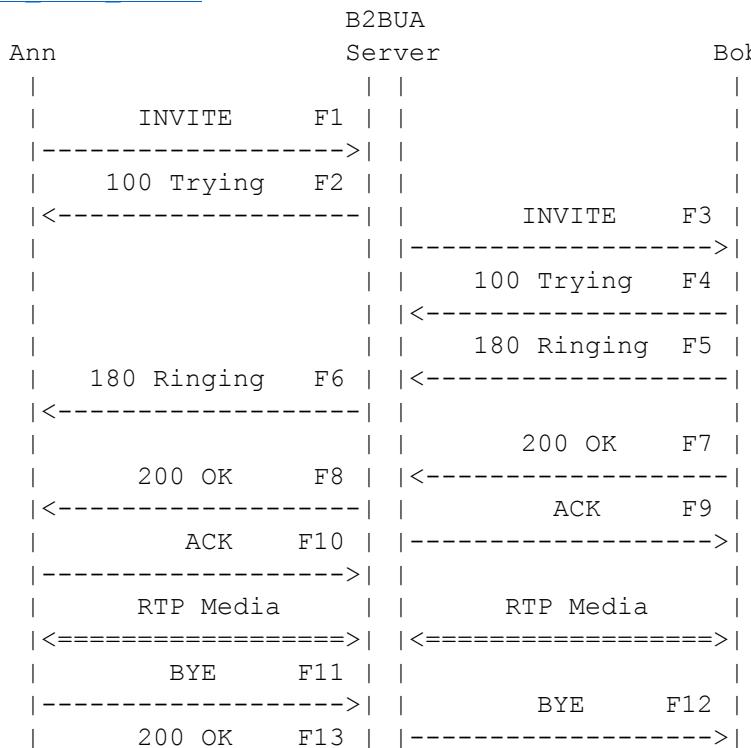
Nəzər yetirin ki, gateway haqqında heç bir xəbərdarlıq yoxdur. Bunun əvəzinə FreeSWITCH zəngi external SIP profile üzərinə ötürür. Digər sözlərlə desək FreeSWITCH serverimiz **9888** nömrəsinə gedən zəngləri **conference.freeswitch.org** üstünə qeydiyyat olmadan göndərir. Bu mümkündür ona görə ki, **conference.freeswitch.org** serveri daxil olan zənglər üçün qeydiyyat tələb eləmir(Bu həmin yerdir ki, gateway işə düşür - əgər hədəf serveri razılaşma olaraq hansısa səbəb ötürürsə onda gateway razılaşmaya istifadəçi adı və şifrə qeydiyyat verilənləri ilə cavab verəcək). Bütün SIP providerlər istifadəçi adı və şifrə ilə olan(digest authentication) qeydiyyat tələb eləmir. E�ə olanları da var ki, qeydiyyat IP ilə aparılır. Bu halda sizin Gateway yaratmağınızda ehtiyac qalmır.

SIP profillərin və istifadəçi agentlərin qısaca müzakirə edilməsi

SIP və istifadəçi qovluğu haqqında olan diskussiyamızı bitirdikdən əvvəl danışmalıyıq ki, bəzi istifadəçilər var ki, kiçik yararlı imkanları tez tapırlar: **SIP profiles**. Daha sərt olaraq desək, FreeSWITCH-də olan SIP profile-i **user agent**-dir. Təcrübədə olan dillə desək bu o deməkdir ki, hər bir SIP profile-i seçilmiş IP ünvanı və portda qulaq asır. Nüsxə quraşdırma faylında **internal** profile-i **5060**-ci və **external** profile isə **5080**-ci portda qulaq asır. Bu profillər nəyin ki, qulaq asır hətta cavab belə verə bilər. Misal üçün, telefon SIP **REGISTER** paketini FreeSWITCH(5060-ci porta)-ə ötürürsə, internal profile-i qeydiyyat müraciətinə həmçinin qulaq asa bilər. Pofile-ların özünün necə aparılmasını

/usr/local/freeswitch/conf/sip_profiles/ qovluğunda olan fayllar təyin edir. Bu profile-larda olan əksər parametrlər FreeSWITCH-in fərqli ssenarilərdə özlərinin necə aparılmasını təyin edir. Əksər hallarda susmaya görə olan mənalar düzgün olur və işləyir. Digər hallarda isə, fərqli VoIP telefonlarının öz xüsusiyyətlərinə görə fərqli tələblər ola bilər və siz problemləri özünüz araşdırmağınızın.

Sonda nəzərə alın ki, profile adlarını heç bir zaman **internal** və ya **external** adlandırmayıñ çünkü, bu böyük qarışığığa gətirib çıxara bilər. Hər bir SIP profile-i adı istifadəçi agentidir hansı ki, spesifik tələblər üçün optimallaşdırılmışdır. **internal** profile-i optimallaşdırılmışdır ki, telefonları qeydiyyatlarını və qeydiyyatdan keçmiş telefonlar arasında zəngləri emal etsin(Hətta daxili LAN-da olmayan telefonlar olarsa da belə). **external profile**-i çıkış gateway qoşulmaları üçün və bəzi **NAT(traversal)** aşılması üçün optimallaşdırılmışdır. İstifadəçi agentlərinin daha dərin müzakirə edilməsi və **back-to-back user agent (B2BUA)** konsepsiyasının daha dərindən başa düşülməsi üçün, http://en.wikipedia.org/wiki/Back-to-back_user_agent linkinə baxın. Aşağıdakı şəkil gözəl misaldır:



```
|<-----| | 200 OK F14 |  
| |<-----| |  
| | | |
```

Notice

Bu başlıqda biz aşağıdakıları müzakirə etdik:

- FreeSWITCH istifadəçilər-i neçə qovluqda cəmləşdirir
- FreeSWITCH-in VoIP protokolunu, SIP-i istifadə edərək necə digər istifadəçilər və dünyaya qoşulmasını anladıq
- SIP email kimidir hansı ki, istifadəçi adı və domain adı var
- Fərqli istifadəçi funksiyalarının istifadə edilməsi, misal üçün voicemail
- Yeni istifadəçinin əlavə edilməsi və Dialplan-da dəyişiklik
- Dünya və kənar Gateway-lərlə əlaqənin qurulması
- İstifadəçi agentləri və SIP profilləri

Bu başlıqda biz **default** XML Dialplan-da ediləcək dəyişiklikləri və XML istifadəçi qovluğunda yeni istifadəçilərin əlavə edilməsində bəzi xırda dəyişikliklərin edilməsini öyrəndik. Artıq bizdə bu dəyişikliklərin necə işləməsi haqqında anlayış olduğu üçün davamlı olaraq onlara əsaslanacaq. Növbəti başlığımızda FreeSWITCH üzərində olan XML Dialplan modulunun işləməsini ətraflı şəkildə öyrənəcəyik çünki, o FreeSWITCH-də olan əsas yönləndirmə moduludur.

5-ci başlıq

XML Dialplan-ın başa düşülməsi

FreeSWITCH yüklenməsində Dialplan çox vacib hissələrdən biridir. Həqiqətən də istənilən PBX-in Dialplan-ı olmalıdır. Bəzi hallarda buna zənglərin səliqəli yönləndirilməsi üçün nömrələmə planı da deyilir. Misal üçün istifadəçi telefonu qaldırıb 1000 nömrəsinə yığıldıqda, sistem bu zəng ilə nə deyəcəyini bilməlidir. Example dialplani artıq **1000 ID** ilə qeydiyyatdan keçmiş telefona zəng edən tərəfin necə qoşulmasını bilir artıq. Ancaq Dialplan zəng edən və zəng edilən abonentlərin birləşməsindən qat-qat artıq çox iş görə bilir. Dialplan özündə zəngin necə edilməsi və özünü necə aparması haqqında instruksiya təşkil edir.

Öncəki başlıqda biz Dialplan-da kiçik dəyişikliklər etdik. Bu başlıqda biz, öncəki əsaslara arxalanaraq zəngin yönləndirilməsi və idarə edilməsini aşağıdakı başlıqlarla müzakirə edəcəyik:

- FreeSWITCH XML Dialplan elementləri
- Zəng ayaqları və kanal dəyişənləri
- Yeni extension(genişlənmə)-ın yaradılması
- Vacib Dialplan programları
- Dialstring formatları

FreeSWITCH XML Dialplan elementləri

FreeSWITCH-in XML Dialplan nüsxəsi XML Dialplan konsepsiyasını öyrənmək üçün, əla ünvandır. Quraşdırma `/usr/local/freeswitch/conf/dialplan/` ünvanında yerləşən 3 əsas fayl və iki qovluqdan ibarətdir:

- **default.xml**: Bu faylda FreeSWITCH-in əsas Dialplan quraşdirmaları olur.
- **public.xml**: Bu fayl FreeSWITCH-ə digər ərazidən gələn zənglərin emal edilməsi üçün quraşdirmaları özündə təşkil edir.
- **features.xml**: Bu faylda spesifik zəng bacarıqlarının emal edilməsi üçün xüsusi kontekst olur.
- **default/**: Bu qovluqda olan fayllar default kontekst tərəfindən **include**(əlavə) edilmişdir.
- **public/**: Bu qovluqda olan fayllar public kontekst tərəfindən **include**(əlavə) edilmişdir.

Nüsxə XML quraşdırmasının zənglərin yönləndirilməsi üçün çoxlu instruksiyaları mövcuddur hansı ki, Dialplan-da başlanğıc qurulma bloklarını öyrədir: **contexts**(kontekstler), **extensions**(genişlənmələr), **conditions**(şərtlər) və **actions**(mənimsədiləcək işlər). Context bir və ya bir neçə genişlənmənin logik qruplaşmasıdır. Genişlənmə özündə riayət ediləcək bir və ya bir neçə şərt saxlayır. Şərtlərdə actions(yerinə yetiriləcək işlər) olur hansı ki, şərtə əsaslanaraq zəng zamanı yerinə yetiriləcək ya da yetirilməyəcək. Bu bloklar haqqında ətraflı danışmadan əvvəl 3-cü başlıqda olan Nüsxə quraşdirmalarının *sınaqdan keçirilməsinə* baxın.

Kontekstlər

Kontekstlər genişlənmələr üçün logik qruplardır. Kontekstlər haqqında dialplanın bölmələri kimi düşünün. Hər bir bölmə xüsusi məqsədə sahibdir və özündə yalnız bu məqsədlə əlaqəli olan genişlənməni saxlayır. Bircə belə məqsəd genişlənmələri bir-birlərindən izolyasiya etməkdən ibarət olur(multitenant-a oxşar misaldır). FreeSWITCH geniş imkana sahibdir ki, eyni zamanda öz konteksti sahib olan bir neçə kirayəçiyə xidmət göstərsin və nömrə konfliktinin olması imkanını tam azaldır. Misal üçün hər bir tenant kirayəçi "operator üçün 0 nömrəsinə zəng" işini görə bilir. Bir tenant-da olan istifadəçilər öz frontdesk genişlənmələrinə çatmaq üçün 0 nömrəsinə zəng edə bilərlər və həmçinin digər tenant-da olan istifadəçilər-də özlərinə aid olan frontdesk genişlənməsilə əlaqələnib tamamilə fərqli genişlənmə ilə əlaqələnmək üçün eyni zamanda 0 nömrəsinə zəng edə bilərlər. Kontekstlər üçün digər məqsəd - təhlükəsizlikdir. Kontext üzərindən ötürülən telefon zəngləri yalnız təyin edilmiş müəyyən zənglərə yetkili olurlar(Misal üçün uzaq məsafəli, dünyaya edilən zənglər ya da digər çox hissəli səsli konfrans zalları resursları ola bilər). Təyin edilə biləcək istənilən hər hansıa rəqəmə limit yoxdur.

Dialplan XML nüsxəsi 3 fərqli konteksti təyin edir hansılara ki, biz aşağıdakı baxırıq.

Default

default konteksti sistemdə qeydiyyatdan keçmiş istifadəçilərin bütün genişlənmə təyinatlarını özündə saxlayır. Biz **1100** genişlənməsini `/usr/local/freeswitch/conf/dialplan/default.xml` faylinə əlavə etməklə əslində **default** kontekstdə dəyişiklik etmiş olduq. Dialplan XML nüsxələrinin əksər imkanları bu faylda təyin edilmişdir.

Public

`public` context-i təhlükəsizlik məqsədi ilə istifadə ediləcək düzgün misaldır. FreeSWITCH serverə gələn bütün qeydiyyatdan keçməyən zənglər `public` kontekst tərəfindən emal ediləcək. "**public**" adı "məqsədinə görə etibarsız"-dır. Susmaya görə FreeSWITCH qeydiyyatdan keçməmiş istifadəçinin sistemdə nə etməsi səbəbindən hədsiz sərt və narahatdır. Ümumi danişsaq, `public` konteksti istifadə edilir ki, daxil olan **DID(Direct Inward Dial)** telefon nömrələrinin spesifik daxili genişlənməyə yönəldirsin(ətraflı məlumat üçün `/usr/local/freeswitch/conf/dialplan/public/00_inbound_did.xml`). Siz `public` konteksti istifadə edərək yalnız öz sisteminizə tələb edilən zənglərin buraxılması üçün istifadə edə bilərsiniz.

Features

`features` konteksti - qruplaşdırılmış genişlənmələr funksiyaları üçün yaxşı misaldır. Bu genişlənmələr həmçinin asanlıqla `default` kontekst-də əlavə edilə bilər ancaq, onların öz kontekslərində saxlanılması işin təşkilatçılığına kömək edir. `features` kontekstində təyin edilən genişlənmələr əksər hallarda son nöqtələr yox, daha çox köməkçi genişlənmələrdir hansı ki, funksiyani yerinə yetirir və sonra zəng edəni digər tərəfə ötürür. Bunun misali - **please_hold** genişlənməsidir hansı ki, musiqini gözləmədə olan tərəf üçün musiqi işə salaraq deyir ki, "Xahiş olunur zənginizi qoşan müddətədək gözləyəsiniz(Please hold while I connect your call)" və sonra zəng edəni son genişlənməyə yönəldirir.

Extensions

Extension(genislənmə) anlayışı hərdən anlaşılmazlığı gətirib çıxara bilər. Ən-ənəvi PBX mühitində `extension`(genislənmə) telefon sisteminə qoşulmuş olan adı telefondur ki, **3** və ya **4** rəqəmli olur. FreeSWITCH-də extension - faktiki olaraq instruksiyalar ardıcılılığıdır hansı ki, zənglə nə ediləcəyini təyin edir. Bu 3 və ya 4 rəqəmli nömrələrə edilən zənglər qədər sadə ola bilər. Faktiki olaraq bu bizim 4-cü başlıqda *SIP və istifadəçi qovluğu* başlığında *istifadəçi əlavə edilməsi* seksiyasında olan gördüyüümüz işlərdir hansı ki, uyğun olan istifadəciyə ona zəngin edilə bilməsi üçün yeni genislənmə əlavə etdik.

Genislənmənin təyin edilməsi `<extension>` tag-la açılır və `</extension>` tag-la bağlanır. Bütün action(işlər) və anti-action(qarşı işlər) bu genislənmə tag-ların arasında təyin edilir. Genislənmələr iki əlavə atributa sahibdir: `name` və `continue`. `name` atributu sizin Dialplan-ı oxunula bilən saxlamaq üçün istifadə edilir. Onlarla `<extension>` tag-a adsız olmadan baxsaz Dialplan çox çətin görünə bilər. `continue` atributu isə uyğun genislənməni tapan kimi, Dialplan-da sintaksis analizini təyin edir. Əgər bu genislənmə üçün bütün şərtlər üst-üstə düşüb `true` cavab verirsə, extension uyğun olduğunu bildirir. Susmaya görə, genislənmələr ilk Dialplan uyğunluğundan sonra davam etmirlər. `continue="true"` təyin edilməsi sintaksis analizatorunu məcbur edir ki, uyğun ola biləcək çoxlu genislənmələr axtarsın(Bu başlığın Dialplan emalının necə işləməsi seksiyasında baxın).

Conditions (şərtlər)

Şərtlər tələbləri yerinə yetirir ki, genişlənmədə sadalanan tədbirləri tətbiq eləsin. Misal üçün, sizdə şərt "999-a zəng edilərsə" ola bilər və sonra bu genişlənmə daxilində olan blok yerinə yetiriləcək. Nəzərinizdə saxlayın ki, şərt yalnız zəng edilən tərəfin nömrəsinə görə məhdudlaşdır. Şərtlər həmçinin tarixə, vaxta, caller ID-yə, göndərən tərəfin IP ünvanına və hətta kanal dəyişənlərinə əsaslanı bilər. Dialplan faylları nüsxəsinə baxmaqla siz əksərən aşağıdakına uyğun olan şərtləri görə bilərsiniz:

```
<condition field="destination_number" expression="^(1234)$">
```

destination_number sütunu sözsüz ki, daha da sınaqdan keçmiş və yayılmışdır ona görə də, vaxtimızın çox hissəsini rəqəmlərə əsaslanan istifadəçi tərəfindən gələn zəngə yönəldirməliyik.

Əksər istifadəçilərdə yalnız birçə şərt olur:

```
<extension name="Simple example">
<condition field="destination_number" expression="^(1234)$">
  <!--ishler(actions) burda menimsedilir -->
</condition>
</extension>
```

Həmçinin siz məntiqi **AND**-lə aşağıdakı misaldaki kimi, şərtlər axını da təyin edə bilərsiniz:

```
<extension name="Two condition tags example">
<condition field="ani" expression="^(1111)$"></condition>
<condition field="destination_number" expression="^(1234)$">
  <!--Öncə göstərilən şərtlərin hər ikisi doğru olarsa, görüləcək işlər
  burda olacaq -->
</condition>
</extension>
```

Öncə göstərilən misalda şərtlər birlikdə **true** olmalıdır ki, ikinci **<condition>** tag daxilində olan işlər yerinə yetirilsin. **ani** sütunu(hansı ki, zəng edən tərəfin telefon nömrəsidir) **1111** olmalıdır və dialed_number(zəng edilən nömrə) sütunu yerinə yetirilən iş üçün **1234** olmalıdır. Bu genişlənmənin asan açıqlanması belə ola bilər "Əgər zəng edən 1111 və zəng edilən 1234 olarsa, göstərilən işləri yerinə yetir". Bir neçə **<condition>** tag-ların bir genişlənmə daxilində yerləşdirmək bəzi hallarda şərtin qatlanmasına gətirib çıxara bilər. Qeyd edin ki, ilk şərt - hələ də əlçatılan XML-dir və bağlayıcı tag tələb edir. XML bunun üçün sintaksis keçidinə izin verir hansı ki, aşağıda göstərilir:

```
<condition field="ani" expression="^(1111)$"/>
```

Bu slash simvoludur "/" hansı ki, **<condition>** tag-i bağlayır.

<condition> tag-nın əlavə bir imkanı **break** atributudur(bəzi məqamlarda siz bunu break flag ya da break parametri kimi eşidəcəksiniz). Çox səviyyəli şərtlərin birləşdirilməsini etdiyinizdə, break atributu sintaksis analizatorunun hər şərtin qiymətləndirilməsində özünün necə aparmasını idarə edə bilər. **break** atributunun aşağıda göstərilən **4** mənəsi ola bilər:

- **on-true:** Əgər mövcud şərt **true**-dursa, bu genişlənmədə növbəti şərtlərin axtarılmasını durduracaq.

- **on-false:** Əgər mövcud şərt **false**-dırsa, bu genişlənmədə olan növbəti şərtlərin axtarılmasını durduracaq(Bu susmaya görə olur).
- **never:** Mövcud şərtin **true** ya da **false** olmasından asılı olmayaraq, növbəti şərtləri axtarmağa davam elə.
- **always:** Mövcud şərtin **true** ya da **false** olmasından asılı olmayaraq, növbəti şərtlərin axtarışını durdur.

Susmaya görə olan şərt əgər birində uğursuzluq görərsə, daha böyük sayıda olan şərtlərin axtarışını durduracaq. Gəlin eyni genişlənməni **break** atributu ilə edək:

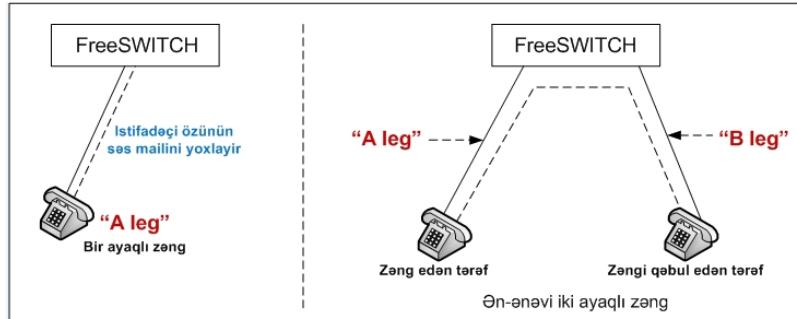
```
<extension name="Two condition tags example">
<condition field="ani" expression="^(1111)$" break="never">
  <!-- Əgər zəng edən 1111-dirssə, işlər burda yerinə yetiriləcək -->
</condition>
<condition field="destination_number" expression="^(1234)$">
  <!-- Zəng edən tərəf 1234 yiğarsa işlər burda olacaq. Zəng edən tərəf 1111
       olmasa da belə -->
</condition>
</extension>
```

Ilk şərtdə biz **break="never"** əlavə elədik. Sintaksis analizatoru artıq bu şərtə çatdıqda, özünü fərqli aparacaq. İlk şərtin dayanmasından asılı olmayaraq, sintaksis analizatoru eyni genişlənmənin daxilində olan növbəti şərtə yönənləncək. **break="never"** atributu olmasa idi, sintaksis analizatoru bu genişlənmə üçün növbəti parçalama işini görmədən dayanacaq və dialplan-ı davam edəcəkdi. Bəs bu şərt **true** olarsa, nə baş verəcək? İlk şərtin daxilində olan işlər görüləcək işlər siyahısına əlavə ediləcək və sonra analizator bu genişlənmədə olan növbəti şərtə kecid alacaq. Son nəticə ondan ibarət olacaq ki, əgər zəng edən tərəf **1234** yiğarsa müəyyən işlər görüləcək. Ancaq bizim digər işlərimizdə vardır hansı ki, əgər zəng edən tərəf **1111** olarsa yerinə yetiriləcək.

Qeyd: Daha çox şərtlər 8-ci başlıqda İrəliləmiş Dialplan Konsepsiyasında açıqlanır.

Zəng ayaqları və kanal dəyişənləri

FreeSWITCH-dən ya da ona gələn zənglər bir və ya bir neçə ayaqdan ibarət ola bilər. Bir ayaqlı zəng qoşulması istifadəçinin öz voicemail-nə yoxlaması üçün zəng etməsinə oxşaya bilər. Ən-ənəvi iki tərəf arasında zəngə iki ayaqlı zəng deyilir. 3-cü başlıqda Nüsxə quraşdirmalarının sınaqdan keçirilməsi başlığında gördüğünüz şəkli yadınıza salın:



Iki fərqli telefon arasında olan zəng **A**(Zəng edən tərəf) **ayağı** və **B**(zəngi qəbul edən tərəfi) **ayağı** kimi göstərilir. Hər bir zəng ayağı səs kanalında olduğu kimi, həmçinin **channel** olaraq tanınır. Hər bir kanalın özünə aid olan məntiqi atribut siyahısı vardır hansı ki, siz bunu konkret zəng ayağı üçün faktlar siyahısı adlandırma bilərsiniz. Bu atributlardan hər biri uyğun olan kanal dəyişənində saxlanılır. Öncəki başlıqda biz öyrəndik ki, qeydiyyatdan keçmiş istifadəçinin təyin edilmiş çoxlu kanal dəyişənləri mövcuddur və bu dəyişənlər zəng edən istifadəçi ayağı tərəfdə əlavə edilir. Zəngdə nə qədər məlumatın olmasına anlamaq üçün bu ideyanı reallıqda görmək üçün, siz məlumat genişlənməsi **9192** nömrəsinə aşağıdakı addımlarla zəng edə bilərsiniz:

1. **fs_cli**-i işə salın və **/log 6** əmrini daxil edin:

```
# fs_cli
freeswitch@internal> /log 6
+OK log level 6 [6]
```

2. İstənilən qeydiyyatdan keçmiş telefondan **9192** nömrəsinə zəng edin.

Siz onlarla sətir məlumat görəcəksiniz və aşağıda onlardan bir qismi göstərilir:

```
2015-09-19 13:41:20.853062 [INFO] mod_dptools.c:1681 CHANNEL_DATA:
Channel-State: [CS_EXECUTE]
Channel-Call-State: [ACTIVE]
Channel-State-Number: [4]
Channel-Name: [sofia/internal/1000@frfs.opensource.az]
Unique-ID: [0942432d-aa5e-e511-9271-0050568873a8]
Call-Direction: [inbound]
Presence-Call-Direction: [inbound]
Channel-HIT-Dialplan: [true]
Channel-Presence-ID: [1000@frfs.opensource.az]
Channel-Call-UUID: [0942432d-aa5e-e511-9271-0050568873a8]
Answer-State: [answered]
Channel-Read-Codec-Name: [G722]
Channel-Read-Codec-Rate: [16000]
Channel-Read-Codec-Bit-Rate: [64000]
Channel-Write-Codec-Name: [G722]
Channel-Write-Codec-Rate: [16000]
```

```

Channel-Write-Codec-Bit-Rate: [64000]
Caller-Direction: [inbound]
Caller-Logical-Direction: [inbound]
Caller-Username: [1000]
Caller-Dialplan: [XML]
Caller-Caller-ID-Name: [fr0]
Caller-Caller-ID-Number: [1000]
Caller-Orig-Caller-ID-Name: [fr0]
Caller-Orig-Caller-ID-Number: [1000]
Caller-Network-Addr: [85.132.57.60]
Caller-ANI: [1000]
Caller-Destination-Number: [9192]
Caller-Unique-ID: [0942432d-aa5e-e511-9271-0050568873a8]
Caller-Source: [mod_sofia]
Caller-Context: [default]
Caller-Channel-Name: [sofia/internal/1000@frfs.opensource.az]
Caller-Profile-Index: [1]
Caller-Profile-Created-Time: [1442652070173033]
Caller-Channel-Created-Time: [1442652070173033]
Caller-Channel-Answered-Time: [1442652080853062]
Caller-Channel-Progress-Time: [0]
Caller-Channel-Progress-Media-Time: [1442652080853062]
Caller-Channel-Hangup-Time: [0]
Caller-Channel-Transfer-Time: [0]
Caller-Channel-Resurrect-Time: [0]
Caller-Channel-Bridged-Time: [0]
Caller-Channel-Last-Hold: [0]
Caller-Channel-Hold-Accum: [0]
Caller-Screen-Bit: [true]
Caller-Privacy-Hide-Name: [false]
Caller-Privacy-Hide-Number: [false]
variable_direction: [inbound]
variable_uuid: [0942432d-aa5e-e511-9271-0050568873a8]
variable_session_id: [4]
variable_sip_from_user: [1000]
variable_sip_from_uri: [1000@frfs.opensource.az]
variable_sip_from_host: [frfs.opensource.az]
variable_channel_name: [sofia/internal/1000@frfs.opensource.az]
variable_sip_call_id: [77495ZThjOTFlZDZkYjg1NTY4NGUxMjZkZDlNTNiODFizGE]
variable_ep_codec_string:
[G722@8000h@20i@64000b, PCMA@8000h@20i@64000b, opus@48000h@20i@2c, PCMU@8000h@20i@64000b]
variable_sip_local_network_addr: [94.20.81.154]
variable_sip_network_ip: [85.132.57.60]
variable_sip_network_port: [57026]
variable_sip_received_ip: [85.132.57.60]
variable_sip_received_port: [57026]
variable_sip_via_protocol: [udp]
variable_sipAuthorized: [true]
variable_Event-Name: [REQUEST_PARAMS]
variable_Core-UUID: [6d46e04e-745c-e511-9271-0050568873a8]
variable_FreeSWITCH-Hostname: [frfs.opensource.az]
variable_FreeSWITCH-Switchname: [frfs.opensource.az]

```

```

variable_FreeSWITCH-IPv4: [94.20.81.154]
variable_FreeSWITCH-IPv6: [::1]
variable_Event-Date-Local: [2015-09-19 13:41:10]
variable_Event-Date-GMT: [Sat, 19 Sep 2015 08:41:10 GMT]
variable_Event-Date-Timestamp: [1442652070173033]
variable_Event-Calling-File: [sofia.c]
variable_Event-Calling-Function: [sofia_handle_sip_i_invite]
variable_Event-Calling-Line-Number: [9056]
variable_Event-Sequence: [32685]
variable_sip_number_alias: [1000]
variable_sip_auth_username: [1000]
variable_sip_auth_realm: [frfs.opensource.az]
variable_number_alias: [1000]
variable_requested_domain_name: [94.20.81.154]
variable_record_stereo: [true]
variable_default_gateway: [example.com]
variable_default_areacode: [918]
variable_transferFallback_extension: [operator]
variable_toll_allow: [domestic,international,local]
variable_accountcode: [1000]
variable_user_context: [default]
variable_effective_caller_id_name: [Extension 1000]
variable_effective_caller_id_number: [1000]
variable_outbound_caller_id_name: [FreeSWITCH]
variable_outbound_caller_id_number: [0000000000]
  
```

variable sətiri ilə başlayan sətirlər uyğun olan kanal dəyişəninin mənasını göstərir. Misal üçün **variable_sipAuthorized**: `[true]` sətiri göstərir ki, **sipAuthorized** kanalının mənəsi `true`-dir. Siz həmçinin digər data elementlərini də çıxışda görəcəksiniz hansı ki, *Unique-ID* və *Call-Direction* onlardandır. Bunlar *info* program dəyişənləridir. Onlardan əksəriyyəti (ancaq hamısı deyil) **read-only** mənaya malikdirlər və onlara yetkini eynilə kanal dəyişənlərində olduğu kimi almaq olur.

Kanal dəyişənlərinə yetkinin alınması

Dialplan çərçivəsində dəyişənlərə yetkini spesifik nəsihətlə alırlar:

`${variable_name}`. Növbəti misala baxın:

```
<action application="log" data="INFO sipAuthorized menasi '${sipAuthorized}'"/>
```

Bu iş FreeSWITCH cli-na aşağıdakı jurnal mesajını çap edəcək:

```
2015-09-09 14:32:48.904383 [INFO] mod_dptools.c:897 sipAuthorized menasi 'true'
```

read-only olanlara olan yetki demək olar ki, eynidir. Bu mənalardan hər birinin uyğun olan kanal dəyişəni adı mövcuddur. Misal üçün:

```
<action application="log" data="INFO Unique-ID menasi '${uuid}'"/>
```

Bu FreeSWITCH CLI-da aşağıdakı jurnal sətirini çap edəcək:

```
2015-09-09 14:46:31.695458 [INFO] mod_dptools.c:897 UniqueID menasi '169ae42e-29f5-4e1c-9505-8ee6ef643081'
```

Qeyd: **info** programı dəyişənlərinin tam siyahısı və onların kanal dəyişənləri adları aşağıdakı ünvandan əldə edilə bilər:

https://freeswitch.org/confluence/display/FREESWITCH/Channel+Variables#Info_Application_Variable_Names_.28variable_xxxx.29

Kanal dəyişənləri 8-ci başlığın Irəliləmiş Dialplan konsepsiyalarında daha ətraflı danışılacaq.

Müntəzəm ifadələr(Regular expressions)

FreeSWITCH XML Dialplan demək olar ki, Perl-compatible regular expression - PCRE (Perl uyğunluqlu müntəzəm ifadələr) geniş tətbiq edir. Müntəzəm ifadələr - **true/false** sınaqlarının sətir simvollarında yerinə yetirilməsi deməkdir. Buna adətən nümunəylə tutuşturulma deyilir. Müntəzəm ifadə sətir simvollarına mənimsədildikdə, biz sadə suala cavab veririk: nümunə uyğundurmu? Əgər cavab **yes** (qəbul)-dirse onda, adətən təyin edilən şərt yerinə yetirilir və ona görə də sualda olan genislənmə yerinə yetirilə bilər. Bəzi hallarda bizdən tələb edilə bilər ki, nümunə tutuşturulması uyğun olmasın (Bu başlığın **actions** və **anti-actions**-una baxın). Perl uyğunluqlu müntəzəm ifadələr çox spesifik sintaksisə malikdir. Öncə çox çətin görünə bilər. Ancaq, əsasları öyrəndikdən sonra görəcəksiniz ki, imkanlar həddən artıq güclü və dəyərlidir. Aşağıda müəyyən müntəzəm ifadələr və onların mənaları göstərilir:

Nümunə	Mənası
123	Istənilən "123" ardıcılığında olan söz uyğundur.
^123	Istənilən "123" ardıcılığı ilə başlayan söz uyğundur.
123\$	Istənilən "123" ardıcılığı ilə bitən söz uyğundur.
^123\$	Yalnız istənilən "123" olan söz uyğundur.
\d	(0-9) aralığında olan istənilən bir rəqəm uyğundur.
\d\d	Iki ardıcıl rəqəm ilə uyğundur.
^\d\d\d\$	Yalnız 3 rəqəm uzunluğunda olan istənilən sözlə uyğundur.
^\d{7}\$	Istənilən 7 simvol uzunluğunda olan sözlə uyğundur.
^\(\d{7}\)\$	Istənilən 7 simvol uzunluğunda olan sözlə uyğun et və uyğun olan mənani spesifik \$1 dəyişənində saxla.
^\1?(\d{10})\$	Əvvəli "1" rəqəmi ilə başlayan, sonra istənilən əlavə 10 rəqəmlə bitən (sonrakı istənilən 10 rəqəm \$1 dəyişəninə mənimsədir) uyğundur.
^(3\d\d\d)\$	"3" rəqəmi ilə başlayan istənilən 4 rəqəmli ilə uyğundur və alınan mənani \$1 dəyişəninə mənimsədir.

Şübhəsiz görmək olur ki, müntəzəm ifadələrlə istənilən nəzərimizə gələn rəqəmləri tutuşturara bilərik. Müntəzəm ifadələrlə həmçinin hərflər və spesifik simvollar uyğunlaşdırara bilərik.

/usr/local/freeswitch/conf/dialplan/default.xml faylinə baxsanız istifadə edilmiş çoxlu fərqli müntəzəm ifadə görə bilərsiniz.

Əgər siz adı sözün spesifik başlıqla tutuşmasını bilmək istəyirsinizsə, FreeSWITCH CLI-da **regex** əmrindən istifadə edin (regex sözü qısa olaraq regular expression-dan götürülmüşdür və adətən "**REG-ex**" kimi elan edilir). **regex** əmri ən azı 2 argument tələb edir: test etmək üçün məlumatlar və yenidən uyğun olmaq üçün nümunə. Argumentler **| (pipe)** simvolu ilə ayrılırlar. Məlumat və

nümunə uyğun olarsa **regex** əmri **true** cavabını qaytaracaq əks halda, **false** qaytaracaq. Siz **fs_cli**-dan aşağıdakı əmrləri yoxlaya bilərsiniz:

```
freeswitch@internal> regex 1234|\d
true
freeswitch@internal> regex 1234|\d\d\d\d
true
freeswitch@internal> regex 1234|\d{4}
true
freeswitch@internal> regex 1234|\d{5}
false
freeswitch@internal> regex 1234|^1234$ 
true
freeswitch@internal> regex 1234|234
true
freeswitch@internal> regex 1234|^234
false
```

regex əmrinin həmçinin ələ keçirmə sintaksisi mövcuddur hansı ki, **true** və **false**-dan əldə edib saxladığı mənani geri qaytara bilir. **%0** ifadəsi bütün uyğun olan, **%1** ilk uyğun olan və **%2** ikinci uyğun olan və.s mənani tutur. Aşağıdakı əmrləri yoxlayın:

```
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d)
true
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d) |%0
18005551212
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d) |%1
800
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d) |%2
555
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d) |%3
1212
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d) |%1%2%3
8005551212
freeswitch@internal> regex 18005551212|1?(\d\d\d) (\d\d\d) (\d\d\d\d) |%1-%2-%3
800-555-1212
freeswitch@internal> regex 18005551212|^7|%0
18005551212
```

Məlumatlar sətirinin və nümunələrin cəld salsaqdan keçirilməsi üçün **regex** əmrindən istifadə edin.

Qeyd: Müntəzəm ifadələr kompyuterlə əlaqəli çoxlu digər şərtlərdə də yararlıdır. Siz FreeSWITCH-in rəsmi səhifəsindən <https://freeswitch.org/confluence/display/FREESWITCH/Regular+Expression> RegEX haqqında daha da ətraflı öyrənə bilərsiniz.

Actions və anti-actions

Actions(islər) faktiki olaraq Dialplan-da uyğunluq olan kimi, öz addımlarına başlayır. Actions həmişə şərt və genişlənmə blokunun daxilində olur. Actions və anti-actions birlikdə FreeSWITCH-ə deyirlər ki, zəngə təsir elə. Bu ikisi

arasında olan fərq çox sadədir: şərt yerinə yetirilərsə iş(action) görülür və anti-action şərt yerinə yetirilməyəndə işə salınır. Aşağıdakı misala baxın(Göstərilən kodu **/usr/local/freeswitch/conf/dialplan/default.xml** faylına yerləşdirib freeswitch daemon-u yenidən işə salın ki, yoxlaya biləsiniz):

```
<extension name="Action ve eksine anti-action nusxesи">
<condition field="destination_number" expression="^(9101)$">
  <action application="log" data="INFO Eger 9101 yiqildisa jurnal yaz"/>
  <anti-action application="log" data="INFO Eger 9101 yiqilmadisa jurnal yaz"/>
</condition>
</extension>
```

Öncəki misalda istifadəçinin etdiyi zəngə əsasən **<extension>** jurnalı FreeSWITCH CLI-a yazacaq. Əgər istifadəçi **9101** nömrəsinə zəng edirse iş görülür və "Əgər 9101 **yiqildisa** jurnal yaz" jurnal sətiri ekrana çıxacaq. Əgər istənilən digər nömrəyə zəng etsəniz anti-action işə düşəcək və "Əgər 9101 **yiqilmadisa** jurnal yaz" sətiri ekrana çap ediləcək.

9101 nömrəsi və istənilən digər nömrəyə zəng etdikdə nəticə aşağıdakı kimi olmalıdır:

```
2015-09-22 07:52:51.409828 [INFO] mod_dptools.c:1662 Əgər 9101 yiqildisa jurnal yaz
2015-09-22 07:53:12.849790 [INFO] mod_dptools.c:1662 Əgər 9101 yiqilmadisa jurnal yaz
```

Yaratdığınız eksər genişlənmələrdə çoxlu action(iş)-lar və bəzi anti-action(əks işləri)-lar olacaq(bu nüsxələr həqiqətən də Dialplan-da mövcuddur). Əksər hallarda actions-lar Dialplan programlarını yerinə yetirir hansı ki, növbə ilə parametrləri ala bilirlər. Öncəki misalda jurnal programı yerinə yetirilir və ötürülməsi üçün data atributunda parametr olur.

Dialplan prosesinin işləməsi

Zəngin gəldiyi anda baş verənlərin vizuallaşdırılması edilsə Dialplan-ın anlaşılması çox asandır. Əksər hallarda biz "the call traverses the Dialplan" ya da "the call hits the Dialplan" kimi ifadələr eşidirik. Dəqiq olaraq bu nədir? Gəlin zəng prosesinin içənə gedək ki, XML Dialplan-ın dəqiqliklə nə etdiyini görə biləsiniz.

Dialplan-ın iki fazası var: **parsing**(sintaksis analizatoru) və **executing**(yerinə yetirilmə). Dialplan sintaksis analizatoru genişlənmə axtarır ki, yerinə yetirsən. O uyğun olan genişlənməni tapan kimi, tapşırıqlar siyahısına yerinə yetiriləcək action(ya da anti-action) əlavə edir. Analizator genişlənmə axtarışını bitirdikdə, yerinə yetirilmə fazası işə başlayır və işlər siyahısında actions(işlər) yerinə yetirilir. Bütün bu işlərin nədən ibarət olduğunu görmək üçün, telefon zəngi edərək debug rejimdə FreeSWITCH console-una baxmaq lazımdır. **fs_cli**-a daxil olun, **9196(echo test)** sınaq zəngi edin və dəstəyi üstünə qoyn. Sonra terminala baxın ki, aşağıdakına uyğun olan sətiri tapasınız:

```
2015-09-22 08:23:05.549824 [INFO] mod_dialplan_xml.c:635 Processing fr0
<1000>->9196 in context default
```

Bu dialplan prosesinin başlanğıcıdır. **fr0** adlı telefon nömrəsi **9196** nömrəsinə zəng elədi(zəng etdiyiniz telefonun uyğun adı sizin konsolunuzda görsənəcək). Növbəti gələn sətirlər Dialplan-la başlayır və debug mesajları göstərir ki, hansı genişlənmələr üst-üstə düşmüşdür və hansı yox. İlk analiz edilmiş genişlənmə **unloop** adlanır. Bu çox maraqlı genişlənmədir amma, bizim Dialplan diskussiyamızda çoxda maraqlı deyil. Analiz edilən növbəti genişlənməyə baxın. Bizim misalda aşağıdakı kimidir:

```
Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default->unloop]
continue=false
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Regex (PASS) [unloop]
${unroll_loops}(true) =~ /^true$/ break=on-false
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [unloop]
${sip_looped_call}() =~ /^true$/ break=on-false
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default-
>tod_example] continue=true
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Date/TimeMatch (FAIL)
[tod_example] break=on-false
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default-
>holiday_example] continue=true
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Date/TimeMatch (FAIL)
[holiday_example] break=on-false
```

tod_example(time of day example) genişlənməsi ilk analizdə göstərilir. **tod_example** genişlənməsi üçün **debug** sətirləri **/usr/local/freeswitch/conf/dialplan/default.xml** faylında olan quraşdırımlara görə əmələ gəlir:

```
<extension name="tod_example" continue="true">
<condition wday="2-6" hour="9-18">
  <action application="set" data="open=true"/>
</condition>
</extension>
```

Bu genişlənmə sadəcə günün vaxtını və həftənin günlərini yoxlayır. Əgər zəng həftə içi (1-ci 5-ci gün arası) (09:00 və 18:59 arası) yerinə yetirilirsə o **channel** dəyişənini **true** üçü açıq təyin edir. Bu zəng 2-ci gün saat 8:58-də edilmişdir. Ona görə də bu **wday**(day of week) sınağına ötürüldü ama **hour**(günün saatına)-a ötürülmədi. Əgər zəng 09:00 və 18:00 aralığında yerinə yetirilərsə hər iki şərt uyğun olacaq və təyin programı iş siyahısına əlavə ediləcək. Nəzər yetirin ki, **tof_example(həftənin günü misali)** genişlənməsində **continue="true"** durur. Bu o deməkdir ki, Dialplan analiz işini **tod_example** uyğun olsa da belə növbəti genişlənmələr üçün edəcək.

Sintaksis analizatoru eksəriyyəti uğursuz olsa da belə, genişlənmələrin uyğunluğunu tutuşdurmaq üçün davam edir:

```
Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default->global-
intercept] continue=false
```

```

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [global-intercept] destination_number(9196) =~ /^886$/ break=on-false

Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default->group-intercept] continue=false

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [group-intercept] destination_number(9196) =~ /^.*8$/ break=on-false

Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default->intercept-ext] continue=false

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [intercept-ext] destination_number(9196) =~ /^.*\*(\d+)$/ break=on-false

Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default->redial] continue=false

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [redial] destination_number(9196) =~ /^(redial|870)$/ break=on-false

Dialplan: sofia/internal/1000@frfs.opensource.az parsing [default->global] continue=true
Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [global] ${call_debug}(false) =~ /true$/ break=never

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (PASS) [global] ${default_password}(freebsd) =~ ^freebsd$/ break=never

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [global] ${rtp_has_crypto}() =~ /^(AEAD_AES_256_GCM_8|AEAD_AES_128_GCM_8|AES_CM_256_HMAC_SHA1_80|AES_CM_192_HMAC_SHA1_80|AES_CM_128_HMAC_SHA1_80|AES_CM_256_HMAC_SHA1_32|AES_CM_192_HMAC_SHA1_32|AES_CM_128_HMAC_SHA1_32|AES_CM_128_NULL_AUTH)$/ break=never

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (PASS) [global] ${endpoint_disposition}(DELAYED NEGOTIATION) =~ ^(DELAYED NEGOTIATION)/ break=on-false

Dialplan: sofia/internal/1000@frfs.opensource.az Regex (FAIL) [global] ${switch_r_sdp}(v=0
o=- 13087372758927860 1 IN IP4 85.132.57.60
s=Bria 4 release 4.2.0 stamp 77495
c=IN IP4 85.132.57.60
t=0 0
m=audio 52128 RTP/AVP 9 8 18 120 0 122 101
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=no
a=rtpmap:120 opus/48000/2
a=fmtp:120 useinbandfec=1; usedtx=1; maxaveragebitrate=64000
a=rtpmap:122 SILK/16000
a=rtpmap:101 telephone-event/8000

```

```
a=fmtp:101 0-15
) =~ /(AES_CM_128_HMAC_SHA1_32|AES_CM_128_HMAC_SHA1_80)/ break=never
```

Aşağıda göstərilən debug sətirləri uyğun olmayan tutuşdurmalardır hansı ki, tam normaldır. Sonra biz bəzi maraqlı çıxışları aşağıdakı kimi yiğacayıq:

Dialplan: `sofia/internal/1000@frfs.opensource.az Absolute Condition [global]`

```
Dialplan: sofia/internal/1000@frfs.opensource.az Action
hash(insert/${domain_name}-spymap/${caller_id_number}/${uuid})
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Action
hash(insert/${domain_name}-
last_dial/${caller_id_number}/${destination_number})
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Action
hash(insert/${domain_name}-last_dial/global/${uuid})
```

```
Dialplan: sofia/internal/1000@frfs.opensource.az Action
export(RFC2822_DATE=${strftime(%a, %d %b %Y %T %z)})
```

Absolute Condition olan debug sətirlərinə diqqət yetirin. *Absolute Condition* sətirləri o mənəni kəsb edir ki, şərt həmişə true-dur və tərkibində olan işlər həmişə yerinə yetirilir. Bu şərt tag-i `/usr/local/freeswitch/conf/dialplan/default.xml` faylında qlobal genişlənməsində yerləşir. Aşağıdakı kimi göstərilir:

```
<extension name="global" continue="true">
  <condition field="${call_debug}" expression="^true$" break="never">
    <action application="info"/>
  </condition>

  <condition field="${default_password}" expression="^freebsd$"
break="never">
    <!--
      <action application="log" data="CRIT WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING "/>
      <action application="log" data="CRIT Open $$conf_dir}/vars.xml and
change the default_password."/>
      <action application="log" data="CRIT Once changed type 'reloadxml' at
the console."/>
      <action application="log" data="CRIT WARNING WARNING WARNING WARNING
WARNING WARNING WARNING WARNING "/>
      <action application="sleep" data="10000"/>
    -->
  </condition>
  <!--
    This is an example of how to auto detect if telephone-event is
missing and activate inband detection
  -->
  <!--
  <condition field="${switch_r_sdp}"
expression="a=rtpmap:(\d+)\stelephone-event/8000" break="never">
    <action application="set" data="rtp_payload_number=$1"/>
```

```

<anti-action application="start_dtmf"/>
</condition>
-->
<condition field="${rtp_has_crypto}"
expression="^($${rtp_sdes_suites})$" break="never">
    <action application="set" data="rtp_secure_media=true"/>
    <!-- Offer SRTP on outbound legs if we have it on inbound. -->
    <!-- <action application="export" data="rtp_secure_media=true"/> -->
</condition>

<!--
    Since we have inbound-late-negotiation on by default now the
    above behavior isn't the same so you have to do one extra step.
-->
<condition field="${endpoint_disposition}" expression="^(DELAYED
NEGOTIATION)"/>
    <condition field="${switch_r_sdp}"
expression="(AES_CM_128_HMAC_SHA1_32|AES_CM_128_HMAC_SHA1_80)" break="never">
        <action application="set" data="rtp_secure_media=true"/>
        <!-- Offer SRTP on outbound legs if we have it on inbound. -->
        <!-- <action application="export" data="rtp_secure_media=true"/> -->
    </condition>

<condition>
    <action application="hash" data="insert/${domain_name}-
spymap/${caller_id_number}/${uuid}"/>
    <action application="hash" data="insert/${domain_name}-
last_dial/${caller_id_number}/${destination_number}"/>
    <action application="hash" data="insert/${domain_name}-
last_dial/global/${uuid}"/>
        <action application="export" data="RFC2822_DATE=${strftime(%a, %d %b
%Y %T %z)}"/>
    </condition>
</extension>

```

global genişlənmədə olan(qırmızı ilə seçilmiş) dördüncü `<condition>` tag-i heç bir sütun ya da ifadəyə sahib deyil və həmçinin həmişə **true** kimi dəyərləndirilir(Buna görə də işlər iş siyahısına əlavə edilir). Bunun nəticəsinə yaxşı sual ortaya çıxır: adı halda sintaksis analizatoru mövcud genişlənmədə şərt işləməyən kimi, öz işini dayandırır və nə üçün sintaksis analizatoru global genişlənməni öz ilk şərti işləmədikdən sonra keçmədi? Cavab **break** atributu sətirindədir. Diqqətlə baxın ki, genişlənməmizin daxilində öncəki şərtlərdə **break="never"** təyin edilmişdir. Bu sintaksis analizatoruna deyir ki, bu genişlənməni şərtin true ya da false olmasından asılı olmayaraq sonadək analiz et(Bu başlıqda öndə olan Conditions sekisiyinə baxın).

Növbəti olan əksər şərtlərdə sintaksis analizatoru music on hold(gözləmədə musiqi)-a çatanadək üst-üstə düşəni olmayıcaq.

Dialplan: `sofia/internal/1000@frfs.opensource.az Regex (PASS) [echo]`
`destination_number(9196) =~ /^9196$/ break=on-false`
Dialplan: `sofia/internal/1000@frfs.opensource.az Action answer()`
Dialplan: `sofia/internal/1000@frfs.opensource.az Action echo()`

Sintaksis analizatoru öz iş siyahısına **answer** və **echo** Dialplan programını əlavə edir:

```
EXECUTE sofia/internal/1000@frfs.opensource.az hash(insert/94.20.81.154-
spymap/1000/ef2658a6-a061-e511-9f6d-0050568873a8)
EXECUTE sofia/internal/1000@frfs.opensource.az hash(insert/94.20.81.154-
last_dial/1000/9196)
EXECUTE sofia/internal/1000@frfs.opensource.az hash(insert/94.20.81.154-
last_dial/global/ef2658a6-a061-e511-9f6d-0050568873a8)
EXECUTE sofia/internal/1000@frfs.opensource.az export(RFC2822_DATE=Wed, 23
Sep 2015 08:10:31 +0500)
2015-09-23 08:10:31.840549 [DEBUG] switch_channel.c:1267 EXPORT (export_vars)
[RFC2822_DATE]=[Wed, 23 Sep 2015 08:10:31 +0500]
EXECUTE sofia/internal/1000@frfs.opensource.az answer()
```

Öncəki misal tamamilə Dialplan-ı keçib emal edilən zəngin nüsxəsidir. Bu prosesi bir neçə dəqiqə müzakirə etməyimizə baxmayaraq, serverdə çox sürətli gedir. Bu "öncə analiz et, sonra yerinə yetir" proses strategiyası XML Dialplan-ı daha da effektiv edir. Misal olaraq çıxışın analiz edilməsi üçün **9664** nömrəsinə(gözləmədə musiqi), bir nömrədən digərinə zəng edib debug çıxışına baxın. Artıq Dialplan-ı işdə gördükdən sonra gəlin yeni bir genişlənmə yaradaq.

Yeni extension(genişlənmə)-ın yaradılması

Gəlin tam yeni genişlənmə yaradaq.

/usr/local/freeswitch/conf/dialplan/default/01_custom.xml faylına yeni sınağımız üçün lazımi sətirləri əlavə edəcəyik ki, sınaqlarımızı edək.

Oeyd: Həmişə özünüzə aid olan Dialplan fayllarının adlarını rəqəm ardıcılılığı ilə adlandırın. Bunun səbəbi ondan ibarətdir ki, XML fayl analizatoru XML faylları ASCII fayl adlarına uyğun olaraq oxuyur.
/usr/local/freeswitch/conf/dialplan/default/ qovluğunda olan analiz ediləcək son fayl **99999_enum.xml** olacaq. Bu faylda enum extension olur və son kimi istifadə edilir ki, yiğilan nömrə heç birinə uyğun olmazsa bu istifadə edilsin. Ətraflı məlumat üçün https://freeswitch.org/confluence/display/FREESWITCH/mod_enum linkinə müraciət edə bilərsiniz.

Dialplan XML faylinin tərkibi bir və ya bir neçə genişlənmədən ibarətdir. Məhdudiyyət yalnız ondan ibarətdir ki, fayl **<include>** tag-la başlamalı və **</include>** tag-la bağlanmalıdır.

Bizim yeni genişlənmə sadəcə bir misal olacaq ancaq, FreeSWITCH-in Dialplan imkanlarını göstərəcək. Genişlənmə aşağıdakı xarakteristikalara sahibdir:

- Zəngə cavab vermə
- Zəng edən istifadəçinin genişlənməsinin nömrəsini iki fərgli formatda oxumaq olur. Hər bir oxunmadan sonra bir saniyə gözlənilir.
- İki saniyəlik durma.
- Zəng edənə goodbye deyir.
- Dəstəyi asma.
- Zəng edən 9101 yiğdiqda genişlənmə yerinə yetiriləcək.

Yeni genişlənmənin yerinə yetirilməsi üçün aşağıdakı addımları atın:

1. Aşağıdakı tərkibi

```
/usr/local/freeswitch/conf/dialplan/default/01_custom.xml faylinə əlavə
edin:
<include>
  <extension name="simple test">
    <condition field="destination_number" expression="^(9101)$">
      <action application="answer"/>
      <action application="say" data="en number iterated ${ani}"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en number pronounced ${ani}"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en name_spelled iterated ab733#/"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en name_spelled pronounced ab733#/"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en name_spelled iterated
${destination_number}"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en number pronounced 12345"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en ip_address iterated
12.34.56.78"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en ip_address pronounced
12.34.56.78"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en CURRENCY PRONOUNCED -1.96"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en short_date_time pronounced [timestamp]"/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en CURRENT_TIME pronounced ${strePOCH()}/>
      <action application="sleep" data="1000"/>
      <action application="say" data="en number counted 12345"/>
      <action application="say" data="en name_spelled pronounced salam"/>
      <action application="playback" data="voicemail/vm-goodbye.wav"/>
      <action application="sleep" data="2000"/>
    <action application="hangup"/>
  </extension>
</include>
```

```
</condition>
</extension>
</include>
```

2. Faylı yadda saxlayın. **fs_cli**-i işə salın **reloadxml** əmrini işə salın və ya **F6** düyməsinə sıxın.

Yeni genişlənmə artıq əlavə edilmişdir. Yeni genişlənməni sınaqdan keçirmək üçün sadəcə **9101** nömrəsinə zəng edin(Həmçinin FreeSWITCH CLI-a baxın ki, extension emali zamanı nələr baş verdiyini anlaya biləsiniz). Sistem çıxışa bir neçə səslə danışmalıdır və sonra dəstəyi asmalıdır. Gəlin hər bir sətir üçün nəyin baş verdiyini araşdırıq:

```
<extension name="simple test">
```

Bu tag sadəcə genişlənmənin başlanması təyin edir. **name** atributu sadəcə istəyə bağlıdır. Ancaq, bu daha da oxunaqlı Dialplan yaradılmasına kömək edir. Həmçinin genişlənmənin adı FreesWITCH jurnallarında da görünəcək ki, troubleshoot zamanı da işimizi asanlaşdırır. Genişlənmə təyinatı **</extension>** tag-lə bitir.

```
<condition field="destination_number" expression="^(9101)$">
```

Condition tag bu genişlənmə üçün uyğun olan parametrləri təyin edir. Burda biz **destination_number** mənasını **^(9101)\$** üçün təyin edirik. Adı sözlə desək bu şərt deyir ki, "Əgər istifadəçi 9101 nömrəsinə zəng edərsə, şərt blokunun daxilində olan işlər yerinə yetiriləcək". Bu tag və bağlayıcı **</condition>** tag arasında olan bütün işlər yerinə yetiriləcək.

```
<action application="answer"/>
```

answer programı sadəcə ona deyilənləri edir: zənglərə cavab verir

```
<action application="say" data="en number iterated ${ani}" />
```

Bu **say** programını yerinə yetirir hansı ki, öncədən yazılmış səsləri rəqəmlərin, hərfərin, pul sayının, IP ünvanının, vaxt möhürüünün, tarixin və.s deyilməsi üçün istifadə etsin. **say** programının ilk 3 argumenti: say motoru(adi halda dil), səslənmə tipi və deyilmə metodu. Bu misalımızda biz **say** programına deyirik ki, **en** motorunu istifadə etsin(bu ingilis səs fayllarını istifadə edəcək) və göstərilən rəqəmləri ardıcıl şəkildə yerinə yetirəcək. **1234** rəqəmi "bir-iki-üç-dörd" kimi səsləndiriləcək. **say** programı bu başlığın Vacib Dialpan programları seksiyasında açıqlanacaq.

```
<action application="sleep" data="1000"/>
```

Öncəki misal sadəcə yatır yəni, işi ancaq 1000 millisaniyə(1 saniyə) üçün durdurur.

```
<action application="say" data="en number pronounced ${ani}" />
```

Bu yenidən say programını işə salır və bu dəfə açıq metod istifadə edilir. **pronounce**(açıq) metodla **say** programı rəqəmləri sadalamaq əvəzinə bütöv rəqəmi deyəcək. **1234** rəqəmi "Min iki yüz otuz dörd(İngiliscə)" kimi səslənəcək.

```
<action application="sleep" data="1000"/>
```

Öncəki misal işi yenidən 1000 millisaniyə(1 saniyə) üçün durdurur.

```
<action application="playback" data="voicemail/vm-goodbye.wav" />
```

Və bir neçə misaldan sonda **playback** programı işə düşəcək. **playback** programı səs faylini zəng edən tərəf üçün oxudacaq. Faylin adı argument kimi göstərilmişdir. Bu halda biz FreeSWITCH voicemail-də istifadə elədiyimiz eyni faylı istifadə edəcəyik. **playback** programı bu başlıqda *Vacib Dialplan programları* seksiyasında daha da detallı açıqlanacaq.

Qeyd: FreeSWITCH-in çoxlu sayda öncədə yazılmış səs faylları mövcuddur. Ingilis fayl adları FreeSWITCH mənbə kodları (**/root/src/freeswitch**) ünvanında **docs/phrase/phrase_en.xml** faylında sadalanmışdır.

```
<action application="sleep" data="2000"/>
```

Öncəki sətir yerinə yetirilməni **2000** millisaniyə (2 saniyə) dayandırır.

```
<action application="hangup"/>
```

Bizim işlərimizin son sətiri isə, öncəki sətirdəki kimidir və dəstəyi üstünə qoyaraq qoşulmanı kəsir.

Budur! Artıq siz ilkin tələb edilən genişlənməni öz sisteminizə əlavə etdiniz. Biz yaratdığımız əksər genişlənmələr **default** kontekstin altına gedəcək ona görə ki, onlar bizim sistemimizdə olan istifadəçilər üçün dizayn edilmişdir. Bəzi hallarda ola bilər ki, biz genişlənməni digər kontekstə əlavə edirik. Misal üçün daxil olan DID zəngin emal edilməsi üçün siz **public** dialplan-da dəyişiklik etməlisiniz. Biz həmçinin özümüzə uyğun olan kontekstin təyin edilməsini bir şirkət daxilində fərqli şöbələrin qurulmasında bir FreeSWITCH server istifadə elədiyimiz halda da edə bilərik.

Vacib Dialplan programları

FreeSWITCH-in 140-dan çox dialplan programı mövcuddur. Ancaq onlardan bəziləri çox vacibdir ona görə ki, əksər istifadə ediləndir. Bu başlıqda biz Dialplan-da daha çox istifadə edilənlərə baxacayıq.

bridge

bridge programı iki son nöqtəni bir-birilə birləşdirir.

Argument sintaksisi:

```
<target_endpoint>[,<target_endpoint>] [|<target_endpoint>]
```

Vergüllə ayrılmış son nöqtələrə eyni zamanda zəng edilir. Kanal simvolu “|” ilə ayrılan nöqtələrə zəng ardıcıl gedir. İlk zəngə cavab verən son nöqtə zəngi qəbul edir və digər son nöqtələrə gedən zəng dayandırılır.

Misallar:

```
<action application="bridge" data="user/1000"/>
<action application="bridge" data="sofia/gateway/my_gateway_name/$1"/>
```

Bu başlıqda olan *Dialstring* formatları seksiyasına baxın.

playback

playback programı sadəcə zəng edən tərəf üçün musiqi faylını oxudur. Fayllar çoxlu formatda ola bilərlər. FreeSWITCH-də əlavə edilmiş səs və musiqi fayllarının hamısı **.wav** formatındadır.

Argument sintaksisi: işə salınacaq səs faylına tam ünvan ya da daha yaxın olan ünvan.

Misallar:

```
<action application="playback" data="/absolute/path/to/sound.wav"/>
<action application="playback" data="voicemail/vm-goodbye.wav"/>
```

Siz həmçinin axının yerləşdiyi ünvandan da belə musiqi faylını işə sala bilərsiniz (**mp3** faylları üçün FreeSWITCH-də **mod_shout** yükleməlisiniz). Bu aşağıdakı kimi olacaq:

```
<action application="playback" data="http://sounds.com/path/to/sound.mp3"/>
```

say

say programı öncədən daxildə olan motoru istifadə edir ki, zəng edənə bəzi məlumatları səslə çatdırırsın. Bu *text-to-speech* deyil. **speak** programına baxın.

Argument sintaksisi:

```
<module_name> <say_type> <say_method> <text>
```

module_name adətən dildir. Misal üçün **en** ya da **ru**.

say_type parametri aşağıdakılardan biri ola bilər:

- number
- items
- persons
- messages
- currency
- time_measurement
- current_date
- current_time
- current_date_time
- telephone_number
- telephone_extension
- url
- ip_address

- e-mail_address
- postal_address
- account_number
- name_spelled
- name_phonetic
- short_date_time

say_method parametri aşağıdakılardan biri ola bilər:

- n/a (metod menimsədilə bilməz *not applicable*)
- pronounced ("Yüz iyirmi üç")
- iterated ("bir iki üç")
- counted (Spesifik işlər üçün "pronounced" kimidir)

Misallar:

```
<action application="say" data="en number pronounced 1234"/>
<action application="say" data="en number iterated 1234"/>
<action application="say" data="en currency pronounced 1234"/>
<action application="say" data="en items pronounced 1234"/>
```

play_and_get_digits

play_and_get_digits programı zəng edənə səs faylini oxudacaq və eyni zamanda da zəng edən tərəfdən yiğilan rəqəmləri dinləyəcək. Bu sizə interaktiv dialplanların IVR yaratmadan yaratmasına kömək edir.

Arqument sintaksisi:

```
<min> <max> <tries> <timeout> <terminators> <file> <invalid_file>
<var_name> <regex> [<digit_timeout>] ['<failure_ext> [failure_dp
[failure_context]]']
```

Arqumentlər:

- min: Yiğilması üçün tələb edilən minimal rəqəm sayı
- max: Yiğilması üçün tələb edilən maksimal rəqəm sayı
- tries: Faylin işə salınması cəhdinin sayı və rəqəmlərin yiğilması
- timeout: Abonentin rəqəmləri daxil edilməsinə başlamasından əvvəl, gözləmənin millisaniyələrlə olan vaxtı
- terminators: Rəqəmlər daxil edilmənin bitməsi üçün istifadə edilir əgər, minimal tələb edilən rəqəm sixilibsa (adətən#)
- file: Rəqəmlərin seçilməsi müddətində işləyəcək səs faylı
- invalid_file: Rəqəmlər regEX arqumentinə uyğun olmayan zaman işə salınacaq səs faylı.
- var_name: Kanalın dəyişəni, hansına ki, rəqəmlər yerləşdirilmiş olmalıdır
- regex: Rəqəmlərə uyğun olmaq üçün müntəzəm ifadə
- digit_timeout (istəkdən asılıdır): Rəqəmlər arası gözləniləcək millisaniyələrlə olan rəqəm (susmaya görə timeout-dan götürülən məna olur)
- failure_ext (istəkdən asılıdır): Əgər abonent girişdə izin verilən mənəni daxil etmirsə, zəng mənsəb genişlənməsinə ötürülməlidir.

- `failure_dp` (`failure_ext`-le birlikdə istəkdən asılıdır): `failure_ext` istifadə etdikdə, mənsəb Dialplan tipi
- `failure_context` (`failure_dp` ilə birlikdə istəkdən asılıdır): `failure_dp` istifadə etdikdə mənsəb Dialplan context-i.

Məsələn:

```
<action application="play_and_get_digits" data="2 5 3 8000 #  
/path/to/sound_file.wav /path/to/invalid_sound.wav my_digits \d+ "/>  
<action application="log" data="User entered these digits: ${my_digits}"/>
```

Bu misal `play_and_get_digits`-i aşağıdakı parametrlərlə işə salır:

- Minimal iki rəqəm gözlənilir
- Maksimal 5 rəqəm gözlənilir
- 3 dəfə cəhd edilə bilər
- Zəng edən tərəfin rəqəmin daxil etməsini bitirməsini təyin edənədək 8 saniyə gözləyir
- Ayırıcı olaraq `#` simvolu istifadə edir
- Rəqəmləri tollayan müddətədək `/path/to/sound_file.wav` səs faylını oxudur.
- Yalnız rəqəmlər daxil edilərsə, `/path/to/invalid_sound.wav` səs faylını oxudur.
- Zəng edilən rəqəmləri kanal dəyişəni `my_digits`-də yerləşdirir.
- Yenidən `\d+` nümunəsi ilə uyğunlaşdırır

ivr

`ivr` programı zəng edəni öncədən təyin edilmiş IVR-a yönləndirir.

Arqument sintaksisi: İşə salınacaq IVR-in adı

Misal:

```
<action application="ivr" data="demo_ivr"/>
```

6-cı başlıqda XML IVR-larin və Phrase Macros-ların istifadəsinə və həmçinin `/usr/local/freeswitch/conf/dialplan/default.xml` faylında olan `ivr_demo` genişlənməsinə baxın.

sleep

`sleep` programı Dialplan yerinə yetirilməsini, təyin edilmiş millisaniyələrlə olan rəqəmə görə fasılələyir.

Arqument sintaksisi: Fasile olası millisaniyələrlə olan rəqəm.

Məsələn:

```
<action application="sleep" data="1000"/>
```

answer

`answer` programı dəstəyi götürərək, zəng edən abonentə səs yolunu qurur. SIP terminləri ilə desək bu, kodekin təyin edilməsi(əgər təyin edilməyibse) üçün,

RTP paketlərin istifadə edilməsi və axını başlanılması məqsədilə FreeSWITCH-ə "200 OK" mesajının yollanılmasını məcbur edir.

Məsələn:

```
<action application="answer"/>
```

pre_answer

pre_answer programı əvvəldə medianın təyin edilməsini çıxsaq, eynilə **answer**-ə oxşayır. SIP termini ilə desək bu, FreeSWITCH-ə "183 Session Progress with SDP" mesajın yollanılmasını məcbur edir.

Misal:

```
<action application="pre_answer"/>
```

hangup

hangup programı səsli yolu kəsir və zəngi durdurur.

Arqument sintaksisi: Əlavə olaraq durmanın səbəbi.

Misallar:

```
<action application="hangup"/>
<action application="hangup" data="USER_BUSY"/>
```

set

set programı kanal dəyişənini təyin edir ya da API əmri Dialplanın içində emal edir (Bu funksiya 8-ci başlıqda Irəliləmiş Dialplan konsepsiyasında açıq göstərilib).

Arqument sintaksisi:

```
<variable_name=value>
```

Məsələn:

```
<action application="set" data="my_chan_var=example value"/>
<action application="log" data="INFO my_chan_var contains ${my_chan_var}"/>
```

transfer

transfer programı zəngi geriyə Dailplan içərisində ötürür. Bu analizator üçün tam yeni bir etap açır və işin yerinə yetirilməsi fazasında olur.

Arqument sintaksisi:

```
<destination number> [destination dialplan [destination context]]
```

Məsələn:

```
<action application="transfer" data="9664"/>
<action application="transfer" data="12345 XML custom"/>
```

Dialstring formatları

Dialplan müzakirəmizi tərk eləməzdən öncə yaxşı olardi ki, bir mövzuya da baxaq: **Dialstrings**. Dialstring - səsləndiyi kimi də, simvollar sətiridir hansı ki, FreeSWITCH tərəfindən zəng ediləcək mənsəb nömrəsini təyin edir. Bütün Dialstringlərin spesifik sintaksisi olur. Sintaksis zəng edilən son nöqtədən asılı olaraq dəyişir. FreeSWITCH-də vacib olan əksər Dialstring tipləri Sofia üçündür ona görə ki, onlar bizim SIP son nöqtələrinə necə zəng etdiyimizi təqdim edir. Ancaq gördüyüümüz kimi, fərqli Dialstring tipləri mövcuddur. Onlar əksərən aşağıda sadalanan iki yerdə istifadə edilir:

- **bridge** programı vasitəsilə Dialplan-da mövcud olan zəng ayağının körpülenməsi
- **originate** əmri ilə CLI-dan yeni zəng ayağının yaradılması

Qeyd: Dialstrin sintaksisi bridge dialplan programının ya da originate API əmrinin istifadə edilməsindən asılı olmayaraq eyni sintaksisə malikdir.

Gəlin bir neçə misalda Sofia Dialstriglə başlayaraq, Dialstring haqqında daha da ətraflı öyrənək:

- **sofia/<profile name>/<user@domain>**
- **sofia/gateway/<gateway name>/<user>**

4-cü başlıqda SIP və istifadəçi qovluğunda öyrəndiyimiz kimi, biz SIP gateway olub və ya olmamasından asılı olmayaraq digər son nöqtəyə zəng edə bilərik. Sofiyadan istifadə vaxtı, SIP profili vasitəsilə yiğmaq üçün, istifadəçini və domeni müəyyən etmək lazımdır. Ancaq, gateway vasitəsilə yiğaraq domeni daxil etmək lazım deyil, çünki bu artıq gateway-in konfiqurasiyasında təyin edilir. Buna görə də, növbəti sətirə icazə verilmir:

```
<!-- Yalnız -->
<action application="bridge" data="sofia/gateway/my_gateway/user@1.2.3.4"/>
```

Düzgün sintaksis aşağıdakı kimidir:

```
<!-- Düzgündür -->
<action application="bridge" data="sofia/gateway/my_gateway/user">
```

internal profile üzərindən gedən zənglər üçün sintaksisin ekvivalenti aşağıdakı kimi olacaq:

```
<!-- Həmçinin düzgündür -->
<action application="bridge" data="sofia/internal/user@1.2.3.4 /">
```

Bu iki sintaksisin biliyi sizin ehtiyaclarınızda olan SIP nömrəsi yiğiminin böyük əksəriyyətini örtecək. Yalnız, çox məhdud hallar da olur. Ancaq, yiğim sətirlərinin tam müzakirəsinə

https://freeswitch.org/confluence/pages/viewpage.action?pageId=3966106#SIP-Specific_Dialstrings linkində baxa bilərsiniz.

Sizin FreeSWITCH serverdə qeydiyyatdan keçən istifadəciyə zəng elədikdə qısa kecid olur:

user/<user id>[@domain]

Bu sintaksis sizin sistemdə olan digər istifadəçiyə zəngin edilməsi işini asanlaşdırır. Faktiki olaraq **/usr/local/freeswitch/conf/dialplan/default.xml** faylı bu metodikani qeydiyyatdan keçmiş istifadəçilərə zəngləri birləşdirmək üçün istifadə edir:

```
<action application="bridge" data="user/${dialed_extension}@${domain_name}" />
```

Əgər sizin FreeSWITCH serverinizdə müəyyən edilmiş yalnız bir domen varsa, **@domain** parametri əlavədir.

Aşağıdakilar Dialplan üçün bir neçə tipdir:

- **loopback/<destination number>**: Bu zəng ayağını yaradır və onu Dialplan-da **<destination_number>**-də yerləşdirir.
- **freetdm//<channel>/<phone number>**: Bu telefon interfeys kartında zəng ayağı yaradır (FreeSWITCH-lə ən-ənəvi telefon avadanlıqlarının istifadəsi haqqında daha ətraflı oxumaq üçün <http://wiki.freeswitch.org/wiki/FreeTDM> linkinə müraciət edə bilərsiniz)
- **error/<error code>**: Bu səhv şərtini simulyasiya edir. Sınaqlar üçün çox yararlı olur.
- **group/<group name>[@domain]**: Bu istifadəçi qrupuna zəng edir (4-cü başlıqda olan SIP və istifadəçi govluğu başlığına baxın)

Bunlardan bəzilərini sinaqdan keçirək. Əgər sizin FreeSWITCH serverdə qeydiyyatdan keçmiş telefonunuz mövcuddursa, **fs_cli**-da **originate** əmrini istifadə edin. **originate** üçün sadə sintaksis aşağıdakı kimidir:

```
originate <dialstring> <destination number>
```

Aşağıdakilarını yoxlayın və baxın görək nə baş verir. 1000 nömrəsini öz telefon nömrənizlə əvəz edin:

```
originate loopback/9664 1000
originate user/1000 9664
originate error/USER_BUSY 1000
originate loopback/9192 1000
originate loopback/4000 1000
```

Gördüyünüz kimi, zəngin yaradılması üçün FreeSWITCH-də çoxlu alətlər mövcuddur. Elə bir virtual ssenari mövcud deyil ki, bu imkan onu emal edə bilməsin.

Iki işləyən 1010 və 1000 nömrəli SIP telefonlar aşağıdakı sintaksislə bir-biri ilə əlaqələndirilir.

```
originate user/1010 1000
```

Nəticə

Bu başlığın tamamlanması yeni istifadəçi üçün çox əhəmiyyətlidir. Burada təqdim edilmiş mənqitin anlaşılan olması FreeSWITCH serverini quraşdırmaq və dəstəkləmək imkanının əhəmiyyətli bir hissəsidir. Müzakirə etdiyimiz başlıqlar aşağıdakılardır:

- FreeSWITCH XML Dialplanın başlanğıc ierarxiyası
 - Dialplan bir və ya bir neçə context-dən ibarət olur.
 - Context bir və ya bir neçə genişlənmədən ibarət olur.
 - Genişlənmələr bir və ya bir neçə şərt təşkil edir.
 - Adətən şərt-də bir və ya bir neçə action-lar və anti-action-lar olur.
- Müntəzəm ifadələr və başlığın tutuşdurulması
- Channel dəyişənləri məntiqinə giriş
- Dialplan sintaksis analizatoru və emaledicisinin neçə işləməsi
- Tələbimizə uyğun olan extension(genişlənmə) yaradılması
- Əksər ümumi və yararlı olan Dialplan programlarının siyahısı

Artıq bu bilgilərə sahib olduqdan sonra, siz bir telefon istifadəcisinin digəri ilə bağlantısının birləşdirilməsindən vacib olan daha da dərin işləri yerinə yetirə bilərsiniz. Dialplan həddən artıq güclü və nizamlanan olsa da, IVR mexanizmi və hansısa programlaşdırma dili ilə müqayisə edilə bilməz. FreeSWITCH-də Dialplan-la işləyən modullar mövcuddur ki, sizin işlək sisteminizə spesifik genişlənmiş imkanlar əlavə edə bilər.

Növbəti başlıqda biz FreeSWITCH-in zəng edən tərəf ilə əlaqəyə girməsinə kömək edəcək iki aspekte baxacayıq: *XML IVR altsistemi* və *FreeSWITCH phrase macros*

6-cı başlıq

XML IVR və Phrase macroslarının istifadə edilməsi

Daxili **IVR**(**I**nteractive **V**oice **R**esponse) motoru FreeSWITCH sistemində güclü komponentdir. Bu, mesajların oxunulması üçün şərait yaradır və interaktiv cavablar(adətən toxunma tonları)-i qayda ilə emal edir ki, zəngləri təyin edilmiş ünvanlara yönləndirsən. Bunun vasitə ilə zəng edən tərəfə canlı şəxsin cavab verilməsinə eytiyac qalmır və zəng edən rahatlıqla məlumatı eşidib, müəyyən funksiyaların işə salınması/dayandırılması ya da istifadəçi hesabının hesablanması üçün datanın daxil edilməsi və.s. işini seçim imkanı vasitəsilə edə bilir.

Əksər insanlar üçün IVR avtooperator kimi anlaşıılır hansı ki, sizin şirkət üçün əsas rəqəmə cavab verir və imkanlar opsiyasını təmin edir ki, şirkətinizdə olan şəxslərə çata biləsiniz(Misal üçün "Satış üçün 1, texniki dəstək üçün 2"). Bu öncədən nəzərdə tutulmayan zəng qəbuledicilərinin parcalanmasından qaçırm və ayrılmış **reception** işçisi tələbini azaldır. Daha da irəliləmiş IVR-lar həmçinin zəng edən tərəfdən məlumatın yiğilması üçün istifadə edilə(Məsələn zəng edənin hesab nömrəsi ya da konfrans korpusunun yaradılması üçün PIN rəqəmi). Bu başlıqda biz FreeSWITCH-in mənbə kodları ilə gələn daxili IVR motorunun iş prinsipini açıqlayacaqıq. Siz 5-ci başlıqda *XML Dialplanın başa düşülməsi* bacarıqlarınızdan, Dialplan vasitəsilə IVR programına zəngləri yönəltmək üçün istifadə edəcəksiniz və daxili quraşdırma faylları istifadə edərək FreeSWITCH üçün IVR menyu yaradacaqsınız.

Aşağıdakı mövzular müzakirə ediləcək:

- IVR motoruna qısa baxış
- IVR XML quraşdırma faylı
- IVR menyünün təyin edilməsi
- IVR menyusunun təyinat yerləri
- Zənglərin sizin IVR-a yönləndirilməsi
- IVR-a əlavələr
- IVR vasitəsilə phrase-ların istifadə edilməsi
- Təkmilləşdirilmiş yönləndirilmə

IVR motoruna qısa baxış

FreeSWITCH-in içində modul kimi əlavə edilən modullardan fərqli olaraq IVR FreeSWITCH-də olan daxili funksionallığıdır. Bu istənilən zamanda istifadə edilir, daxil ediləni oxuyur və rəqəmlər toplanır. Hətta siz IVR-i birbaşa Dialplan-dan istifadə etməsənizdə, siz IVR-la əlaqəli olan funksiyaları digər programlarda da görəcəksiniz. Misal üçün voicemail programı IVR funksionallığından mesajların oxudulması, silinməsinin idarəedilməsində rəqəmlərin gözlənilməsi, yadda saxlanılması və digər voicemail-lərin idarəedilməsində istifadə edir. Bu seksiyada biz yalnız IVR funksionallığına baxacayıq hansı ki, ivr Dialplan-ın içindən göstərilir. Bu funksionallıq adətən avto uyğunlaşma menyusunu yaratmaq üçün istifadə edilir amma, digər funksionallıqlar da mövcuddur.

IVR XML quraşdırma faylı

FreeSWITCH artıq IVR menyu nüsxəsi ilə öncədən quraşdırılmış gəlir. Adətən **5000** nömrəsinə yiğmaqla həmin menyunu işə salmaq olur. **5000** nömrəsinə yiğdiqda sizin FreeSWITCH-ə xoş gəlmənizin səslə qarşılanmanızı eşidəcəksiniz və menyu size təqdim ediləcək. Menyu opsiyaları FreeSWITCH konfrans-a zəng etmək, echo genişlənməsinə zəng etmək, gözləmədə musiqini dinləmək ya da alt menyuya gedişdən ibarətdir. XML nüsxəsinə baxaq.

```
/usr/local/freeswitch/conf/ivr_menus/demo_ivr.xml faylinı açırıq hansı ki,
aşağıdakı XML-dən ibarətdir:
<!-- demo IVR setup -->
<!-- demo IVR, Main Menu -->
<menu name="demo_ivr"
      greet-long="phrase:demo_ivr_main_menu"
      greet-short="phrase:demo_ivr_main_menu_short"
      invalid-sound="ivr/ivr-that_was_an_invalid_entry.wav"
      exit-sound="voicemail/vm-goodbye.wav"
      confirm-macro=""
      confirm-key=""
      tts-engine="flite"
      tts-voice="rms"
      confirm-attempts="3"
      timeout="10000"
      inter-digit-timeout="2000"
      max-failures="3"
      max-timeouts="3"
      digit-len="4">

<!-- The following are the definitions for the digits the user dials -->
<!-- Digit 1 transfer caller to the public FreeSWITCH conference -->
<entry action="menu-exec-app" digits="1" param="bridge
sofia/$${domain}!888@conference.freeswitch.org"/>
```

```

<entry action="menu-exec-app" digits="2" param="transfer 9196 XML
default"/>      <!-- FS echo -->
<entry action="menu-exec-app" digits="3" param="transfer 9664 XML
default"/>      <!-- MOH -->
<entry action="menu-exec-app" digits="4" param="transfer 9191 XML
default"/>      <!-- ClueCon -->
<entry action="menu-exec-app" digits="5" param="transfer 1234*256 enum"/>
<!-- Screaming monkeys -->
<entry action="menu-sub" digits="6" param="demo_ivr_submenu"/>
<!-- demo sub menu -->
<!-- Using a regex in the digits tag lets you define a dial pattern for the caller
     You may define multiple regexes if you need a different pattern for some reason -->
<entry action="menu-exec-app" digits="/^(10[01][0-9])$/" param="transfer
\$1 XML features"/>
<entry action="menu-top" digits="9"/>                                <!-- Repeat this menu -->
</menu>

<!-- Demo IVR, Sub Menu -->
<menu name="demo_ivr_submenu"
      greet-long="phrase:demo_ivr_submenu"
      greet-short="phrase:demo_ivr_submenu_short"
      invalid-sound="ivr/ivr-that_was_an_invalid_entry.wav"
      exit-sound="voicemail/vm-goodbye.wav"
      timeout="15000"
      max-failures="3"
      max-timeouts="3">

<!-- The demo IVR sub menu prompt basically just says, "press star to return to previous menu..." -->
<entry action="menu-top" digits="*"/>
</menu>

```

Öncəki misalda iki ədəd IVR menyu təyin edilmişdir: **demo_ivr** və **demo_ivr_submenu**. Gəlin IVR menyu təyinatı ilə başlayaraq IVR-in müəyyən bir hissəsini dağlıq və eynilə təhlil edək.

IVR menyunun təyin edilməsi

Aşağıdakı XML **demo_ivr** adlı IVR menyunu təyin edir:

```
<menu name="demo_ivr"
      greet-long="phrase:demo_ivr_main_menu"
      greet-short="phrase:demo_ivr_main_menu_short"
      invalid-sound="ivr/ivr-that_was_an_invalid_entry.wav"
      exit-sound="voicemail/vm-goodbye.wav"
      confirm-macro=""
      confirm-key=""
      tts-engine="flite"
      tts-voice="rms"
      confirm-attempts="3"
      timeout="10000"
      inter-digit-timeout="2000"
      max-failures="3"
      max-timeouts="3"
      digit-len="4">
```

Biz önce adını çəkdiyimiz menyunun adını zənglərin Dialplan-dan IVR-a ötürülməsində istifadə edəcəyik. Addan sonra, IVR-in özünü necə aparması üçün fərqli XML atributları təyin edilir. Aşağıdakı opsiyalar IVR-in özünü necə aparmasını təyin edərək mövcud olur.

greet-long

greet-long atributu abonent IVR-a çatan kimi başlanğıc IVR salamlaşmasını oxutma işini görür. Bu **greet-short** səs faylından fərqlənir hansı ki, yalnız "XYZ şirkətinə zəng etdiyiniz üçün təşəkkür edirik" kimi qısa girişlərin oxunulmasına şərait yaradır. Demo IVR-da **greet-long** atributu **Phrase-Macro**-dur hansı ki, zəng edən tərəfə giriş mesajını oxudur ("Welcome to FreeSWITCH"), menyu bəndləriylə müşayiət edilərək zəng edən tərəfə seçim imkanı yaradır.

Arqument sintaksisi: Səs faylin adı(ya da tam ünvan + ad), TTS ya da Phrase Macro.

Nüsxələr:

```
greet-long="my_greeting.wav"
greet-long="phrase:my_greeting_phrase"
greet-long="say:Welcome to our company. Press 1 for sales, 2 for support."
```

greet-short

greet-short atributu salamlaşma təyin edir hansı ki, zəng edən tərəf izin verilməyən məlumatı daxil etdiğdə və ya heç bir məlumat daxil etmədikdə oxumağa başlayır. Bu adətən girişsiz **greet-long**-da olan səs faylı ilə eyni olur. IVR nüsxəsində **greet-short** atributu **Phrase Macro**-dur hansı ki, menyu hissələrini zəng edən tərəfə oxudur və **greet-long**-da tapılan uzun girişini oxutmur.

Arqument sintaksisi: Səs faylı(ya da tam ünvan + ad), TTS ya da Phrase Macro.

Misallar:

```
greet-short="my_greeting_retry.wav"
```

```
greet-short="phrase:my_greeting_retry_phrase"
greet-short="say:Press 1 for sales, 2 for support."
```

invalid-sound

invalid-sound atributu zəng edən yalnız məlumat daxil elədikdə işə salınacaq səs faylini təyin edir.

Arqument sintaksisi: Səs faylinin adı(ya da tam ünvan + ad), TTS ya da Phrase Macro.

Misallar:

```
invalid-sound="invalid_entry.wav"
invalid-sound="phrase:my_invalid_entry_phrase"
invalid-sound="say:That was not a valid entry"
```

exit-sound

exit-sound atributu zəng edən tərəfin həddən artıq çoxlu yalnız daxiletməsində və ya çoxlu gözləmə vaxtının bitməsi hallarında işə salınacaq səs faylini təyin edir. Bu fayl IVR-dan çıxışdan önce oxunur(Zəng Dialplan-da davam edəcək).

Arqument sintaksisi: Səs faylinin adı(ya da tam ünvan + ad), TTS ya da Phrase Macro.

Misallar:

```
exit-sound="too_many_bad_entries.wav"
exit-sound="phrase:my_too_many_bad_entries_phrase"
exit-sound="say:Hasta la vista, baby."
```

timeout

timeout atributu IVR-in qarşılama mesajından sonra istifadəçinin rəqəmləri daxil etməsi üçün gözlədiyi maksimal vaxt sayını təyin edir. Əgər bu limit aşılırsa menyu **max-timeouts** atributunda olan sayadək təkrarlanacaq.

Arqument sintaksisi: Millisaniyələrlə olan istənilən rəqəm.

Misallar:

```
timeout="10000"
timeout="20000"
```

inter-digit-timeout

inter-digit-timeout atributu zəng edənin daxil elədiyi hər bir rəqəm arasında olan maksimal gözləmə vaxtını təyin edir. Bu tam timeout-dan fərqlənir. Bu abonentin çoxlu rəqəm daxil etməsi tələbi yarandıqda, kəsinti olmadan lazımi uzunluqda vaxtin təyin edilməsində çox yararlı olur. Misal üçün, əgər 1000 və 1 rəqəmləri IVR-in düzün verilənləridirse, sistem bunun son yazı olmasını təyin edənədək 1 rəqəmi daxil edildikdən sonra **inter-digit-timeout**-un bitməsini gözləyəcək.

Arqument sintaksisi: Millisaniyələrlə olan istənilən rəqəm.

Məsələn:

```
inter-digit-timeout="2000"
```

max-failures

max-failures atributu IVR-dan çıxmazdan öncə, yalnız daxil edilən verilənlərin təkrarlanma sayını təyin edir(Zəng Dialplan-da davam edəcək).

Arqument sintaksisi: Istənilən tam rəqəm.

Məsələn:

```
max-failures="3"
```

max-timeouts

max-timeouts atributu IVR-dan çıxmazdan öncə nə qədər timeout gözləyəcəyini təyin edir(Zəng Dialplan-da davam edəcək).

Arqument sintaksisi: Istənilən tam rəqəm.

Məsələn:

```
max-timeouts="3"
```

digit-len

digit-len atributu yazının tam olmasını təyin edənədək, istifadəçinin daxil edə biləcəyi maksimal rəqəm sayını təyin edir.

Arqument sintaksisi: Birinə bərabər ya da böyük olan istənilən rəqəm.

Məsələn:

```
digit-len="4"
```

tts-voice

tts-voice atributu istifadə ediləsi olan spesifik Text-To-Speech səsini təyin edir.

Arqument sintaksisi: Düzgün olan istənilən Text-To-Speech səsi.

Məsələn:

```
tts-voice="Mary"
```

tts-motoru

tts-motoru atributu istifadə ediləcək spesifik Text-To-Speech motorunu təyin edir.

Arqument sintaksisi: Düzgün olan istənilən Text-To-Speech motoru.

Məsələn:

```
tts-engine="flite"
```

confirm-key

confirm-key atributu mənsəb genişlənmə rəqəmini yoxlamaq üçün, istifadəçi sixabiləcək düyməni təyin edir. Bu **confirm-macro** atributu ilə birlikdə istifadə edilir.

Argument sintaksisi: Istənilən düzgün DTMF rəqəmi.

Məsələn:

```
confirm-key="#"
```

confirm-macro

confirm-macro atributu **confirm-key** düyməsi sixılanadək gözlənilən müddətdə hansı Phrase Macro-nun oxunmasını təyin edir. Təyin edildikdən sonra, zəng edənin mənsəb genişlənmə rəqəmini IVR-da daxil etməsindən sonra macro işə salınır.

Argument sintaksisi: Istənilən düzgün olan Phrase Macro.

Məsələn:

```
confirm-macro="my_confirm_macro"
```

Bu atributlar IVR-in ümumi davranışını diktə edir.

IVR menyusunun təyinat yerləri

IVR-in qlobal atributlarını təyin elədikdən sonra, siz zəng edən tərəfin sixa bilməsi üçün mövcud olan mənsəbləri(ya da opsiyaları) təyin etməlisiniz. Siz bunu **<entry>** XML elementi ilə edirsiniz. Gəlin bu IVR istifadə elədiyi ilk **6** XML opsiyalarına baxaq:

```
<entry action="menu-exec-app" digits="1" param="bridge
sofia/$${domain}/888@conference.freeswitch.org"/>
<entry action="menu-exec-app" digits="2" param="transfer 9196 XML default"/>
    <!-- FS echo -->
<entry action="menu-exec-app" digits="3" param="transfer 9664 XML default"/>
```

```

<!-- MOH -->
<entry action="menu-exec-app" digits="4" param="transfer 9191 XML default"/>
  <!-- ClueCon -->
<entry action="menu-exec-app" digits="5" param="transfer 1234*256 enum"/>
  <!-- Screaming monkeys -->
<entry action="menu-sub" digits="6" param="demo_ivr_submenu"/>
  <!-- demo sub menu -->
<!-- Using a regex in the digits tag lets you define a dial pattern for the
caller. You may define multiple regexes if you need a different pattern for
some reason -->
<entry action="menu-exec-app" digits="/^(10[01][0-9])$/" param="transfer $1
XML features"/>

```

Hər bir yazı ayrıca işi təyin edir hansı ki, zəng edən tərəf menyunun daxilindən mənimsədə bilər. Hər bir verilən üçün **3** parametr mövcuddur – götürüləcək **action**, zəng edənin bu işi yerinə yetirilməsi üçün sixacağı rəqəmlər və işə ötürüləcək parametrlər(**param** atributu). Əksər hallarda siz yəqin ki, **menu-execapp** işindən istifadə edəcəksiniz hansı ki, zəng idarə etməsini Dialplan parametrləri ilə yanaşı Dialplan programına köçürür(Sizin müntəzəm Dialplanda olduğunuz kimi, **bridge**, **transfer**, **hangup** və.s).

Öncəki misalda olan mövcud opsiyalar çox sadədirler – onlar bir rəqəm təyin edirlər(Misal üçün **digits = "3"**) hansı ki, sixildiqda zəngi bridge vasitəsilə birbaşa son nöqtəyə ötürür(1 rəqəmi) ya da zəngi Dialplan(2,3,4 və 5 rəqəmləri)-da olan genişlənməyə ötürür. Sonda 6 və 9 rəqəmləri IVR alt menyusundadır(Növbəti seksiyada açıqlanır).

Bir yazıda var ki, digərlərinən fərqlənir və IVR yazılarında sonuncudan əvvəldə durur. Ancaq buna elə də ciddi baxmağa ehtiyac yoxdur.

```
<entry action="menu-exec-app" digits="/^(10[01][0-9])$/" param="transfer $1
XML features"/>
```

Bu verilən təyinatı **digits** sütunu üçün regex-i bildirir. Bu müntəzəm ifadə sütunu sizin Dialplan-da istifadə elədiyiniz ifadələrlə identikdir. Bu misalda IVR **1000-1019** aralığında olan istənilən 4 rəqəmli genişlənməni axtarır(Bu rəqəmlər katalogda olan susmaya görə istifadə edilən genişlənmələrdir). Nəzərə alın ki, müntəzəm ifadələr mötərizə daxilində olur hansı ki, uyğun olan mənəni alır. Bu alınan məna spesifik **\$1** dəyişənində yerləşdirilir və transfer programında argument kimi istifadə edilir. Bu IVR-a effektiv şəkildə **1000-1019** aralığında olan verilənlərin qəbul edilməsinə və zəng edəni IVR-da daxil edildikdə birbaşa həmin genişləmlərə ötürülməsinə şərait yaradır.

Qalan IVR action yazıları biraz fərqlidir. Onlar alt-menyunu iş kimi təqdim edirlər hansı ki, zəng edəni IVR alt menyusuna və əsas menyuya ötürürler(Bu mövcud IVR-i yenidən işə salır və menyunu oxudur).

```
<entry action="menu-sub" digits="6" param="demo_ivr_submenu"/>
<entry action="menu-top" digits="9"/>
```

Bu iki verilən işlək IVR idarəetməsini yeni IVR-a ötürür. Nəzərə alın ki, bu üsulla alt-menyunun ötürülməsi ilə IVR tarixçəsini yaradırsınız. Həmçinin **go back** düyməsidə zəng edən üçün mövcuddur hansı ki, onları öncəki IVR-a qaytarır.

Digər işlər də mövcuddur ki, IVR çərçivəsində istifadə edilə bilər. IVR daxilində istifadə edilə biləcək tam işlərin siyahısı növbəti başlıqlarda açıqlanır.

menu-exec-app

param sütunu ilə kombinasiya edilmiş **menu-exec-app** işi xüsusi programı işə salır və siyahilanın parametrləri programaya ötürür. Bu sizin Dialplan-da istifadə elədiyiniz **<action application="app" data="data">** ekvivalentidir. **menu-exec-app** əsasən Dialplan-da zəng edəni digər genişlənməyə ötürəndə istifadə edilir.

Arqument sintaksis: **application <params>**

Məsələn:

```
<entry digits="1" action="menu-exec-app" param="application param1 param2
param3 ..."/>
<entry digits="2" action="menu-exec-app" param="transfer 9664 XML default"/>
```

menu-play-sound

Spesifik səs faylin işə salınması üçün **param** sütunu ilə birlikdə **menu-play-sound** istifadə edilir.

Arqument sintaksis: Keçərli səs faylidir.

Məsələn:

```
<entry digits="1" action="menu-play-sound" param="screaming_monkeys.wav"/>
```

menu-back

menu-back işi öncəki IVR menyusunu qaytarır, əgər istəniləndirsə.

Arqument sintaksisi: Heç biri.

Məsələn:

```
<entry digits="1" action="menu-back"/>
```

menu-top

menu-top işi IVR menyunu yenidən işə salır.

Arqument sintaksis: Heç biri.

Məsələn:

```
<entry digits="1" action="menu-top"/>
```

Zənglərin sizin IVR-a yönləndirilməsi

IVR-ı yaratdıqdan sonra siz hansısa bir üsulla daxil olan zənglərin ona yönləndirilməsini məcbur etməlisiniz. Zənglərin öz IVR-niza yönləndirilməsi çox asandır və sizin Dialplan-ın daxilindən edilə bilər. IVR-ı çağırmaq istədiyiniz yerdə aşağıdakı XML programını öz Dialplan genişlənmənizə əlavə edin:

```
<action application="ivr" data="demo_ivr"/>
```

Bu sətir FreeSWITCH-i məcbur edəcək ki, **demo_ivr** adlı IVR-ı axtarsın və onu çağırırsın.

demo_ivr-ı çağırmaq üçün XML Dialplan verilən-i FreeSWITCH nüsxə faylarının tərkibində gəlir və aşağıdakı kimidir:

```
<!-- a sample IVR -->
<extension name="ivr_demo">
    <condition field="destination_number" expression="^5000$">
        <action application="answer"/>
        <action application="sleep" data="2000"/>
        <action application="ivr" data="demo_ivr"/>
    </condition>
</extension>
```

Qeyd: Qeyd edin ki, öncəki misalda **sleep** programı, IVR yerinə yetirilməzdən önce işə salınır. Bu vacibdir ona görə ki, IVR salamlaşa oxumağa başlamazdan önce, o media-nı sizin zəng edən tərəflə FreeSWITCH prosesi arasında işə salmağa şərait yaradır

IVR-lara əlavələr

IVR-a əlavə edilmənin ya da artırılmanın iki üsulu mövcuddur. İlk üsulu öncə danışdığımız **SubMenu** sistem siyahısıdır. İki sadə IVR menyusunu elə yaradın ki, onlar bir-birlərindən asılı olmasın və hər birinin unikal adı mövcud olsun. Sonra əsas IVR-dan bir iş yaradın ki, alt menyu işini görsün və **param** sütunu alt səviyyə IVR-ın adını özündə təşkil eləsin.

```
<entry digits="1" action="menu-sub" param="child_ivr"/>
```

Öz menyularınızın bu üsulla yaradılmasının üstünlüyü ondan ibarətdir ki, siz zəng edənlərin geriyə öncəki IVR menyuya qayıda bilməsi üçün **menu-back** işini

istifadə etməyə imkan alırsınız. Yararlıdır o halda ki, sizin çoxlu valideyin zəng edənləri eyni alt menyunu çağırır.

Alt menyuların istifadə edilməsinin digər üsulu hər bir IVR-a unikal genişlənmə rəqəminin təyin edilməsi və sadəcə zəng edəni bir genişlənmədən digər genişlənməyə transfer etməkdir(səliqə ilə valideyn və bala menyulara çatmaq üçündür). Bu üsulla siz təminat verə bilərsiniz ki, IVR-in necə çağırılmasından asılı olmayaraq, bir IVR-dan digərinə və geriyə gedə bilərsiniz. Bu həmçinin sizin Dialplan-ı daha da qarşidurmaz edir və təyin edilmiş əlavə IVR-ların(onların genişlənmələrinə birbaşa zəng etməklə) istənilən zamanda bütün ağac boyu hərəkət etmədən sınaqdan keçirilməsini nəzərdə tutur.

IVR vasitəsilə phrase-ların istifadə edilməsi

Yəqin ki artıq görmüsünüz, **greet-long** və **greet-short** opsiyaları misallarda təyin edilmiş fayl və tam ünvanından fərqli olaraq **phrase: demo_ivr_main_menu** istifadə edir. IVR-lar size şərait yaradır ki, **phrase** və **Text-To-Speech** macros istifadə edərək səs fayllarını təyin edəsiniz. Bu fərqli səbəblərə görə istifadəyə yararlıdır: Ən əsas imkanı bir neçə səs faylini bir phrase-la birləşdirmək və zəng edənin informasiyalarına əsaslanaraq zəng edən tərəfə fərqli dillərin təqdim edilməsi imkanıdır.

Phrase Macros-a zəngin edilməsi

Phrase Macros-u Dialplan, IVR ya da Dialplan script tərəfindən(LUA scripti kimi, haqqında növbəti başlıqda danışılacaq) çağırıla bilər. Sonuncu növbəti başlıqda açıqlanacaq. Səs fayl adı istifadə edilən istənilən yerdə, Phrase Macros virtual olaraq istifadə edilə bilər. Phrase Macros yalnız işə salınmaq məqsədlərində istifadə edilir və bu halda onlar fayl adı təyin edilib yazılıma olan əməliyyatlarda istifadə edilə bilməz. Biz həmçinin öz XML IVR quraşdırma fayllarımızda phrase-alar nüsxələrinə baxdıq. Aşağıdakı bir neçə misal Dialplan-da Phrase Macros istifadə edilməsidir:

```
<action application="playback" data="phrase:myphrase:arg1:arg2:arg3"/>
<action application="play_and_get_digits" data="2 5 3 7000 #
phrase:myphrase:arg1 /invalid.wav my_var\d+/">
```

Nəzərə alın ki, orda argumentin olmasına tələb yoxdur. Aşağıdakı kodda həmçinin doğrudur:

```
<action application="playback" data="phrase:myphrase"/>
```

Indi gəlin bir neçə phrase-a baxaq görək bizim üçün nə edə bilər.

Phrase Macro nüsxələri - voicemail

Yadda saxlayın ki, FreeSWITCH voicemail sistemi IVR-in ağır istifadəçisidir. Bu həmçinin Phrase Macros-ların istifadə edilməsi üçün bir nüsxədir ki, qeyd edilmiş bir neçə səs faylinin təkrar istifadə edilməsinə şərait yaratır. FreeSWITCH-də olan voicemail-də Phrase Macros tətbiqinə baxdıqda biz öyrənə bilərik ki, bu mükəmməl alətlərin hamısını öz məqsədlərimiz üçün öyrənməliyik.

`/usr/local/freeswitch/conf/lang/en/vm/sounds.xml` faylinı hər hansıa bir XML mətn redaktorunda açın və analiz edin ki, `<include>` tag-ların daxilində ardıcıl `<macro>` tag-lar görəcəksiniz. Ancaq ardıcıl macroslara baxdıqda artıq onların nə etdiklərini anlaya bilərsiniz.

Phrase Macro üçün başlanğıc sintaksis aşağıdakı kimidir:

```

<macro name="<macro name>">
    <input pattern="<pattern>">
        <match>
            <action/>
        </match>
        <nomatch>
            <action/>
        </nomatch>
    </input>
    <input pattern="<pattern>">
        <match>
            <action/>
        </match>
        <nomatch>
            <action/>
        </nomatch>
    </input>
</macro>
  
```

Macro mətni `<macro>` və `</macro>` tag-ları daxilində təyin edilir. `input` `pattern` ifadəsi müntəzəm ifadədir hansı ki, Phrase Macro-ya gələn istənilən arqumətlə tutuşdurulur. Doğru tutuşdurulma olarsa, `<match>` və `</match>` tag-ların daxilində olan işlər eks halda isə, `<nomatch>` və `</nomatch>` tag-ların daxilində olanlar yerinə yetiriləcək. Əgər tutuşdurulma olarsa, spesifik alınan müntəzəm ifadə (`$1, $2` və.s.) `<match>` node-un daxilində mövcud olacaq. Nəzərə alın ki, sizin çoxlu giriş parametrləriniz ola bilər. Bu funksiyalar eynilə XML dialplan funksiyalarına oxşayırlar. Çoxlu `input pattern` node-ları istifadəsi üçün aşağıdakı koda baxın.

Gəlin adı macros nüsxəsinə baxaq. `voicemail_goodbye` macro olan ünvani `/usr/local/freeswitch/conf/lang/en/vm/sounds.xml` faylında tapın:

```

<macro name="voicemail_goodbye">
    <input pattern="(.*)">
        <match>
            <action function="play-file" data="voicemail/vm-goodbye.wav"/>
        </match>
    </input>
</macro>
  
```

Bu macro zəng edən sistemdən çıxış elədikdə, voicemail sistemi tərəfindən çağırılır. Bu halda daxil edilən nüsxə (*.*) olacaq hansı ki, həmişə uyğun olacaq hətta, phrase argumentsız çağırılsa da belə. Bu nüsxə Phrase Macros-da olan ümumi nüsxədir. Bir tərəfdən sizə mənasız görünə bilər ki, bir səs faylinin işə salınması üçün **7** sətir kod yazılır. Yalnız bu Macro-nun istifadə edilməsi ilə istifadəçinin voicemail-dən çıxış etməsini təyin edə bilirik və biz bunu mənbə kodlarında heç bir dəyişiklik etmədən edirik. Həmçinin başqa üstünlükler də var.

```
/usr/local/freeswitch/conf/lang/en/vm/sounds.xml faylında
voicemail_enter_pass macro yerləşən ünvanı tapın:
<macro name="voicemail_enter_pass">
<input pattern=".+">
<match>
<action function="play-file" data="voicemail/vmenter_pass.wav"/>
<action function="say" data="$1" method="pronounced" type="name_spelled"/>
</match>
</input>
</macro>
```

Nəzərə alın ki, bu macro argumentləri alır və onları spesifik **\$1** dəyişənində yerləşdirir. Nüsxə quraşdırmasında voicemail modulu # simvolunu argument kimi yollayır. Zəng edən tərəf voicemail-də qeydiyyatdan keçdiyi anda, bu macro dialoqu kontrol edir. Təyin edildiyi kimi, bu səs faylini işə salaraq deyir "**Please enter your password, followed by...**" və sonra **say** programını istifadə edir ki, "**pound**" sözünü desin. Bu macro bizə imkan verir ki, zəng edən tərəf şifrə daxil elədikdə nəyi eşitməsini təyin edə bilək.

Qeyd: voicemail-ə daxil olub mesajlarınızı yoxladığda **fs_cli** əmrini daxil edin və phrase-ların necə işlədiyini görə biləcəksiniz.

Bu hissədə biz gördük ki, zəng edən tərəfdə işə salmaq üçün, Phrase Macros-lar sayesində bütöv bir cümlə yaradaraq fərqli səs deyilişlərini birlikdə tikə bilək. Bunun klassik nüsxəsi IVR menyudur. VoiceMail əsas menyusu həqiqətən də sadəcə xüsusi məqsədlər üçün IVR menyudur. Bu dialogdir ki, zəng edənə deyir "**To listen to new messages, press one. To listen to saved messages, press two. For advanced options, press five. To exit, press pound**". Gəlin aşağıdakı macro-da baxaq.

```
/usr/local/freeswitch/conf/lang/en/vm/sounds.xml faylında voicemail_menu
macro-suna baxın aşağıdakı kimidir:
<macro name="voicemail_menu">
<input pattern="^([0-9#*]):([0-9#*]):([0-9#*]):([0-9#*])\$">
<match>
<!-- To listen to new messages --&gt;
&lt;action function="play-file" data="voicemail/vm-listen_new.wav"/&gt;
&lt;action function="play-file" data="voicemail/vm-press.wav"/&gt;
&lt;action function="say" data="$1" method="pronounced" type="name_spelled"/&gt;
&lt;action function="execute" data="sleep(100)"/&gt;

<!-- To listen to saved messages --&gt;
&lt;action function="play-file" data="voicemail/vm-listen_saved.wav"/&gt;</pre>

```

```

<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="say" data="$2" method="pronounced" type="name_spelled"/>
<action function="execute" data="sleep(100)"/>

<!-- For advanced options -->
<action function="play-file" data="voicemail/vm-advanced.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="say" data="$3" method="pronounced" type="name_spelled"/>
<action function="execute" data="sleep(100)"/>

<!-- To exit -->
<action function="play-file" data="voicemail/vm-to_exit.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="say" data="$4" method="pronounced" type="name_phonetic"/>
</match>
</input>
</macro>

```

Bu frazaların eksəriyyəti aydınlaşdırır. Əsas məlumat faktiki olaraq, seçilmiş nüsxədə olur. Voicemail modulu bu macro-nu göstərilən argument siyahısı ilə çağırır: **1:2:5#**. Daxil edilən nüsxə sadəcə müntəzəm ifadədir hansı ki, bu mənaları analiz edir belə ki, **\$1-də 1 olsun**, **\$2-də 2 olsun**, **\$3-də 5 olsun** və **\$4-də # olsun**.

Qeyd: Sual vermək olar ki, düymələrin sıxılması harda təyin edilir. Yəni FreeSWITCH voicemail moduluna zəng edənin **1** sıxanda yəni mesajlara, **2** sıxanda yadda saxlanılmış mesajlara və digərləri olan ünvan harda təyin edilir. Cavab **/usr/local/freeswitch/conf/autoload_configs/voicemail.conf.xml** faylındadır. Susmaya görə olan **profile** üçün **<profile>** node-una baxın. Nəzərə alın ki, eksər adı olan parametrlərin sonu **key**-lə bitir. Bunlar hamisi dəyişdirilə bilər. **play-newmessages-key** parametri istifadəçinin yeni mesajları dinləməsi üçün sıxacağı düyməni təyin edir. **config-menu-key** parametri istifadəçinin irəliləmiş opsiyalar menyusunu açmaq üçün sıxacağı düyməni təyin edir. Sınaqlar üçün fərqli dəyişikliklər edib yoxlamaqdan çəkinməyin. Sınaqlarınızı etməzdən önce nüsxə çıxarın ki, nəyisə səhv elədikdə asanlıqla bərpa edə biləsiniz.

Gəlin Phrase Macro istifadəsinin digər bir misalına baxaq ki, daha da çətin IVR ssenarilərində olan problemlər həll edə bilək. **voicemail_message_count** macro-su iki fərqli problemləri həll edir. İlk olaraq bizim iki fərqli tipli voicemail mesajımız var(yeni və saxlanılmış). Sonra bizim problemimiz vardır ki, zəng edənə nə qədər mesajının qalmasını dedikdə nə zaman çoxlu və nə zaman tək mesajları istifadə edəcəyik. Baxın görək **voicemail_message_count** macro-su bu problemi necə gözəl həll edir:

```

<macro name="voicemail_message_count">
  <input pattern="^(1):(.*$)" break_on_match="true">
    <match>
      <action function="play-file" data="voicemail/vm-you_have.wav"/>
      <action function="say" data="$1" method="pronounced" type="items"/>
      <action function="play-file" data="voicemail/vm-$2.wav"/>
      <action function="play-file" data="voicemail/vm-message.wav"/>
    </match>

```

```

</input>
<input pattern="^(\d+):(.*?)$">
<match>
  <action function="play-file" data="voicemail/vm-you_have.wav"/>
  <action function="say" data="$1" method="pronounced" type="items"/>
  <action function="play-file" data="voicemail/vm-$2.wav"/>
  <action function="play-file" data="voicemail/vm-messages.wav"/>
</match>
</input>
</macro>

```

Yenə də kodun eksər hissəsi aydınlaşdır və önceki misalda olduğu kimi açar başa düşüləcək hissə qalınlaşdırılmışdır. Vociemail modulu bu macronu **x:new** ya da **x:saved** arqumenti ilə çağıraraq, yadda saxlanılmış yeni ya da mesaj olaraq təqdim edir. Mesajların nömrəsi **\$1** vasitəsilə tutulur və sonra mesajın tipi(yeni ya da saxlanılmış olaraq) **\$2** dəyişənində saxlanacaq. Macro **\$2**-ni **voicemail/vm-new.wav** ya da **voicemail/vm-saved.wav** fayllarının nə zaman oxudulmasını təyin etmək üçün istifadə edir. Necə olardı ki, mesaj mesaja qarşı olsayıdı?

Qeyd edin ki, ilk daxil edilmənin **break_on_match** adlı əlavə bir atributu da var. Bu atributun **true** təyin edilməsi macro-ya deyəcək ki, marcos daxilində gələn digər daxiledilmələrə fikir verməsin. Əgər istifadəçinin bir yeni mesajı varsa onda, phrase bunu **1:new** arqumenti ilə çağıracaq(Uyğun olaraq yadda saxlanılmış mesaj üçün də o **1:saved** kimi çağıracaq). İlk başlıq uyğun olacaq və uyğunlaşma üçün daha başqa axtarış edilməyəcək. Sonra ilk **<match>** üçün işlər yerinə yetiriləcək. Bu halda Phrase Macro iki frazani birlikdə tikərək deyir, "You have ... one ... new ... message". Əgər orda birdən çox(ya da sıfır) mesaj olarsa, arqument **2:new** kimi olacaq. Bu halda ilk daxil edilən uyğunlaşma uğursuz olacaq və sonra o uyğun olan ikinci başlıqla davam edəcək. Bu **<match>** daxilində olan işlər "You have ... two ... new ... messages" frazاسını deyərək bəhrəsini verəcək. Daha da spesifik giriş başlıqlarını **break_on_match**-la **true** və ümumi giriş başlığını sonra istifadə elədikdə biz bütün dillərdə olan problemi sadə və gözəl bir üsulla həll edirik.

Bu prinsipləri yadınızda saxlayın ona görə ki, buna 7-ci başlıqda LUA ilə Dialplan Script yazma-da baxacayıq.

Təkmilləşdirilmiş yönləndirmə

IVR-lar yalnız menyulara limitlənmirlər. Misal üçün siz daha da çətin IVR strukturunu programlaşdırmaq istəyirsiniz. Bunu daxili XML IVR-la və ya başqa üsullar la da eləmək olar.

Misal üçün deyə bilərik ki, siz zəng edənin spesifik cavablama xidmətinə çatması üçün səliqə ilə PIN rəqəmini tələb edirsiniz. Siz yalnız tərkibində PIN rəqəmləri olan IVR yarada bilərsiniz və müraciət edilən PIN rəqəmi, yalnız daxiletmə səsi, yalnız şifrə səsi ilə salamlaşma səs fayllarını əvəz edin. Mənim nüsxəsi aşağıdakı kimi olacaq:

```
<menu name="enter_pin"
greet-long="phrase:enter_your_pin" invalid-sound="phrase:invalid_pin" exit-
sound="phrase:invalid_pin" timeout="15000" max-failures="3" max-timeouts="3">
<entry digits="1828" action="menu-exec-app" param="transfer after_hours XML
default"/>
</menu>
```

Bu səliqə ilə öncəki misalda olduğu kimi, **1828** şifrəsinin daxil edilməsini tələb edəcək və üç yalnız cəhddən sonra çıxaracaq.

Digər opsiya kimi, siz müəyyən datanı yığan və gələcək DialPlanınızda istifadə ediləcək olan IVR yarada bilərsiniz. Gəlin deyək ki, siz zəng edənin Caller ID-sinin daxil etməsini istəyirsiniz ki, onların növbəti çıxış zənglərində öz ID-ləri əmələ gəlsin. Siz IVR yarada bilərsiniz ki, **10** rəqəmi özündə yığır və nəticəni digər genişlənməyə ötürür, o da öz növbəsində rəqəmləri mövcud Caller ID-yə mənimsədir və sonda son mənsəbə gedir. Sizin IVR menyusu aşağıdakı kimi ola bilər:

```
<menu name="set_callerid" greet-long="phrase:enter_your_callerid" invalid-
sound="phrase:invalid_callerid" exit-sound="phrase:invalid_callerid"
timeout="15000" digit_len="10" max-failures="99" max-timeouts="99">
<entry digits="/^(\d{10})$/" action="menu-exec-app" param="transfer $1 XML
set_callerid"/>
</menu>
```

Sonra öz Dialplan-ınızda spesifik kontekst yarada bilərsiniz. Aşağıdakı kimi:

```
<context name="set_callerid">
<extension name="SetIt">
  <condition field="destination_number" expression="^(\d{10})$">
    <action application="set" data="effective_caller_id_number=$1"/>
    <action application="bridge" data="sofia/external/18005551212"/>
  </condition>
</extension>
</context>
```

Öncəki IVR kombinasiyası və köməkçi konteksti yalnız öz ID-lərini daxil edən zəng edənlərə izin verəcək və zəng müddəti **bridge** programına bunun üçün üstünlük təyin edir.

Notice

FreeSWITCH-də IVR sistemi zəng ediləndə daxil edilmə istəkləri üçün çox güclü və dinamikdir. Zəng edənlərin dinamik yönləndirmə və fərqli axınların yaradılma imkanı üçün FreeSWITCH-i digər programlarla əlaqələndirdikdə isə, bu imkanlar da sonsuz olur.

Artıq biz XML IVR sistemi və Phrase Macro sistemi açıqladıq. Gəlin öz diqqətimizi zəng edən tərəfin daha da çətin yönləndirilməsi və əlaqələndirilməsi üçün LUA Dialplan programlaşdırma dili ilə tanış olaq.

7-ci başlıq

LUA vasitəsilə Dialplan Script yazma

Öncəki başlıqda biz **Interactive Voice Response (IVR)** programının başlanğıc imkanlarını daxili **XML IVR motoru** istifadə edərək araşdırırdıq. XML IVR motoru statik sadə olan IVR proqramlarının düzəldilməsi üçün çox rahatdır. FreeSWITCH-in IVR-ların yaradılması üçün XML IVR motorundan daha da güclü olan proqramları mövcuddur. Bir üsulu FreeSWITCH-lə integrasiya edilmiş skript dilinin istifadə edilməsidir. FreeSWITCH səs proqramlarının qurulması üçün aşağıdakı skript dillərini dəstəkləyir:

- JavaScript
- Lua
- Perl

IVR proqramlarının qurulması üçün önce göstərilən dillərdən hər hansı biri istifadə edilə bilər. Bu başlıqda biz öz diqqətimizi yüngül çəkili olan LUA(<http://www.lua.org>) dilinə yönəldəririk ona görə ki, bu dil məhz digər proektlərlə birlikdə işlənilə bilmək üçün təkmilləşdirilir. Real misal olaraq WarCraft dünyasını deyə bilərik.

Oeyd: Hər bir script dilinin öz üstünlükləri və çatışmamazlıqları mövcuddur. Lua düzgün seçimdir ona görə ki, sürətlidir, genişlənəndir və öyrənilməsi çox asandır. Demək olar ki, Lua istənilən Dialplan üçün yazılaçq skriptlərdə düzgün seçimdir.

Bu başlıq aşağıdakı bölmələri açıqlayacaq:

- Lua ilə başlayaq
- Səs proqramlarının qurulması
- Irəliləmiş IVR konsepsiyaları
- Script tipləri

Lua ilə səs proqramlarının öz misallarımızda qurulmasında geniş şəkildə özümüzə uyğun olan marco frazalarından istifadə edəcəyik.

Lua ile başlayaq

Susmaya görə Lua nüsxə qurasdirmaları FreeSWITCH-də olur. Yeni adı standart kompilyasiya etsəniz artıq LUA olacaq. Buna əmin olmaq üçün isə, **fs_cli**-dan **lua** əmrini daxil etmək lazımdır. Aşağıdakı səhvi görməlisiniz:

```
freeswitch@internal> lua
```

-ERR no reply

Əgər sizdə **command not found** səhvi çap edilərsə bu o deməkdir ki, sizin sistemdə lua yoxdur və yenidən **mod_lua** modulla kompilyasiya edib yükleməlisiniz. Bunu **mod_flite** yüklədiyimiz qayda da edə bilərsiniz. [2-ci başlıqda](#) Linux/Unix/Mac OS X sistemlərində kompilyasiya və yüklənmə-də buna baxa bilərsiniz.

Lua dilini ayrıca olaraq FreeBSD maşına yüklemek istesəniz, aşağıdakı portdan istifadə edə bilərsiniz:

```
root@frfs:~ # cd /usr/ports/lang/lua52  
root@frfs:/usr/ports/lang/lua52 # make -DBATCH install
```

Lua scriptlerin Dialplan-dan işe salınması

Lua dialplan programı **<action>** tag-lərlə aşağıdakı sintaksisə uyğun olaraq işə salınır:

```
<action application="lua" data="my_script.lua arg1 arg2 arg3"/>
```

Scriptə ötürürlən argumentlər boşluqlarla ayrılırlar. Boşluq təşkil edən argumenti əlavə etmək üçün tək dırnaqdan istifadə eləmək lazımdır:

Əgər siz öz scriptinizi FreeSWITCH-in susmaya görə olan qovluğunda olan **scripts** qovluğuna yəni, **/usr/local/freeswitch/scripts/** ünvanına yerləşdirmisinizsə, onda tam ünvanı yazmağa gərək yoxdur. Ancaq ehtiyac olarsa tam ünvanı da yaza bilərsiniz. Misal üçün Linux/UNIX serverlərdə aşağıdakı kimi olacaq:

`<action application="lua" data="/full/path/to/my_script.lua"/>`

Windows maşınlarda isə aşağıdakı kimi olacaq:

```
<action application="lua" data="C:\full\path\to\my_script.lua"/>
```

Yazmağa başlamazdan önce gəlin qısa şəkildə Lua scriptlərin sintaksisinə baxaq.

Lua başlangıç sintaksisi

Lua sintaksisi həm öyrənmək və həm də yazmaq üçün çox asandır. Aşağıdakı sadə scriptdir:

```
-- Bu adı LUA scriptidir
-- İki tire simvolları ilə başlayan sətir, tək sətirli şərh deməkdir
-- [[
Çoxsətirli şərh.
ikiqat kvadrat mötərizələrin daxilində isə çoxsətirli şərhlər olur.
]]
-- Lua sadə yiğilir
var = 1 -- Bu şərhdir
var ="alpha" -- Digər şərh
var ="A1" -- Fikriniz var...
-- [[
Lua skripti Dialplan-dan çağırıldığda, sizin azca sehirli obyektləriniz olur.
Onlarda biri 'freeswitch' obyektidir hansı ki, aşağıdakı rahatçılığı yaradır:
freeswitch.consoleLog("INFO","This is a log line\n")
Digər vacib olanı isə, 'session' obyektidir hansı ki, size zəngin
manipulyasiyasına izin verir:
session:answer()
session:hangup()
]]
-- Lua geniş cədvəl istifadəsinə sahibdir
-- Cədvəllər massivlərin və assosiativ massivlərin hibrididir
val1 = 1
val2 = 2
my_table = {
    key1 = val1,
    key2 = val2,
    "index 1",
    "index 2"
}
freeswitch.consoleLog("INFO","my_table key1 is '" .. my_table["key1"]
<...'\n")
freeswitch.consoleLog("INFO","my_table index 1 is '" .. my_table[1]
<...'\n")
-- Yetki hüquqları keçdi
arg1 = argv[1] -- ilk argument
arg2 = argv[2] -- ikinci argument
-- Sade if/then
if ( var =="A1" ) then
    freeswitch.consoleLog("INFO","var is 'A1'\n")
end
-- Sadə if/then/else
if ( var =="A1" ) then
```

```

freeswitch.consoleLog("INFO","var is 'A1'\n")
else
  freeswitch.consoleLog("INFO","var is not 'A1'!\n")
end
-- Sətirlərin birləşdirməsindən istifadələr ..
var ="This" .." and ".. "that"
freeswitch.consoleLog("INFO","var contains '" .. var .. "'\n")
-- Son
  
```

Dialplandan yerinə yetirilən hər bir Lua scripti **session** obyekti qaytarır hansı ki, işə düşən zəng ayağını göstərir. session obyekti zənglərin idarə edilməsi üçün əsas üsuldur və LUA skriptlərində geniş istifadə edilir.

Səs programlarının qurulması

Artıq Lua scriptin başlanğıc sintaksisini öyrəndikdən sonra, gəlin adı Lua scripti və ona uyğun olan Dialnplan verilənlərini yaradaq. Yeni DialPlan genişlənməsi yaradaq hansı ki, **9910** nömrəsini yiğdiqdə LUA scriptini işə salacaq:

1. 5-ci başlıqda XML Dialplanın başa düşülməsi mövzusunda yaratdığımız **/usr/local/freeswitch/conf/dialplan/default/01_custom.xml** faylini açın və aşağıdakı genişlənməni əlavə edin:


```

<extension name="Simple Lua Test">
  <condition field="destination_number" expression="^(9910)$">
    <action application="lua" data="test1.lua"/>
  </condition>
</extension>
      
```
2. Faylı yadda saxlayıb **fs_cli** əmrini daxil edin və **reloadxml** əmrini də həmçinin konsol-dan daxil edin(Ya da **F6** düyməsi). Bizim dialplan artıq hazırlıdır ki, **test1.lua** adlı skripti çağırırsın. Bu yeni scripti aşağıdakı kimi yaradın.


```

2. Aşağıdakı sətirləri /usr/local/freeswitch/scripts/ qovluğunda test1.lua
adında fayla əlavə edin(Faylı yadda saxlayıb çıxın):
-- test1.lua
-- Zəngə cavab ver, köməkçini işə sal, dəstəyi üstə qoy
-- Ünvan ayıricısını təyin et
pathsep = '/'
-- Windows istifadəçiləri tərsinə slash istifadə edəcək:
      
```

```
-- pathsep = '\\'
-- Zəngə cavab ver
session:answer()
-- Səs faylinin adı və ünvanından sətir yaradaq
prompt ="ivr" .. pathsep .. "ivr-welcome_to_freeswitch.wav"
-- Jurnal mesajını çap elə(Yəni: ivr/ivr-welcome_to_freeswitch.wav)
freeswitch.consoleLog("INFO","Prompt file is '" .. prompt .. "'\n")
-- Köməkçini işə sal
session:streamFile(prompt)
-- Dəstəyi üstünə qoy
session:hangup()
```

Yuxarıda yazdığımız skript artıq sınadalarımız üçün hazırlıdır. FreeSWITCH-də qeydiyyatdan keçmiş hər hansıa telefon istifadə edərək **9910** nömrəsinə yiğsaq FreeSWITCH-ə giriş mesajını eşidəcəyik və sistem qoşulmayı kəsəcək.

Qeyd: Lua skriptində edilən dəyişiklikdən sonra **reloadxml** əmrinin daxil edilməsinə ehtiyac yoxdur. Skript faylı yadda saxlanılan kimi, Dialplandan çağırılan lua programı yenilənmiş skripti istifadə edəcək.

Gəlin bu skriptdə bir neçə sətirə baxaq və onların funksiyalarını açıqlayaq.

```
pathsep = '/'
```

Öncəki sətir pathsep adlı dəyişən yaradır və mənasını '/' (slash) simvolu təyin edir. Bu Linux/UNIX tipli sistemlər üçün ünvan ayıricısı olaraq istifadə edilir. Windows istifadəçiləri isə ünvan ayıricısı olaraq '\\' simvolundan istifadə edir.

Qeyd: Ünvan ayıricısının skriptlərdə istifadə edilməsi onun əməliyyat sistemləri arasında köçürülməsini daha da asan edir.

Sətir kodu zəngə cavab verir. Əksər skriptlər ilk işlərində zəngə cavab verir.

```
session:answer()
```

Sətir kodu **prompt** adlı dəyişən yaradır hansı ki, mövcud olan səs fayllarından birinə olan ünvanı təyin edir.

```
prompt ="ivr" .. pathsep .. "ivr-welcome_to_freeswitch.wav"
```

Sətir kodu FreeSWITCH-in konsoluna məlumat xarakterli informasiyani çap edəcək. Bu problemin araşdırılması üçün çox rahat bir imkandır. Əgər siz **9910** nömrəsinə zəng etdiyiniz müddətdə FreeSWITCH-in konsoluna baxsanız 2015-10-19 16:35:13.557606 [INFO] switch_cpp.cpp:1328 Prompt file is 'ivr/welcome_to_freeswitch.wav' sətirini görəcəksiniz.

```
freeswitch.consoleLog("INFO","Prompt file is '" .. prompt .. "'\n")
```

Qeyd: Əmin olun ki, freeswitch.consoleLog istifadə elədikdə sonda '\n' yeni sətir simvolunu təyin edirsınız.

Növbəti göstərilən kod sətiri **session** obyektinin **streamFile** metodunu istifadə edir ki, audio faylı zəng edən tərəf üçün oxutsun. Nəzərə alın ki, uyğun ünvan adının təyin edilməsində FreeSWITCH zəngin tezlik aralığına görə fərq

qoyur. Əksər hallarda o **8000** istifadə edəcək ona görə ki, aralıq nüsxəsi 8000Hz(8kHz) adı telefon zəngidir. Bu misalda səs faylı olaraq istifadə elədiyimiz Linux ünvani `/usr/local/freeswitch/sounds/en/us/callie/ivr/8000/ivr-welcome_to_freeswitch.wav` olacaq.

```
session:streamFile(prompt)
```

Qeyd: FreeSWITCH səs fayllarının tam siyahısı və onların tərkibini mənbə kodları olan ünvanda `docs/phrase/phrase_en.xml` faylından əldə edə bilərsiniz.

Sonda kod sətiri isə sadəcə dəstəyi üstünə qoyub zəngi dayandırır:

```
session:hangup()
```

Bu tip scriptlərin LUA ilə yaradılması çətin deyil. Gəlin bir script yazaq ki, zəng edən tərəflə müəyyən bir qarşılıqlı əlaqəyə girə bilsin.

Zəng edən ilə əlaqəyə girən IVR nüsxəsi

Ekser IVR programları zəng edən tərəfindən gələn müəyyən daxil edilən məlumatın çeşidlənməsini tələb edir. Misal üçün əksər istifadə edilən IVR programlarında zəng edəndən tələb edilir ki, PIN ya da hesab nömrəsini daxil etsin və sonra uyğun olan iş görülsün. Gəlin kiçik bir skript yazaq ki, o zəng edəndən daxil edilmiş müəyyən rəqəmləri özünə götürür və sonra onları iki səs metodu ilə oxuyur.

1. `/usr/local/freeswitch/conf/dialplan/default/01_custom.xml` faylinə aşağıdakı genişlənməni əlavə edin:


```
<extension name="Read Back Entered Digits">
  <condition field="destination_number" expression="^(9911)$">
    <action application="lua" data="read_back_digits.lua"/>
  </condition>
</extension>
```
2. Faylı yadda saxlayın `fs_cli` əmrini daxil edin və konsolda da `reloadxml` əmrini daxil edin. Artıq bizim dialplan `read_back_digits.lua` adlı scripti işə salmaq üçün hazırlıdır.

Yeni scripti aşağıdakı kimi yaradın:

1. `/usr/local/freeswitch/scripts/read_back_digits.lua` adlı fayl yaradın və tərkibinə aşağıdakı sətirləri əlavə edin:


```
-- read_back_digits.lua
-- Zəngə cavab ver
session:answer()
-- Ünvan ayırıcısını təyin eley
pathsep = '/'
-- Windows istifadəçiləri bu tip slash istifadə edəcək:
-- pathsep = '\'
-- Dəyişən təyin edirik ki, işə salınacaq səsi mənimsədir
prompt ="ivr" .. pathsep .. "ivr-
please_enter_extension_followed_by_pound.wav"
-- Yalnız edilən işin səsi işə salacaq dəyişəni təyin edirik
invalid ="ivr" .. pathsep .. "ivr-that_was_an_invalid_entry.wav"
-- Faylı oxut və rəqəmləri bir yerə topla
-- 'digits' dəyişəni yüksəlmış rəqəmləri özündə saxlayacaq
```

```
-- Ən azı 3 rəqəm və ən çox 5 rəqəm keçərilidir
-- İşin bitməsi üçün zəng edən '#' (diyez-pound) simvol daxil etməlidir
digits = session:playAndGetDigits(3, 5, 3, 7000,"#", prompt, invalid,
"\\d+")
-- Rəqəmləri ardıcıl olaraq say, de və sonra ara ver
-- Məsələn: "one two three four five"
session:execute("say","en number iterated " .. digits)
session:sleep(1000)
-- Rəqəmləri görünüyü şəkildə say və ara ver
-- Məsələn: "twelve thousand, three hundred forty-five"
session:execute("say","en number pronounced " .. digits)
session:sleep(1000)
-- Mədəni sağollaşma
thankyou = "ivr" .. pathsep .. "ivr-thank_you.wav"
goodbye = "voicemail" .. pathsep .. "vm-goodbye.wav"
session:streamFile(thankyou)
session:sleep(250)
session:streamFile(goodbye)
-- Dəstəyi üstüne qoy
session:hangup()
```

2. Faylı yadda saxlayın.

Script artıq sınaqlarımızı etmək üçün hazırlıdır. **9911** nömrəsinə zəng edin və sizə deyiləcək ki, genişlənməni daxil edin. Bir neçə rəqəm daxil et və **#** simvolunu daxil et. Sistem bu rəqəmləri geriyə ardıcıl şəkildə(bir iki üç dörd kimi) sayıb ara verəcək və sonra vahid olaraq(min iki yüz otuz dörd) sayacaq və sonda "**Thank you, goodbye**" deyib dəstəyi üstüne qoyacaq.
playAndGetDigits metoduda həmçinin yalnız daxil edilməni sizin üçün emal edəcək. Ulduz simvolu və iki rəqəmin daxil edilməsi size **invalid entry** sözünü deyəcək. Əgər zəng edən 3 dəfə ardıcıl yalnız daxil edərsə, **playAndGetDigits** qoşulmanı kəsəcək.

Şərtlər və dövrlər

Öncəki misallar zəng edənin adı dialogunu göstərir. Gəlin indi skript istifadə edək ki, onda şərt və dövr olsun. Biz həmçinin 6-cı başlıqda XML IVR və Phrase Macroların istifadə edilməsində öyrəndiyimiz kimi, geniş səslərin içində fərqli səslərin yığılması üçün yeni Phrase Macro tətbiq edəcəyik.

Gəlin birinci Phrase Macro yaradaq. Bize Phrase Macro lazımdır ki, seçilmiş səs fayllarını bir deyilişin içində birləşdirib desin "To continue, press one; to exit, press two". **/usr/local/freeswitch/conf/lang/en/demo/custom-phrases.xml** adlı fayl yaradırıq və tərkibinə yeni makro əlavə edirik.

- /usr/local/freeswitch/conf/lang/en/demo/custom-phrases.xml** faylı yaradıb tərkibinə aşağıdakı sətirləri əlavə edirik:


```
<macro name="read_digits2_phrase" pause="100">
<input pattern="(.*)">
<match>
<action function="play-file" data="voicemail/vm-continue.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="play-file" data="digits/1.wav"/>
<action function="execute" data="sleep(250)"/>
<action function="play-file" data="voicemail/vm-to_exit.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
```

```

<action function="play-file" data="digits/2.wav"/>
<action function="execute" data="sleep(250)"/>
</match>
</input>
</macro>
```

- Faylı yadda saxlayırıq **fs_cli**-a daxil oluruq və **reloadxml** (Ya da **F6**) edirik.

Artıq **read_digits2_phrase** adlı macro işləməyə hazırlıdır.

Qeyd: Yadda saxlayın istənilən zaman hər hansıa XML sənədlə işlədikdən sonra FS konsolda mütləq **reloadxml** əmrini (ya da **F6**) daxil etmək lazımdır. Konsoldan əmri daxil etmək yaxşı vərdişdir ona görə ki, siz yenidən yüklənmə müddətində çıxan səhvleri həmin anda da konsolda görəcəksiniz.

- /usr/local/freeswitch/conf/dialplan/default/01_custom.xml faylini açın və göstərilən genişlənməni əlavə edin:

```

<extension name="Read Back Entered Digits #2">
<condition field="destination_number" expression="^(9912)$">
<action application="lua" data="read_back_digits2.lua"/>
</condition>
</extension>
```

- Faylı yadda saxlayın və **fs_cli**-dan **reloadxml** (ya da **F6**) yerinə yetirin.

Bizim Dialplan artıq **read_back_digits2.lua** adlı lua skriptinin çağırılmasına hazırlıdır.

Aşağıdakı yeni skripti yaradın:

```

1. /usr/local/freeswitch/scripts/read_back_digits2.lua fayl yaradıb
tərkibinə aşağıdakı sətirləri əlavə edin:
-- read_back_digits2.lua
-- while loop ve session:ready() göstərir
-- Zəngə cavab verir
session:answer()
-- Ayırıcıni təyin edir
pathsep = '/'
-- Windows istifadəçiləri əksinə slash istifadə edir
-- pathsep = '\'
-- Sesi işə salacaq prompt adlı dəyişən təyin edirik
prompt = "ivr" .. pathsep .. "ivr-
please_enter_extension_followed_by_pound.wav"
-- Yalnız mesajları işə salacaq invalid adlı səs dəyişənin təyin edirik
invalid = "ivr" .. pathsep .. "ivr-that_was_an_invalid_entry.wav"
-- Davam eləmək ya da çıxməq üçün flaş təyin edirik
continue = true
-- while loop işə salırıq
-- Zəng edən tərəf donanadək ya da çıkış seçənədək dövr davam edəcək
while(session:ready() == true and continue == true) do
-- Səs faylini işə sal və rəqəmləri bir yerə topla
-- 'digits' dəyişəni toplanmış rəqəmləri özündə saxlayacaq
-- Ən azı 3 və ən çox 5 rəqəm daxil edilərsə düzgün olacaq
-- İşin bitməsi üçün zəng edən '#' (diyez-pound) simvol daxil
etməlidir
```

```

digits = session:playAndGetDigits(3, 5, 3, 7000, "#", prompt,
invalid, "\d+")
-- Rəqəmləri ardıcıl olaraq say, de və sonra ara ver
-- Məsələn: "one two three four five"
session:execute("say","en number iterated " .. digits)
session:sleep(1000)
-- Rəqəmləri göründüyü şəkildə say və ara ver
-- Məsələn: "twelve thousand, three hundred forty-five"
session:execute("say","en number pronounced " .. digits)
session:sleep(1000)
-- Zəng edəndən, davam edək ya çıxaq soruş?
digits = session:playAndGetDigits(1, 1, 2, 4000, "#",
"phrase:read_digits2_phrase", invalid, "\d{1}")
freeswitch.consoleLog("INFO","digits is " .. digits .. "'\n")
if (digits == "2") then
    continue = false
    freeswitch.consoleLog("INFO","Preparing to exit...\n")
end
end
-- Səliqə ilə dəstəyi asırıq
thankyou = "ivr" .. pathsep .. "ivr-thank_you.wav"
goodbye = "voicemail" .. pathsep .. "vm-goodbye.wav"
session:sleep(250)
session:streamFile(thankyou)
session:sleep(250)
session:streamFile(goodbye)
-- Dəstəyi asırıq
session:hangup()

```

2. Faylı yadda saxlayın.

Artıq sınaqlarımızı edə bilərik. FreeSWITCH konsolunda **/log 6** əmrini daxil edin çoxlu debug mesajları görsənməsin. Genişlənməyə zəng etdikdə FS konsola baxın. **9912** nömrəsinə zəng edin və geriyə oxuması üçün mənani daxil edin. Rəqəmi daxil etdikdən sonra, yeni səslə sizdən soruşulacaq ki, davam etmək üçün 1 çıxməq üçün isə 2 düyməsinə sıxın(Texniki olaraq 2 rəqəmindən başqa istənilən rəqəm skriptə davam edəcək). Hər iki opsiyanı yoxlayın və konsola baxın. Siz konsol jurnallarında skriptdən daxil etdiyiniz rəqəmləri görəcəksiniz.

Gəlin skriptimizdə iki seçilmiş sətiri analiz edək. Bu sətir while dövrü inisializasiya edir. Nəzərə alın ki, orda iki şərt **true** olmalıdır əks halda while dövrü çıkış edəcək. **session:ready()** metodu **true** olmalı və **continue** dəyişəni **true** olmalıdır. **session:ready()** metodu zəng edənin dəstəyi asıb-asmadığını təyin eləmək üçün yaxşı üsuldur. Zəng edən dəstəyi asdıqda **session:ready()** metodu true olmayan mənani mənimseyəcək. Digər şərt isə **continue** dəyişənini sınaq edir hansı ki, biz mənasını **true** təyin eləmişdik. O zəng edənin 2 düyməsinə sıxan müddətədək **true** olaraq qalır və 2 sıxıldığda isə **false** mənasını mənimseyir.

```
while(session:ready() == true and continue == true) do
```

Aşağıdakı sətir yoxlayır əgər zəng edən 2 düyməsini sıxarsa, səliqə ilə çıx. Əgər daxil edilən rəqəm ikidirsə, onda **continue** mənası **false** olur və **while** dövrü bunun nəticəsində çıkış edir.

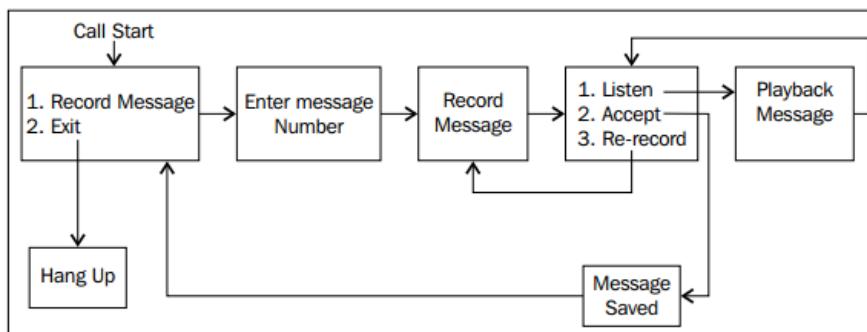
```
if (digits == "2") then
```

Qeyd: Nəzərə alın ki, **play_and_get_digits**-dən alınan məlumat sətir tiplidir tam rəqəm deyil. Zəng edən tərəfindən daxil edilən rəqəm * və # simvolları ola bilər.

Bu misal sizə şərtlərin istifadəsini və üstünlüyünü göstərdi. **session:ready()** metodу zəng edənin dəstəyi aslığı halda script vasitəsilə dövrdə işin dayandırılmasının əla yoludur.

Daha çox şərt və dövrələr

Gəlin daha da böyük iş görə bilən script yazaq ki, zəng edənə səsini yazma imkanı yaratsın. Bu skript istifadəçiyə rəqəmlər seriyası təklif edəcək hansı ki, zəng edənə daxil etdiyini yazmaq üçün fayl adlarında istifadə ediləcək, istifadəçiyə yazdığını qəbul eləmək ya da yenidən yazmaq və sonda zəng edənə digər yazılımanın daxil edilməsini təklif eləmək və ya skriptdən çıxışı təklif edəcək. Həmçinin fərqli sixılan açarların emal edilməsi üçün **setInputCallback** istifadə edəcək. Sonda biz iki ədəd yeni Phrase macro və bir ədəd önce yaratdığımızı istifadə edəcəyik. Kiçik zəng axını aşağıdakı diaqram kimi olacaq:



Dialplana genişlənmənin əlavə edilməsi ilə başlayaq:

1. `/usr/local/freeswitch/conf/dialplan/default/01_custom.xml` faylını açın və `<include>` tag-in daxilinə aşağıdakı genişlənməni əlavə edin:


```

<extension name="Record Sound Files Utility">
  <condition field="destination_number" expression="^(9913)$">
    <action application="lua" data="record_sound_files.lua"/>
  </condition>
</extension>
      
```

2. Faylı yadda saxlayın **fs_cli**-dan **reloadxml** (Ya da **F6**) əmrini daxil edin.

Artıq bizim dialplan hazırlıdır ki, **record_sound_files.lua** adlı skripti çağırırsın. Phrase macro-ları yaradaq. İlk Phrase Macro zəng edənə deyir ki, "To record a greeting, press one; to exit, press two". Nəzərə alın ki, bu phrase macro demək olar ki, öncəki misalımızda yaratdığımız **read_digits2_phrase** phrase macrosu ilə identikdir. Digər Phrase macro

istifadəçiye deyir ki, mesajın rəqəmini daxil edin(bu faylin adıdır) və diyez(#) sıxın. Aşağıda göstərilən qayda da iki yeni Phrase Macro əlavə edin:

3. `/usr/local/freeswitch/conf/lang/en/demo/custom-phrases.xml` faylini açın və tərkibinə aşağıdakı kodları əlavə edin:


```
<macro name="record_greeting_or_exit" pause="100">
<input pattern="(.*)">
<match>
<action function="play-file" data="voicemail/vm-to_record_greeting.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="play-file" data="digits/1.wav"/>
<action function="execute" data="sleep(250)"/>
<action function="play-file" data="voicemail/vm-to_exit.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="play-file" data="digits/2.wav"/>
<action function="execute" data="sleep(250)"/>
</match>
</input>
</macro>
```

```
<macro name="enter_file_number" pause="100">
<input pattern="(.*)">
<match>
<action function="play-file" data="ivr/ivr-please_enter_the.wav"/>
<action function="play-file" data="ivr/ivr-file.wav"/>
<action function="play-file" data="ivr/ivr-number.wav"/>
<action function="execute" data="sleep(250)"/>
<action function="play-file" data="currency/and.wav"/>
<action function="play-file" data="voicemail/vm-press.wav"/>
<action function="play-file" data="digits/pound.wav"/>
<action function="execute" data="sleep(250)"/>
</match>
</input>
</macro>
```

4. Faylı yadda saxlayın, `fs_cli`-dan `reloadxml`(Ya da **F6** sıxın) daxil edib enter sıxın.

Artıq `record_greeting_or_exit` və `enter_message_number` phrase macro-ları istifadəyə hazırlırlar. Sonda yeni skriptimizi aşağıdakı kimi yaradaq:

1. `/usr/local/freeswitch/scripts/` ünvanında `record_sound_files.lua` adlı skript yaradaq və tərkibinə aşağıdakı sətirləri əlavə edək:


```
-- record_sound_files.lua
-- İstifadəçiye bir və ya bir neçə səs faylları yaratmağa imkan yaradır
-- Səsler ${sounds_dir} qovluğunda saxlanılır
-- Yeni /usr/local/freeswitch/sounds qovluğunda
-- Yazılma müddətində yiğilmiş rəqəmlərin emal edilməsi üçün geriyə
qayıdış
function onInput (s, type, obj)
  if ( type == 'dtmf' ) then
    return"break" -- Bu yazılmayı dayandırır
  end
end
```

```

-- Zəngə cavab verir
session:answer()
session:sleep(500)
-- Ünvan ayırıcısını təyin edirik
pathsep = '/'
-- Windows istifadəçiləri əksinə slash istifadə edəcək:
-- pathsep = '\\'
-- Girişdə işə salınacaq səsin dəyişənini təyin edirik
prompt ="ivr" .. pathsep .."ivr-
please_enter_extension_followed_by_pound.wav"
-- invalid mesajı səsinin dəyişənini təyin edirik
invalid ="ivr" .. pathsep .."ivr-that_was_an_invalid_entry.wav"
-- Davam etmək ya da çıxış üçün flag təyin edirik
continue = true
-- Yazılma gedən müddətdə zəng edən hansısa rəqəm sixarsa nə iş
görülməsini təyin edirik
session:setInputCallback("onInput","")
-- while dövrü inisializasiya edirik
-- Dövr zəng edən dəstəyi asana ya da çıxış seçenekdək davam edəcək
while( session:ready() and continue ) do
    -- İlk menyu:
    -- 1 = Yazma
    -- 2 = Çıxış
    digits = session:playAndGetDigits(1, 1, 3, 7000, "#",
"phrase:record_greeting_or_exit", invalid,"\\d{1}")
    if (digits == "2") then
        continue = false
        freeswitch.consoleLog("INFO","Preparing to exit...\n")
    else
        -- Zəng edəndən gələn mesajın rəqəmini bir yerə yiğ
        -- 'digits' dəyişəni yiğilmiş rəqəmləri özündə saxlayır
        -- Ən azı 3 rəqəm və ən çoxu 5 rəqəm daxil edilə bilər
        -- Çıxış üçün zəng edən # (ya da diyez) simvolu sixır
        msgnum = session:playAndGetDigits(3, 5, 3, 7000,"#",
"phrase:enter_file_number", invalid,"\\d+")
        -- Faylin rəqəmini oxu
        session:execute("say","en number iterated " .. msgnum)
        session:sleep(1000)
        -- Yeni dövr: qəbul edilib ya yox
        accepted = false
        while ( session:ready() and not accepted ) do
            -- Səs faylini yaz
            session:streamFile("phrase:voicemail_record_message")
            -- Yazmaq üçün "bong" tone-a üstünlük verilir
            session:streamFile("tone_stream://v=-7;%100,0,941.0,1477.0;v=-
7;>2;+=.1;%1000, 0, 640") )
            filename = session:getVariable('sounds_dir') .. pathsep .. msgnum
            .. ".wav"
            session:recordFile(filename,300,100,10)
            -- Yeni dövr: Zəng edəndən soruş, qulaq as, qəbul elə ya da
            yenidən yaz.
        listen = true
        while ( session:ready() and listen ) do
            session:streamFile(filename)
            -- voicemail module-un rahat record_file_check macro-sunu
            istifadə edirik

```

```

local digits = session:playAndGetDigits(1, 1, 2,
4000,"#", "phrase:voicemail_record_file_check:1:2:3", invalid, "\\\d{1}")
if ( digits == "1" ) then
    listen = true
    accepted = false
    session:execute("sleep","500")
elseif ( digits == "2" ) then
    listen = false
    accepted = true
    -- Zəng edənə bildir ki, mesaj yadda saxlanıldı
    -- Qeyd: Siz bunu phrase macronun içində yerləşdirə
    bilərsiniz.
    session:streamFile("voicemail/vm-message.wav")
    session:execute("sleep","100")
    session:execute("say","en number iterated " .. msgnum)
    session:execute("sleep","100")
    session:streamFile("voicemail/vm-saved.wav")
    session:execute("sleep","1500")
elseif ( digits == "3" ) then
    listen = false
    accepted = false
    session:execute("sleep","500")
end -- if ( digits == "1" )
end -- while ( session:ready() and listen )
end -- while ( session:ready() and not accepted )
end -- if ( digits == "2" )
end -- while ( session:ready() ve continue )
-- Mədəni şəkildə
thankyou = "ivr" .. pathsep .. "ivr-thank_you.wav"
goodbye = "voicemail" .. pathsep .. "vm-goodbye.wav"
session:sleep(250)
session:streamFile(thankyou)
session:sleep(250)
session:streamFile(goodbye)
-- Dəstəyi as
session:hangup()

```

2. Faylı yadda saxlayın.

9913 nömrəsinə zəng edərək skripti sınaqdan keçirin. İlk menyu deyəcək "*To record a greeting, press one; to exit, press two*". **1**-i sıxırıq. Sistem sizdən sonra mesajın rəqəmini soruşacaq və ardınca **#** simvolu. **1234#** aralığında bir rəqəm daxil edin. Sonra sizə mesajın saxlanılması soruşulacaq. Mesajı yaz və sonra yazınızı dayandırmaq üçün istənilən simvolu daxil edin. Yazılmış səs yenidən oxuyacaq və sonra sizə son menyu açılaraq növbəti opsiyalarla soruşulacaq: qulaq as, qəbul elə ya da yenidən işə sal. Skriptin işləməsini tam anlamaq üçün bunlardan hər birini yoxlayın. Tələbinizə uyğun olacaq çoxlu səs yaza bilərsiniz.

Oeyd: Yazdığınız fayllar sounds qovluğunda yerləşir. sounds qovluğunun hansı ünvan olmasını tapmaq üçün **fs_cli**-dan **eval \${sounds_dir}** əmrini daxil edərək baxa bilərsiniz. Həmçinin nəzərə alın ki, **recordFile** metodу mövcud olan faylı siləcək və onun üstünə yenisini yazacaq(ehtiyatlı olun).

Gəlin skriptimizin bəzi açar hissələrini açıqlayacaq. **onInput** funksiyası ilə başlayaqq:

```
function onInput (s, type, obj)
    if ( type == 'dtmf' ) then
        return "break" -- Bu yazılmani dayandırır
    end
end
```

Bu funksiya sadəcə daxil edilmənin tipini yoxlayır və əgər istifadəçi DTMF rəqəmləri daxil edərsə **"break"** mənasını qaytarır. Bu funksiya yerinə yetirilərsə? Bu aşağıdakı kod hissəsi ilə six əlaqədədir:

```
session:setInputCallback("onInput", "")
```

setInputCallback metodu istifadəçinin rəqəmləri daxil elədiyində nə edilməsini təyin edir. Hər dəfə rəqəmlər daxil edildiyində, təyin edilən funksiya çağırılır(Nəzər yetirin ki, **session:playAndGetDigits** yerinə yetirildikdə bu mənimsədilmir hansı ki, özünə aid olan rəqəmləri emal edir). Bizim skriptimizdə **onInput** funksiyası zəng edən tərəf rəqəmləri yiğdiyi halda işə düşür.

Funksiya sadəcə **"break"** qaytarır hansı ki, həm yazmanı həm də oxumanı dayandırır. Bu o deməkdir ki, rəqəmi sixaraq yazını dayandırma imkanına yalnız siz sahib deyilsiz(Siz həmçinin oxunmada fərqli köməkçiləri də ötürə bilərsiniz). **9913** nömrəsinə yenidən yiğin və siz artıq səs menyusu "*Record your message at the tone*" eşitsəniz, rəqəm sixın ki, bu hissəni ötürəsiniz.

Diqqət yetirin ki, biz while dövrünün **1** cütünü əsas **while** dövrünün içində yerləşdiridik. Əsas **while** dövrü zəng edən **2** sixana ya da dəstəyi asanadək davam edəcək. Ortada olan **while** dövrü isə **accepted** dəyişəni **true** olduğu müddətədək davam edəcək. Daxildə olan **while** dövrü isə **listen** dəyişəni **false** olduğu müddətədək davam edəcək. Daxili **while** dövrü zəng edənə izin verir ki, öz yazdığını qəbul etməzdən önce istədiyi qədər qulaq asa bilsin və ortada olan while dövrü zəng edənə izin verir ki, istədiyi qədər öz qeyd elədiyi yazını silib üstünə yazzın. Kənar while dövrü isə zəng edənlərə şərait yaradır ki, istədikləri qədər səs faylları yaza bilsinlər.

Qeyd: Yəqin ki, hər bir **while** dövrünün içində **session:ready()** yoxlanışı olmasından xəbəriniz oldu. Bu zəng edənin ortada olan while dövrü yerinə yetirildiyi müddətdə dəstəyi asdıığı hallarda tələb ediləcək. Qayda olaraq, istənilən zaman sizin Dialplan skriptində while dövrü olarsa, siz statusu **session:ready()** vasitəsilə yoxlamalısınız. Bunun yerinə yetirilməməsi Lua skriptini zombi edə bilər və söndürülmüş sessiyanın mənasız yerə gözlənilməsinə gətirib çıxara bilər.

Script bütün səs fayllarını **.wav** faylları kimi saxlayacaq. Siz faylin tipini fərqli genişlənmə seçərək dəyişə bilərsiniz(Məs: **.ul** və ya **.gsm**). Ancaq FreeSWITCH-in developerləri əsaslı səbəbiniz yoxdursa, məsləhət görürər ki, **.wav** fayllardan istifadə edəsiniz.

Aşağıda başqa bir maraqlı sətir var:

```
session:streamFile("tone_stream://v=-7;% (100,0,941.0,1477.0);v=-7;>=2;+=.1;% (1000, 0, 640)"
```

Öncəki sətir FS daxilində olan **Tone Generation Markup Language (TGML)** istifadə edir ki, abonent üçün yazıya başlamazdan önce "bong" tonu işə düşsün. FreeSWITCH sizə geniş aralıqda tonlar istifadə edilməsi üçün imkan verir. Ətraflı məlumat üçün <https://freeswitch.org/confluence/display/FREESWITCH/TGML> linkinə baxa bilərsiniz.

Səs fayllarının kombinasiyasını istifadə edilməsi zəng edənin daxil etməsini toxunulan tonlar vasitəsilə etməsinə imkan verir. Misal üçün rəng edənin səsinin yazılması. Siz özünüzə aid olan fərqli səs programınızı yaza bilərsiniz.

Oeyd: Lua haqqında ətraflı oxumaq istəsəniz, https://freeswitch.org/confluence/display/FREESWITCH/mod_lua linkinə müraciət edə bilərsiniz.

Burda biz LUA script dili vasitəsilə XML IVR motorundə bir neçə nüsxəni göstərdik. Artıq daha da çətin konsepsiyalarda LUA scriptin üstünlüklerini göstərmək zamanıdır.

Irəliləmiş IVR konsepsiyaları

Programlaşdırma konstruksiyasında şərt və dövr olduğu kimi həmçinin başqa vacib funksionallıqlar da vardır. Bu funksionallıqlardan biri də IVR-in üçüncü tərəf baza ilə əlaqəyə girməsidir. Bəzi hallarda bu web baxış funksiyasına oxşayır. Başqa hallarda desək tələb ola bilər ki, hesab ya da ID rəqəmi ilə PIN bazaya sorğu edilir. Gəlin hər bir metod üçün bir neçə nüsxə misal çəkək.

LuaSQL vasitəsilə verilənlər bazasına qoşulaq.

LuaSQL interfeysi adı bir aralıqdır ki, LUA və DBMS arasında əlaqə yaradır (LuaSQL interfeysi KEPLER proekti tərəfindən yaradılmışdır). Daha ətraflı məlumat üçün <https://github.com/keplerproject/lua-compat-5.3> linkinə müraciət edə bilərsiniz).

Qeyd: Bu misalda sizdən verilənlər bazası ilə işləmək üçün müəyyən bilik tələb edilir. Misalımızda FreeBSD10.1 x64 üzərində PostgreSQL9.3 qurulmuşdur.

LUA-nı PostgreSQL verilənlər bazasına qoşmaq üçün, serverimizdə PostgreSQL verilənlər bazasını yükləyib işlək vəziyyətə gətirmək lazımdır. Bunu aşağıdakı addımlarda edirik.

1. İlk işimiz FreeBSD portlardan luasql-postgres-i yükleməkdir çünki, bu LUA-nın pgSQL-ə qoşulması üçün tələb edilir:

```
root@frfs:~ # cd /usr/ports/databases/luasql-postgres
root@frfs:/usr/ports/databases/luasql-postgres # make -DTABCH install
```
2. **luasql-postgres** paketi mütləq **postgresql93-client** tələb elədiyinə görə də biz postgresql93-server yükləyirik:

```
root@frfs:~ # pkg install postgresql93-server
```
3. PostgreSQL inisializasiyasını işə salırıq:

```
root@frfs:~ # /usr/local/etc/rc.d/postgresql initdb
```
4. **/usr/local/pgsql/data/postgresql.conf** faylında aşağıdakı sətirin qarşısından şərhi silirik:

```
listen_addresses = 'localhost'
```
5. **/usr/local/pgsql/data/pg_hba.conf** faylında **host all all 127.0.0.1/32 trust** sətirini dəyişib aşağıdakı kimi edirik:

```
host all all 127.0.0.1/32 md5
```
6. PostgreSQL daemonu işə salırıq:

```
root@frfs:~ # /usr/local/etc/rc.d/postgresql start
```
7. İşə düşməsini yoxlayırıq

```
root@frfs:~ # sockstat -l | grep postgres
pgsql    postgres    23313  3  tcp6      :::1:5432          *:*
pgsql    postgres    23313  4  tcp4      127.0.0.1:5432        *:*
pgsql    postgres    23313  5  stream   /tmp/.s.PGSQL.5432
```
8. Artıq pgsql istifadəçisi üçün şifrə təyin edirik:

```
root@frfs:~ # passwd postgres
Changing local password for postgres
New Password: pgsql_shifresi
Retype New Password: pgsql_shifresi_tekrar
```
9. pgsql istifadəçi adı ilə daxil oluruq, FreeSWITCH üçün istifadəçi və bu istifadəçinin qoşulması üçün verilənlər bazası yaradırıq:

```
root@frfs:~ # su postgres
$ createuser -sdrP fsuser
Enter password for new role: shifre
Enter it again: tekrar_shifre
```
10. **fsdb** adlı verilənlər bazası yaradırıq və **fsuser** istifadəçisini sahib edirik:

```
$ createdb fsdb --owner=fsuser

11. Konsoldan çıxırıq:
$ exit

12. PostgreSQL servisini yenidən işə salırıq:
root@frfs:~ # service postgresql restart
```

Verilənlər bazasını aşağıdakı şəkildə quraq:

1. Verilənlər bazası **fsdb**, istifadəçi adı **fsuser** və şifrə **freebsd** olan istifadəçi artıq yaradılmışdır.
2. **fsuser** adlı istifadəçi ilə **fsdb** adlı verilənlər bazasına qoşulaq.
`root@frfs:~ # psql -h 127.0.0.1 -p5432 -U fsuser fsdb`
3. **users** adlı cədvəl yaradırıq:
`CREATE TABLE users (
 name character varying(20),
 pin integer,
 acct integer,
 balance numeric(9,2),
 PRIMARY KEY(acct)
);`
4. Cədvələ müəyyən dataları əlavə edirik:
`INSERT INTO users(name, pin, acct, balance) VALUES('Anthony', 7654, 9898, 123.45);
INSERT INTO users(name, pin, acct, balance) VALUES('Michael', 9642, 1771, 0.00);
INSERT INTO users(name, pin, acct, balance) VALUES('Darren', 3756, 2316, 15.75);
INSERT INTO users(name, pin, acct, balance) VALUES('Raymond', 5825, 4639, 22.22);`

Sınaq üçün yaratdığınız istifadəçi adı və şifrə ilə verilənlər bazasına qoşulmağa çalışın əks halda lua qoşulmaq imkanına sahib olmayıcaq. Aşağıdakı qaydada yeni genişlənmə əlavə edin.

5. `/usr/local/freeswitch/conf/dialplan/default/01_custom.xml` faylinə aşağıdakı yeni genişlənməni əlavə edin:
`<extension name="Simple db connection">
<condition field="destination_number" expression="^(9914)$">
<action application="lua" data="db_connect.lua"/>
</condition>
</extension>`
6. Faylı yadda saxlayın **fs_cli**-dan **reloadxml** (ya da **F6**) əmrini daxil edin.

Bizim Dialplan artıq hazırdır ki, **db_connect.lua** adlı Lua scriptini çağırırsın. Script verilənlər bazasına qoşulma və SQL sorğu edilməsinin kiçik bir konsepsiyasını göstərəcək. Biz zəngdən hesab rəqəmini və PIN-i qəbul edəcəyik. Verilənlər bazasına sorğu yollayıb PIN-in doğruluğunu yoxlayacayıq və əgər doğrudursa, script müştəri balansını oxuyacaq. Gəlin skriptimizi aşağıdakı qaydada yaradaq.

7. /usr/local/freeswitch/scripts/ ünvanında **db_connect.lua** adlı skript yaradın və tərkibinə aşağıdakı sətirləri əlavə edin:

```

-- db_connect.lua
-- Verilənlər bazasına qoşulur, PIN-i yoxlayır və balansı oxuyur.
-- LuasQL yükləyir
-- local luasql = require "luasql.postgres"
luasql = require "luasql.postgres"

DBHOST = '127.0.0.1'
DBNAME = 'fsdb'
DBUSER = 'fsuser'
DBPASS = 'freebsd'
DBPORT = 5432

-- hangup funksiyası kodu biraz təmiz edir
function hangup_call ()
    session:streamFile("ivr/ivr-thank_you.wav")
    session:sleep(250)
    session:streamFile("voicemail/vm-goodbye.wav")
    session:hangup()
end

-- Tələb olunarsa sil
function close_db_conn()
    cur:close()
    con:close()
    env:close()
end

-- Verilənlər bazası mühiti obyekti yarat
env = assert (luasql.postgres())

-- Verilənlər bazası qoşulması obyekti yarat
con = env:connect(DBNAME,DBUSER,DBPASS,DBHOST,DBPORT)

-- Yalnız verilən səs faylı təyin edirik
invalid = "ivr/ivr-that_was_an_invalid_entry.wav"

-- Zəng edəni salamlayırıq
session:answer()
session:streamFile("ivr/ivr-hello.wav")

tries = 0
while (tries < 3) do
-- Hesab nömrəsini birlikdə toplayırıq
    acct = session:playAndGetDigits(3, 5, 3, 7000, "#",
"phrase:enter_file_number", invalid, ".+")

    -- Hesabı verilənlər bazasından dərtirir
    cur = assert(con:execute("SELECT * FROM users WHERE acct = '" ..
acct .. "'"))

    -- Nəticəni əlifba ardıcılılığında sütun adlarına görə indeks
    -- edilmiş şəkildə al
    row = cur:fetch ({}, "a")
  
```

```

-- Təsdiq et ki, biz yazını aldıq
if (cur:numrows() == 1) then
    -- Artıq hesab var, PIN-i bir yerdə cəmləşdir və yoxla
    tries = 0
    while (tries < 3) do
        pin = session:playAndGetDigits(3, 5, 3, 7000, "#",
"ivr/ivr-please_enter_pin_followed_by_pound.wav", invalid, "\d+")
        if (pin == row.pin) then
            bal = row.balance
            user_repeat = true
            while(user_repeat == true) do
                session:streamFile("voicemail/vm-you_have.wav")
                session:execute("sleep",200)
                session:execute("say", "en currency pronounced " ..
bal)
                session:execute("sleep",200)
                digits =
session:playAndGetDigits(1,1,3,7000,"#", "ivr/ivr-
to_repeat_these_options.wav",invalid,"\\d+") -- tekrarla y/n
                freeswitch.consoleLog("INFO","User entered '' ..
digits .. "'\n")
                if (digits == "1") then
                    user_repeat = true
                else
                    close_db_conn()
                    hangup_call()
                    break
                end
            end
        else
            -- Zəng edən yalnız PIN daxil etmişdir
            session:streamFile("ivr/ivr-
that_was_an_invalid_entry.wav")
            tries = tries + 1;
        end
    end
    if (tries > 2) then
        -- PIN-in daxil edilməsində çoxlu yalnız cəhd oldu
        session:streamFile("voicemail/vm-abort.wav")
        close_db_conn()
        hangup_call()
        break
    end
else
    -- Biz bu hesabi tapa bilmədik
    session:streamFile(invalid)
    tries = tries + 1;
end
end -- while (tries < 3)

if (tries > 2) then
    session:streamFile("voicemail/vm-abort.wav")
    close_db_conn()
    hangup_call()
end

```

8. Faylı yadda saxlayın.

9914 nömrəsinə zəng edərək yeni genişlənməni yoxlayın. Bazanıza əlavə elədiyiniz **4** rəqəmli hesab nömrəsini **9898** daxil edin və **#** simvolunu sıxın. Sonra həmin hesab nömrəsinin uyğun olan **7654** rəqəmli PIN-ni daxil edin və **#** simvolunu sıxın. Sistem daxil edilən hesab üçün bazada axtarış edəcək və şəxsə aid olan balansı səslə bildirəcək. Fərqli kombinasiyalarda mövcud olan/olmayan hesabla mövcud olan/olmayan PIN daxil etməklə fərqli sınaqlar keçirib scriptin özünü necə aparmasını təyin edin.

Gəlin bu scriptdə olan yeni imkanları açıqlayaq. Siz ilk dəfə gördünüz ki, biz bir neçə funksiyadan istifadə elədik. Onlar tələb edilmir. Amma kodu daha da oxunaqlı edə bilirlər. **hangup_call** funksiyası sadəcə zəngi bitirir. **close_db_conn** funksiyası açılmış verilənlər bazası qoşulmasını bağlayır. Bu funksiyalar bizim skriptimizdə fərqli ünvanlarda çağırılırdı ki, biz öz skriptimizdə həzin şəkildə çıxış edə bilək.

Verilənlər bazası qoşulması bir neçə sətirdə baş verir. Əsas olani seçilmişdir:

```
luasql = require "luasql.postgres"
```

Bu sətir **luasql.postgres** modulunu yükleyir. Sizin verilənlər bazası mühitinizdən asılı olara o **luasql.mysql**, **luasql.odbc** və hətta Oracle verilənlər bazası üçün **luasql.oci8**-də ola bilər.

Oeyd: Bəzi istifadəçilərin LuasQL-in MySQL verilənlər bazası ilə istifadəsində ciddi şəkildə yaddaşınitməsi problemləri olmuşdur. Belə hallarla qarşılaşmamaq üçün MySQL-i ODBC connector ilə istifadə eləmək məsləhətdir.

Aşağıdakı sətir verilənlər bazası mühiti obyekti yaradır:

```
env = assert (luasql.postgres())
```

Aşağıdakı sətirin biraz işi çoxdur çünkü, o bizim verilənlər bazasına qoşulmamız üçün qoşulma obyekti yaradır:

```
con = assert (env:connect("fsbook","fsuser","fspass", "localhost"))
```

connect işi üçün istifadə edilən arqumentlər aşağıdakı kimidir:

- Verilənlər bazasının adı
- İstifadəçi adı
- Şifrə
- Host ünvanı ya da IP ünvan

Əgər qoşulma uğurlu olarsa, qoşulma obyekti bizə izin verəcək ki, SQL müraciətləri hədəf bazamıza edə bilək:

```
cur = assert(con:execute("SELECT * FROM users WHERE acct = '" .. acct .. "'"))
```

Bu sətir əslində hədəfləndiyimiz verilənlər bazasında SQL SELECT funksiyasını users table-ında edir. O **cursor** obyektini qaytarır. **cursor** obyekti hədəflənən bazada yerinə yetirilmiş SQL şərtin nəticəsini göstərir. Bizim misalımız

cursor obyektini istifadə edir ki, verilənlər sətirini aşağıdakı kimi qaytarısın:

```
row = cur:fetch ({}, "a")
```

fetch metodu məhz iki arqument götürür ki, sətir obyektinin əldə edəcəyi Lua cədvəl adı (istəkdən asılıdır) və ya "a" ya da "n" indeks görünüşünü təqdim eləsin:

- "a" : Bu rəqəmli-hərfli indekslər mənasını verir, sütunlara məhz sütunun adı ilə müraciət olunacaq.
- "n" : Bu rəqəmli indekslər mənasını verir, sətirdə olan sütun pozisiyasında rəqəmli indekslər olaraq müraciət ediləcək.

table istəkdən asılı olan arqumentdir hansı ki, Lua cədvəlini sətirlərlə dolduracaq. Bizim misalda boş fiqurlu {} mötərizələrdən istifadə edirik ki,Lua cədvəlinin istifadə etməməyimizi göstərək. **fetch** metodu sətir datasını ya da başqa sətir data tapılmazsa **nil**(sıfır) qaytaracaq. **fetch** metodu programçıya izin verir ki, nəticələrin içinde hər sətir üçün bir dəfə dövr etsin. Bizim misalda biz **acct** sütununa əsaslanan sətir datasını seçmişik hansı ki, əsas açardır. Ona görə də bizim sətirdə ya ümumiyyətlə yazı olmayıacaq ya da bir yazı olacaq. Öz nəticələrimizi yenidən aşağıdakı kimi yoxlayacayıq:

```
if (cur:numrows() == 1) then
```

*PostgreSQL, MySQL və Oracle LuasQL driverləri **numrows()** metodunu dəstəkləyir.* Misalımızda bilmək istəyirik ki, həqiqətən də zəng edənin daxil elədiyi hesab rəqəmi olan dəqiq bir sətir datası almışığmı. Biz təsəvvür edirik ki, əgər müraciət dəqiq 1 sətir data qaytarırsa onda, zəng edən yalnız hesab rəqəmi daxil etmişdir. Eyni qayda ilə əgər hesab doğru daxil edilərsə, abonentin daxil etdiyi PIN-də yoxlanılır.

```
if (pin == row.pin) then
```

Bu zəng edənin daxil elədiyi PIN-in bazadan oxumasılına əmin olmaq üçün yoxlanış edir. Əgər bazada yoxdursa zəng edən PIN-i yenidən daxil edir. 3 yalnız cəhdən sonra script çıxış əmri yollayacaq.

Qeyd: Qoşulma və cursor obyektləri haqqında ətraflı məlumatı <https://github.com/keplerproject/luasql/commits/master> səhifisindən əldə edə biərsiniz.

Verilənlər bazasına LuasQL vasitəsilə qoşulma nisbətən asandır. Gəlin indi alternativ metod istifadə edərək kənar verilənlər bazasına qoşulaq.

CURL istifadə edərək WEB zəng edək

Bəzi hallarda tələb ola bilər ki, öz scriptimizdə WEB zəng edək. Bunun üçün istifadə edilən çoxlu programlar mövcuddur, misal üçün hava haqqında olan məlumat ya da VoIP programının web programla integrasiya edilməsi. Bu misalımızda biz sadəcə bir web zəng edəcəyik ki, hərbi dəniz donanması saytından U.S. üçün UTC vaxtını əldə edək. Biz həmçinin bir neçə yeni konsepsiya **freeswitch.API** obyekti, Phrase Macro-ya parametrlərin ötürülməsi,

Lua sətir idarə edilməsi funksiyaları sayəsində datanın açılması və başlıq tutuşdurulması haqqında danışacayıq.

Ilk işimiz mod_curl-in yüklənməsidir. Siz bunu 2-ci başlıqda Qurulma və yüklənmədə mod_flite üçün etdiyimiz eyni prosedurda edə bilərsiniz. Prosedur uyğun şəkildə yerinə yetirildikdən sonra isə aşağıdakı addımları edin:

1. Kompilyasiyadan önce **/root/src/freeswitch/modules.conf** faylında aşağıdakı sətrin qarşısından şərhi(diyez simvolu) silirik və faylı yadda saxlayırıq:
#applications/mod_curl
2. Kompilyasiyadan sonra **/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml** faylında aşağıdakı sətinin qarşısından **<!--** və sonundan **-->** simvollarını silirik:
<!-- <load module="mod_curl"/> -->
3. FreeSWITCH daemonu yenidən işə salırıq ki, dəyişiklikləri mənimsətsin:
root@frfs:~ # /usr/local/etc/rc.d/freeswitch stop
root@frfs:~ # /usr/local/etc/rc.d/freeswitch start
4. **fs_cli**-i işə salırıq və **help** əmrini daxil edib nəticəyə baxırıq. Əgər mod_curl uğurla yüklənmişdirse onda, siz curl-in API əmri kimi mövcud olmasını aşağıdakı sintaksislə görməlisiniz:
curl, curl url [headers|json|content-type <mime-type>|connect-timeout <seconds>|timeout <seconds>] [get|head|post|delete|put [data]], curl API, mod_curl

Sonra Dialplan-ımıza yeni genişlənməni əlavə edək:

1. **/usr/local/freeswitch/conf/dialplan/default/01_custom.xml** faylimizə aşağıdakı yeni genişlənməni əlavə edirik:
<extension name="Web Lookup">
<condition field="destination_number" expression="^(9915)\$">
<action application="lua" data="web-call.lua"/>
</condition>
</extension>
2. Faylı yadda saxlayın **fs_cli**-dan **reloadxml** və **ENTER** edin.

Artıq bizim dialplan **web-lookup.lua** adlı Lua scriptin çağırılması üçün hazırlıdır. Artıq gəlin yeni Pharse Macro yaradaq hansı ki, **hh:mm:ss** arqumentini alacaq və geriyə vaxtı oxuyacaq. Aşağıdakı addımları edin:

1. **/usr/local/freeswitch/conf/lang/en/demo/custom-phrases.xml** faylini açın və sonuna aşağıdakı sətirləri əlavə edin:
<macro name="simple_time" pause="50">
<input pattern="(\d\d):(\d\d):(\d\d)">
<match>
<action function="execute" data="sleep(250)"/>
<action function="say" data="\$1" method="pronounced" type="number"/>
<action function="execute" data="sleep(50)"/>
<action function="say" data="\$2" method="pronounced" type="number"/>
<action function="execute" data="sleep(50)"/>
<action function="play-file" data="currency/and.wav"/>

```

<action function="execute" data="sleep(50)"/>
<action function="say" data="$3" method="pronounced" type="number"/>
<action function="execute" data="sleep(50)"/>
<action function="play-file" data="time/seconds.wav"/>
<action function="execute" data="sleep(250)"/>
</match>
</input>
</macro>

```

2. Faylı yadda saxlayın və **fs_cli**-dan **reloadxml** əmri(Ya da **F6**) yazıb **ENTER** sıxın.

Artıq **simple_time** adlı phrase macro istifadə üçün hazırlanırdır. O vaxt arqumentini **hh:mm:ss** formatında alır və daxil edilən **(\d\d):(\d\d):(\d\d)** başlığına \$1, \$2 və \$3 dəyişənləri qaytarır hansı ki, uyğun olaraq saat, dəqiqə və saniyelərdən ibarət olur. Sonda LUA scriptini aşağıdakı kimi yaradacayıq:

- Mətn redaktoru istifadə edərək **/usr/local/freeswitch/scripts/** ünvanında **web-call.lua** adlı script yaradın.


```

-- web-call.lua
-- http://tycho.usno.navy.mil/cgi-bin/timer.pl saytına curl çağırış
edirik
-- Vaxt məlumatını açır və geriye zəng edənə oxuyur
-- Dəyişəni mənsəb URL təyin edirik
web_url = "http://tycho.usno.navy.mil/cgi-bin/timer.pl"
-- Zəng edənə oxuduğumuz vaxtin sayı
num_reads = 0
-- FreeSWITCH API obyektini alırıq
api = freeswitch.API()
session:answer()

while(session:ready() == true and num_reads < 10) do
    freeswitch.consoleLog("INFO","URL:  " .. web_url .. "\n")
    raw_data = api:execute("curl", web_url)
    freeswitch.consoleLog("INFO","Raw data:\n" .. raw_data ..
"\n\n")

    -- <BR>MMM. dd, hh:mm:ss UTC strukturuna uyğun olan sətiri axtar
    date_time = string.match(raw_data,"<BR>.-UTC",1)
    if (date_time == nil) then
        freeswitch.consoleLog("INFO","UTC date and time not
found\n")
    else
        freeswitch.consoleLog("INFO","UTC date and time is '"
.. date_time .. "'\n")
        -- Artıq seçilmiş elementləri kiçik sözlərlə parçala
        time = string.gsub(date_time,".-(%d+:%d+:%d+).+","%1")
        freeswitch.consoleLog("INFO","Time is '" .. time ..
"\n\n")
        session:streamFile("phrase:simple_time:" .. time)
    end

    num_reads = num_reads + 1

```

```

    session:execute("sleep","1000")
end

session:hangup()

```

2. Faylı yadda saxlayın.

Qeyd: Əgər siz vaxtı öz web serverinizdən alıb istifadə eləmək istəsəniz, sadəcə web server və php işləyən bir serverdə istənilən adla bir php script yaradıb tərkibinə aşağıdakı sətirləri əlavə edə bilərik(**/usr/local/etc/php.ini** faylında **date.timezone = 'Asia/Baku'** sətirini uyğun eləmək lazımdır).

```

<?php
    $now = date("H:i:s");
    echo $now;
?>
Eynilə web-lookup.lua scriptimizdə web_url=
"http://frfs.opensource.az/index.php" dəyişəni serverimiz üçün edirik.
Və əmrin çıxışı yalnız saatı qaytarlığı üçün while-da olan date_time =
raw_data dəyişəni uyğun olaraq edirik.

```

9915-ə zəng edin və qulaq asın. Script FreeSWITCH **curl** API istifadə edərək web axtarış edəcək. Əgər çağırış uğurlu olarsa sətir datası saat, dəqiqə və saniyələrə parçalanacaq və bundan sonra həmin mənalar **simple_time** Phrase Macrosunun içində kecid alacaq. Sonra bu Macro vaxtı geriyə zəng edənə oxuyacaq. **10** dövrdən sonra script çıkış edəcək. İstənilən zaman dəstəyi asa bilərsiniz.

Qeyd: Əmin olun ki, sizin FreeSWITCH serverin internet yetkisi mövcuddur. Əks halda **web-call.lua** skripti işləməyəcək.

Gəlin misalımızda olan yeni konsepsiyyaya nəzər yetirək. Aşağıdakı iki sətir koda diqqət yetirin:

```

api = freeswitch.API()
raw_data = api:execute("curl", web_url)

```

İlk sətir FreeSWITCH API obyektini yaradır hansı ki, bize öz skriptimizdən API əmrlərin ötürülməsinə şərait yaradır(Yadda saxlayın ki, API əmrlər FreeSWITCH-in konsolunda olan əmrlərlə eynidir). İkinci sətir sadəcə curl əmrini işə salır və URL-dən gələn nəticəni ələ keçirir. Script curl-dən aldığı emal olunmamış datanı çap edir hansı ki, ümumilikdə aşağıdakı kimi görünür:

```

!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final"//EN>
<html>
<body>
<TITLE>What time is it?"</TITLE>
<H2> US Naval Observatory Master Clock Time</H2> <H3><PRE>
<BR>Feb. 28, 06:39:03 UTC Universal Time
<BR>Feb. 28, 01:39:03 AM EST Eastern Time
<BR>Feb. 28, 12:39:03 AM CST Central Time
<BR>Feb. 27, 11:39:03 PM MST Mountain Time
<BR>Feb. 27, 10:39:03 PM PST Pacific Time

```

```
<BR>Feb. 27, 09:39:03 PM AKST Alaska Time
<BR>Feb. 27, 08:39:03 PM HAST Hawaii-Aleutian Time
</PRE></H3><P><A HREF="http://www.usno.navy.mil"> US Naval
Observatory</A>
</body>
</html>
```

Bütün bu verilənlər bir sətir mətn üzərində gəlir. Aşağıdakı sətir kodu UTC vaxtı ilə olan sətiri ayırır:

```
date_time = string.match(raw_data,"<BR>.-UTC",1)
```

date_time dəyişəni artıq **
 Oct. 21, 12:34:34 UTC** məlumatına sahibdir. **string.match** funksiyası başlıq tutuşdurmasını **raw_data** sətirinə mənimsədir. Tutuşdurduğumuz başlıq **
.-UTC** -dir. Bu başlıqda iki meta verilən mövcuddur. Dəqiq desək, period və tire. Digər simvollar hərflərdir. Adı dildə desək bu başlıqlar sətiri belə oxuyur - "sətirin əvvəli
 ilə başlayanı və UTC-yədək tapılan istənilən simvolları tutuşdur". Bu .- o deməkdir ki, tutuşa biləcək ən az simvolları uyğunlaşdır.

Qeyd: Lua sətir manipulyasiyası haqqında daha ətraflı <http://www.lua.org/manual/5.1/manual.html#5.4> linkindən oxuya bilərsiniz.

Artıq bizdə tək mətn sətri var. Biz vaxtdan saat, dəqiqliq və saniyəni ayırmalıyıq:

```
time = string.gsub(date_time,".-(%d+:%d+:%d+).+","%1")
```

Bu sətir **string.gsub** funksiyasını istifadə edir ki, təyin edilmiş sətirdə "tutuşdur və dəyişdir" funksiyasını yerinə yetirsən. **string.gsub** argumentləri tutuşdurma üçün sözlər, yenidən tutuşdurma üçün nüsxə və "dəyişdiriləcək" mənadır. Bizim misalda biz sətirdən **hour:minute:second** məlumatını ayırmak istəyirik. **strings.gsub** funksiyası daxil edilən sətirin hissəsi ya da tam tutuşdurlması işini görür və sonra dəyişdirilmiş mənanı qaytarır. Bu nüsxə tutuşdurulması və sətirin emal edilməsi tiplərinə görə digər dillərdən biraz fərqlənir. Ancaq effektivdir. Bu misalda istifadə elədiyimiz başlıq aşağıdadır:

```
.- (%d+:%d+:%d+).+
```

Bu nüsxə yoxlanışı sətirin əvvəlindən başlayır və vaxta çatanadək ola biləcək ən az simvollar götürülür. On vaxt mənasını (**hh:mm:ss**) "ələ keçirir" və sonra, uyğunlaşdırmanın sətirin sonuna çatanadək tutuşdurur. **hh:mm:ss** mənası **%1** kimi göstərilir. **%1**-in "dəyişdiriləcək" argument kimi yerləşdirilməsi **strings.gsub**-a deyir ki, yalnız tutulmuş bizi lazımlı olan datanı qaytar. Artıq bizim vaxt informasiyamız mövcuddur və biz onu Phrase Macro-muza aşağıdakı qaydada ötürə bilərik:

```
session:streamFile("phrase:simple_time:" .. time)
```

hh:mm:ss formatında olan vaxt mənası Phrase Macro üzərinə düşür hansı ki, orda da yenidən daxil edilən nüsxə olaraq aşağıdakı kimi tutuşdurulur:

```
<input pattern="(\d\d):(\d\d):(\d\d)">
```

Saat, dəqiqə və saniyə mənaları uyğun olaraq **\$1, \$2** və **\$3** kimi götürülür və geriyə zəng edənə oxudulur. Ssenari zəng edən özü dəstəyi üstünə qoymayanadək **9** dəfə təkrarlanır və sonra durur.

Bəzi hallarda tələb edilə bilər ki, sətir datasını kodlaşdırmaq və yenidən açmaq tələb edilə bilər. Aşağıdakı funksiya nüsxələrini siz webdən sətirlərin URL-kodlaşdırma və URL-dekodlaşdırması üçün istifadə edə bilərsiniz:

```
function urldecode (s)
    return (string.gsub (string.gsub (s, "+", ""), "%(%x%x)", 
        function (str)
            return string.char (tonumber (str, 16))
        end ))
end

function urlencode (s)
    return (string.gsub (s,"%W",
        function (str)
            return string.format ("%%%02X", string.byte (str))
        end ))
end
```

Mövcud vəziyyət üçün artıq biz anlayırıq ki, kənar məlumatların əldə edilməsi (verilənlər bazası üzərindən ya da Web üzərindən) nisbətən birbaşadır. Real iş mövcud istisnalar və verilənlərin alınması işinin emalı prosesindədir.

LUA başlıqları ya Müntəzəm Ifadələr (RegEX)

Texniki desək, LUA mənbədən olaraq müntəzəm ifadələri dəstəkləmir. Ancaq LUA nüsxə-tutuşdurulması sintaksisin ənənəvi **Perl Compatible Regular Expressions (PCRE)** ilə ümumi oxşarlıqları var. Aşağıdakı cədvəl LUA başlıqları və PCRE alternativləri göstərir:

LUA MetaSimvol	PCRE Metasimvol
.	.
+	+
-	*?
*	*
%	\

Aşağıdakı cədvəl LUA simvolu klasi və onun PCRE alternativini göstərir:

LUA Simvol Class	PCRE Simvol Class
%d	\d
%w	\w
%s	\s

Lua nüsxə sintaksisi sənədini tam şəkildə <http://www.lua.org/manual/5.1/manual.html#5.4.1> linkindən əldə edə bilərsiniz.

Kiçik praktika ilə, öncədən müntəzəm ifadələrlə işləyən istənilən şəxs effektiv LUA nüsxələri yaza bilərlər.

Script tipləri

DialplandanLua scriptlərlə işlədikdə bəzi məqamlar vardır ki, yadımızda saxlamalıyıq:

- Lua scripti bitdikdə, zəng avtomatik dayanacaq. Əgər siz Dialplan üçün emal işini davam etdirmək istəyirsinizsə əmin olun ki,
session:setAutoHangup(false) yerinə yetirmisiz. Aşağıdakı Dialplan hissəsinə baxın:

```
<condition>
  <action application="lua" data="my_script.lua"/>
    <!-- setAutoHangup false deyilsə aşağıdakı yerinə yetirilməyəcək -->
  <action application="transfer" data="$1 XML default"/>
</condition>
```

- Lua ssenarisində düzgün çıxış üsulu əmrləri bitirməkdir. Öncə dediklərimizdə ssenarıdan çıxış üçün heç bir açıq aydın əmr yox idi ancaq, indi **error()** funkisiyası istifadə edə bilərik ki, scripti məcburi dayandırıraq. Əgər **session:setAutoHangup(false)** təyin edilibsə, onda DialPlan prosesi davam edəcək.
- Öncəki hissəni yadda saxlayaraq nəzərə alın ki, calling **session:bridge()** ya da **session:transfer()** gözlədiyiniz kimi işləməyə bilər. Işin **bridge** ya da **transfer** edilməsi işi ssenari çıxanadək yerinə yetirilməyəcək. Növbəti kod fragməntinə baxın:
`freeswitch.consoleLog("INFO","Before transfer...\n")
session:transfer("9664 XML default")
freeswitch.consoleLog("INFO","After transfer...\n")`

transfer işi ikinci **consoleLog** çağırışı və ardıcıl kod sətirləri yerinə yetirilməyənədək baş verməyəcək. Əgər siz **bridge** ya da **transfer** işini görmək istəyirsinizsə, əmin olun ki, onlar sizin məntiqi scriptin sonunda yerinə yetirilir.

- Dialplan əvəzi olaraq LUA(ya da istənilən digər script dili) istifadə etməyin. Zənglərin yönləndirilməsi üçün XML Dialplan extremal şəkildə effektivdir və heç bir script dili onunla rəqabətə girə bilməz. Lua-nı zəng edənlə interaktiv əlaqəyə girmək üçün istifadə edin ya da hər

hansısa bir işin asanlıqla Dialplan vasitəsilə yerinə yetirilməsi mümkün olmazsa istifadə edin. Düzgün misal ondan ibarətdir ki, əgər bir işi Dialplan vasitəsilə görmək mümkündürsə onda onunla da edin.

- Dialplan-dan çağırılan scriptləri təkrar istifadə etməyin. Əgər siz istəsəniz ki, scriptlərlə zənglərin idarə edilməsi, hesablanması, üçüncü tərəfin idarə edilməsi və.s. işini görəsiniz o halda durun çünki, 10-cu başlıqda FreeSWITCH-in kənardan idarə edilməsi bölməsində **event socket** vasitəsilə heyvətəmiz işlər görəcəksiniz.

Notice

Lua səs programlarının zəng edənlərlə interaktiv əlaqə yaratması üçün elegant və sadə bir seçimdir. O çox sadədir və ona görə də, miqyası böyüdəndir. Öyrənilməsi sadə, asan sintaksis və online sənədə sahibdir.

Bu başlıqda biz çoxlu hədəflər yerinə yetirdik:

- Lua-nın əsas sintaksisi ilə tanış olduq və idarədilməsini öyrəndik
- Zəng edənlə əlaqəyə girmək üçün cavablama, dəstəyi asma, səs fayalarını oxutma, Phrase Macros-larını oxutma və zəng edəndən daxil etməni etməklə bir neçə script yazdıq.
- FreeSWITCH konsola Jurnal mesajlarını yollamaq üçün **freeswitch** obyektinin istifadə edilməsi və API əmrlərinin yerinə yetirilməsini öyrəndik.
- LuaSQL-in yüklenməsi və PostgreSQL verilənlər bazasına Lua script vasitəsilə qoşulmayı öyrəndik.
- **mod_curl**-in kompilyasiyası və susmaya görə yüklenməsini öyrəndik.
- Lua script vasitəsilə web çağrıları edilə bilməsi üçün **curl** müraciətlərin yollanılmasını göstərdik.
- Lua nüsxə tutuşdurulması sintaksisi ilə tanış olduq

Artıq bizdə zəng edənlə əlaqəyə girmək üçün script yazılmasının başlanğıc bilikləri var və Dialplana yenidən baxsaq yaxşı olar. 5-ci başlıqda XML Dialplan-ın başa düşülməsinə yenidən baxın və öz fərqli səviyyələr üçün fərqli dialplan seçin.

8-ci başlıq

İrəliləmiş Dialplan konsepsiyaları

Öncəki başlıqlarda siz bir az FreeSWITCH-də olan XML quraşdırma fayllarının istifadəsinin gücü haqqında öyrəndiniz. Ümumi quraşdırma təyinatlarının edilməsi üçün spesifik olaraq Dialplan verilənləri və XML istifadəsi haqqında öyrəndiniz. Bu başlıqda biz Dialplan strukturunun dərinliklərini, XML Dialplan emali sisteminin imkanlarını və çox əsas xarakteristikaları öyrənməklə çox çətin nəticələri əldə edə bilməni görəcəksiniz.

Bu başlıqda olan bəzi elementlər təkrar kimi görünə bilər amma, biz öncə yuxarıda danışdığınız başlıqlarda olan Dialplan funksiyalarına qayıtmalıyıq ki, Dialplan sistemin necə və nə üçün olmasını anlayaqq. Əksərən insanlar FreeSWITCH Dialplan-ı tam anlamadan istifadə edirlər. Bu başlıq sizi Dialplan çərçivəsində edilən işlər üçün ekspert etməyə hazırlaşır.

Başlıqda nəzərdə tutulur ki, sizdə artıq FreeSWITCH üzərində olan zənglərin yönləndirilməsi, zəng prosesinin işi və XML quraşdırılması haqqında biliklər mövcuddur. Bu başlığın oxunulmasına dək FreeSWITCH-in susmaya görə olan yüklənməsində demo telefonların qurulması və istifadəsi problemsiz işləyəcək.

Bu başlıqda aşağıdakı bölmələri açıqlayacaqıq:

- Dialplana yenidən baxış
- Ümumi Dialplan konsepsiyaları
- XML Dialplan moduluna analiz
- Tələlərdən qaçış
- XML Dialplan programları
- İstifadə edilən dəyişənlər
- Dəyişənlərin zəng başlıqları ilə ötürülməsi

Dialplana yenidən baxış

FreeSWITCH-də Dialplan motoru inanılmaz dərəcədə elastik bir program hissəsidir. Əgər sizin digər əlaqələndirici sistemləri ilə bilikləriniz mövcuddursa ola bilər ki, siz müəyyən **statements-you pre-program**(zənglərə cavab verir, faylları oxudur, rəqəmləri bir yerə toplayır və zəngləri yönləndirir) logik operatorlar Dialplan birləşdirmə konsepsiyaları ilə tanışsınız hansı ki, işi hər bir zəng üçün edir. İstənilən iş öncədən switch daxili mövcud olan logiki deyilişlərlə yerinə yetirilə bilmirsə, ümumiyyətlə yerinə yetirilməyəcək.

FreeSWITCH-də Dialplan emalı faktiki olaraq yüklənilən modullarla emal edilir. Bu modullarda olan məntiq hər dəfə zəng emal ediləndə istifadə edilir və siz öz məntiqi tələbinizdən asılı olaraq zəngin fərqli istiqamətlərdə getməsini etmək üçün çoxlu Dialplan modullarını yükleyə bilərsiniz. Bu FreeSWITCH-lə digər sistemlər arasında olan çox vacib fərqdir və adətən diqqətdən yayılır. Dialplan emalının modullu edilməsi ilə zənglərin yönəldirilməsinin daha azad bir forması ortaya çıxdı. Özünüzə aid olan şəxsi bir modul yaza ya da alternativ modullar istifadə edə bilərsiniz ki, sizin Dialplanın emal edilməsi üçün yeni çoxlu əmrlər açsın. Bu azadlıq digər keçiricilərlə müqayisə ediləndir hansı ki, sizə Dialplanın emal edilməsi üçün kənar scriptlərin yerinə yetirilməsinə şərait yaradır. FreeSWITCH size kənar script dillərinin istifadə edilməsinin əvəzinə, hər şeyi **C**-də saxlamağa onun daxili API-larını ya da link edilmiş kitabxanalarını istifadə(tələb edilərsə) üçün daha çətin bir integrasiya verir. Bu size sisteminizin resurslarından daha da az istifadə eləmək üçün çox rahat şərait verir.

Nə üçün Dialplan-ı modullu edirik? Vacibdir ki, önce sizdə Dialplan-ın niyə olmasını anlayasınız.

Gəlin müəyyən zaman ucun programlaşdırma haqqında unudaq və nəzər yetirek ki, hansı insanlar öz keçirici sistemlərini dəyişmək istəyirlər. Əgər siz böyük zəng axınızı qəbul edən əksər keçirici sistemlərə baxış keçirsəniz görəcəksiniz ki, əksər telefon zəngləri blok-sxem məntiqi struktur istifadə edir. Əgər siz əksər istifadəçilərdən soruşsanız ki, onlar öz telefonlarının necə aparılmasını istəyərdilər, onda cavab bu iş üçün yoxdur olacaq ya da cavab əksər hallarda əsas blok-sxemə çevrilmənin özü olacaq. Nə etmək istədiyinizdən asılı olmayaraq əgər siz qərarlar verən iş axını prosesinin diaqramını qurmusunuzsa, artıq öz Dialplan-nizi qurmusunuz. Faktiki olaraq, siz həmçinin Dialplan modulu emal edə bilecəyi olduğu həllər tələblərini nəzərə almısınız.



Gəlin ümumi zənglər axınının misalını götürək və onu hissələrə ayıraq. Bu misala yaxından baxdıqda görə bilərik ki, blok sxemi məntiqi axının strukturunu Dialplanın necə emal etməsini göstərir. Misal olaraq sizin hal-hazırda biznes üçün açıq olmanızı təyin edə bilmək üçün sizin Dialplan prosessorunun sistemin tarixi və vaxtına yetkisi olmalıdır ki, sizin işə açıq olan rahat günlərlə müqayisə işini görə bilsin. Zəng edən tərəfin 1 rəqəmini sıxmasını təyin edə bilmək üçün sizin Dialplan-in toxunulan tonlarını interpretasiya eləmək imkanı olmalıdır. Təyin edilən şərtlərə əsasən siz zənglə nə edəcəyinizin qərarına sahib olmalısınız – yönəldirmə, faylı oxuma, dəstəyi asma və.s. Bütün bunlar məntiqi və sintaksis əmrlərinə sahibdir hansı ki, Dialplan yerinə yetirir. Bəzi sistemlərdə bu tip qərarların kod yazıları qəribə forma ala bilər və nəticədə tam qarışqlıq olacaq (Müəyyən məhdudiyyətlər yaradır). FreeSWITCH-də məntiq fərqli dillərdə ola bilər ya da özünüzə uyğun olan yaza bilərsiniz.

Geriyə tarixə baxsaq heç bir Dialplan əmrləri interpretasiyası unikal olmayıb. Bir sistemdən digərinə keçidlə insanların öz əmrlərini strukturlaşdırması üsulu da dəyişirdi. FreeSWITCH developerləri bunu öncədən görürdülər və qərara aldılar ki, Dialplan-in özünü modullu şəklində etməsi vacibdir.

Saysız Dialplan modullarının yüklənilə bilən olmasının texniki üstünlükleri çoxdur. İlk olaraq keçirici sistem özü qayda ilə Dialplan quraşdırmasını götürür sonra, istənilən kənar kitabxanalara (SQL kitabxanalar, YAML kitabxanalar, CURL HTTP kitabxanalar və.s) link təqdim edə bilir və onu FreeSWITCH-ə zəngin emal etməsi üçün gözlənilən məntiqi dizayn içəinə əlavə edir. Ikinci isə Dialplan emalı modulu sistemin digər hadisə idarə edən modullarına da güvənə bilər hansı ki, sayəsində zəngin instruksiyalarını uzaq WEB serverdən ala bilərsiniz.

Növbəti üstünlükler odur ki, zəngləri açıq şəkildə yönəldirmək üçün məntiqi yüklənmiş modulun içində dəyişdirə bilərsiniz hansı ki, C programlaşdırma dilinin (FreeSWITCH yazıldığı) quruluşuna əsaslanır. Bu size FreeSWITCH üzərində qurmaq əvəzinə öz Dialplan prosesinin qurulmasına imkan verir. Sizin öz alt programlarınız sistemdə olan kitabxanaları C bazlı login API-ları

birgə əlaqələndirə (Məsələn: SQL) bilər. Misal üçün siz rahatlıqla SQL verilənlər bazasına sorğu yollaya bilərsiniz ki, istifadəçinin öz zəngini proxy edilməsini istəməsini öyrənəsiniz və sonra birbaşa FreeSWITCH API çağırıa bilərsiniz ki, proxy-ni işə salsın (Bütün bunlar bir neçə C-də yazılmış kodlarla Dialplan prosessorunda yerinə yetirilir). Bu sizə böyük dinamik mühit yaradır ona görə ki, heç bir kənar proses və resurs istifadə eləmədən öz Dialplan-nızı (Misal üçün verilənlər bazası SQL-dən true/false mənasının alınmasını PHP ya da SHELL scriptlə edə bilərsiniz) emal edə bilərsiniz. Bu FreeSWITCH-ə daha böyük zənglərin emal edilə bilməsi şəraitini yaradır.



Əsas yalnız düşüncə ondan ibarətdir ki, FreeSWITCH Dialplan XML-ə əsaslanması və onun tələb edilməsidir. Bu doğru deyil. Əgər siz adı fayllara üstünlük verirsinizsə onda öz Dialplan quraşdırmanızı orda saxlaya bilərsiniz. Əgər YAML-a üstünlük verirsinizsə, onu da istifadə edə bilərsiniz. Siz yalnız FreeSWITCH-də istifadə eləmək istədiyiniz düzgün olan C bazalı Dialplan modulunu yükləməlisiniz ki, saxladığınız məntiqi təyin edilmiş quraşdırma faylı tipi üçün interpretasiya eləsin.

FreeSWITCH-də daha geniş yayılmış Dialplan emali mexanizmi hələ də XML modulu üsulu ilədir. FreeSWITCH tərəfindən yayımlanmış Dialplan nüsxələri və şəbəkədə paylaşılmış əksər Dialplan nüsxələri XML-dədir. Bu səbəbdən də biz bu başlıqda diqqətimizi XML-ə yönəldirəcəyik. Özünüzə aid olan C modullarının yaradılması bu kitabın çərcivəsini aşır amma, siz bilməlisiniz ki, belə bir imkan var. Siz FreeSWITCH-i həddən artıq inkar elədikdə görə bilərsiniz ki, mövcud XML Dialplan prosessoru artıq sizin tələbləri qarşılıya bilmir və bu halda yadda saxlamalısınız ki, FreeSWITCH sizi yalnız XML ilə məhdudlaşdırır.

Spesific XML Dialplan konsepsiyalarına daxil olmadan, gəlin önce ümumilikdə Dialplan konsepsiyalarını açıqlayaq.

Ümumi Dialplan konsepsiyaları

Gelin qısa şəkildə 5-ci başlıqda XML Dialplan-ın başa düşülməsi mövzusunda danışdığımız konsepsiyaları açıqlayaq. Ümumi şəkildə desək, Dialplan iş siyahısının generasiya edilməsinə köməklik göstərir ki, zəng edən danışmaq istədiyi şəxs və ya şəxslərə çata bilsin. Dialplan modulu qərar vermə prosesini realizasiya edir və bu da işin görülməsinə gətirir. Dialplan modulu zəngin necə emal edilməsi üçün istənilən məqsədin realizasiya edilməsində azad olduğu halda **3** konsepsiaya sahibdir. Bu **3** məntiq hər bir zəngdə eyni 3 sualın verilməsi ilə qırıla bilər:

- **Contexts:** Harda ki, biz ümumi çata biləcəyimiz ərazinin ümumi siyahısını ya da funksiyasını axtarırıq hansı ki, mövcud zəng edən tərəfə çatmağa şərait yaradır?
- **Şərtlər:** Zəng edənin çatmaq istədiyi şəxsin kim olduğu?
- **Actions:** Qarşı tərəfə çatdıqda nə iş görülməlidir?

Bu 3 sual adətən 3 konsepsiyanın daxilində yönləndirmə istəyinin qırılması ilə cavablandırılır - zəng edənin konteksti, şərti tutuşdurulması və işlər. Bu anlayışlar FreeSWITCH üçün mütləq şəkildə unikal deyil. Biz bu başlıqda onların hər birini ayrıraqda analiz edəcəyik.

Istənilən Dialplan həlli üçün nəticə çoxlu işlər yaradır. Hər bir Dialplan modulu A tərəfindən B tərəfinə çatmış uğurlu nəticələri biri digərindən sonra yerinə yetirilən işlər siyahısını qaytarmalıdır.

Contekstlər

Biz kontekst haqqında danışdıqda, həqiqətən də zəng edənin çatmasına izin verdiyi seçilmiş mənsəblər siyahısına istinad edirik. Bu hissələr ayrılmış insanlar ola bilməz: onlar interaktiv imkanlara sahib olan insanlar qrupu ola bilər (Misal üçün voicemail) ancaq nəticə olaraq, zəng edən kiməsə ya da nəyəsə qoşulmaq isteyir. **Contekst** qaydalar toplusudur hansı ki, zəng edənin çatmaq istədiyi ünvanın və onlara istədiyi mənsəbə çatmağa izin verməsinin və ya verməməsinin təyin edilməsinə kömək edir. Bu qaydalara "**extensions**" deyilir.

Zəng edənin konteksti haqqında məntiqi operatorların tam qrupu olaraq, zəng edilə bilən ümumi mənsəblər kimi düşünülə bilər. Dialplan nüsxələrində əksər istifadə edilən kontekstlər "**internal**" və "**public**" kontekstlərdir. "**internal**" konteksti adətən daxildən istifadəçilər tərəfindən edilən zənglərə və ya sistem daxili edilən zənglərə müraciət edir (Misal üçün işçilər mərkəzi binanın ofis daxilində əyləşirlər). Bu insanlar digərləri ilə **4** rəqəmli nömrələri yığmaqla danışa bilərlər ya da kənar nömrələrə **9** rəqəmi yığıb sonra zəng edə bilərlər. "**public**" kontekstinə isə sistemdən kənar istifadəçilərin zəng edilməsində istinad edilir. Bu insanlar yalnız sistem daxilində təyin edilmiş son ünvanlara zəng edə və ya seçilmiş şəxslərə zəng etməyə bilər.

Ümumilikdə nə üçün bizim mənsəb qruplarımı var? Nə üçün sadəcə daxili və dünya nömrələri yoxdur? Əksər şirkətlər buna baxdıqda daha genişlənə bilən tələb edirlər. Misal üçün ümumi olan otel telefonları ssenarisinə baxa bilərik. Əgər biz otel-də olan ümumi telefon tələblərinə baxsaq, biz onları 3 ümumi kontekstin daxilində bölgə bilərik - *daxili işçilər, daxili qonaqlar və*

kənar zəng edənlər. Ümumi danışıqda istifadəmiz üçün biz onları qısa şəkildə "**staff**", "**guests**" və "**external**" adlandıracayıq.

"external" zəng edənlər üçün qrup yaradılmanın səbəbi kənardan gələn zənglərin bir iki əsas nömrələrə zəng etməklə front desk işçilərinə və restorana zəng etmələrinə şəraitin yaradılmasıdır. Eynilə də biz istəmirik ki, kənar zəng edən otel daxili imkanlara və otaqlara birbaşa zəng edə bilsinlər (Misal üçün zənglə oyadıcı servisinə). Ona görə də biz bütün mövcud olan rəqəmlərin siyahısını bir "**external**" adlı kontekstin içində birləşdirəcəyik və çöldən gələn bütün zəngləri bu kontekstə emal edilməsi üçün yönləndirəcəyik.

Qonaqlar isə digər otaqlara zəng edə bilməli, zənglə oyatmaq servisindən yararlanmalı və front desk işçilərinə zəng edə bilməlidirlər. Yenə də bu qrup nömrələri "**external**"-da olandan fərqlidirirlər və ona görə də onların özlərinə aid olan kontekstləri olacaq.

Sonda işçilərin spesifik funksiyaları mövcuddur, misal üçün telefona qeyd qoymaq "**checked out**" bacarığı, otaq boş olduğu müddətdən etibarən həmin telefona zəng edilməsi qadağasının qoyulması (təhlükəsizlik üçün dünya danışıqlarının qarşısını alırıq). Bu funksiyalar qonaqlar üçün və kənardan gələn zənglər üçün mövcud deyil. Beləliklə də işçilərin özlərinə aid olan kontekstləri mövcuddur.

Visual olaraq yaradılacaq məntiq aşağıdakı kimi olacaq:

	Zəng edən(context)	Daxili Qonaqlar	Daxili işçilər	Kənar Zəng edən
zəng edir (mənsebə)	Front Desk	✓	✓	✓
	Restoran	✓	✓	✓
	Otel otaqları	✓	✓	✗
	Oyandır zəngi	✓	✗	✗
	Checked-out flag	✗	✓	✗

Artıq biz kontekstlər üçün tələblərimizi görürük. Göstərilən cədvəldə hər bir zəng edənin individual tipə sahib olduğu göstərilir (kimə zəng etmək olar və kimə yox).

Diqqətli olun ki, kontekst anlayışı seçilmiş tərəfə zəng edə bilmək imkanı deyil, o qruplaşdırılmış nömrələr siyahısını təşkil edir hansı ki, tərəflərə çatmaq üçün yığılı bilər. Aktual istək kimə zəng etmək istəyimiz və o şəxsə hənsi nömrə təyin edilmişdir işi şərtlər tərəfindən emal edilir hansı ki, aşağıdakı seksiyada açıqlayırıq.

Şərtlər (Conditions)

Sistem zəng edilə biləcək şəxslər siyahısını əldə edən kimi, o dəqiq təyin etməlidir ki, zəng edən kimdir və kimə zəng edilir. Bu iş şərtlərin istifadə edilməsi ilə edilir.

Şərtləri bir və ya bir neçə məntiqi strukturdur hansı ki, zəngin hara getməsini qərarlaşdırır. Adətən zəng edənlə (məsələn hansı nömrəyə zəng edilmişdir və ya zəngin Caller ID-si nə idi) qaydalar toplusu müqayisə

edilir. Bu informasiya Dialplan dəyişənlərindən yığılır(bu başlıqda öncə danışıldı) və yenidən müntəzəm ifadələrlə, sətirlərlə və digər dəyişənlərlə tutuşdurulur.

Şərtlər adətən zəng edilən nömrə ilə spesifik ünvanın spesifik telefonuna xəritələnmə üçün tutuşdurulmada istifadə edilir. Misal üçün, biz sınaqdan keçirərək görə bilərik ki, əgər 415-886-7949 nömrəsinə zəng edilərsə, o 7949 genişlənməli telefona düşəcək.

Bəzi hallarda mənsəbin tutuşdurulması zəng edilən nömrədən kənar sütunlarla işləyirlər. Misal üçün siz zəng edənin Caller ID-sini yoxlaya bilərsiniz və əgər o telemarketerdirse məşğuldur signali təyin edə bilərik. Səliqə ilə zəngin yönləndirilməsi üçün həddən artıq sütun və mənalar mövcuddur ki, istifadə edəsiniz. Hətta texniki və ya verilənlər bazası tərəfindən idarə edilən quraşdırmałara belə baxmaq olar hansı ki, kiminse NAT arxasında olmasını təyin edir ya da onların call-forwarding məlumatı verilənlər bazasında saxlanılıbmı.

Həmçinin Dialplani elə təyin eləmək mümkündür ki, eyni zəngin içində çoxlu şərtlərə çoxlu işlər mənimsetmək olsun. Misal üçün zəng edən spesifik əraziyə zəng elədikdə orda dəstək asılonda zəng davam edir və onlara açıqlama oxumaq başlayır. Sizin Dialplan prosessorundan asılı olaraq fərqli tələblər üçün fərqli şərtlər mövcuddur.

Öncəki otel misalında gördüyüümüz məntiqi qursaq 3 kontekst istifadə ediləcək və onların hər biri fərqli mənsəbləri təyin edə bilmək üçün fərqli şərtlərə sahib olacaq(əşağıdakı kimi):

"Internal Guest" context	"Internal Staff" Context	"External" Context
0-a zəng etdilər?	0-a zəng etdilər?	
-Front deskə get	-Front deskə get	
2929-a zəng etdilər?	2929-a zəng etdilər?	646-222-2929-a zəng etdilər?
-Restorana get	-Restorana get	-Restorana get
3000-3999 aralıqda z.e?	3000-3999 aralıqda z.e?	
-Zəng otağı 3000-3999	-Zəng otağı 3000-3999	
*5-ə zəng etdilər?		
-Oyandır	*6-ya zəng etdilər?	
	-"Checked Out" flag qoy	

Susmaya görə və əksər hallarda Dialplan uyğunlaşma tapılanadək işləyir. Ətraflı məlumat üçün aşağıdakı genişlənmələrə baxın.

Işlər (Action)

Action şərt tutuşdurulması uyğun olduğundan sonra yerinə yetirilən addımdır. Bu freeswitch yerinə yetirməsi üçün Dialplan-ın generasiya elədiyi işlər siyahısıdır(Adətən zəng edəni mənsəbə çatdırır). İşlərə **"answering"**, **"bridging"**, **"routing to voicemail"** və.s daxildir.

Anlamanız vacibdir ki, Dialplan yalnız yerinə yetiriləcək olan siyahını yaradır - o həmin işləri real vaxtda yerinə yetirmir! Bu qayda üçün müəyyən

şərtlər var amma, Dialplan strukturunun ümumi baxışı modul ətrafında dövr edir və moduldan çox çətin bir iş tələb edilmir (faktiki zəng birləşdir və işi yerinə yetir).

Bunu aşağıdakı kimi açıqlamaq üçün, Dialplan haqqında zəng üzərində nə baş verəcək mövcud işlərin siyahısının generatoru kimi düşünün. Hal-hazırkı işlər siyahısı aşağıdakı kimi ola bilər:

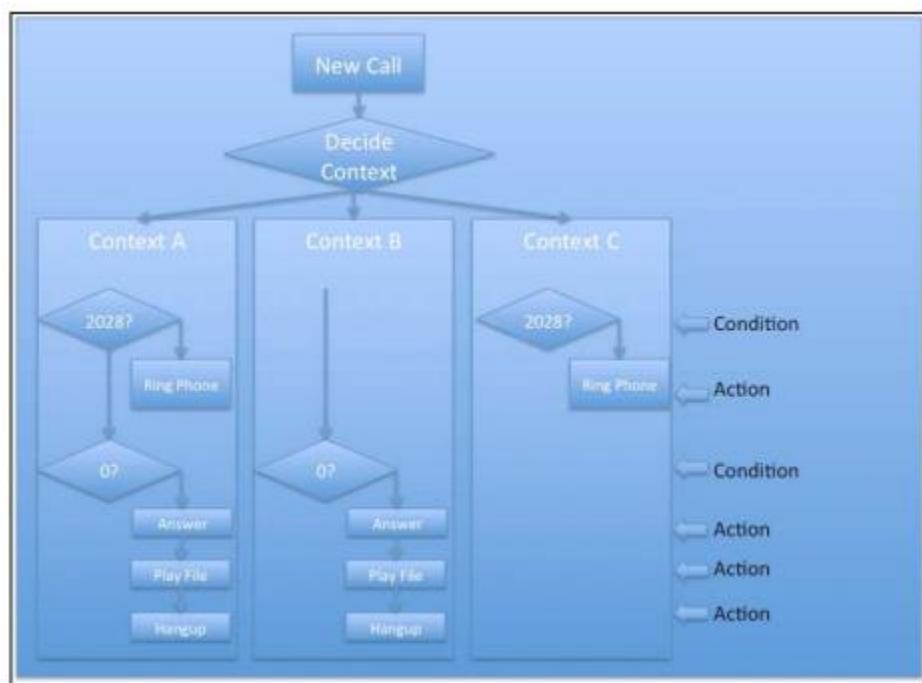
- Answer
- Play welcome file
- Transfer to user "John Doe"
- Hang up

Bu siyahı FreeSWITCH-ə qaytarıla bilər ki, yerinə yetirsən. Dialplan faktiki zəng axını üçün interaktiv olması nəzərdə tutulmur. Əgər siz zəng prosesində tamalı əlaqəyə girmək istəyirsinizsə siz FreeSWITCH üzərində olan script dilini öyrənməlisiniz. Bunun haqqında 7-ci başlıqdə *LUA vasitəsilə Dialplan scripting-də ətraflı oxuya bilərsiniz.*

Hamısının birləşdirilməsi

FreeSWITCH-in emal elədiyi bütün kontekstlər, genişlənmələr, şərtlər və işlər faktiki olaraq ROUTE phrase-da təyin edilmişdir. Bütün zənglər ROUTE statusu üzərindən keçir. Yönləndirmə statusu - FreeSWITCH zəng idarə etməsini Dialplan modul üçün istifadəyə ötürükdə təyin edilir və öncəki 4 anlayış istifadə edilir ki, işlərin siyahısı hazırlanın.

Proses ümumiyyətlə aşağıdakı kimi görünür:



Işlərin nəticəsi sonda FreeSWITCH-ə aşağıdakı şəkildə gələcək:

- Cavabı yerinə yetir

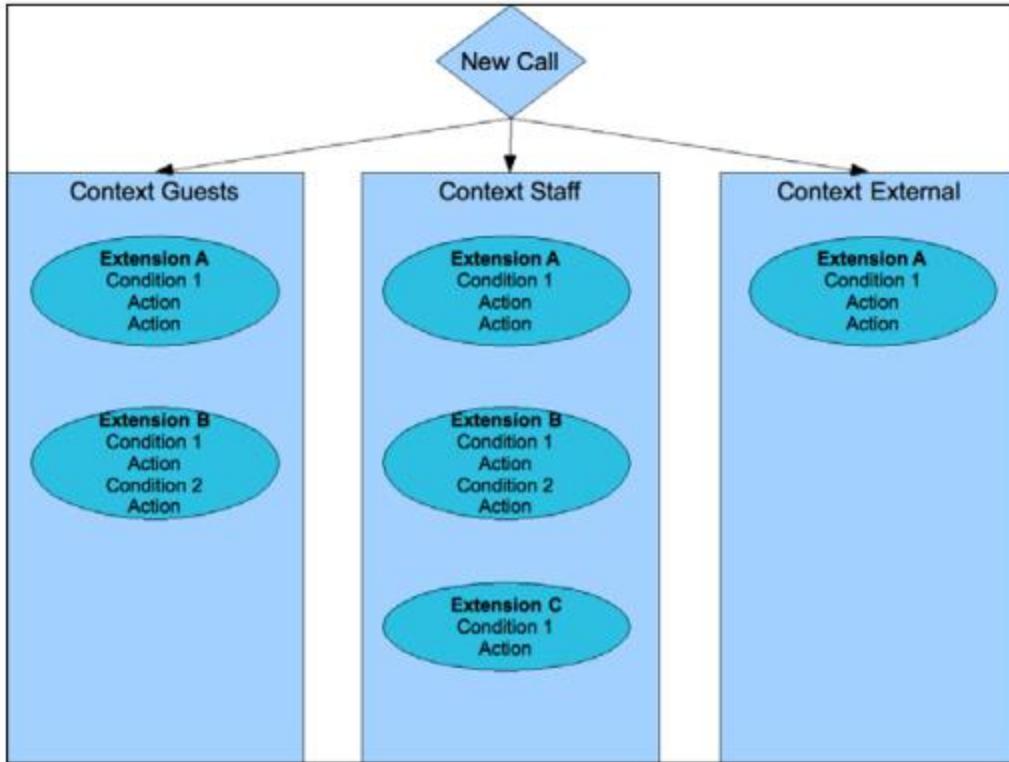
- Yerinə yetir və file.wav faylını oxut
- Dəstəyi üstünə qoymaşı yerinə yetir

Qeyd: FreeSWITCH cəmiyyətinin üzvləri tez-tez Dialplan "phases" haqqında danışırlar - onlar əksər hallarda bu iki **ROUTE** və **EXECUTE** proses fazasına istinad edirlər. Hərdən siz elə insanlar görə bilərsiniz ki, ROUTE frazاسını **"parsing"** frazası kimi adlandırırlar. "parsing" termini ROUTE fazası işləyən zaman nə baş verdiyini asanlıqla başa salır. Biz həmçinin **ROUTE** frazاسına sinonim olaraq **"hunting"** ifadəsindən istifadə edəcəyik. **ROUTE**, **parsing** və **hunting** eyni mənsəbə istinad edir.

Bu iki fazalı prosesin başa düşülməsi, XML dialplanın idarə edilməsi əməliyyatları üçün çox vacibdir.

XML Dialplan moduluna analiz

5-ci başlıqda XML Dialplan-ın başa düşülməsində danışdığımız kimi XML Dialplan modulu FreeSWITCH-də quraşdırılan əsas məhsur üsuldur. Kitabın yazılması müddətində də bu ən dayaniqli oları idi. Bu kontekstləri dəstəkləyir hansı ki, tərkibində genişlənmələr siyahısı olur, hər genişlənmədə bir ya da bir neçə şərt olur və hər bir şərtdə yerinə yetiriləcək olan işlərin siyahısı olur. Gəlin bir neçə konsepsiyaya baxaq ki, sizin onlarla işləmənin komfortlu olmasına əmin olaq. Dialplan verilənlərinin axtarışı və yazılarının emal edilməsi gözləmə mühitinə əsaslanır hansı ki, çox ölçülü ağaç kimi nəyəsə baxır.



Sizin Dialplana və onun necə istifadə edilməsinə qısa şəkildə baxdıqdan sonra, anlaşıla bilər ki, nəyə görə Dialplan üçün XML ən düzgün seçimdir. XML əlavəsi atributları göstərilən szenari üçün tamam yararlıdır. FreeSWITCH isə Dialplan quraşdırması parametrləri ağacına güvənir və XML öz növbəsinə təbii ki, məhdudiyyətsiz ağaç quruluşlu struktura sahibdir hansı ki, ağacda olan hər bir yarpaq üçün mənaların düzülməsini nəzərə alır. Bundan başqa siz istədiyiniz tag-lər və atributları istənilən zaman əlavə edə bilərsiniz, hətta FreeSWITCH tərəfindən məhəl qoyulmayanlar amma, sizin seçilmiş program təminatına XML-ə oxuma və yazma da yaralı olsa da belə. Bu böyük uyğunluqdur.

Bu nöqtədə artıq kontekstlər, genişlənmələr, şərtlər və işlər haqqında kifayət qədər məlumatlı olmalıyıq. Gəlin XML Dialplan-da daha da dərin işlər görməyə başlayaqla ki, Dialplan-da olan fərqli funksiyaların özlərini necə aparmalarını görə bilək.

Genişlənmələr

Genişlənmələr adı XML konteynerlərdir, tərkibinə şərtlər qrupu və işləri yığır. Diqqət yetirin ki, "**extensions**" adı sizi biraz qarşıqlığa gətirib çıxara bilər. Əksəriyyət insanlar genişlənmə anlayışını nəyəsə zəng etdikdə çata bilmək kimi anlayırlar. Məsələn 2900 yada 3000 nömrələri kimi amma, FreeSWITCH-də olan genişlənmələr anlayışı ümumiyyətlə sizi zəng etdinizlə heç əlaqələnmir. Onlar sadəcə Dialplan kontekstinin adıdır. Misal üçün **2900** nömrəsinə zəng etdikdə çatmaq üçün, siz ola bilər ki, faktiki **darren** adlı genişlənməyə düşürsünüz (Aşağıdakı kimi):

```

<extension name="darren">
  <condition field="destination_number" expression="^2900$">
  
```

```
<action application="bridge" data="user/darren"/>
</condition>
</extension>
```

Bu misalda genişlənmə darren adındadır amma **2900**-a zəng elədikdə Darrenə necə çatma işinə görə şərtə təşəkkür eləməlisiniz.

Genişlənmələrin çoxlu dəyişdirilə bilən hərəkətləri ola bilər. Susmaya görə FreeSWITCH genişlənməni tapan kimi ona şərtləri menimsədir və sonra genişlənmədə axtarışı dayandırır. Misal üçün sizdə genişlənmə var ki, sizin CALLER ID-ni dəyişir, genişlənmənin ardı ilə gedərək faktiki zəngi yönləndirir, FreeSWITCH heç bir zaman susmaya görə ikinci genişlənməyə çatmayacaq. Misal olaraq heç kəsə çatacaq statusda olmayıacaq:

```
<extension name="set_callerid">
  <condition field="destination_number" expression="^2900$">
    <action application="set" data="effective_caller_id_number=4158867900"/>
  </condition>
</extension>
<extension name="darren">
  <condition field="destination_number" expression="^2900$">
    <action application="bridge" data="user/darren"/>
  </condition>
</extension>
```

Öncəki misalda ikinci genişlənmə heç bir zaman işləməyəcək. Əgər siz istəsəniz ki, uğurlu tutuşdurmadan sonra əlavə genişlənmələr işə düşsün, siz **continue** flag-dan istifadə eləməlisiniz. Əgər öncəki misalda dəyişiklik eləsək Dialplan aşağıdakı kimi olacaq:

```
<extension name="set_callerid" continue="true">
  <condition field="destination_number" expression="^2900$">
    <action application="set" data="effective_caller_id_number=4158867900"/>
  </condition>
</extension>
<extension name="darren">
  <condition field="destination_number" expression="^2900$">
    <action application="bridge" data="user/darren"/>
  </condition>
</extension>
```

Nəzər yetirin ki, biz **continue="true"** -nu yalnız ilk genişlənmədə əlavə elədik ki, hətta bu genişlənməyə uyğun olan tutuşdurma olsa da belə emal davam eləsin. Biz bunu ikinci genişlənmə üçün əlavə eləmədik ona görə ki, biz emalın davamını ikinci tutuşdurmadan sonra istəmirik.

Şərtlər (Conditions)

Şərtlər müntəzəm ifadələrin dəyişənlərə qarşı sınaqdan keçirilməsini nəzərə alır. Şərtlər genişlənmə tag-ləri ilə birgə mövcud olurlar. Biz mövcud olan fərqli dəyişən tiplərini bu başlıqda birazdan danışacayıq.

Şərtlər bir və ya bir neçə genişlənmənin yoxlanılması üçün istifadə edilə bilər. Susmaya görə və onların əsas başlanğıc səviyyəsində bir və ya bir neçə

şərtlər nəticəsi **true** olduqda, yerinə yetirləcək işlər qrupu təyin edilə bilər.

Misal olaraq aşağıdakı kod hissəsinə baxa bilərik:

```
<extension name="test_two_things">
  <condition field="network_addr" expression="^192\.168\.1\.1$"/>
  <condition field="destination_number" expression="^2900$">
    <action application="playback" data="i-know-you.wav"/>
  </condition>
</extension>
```

Bu misaldaki şərt sadalanır. Qeyd, ilk **condition** tag-in sonunda **/>** simvolla ehtiyatlı olun. O vaxtadək ki, orda heç bir iş təyin edilməyib, o hələ də sınaqdan keçirilir və əgər o işini dayandırarsa bütün genişlənmələr buraxılır. Əgər hər iki şərtdən keçilərsə, onda o deməkdir ki, zəng edən tərəf **2900** nömrəsinə yiğir, onun IP ünvanı **192.168.1.1**-dir və onun işi yerinə yetirləcək. Bu effektiv olaraq iki şərt arasında **AND** məntiqini yaradır - hər ikisi **playback** işi yerinə yetirilməzdən önce tutuşdurulmalıdır.

Bu necə işləyir? Sırr **break** atributundadır(Həmçinin **break flag** adlanır). Hər bir **condition** operatorunun break **flag**-ı mövcuddur hansı ki, **expression** dəyərləndiriləndən sonra nə baş verməsini təyin edir. Susmaya görə **break** flagın mənası **on-false**-dir(Bu o deməkdir ki, dəyərləndirmə müddətində **false** ya da negativ nəticə alarıqsa şərtlər və genişlənmələr emalını tam dayandıracaq).

Siz **break** parametrinə aşağıdakı 4 mənadan birini mənimsədə bilərsiniz:

- **on-false**: İlk uğursuz tutuşdurmadan sonra axtarışı dayandır(susmaya görə aparış qaydası)
- **on-true**: İlk uğurlu tutuşdurmadan sonra şərtlərdə axtarışı dayandır
- **always**: Tutuşmanın olub olmamasından asılı olmayaraq, bu şərtdə dayandır
- **never**: Tutuşmanın olub olmamasından asılı olmayaraq, axtarışa davam elə

Siz həmçinin şərtlər istifadəsi üçün pseudo **if/then/else** emal seksiyası yarada bilərsiniz. Gələnin davamında bu atributlardan birini istifadə edək. Məsələn **never** flag.

Gəlin deyək ki, siz genişlənmə yaratmaq istəyirsiniz ki, o zəngi yalnız təyin edilmiş IP ünvanından edir və yoxlayır görək istifadəçi ***72** ya da ***73** nömrəsinə zəng edirmi. Siz bu işi aşağıdakı qaydada iki ayrıca genişlənmədə edə bilərsiniz:

```
<extension name="extension_72">
  <condition field="network_addr" expression="^192\.168\.1\.1$"/>
  <condition field="destination_number" expression="^*\*72$">
    <action application="playback" data="forward-on.wav"/>
  </condition>
</extension>

<extension name="extension_73">
  <condition field="network_addr" expression="^192\.168\.1\.1$"/>
  <condition field="destination_number" expression="^*\*73$">
    <action application="playback" data="forward-off.wav"/>
  </condition>
```

```
</extension>
```

Bu işi eynilə güclü **break** flagın köməkliyi ilə bir genişlənmənin daxilində də edə bilərsiniz:

```
<extension name="extension_72_or_73">
  <condition field="network_addr" expression="^192\.168\.1\.1$"/>
  <condition field="destination_number" expression="^\*72$" break="on-true">
    <action application="playback" data="forward-on.wav"/>
  </condition>
  <condition field="destination_number" expression="^\*73$">
    <action application="playback" data="forward-off.wav"/>
  </condition>
</extension>
```

Gördüyünüz kimi, ilk şərtdə **break="on-true"** flagın əlavə edilməsi ilə biz emalı şərtin **true** olduğu halında dayandırırıq. Əks halda emal davam edəcək. Buna **if/then/else-if** operatoru kimi baxın; əgər ilk şərt uyğundursa işə sal və emalı dayandır eks halda, tutuşdurma olarsa ikinci şərti işə sal.

Digər misalda olduğu kimi, həmçinin **if/then** konsepsiyasına da baxın. Siz bu məntiqi **break="never"** flagı ilə simulyasiya edə bilərsiniz. O vaxtadək ki, nasaz şərtlərin daxilində olan işlər yerinə yetiriləcək addımlara əlavə edilməyəcək, ardıcıl şərtlər hələ də emal ediləcək. Misala baxın, harda ki biz iki fərqli şəbəkə parametrini eyni blok genişlənmədə yoxlamaq istəyirik:

```
<extension name="decide_routing">
  <condition field="network_addr" expression="^192\.168\.1\.1$" break="never">
    <action application="set" data="inhouse=true"/>
  </condition>
  <condition field="source" expression="PortAudio"/>
    <action application="set" data="portaudio=true"/>
  </condition>
</extension>
```

Öncəki XML yoxlanış edəcək əgər, istifadəçi **192.168.1.1** IP ünvanından zəng edirse və **set** dəyişəni **true**-dursa. O hər bir halda ikinci şərtə davam edəcəkdir, istifadəçinin **PortAudio** istifadə etməsini yoxlayacaqdı və əgər **true**-dursa **set** dəyişənini təyin edəcək. Bu effektiv **if/then** şərtini digər **if/then** ardından verir.

Şərt hər bir extension tag-in içində mövcud olmalıdır hətta, siz şərt üçün həmişə **true** təyin etsəniz belə. Aşağıdakı kod hissəsi düzgün deyil:

```
<extension name="set_callerid">
  <action application="set" data="effective_callerid_number=4158867900"/>
</extension>
```

condition tag-ı mövcud olmadığına görə bu genişlənməyə məhəl qoyulmayacaq və iş heç bir zaman yerinə yetirilməyəcək.

Spesifik şərt dəyişənləri

Gördüyünüz əksər şərtlərdə siz dəyişənlər və ifadələr istifadə edəcəksiniz amma, şərtlər elə yazılı bilər ki, spesifik şərt dəyişənləri istifadə edərək səliqə ilə emal işini sadə və dinamik eləsin.

Aşağıdakı dəyişənlər sizin XML **condition** tag-larında istifadə eləmək üçün **field** atributlarıdır:

- **context**: Mövcud Dialplan konteksti
- **rdnis**: Yönündirilən rəqəm; qovluq rəqəmi hansı ki, zəng elan edilmişdir
- **destination_number**: Zəng edilən rəqəm; bu, çatmaq istədiyimiz zəngin rəqəmidir
- **dialplan**: İstifadə olunan Dialplan modulunun adı
- **caller_id_name**: Əgər mövcuddursa, zəng edənin adı
- **caller_id_number**: Əgər mövcuddursa, zəng edən tərəfin rəqəmi
- **ani**: Zəng edən tərəfin Avtomatik rəqəm indentifikasiyası(The Automatic Number Identification)
- **aniii**: Əgər mövcuddursa, zəngdə yerləşdirilən avadanlıq tipi(misal üçün, *payphone*)
- **uuid**: Mövcud zəng üçün unikal identifikasiator
- **source**: Zəngi qəbul edən FreeSWITCH modulunun adı(misal üçün, PortAudio)
- **chan_name**: Mövcud kanalın adı(misal üçün, *PortAudio/1234*)
- **network_addr**: Zəng üçün siqnal mənbəsinin IP ünvanı

Müəyyən kiçik nüsxələr aşağıdadır:

```
<condition field="network_addr" expression="^1\.2\.3\.4$">
<condition field="caller_id_number" expression="^1[01]\d\d"/>
```

Aşağıdakılar birbaşa olaraq **condition** tag-in daxilində atribut kimi, istifadə edilə bilər:

- **year**: Təqvim ilini ifadə edir, 0-9999
- **yday**: Ilin gününü ifadə edir, 1-366
- **mon**: Ayı ifadə edir, 1-12(Misal üçün, Yanvar **1** deməkdir)
- **mday**: Ayın gününü ifadə edir, 1-31
- **week**: Həftənin gününü ifadə edir, 1-53
- **mweek**: Ayın həftəsini ifadə edir, 1-6
- **wday**: Həftənin gününü ifadə edir, 1-7(Misal üçün, **Sunday** bərabərdir **1** və **Monday** bərabərdir **2**)
- **hour**: Saati ifadə edir, 0-23
- **minute**: Dəqiqəni ifadə edir (saatdan dəqiqə), 0-59
- **minute-of-day**: Günün dəqiqəsini ifadə edir, (1-1440) (misal üçün, gecə saat 1, 1 A.M vaxtı 60-a bərabərdir və günorta 720-ə bərabərdir)

Bu şərt atributları aşağıdakı kimi istifadə edilə bilər:

```
<extension name="holiday_example" continue="true">
<condition mday="1" mon="1">
<action application="set" data="open=false" inline="true"/>
</condition>
```

```
</extension>
```

Misal yeni il gunu üçün **open** dəyişənini **false** təyin edir. Qeyd edin ki, condition sətiri **mon** və **mday** şərt dəyişənlərini istifadə edir. Adı day/time yoxlanışı aşağıdakı kimi olardı:

```
<extension name="holiday_example" continue="true">
  <condition wday="2-6" hour="8-16">
    <action application="set" data="open=true" inline="true"/>
  </condition>
</extension>
```

Əgər zəng 8 A.M. və 5 P.M. aralığında gələrsə, bu kanal dəyişəni **open-i "true"** təyin edir.

Daxili yerinə yetirilmə

FreeSWITCH original dizaynda routing/Dialplan fazası müddətində baş verən istənilən işləri bağlayır. Burda daxili problem var idi hansı ki, insanları fikir dəyişikliyinə gətirirdi və Dialplan daxilindən zəng axını ilə dəyişənləri idarə edirdi. Bunun əvəzinə bu işi fərqli programlaşdırma dillərində saxladılar. Məntiq sadədir - Dialplan məhdudiyyətində istifadə edilə biləcək dəyişənləri qiymətləndirmək, məntiq yaratmaq üçün aydın olmayan və məhdudiyyətli əmrlər toplusu əvəzinə FreeSWITCH script dillərinə müraciət eləməyi daha üstün tutur. Misal üçün Perl ya da Lua(Əla sənədləşmə) dillərində FreeSWITCH-də olan emaledici məntiqdən daha geniş imkan var. Bu nəinki geniş imkana sahibdir həmçinin, istifadəçilərin giriş və çıkışda XML emaledici sistemin analiz edilməsinin qarşısını alır.

Ancaq FreeSWITCH genişləndikcə, zəng axının emali işini elə XML Dialplan dilinin özündə eləmək eksər tələbi insanlardan gəlməyə başladı. Dialplan yönləndirilməsində ən böyük çatışmamazlıqlardan biri də o idi ki, dəyişəni təyin eləmək(həmçinin üstünə başqasını yazmaq) və sonra sınaqdan keçirmək mümkün deyildi. Core programçılara saysız müraciətlərdən sonra, **inline** flag-ı XML Dialplan emalına əlavə edildi.

inline flag-ı bəzi əmrlərə(hamısına deyil) izin verir ki, öncədən təyin edilən qaydaları zədələyərək Dialplan fazasında yerinə yetirilsin. Bu bizi sarsıtsa da, bəzi hallar olur ki, əlavə funksionallıq mütləq tələbdir və kodu daha aydın şəkildə oxumaq lazım olur. Buna görə də daxili yerinə yetirilməyə ehtiyac olur. Aşağıda kod hissəsi göstərilir:

```
<extension name="check_for_user" continue="true">
  <condition field="${callerid}" expression="2035551212">
    <action application="set" data="user=yes"/>
  </condition>
</extension>

<extension name="route_users_only">
  <condition field="${user}" expression="yes">
    <action application="answer"/>
    <action application="playback" data="tada.wav" />
    <action application="hangup"/>
  </condition>
</extension>
```

Öncəki kod hissəsi iki genişlənmə təyin edir hansı ki, biri digərinə güvənir - ilk genişlənmə gördüyüümüz kimi əgər kimsə 203-555-1212 nömrəsindən zəng edərsə, **user** dəyişəninə **yes** təyin edilir və ikinci dəyişəndə deyilir ki, user dəyişəni təyin edilən bütün zəngləri yönləndir ki, **tada.wav** faylini işə salsın. Bu üsulda kod işləməyəcək. Gəlin nə üçün işləmədiyinə baxaq:

Yada salın ki, şərtlər istənilən yerinə yetirilmiş işlərdən önce dəyərləndirilir. Öncəki misalımızda **set** iş programıdır. Bu səbəbdən də, bütün şərtlər dəyərləndirilməyənədək o yerinə yetirilməyəcək. **condition** baxdığı **\${user}** dəyişəni olan istənilən yerdə işləməyəcək ona görə ki, işləmək üçün dəyişəni təyin edən kod daxili emalda olmasına **inline** emal tələbi edir.

Əgər XML işi oxumağa dəyişirse:

```
<action application="set" data="user=yes" inline="true"/>
```

XML Dialplan prosessoru faktiki olaraq **set** programını gördüyü istənilən yerdə yerinə yetirəcək. Artıq XML kodu istədiyimiz kimi işləyəcək.

Texniki səbəblərdən bu imkandan həddən artıq istifadə edilmənin qarşısını almaq üçün, **inline** flag-ı yalnız çox sürətli işləyən proqramlar və bəzi dəyişənlərin götürüle bilməsi üçün məhdudlaşdırılmışdır. Həmçinin mövcud sessiya üçün **inline** funksiyalarının statuslarının dəyişdirilməsinə yetkini məhdudlaşdırmaq lazımdır.

Inline emal üçün mövcud işlərin siyahısı aşağıda sadalanır:

- **check_acl**: Giriş idarə etməsi siyahısını yoxla
- **eval**: Daxili API yerinə yetir ya da konsola müəyyən mətni göndər
- **event**: Təsadüfi eventləri işə sal
- **export**: Dəyişəni kanalda təyin elə hansı ki B ayağı ötürülməsinə yarayacaq
- **log**: Jurnal verilənini əllə yaradırıq
- **presence**: **PRESENCE_IN** və **PRESENCE_OUT** event-lərini yolla
- **set**: Kanala dəyişəni təyin elə
- **set_global**: Qlobal dəyişəni təyin elə
- **set_profile_var**: İstifadə edəcəyimiz istifadəçi profile-nı təyin elə
- **set_user**: Hal-hazırkı istifadəçini təyin elə və onun bütün kanal dəyişənlərini aktiv jurnalaya yaz
- **sleep**: Emala ara ver
- **unset**: Dəyişəni təmizlə
- **verbose_events**: Hadisənin çıxışını daha detallı çap elə
- **cidlookup**: **caller_id_name** sütununun **CNAM** axtarışı ilə təyin elə
- **curl**: HTTP müraciət elə və cavab al
- **easyroute**: Verilənlər bazası DID yönləndirmə motoruna birbaşa qoşulma
- **enum**: Sadalanan axtarışlar ya da uyğun servislər üçün
- **lcr**: Yönləndirmə həllərinin daha az dəyərinin götürülməsi
- **nibblebill**: Hesab dəqiqliyə əsaslanaraq sayıır
- **odbc_query**: Əlimizlə ODBC müraciətlər edirik

Bu əmrlərin hər birinin detali FreeSWITCH wikipediada açıqlanır.

Actions və anti-actions

Digər mövcud olan zəng idarə edilməsi mexanizmi **action** və **anti-action** kombinasiya tag-larıdır. Şərtlərdən fərqli olaraq əgər **condition** yerinə yetirilməzsə, bu tag-lar size səhv baş verən halda alternativ işlər yiğimina keçidə izin verirlər. Bu if/then/else şərtinin digər bir versiyasıdır ancaq, rahat oxunula və idarə edilə bilər. Əsasən də eyni şərtin daxilində bir neçə işlərə cəhd edilərsə.

Gəlin işləməsi üçün bir neçə misala baxaq:

```
<condition field="${inhouse}" expression="true" break="never">
  <action application="log" data="This is an in-house call"/>
  <anti-action application="log" data="Not in-house call"/>
</condition>
```

Öncəki misalda əgər **\${inhouse}** dəyişəni **true** olarsa, jurnalda "This is an in-house call" yazılıcaq əks halda, jurnalda "Not in-house call" yazılıcaq. İki ədəd **if/then/else** şərt operatorlarında ayırmak əvəzinə bu daha da oxunaqlıdır.

RegEX operatoru

Bəzi hallarda daha da çətin məntiqi strukturların yaradılması tələbi olur.

Məsələn **destination_number** ifadəsi **1000** ya da **1001**-dirmi:

```
<condition field="destination_number" expression="^100[01]$">
```

Həqiqətdə, müntəzəm ifadə belə deyir "Match if the destination number is 1000 or is 1001". Əgər sizdə məntiqi **OR**-un iki fərqli sütundada istifadə etmək sizin tələbi varsa? Bu halda **regex** operatoru kömək edəcək.

regex üçün başlanğıc sintaksis:

```
<condition regex="all|any|xor">
  <regex field="some_field" expression="Some Value"/>
  <regex field="another_field" expression="^Another\s*Value$"/>
  <action(s) ...>
  <anti-action(s) ...>
</condition>
```

Sizin şərt daxilində çoxlu **regex** tag-ləri yazma tələbi ola bilər. RegEX operatorunun 3 fərqli mənası ola bilər:

- **all**: Bu məntiqi **AND** operatorunun ekvivalentidir. İşin yerinə yetirilməsi üçün şərtin daxilində olan bütün ifadələr **true** olmalıdır.
- **any**: Bu məntiqi **OR** operatorunun ekvivalentidir. İşin yerinə yetirilməsi üçün şərtin daxilində olan istənilən ifadələrdən biri **true** olmalıdır.
- **xor**: Bu **XOR** əməliyyatının ekvivalentidir. İşin yerinə yetirilməsi üçün ifadələrdən yalnız biri **true** ola bilər.

Iki fərqli sütundada olan **OR** kimi daha çətin məntiqi operatorun yerinə yetirilməsi üçün aşağıdakını yoxlayın:

```
<extension name="Regex OR example" continue="true">
  <condition regex="any">
```

```
<!-- Əgər hansısa true-dursa işi yerinə yetir -->
<regex field="caller_id_name" expression="Some User"/>
<regex field="caller_id_number" expression="^1001$"/>
<action application="log" data="INFO At least one matched!"/>
<!-- Əgər *none* true-dursa onda, əks işlər gör -->
<anti-action application="log" data="WARNING None of the conditions
matched!"/>
</condition>
</extension>
```

Öncəki misalda action tag-ı yalnız **caller_id_name** ifadəsi "Some User" ya da **caller_id_number** ifadəsi "1001" olarsa yerinə yetiriləcək. Əgər heç bir **regex** true olmazsa, onda **anti-action** tag yerinə yetiriləcək.

Sizin daha da çətin şərtiniz ola bilər. Misal üçün deyək ki, "Some User" üçün birdən çox genişlənmə mövcuddur. Onda siz aşağıdakı regex-i istifadə edə bilərsiniz:

```
<regex field="caller_id_number" expression="^1001|1005$"/>
```

Geniş **AND** operatoru istifadə eləmək üçün siz aşağıdakı kimi genişlənmə istifadə edə bilərsiniz:

```
<extension name="Regex AND example" continue="true">
<condition regex="all">
<!-- Əgər hər ikisi true-dursa onda işlər görüləcək -->
<regex field="caller_id_name" expression="Some User"/>
<regex field="caller_id_number" expression="^1001$"/>
<action application="log" data="INFO Both matched!"/>
<!-- Əgər istənilən biri *any* false-dırsa onda anti-actions işləyir -->
<anti-action application="log" data="WARNING At least one failed!"/>
</condition>
</extension>
```

Yenə də deyirik, siz çoxlu **<regex>** tag-ləri şərtin daxilində istədiyiniz kimi istifadə edə bilərsiniz. Onların hamısı programları action tab-ların daxilində səliqə ilə yerinə yetirmək üçün **true** olmalıdır. Əgər regex-lərin sınaqlarından hansısa biri uğursuz olarsa onda, anti-action tag-ın daxilində olan programlar işə düşəcək.

Bəzi nadir hallar ola bilər ki, siz **XOR** ya da exclusive **OR** əməliyyatını yerinə yetirmək istəyirsiniz. Bu həmən yerdir hansı ki, bir və yalnız bir şərt **true** ola bilər. Əgər şərtlərdən heç biri **true** deyilsə ya da şərtlərdə 1-dən çox **true** varsa onda, bütöv məntiqi əməliyyat **false** olacaq. Aşağıdakı misal kimi:

```
<extension name="Regex XOR example" continue="true">
<condition regex="xor">
<!-- Əgər bircə dənə true varsa aşağıdakı işlər görüləcək -->
<regex field="caller_id_name" expression="Some User"/>
<regex field="caller_id_number" expression="^1001$"/>
<action application="log" data="INFO exactly one matched!"/>
<!-- Əgər hər ikisi true ya da hər ikisi false olarsa XOR işləməyəcək -->
<anti-action application="log" data="WARNING XOR evaluated false!"/>
</condition>
</extension>
```

Əlavə edilmiş şərtlər

FreeSWITCH 1.2.6 versiyasından etibarən siz <**condition**> tag-ləri fərqli situasiyalarda yerləşdirə bilərsiniz. Belə bir şərt məntiqi strukturda göstərilir:

```

IF condition1 TRUE THEN
    IF condition2 TRUE THEN
        DO actions
    ELSE
        DO other actions
    ENDIF
    IF condition3 TRUE THEN
        DO actions
    ELSE
        DO other actions
    ENDIF
    IF condition4 TRUE THEN
        DO actions
    ELSE
        DO other actions
    ENDIF
    ELSE
        DO other actions
    ENDIF

```

Bu işin görünməsi üçün şərtlərin əlavə ediləsi hər şeyi asanlaşdırır. Aşağıdakı quruluş bizim məntiqi strukturumuzda **condition1 destination_number**-e ekvivalenti göstərilir. Əgər digər sutunlarda varsa, onlar **condition2**, **condition3** və **condition4** üçün ekvivalentdirlər. Qısa olaraq biz **action** və **anti-action** tag-ləri təyin eləmədik:

```

<condition field="destination_number" expression="^1000$"
require_nested="false">
<action...>
<anti-action...>
<condition field="caller_id_number" expression="^1001$">
<action...>
<anti-action...>
</condition>
<condition field="caller_id_number" expression="1?408\d+">
<action...>
<anti-action...>
</condition>
<condition field="caller_id_number" expression="^1001$">
<action...>
<anti-action...>
</condition>
<action...>
<anti-action...>
</condition>

```

Vacib qeydi yadınızda saxlayın, bütün əlavə edilmiş şərtlər əsas şərtdən öncə yerinə yetirilir. Bu o deməkdir ki, valideyn şərtdə olan bütün action və anti-action ifadələri həmişə **actions/anti-actions** əlavə edilmiş şərtlərdən sonra yerinə yetiriləcək. Digər sözlə desək Dialplan parçalayıcısı işlər üçün axtarış fazasında baxdıqda(ya da **anti-actions**) o, həmişə actions/anti-actions-ları valideyn şərtdən öne əlavə edilmiş şərtlərin içine atacaq.

Əvvəlki nümunədə hər şey üç əlavə edilmiş şərtlərdə qiymətləndirilmiş olacaqlar (hansılar ki, **caller_id_number**-i yoxlayırlar), və onların **action/anti-action**-ları valideyn şərtləri **action/anti-action**-dan əvvəl işə düşəcək(hansı ki, **destination_number**-i yoxlayır). Adı sözlə desək bala şərtlər valideyn şərtlərdən tez gəlir.

Aşağıdakı seksiya Dialplan işləməsinin iki dənən çox fazasını və yeni istifadəçilərin adətən elədikləri səhvləri açıqlayır.

Tələlərdən qacış

FreeSWITCH-də iki əsas yer vardır ki, Dialplan dizaynında istifadəçiləri qarışığığa gətirib çıxara bilər - xüsusən də sizin Asterisk-lə təcrübəniz varsa belə olur. İlk olaraq dəyişənlərin işə düşməsi zamanı necə emal edilmələrini və sonra jurnalların interpretasiyasını anlamalısınız.

Yadda saxlayın ki, Dialplan iki fazlı prosesdir, ilk olaraq nəyi yönləndirməli(həmçinin axtarış və parçalama kimi də başa düşülür) fazasıdır. Bu faza yerinə yetirilən digər əmrlərin içində tamamilə üstün olanıdır. Gəlin yenə düzgün işləməyən bir misala baxaq. Bu dəfə əksər insanların öz XML-lərini **debug** elədiyi qaydada edəcəyik - biz XML-imizə info programını əlavə edəcəyik ki, kanalda təyin edilən dəyişənlərin siyahısını konsolumuzda görə bilək.

```
<extension name="check_for_user" continue="true">
  <condition field="${callerid}" expression="2035551212">
    <action application="set" data="user=yes"/>
    <b><b><action application="info"/>
  </condition>
</extension>
<extension name="route_users_only">
  <condition field="${user}" expression="yes">
    <action application="answer"/>
    <action application="playback" data="tada.wav"/>
    <action application="hangup"/>
  </condition>
```

```
</extension>
```

Bu kod hissəsi işləyən kimi, **info** programı kanalda təyin edilən bütün dəyişənləri ekrana çap edəcək və istifadəçi görəcək ki, "**user**" dəyişəninə "**yes**" təyin edilib. Ancaq ardınca gələn **condition** hansı ki, **user** dəyişənini sınaqdan keçirir işə düşməyəcək. Əksər istifadəçilər düşünəcək ki, FreeSWITCH-də problem var ancaq, əslində hər kəs gördükələri çıxışı düzgün oxumurlar. Diqqətli analizdən sonra görəcəksiniz ki, sizin jurnallarda **set** programı sınaqdan keçirilən bütün şərtlərdən sonra yerinə yetirilmişdir və **info** programı ardıcıl olaraq dəyişənin təyin edilməsini göstərərək işə düşmüştür. Amma şərtin yoxlanılması bu baş verməsindən xeyli uzun müddət öncə edilmişdir. Əgər siz jurnalları diqqətli oxumasanız bu bir neçə saatın boşuna itkisinə gətirib çıxara bilər.

Jurnal göstərəcək ki, planlaşdırılmış Dialplan verilənləri həqiqətən də uyğundurmu və bu da insanları düşüncəyə gətirə bilər ki, o Dialplan seksiyaları doğrudan da işləmişdir. Əslində siz bunu görmək üçün jurnalların sonunadək analiz etməlisiniz. **EXECUTE** jurnal operatorunu analiz etməyə öyrənin ki, nə baş verdiyini daha aydın başa düşəsiniz. Əgər elementlər gözlədiyiniz kimi yerinə yetirilməzsə, onda sizin şərtlər düşündüyüünüz kimi işləməmişdir.

XML Dialplan programları

Biz Dialplan-ın bütün əmrlərinin siyahısının hamısını öyrənmək istəsək də, siyahı çox böyükdür və bu başlıqə yerləşməyəcək. Ona görə də biz Dialplan əmrlərini 3 aralıqda danışacayıq - Dialplan alətləri, Sofia əlaqələnməsi və ümumi API əmrləri. Bu dialplan əmrləri **mod_dptools**, **mod_sofia** və **mod_commands** tərəfindən təmin edilir. Həmçinin orda gündəlik istifadə edilən əmrlər toplusu da mövcuddur.

mod_dptools

mod_dptools - Dialplan idarə edilməsi üçün əmrlər toplusudur. Bu Dialplan çərçivəsində çoxlu programlar mövcuddur. Siz həmçinin **answer**, **hangup**, **bridge** və **set** kimi bəzi başlangıç əmrlər sayesində də öyrəndiniz. Gəlin daha da qabaqcıl əmrləri açıqlayaq:

- **bing_meta_app**: Bu əmr programı spesifik zəng ayağı(ları)na birləşdirir. Bridge edilmiş zəng müddətində, birləşdirilmiş zəng ayağında DTMF ardıcılılığı programın yerinə yetirilməsini insializasiya edəcək.

Birləşdirilməyən zəng ayağı yığım müddətində DTMF ardıcılılığını eşitməyəcək. Yalnız tək rəqəmi birləşdirə bilərsiniz və əlaqə adətən * açarının sıxılması ilə baş verir. Misal olaraq, deyə bilərik ki, siz *2 sıxımı ilə zəngin yazılımasını istəyirsiniz.

Zəng edən tərəf *2 sıxarsa, yazılıma başlayacaq. Bu halda siz aşağıdakı Dialplan kod hissəsini istifadə edə bilərsiniz:

```
<action application="bind_meta_app" data="2 a s
record_session::recording.wav"/>
<action application="bridge" data="sofia/sipprovider/+14158867900">
```

Bu iş kanalda olan A-ayağına imkan verir ki, *2 sıxmaqla eyni zəng ayağında olan zəngi yaza bilsin (üçüncü parametr **same leg** - eyni ayaq deməkdir).

Qeyd edin ki, əgər başqa cür təyin edilməyibse, **bind_meta_app** programı * simvolunu elə "**meta key**" kimi istifadə edəcək. Bunu dəyişmək üçün **bind_meta_key** kanal dəyişəninin mənasını fərqli edin. Misal üçün * evəzine # istifadə eləməklə bunu edə bilərsiniz:

```
<action application="set" data="bind_meta_key=#"/>
```

bind_meta_app parametrlərinin formatına diqqət yetirin:

```
<action application="bind_meta_app" data="KEY LISTEN_TO RESPOND_ON
APPLICATION[::PARAMETERS]"/>
```

Aşağıdakı siyahı öncəki parametrləri açıqlayır:

- **KEY:** Qulaq asılacaq açarı təyin edir.
- **LISTEN_TO:** Bu hansı zəng ayağının açar olaraq qulaq asmasını təyin edir. Mümkün ola biləcək parametrlər **a,b** ya da **ab**-dir.
- **RESPOND_ON:** Təyin edir ki, açar zəng edildikdə hansı zəng ayağı cavab verməlidir. Misal üçün cavab əmri olaraq fayl oxudulduğda hansı ayaq **playback** eşidəcək. Eyni zəng ayağı üçün qəbul edilən parametr **s** -dir. Bu həmin ayaqdır hansında ki, açar sıxılmışdır ya da ○ əks ayaq üçündür.
- **APPLICATION:** Bu hansı programın işə düşməsini təyin edir.
- **PARAMETERS:** Bu programaya keçmək üçün parametri təyin edir. Qeyd edin iki ardıcıl iki nöqtə ilə siz (**APPLICATION::PARAMS**) programları ayırirsiniz.

Qeyd: Zəng ayağına birləşən kimi, programın birləşməsi zəng yaşadığı müddətədək işlək vəziyyətdə olacaq.

Irəliləmiş DTMF açar birləşdirilməsi üçün **bind_digit_action** mexanizminə baxın.

- **bind_digit_action:** Bu əmr **bind_meta_app**-a baxdıqda daha da irəliləmiş və elegant açar birləşməsi mexanizmini təqdim edir. O sizə istənilən rəqəm kombinasiyasını yığmağa şərait yaradır və onun * simvolu ilə başlamasını tələb eləmir.

bind_digit_action parametrinin formatı:

```
<action application="bind_digit_action"
data="REALM,KEY|REGEX,COMMAND,COMMAND_ARGUMENTS,LISTEN_TO,RESPOND_ON"/>
```

Aşağıdakı siyahı öncəki parametrləri açıqlayır:

- **REALM:** İstifadəçini təyin edir.
- **KEY|REGEX:** Qulaq asacağı açarı təyin edir ya da qulaq asmaq üçün tərkibində **key** parametrini tutan **regex**.
- **COMMAND:** İşə salmaq üçün Dialplan programını təyin edir. Bu istənilən dialplan əmri(**exec**: prefiks ilə) ya da API əmri(**api**: prefiks ilə) ola bilər.
- **COMMAND_ARGUMENTS:** Yerinə yetiriləcək əmrin argumentlərini təyin edir.
- **LISTEN_TO:** Açıq üçün hansı zəng ayağı olacağını təyin edir. Qəbul ediləcək parametrlər **a,b** ya da **ab**-dir.
- **RESPOND_ON:** Cavab verəcək zəng ayağını təyin edir. Misal üçün fayl cavab əmri olaraq işə düşərsə, hansı zəng ayağı bunu eşitməlidir. Eyni ayaq üçün qəbul ediləcək parametrlər **s**-dir. Bu həmin ayaqdır hansı ki, açar sıxılmışdır ya da **o** açarı qarşidakı ayaq üçündür.

Burda misal göstərilir. Aşağıdakı əmr sistemdə olan bütün zəngləri gizli kod **9348234** yığdıqda kəsəcək. O **hupall** API əmrini işə salır.

```
<action application="bind_digit_action" data="my_digits,9348234,api:hupall"/>
```

Bilin ki, **bind_digit_action** yığılmış əmrləri **bind_digit_action** tərəfindən elə istifadə ediləcək ki, digər programlara ötürülməyəcək.

- **eavesdrop:** Bu əmr sizə digər kanallara qulaq asmağa imkan yaradır. Misal olaraq, aşağıdakı Dialplan əmri size **\$1**-də olan **UUID**-yə qulaq asmağa kömək edəcək.

```
<action application="eavesdrop" data="$1"/>
```

Siz **\$1**-i istədiyiniz digər UUID ilə dəyişə bilərsiniz ya da verilənlər bazasında olan UUID-ni aşağıdakı qayda da ala bilərsiniz:

```
<action application="eavesdrop" data="${db(select/spymap/${extension})}"/>
```

Bu misalda **extension** dəyişəni istifadə edilir hansı ki, Dialplan-a prioritət təyin edə bilir və axtarış parametri kimi istifadə edilir ki, verilənlər bazasında təyin edilən **UUID** üçün axtarış eləssin. Bu sənari-də əger siz genişlənmə rəqəmini yazmışınızsa və bütün aktiv zənglərin UUID-si verilənlər bazasının **spymap** cədvəlinde yerləşirse siz qulaq asmaq üçün həmin məlumatı sonra burdan ala bilərsiniz.

execute_extension: Hər hansıa genişlənməni başqa bir genişlənmənin içindən bu Dialplan programı ilə yerinə yetirə bilərsiniz. Məqsəd o ola bilər ki, zəngi müvəqqəti olaraq bir yere yönləndirəsiniz və sonra geriyə əvvələ qaytarasınız. Bu digər keçiricilərdə olan **loopback** funksiyası ilə eynidir. **execute_extension** genişlənməni macro kimi yerinə yetirir və sonra qaytarır. Bu transferdən fərqlidir hansı ki, birbaşa digər genişlənməyə gedir və geri qayıtmır. **execute_extension** əmri hal-hazırkı konteksti saxlayacaq, müvəqqəti genişlənmə yaradıb onu yerinə yetirəcək və geriyə zəng edilən yerə qayıdacaq.

```
<action application="execute_extension" data="destination_number [Dialplan] [context]"/>
```

Əgər siz Dialplan və kontekst təyin etməmisinizsə, o susmaya görə olan mənani alacaq.

execute_extension o halda istifadə edin ki, sizin əmri yerinə yetirib və sonra Dialplan emalına(çıxdığınız yerə) geriyə qayıtmağa eyhtiyacınız var. Əgər başqa heç bir şeyə ehtiyac yoxdursa, **transfer** programını istifadə edin. Əgər siz programçısınızsa, onda bu analogiyada: **execute_extension** oxşayır **gosub**-a harda ki, **transfer goto** kimidir.

- **send_display:** Siz seçilmiş **SIP INFO** mesajını telefonə yollaya bilərsiniz hansı(bəzi telefon modellərində) ki, mesajı telefonun ekranında göstərəcək.

Istifadə misali:

```
<action application="send_display" data="Support Call"/>
```

Bu istifadə edilə bilər ki, zəng edən tərəfin departamentini ya da bölməsini zəng edilən tərəfdə göstəriləsin.

Həmçinin çoxlu əmrlər mövcuddur ki, FreeSWITCH rəsmi saytından baxa bilərsiniz:

https://freeswitch.org/confluence/display/FREESWITCH/mod_dptools

mod_sofia

mod_sofia əmri SIP altında olan bütün imkanlar üçün əlçatilandır. Bura həmçinin son nöqtələr üçün SIP zənglərinin göndərmə və qəbul edilməsi, SIP qeydlərinin idarə edilməsi və kontakt məlumatları. **mod_sofia** çərçivəsində fərqi əmrlər mövcuddur hansı ki, təkcə zənglərin qəbul edilməsi və inisializasiya işini görmür, həmçinin son nöqtənin idarə edilməsi və dağıdırılması işini görə bilər(Misal üçün qeydiyyatdan keçmiş IP ünvan və alət NAT arxasındadırımı)

Baxmayaraq ki, **Sofia** sizə birbaşa Diaplan-dan yerinə yetiriləcək programları vermir, bu çoxlu əmrlər parametrlərində istifadə edilir.

Sofia ümumiyyətlə SIP zənglərin körpülənməsində yetki alır. **A** ayağının yeni inisializasiya edilmiş **B** ayağına qoşulmasına **Bridging** deyilir. Zəngləri körpüləndirirsizsə, ümumi format aşağıdakı kimidir:

```
<action application="bridge" data="sofia/profile/endpoint[@domain]"/>
```

Gəlin data porsiyasını analiz edək.

bridge əmri ümumi məqsədli əmr olduğuna görə, o birbaşa olaraq SIP zəngləri körpü eləmir. Siz öncə **sofia/** parametrini təyin etməlisiniz ki, SIP zəng etməyinizi təyin edəsiniz. **sofia/** parametrindən sonra zəngin qoşulması üçün hansı yolu və ya SIP profilini istifadə etməni yazmalısınız. Əgər siz yol olaraq öz SIP quraşdırmalarınızda olanı təyin eləmisinizsə, qayıdan server üçün domain adı artıq məlum olur(yeni **@domain** yazmağa gərək qalmır) və zəng sətirinin sonuna əlavə edilməli deyil. Əgər siz profaylin adını ikinci parametr kimi təyin edirsizsə SOFIA-ya deyirsiniz ki, zəng qoşulması üçün hansı IP ünvanı, Port və parametrlər istifadə edilməlidir. Bu halda siz domain adı ya da IP ünvanı **bridge** programının sonunda təyin etməlisiniz. Axırda, **endpoint** parametri uzaq sistemə göndərilecək istifadəçi adını təyin

edir. Bu adətən DID ya da genişlənmə rəqəmi olur ki, qarşı tərəf onlarla kimin əlaqəyə girməsini bilsin.

Aşağıda müxtəlif hallar üçün zəngin bridge edilməsi göstərilir. Aşağıdakı misallarda təsəvvür edək ki, bizim **supersip** adlı gateway-imiz və **external** adlı sofia profilimiz var.

```
<action application="bridge" data="sofia/supersip/+14158867900"/>
```

Bu zəng edəni (415) 886-7900 nömrəsi ilə **supersip** təçizatçı məlumatı istifadə edərək bridge edəcək.

```
<action application="bridge"  
data="sofia/external/+14158867900@sip.supersip.com"/>
```

Bu zəng edəni (415) 886-7900 nömrəsi ilə **supersip** təçizatçı istifadə edərək bridge edəcək amma, biz zəngi **external** təchizatçı üzərindən edəcəyik(hansi ki, port faylında IP ünvan və port təyin edilmişdir) və zəngi **sip.supersip.com** təçizatçısı serverinə yönləndiririk.

```
<action application="bridge"  
data="sofia/external/someuser@otheroffice.com:5080"/>
```

Bu zəng edəni **otheroffice.com** serverdə olan və **5080**-ci portda qulaq asan **someuser** adlı istifadəçiye bridge edəcək. Bu o deməkdir ki, siz zəngləri FreeSWITCH, Asterisk və istənilən VoIP təçizatçı arasında bridge edə bilərsiniz. Bu üsulla istifadəçi adı, server, port və zənglər üzərində olan yönləndirilmə tam olaraq sizin əlinizdədir.

Artıq xeyli misallar çəkdiyimizdən sonra gördük ki, sofia parametrləri zəng sətirinin sonunda alır ki, qabaqcıl opsiyaları yerləşdirərək zəngin özünü necə aparmasını təyin etsin. Misal olaraq deyə bilərik ki, susmaya görə olan UDP protocol əvəzinə TCP istifadə eləmək istəyirik. Zəng sətirinin sonuna nöqtə vergül qoymaqla bunu edə bilərsiniz. Bu tapşırıq üçün biz **transport=tcp** opsiyasını zəng sətirinin sonuna əlavə edirik. Bu aşağıdakı kimi ola bilər:

```
<action application="bridge" data=  
"sofia/external/+14158867900@sip.supersip.com;transport=tcp"/>
```

Misalların göstərməyimizin vacibliyi və məqsədi Sofia sistemlərinin Dialplan üzərində olan güclərinin göstərilməsindən ibarətdir. Programlar profile-larda təyin edilən başlanğıc bridge funksionallıqlarının istifadəsi ilə limitlənmirlər və siz istənilən ünvanda olan zənglərə yaradıcı olan dəyişənlər, opsiyalar və Dialplan funksiyaları istifadə edərək qoşula bilərsiniz.

mod_commands

Sistem inzibatçısına CLI-dan əmrlər üçün şərait yaradır. Bəzi hallarda FreeSWITCH-də CLI əmrləri zəng prosesində də xeyirli ola bilər. O halda ki, ümumiyyətlə CLI əmrləri Dialplan-dan çağırılan programlardan fərqlidir yenə də istədiyiniz əmri CLI-dan sətir kimi daxil edə bilərsiniz.

Misal olaraq **hupall**(NORMAL_CLEARING) CLI əmri normalda bütün aktiv zəngləri kəsir(xətdə kəsir) və onları **NORMAL_CLEARING** səbəbi ilə ayırır. Bu əmr adətən yalnız CLI-dan işə düşür. Əgər bunun 999 nömrəsinə zəng edərək Dialplan-dan mövcud olmasını istəsəniz, aşağıdakı genişlənməni təyin edə bilərsiniz.

```
<extension name="Make API call from Dialplan">
<condition field="destination_number" expression="^(999)$">
<action application="set" data="api_result=${hupall(normal_clearing)}"/>
</condition>
</extension>
```

Nəzər yetirin ki, tünd qara rənglər qeyd etdiyimiz **hupall** əmri CLI-dan **\${hupall(normal_clearing)}** ifadəsi kimi genişlənmə daxilində işə salacaq. Əlavə olaraq, əmrin nəticəsi **api_result** dəyişəninə mənimsədiləcək ona görə ki, **set** əmrini ifadədə istifadə elədik.

Göstərdiyimiz misal real həyatda elə də istifadə edilən olmasa da, göstərdik ki, Dialplan-dan CLI əmrlərini yollamaq mümkündür. FreeSWITCH üzərində olan əmrlərin tam siyahısını əldə etmək üçün https://freeswitch.org/confluence/display/FREESWITCH/mod_commands linkinə müraciət edə bilərsiniz.

Istifadə edilən dəyişənlər

Bura gələnədək kanal dəyişənlərinin başlanğıc istifadəsi haqqında çox danışdıq. FreeSWITCH-in dəyişənlərin istifadə edilməsi üçün global dəyişənləri də əlavə eləməklə, çoxlu üsulları mövcuddur. Gəlin başa düşmək üçün bu dəyişənlərin müəyyən bir hissəsinə baxaqla.

Müntəzəm ifadələrlə dəyişənlərin sınaqdan keçirilməsi

Biz artıq XML tag-ın daxilində şərtin istifadə edilməsi və məqsədi haqqında artıq danışmışdıq. Artıq sizin sınaqdan keçirə bilməniz üçün zəngi emaldan keçirilməsində fərqli elementlər haqqında danışacayıq.

FreeSWITCH sizin sınaqdan keçirməniz üçün dəyişənləri 3 ümumi hissəyə böllür - zəng edən profile sütunları, kanal dəyişənləri və global dəyişənlər. Əlavə olaraq siz macros və API əmrlərdən istifadə edə bilərsiniz ki, onların çıxışlarını şərtlərinizdə istifadə edəsiniz. Bunlardan hər birinə detalllı şəkildə baxacayıq.

Zəng edənin profile sütunları

Zəng edənin profile sütunları dəyişənləri onun qeydiyyatdan keçdiyi müddətdə alınır. Bu dəyişənlər **directory** ilə təyin edilir və zəng edənin ərazi kodunu,

kodek quruluşunu və.s. məlumatları əlavə edə bilər. Siz Dialplan-ı emal elədikdə zəng edənin profile sütunlarını aşağıdakı kimi istifadə edə bilərsiniz:

```
<condition field="${caller_profile_field}">
```

Bu dəyişənlər sizin qovluq çərcivəsində aşağıdakı kimi təyin edilir:

```
<user id="bob">
  <variables>
    <variable name="caller_profile_field" value="1234"/>
  </variables>
</user>
```

Bu misalda bob adlı istifadəçi qeydiyyatdan keçdikdə hal-hazırkı zəng profile-ni təyin elədi. Nəticə budur ki, bu profile tərkibində olan bütün dəyişənlər digər kanal dəyişənləri kimi əlçatılmalıdır.

Istifadəçi qovluğu 4-cü başlıqda SIP və istifadəçi qovluğunda açıqlanmışdır.

Kanal dəyişənləri

FreeSWITCH-də olan hər bir kanalın dəyişənləri sayəsində statusu izləmək, quraşdırımlar və zəng haqqında digər məlumatları əldə etmək ola bilər. Kanal dəyişənləri aşağıdakı formatda istifadə edilir:

```
${variable}
```

Kanal dəyişənləri Dialplanda, programda və qovluqda təyin edilə bilər. Onlar zəngin quruluşuna və gedişinə təsir edə bilər. Onlar Dialplan şərtləri, program əmrləri və digərləri kimi, dəyişənin emalı olan istənilən yerdə istifadə edilə bilər.

```
<condition field="${channel_variable}">
```

Kanal dəyişənləri demək olar ki, FreeSWITCH-də olan zəng emalının vacib hissələrindən biridir. Adı zəngin tərkibində emal edilmə üçün tələbdən də qat-qat çox olan dəyişənlər mövcuddur. Siz bütün dəyişənlər siyahısına <https://freeswitch.org/confluence/display/FREESWITCH/Channel+Variables> linkindən baxa bilərsiniz.

Kanal dəyişənləri və zəngin qurulması

Siz kanal dəyişənlərini, zəngi qurdugda ya da spesifik zəng ayaqlarını qurdugda təyin edə bilərsiniz. Misal üçün bridge əmri ilə **A** ayağı ilə **B** ayağını körpüledikdə. Bu nüsxələrdə kanal dəyişənlərinin təyin edilməsinin iki üsulu mövcuddur - {} fiqurlu mötərizələr və [] kvadrat mötərizələr. Hər biri fərqli işləyir və zəngin çoxlu tərəflərdə eyni zamanda bridge edilməsində istifadə edilir.

Fiqurlu mötərizələr zəng müddətində "**global**" olanda istifadə edilir. Bunu misal götürərək, zəngi Darren(istifadəçi) istifadəçinin GSM nömrəsinə 203-555-1212 bridge edirik. Voicemail-ə düşmənin qarşısını almaq üçün zəngi sadəcə **20** saniyəlik edirik.

```
<application action="bridge"
  data="{call_timeout=20}sofia/gateway/my_gw/2035551212"/>
```

Fiqurlu mötərizələrin daxilində olan dəyişən yalnız, `sofia/gateway/my_gw/2035551212` yeni kanal qurulduğda istifadə ediləcək. Indi isə gəlin Darren üçün ofis telefon əlavə edək. Ofis telefonu üçün zəngin getməsi müddətini 30 saniyə, GSM isə yenə də 20 saniyə edirik. Biz bunu kvadrat mötərizə ilə aşağıdakı kimi edə bilərik:

```
<application action="bridge"
data="[call_timeout=20]sofia/gateway/my_gw/2035551212
,[call_timeout=30]sofia/gateway/my_gw/4158867901"/>
```

Dəyişənlərin kvadrat mötərizədə təyin edilməsi ilə o hər bir zəng ayağı üçün spesifik zəng mənimsdədir.

Oeyd: Fiqurlu mötərizələr yalnız zəng sətirin əvvəlində keçərli olur. Həmçinin yeni sətirlər keçərli deyil və sizin bridge sətiri bir sətirdə olmalı, mötərizələr arasında və **sofia/** porsiyasında boşluq olmamalıdır.

Siz həmçinin "**clobber**" dəyişənlərini fiqurlu mötərizə daxilində istifadə edə bilərsiniz ki, gələcəkdə kvadrat mötərizələr daxilində istifadə edəsiniz. Siz **local_var_clobber** adlı flag təyin eləməlisiniz ki, bu işi görəsiniz. Öncəki misalımızı yenidən göstərərək istənilən ayaq üçün "**default**" vaxt bitməsini 30 saniyə və yalnız GSM üçün 20 saniyəni aşağıdakı kimi təyin edə bilərik:

```
<application action="bridge" data="{local_var_clobber=true,call_timeout=30}
[call_timeout=20]sofia/gateway/my_gw/2035551212
,sofia/gateway/my_gw/4158867901"/>
```

Çoxlu dəyişənlərin təyin edilməsi üçün vergül ayricisından istifadə edə bilərsiniz. Misal üçün aşağıdakı kimi təyin edə bilərsiniz:

```
{call_timeout=20,sip_secure_media=true}
```

Öncəki kodun sayəsində bütün kanallara iki dəyişən təyin edilə bilər ya da:

```
[call_timeout=20,sip_secure_media=true]
```

Öncəki kod seçilmiş kanallara təyin etmək üçün istifadə edilə bilər. Enterprice originate-lə spesifik notasiya istifadə edilir. Kvadrart mötərizələr əvəzinə(hər kanal üçün) ya da fiqurlu mötərizələr əvəzinə bundan-kiçikdir və bundan-böyükür simvolları istifadə edirik. Aşağıdakı misala baxın:

```
<action application="bridge" data="<ignore_early_media=true>{var1=val1}
sofia/gateway/my_gw/${dest1}:_{:}{var1=val2}sofia/gateway/my_gw/${dest2}" />
```

Zəng sətirinin əvvəlində **<ignore_early_media=true>** istifadə edilməsini qeyd edin. Bu **ignore_early_media** dəyişənini məcbur edir ki, hər iki "**originates**" üçün **true** təyin eləsin.

Enterprice originate istifadəsi haqqında bu başlığın növbəti XML Dialplan hissələrində danışılacaq.

Qlobal dəyişənlər

FreeSWITCH işə düşən kimi, o sizin mövcud XML quraşdırmanızı RAM-a yükleyir. Bu proses müddətində o aşağıdakı kodu axtarır:

```
<X-PRE-PROCESS cmd="set" data="domain=127.0.0.1"/>
```

Bu kod qlobal dəyişənləri təyin edir.

Global dəyişənlər FreeSWITCH işə düşdükdə bu inisializasiya yüklənmə prosesində genişlənir. **X-PRE-PROCESS** tag-ı faktiki XML yüklenməsində yerinə yetiriləcək əmri təyin edir. Siz dəyişəni bu faza müddətində təyin etsəniz, bu dəyişən faktiki olaraq qlobal elan edilir və XML-in digər ünvanından proqramlar üçün **\$\$ {variable}** kimi mümkün olur.

Həmçinin qeyd edin ki, siz öz XML-inizdə **\$\$ {variable}** istifadə elədikdə, oda XML yüklənməsi zamanı təyin edilmiş **X-PRE-PROCESS** tag-la dəyişdiriləcək.

```
<X-PRE-PROCESS cmd="set" data="domain=127.0.0.1"/>
<param name="domain" value="$$ {domain}"/>
```

Misal üçün öncəki XML kodu birləşdirilib FreeSWITCH tərəfindən bir sətir kodu kimi aşağıdakı qaydada edilə bilər:

```
<param name="domain" value="127.0.0.1"/>
```

Bu funksionallıq XML analizatorunun bacarığıdır FreeSWITCH deyil. Qlobal dəyişənlərin ilkin emalı FreeSWITCH tərəfindən istifadə edilən proses ya da hadisənin XML faylından öncə yerinə yetirilir.

FreeSWITCH kompilyasiya edilmiş çıxışı diskə çıxarır. Siz bu fayla baxaraq öz pre-prosessor əmrlərinizlə nə baş verdiyini və global elan edilmiş dəyişənləri görə bilərsiniz. Adətən bu fayl **/usr/local/freeswitch/log/freeswitch.xml.fsxml** ünvanında yerləşir.

Siz öz şərtlərinizdə global dəyişənləri, öz dəyişəninizi və parametr elanını aşağıdakı kimi təyin edə bilərsiniz:

```
<condition field="$$ {global_variable}">
```

Dialplan funksiyaları

Dialplan funksiyaları - funksiyanın kiçik hissələrindən ibarətdir hansı ki, Dialplan şərtlərini emal edərək real vaxtda işləyir. Onlar istifadə edilə bilər ki, öz şərtlərinizi yazdıqda daha geniş idarəetmə bacarığı və dinamiklik gətirsin.

Dialplan funksiyaları faktiki olaraq dialplan-dan kənarda da istifadə edilə bilər. Onlar XML ilə əlaqəli deyillər və FreeSWITCH sətir prosessoru çağırılan istənilən yerdə işə salına bilər. Digər yerlərin nüsxələri, həmçinin kənar scriptlərdə əlavə edilə bilər ki, dəyişənləri təyin eləsin(bu scriptlər dəyişənlər təyin edir, proqramları **bridge** və **transfer** edir).

Dialplan funksiyaların ümumi formatı:

```
 ${api_func(api_args ${var_name})}
```

api_func olan yerdə Dialplan funksiyası yazılır, **api_arg** funksiyaya ötürüləcək argumentin adı və **\${var_name}** işə istəkdən asılı olan dəyişəndir hansı ki, funksiyaya ötürülür. **api_arg** üçün format və gözlənilən parametrlər istifadə edilən funksiyalardan asılıdır. Hər bir mövcud olan Dialplan funksiyası aşağıdakı seksiyada daha detallı açıqlanır.

Qeyd: Virtual olaraq **fs_cli**-dan yerinə yetirilən istənilən API həmçinin **\${api{args}}** notasiyası istifadə edilərək Dialplandan da yerinə yetirilə bilər.

Real vaxt şərt dəyəri

Siz şərt dəyərini şərt ifadələrində **cond** funksiyasını istifadə edərək təyin edə bilərsiniz.

condition funksiyasının ümumi formatı aşağıdakı kimidir:

```
 ${cond(<expr> ? <true val> : <false val>)}
```

Misal olaraq şərt funksiyasının istifadə edilməsi:

```
 ${cond(5 > 3 ? true : false)}
```

Bu ifadə **true** qaytarmalıdır. Izin verilən müqayisələr aşağıdakılardır:

- **==** bərabərdir deməkdir
- **!=** bərabər deyil deməkdir
- **>** böyükdür
- **>=** böyük ya da bərabərdir
- **<** kiçikdir
- **<=** kiçik ya da bərabərdir

Qeyd edin ki, sətirləri sətirlərlə və rəqəmləri rəqəmlərlə müqayisə edə bilərsiniz amma, əgər siz sətiri rəqəmlə müqayisə edirsinizsə onlar **strlen(string)** və **number** kimi müqayisə ediləcək.

Sətir şərtləri

Siz dəyişənlər mənasının porsiyasını dəyişəni **\${var:offset:length}** tag-lərə çevirməklə seçə bilərsiniz (misal üçün istənilən program dilində olan **substr**).

Aşağıdakılardır:

- **var:** Sətir dəyişəni. Bu hərfi sətir ya da **\${caller_id}** kimi dəyişən ola bilər.
- **offset:** Datanın nüsxələnməsi üçün ünvan. **0** mənəsi ilk simvolu bildirir.
- **length:** Axtarılacaq simvolların rəqəmi. Bu istəkdən asılıdır və təyin edilməyibsə, sətirin qalanı nüsxələnir.

Aşağıdakılardır:

```
 var = 1234567890
 ${var:offset:length}
 ${var:0:1} // 1
 ${var:1} // 234567890
 ${var:-4} // 7890
 ${var:-4:2} // 78
 ${var:4:2} // 56
```

Misal olaraq bu API-1 çağıraraq çıxan zəngin Caller ID-sində ilk 3 rəqəmin(U.S. ərazi kodu) tutulması **\${callerid}** adlı dəyişəndə Dialplan üzərindən aşağıdakı kimi, olacaq:

```
<application name="set" data="areacode=${callerid:0:3}"/>
```

0-a bərabər ya da kiçik olan istənilən istifadə edildikdə sətirin sonuna spesifik pozisiyadan qaytaracaq.

Verilənlər bazası müraciətləri

Siz təsadüfi şəkildə **insert**, **delete**, **select** və **update** mənalarını daxili FreeSWITCH daxili bazasından edə bilərsiniz.

Verilənlər bazası üçün ümumi əmr aşağıdakı kimidir:

```
 ${db(command/realm/key/value)}
```

Verilənlər bazası əmrləri **insert**, **select** ya da **delete** ardınca cədvəl ya da ərazi, ardınca açar və mənası gəlir.

Misalda olduğu kimi, biz spesifik program gözləmədə musiqini Caller ID-yə əsaslanaraq yaza bilərik:

```
<action application="playback"
data="${db(select/music/${caller_id_number})}"/>
```

Digər misalda olduğu kimi, biz datanı bazaya **insert** edə bilərik. Bu misalda biz mövcud zəng edənin UUID-sini **spymap** adlı cədvələ zəng edənin Caller ID-sini açar olaraq istifadə edərək əlavə edəcəyik. Kimsə sonra son UUID-ə əsaslanan spesifik Caller ID-ni əldə edə biləcək.

```
<action application="db" data="insert/spymap/${caller_id_number}/${uuid}"/>
```

Siz FreeSWITCH bazasında yazdıqda sisteminiz üçün SQLite ya da ODBC ilə verilənlər bazası quraşdırılısınız. Bu sizin datanın həmişəlik saxlanılmasına kömək edəcək. Bəzi hallarda bu praktik ya da istənilən deyil və sizin müvəqqəti datanız olur hansı ki, yalnız ram-da qalması tələb edilir. Bu tip hallarda öncə danışdığımız kimi, **db** üçün **hash** çıxarmırıq.

Misal üçün biz hash-dən darta bilərik:

```
<action application="playback"
data="${hash(select/music/${caller_id_number})}"/>
```

Ya da biz hash-də saxlaya bilərik:

```
<action application="hash" data="insert/spymap/${caller_id_number}/${uuid}"/>
```

Hash sadəcə yaddaş daxili cədvəldir hansı ki, **key/value** cütlüyünü saxlayır. Əgər siz FreeSWITCH-i yenidən işə salsanız, hash-də olan istənilən data-nı itirəcəksiniz.

SIP əlaqə parametrləri

Siz **sofia_contact** əmrini istifadə edərək, qeydiyyatdan keçmiş Sofia kontakt-1 üçün əlaqə sətirini və parametrlərini əldə edə(və idarə edə bilərsiniz) bilərsiniz:

```
 ${sofia_contact(profile/foo@bar.com)}
```

Bu çoxlu səbəblərdən istifadə ediləndir. Misal üçün deyə bilərik ki, istifadəçinin harda yerləşdiyinə görə NAT təyin edilməsi ya da istifadəçi ümumiyyətlə qeydiyyatdan keçibmi. Daha ümumi misal desək, siz bu imkanı istifadə edə bilərsiniz ki, kontakt sətirini hissələrə böləsiniz və gələcəkdə istifadə edəsiniz.

Aşağıdakı XML kodu [foo@domain.com](#) istifadəcisinə baxış keçirir və istifadəçi domain adı ya da IP ünvanını və contact parametrlərini qeydiyyat formasının contact sətirindən əldə edir. O istifadəçi adını istifadəçi contact yazısının əvvəlindən ayıracaq.

```
<condition field="\${sofia_contact($user_id@domain.com)}">
  expression="^[@]+@(.*)"
    <action application="set" data="to_domain=$1"/>
</condition>
```

[domain.com](#)-i özünüzə aid olanla dəyişin. Müntəzəm ifadəyə diqqət yetirin:
`^[@]+@(.*)`

Bu nüsxə sətirin əvvəlindən @ simvolu tapanadək uyğun gəlir və sonra, @ simvolundan sonra gələn istənilən məlumatı \$1 parametrinə ötürür.

Istifadəçi adını ayırdıqdan sonra, siz onu yeni istifadəçi adı ilə dəyişə bilərsiniz. Bu adətən DID-in müştəri PBX-nə yönləndirilməsində istifadə edilir - siz qəbul edən istifadəçinin adını DID adı ilə dəyişə bilərsiniz:

```
<condition field="\${sofia_contact($user_id@domain.com)}">
  expression="^[@]+@(.*)"
    <action application="bridge" data="sofia/external/${DID_number}@$1"/>
</condition>
```

Bu misalda əgər dəyişən \${DID_number} sütununda təyin edilibsə, o istifadəçinin IP ünvanı və kontakt yönləndirmə məlumatı ilə birləşdiriləcəkdi. Beləliklə əgər istifadəçi [frank@72.44.12.28](#) kimi qeydiyyatdan keçibse, o bizim misalda [2035551212@72.44.12.28](#) kimi dəyişdirilə bilər.

Qeyd: Əgər sizin istifadəcilərin öz alətlərini qeydiyyatdan keçirmələri üçün çoxlu SIP profilləriniz varsa, o halda siz **sofia_contact** istifadə elədikdə qeydiyyat müddətində **ERR/USER_NOT_REGISTERED** səhvləri ala bilərsiniz. sofia_contact əmri işə salındıqda bu səhvi * istifadə eləməklə düzəldin:

```
 sofia_contact(*@user@domain)
 * simvolu Sofia-ya deyəcək ki, istifadəçi üçün bütün SIP profillərə çat.
```

Set, export və ayaqlar

Iki fərqli zəng ayaqlarına qoşulmaq üçün bridge elədikdə, görə bilərsiniz ki, sizin ayağın(A ayağı) çıxışında kanal dəyişəni var və eynilə onun B ayağında olmasına da istəyirsiniz. Elə hallar ola bilər ki, konkret olaraq hansısa bir

ayaqda təyin eləmək istəyirsiniz. Bu bölmədə göstərilən göstərilən metodlar həmin problemin həlli üçün üsullardır.

set qarşıdır export-a

Zənglər haqqında məlumatın təyin edilməsi, dəyişdirilməsi üçün və zəng emalının keçirilməsi üçün, iki ümumi Dialplan programı mövcuddur. Bu programlar **set** və **export** adındadır.

set programı kanal üçün dəyişənləri kanal aralığı üçün təyin edir. Bu dəyişənlər sonra programlar tərəfindən əldə edilə ya da Dialplan şərt sınaqlarında yoxlanıla bilər. Biz bu kitab içində artıq set programından dəfələrlə istifadə eləmişik.

export programı sonradan **set** programını götürür. O dəyişənləri mövcud kanala təyin edir amma həmçinin gələcək yaradılmış kanalların istifadəçi üçün ya da Dialplan konteksti üçün dəyişənləri yadda saxlayır. Digər sözlə desək, export dəyişənləri hər iki tərəfə yeni mövcud **A** zəng ayağı və gələcək **B** zəng ayaqlarına təyin edir.

Bu iki program arasında olan fərq çox kiçik ola bilər və siz bu tələbi B ayağında (zənglərin transfer edilməsi) olan məlumatı düzgün almadığınız anda hiss edəcəksiniz. Export programı çoxlu zəng hissələrində ziddiyətsizlik yaratmaq üçün çox yararlı olur.

Aşağıdakı misallara baxın:

```
<!-- "foo" dəyişəni hər iki ayaqda təyin edilib -->
<action application="export" data="foo=bar"/>
<action application="bridge" data="/user/1001"/>
<!-- "foo" dəyişəni yalnız b ayaqda təyin edilib -->
<action application="export" data="nolocal:foo=bar"/>
<action application="bridge" data="/user/1001"/>
```

Elə hallar ola bilər ki, siz foo dəyişənini hər iki ayaqda təyin etməlisiniz. Ancaq elə hallar da ola bilər ki, misal üçün CDR-ların işə salınmasında müəyyən mənəni yalnız **B** ayağında işə salmaq istəyirsiniz. Tünd qara hərflərlə olan nüsxə **nolocal**: directivi bunu göstərir hansı ki, kanal dəyişəni foo üçün bar mənasını **B** ayağında təyin edir amma, kanal dəyişəni foo-nu A ayağına təyin eləmir.

Dəyişənlərin zəng başlıqları ilə ötürülməsi

Bəzi hallar ola bilər ki, siz çıxış zənglərində özünüzə aid olan başlıqları təyin etmək istəyirsiniz. SIP stack bu işin görülməsi üçün əsas vacib yerdir.

Siz çıxış SIP zənglərinə eyni set və export istifadə edərək, təsadüfi başlıqları əlavə edə bilərsiniz amma, dəyişən adlarını **sip_h_** prefiksi ilə təyin eləmək lazımdır. Misal üçün **CallerLikesTacos=1** header-ni zəngə əlavə eləmək üçün, siz **set** programının prioritetini **bridge** programına aşağıdakı kimi təyin edə bilərsiniz:

```
<action application="set" data="sip_h_X-CallerLikesTacos=1"/>
<action application="bridge" data="sofia/mydomain.com/1000@example.com"/>
```

Əgər siz başlıqların **BYE** müraciətinə əlavə edilməsini istəyirsinzsə, siz **sip_bye_h_** prefiksini kanal dəyişənində istifadə eləməlisiniz.

Qeyd: Əgər tələb edilmirsə siz öz headerlərinizi **X-** prefiksi ilə təyin eləməlisiniz ki, SIP-in digər stekləri ilə funksional uyğunluq problemi olmasın. **X-** başlıqları adətən digər header-lər kimi görünür və əgər təyin edilməyib, məhəl qoyulmur.

XML Dialplan qısaca

Biz gündəmdə ümumi istifadə ediləcək ssenarilərdən bir neçəsini açıqladıq. Burda göstərilən nüsxələr sınaqdan keçirilməsi üçün tam işlək vəziyyətdə göstərildi. Bu ssenariləri öz sınaqlarınızda dəyişib yoxlamaqdən çəkinməyin.

Zəng edənin nömrəsi və IP ünvanı ilə tutuşdurma

Aşağıdakı misalda göstərilən genişlənmə o halda seçiləcək ki, zəng edilən son nöqtənin IP ünvanı **192.168.1.1**-dir. Ikinci şərtdə zəng edilən nömrə **\$1** dəyişəninində açılmış və bridge dəyişəninə əlavə edilmişdir ki, qayda ilə **192.168.2.2** IP ünvanına zəng eləmək olsun:

```
<extension name="Test1">
  <condition field="network_addr" expression="^192\.168\.1\.1$"/>
  <condition field="destination_number" expression="^(\d+)$">
    <action application="bridge" data="sofia/profilename/$1@192.168.2.2"/>
  </condition>
</extension>
```

İlk şərt sütunu slash ilə kəsilir. Son şərt sütununda action tag olur hansı ki, **</condition>** tag ifadəsi ilə bağlanır. Həmçinin nəzərə alın ki, öncəki nüsxə bu misalla eyni deyil:

```
<extension name="Test1Wrong">
  <condition field="destination_number" expression="^(\d+)$"/>
  <condition field="network_addr" expression="^192\.168\.1\.1$">
    <action application="bridge" data="sofia/profilename/$1@192.168.2.2"/>
  </condition>
</extension>
```

Test1Wrong misali zəngi düzgün yönləndirməyəcək ona görə ki, **\$1** dəyişəninin heç bir mənası yoxdur ona görə ki, tam rəqəm fərqli şərtlərin sütunlarında mövcud olmuşdur.

Siz həmçinin Test1Wrong misalının səhvini ilk şərtdə dəyişən təyin eləməklə aradan qaldırı bilərsiniz hansı ki, sonra ikinci şərtin daxilində işi yerinə yetirəcəksiniz:

```
<extension name="Test1.2">
  <condition field="destination_number" expression="^(\d+)$">
    <action application="set" data="dialed_number=$1"/>
  </condition>
  <condition field="network_addr" expression="^192\.168\.1\.1$">
    <action application="bridge"
data="sofia/profile/${dialed_number}@192.168.2.2"/>
  </condition>
</extension>
```

Siz set dəyişənini genişlənmənin daxilində növbəti şərtlər/uyğunlamalar üçün istifadə edə bilərsiniz ona görə ki, işi çağırılarda genişlənmə dəyərləndirilir.

Əgər siz genişlənmənin daxilində dəyişənlərə əsaslanan fərqli işlərin görülməsini təyin eləmək istəyirsinizsə siz ya dəyişənin təyin edilməsi üçün **execute_extension** istifadə eləməlisiniz ki, zəngi yönləndirəsiniz ya da daxili emali istifadə edin(Bu başlıqda əvvəldə olan *Daxili* yerinə *yetirilməyə baxın*)

IP ünvan və Caller ID-nin tutuşdurulması

Bu misalımızda 1 prefaksi ilə zəng edilən rəqəmi tutuşdurmalı və daxil olan IP ünvanı eyni zamanda tutuşdurmalıyıq:

```
<extension name="Test2">
  <condition field="network_addr" expression="^192\.168\.1\.1$"/>
  <condition field="destination_number" expression="^1(\d+)$">
    <action application="bridge" data="sofia/profilename/$0@192.168.2.2"/>
  </condition>
</extension>
```

Burda da həmçinin **^1(\d+)\$** qaydası ilə müqayisə edirik, **\$1** dəyişəni istifadə eləmədik hansı ki, yalnız **1** rəqəmindən sonrakı hissəni götürəcəkdir. Onun yerinə biz **\$0** dəyişəni istifadə elədik hansı ki, original mənseb nömrəni tərkibində saxlayır.

Nömrənin tutuşdurulması və rəqəmlərin ayrılması

Biz bu misalımızda zəng edilən nömrə 00-la başlayanı tutuşdurmalıyıq ancaq, həmçinin aparıcı rəqəmləri ayırmalıyıq. Təsəvvür edək ki, FreeSWITCH 00123456789 nömrəsini qəbul edir və biz 00 rəqəmlərini ayırmalıyıq. Onda aşağıdakı genişlənməni istifadə edə bilərik:

```
<extension name="Test3.1">
  <condition field="destination_number" expression="^00(\d+)$">
    <action application="bridge" data="sofia/profilename/$1@192.168.2.2"/>
  </condition>
</extension>
```

Digər sözlə, əgər siz rəqəm olmayanları gözləyirsinizsə ya da siz rəqəmlərdən fərqli olanları da tutuşdurmaq istəyirsinizsə, **\d+** əvəzinə **.+** istifadə edin ona görə ki, **\d+** yalnız sayılan rəqəmləri tutuşdurur hansı ki, **.+** hal-hazırkı pozisiyadan sətirin sonunadək bütün simvolları tutuşdurur.

```
<extension name="Test3.2">
  <condition field="destination_number" expression="^00(.+)\$">
    <action application="bridge" data="sofia/profilename/\$1@192.168.2.2"/>
  </condition>
</extension>
```

Qeyd: Texniki olaraq, biz istəmədiyimizi kənarlaşdırırıq amma, istədiyimiz rəqəmləri tuturuq. İlk dairəvi mötərizə daxilində olan tutusma **\\$1**-də saxlanılır. Məqsəd odur ki, bizim rəqəmlərimiz və onları dəyişənə mənimsədib istədiyimiz məqamda istifadə etmək imkanı olsun.

Nömrəni tutuşdur, rəqəmləri ayır və prefiks əlavə et

Bu misalda biz öndə gələn rəqəmləri ayırmalıyıq ancaq, zəng edilən nömrənin önünə yeni prefix-də əlavə eləməliyik. Nəzərdə tutulur ki, həmin FreeSWITCH **00123456789** nömrəsini qaytarır və bize lazımdır ki, **00** nömrəsini **011** ilə əvəz edək. Aşağıdakı genişlənməni istifadə edə bilərik.

```
<extension name="Test4">
  <condition field="destination_number" expression="^00(\d+)\$">
    <action application="bridge" data="sofia/profilename/011\$1@x.x.x.x"/>
  </condition>
</extension>
```

Qeydiyyatdan keçmiş alətə zəng edək

Bu misal sizin FreeSWITCH sisteminizdə qeydiyyatdan keçmiş alətin necə bridge edilməsini göstərir. Bu misalda biz nəzərdə tuturuq ki, sizin **local_profile** adlı Sofia profile-niz qurulmuşdur və sizin telefonları **example.com** domain adına qoşulurlar. Qeyd zəng sətirində @ əvəzinə % yazın:

```
<extension name="internal">
  <condition field="source" expression="mod_sofia"/>
  <condition field="destination_number" expression="^(4\d+)\$">
    <action application="bridge" data="sofia/local_profile/\$0@example.com"/>
  </condition>
</extension>
```

@ əvəzinə % istifadə edilməsi FreeSWITCH-ə xas olan spesifikasiyadır. **user%domain** FreeSWITCH-ə deyir ki, istifadəçi bu domain adı ilə qeydiyyatdan keçmişdir və bu domain FreeSWITCH kataloqu tərəfindən xidmət edilir.

A tərəfi və sonra B tərəfi yoxla

Aşağıdakı misal göstərir ki, necə birinci işdə problem olarsa ikincini çağırmaq olar.

Əgər ilk iş uğurlu olarsa, zəng 111@example1.company.com -a bridge ediləcək və tərəflərdən biri dəstəyi asanadək mövcud olacaq. Bundan sonra, heç bir iş

yerinə yetirilməyəcək çünki, zəng edənin kanalı bağlanmışdır (Digər sözlə, 1111@example2.company.com-a zəng edilməmişdir).

Əgər ilk 1111@example1.company.com-a olan çağırış uğurlu olmazsa, kanal bağlanmayıcaq və ikinci işi çağıracaq.

```
<extension name="find_me">
  <condition field="destination_number" expression="^1111$">
    <action application="set" data="hangup_after_bridge=true"/>
    <action application="set" data="continue_on_fail=true"/>
    <action application="bridge"
data="sofia/local_profile/1111@example1.company.com"/>
    <action application="bridge"
data="sofia/local_profile/1111@example2.company.com"/>
  </condition>
</extension>
```

DID-lərin genişlənmələrə yönləndirilməsi

Təyin edilmiş DID-lə gələn, daxil olan zənglərin public kontekst üzərindən seçilmiş genişlənme **inhouse**-a yönləndirilməsi üçün aşağıdakılari edin:

```
<context name="public">
  <extension name="test_did">
    <condition field="destination_number" expression="^\d{6}(\d{4})$">
      <action application="transfer" data="$1 XML inhouse"/>
    </condition>
  </extension>
</context>
```

Bu ancaq **10** rəqəmli nömrədən son **4** rəqəmini tutacaq və zəng edəni inhouse konteksti üzərindən bu nömrəyə yönləndirəcək. Nəzər yetirin ki, **\d{4}** ətrafında olan dairəvi mötərizələr bizə imkan verir ki, yalnız son **4** rəqəmləri tuta bilək.

Alternativ çıxış zəngləri

Bu misalımızda biz **10** rəqəmli zəngi **OfficeA**-dan **gateway1**-ə və **OfficeB**-dən **gateway2**-yə ötürəcəyik. Bu o deməkdir ki, OfficeA və OfficeB eyni FreeSWITCH istifadə edir amma çıxış zəngləri üçün fərqli yönləndirməyə tələb var. Bu o deməkdir ki, hər iki ofisin **4** rəqəmli genişlənməsi var. OfficeA-nın genişlənmələri **2** ilə, OfficeB-nın genişlənmələri isə **3** ilə başlayırlar.

```
<extension name="officeA_outbound">
  <condition field="caller_id_number" expression="^2\d{3}$"/>
  <condition field="destination_number" expression="^(\d{10})$">
    <action application="set" data="effective_caller_id_number=8001231234"/>
    <action application="set" data="effective_caller_id_name=Office A"/>
    <action application="bridge" data="sofia/gateway/myswitch.com/$1"/>
  </condition>
</extension>
<extension name="officeB_outbound">
  <condition field="caller_id_number" expression="^3\d{3}$"/>
  <condition field="destination_number" expression="^(\d{10})$">
    <action application="set" data="effective_caller_id_number=8001231235"/>
```

```
<action application="set" data="effective_caller_id_name=Office B"/>
<action application="bridge" data="sofia/gateway/otherswitch.com/$1"/>
</condition>
</extension>
```

Enterprice ilə çoxsaylı son nöqtələr

Bu misala baxsaq: Müştəri bilmək istəyir ki, onlar zəngi iki fərqli insana yönləndirə bilərlərmi. İlk şəxs(Alice) istəyir ki, onun stolüstü telefonu önce və sonra mobil telefonu zəng çalsın. İkinci şəxs(Bob) istəyir ki, onun stolüstü və mobil telefonları eyni anda zəng çalsınlar. Alice ya da Bob ilk telefona birinci cavab vermələrindən asılı olmayıaraq, biri telefona cavab verdikdə digər telefonlara gələn zənglər dayanacaq.

Bu çətin ssenari tələb edir ki, FreeSWITCH-i enterprise originate olaraq istifadə edəsiniz. Əsas idea ondan ibarətdir ki, hansısa şəxs daha da geniş mühitə qoşulur. Bizim misalda, Alice-in telefonuna aşağıdakı kimi zəng edilə bilər:

```
<action application="bridge" data="[leg_timeout=10]user/Alice|
[leg_timeout=20]sofia/gateway/my_gw/${alice_mobile}">
```

Və Bob üçün:

```
<action application="bridge" data="[leg_timeout=10]user/Bob|
[leg_timeout=20]sofia/gateway/my_gw/${bob_mobile}">
```

Alice ya da Bob üçün ayrılıqda olan körpüləmə cəhdi zəng üçün ayrılıqda işləyəcəkdi amma ikisinində birlikdə işləməyəcək. Bu işi görmək üçün isə, onların hər ikisini bir bridge işinin içində yerləşdirin və onları `:_:` simvolu ilə spesifik ayırıçı ilə ayırin.

```
<action application="bridge"
data=<ignore_early_media=true>[leg_timeout=10]user/Alice|
[leg_timeout=20]sofia/gateway/my_gw/${alice_mobile}:_|
[leg_timeout=10]user/Bob|
[leg_timeout=20]sofia/gateway/my_gw/${bob_mobile}">
```

Bu halda, əgər daxil olan ayaq **bridge** programına düşürsə, FreeSWITCH iki ayrıca "**originate**" insializasiya edəcək - biri Alice-ə çatmaq üçün və digəri isə Bob-a çatmaq üçündür. Burda FreeSWITCH-in effekti odur ki, o eyni zamanda Alice-in iş telefonu, mobil telefonu və eyni zamanda Bob-un iş telefonu, mobil nömrəsinə çatmağa çalışır. Əgər kimsə cavab verirse, onda bütün zənglər dayanır və zəng yalnız ilk cavab verənə qoşulmuş vəziyyətdə olur.

Nəzərə alın ki, **ignore_early_media=true** məcburi istifadə elədik ona görə ki, biz çoxlu zəng ayaqları yaradırıq. Elə bir üsul yoxdur ki, zəngi bir mənbəyə ötürək və istifadə edək. Əmin olun ki, əgər siz müəyyən bir signali zəng etdiyiniz tərəfə çatdırmaq istəyirsizsə, **ringback** və ya **transfer_ringback** dəyişənlərini təyin eləmişiniz.

Notice

Bu başlıqda biz FreeSWITCH Dialplan-da çox dərindən işləri araşdırırdıq. 5-ci başlıqda olan XML Dialplanın başa düşülməsinə əsaslanaraq biz Dialplan-da çoxlu qabaqcıl məntiqləri araşdırırdıq:

- Dialplan analizatorunun işləməsini
- Global dəyişənlər və kanal dəyişənlərinin istifadə edilməsini
- Qabaqcıl istifadə və müntəzəm ifadələr
- Fərqli qabaqcıl yönləndirmə konsepsiyaları

FreeSWITCH-də olan Dialplan sistemi dünyada olan öyrənə biləcəyiniz konsepsiyalardan çox vacib olanıdır. FreeSWITCH-in gücünün bir hissəsi həqiqətən Dialplan sistemi sayəsində də olur və FreeSWITCH-də fərqli funksiyaların istifadəsi sizi düzgün seçimə gətirib çıxaracaq.

Növbəti başlıqda biz FreeSWITCH-də statik XML fayllarına əsaslanmayan çox güclü olan quraşdırımlara son qoyacayıq.

9-cu başlıq

Statik XML quraşdirmaları sərhədinin aşılması

Bu andan etibarən, biz diqqətimizi FreeSWITCH-də susmaya görə yüklenmiş statik XML fayllarına ayıracayıq. Başlıqda FreeSWITCH-i minimal static XML fayllarla necə tam şəkildə quraşdırmanın yollarını öyrənəcəyik. FreeSWITCH bunun dinamik idarə edilməsi üçün müxtəlif yollar təklif edir. Həmçinin hər bir metodun fərqli istiqaməti var amma bu funksiyalardan bəziləri üst-üstə düşə bilər. Misal üçün, **mod_xml_curl** və dillə birləşmə sizə imkan yaradır ki, dinamik quraşdirmaları yaradasınız. Bu başlıqda biz aşağıdakı metodları açıqlayacaqıq:

- **mod_xml_curl:** Bu modul sizə imkan verir ki, quraşdırma faylini web serverdən götürəsiniz. Quraşdirmalara Dialplan-lar, istifadəçi qovluğu və ümumi quraşdırma faylları aiddir. Bu həmçinin serverin sıradan çıxdığı halında statik neytralizasiya ilə dinamik quraşdırma imkan verir.
- **Dillərin əlaqələndirilməsi vasitəsilə quraşdirmaların dinamik generasiyası:** **mod_xml_curl** istifadə elədikdə siz dəstəklənən dillərdən(Lua, Perl, Java, Python və.s.) istifadə edə bilərsiniz ki, dinamik quraşdirmaları generasiya edəsiniz.
- **Zənglərin API-la CLI-dan edilməsi:** originate API unikaldır ona görə ki, o sizin sistemdə yeni telefon zəngləri yarada bilər(burdan da **originate** adı əmələ gəlir). Biz **fs_cli** vasitəsilə originate API istifadə edərək yeni zəngin yaradılmasını qısa şəkildə göstərəcəyik.
- **Əmrlərin yerinə yetirilməsi üçün ESL-in istifadə edilməsi:** Event Socket və ESL(Event Socket Library) FreeSWITCH-in idarə edilməsi üçün çox güclü bir imkandır. Biz bu başlıqda Event Socket-ə qısa şəkildə baxacayıq və sonra daha detallı şəkildə 10-cu başlıqda FreeSWITCH-in kənardan idarə edilməsinə baxacayıq.

mod_xml_curl

mod_xml_curl modulu bildiyimiz CURL kitabxanalarını istifadə edir ki, WEB serverdən quraşdırma faylları dartsın. FreeSWITCH bu faylları analiz edə və istifadə edə bilər ona görə ki, o statik XML quraşdırma faylları istifadə edərdi. Siz WEB serveri idarə elədiyinizə görə, imkanınız var ki, bir müraciətdən digərinə gedən müraciətdə XML-in dəyişdirilməsi imkanından yararlanasınız. Bu bir WEB server vasitəsilə bir neçə FreeSWITCH server qurduqda xeyirli ola bilər. Bunun böyük bir üstünlüyü ondan ibarətdir ki, cluster-də işləyən serverlər üçün quraşdırma fayllarının dəyişikliyinin bir serverdən edilməsidir.

Bu başlıqdə olan bütün **mod_xml_curl** misalları PHP işləyə bilən WEB server tələb edir. Misallar **apache24** üzərində olan **mod_php** ilə sınaqdan keçirilmişdir və siz eyni mühiti Linux/UNIX və ya Windows-da qura bilərsiniz. Nəzərdə tutulur ki, WEB serverimiz artıq FreeSWITCH olan serverimizin üzərində qurulmuşdur və hazırda (Apache24, php5.6 və MySQL 5.6 qurulmuş vəziyyətdədir). Həmçinin FreeSWITCH kompilyasiya müddətində **src/freeswitch/modules.conf** faylında **xml_int/mod_xml_curl** aktivləşdirilmişdir.

FreeSWITCH yenidənyüklənməsində **mod_xml_curl**-in həmişə aktiv olmasını istəyirikse aşağıdakı addımları edirik:

1. **/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml** faylında aşağıdakı sətirin qarşısından **<!--** və sonundan **-->** şərh simvollarını silirik:
<load module="mod_xml_curl"/>
2. Faylı yadda saxlayırıq.

Qeyd: mod_xml_curl susmaya görə Windows məşinlərdə kompilyasiya edilir, amma UNIX/Linux-larda isə olmur. UNIX/Linux-lar üçün əmin olun ki, **mod_xml_curl**-in qarşısından **modules.conf** faylında şərh silinib və **make mod_xml_curl-install** əmri ilə yüklənilib.

Təsəvvür elədiyimiz kimi, biz FreeSWITCH-ə URL ünvan təqdim eləməliyik ki, tələb elədiyi quraşdırma faylların dərtılması üçün onu istifadə eləsin. Biz bunu **xml_curl.conf.xml** faylında edəcəyik:

3. **/usr/local/freeswitch/conf/autoload_configs/xml_curl.conf.xml** faylini istənilən mətn redaktorunda açın və aşağıdakı sətirləri əlavə edin:
<configuration name="xml_curl.conf" description="cURL XML Gateway">
<bindings>
<binding name="arbitrary_name">
<param name="gateway-url" value="http://localhost/index.php"
bindings="configuration|dialplan|directory"/>
</binding>
</bindings>
</configuration>
4. Faylı yadda saxlayırıq.
5. FreeSWITCH-i yenidən işə salırıq.

binding name atributu açıqlama verə biləcək istənilən məlumat ola bilər. binding daxilində olan param-da isə hər dəfə quraşdırma tələb olunacaq URL-

yazılır. Biz öz misalımızda **http://localhost** ünvanında olan PHP skriptdən istifadə edirik. Gördüyünüz kimi, binding atributunun ardınca biz **configuration**, **dialplan** və **directory** seçmişik ki, FreeSWITCH bunlar üçün WEB serverə müraciət edə bilsin(Biz bunu birazdan **index.php**-də **section** da müraciət edəcəyik).

Bu kritikdir ona görə ki, siz müəyyən ssenarini emal edirsiniz və FreeSWITCH sizin scriptlərdən müəyyən müraciət edir hansı ki, necə emal etməsini bilmir. Aşağıdakı cavabı istifadə edərək FreeSWITCH-ə deyin ki, siz müraciətin necə emal ediləcəyini bilmirsiniz və o daxili quraşdırma faylı axtarmalıdır:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="freeswitch/xml">
  <section name="result">
    <result status="not found"/>
  </section>
</document>
```

Bu o halda çox yararlı olur ki, siz yalnız müəyyən genişlənmələri və ya Dialplan-da olan konteskti emal elemək istəyirsiniz ya da siz yalnız bir neçə modul üçün spesifik opsiyaları ötürmək istəyirsiniz. Aşağıdakı seksiyalarda biz sizə bunun yerinə yetirilməsinin bir neçə misalını çəkəcəyik. Önce **index.php** faylini dolduraq hansı ki, bütün müraciətlər üçün sürücü rolunu oynayır. Aşağıdakı sətirləri **index.php** faylında VirtualHost adı üçün təyin elədiyiniz qovluqda yerləşdirmək lazımdır:

```
<?php
function not_found( $msg = '' ) {
print "<document type=\"freeswitch/xml\">\n";
print " <section name=\"result\">\n";
print "   <result status=\"not found\"/>\n";
print " </section>\n";
print " <!-- $msg -->\n";
print "</document>\n";
exit;
}
header( "Content-Type: text/xml" );
print "<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"no\"?>\n";
if ( !array_key_exists( 'section', $_REQUEST ) ) {
    not_found( 'no section passed' );
}
if ( !preg_match( '/^(directory|dialplan|configuration)$/' ,
$_REQUEST['section'] ) ) {
    not_found( 'section not valid' );
}
$gen_file = "{$_REQUEST['section']}.php";
if ( file_exists( $gen_file ) && is_readable($gen_file) ) {
    include $gen_file;
} else {
    not_found( '$gen_file not found' );
}

not_found() funksiyası yalnız tapmadığı XML blokunu çap edəcək hansı ki, haqqında önce danışıq. Biz qərar versək ki, static XML bloku müraciəti emal
```

not_found() funksiyası yalnız tapmadığı XML blokunu çap edəcək hansı ki, haqqında önce danışıq. Biz qərar versək ki, static XML bloku müraciəti emal

eləsin ya da müraciət ilə nə isə doğru olmazsa, onda biz cavab verəcəyik ki,
not found XML block.

Növbəti cüt sətir (əvvəli **header** və **print** olan) əmin olmaq istəyir ki, cavabı XML kimi təyin edən və uyğun olaraq emal edən FreeSWITCH-dirmi ya web browser.

Növbəti 3 sətir blok təyin edir ki, müraciətdə section olmasını təyin eləsin və bunun sayəsində də FreeSWITCH üçün geriyə qayıdacaq olanları təyin edirik. Əgər seksiya yoxdur, *not found XML* qaytaracayıq.

Əmin olmaq üçün, yoxlanışdan sonraki 3 sətir, **section**-un ya **dailplan**, **directory** ya da **configuration** olmasını təyin edir. Əgər istənilən digər müraciətlər cavab olaraq *not found XML* alarsa o, tələb elədiyi müraciəti FreeSWITCH serverin özündə daxildə axtarmalıdır.

Sonda olan **if** bizi əmin edir ki, emal üçün tələb edilən fayl yerindədir və biz onu oxuya bilirik.

Aşağıdakı seksiyada biz Dialplan müraciətlərinin **dialplan.php(index.php** ilə eyni qovluqda olacaq) faylında necə emal edilməsinə baxacayıq.

mod_xml_curl Dialplan

Bu seksiyada biz size necə **book_test** dialplan kontekstinin **mod_xml_curl** vasitəsilə emal edilməsini və digər kontekstlərin static XML Dialplan-a geri qayıtmasını göstərəcəyik. Aşağıdakı sətirləri **index.php** fayliniz yerləşən qovluqda **dialplan.php** adlı faylına əlavə edin:

```
<?php
if ( !array_key_exists( 'Hunt-Context' , $_REQUEST ) || $_REQUEST['Hunt-Context'] != 'book_test' ) {
    not_found('not our context');
}
print "<document type=\"freeswitch/xml\">\n";
print " <section name=\"dialplan\">\n";
print " <context name=\"$$_REQUEST['Hunt-Context']\">\n";
print " <extension name=\"no_name\">\n";
print " <condition>\n";
print " <action application=\"info\">\n";
print " </condition>\n";
print " </extension>\n";
print " </context>\n";
print " </section>\n";
print "</document>\n";
?>
```

Hər müraciətdə FreeSWITCH çoxlu **POST** parametrlərini bizim WEB servere yollayacaq. Bir cüt tez-tez istifadə edilən parametrlər **Hunt-Context** və **Hunt-Destination-Number**-dir hansı ki, statik XML Dialplan-da gördüyüümüz context-ə və **destioantion_number**-ə uyğundur. **dialplan.php** faylında ilk 3 sətir yoxlayır ki, bizim müraciətdə **Hunt-Context** varmı və **Hunt-Context**-i **book_test**-dirmi,

Əks halda artıq bildiyimiz **not found XML** cavabını qaytarıb FreeSWITCH-ə deyəcək ki, kontekst üçün başqa yerdə axtarış elə. Bu skriptin qalanı isə bizim bir genişlənmədən ibarət olan Dialplan konteksti üçün XML-i çap edir.

Web serveri sinaqdan keçirmək üçün CLI-dan istifadə edilən əla alət **curl**-dir. İstənilən UNIX/Linux-da aşağıdakı əmri istifadə edin:

```
root@frfs:~ # curl -D- http://localhost/index.php -d 'section=dialplan&Hunt-Context=book_test'
```

Nəticə aşağıdakı oxşar çap edilməlidir:

```
HTTP/1.1 200 OK
Date: Tue, 27 Oct 2015 06:09:54 GMT
Server: Apache/2.4.16 (FreeBSD) OpenSSL/1.0.1j-freebsd PHP/5.6.13
X-Powered-By: PHP/5.6.13
Content-Length: 278
Content-Type: text/xml; charset=UTF-8
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="freeswitch/xml">
  <section name="dialplan">
    <context name="book_test">
      <extension name="no_name">
        <condition>
          <action application="info">
        </condition>
      </extension>
    </context>
  </section>
</document>
```

Misallarımızdan əksərində siz görəcəksiniz ki, CLI-dan curl əmri istifadə edilir. Biz adətən **-D** opsiyasını istifadə edəcəyik ki, cavabın başlıqlarını ekrana çıxaraq. Bizim misalda **-D** deyir ki, başlıqları fayl əvəzinə ekrana çap elə.

Nəzər yetirin ki, **<condition>** tag-ın başlıqları yoxdur. Səbəb ondan ibarətdir ki, biz routing həllərini öz skriptimizdə real vaxtda edə bilərik və XML-i yalnız qəbul elədiyimiz həll üçün qaytararıq. Ancaq, sizin hələ də seçiminiz var ki, şərtləri **field** və **expression** atributu ilə çap edib, FreeSWITCH-ə onları dəyərləndirməyə və qərar verməyə şərait yaradasınız. Əksər tətbiqlər tək genişlənmə qaytaracaq ancaq, digərləri mövcud **context** qaytaracaq və **mod_xml_curl** sadəcə çoxlu maşınlar üzərindən universal quraşdırma formasını saxlamaq üçün istifadə edilir.

Qeyd: Bu misalda olan misallar curl-in Linux/UNIX CLI-dan istifadəsini göstərir. Həmçinin curl.exe Windows 32 bit və 64 bit versiyaları da mövcuddur. Endirmək üçün <http://curl.haxx.se/download.html#Win32> ünvanına daxil olub baxa bilərsiniz.

mod_xml_curl qovluğu

Bu seksiyada qovluğa olan cavabların necə FreeSWITCH-dən qaytarılmasını göstərəcəyik. Misal istənilən istifadəçiye **1234** şifrəsi ilə qoşulmağa izin verəcək. Bu yalnız FreeSWITCH serverə düzgün XML-in qaytarılmasını göstərmək üçün istifadə edilir və təcrübədə bu imkani təhlükəsizlik baxımından heç vaxt istismara buraxmayıñ.

Sizin **index.php** faylı yerləşən ünvanda **directory.php** adlı fayl yaradın.

Aşağıdakı sətirləri **directory.php** faylinə əlavə edin:

```
<?php
if ( !array_key_exists( 'domain', $_REQUEST ) || !array_key_exists( 'user',
$_REQUEST ) ) {
    not_found( 'missing domain or user' );
}
print "<document type=\"freeswitch/xml\">\n";
print " <section name=\"directory\">\n";
print " <domain name=\"$".$_REQUEST['domain']."\">\n";
print " <groups>\n";
print " <group name=\"default\">\n";
print " <users>\n";
print " <user id=\"$".$_REQUEST['user']."\">\n";
print " <params>\n";
print " <param name=\"password\" value=\"1234\">\n";
print " </params>\n";
print " <variables>\n";
print " <variable name=\"user_context\" value=\"default\">\n";
print " </variables>\n";
print " </user>\n";
print " </users>\n";
print " </group>\n";
print " </groups>\n";
print " </domain>\n";
print " </section>\n";
print "</document>\n";
?>
```

dilaplan müraciətlərində olduğu kimi, **directory** müraciətləri də çoxlu **POST** müraciətləri yollayacaq hansı ki, sizə müraciətə necə cavab verməyinizi təyin edəcək. Əksər istifadə edilən **domain** və **user** parametrləridir. Scriptin ilk **3** sətirində əmin oluruq ki, onların hamısı işə düşməzdən önce mövcuddur. Öncəki misallarımızda olduğu kimi, əgər tələb elədiyimiz parametrlərdən hər hansı biri tapılmazsa bizə **not found XML** qaytaracayıq. Scriptimizin qalan hissəsi isə, çap edilən FreeSWITCH-in emal edəcəyi XML-dir. Qeyd edin ki, **<domain>** və **<user>** tag-lərində biz sadəcə bizdən tələb ediləni geriyə qaytarırıq. Bu istənilən istifadəçi adı və istənilən domain üçün düzgün XML cavabını 1234 şifrəsi ilə qaytaracaq. Bir daha bildiririk ki, bu misalları istismara buraxmayıñ əks halda hücumçu özünə aid olmayan kontekstə düşüb istənilən ünvana zəng edə biləcək. Misalda sadəcə göstərilir ki, directory-ə edilən müraciət-ə düzgün cavab qaytara bilir. Istismarda olan sistemlərdə əksər hallarda elə olur ki, siz istifadəçi məlumatlarınızı verilənlər bazasında saxlayırsınız və sizin script bazaya qayda ilə müraciət yollayıb tələb edilən cavabı alır.

Sistemin CLI-dan aşağıdaki əmri işə salın:

```
root@frfs:~ # curl -D- http://localhost/index.php -d
'section=directory&domain=example.com&user=1000'
```

Nəticə aşağıdakina uyğun olmalıdır:

```
HTTP/1.1 200 OK
Date: Tue, 27 Oct 2015 07:49:30 GMT
Server: Apache/2.4.16 (FreeBSD) OpenSSL/1.0.1j-freebsd PHP/5.6.13
X-Powered-By: PHP/5.6.13
Content-Length: 415
Content-Type: text/xml; charset=UTF-8
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="freeswitch/xml">
<section name="directory">
<domain name="example.com">
<groups>
<group name="default">
<users>
<user id="1000">
<params>
<param name="password" value="1234">
</params>
<variables>
<variable name="user_context" value="default">
</variables>
</user>
</users>
</group>
</groups>
</domain>
</section>
</document>
```

Ötəri baxışla **name** atributu və **<domain>** tag-ına, **<user>** tag-nın **id** atributuna baxın və nəzərinizdə saxlayın ki, əgər siz fərqli parametrləri curl müraciətindən post edərsinizsə onlar necə dəyişəcək. Yadda saxlayın ki, **<domain>** tag-ın daxilində olan bütün qruplar və istifadəçilər eyni domain-ə aid olurlar. Eynilə **<group>** tag-ın daxilində olan bütün istifadəçilər eyni qrupa aid olacaqlaq.

mod_xml_curl quraşdırılması

Siz **mod_xml_curl** istifadə edərək FreeSWITCH-ə çoxlu sayda quraşdırma faylları mənimsədə və quraşdırımları əlaqələndirə bilərsiniz. Bu seksiyada biz dinamik olaraq **sofia.conf** faylini generasiya edəcəyik və platformaya uyğun olan bir mühit yaradacayıq ki, digər quraşdırımları asanlıqla edə bilək. Biz bir neçə yeni fayl və alt qovluqları öz WEB serverimizə əlavə edəcəyik.

index.php faylı yerləşən eyni ünvanda **configuration.php** adlı fayl yaradın və tərkibinə aşağıdakı sətirləri əlavə edin:

```
<?php
if ( array_key_exists( 'key_value', $_REQUEST ) ) {
    $conf = $_REQUEST['key_value'];
    if ( is_file( "configuration/$conf.php" ) ) {
        include_once( "configuration/$conf.php" );
    } else {
        not_found( "unable to find config script ($conf.php)" );
    }
}
?>
```

configuration.php faylı spesifik quraşdırma üçün **configuration** adlı alt qovluğa baxacaq. **configuration.php** və **index.php** faylları yerləşən eyni qovluğun altında **configuration** adında alt qovluğu yaradın.

Sonra **sofia.conf.php** adlı faylı **configuration** adlı alt qovluqda yaradıb, tərkibinə aşağıdakı sətirləri əlavə edin:

```
<?php
print "<document type=\"freeswitch/xml\">\n";
print " <section name=\"configuration\">\n";
print " <configuration name=\"sofia.conf\">\n";
print " <profiles>\n";
print " <profile name=\"internal\">\n";
print " <settings>\n";
print " <param name=\"sip-ip\" value=\"$[_SERVER['REMOTE_ADDR']]\"/>\n";
print " <param name=\"rtp-ip\" value=\"$[_SERVER['REMOTE_ADDR']]\"/>\n";
print " <param name=\"sip-port\" value=\"5060\"/>\n";
print " </settings>\n";
print " </profile>\n";
print " </profiles>\n";
print " </configuration>\n";
print " </section>\n";
print "</document>\n";
?>
```

Orda **sofia.conf.php** faylında FreeSWITCH-ə olan cavabın qaytarılmasından başqa heçnə yoxdur. Biz nəzərdə tuturuq ki, **configuration.php** uğurla bu faylı tapmışdır və yükləmişdir, sonra hissyatlı hissə XML-in çap edilməsidir. Qeyd etməli olduğumuz bir şey var bu **rtp-ip** və **sip-ip**-dir hansı ki, göndərilən müraciətə IP ünvan təyin edir. Beləliklə biz misalımızda **127.0.0.1** istifadə edirik.

Gəlin quraşdirmamızı sinaqdan keçirək. Sistemin CLI-indən aşağıdakı əmri işə salın:

```
root@frfs:~ # curl -D- http://localhost/index.php -d
'section=configuration&key_value=sofia.conf'
```

Çıxış aşağıdakı kimi olacaq:

```
HTTP/1.1 200 OK
Date: Tue, 27 Oct 2015 19:10:42 GMT
Server: Apache/2.4.17 (FreeBSD) PHP/5.6.13
X-Powered-By: PHP/5.6.13
Content-Length: 410
```

Content-Type: text/xml; charset=UTF-8

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="freeswitch/xml">
  <section name="configuration">
    <configuration name="sofia.conf">
      <profiles>
        <profile name="internal">
          <settings>
            <param name="sip-ip" value="127.0.0.1"/>
            <param name="rtp-ip" value="127.0.0.1"/>
            <param name="sip-port" value="5060"/>
          </settings>
        </profile>
      </profiles>
    </configuration>
  </section>
</document>
```

Öncə dediyimiz kimi, **rtp-ip** və **sip-ip** hər ikisi də IP ünvanlardır hansı ki, müraciət-dən qayıdır. Bizim halda 127.0.0.1 istifadə elədik hansı ki, bütün hallar üçün yararlıdır. Təsəvvür edin ki, sizin cluster-də işləyən çoxlu sayda FreeSWITCH serverləriniz mövcuddur hansı ki, əlaqələnəcək IP ünvanları çıxmaq şərtilə bütün profillər eynidir. Birdən real cəmiyyətdə bütövlükə istifadə ediləcək işarələri göstərir.

Görə biləcəyiniz başqa məqamda var hansı ki, müraciətləri analiz elədikdə quraşdırma faylları üçün bəzi müraciətlər **post_load_** prefiksələ başlayırlar. Bu müraciət edilən fayllar size imkan verir ki, fayllarda quraşdırılmaları **mod_xml_curl** yüklenə bilməsindən öncə, təyin edəsiniz (misal üçün, **modules.conf.xml** və **switch.conf.xml**). **post_load_** fayllarında olan tərkib eynilə statik XML fayllarla eynidir. Misal üçün, sizin **mod_xml_curl** yükleyən **modules.conf.xml**-iniz var və sonra qalan tələb edilən modulları **mod_xml_curl** cavabı ilə **post_load_modules.conf** yükleyə bilərsiniz.

mod_xml_curl nəticə

Göstərdiyimiz **mod_xml_curl** misallarının əksəriyyətini siz eyni XML-də hər dəfə çap edəcəksiniz və beləliklə də onlar müəyyən kiçik şəyləri saymasaqlı, həddən artıq statikdirlər.

Siz yəqin ki, diqqət yetirdiniz ki, biz XML faylini çıxışa ötürmək üçün həddən artıq çox print operatorundan istifadə elədik. Digər dillərdə olduğu kimi PHP-də də XML generasiyası üçün library/class-lar mövcuddur hansı ki, size fərqli simvollar kodlaşdırılmalarında düzgün XML generasiyasına kömək edəcək. Misal üçün, PHP-nin SimpleXML adlı genişlənməsini (<http://php.net/manual/en/book.simplexml.php>) istifadə edə bilərsiniz. Ümumiyyətlə hər hansısa bir ssenarinin planlaşdırmasını edirsinizsə, kitabxanalardan istifadə eləməniz daha düzgündür.

Dillərin əlaqələndirilməsi vasitəsilə quraşdırmacların dinamik generasiyası

Əgər siz ssenarilərin emalı üçün WEB server qaldırmaq istəmirsinizsə, FreeSWITCH sizə daxili script dillərinin sayəsində eyni qosulma müraciətlərinin emalına izin verir. Bütün script dili modulları sizə izin verir ki, təyin elədiyiniz parametrləri eynilə **mod_xml_curl** elədiyi kimi öz skriptiniz emal eləsin. Ümumi istifadə edilən dillər aşağıdakılardır:

- **mod_lua** ilə Lua
- **mod_perl** ilə Perl
- **mod_python** ilə Python

Qeyd: FreeSWITCH həmçinin Microsoft-un **.NET** dilinidə dəstəkləyir hansı ki **mod_managed** tərəfindən idarə edilir. Bunun istifadəsi script dilləri olan Lua, Perl və Python-dan fərqlidir. Ətraflı məlumatı http://wiki.freeswitch.org/wiki/Mod_managed linkindən əldə edə bilərsiniz.

/usr/local/freeswitch/conf/autoload_configs qovluğuna baxsanız siz hər bir dil üçün quraşdırma fayllarını görə bilərsiniz:

- `lua.conf.xml`
- `perl.conf.xml`
- `python.conf.xml`

Onlardan istənilənini açın və siz aşağıdakılara oxşar parametrləri görəcəksiniz:

```
<param name="xml-handler-bindings" value="dialplan|directory|configuration"/>
<param name="xml-handler-script" value="/path/to/script.ext"/>
```

Gördüyüümüz kimi, **xml-handler-bindings** quraşdırma seksiyası sizin skriptinizin tərəfindən emal ediləcək hissədir. Həmçinin scriptdə müraciətlərin emal edilməsi üçün **xml-handler-script** ünvani var.

Qeyd: Biz öz misallarımızda **mod_lua** istifadə edəcəyik amma, prinsip eynilə **mod_perl** və **mod_mython**-a da aiddir.

mod_xml_curl və **mod_lua** arasında olan əsas fərq ondan ibarətdir ki, **mod_lua** FreeSWITCH-in daxilindədir və orda FreeSWITCH-in oxuması üçün XML çıxışda **Content-Type** təyin edib çap eleməyə ehtiyac yoxdur. Onun yerinə spesifik **XML_STRING** adlı spesifik dəyişən təyin edirik hansı ki, tərkibində FreeSWITCH tərəfindən parçalanacaq XML tərkib olur. Siz bunu aşağıdakı misalda görə bilərsiniz:

```
local xml_header = [[<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="freeswitch/xml">]]
local xml_body
if XML_REQUEST['section'] == 'configuration' and XML_REQUEST['key_value'] == 'sofia.conf' then
    local ip_v4 = params:getHeader( 'FreeSWITCH-IPv4' )
    xml_body = string.format([[<section name="configuration">
        <configuration name="sofia.conf">
```

```

<profiles>
  <profile name="internal">
    <settings>
      <param name="sip-ip" value="%s"/>
      <param name="rtp-ip" value="%s"/>
      <param name="sip-port" value="5060"/>
    </settings>
  </profile>
</profiles>
</configuration>
</section>]], ip_v4, ip_v4)
else
xml_body = [[
  <section name="result">
    <result status="not found"/>
  </section>
]]
end
local xml_footer = [[</document>]]
XML_STRING = xml_header .. xml_body .. xml_footer
  
```

Bu misal artıq sizə çox tanış gəlməlidir. Başlığın əvvəlində **mod_xml_curl** vasitəsilə elədiyimiz eyni işi görür. Biz eyni olaraq XML açan və bağlayan sekisiyaları generasiya elədik. Sonra əgər, sekсиya **configruation** və **key_value** **sofia.conf**-dursa biz minimal **sofia.conf** generasiya elədik. Əgər orda istənilən digər sekсиya ya da istənilən digər quraşdırma üçün müraciət edilibsə, onda biz sadəcə başlığın əvvəlində **mod_xml_curl** misallarında danışdığımız eyni **not found XML** blokunu qaytaracayıq.

İlk baxışdan bu biraz qarışiq görünə bilər hansı ki, bu ssenariləri clusterdə olan maşınlarda sinxronizasiya edərək saxmalaq istəyir. Ancaq əgər siz faylların saxlanılması üçün hansısa Box, Dropbox ya da istənilən digər imkanlardan birini istifadə etməyi nəzərdə tutursunuzsa, onda siz çox gözəl alternativi **mod_xml_curl**-i istifadə edə bilərsiniz hansı ki, web serverin istifadə eləməsini tələb eləmir.

Zənglərin API-la CLI-dan edilməsi

Siz zəngləri sistem istifadəçisi olmadan da edə bilərsiniz. Bunun üçün nəzərdə tutulur ki, sizin qeydiyyatdan keçmiş son nöqtəniz var və siz ona qeydiyyatsız zəng edə bilərsiniz. Son nöqtə IP telefon, program telefonu ya da bir və ya iki istifadəçi ilə qeydiyyatdan keçmiş FreeSWITCH server ola

bilər. Yalnız UI tələb edilir ki, hansı ki sizin zənginiz cavab veriləcək telefondur. Bizim misalda **my.open.endpoint.example.com** mənsəb domainindir. Əmin olun ki, öz quraşdırılmalarınıza uyğun olan domain ya da IP ünvan istifadə edirsiniz.

fs_cli-i açın və aşağıdakı əmri yerinə yetirin:

```
originate sofia/internal/1001@frfs.opensource.az &echo()
```

Sözsüz ki, real həyatda bu elədə yararlı alət deyil. Eşidərək "Salam bir, iki, üç" deyilişi ən yaxşı sınaqdır amma, bunu daha produktiv misal eləmək üçün biz daha da yaxşı zəng edə bilərik.

Aşağıdakı misalda biz, son nöqtəmizə zəng edəcəyik və sonra onu public FreeSWITCH-ə bridge edəcəyik ki, susmaya görə olan Dialplan-da konfransa düşsün hansı ki, FreeSWITCH-də yüklənmədə susmaya görə olur.

```
originate sofia/internal/1001@frfs.opensource.az &transfer(9888 XML default)
```

Öncəki misalda olduğu kimi, buda ilk olaraq bizim son nöqtəyə zəng edəcək. Artıq qoşulma olan kimi, biz default XML dialplan context-inə düşəcəyik hansı ki, **default** kontekstdə olan **9888** nömrəsi **freeswitch_public_conf_via_sip** genişlənməsilə tutuşdurulacaq və zəng bridge ediləcək.

Növbəti misalda həmçinin bizi SIP üzərindən **public** FreeSWITCH konfransına bridge edəcək amma, Dialplan üçün tələb olmadan biz originate əmri ilə bridge edirik:

```
originate sofia/internal/1001@frfs.opensource.az &bridge(sofia/internal/888@conference.freeswitch.org)\
```

Öncəki iki misala baxsanız yenə də düşünə bilərsiniz ki, bunlar da elə də yararlı deyil. Ancaq bu misalları biraz dəyişdirə bilərik ki, web səhifə üçün script işləsin. Siz öz tərəfinizdən sayt yarada bilərsiniz hansı ki, istifadəçilər öz telefon nömrələrini daxil edirlər və forma emal edilməsi üçün doldurulub göndərilir əsas IVR menyusuna.

```
originate sofia/gateway/my_provider/12125551234 &ivr(main_menu)
```

Bu **12125551234** nömrəsinə zəngi **my_provider** adında gateway üzərindən edəcək və **main_menu** adlanan IVR-da cavab verən şəxsi ayıracaq. Siz ola bilər ki, narahatsınız ki, necə həmin nömrəni web formadan **fs_cli**-in daxilinə almaq lazımdır.

```
fs_cli -x 'originate sofia/gateway/my_provider/12125551234 &ivr(main_menu)'
```

Mütləq **fs_cli**-i **-x(execute)** parametri ilə istifadə edin. **fs_cli**-in **-x** əmri istifadəsi çox asandır. Ancaq sizin ssenarinizdə asılı olaraq ola bilər ki, bu ən genişlənən və effektiv metod deyil. Aşağıdakı seksiya size FreeSWITCH-də olan ən vacib aspektlərdən birini açıqlayır, event socket və ESL.

Əmrlərin yerinə yetirilməsi üçün ESL-in istifadə edilməsi

Bizim öncəki misallarımızda zəngi **fs_cli**-dan yerinə yetirirdik. Bu seksiyada xeyli kod nüsxələrinə baxacayıq hansı ki, sizin sevilən dildən eyni işi görəcək. Bizim misallar üçün Lua istifadə edirik ancaq siz yenə də istədiyinizi seçə bilərsiniz. Bizim misallarımız üçünLua istifadə edəcəyik yəqin ki, siz artıq ona öyrəşmişiniz artıq. API ESL sizin Python, Lua, Perl, PHP ya da hansısa istifadə elədiyiniz dil kimidir. Ona görə də ağıllı şəxslər bu imkanları istənilən dildə istifadə edə bilər.

Qeyd: **ESL scriptləri ya da daxili dillər.** Yadda saxlayın ki, ESL bazalı programlar daxili istifadə edilən dillərlə eyni deyil. FreeSWITCH event socket FreeSWITCH-in özünə qoşulmaq üçün TCP bazalı qoşulmadır. ESL ümumi kitabxanalara sahibdir hansı ki, təkcə FreeSWITCH-ə aid olan dillərə daxil deyil. Öncə siz sadəcə ESL-dən öncə Lua, Perl, Python ya da PHP-ni öz sisteminiz üçün yükləyin. Lua üçün <http://www.lua.org> saytına baxa bilərsiniz.

Necə ki, ESL modulları da susmaya görə yüklenmənin hissəsi deyil, onu işlərimizi başlamazdan öncə yükleməliyik. Linux/UNIX istifadəçiləri aşağıdakı addımları etməlidirlər:

1. FreeSWITCH ESL qovluğuna keçid alın:

```
root@frfs:~ # cd /usr/src/freeswitch/libs/esl/
```

2. Aşağıdakı əmri işə salırıq:

```
root@frfs:/usr/src/freeswitch/libs/esl # make luamod
```

Bu əmrlərin uğurla yerinə yetirilməsindən sonra, artıq sizin kompilyasiya edilmiş Lua ESL modulunuz **/usr/src/freeswitch/libs/esl/lua** qovluğunda **ESL.so** adında mövcud olmalıdır.

Qeyd: Əgər sizin Lua ESL modulu yüklenmənizdə problemlər çıxırsa, onda öz sisteminizə uyğun olan **lua-devel** paketini yükləyin.

Siz **ESL.so** faylini Lua modulunun axtardığı ünvanlardan birinə köçürməlisiniz ki, hansı qovluqda yerləşmənidən asılı olmayaraq yüklenilə bilən olsun. Ünvanın tapılmasının qısa yolu mütləq və mütləq **/usr/src/freeswitch/libs/esl/** qovluğunda yerləşən **./lua/single_command.lua** scriptinin işə salınmasıdır amma, scripti işə salmazdan öncə **which lua** əmri ilə onun binar fayl ünvanını tapıb, **single_command.lua** faylinin ilk sətirində(**ShaBang**) düzgün dəyişməlisiniz. Nəticədə aşağıdakı sətirləri görməlisiniz:

```
[root@Xmlcurl esl]# ./lua/single_command.lua
/usr/bin/lua: ./lua/single_command.lua:2: module 'ESL' not found:
no field package.preload['ESL']
no file './ESL.lua'
no file '/usr/share/lua/5.1/ESL.lua'
no file '/usr/share/lua/5.1/ESL/init.lua'
no file '/usr/lib64/lua/5.1/ESL.lua'
no file '/usr/lib64/lua/5.1/ESL/init.lua'
no file './ESL.so'
no file '/usr/lib64/lua/5.1/ESL.so'
no file '/usr/lib64/lua/5.1/loadall.so'
stack traceback:
[C]: in function 'require'
```

```
./lua/single_command.lua:2: in main chunk
[C]: ?
```

Gördüyüümüz kimi sistem qovluğun(Yəni . qovluğun) içində **ESL.so** və çoxlu daxili kitabxanaların ünvanlarını axtarır. Kitabxananı **/usr/lib64/lua/5.1/ESL.lua** ünvanına nüsxələyirik. Bu misalda bizim serverimiz x64 olduğuna görə də, kitabxananı həmin ünvana nüsxələyirik:
cp /usr/src/freeswitch/libs/esl/lua/ESL.so /usr/lib64/lua/5.1/ESL.lua

Yenidən əmri işə salıb yoxlayırıq ki, **ESL.so**-nun düzgün ünvanda olmasını təyin edək.

Qeyd: Windows istifadəçiləri mod_esl-i Microsoft Visual Studio ilə kompilyasiya etməlidirlər. mod_esl üzərində sağ düyməni sıxın və **Build** düyməsinə sıxın. O **esl.dll** yaradacaq. Bu faylı lua yüklenən **clib** alt qovluğun içini yükləyin. Lua scriptini Windows-da işə saldıqda, CLI-i istifadə edin və sonra **lua**-ni scriptin adının ardınca yerinə yetirin. Misal üçün: **lua single_command.lua**

Əgər hər şey düşündüyüümüz kimi gedirse, siz aşağıdakina uyğun olan sətirləri görəcəksiniz:

```
/usr/bin/lua: Error in api (arg 2), expected 'char const *' got 'nil'
stack traceback:
[C]: in function 'api'
./lua/single_command.lua:9: in main chunk
[C]: ?
```

Əgər belə bir səhv çap edilərsə, o deməkdir ki, ötürülən əmr bizim **single_command.lua** scriptimiz üçün düzgün deyil. Bunu aşağıdakı əmrlə edə bilərsiniz:

```
./lua/single_command.lua status
```

API-ın **status** əmrini öz **single_command.lua** scriptimizə ötürdükdə, lua scripti biziə aşağıdakı çıxışı verəcək:

```
UP 0 years, 0 days, 5 hours, 6 minutes, 27 seconds, 223 milliseconds, 190
microseconds
FreeSWITCH (Version 1.2.8) is ready
0 session(s) since startup
0 session(s) - 0 out of max 30 per sec
1000 session(s) max
min idle cpu 0.00/96.00
Current Stack Size/Max 240K/240K
```

Əgər əmin nəticəsində gözlədiyiniz məlumatı görünüzsə, bu o deməkdir ki, API-a **single_command.lua** scripti sayəsində onun dəstəklədiyi istənilən əmri ötürmək olar. Beləliklə biz bu scriptə önce istifadə elədiyimiz əmrləri ötürə bilərik:

```
./lua/single_command.lua 'originate sofia/gateway/my_provider/12125551234
&ivr(main_menu)'
```

Bunun çox yararlı olması o halda olacaq ki, siz FreeSWITCH serveriniziə tələb edilən əmrləri tamam başqa bir maşından ötürmək istəyəcəksiniz. **mod_event_socket** quraşdırılması sizə imkan yaradır ki, kənardan FreeSWITCH serverinizi idarə edə biləsiniz hansı ki, növbəti başlıqda **FreeSWITCH-in kənardan idarə** edilməsində daha da ətraflı danışılır.

Notice

Bu başlıqda olan quraşdirmalarla siz minimal XML fayllarla bir WEB server üzərindən bütün FreeSWITCH klaster maşınlarına paylaşım edə bilərsiniz. Heç bir digər **Directory** və ya **Dialplan**-a ehtiyac yoxdur ona görə ki, onlarda WEB server tərəfindən ötürürlə bilər. ESL üzərindən sizi əlavə edən prosesin bridge edilməsini xoşunuza gələn dillə yerinə yetirin, voicemail-ləri idarə edin və.s. Artıq siz bilirsiniz ki, klaster-də olan çoxlu sayda FreeSWITCH serverləri idarə eləmək üçün fərqli konsollar-da olan SHELL-ə ehtiyac yoxdur.

Bu FreeSWITCH-in XML faylları olmadan idarə edilməsinin sadə yoludur. Növbəti başlıqda biz çox güclü olan bir modul **FreeSWITCH event socket** haqqında danışacayıq.

10-cu başlıq

FreeSWITCH-in kənardan idarə edilməsi

FreeSWITCH-in event system-i ən çox güclü olan komponentlərindən biridir. Siz həmçinin FreeSWITCH-in fərqli quraşdırma faylları və script dillərində əlaqədə işlədiyi hallarda özünүn necə aparmasını öyrənmisiz. Event sistemi sizə çox geniş dinamik funksionallıq verir ki, FreeSWITCH-i idarə edə biləsiniz. Event sistemin istifadə edilməsi ilə, həqiqətən də FreeSWITCH dirilməyə başlayır.

Event sistemi kənar programlara imkan yaratdır ki, sistemdə baş verən işlər üçün qulaq asan kimi iştirak edə bilsinlər. Bu, telefon əməliyyatları ilə real vaxtda telefon softswitch-lə kənar işləyən program və ya avadanlıqla əlaqə yaratmağa şərait yaratdır. Demək olar ki, FreeSWITCH çərcivəsində işləyən bütün hadisələr qısa şəkildə **event message** generasiya edir. Bu eventlərə kənar obyektlərlə baxıla bilər. Bu publish/subscribe (ya da "**pub-sub**") sistemlərinin quruluşlarında istifadə edilən mesaj növbələşməsi program həlləri kimidir və eynilə də FreeSWITCH-də istifadə edilir.

Event sistemi iki istiqamətlidir: Əlavə olaraq kənar programlara hadisələri (**event**) dinləməklə eynilə də hadisələri(event) FreeSWITCH-ə ötürmək üçündə şərait yaratdır. Bu birləşmə sizə təsəvvürünüzdə olan istənilən yerdə FreeSWITCH-in istifadə edilməsinə şərait yaratır.

Başlıqda biz aşağıdakıları müzakirə edəcəyik:

- Event sisteminə ümumi baxış
- Event sistem arxitekturası
- Event socket quraşdırılmaların edilməsi
- Event socket kitabxanası
- Təcrübədə olan eventlər

Event sisteminə ümumi baxış

Event sistemi FreeSWITCH-in mərkəzi əsəb quruluşudur hansı ki, daxili və xarici programlara imkan verir ki, sistemin daxilində baş verən aktivliyin axınına üzv ola bilsinlər. FreeSWITCH-də baş verən hər şey event generasiya(ya da yandırır) edir. Yeni zəngin gəlməsi hansısa event-in nəticəsidir. Zəngin dayandırılması nəticəsi event-dir. Nəticələrin diskimizə jurnallanması bir eventdir. Hətta sakit danişiq belə event generasiya edə bilər. Hər bir hadisə **event stream**-in hissəsi olaraq gəlir hansı ki, **event type**, **event category** ilə və event haqqında çoxlu digər detallarla **tag** edilir. Programın digər hissələri bu eventlər üçün qulaq asa bilər və onlarla istəklərindən asılı olaraq əlaqəyə gire bilərlər. Misal olaraq onları TCP socket qoşulması üzərindən açıq mətnlərə axına ötürmək.

FreeSWITCH funksionallığının böyüdülməsi üçün events digər bir üsuldur. Events lər keçirici ya da modullardan fərqlidir(hansı ki, real vaxtda emal və zəngin işə salınmasını edə bilir). Events sistem üzərində olan aktivliyin izlənilməsi üçün asinxron(ya da bloklanmayan növbələr) metod təqdim edir. Onlar programın bir hissəsi ilə bir yerdə generasiya edilir və sonra programın digər hissəsi tərəfindən istifadə edilir. Təcrübədə bunu external programdan istifadə eləmək əvəzinə, çox aktivlik olan yerlərdə istifadə edə bilərsiniz.

Misal olaraq, birdən zəng səsinin artırılmasında qəribə səs ola bilər ki, yeni zəng eventlərini generasiya edir. Siz həmçinin cəhd edə bilərsiniz ki, bu eventləri web browser tərəfindən idarə edəsiniz ancaq, web browser həmin anda da baş verən yeni zəng rəqəmi ilə saxlaya bilməyəcək. Növbələnmiş event sistemini istifadə eləməklə, siz core keçirtmə motorunun(və həmçinin asılı olmadan zəngləri bloklamaqla) emal edilməsindən qazmış olursunuz hansı ki, bu vaxt web browserin baş verən səsin emal etməsini gözləyirsınız.

Başlıqda sistem eventlərinin bütün fərqli hissələrinə kənar programdan FreeSWITCH-ə eventlərin göndərilməsi, qəbul edilməsinə baxacayıq. Biz modullara baxacayıq hansı ki, event sistemi kənardan işləmək üçün işə salır. Event tiplərini generasiya eləmək və eventi işə salacaq üsulları öyrənəcəyik. Sonda bir ssenariyə baxacayıq ki, özünüzə aid olan bir code-la necə FreeSWITCH-i idarə edə bilərsiniz.

Event sistemi arxitekturası

FreeSWITCH üzərində olan event alt sistemi dizayn edilmişdir ki, keçiricilik qabiliyyəti və tiplərindən asılı olaraq sistemin yüklənməsində olan eventlərə prioritetlər təyin eləmək olsun. FreeSWITCH-də olan event sisteminin iki səviyyəsi var. İlk qat daxili event emalı rutini və FreeSWITCH-in özü ilə bu eventləri qəbul eləmək üçün interfeys təqdim edir. İkinci qat modullu arxitektur tərəfindən təqdim edilir və bu eventlərə müştəri görünüşlü yetki verir. Bu iki funksionallığı ayırdıqda, event sistemin publish/subscribe strukturu aydın olur.

Daxili event səviyyəsi ilə FreeSWITCH mərkəzi funksionallıq verir ki, **channel-level** və **system-level**-də baş verən eventləri emal eləsin. Event-lər sistemin istənilən hissəsi ilə yayılana bilər(modullar daxil olmaqla). Eventlərin ümumilikdə iki özəl tipi mövcuddur - sistem eventləri və jurnallama eventləri. Sistem eventləri core alt sistemi komponentləri ya da modullar vasitəsilə generasiya edilir. Onlar sistemin daxili ürək döyüntüsü vaxtıda daxil olmaqla hər şeyi konfrans alt sisteminə əlavə edirlər(Misal üçün konfrans otağına üzv olmaq ya da çıxış eləmək). Jurnallama eventləri hər dəfə FreeSWITCH jurnal faylinə jurnal verilənini generasiya edir. Bu alt sistemləri əslində 3 event hissəsindən ibarət olur. Hər birinin özünə aid olan axını və üstünlük səviyyəsi olur. Əgər növbələşmə dolursa, bütün event sistemi dolmayanadək digər event növbələşməsinə kecid olacaq. Zənglər ya da sistem funksiyalarının irəliləyişi kimi, hadisələr də daxili üzv olanlardan yüklənmə gözlədikləri halda da, backend-də olan axınlarla yerinə yetirilir və yaddaşa saxlanılır. Mesaj bütün modullar və alt sistemləri tərəfindən götürülən kimi, event mesajı silinir. Bu event sisteminə eventlərin özləri kimi yaxşı genişlənmə imkanı verir ona görə ki, atılmış eventlər növbələnməni düzəzmək üçün, onların yarananlarının gözlədikləri zamanda zəngi blok etmir.

FreeSWITCH bu modullu arxitekturu istifadə edir ki, kənar programdan eventləri etmək imkanı olsun. Event emal edən modulu daxili event mesajlarına üzv ola, onları formatlaya və onları kənar programa yollaya bilər. Belə modullara uyğun olaraq **event handlers** deyilir. FreeSWITCH-ə çoxlu sayıda event emalediciləri daxil edilməyib amma mövcud olanlarda həddən artıq funksionallıq sahibdir ona görə ki, core sistemin özü eventlərlə boldur. Biz bu modullara və onların istifadə edilməsinə baxacayıq.

Event bazalı modullar

Eventləri emal edəcək çoxlu modullar var. Ancaq ən çox istifadə ediləni **mod_event_socket**-dir. Biz əsas diqqətimizi bu modula yönəldəcəyik və sonra xırda olaraq digərlərinə də baxacayıq.

mod_event_socket

mod_event_socket modulu FreeSWITCH-də 3-cü tərəf programlardan eventlərin qəbul edilməsi və göndərilməsi üçün əsas ümumi olanıdır. Bu modul TCP socket verir hansı ki, kənar programlardan ona qoşula bilərsiniz. Qeydiyyatdan keçdiğindən sonra, siz asanlıqla plan-text formatında event məlumatlarını göndərə və qəbul edə bilərsiniz. Bunu parçalamaq və anlamaq çox asandır. Bu FreeSWITCH-də qəbul edən və göndərən eventlər üçün ikitərəfli qoşulmaya şərait yaradır.

Event socketlərin istifadə edilməsi ümumilikdə çox asandır. İlk olaraq, öncədən `mod_event_socket` üçün quraşdırılmış socket-ə kənar programdan qoşulursunuz. Siz sistemdə qeydiyyatdan keçirsiniz və sonra FreeSWITCH-ə event mesajlarının yollanmasına başlayırsınız. Siz həmçinin eventlərin qəbul edilməsi üçün müraciət-də yollaya bilərsiniz hansında ki, `mod_event_socket`-i event listenerləri event sisteminə əlavə edəcək, event mesajlarını növbələyəcək və onları sizə emal edə biləcəyiniz sürətdə də geriyə qaytaracaq.

`mod_event_socket` modulu sizə müraciətləri plain tekst-də göndərmək və qəbul eləmək üçün, eventlərin serialaşdırılmış nüsxələnməsi və özünüzə aid olacaq eventlərin generasiya edilməsi üçün interfeys təqdim edir. Siz istəyinizdən asılı olaraq həmçinin XML kimi formatlanmış datanın qayıtməsi üçün müraciət yollaya bilərsiniz. Modul event süzgəcləmə mexanizmini də təqdim edir ki, yalnız sizə maraqlı olan event tipinə üzv ola biləsiniz. Misal üçün sizin program dizayn edilə bilər ki, yalnız konfranslarla işləsin. Bu halda sizdə tələb olacaq ki, yalnız konfransa aid olan eventlər gəlsin. Modulun özü daxili event sistemindən gələn eventləri ələ keçirib bütün aktiv olan TCP socket qoşulmalarına həmin eventləri ötürə bilmək imkanına sahibdir. O hər bir TCP qoşulması üçün fərqli event səviyyəsi təyin edərək, ayrıca individual iş görərək növbə yaradır. Növbələşmə işinin özüdə elə FreeSWITCH-in core funksionallığının bir hissəsidir.

Event socket quraşdırımaların edilməsi

Siz event socket-i sadəcə `mod_event_socket`-i `/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml` faylında şərhi silib yükleməklə işə sala bilərsiniz. `mod_event_socket` yükləndikdən sonra, `/usr/local/freeswitch/conf/autoload_configs/event_socket.conf.xml` faylında dəyişiklik edə bilərsiniz. Aşağıdakı parametrlər mövcuddur:

- **listen-ip:** Event socket qoşulmaları üçün qulaq asacaq IP ünvanı. Kənar programlar bu IP ünvana qoşulacaqlar. Susmaya görə olan siyaset yalnız **127.0.0.1** ünvanından qoşulmalara izin verir. Siz spesifik IP ünvan təyin edə ya da **0.0.0.0** IP ünvan təyin edə bilərsiniz ki, bütün IP ünvanlarda qulaq asasınız.

```
<param name="listen-ip" value="127.0.0.1"/>
```

- **listen-port:** Gələn qoşulmalar üçün qulaq asdığı TCP port.

```
<param name="listen-port" value="8021"/>
```

- **password:** Porta qoşulduqda istifadə ediləcək şifrə.

```
<param name="password" value="ClueCon"/>
```

- **apply-inbound-acl:** Access Control List(ACL) porta gələn qoşulmaları idarə eləmək üçün istifadə edilir. Faktiki olaraq siz bununla önce danışdığınız IP:Port qoşulmasına gələcək qoşulmaları bir növ süzgəcdən keçirmiş olacaqsınız. Siz bununla həmçinin bildiyimiz access control list(hansı ki, öncədən /usr/local/freeswitch/conf/autoload_configs/acl.conf.xml faylında təyin etmisiniz) adı da istifadə edə bilərsiniz ya da IP ünvan/aralıq təyin edə bilərsiniz.

```
<param name="apply-inbound-acl" value="<acl_list|cidr>"/>
```

Aşağıda **apply-inbound-acl** üçün misal göstərilir:

```
<param name="apply-inbound-acl" value="known_machines"/>
```

```
<param name="apply-inbound-acl" value="10.20.0.0/16"/>
```

Nəzərə alın ki, çoxlu **apply-inbound-acl** parametrləri işləməyəcək.

Eventlərin oxunması

Eventləri mod_event_socket-dən oxuduqda, data iki nöqtə ilə ayrılan **name/value** cütlüyü formatında olacaq. Event mesajı iki sətir sonu ardıcılılığı ilə bitir. FreeSWITCH **carriage-return linefeed** (CRLF) simvollarında ən-ənəvi DOS/Windows EOL ardıcılılığı istifadə edir. Sizin kənar program event socket-ə qoşulmalıdır və iki sətir keçidi görənədən bacardığı qədər çox simvol oxumalıdır. Aşağıda adı tək **key/value** cütlüyü sətiri göstərilir:

Event-Name: CHANNEL_EVENT

Bəzi key/value cütlükleri mənanın özü ilə birlikdə, çoxlu sətirin kəsilməsinə sahib olurlar. Bu ssenaridə FreeSWITCH hələ də mənani bir sətirdə göndərmək istəyir. Bunu eləmək üçün, FreeSWITCH bir sətirdə data-nı URL kodlaşdırma edəcək. Aşağıda çoxlu sətiri olan cavabın qayıdışı göstərilir:

```
variable_switch_r_sdp: v=3D0%0D%0Ao%3DUAC%206407%206867%20IN%20IP4%20
192.168.27.72%0D%0As%3DSIP%20Media%20Capabilities%0D%0Ac%3DIN%20
IP4%2061.231.8.102%0D%0At%3D0%200%0D%0Am%3Daudio%2012916%20RTP/AVP%20
0%2018%20101%0D%0Aa%3Drtpmap%3A0%20PCMU/8000%0D%0Aa%3Drtpmap%3A18%20
G729/8000%0D%0Aa%3Dfntp%3A18%20annexb%3Dno%0D%0Aa%3Drtpmap%3A101%20
telephone-event/8000%0D%0Aa%3Dfntp%3A101%200-15%0D%0Aa%3Dmaxptime%3A2
0%0D%0A
```

Öncə göstərilən nümunədə FreeSWITCH emal elədiyi zəngin SDP tərkibinin kodlaşdırılmış URL-idir. Bu originalda aşağıdakı kimidir:

```
variable_switch_r_sdp: v=0
o=UAC 6407 6867 IN IP4 192.168.27.72
s=SIP Media Capabilities
c=IN IP4 61.231.8.102
```

```
t=0 0
m=audio 12916 RTP/AVP 0 18 101
a=rtpmap:0 PCMU/8000
a=rtpmap:18 G729/8000
a=fmtpt:18 annexb=no
a=rtpmap:101 telephone-event/8000
a=fmtpt:101 0-15
a=maxptime:20
```

Əgər ad/məna cütlüyündən biri **Content-Length** header olarsa, siz mütləq dəqiqliklə başlıqların yüklənməsindən sonra və iki CRLF-lər görünən kimi, socket-dən çoxlu baytları oxumalısınız. Content length-də olan bütün baytları oxuduqdan sonra, növbəti paket ardıcıl baytda işə düşəcək. Sizdə Content-Header-i özündə saxlaya biləcək event olarsa, bu event tərəfindən generasiya edilmişdir əlavə indikatordur hansı ki, **key/value** formasında olmur və özünə uyğun olan format təşkil edə bilər.

Aşağıdakı misal kanal statusunun dəyişilməsi haqqında event-dir:

```
Content-Length: 646
Content-Type: text/event-plain
Channel-State: CS_EXECUTE
Channel-State-Number: 4
Channel-Name: sofia/default/1006%4010.0.1.250%3A5060
Unique-ID: 74775b0d-b112-46e2-95af-c28258650b1b
Call-Direction: inbound
Answer-State: ringing
Event-Name: CHANNEL_STATE
Core-UUID: 2130a7d1-c1f7-44cd-8fae-8ed5946f3cec
FreeSWITCH-Hostname: localhost.localdomain
FreeSWITCH-IPv4: 10.0.1.250
FreeSWITCH-IPv6: 127.0.0.1
Event-Date-Local: 2012-12-16%2022%3A33%3A18
Event-Date-GMT: Mon,%2017%20Dec%202012%2004%3A33%3A18%20GMT
Event-Date-timestamp: 1197865998931097
Event-Calling-File: switch_channel.c
Event-Calling-Function: switch_channel_perform_set_running_state
Event-Calling-Line-Number: 620
```

Nəzər yetirin ki, tünd qara rəngdə olan sətir göstərir ki, **Event-Name**-i **CHANNEL_STATE**-dir. Bu event kanalın statusunun dəyişilməsinə aiddir.

Minimum event məlumatları

FreeSWITCH-də mod_event_socket üzərindən qayidian hər bir event-in tərkibində hadisənin tipindən asılı olmayıaraq, event-i təyin edən minimal sayda məlumat olur. Bu sütunlar sizə təkcə gözlədiyiniz event-in nə etməsini sizə başa salmaq üçün deyil həmçinin, eventin nə etdiyini və hansı serverdə etdiyini də başa salır. Multi server mühitlərində bu sütunlar adətən Core-UUID headerlə istifadə edilə bilər ki, hansı sistemin eventi generasiya elədiyini təyin edəsiniz o halda ki, vaxt möhürləri eventlərin uyğun qaydada bərpa edilməsinə və emalına zəmanət verir.

Hər bir event-də əldə etdiyiniz sütunlar aşağıdakı event-də göstərilir:

```
Event-Name: CHANNEL_EVENT
Core-UUID: 689fd828-e85b-ca43-a219-39332bc55860
Event-Date-Local: 2012-05-09%2018%3A48%3A59
Event-Date-GMT: Wed,%2009%20May%202012%2016%3A48%3A59%20GMT
Event-Calling-File: switch_channel.c
Event-Calling-Function: switch_channel_set_caller_profile
Event-Calling-Line-Number: 840
```

Öncəki misal hansı eventin baş verməsindən asılı olmayaraq həmişə əlavə edilir. Bu o deməkdir ki, hər bir hadisə aşağıdakı tag-lə qeydə alınacaq:

- **Event-Name:** Baş verən hadisənin açıqlamasını verən event adı.
- **Core-UUID:** FreeSWITCH core-un mövcud instansiyasının UUID-si
- **Event-Date-Local:** Sistem saatına uyğun olan event date/time-ı
- **Event-Date-GMT:** UTC vaxtına əsaslanan event date/time-ı
- **Event-Calling-File:** Eventi işə salan C mənbə faylı
- **Event-Calling-Function:** Eventi işə salan funksiyanın adı
- **Event-Calling-Line-Number:** Eventi işə salan C mənbə faylinin dəqiq sətir rəqəmi

Son 3 header məhz problemlərin tapılması və təyin edilməsi üçün istifadə edilir. Əvvəlki informasiyadan sonra hadisə üçün spesifik informasiya göndərilən hadisənin tipindən asılı olaraq daxil ediləcək. Orda event key/value və event-spesifik key/value cütlüyünün arasında sətir ayırıcısı ya da boşluq yoxdur.

Eventlərin göndərilməsi

Siz eventi FreeSWITCH-i core-a **mod_event_socket** üzərindən alığınız eventin eyni TCP qoşulması ilə göndərə bilərsiniz(qoşulma ikitərəflidir). Bütün əmrlər əmrin adı və arqumentləri formatında olur. Bəzi əmrlər əlavə sütunlar tələb edir. Eventi yolladıqda, əlavə sütunlar üçün format eynilə eventlərin qəbul edilməsi formatı ilə eynidir. Siz FreeSWITCH-ə eventin adını təyin edən key/value siyahısı, flagları göndərirsiniz və FreeSWITCH bu mesajı emal məqsədilə event alt sisteminə ya da FreeSWITCH core-a ötürəcək.

Başlanğıc əmr misali aşağıdakı kimidir:

```
api sleep 5000
```

Bu əmr API vasitəsilə **sleep** əmrini işə salacaq və **5000** arqumentini **sleep** əmrinə ötürəcək. Bu o deməkdir ki, sistem **5** saniyəlik yatacaq.

Daha çətin misal mesajın FreeSWITCH event növbələşmə sisteminə birbaşa ötürülməsidir. Siz eventləri FreeSWITCH sisteminə **sendevent** əmri ile uyğun parametrlərlə yollaya bilərsiniz.

sendevent misali aşağıdakı kimidir:

```
sendevent NOTIFY
profile: internal
content-type: application/simple-message-summary
event-string: check-sync
```

```
user: 1005
host: 192.168.10.4
content-length: 5
hello
```

Bu **NOTIFY** eventini uyğun məlumatla yollayacaq. Bu misalda biz deyirik ki, **NOTIFY** mesajı Sofia **internal** profile-da olan 1005@192.168.10.4 istifadəçisinə göndərilsin. Əgər mod_sofia bu tip mesajlar üçün yüklənmişdirsə və qulaq asırsa o müraciət edilən **NOTIFY** mesajı üçün uyğun olan SIP paketi **1005** istifadəçisi üçün generasiya edəcək və **content-type**, **event-string** header-i kontentin özü ilə birlikdə(Bizim halda **hello**) SIP mesajına əlavə edəcək.

Qeyd edin ki, sizin FreeSWITCH-ə yolladığınız bütün event mesajları iki **CRLF** simvol ardıcılılığı ilə qırılmalıdır.

FreeSWITCH-ə yollaya biləcəyiniz bütün event siyahısını bu başlıqda FreeSWITCH event sistemi əmrləri seksiyasında detallı şəkildə görəcəksiniz.

Dialplan-dan gələn eventlər

mod_event_socket-i **socket** adlı Dialplan programı təqdim edir hansı ki, çıxış TCP qoşulmalarına IP və port daxil edilməsinə imkan verir və digər tərəf isə əmrləri geriyə FreeSWITCH-ə axın olaraq qaytara bilər. Bu Asterisk üzərində olan şəbəkə bazalı fast-AGI(FAGI)-a oxşayır amma, bu gözləmədən istənilən əlavə eventlər ya da cavabların idarə edilməsi və yerinə yetirilməsi üçün əmrlərə izin verir ki, tam asinxron rejimdə işləyə bilsinlər.

Siz çıxış **socket**-ini çağırıldığda, FreeSWITCH avtomatik olaraq zəngi yerləşdirir. Zənglərin yerləşdirilmiş statusuna **CHANNEL_PARK** eventi vasitəsilə baxa bilərsiniz.

Dialplan-dan **socket**-in çağırılması sintaksisi aşağıdakı kimidir:

```
<ip>:<port> [<keywords>]
```

Bunun Dialplan-da necə istifadə edilməsinin misalları aşağıda göstərilir:

```
<action application="socket" data="127.0.0.1:8084"/>
<action application="socket" data="127.0.0.1:8084 async"/>
<action application="socket" data="127.0.0.1:8084 full"/>
<action application="socket" data="127.0.0.1:8084 async full"/>
```

Istəkdən asılı olan açar sözləri **async** və **full** aşağıdakı kimi açıqlanır:

- **async**: Bu o deməkdir ki, bütün əmrlər gözləmədən qayıdacaq. Bu imkan verir ki, əmrlər axını yerinə yetirilən müddətədək eventlər üçün socket-i monitoring edə bilək. Əgər async açar sözü olmazsa, hər bir event socket-i bitənədək blok ediləcək.
- **full**: full açar sözü onu deyir ki, digər tərəfin event socket-i üçün tam əmrlər siyahısı mövcuddur. Bu həmin əmrdir hansı ki, giriş socket qoşulması eventinə sahibdir və beləliklə də siz API əmrlərini əldə edərək də və.s yerinə yetirə bilərsiniz. Əgər full açar sözü olmazsa, onda əmrlərin təyinatı və eventlər bu zəng üçün limitli olacaq. Digər sözlə desək, əgər **full** təyin edilməzsə onda bu socket qoşulmasına ötürülmüş əmrlər yalnız hal-hazırkı işlək kanala təsir edə bilər. Uyğun

olaraq, socket qoşulması yalnız bu kanala aid olan eventləri qəbul edəcək.

mod_event_multicast

mod_event_multicast modulu **mod_event_socket**-ə çox oxşayır. Bu multicast şəbəkələri üzərindən 3-cü tərəf programlara və digər FreeSWITCH nüsxələrinə adı mətn event məlumatlarının göndərilməsi və qəbul edilməsinə şərait yaradır. Digər maşınlar özlərində olan eventləri işə salaraq, qulaq asmaq və parçalamaq üçün quraşdırıla bilər.

Event başlıqları "**Orig-**" prefiksle olan original başlıqları və **multicast::event** alt tipi olan **CUSTOM** tipləri çıxmaq şərtilə adı eventlərlə eynidir. **Multicast-Sender** header-idə həmçinin əlavə edilmişdir. Aşağıdakı **mod_event_multicast** tərəfindən FreeSWITCH-ə kənardan gələn paket nüsxəsi göstərilir:

```
Event-Name: CUSTOM
Core-UUID: 12938281-57ce-11de-9be6-99a22d850f40
FreeSWITCH-Hostname: SYS1
FreeSWITCH-IPv4: 192.168.1.12
FreeSWITCH-IPv6: %3A%3A1
Event-Date-Local: 2010-01-16%2018%3A15%3A10
Event-Date-GMT: Tue,%2016%20Jun%202009%2022%3A15%3A10%20GMT
Event-Date-Timestamp: 1245190510366825
Event-Calling-File: mod_event_multicast.c
Event-Calling-Function: mod_event_multicast_runtime
Event-Calling-Line-Number: 313
Event-Subclass: multicast%3A%3Aevent
Multicast: yes
Multicast-Sender: 5211c5b8-ac42-11e2-8176-d16e41886f24
Orig-Event-Name: CUSTOM
Orig-Core-UUID: 8784372-5ecc-4eaa-9002-9992b7ab7c4d
```

Siz **mod_event_multicast** modulunu **/usr/local/freeswitch/conf/autoload_configs/event_multicast.conf.xml** faylında quraşdırıa bilərsiniz:

- **address**: Mənsəbin IP ünvani hara ki, event yollanacaq.
- **port**: Mənsəbin TCP port-u hara ki, event yollanacaq.
- **bindings**: Bindings təyin edir ki, hansı eventləri multicast ünvanına göndərmək lazımdır. Bu başlığın əvvəlində göstərdiyimiz event **<arg><arg>** formatı ilə eynidir.
- **ttl**: Siz paketlər üçün TTL(Time To Live)-i təyin edə bilərsiniz ki, o paketlər uyğun vaxtında təqdim edilməzsə, silinsin. Bu sizin LAN/WAN keçirici avadanlığınızdan asılıdır. Bu rəqəm keçəcək "**hop**" sayını təyin edir.

Aşağıda **mod_event_multicast** üçün nüsxə göstərilir:

```
<param name="address" value="225.1.1.1"/>
<param name="port" value="4242"/>
<param name="bindings" value="PRESENCE_IN CUSTOM sofia::register CUSTOM
multicast::event"/>
<param name="ttl" value="1"/>
```

FreeSWITCH event sistem əmrləri

FreeSWITCH-ə qoşulmaq üçün istənilən event bazalı utilitlərin istifadəsində aşağıda mövcud ola biləcək əmrlərin siyahısı sadalanır. Siz bu əmrləri ESL(FreeSWITCH Event Socket Library hansı ki, başlığın əvvəlində haqqında danışmışdır)-dən də **mod_event_socket** üzərindən və event sisteminə giriş üçün FreeSWITCH tərəfindən təqdim edilən digər standart interfeyslə istifadə edə bilərsiniz. Sintaksis bütün yetki metodları üçün eynidir həmçinin, seçilmiş modullar üçün fərqli variasiyalarda formatlama və kodlaşdırma mövcuddur.

auth <password>

Siz **mod_event_socket** üzərindən, FreeSWITCH event socket sisteminə ilk dəfə qoşulduqda qeydiyyatdan keçməlisiniz. Aşağıdakı əmr sizə imkan verir ki, authentifikasiya parametrlərini ötürəsiniz:

```
auth ClueCon
```

api

api əmri **Application Programming Interface (API)** işə salır. İstənilən API əmri FreeSWITCH CLI-dan işə salına bilər. Bu uyğun əmri blok rejimində işə salır hansı ki, bu əmr bitənədək heç bir əmrin yerinə yetirilməsinə izin vermir və açıq socket qaytarır.

Sintaksisi: **api <command> <arg>**

Misallar:

```
api originate sofia/mydomain.com/4158867999@telco.com 1000
```

Bu zəngi (415)886-7999-a telco.com üzərindən inisializasiya edəcək və onu daxili **1000** genişlənməsinə qaytaracaq.

```
api sleep 5000
```

Bu sleep əmrini 5(5000 millisaniyə) saniyə müddəti üçün işə salır.

bgapi

Işı arxa fonda işə salır və Job-UUID sütunu məsləhətçi ilə arxa fona qaytarır. Əmr həqiqətəndə yerinə yetirildikdə və bitdikdə, nəticə event kimi, inisializasiyə edilmiş uyğun olan UUID ilə göndəriləcək.

bgapi əmri eyni argumentləri əmrlər üçün **api** əmri kimi öncə açıqlandığı qaydada qaytaracaq. Fərq yalnız ondan ibarətdir ki, server durmadan cavabı qaytarır və çoxlu əmrlərin emalı üçün mövcud olur.

Sintaksisi: **bgapi <command><arg>**

Misallar:

```
bgapi originate sofia/example/300@foo.com 8600
Content-Type: command/reply
Reply-Text: +OK Job-UUID: c7709e9c-1517-11dc-842a-d3a3942d3d63
```

Əmr yerinə yetirilməni bitirdikdə, FreeSWITCH hadisəni Job-UUID sütununda eyni UUID ilə işə salacaq. Cavab mesajı hadisəsinin tipi **BACKGROUND_JOB** olacaq, beləliklə, siz hadisələrin o tiplərini almaq, cavabı görmək üçün üzv olmalı olacaqsınız. Nüsxə cavabı misal üçün aşağıdakı kimi ola bilər:

```
Content-Length: 625
Content-Type: text/event-plain
Job-UUID: c7709e9c-1517-11dc-842a-d3a3942d3d63
Job-Command: originate
Job-Command-Arg: sofia/default/300%20foo.com
Event-Name: BACKGROUND_JOB
Core-UUID: 42bdf272-16e6-11dd-b7a0-db4edd065621
FreeSWITCH-Hostname: ser
FreeSWITCH-IPv4: 192.168.1.104
FreeSWITCH-IPv6: 127.0.0.1
Event-Date-Local: 2008-05-02%2007%3A37%3A03
Event-Date-GMT: Thu,%2001%20May%202008%2023%3A37%3A03%20GMT
Event-Date-timestamp: 1209685023894968
Event-Calling-File: mod_event_socket.c
Event-Calling-Function: api_exec
Event-Calling-Line-Number: 609
Content-Length: 41
+OK 7f4de4bc-17d7-11dd-b7a0-db4edd065621
```

Arxa fonda olan işin cavabında Orijinal Job ID-si Job-UUID-də sadalanacaq o vaxtadək ki, zəngin UUDI-si əlavə tərkib datanın icində originate əmri yaradır (Bu originate əmrinin nəticəsidi), bu halda **+OK 7f4de4bc-17d7-11dd-b7a0-db4edd065621**. Əgər siz FreeSWITCH-lə asinxron əlaqəyə girəcək program yaratsanız əmin olun ki, əmrlərin yerinə yetirilməsi üçün bgapi istifadə edilib və BACKGROUND_JOB eventləri üçün üzv olun. **bgapi** əmrinin uyğun BACKGROUND_JOB event ilə tutuşdurulması üçün, **Job-UUID** sütun mənasını istifadə edin. **BACKGROUND_JOB** işi "**final**" nəticəni özündə saxlayacaq hansı ki, **bgapi** üzərindən göndərilmişdir.

event

event əmri eventlərin axının işə salır və ya dayandırır. Module üzərindən axın edilir ki, event əmrini yerinə yetirsin (bu **mod_event_socket** TCP qoşulması və **mod_erlang_event** və.s.). Siz spesifik event klas tiplərinə üzv ola ya da bütün tiplərə üzv ola bilərsiniz.

'**event**'-in ardıcıl çağrıları önce təyin edilən və müraciət edilən event təyinatını silib üstünə yazacaq.

Sintaksisi: **event plain <list of event types | all>**

Misallar:

```
event plain ALL
```

Bu bütün eventlərin nüsxələnməsinə müraciət edir.

```
event plain CUSTOM conference::maintenance
```

Bu yalnız spesifik konfrans modulları, təkmilləşmə eventləri, **CUSTOM** eventlərin nüsxələnməsi üçün müraciət edir.

```
event plain CHANNEL_CREATE CHANNEL_DESTROY CUSTOM
conference::maintenance sofia::register sofia::expire
```

Bu yalnız kanalın yaradılması və ya məhv edilməsi eventlərini, **conference** modullarından bütün seçilmiş eventlər, sistem təkmilləşməsi, Sofia qeydiyyatı və sistemin vaxt bitməsi üçün eventləri nüsxələyir.

noevents

Bu əmr öncə event əmri ilə işə salınan bütün eventləri dayandırır.

Istifadəsi:

```
noevents
```

divert_events

divert_events əmri eventlərə izin verir ki, daxili script sayəsində içəri event socketinə zəng ötürülsün. Bu o deməkdir ki, FreeSWITCH-də işləyən script-n tələbi var ki, daxil edilmə həqiqətən də bunu kənar qoşulma programından ala bilsin hansı ki, event cavabını socket qoşulması üzərindən göndərir.

Girişdə olan geriyə zəng daxili **setInputCallback()** scriptdən istifadə edilərək qeydiyyatdan keçə bilər (Biz **setInputCallback()** istifadəsini 7-ci başlıqda Lua istifadə edərək Dialplan scripting-də göstərmışık). **divert_events**-in **on** təyin edilməsi chat mesajlarında istifadə edilə bilər. Misal üçün Gtalk channel, **automatic speech recognition(ASR)** eventləri və digərləri.

Sintaksisi: **divert_events <on|off>**

Misallar:

```
divert_events on
divert_events off
```

filter

Qulaq asması üçün event tipini təyin edir. Socket qoşulmasında çoxlu sayıda filterə izin verilir. Qeyd edin ki, bu əmr "**filter in**"-dir filter out deyil (**nixevent** seksiyasına baxın). Çoxlu süzgəclər təyin edin ki, görmək istədiyiniz ən az event tipləri göstərsin.

Sintaksisi: **<EventHeader> <ValueToFilter>**

Misallar:

Aşağıdakı misal bütün eventlərə üzv olacaq:

```
events plain all
Content-Type: command/reply
Reply-Text: +OK event listener enabled plain
```

Bütün eventlərə üzv olur.

```

filter Event-Name CHANNEL_EXECUTE
Content-Type: command/reply
Reply-Text: +OK filter added. [filter]=[Event-Name CHANNEL_EXECUTE]
filter Event-Name HEARTBEAT
Content-Type: command/reply
Reply-Text: +OK filter added. [Event-Name]=[HEARTBEAT]

```

Filterlər yalnız tipi **CHANNEL_EXECUTE** və **HEARBEAT** olanları qaytaracaq.

Siz istənilən event header-inə görə filter edə bilərsiniz. Spesifik kanal süzgəci üçün siz **uuid** sintaksisi istifadə edərək yazmalısınız:

```
filter Unique-ID d29a070f-40ff-43d8-8b9d-d369b2389dfe
```

Süzgəclər kombinasiyası istifadə edin ki, socket-dən gələn eventlərin sayını azaldasınız.

filter delete

Dayandırmaq istədiyiniz event süzgəcini təyin edin. Bu o zaman yararlı olur ki, birdən həddən artıq event datası süzgəcdən keçir və lazımlarda görünmür.

Sintaksisi: **filter delete <EventHeader> <ValueToFilter>**

Misallar:

```

filter delete Event-Name HEARTBEAT
filter delete Unique-ID d29a070f-40ff-43d8-8b9d-d369b2389dfe

```

Bu göstərilmiş **Unique-ID**-yə mənimsədilmiş filter-i siləcək. Bundan sonra, siz **Unique-ID** üçün heç bir event qəbul eləməyəcəksiniz.

```
filter delete Unique-ID
```

Bu **Unique-ID** üçün mənimsədilən bütün süzgəcləri siləcək.

nixevents

Bu əmr filter əmrinin əksidir hansı ki, təyin edilmiş event tipinin qayıtmاسının qarşısını alır.

Istifadəsi:

```
nixevents <event types | ALL | CUSTOM custom event sub-class>
```

sendevent

Event sisteminin içində eventi yolla(başlıqlar üçün çoxsətirli giriş).

Sintaksisi: **sendevent <event-name>**

Bu FreeSWITCH daxili event sistemi ilə eventi generasiya edir. Bu eventə üzv olan istənilən modullar və ya sistem prosesləri həmin eventi alacaq.

Əgər siz **sendevent**-i event tipi təyin eləmədən istifadə edirsinizsə və Event-Name header-ni istədiyiniz event adı ilə əlavə edirsinizsə, istənilən event tipi təyin edə bilərsiniz. Misal üçün:

```
sendevent SOME_NAME
Event-Name: CUSTOM
Event-Subclass: albs::Section-Alarm
Section: 33
Alarm-Type: PIR
State: ACTIVE
```

sendevent əmrinin misali aşağıdakı kimidir:

```
sendevent NOTIFY
profile: internal
content-type: application/simple-message-summary
event-string: check-sync
user: 1005
host: 192.168.10.4
content-length: 5
hello
```

sendmsg <uuid>

UUID(**call-command execute** ya da **hangup**)-si götürülən zəngə mesajı yolla. Bu əmri istifadə elə ki, progress-də olan zənglərin davranışını idarə edə biləsiniz. Zəng üçün UUID-ni təqdim eləməlisiniz.

Qayda ilə mesajın göndərilməsini istifadə eləməklə zəngləri idarə eləmək üçün onlar park edilməlidir. Park edilmiş zəng bir növ statusu təyin edilməyən kanala deyilir hansı ki, sizə həmin kanalda artıq yerinə yetirilən (**Qeyd:** Park edilmiş zəng heç bir medianı qəbul eləməyəcək, **music-on-hold**-da daxil olmaqla) digər programları dayandırmadan programı yerinə yetirməyə şərait yaradır.

Siz **&park()** sintaksi istifadə eləməklə zəngi birbaşa park-a ötürə bilərsiniz:

```
originate sofia/example/300@foo.com &park()
```

Kanalda yerinə yetirilə biləcək kök işlərinin iki tipi mövcuddur: **execute** və **hangup**. Bu iki iş aşağıdakı seksiyalarda açıqlanmışdır.

execute

execute əmri Dialplan programlarının yerinə yetirilməsi üçün istifadə edilmişdir. Siz programın adını və programın argumentlərini yerinə yetirilən müraciətin içini əlavə edə və programı çoxlu sayda dövrə sala bilərsiniz. Aşağıdakı nüsxə adı **.wav** faylini əlavə edə bilər.

Format aşağıdakı kimidir:

```
SendMsg <uuid>
call-command: execute
execute-app-name: <one of the applications>
execute-app-arg: <application data>
loops: <number of times to invoke the command, default: 1>
```

Misal olaraq deyə bilərəm ki, aşağıdakı **SendMsg** əmri təyin edilmiş **<uuid>** adlı kanalda **test.wav** adlı faylı oxudacaq.

```
SendMsg <uuid>
call-command: execute
execute-app-name: playback
execute-app-arg: /tmp/test.wav
```

Əgər sizin **SendMsg** əmri ilə ötürəcəyiniz arqument **2048** simvolu aşarsa biraz fərqli formatı istifadə edə bilərsiniz:

```
SendMsg <uuid>
call-command: execute
execute-app-name: <one of the applications>
loops: <number of times to invoke the command, default: 1>
content-type: text/plain
content-length: <content length>
<application data>
```

Tünd qara rəngli sətirlərə diqqət yetirin. Siz programda ötürüləcək olan mətn uzunluğunu təyin edə və sonra bütöv şəkildə proqrama ötürə bilərsiniz.

hangup

Bu əmr aktiv zəngləri durdurur.

Format:

```
SendMsg <uuid>
call-command: hangup
hangup-cause: <recognized hangup cause>
```

noMedia

nomedia əmri ilə real vaxtda FreeSWITCH-in media ünvanında olub olmamasını idarə eləmək mümkündür. Bu əmr size seçilən kanal üçün medianın işə salınması və ya dayandırılmasını idarə eləmək üçün imkan yaradır.

Istifadəsi:

```
SendMsg <uuid>
call-command: nomedia
nomedia-uuid: <noinfo>
```

log <level>

Bu əmr jurnalların çıxışına imkan yaradır. Siz görmək istədiyiniz jurnal səviyyəsini təyin edə bilərsiniz. Bu size FreeSWITCH CLI-da olan bütün jurnal hadisələrini görməyə şərait yaradır.

Istifadəsi:

```
log <level>
```

nolog

Bu əmr öncəki **log** əmri ilə işə salınan jurnal səviyyəsini durdurur.

Istifadəsi:

nolog

linger

FreeSWITCH-ə deyir ki, kanal dayandırıldıqda socket-i bağlama. Bu socket client tərəfindən kanala gələn uyğun eventi qəbul eləmədiyi müddətədək socket qoşulmasını aktiv saxlayacaq.

Istifadəsi:

linger

noligner

Bu əmr öncə **linger** əmri ilə işə salılmış olan funksionallığı dayandırır.

Istifadəsi:

noligner

FreeSWITCH konsole programı

Əksər insanlar bunu eləmirlər ama onlar FreeSWITCH Console **fs_cli** programını istifadə etmişlərsə onda, onlar artıq FreeSWITCH event socket alt sistemini istifadə etmişlər. **fs_cli** programı C dilində yazılmışdır və FreeSWITCH-in event socket-inə **mod_event_socket** vasitəsilə qoşulur. O bütün sistem jurnallarını istifadə edir, rəngləyir və event mesajları formasında geriyə qayıtmaları üçün interfeys təqdim edir. Mövcud FreeSWITCH konsolu bu program tərəfindən yenidən yaradılmışdır (Siz **fs_cli**-in mənbə koduna FreeSWITCH mənbə kodları olan **libs/esl/fs_cli.c** faylda baxa bilərsiniz)

Event Socket kitabxanası

FreeSWITCH Event Socket Library API-lar standartlarıdır hansı ki, fərqli program dillərinin istifadə edilə bilməsi üçün onları modul olaraq yüklenmə şəraiti yaradır. Başqa sözlə desək seçdiyiniz dil üçün API-lar yükləndikdə FreeSWITCH event funksionallığına heç bir TCP socket qoşmadan və ya şəbəkədə başqa iş görmədən yetki almış olacaq.

Dəstəklənən kitabxanalar

FreeSWITCH standartlaşdırılmış API-larin yaradılması üçün SWIG (<http://www.swig.org/>) istifadə edir. SWIG təyin edilmiş dəyişənlər, zənglərin funksiyalarının siyahısını götürür və avtomatik kitabxanalar yaradır hansı ki, FreeSWITCH kök kodlarının yüklənilə bilən dillər modullarına link edilir. Aşağıdakı dillər susmaya görə dəstəklənir:

- Perl
- PHP
- LUA
- Python
- Ruby
- C
- TCL
- .NET

Növbəti obyektlər və metodlar **ESL** genişlənməsi yarada bilən istənilən dilə mənimsədilə bilər. Artıq öz seçdiyiniz dil üçün uyğun modulu yükledikdən sonra, istənilən standart ESL obyektlərini, funksiyalarını və dəyişənlərini istifadə edə bilərsiniz. Ümumi funksiyalar Aşağıdakı cədvəldə sadalanır. FreeSWITCH ESL sizin əksər dəyişdirmə tiplərinizdə, özünüzə aid olan tiplərin yaradılmasında, dəyişən strukturlarında və bu əmrlərin istifadə edilməsində **SWIG** istifadə edir.

ESLObject

ESL obyektin özəyi **ESLObject**-dir. Siz FreeSWITCH-dən qayıdacaq eventlər üçün **loglevel** məlumatını təyin edə bilərsiniz.

eslSetLogLevel(\$loglevel)

eslSetLogLevel(\$loglevel) serverdə jurnal səviyyəsini təyin edir. **\$loglevel** özü 0 və 7 arasında olan tam rəqəmdir. **\$loglevel** üçün mənalar Aşağıdakılardır:

- 0 is EMERG
- 1 is ALERT
- 2 is CRIT
- 3 is ERROR
- 4 is WARNING
- 5 is NOTICE
- 6 is INFO
- 7 is DEBUG

ESLevent object

Event qayıtdıqda, siz ESLevent obyekti əldə edəcəksiniz. Bu obyektin fərqli köməkçi funksiyaları olur ki, əldə edilən eventi analiz edib emal edə biləsiniz.

serialize([\$format])

`serialize([$format])` eventi iki nöqtə ilə ayrılan "name: value" cütlüyüünə

SIP/e-mail paketinə uyğun olaraq ötürür.

setPriority([\$number])

setPriority([\$number]) işə düşdürü anda, **\$number** üçün event-e prioritet təyin edir.

getHeader(\$header_name)

getHeader(\$header_name) hadisə obyektindən **\$header_name** açarlı başlığı alır.

getBody()

getBody() event obyektinin tərkibini alır.

getType()

getType() event obyektinin tipini alır.

addBody(\$value)

addBody(\$value) event obyektinin tərkibinə **\$value** əlavə edir. Bu eyni event obyekti üçün çoxlu sayda çağırıla bilər.

addHeader(\$header_name, \$value)

addHeader(\$header_name,\$value) başlığın **\$header_name** açarı olduğu yerlərə header əlavə edir və event obyekti mənasını **\$value** təyin edir. Bu eyni event obyekti üçün çoxlu sayda çağırıla bilər.

delHeader(\$header_name)

delHeader(\$header_name) headeri **\$header_name** açarı ilə event obyektindən silir.

firstHeader()

firstHeader() göstəricini event obyekti növbəti başlığına təyin edir və açar adını qaytarır. Bu nextHeader çağırılmışdan önce işə salınmalıdır.

nextHeader()

nextHeader() göstəricini event obyekti növbəti başlığına köçürür və açar adı qaytarır. **firstHeader** yönləndiricini təyin eləmək üçün bu metoddan önce çağırılmalıdır. Əgər bu metod çağırılışında siz son header-də olsanız, cavab **NULL** qayıdacaq.

ESLconnection object

ESLconnection obyekti event emalı üçün FreeSWITCH-ə qoşulmayı dəstəkləyir. Bu obyekti FreeSWITCH-ə qoşulmayı dəstəkləyir və mesajların gediş/qayıdışını emal edir.

new(\$host, \$port, \$password)

Bu əmr ESLconnection-un yeni prosesini inisializasiya edir və **\$host** hostuna **\$port** portu ilə qoşulub FReeSWITCH serverə **\$password** təqdim edir. Bu yalnız event socket-də **inbound** rejimində olur. Bu funksiyani FreeSWITCH-lə əlaqə yaranan da istifadə edin hansı ki, başlıqda heç bir təyin edilməyən zəng və kanalla əlaqəyə girmir.

new(\$fd)

Bu əmr tərkibində **\$fd** olan mövcud fayl rəqəmini(**file descriptor**) istifadə edərək **ESLconnection** üçün, yeni əmri inisializasiya edir.

Siz bunu Event socket outbound qoşulmalarında istifadə edə bilərsiniz. Hətta düzgün daxili socket-ə keçsədə belə, inbound qoşulmalarında uğursuz olacaq.

socketDescriptor()

Bu əmr qoşulma obyekti üçün(əgər qoşulma mövcuddursa) UNIX fayl deskriptorunu qaytarır. Bu outbound rejimində istifadə ediləndə yeni **\$fd**-yə ötürülmən fayl deskriptorudur.

connected()

Bu əmr qoşulma obyektinin qoşulmuş olmasını yoxlayır. Qoşulduğu halda **1**, əks halda isə **0** qaytarır.

getInfo()

FreeSWITCH "Event Socket Outbound" emaledicisi qoşulduğda, o **CHANNEL_DATA** eventini ilk event kimi, hətta ilk qoşulmadan sonra yollayacaq. **getInfo()** bu Channel Data tərkibli **ESLevent**-i qaytarır.

"Event Socket Outbound" qoşulması istifadə edildikdə, **getInfo()** NULL qaytarır.

send(\$command)

Bu əmr FreeSWITCH-ə əmr yollayır və cavabı gözləmir. Siz cavabı almaq üçün dövrə **recvEvent** və **recvEventTimed** eventlərini çağırı bilərsiniz. Cavab eventinin **content-type** adlı başlığı olur hansı ki, **api/response** ya da **command/reply** cavablarına malik olur.

Avtomatik cavab eventini gözləmək üçün, **send()** əvəzinə **sendRecv()** istifadə edin.

sendRecv(\$command)

Daxildə **sendRecv(\$command)**, **send(\$command)** və sonra **recvEvent()** çağırır və **ESLevent** instansını qaytarır.

recvEvent() dövrə content-type başlığı qaytarmayanadək işə salınır hansı ki, mənimsədilmiş mənaları **api/response** və **command/reply** olacaq və sonra onları ESLevent instansi kimi qaytaracaq.

recvEvent() tərəfindən gələn istənilən event növbədə olan tranzaksiya ilə heç bir əlaqəsi olmur və sizin programda olan növbəti ardıcıl **recvEvent()**-ə qaytarılacaq.

api(\$command[, \$arguments])

API əmrini FreeSWITCH serverə yollayır. Bu metod əmr yerinə yetirilməyənədək sonrakı yerinə yetirilməni dayandırır.

api(\$command,\$args) əmri **sendRecv("api \$command \$args")** ilə identikdir.

bgapi(\$command[, \$arguments])

FreeSWITCH-in öz axınında işə düşəcək və blok edilməyəcək API əmrini FreeSWITCH serverə ötürür.

bgapi (\$command,\$args) əmri **sendRecv("bgapi \$command \$args")** ilə identikdir.

sendEvent(\$send_me)

FreeSWITCH event sisteminə hadisəni daxil et. Bu sizə hadisə müştəriləri üçün istifadə və yerinə yetirə bilməsi üçün, FreeSWITCH-ə hadisənin yollanmasına kömək edir.

recvEvent()

Bu FreeSWITCH-dən gələn növbəti hadisəni qaytarır. Əgər heç bir hadisə gözlənilmirsə, bu zəng hər hansıa bir hadisə gəlməyənədək blok ediləcək. Əgər hər hansı hadisələr **sendRecv()** zəngi zamanı növbəyə qoyulmuşsa, onda birinci növbədən silinmiş qaytarılacaq. Əks halda növbəti hadisə qoshulmadan oxunacaq.

recvEventTimed(\$milliseconds)

Bu əmr **recvEvent()**-ə oxşayır amma **\$milliseconds**-da təyin edilən vaxtı aşarsa blok edəcək. **recvEventTimed(0)**-a gedən zəng gözlənilmədən qayıdacaq. Bu zəngin atılması üçün yararlıdır.

filter(\$header, \$value)

Event socket **filter** əmrinə baxın.

events(\$event_type,\$value)

\$event_type-in **plain**, **json** ya da **xml** mənaları ola bilər. **\$event_type** üçün təyin edilən istənilən digər mənalar **plain** ilə dəyişdiriləcək.

execute(\$app[, \$arg][, \$uuid])

Dialplan programını yerinə yetir və serverdən cavab üçün gözlə. Soket qoşulmaları olan kanalda (tez-tez, daxil olan hadisənin socket qoşulmaları ilə) bağlantılarda üç parametr tələb olunur. **\$uuid** kanalı müəyyən edir, hansında ki, əlavəni (programı) yerinə yetirmək olar.

execute() serverdən qaydan ESLevent obyektini təşkil edir. Bu ESLevent obyektin **getHeader("Reply-Text")** metodu serverin cavabını qaytarır. Server uğurlu cavabda **+OK [Success Message]** qaytaracaq əksinə, səhv olduqda **-ERR [Error Message]** cavabı qaytaracaq.

executeAsync(\$app[, \$arg][, \$uuid])

Bu əmr **execute** ilə eynidir; o halda ki, o serverdən gələn cavab üçün gözləmir (Programlaşdırımda **executeAsync** mənasi "**non-blocking**" deməkdir).

Bu kanala göndərilən mesajın sonuna **async:true** header əlavə etməklə başlanğıc zəngin **execute()** etməsini məcbur edir.

setAsyncExecute(\$value)

async rejimini socket qoşulması üçün məcburi işə salır. Bu əmrin çıxış socket qoşulmaları üçün effekti yoxdur hansı ki, daxili socket qoşulmalarında Dialplan-da async təyin edilir ona görə ki, bu qoşulmalar artıq **async** rejimdə qoşulu olur.

Məcburi async rejimi işə salmaq üçün **\$value** mənəsi **1** və dayandırmaq üçün **0** olmalıdır.

Spesifik olaraq **setAsyncExecute(1)** gələcək zənglərin kanalın içində **async:true** başlığı ilə **execute()** etməsinə məcbur edir. Digər event socket kitabxanaları rutini bu zəngə təsir göstərmir.

setEventLock(\$value)

socket qoşulması üçün **sync** rejimi gücləndirir. Bu əmrin çıkış socket qoşulmalarında effekti yoxdur hansı ki, Dialplan-da async təyin edilmir ona görə ki, bu qoşulmalar artıq sync rejimdə təyin edilmişdir.

Məcburi **sync** rejimi işə salmaq üçün **\$value** mənəsi **1** və dayandırmaq üçün **0** olmalıdır.

Spesifik olaraq **setEventExecute(1)** gələcək zənglərin kanalın içində **event-lock:true** başlığı ilə **execute()** etməsinə məcbur edir. Digər event socket kitabxanaların rutini bu zəngə təsir göstərmir.

disconnect()

FreeSWITCH serverə socket qoşulmasını bağlayır.

Təcrübədə olan eventlər

Gelin təcrübədə bir neçə event istifadəsinə baxaq.

Event Socket Library misali - əmrin yerinə yetirilməsi

Aşağıdakı PHP nüsxəsi size bir sətir əmrin necə yerinə yetirilməsini göstərir. FreeSWITCH-in Event Socket Library-sini istifadə edərək siz bu əmrləri FreeSWITCH-ə yollaya və cavabını gözləyə bilərsiniz.

```
// FreeSWITCH ESL kitabxanasını əlavə edir. Qeyd ESL.php kodu
// FreeSWITCH PHP ESL modulu ilə gəlir.
require_once('ESL.php');
if ($argc <= 1) {
    printf("ERROR: You Need To Pass A Command\nUsage:\n\t%s <command>\n",
$argv[0]);
    exit();
}
```

```
// Yerinə yetirilən addan izolyasiyani götürürük($argv[0]) array_shift($argv);
$command = sprintf('%s', implode(' ', $argv));
printf("Command to run is: %s\n", $command);
// FreeSWITCH-ə qoşulur
$sock = new ESLconnection('localhost', '8021', 'ClueCon');
// Əmri yollayır
$res = $sock->api($command);
// Cavabı çap edir
printf("%s\n", $res->getBody());
```

Eventlərin FreeSWITCH-ə göndərilməsi misalları

Aşağıdakı misallar size FreeSWITCH event sisteminin daxilinə hadisənin yollanılması ilə nələr edə biləcəyinizi göstərir.

Telefon işıqlarının təyin edilməsi

Əksər telefonlar SIP presence mesajları vasitəsilə işıqların yandırılması və söndürülməsini dəstəkləyirlər.

Işıqların yandırılması

Siz telefonun işıqlarını FreeSWITCH-ə presence eventi yollamaqla yandırma bilərsiniz hansı ki, sonra presence mesajını telefona yollayacaq. FreeSWITCH event socket-nə qoşulun və Aşağıdakı event-i yollayın:

```
sendevent PRESENCE_IN
proto: sip
from: 1000@example.com
login: 1000@example.com
event_type: presence
alt_event_type: dialog
Presence-Call-Direction: outbound
answer-state: confirmed
```

1000@example.com kanal düyməsi olan istənilən şəxs işığın yanmasını görcək. Qeyd edin ki, **answer-state** header ilə təsdiq **key/value** cütlüyü ilə ehtiyatlı olun. Bu o deməkdir ki, aktiv zəng baş verir və işıq yanılı olmalıdır.

Işıqları söndürürük

Siz FreeSWITCH-ə presence eventi yollayaraq işıqları söndürə bilərsiniz. FreeSWITCH event socketine qoşulduğdan sonra Aşağıdakı event-i yollayın:

```
sendevent PRESENCE_IN
proto: sip
from: 1000@frfs.opensource.az
login: 1000@frfs.opensource.az
event_type: presence
alt_event_type: dialog
Presence-Call-Direction: outbound
answer-state: confirmed
```

```
root@frfs:~ # telnet localhost 8021
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
```

Content-Type: auth/request

```
auth ClueCon
sendevent PRESENCE_IN
proto: sip
from: 1000@frfs.opensource.az
login: 1000@frfs.opensource.az
event_type: presence
alt_event_type: dialog
Presence-Call-Direction: outbound
answer-state: confirmed
```

Content-Type: command/reply
 Reply-Text: +OK accepted

answer-state headerinin terminated statusu ilə ehtiyatlı olun bu o deməkdir ki, bu liniyada zəng yoxdur(işığı söndür).

Telefonun yenidən yüklənməsi

FreeSWITCH-in imkanı var ki, SIP telefonların yenidən yüklənmələri üçün müraciət yollaya bilsin. Bu o hallarda yararlı olur ki, siz yeni quraşdirmalar etmisiniz və telefonu həmin quraşdirmalrı götürməsi üçün yenidən yükləyirsiniz hansı ki, əksər telefonlar yüklənmə müddətində quraşdırma fazasında avtomatik olaraq edəcək.

Siz qeydiyyatdan keçmiş yenidən yüklənmə etmək istədiyiniz telefonun **Call-ID** sütunu bilməlisiniz. Sizə lazım olan son nöqtəni **sofia status profile <profile_name> reg** əmrini **fs_cli**-dan daxil edərək əldə edə bilərsiniz. Sonradan sadəcə **sofia profile <profile_name> check_sync <call_id> reboot** əmrini yazmaqla telefonu yenidən yüklənməyə yollaya bilərsiniz. Ancaq biz bunu Event Socket üzərindən Aşağıdakı kimi edəcəyik(Əmrler və vacib olan yerlər tünd qara rənglə seçilmişdir):
 root@frfs:~ # telnet 127.0.0.1 8021
 Trying 127.0.0.1...
 Connected to localhost.
 Escape character is '^]'.
 Content-Type: auth/request

auth ClueCon

Content-Type: command/reply
 Reply-Text: +OK accepted

api sofia status profile internal reg

Content-Type: api/response
 Content-Length: 719

Registrations:

Call-ID: 78100NWVkm2J1MWIxOWY0NGNiZTU4MzZiNDlhMzEwOTZmODE

```
User:          1000@94.20.81.154
Contact:       "fr1000"
<sip:1000@217.168.189.168:56917;rinstance=2e190895bf2b01a5>
Agent:         Bria 4 release 4.2.1 stamp 78100
Status:        Registered(UDP) (unknown) EXP (2015-11-12 00:58:39)
EXPSECS (3573)
Ping-Status:   Reachable
Host:          frfs.opensource.az
IP:            217.168.189.168
Port:          56917
Auth-User:     1000
Auth-Realm:    frfs.opensource.az
MWI-Account:  1000@94.20.81.154
```

Total items returned: 1

```
api sofia profile internal check-sync
78100NWVkJlMWIxOWY0NGNiZTU4MzZiNDlhMzEwOTZmODE reboot
```

Telefonun yenidən quraşdırılması üçün müraciət edək

Bəzi telefonlar yenidən qurulmayı dəstəkləyirlər. Əgər SNOM telefonuna sahib olsanız, onun quraşdırımlarını Aşağıdakı sətirləri istifadə edərək yeniləyə bilərsiniz:

```
sendevent NOTIFY
profile: internal
event-string: check-sync;reboot=false
user: 1000
host: 85.132.57.60
content-type: application/simple-message-summary
```

Seçilmiş xəbərdarlıq mesajları

Siz fərqli kontente sahib olan event mesajlarını ötürə bilərsiniz. Misal üçün Aşağıdakı ola bilər:

```
sendevent NOTIFY
profile: internal
content-type: application/simple-message-summary
event-string: check-sync
user: 1000
host: 85.132.57.60
content-length: 2
OK
```

Aşağıdakına oxşar olan bir paket generasiya ediləcək:

```
NOTIFY sip:1000@85.132.57.60 SIP/2.0
Via: SIP/2.0/UDP 85.132.57.55;rport;branch=z9hG4bKpH2DtBDcDtg0N
Max-Forwards: 70
From: <sip:1000@85.132.57.55>;tag=Dy3c6Q1y15v5S
To: <sip:1000@85.132.57.55>
Call-ID: 129d1446-0063-122c-15aa-001a923f6a0f
CSeq: 104766492 NOTIFY
```

```
Contact: <sip:mod_sofia@85.132.57.55:5060>
User-Agent: Freeswitch-mod_sofia/1.0.0.trunk-9578:9586
Allow: INVITE, ACK, BYE, CANCEL, OPTIONS, PRACK, MESSAGE, SUBSCRIBE, NOTIFY,
REFER, UPDATE, REGISTER, INFO, PUBLISH
Supported: 100rel, timer, precondition, path, replaces
Event: check-sync
Allow-Events: talk, presence, dialog, call-info, sla, includesession-
description, presence.winfo, message-summary
Subscription-State: terminated;timeout
Content-Type: application/simple-message-summary
Content-Length: 2
OK
```

Qeyd edin ki, generasiyada SIP məlumat vermə mesajından başqa əlavə elədiklərimiz sütunlar da, bizim müraciətimizlə SIP-ə ötürülmüşdür.

Nəticə

Artıq FreeSWITCH event sisteminin iş prinsipini öyrəndikdən sonra, ağlınıza gələn istənilən ideyanı bu mühitdə reallığa çevirə bilərsiniz. Real vaxt rejimində istifadəçilərlə əlaqə və digər fərqli imkanları sınaqdan keçirməniz biliklərinizi bərkidəcək. Event sistemi limitsizdir və siz istənilən dizaynı bu mühitin üzərindən reallığa çevirə bilərsiniz.

Növbəti Başlıqda biz FreeSWITCH-in daha asan idarə edilməsinin üsuluna baxacayıq. FreeSWITCH-in **mod_httapi** vasitəsilə idarəedilməsinə baxacayıq.

11-ci başlıq

mod_httapi vasitəsilə WEB bazalı zəngin idarə edilməsi

mod_httapi program təminatı sizin IVR-lar və zəngin idarə edilməsi program təminatlarının daha dinamik edilməsi üçün yaradılmışdır. Bununla siz özünüz seçdiyiniz IVR bazalı istifadəçi girişini yarada bilərsiniz. FreeSWITCH-in **mod_httapi**-i adı **HTTP POST** əməliyyatı FreeSWITCH-in zəng axınının fərqli bit məlumatlarını göndərmək üçün asan üsuldan istifadə edir. Bu başlıqda biz Aşağıdakılari açıqlayacaq:

- HTTPAPI sintaksisi
- mod_httapi quraşdırma faylı
- HTTAPI-da demo IVR

Bu başlıqda oxuduğumuz kimi, **mod_httapi** iteraktiv zəng emal prosesini istifadə edir; Bu da o deməkdir ki, bir telefon zəngi üçün təkrarlanan HTTP POST müraciətləri olur. Bu programı hazırlayan şəxs üçün geniş elastiklik və programın dizaynında güc verir. Bir cavabın içinde bütün mümkün ola biləcək zəng axını məntiqini generasiya eləmək tələb edilmir. **httapi** programı tərəfindən idarə edilən telefon zəngi işləri HTTP cavabda yerinə yetirəcək(bu HTTAPI "document"-dir) və sonra digər HTTP POST müraciəti serverə yollayacaq. Həqiqətdə **httapi** programı instruksiyani serverdən alır, işə salır və sonra demək üçün web serverlə əlaqəyə girərək deyir "Bu işi bitirdim, sonra?". Bu iterasiya zəng bitənədək ya da **httapi** Dialplan programından kənara çıxanadək davam edəcək.

Qeyd: Iterasiya dövrdə işləyən programın bir dövrünə deyilir. Yeni bir dövr bir iterasiyadır.

HTTAPI sintaksisi

HTTAPI qeydi - adı emal edilmiş XML-dən fərqli bir şey deyil. Əksər hallarda başlanğıc HTTAPI sənədi aşağıdakı kimi olur:

```
<document type="text/freeswitch-httapi">
<variables/>
<params/>
<work/>
</document>
```

Bu sənəd müraciət edilən HTTP POST-un cavabında web serverdən qaytarılmışdır.

HTTAPI cavabin tərkib tipi **text/xml** olmalıdır. Bütün HTTAPI cavablar sənədin tag-na **text/freeswitch-httapi** atribut tipini əlavə etməlidir. Bununla da belə siz verilən cavabda bala tag-larda istənilən biri ya hamisini istifadə edə bilərsiniz. Bala taglar aşağıdakılardan kimi ola bilər:

- **params:** Bu POST **params**(parametrlər)-lari hər müraciətdə FreeSWITCH web serverə ötürür. Siz FreeSWITCH-ə seçdiyiniz POST parametrlərin ötürülməsi üçün **<params>** tag-dan istifadə edə bilərsiniz.
- **variables:** Bu kanal dəyişənləri kanaldan **httapi** Dialplan programını çağırır. **<variables>** tag-i sizə imkan yaradır ki, kanal dəyişənlərini ardıcıl müraciətlərdə FreeSWITCH Dialplan-da ya da geriyə httapi-da istifadə edə biləsiniz(Bu başlıqda daha detallı açıqlanır).
- **work:** Bu çoxlu maraqlı tərkibin baş verdiyi ünvandır. Burda **<work>** tagının bala tag-i olaraq çoxlu **action** tag istifadə edilə bilər ki, FreeSWITCH-ə istənilən idarə edilə bilən telefona telefon zəngində hər hansıa bir işin görülməsinə məcbur edilsin. Mesajların konsol-dan jurnallanması, səs fayllarının oxudulması, Automatic Speech Recognition edilməsi, DTMF açarlarının sıxılması və s. Növbəti başlıqda mövcud olacaq action tag-lar və onlara qoşula biləcək atributlar göstərilir.

Aşağıda göstərilən işlərdən əksəriyyətinin imkanı var ki, **bindings** əlavə eləsin və buda bizə izin verir ki, FreeSWITCH informasiyani toplayıb geriyə bizə qaytarsın. Bu web səhifədə olan WEB forma kimi emal edilir. Hər bir elementin "name"-i var və bu element üçün toplanan istənilən data eyni adın **POST param**-ı kimi sizin WEB programınıza ötürüləcək. Birləşmənin yenidən tutuşdurulması üçün müntəzəm ifadəsi(**regex**) olacaq hansı ki, daxil olan mənanın sonundan ayırmak üçün istifadə edilən əlavə rəqəm mənasıdır.

İşin hərəkətləri

HTTAPI **work** işləri bu seksiyada açıqlanır. Göründüyü kimi, ***DATA*** kontentin tag-dir(yəni, **<tag>*DATA*</tag>**).

Bütün iş tapşırıqlarının həmişə mövcud olan iki tag-ı var:

- **action:** Yəni susmaya görə olan mənsəb URL-i dəyişir.
- **temp-action:** Mənsəb URL-i növbəti müraciətə mənimsetmək üçün dəyişir. Ardıcıl müraciətlər susmaya görə olan URL-i istifadə edəcək ya da action tag-da təyin edilməsindən asılı olmayıaraq.

Aşağıdakilar **work** işlərinin siyahısı və açıqlamalarıdır.

playback

playback faylı oxudur və təyinatdan asılı olaraq daxil ediləni toplayır.

Aşağıdakı atributlara sahibdir:

- **file:** Oxudulacaq faylin ünvanı
- name:** Nəticənin saxlanılması üçün **Param** adı
- **error-file:** Yalnız daxil edilmədə oxundulacaq error faylı
- **digit-timeout:** Fayl oxunulmasından sonra vaxt bitməsini gözləyən rəqəmlər üçün (daxil edilmiş birləşmə mövcud olduqda)
- **input-timeout:** multi-digit daxiledilmədə çoxlu rəqəmlər üçün vaxt bitməsinin gözlənilməsi
- **loops:** Fayl oxudacaq vaxtin maksimal rəqəmi (Daxil edilən birləşim mövcud olarsa)
- **asr-engine:** Automated Speech Recognition (ASR) motorun istifadə edilməsi
- **asr-grammar:** Automated Speech Recognition (ASR) qramatikasının istifadə edilməsi
- **terminators:** Yığılmış rəqəmlərin işə salınması və durmadan dayandırılmasının istifadə edilməsi üçün açarlar

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <playback action="http://newurl/index.php" temp-
      action="http://newtempurl/index.php" name="playback_user_input"
      error-file="ivr/ivr-error.wav" file="ivr/ivr-
      welcome_to_freeswitch.wav" asr-engine="pocketsphinx" asr-
      grammar="my_default_asr_grammar" digit-timeout="5" input-
      timeout="10" loops="3" terminators="#">
      <bind strip="#">~\d{3}</bind>
    </playback>
  </work>
</document>
```

playback işi **playback** Dialplan programları üçün analoqdur.

vmname

vmname voicemail adını oxudur və istəkdən asılı olaraq daxil edilməni toplayır. Aşağıdakı atributlara sahibdir:

- **id:** user@domain kimi oxudulacaq istifadəçilərin adı
- **name:** Nəticəni saxlamaq üçün **Param** adı
- **error-file:** Yalnız daxil edilmədə oxundulacaq olan error faylı
- **digit-timeout:** Faylin oxudulmasından sonra (daxil edilmiş birləşmə mövcud olduqda), rəqəmlər üçün vaxt bitməsinin gözlənilməsi
- **input-timeout:** multi-digit daxil edilmədə (daxil edilmiş birləşmə mövcud olduqda) çoxlu rəqəmlər üçün vaxt bitməsinin gözlənilməsi
- **loops:** Fayl oxudulacaq olan sayı maksimal sayı (daxil edilmiş birləşmə mövcud olduqda)

- **terminators:** Yiğilmiş rəqəmlərin işə salınması və durmadan dayandırılmasının istifadə edilməsi üçün açarlar

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <vmname action="http://newurl/index.php"
      temp-action="http://newtempurl/index.php"
      name="vmname_user_input"
      error-file="ivr/ivrerror.wav"
      id="1007@192.168.1.101"
      digit-timeout="5"
      input-timeout="10"
      loops="3"
      terminators="#">
      <bind strip="#">~\d{3}</bind>
    </vmname>
  </work>
</document>
```

record

record faylı yazar, istəkdən asılı olaraq daxil edilməni bir yərə toplayır və faylı geriyə mənsəb URL-ə ötürür. Aşağıdakı atributlara sahibdir:

- **file:** Yazılma üçün faylin ünvani
- **name:** Nəticəni saxlamaq üçün **Param** adı (Fayl yükləməsinin multihissəli forması olacaq)
- **error-file:** Yalnız daxil edilmədə oxudulacaq olan error faylı
- **beep-file:** Mesajın yazılmasını indicator(voicemail beep-dir) olaraq təyin edən oxudulacaq fayldır
- **digit-timeout:** Fayl oxudulmasından sonra rəqəmlər üçün vaxt bitməsinin gözlənilməsi (Daxil edilmiş birləşim mövcud olarsa)
- **limit:** Yazılacaq saniyələrin yuxarı limit sayı
- **terminators:** Yiğilmiş rəqəmlərin işə salınması və durmadan dayandırılmasının istifadə edilməsi üçün açarlar

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <record action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      name="playback_user_input"
      error-file="ivr/ivr-error.wav"
      beep-file="tone_stream://${beep}"
      file="12345.wav"
      digit-timeout="5"
      limit="60"
      terminators="#">
      <bind strip="#">~\d{3}</bind>
    </record>
  </work>
</document>
```

record işi **record** Dialplan programları üçün analogdur.

pause

pause spesifik ayrılmış vaxt üçün gözləyir. Aşağıdakı atributlara sahibdir:

- **milliseconds**: Ara vermek üçün millisaniyelərdə olan rəqəm
- **name**: Nəticəni saxlamaq üçün **param** adı
- **error-file**: Yalnız daxil edilmədə oxunulacaq error faylı
- **digit-timeout**: Fayl oxudulmasından sonra rəqəmlər üçün vaxt bitməsinin gözlənilməsi (Daxil edilmiş birləşim mövcud olarsa)
- **input-timeout**: multi-digit daxil edilmədə çoxlu rəqəmlər üçün vaxt bitməsinin gözlənilməsi
- **loops**: Daxil edilmiş birləşmə mövcud olduqda, oxudulacaq olan faylin maksimal sayı
- **terminators**: Yiğilmiş rəqəmlərin işe salınması və durmadan dayandırılmasının istifadə edilməsi üçün açarlar

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <pause action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      name="pause_user_input"
      error-file="ivr/it_was_that_bug.wav"
      digit-timeout="5"
      milliseconds="15000"
      terminators="#">
      <bind strip="#">~\d{3}</bind>
    </pause>
  </work>
</document>
```

speak

speak mətni zəng edənə **TTS (Text-to-Speech)** motoru istifadə edərək oxuyur, istəkdən asılı olaraq daxil edilməni toplayır bir yerə. Aşağıdakı atributlara sahibdir:

- **text**: Zəng edənə oxudulacaq mətn
- **name**: Nəticəni saxlamaq üçün Param adı
- **error-file**: Yalnız daxil edilmənin oxudulması üçün səhv faylı
- **digit-timeout**: Fayl oxudulmasından sonra rəqəmlər üçün vaxt bitməsinin gözlənilməsi (Daxil edilmiş birləşim mövcud olarsa)
- **input-timeout**: multi-digit daxil edilmədə çoxlu rəqəmlər üçün vaxt bitməsinin gözlənilməsi
- **loops**: Daxil edilmiş birləşmə mövcud olduqda, oxudulacaq olan faylin maksimal sayı
- **engine**: İstifadə üçün Text-to-Speech (TTS) motoru
- **voice**: İstifadə üçün Text-to-Speech (TTS) voice
- **terminators**: Yiğilmiş rəqəmlərin işe salınması və durmadan dayandırılmasının istifadə edilməsi üçün açarlar

Məsələn:

```
<document type="text/freeswitch-httapi">
```

```

<work>
  <speak action="http://localhost/newurl.php"
    temp-action="http://localhost/newtempurl.php"
    name="speak_user_input"
    error-file="ivr/ivr-error.wav"
    digit-timeout="5"
    engine="flite"
    voice="slt"
    text="Hello from flite text to speech engine"
    terminators="#">
    <bind strip="#">~\d{3}</bind>
  </speak>
</work>
</document>

```

speak işi **speak** Dialplan programları üçün analoqdur.

say

İnsan danışığını modelləşdirmək üçün FreeSWITCH **say** motorunu istifadə edin ki, səsləri iterasiya edəsiniz. Aşağıdakı atributlara sahibdir:

- **text**: Səsləndirmək üçün mətn, danışiq, yazım və s.
- **name**: Param adı hansında ki, nəticə yadda saxlanılacaq
- **error-file**: Yalnız daxil edilmədə oxunulacaq olan səhv faylı
- **digit-timeout**: Fayl oxudulmasından sonra rəqəmlər üçün vaxt bitməsinin gözlənilməsi (Daxil edilmiş birləşim mövcud olarsa)
- **input-timeout**: multi-digit daxil edilmədə çoxlu rəqəmlər üçün vaxt bitməsinin gözlənilməsi
- **loops**: Daxil edilmiş birləşmə mövcud olduqda, oxudulacaq olan faylin maksimal sayı
- **language**: Səsləndirilecek dil
- **type**: Tip (say interfeys parametri)
- **method**: Metod (say interfeys parametri)
- **gender**: Cins (say parametri)
- **terminators**: Yiğilmiş rəqəmlərin işə salınması və durmadan dayandırılmasının istifadə edilməsi üçün açarlar

Məsələn:

```

<document type="text/freeswitch-httapi">
  <work>
    <say action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      name="say_user_input"
      error-file="ivr/ivr-error.wav"
      digit-timeout="5"
      language="en"
      type="name_spelled"
      method="pronounced"
      text="This is what the caller will hear"
      terminators="#">
      <bind strip="#">~\d{3}</bind>
    </say>
  </work>
</document>

```

```
</work>
</document>
```

say işi Dialplan **say** programı üçün analogdur. say məlumatlarına 5-ci başlıqda XML Dialplanın başa düşülməsində Vacib Dialplan proqramlarında baxın.

execute

execute FreeSWITCH-in dialplan programını yerinə yetirir. Aşağıdakı atributları mövcuddur:

- **application**: İşə salınacaq Dialplan programı
- **data**: Program datası üçün alternativ mənbə
- ***DATA***: Program datası

Məsələn:

```
<document type="text/freeswitch-httplib">
  <work>
    <execute action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      application="log"
      data="INFO this is an info log message"/>
  </work>
</document>
```

sms

sms mesajları yollayır. Aşağıdakı atributlara sahibdir:

- **to**: Mənseb rəqəmi
- ***DATA***: Mesajın datası

Məsələn:

```
<document type="text/freeswitch-httplib">
  <work>
    <sms action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      to="sip:1007@192.168.1.101">Message text here</sms>
  </work>
</document>
```

Qeyd: Bu **mod_sms** tələb edir hansı ki, kompilyasiya edilməli və yüklənməlidir. Ötraflı məlumat üçün

https://freeswitch.org/confluence/display/FREESWITCH/mod_sms linkinə müraciət edə bilərsiniz.

dial

dial çıxış zəngini yerləşdirir ya da yönləndirir. Aşağıdakı atributlara sahibdir:

- **context**: Dialplan konteksti
- **Dialplan**: Dialplan tipi (Adətən XML)
- **caller-id-name**: Zəng edənin ID adı
- **caller-id-number**: Zəng edənin ID rəqəmi

- ***DATA***: Zəng ediləcək rəqəm ya da originet ediləcək sətir

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <dial action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      caller-id-name="HTTAPI Test"
      caller-id-number="19193869900"
      context="default"
      Dialplan="XML">
      sip:2019@10.1.1.12
    </dial>
  </work>
</document>
```

dial işi zəngi Dialplan üzərindən yerləşdirəcək və əgər yeni zəng ayağı yaradılmışsa, HTTAPI tərəfindən idarə edilən yeni zəng ona qoşulmuş olacaq.

recordCall

recordCall rəngin yazılımasını inisializasiya edir. Zəng bitdikdən sonra fayl **post** ediləcək. Aşağıdakı atributlara sahibdir:

- **limit**: Saniyələrlə olan vaxtin bitməsi.
- **name**: Əgər bu http:// ilə başlayırsa, o halda bu URL-i təyin etməlidir harda ki, FreeSWITCH fayla **PUT** edəcək. Sizin web server PUT müraciəti qayda ilə emal edilməsi üçün quraşdırılmalıdır. Əgər buraxılıbsa, FreeSWITCH müvəqqəti qovluğa yazacaq.

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <recordCall action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      name="http://localhost/newfile.wav"
      limit="60"/>
  </work>
</document>
```

recordCall işi record Dialplan programının alternatividir.

conference

conference konfrans zəngini işə salır. Aşağıdakı atributlara sahibdir.

- **profile**: İstifadə üçün konfrans profaylı.
- ***DATA***: Konfrans adıdır hansında ki, zəng yerləşdiriləcək.

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <conference action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
```

```

        profile="my_new_profile">
        My_Conference
    </conference>
</work>
</document>
```

conference işi **conference** Dialplan programı üçün analogdur.

hangup

hangup zəngi dayandırır. Aşağıdakı attributlara sahibdir:

- **cause:** Dayandırma səbəbini yollayır

Məsələn:

```

<document type="text/freeswitch-httapi">
    <work>
        <hangup action="http://localhost/newurl.php"
            temp-action="http://localhost/newtempurl.php"
            cause="NORMAL_CLEARING"/>
    </work>
</document>
```

hangup işi hangup Dialplan programının analogudur.

break

httapi programından break çıxır və Dialplan-da davam edir.

```

<document type="text/freeswitch-httapi">
    <work>
        <break/>
    </work>
</document>
```

log

log jurnal sətirlərini **fs_cli** konsola və jurnal faylinə yazır.

- **level:** İstifadə ediləcək jurnal səviyyəsi
- **clean:** Əgər **true** mənasını təyin edəriksə onda, jurnal sətiri jurnal prefiksini çap etməyəcək.

Məsələn:

```

<document type="text/freeswitch-httapi">
    <work>
        <log action="http://localhost/newurl.php"
            temp-action="http://localhost/newtempurl.php"
            level="info">this is a log message with a prefix</log>
        <log level="warning"
            clean="1">and this is one without</log>
    </work>
</document>
```

log işi Dialplan-ın **log** programının analogudur. Qeyd edin ki, **log** Dialplan programının **clean** opsiyası mövcud deyil hansı ki, httapi log işi yerinə yetirir.

continue

continue spesifik olmayan işləri görür və davam edir(bu **no-op** işidir). Əgər siz **getVar** nəticələrinə ya da oxşarına əsaslanan fərqli action URL-lərinə edilən müraciətləri istifadə eləmək istəyirsinizsə:

```
<document type="text/freeswitch-httapi">
  <work>
    <continue action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"/>
  </work>
</document>
```

getVar

getVar kanal dəyişəni kontentini alır(yetkilərdən asılıdır). Aşağıdakı atributlara sahibdir:

- **permanent**: true mənasını təyin elədikdə, dəyişən bu zəng üçün HTTAPI ardıcıl müraciətlərin hamısı üçün göndərılır. Əks halda bu yalnız növbəti müraciətə göndərılır.
- **name**: Kanaldan oxumaq üçün dəyişən adı(Misal üçün **caller_id_name**)

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <getVariable name="caller_id_name"
      action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      permanent="1"/>
  </work>
</document>
```

voicemail

voicemail heç bir **"execute"** yetkisi tələb etmədən, voicemail Dialplan programını çağırır. Aşağıdakı atributlara sahibdir:

- **check**: Mənasını **true** təyin elədikdə, bu zəng edənə mesajların yoxlanılmasına izin verir. Bu mail yeşiyini istifadəçisidir. Əgər ötürülsə, zəng edənə səs mesajlarının yeşiyinin tərk edilməsi çıxacaq.
- **auth-only**: Yalnız qeydiyyatdan keçir və davam et. Sonda, bu rejim seçilir, uğurlu qeydiyyatdan sonra kanalda iki yeni dəyişən təyin ediləcək:
 - **variable_user_pin_authenticated** true təyin edilib
 - **variable_user_pin_authenticated_user** uğurlu qeydiyyatdan keçmiş istifadəçinin adıdır.
- **profile**: İstifadə ediləcək voicemail-in profil adı("default"-a ötürülür)
- **domain**: İstifadə ediləcək domain(Global **domain** dəyişəninə ötürülüb)
- **id**: İstifadə ediləcək ID(**id** üçün soruşulmasına ötürülüb)

Məsələn:

```
<document type="text/freeswitch-httapi">
  <work>
    <voicemail action="http://localhost/newurl.php"
      temp-action="http://localhost/newtempurl.php"
      auth-only="1"
      check="1"
      domain="192.168.1.101"
      id="1010"
      profile="default"/>
  </work>
</document>
```

Voicemail işi voicemail Dialplani üçün analoqdur.

mod_httapi quraşdırma faylı

mod_httapi quraşdırma faylı **conf/autoload_configs** qovluğunda tapılmışdır və adı **httapi.conf.xml**-dir. Bu **profiles** seksiyalarında olduğu kimi çoxlu **settings** parametrlərinə sahibdir. Nüsxə quraşdirmaların tərkibində **default** HTTAPI profile olur ya da siz özünüzə aid olan profillər yarada bilərsiniz.

profile tag-in daxilində sizə param məlumatların rəqəmləri bildirilir. Bu susmaya görə olan çoxlu iş addımlarını yetkilərin idarə edilməsi kimi işləri tənzimləyir və HTTP müraciətlərin istifadə edilməsi üçün susmaya görə olan URL.

Siz ola bilər ki, 9-cu başlıqda statik XML quraşdırmasının köçürülməsi bölümündə keçdiyimiz **mod_xml_curl** quraşdırmasında olan **gateway-url** parametrindən istifadə edəcəksiniz və mod_xml_curl-in web serverdən quraşdirmaların necə alınması üçün istifadə edəcəksiniz. **mod_httapi**-da eyni olan gateway-url parametri FreeSWITCH-də baza URL-i olaraq istifadə edilə

bilər hansı ki, ötürüləcək və götürüləcək məlumatların yerləşdiyi ünvanı təyin edəcək. **httapi** Dialplan programı kimi, imkanımızda var ki, istəyə əsaslanan URL seçə bilək. Növbəti iki misala baxsaq, diqqət yetirin ki, ilk olanda məlumatları proqrama ötürürük. Bu o deməkdir ki, o öncədən quraşdırılmış quraşdırma faylından **gateway-url** parametrini istifadə edəcək. İkinci misalda, biz httapi URL-ni ötürürük:

Misal bir:

```
<action application="httapi" />
```

Misal iki:

```
<action application="httapi"
  data="http://localhost/httapi/index2.php"/>
```

Aşağıdakı bir neçə misalda biz sizin web serverə necə POST parametrlərini ötürməyi öyrədəcəyik. Növbəti göstərilən misallar birlikdə funksionallıq baxımından "Misal iki" ilə identikdir.

Aşağıdakı misal URL adlandırılmış parametr kimi figurlu mötərizələrin daxilində ötürülür(bunlar {} simvollardır) hansı ki, FreeSWITCH web serverə ötürməzdən önce interpretasiya edir. url parametri spesifikdir və sözün əsl mənasında bildirir ki, "URL üçün bunu istifadə elə".

```
<action application="httapi"
  data="{url=http://localhost/httapi/index2.php}">
```

Bizim növbəti misalda, biz **url** parametrini öncədən ötürmüşük hansı ki, HTTP metod üçün yeni parametr ötürürük(Misal üçün GET, PUT, ya da POST).

```
<action application="httapi"
  data="{method=POST,url=http://localhost/httapi/index2.php}">
```

Yetkilər

httapi-da olan bütün idarəetmə ilə bəzi hallarda tələb olur ki, **permissions**(yetkilər)-i elə quraşdırıraq ki, təyin edilən dəyişənlər dəyişdirilə bilməsin ya da program və API-lar məqsədli olmadan yerinə yetirilməsin. **httapi.conf.xml** quraşdırma faylında siz **<permissions>** tag-nı **default** profile altında görəcəksiniz. Permissions tag-lə siz çoxlu işə sala biləcəyiniz yetkiləri tapacaqsınız hansı ki, bəzi məqamlarda yetkilərin çox kiçik səviyyəyədək idarə edilməsi üçün imkan yaradır.

set-params hüquqdur hansı ki, sizə eyni parametr üçün quraşdırmanı məhdudlaşdırmaq imkanı yaradır. Bu parametrlər httapi-dan {}-larla Dialplan-dan çağırıldığında çağırışın yaşam müddətinədək istifadə edilə biləcək.

set-vars kanal dəyişənləri üçün məhdudiyyətin təyin edilməsinə imkan yaradır. Bu hüquq növbəti addımda set-params təyin edir və sizə imkan yaradır ki, izin vermək istədiyiniz dəyişənləri təyin edəsiniz(**acl.conf.xml** faylında olan hüquqların emal edilməsinə oxşayır). Növbəti misal faylında susmaya görə olan siyasetlərə və yetki hüququ vermək imkanına nəzər yetirin:

```
<permission name="set-vars" value="true">
  <variable-list default="deny">
    <!-- Variables here may be changed -->
    <variable name="caller_id_name"/>
  </variable-list>
```

```
</permission>
```

Göstərilən kod siyahısı deyir ki, effektiv olaraq dəyişənlərin təyin edilməsi imkanı əgər, onlar spesifik olaraq **variable-list** siyahısında təyin edilərsə söndürülmüşdür.

Yəqin **<variable>** tag-da heç bir atributun olmasına artıq diqqət yetirdiniz hansı ki, öz növbəsində verilən dəyişən üçün imkana izin verilməsini deyir. Orda düzgün **type** atributu eyni olaraq ACL-lərin type atributu kimi işləyir ancaq, atribut tipi təyin edilməzsə susmaya görə olacaq mənə siyasetin əksi olmalıdır. Bu onu deyir ki, əgər sizin susmaya görə olan listiniz deny-dırsa, onda **type** yazı atributunun saxlanılması bu obyekt üçün yerinə yetirilməyə izin verəcək. Əksi doğru deməkdir:

```
<permission name="set-vars" value="true">
  <variable-list default="allow">
    <!-- Variables here may *not* be changed -->
    <variable name="caller_id_name"/>
  </variable-list>
</permission>
```

Qeyd: allow və deny verilənlərinizə diqqətli nəzər yetirin çünki, xəbəriniz olmadan çox vacib məlumatlara yetkini açıq saxlaya bilərsiniz.

Gördüyünüz kimi, **get-vars** hüquqları sizə zəngdən kanal dəyişənlərini oxumağa imkan yaradır. Bu hüququn **set-vars** opsiyalarının verdiyi eyni xırda modullu yetkiləri vardır. Orda dəyişənlərin təyin edilməsi kimi, dəyişənlərin götürülməsi üçün, eyni ACL tipli idarəetmə siyahıları mövcuddur. Bu mövcuddur ona görə ki, siz istədikdə öz programlarınızna bəzi dəyişənlərə təyinat imkanı verə bilərsiniz o halda ki, digərləri yalnız **read-only** rejimində qalır və bəzilərinə yetki tamamilə bağlanmış olur.

extended-data yetkisi həddən artıq məlumatı sizin web proqrama ötürəcək. Susmaya görə olan özünü aparış qaydası kanalın sıxılmış icmalini ötürəcək və sizə izin verəcək ki, HTTAPI əmrləri vasitəsilə tələb edilən məlumatları, ardıcıl geri qaydan zəngləri əldə edə biləsiniz. Əgər siz öz programınıza ötürülən müraciətin daxilində hər bir kanal dəyişəninin təyin edilməsini istəyirsinizsə, yalnız bu opsiyanı aktivləşdirməlisiniz.

Əgər siz **execute-apps** hüququnu təyin etsəniz onda sizin imkanınız olacaq ki, Dialplan programlarını httapi web proqramı vasitəsilə çağırı biləsiniz. Bu sizə imkan yaradacaq ki, öncə danışlığımız kimi programları **<execute>** tag-la istifadə edə biləsiniz. Bu hüququn ACL-də olduğu kimi eyni çeşidləmə hüquqları var. Həmçinin imkan var ki, bütün programlara yetki verəsiniz və ya onları biri digərinin ardınca işə sala ya da dayandırıa biləsiniz. Aşağıdakı misalda quraşdırma faylından götürdüyüümüz kimi göstərilir ki, biz programlara susmaya görə olan **deny** politikasının istifadəsinə imkan yaradırıq və **info** ilə **hangup** programlarına izin veririk:

```
<permission name="execute-apps" value="true">
  <application-list default="deny">
    <application name="info"/>
    <application name="hangup"/>
  </application-list>
</permission>
```

expand-vars hüquqları size öz programlarınızda dəyişənləri istifadə edilməsinə imkan yaradır hansı ki, normalda siz bunu Dialplan-dan edə bilərsiniz. **\${caller_id_number}** kimi dəyişənlər sətirin daxilində açıqlanacaq. **\${caller_id_number}** size zəng edən tərəfin nömrəsini verəcək. Bu həmçinin size öz web programlarınızda API əmrlərin istifadə edilməsində imkanlar yaradır. Aşağıdakı misala baxın:

```
 ${sofia_contact(1010@192.168.1.100)}
```

Bu sətir **sofia_contact** API əmrinə ötürünlən argument üçün yerinə yetirəcək və əlavə ediləcək yerdə nəticəsi olacaq. Bunu oxuduqda anlayacaqsınız ki, sizin programdan istənilən API əmrlərinə hüququ var və bu təhlükəsizlik baxımından doğru deyil. Narahat olmayın! ACL tipli idarəetmə siyahıları mövcuddur. Siz öz tələbinizdən asılı olaraq, əksər API əmrləri işə sala ya da dayandırıa bilərsiniz. Aşağıdakı XML koduna nəzər yetirin:

```
<permission name="expand-vars" value="true">
    <variable-list default="deny">
        <variable name="caller_id_name"/>
        <variable name="caller_id_number"/>
    </variable-list>
    <api-list default="deny">
        <api name="expr"/>
        <api name="lua"/>
        <api name="sofia_contact"/>
    </api-list>
</permission>
```

Aşağıdakı kod bütün digərlərindən başqa **caller_id_name** və **caller_id_number** kanal dəyişənləri üçün quraşdırılmalara izin verir. Bu bütün digərlərindən başqa, **expr_lua** və **sofia_contact** API əmrləri üçün yerinə yetirilməyə izin verir. Bu misal sizin FreeSWITCH sisteminizdə işləyən HTTAPI programları vasitəsilə bir programçı olaraq öz programınızda ya da sistem inzibatçısı olaraq işlək sisteminizdə kiçik modullu idarəetmə üsullarını göstərdi.

dial hüququ size imkan verir ki, öz programınızdan nömrəyə yığa bili və Dialplana düşüb uyğun olaraq yönləndirə bilsiz.

dial-set-Dialplan və **dial-set-context** hüquqları size imkan verir ki, Dialplan-ı dəyişəsiniz və əgər mənimsədilə biləndirsə Dialplan konteksti nömrəyə yığmaq üçün istifadə edəcək.

dial-set-cid-name və **dial-set-cid-number** hüquqları size zəng elədikdə imkan yaradır ki, zəng edənin ID adını və zəng edənin ID rəqəmini təyin edəsiniz.

dial-full-originate hüququ size tam endpoint/profile/number sintaksisi istifadə edilməsinə imkan yaradır(Məsələn:
sofia/internal/1010@192.168.1.100).

dial-set-context, **dial-set-Dialplan**, **dial-set-cid-name**,
dial-set-cid-number ya **dodial-full-originate** birinin istifadə edilməsi **dial** yetkisini işə salacaq hətta, quraşdırma faylında hardasa false təyin edilərsə belə.

conference hüququ sizə imkan yaradacaq ki, konfranslara zəng eləməyə izin verəcək hətta, **conference-set-profile** hüququ sizə hər bir istifadə edilən müraciət üçün konfrans profaylinin dəyişdirilməsinə izin verəcək. Əgər **conference-set-profile** işə salınıbsa, hətta quraşdırma faylında hansısa bir yerdə **false** təyin edilərsə də belə konfrans işə salınacaq.

Qeyd edin ki, istənilən ACL tipli idarəetmə siyahıları adı **<permission>** tag-ı əlavə edilmə siyahısı istifadə etmədən ötürünlə bilər. Bu susmaya görə hər şeyə izin verəcək ona görə ki, siz siyahınızı **default="allow"** ilə yaratmışsınız. Aşağıdakı iki misallar eynilə bu qaydada işləyəcək.

Misal bir:

```
<permission name="set-vars" value="true"/>
```

Misal iki:

```
<permission name="set-vars" value="true">
    <variable-list default="allow">
        </variable-list>
    </permission>
```

Çıxış

Dəstəyin asılması eventində FreeSWITCH "**exiting**" parametrini ötürəcək ki, zəngin bitməsini sizə bildirsin, artıq hansısa açığınız istənilən sessiyaları ayıra bilər və izlədiyiniz hansısa hesabatı bitirə bilərsiniz. Hətta siz tamamilə bu müraciətə məhəl qoymaya bilərsiniz və FreeSWITCH çox rahat bərpa olunacaq(O sadə "**OK**" cavabını gözləyir).

Məlumatların ardıcıl müraciətlərdə saxlanması

Artıq yeni əldə etdiyiniz biliklərlə siz hansısa programın yazılması barədə düşünməyə başlamışınızsa, bir müraciətdən digərinə məlumatların hissələrinin saxlanılmasında hansı üsulun istifadə edilməsindən narahat olacaqsınız.

Aydındır ki, siz kanal dəyişənlərini təyin edə və götürə bilərsiniz ancaq, bu **set/get** əməliyyatlarında çoxlu resurs itkisinə səbəb olacaqdır. Əgər hansıa istifadəçinin daxil olub nələr istifadə edilməsi üçün bir program yazardınızsa(misal üçün onlayn bank sistemləri), siz bu məlumatları gələcək istifadə edilməsi üçün sessiyada saxlaya bilərdiniz. **Httapi-a** minnətdarlığımızı bildirək ki, bu imkanı bizə verir. Hər bir müraciət özünə **session_id** **POST** və **GET** parametrini daxil edəcək. Bu **session_id** **parametr** mənası sessiyani inisializasiya elemək statusuna sahib olmalıdır hansı ki, o seans identifikatoru kimi istifadə edilir. Bu tip idarəetmə səviyyəsi istənilən web programlaşdırma dilində mövcuddur. PHP-də bunun edilməsi aşağıdakı kodda göstərilir:

```
if ( array_key_exists( 'session_id', $_REQUEST ) ) {
    session_id( $_REQUEST['session_id'] );
}
session_start();
```

Siz sessiyanı başladığdan sonra, sizin sessiya dəyişəninə yetkiniz olacaq ya da obyekt informasiyanın ardıcıl müraciətlərin tərəfindən istifadə edilməsi üçün saxlanıla bilər.

Bəzi müraciətlərdən bəzi parametrlər çatışır

Əgər siz nüsxə quraşdırmasını istifadə edirsinizsə, hər bir HTTP müraciətdən bütün məlumatları əldə etməlisiniz. Qeyd edin əgər siz, nüsxə quraşdirmalarında dəyişikliklər edib, **extended-data** işə salırsınızsa onda, öz programınıza gedəcək müəyyən dataları itirməyə başlayacaqsınız. Bunun səbəbi isə müraciətdə olan susmaya görə olan metod **GET**-dir. **extended-data** ilə ötürülən bütün məlumat əksər hallarda **CGI** parametrlərdə göstərilən maksimal izin verilən data həddini aşa bilər hansı ki, müraciətin sonunda ötürülən data formasının müəyyən hissəsini kəsib ötürəcək. Problemin həll edilməsinin çoxlu üsulları mövcuddur. Əsas həmişəlik olan yol isə aşağıda göstərildiyi kimi, **httapi.conf.xml**-də **method** parametrinin təyin edilməsidir:

```
<param name="method" value="POST"/>
```

Yalnız, hər bir müraciətə əsasən metodun təyin edilməsi üçün yol vardır hansı ki, səbəbə əsaslanaraq istəkdən asılı olaraq POST müraciətlərə təyin eləmək olar. Siz metodu URL sətirində aşağıdakı göstərilən qaydaya uyğun olaraq təyin edə bilərsiniz:

```
<action application="httapi"
data="{url=http://localhost/httapi/index.php,method=POST}"/>
```

Daha da asan edək

Öncəki başlıqlarımızdan gördükümüz kimi, siz httapi-in gücünü və asanlığını anladıqca, siz FreeSWITCH-in idarə edilməsi üçün XML-də olan formatı istifadə eləmək istəməyəcəksiniz. Həmçinin bütün bu XML-lərin elə çap edilməsi qarşıdurmaya səbəb ola bilər. Biz daha çoxunu qəbul edə bilərik. Məhz buna görə də HTTAPI XML yazıldı ki, köməkçi kitabxanaların sayesində özünüzə uyğun dili seçib işlərinizi görə bilərsiniz. Bu tip xeyli kitabxanalar **freeswitch-contrib** reposunda **PHP** və Python üçün tapıla bilər.

Qeyd: FreeSWITCH

GIT (<https://freeswitch.org/confluence/display/FREESWITCH/Tips+For+Using+Git>) server çoxlu reposlardan ibarətdir və bunlardan biri **freeswitch-contrib**-dir. Bu repositoriya(Ya da "**repo**") istifadəçi əlavəli kod nüsxələrindən ibarətdir.

Aşağıda biz sizə IVR-larda istifadə ediləcək kiçik nüsxələri PHP vasitəsilə göstərəcəyik. Dövrümüzdə olan bütün IDE-lərlə siz httapi XML istifadə edərək özünüzə aid olan funksiya və metod adları, dəyişən adları istifadə edərək kitabxanalarla özünüzə aid olan zəng axınını idarə edə bilərsiniz.

HTTAPI-da demo IVR

İşə başlamazdan önce nəzərdə tutulur ki, sizin PHP faylları emal edəcək PHP web serveriniz artıq mövcuddur. Web serverin qurulması bu sənəddən kənar olan bir mövzudur və əgər sizdə hazır server yoxdursa, ilk olaraq onu etməlisiniz.

PHP kodları emal edə bilən web serveriniz hazır olduqdan sonra işə, siz PHTTAPI kitabxanaları **freeswitch-contrib** repo-dan endirə bilərsiniz. Bütün class-lar isimizin asanlaşdırılması üçün, bir PHP faylinin içində yazılmışdır. Faylı <https://freeswitch.org/fisheye/browse/freeswitch-contrib/intralanman/PHP/phttapi/phttapi.php> ünvanında **source** tab-ın altından əldə edin. Öncə danışlığımız kimi, sizə məlum olan web server ünvanında yadda saxlayın. Hətta siz scripti demo IVR-in endirilməsi və yüklənməsi üçün özünüzə lazım olan WEB server qovluğunda da yerləşdirə bilərsiniz. Növbəti başlıqda biz **demo-ivr.php** scripti haqqında danışacayıq.

1004_11_codes.php adlı scripti kod nüsxələrindən götürüb öz web serverinizin Public_HTML qovluğuna **demo-ivr.php** adında nüsxələyin(Bu faylı mütləq **phttapi.php** faylı olan qovluğun yanından qeyd eləmək lazımdır). Mətn redaktorunu açın və aşağıdakı sətirlər olan hissəyə diqqət yetirin ona görə ki, biz kodları ardıcıl olaraq analiz edəcəyik.

```
if ( array_key_exists( 'session_id', $_REQUEST ) ) {
    session_id( $_REQUEST['session_id'] );
}
session_start();
```

Bu kod bloku **session_id** istifadə edərək sessiyani işə salır hansı ki, bu başlığın əvvəlində müzakirə eləmişdiq.

```
if ( array_key_exists( 'exiting', $_REQUEST ) ) {
    session_destroy();
    header( 'Content-Type: text/plain' );
    print "OK";
    exit();
}
```

Əgər biz **exiting** parametrini görürükse, PHP sessiyani dağıdırıq və FreeSWITCH-ə deyirik ki, biz başa düşürük və ardınca skriptdən çıxırıq:

```
$demo = new phttapi();
```

Burda biz **httapi** obyektini yaradırıq. Bu obyekt (**\$demo**) bize imkan yaradır ki, iş addımlarını yerinə yetirək.

```
$opt = array_key_exists( 'main_menu_option', $_REQUEST ) ?
$_REQUEST['main_menu_option'] : '';
```

Bu adı if/then/else quruluşunda olan şərtidir hansı ki, **\$opt**-nin həmişə təyin edilməsini yoxlayır, hətta əgər **main_menu_option** boş olarsa da belə.

main_menu_option-u işə biz birazdan opsiyalara birləşdirəcəyik hansı ki, zəng edənin sıxlığı açarlarla doldurulmağa gedəcək.

```
if ( preg_match( '/^10[01][0-9]$/ ', $opt ) ) {
    $xfer = new phttapi_dial( $opt );
    $xfer->context( 'default' );
    $xfer->dialplan( 'XML' );
    $demo->add_action( $xfer );
```

```
    } else {
```

Bu kod bloku müntəzəm ifadələrdə olan opsiyaların tutuşdurulması şərtini sınadın keçirir. Əgər belədirsə, o halda biz yeni **phttapi_dial** obyekti (**\$xfer**) qururuq, mənsebi təyin edirik və sonra işi **\$demo** obyektinə təyin edirik. Əgər bu genişlənmə deyilsə, bu sınaqları hər bir açar söz opsiyaların sıxılması üçün switch şərtində kəsirik.

```
    case '1':
        $conf = new phttapi_dial( '9888' );
        $conf->caller_id_name( 'another book reader' );
        $conf->context( 'default' );
        $conf->dialplan( 'XML' );
        $demo->add_action( $conf );
        break;
```

Əgər 1 opsiyası sıxılarsa, onda biz **dial** opsiyasını yaradırıq hansı ki, bu başlığın əvvəlində danışdığımız dial tag-ına uyğun gəlir. tag-da olan hər bir atributun **phttapi_dial** klasında uyğun olan metodu var. Misal üçün **context** metodu **context** atributunu təyin edir, **Dialplan** metodu isə Dialplan **attribute** təyin edir və.s(1 opsiyası zəng edəni public FreeSWITCH konfrans serverinə yollayacaq).

2-ci haldan 5-ədək hamısı dial obyektləridir və eyni məntiqi struktura sahibdirlər hansı ki, hər bir opsiyada arzu edilən nəticəni əldə eləmək fərqli atributları təyin edirlər.

```
    case '6':
        if ( array_key_exists( 'sub_menu_option',
$_REQUEST ) && $_REQUEST['sub_menu_option'] == '*' ) {
            unset(
$_SESSION['first_sub_play_done'] );
            $demo->add_action( $c = new
phttapi_continue() );
            break;
        }
        $demo->start_variables();
        $demo->add_variable( 'main_menu_option', 6 );
        $demo->end_variables();

        $sub = new phttapi_playback();
        $sub->error_file( 'ivr/ivr-
that_was_an_invalid_entry.wav' );
        $sub->loops( 3 );
        $sub->digit_timeout( '15000' );

        if ( !array_key_exists(
'first_sub_play_done', $_SESSION ) ) {
            $_SESSION['first_sub_play_done'] =
TRUE;
            $sub->file(
'phrase:demo_ivr_sub_menu' );
        } else {
            $sub->file(
'phrase:demo_ivr_sub_menu_short' );
```

```
}
```

```
$star = new phttapi_action_binding( '*' );
$sub->add_binding( $star );
$sub->name( 'sub_menu_option' );

$demo->add_action( $sub );
break;
```

6-ci opsiya biraz hiyləgərdir və yəqin ki, oz faylında qırılacaq ona görə ki, bu texniki olaraq ayrıca IVR-dir. Buz bunu sizin üçün bir faylda əlavə etdik ki, qurmaq və sinaqdan keçirmək asan olsun(6-ci opsiya IVR alt menyunu göstərir).

```
case '9':
    $continue = new phttapi_continue();
    $demo->add_action( $continue );
    break;
```

Bu kod hissəsi ilə biz sadəcə **continue** edirik hansı ki, nəticəsinə opsiyaların təkrarlanması işini görür ona görə ki, orda heç bir birləşmə və üsul yoxdur ki, **main_menu_option** parametrini yollayaq.

```
default:
    $intro = new phttapi_playback();
    $intro->error_file( 'ivr/ivr-
that_was_an_invalid_entry.wav' );
    $intro->loops( 3 );
    $intro->digit_timeout( '2000' );
    $intro->input_timeout( '10000' );
    $intro->name( 'main_menu_option' );

    if ( !array_key_exists( 'first_play_done',
$_SESSION ) ) {
        $_SESSION['first_play_done'] = TRUE;
        $intro->file(
'phrase:demo_ivr_main_menu' );
    } else {
        $intro->file(
'phrase:demo_ivr_main_menu_short' );
    }
```

default halının işi faylin içini oxumaqdır. Tam olaraq **ivr** Dialplani anlamadır üçün, biz sadəcə session içinde bir məlumat saxlayacayıq hansı ki, bizə uzun girişin oxunub və ya oxunmaması haqqında məlumat verəcək(Uzun və qısa salamlayıcıları 6-ci başlıqda XML IVR-ların və Phrase Macros-larda IVR menyünün təyin edilməsi alt başlığında açıqlanır).

```
$b1    = new phttapi_action_binding( 1 );
$b2    = new phttapi_action_binding( 2 );
$b3    = new phttapi_action_binding( 3 );
$b4    = new phttapi_action_binding( 4 );
$b5    = new phttapi_action_binding( 5 );
$b6    = new phttapi_action_binding( 6 );
$b9    = new phttapi_action_binding( 9 );
```

```

$bext = new phttapi_action_binding(
'~10[01][0-9]' );
$bext->strip( '#' );

$intro->add_binding( $bext );

$intro->add_binding( $b1 );
$intro->add_binding( $b2 );
$intro->add_binding( $b3 );
$intro->add_binding( $b4 );
$intro->add_binding( $b5 );
$intro->add_binding( $b6 );
$intro->add_binding( $b9 );

$demo->add_action( $intro );

Siz görə bilərsiniz ki, biz hər bir seçilən rəqəm üçün birləşim obyektini
yaratdıq və sonra hər bir birləşim üçün playback işini əlavə elədik. Sonra
öncəki misallarımızda olduğu kimi, $demo obyektinə işi əlavə elədik.
Göründüyü kimi, biz bir birləşmə yaradıb bir müntəzəm ifadə ilə bütün
rəqəmlər üçün birləşməni təyin edə bilərik. Bunu eləməklə göstərdik ki, sizin
çoxlu birləşiminiz tək rəqəmlərlə və ya müntəzəm ifadələrlə ola bilər və hər
şey işləyəcək.

header( 'Content-Type: text/xml' );
print $demo->output();

```

Biz burda qayıdan cavabın tərkib tipini **text/xml** təyin elədik və qurduğumuz
obyektin çıxışını çap elədik. Əgər siz **content-type**-da **text/xml**-dən fərqli
istifadə etmək istəsəniz FreeSWITCH başa düşmək istəməyəcək. Ona görə də əmin
olun ki, seçdiyiniz dildə bu edilmir.

Notice

Bu başlığımızda yeni buraxılan bir funksionallıq **mod_httapi**-a baxdıq. Web browserin FreeSWITCH-lə kobinasiyası ilə artıq mümkündür ki, zəng idarə etməsini HTTAPI qeydiyyatı ilə istifadə edəsiniz. Ardınca biz PHP(**phttpapi.php**) kitabxanası haqqında danışdıq hansı ki, aralıq bir səviyyə verir. **mod_httapi** istifadə edilməsində təşkilatlar telefon programının yazılımasına öz programçılarının web programlaşdırma biliklərini artırıa bilərlər. Bundan başqa programcılar FreeSWITCH inzibatçı bilməli olduğu işləri öyrənməli deyillər. Onlar HTTAPI öyrənməli və həmin imkan üzərindən şəbəkə vasitəsilə nələr eləyə bilmələrini araşdırmalıdır.

Növbəti başlıqdə biz VoIP inzibatçı üçün çox vacib olan bir mövzu haqqında danışacayıq. Bu NAT-ın emal edilməsidir.

12-ci başlıq

NAT emal edilməsi

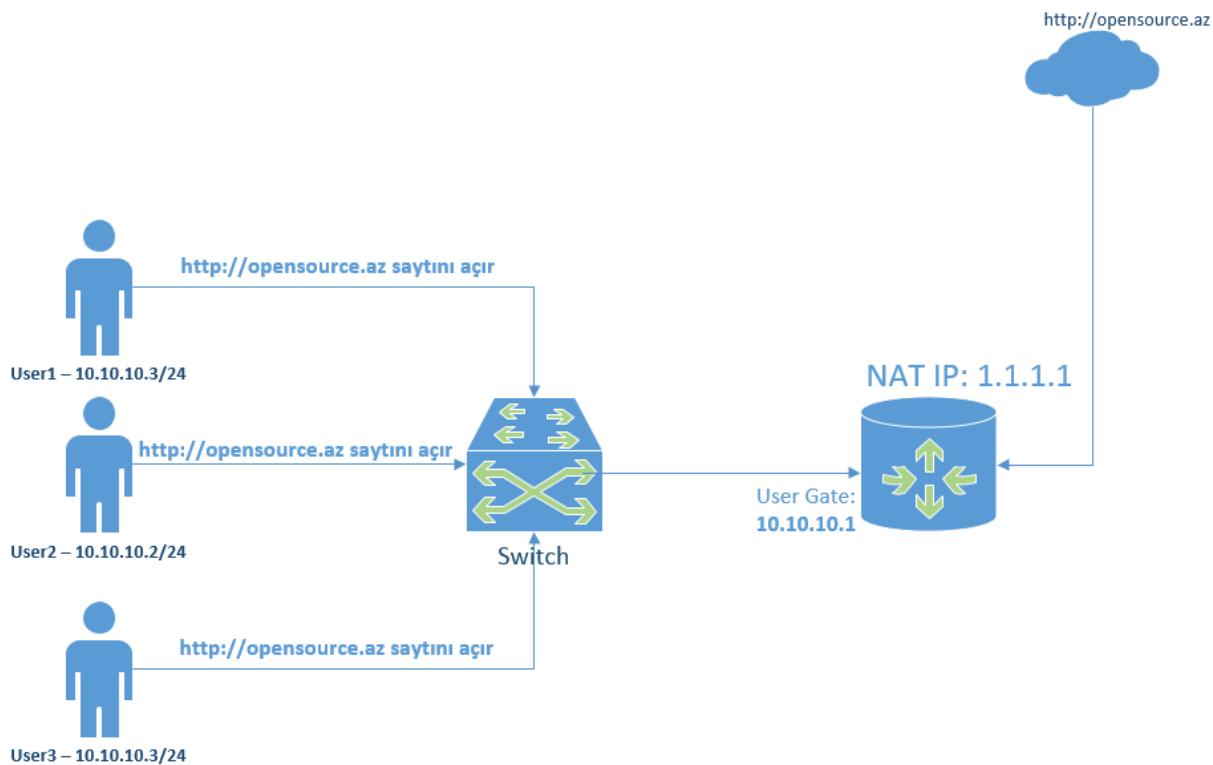
Kitabın əvvəlində biz nəsillik haqqında danışmışdıq. Bütün texnologiya nə qədər də dəyişsə, hər kəs istifadə elədiyi mühitin bir o qədər köhnə olmasını istəyir. Avtomobil lərdə hələ də sürət ölçəni iynə ilə simulyasiya ilə edirlər çünki, insanlar buna öyrəşiblər. Hətta sevimli saytimizda olan qrafika belə bizim köhnədən gördüyüümüz düymə və keçiricilərdən ibarət olur. Hətta cəmiyyət olaraq çalışırıq ki, "Qırılmayanadək əl dəyməyin işləsin" şūraları ilə yaşayaq. Eyni şūrlar hətta belə ən yeni şeylərə də mənimsədilir "Internetə çıxırıq". Dövrümüzdə VoIP-lə məşğul olan şəxslər NAT(Network Address Translation)-la quraşdırmaları ilk edənədək hansı problemlərlə qarşılaşacaqlarını heç belə ağıllarına da gətirə bilməzlər.

Bu başlıqda biz aşağıdakılari açıqlayacayıq:

- NAT-a qısa giriş və tarixi
- NAT-ın dörd tələsi
- NAT-ı aşmağa kömək edən FreeSWITCH quraşdırmaları
- NAT vəziyyətlərində FreeSWITCH-in daha da yaradıcı istifadə edilməsi

NAT-a qısa giriş

NAT haqqında anlayışı olmayan bir şəxsə NAT-ı başa salmaq başlanğıcda çətin görünə bilər. Yalnız bu əslində o dərəcədə də çətin deyil. Hər bir müəssisənin daxili İP aralığından seçdiyi bir şəbəkə quruluşu mövcuddur. Daxili aralıqdan seçilmiş şəbəkə də istifadəçilər Internetə çıxdıqda NAT edilən yeganə PUBLIC İP üzərindən istifadə edirlər. Təsəvvür edin ki, eyni anda 5 istifadəçi <http://opensource.az> saytını açmaq istəyirlər. Bu halda NAT İP ünvani özündən çıxıb opensource.az-a gedən paketin cavabını geri qayıtdıqda hansısa bir üsulla təyin etməlidir ki, hansı daxili istifadəçi üçün qaytarmaq lazımdır. Bu halda sizin köməyinizə portlar çatır. Hər bir istifadəçi DNS-dən opensource.az adı üçün İP ünvani təyin edir və NAT İP-si vasitəsilə həmin İP-yə keçid edir ancaq, hər bir istifadəçi öz çıxış paketinə müəyyən bir port təyin edir və uyğun port NAT İP ünvani ilə opensource.az saytına çıxış etdikdə də istifadə edilir. Beləliklə paket opensource.az saytından geri qayıtdıqda təyinatında olan portun sorğusunu edir və portu aydınlaşdırıb lazımlı olan istifadəciyə cavabı qaytarır. Aşağıdakı şəkildə hər şey daha aydın görünür:



Qeyd: NAT PAT-a qarşı

Texniki olaraq NAT ilə PAT arasında texniki fərq var. Ümumiyyətlə VoIP mühitində NAT termini tez-tez istifadə edilir. Bu başlıqla biz NAT və PAT arasında olan fərqləri açıqlayırıq.

Məsələlə NAT-a gəldikdə dediyimiz kimi, PUBLIC IP ünvallardan az istifadə eləmək üçün daxili LAN şəbəkədə olan bütün istifadəçilərin dünyaya bir IP ünvan kimi tanınmasını anlayırıq. Həmçinin istifadəçiləri NAT arxasında yerləşdirməklə onları müəyyən mənada hücumdan da qorumuş olursunuz.

Ümumiyyətlə təhlükəsizlik haqqında daha ətraflı siz 13-cü başlıqda **VoIP təhlükəsizlik** mövzusunda oxuyacaqsınız.

NAT təkmilləşdirilməsinin başa düşülməsi

Dövrün inkişafı ilə əlaqədar olaraq IP ünvanlara tələb artmağa başladı və IP ünvanının tükənməsinə gətirib çıxardı. Lakin bu problemin aradan qaldırılması üçün NAT və IPv6 yarandı. NAT öz sayəsində çoxlu sayıda daxili şəbəkə avadanlıqlarını dünyaya tək IP ilə yayımlaşığı sayəsində çox məhsurlaşdı. NAT demək olar ki, 1990-ci illərdə çox sürətlə inkişaf elədi və IP bitməsinin qarşısını aldı hətta, elə məhsurlaşdı ki, rahatçılıqdan artıq insanlar ondan əl çəkmək istəmirlər. Yalnız NAT probleminin həlli üçün IPv6 bizə demək olar ki, limiti olmayan bir sayıda IP ünvan verir. IPv6 spesifikasiyası 1998-ci ildə yayıldı və yaşayış-yavaş inkişaf eləməyə başladı. Yalnız hələ də IPv6 IPv4(1970-ci illərdə qəbul edilib)-ün kölgəsində qalaraq inkişaf eləməkdədir. Hətta biz IPv6-nı tam tətbiq etsəkdə belə, NAT-dan uzun müddət əl çəkə bilməyəcəyik.

Qeyd: Əgər FreeSWITCH-in IPv6 üzərindən SIP və RTP-nin dəstəkləməsi haqqında düşünməyə başlasanız, https://wiki.freeswitch.org/wiki/Main_Page səhifəsindən daha ətraflı məlumat əldə etmiş olacaqsınız.

IP telefon dünyasında əksəriyyətimiz üçün, NAT anlayışında bilməli olduğunuz çox vacib hissələr. Cənki, ələ bir zaman ola bilər ki, NAT-a görə xırda bir problemə görə bir neçə gün mənasız vaxt itirə bilərsiniz. Başlığımızın məqsədi sizə NAT-ın çıxardığı problemlərin FreeSWITCH üzərindən ANTI-NAT bacarıqları ilə qarşısının alınmasının açıqlanmasıdır. NAT və VoIP ilə həll etmək istədiyimiz əsas problemimiz ondan ibarətdir ki, NAT arxasında olan telefon(alət) birbaşa internet-də görünmür və sizin onunla əlaqə qurmanız problemə çevrilir. Digər bir problem ondan ibarətdir ki, SIP protokolu NAT arxasında olduqda qırıla bilər. Sözün düzünü desək FreeSWITCH qurucularının NAT-la əlaqəli olan ilkin pozisiyaları "Bizim problem deyil" idi! Yalnız sonradan istifadəçilərdən NAT arxasında qalan avadanlıqların istifadə edilə bilməməsi haqqında çoxlu sorğular yarandı. Və FreeSWITCH qurucuları NAT probleminin həlli üçün yeni təkmilləşmə metodları haqqında düşünməyə başladılar.

NAT-ın dörd tələsi

NAT-la hər kəsin bilməli olduğu dörd tələ var. Bu tələləri tamlıqla anladığdan sonra siz NAT-la qarşılaşdığınız istənilən problemi çox rahat həll etmiş olacaqsınız:

- Ağliniza belə gəlmədiyiniz məqamlarda NAT işləyə bilər. Internet sənədlü olmalı deyil.
- NAT-a qalib gəlmək üçün istənilən iki metod birlikdə istifadə edilərsə, biri digərini dayandıracaq.
- NAT-a qalib gəlmək üçün bəzi alətlər SIP ALG(Application Layer Gateway) istifadə edir.
- NAT-ın qarşısının alınmasının üsulları təyinatının yalnız məlumatları ilə vəziyyəti daha da pisləşdirə bilər.

Bu tələlərlə tanış olun çünki, başlığımızda onlara haqqında çox müzakirə edirik.

Gəlin onlar haqqında ətraflı danışaq:

- Hətta siz bilməsəniz də belə NAT orda ola bilər. Internet yanlış olmalı deyil.

Əgər siz cabel ya da telefon şirkətindən Internet istifadə edirsinizsə bilin ki, xidmət təçizatçıları əksər hallarda IPv4 qıtlığına görə öz daxili istifadəçilərini NAT üzərində dünyaya çıxarırlar. Hətta bu axın bir neçə dəfə təkrarlana bilər və siz buna təsir edə bilməzsəniz. VoIP istifadə edilən yerlərdə isə bu ciddi problemdir. Əksər VoIP protokolların NAT probleminin həll edilməsi üçün çox kiçik bacardığı bir imkanı var və əksər hallarda da bu üsul uğursuz olur. Beləliklə, evdə VoIP istifadə elədiyiniz hallarda qarşılaşdığınız NAT problemi təçizatçıdan qaynaqlana bilər.

- NAT probleminin qarşısının alınması üçün iki texnika birlikdə istifadə edilərsə, biri digərini dayandıracaq.

Bu o hallarda yaranır ki, sizin telefon(alət) NAT funksionallığına sahibdir. Siz onu işə salmısınız və eyni zamanda da FreeSWITCH-də NAT funksionallığını işə salmısınız. Bu zaman heç biri işləməyəcək və siz problemlə qarşılaşmış olacaqsınız hansı ki, tərəflərdən birində səs eşidilməyəcək.

• NAT problemin həll edilməsi üçün bəzi alətlər SIP ALG istifadə edirlər. Əslində SIP ALG metodu ilk iki metoddan pis işləyir. Həmçinin elə hallar olur ki, o sizin SIP trafiki gateway üzərində yönləndirmənin təyin edilməməsinə gətirib çıxara bilər. Çünkü bu üsul sizin router üstündə SIP paketin dəyişikliyini edib yollayırlar. Bu metod tam qarışığıliga səbəb ola bilər. Əksər hallarda elə əksinə SIP ALG-ın söndürülməsi NAT problemini həll etmiş olur.

- NAT probleminin həll edilməsi aldadıcı şəkildə vəziyyəti təyin edə bilər və faktiki olaraq hər şeyi daha da pis edə bilər.

Bu sizə öz mühitinizin başa düşülməsinə kömək edəcək və siz antiNat funksiyasının işə salınmasını faktiki olaraq təyin edə biləcəksiniz. Bəzi SIP alətləri daha gizli SIP aspektləri istifadə edirlər və həqiqətdə özlərinə paketdə şəbəkə ünvanları göstərirler. Beləliklə, NAT probleminin həll edilməsi daha da böyük problemə gətirib çıxara bilər. Siz Cisco alətlərdə olan NAT problemləri ilə ehtiyatlı olmalısınız ona görə ki, onlar NAT arxasında çox pis işləyirlər.

NAT-ı aşmağa kömək edən FreeSWITCH quraşdirmaları

Artıq NAT-da qarşılaşacağımız problemləri gördükdən sonra, gəlin NAT ilə qarşılaşacağımız fərqli tiplərin üstündən keçək. NAT problemin həll edilməsinin bir neçə üsulu var yalnız biz hal-hazırda diqqətimizi buna yönləndirmirik. Əksər hallarda elə olacaq ki, ya sizin PBX ya da telefon NAT arxasında olacaq və eynilə də əlaqəyə girəcək son nöqtə NAT arxasında olmayıcaq(və əksinə). Daha pis ikili NAT var hansı ki, hər iki tərəfə NAT arxasındadır. İkili NAT aşağıdakı quruluşa sahibdir:



Gəlin adı qaydada düşünək ki, NAT arxasında olan istifadəçi öz evindən PUBLİC-də olan FreeSWITCH server VoIP telefonu ilə qeydiyyatdan keçmək istəyir. Yaxşı xəbər odur ki, FreeSWITCH susmaya görə öz nüsxə quraşdirmalarında bunu açıqlayır. FreeSWITCH-in core-unda ACL(Access Control Lists) adlı bir bacarıq mövcuddur. ACL sizə imkan yaradır ki, öz mənsəbindən gələn ünvanın təyinatı üçün, şəbəkə ünvanlarını yaradaraq hüquqları idarə edə biləsiniz. Təyinat bərabərliklə verilir yəni, avadanlıq ya bizim təyinatımıza tutuşdurulub ya da ki, yox. Belə olanda siz kənar avadanlıqların qeydiyyatını IP ünvanla aparıb ya izin verə bilərsiniz ya da ki, tamamilə bağlaya bilərsiniz. Bu halda da biz istifadəçinin NAT arxasında olmasını ACL vasitəsilə təyin edib etməməyimizi ordan qərar verəcəyik.

NAT arxasında olan alətin RFC-1918-də qeyd edilmiş daxili IP aralığında olan bir IP-si olur. Bu IP ünvanlar istənilən halda NAT arxasında istifadə edilə bilər və heç bir zaman dünyada görünə bilməz. Bunlar 192.168.0.0/16, 172.16.0.0/12 və 10.0.0.0/8 aralığında olan IP ünvanlardır.

Qeyd: Haqqında daha ətraflı oxumaq üçün işə https://en.wikipedia.org/wiki/Private_network ünvanına müraciət edə bilərsiniz.

Bu o deməkdir ki, NAT arxasında daxili aralıqda olan istifadəçilər hamisin öz ROUTER-larınə qoşulub dünyaya bir IP ünvan kimi çıxırlar. Axın istifadəçilərində gəlir NAT serverə və dünyaya port xəritələnməsi prinsipinə əsaslanaraq çıxırlar. Trafik geri qayıtdıqda işə Router portID-ni təyin edərək axını geriyə istifadəciyə qaytarır. Lakin istifadəçi FreeSWITCH-ə qoşulduğda o həmişə gələn trafikin eyni mənbədən olduğunu görür və daxil olan zəngin ona qaytarmasına çətinlik çəkir. Bu halda da ACL köməyimizə çatır.

FreeSWITCH-in **apply-nat-acl** adında **mod_sofia** profaylları üçün quraşdırma parametri var. Bu parametr çoxlu sayda istifadə edilə bilər və ACL-in adını gözləyir. **mod_sofia** öz növbəsində **SIP REGISTER** ya da **INVITE** paketlərini aldıqda o qoşulma ünvanına baxır və ACL-də təyin edilmiş **contact** başlığının

ünvanladığı IP-ni yoxlayır. Əgər onlar uyğundursa, o qərar verir ki, alət NAT arxasındadır. NAT arxasında olan IP ünvanı təyin eləmək çox çətindir ama köməkçimiz var. Yadda saxlayın ki, NAT yalnız daxili IP aralığından gələn IP ünvanlara deyilir və kənar IP ünvanlar üçün bunu etməyin.

Yalnız nəzərə alın ki, NAT arxasından gələn istifadəçilərin hamısı həmişə yaxşı niyyətli olur. İşinin həlli üçün bu sizə həmişə köməyə çata bilməsə də, əksər hallarda yararlı olur. Lakin nəzərə alın ki, FreeSWITCH serverinizin özü də NAT arxasında işləyə bilər. Ancaq FreeSWITCH işə düşdükdə, onun üçün **nat.auto** adlı spesifik ACL yaradılır. Bu spesifik ACL RFC-1918-də olan IP aralığını özündə təşkil edir amma eynilə də FreeSWITCH olan serverin öz IP ünvanını da yoxlayır əgər, FreeSWITCH-də olan IP ünvan daxili aralıqda olanlardan hansısa birinə sahibdirse onu siyahıdan çıxarıır. Beləliklə də FreeSWITCH serverlə eyni şəbəkədə olan telefonların zəngini siz qəbul etməyəcəksiniz. Eynilə-də bu ACL vasitəsilə NAT arxasında olan telefon təyin edilə bilər.

FreeSWITCH susmaya görə **nat-auto(/usr/local/freeswitch/conf/sip_profiles/internal.xml** faylında)-da təyin edilmiş **applynat-acl**-lə gəlir və əksər NAT arxasına qalan tipik alətlərin problemlərini həll edə bilir.

Biz bu problemi necə həll elədik? Əksər hallarda NAT arxasında qeydiyyatdan keçən telefon təyin edildikdə biz onun IP və portunu daxili bazamızda əlçatılmaz LAN-ların yanında qeydə alırıq. Bu telefon ilə əlaqə qurmaq istədikdə isə öz verilənlər bazamızdan məsləhət alırıq və mesajın göndəriləcək olması olan ünvanın PUBLIC **IP:Port**-u təyin edirik. Yalnız SIP başlıqlarda daxili IP:Port mənaları görünəcək hansı ki, qarşı tərəfdə olan telefonumuz bu mənaları da gözləyir. Biz həmçinin telefona deyirik ki, tez-tez qeydiyyatdan keç çünkü, xəritilənmə açıq qalmalıdır(Əksər hallarda NAT routerlər translayasiya yolunu çox qısa bir zaman üçün açıq saxlayırlar). Bu metod 4-cü tələyə baxdıqda daha effektivdir çünkü, biz SIP paketin içində ALG kimi heç bir şey dəyişmirik. Bu o deməkdir ki, telefon görmək istədiyi hər şeyi görəcək və daha başqa məlumatə ehtiyacı olmayacaq.

Aşağıda FreeSWITCH **CLI(Command Line Interface)**-in çıxışı göstəriləcək. Müştəri NAT arxasında işləyir və PUBLIC ünvanda işləyən FreeSWITCH serverdə qeydiyyatdan keçmək istəyir. Nəzər yetirin ki, Contact sütunu **10.0.1.85** IP ünvanı istifadə edir hansı ki, daxili IP aralığına aiddir. Status göstərir ki, UDP-NAT təyin edilmişdir və **nat.auto** ACL-inə təşəkkürümüzü bildiririk. Qəbul Contact-in sonunda baş verir. Əlavə olan parametrlər **fs_nat** və **fs_path** telefon qeydiyyatında olan Contact ünvanına əlavə edilir ki, NAT-la qərar verməmizə köməkçi olsun. Aşağıdakına baxın:

```
fs_nat=yes
fs_path=sip%3A1006%40206.22.109.244%3A43425%3Brinstance%3Db67dbafc9baa9465%3B
transport%3Dudp.
```

fs_path sütunu SIP **URI(Uniform Resource Identifier)**-dir və geriyə NAT üzərindən çıkış edəcək. Bu şifrələnmiş URL-dir və URI-də olan spesifik simvollar real kontakt-la konfliktə girmir. Deşifrə edilmiş versiyası aşağıdakı kimidir:

```
sip:1006@206.22.109.244:43425;rinstance=b67dbafc9baa9465;transport=udp
```

Əgər biz telefona zəng eləmiş olsaq, INVITE paketini 206.22.109.244:43425 ünvanına yollayacayıq, biz bunu 10.0.1.85:5060 ünvnanında saxlayacayıq və bu onun işini bitirməsindən olacaq yeridir o vaxtadək ki, NAT translyasiya paketi yerini götürür və daxili telefon onu router-ə çatdırır. Siz bütün qeydiyyat informasiyasını sofia status profile internal reg əmri ilə görə bilərsiniz. Aşağıdakı misaldaki kimi:

```
freeswitch@fs> sofia status profile internal reg
```

Registrations:

```
=====
Call-ID: ZWU1Mjd1ZTI2MTg2MmVhNTc5NTk3MDY5YjFmOTVkMTU.
User: 1006@myhost.freeswitch.org
Contact: "TEST" <sip:1006@10.0.1.85:10118;rinstance=b67dbafc9baa94
65;transport=udp;fs_nat=yes;fs_path=sip%3A1006%40184.58.189.244%3A43425%3
Brinstance%3Db67dbafc9baa9465%3Btransport%3Dudp>
Agent: eyeBeam release 1104g stamp 54685
Status: Registered(UDP-NAT) (unknown) EXP(2012-12-09 10:18:07)
EXPSECS(88)
Host: myhost
IP: 206.22.109.244
Port: 43425
Auth-User: 1006
Auth-Realm: myhost.freeswitch.org
MWI-Account: 1006@fs.freeswitch.org
Total items returned: 1
=====
```

Nəzərə alın ki, **Contact**: başlığında həm **fs_nat** və həm də **fs_path** parametrləri olur. FreeSWITCH tərəfindən **1006** istifadəçisinə göndərilecək olan istənilən SIP trafiki URI istifadə edəcək hansı ki, **fs_path** parametrində təyin edilmişdir.

Media axınının yaradılması

Artıq SIP mesajları FreeSWITCH-dən telefona düzgün axın etdikdən sonra, media necə olacaq? Telefon zəngi eventlərlə varlı deyil hətta, bir-birini düzgün eşitməzsə də belə? Sizin yəqin ki, NAT-la audio(RTP trafiki) problemləriniz çox olmuşdur hansı ki, səsə gəldikdə ya bir tərəfli ya da ümumiyyətlə ikitərəfli olmur. Bunun üçün FreeSWITCH tərəfindən spesifik opsiya yaradılmışdır hansı ki, həmişə işə düşür və yalnız ən çıxılmaz hallarda əllə dayandırıla bilər. Bu funksiyaya **RTP audio-adjust** deyilir. Buna tələbimiz ona görə yaranır ki, NAT arxasında olan istifadəçi bizi zəng eləmək istədiqdə o audio-nun mənsebi üçün öz LAN ünvanını FreeSWITCH-ə əlçatılmaz kimi yayımlayacaq.

Biz təsəvvür edə bilərik ki, alət NAT arxasındadır və biz audio trafikini SIP mesajında yadda saxladığımız eyni ünvana yollamalıyıq. Ancaq bu üsul bütün NAT-larda kömək olmur ona görə ki, bəzi NAT-larda məhdudiyyətlər var və NAT arxasında qalan alətin port ünvanı onun paket yollaması müddətinə qədər görsənmir. Realliqda audio paketin alətə necə düzgün yollanılmasının çıxış yolu bizim anlayışımızda olmaya bilər. auto-adjust funksiyasının bacarığı sayəsində, bizim hələ də mübarizə şansımız var. Biz telefona düzgün bir ünvan verərək onun bizim üçün audio-nu yollamasını gözləyə bilərik o vaxtadək ki, o

bizə heç olmazsa bir paket yollamalıdır. Bir paket yollandıqdan sonra isə, biz onun içindən audio axının geriyə yollanılması ünvanını əldə edə bilərik. Bu 100% təminat verməsə də, bəzi çıxılmaz hallarda çox kömək edir. Yalnız təhlükəsizlik üçün bunu axının əvvəlində edirik çünki, istifadəçi audio axınını oğurlamaq istəyən şəxslər çıxa bilər.

Öncə dediyimiz kimi, bu **RTP audio-adjust** susmaya görə işə salınır və avtomatik olaraq da dayandırılır. Jurnallarınızda onun işləməsini tapmaq üçün siz aşağıdakı sətiri axtarmalısınız. Mesaj size təyin elədiyi original və yeni media mənbəsini göstərir. Bu jurnal sətiri **auto-adjust** işə düşən kimi, istənilən zəngin əvvəlindən çap edilir.

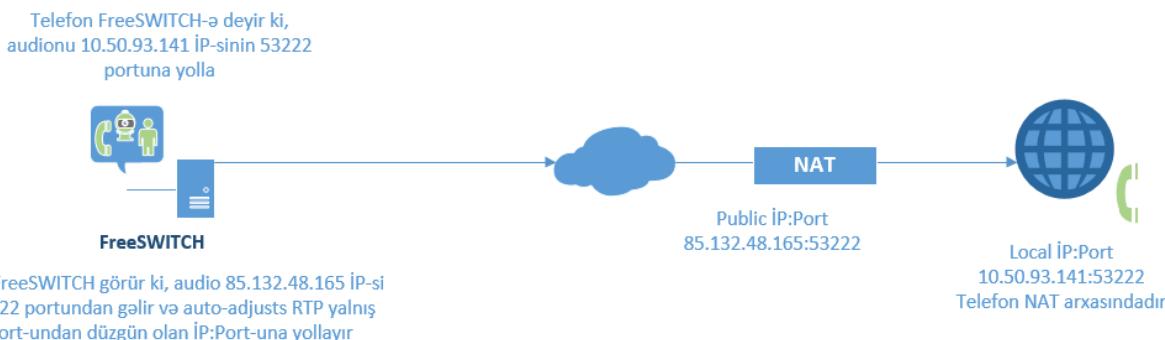
```
2015-11-26 07:20:26.965770 [INFO] switch_rtp.c:5857 Auto Changing port from 10.50.93.141:53222 to 85.132.48.165:53222
```

```
2015-11-26 07:20:27.885773 [INFO] switch_rtp.c:5857 Auto Changing port from 10.50.93.101:54206 to 85.132.48.165:54206
```

```
2015-11-26 08:29:46.195758 [DEBUG] switch_core_media.c:5177 AUDIO RTP [sofia/internal/1013@85.132.48.165:59002] 94.20.81.154 port 30198 -> 10.50.93.101 port 54206 codec: 0 ms: 20
```

```
2015-11-26 08:29:43.485908 [DEBUG] switch_core_media.c:5177 AUDIO RTP [sofia/internal/1010@frfs.opensource.az] 94.20.81.154 port 27548 -> 10.50.93.141 port 56340 codec: 0 ms: 20
```

Aşağıdakı diaqram daha da aydın başa salır:



Əgər sizdə həqiqətən-də NAT tələlərindən 4-cüsü baş veribsə, onda aşağıdakı sətiri öz SIP profilinizə əlavə eləməlisiniz:

```
<param name="disable-rtp-auto-adjust" value="true"/>
```

Digər hallarda siz bunu hər bir zəng üçün **rtp_auto_adjust=false** kanal dəyişəni təyin edərək dayandırı bilərsiniz.

Genişlənmiş opsiyalar və quraşdırılmalar

Artıq bunun necə işləməsini təyin elədikdən sonra, biz NAT-ın təyin edilməsinin digər üsullarını öyrənə bilərik. Bəzi hallarda ACL kifayət eləmir ona görə ki, təbəəl ALG paketi qarışdırılmışdır ya da trafik proxy üzərindən keçir ya da telefon düşünə bilər ki, o NAT emalı işini özü görməlidir və düzgün emal etmir.

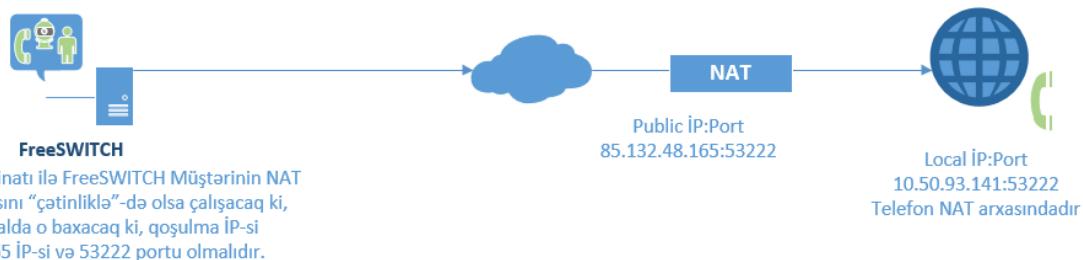
Bizim digər opsiyamız vardır və susmaya görə işə salınmır ona görə ki, 4-cü tip NAT tələsinin üzünə gülür və düşünür ki, hər şey NAT üzərindən gəlir. Bu parametr təhlükəlidir amma seçiminiz olmayan yerdə çox effektivdir.

Parametrin adı **aggressive-nat-detection**-dur və sizin SIP profaylinizda onun **true** təyin edilməsi bütün SIP trafikində işə salacaq. Susmaya görə o SIP paketlərinə baxır və əgər çoxlu başlıqlarda çoxlu IP ünvanlar görünürse o, məntiqi hesablama aparır ki, mənbə ünvanını təyin etsin. Burdan da görürük ki, bu ACL-də olan funksionallığıla eyni deyil ona görə ki, o həmişə mənbə ünvanından ibarət olmaya bilər və onu bazaya yazır.

```
<param name="aggressive-nat-detection" value="true"/>
```

Aşağıdakı şəkil situasiyanı göstərir:

Aqressiv NAT təyinatı olmadan
telefon əlaqə 10.50.93.141 İP ünvanı
və 53222 portu ilə əlaqəyə gira bilər



Aqressiv NAT təyinatı ilə FreeSWITCH Müştərinin NAT arxasında olmasını "çətinliklə"-də olsa çalışacaq ki,
tapsın. Bu halda o baxacaq ki, qoşulma İP-si
85.132.48.165 İP-si və 53222 portu olmalıdır.

FreeSWITCH ailəvi parametri mövcuddur hansı ki, "No Device Left Behind" NLDB kimi açıqlayıraq. Bu həmin haldır ki, həqiqətdə işləməyəndə biz client-ə özümüzü elə aparırıq ki, hər şey qaydasındadır və işləməlidir. NAT ilə mübarizə aparılmasında belə bir parametrin adı **force-rport**-dur. SIP-də **rport** atributun məqsədi müraciətə **rport** əlavə edərək NAT-a qarşı mübarizə aparmaqdır.

FreeSWITCH bu atributu gördükdə o, **rport=host:ip** atributunu telefon'a yollayacaq və o da öz növbəsində anlayacaq ki, NAT arxasındadır. Gülməli orasıdır ki, bəzi telefonlar **rport** görən kimi qərar verə bilirlər amma, heç bir zaman onu soruştururlar. **force-rport** parametri FreeSWITCH-i məcbur edir ki, onun danışdığı hər bir alət **rport** parametrinə sahib olmalıdır və beləliklə biz cavab veririk ki, onlar ardıcıl getmişlər və funksionallığı yaratmışlar hansı ki, başqa yolla əlçatılmaz olardı.

Bu parametr də əksər NDLB opsiyalarında olduğu kimi, susmaya görə aktiv olmur. Bu 4-cü tələ üçün boşluq açır ona görə ki, o alətləri zədələyə bilər. Siz bunu **true** təyin edə bilərsiniz ki, həmişə **rport** ala biləsiniz ya da **safe** təyin edə bilərsiniz ki, yalnız işləməsi tələb olan avadanlıqlar üçün bunu alaqlı. Siz həmçinin **client-only** ya da **server-only** təyin edə bilərsiniz zəngin istiqamətinə əsaslanaraq edəsiniz amma, hər bir uğurla siz bu opsiyalara heç bir zaman ehtiyac duymayacaqsınız.

Qeyd: Polycom telefonları klassi **ndlbg-force-rport** imkanı verir ki, **true** ya da **safe** təyin edəsiniz çünki, **rport**-u dəstəkləmər. Bütün hallarda, siz **safe** istifadə eləmək istəyirsiniz çünki, əksər telefonlar **rport**-u dəstəkləyir. Əgər bəzi səbəblər üçün sizin **ndlbg-force-rport=true** istifadə eləməyinizi ehtiyac varsa, onda bu parametrlərlə yeni SIP

profilə yaradın. Əmin olun ki, yalnız Polycom telefonları bu SIP profile-i istifadə edir.

FreeSWITCH client tərəfdə

Öncəki misallarımızda biz NAT haqqında danışdıqda nəzərə alırdıq ki, FreeSWITCH özü PUBLIC IP ilə işləyir və client tərəflər NAT arxasında işləyir. Gəlin təsəvvür edək ki, FreeSWITCH server özü NAT arxasında işləyir və hansısa PUBLIC-də olan SIP serverə qoşulur.

Məsləhətdir ki, tam başa düşmək üçün danışilan situasiyaların hamısını özünüz üçün real mühitdə yaradıb sınaqlardan keçirəsiniz. FreeSWITCH client tərəfdə olan iki NAT protokol dəstəkləyir və susmaya görə onlar NAT-PMP və UPnP adlanır. Bu protokolların hər ikisi fərqli metodlar istifadə edir amma məqsəd eynidir.

Hər iki metod şəbəkə protokolu istifadə edir ki, NAT router-i tapıb onunla əlaqəyə girsin. NAT-ın dayanmadan görsənməsini yaratmaq əvəzinə o Router-dən portun açılmasını istəyir və faktiki olaraq açılan portun detallarını araşdırır ona görə ki, FreeSWITCH digər serverlə danışdıqda o düzgün informasiyanı SIP paketində yerləşdirir. Əla! Ehtiyatlı olun ona görə ki, biz indi bütün tələlər üçün qapı açırıq.

Qeyd: NAT-PMP haqqında və UPnP haqqında əlavə məlumatı

https://en.wikipedia.org/wiki/NAT_Port_Mapping_Protocol və

https://en.wikipedia.org/wiki/Universal_Plug_and_Play linkindən əldə edə bilərsiniz.

Artıq biz özümüzün NAT arxasında olmasını gizlətdik və əks tərəf bizi tapa bilməyəcək. Qarşı tərəf isə aqressiv NAT təyinatı ya da ikili NAT tutulmasını istifadə edə bilər. Sizin sərt NAT router ya da Firewall-nız olsa yaxşı olar çünki, bu həm bağlı portların qarşısını ala bilər və həmdə NAT probleminin həlli ola bilər. Siz həmçinin FSAPI modulu istifadə edə bilərsiniz ki, öz programınızda xəritələnəcək portları eventlərlə **map** və ya **unmap** edəsiniz.

SIP profile-da **ext-sip-ip** və **ext-rtp-ip** adlı parametrlər vardır. Bu parametrlər istifadə edilir ki, serverimizin public IP ünvanlarına necə çatmağı bəyan eləsin. Susmaya görə olan quraşdılmalar bu parametlərin hər ikisini **auto-nat** ilə təyin edir. Bu NAT router ilə birlikdə istifadə edilir ki, işimiz düzgün yerinə yetirilsin. Bəzilərimizin NAT-PMP ya da UPnP dəstəkləyən Router-i olmaya bilər ya da daha pis biri var digəri yoxdu.

Biz FreeSWITCH daemon-u **-nonat** opsiyası ilə işə saldıqda bu funksionallığı söndürə bilərik. Söndürülmüş bu opsiya ilə birlikdə belə bizim hələ, əlimizdə yerinə yetirə biləcəyimiz bir neçə üsulumuz mövcuddur. Siz IP ünvanın external olmasından əmin olduqdan sonra onu istənilən sütuna əlavə edə bilərsiniz. Düzgün üsul olsun **autonat:x.x.x.x**(x.x.x.x olan yer sizin PUBLIC IP ünvanla dəyişdirilməlidir)-da təyin edilməsidir ona görə ki, o bildiyi PUBLIC IP ünvanı istifadə edəcək.

Digər bir üsul isə siz host:my.domain.com ya da
stun:stun.myhost.com(my.domain.com ya da stun.myhost.com sizin özünüzə aid

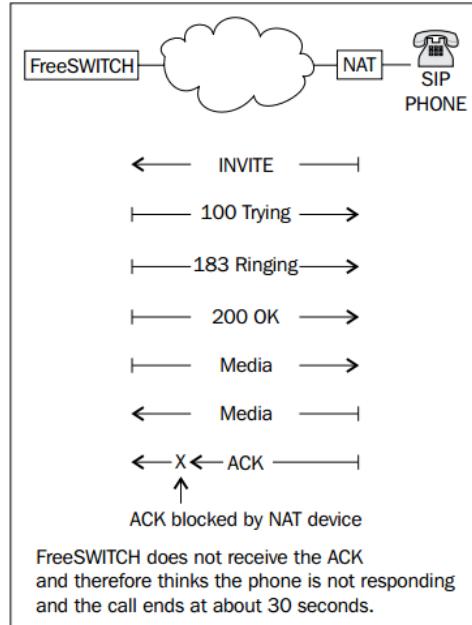
olan dynamic domain ya da STUN serverlər nəzərdə tutulur) təyin eləməklə, dinamik DNS istifadə edə bilərsiniz tələblərinizi qarşılıyayınız. Bunun çatışmamazlıqları var çünkü, o gec işləyə və ya tam dayana bilər. Amma ən yaxşı üsul özünüzün avadanlığınızın olmasıdır. Həmçinin NAT Router vasitəsilə spesifik portları FreeSWITCH ya da telefonlarınızın birbaşa xəritələyə bilərsiniz və bu halda **ext-sip-ip/ext-rtp-ip** sizin köməyinizi çatacaq.

NAT vəziyyətlərində FreeSWITCH-in daha da yaradıcı istifadə edilməsi

FreeSWITCH həmçinin alətlərin arasında NAT-ın ötürücüsü kimi istifadə edilə bilər. Siz daxili FreeSWITCH server qura bilərsiniz sonra daxildə olan bütün telefonları onun üzərində qeydə ala bilərsiniz ardınca da daxili FreeSWITCH serveri bütün daxili telefonların adından dünyada olan hansısa SIP təçizatçısında qeydiyyatdan keçirə bilərsiniz (Bu quruluş NAT üzərində bir deşik açır amma hamı xoşbəxt olur ☺). Həmçinin siz müəyyən bir PUBLIC IP ünvanda FreeSWITCH qura bilərsiniz və bütün daxili telefonları bu ünvanın üzərində eynilə daxili olan FreeSWITCH-ləri də bu public FS-də qeydiyyatdan keçirirsiniz. Onlar hətta NAT arxasında qalsalarda belə yenə də öz aralarında zəngi problemsiz edəcəklər.

Notice

Ümid edirik ki, ALG-la qarşılaşış hansısa bir çətinliyə düşdükdə bu başlıq sizin köməyinizi çatacaq. Əgər bu başlığı oxusunuz və yenə də problemlərinizin həllini tam dəqiq tapmasanız, FreeSWITCH cəmiyyətinə suallarınızı verib məlumat bölüşməsi edə bilərsiniz. Əlavə B FreeSWITCH onlayn cəmiyyətinə baxın. Bir diqqət yetirəcək məqamı deyək ki, əgər siz kimnənsə eşitsəniz ki, onların zəngləri başlanğıcdan 30 saniyə sonra dayanır onda, bu NAT problemidir. Bu aşağıdakı diaqramda göstərilir.



Bəzi hallarda FreeSWITCH özü problemi həll edə bilir və bu halda siz NAT alətini yığışdırmağınız.

Noticə

Bu başlıqda biz FreeSWITCH server üzərində NAT probleminin həll edilməsinin üsullarını öyrəndik. Həmçinin biz FreeSWITCH üzərində NAT problemin həlli üçün lazım olan opsiyaları da öyrəndik. Təhlükəsizliyə keçməzdən önce sizə NAT ilə qarşılaşacağınız əsas vacib məqamları vurğulayaq:

- NAT-da olan 4 tələlər haqqında oxuyun. Çünkü bu səhvlerin biri ilə qarşılaşdıqda çox böyük çətinliyə düşə bilərsiniz. Məsləhətdir ki, addımlarınızı diqqətlə dəfələrlə izləyəsiniz.
- NAT-in işləməsi üçün çalışın ki, az dəyişikliklər edəsiniz. NAT ilə dəyişiklikləri çox elədikdə hansısa bir səhvi buraxa bilərsiniz. Hətta elə ola bilər ki, bir tərəfi düzəlt dikdə digər tərəfi sıradan çıxarasınız.
- Əgər sizin ROUTER-ə girişiniz varsa, onda onu tələbdə olan NAT üçün quraşdırın. Ən yaxşı quraşdırma üsulu isə, susmaya görə olan minimal olanlardır.

Növbəti başlıqda fikirlərimizi NAT-dan çəkib VoIP üzərində olan təhlükəsizliyə yönəldirəcəyik.

13-cü başlıq

VoIP Təhlükəsizliyi

VoIP təhlükəsizliyi FreeSWITCH sisteminin qorunması üçün vacib olan başlıqdır. Qorunma strategiyaları hər iki önləyici və müdafiə edilən texnologiyaları daxil edir. FreeSWITCH-də olan proaktiv texnologiyalara SIP və RTP qoşulmaları üçün fərqli tip şifrələnmə daxildir hansı ki, telefon zənglərinə qulaq asmanın qarşısını alır. FreeSWITCH-də olan müdafiə texnologiyaları isə onun hansısa digər açıq qaynaqlı alətlərlə kombinasiyasıdır hansı ki, tanimayan mənbələrdən zərərli axını blok edir. FreeSWITCH imkanlarının ümumi mövcud olan açıq qaynaqlı VoIP alətlərlə kombinasiyası işlək istismarda olan mühitlərdə çox vacibdir.

Bu başlıq aşağıdakı 4 alt başlıqdan ibarətdir:

- Şəbəkə səviyyəsinin qorunması
- SIP siqnal səviyyəsinin qorunması
- Səsin qorunması
- Şifrələrin qorunması

Şəbəkə səviyyəsinin qorunması

Əksər pis niyyətli şəxslər açıq portları qırıb, VoIP şəbəkəsinə daxil olmaq istəyir. Onlar asan şifrələrdən tutmuş program səhvlerinə, exploitlərin tətbiq edilməsinə, quraşdırılmaların dəyişdirilməsinə və telefon sisteminin yönləndirilməsinədək hər şeyə baxırlar. Ümumi məqsəd oğurluq, səsin dinlənilməsi ya da hansısa məlumatların oğurlanmasıdır.

Şəbəkə sizin sistemə giriş nöqtəsi olduğu müddətədək, öz şəbəkənizin quruluşunu dəqiq öyrənməli və bəzi funksionallıqlar üzərində üstünlüyü əldə etməlisiniz ki, FreeSWITCH sistemini davamlı təhlükəsiz edəsiniz.

Interfeyslərin ayrılması və trafikin məhdudlaşdırılması

SIP texnologiyası elədir ki, ümumi halda internetdə açıq olmasını tələb edir. Əksər hallarda da pis niyyətli Hacker-lər UDP **5060**-ci port üçün IP aralığını skan edib cavab gözləyirlər. Cavab verən serveri tapdıqdan sonra isə onlar şifrələrin tapılması üçün **brute-force** edirlər ya da zəng eləməyə çalışırlar. Bəzi hallarda onlar həmçinin yalançı qeydiyyatlarla ya da digər paketlərlə serverin özünü də **flood** edirlər ki, sistemin davamlılığını aşağı salsınlar.

FreeSWITCH sisteminizin qorunmasının əsas yollarından biri onun qulaq asdıığı interfeysi firewall vasitəsilə digərlərindən ayırmadır.

Öncəki başlıqlardan öyrəndiyimiz kimi, FreeSWITCH sizə sisteminizdə olan fərqli IP və portlar üzərindən fərqli Sofia SIP interfeyslərini yaratmağa şərait yaradır hansı ki, siz SIP trafikini fərqli interfeyslərlə ala və ötürə bilərsiniz. Bu təhlükəsizliyin artırılması və dayanıqlığın artırılması üçün də istifadə edilir.

Təhlükəsizlik termini ilə desək, Sofia SIP profillərində default kontekst var hansı ki, onlara əsaslanaraq daxili zəng yönləndirilir. Bu kontekstlər haqlı olaraq məhdud Dialplan-a sahib olurlar. Əgər siz məhdud kontekstləri və Dialplanları münasib SIP profillə kombinasiya edirsizsə daha az şans verirsiniz ki, kimsə oğurluq trafiki sizin SIP sistemin üzərindən ötürə bilsin.

Əlavə olaraq, hər bir Sofia SIP profile-ında fərqli **Access Control List(ACL)** ola bilər. Bu halda siz, daha sərt məhdudiyyətləri PUBLIC IP ünvana və daha az məhdudiyyətləri daxili IP ünvana təyin edə bilərsiniz.

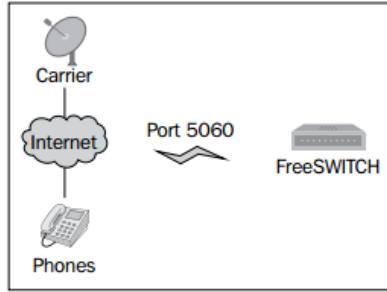
Dayanıqlıq və davamlılıq baxımından desək, FreeSWITCH haqqında olan fakt özünü göstərir ki, hər bir Sofia SIP interfeysi bir axındır. Bu o deməkdir ki, onların hər biri bir axın üçün bir IP və port istifadə edir ki, kiminsə sistemin çökdürməsinin qarşısı alınsın.

Adi qurulus misali

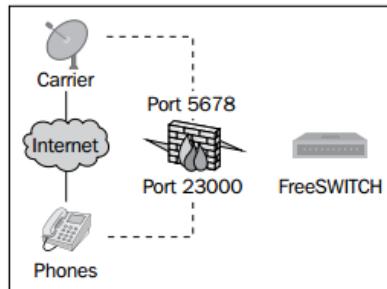
Həmişə istifadə edilən adi formada interfeysə sahib olduqda ya da sizin xidməti təcizatçınız və ya Internet Telephone Service Providers (ITSPs)-a telefonların birbaşa qoşulması olduqda, daha sərf edəni telefonların ITSP-ə qoşulmasıdır. Əksər pis niyyətli lər sizin 5060-ci SIP portunda müraciətlərin qəbul edilməsi və cavablandırılması imkanına sahib olduğunuzdan sonra

hərəkətə keçirlər. Bu nöqtədən sonra onlar fərqli üsullarla qeydiyyat mexanizmlərini sınaqdan keçirəcək ki, nəsə əldə etsinlər və əgər etməzlərsə həmin porta zərər yetirmək istəyəcəklər. Əgər siz bu IP-yə məhdudiyyət təyin eləsəniz, portu fərqli rəqəmlə dəyişsəniz və seçilmiş xidmət təçizatçılarına görə girişə limit təyin eləsəniz, siz avtomatik olaraq öz sisteminizə giriş hamı üçün bağlamış olacaqsınız (Hətta sizin sistemdə olan istifadəçi adı və şifrəni ələ keçirmiş olsa da belə).

Aşağıdakı diaqram hər kəsin FreeSWITCH-i susmaya görə necə qurduğunu göstərir:



Alternativ üsulla fərqli istifadə edilməyən portu təyin eləməklə, siz hücumunun işini çətinləşdirə bilərsiniz. Əlavə olaraq siz Firewall istifadə edə bilərsiniz ki, təçizatçılar üçün girişə bir port üçün məhdudiyyət və telefonlar üçün girişə digər port üçün məhdudiyyət təyin edəsiniz. Bu aşağıdakı diaqramda göstərilir:



Öncəki misalda olan **5678** portunda işləyən təçizatçının və **23000**-ci portunda işləyən telefonların qəbulunu yerinə yetirə bilmək üçün, siz quraşdırılmaları aşağıdakı kimi edə bilərsiniz (`/usr/local/freeswitch/conf/sip_profiles/pstn.xml` faylı yaradırıq və məzmununa əlavə edirik):

```

<profile name="incoming_from_pstn">
  <settings>
    <param name="auth-calls" value="true"/>
    <param name="apply-inbound-acl" value="my_carriers"/>
    <param name="context" value="inbound_call"/>
    <param name="sip-port" value="5678"/>
    <!-- Elave imkanlar olacaq -->
  </settings>
</profile>
  
```

Öncəki Sofia Profili üçün təçizatçılardan gələn trafik **5678**-ci portda qəbul edilir. Onlar bu porta düşən kimi, **my_carriers** adlı ACL tətbiq ediləcək ki,

yalnız təçizat üzvlərinə giriş yoxlanılsın. Əgər sizin **my_carriers** ACL-nizdə səhv olarsa, narahatçılığa səbəb yoxdur çünki, zəng edən tərəf **inbound_calls** kontesktinə düşəcək hansı ki, yalnız daxil olan zənglərə izin verir və çıxışı bağılayır. Bu tam sərt məhdudiyyətdir.

Əlavə olaraq, siz təçizatçılarınızın IP ünvanlarını təyin elədikdən sonra yalnız o IP ünvanların 5678-ci porta girişini Iptables, IPFW, PF və ya digər firewall vasitəsilə məhdudlaşdırı bilərsiniz. Bu təhlükəsizlik üçün istifadə edilən dayanıqlı metoddur.

O halda öz istifadəçiləriniz üçün də, aşağıdakı profile-ı yaratmalısınız ki, qoşula bilsinlər (**/usr/local/freeswitch/conf/sip_profiles/istifadeciler.xml** adlı fayl yaradın və tərkibinə aşağıdakı sətirləri əlavə edin):

```
<profile name="user_access">
  <settings>
    <param name="auth-calls" value="true"/>
    <param name="apply-inbound-acl" value="default_deny_list"/>
    <param name="context" value="customer_call"/>
    <param name="sip-port" value="23000"/>
    <!-- Elave imkanlar burda olacaq -->
  </settings>
</profile>
```

Bu Sofia profilində sizin müştərilər **23000**-ci porta müraciət etməlidirlər. Bu port qeydiyyat tələb edir və **default_deny_list** adlı ACL istifadə edir hansı ki, susmaya görə bütün trafiki bağlayır. Bütün trafiki məcbur edir ki, qeydiyyatdan keçsin. Bu o deməkdir ki, düzgün istifadəçi adı və şifrə təqdim edilməlidir ki, sistemə daxil olsun. İstifadəçilər düzgün məlumatı daxil elədikdən sonra isə, onlar zəngi yerinə yetirə bilmək üçün **customer_call** kontekstinin üstünə düşəcəklər.

Kompleks yüklənmənin göstərilməsi

Digər daha çox istifadə edilən şəbəkə nəzərdə tutur ki, sizin yönləndirilmə imkanı olan şəbəkəniz var və siz şəbəkəyə yetkiləri fiziki və ya VLAN-lara ayıra bilərsiniz.

Əksər hallarda ayrı fiziki şəbəkə kartında olan fərqli şəbəkələr istifadə edilir. Bu SIP portlarının fərqli interfeyslərdə xəritələnməsi üçün istifadə edilir. Siz həmçinin event socket və digər FreeSWITCH portlarını idarə edici IP ünvana xəritələyə bilərsiniz.

Bunu aşağıdakı qaydada Sofia SIP profile-ında edə bilərsiniz (**/usr/local/freeswitch/conf/sip_profiles/umumiprofile.xml** adlı fayl yaradıraq və bayaq yaratdığınıza faylları silib əvəzinə bunu istifadə edirik. Lakin burda göstərmək məqsədilə daxili IP istifadə edilməmişdir və **10.10.10.10** IP-si istifadə edilmişdir):

```
<profile name="incoming_from_pstn">
  <settings>
    <param name="auth-calls" value="true"/>
    <param name="apply-inbound-acl" value="my_carriers"/>
    <param name="context" value="inbound_call"/>
```

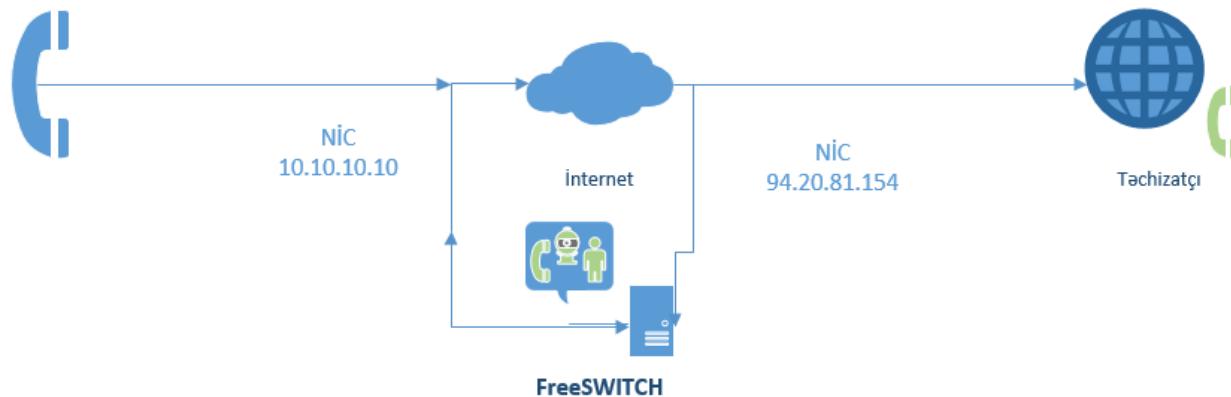
```

<param name="sip-port" value="5678"/>
<param name="sip-ip" value="94.20.81.154"/>
</settings>
</profile>
<profile name="user_access">
<settings>
    <param name="auth-calls" value="true"/>
    <param name="apply-inbound-acl" value="default_deny_list"/>
    <param name="context" value="customer_call"/>
    <param name="sip-port" value="23000"/>
    <param name="sip-ip" value="10.10.10.10"/>
</settings>
</profile>
  
```

Öncəki ssenaridə **sip-ip** quraşdırması ilə ehtiyatlı olun hansı ki, hər bir interfeys üçün fərqlidir. Şəbəkə kartlarının IP ünvanlarından biri **94.20.81.154** və digəri təsəvvür edək ki, daxili olanı 10.10.10.10-dur.

FreeSWITCH server yalnız bu IP ünvanlarda təyin edilən portlar üçün SIP profile yaradacaq. Bu size şərait yaradır ki, hər bir interfeys üçün fərqli firewall və şəbəkə linklərini istifadə edərək FreeSWITCH-i daha təhlükəsiz edəsiniz. Bu ssenaridə 10.10.10.10 daxili IP ünvandır, dünyaya yayılana bilməz və **94.20.81.154** dünya IP ünvanıdır və dünyaya yayılanır. Əgər sizin daxili şəbəkədən kənardə olan telefon istifadəçinizin yoxdur, **94.20.81.154** IP ünvanı yalnız telefon təçizatçısı tərəfindən qoşulma qəbul edir. Program telefonları üçün isə alternativ olaraq VPN quraşdırı bilərsiniz ki, daxili şəbəkəni görsünlər. Bu daha təhlükəsiz yol olacaq.

Öncəki ssenari aşağıdakı diaqramda göstərilə bilər:



Öncəki diaqramda telefonlar 10.10.10.10 şəbəkə kartında danışırular və həmçinin 94.20.81.154 IP ünvanında isə təçizatçı ilə əlaqələnirik.

VLAN-lar

VLAN-lar sizin telefonları daxili şəbəkənizdə data qoşulmalarından ayırmayı üçün əla üsuldur və əgər düzgün edilərsə, bu pis niyyətlilərin qarşısını alaraq zəngin keyfiyyətini də artırı bilər.

VLAN-ları adətən əlavə funksionallıq olmasından yalnız anlayırlar və çoxlu problemlərə də səbəb ola bilər. Bu sadəcə alətin IP ünvanını təyin eləmək və onlar eyni şəbəkədə olduqda daxil olmaq üçündür. Ordan da, əksər telefonların SIP istifadəçi adı və şifrələri götürmək mümkün olur. Misal üçün əgər siz Polycom alətinə daxil olsanız telefona aid olan məlumatların quraşdırılmalarını çıxarış edə bilər və onlara açıq mətnində baxa bilərsiniz. Telefon ayrılmış şəbəkə üzərində olduqda, bunu eləmək daha da istək yaradır.

Əlavə olaraq telefonun birbaşa idarə edilməsi, VLAN-lar yalnız məlumatların səs şəbəkəsinə göndərmək üçün istifadə edilən alətlərin qarşısını alır. Bura qeydiyyatdan keçməyən soft telefonlar da daxildir(yalançı BYE mesajları yollamaqla, zəngi məqsədli şəkildə dayandırmaq üçün istifadə edilir, genişlənmələri oğurlamaq üçün istifadə edilir).

Qeyd edək ki, VLAN-lar əksər şəbəkələrdə porta tag edilmiş kimi olur hansı ki, switch-də olan fiziki port virtual LAN-in hissəsi olur. Həmçinin program bazalı tag edilmə olur hansı ki, şəbəkə kartı(Ya da əməliyyat sistemi) hər bir paketi spesifik virtual LAN rəqəmi ilə IP başlığında işarələyir.

VLAN mövzusu bu kitabı aşır amma qeyd eləmək lazımdır ki, əksər desktop telefonlar VLAN-ı dəstəkləyir. İstər şəbəkə bazalı və ya istərsə də program təminatı olan VLAN-larla da şərhsiz işləyir.

Girişin təyin edilməsi

Sistemimizə giriş cəhdlerinin araşdırılması fərqli trafiki tipinə, müraciət formasına və edilən tələbin tipinə görə təyin edilə bilər. Yalnız təyin etmək sadəcə məlumatlandırmaq üçündür. Avtomatik bağlamaq üçün isə IPS qurmaq tələb edilir.

Qeydiyyat monitorinqi

Bəzi alətlər SIP-də istifadə edilən müraciətə cavab vermədən yalançı qeydiyyat cəhdlerini yollayırlar ki, bu da VoIP sistemini aşır. Bu alətlərdən biri isə DoS xasiyyətli kimi tanınan **SIPvicious** skaneridir. Bu tip alətlər yalançı müraciətlərlə sistemi yükleyir ki, real müraciətlərə cavab verməyə vaxtı qalmاسın və sistem gərgin vəziyyətdə işləyir. Əlavə olaraq bəzi şübhəli şəxslər qısamüddətli ölkədən kənar zənglərin edilməsi cəhdləri ilə təyin edilə bilər.

FreeSWITCH sisteminin daxilində olan qeydiyyat hesablarının yerinə yetirilməsi cəhdli olduqda, dərhal jurnallama imkanına sahibdir. Fail2ban adlı program təminatı isə bu jurnal sətirini analiz edib nə qədər tez-tez cəhd edilməsini təyin edə bilir. Əgər təyin edilmiş vaxt aralığı üçün təkrarlanmanın sayı çox olarsa, müraciət edən IP ünvan bağlanması üçün Firewall-a ötürülür. Adı qaydaya əsasən həmişə bir mənbə IP ünvandan gələn

müraciət təyin edilən vaxt aralığında məhdudiyyəti aşarsa, müraciət edən IP ünvanı bağlanır.

Bu məlumatın generasiya edilməsindən əmin olmaq üçün, FreeSWITCH yalnız qeydiyyat cəhdi alıqda siz öz SIP profillərinizi dəyişdirə və aşağıdakı sətiri əlavə edə bilərsiniz (`/usr/local/freeswitch/conf/sip_profiles/internal.xml` və ya `/usr/local/freeswitch/conf/sip_profiles/external.xml`):

```
<param name="log-auth-failures" value="true"/>
```

Qeydiyyat cəhdləri üçün generasiya ediləcək jurnal sətiri aşağıdakı kimi görünəcək:

```
[WARNING] SIP auth challenge (REGISTER) on sofia profile 'customer_access'
for [user_rdkj7h@2600hz.com] from ip 184.106.157.100
```

Bu sətir avtomatik olaraq sayıla bilər və mənbə IP **184.106.157.100** ünvanı bağlanma üçün ötürülə bilər.

Fail2Ban

Bu üçüncü tərəf program təminatıdır hansı ki, arxa fonda jurnalları oxuyur. Əgər jurnal faylında öncədən təyin edilmiş jurnal sətiri öncədən təyin edilən sayıda təkrarlanarsa, Fail2Ban öz işini yerinə yetirəcək. Təyin edilən aralıq müddəti üçün o qurula bilər ki, məktub yollasın ya da IP ünvanının bağlanması üçün IPTables-a əmr ötürülə bilər ki, həmin IP ünvanı bağlaşın.

Fail2Ban ümumilikdə bizim kitabdan kənar olan mövzudur lakin, onun FreeSWITCH-lə qurulmasını bu başlıqda göstəririk.

FreeBSD əməliyyat sistemi üzərində Fail2Ban qurulması

Bu bizə çoxlu hostların müəyyən bir seriya düzgün sayılmayan müraciətlərini bağlamağa kömək edəcək. Yəni bu program təminatı istənilən servisin jurnallarını oxumaq imkanına malikdir. Misal olaraq SSH-a gələn müraciətlərin sayı 3-ü aşarsa Fail2Ban edəcəyik. Bizim halda IPFW firewall-dan istifadə edəcəyik.

İlk önce Kerneli kompilyasiya edək və startup-a əlavə edək.

```
ee /sys/amd64/conf/GENERIC      # Faylin sonuna aşağıdakı sətirləri əlavə edirik.
# ipfw
options          IPFIREWALL
options          IPFIREWALL_VERBOSE
options          IPFIREWALL_VERBOSE_LIMIT=3
options          DUMMYNET
options          IPFIREWALL_FORWARD
options          IPFIREWALL_NAT
options          LIBALIAS

cd /usr/src/ ; make buildkernel      # Kerneli kompilyasiya edirik
cd /usr/src/ ; make installkernel    # Kompilyasiya edilmiş kerneli yükleyirik.

portsnap fetch extract update      # Bütün portları yeniləyirik.
reboot                # Yenidənyüklənmə edirik ki, Kernel işə düşsün.
```

```
ee /etc/rc.conf          # Startup faylina aşağıdakı sətirləri əlavə edirik
firewall_logging="YES"
firewall_enable="YES"
firewall_type="UNKNOWN"
firewall_script="/etc/ipfw.conf"
```

Bu IPFW faylinda biz **100** rəqəmli cədvəldə fail2ban-dan gələn IP ünvanları saxlayacaq ona görə ki, bu cədvəldə həmin IP-lər block edilir.

```
ee /etc/ipfw.conf # IPFW startup scriptinin tərkibi aşağıdakı kimi olacaq ona
ipfw add 10 deny ip from table'(100)' to any
ipfw add 65000 allow ip from any to any
```

/usr/local/freeswitch/conf/sip_profiles/ qovluğunda olan bütün SIP profillər üçün aşağıdakı sətiri uyğun olaraq edirik(Yəni **internal.xml** və **external.xml**):

```
<param name="log-auth-failures" value="true"/>
```

Fail2Ban-ı yükleyək.

```
cd /usr/ports/security/py-fail2ban      # Portuna daxil oluruq.
make install                            # yükləyirik
```

```
echo 'fail2ban_enable="YES"'           # Startup-a əlavə edirik.
```

```
# Startup Log rotate əlavə edirik. CLI-dan əmri daxil edirik.
echo "/var/log/fail2ban.log           600    7    200 *      JC" >>
/etc/newsyslog.conf
```

/usr/local/etc/fail2ban/jail.conf susmaya görə olan faylinda aşağıdakı məzmunu uyğun olaraq dəyişirik.

```
[DEFAULT]
ignoreip = 127.0.0.1/32
# BAN ediləcək host-un saniyələrlə olan vaxtı
bantime = 600
```

```
# Yoxlanış müddəti hansı ki, bu aralıqda hadisə təkrarlanı bilər.
# Belə hal olan kimi tutub bağlaşın
findtime = 900
```

```
# Maximum neçə dəfə qanun pozuntusu ola bilər.
maxretry = 3
```

```
# Jurnalları parçalamaq metodikası
backend = auto
```

FreeSWITCH üçün filter faylı fail2ban ilə birlikdə yükləndikdə gəlir və **/usr/local/etc/fail2ban/filter.d/freeswitch.conf** olaraq yerləşdirilir. Faylin tərkibi aşağıdakı kimidir və susmaya görə olaraq qalır:

```
[Definition]
```

```
failregex = ^\.\d+ \[WARNING\] sofia_reg\.c:\d+ SIP auth (failure|challenge)
 $\backslash((REGISTER|INVITE)\backslash) \text{ on } \text{sofia profile } \backslash[^']+ \backslash \text{ for } \backslash[.*\backslash] \text{ from } \text{ip } <\text{HOST}>\$$ 
```

```

^.\.\d+ \[WARNING\] sofia_reg\.c:\d+ Can't find user
\[ \d+@\d+\.\d+\.\d+\.\d+\] from <HOST>$

ignoreregex =


/usr/local/etc/fail2ban/jail.local faylı yaradırıq və tərkibinə aşağıdakı
sətirləri əlavə edirik:
[freeswitch]
# işə salırıq
enabled = true
# Bu portlar üçün analiz edilir
port = 5060,5061,5080,5081
# freeswitch adlı filterdən istifadə edilir hansı ki,
# /usr/local/etc/fail2ban/filter.d/freeswitch.conf faylından oxunur
filter = freeswitch

# FreeSWITCH-in mənbə kodlarında kompilyasiya edildiyi jurnal faylı ünvanı
# bu fayl veriləcək qərarlar üçün analiz edilir
logpath = /usr/local/freeswitch/log/freeswitch.log
# Əgər təkrar sətir 6 dəfə olarsa
maxretry = 6

# Mənimsemə üçün /usr/local/etc/fail2ban/action.d/bsd-ipfw.conf faylından
istifadə eləmək
# Kvadrat mötərizələrdə dəyişənin mənasını təyin edirik,
# Bizim halda göstəririk ki, tablearg=5060 olan 100 cədvəlinə əlavə elə
action = bsd-ipfw[table=100, tablearg=5060]

# Hükum edən IP ünvanının bağlanması müddəti
bantime = 259200

```

Həmçinin **/usr/local/etc/fail2ban/action.d/bsd-ipfw.conf** faylında aşağıdakı 2 sətiri uyğun formaya gətiririk.

```

actionban = ipfw table <table> add <ip> <tablearg>
actionunban = ipfw table <table> delete <ip>

/usr/local/etc/rc.d/fail2ban start      # İşə salaq.

```

Sınaq üçün həmin serverin IP ünvanına səhv şifrələrilə SIP qoşulma eləsəniz aşağıdakı jurnalı fail2ban-da göstərilməlidir.

```

tail -f /var/log/fail2ban.log
2015-11-27 10:24:48,454 fail2ban.jail      [627]: INFO    Jail 'freeswitch'
stopped
2015-11-27 10:24:48,461 fail2ban.server    [627]: INFO    Exiting Fail2ban
2015-11-27 10:25:32,013 fail2ban.server    [510]: INFO    Changed logging target to
/var/log/fail2ban.log for Fail2ban v0.9.3
2015-11-27 10:25:32,026 fail2ban.database   [510]: INFO    Connected to fail2ban persistent
database '/var/db/fail2ban/fail2ban.sqlite3'
2015-11-27 10:25:32,048 fail2ban.jail      [510]: INFO    Creating new jail
'freeswitch'

```

```

2015-11-27 10:25:32,083 fail2ban.jail      [510]: INFO    Jail 'freeswitch'
uses poller
2015-11-27 10:25:32,126 fail2ban.filter   [510]: INFO    Set jail log file
encoding to US-ASCII
2015-11-27 10:25:32,127 fail2ban.jail     [510]: INFO    Initiated 'polling'
backend
2015-11-27 10:25:32,152 fail2ban.filter   [510]: INFO    Added logfile =
/usr/local/freeswitch/log/freeswitch.log
2015-11-27 10:25:32,153 fail2ban.filter   [510]: INFO    Set maxRetry = 6
2015-11-27 10:25:32,155 fail2ban.filter   [510]: INFO    Set jail log file
encoding to US-ASCII
2015-11-27 10:25:32,156 fail2ban.actions  [510]: INFO    Set banTime =
259200
2015-11-27 10:25:32,158 fail2ban.filter   [510]: INFO    Set findtime = 600
2015-11-27 10:25:32,205 fail2ban.jail    [510]: INFO    Jail 'freeswitch'
started
2015-11-27 10:25:32,335 fail2ban.actions  [510]: NOTICE   [freeswitch] Ban
94.20.81.130

```

ipfw table 100 list - Block edilmiş IP ünvanlar üçün cədvəlimizə baxırıq
94.20.81.130/32 5060

Cədvəldən hansısa IP ünvanı silmək istəsəniz aşağıdakı əmrən istifadə edə bilərsiniz:

ipfw table 100 delete 94.20.81.130

Fail2Ban-ın özü ilə blok edilmiş IP ünvanlarının siyahısına baxırıq:

```

root@frfs:~ # fail2ban-client status freeswitch
Status for the jail: freeswitch
`- Filter
|  |- Currently failed: 1
|  |- Total failed:      14
|  `- File list:         /usr/local/freeswitch/log/freeswitch.log
`- Actions
  |- Currently banned: 1
  |- Total banned:     1
  `- Banned IP list:   94.20.81.130

```

Qeyd: Siz Fail2Ban-da jurnal faylinin oxunması sayında sətir limitini FreeSWITCH serverinizi hansı vaxt aralığında gələ biləcək maksimal istifadəçi sayına görə analiz edib qərar verməlisiniz. Əks halda həmin aralıqda real istifadəçi də müraciət eləsə bağlanmış olacaq.

Şifrələnmə

Səsin təhlükəsiz saxlanılması vacibdir. Xüsusən də, sizin qoşulmanız PSTN üzərindən keçirssə daha önemlidir çünki, Internet üzərindən dünyaya çıxış olur.

VoIP şifrələnməsi iki konsepsiyyaya əsaslanır - siqnalın şifrələnməsi və media(**audio/video**) qoşulmalarının şifrələnməsi. Standart şifrələnmə mexanizmləri kimi, VoIP şifrələnməsi də şifrələnmə kitabxanaları istifadə

edir və açar dəyişikliyini şifrə danışmalarını gedib qayıdan məlumatlara əlavə edir. VoIP-də istifadə edilən əsas şifrələnmə alqoritmləri Web üzərində işləyən SSL-ə və açarların dəyişilməsi isə SSH serverə qoşulmasına oxşayır. İstənilən dəyişmə tipində şifrələnmə alqoritminin əsas məqsədi odur ki, iki tərəf arasında olan əsas şifrələnmə açarını yalnız iki tərəfin özü bilməlidir hansı ki, bu şifrələnmə açarı telefon zəngində olan real datanın şifrələnməsi və deşifrələnməsində istifadə ediləcək.

Əksər insanlar seçim üçün TLS, SSL və SRTP arasında qalırlar və tam işləmələrini başa düşmürlər. Bu qoşulmaların tam müdafiəsi üçün, hər ikisi seçilməlidir yəni, signal səviyyəsi və audio səviyyəsi şifrələnməsi də seçilməlidir.

Aşağıdakı seksiyada biz hər bir şifrələnməni detallı açıqlayacaqıq.

SIP siqnal səviyyəsinin qorunması

Şifrələnmə üçün SIP siqnalları çox vacibdir. O sizin telefonunuzun yerinə yetirdiyi hər iki qeydiyyat məlumatlarına sahib olur, zəngləri qəbul edir, Caller ID adı, zəng edənin nömrəsi və qəbul edənin nömrəsinə susmaya görə sahib olur. Bunu **sniff** edib **spoof** eləmək çox asandır. Şifrələnmə çətin ola bilər. Əlavə olaraq siz **SRTP**(Secure RTP) istifadə edirsinizsə hansı ki, SIP siqnallaşması tərkibində açar olur və audionun təhlükəsiz olması üçün istifadə edilir. Kimsə bu açarı açıq şəkildə ələ keçirse asanlıqla media-nı deşifrə edilmiş şəklə gətirə bilər.

Şifrələnmə opsiyaları arasında seçimin edilməsi

FreeSWITCH-də çoxlu şifrələnmə opsiyaları mövcuddur. Siz signallaşma səviyyəsini (Bu SIP mesajlarıdır), medianı (Bu RTP axının içində olan audio-dur) ya da hər ikisini şifrələyə bilərsiniz. **Transport Layer Security (TLS)** v1 TCP üzərindən keçən bütün hər şeyi şifrələyir. Bunun zəif cəhəti ondan ibarətdir ki, TCP-yə görə gecikə bilər. RTP üçün ümumiyyətlə UDP çox üstün tutulandur və **TLSv1**-in istifadəsi biraz artıq trafikin istifadə edilməsinə gətirəcək. Sondan-sona qoşulmalar üçün **Z RTP** şifrələnməsi açarları dəyişmədən və sertifikatlar olmadan əla seçimdir amma, qurulması biraz ümumidir (Əsasən də bəzi müştərilər üçün). **Secure Sockets Layer (SSL) v2/3** isə SSL sertifikatları istifadə edərək SIP idarə edici kanalı TCP qoşulması üzərindən şifrələyir ancaq, susmaya görə RTP data üçün heçnə təqdim etmir. Giriş məlumatı, zəngin metadataşı idarə edici kanal üzərindən ötürülür və əgər siz bunun haqqında düşünürdüñüsə RTP data gecikməyəcək. Əgər siz səs datasının özünü də şifrələmək istəyirsinizsə bunu SRTP ilə edə bilərsiniz. SRTP böyük gecikmə yaratmadan hər bir RTP UDP paketləri üçün RTP datasını şifrələyir. Bunun üstünlüyü ondan ibarətdir ki, zəngin datası şifrələnir və UDP üzərindən işləyir. Beləliklə demək olar ki, SRTP işə salınanda və ya sönülü olanda elə də böyük fərq olmur. SRTP şifrələnmə açarı (**SSLv2/3** şifrələnməsidir) idarə edici kanalı üzərindən mübadilə edilir və size hər ikisi üçün yaxşı nəticə verir. Susmaya görə **SSLv2/3 + SRTP** firewall-la çox dostluq edir. Həmçinin FreeSWITCH server üzərində **SSLv2/3 + SRTP**-ni quraşdırmaq çox asandır və əksər SIP telefonlar üçün bu şifrələnmə metodikası dəstəklənir.

Qeyd: Z RTP - protokoludur hansı ki, PGP şifrələnməni yaradan tərəfindən yaradılmışdır. Haqqında daha ətraflı məlumatı <http://zfone.com/> ünvanından oxuya bilərsiniz.

SSL vasitəsilə şifrələnmə

SSL Şifrələnməsi WEB saytin və ya digər SSL bazalı servisin razılışması prinsipi ilə eyni olaraq işləyir. Göndərən və qəbul edənin arasında mesajların mübadiləsi üçün üçüncü tərəf istifadə edilir ki, sertifikatı yoxlasın (imzalasın). Nəzəriyyədə olan public və private hash-ə əsaslanan bu sertifikatlar telefonun özündə və serverdə uyğun olaraq yüklənəcək.

FreeSWITCH SIP paketlərin SSLv2/3 şifrələnməsini dəstəkləyir. Susmaya görə SSL-in işə salınması yalnız SIP-i işə salır və istənilən RTP data ya da media üçün məlumat vermir. Medianın necə şifrələnməsi haqqında olan məlumat isə SSL idarə etməsi kanalın üzərindən verilir ki, SRTP işə salınarsa DTP data şifrələnə bilsin. Zəng məlumatı və zəng metadataşı qorunmuş qoşulma üzərindən ötürülür. Əgər siz müdafiə haqqında narahat olursunuzsa, siz SSL şifrələnməni heç bir RTP gecikməsi olmadan istifadə edə bilərsiniz. Misal üçün, əgər siz telefonlarda paketin sniff edilməsinin işləməsini istəmirsinizsə SSLv2/3 sizin SIP paketləri şifrələnməsinə kifayət edəcək. Əgər siz səs datasını da şifrələmək istəyirsinizsə bunu SRTP ilə birlikdə edə bilərsiniz.

SSLC2/3-ün işə salınması

SSL şifrlənməsinin qayda ilə işə salınması üçün FreeSWITCH OpenSSL kitabxanaları ilə kompilyasiya edilməlidir. Sonra da özünüzün SSC(Self Sign Certificate)-ini generasiya eləməlisiniz. Bu sertifikatlar tamamilə WEB server üzərində olan sertifikatlarla eyni olaraq işləyirlər.

FreeSWITCH-i OpenSSL kitabxanaları ilə birlikdə kompilyasiya eləmək üçün əmin olun ki, OpenSSL qurucu kitabxanaları artıq sisteminizdə yüklənmişdir. Sonra **./configure** əmrini **--with-openssl** flagla işə salın. Nəzərə alın ki, bu opsiya onsuza susmaya görə qoşulu olur amma, hər hal üçün bilin ki, bununla etmək lazımdır.

FreeSWITCH-in daxilində sadə scriptlər var hansı ki, sizə internal SIP profaylda özü-özünü imzalayan sertifikatın tez yaradılmasında kömək edəcək. Əgər sizin serverinizin daxili host adı **frfs.opensource.az**-dirsa siz aşağıdakı əmri işə sala bilərsiniz.

```
# cd /usr/local/freeswitch/bin/
# ./gentls_cert setup -cn frfs.opensource.az -alt DNS:frfs.opensource.az -org
opensource.az
Creating new CA...
Generating a 2048 bit RSA private key
.....+++  

.....+++
writing new private key to '/usr/local/freeswitch/conf/ssl/CA/cakey.pem'
-----
DONE

# ./gentls_cert create_server -cn frfs.opensource.az -alt
DNS:frfs.opensource.az -org opensource.az
Generating new certificate...
-----
CN: "frfs.opensource.az"
ORG_NAME: "opensource.az"
ALT_NAME: "DNS:frfs.opensource.az"

Certificate filename "agent.pem"

[Is this OK? (y/N) ]
Y
Generating a 2048 bit RSA private key
.....+++  

.....+++
.....+++
writing new private key to '/tmp/fs-ca-641-20151127141731.key'
-----
Signature ok
subject=/CN=frfs.opensource.az/O=opensource.az
Getting CA Private Key
DONE
```

Bəzi telefonlar hostname-in uyğun olmasını yoxlayır ona görə də öncəki əmrde hostname-in düzgün olmasından əmin olun. Tələb edilərsə, hostunuzda siz çoxlu sertifikatları hər bir domain adı üçün generasiya edə bilərsiniz. Bu scriptlər sertifikatları generasiya edib və imzaladıqdan sonra avtomatik olaraq onları **/usr/local/freeswitch/conf/ssl/** qovluğuna yerləşdirəcək.

Generasiya elədikdən sonra isə, siz FreeSWITCH-ə deməlisiniz ki, hansı SIP profile-i bu sertifikatları istifadə edəcək və SSL şifrələnməsini bu profile-larda işə salmalısınız. Bunu eləmək üçün öz Sofia SIP profile-inizin içində aşağıdakı sətirləri tapıb, uyğun olaraq dəyişməlisiniz(Yəni **/usr/local/freeswitch/conf/sip_profiles/internal.xml** ya da **/usr/local/freeswitch/conf/sip_profiles/external.xml** faylında):

```
<param name="tls" value="true"/>
<param name="tls-version" value="sslv23"/>
<param name="tls-cert-dir" value="/usr/local/freeswitch/conf/ssl"/>
```

Əgər siz generasiya elədiyiniz sertifikatların özünü qorumaq üçün şifrə təyin etmişsinizsə, şifrəni aşağıdakı direktivdə təyin edə bilərsiniz:

```
<param name="tls-passphrase" value="" />
```

Əlavə təhlükəsizlik üçün siz fərqli profile-lar üçün fərqli sertifikatları generasiya edə bilərsiniz.

Artıq siz sertifikatları generasiya eləyib FreeSWITCH-də profiliniz üçün işə saldıqdan sonra, son nöqtədə olan SIP telefon üçün bunu işə sala bilərsiniz.

TLS ilə şifrələnmə

Təhlükəsiz signallaşma üçün TLS alternativ şifrələnmə mexanizmidir. Bu ümumilikdə daha yetkin startegiyadır. TCP qoşulması üzərindən keçən hər şeyi TLS şifrələyir və bu qoşulmayı danişiq pəncərəsinin davamıyyəti üçün dəstəkləyir.

SSL-də olduğu kimi də, TLS-də OpenSSL kitabxanalarını tələb edir. Həmçinin əmin olun ki, sisteminizdə OpenSSL kitabxanaları mövcuddur və FreeSWITCH --**with-openssl** flagi ilə quraşdırılmışdır.

Sıgnal səviyyəsində TLS şifrələnməsinin işə salınması üçün aşağıdakı sətiri dailpla-nıñiza əlavə edin:

```
<param name="tls-version" value="tlsv1"/>
```

TLS ilə əlaqəli çox qarışılıq və səhvlər mövcuddur. **TLS** UDP ilə yox **TCP** ilə işləyir. Bunun pis tərəfləri də mövcuddur hansı ki, FreeSWITCH telefonu qoşulmalıdır və əgər telefon NAT ya da firewall arxasındadırsa, əlçatılmaz ola bilər. Siz əmin olmalısınız ki, bütün firewalllar TCP giriş trafiki ilə işləmək üçün öncədən quraşdırılmışdır. Həmçinin əmin olun ki, son nöqtə də vaxt düzgün quraşdırılmışdır əks halda, siz pis sertifikat yalnız səhvləri alacaqsınız və əl sıxışması yaranmayacaq.

Öncəki göstərilən SSL misalında olduğu kimi, siz həmçinin SSL sertifikatlarının yerləşdiyi ünvanı göstərməlisiniz.

```
<param name="tls-cert-dir" value="/usr/local/freeswitch/conf/ssl"/>
```

Əgər siz öz sertifikatınızı şifrə ilə qoruyursunuzsa onda aşağıdakı qaydada şifrəni yaza bilərsiniz.

```
<param name="tls-passphrase" value="" />
```

Artıq siz sertifikatı qurduqdan və FreeSWITCH-i işə saldıqdan sonra öz clientinizdə TLS-i işə sala və SIP paketləri şifrələyə bilərsiniz.

Səsin qorunması

Audio tərkibi (həmişə RTP kimi tanınır) ola bilər ki, VoIP mübadiləsinin ən vacib hissəsidir. Bu VoIP təhlükəsizliyin ən vacib hissələrindən biridir. RTP axının şifrələnməsi o deməkdir ki, artıq real telefon danışçıları dinlənilə, götürülə və illegall üsulla əldə edilə bilməz. Bu işi görmək üçün çoxlu yollar vardır. Şifrələnmə alqoritmləri mövzusunda lazımdır ki, hər iki tərəf həm şifrələnmə və həm də deşifrələnmə üçün aralarında razılığa gəlməlidirlər. Başqa sözlə desək siz hər iki tərəfin dəstəkləmədiyi şifrələnmə metodundan istifadə edə bilməzsınız. Bundan başqa olaraq şifrələnmə alqoritmləri zəngin əvvəlində olan açar mübadiləsinə əsaslanır. Bu açar mübadiləsi hər iki tərəf üçün şifrə mübadiləsinə oxşayır ancaq elektronlaşdırılmış və avtomatlaşdırılmış üsulla.

Orda audio və audio axınlarda şifrələnmənin iki məhşur yolu istifadə edilir. Bunlar SRTP və ZRTP-dir.

SRTP 2004-cü ildə Cisco və Ericsonda işləyən IP protocol və şifrələnmə experleri olan kiçik qrup tərəfindən yaradılmışdır. SRTP vasitəsilə RTP

verilənlərinə mesajın bütövlüyü və qeydiyyatı yerinə yetirilir. Bu **unicast** və **multicast** programlarda işləmək üçün dizayn edilmişdir. Ona görə ki, bu köhnədir və əsas IP telefonları avadanlıqları ilə işləyənlər tərəfindən qurulmuşdur. Hal-hazırda işləyən əksər avadanlıqlara mənimsədilmişdir. Hal-hazırda SRTP əksər alətlərdə var və dəstəklənir.

Z RTP isə 2006-ci ildə PGP-ni yaradan şəxs Phil Zimmermann tərəfindən yaradılmışdır. Bu daha yenidir və açarın razılışdırılmasını avtomatik olaraq edir (RTP zənglərdə şifrələnmə və mübadilə işini çox asanlaşdırır). Həmçinin server tərəf şifrələnmədən asılı deyil. Şifrələnmə serverlər arasında RTP axının anlamadan da baş verir. Bu qəbul etmənin sürətini kifayət qədər artırır ona görə ki, asılılıq kifayət azalır. Ancaq əksər avadanlıq istehsalçıları bu protocol üçün Z RTP-ni reallığa çevirməlidirlər ki, uğurlu iş getsin.

FreeSWITCH tərəfindən hər iki protocol dəstəklənir və bu başlıqda hər ikisi açıqlanır.

S RTP vasitəsilə şifrələnmə

S RTP şifrələnməsi mexanizmi SIP üzərinən zəng müddətində razılaşır. SIP müzakirəsində hər iki tərəf RTP şifrələməni dəstəkləməlidir və SIP paketlərin şifrələnməsi üçün açarları mübadilə etməlidirlər. S RTP üçün istifadə edilən şifrələnmə açarı kanal idarə edilməsi üzərindən mübadilə edilir. Bu məlumatda sonradan audionun şifrələnməsi üçün istifadə edilir.

RTP datada olan şifrələnməni UDP paketlərin sayəsində S RTP sayılmaz dərəcədə gecikməni azaldır. Bunun üstünlüyü ondan ibarətdir ki, data şifrələnir amma yenə də UDP üzərindən gedir. Burda şəbəkə üzərindən keçən data şifrələnsə də adı qaydada gedən şifrələnməyən trafik qədər sürətli keçəcək.

Ümumlilikdə SSLv2/3 və S RTP daha çox firewall-la dostluq edən strategiyaya sahibdir. SSL və S RTP-nin quraşdırılması da çox asandır.

Nəzər yetirin ki, SIP paketlərin üzərində şifrələnməni işə salanadək S RTP üçün açar açıq şəkildə gedir. FreeSWITCH ilə telefon arasında tam şifrələnmə istəyirsinizsə, SIP şifrələnməni S RTP ilə kombinasiya eləməlisiniz. Bu istənilən snoop və ya MITM hucumlarının qarşısını alacaq. Əgər yalnız S RTP işə salınarsa, yalnız RTP paketlərin xeyirli yüklənməsi olan paketləri şifrələnəcək.

S RTP-nin işə salınması

Siz S RTP-ni öz Dialplanınızdan hər zəng üçün işə sala bilərsiniz.

```
<action application="set" data="sip_secure_media=true"/>
```

Bu hər iki ayaq üçün hər iki istiqamətdə edilməlidir ki, tam effektə sahib olsun. Sözsüz ki, sizin təçizatçı S RTP-ni dəstəkləməyə bilər amma, siz bunu FreeSWITCH-ə qoşulu olan ayaqlar üçün edə bilərsiniz.

Siz öz Dialplanınızda medianın düzgün təhlükəsiz olmasını **\${sip_secure_media_confirmed}** dəyişənin təyin edilməsi ilə yoxlaya

bilərsiniz. Aşağıdakı göstərilən kod bloku SIP mediası təhlükəsiz olan kimi, **bong** tonunu oxudacaq:

```
<extension name="is_secure">
    <condition field="${sip_secure_media_confirmed}" expression="^true$">
        <action application="sleep" data="1000"/>
        <action application="gentones" data="${bong-ring}"/>
    </condition>
</extension>
```

Şifrələnməni **debug** elədikdə köməkçi elə SIP paketinin içində olur hansı ki, telefon ondan düzgün şifrələnmə üçün müraciət edir. SIP paketlərində əgər siz öz SIP qurulmanızda şifrələnmiş RTP təqdim etmişsinizsə **a=crypto** sətirini görəcəksiniz.

Z RTP vasitəsilə şifrələnmə

Z RTP elə SRTP-yə oxşar şifrələnmə alqoritminə sahibdir yalnız media axınında şifrələnmənin açar mübadiləsinə görə fərqlənirlər. Həmçinin şifrələnəndə daha da təhlükəsiz və protokolu anlamayan serverlər üçün daha rahatdır. Bu SRTP-yə baxanda Z RTP-yə daha da geniş şərait yaradır və son nöqtələrə tam idarə imkanı yaradır ki, bütün səviyyələri emal eləsin. Tələb edilən şifrələnmə isə MITM riskini saxlamır. Z RTP-də həmçinin media qurulmasında açar mübadiləsini tələb eləmir. Açıların mübadiləsi RTP danışmasının qurulması müddətində baş verir.

Z RTP uğurlu açarları RTP üzərindən Diffie-Hellman açar mübadiləsi protokolu ilə təhlükəsiz statusa sahib olduqdan sonra ötürür. Z RTP protokolu tamamilə RFC6189-da yazılmışdır.

Oeyd: Şifrələnmə haqqında oxuyun. İlk dəfə şifrələnmə haqqında oxuduqda çox çətin gələcək. Misal üçün Diffie-Hellman alqoritmi və açar mübadiləsi. Onlar geniş şifrələnmə mühitinin hissələridir (**PKI - Public Key Infrastructure**). Məsləhətdir ki, şifrələnmə haqqında müəyyən bir sənədləri oxuyasınız.

Z RTP-nin əsas üstünlüklerindən biri də odur ki, o Proxy üzərindən işləyə bilir. Adətən SRTP-də, şifrələnmiş kanal üzərindən hər bir qoşulanın şifrələnmə protokollarından xəbəri olmalıdır və həmçinin audio axının şifrələmə/deşifrələmə imkanına sahib olmalıdır. Əgər sizin axının birləşdiyi yerə yetkiniz varsa, bu şifrələnmiş axının ortasında gizli qulaq asmağa şərait yaradır. Z RTP-də isə əksinə ortada olan serverin hər kəs istifadə elədiyi şifrələnmə protokolundan xəbəri olmasına ehtiyac yoxdur. Onlar hesab edirlər ki, onlar sadə RTP paketlərdirlər. Sevrerin RTP axınında nə olmasını bilməsinə ehtiyacın olmadığına görə, iki son nöqtələr bir-birlərinə ötürülən axının təhlükəsiz olması üçün Z RTP-ni dəstəkləməlidirlər. Şifrələnmə məlumatını ötürmək anlamaq üçün proxy-ə ehtiyac yoxdur.

Z RTP-nin FreeSWITCH-də olan digər üstünlüyü isə onun susmaya görə aktiv olmasınaidir. Z RTP protokolu özü də razılışma paketlərin hər bir RTP danışmasının içində əlavə edir və cavabı gözləyir. Əgər cavab qayıdarsa Z RTP şifrələnməsi digər tərəf müraciət edən kimi işə düşür. Z RTP elə də məhşur

olmadığına görə o eksər telefon avadanlıqlarında dəstəklənmir amma ZRTP program proxy-sində işləyir. Bu proxy sizə şərait yaradır ki, ZRTP əlavəsi olan programı (**Zfone**) yükləyəsiniz və RTP trafiki avtomatik olaraq şifrələnəcək hətta, susmaya görə ZRTP-ni dəstəkləməyən telefonlarda belə. Zfone arxa fonda sakitcə işləyir və sizə açar mübadiləsi baş verən kimi, xəbər edir. Orda əlavə olaraq SDK var ki, programçılar özlərinə aid olan ZRTP işlətmək istədikləri program təminatlarını yaza bilsinlər.

Siz ZRTP dəstəklənməsini öz Dialplan-nızda aşağıdakı sətirlə işə sala ya da dayandırıa bilərsiniz:

```
<action application="set" data="zrtp_secure_media=[true|false]"/>
```

ZRTP razılaşdıqdan sonra siz aşağıdakı sətiri FreeSWITCH konsolunda görəcəksiniz

```
[DEBUG] switch_rtp.c:928 [ zrtp main]: START SESSION INITIALIZATION.
```

ZRTP protokolu ZRTP razılaşması paketlərini RTP axının içində yeridəcək. Əgər ZRTP uğurla sessiyaya başlayarsa, siz çoxlu sayıda ZRTP mesajlarını görəcəksiniz. Aşağıda təsdiq mesajıdır hansı ki, kanalın artıq təhlükəsiz olmasını bildirir:

```
[zrtp protocol]: Enter state SECURE (DH).
```

Siz həmçinin görəcəksiniz ki, seçilmiş paylaşmış **cache** avtomatik olaraq yerləşdirmişdir hansı ki, növbəti zəngdə müqayisə üçün istifadə ediləcək:

```
[ zrtp cache] Storing ZRTP cache to </usr/local/freeswitch/db/zrtp.dat>...
```

Siz həmçinin əmin olmalısınız ki, ZRTP-ni FreeSWITCH-in kompilyasiyası zamanı işə salmısınız. Əgər əmin deyilsinizsə bunu yenidən kompilyasiya zamanı **./configure --enable-zrtp** əmri ilə edə bilərsiniz.

Şifrələrin qorunması

Şifrələr FreeSWITCH-də telefon qeydiyyatından keçdikdə istifadə edilir. FreeSWITCH kənar gateway-lər üzərində qeydiyyatdan keçdikdə və inzibatçılar FreeSWITCH sistemin üzərində qeydiyyatdan keçdikdə. Əksər zamanlarda bunların hər ikisində çox asan şifrələr olur.

Əlavə olaraq, əksər istifadəçilər öz şifrələrini çox asan söz birləşmələri kimi təyin edirlər. Əksəriyyəti də öz voicemail qutularının şifrələrini susmaya görə saxlayır və dəyişmirlər.

Bu şifrlər adətən eksəriyyətinin hədəflərinə çevrilir.

Bu mexanizimlərin asanlaşdırılmasının bir neçə mexanizmi mövcuddur.

Şifrlərin qeydiyyatı

Məlumatların qeydiyyatını ötürməyə ya da diskdə açıq şəkildə saxlanılmasına ehtiyac yoxdur. SIP verilənlərin öz qovluğunuzda təyin elədikdə aşağıdakı sətirin əlavə edilməsinin əvəzinə:

```
<param name="password" value="samiam"/>
```

Şifrənin **a1-hash** alqoritmi ilə hesablanması aşağıdakı şəkildəki kimi təyin edin:

```
<param name="a1-hash" value="405b86b7bb4ba72e1ec39822f0d0ad22"/>
```

a1-hash-in generasiya edilməsi üçün **username:domain:password** strukturunda olan sətirin **md5**-ni əldə edin hansı ki, sizin istifadəçi adı, domain adı və şifrə iki nöqtə ilə ayrılmışdır. Linux üçün aşağıdakı misal kimi:

```
echo -n "jamal:opensource.az:Shifre0pEr" | md5sum
```

```
405b86b7bb4ba72e1ec39822f0d0ad22 -
```

FreeBSD və MacOS üçün aşağıdakı kimi:

```
md5 -s "jamal:opensource.az:Shifre0pEr"
```

```
MD5 ("jamal:opensource.az:Shifre0pEr") = 405b86b7bb4ba72e1ec39822f0d0ad22
```

Əldə elədiyiniz hash-i öz qovluğunuzda təyin etməlisiniz. Bu o deməkdir ki, actual SIP qeydiyyatı məlumatını öz diskinizdə saxlamırsınız və kimsə onu tapsa o şifrəni açıq şəkildə görə bilməyəcək.

Misal üçün bir istifadəçi üçün tam misal aşağıdakı kimi olacaq:

```
<user id="jamal">
  <params>
    <param name="a1-hash" value="405b86b7bb4ba72e1ec39822f0d0ad22"/>
  </params>
</user>
```

Voicemail şifrləri

Voicemail qutuları üçün öz tarixində xeyli gözdən salmalar mövcuddur.

Həmçinin kiminsə voicemail yesiyinin qulaq asılmasının nəticəsində uzaqdan zəngin yönləndirilməsi, geriyə zəng imkanlarını da uzaqdan ələ keçirmək olar.

Məşhur üsullardan biri isə odur ki, istifadəçinin voicemail-i sindirmaq və həmin şəxsin zənglərini bahalı dünya zənglərinin üzərinə yönəldirməkdir.

Mövcud günümüzədək Voicemail şifrəsi çox sindiriləbiləndir.

Asan voicemail şifrlərinin qorunması da çox asandır. FreeSWITCH voicemail şifrlərini açıq şəkildə öz bazasında saxlayır. Həmçinin şəbəkədə istifadəçilərin mail yesiklərini onların genişlənmələrinə görə ya da 1234, 1111 nömrələrinə görə axtarış edə bilərsiniz.

Şifrlərin axtarışı üçün siz script yazmalısınız ki, voicemail quraşdırılması bazasında şifrləri axtarış eləsin. Nəzərə alsaq ki, siz susmaya görə olan

FreeSWITCH istifadə edirsinizsə voicemail bazası FreeSWITCH DB qovluğunda SQLite faylda saxlanılır. Bu qovluq sizin mənbə kodlarını hara yüklediyinizdən asılıdır. Mənim halimdə bu **/usr/local/freeswitch/db/-dir.**

Sizin bazarınızın yoxlanılmasının asan yolu aşağıdakı əmrlə SQLite əmr yollamaqdır(Öncə baza yerləşən qovluğa daxil oluruq):

```
cd /usr/local/freeswitch/db
```

```
sqlite3 voicemail_default.db "select * from voicemail_prefs where password=1234 or password=1111"
```

Bu əmr SQLite3 linux client istifadə edəcək ki, **voicemail_prefs** cədvəline 1111 və 1234 şifrələri üçün baxsın. O bütün mail yesikləri üçün məlumatları ekrandan çap edəcək(Bu şifrələri istifadə edənlər üçün istifadəçi adı və şifrəni də çap edəcək). Sonra da şifrələrin dəyişdirilməsi üçün istifadəçilərə müraciət edə ya da özünüz sərt şəkildə dəyişə bilərsiniz.

Nəticə

Bu başlıq mövcud günümüzdə olan VoIP-lə bağlı təhlükəsizlik işləri haqqında danışdı. Həmçinin səslə bağlı təhlükəsizlik haqqında daha ətraflı oxumaq üçün <http://www.hackingvoip.com/> saytına müraciət edib baxa və Hacking Exposed VoIP: Voice Over IP Security Secrets & Solutions kitabı oxuya bilərsiniz.

Bu başlıq sizə VoIP-ə edilən ümumi hücumların qarşısının necə alınmasını göstərdi. Bura DoS hucumlar-da daxil idi.

14-çü başlıq

Qabaqcıl imkanlar və əlavə oxumalar

FreeSWITCH-i istifadə edən iki əsas program mövcuddur - biri FreeSWITCH-in daxilində olan C modulu və digərləri FreeSWITCH-in kənardan idarə edilməsi üçündür. Hər iki mövzu bu başlıqda açıqlanacaq.

FreeSWITCH-də çoxlu program və modullar mövcuddur ki, zəngin işlək müddətində birbaşa zəng edə və zəng müddətində zənglər arası kecid qərarıbilərsiniz. Bu modullar Caller ID axtarışı, real-time hesablama və çox hissəli konfrans modullar aralığındadır. Modullar birlikdə istifadə edilə bilər ki, zəngləri idarə edə, ümumi Dialplan programının tərkibi genişləndirilə və ya digər hansıa funksionallığı yerinə yetirə bilərsiniz.

Əlavə olaraq bütün açıq qaynaqlı FreeSWITCH cəmiyyəti elə bir mühit formalasdırmışdır ki, əksər programlar tamlıqla FreeSWITCH-i idarə edə bilsin.

Bu başlıqda artıq nəzərdə tutulur ki, siz FreeSWITCH-də biliklərə sahibsiniz və fərqli program təminatları və modullar istifadə ediləcək ki, FreeSWITCH-in imkanlarını genişləndirək. Həmçinin 3-cü tərəf program təminatı istifadə edərək FreeSWITCH-in əlavə funksionallığı sahib olmasını göstərəciyik.

Başlıqda aşağıdakı mövzular müzəkirə edilir:

- Çox istifadəçili konfranslar(*mod_conference*)
- Real-time billing (*mod_nibblebill*)
- Alternativ son nöqtələr: Skype, GSM və TDM
- Web GUI-lər və digər proektlər

Çox istifadəçili konfranslar (mod_conference)

FreeSWITCH-in tərkibində çox güclü olan çox istifadəçili modul mod_conference var hansı ki, çox istifadəçili audio konfrans sistemlərində audio kanallarının birləşməsinə şərait yaradır. Bu sistem həmcinin sizə bütün audio birləşmələr üzərində idarə etməyə, qarşılıqlı əlaqəyə (misal üçün toxunuş tonları, hər kanal üçün gedib qayıdan audio ünvanları, səsin idarə edilməsi, idarə edilmənin artırılması və.s) tam şərait yaradır. Siz sistemin resurslarından (CPU, RAM) asılı olaraq, istədiyiniz kimi, çoxlu konfranslar yarada bilərsiniz.

Quraşdırılma

mod_conference quraşdırılması XML faylların conference seksiyasında quraşdırılmışdır. Bu ümumilikdə

/usr/local/freeswitch/conf/autoload_configs/conference.conf.xml faylında yerləşir. Bu profillər Dialplanda yaradıldıqda konfranslara mənimsədilə bilər. Conference quraşdırma faylı çoxlu seksiyaların tərkibində hər biri özünə aid olan parametrlərlə parçalanmışdır. Bu seksiyalar başlığımızın tərkibində açıqlanacaq.

Konfrans profilləri

Konfrans profilləri quraşdırmalarının şablonlarıdır hansı ki, adı konfransa mənimsədilə bilər. Caller-Controls birləşməsində (bu seksiyada danışılır) konfrans profilləri şəxsi konfranslarda özünü aparma qaydalarının tam bizim tərəfimizdən dəyişdirilə bilməsinə izin verir. Siz şablon tipini yarada bilər və onları çoxlu konfranslarda mənimsədə bilərsiniz. İstifadə eləmək istədiyiniz hər bir konfrans üçün profil yaradın ya da susmaya görə olanı istifadə edin.

Konfrans profilləri profiles olaraq adlandırılır və profilə elementinin tərkibində parametrlərdən ibarət olur. Ümumi strukturu aşağıdakı kimidir:

```
<profiles>
    <profile name="default">
        <param name="paramName" value="paramValue"/>
    </profile>
</profiles>
```

Siz istənilən sayıda **<profile>** tag istifadə edə bilər və hər bir **<profile>** tag-da istənilən sayıda **<param>** tag-lar ola bilər. Aşağıda mövcud ola biləcək parametrlərin siyahısı sadalanır:

- **rate:** rate parametri səviyyəyə görə susmaya görə olan mənani təyin edir (və ən yüksək olanı) hansı ki, konfrans körpüsü istifadə edir. Bu kanala zəng edənin hər biri öz audio səviyyələrini bu səviyyə üçün transkodlaşdırılmış olmasa, öz səviyyəsini transkodlaşdırır. Səsin qarışması səbəbindən bu sistemə baxdıqda ən aşağı səviyyəni təyin edir. Əgər iki zəng edənin HD telefonu olarsa və konfrans otağında olan aralıq 8000 olarsa, zəng edənlərdə həmin səviyyəyədək endiriləcək.
 - Parametr sintaksisi: **<param name="rate" value="8000"/>**
 - Susmaya görə: 8000
 - Mövcud opsiyaları (Gələcək versiyalarda daha da çox olacaq): 8000, 12000, 16000, 24000, 32000, and 48000
- **caller-controls:** Bu parametr **caller-controls** profile-ni bu konfrans körpüsü ilə istifadə etmək üçün təyin edir.

- Parametr sintaksisi: `<param name="caller-controls" value="default"/>`
- **auto-record:** Bu parametr konfransın harda avtomatik record olub olmamasını təyin edir. Yazma kanalda bir dəfə iki və daha çox tərəf üçün başlayır. Bu opsiya təyin edilərsə, konfransı yazmaq məqsədilə ünvandan ibarət olmalıdır.
 - Parametr sintaksisi: `<param name="auto-record" value="filename"/>`
 - Susmaya görə: `off`
 - Nüsxə faylı: `/usr/local/freeswitch/sounds/conferences/${conference_name}.wav`

Sadalanan nüsxə faylin adı konfransı konfrans bridge adından istifadə edərək faylin adına qeyd edərək yazmaq istəyir.
- **interval:** Bu parametr hər **frame** üçün milli saniyələrlə olan mix edilmiş vaxtdır. Bu **ptime** işlədiyi kimi işləyir ancaq, ptime kimi zəng edənin mövcud vaxtına ehtiyacı yoxdur. Yüksək rəqəmlər daha az CPU istifadə edir amma, konvertasiya keyfiyyəti problem ola bilər. Susmaya görə olan adətən ən yaxşısidır.
 - Parametr sintaksisi: `<param name="interval" value="20"/>`
 - Susmaya görə: 20
- **energy-level:** Bu parametr digər istifadəçilərə göndərmək üçün enerji səviyyəsini təyin edir(ya da audionun gücünü təyin edir). Energetika səviyyəsi - məhdudiyyətdir, səviyyəni diktə edir, hansında ki, insan özünü fon səs-küyünə qarşı danışmağa (deməyə) formalasdırır. Bu funksiya konfransa qoşulmaq üçün, fonda olan səs-küyü silməyə (uzaqlaşdırmağa) kömək edir. Əgər bu seçim çox yüksəkdirsə, o əvvəlcə kəsilməyə gətirib çıxara bilər. Qiymət **0** söndürmə, aşkar etmə və bütün paketləri körpüylə birləşdirəcək, hətta əgər onlar yalnız fon səs-küyü olacaqlarsa.
 - Parametr sintaksisi: `<param name="energy-level" value="20"/>`
 - Susmaya görə: **20**
 - **0** təyin edilməsi tamamilə dayandırır
- **member-flags:** Bu parametr üzvə əsaslanan spesifik flagların ya da seçilmiş konfrans üzvlərinin təyin edilməsinə imkan verir. Bu opsiyalar daxil edir. Bu seçimlərə daxildir: paket qarışdırmasıyla israfçı olmaqmı (yəni konfransda heç bir danışq olmasa da audionu insanlara göndərin), müəyyən element konfransın lideridirmi(və beləliklə onlar getdikdə konfrans qurtarmalıdır) və sair. Seçimlər kanalla bölünmüş olmalıdır | simvolu.
 - Parametr sintaksisi: `<param name="member-flags" value="waste|endconf"/>`
 - **member-flags** parametri üçün opsiyalar aşağıdakılardır:
 - **deaf:** Susmaya görə konfransda olan bir üzvün digərlərinə qulaq asa bilməsinin qarşısını alır(Bu konfrans başlıdıqdan sonra da belə eventler vasitəsilə dəyişdirilə bilər).
 - **waste:** Hətta danışqlar olmasa da belə, audio-nu kanala göndərir.
 - **dist-dtmf:** Hər kanal üçün DTMF signalları yayımılayır. Əgər kimse, DTMF ton sıxarsa bu normalda ələ keçirilir və

FreeSWITCH tərəfindən yerinə yetirilir. Bu opsiya həmin formanın baş verməsinin qarşısını alır və əvəzinə DTMF tonu digər üzvlərə **əks-səda** signalları ilə yollayır.

- **endconf:** Bu tərəf bitdikdən sonra, konfransın bitməsini təyin edir.
- **conference-flags:** Bu parametr konfrans genişlənmələri üçün flagları təyin edir hansı ki, konfransın arxada özünü necə aparmasını təyin edir. Ancaq hazırda mövcud olan opsiya istifadəçilərin konfransa girməzdən önce moderatorun gözləmələridir. Moderatorlar konfransa körpüləndikdən sonra, əlavə flag ötürümləkə Dialplan tərəfindən təyin edilirlər. İstifadəçi moderatorun girişini gözlədiyi müddətədək kanalda müsiqini dinləyəcək.
 - Parametr sintaksisi: `<param name="conference-flags" value="wait-mod"/>`
- **tts-engine:** Bu parametr konfrans körpüsündə istifadə ediləcək **Text-To-Engine** parametrini təyin edir.
 - Parametr sintaksisi: `<param name="tts-engine" value="cepstral"/>`
- **tts-voice:** Bu parametr konfrans körpüsündə istifadə ediləcək Text-To-Engine səs parametrini təyin edir.
 - Parametr sintaksisi: `<param name="tts-voice" value="david"/>`
- **pin:** Bu parametr istifadəçinin konfransa girməsindən önce daxil edəcəyi PIN kodu təyin edir(bu şifrədir).
 - Parametr sintaksisi: `<param name="pin" value="12345"/>`
- **max-members:** Bu konfransda iştirak edə biləcək maksimal üzv sayı. Əgər bu həddə çatılıbsa, əlavə qoşulmaq istəyən istifadəçilər **max-members-sound** müsəqisini eşidəcək və zəng edən konfrans körpüsünə qəbul edilməyəcək.
 - Parametr sintaksisi: `<param name="max-members" value="20"/>`
- **caller-id-name:** Bu parametr başa salır ki, konfrans körpüsündən kənara zəng elədikdə zəng edənin ID adını təyin edir.
 - Parametr sintaksisi: `<param name="caller-id-name" value="John Doe"/>`
- **caller-id-number:** Bu parametr başa salır ki, konfrans körpüsündən kənara zəng elədikdə zəng edənin ID nömrəsini təyin edir.
 - Parametr sintaksisi: `<param name="caller-id-number" value="4158867900"/>`
- **comfort-noise:** Bu parametr konfransda əlavə edilən arxa fonda olan ağ səs-küyünü səviyyəsini göstərir. Bəzən abunəçilər səsin səviyyəsi çox azaldıqda düşünürlər ki, onlar konfransdan çıxardılmışdır. Bu parametr xətdə ağ səs-küyü rahat təmin edir, ona görə də abunəçi əvvəlki kimi xəttə qoşulmuş olmasına bilmər. Diqqəti yetirin ki, audio nüsxələrinin daha yüksək sürətdə təyinatı nəticəsində, bu səs-küy zəhlətökən ola bilər. Əgər siz artırmanı 8000 Hz tezliyindən yuxarı edirsinizə, siz bu parametri kökləyə bilərsiniz.
 - Parametr sintaksisi: `<param name="comfort-noise" value="1000"/>`
- **announce-count:** Bu parametr yeni şəxsi konfransə üzv olduğuda konfransda olan ümumi saya baxacaq ancaq, məhdudiyyət təyin edilərsə bu parametr çatılacaq. Bu keçərli olan text-to-speech motoru tələb edir.

- Parametr sintaksisi: `<param name="announce-count" value="5"/>`
- **suppress-events:** Bu parametr FreeSWITCH event sistemi ilə istifadə üçündür. Spesifik quraşdırma opsiyası göstərir ki, təyin edilən eventlər digər tərəflərdə istifadə edilə bilməz ona görə ki, konfransa qulaq asila bilsin.
 - Parametr sintaksisi: `<param name="suppress-events" value="true"/>`
- **sound-prefix:** Bu konfrans audio faylların aldığı susmaya görə olan ünvanı parametr olaraq təyin edir.
 - Parametr sintaksisi: `<param name="sound-prefix" value="/usr/local/freeswitch/sounds//>`

Aşağıdakı parametrlər təyin edilmiş işi gordükdə, konfrans körpüsünün içindən istifadəçi səsləri üçün musiqinin işə salınmasında yararlı olur. Bütün səslər **enter-sound** və **exit-sound** çıxmaq şərtilə(hansi ki, bütün iştirakçılarda oxudulur) konfransda olan bütün tərəflər üçün yox, seçilmiş zəng edənin kanalında oxudulur.

Bütün seçilmiş səslər spesifik formatda təyin edilir: `<param name="sound-name" value="file.wav"/>`

Seçilmiş səslər aşağıdakı kimi olacaq:

- **muted-sound:** Bu səs zəng edəndə səsin kəsilməsi baş verdikdə oxudulacaq.
- **unmuted-sound:** Bu səs zəng edəndə uzun müddət səsin kəsilməsi baş vermədikdə oxudulur.
- **alone-sound:** Bu səs zəng edən tərəfə yeganə qalan tərəf olduqda oxudulur.
- **enter-sound:** Bu səs konfrans üzv olan yenisini qoşulduğda, yerdə qalan hər bir üzvə oxudulacaq.
- **exit-sound:** Bu səs zəng edənin konfransı tərk eləməsində bütün üzvlərdə oxudulacaq.
- **kicked-sound:** Zəng edən tərəf konfransdan çıxarıldığından bu səs oxuyacaq.
- **locked-sound:** Bu səs məhdud edilmiş konfrans qoşulmaq istəyən zəng edən tərəflər üçün oxudulur.
- **is-locked-sound:** Bu səs konfrans bağlılıqda onun üzvlərinə oxudulacaq səsdır.
- **is-unlocked-sound:** Konfrans yenidən işə düşdükdə bu səs bütün konfrans üzərində oxudulacaq.
- **pin-sound:** Bu konfrans pini soruşulan müddətdə oxunulacaq.
- **bad-pin-sound:** Bu səs yanlış PIN rəqəmi daxil edildikdə işə salınır.
- **perpetual-sound:** Xüsusi qurulmadır – Tərəflər konfransda olduqda bu səs dövrüdə sonsuzadək oxuyacaq.
- **moh-sound:** Fayl ya da resurs emalı hansı ki, seçilmiş musiqini kofransda qalmış bir şəxsə **music-on-hold**-a işə salır. İkinci üzv qoşulduğda, **mod-wait** quraşdırılması təyin edilənədək audio dayanacaq.
- **max-members-sound:** Əgər maksimal qoşulma sayı olan konfrans, kimsə qoşulmaq istəyirse bu musiqi işə düşəcək.

Abonentlərin idarə edilməsi

Konfranslar zəng edənin hansı komandaları içəridən aktiv konfransın sensor tonları vasitəsilə abunəçilər üçün mümkün olması idarə etmələrinə icazə verir. Komandalar konfransın səsinin dəyişikliklərini özüne daxil edə bilər, səsin kəsilməsi/səsin əlavə etmələri və ya daha çox imkanlar. Ya da daha da qabaqcıl olan menyunun oxudulması, ya da əhalinin bir konfransdan digərinə ötürülməsi.

Abonentlərin idarə edilməsi konfrans ilk işə düşəndə öncədən qurulmuş şablonlara əsaslanır. Misal üçün siz mövcud olan idarə edilmənin siyahısını təyin edə bilərsiniz(misal üçün **0** açarı səsin kəsilməsi üçün, **1** səsin azalması üçün və **3** səsin artırılması üçün) və sonra bu idarə edilmələri **3** fərqli konfranslara məniməsə bilərsiniz. Quraşdırılmalar konfrans başlayan kimi işə düşür və konfrans müddətində işlək olaraq qalır.

Yadda saxlayın ki, siz bir konfransı özüne daxil olan idarə etmə seçimləri imkanları ilə və digər tərəfini digər idarə etmə seçimləri imkanları ilə birlikdə qura bilməzsınız.

Qeyd: Diqqət! Öz **caller-controls**-nu **default** ya da **none** adla təyin etməyin. Bu sözlər **default key** xəritələnməsi ya da **no key** xəritələnməsi rezerv edilmişdir.

Aşağıdakı **caller-controls** quraşdırması üçün misaldır:

```
<caller-controls>
<group name="standard-keys" >
    <control action="vol talk dn" digits="1"/>
    <control action="vol talk zero" digits="2"/>
    <control action="vol talk up" digits="3"/>
    <control action="transfer" digits="5" data="100 XML default"/>
    <control action="execute_application" digits="0" data="playback
conf_help.wav"/>
    <control action="execute_application" digits="#" data="execute_dialplan
conference-menu"/>
</group>
</caller-controls>
```

Öncəki misal **standard-keys** adlı caller-control profile-nın yaradılmasını göstərir. **1,2** və **3** açarları səsi alır, artırır və uyğun olaraq normallaşdırır. **5** açarı isə zəng edəni **100** genişlənməsinə yönləndirir və **0** ilə **#** açarları hər biri specific Dialplan programını yerinə yetirir.

Yayımlamaq

Konfrans quraşdırma faylinin **advertise** seksiyası sizə imkan yaradır ki, xidmətlərə mövcudluq eventləri(xəbərdarlıqlar) generasiya edəsiniz və FreeSWITCH event sistemi üzərindən hissələrə üzv olasınız. İdeya ondan ibarətdir ki, həmişəlik otaq adları təyin edək ki, telefon ya da digər alət istədikdə mövcudluq eventi yarada bilək. Kənar program təminatı isə sonra konfrans otağının istifadə edilməsi və ya edilməsini monitor edə biləcək.

Yayımlayıcı quraşdirmalarda hər element üçün **advertise** tag-ı olan otağın adı tərkibinə sahib olur. Aşağıdakı misaldakı kimi:

```
<advertise>
  <room name="888@${domain}" status="FreeSWITCH"/>
</advertise>
```

XMPP eventlərin göndərilməsi və qəbul edilməsi

Konfrans modulu XMPP serverlər üçün imkan yaradır misal üçün, Gtalk əmrləri Jabber/XMPP üzərindən qəbul edir. Bu əmrlərə istifadəçilərin çıxarılması, zənglərin yönləndirilməsi və digərləri daxildir. İstifadə üçün quraşdırması çox asandır və aşağıdakı misaldakı kimi göstərilir:

```
<chat-permissions>
  <profile name="default">
    <user name="bob@somewhere.com" commands="all"/>
    <user name="harry@somewhere.com"
      commands="|deaf|dial|energy|kick
                |list|lock|mute|norecord
                |play|record|relate|say|saymember
                |stop|transfer|undeaf|unlock
                |unmute|volume_in|volume_out|"/>
  </profile>
</chat-permissions>
```

Zəng edənlərin konfransa qoşulması

Zəng edənlər konfranslara **conference** programı vasitəsilə çatırlar hansı ki, adətən XML Dialplan-dan çağırılır ya da API çağrılarıla event socket-dən alınır. Qoşulma üçün ümumi sintaksis aşağıdakı kimidir:

```
<action application="conference" data="confname@profilename"/>
```

confname konfrans otağının adıdır və **profilename** isə konfrans quraşdırma faylından (Başlığımızın əvvəlində danışdığımız kimi) istifadə eləmək üçün profile-dır.

Siz istəkdən asılı olaraq konfrans profile adının sonuna **+flags** əlavə etməklə müəyyən parametrləri konfransa ötürə bilərsiniz. Aşağıda göstərildiyi kimi:

```
<action application="conference"
  data="confname@profilename+ConfPIN+flags{
    mute|deaf|waste|moderator
  }"/>
```

Konfranslar tələbinə uyğun olaraq ilk üzv bridge edilən kimi, yaradılır. Təyin edilmiş PIN rəqəmini göstərməklə aktiv profilin quraşdirmalarının yaradılmasından sonra konfransla yaddaşa yazılır. Bunu qeyd eləmək vacibdir ona görə ki, yaddaşa yükənmiş olan etdiyiniz dəyişikliklər mövcud konfranslara qüvvəyə minməyəcək. Misal üçün konfrans PIN rəqəmi ilə işə düşən kimi, konfransa növbəti üzv olmaq istəyən iştirakçılar bu PIN rəqəmini daxil etməlidirlər.

Daxil etdiyiniz profile adı sizin **/usr/local/freeswitch/conf/autoload_configs/conference.conf.xml** faylında olanla üst-üstə düşməlidir.

Dinamik yaradılmış konfranslar özündə olan üzvlərin sayı sıfıra düşənədək aktiv qalacaqlar.

Aşağıdakı mənaların təyin edilməsi müəyyən misallardır ki, konfransda zəngin körpülməsində data seksiyasını təyin edir:

İşin Datası	Açıqlanması
confname	Profile "default", flag ya da PIN yoxdur
confname+1234	Profile "default", PIN 1234-dur
confname@profilename+1234	Profile "default", PIN 1234-dur, flag yoxdur
confname@profilename++flags{mute waste}	Profile "default", coxlu flag, PIN yoxdur
confname+1234+flags{mute waste}	Profile "default", coxlu flag, PIN var

Nəzərə alın ki, bəzi parametrlər istəkdən asılıdır amma, onların ardıcılılığı çox önemlidir.

Aktiv konfransların idarə edilməsi

Aktiv konfransların idarə edilməsi üçün çoxlu sayda CLI və API əmrləri mövcuddur. Əksər istifadə edilən əmrlə istifadəçilə zərbə, səsin köklənməsi və zənglərin yaradılmasıdır (şəxsləri konfransa əlavə etmək). Bu mövzu bizim kitabdan kənar olduğuna görə siz FreeSWITCH-in rəsmi saytından konfrans körpüsünün idarə edilməsi üçün CLI əmrlərinin necə istifadə edilməsini oxuya bilərsiniz.

Qeyd: Konfrans haqqında əlavə ətraflı məlumatı siz onlayn olaraq http://wiki.freeswitch.org/wiki/Mod_conference linkindən əldə edə bilərsiniz.

Real-time billing (mod_nibblebill)

mod_nibblebill modulu FreeSWITCH üçün credit/debit işini görür. Modul əvvəlcə Darren Schreiber tərəfindən peşəkar səviyyənin magistral sisteminin qırılmalarını doldurmaq üçün yazılmışdı hansı ki, əvvəlcə real vaxtda dələduzluğu aşkar etmək imkanları baxımından sıxıntı var idi. Onun hədəfi

zəng baş verən zamanda məlumat bazasından kreditin və ya nağd pulun real vaxtda debitine icazə vermək üçün şərait yaratmaqdır.

Darren aşağıdakı hissələri yaratdı:

- Hesablardan real vaxt rejimində debit credit/hash işinin aparılması
- Bir zəng müddətində fərqli ölçülərdə billing imkanına icazə
- İstifadəçilərin balansı kiçildikdə onlara məlumatın ötürülməsi (audio kanal üzərindən)
- Balans tükəndikdə zənglərin qırılması və ya başqa ünvana yönləndirilməsi
- Çoxlu sayda olan zənglərlə önce sadalanan billing funksiyalarına icazə

Istifadə halları

mod_nibblebill çoxlu yerlərdə istifadə edilə bilər. Onlardan bir neçəsini aşağıda sadalayırıq.

Billing (öncədən-ödəniş)

Siz insanlara şərait yarada bilərsiniz ki, öz hesablarına pul yerləşdirsinlər və işləri ilə məşğul olsunlar. Bundan əlavə olaraq istifadəçi öz hesabında olan pulları bitirdikdən sonra, onda xüsusi musiqi və ya ton (Ya da başqa hansısa iş yerinə yetirilə bilər) işə düşə bilər. Hesabin tam tükənməsində zəng xüsusi genişlənməyə yönəldirilə bilər ki, öz balanslarını toxunma tonları ilə yenidən doldursunlar yada sadəcə zəng qırıla bilər.

Billing (Ödəniş mesajı)

Əgər sizin verilənlər bazasının sütunları buna izin verirsə siz xəbərdarlıq yarada bilərsiniz və nağd ödəniş neqativ rəqəmlərindən çıxarış edə bilərsiniz. Zəng edən verilənlər bazasında neqativ nömrələrin içərinə düşə bilərlər və sonra onların istifadə etmələrindən sonra onları tarifləşdirə bilərlər. Bu halda siz həmçinin özünüzü artıq istifadədən qoruya biləcəksiniz ona görə ki, zəng edən tərəf sizin təyin elədiyiniz hansısa həddə çatacaqlar (Yəni bir aydan sonra çox pul xərcləyəcəklər).

Bu adəti istifadə edilən qoşulma imkanları üçün ödəniş strukturudur və nəzərdə tutur ki, heç bir ödəniş edilmədən lazımdan artıq istifadə tapılarsa bağlanılacaq.

Zəngə görə ödəniş sistemi

Siz spesifik servis üçün ayrılmış ödəniş ya da müəyyən hadisədən sonra dəqiqlik ödəniş imkanı yarada bilərsiniz (Misal üçün kredit kartının nömrəsi daxil edilir və sonra qəbul olunur).

Maksimal kredit və/ya oğurluğun qarşısının alınması

Kredit sütunu qurmaq olar hansı ki, öncə gördüyüümüz əvvəlcədən ödəniş kimi, sizin istifadəçiləriniz tərəfindən tükənilir amma bu haqda heç kimə demirik. Onlar öz kreditlərini gün, həftə ay və daha da çox hansısa müddətdə bitirərlərsə daha zəng edə bilməyəcəklər. Başqa ssenari istifadə edilə bilər ki, onların hesabına təyin edilmiş intervalda daha da böyük kredit yerləşdirilsin. Bu bir gündə "100 dəqiqə" və daha da çox hansısa imkana şərait yaradardı.

Məqsədlərin dizaynı

Əgər siz **mod_nibblebill** istifadə eləmək niyyətində olsaz, nəzərə alın ki, modul aşağıdakı dizayndan ibarətdir:

- **Paralel dizayn:** Bu size çoxlu inkişafda olan kanallarda eyni hesab/hesab kodunda idarəetmə imkanlarına şərait yaratır.
 - **Scalability(Genişlənmə):** Bu fərqli ürək döyüntüsü intervallarına izin verir(ya da zəng müddətində idarə etmənin tamam dayandırılması). Bu inzibatçıya sistem yüklənməsindən asılı olaraq yoxlanış imkanı yaratır.
 - **Flexibility(Elastiklik):** Bu xəbərdarlıq səviyyələrinə və "out-of-funds" səviyyələrinə globalda və hər bir istifadəçiye əsaslanan elastikliyə şərait yaratır. Həmçinin zəng edən fondan kənarda qalanda seçilməyə şərait yaratır.
 - **Seçim imkanı:** Bu quraşdılmalar seçimdən asılı olmalıdır. İstifadəçilər kəsilərsə ya da xəbərdarlıq edilərsə baş verilmə təyin edilməlidir.

Yüklenmesi ve qurulması Mod Nibblebill yüklenmesi ve quraşdırılması

Oyud: Ýgér biz FreeSWITCH-in core səviyyədə ODBC dəstəklənməsini istəsək onda, ODBC-ə aid olan paketləri yükləməliyik. Bunu aşağıdakı kimi edə bilərsiniz.

PostgreSQL üçün PostgreSQL connector-dan istifadə edilir:

```
root@iris:~ # make -DBATCH install
```

PostgreSQL verilənlər bazasını yükləyirik və istədiyimiz hansısa baza ilə istifadəçi verilənlərini yaradırıq:

postgresql93-server yükleyirik:

```
root@frfs:~ # pkg install postgresql93-server
```

PostgreSQL inisializasiyasını işə salırıq:

```
root@frfs:~ # /usr/local/etc/rc.d/postgresql initdb
```

```
/usr/local/pgsql/data/postgresql.conf faylında aşağıdakı sətirin qarşısından
şerhi silirik:  

listen_addresses = 'localhost'
```

```
/usr/local/pgsql/data/pg_hba.conf faylında host all all 127.0.0.1/32 trust
sətirini dəyişib aşağıdakı kimi edirik:  

host all all 127.0.0.1/32 md5
```

PostgreSQL daemonu işə salırıq:

```
root@frfs:~ # /usr/local/etc/rc.d/postgresql start
```

İşə düşməsini yoxlayırıq

```
root@frfs:~ # sockstat -l | grep postgresql
pgsql      postgres    23313  3   tcp6     ::1:5432          *:*
pgsql      postgres    23313  4   tcp4     127.0.0.1:5432        *:*
pgsql      postgres    23313  5   stream  /tmp/.s.PGSQL.5432
```

Artıq pgsql istifadəcisi üçün şifrə təyin edirik:

```
root@frfs:~ # passwd postgres
Changing local password for postgres
New Password: postgres_şifrəsi
Retype New Password: postgres_şifrəsi_təkrar
```

pgsql istifadəçi adı ilə daxil oluruz, FreeSWITCH üçün istifadəçi və bu istifadəçinin qoşulması üçün verilənlər bazası yaradırıq:

```
root@frfs:~ # su postgres
$ createuser -sdrP nibbleuser
Enter password for new role: şifrə
Enter it again: təkrar_şifrə
```

fsdb adlı verilənlər bazası yaradırıq və **fsuser** istifadəçisini sahib edirik:

```
$ createdb nibblebill --owner=nibbleuser
```

Konsoldan çıxırıq:

```
$ exit
```

PostgreSQL servisini yenidən işə salırıq:

```
root@frfs:~ # service postgresql restart
```

Nəzərdə tutulur ki, aşağıdakı əmrlə bazamıza qoşulmuşuq artıq:

```
# su postgres
$ psql -h 127.0.0.1 -p5432 -U nibbleuser nibblebill
```

Verilənlər bazamızın içində əvvəlki DB-lərə aşağıdakı qaydada baxırıq(\l əmri ilə, mövcud bazamız **nibblebill**-dir):

```
nibblebill=# \l
                                         List of databases
   Name    |  Owner   | Encoding | Collate | Ctype | Access privileges
-----+-----+-----+-----+-----+-----+
 fsdb    | fsuser   | UTF8    | C      | C      |
nibblebill | nibbleuser | UTF8    | C      | C      |
```

```

postgres | postgres      | UTF8      | C          | C          |
template0 | postgres      | UTF8      | C          | C          | =c/postgresql      +
           |                         |           |           |           | postgres=CTc/postgresql
           |                         |           |           |           | =c/postgresql      +
template1 | postgres      | UTF8      | C          | C          | =c/postgresql      +
           |                         |           |           |           | postgres=CTc/postgresql
(5 rows)

```

nibblebill adlı bazanı seçirik:

```
nibblebill=# \c nibblebill
```

You are now connected to database "nibblebill" as user "nibbleuser".

Cədvəllərimizə aşağıdakı qaydada baxırıq(\d əmri ilə, mövcud cədvəl **accounts**-dır):

```
nibblebill=# \d
```

List of relations			
Schema	Name	Type	Owner
public	accounts	table	nibbleuser
public	accounts_id_seq	sequence	nibbleuser

Cədvəlimizdə olan sütunlara baxırıq(\d **accounts** əmri ilə):

```
nibblebill=# \d accounts
```

Table "public.accounts"			
Column	Type	Modifiers	
id	bigint	not null default nextval('accounts_id_seq'::regclass)	
name	character varying(256)		
cash	double precision	not null	

accounts adlı cədvəlimizin tərkibini açıqlayırıq:

```
nibblebill=# \dt+ accounts
```

Column	Type	Modifiers	Table "public.accounts"	Storage	Stats target	Description
id	bigint	not null default nextval('accounts_id_seq'::regclass)	plain			
name	character varying(256)		extended			
cash	double precision	not null	plain			

Has OIDs: no

İndi isə PostgreSQL üçün ODBC-ni qurasdırıraq. Aşağıdakı Driver PostgreSQL üçün istifadə ediləcək.

```
-rwxr-xr-x 1 root wheel 523616 Nov 29 12:43 /usr/local/lib/psqlodbcw.so*
```

```
# odbcinst -j
unixODBC 2.3.4
DRIVERS.....: /usr/local/etc/odbcinst.ini
SYSTEM DATA SOURCES: /usr/local/etc/odbc.ini
FILE DATA SOURCES...: /usr/local/etc/ODBCDataSources
USER DATA SOURCES...: /root/.odbc.ini
```

```
SQLULEN Size.....: 8
SQLLEN Size.....: 8
SQLSETPOSIROW Size.: 8
```

/usr/local/etc/odbc.ini faylinin tərkibinə aşağıdakı sətirləri əlavə edək:

```
[nibblebill]
Description=ODBC for PgSQL
Driver=/usr/local/lib/pgsqlodbcw.so
SERVER = 127.0.0.1
PORT = 5432
DATABASE = nibblebill
USER = nibbleuser
PASSWORD = freebsd
Protocol = 6.4
FetchBufferSize=99
ReadOnly = no
Debug = 1
CommLog = 1
```

/usr/local/etc/odbcinst.ini faylinin tərkibini aşağıdakı kimi edirik:

```
[PostgreSQL]
Description=ODBC for PgSQL
Driver=/usr/local/lib/pgsqlodbcw.so
UsageCount=3
```

ODBC driver yükləyirik:

```
# odbcinst -i -d -f /usr/local/etc/odbcinst.ini
odbcinst: Driver installed. Usage count increased to 4.
Target directory is /usr/local/etc
```

Verilənlər bazasına **freeswitch** adı ilə, **fsuser** istifadəçi adı və **freebsd** şifrəsi ilə qoşuluruq:

```
# isql -v nibblebill nibbleuser freebsd
```

Maraq üçün öncədən yaratdığımız cədvəlin tərkibinə baxırıq:

```
SQL> select * from accounts
```

İşimizin görülməsi üçün bizə FreeSWITCH-in ODBC dəstəklənməsi lazımdır. Ancaq FreeSWITCH özü sistemdə axtarış edir və əgər unixODBC üçün lazım olan driverlər yüklənilibsə, o avtomatik olaraq **configure** bölümündə onu tapıb hər şeyi quraşdıracaq.

Qeyd: Əgər FreeSWITCH-in core səviyyəsində ODBC (verilənlər bazasına qoşulmaq üçün connector) dəstəklənməsini istəyirsinizsə onda **./configure --enable-core-odbc-support** etməlisiniz.

FreeSWITCH-i adı qaydada yükləyirik yalnız aşağıdakı modulları seçirik ki, **mod_nibblebill** və digər tələb edilən modullarımız yüklənsin.

```
# cd /root/src/freeswitch
# ./bootstrap.sh -j
```

```
# cat /root/src/freeswitch/modules.conf | grep -v '#'
applications/mod_callcenter
applications/mod_commands
applications/mod_conference
applications/mod_curl
applications/mod_db
applications/mod_dptools
applications/mod_enum
applications/mod_esf
applications/mod_esl
applications/mod_expr
applications/mod_fifo
applications/mod_fsv
applications/mod_hash
applications/mod_httapi
applications/mod_nibblebill
applications/mod_sms
applications/mod_spandsp
applications/mod_spy
applications/mod_valet_parking
applications/mod_voicemail
asr_tts/mod_flite
codecs/mod_amr
codecs/mod_bv
codecs/mod_b64
codecs/mod_g723_1
codecs/mod_g729
codecs/mod_h26x
codecs/mod_vp8
codecs/mod_opus
dialplans/mod_dialplan_asterisk
dialplans/mod_dialplan_xml
endpoints/mod_rtc
endpoints/mod_verto
endpoints/mod_loopback
endpoints/mod_skinny
endpoints/mod_sofia
event_handlers/mod_cdr_csv
event_handlers/mod_cdr_sqlite
event_handlers/mod_event_socket
formats/mod_local_stream
formats/mod_native_file
formats/mod_shout
formats/mod_sndfile
formats/mod_tone_stream
languages/mod_lua
loggers/mod_console
loggers/mod_logfile
loggers/mod_syslog
say/mod_say_en
xml_int/mod_xml_cdr
xml_int/mod_xml_curl
```

```
xml_int/mod_xml_rpc
xml_int/mod_xml_scgi
```

Quraşdırırıq və yükləyirik:

```
# ./configure
# gmake
# gmake install cd-sounds-install cd-moh-install
```

`/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml` faylında aşağıdakı sətiri uyğun formaya gətiririk:

```
<load module="mod_nibblebill"/>
```

`/usr/local/freeswitch/conf/autoload_configs/nibblebill.conf.xml` faylında aşağıdakı sətirləri uyğun formaya gətiririk (Aşağıdakı sətirlərdə **nibblebill** adlı verilənlər bazasına **nibbleuser** adlı istifadəçi və **freebsd** şifrəsi ilə qoşuluruq. **accounts** cədvəlinde **cash** və **id** sütunlarını seçirik):

```
<param name="odbc-dsn" value="nibblebill:nibblesuser:freebsd"/>
<param name="db_table" value="accounts"/>
<param name="db_column_cash" value="cash"/>
<param name="db_column_account" value="id"/>
```

PostgreSQL-de olan bazamıza qoşulub cədvəli yaradırıq

```
# su postgres
$ psql -h 127.0.0.1 -p5432 -U nibbleuser nibblebill
create table accounts (
    id bigserial not null,
    name varchar( 256 ),
    cash double precision not null
);
```

Qeyd: Əgər cədvəli MySQL üçün yaratmaq istəsəydiniz aşağıdakı qaydada etməli idiniz:

```
CREATE TABLE accounts
(
    id int NOT NULL PRIMARY KEY,
    name VARCHAR(255),
    cash double precision NOT NULL
);
```

Cədvəlimizə müəyyən bir məlumatları əlavə edirik:

```
nibblebill=# INSERT INTO accounts (id, name, cash) VALUES (1, 'Darren', 41.4161);
INSERT 0 1
nibblebill=# INSERT INTO accounts (id, name, cash) VALUES (2, 'Joe', 50);
INSERT 0 1
nibblebill=# INSERT INTO accounts (id, name, cash) VALUES (3, 'tester9', 50);
INSERT 0 1
nibblebill=# INSERT INTO accounts (id, name, cash) VALUES (10, 'tester10', 44.8213);
INSERT 0 1
nibblebill=# INSERT INTO accounts (id, name, cash) VALUES (837269, 'My Company', 50);
INSERT 0 1
```

Sonra **accounts** cədvəlinin tərkibinə baxırıq:

```
nibblebill=# select * from accounts;
+-----+-----+
| id   | name    | cash |
+-----+-----+
| 1    | Darren   | 41.4161|
| 2    | Joe      | 50    |
| 3    | tester9  | 50    |
| 10   | tester10 | 44.8213|
| 837269 | My Company | 50    |
+-----+
(5 rows)
```

Qeyd: Data Source Names (DSN) haqqında siz FreeSWITCH üçün http://wiki.freeswitch.org/wiki/Data_source_name linkinə müraciət edə bilərsiniz.

Zəngin hesablanması

Zəngin hesablanması üçün çoxlu üsullar var. Bu seksiya həmin metodların istifadə edilməsi və onların tərkibində olan üsulların istifadə edilməsini açıqlayır.

Susmaya görə olan nibble metodu

FreeSWITCH-də susmaya görə olan billing metodu nibble-dir. Hər bir **x** saniyəsinə görə, biz hesabdan **y** dəyəri çıxırıq.

Zəngin hesablanması üçün sizə ən azı iki dəyişəni proqresdə olan kanala təyin etməlisiniz. Dəyişənlər **nibble_rate** və **nibble_account**-dur. Səliqəli imkan olaraq, **hangup** baş verənədək **mod_nibblebill** həqiqətdə sizin hesablama dəyişənlərinizin harda təyin etməsinə diqqət yetirmir. Bu o deməkdir ki, siz onları Dialplan-da directory-nin içinde Lua scriptlə kanalın manipulyasiya edilən istənilən yerində təyin edə bilərsiniz.

Bu adı formatı siz istifadəçi qovluğu veriləninə əlavə edə bilərsiniz:

```
<variable name="nibble_rate" value="0.03"/>
<variable name="nibble_account" value="18238"/>
```

Artıq istifadəçi hər gələn və gedən zəngdə dəqiqə üçün **\$0.03** dəyərində hesablanacaq. Tarif istifadəçi hesabı **18238** ilə hesablanacaq.

Susmaya görə ürək döyüntüsü **60** saniyə təyin edilib. Bu o deməkdir ki, hər bir 60 saniyə üçün hesabdan **0.03\$** məbləğ çıxılacaq. Nəzərə alın ki, bütün riyazi hesablamalar FreeSWITCH-in daxilində olan mikrosaniyərlərə hesablanır. Bu bir neçə şeyin anlamına gəlir:

- Əgər ürək döyüntüsü düzgün vaxtı hesablamazsa, siz tarif hesabının sentinin qarşısını alacaqsınız. Ancaq nəzərə almalısınız ki, sizin qoşduğunuz verilənlər bazası bunu dəstəkləməlidir.
- Sağaclar vaxt arası tikləri dəqiq hesablayırlar. Orda hesab itkisi yoxdur.
- Minimum tarifikasiya mövcud deyil.

Siz hesab intervalını qlobal olaraq aşağıdakı parametrlə təyin edə bilərsiniz:

```
<param name="global_heartbeat" value="300">
```

Bu hesablama işini hər 5 dəqiqədən bir edəcək. Hesablamalar hər bir saniyədə edilə bilər amma bu yaxşı deyil çünkü, siz öz verilənlər bazanıza hər saniyə hər bir kanal üçün müraciət edib yükləyəcəksiniz.

Hesablari aparmaq üçün Nibble billing alternativi

Siz bu modulu hesab tezliyi istifadə etmədən də işlədə bilərsiniz. Bu o deməkdir ki, siz zəngin dəyərini zəng bitdikdən sonra hesablaya bilərsiniz. Öncə danışdığınız dəyişənləri təyin etməlisiniz və həmcinin əlavə aşağıda göstərilən dəyişəni **mod_nibblebill.conf.xml** faylına əlavə etməlisiniz:

```
<param name="global_heartbeat" value="off">
```

Bunu etməklə hesablama yalnız zəngin sonunda yerinə yetiriləcək(dəstək asılanda). Vaxt hesablaması zəngə başlığından edilir. Əgər zəngə heç bir zaman cavab verilməzsə, hesablama işi aparılmayacaq.

Zəngin hesablanması formulu bu parametr aşağıdakı kimi, təyin edildikdə hesablanır:

$$([\text{time call ended}] - [\text{time call answered}]) \times [\text{rate per minute}] = \text{total charge}$$

Qeyd: Metod proqresdə olan zənglər nəzərə almır. Bu o deməkdir ki, oğurluq ola bilər və insanlar onlara təyin edilən məhdudiyyətləri aşa bilərlər.

Misallar

Aşağıdakı misallar billing senarilərinin necə yerinə yetirilməsini göstərir.

Hər bir istifadəçi üçün fərqli aralıq

Mümkündür ki, hər bir istifadəçi üçün hər dəqiqə fərqli aralıq təyin edəsiniz. Əlavə olaraq bu Dialplan misallarında da aşağıda göstərilən qaydada işləyə bilər ancaq ehtiyatlı olun ki, dəyişənləri silməyəsiniz. Aşağıdakı misala baxın.

Gəlin baxaq ki, sizdə iki istifadəçi mövcuddur - biri dəqiqə üçün **0.05\$** digəri isə, dəqiqə üçün **0.10\$**. Əgər onlar **800** pulsuz olan rəqəmə zəng edərlərsə pul hesablanmayıacaq. Siz onların qovluğunu aşağıdakı qaydada işə salmalısınız:

```
<user id="dschreiber">
<params>
<param name="password" value="1234"/>
</params>
<variables>
<variable name="nibble_rate" value="0.05"/>
<variable name="nibble_account" value="8182"/>
<variable name="default_areacode" value="415"/>
<variable name="toll_allow"
value="domestic,international,local"/>
<variable name="user_context" value="default"/>
</variables>
</user>
<user id="expensive_guy">
<params>
<param name="password" value="1234"/>
```

```

</params>
<variables>
<variable name="nibble_rate" value="0.10"/>
<variable name="nibble_account" value="2932"/>
<variable name="default_areacode" value="212"/>
<variable name="toll_allow"
          value="domestic,international,local"/>
<variable name="user_context" value="default"/>
</variables>
</user>

```

Sonra öz Dialplan-nızda hesablama aralığını pulsuz olan zenglər üçün də edin:

```

<extension name="tollfree800">
    <condition field="destination_number"
               expression="^1"(800\d{7})$">
        <action application="set" data="nibble_rate=0"/>
        <action application="bridge"
               data="sofia/gateway/flowroute/$1"/>
    </condition>
</extension>

```

İstənilən **800**-lə olmayan nömrələr istifadəçi üçün təyin edilən hesablanmada aparılacaq o vaxtadək ki, havayı zenglər **0** ilə dəyərləndiriləcək(Yəni heç bir tarif olmayıcaq).

Bütün istifadəçilər üçün tek hesablama

Öz istifadəçi hesabınızda aşağıdakı sətirləri əlavə edin:

```

<user id="mercutioviz">
    <params>
        <param name="password" value="1234"/>
    </params>
    <variables>
        <variable name="toll_allow" value="domestic,international,local"/>
        <variable name="user_context" value="default"/>
        <variable name="nibble_account" value="1"/>
    </variables>
</user>

```

Aşağıdakı kodda əlavə edin:

```

<user id="dschreiber">
<params>
<param name="password" value="1234"/>
</params>
<variables>
    <variable name="toll_allow"
              value="domestic,international,local"/>
    <variable name="user_context" value=" default"/>
    <variable name="nibble_account" value="2"/>
</variables>
</user>

```

Hesab aparmaq istədiyiniz genişlənmə üçün isə aşağıdakı sətirləri genişlənməyə əlavə edin:

```
<extension name="outbound">
    <condition field="destination_number"
        expression="^91"(\d{10,})$">
        <action application="set" data="nibble_rate=0.05"/>
        <action application="set"
            data="nibble_account=${nibble_account}"/>
        <action application="bridge"
            data="sofia/gateway/flowroute/1\$1"/>
    </condition>
</extension>
```

Ərazi koduna görə fərqli aralıqlar

Bu misal bütün zəngləri dəqiqə üçün **0.05\$** çərçivəsində hesablayır yalnız ərazi kodu 919-u və 800 olan nömrələri çıxmaq şərtilə hansı ki, havayıdırılar. Zənglər istifadəçi hesabının istənilən koduna görə tarifləndirilir və istifadəçi üçün öz qovluq profilində təyin edilir.

Aşağıdakı misalda biz Dialplandan aralıq təyin edirik. Ehtiyatlı olun. Bu istifadəçi qovluğunda olan bütün təyin edilmiş dəyişənləri silib üstünə yazacaq:

```
<extension name="tollfree800">
    <condition field=" destination_number" expression="^1?(800\d{7})$">
        <action application="set" data="nibble_account=${accountcode}"/>
        <action application="set" data="nibble_rate=0"/>
        <action application="bridge" data="sofia/gateway/flowroute/1\$1"/>
    </condition>
</extension>
<extension name="special919rate">
    <condition field="destination_number" expression="^1?(919\d{7})$">
        <action application="set" data="nibble_account=${accountcode}"/>
        <action application="set" data="nibble_rate=0.07"/>
        <action application="bridge" data="sofia/gateway/flowroute/1\$1"/>
    </condition>
</extension>
<extension name="domestic">
    <condition field="destination_number" expression="^1?(\d{10})$">
        <action application="set" data="nibble_account=${accountcode}"/>
        <action application="set" data="nibble_rate=0.05"/>
        <action application="bridge" data="sofia/gateway/flowroute/1\$1"/>
    </condition>
</extension>
```

Çatdırılma xidmətinə görə fərqli səviyyələr

Bu ideya **nibble_rate** məntiqini zəng müddətində dəyişir.

İdeya burdadır: zəng edən daxilə zəng edə bilər və zəngin ilk hissəsi üçün onlar dəqiqə üçün **\$1.00** ödəyəcəklər(Ola bilər ki **tier1** dəstəklənməsinə zəng edilib). Əgər onların **tier2** dəstəklənməsinə ehtiyacları varsa, dəyər dəqiqə üçün **\$5.00** olacaq. Tarif zəng yönləndirilən kimi dəyişəcək(dəyişənin dəyişdirilməsi kimi). Hətta zəng edən tərəf gözləmədə olanda ya da FIFO-da olanda dəyəri **0** təyin edə bilərsiniz. Aşağıdakı misalda **2000** genişlənməsi ilk

səviyyədə olan agenti **1000** genişlənməsinə yönləndirir. **2001** genişlənməsi ikinci səviyyə agenti **1001** genişlənməsinə yönləndirir:

```

<extension name="tier1">
    <condition field="destination_number" expression="^2000$">
        <!-- Onceki seviyyenin hesablanmasını saxlayır -->
        <action application="nibblebill" data="flush"/>
        <!-- Seviyyeni deyishirik -->
        <action application="set" data="nibble_rate=1.00"/>
        <!-- Tier1 reputasiyasına yonlendirir -->
        <action application="transfer" data="1000 XML default"/>
    </condition>
</extension>
<extension name="tier2">
    <condition field="destination_number" expression="^2001">
        <!-- Onceki seviyyenin hesablanmasını saxlayır -->
        <action application="nibblebill" data="flush"/>
        <!-- Seviyyeni deyishir -->
        <action application="set" data="nibble_rate=5.00"/>
        <!-- Tier2 reputasiyasına yonlendirir -->
        <action application="transfer" data="1001 XML default"/>
    </condition>
</extension>
```

Bunun istifadə edilməsinin başqa yolu isə zəng edənin texniki dəstək ilə danışması müddətində onlara seçim vermək lazımdır. Aşağıdakı kimi ediləndən başqa önceki misalda olduğu kimi, eyni konsepsiyadır hansı ki, 2002 görünmə üçün yönləndirilir:

```

<extension name="survey-after-call">
<condition field="destination_number" expression="^2002">
    <!-- 1001 genishlenmesi uzerinden destek hesablanması $1.00/deqiqə-->
    <action application="set" data="nibble_rate=1.00"/>
    <action application="set"
    data="hangup_after_bridge=false"/>
    <action application="bridge"
    data="sofia/internal/1001@${domain}"/>
    <action application="nibblebill" data="flush"/>
    <!-- Aralıq 0 teyin ele, sonra zeng edəni acıqlama ucun xfer IVR-a yolla -->
    <action application="set" data="nibble_rate=0.00"/>
    <action application="bridge"
    data="sofia/internal/1002@${domain}"/>
</condition>
</extension>
```

Qeyd: Bu metod üçün "**catch**" var. Zənglərin səviyyələri dəyişməzdən önce siz mövcud zənglərin hesablarını bazanızda sıfırlamalısınız. Bu istənilən hesablanan saniyələri verilənlər bazasına köhnə səviyyə ilə yazır. Həmçinin bu başlıqdə Application/CLI/API əmirlər siyahısında **flush**-a baxın.

Balans bitdikdə zəngi dayandır

Hesab **nabal_amt**-də təyin ediləndən aşağı düşdükdə zəng sizin seçdiyiniz genişlənməyə ötürülür. Bu size "Pulunuzun çatışmamasına görə zənginiz dayandırıldı" kimi mesajın oxumasına imkan verir. Biz istifadəçinin zəngini həqiqətən də dayandırıb tarif planına ötürdüyümüzə görə, onu məcbur edə bilərik ki, kredit kartı ilə ödəniş edə bilsin.

Öz `conf/autoload_configs/nibblebill.conf.xml` faylıınızda aşağıdakina uyğun olan sətiri əlavə edin:

```
<param name="nabal_amt" value="0"/>
<param name="nabal_action" value="hangup XML default"/>
```

Bu misalda, "`hangup XML default`"-in parametri **nabal_action** susmaya görədir. Burda **mod_nibblebill**-ə deyilir ki, zəngi hangup adlı genişlənməyə öz XML Dialplanınızda **default** kontekstdə ötürün o vaxt ki, balans **nabal_amt** həddinə çatır. Sonra siz aşağıdakı sətirləri dialplanınıza əlavə edə bilərsiniz:

```
<extension name="hangup">
  <condition field="destination_number"
    expression="^ (hangup) \$">
    <action application="playback" data="no_more_funds.wav"/>
    <action application="hangup"/>
  </condition>
</extension>
```

Bu misalda zəng ədədlərin balansı sıfıra çatdıqda onların zəngləri **hangup** genişlənməsinə yönəldiriləcək. Bu genişlənmə fondun bitməsi haqqında musiqi oxuyacaq (nəzərdə tutulur ki, siz **no_more_funds.wav** adlı musiqi yazmısınız) və zəng kəsiləcək.

Qeyd: Ehtiyatlı olun ki, **B** ayağı həmçinin hazırda yönləndirilmiş zəngi eyni genişlənmədə alır. Digər sözlə desək qarşı tərəfdə fondun bitməsi haqqında səsi eşidəcək.

Application/CLI/API əmrləri

Aşağıdakı əmrlər Dialplandan, CLI ya da API-dan istifadə edilə bilər. Sintaksis yalnız program formatlarında olan fərqlərindən başqa, hamısı üçün eynidir.

```
<action application="nibblebill" data="action [params]"/>
```

CLI ya da API əmrləri üçün aşağıdakı kimidir:

```
nibblebill <channel-uuid> <action> [params]
```

Check

Programın içində **check** əmrinin əlavə edilməsi ya da CLI-da UUID istifadə edilməsi yaxında hesablanan balansı qaytaracaq. Bura verilənlər bazasına yazılmayan inkrementlər daxil deyil.

```
<action application="nibblebill" data="check"/>
```

Flush

Aşağıdakı kodu sizin dialplana əlavə edir:

```
<action application="nibblebill" data="flush"/>
```

öncə göstərilən koda gözləmədə olan hesabi billing bazasına yazacaq. Billing davam edəcək amma, bu nöqtədə vaxtında istənilən hesablanma üçün tələb ediləcək hesablanacaq və yazılıcaq. Hesablanma dayananda bunun effekti qalmır.

Pause

Aşağıdakı kodu öz Dialplanınıza əlavə edin:

```
<action application="nibblebill" data="pause"/>
```

Bu hesaba ara vermək üçün flagı təyin edir. Əgər hesab müddətində zəng dayandırılıbsa, ara verilmiş zəng vaxtı hesablanmayanadək hesablanma olmayacaq ancaq, hesablanma yazılmayanadək **pause**-a üstünlük verəcək. Siz həmçinin sonra zəng müddətində **resume** əmrini istifadə etəməklə hesablaya bilərsiniz.

Qeyd əgər siz zəngə artıq ara verilməsindən sonra **pause** əmrini işə salmışsınızsa, onda **pause** əmrinə məhəl qoyulmayacaq.

Resume

Aşağıdakı kodu öz Dialplanınıza əlavə edin:

```
<action application="nibblebill" data="resume"/>
```

Bu önce ara verilmiş hesabi zəng müddətində bərpa edəcək. Ara verilmə ilə qaytarılma arasında olan vaxt işə hesablanmayacaq. Nəzərə alın ki, siz zəngi çoxlu sayda **pause** edib və **resume** edə bilərsiniz. Hər bir edilən **pause** və **resume** arasında olan vaxt hesablanmayacaq.

Reset

Aşağıdakı kodu öz Dialplanınıza əlavə edin:

```
<action application="nibblebill" data="reset"/>
```

Bu billing vaxtını hazırkı vaxta təyin edir. Ancaq nəzərə alın ki, burda elədiyiniz hər şey mövcud zaman üçün zəngi izləyən daxili sayğacları sıfırlayır. Beləliklə bu vaxtadək tariflənən istənilən zaman (amma hələki diskə yazılmayan) itəcək ya da pulsuz hesab ediləcək.

Qeyd: Təyin edilmiş istifadəçi hesabi üçün verilənlər bazasında hesablanma biliçək istənilən say son müqavilə hesab edilir. Bu əmr artıq diskdə verilənlər bazasında qeyd edilmişlər üçün təsir etmir.

Fondların əlavə edilməsi və hesablanması adjust əmrini öz Dialplanınıza əlavə edin:

```
<action application="nibblebill" data="adjust 5.00"/>
```

Bu istifadəçi hesabına müəyyən fonduna əlavə edir ya da silir (Bu halda **\$5.00** əlavə edilir). Qeyd edin ki, bu durmadan baş verir və verilənlər bazası işləməyincə bütün müdafiəni aşır. Sizin öhdəliyiniz ondan ibarətdir ki, bu əmr istifadə edildikdə verilənlər bazası həmişə işlək vəziyyətdə olsun. Mənfi rəqəmlər istifadə edin ki, istifadəçi hesabından çıxasınız.

Sessiyanın döyüntüsünün işə salınması

Sessiya üçün heartbeat aşağıdakı əmrlə işə salınır:

```
<action application="nibblebill" data="heartbeat 60"/>
```

Bu döyüntünü mövcud zəng üçün 60 saniyəlik təyin edir. Siz bunu hər zəng üçün müxtəlif təyin edə bilərsiniz.

Hesablama yalnız B ayağında baş verir.

Əgər siz yalnız B ayağını hesablamaq istəyirsinizsə, **enable_heartbeat_events** dəyişəni B ayağı kanalında işə salınmalıdır. Siz Heartbeat hadisələrini **bridge** əmri ilə işə salaraq ala bilərsiniz. Bu kitabda öncədən danışdığımız kimi, fiqurlu mötərizələr daxilində olan dəyişənlər **bridge** əmri ilə B ayağına ötürülür. Burda misal var:

```
<action application="bridge"
  data="{enable_heartbeat_events=5,nibble_rate=1,nibble_
  account=0838833133}sofia/external/$1@tel.co."/>
```

Alternativ son nöqtələr

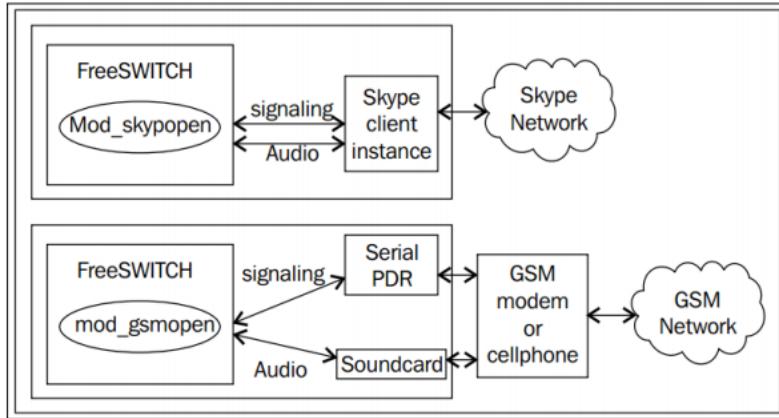
O vaxtadək ki, əksər FreeSWITCH istifadəçiləri SIP(həmçinin **mod_sofia**) istifadə edəcək, orda FreeSWITCH-in digər yolları vardır ki, dünya ilə əlaqəyə girsin. Aşağıda bu metodlardan 3-ü qısa şəkildə açıqlanır.

Skype və GSM son nöqtələri

Hər kəsin GSM (Global System for Mobile Communications) ya da Skype şəbəkəsinə qoşulmaq üçün avadanlıq ya da xidmət almağa büdcəsi olmur, xüsusən də ilk məqsəd öyrənməkdirsə. FreeSWITCH-in alternativi vardır ki, bunu təklif edə bilər: bir neçə son nöqtə(Yeni kanal driverləri) yəni, ucuz üsullar daxil olan, çıxan zəngləri və mesajları ötürə bilək.

mod_skypeopen və **mod_gsmopen** modullarının FreeSWITCH imkanları ilə tam integrasiyası mövcuddur. Diaqnostika, tam idarə eləmək, tam hadisə əlaqəsi üçün imkanlara sahibdir və Sofia SIP qaçan atı ilə eyni yolda işlədilə bilən CLI əmrləri mövcuddur.

mod_skypeopen və **mod_gsmopen** modulları eyni ümumi struktura sahibdirler. Onlar kənar obyekti("interface") özlərinə aid olan signal protokolu vasitəsilə idarə edirlər və audio axını FreeSWITCH-ə/-dən interfeyslə mənsəb şəbəkəyə yönləndirir.



mod_skypeopen üçün "interface" prosesi adı Skype client programıdır hansı ki, birbaşa Skype şəbəkəsi ilə əlaqəyə girir, audio axını FreeSWITCH-ə ötürür. Skype client prosesi **mod_skypeopen** tərəfindən Skype API-a aid olan əmrlərlə(signallaşma) idarə edilir.

mod_gsmopen üçün "interface" GSM modemdirdir(ya da ikici əl mobil telefon) hansı ki, GSM şəbəkəsi ilə özu əlaqəyə girir. GSM interfeysi isə **mod_gsmopen** tərəfindən serial port əmrləri vasitəsilə idarə edilir(əksər isitfadə edilən AT əmrləri), o halda ki, audio axını səs kartı üzərindən keçir.

Hər iki **mod_skypeopen** və **mod_gsmopen** adı FreeSWITCH API-larını, Dialplanları və eventləri istifadə eləməklə tam şəkildə səsli zəngləri və çatı dəstəkləyir. Beləliklə əgər sizin program SIP ya da Jabber ilə işləyirsə, həm səs və həmdə mesajlar üçün bu GSM və Skype şəbəkələrində daxil olan və gedən müzakirələrdə SMS mesajlarda da işləyəcək.

mod_skypeopen vasitəsilə Skype

mod_skypeopen modulu effektiv şəkildə FreeSWITCH serverdə işlək Skype prosesinə public Skype API-lar istifadə edərək qoşulur. Skype geriyə bərpa edilməyib və bu bacarığın işləməsi üçün bu imkana ehtiyacı yoxdur. **mod_skypeopen** öz növbəsində Skype API-dan legal şəkildə istifadə edir. O halda ki, **mod_skypeopen** Skype API-i tam uyğunluqla Skype lisenziyasını qəbul edərək istifadə edir və bu heç bir yerdə təsdiqlənmiş, sertifikatlaşmış deyil ya da hansısa yolla Skype tərəfindən təsdiqlənməmişdir.

Ona görə ki, **mod_skypeopen** dünyada olan Spype API-larına yetki almaq üçün, Skype-ın işlək versiyasını tələb edir. Skype-ın işləməsi adətən X Windows tələb edir ki, bəzi resursları işə salsın. Proses çox sadədir, **mod_skypeopen** adı audio driver yaradır hansı ki, Skype ona/ondan göndərir/qəbul edir audionu. Bu virtual driver faktiki olaraq səs kartına gedən/gələn verilənləri FreeSWITCH-dən/ə qəbul edir və ya ötürür.

Qayda ilə **mod_skypeopen** istifadə edilməsi üçün siz **mod_skypeopen** modulunu kompilyasiya etməli və FreeSWITCH işləyən eyni serverdə ən azı bir ədəd **Skype client** prosesi işə salmalısınız.

Linux və Windows tam şəkildə dəstəklənir və sizdə 10-larla paralel Skypeopen zənglər eyni anda işləyə bilər(Yalnız maşınınızda Skype paralel sessiyaların işləməsi üçün kifayət qədər DDR və CPU gücü olmalıdır). MaC üçün yalnız bir Skype prosesi işləyə bilər.

Linux və Windows-da siz daxil olan Skype zəngləri/çatları üçün eyni istifadəçi adı ilə cavablaşmadan çoxlu proseslərinə sahib ola bilərsiniz. Misal üçün "**mycompany_tech_support**" 10-larca daxil olan zəng.

Həmçinin Linuxda da çoxlu çıkış zəngləri/çatları eynilə Skype istifadəçi adından edilə bilər(Yəni ki, 10-larla paralel zənglər "**mycompany_sales**" adlı Skype adında yerləşdiriləcək) ancaq, Windows-da hər bir çıkış zəngi fərqli istifadəçi adı ilə yerləşdirilməlidir ("**mycompany_sales01**", "**mycompany_sales02**" və.s)

Windows və Linux Skypeopen-lərində Skype clienti tələbdən asılı olaraq görüntüsüz ("**headless**") yüklənə və işə salına bilər(Linux serverdə Xvfb əmri ilə).

Skypeopen daxil olan/çıxan zəngləri həmçinin digər Skype istifadəçi adlarından/adlarına(classic Skype-to-Skype zəngləri) da, yerləşdirməyə izin verir ki, "SkypeOut" (Siz Skype-da skype istifadəçi adı üçün kredit almalısınız) zənglərin PSTN-ə yada telefon nömrələrini dünyaya yayımlamaq(), originalda zənglərin "SkypeIN"-nin dünyadan PSTN(Skype üçün xidmət almalısınız)-ə hədəflənən nömrələrin qəbul edilməsi və daxili yazışma üçün digər skype istifadəçi adları.

Qeyd: Əlavə olaraq mod_skypeopen modulun haqqında ətraflı şəkildə oxumaq istəsəniz, https://freeswitch.org/confluence/display/FREESWITCH/mod_skypopen linkinə müraciət edə bilərsiniz.

mod_gsmopen-lə GSM

Siz **mod_gsmopen** modulunu yükleməli və kompilyasiya etməlisiniz. Ardınca da aşağıdakılari etməlisiniz:

- Bir və ya bir neçə GSM modemləri ya da ikinci əl telefonlar(interface)
- Bir və ya bir neçə serial portlar(əksər hallarda USB port)
- Bir və ya bir neçə səs kartları(əksər hallarda ucuz USB "dongle" açar)
- İnterfeys və serial səs/port kartı ilə audio və serial qoşulması üçün kabel.

Hər bir "**interafe**" aşağıdakı elementlərdən ibarətdir.

Əksər telefonlar verilənlərin ötürülməsi üçün birbaşa serial port(USB) ilə, öz kabelləri ilə qoşula bilərlər o halda ki, siz audio kabel üçün əlinizlə audio kart üçün etməlisiniz.

Marketlərdə çox ucuz olan GSM modemləri mövcuddur üstündə daxili səs kartı və daxili USB hub-da olur. Bu alətlərlə siz yalnız FreeSWITCH maşından "black box" maşına standart USB kabel gedir.

Hər bir GSM modem bir parallel zəngi dəstəkləyir. Coxlu liniyanın(telefonların) əlavə edilməsi üçün sadəcə USB hub ya da əlavə alətləri qoşun. Əlavə detallar və dəstəklənən telefonlarla alətlər üçün Wikipedia-ya baxın. SMS gateway-in istifadəsi üçün siz yalnız "**interface**"-in serial tərəfini istifadə eleməlisiniz(səs və ya audio kabelə ehtiyac yoxdur). Siz həmçinin çoxlu ikinci əl telefonları qoşub USB hublarla(çoxlu serial dəstəklənməsi Linuxda yaxşı işləyir) istifadə edə bilərsiniz və tam yüklü prosessorda SMS buraxılması elə də güclü deyil.

Hər bir "inteface"-də özünə aid olan mobil nömrə ilə SIM card olur. İdealda sizin GSM təçizatçınızdan yaxşı mobil plan olur. Qeyd edin ki, əksər təçizatçılar daxili zənglər üçün pulsuz xidmət təşkil edirlər "SME planları", "Ailə planları" azad dəqiqələr ya da spesifik saatlar üçün spesifik təkliflər.

Siz fərqli təçizat planları ilə fərqli planları tutuştura və uyğunlaşdırı bilərsiniz ki, çıxışda SMS və zənglər üçün Dialplanda daha ucuzundan istifadə edəsiniz.

Artıq GSM Open həm Linux/UNIX və həm də Windows-da işləyir.

Qeyd: GSM Open haqqında ətraflı məlumat almaq istəsəniz <http://wiki.freeswitch.org/wiki/GSMopen> linkinə müraciət edə bilərsiniz.

FreeTDM ilə TDM

FreeSWITCH demək olar ki, əksər telefon interfeys kartları ilə uyğunlaşır. Əksər telefon kartları Sangoma, Digium, OpenVox, Rhino Technologies, RedFone və digərləri tərəfindən istehsal edilir. FreeSWITCH qurucuları originalda BSD lisenziyalı olan OpenZAP kitabxanaları üçün yaratmışlar. Bu abstract səviyyə FreeSWITCH-ə izin verir ki, Sangoma və Digium bazalı kartlarla və onların tələb edilən driverləri ilə işləyə bilsinlər. FreeTDM Sangoma texnologiyaları şirkətləri tərəfindən dəstəklənir. Yeni bir səviyyədir hansı ki, tamamilə OpenZAP ilə əvəz edilmişdir.

Qeyd: Analoq və rəqəmsal(**T1/E1** və **PRI**) kartların qurulmasının ətraflı informasiyası <http://wiki.freeswitch.org/wiki/FreeTDM> linkindən əldə edilə bilər.

Quraşdırma alətləri və əlaqəli proektlər

FreeSWITCH cəmiyyəti demək olar ki, onu fərqli məqsədlər üçün istifadə edən şəxslər tərəfindən böyüdü və inkişaf elədi. Məqsədlər kiçik evdən tutmuş böyük şirkətlərin telefon sistemlərinə qədər dəyişir. Üstünə gələn tələblərə əsalanaraq öz kodunu fərqli məsəqlər üçün təkmilləşdirdi ki, bütün yenilikləri qarşılığa bilsin. Platformalar WEB GUI-dən tutmuş kitabxana mühitlərinədək parçalanır. Bunlardan bir neçəsini birazdan müzakirə edəcəyik.

WEB GUI-lər və digər proektlər

Günümüzdə FreeSWITCH üçün saysız WEB GUI-lər mövcuddur. Elələri vardır ki, XML faylları da generasiya edirlər amma elələri də var ki, istifadəçi ilə tam əlaqəyə girib çox rahat idarə etmə funksionallığı təqdim edir. Biz onlardan bir neçəsini aşağıda açıqlayırıq.

Nəzərə alın ki, sizin tələbinizdən asılı olaraq düzgün GUI-nin seçilməsi çox önemlidir. Bəziləri dilə, bəziləri imkanlara, bəziləri genişlənmə imkanlarına üstünlük verir. Elələrində var ki, artıq qrafik şəkildə sistem ISO-ları verir. Siz hamısını yoxlamalısınız ki, özünüzə uyğun olanı tapasınız.

FusionPBX

FreeSWITCH üçün PHP-də yazılmış GUI-dir. Proekt Pfsence firewall ilə başlanılmışdır. Sonrada FusionPBX(Linux/BSD/Windows/MacOS) adına dəyişdirilmişdir və multiplatforma dəstəyi ilə yayılmışdır. Eynilə çoxlu verilənlər bazaları(PostgreSQL, MySQL və SQLite) ilə işləyə bilir.

İmkanlara limitsiz genişlənmələr, IVR menyu, email-ə voicemail, qrupların axtarışını, fax server, interaktiv konfransların idarə edilməsi, aktiv zənglərə baxma və genişlənmələr, növbələr, zəngin yönləndirilməsi, click-to-call, DISA, qurulma, multi-tenant və.s.

FusionPBX çox dinamikdir. Misal üçün "**superadmin**" qrupuna mənimsədilmiş istifadəçi web interfeysə daxil ola bilər və web interfeysə giriş edib bəzi administrativ işləri görə bilər(bunlara menyu, tərkib, mövzular, istifadəçi qrupları, multi tenant quraşdirmalar və yetki imkanları daxildir).

Əlavə məlumat, misal üçün sənədlər və şəkilər rəsmi <http://www.fusionpbx.com/> saytında mövcuddur. IRC kanalını **irc.freenode.net**-də **#fusionpbx**-dir.

FreePyBX

Noel Morgan tərəfindən yazılmış FreeSWITCH WEB GUI-dir. Bu FreeSWITCH-in web bazalı idarə edilməsinə və qurulmasına şərait yaradır. O çoxlu funksionallığa sahibdir(həmçinin multi-tenant, call center, daxili bilet sistemi və.s). Haqqında daha da ətraflı <http://www.freepybx.org/> ünvanından oxuya bilərsiniz.

blue.box

blue.box açıq qaynaqlı PHP/MySQL bazalı proektdir hansı ki, 2600hz konamdası tərəfindən FreeSWITCH-in XML fayllarının idarə edilməsi üçün qrafik interfeys imkanları verir. Bu kiçik və orta FreeSWITCH qurulmaları üçün nəzərdə

tutulmuşdur. Çox dəyişmə funksionallığına sahibdir və tələbdən asılı olaraq dəyişmə bacarığına görə modullu imkana sahibdir. Həmçinin multi-tenant-i dəstəkləyir.

Əlavə məlumatı <http://www.2600hz.org/> linkindən əldə edə bilərsiniz. Həmçinin irc.freenode.net-də IRC kanala sahibdir.

Kazoo

Kazoo 2600hz komandası tərəfindən yazılmış açıq qaynaqlı platformadır və məqsədi kütləyə paylaşılmış cloud qoşulmalarını çatdırmaqdır. Proekt ümumlikdə bütün açıq qaynaqlı telefon sistemləri üçün nəzərdə tutulmuşdur. Bu orta və böyük səviyyəli xidmət təcizatçılarına kömək üçün yaradılmışdır və imkan yaradır ki, dayanıqlığı şəbəkə ya da Internet üzərindən əldə edə bilsinlər. Bütün stek yaradılmışdır ki, öz bacarıqlarını API-la təqdim etsinlər və beləliklə də bütün funksiyalar kənar program təminatları üçün tam avtomatlaşdırıla bilər. GUI, API-ların emal edilməsi, verilənlər bazası anbarı motorunu və mesajlaşma motorunu hamısı asılı olmayan moldullarla yayındırılmışdır və sizə hansı hissənin öz programınızda istifadəsinə seçim verir.

Əgər siz bir host üzərində yayımlanmış qoşulmalar marketi ilə maraqlanırsınızsa, siz Kazoo-nu yoxlamalısınız. Bunun haqqında <http://2600hz.com/platform.html> platformasından oxuya bilərsiniz.

Kitabxanaların dəstəklənməsi

Əlavə olaraq, **Event Socket Library (ESL)** kitabxanaları da FreeSWITCH-ə mənimsədilmişdir. ESL seçdiyiniz dil üçün əlavə geniş funksionallıq təqdim edir.

Liverpie (Ruby)

Liverpie(language independent IVR proxy) Ruby-də yazılmış açıq qaynaqlı program təminatıdır ki, FreeSWITCH-lə bir tərəfdə danışır və dilindən asılı olmayıaraq istənilən web program təminatı digər tərəfdə danışır. O FreeSWITCH-in **mod_event_socket** dialoqunu HTTP qeydlərinə(fərqli parametrlərin HTTP başlıqlarında yerləşdirilməsi) tərcümə edir. Beləliklə özümüzə aid olan danışan HTTP Avtomat yaza bilərik ki, Liverpie vasitəsilə FreeSWITCH-ə qoşaq. Nəzərə alın ki, Liverpie həmçinin YAML-a cavab gözləyir(Əgər siz Liverpie-dən razısınızsa bu o deməkdir ki, siz XML qaytarmaq məcburiyyətində deyilsiz artıq).

Siz Liverpie haqqında <http://www.liverpie.com/> linkindən daha ətraflı oxuya bilərsiniz.

FreeSWITCHer (Ruby)

FreeSWITCHer - FreeSWITCH-lə əlaqəyə girmək üçün *EventMachine* bazalı Ruby kitabxanalarıdır. FreeSWITCHer kitabxanaları **mod_event_socket** üzərindən əlaqəyə çıxırlar. O həm giriş və həm də çıkış event siyahılarını yarada və mövcud zəngi Ruby kitabxanalarından işə sala bilər. İşlərin görülməsi üçün

mövcud sənədə və kod nüsxələrinə sahibdir. Haqqında daha da ətraflı oxumaq istəsəniz, <https://github.com/bougyman/freeswitcher> linkinə müraciət edə bilərsiniz.

Librevox (Ruby)

Librevox demək olar ki, FreeSWITCH-in əsas yenidən yazılıması məqamında dünyaya gəlmışdır. FreeSWITCH-in yazılımasında iştirak edən əsas yazarlardan biri Harry Vangberg qərara alır ki, FreeSWITCH-in yenidən yazısın və bu da nəticəsidir. Web saytından desək, "Librevox və FreeSWITCH kənardan bir-birlərinə oxşaya bilərlər amma, librevox daxili hissələrdə işinə çox sadə yanaşır". FreeSWITCH kimi, Librevox-unda kifayət qədər sənədləri və kod nüsxələri mövcuddur ki, işinizi başlaya biləsiniz. Haqqında daha da ətraflı oxumaq üçün <https://github.com/vangberg/librevox> linkinə müraciət edə bilərsiniz.

EventSocket (Python/Twisted)

EventSocket - FreeSWITCH event socket üçün çevrilmiş protokoldur. Bu protocol event socketin bir fayl klasında daxili və xarici metodları üçün, dəstək şəraiti yaradır. Bu fərqli məqsədlər üçün istifadə edilə bilər. Çalışır ki, sadə və genişlənilə bilən olsun və FreeSWITCH-in bütün funksionallığını çevrilmiş program təminatlarına ötürsün. Bundan başqa bu ümumilikdə event əsaslıdır və çətin realizasiyalarda FreeSWITCH-in event socket vasitəsilə idarə edilməsi üçün asan imkanlar yaradır.

Bu kod özək Nuswit Telephony API(<http://nuswit.com/>)-nin bir hissəsidir və tam imkanlı web bazalı zəng artıq Brazilia və Colombia-da istifadə edilir. Mənbə kodları, nüsxələr və sənədlər <https://github.com/fiorix/eventsocket> linkində əldə edilə bilər.

FSSocket (Perl)

Perl Object Environment(POE) mühitində formalasılmış kitabxanalardır. Perl-dən FreeSWITCH-in event socket sistemi ilə asan integrasiyası şərait yaradır. O FreeSWITCH eventlərini hash-lərin içini parçalayır. Siz bütün hər şey üçün və ya fərqli event tiplərinə görə sorğu edə bilərsiniz. Mənbə kodları, nüsxələr və sənədləşmə <http://search.cpan.org/~ptinsley/POE-Filter-FSSocket-0.07/> linkində mövcuddur.

Vestec Automatic Speech Recognition

O vaxtadək ki, FreeSWITCH PocketSphinx proekti ilə işləyir və kimsə səsin təyin edilməsi ilə ciddi məşqul olursa, mütləq Vestec-ə baxmalıdır. Vestec-in ASR platforması FreeSWITCH-lə əla işləyir və real programlarda istifadə ediləndir. Vestec biznes xarakterlidir şirkət pulsuz, saysız tam imkanlı qurucu lisenziyasını FreeSWITCH istifadə edən hər kəsə vermişdir. Ətraflı məlumat üçün Vestec ilə info@vestec.com email ünvanında əlaqəyə görə bilərsiniz. Dəstəklənən dillər ünvanında <http://www.vestec.com/products/> sadalanmışdır.

Nəticə

FreeSWITCH programlarının yaradılması üçün çoxlu sayıda program alətləri ilə bizi təmin edir. FreeSWITCH-lə paketlənən modullar bizə imkan yaradır ki, daxili API vasitəsilə geniş funksionallıqları yarada bilək. Əlavə olaraq, FreeSWITCH cəmiyyəti günümüzün tələbində olan ən son yenilikləri müzakirə edir və birlikdə tətbiq edir.

15-ci başlıq

Sınaqdan keçirilmiş əlavələr

Öncəki balıqlarda biz Multitenant haqqında danışmışdıq hansı ki, eyni PUBLIC İP ünvan üzərində çıxlu müştərilərə xidmət vermək üçün şərait yaradır. Başlığımızda Multitenant üçün təcrübədə olan BlueBox və FusionPBX(FreeSWITCH üçün grafik interfeys) qurulacaq.

Təlükəsizlik yoxlanışlarının sınaqdan keçirilməsi üçün bizə öncəki başlıqlarda haqqında danışdığınız SIPvicious tələb edilir. Başlıqda onun qurulub sınaqdan keçirilməsi açıqlanacaq.

Səs yazılımasının təcrübəmdə istifadə etdiyim dəyişənləri göstəriləcək.

Dayanıqlıq üçün FreeSWITCH-in Linux Ha vasitəsilə clusterinin qurulması-da bu başlıqda açıqlanacaq.

Mod_Event-Socket dilinin PHP və Python istifadəsi açıqlanır.

Mod_XML_CURL-in təcrübədə qurulması açıqlanır.

Mod_Call_Center-in təcrübədə qurulması açıqlanır.

Başlıqda aşağıdakı mövzular müzəkirə edilir:

- FreeBSD əməliyat sistemində BlueBox yüklenməsi və qurulması
- BlueBox üzərində FreeSWITCH üçün Multi Tenant Domain qurulması
- FreeBSD əməliyyat sistemində SIPvicious qurulması və sınaqdan keçirilməsi
- FreeSWITCH üzərində səslərin yazılıması
- FreeSWITCH recover və Linux HA vasitəsilə Clusterin qurulması
- Mod_Event_Socket dili vasitəsilə PHP və Python-da misallar
- FreeSWITCH SIP istifadəçiləri MySQL-də [Mox_XML_CURL]
- FreeSWITCH vasitəsilə Mod_callcenter üzərində növbələmə
- API vasitəsilə FreeSWITCH serverdən statusların alınması

FreeBSD əməliyat sistemində BlueBox yüklənməsi və qurulması

Məqsədimiz FreeBSD 10.1 X64 məşinə Bluebox-un yüklənməsidir. Bluebox web interfeysdir hansı ki, sayesində domain adına əsaslanan virtual ATS-ləri kirayə vermək olur.

Öncə portları yeniləyirik:

```
# portsnap fetch extract update
# reboot - Sistemə yenidənyüklənmə edirik
```

Qeyd: Bluebox rəsmi saytı olan http://www.2600hz.org/bluebox_download.html səhifəsində bildirilir ki, Apache 2.2, MySQL 5.1 və PHP 5.3 istifadə etmək lazımdır. Mənim halimda Apache2.4, MySQL5.5 və PHP5.3 (Bluebox PHP5.6 ilə işləmədi) istifadə edilmişdir.

/etc/rc.conf startup faylimizə aşağıdakı sətirləri əlavə edirik ki, sistem yenidənyüklənməsindən sonra, lazımsız servislər işə düşməsin:

```
syslogd_enable="YES"
syslogd_program="/usr/sbin/syslogd"
syslogd_flags="-ss"
sendmail_enable="NO"
sendmail_submit_enable="NO"
sendmail_outbound_enable="NO"
sendmail_msp_queue_enable="NO"
sendmail_rebuild_aliases="NO"
```

WEB server apache24-ü yükləyək və quraşdırıq:

```
# cd /usr/ports/www/apache24 - Port ünvanına daxil oluruq
# make config - Lazımı modulları seçirik(Susmaya görə seçdim)
# make install - Yükləyirik
```

Aşağıdakı sətiri apache quraşdırma faylinin sonuna əlavə edirik ki, yəni VirtualHostlarım üçün qovluq əlavə edilsin:

```
# echo "Include /usr/local/domen/*" >> /usr/local/etc/apache24/httpd.conf
```

VirtualHostlar üçün qovluq yaradırıq:

```
# mkdir /usr/local/domen/
```

Subdomain **trbox.opensource.az** üçün quraşdırma faylinin

/usr/local/domen/trbox.opensource.az içiniə aşağıdakı sətirləri əlavə edirik:
<VirtualHost *:80>

```
ServerAdmin jamal.shahverdiyev@opensource.az
ServerName trbox.opensource.az
AcceptPathInfo On
DocumentRoot /usr/local/www/bluebox/
<Directory "/usr/local/www/bluebox">
    AllowOverride All
    Require all granted
</Directory>
</VirtualHost>
```

Lazımı yetkiləri veririk:

```
# chown -R www:www /usr/local/domen/trbox.opensource.az
```

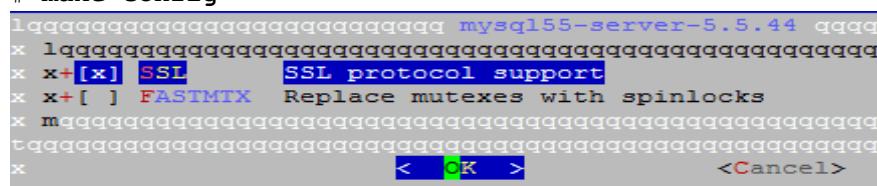
/etc/hosts faylını aşağıdaki şəklə gətiririk:
 127.0.0.1 localhost localhost.my.domain
 150.170.81.148 trbox.opensource.az trbox

Apache web serveri startup-a əlavə edirik:
`echo 'apache24_enable="YES"' >> /etc/rc.conf`

/usr/local/etc/apache24/httpd.conf faylında aşağıdaki sətirlərin qarşısından şərhi mütləq silin(Bluebox tələb edir):
LoadModule ssl_module libexec/apache24/mod_ssl.so
LoadModule rewrite_module libexec/apache24/mod_rewrite.so

Web serverə yenidənyüklənmə edirik(**bluebox** qovluğun olmamasından narahat olmayın. Birazdan yüklədikdə əmələ gələcək):
`# /usr/local/etc/rc.d/apache24 restart`

Verilənlər bazası MySQL-i yükleyirik:
`# cd /usr/ports/databases/mysql55-server/`
`# make config`



The screenshot shows a configuration dialog for MySQL. It lists several options, with 'SSL protocol support' being the one that is currently selected. Other options like 'FASTMTX' and 'm' are also visible. At the bottom of the dialog, there are 'OK' and 'Cancel' buttons.

`# make install`

MySQL serveri StartUP-a əlavə edirik və işə salırıq:
`# echo 'mysql_enable="YES"' >> /etc/rc.conf`
`# /usr/local/etc/rc.d/mysql-server start`

MySQL susmaya görə olan dəyişənlərdə müəyyən dəyişikliklər edirik(Scripti işə salırıq və uyğun suallara cavab veririk):
`# /usr/local/bin/mysql_secure_installation`
 Enter current password for root (enter for none): **ENTER_Salırıq**
 Set root password? [Y/n] **Enter**
 New password: **şifre**
 Re-enter new password: **şifre_təkrar**
 Remove anonymous users? [Y/n] **Enter**
 Disallow root login remotely? [Y/n] **Enter**
 Remove test database and access to it? [Y/n] **Enter**
 Reload privilege tables now? [Y/n] **Enter**

/etc/my.cnf faylinə aşağıdaki sətirləri əlavə edirik:
[client]
port = 3306

```

socket = /tmp/mysql.sock

[mysqld]
log = /var/log/mysql.log
bind-address = 127.0.0.1
port = 3306
socket = /tmp/mysql.sock
skip-external-locking
key_buffer_size = 256M
max_allowed_packet = 1M
table_open_cache = 256
sort_buffer_size = 1M
read_buffer_size = 1M
read_rnd_buffer_size = 4M
myisam_sort_buffer_size = 64M
thread_cache_size = 8
query_cache_size= 16M
thread_concurrency = 8
log-bin=mysql-bin
binlog_format=mixed
server-id = 1

[mysqldump]
quick
max_allowed_packet = 16M

[mysql]
no-auto-rehash

[myisamchk]
key_buffer_size = 128M
sort_buffer_size = 128M
read_buffer = 2M
write_buffer = 2M

[mysqlhotcopy]
interactive-timeout

```

Jurnal faylini yaradırıq və lazımı hüquqları veririk:

```
# touch /var/log/mysql.log
# chown mysql:mysql /var/log/mysql.log
```

MySQL-i yenidən işə salırıq ki, dəyişikliklər işə düşsün:

```
# /usr/local/etc/rc.d/mysql-server restart
```

MySQL CLI-a daxil olub bluebox üçün verilənlər bazası, istifadəçi adı və şifrə yaradırıq:

```
# mysql -uroot -pmysq1pass
mysql> CREATE database bluebox;
mysql> GRANT ALL PRIVILEGES ON bluebox.* TO 'bluebox'@'localhost' IDENTIFIED
BY 'blueb0xp@$$w0rd';
```

```
mysql> FLUSH PRIVILEGES;
mysql> \q
```

PHP53-u və onun genişlənmələrini portlardan yükleyək:

```
# cd /usr/ports/lang/php53      - Port ünvanına daxil oluruq
# make config
php53-5.3.29_2
AP2FILTER Use Apache 2.x filter interface (experimental)
APACHE Build Apache module
CGI Build CGI version
CLI Build CLI version
DEBUG Build with debugging support
FPM Build FPM version (experimental)
IPv6 IPv6 protocol support
LINKTHR Link thread lib (for threaded extensions)
MAILHEAD mail header patch
MULTIBYTE zend multibyte support
SUHOSIN Suhosin protection system
< OK > <Cancel>
```

```
# make install      - Yükləyirik
```

/usr/local/etc/apache24/httpd.conf faylında **DirectoryIndex** bölümünün qarşısına **index.php** artırırıq və faylin sonuna aşağıdakı iki sətiri əlavə etdiqdən sonra yadda saxlayıb çıxırıq:

```
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

```
# cd /usr/ports/lang/php53-extensions/      - Genişlənmələrin port ünvanına daxil oluruq
# make config      - Aşağıdakı modulları seçirik
```

```

php53-extentions-1.6
  Bcmath
  Bz2
  Calendar
  Ctype
  Curl
  DBA
  DOM
  EXIF
  FileInfo
  Filter
  Ftp
  Gd
  Gettext
  Gmp
  Hash
  Iconv
  Imap
  Interbase
  Json
  LDAP
  Mbstring
  Mcrypt
  Mssql
  MySQL
  MySQLi
  Odbc
  OpenSSL
  Pcntl
  Pdf
  PDO
  PDO_MySQL
  PDO_PgSQL
  PDO_SQLite
  Pgsql
  Phar
  Posix
  Pspell
  Readline
  Recode
  Session
  Shmop
  Simplexml
  Snmp
  Soap
  Sockets
  SQLite
  SQLite3
  Sybase_ct
  Sysvmsg
  Sysvsem
  Sysvshm
  Tidy
  Tokenizer
  Wddx

  XML support
  XMLReader support
  XMLRPC-EPI support
  XMLWriter support
  XSL support (Implies DOM)
  Zip support
  ZLIB support

< OK > <Cancel>

```

make -DBATCH install

Web serveri yenidən işə salırıq ki, dəyişiklik dərhal işə düşsün.
 # /usr/local/etc/rc.d/apache24 restart

Portlardan BASH yükleyirik çünkü bluebox tələb edir:

```

cd /usr/ports/shells/bash
make config

bash-4.3.42
  Colonbreakwords
  Docs
  Help
  Implicitcd
  Importfunctions
  Nls
  Static
  Syslog

< OK > <Cancel>

```

make install

BASH-ın işlemesi üçün aşağıdaki iki addımı yerinə yetiririk:

```

echo "fdesc  /dev/fd      fdescfs      rw      0      0" >>
/etc/fstab
mount -t fdescfs fdesc /dev/fd

```

Bununla da başlangıç işlərimiz bitdi. Növbəti işlərimizdə öncə FreeSWITCH-i qururuq və sonra Bluebox-u qururuq ki, FreeSWITCH-ə WEB vasitəsilə qoşula bilsin.

FreeSWITCH serverimizi hazırlayaq

Tez olsun deyə paketlərdən istifadə edəcəyik. Ona görə də paketləri yeniləyək və lazımları yükləyək:

```
# pkg update -f
# pkg install autoconf automake curl git gmake jpeg ldns libedit libtool
openssl pcre pkgconf speex sqlite3 wget sudo
```

FreeSWITCH mənbə kodları üçün qovluq yaradırıq və ora daxil oluruq:

```
# mkdir /root/src
# cd /root/src/
```

FreeSWITCH 1.4 versiyasının mənbə kodlarını endiririk:

```
# git clone -b v1.4 https://stash.freeswitch.org/scm/fs/freeswitch.git
```

Endirilən mənbə kodlarının qovluğuna daxil oluruq və qurulmasını işini ardıcıl görürük(Səbirli olun bu biraz vaxt alacaq):

```
# cd /root/src/freeswitch/
# ./bootstrap.sh -j
# ./configure
# gmake
# gmake install cd-sounds-install cd-moh-install
```

FreeSWITCH-i daemon kimi işə salmaq üçün Startup skriptini yaradaq və **/usr/local/etc/rc.d/freeswitch** faylinə aşağıdakı sətirləri əlavə edək:

```
#!/bin/sh
#
# PROVIDE: freeswitch
# REQUIRE: LOGIN cleanvar
# KEYWORD: shutdown
#
# Add the following lines to /etc/rc.conf to enable freeswitch:
# freeswitch_enable:      Set it to "YES" to enable freeswitch.
#                         Default is "NO".
# freeswitch_flags:       Flags passed to freeswitch-script on startup.
#                         Default is "".
#
. /etc/rc.subr
name="freeswitch"
rcvar=${name}_enable
load_rc_config $name
: ${freeswitch_enable="NO"}
: ${freeswitch_pidfile="/usr/local/freeswitch/run/freeswitch.pid"}
start_cmd=${name}_start
stop_cmd=${name}_stop
pidfile=${freeswitch_pidfile}
freeswitch_start() {
    /usr/local/freeswitch/bin/freeswitch ${freeswitch_flags}
```

```

        echo -n "Starting FreeSWITCH: "
}
freeswitch_stop() {
    /usr/local/freeswitch/bin/freeswitch -stop
}
run_rc_command "$1"

```

Skripti yerinə yetirən edirik:

```
# chmod u-w,ugo+x /usr/local/etc/rc.d/freeswitch
```

/etc/rc.conf faylına aşağıdakı sətirləri əlavə edirik ki, FreeSWITCH serverimiz sistem yenidənyüklənməsindən sonra avtomatik işə düşsün:

```
freeswitch_enable="YES"
freeswitch_flags="-nc"
```

Web servis Bluebox tərəfindən FreeSWITCH-in istifadə edilə bilməsi üçün aşağıdakı yetkiləri təyin edirik:

```
# chmod -R 775 /usr/local/freeswitch/
```

FreeSWITCH-i işə salırıq:

```
# /usr/local/etc/rc.d/freeswitch start
```

FreeSWITCH-in qulaq asdığı portlarına baxırıq:

```
# sockstat -l | grep freeswitch
root      freeswitch 14016 30  udp6      ::1:5080          *:*
root      freeswitch 14016 32  tcp6      ::1:5080          *:*
root      freeswitch 14016 34  tcp4      150.170.81.148:5080  *:*
root      freeswitch 14016 35  udp4      150.170.81.148:5080  *:*
root      freeswitch 14016 39  udp4      150.170.81.148:5060  *:*
root      freeswitch 14016 40  tcp4      150.170.81.148:5060  *:*
root      freeswitch 14016 46  udp6      ::1:5060          *:*
root      freeswitch 14016 47  tcp6      ::1:5060          *:*
root      freeswitch 14016 50  tcp4      *:8081           *:*
root      freeswitch 14016 51  tcp4      *:8082           *:*
root      freeswitch 14016 52  udp4      *:1337           *:*
root      freeswitch 14016 53  udp4      *:*
root      freeswitch 14016 63  tcp4      127.0.0.1:8021     *:*
```

/root/.cshrc faylında path dəyişənini aşağıdakı şəklə gətiririk:

```
set path = (/sbin /bin /usr/sbin /usr/bin /usr/local/freeswitch/bin
/usr/local/sbin /usr/local/bin $HOME/bin)
```

Bluebox yükləyək və quraşdırıraq

Məhz Bluebox sayəsində virtual doman adlarına görə virtual ATS-lər vermək olur.

Öncə mənbə kodlarını endirək və lazımı yetkiləri təyin edək.

```
# cd /usr/local/www/
```

```
# git clone git://github.com/2600hz/bluebox.git bluebox
```

Lazımı hüquqları təyin edək:

```
# chown -R root:www /usr/local/www/bluebox/
# chmod -R 775 /usr/local/www/bluebox/
```

Bluebox qovluğuna daxil olaq və **preinstall.sh** scriptində baş ünvanını asağıdakı səklə qətirək:

```
# cd /usr/local/www/bluebox/  
# cat preinstall.sh | grep bash  
#!/usr/local/bin/bash
```

Skripti işə salırıq və FreeBSD serverimize uyğun olan WEB istifadəçi, FreeSWITCH yüklenmiş uyğun ünvanları təyin edirik(Mənim situasiyamda hər şey susmaya görə idi. Skript avtomatik olaraq lazımlı olan hüquqları təyin edəcək):

```
# ./preinstall.sh
```

"... , www.pchanscall.com

- - - Our free software. Your next VoIP system!

SELINUX

Assuming SELINUX is not on this server . . . OK

Checking config files

```
# Copying bluebox/config/config.php
# Copying bluebox/config/database.php
# Copying bluebox/config/email.php
# Copying bluebox/config/locale.php
# Copying bluebox/config/session.php
# Copying bluebox/config/telephony.php
# Copying bluebox/config/upload.php
# Copying modules/freeswitch-1.1.1/config/freeswitch.php
# Copying modules/asterisk-1.0/config/asterisk.php
```

WEB USER

We need to verify the web user so we can correctly set the permissions on certain folder/files.

Web user name [www] ? www

BLUEBOX PRIVILEGES

Updating the permissions so Blue.box can write to its configuration files.

```
# chgrp -R www bluebox/logs/
# chmod -R g+w bluebox/logs
```

```
# chgrp -R www bluebox/cache/
# chmod -R g+w bluebox/cache
# chgrp -R www bluebox/config/
# chmod -R g+w bluebox/config/
# chgrp -R www modules/freeswitch-*/* config/freeswitch.php
# chmod -R g+w modules/freeswitch-*/* config/freeswitch.php
# chgrp -R www modules/asterisk-*/* config/asterisk.php
# chmod -R g+w modules/asterisk-*/* config/asterisk.php
# chgrp -R www uploads/
# chmod -R g+w uploads/
```

SOFTSWITCH PRIVILEGES

We need to verify the path to your softswitch confs directory so we can update its permissions.

Softswitch conf dir [/usr/local/freeswitch/conf]?

```
# chgrp -R www /usr/local/freeswitch/conf
# chmod -R g+w /usr/local/freeswitch/conf
```

SOUND FILE PRIVILEGES

We need to verify the path to your sound files.

Sound file dir [/usr/local/freeswitch/sounds]?

```
# chgrp -R www /usr/local/freeswitch/sounds
# chmod -R g+w /usr/local/freeswitch/sounds
```

RECORDING FILE PRIVILEGES

We need to verify the path to your recording files.

Record file dir [/usr/local/freeswitch/recordings]?

```
# chgrp -R www /usr/local/freeswitch/recordings
# chmod -R g+w /usr/local/freeswitch/recordings
```

PLEASE SET UP YOUR DB

You must now ensure your database has a user configured for Blue.box, you will enter the credentials in the next phase.

COMPLETE!

Now point your web browser to your server and the Blue.box installer will get your system ready.

Example <http://127.0.0.1/>

Welcome and thank you for using Blue.Box!

Artıq <http://trbox.opensource.az/> linkinə müraciət edirik və Bluebox ilk səhifəsinə açırıq. Şəkildə göstərildiyi kimi **I Accept** seçirik və **Continue** düyməsinə sıxırıq:

trbox.opensource.az/index.php/installer

[Restart Wizard](#) | [Wiki](#)

Welcome to Bluebox 1.0.4-dev Setup Wizard

Our free software. Your next voip system.

About blue.box
 blue.box provides a modular approach to traditional telephony systems. Instead of pre-determined configuration screens and business logic, Bluebox allows you to install Applications and Plug-ins to custom tailor the system's behavior to your needs.

Installation Environment
 No errors were detected with your installation environment. This system should be capable of running blue.box

Terms and Conditions

blue.box is the combination of many works. Both the FreePBX v3 license and the 2600hz license are below. You are agreeing to both licenses by installing this software.

MOZILLA PUBLIC LICENSE
 Version 1.1

1. Definitions.

1.0.1. "Commercial Use" means distribution or otherwise making the Covered Code available to a third party.

1.1. "Contributor" means each entity that creates or contributes to the creation of Modifications.

1.2. "Contributor Version" means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. "Covered Code" means the Original Code or Modifications, or the

I Accept

Continue

Verilənlər bazasının düzgün məlumatlarını daxil edib, **Continue** düyməsinə sıxırıq:

[Restart Wizard](#) | [Wiki](#)

Initial Configuration

Our free software. Your next voip system.

Database Connection

Database Type:

mysql

Database Host:

127.0.0.1

Database Name:

bluebox

Database Username:

bluebox

Database Password:

blueb0xp@\$\$w0rd

Database Port:

3306

System Defaults

Site Domain

http://trbox.opensource.az/

Upload Directory

/usr/local/www/bluebox/uploads

Default Timezone

Baku

Install Sample Data

Anonymous Usage Statistics

This option enables reporting on which modules you use. The data is sent to the Bluebox team via the Internet. We DO NOT transmit any configuration options or personally identifiable information. This feature ensures our focus remains on the modules people use most. Thank you for your support!

Allow Anonymous Statistic Collection

Continue

Back

Bizə xəbərdarlıq ediləcək ki, mövcud baza tamamilə silinəcək. Razılaşırıq
Continue düyməsinə sıxırıq:

[Restart Wizard](#) | [Wiki](#)

Initial Configuration

Our free software. Your next voip system.

The existing database will be permanently erased if you continue!

Click continue again to proceed...

Database Connection

Database Type:

Database Host:

Database Name:

Database Username:

Database Password:

Database Port:

System Defaults

Site Domain

Upload Directory

Default Timezone

Install Sample Data

Anonymous Usage Statistics

This option enables reporting on which modules you use. The data is sent to the Bluebox team via the Internet. We DO NOT transmit any configuration options or personally identifiable information. This feature ensures our focus remains on the modules people use most. Thank you for your support!

Allow Anonymous Statistic Collection

[Continue](#)

[Back](#)

Bluebox istifadəçisi üçün inzibatçı adı və şifrəni daxil edirik:

[Restart Wizard](#) | [Wiki](#)

Create Main Administrator

Our free software. Your next voip system.

Master Administration Account

Email Address:

Password:

Confirm Password:

[Continue](#)

[Back](#)

Konfliktdə olan faylların silinməsi xəbərdarlığı ediləcək və razılaşaraq
Continue düyməsinə sıxırıq:

[Restart Wizard](#) | [Wiki](#)

Telephony Engine

Our free software. Your next voip system.

Softswitch Selection

Telephony Driver:

Telephony Configuration

Conf Directory:

Global Sound File Directory:

Event Socket

ESL Host:

ESL Auth:

ESL Port:

Conflicting Files

```
/usr/local/freeswitch/conf/directory/default.xml
/usr/local/freeswitch/conf/autoload_configs/conference.conf.xml
/usr/local/freeswitch/conf/autoload_configs/ivr.conf.xml
/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml
/usr/local/freeswitch/conf/autoload_configs/xml_cdr.conf.xml
/usr/local/freeswitch/conf/autoload_configs/callcenter.conf.xml
/usr/local/freeswitch/conf/autoload_configs/distributor.conf.xml
/usr/local/freeswitch/conf/autoload_configs/directory.conf.xml
/usr/local/freeswitch/conf/autoload_configs/fax.conf.xml
/usr/local/freeswitch/conf/autoload_configs/external-ipv6.xml
/usr/local/freeswitch/conf/autoload_configs/external.xml
/usr/local/freeswitch/conf/autoload_configs/internal-ipv6.xml
/usr/local/freeswitch/conf/autoload_configs/internal.xml
```

[Continue](#)
[Back](#)

Konflikt faylların silinməsinin son xəbərdarlığı və **Continue** edirik:

[Restart Wizard](#) | [Wiki](#)

Telephony Engine

Our free software. Your next voip system.

Conflicting configuration files will be permanently erased if you continue!

Click continue again to proceed...

Softswitch Selection

Telephony Driver:

Telephony Configuration

Conf Directory:

Global Sound File Directory:

Event Socket

ESL Host:

ESL Auth:

ESL Port:

Conflicting Files

```
/usr/local/freeswitch/conf/directory/default.xml
/usr/local/freeswitch/conf/autoload_configs/conference.conf.xml
/usr/local/freeswitch/conf/autoload_configs/ivr.conf.xml
/usr/local/freeswitch/conf/autoload_configs/acl.conf.xml
/usr/local/freeswitch/conf/autoload_configs/xml_cdr.conf.xml
/usr/local/freeswitch/conf/autoload_configs/callcenter.conf.xml
/usr/local/freeswitch/conf/autoload_configs/distributor.conf.xml
/usr/local/freeswitch/conf/autoload_configs/directory.conf.xml
/usr/local/freeswitch/conf/autoload_configs/fax.conf.xml
/usr/local/freeswitch/conf/autoload_configs/external-ipv6.xml
/usr/local/freeswitch/conf/autoload_configs/external.xml
/usr/local/freeswitch/conf/autoload_configs/internal-ipv6.xml
/usr/local/freeswitch/conf/autoload_configs/internal.xml
```

[Continue](#)
[Back](#)

Sonda yüklenmə prosesinin işə salınması üçün bizdən **Continue** düyməsinə sıxılması istəniləcək və istədiyini edirik:

Installation

Our free software. Your next voip system.

Ready to Install

[Click continue to install...](#)

[Continue](#)

[Back](#)

Yüklənir:

Installation

Our free software. Your next voip system.

Ready to Install

Please Wait...



Uğurlu nəticə görünən səhifəni çap edəcək və linkə sıxırıq:

Complete!

Our free software. Your next voip system.

Installation Complete!

[Click here to use Bluebox 1.0.4-dev!](#)

İşimizin uğurlu nəticəsi aşağıdakı şəkildəki kimi olacaq:



Welcome Account Admin | [Logout](#) | [Language](#)

The screenshot shows the Bluebox Cloud Telecom web interface. At the top, there's a navigation bar with tabs: System (which is selected), Connectivity, Applications, Status, Routing, Organization, and Media. Below the navigation bar, there are several icons with labels: Account Manager (with a house icon), Report Bug (with a bug icon), Custom Feature (with a hash icon), ODBC (with a database icon), Package Manager (with a circular arrow icon), and XML File Editor (with an XML icon). A prominent yellow banner at the bottom of the interface says "Welcome to blue.box!". At the very bottom left, there's a small link: "powered by blue box".

Sonda səhifəmizin sınaqdan keçirilməsi üçün XML mətn redaktorunu açmışıq:

Welcome Account Admin | [Logout](#) | [Language](#)

2600hz CLOUD TELECOM

System Connectivity Applications Status Routing Organization Media

Account Manager Report Bug Custom Feature ODBC Package Manager XML File Editor

Current File: /usr/local/freeswitch/conf/autoload_configs/alsa.conf.xml

```

<configuration name="alsa.conf" description="Soundcard Endpoint">
  <settings>
    <!--Default dialplan and caller-id info -->
    <param name="dialplan" value="XML"/>
    <param name="cid-name" value="N800_Alsa"/>
    <param name="cid-num" value="5555551212"/>
  </settings>
</configuration>

```

Position: Ln 13, Ch 1 Total: Ln 13, Ch 410

Toggle editor

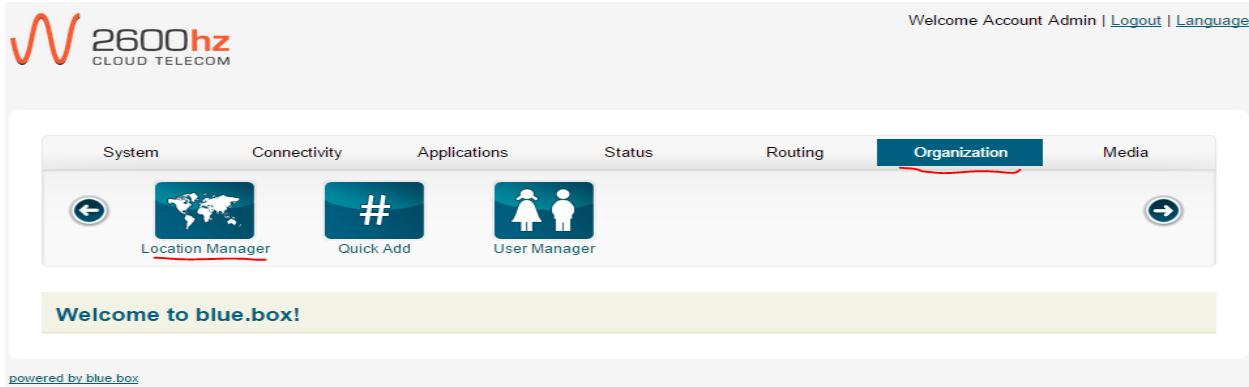
Thanks to [the folks who developed EditArea](#) for this great javascript editor.

powered by blue_box

BlueBox üzərində FreeSWITCH üçün Multi Tenant Domain qurulması
Məqsədimiz Bluebox sayəsində fərqli domain adlarına virtual ATS-lərdən istifadə etməyə şəraitin yaradılmasıdır. Domainlər aşağıda sadalanır:

main.opensource.az - Serverimizin əsas domain adı.
tenant.opensource.az - serverimizin ikinci(virtual) domain adı.

Bluebox-un web səhifəsinə ilk daxil olduqdan sonra, **Organization** tab-ın altında **Location Manager** düyməsinə sıxırıq (Şəkildə göstərilir):



Welcome Account Admin | [Logout](#) | [Language](#)

System Connectivity Applications Status Routing **Organization** Media

[Location Manager](#) [Quick Add](#) [User Manager](#)

Welcome to blue.box!

powered by blue box

Açılan səhifədə **Main** olan location seçib **Edit** düyməsinə sıxırıq:



Name	Domain Name/Realm	Actions
Main Location	[REDACTED]	Edit Delete

Page 1 of 1 100 View 1 - 1 of 1

Və öncə yazdığını **main.opensource.az** domain adını əsas olaraq əlavə edirik sonra **Save** düyməsinə sıxırıq:



Welcome Account Admin | Logout | Language

2600hz CLOUD TELECOM

System Connectivity Applications Status Routing Organization Media

Account Manager Report Bug Custom Feature ODBC Package Manager XML File Editor

Edit

Location Information

Location Name: Main Location
Domain Name/Realm: main.opensource.az

Address Information

Address: Azerbaijan
City: Baku
State: none
Zip: AZ1025

Interface Management

Bind To Interface: Default (Authenticated SIP)

Users

Email Address	First Name	Last Name	Location	User Type	Actions
jamal.shahverdiev@opensource.az	Account	Admin	Main Location	System Admin	Edit Delete
peter@initech.com	Peter	Gibbons	Main Location	User	Edit Delete
michael@initech.com	Michael	Bolton	Main Location	User	Edit Delete
samir@initech.com	Samir	Nagheenanajar	Main Location	User	Edit Delete
bill@initech.com	Bill	Lumbergh	Main Location	User	Edit Delete
milton@initech.com	Milton	Waddams	Main Location	User	Edit Delete

Page 1 of 1 View 1 - 6 of 6

Add New User

Save **Close**

powered by blue box

Sonra **System** tab altında **Account Manager** bölməsinə daxil oluruq və **Add Account** düyməsinə sıxırıq:

Welcome Account Admin | Logout | Language

2600hz CLOUD TELECOM

System Connectivity Applications Status Routing Organization Media

Account Manager Report Bug Custom Feature ODBC Package Manager XML File Editor

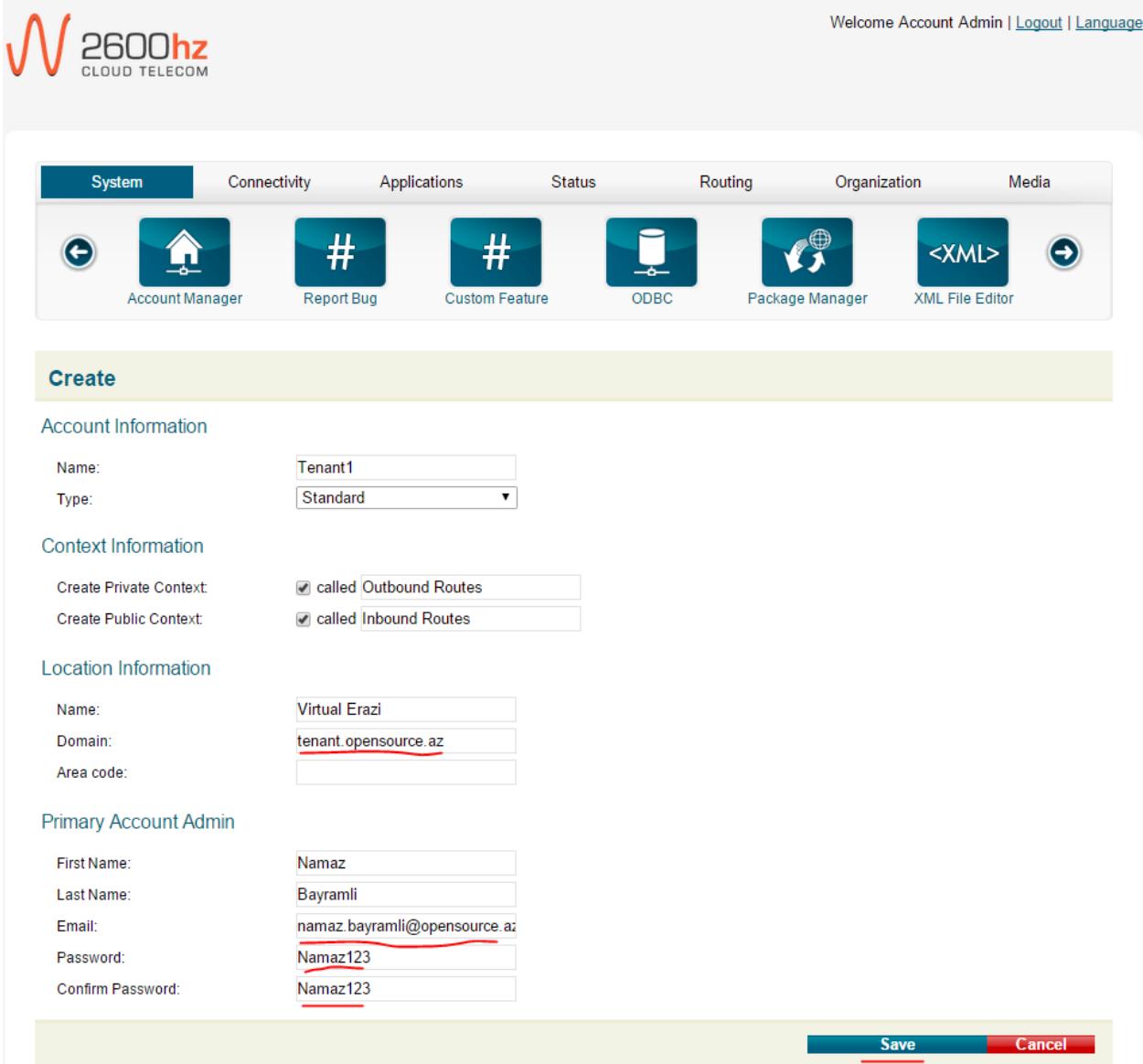
Accounts

Name	Type	Created	Actions
Master Account	Normal	2015-08-24 15:50:01	Edit Delete Rebu

Page 1 of 1 View 1 - 1 of 1

powered by blue box

Virtual domain adı yəni **tenant.opensource.az** üçün verilənləri daxil edirik və **Save** düyməsini sıxırıq:



Welcome Account Admin | [Logout](#) | [Language](#)

System **Connectivity** **Applications** **Status** **Routing** **Organization** **Media**

Create

Account Information

Name:	Tenant1
Type:	Standard

Context Information

Create Private Context:	<input checked="" type="checkbox"/> called Outbound Routes
Create Public Context:	<input checked="" type="checkbox"/> called Inbound Routes

Location Information

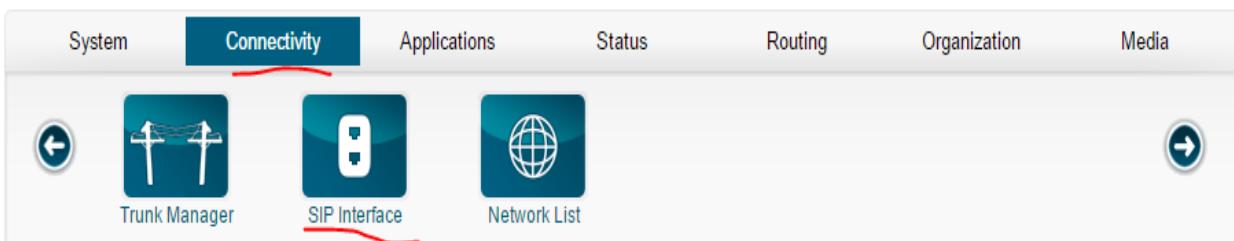
Name:	Virtual Erazı
Domain:	tenant.opensource.az
Area code:	

Primary Account Admin

First Name:	Namaz
Last Name:	Bayramli
Email:	namaz.bayramli@opensource.az
Password:	Namaz123
Confirm Password:	Namaz123

Save **Cancel**

Ardınca **Connectivity** tab-ı və **SIP Interface** bölümünüə daxil oluruq:



System **Connectivity** **Applications** **Status** **Routing** **Organization** **Media**

Trunk Manager **SIP Interface** **Network List**

Açılan pəncərədə hər bir **SIP** interfeysi seçərək **Edit** düyməsinə sixırıq:

SIP Interfaces					Add SIP Interface
Interface Name	IP Address	Port	Authentication	Actions	
Authenticated SIP	Auto Detect	5060	Required	Edit	Delete
Authenticated SIP - NAT	Auto Detect	5070	Required	Edit	Delete
Unauthenticated SIP	Auto Detect	5080	None	Edit	Delete

Page 1 of 1 | 100 | View 1 - 3 of 3

Authenticated SIP üçün açılan səhifədə **Force Registration Domain** və **Default Incoming Context** üçün **Auto(multitenant)** seçib **Save** düyməsinə sixırıq:

[Edit](#)

SIP Interface Information

SIP Interface Friendly Name:	Authenticated SIP
<input checked="" type="checkbox"/> IP Address to Bind To: Leave blank to auto-detect	<input type="text"/>
Port to Bind To:	5060
SIP Traffic Protocol:	<input checked="" type="radio"/> Both
<input checked="" type="checkbox"/> External SIP IP Address: Leave blank to auto-detect	<input type="text"/>
<input checked="" type="checkbox"/> Log authentication failures:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Use Compact Headers:	<input type="checkbox"/>
Default Interface	<input checked="" type="radio"/> Authenticated SIP

NAT

<input checked="" type="checkbox"/> Server is behind NAT? Detect IP, even when behind NAT	<input checked="" type="checkbox"/> NAT Detection Mechanism: How to detect the public IP
<input checked="" type="checkbox"/> SIP Ping Registered Devices Send registered devices SIP option pings	<input checked="" type="checkbox"/> Aggressive NAT Detection Detect SIP/NAT IP & Port on Registration
<input checked="" type="checkbox"/> Use Network IP & Port for RTP Equivalent to forcing port	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

Network Lists

<input checked="" type="checkbox"/> NAT List: Matches force NAT traversal mechanisms	<input checked="" type="radio"/> Private Network (auto)
<input checked="" type="checkbox"/> Inbound ACL: Matches do not require authentication	<input checked="" type="radio"/> Trunks (auto)
<input checked="" type="checkbox"/> Register ACL: Matches can register with no credentials	<input checked="" type="radio"/> None

Registration

<input checked="" type="checkbox"/> Enable Auth/Challenge? Require a SIP username/password?	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> Allow multiple registrations:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Force Registration Domain Equivalent to forcing port	<input checked="" type="radio"/> Auto (multitenant)

Inbound Calls

Default Incoming Context:	<input checked="" type="radio"/> Auto (multitenant)
---------------------------	---

ODBC Connection

DSN	<input type="text" value="None"/>
-----	-----------------------------------

[Save](#) [Close](#)

Authenticated SIP - NAT üçün açılan səhifədə **Force Registration Domain** və **Default Incoming Context** üçün **Auto(multitenant)** seçib **Save** düyməsinə sixırıq:

Edit

SIP Interface Information

SIP Interface Friendly Name:	Authenticated SIP - NAT
<input checked="" type="checkbox"/> IP Address to Bind To:	Leave blank to auto-detect
Port to Bind To:	5070
SIP Traffic Protocol:	Both
<input checked="" type="checkbox"/> External SIP IP Address:	Leave blank to auto-detect
<input checked="" type="checkbox"/> Log authentication failures:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Use Compact Headers:	<input type="checkbox"/>
Default Interface	Authenticated SIP

NAT

<input checked="" type="checkbox"/> Server is behind NAT?	Detect IP, even when behind NAT
<input checked="" type="checkbox"/> NAT Detection Mechanism:	Detected public IP
<input checked="" type="checkbox"/> SIP Ping Registered Devices	Send registration detection ping attempts
<input checked="" type="checkbox"/> Aggressive NAT Detection	Detect SIP NAT IP & Port on Registration
<input checked="" type="checkbox"/> Use Network IP & Port for RTP	Equivalent to forcing port

Network Lists

<input checked="" type="checkbox"/> NAT List:	Private Network (auto)
<input checked="" type="checkbox"/> Inbound ACL:	None
<input checked="" type="checkbox"/> Register ACL:	None

Registration

<input checked="" type="checkbox"/> Enable Auth/Challenge?	Require a SIP username/password?
<input checked="" type="checkbox"/> Allow multiple registrations:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Force Registration Domain	Auto (multitenant)
Equivalent to forcing port	

Inbound Calls

Default Incoming Context:	Auto (multitenant)
---------------------------	--------------------

ODBC Connection

DSN:	None
------	------

[Save](#)
[Close](#)

Unauthenticated SIP üçün açılan səhifədə **Force Registration Domain** və **Default Incoming Context** üçün **Auto(multitenant)** seçib **Save** düyməsinə sıxırıq:

Edit

SIP Interface Information

SIP Interface Friendly Name:	<input type="text" value="Unauthenticated SIP"/>
<input checked="" type="checkbox"/> IP Address to Bind To: Leave blank to auto-detect	<input type="text" value="5080"/>
Port to Bind To:	<input type="text" value="Both"/>
SIP Traffic Protocol:	<input checked="" type="checkbox"/> External SIP IP Address: Leave blank to auto-detect
<input checked="" type="checkbox"/> Log authentication failures:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Use Compact Headers:	<input type="checkbox"/>
Default Interface	<input type="text" value="Authenticated SIP"/>

NAT

<input checked="" type="checkbox"/> Server is behind NAT? Detect IP, even when behind NAT	<input checked="" type="checkbox"/> NAT Detection Mechanism: <input type="text" value="Detect IP via uPnP"/>
<input checked="" type="checkbox"/> SIP Ping Registered Devices Send registered devices SIP option pings	<input type="checkbox"/>
<input checked="" type="checkbox"/> Aggressive NAT Detection Detect SIP NAT IP & Port on Registration	<input type="checkbox"/>
<input checked="" type="checkbox"/> Use Network IP & Port for RTP Equivalent to forcing port	<input type="checkbox"/>

Network Lists

<input checked="" type="checkbox"/> NAT List: Matches force NAT traversal mechanisms	<input type="text" value="Private Network (auto)"/>
<input checked="" type="checkbox"/> Inbound ACL: Matches do not require authentication	<input type="text" value="None"/>
<input checked="" type="checkbox"/> Register ACL: Matches can register with no credentials	<input type="text" value="None"/>

Registration

<input checked="" type="checkbox"/> Enable Auth/Challenge? Require a SIP Username/Password?	<input type="checkbox"/>
<input checked="" type="checkbox"/> Allow multiple registrations:	<input type="checkbox"/>
<input checked="" type="checkbox"/> Force Registration Domain Equivalent to forcing port	<input type="text" value="Auto (multitenant)"/>

Inbound Calls

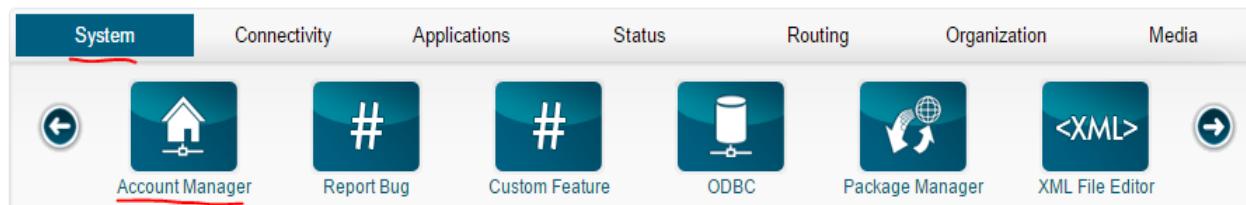
<u>Default Incoming Context:</u>	<input type="text" value="Auto (multitenant)"/>
----------------------------------	---

ODBC Connection

DSN	<input type="text" value="None"/>
-----	-----------------------------------

Save **Close**

Ardınca **System** və **Account Manager** bölümünə daxil oluruq və yaratdığımız istifadəçilərə baxırıq:



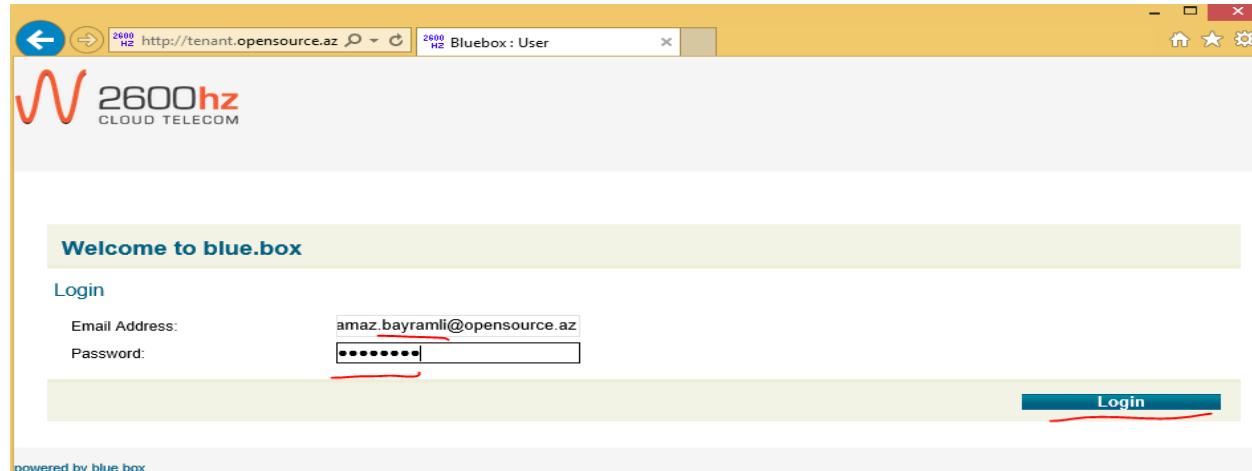
The dashboard includes icons for Account Manager, Report Bug, Custom Feature, ODBC, Package Manager, and XML File Editor. The 'Account Manager' icon is highlighted with a red underline.

Accounts

Name	Type	Created	Actions
Master Account	Normal	2015-08-24 15:50:01	Edit Delete Re
Tenant1	Normal	2015-08-25 03:40:20	Edit Delete Re

Page 1 of 1 100 View 1 - 2 of

Sonra <http://tenant.opensource.az> səhifəsinə daxil olaraq Tenant domain üçün yaratdığımız istifadəçi adı ilə şifrəni daxil edib, **Login** düyməsinə sıxırıq:



FreeBSD əməliyyat sistemində SIPvicious qurulması və sinaqdan keçirilməsi

FreeBSD maşına SIPvicious yüklənməsi və sinaqdan keçirilməsi üçün, aşağıdakı addımları edirik. SVMAP aləti SIPVicious paketinin tərkibində olan biridir. SVMAP bir və ya IP aralığı üçün scan edib fingerprintin təyin edilməsi üçün istifadə edilə bilər. SVMAP imkan yaradır ki, skanda istifadə ediləcək müraciət metodunu təyin edəsiniz. Susmaya görə olan metodu OPTIONS-dur. O bizə hətta debug, verbosity opsiyalarını və SIP mənsəb domain üçün SRV yazılarının skan edilməsinə də izin verir. Siz **./svmmap.py -h** əmri ilə bütün argumentlərin siyahısını əldə edə bilərsiniz.

SIPViviuos programı SIP bazalı VoIP sistemlərin auditini üçün istifadə edilən alətlər toplusudur. Hal-hazırda aşağıdakı alətlərdən ibarət tərkibə sahibdir:

- **svmmap** - bu skannerdir. IP aralığda tapılan SIP alətlərin siyahısını çıxarır.
- **svwar** - PBX-da aktiv genişlənmələri təyin edir.
- **svcrack** - SIP PBX üçün onlayn şifrə sindirandır.

- **svreport** - Sessiyani idarə edir və hesabatları fərqli formatlarda çıxarış edir.
- **svcrash** - Qeydiyyatı olmayan svwar və svcrack scan-ların cəhd'lərini dayandırmaq istəyir.

Endiririk və açırıq:

```
root@odbc:~ # fetch http://sipvicious.googlecode.com/files/sipvicious-0.2.8.tar.gz
sipvicious-0.2.8.tar.gz      100% of    78 kB  454 kBps 00m01s
```

```
root@odbc:~ # tar zxf sipvicious-0.2.8.tar.gz
root@odbc:~ # cd sipvicious-0.2.8
root@odbc:~/sipvicious-0.2.8 # chmod +x *.py
```

FreeSWITCH olan serverimizi skan edirik:

SIP Device	User Agent	Fingerprint
150.170.81.154:5060	FreeSWITCH-mod_sofia/1.5.final+git~20150528T173517z~6a2fc5e0f7~64bit	disabled

1000 və 1019 aralığı üçün istifadəçi adlarını brute-force edib jurnal faylinə yazırıq:

```
# ./svwar.py --force -e 1001-1019 frfs.opensource.az > auth_log.txt
```

FreeSWITCH üzərində səslərin yazılıması

Məqsədimiz FreeSWITCH üzərində bütün zənglərin yazılımasıdır. Bunun üçün, `/usr/local/freeswitch/conf/dialplan/default.xml` faylında `<extension name="Local_Extension">` tag-nı tapırıq və daxili genişlənmələrə aid olan ilk şərt-dən yəni `<condition field="destination_number" expression="^(10[01][0-9]|1100)$">` tag-dan sonra aşağıdakı sətirləri fayla əlavə edirik:

```
<action application="set" data="RECORD_TITLE=Recording ${destination_number} ${caller_id_number} ${strftime(%Y-%m-%d %H:%M)}"/>
<action application="set" data="RECORD_COPYRIGHT=(c) 2015"/>
<action application="set" data="RECORD_SOFTWARE=FreeSWITCH"/>
<action application="set" data="RECORD_ARTIST=FreeSWITCH"/>
<action application="set" data="RECORD_COMMENT=FreeSWITCH"/>
```

```
<action application="set" data="RECORD_DATE=${strftime(%%Y-%%m-%%d %%H:%%M)}"/>
<action application="set" data="RECORD_STEREO=true"/>
<action application="record_session"
data="/usr/local/freeswitch/recording/archive/${strftime(%%Y-%%m-%%d-%%H-%%M-
%%S)}_${destination_number}_${caller_id_number}.wav"/>
```

Bu sətirlərlə deyirik ki, **/usr/local/freeswitch/recording/archive/** qovluğuna il, ay, gün, saat, dəqiqə, saniyə, zəng edilən nömrə və zəng edən nömrə strukturunda **wav** genişlənmə ilə bütün zəngləri qeyd elə.

Ancaq **/usr/local/freeswitch/recording/archive/** qovluğunu yaratmağı unutmayın. Sonda qovluğu yaradaq və freeswitch-ə restart edək:

```
# mkdir -p /usr/local/freeswitch/recording/archive/
# /usr/local/etc/rc.d/freeswitch stop
# /usr/local/etc/rc.d/freeswitch start
```

Yoxlamaq üçün iki genişlənmə arasında zəng edin və səsləri recordings qovluğunda yoxlayın. Aşağıdakı şəkildə fayllar yazılacaq:

```
-rw-r--r-- 1 root wheel 731750 Sep 20 20:13 2015-09-20-20-12-57_1000_1015.wav
-rw-r--r-- 1 root wheel 4879590 Sep 20 20:23 2015-09-20-20-22-02_1015_1000.wav
-rw-r--r-- 1 root wheel 3266150 Sep 20 20:28 2015-09-20-20-26-28_1000_1100.wav
-rw-r--r-- 1 root wheel 2138470 Sep 21 07:52 2015-09-21-07-51-37_1000_1010.wav
-rw-r--r-- 1 root wheel 2313190 Sep 21 07:53 2015-09-21-07-53-11_1010_1000.wav
```

FreeSWITCH recover və Linux HA vasitəsilə Clusterin qurulması

Məqsədimiz FreeSWITCH serverin **mod_sofia recover** funksionallığı vasitəsilə sıradan çıxmış zənglərin bərpa edilməsidir. Sayəsində birinci node-da hər hansıa bir hadisə baş verdiyi halda onun üzərində olan sessiyalar ikinci noda miqrasiya edilir(Diqqət!! RTP-nin proksi edilməsi ilə birlikdə və ya ayrı).

Tətbiqi çox asandır. Bunun üçün kənar bir baza istifadə edilir və SIP stek orda orda saxlanılır. Beləliklə də, **mod_sofia** özünə aid olan bütün məlumatları kənar bazada saxladığına görə, birinci node sıradan çıxdığı halda klasterin ikinci nodunun imkanı var ki, bu quraşdırılmaları oxusun və zəngləri özünə götürsün.

Bizə tələb edilənlər aşağıdakılardır:

- FSNode1 (FreeSWITCH - CentOS 6.7 x64)
 - FSNode2 (FreeSWITCH - CentOS 6.7 x64)
 - MySQL (DB server - FreeBSD 10.1 x64)

İlk işimiz hər üç serverdə reposlar və paketlərin yenilənməsidir:

```
[root@fsnode1 ~]# yum update -y  
[root@fsnode2 ~]# yum update -y  
root@odbc:~ # portsnap fetch extract update
```

Sistemi yenidən yükləyirik:

```
root@odbc:/usr/src # reboot
```

Verilənlər bazası yerləşən serverimizə MySQL yükleyirik quraşdırırıq və yalnız nodlarımızdan qırışə izin veririk.

```
root@odbc:~ # cd /usr/ports/databases/mysql55-server/
root@odbc:~ # make config
```

```
root@odbc:/usr/ports/databases/mysql55-server # make -DBATCH install
```

MysQL-i qurasdırırıq və baza varadırıq:

```
# cp /usr/local/share/mysql/my-large.cnf /etc/my.cnf
```

`/etc/my.cnf` faylında [`mysqld`] başlığına aşağıdaki satırı ekleyerek:

```
root@odbc:~ # touch /var/log/mysql.log  
root@odbc:~ # chown mysql:mysql /var/log/mysql.log
```

```
root@odbc:~ # echo 'mysql_enable="YES"' >> /etc/rc.conf  
root@odbc:~ # /usr/local/etc/rc.d/mysql-server start
```

MySQL-in root istifadəcisinə sifra təyin edirik:

```
MySQL-in root istifadəçisini şifre təyin edirik:  
root@odbc:~ # mysqladmin -u root password freebsd  
root@odbc:~ # mysql -uroot -p  
mysql> CREATE DATABASE freeswitch;  
mysql> grant all privileges on freeswitch.* to freeswitch@150.170.81.131  
identified by 'freebsd';  
mysql> grant all privileges on freeswitch.* to freeswitch@150.170.81.132  
identified by 'freebsd';
```

```
mysql> grant all privileges on freeswitch.* to freeswitch@150.170.81.133
identified by 'freebsd';
```

Təhlükəsizlik üçün FireWALL-imizi aşağıdakı opsiyalarla kompilyasiya edirik(GENERIC faylina **IPFWIREWALL** opsiyaları əlavə edirik):

```
root@odbc:~ # cd /sys/amd64/conf/
root@odbc:/sys/amd64/conf # cat /sys/amd64/conf/GENERIC | grep -i firewall
options          IPFIREWALL
options          IPFIREWALL_VERBOSE
options          IPFIREWALL_VERBOSE_LIMIT=3

root@odbc:/sys/amd64/conf # cd /usr/src/
root@odbc:/usr/src # make buildkernel && make installkernel
```

StartUP quraşdırma faylına aşağıdakı sətirləri əlavə edirik və serveri yenidən yükləyirik:

```
root@odbc:~ # echo 'firewall_enable="YES"' >> /etc/rc.conf
root@odbc:~ # echo 'firewall_type="UNKNOWN"' >> /etc/rc.conf
root@odbc:~ # echo 'firewall_script="/etc/firewall.conf"' >> /etc/rc.conf
```

/etc/firewall.conf faylına aşağıdakı sətirləri əlavə edirik(MySQL-ə girişi yalnız seçilmiş şəbəkə üçün veririk):

```
ipfw add 10 allow ip from any to me dst-port 22 keep-state
ipfw add 20 allow ip from 150.170.81.129/27 to me dst-port 3306 keep-state
```

Sistemi yenidən yükləyirik:

```
root@odbc:/usr/src # reboot
```

Əvvəlki başlıqlarda oxuduğumuz qayda da hər iki node üçün FreeSWITCH-i yükleyirik və quraşdırırıq.

CentOS node-larda Selinux və FireWall-ı söndürürük(Hər iki node-da edirik).

/etc/selinux/config faylında aşağıdakı sətiri uyğun olaraq edirik:

```
SELINUX=disabled
```

CentOS node-larda lazımsız servisləri söndürürük(Hər iki node-da edirik):

```
# chkconfig --level 0123456 postfix off
# chkconfig --level 0123456 kdump off
# chkconfig --level 0123456 cups off
# chkconfig --level 0123456 ip6tables off
# chkconfig --level 0123456 iptables off
```

Hər iki node-da **/etc/hosts** faylı aşağıdakı kimi olacaq:

```
127.0.0.1   localhost localhost.localdomain
150.170.81.131 fsnode1
150.170.81.132 fsnode2
```

Hər iki node-da sistemə yenidən yüklənmə əmri veririk:

```
# reboot
```

Hər iki node üçün ODBC və MySQL connector-u yükleyirik:

```
# yum install mysql-connector-odbc.x86_64 unixODBC.x86_64
```

Hər iki node üçün ODBC yükleyicisini çap edərək informasiyaya baxırıq:

```
# odbcinst -j
unixODBC 2.2.14
DRIVERS.....: /etc/odbcinst.ini
SYSTEM DATA SOURCES: /etc/odbc.ini
FILE DATA SOURCES...: /etc/ODBCDataSources
USER DATA SOURCES...: /root/.odbc.ini
SQLULEN Size.....: 8
SQLLEN Size.....: 8
SQLSETPOSIROW Size.: 8
```

Hər iki node-da **/etc/odbc.ini** faylını aşağıdakı mətnlə edirik:

```
[freeswitch]
Driver = MySQL
TraceFile = stderr
SERVER = 150.170.81.130
PORT = 3306
DATABASE = freeswitch
USER = freeswitch
PASSWORD = freebsd
OPTION = 67108864
```

Hər iki node-da **/etc/odbcinst.ini** faylını aşağıdakı mətnlə edirik:

```
[MySQL]
Description = ODBC for MySQL
Driver = /usr/lib/libmyodbc5.so
Setup = /usr/lib/libodbcmyS.so
Driver64 = /usr/lib64/libmyodbc5.so
Setup64 = /usr/lib64/libodbcmyS.so
FileUsage = 1
UsageCount = 2
Threading = 0
MaxLongVarcharSize = 65536
```

Hər iki node-da aşağıdakı əmrlərlə qoşulmayı yoxlayırıq:

```
# echo "select 1" | isql -v freeswitch
+-----+
| Connected! |
| |
| sql-statement |
| help [tablename] |
| quit |
| |
+-----+
```

```
# isql -v freeswitch
+-----+
| Connected! |
| |
| sql-statement |
| help [tablename] |
| quit |
| |
+-----+
```

```

|                               |
+-----+
SQL> select 1
+-----+
| 1           |
+-----+
| 1           |
+-----+
SQLRowCount returns 1
1 rows fetched
SQL>

```

Biz isteyirik ki, FS-imiz ümumi İP ünvanda işləsin və bunun üçün hər iki node-da **/usr/local/freeswitch/conf/vars.xml** faylina aşağıdakı sətiri əlavə edirik(Həmçinin susmaya görə olan şifrəni dəyişirik):

```

<X-PRE-PROCESS cmd="set" data="local_ip_v4=150.170.81.133"/>
<X-PRE-PROCESS cmd="set" data="default_password=freebsd"/>

```

Recover işləməsi üçün isə SIP profillərdə səslərin izləməsini aktivləşdirmək lazımdır. **/usr/local/freeswitch/conf/sip_profiles/** ünvanına gedirik və hər iki profil(internal,external) üçün, aşağıdakı sətiri əlavə edirik(Hər iki node-da edirik):

```
<param name="track-calls" value="true"/>
```

Hər iki node-da hər iki profil üçün verilənlər bazasında qoşulmayı aşağıdakı sintaksislə təyin edirik(**/usr/local/freeswitch/conf/sip_profiles/** ünvanında **internal.xml** və **external.xml** fayllarında bazaya qoşulmayı quraşdırırıq):

```
<param name="odbc-dsn" value="odbc://freeswitch:freeswitch@freebsd"/>
```

Hər iki node-da **/usr/local/freeswitch/conf/autoload_configs/switch.conf.xml** faylında da aşağıdakı sətirin qarşısından şərhi silirik və uyğun şəklə gətiririk:

```
<param name="core-db-dsn" value="odbc://freeswitch:freeswitch@freebsd"/>
```

Artıq FreeSWITCH-i işə saldıqda avtomatik olaraq bazalar yaradılacaq.

Arping yükleyirik və system control-da bir parametri aktivləşdiririk(Hər iki node-da yükleyirik):

```
# yum install arping2.x86_64
```

Sistem control-u əlavə edirik:

```
# echo 'net.ipv4.ip_nonlocal_bind=1' >> /etc/sysctl.conf
```

Ardınca **HeartBeat** quraşdırırıq. Hər iki node-da heartbeat yükleyirik və quraşdırırıq:

```
# yum install heartbeat.x86_64
```

/etc/ha.d/authkeys faylına aşağıdakı sətirləri əlavə edirik(Hər iki node üçün edirik):

```
auth 1
1 shal mega_super_secure_key
```

Hər iki noda-da açarlara **600** yetkisi təyin edirik:

```
# chmod 600 /etc/ha.d/authkeys

/etc/ha.d/ha.cf faylına aşağıdakı sətirləri əlavə edirik(Hər iki node üçün
edirik):
# Jurnallar
logfacility local0
logfile /var/log/ha-log
# Vaxt bitmələri
keepalive 100ms
deadtime 2
warntime 1
initdead 120
# Node-lar öz aralarında necə danışacaqlar
udpport 694
bcast eth0
# Bizim klasterdə hansı node-lar iştirak edirlər(hostname-lər)
node fsnode1 fsnode2
# Bərpa edildikdən sonra, resurslar birinci node-a qayıtsınmı?
auto_failback on

/etc/ha.d/haresources faylına aşağıdakı sintaksisi əlavə edirik(ikinci node-
da uyğun olaraq fsnode1-i fsnode2 ilə əvəz etmək lazımdır):
fsnode1 IPaddr2::150.170.81.233/32/eth0 freeswitch::fsrecover

freeswitch::fsrecover - init.d skriptinin hissəsidir hansı ki, bizim üçün
profilərimizi işə salacaq. Onu biz aşağıdakı addımla əlavə edirik.

HeartBeat-i hər iki noda-da startup-a əlavə edirik və işə salırıq:
# chkconfig heartbeat on
# service heartbeat start

/etc/init.d/freeswitch skriptində restart və reload arasına aşağıdakı
sətirləri əlavə edirik(Həmçinin echo istifadəsində də əlavə edirik)
fsrecover)
$FS_HOME/bin/fs_cli -x "sofia profile internal start"
$FS_HOME/bin/fs_cli -x "sofia profile external start"
/bin/sleep 1
$FS_HOME/bin/fs_cli -x "sofia recover"
;;
```

Artıq node-larımızdan biri çökdüyü halda, ikincisi ümumi İP-ni işə salacaq və
 FS profiləri yenidən işə salacaq və bundan sonra da zənglərin ələ
 keçirilməsi əmri işləyəcək.

Səhvləri araşdırmaq üçün **/usr/local/freeswitch/log/** ünvanında olan
freeswitch.log faylını araşdırmaq lazımdır.

Mod_Event_Socket dili vasitəsilə PHP və Python-da misallar

Nəzərdə tutulur ki, artıq freeswitch kompilyasiya edilmişdir. Həmçinin **mod_python**, **mod_esl** modulları **/root/src/freeswitch/modules.conf** faylından aktivləşdirilib kompilyasiya edilmişdir.

Qeyd: Ümumiyyətlə API-a qoşulma üçün Python ya da PHP **127.0.0.1** ünvanında **8021**-ci porta **ClueCon** şifrəsi ilə qoşulur. Bu quraşdirmalar isə **/usr/local/freeswitch/conf/autoload_configs/event_socket.conf.xml** faylında edilir. Siz onu dəyişmək istəsəniz, unutmayın ki, hər bir dulin qoşulmalarında da dəyişməlisiniz.

Sınaq üçün consol-dan **8021**-ci porta qoşuluruq və **auth ClueCon** əmri ilə **ClueCon** şifrəsini daxil edirik(**Qeyd:** Şifrəni daxil etdikdən sonra iki dəfə **ENTER** sixmaq lazımdır. Ümumiyyətlə istənilən əmrlərin kombinasiyasını daxil etdikdən sonra iki dəfə **ENTER** sixmaq lazımdır):

```
root@frfs:~ # telnet localhost 8021
```

```
Trying 127.0.0.1...
```

```
Connected to localhost.
```

```
Escape character is '^]'.
```

```
Content-Type: auth/request
```

```
auth ClueCon
```

Eynilə PHP və Python artıq sistemə yüklənmiş vəziyyətdədir.

Aşağıdakı qovluğa daxil oluruq:

```
# cd /root/src/freeswitch/libs/esl/
```

Kompilyasiya edirik:

```
# make
```

PHP-ni kompilyasiya edirik

```
# make phpmod
```

PHP modulu yükleyirik

```
# make phpmod-install
```

Sonra aşağıdakı əmrlə php binar faylinin ünvanını tapırıq:

```
# which php
```

```
/usr/local/bin/php
```

Ardınca **/root/src/freeswitch/libs/esl/php/single_command.php** faylında php shabang-in ünvanı dəyişib **which** əmrindən aldığımız binar **path**(ünvan)-la əvəz edirik:

```
#!/usr/local/bin/php
```

Scripti yerinə yetirən edirik:

```
# chmod +x /root/src/freeswitch/libs/esl/php/single_command.php
```

Nəhayət sonda ESL-ə yetki alırıq:

```
# /root/src/freeswitch/libs/esl/php/single_command.php status
Command to run is: status
UP 0 years, 0 days, 1 hour, 17 minutes, 33 seconds, 841 milliseconds, 115
microseconds
FreeSWITCH (Version 1.5.final 64bit) is ready
2 session(s) since startup
0 session(s) - peak 1, last 5min 0
0 session(s) per Sec out of max 30, peak 1, last 5min 0
1000 session(s) max
min idle cpu 0.00/100.00
Current Stack Size/Max 524288K/524288K
```

Eynilə Python üçün kompilyasiya edək və yükləyək.

```
# cd /src/freeswitch/libs/esl
# make pymod
# make pymod-install
```

Nəticədə python-la da işləməsinə baxırıq:

```
# python /root/src/freeswitch/libs/esl/python/single_command.py -c status
UP 0 years, 0 days, 1 hour, 29 minutes, 41 seconds, 245 milliseconds, 97
microseconds
FreeSWITCH (Version 1.5.final 64bit) is ready
3 session(s) since startup
0 session(s) - peak 1, last 5min 0
0 session(s) per Sec out of max 30, peak 1, last 5min 0
1000 session(s) max
min idle cpu 0.00/100.00
Current Stack Size/Max 524288K/524288K
```

```
# python /root/src/freeswitch/libs/esl/python/single_command.py -c "show
status"
UP 0 years, 0 days, 1 hour, 33 minutes, 5 seconds, 975 milliseconds, 979
microseconds
FreeSWITCH (Version 1.5.final 64bit) is ready
7 session(s) since startup
1 session(s) - peak 1, last 5min 1
0 session(s) per Sec out of max 30, peak 1, last 5min 1
1000 session(s) max
min idle cpu 0.00/100.00
Current Stack Size/Max 524288K/524288K
```

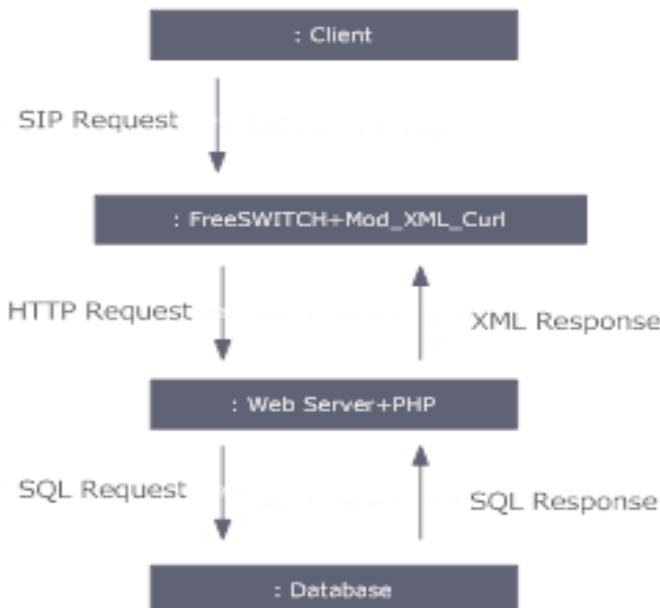
FreeSWITCH SIP istifadəçiləri MySQL-də [Mod_XML_CURL]

mod_xml_curl qısa şəkildə desək "**FreeSWITCH SIP Realtime**" deməkdir. Bu eynilə Asterisk-də olan realtime kimidir hansı ki, SIP istifadəçiləri verilənlər bazasına yeridilir və asterisk daxil olan sip istifadəçilərinin qeydiyyatını verilənlər bazasında aparır.

Dəqiqliklə desək, FreeSWITCH tam da Asterisk kimi işləmir. FreeSWITCH istifadəçilər üçün XML quraşdırımları tələb edir. Ancaq **mod_xml_curl** sayəsində siz SIP istifadəçilərin quraşdırımlarını verilənlər bazasında saxlayaraq eyni işi görə bilərsiniz.

Mətiq çox sadədir. FreeSWITCH-ə daxil olan SIP müraciətləri onun haqqında olan detalları WEB serverdən soruşur. WEB serverdə öz növbəsində verilənlər bazasına müraciət edir və əgər müraciətdə olan istifadəçi məlumatları verilənlər bazasında doğrudursa, web server cavabı XML formatında qaytarır. FreeSWITCH isə XML cavabda olan SIP müraciətə əsaslanaraq qəbul və ya imtina edir. Struktur aşağıdakı şəkildəki kimi olacaq:

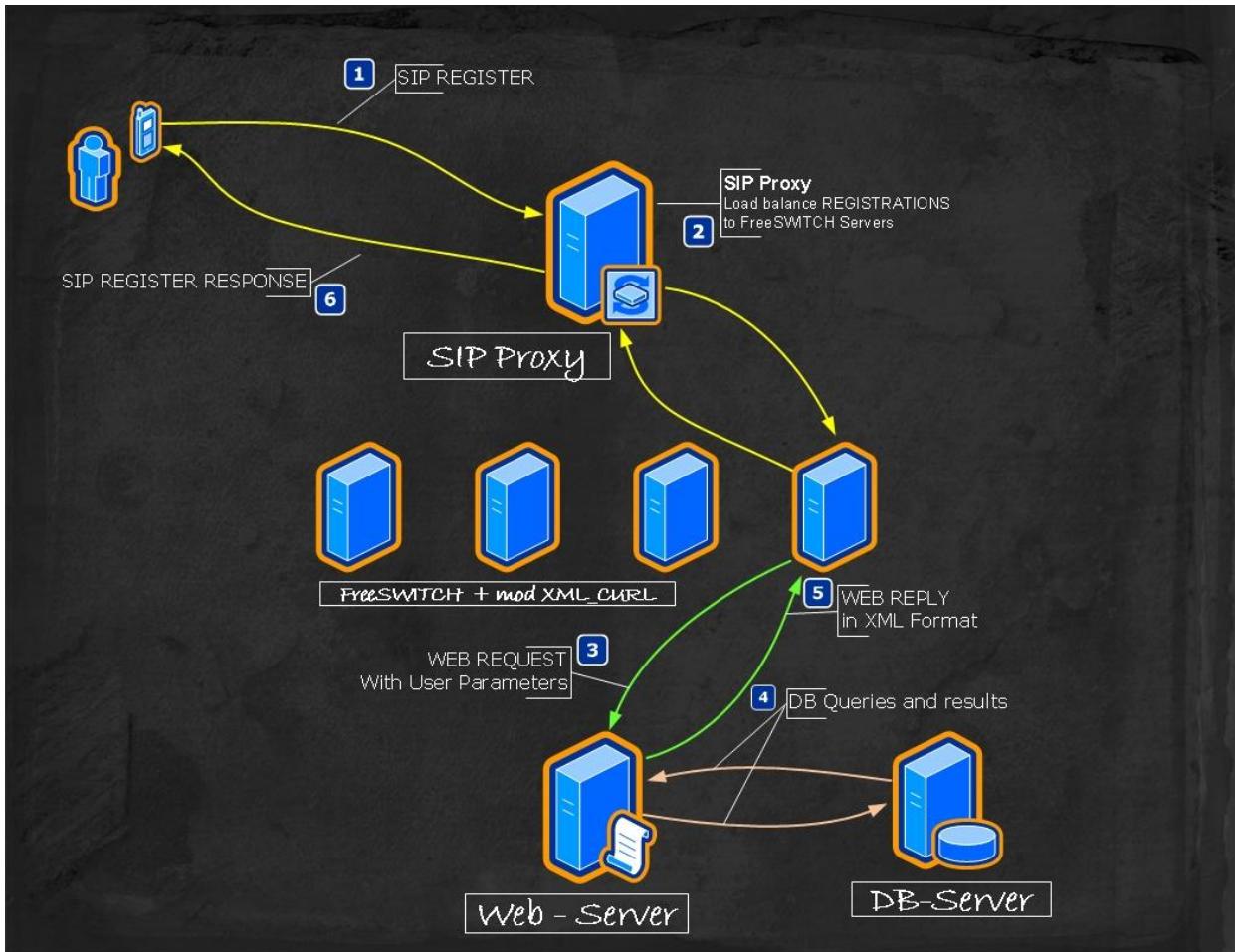
Mod_XML_Curl: during request



Rəsmi səhifəsi

https://freeswitch.org/confluence/display/FREESWITCH/mod_xml_curl ünvanından daha da ətraflı oxuya bilərsiniz.

Aşağıdakı şəkildə daha da başqa üsulla axın göstərilir:



Quraşdirmalarımız CentOS7 üzerinde aparılmıştır. CentOS7 машинимизда DNS adı **xmcurl.opensource.az** və IP ünvanı **150.170.81.135**-dir.

wget və **git**-i yükleyirik:

```
[root@web ~]# yum install wget git
```

CentOS üzerinde Apache MySQL PHP-nin qurulması üçün internetdən rahat şəkildə məlumat əldə edə bilərsiniz. Nəzərdə tutulur ki, artıq LAMP(Linux Apache MySQL PHP) hazırlıdır.

Qeyd: FreeSWITCH-i mütləq **mod_xml_curl** modulla kompilyasiya eləmək lazımdır. Həmçinin **/usr/local/freeswitch/conf/autoload_configs/modules.conf.xml** faylında **<load module="mod_xml_curl"/>** sətirinin qarşısından və sonundan şərhi silmək lazımdır.

Sonra **/usr/local/freeswitch/conf/autoload_configs/xml_curl.conf.xml** faylında aşağıdakı sətirləri uyğun olaraq edirik:

```
<configuration name="xml_curl.conf" description="cURL XML Gateway">
<bindings>
<binding name="example">
```

```

<param name="gateway-url" value="http://xmlcurl.opensource.az/"
bindings="directory"/>
</binding>
</bindings>
</configuration>

```

Artıq **mod_xml_curl** üçün tələb edilən işlək PHP kodları endirək:

```

[root@web src]# cd /root/
[root@web ~]# cd /usr/src/
[root@web src]# git clone https://freeswitch.org/stash/scm/fs/freeswitch-
contrib.git

```

Oeyd: Mənbə kodlarının GIT ünvanını

<https://freeswitch.org/stash/projects/FS/repos/freeswitch-contrib/browse> linkindən əldə edə bilərsiniz.

Endirdiyimiz kodları PUBLIC_HTML qovluğumuza köçürək:

```

[root@web src]# cd /usr/src/freeswitch-contrib/intralanman/PHP/fs_curl/
[root@web fs_curl]# cp -rf * /var/www/html/
[root@web fs_curl]# cd /var/www/html/

```

FS_CURL üçün qlobal quraşdırma faylında verilənlər bazasının quraşdırımlarında və degug quraşdırımlarda dəyişiklik edin.

/var/www/html/globalDefines.php faylında dəyişiklik edərək, posgresql üçün şərh edib MySQL üçün əlavə edirik(Aşağıdakı kimi):

```

// define('DEFAULT_DSN', 'pgsql:dbname=freeswitch;host=127.0.0.1');
define('DEFAULT_DSN', 'mysql:dbname=FS_DB;host=127.0.0.1');
define('DEFAULT_DSN_LOGIN', 'freeswitch');
define('DEFAULT_DSN_PASSWORD', 'freebsd');

```

//Həmçinin Debugları aşağıdakı kimi quraşdırırıq:

```

define('FS_CURL_DEBUG', 9);
define('FS_DEBUG_FILE', '/var/log/fs_curl.debug');

```

Təyin etdiyimiz debug faylini yaradırıq və apache üçün lazımı yetkiləri veririk:

```

[root@web html]# touch /var/log/fs_curl.debug
[root@web html]# chown apache:apache /var/log/fs_curl.debug

```

Verilənlər bazasını yaradaq və daxili strukturunu düzəldək(Eynilə cədvələ qoşulacaq domain, istifadəçi adı və şifrə əlavə edək):

```

[root@web html]# cd /var/www/html/sql/
[root@web sql]# mysql -uroot -pmysqlpss
MariaDB [(none)]> create database FS_DB;
MariaDB [(none)]> GRANT ALL PRIVILEGES on FS_DB.* to freeswitch@localhost
identified by 'freebsd';

```

```

MariaDB [(none)]> use FS_DB
Database changed
MariaDB [FS_DB]> \. mysql-5.0-with-samples.sql

```

```
MariaDB [FS_DB]> insert into directory_domains values (3,'150.170.81.137');
MariaDB [FS_DB]> INSERT into directory (username, domain_id) VALUES
("19881",3);
Query OK, 1 row affected, 1 warning (0.00 sec)
```

19881 nömrəli yaratdığımız istifadəçinin hansı id-də yerləşməsi üçün **directory** cədvəlini select edib baxırıq:

```
MariaDB [FS_DB]> select * from directory;
+----+-----+-----+
| id | username | domain_id | cache |
+----+-----+-----+
| 1  | 1000    | 1          | 0      |
| 2  | 1001    | 2          | 0      |
| 3  | 1002    | 1          | 0      |
| 5  | 1003    | 2          | 0      |
| 6  | 1004    | 1          | 0      |
| 7  | 1005    | 2          | 0      |
| 8  | 1006    | 1          | 0      |
| 9  | 1007    | 2          | 0      |
| 10 | 2000   | 1          | 0      |
| 11 | 1009    | 2          | 0      |
| 12 | 19881  | 3          | 0      |
+----+-----+-----+
11 rows in set (0.00 sec)
```

12 ID-li istifadəçi üçün **freebsd** şifrəsi təyin edirik:

```
MariaDB [FS_DB]> insert into directory_params
(directory_id,param_name,param_value) VALUES (12,'password','freebsd');
Query OK, 1 row affected (0.00 sec)
```

Bununla da bitdi, sonda serverimizi yenidən yükleyirik və **fs_cli**-a daxil olub, **XML_CURL_DEBUG**-i aktiv edirik:
`[root@web sql]# reboot`

FreeSWITCH consola daxil oluruq və xml debug aktiv edirik:

```
[root@web log]# fs_cli
freeswitch@internal> xml_curl debug_on
OK
```

Artıq istənilən soft telefonla SQL-də yeni yaratdığımız istifadəçi adı və şifrə ilə FreeSWITCH serverimizə qoşulmağa çalışın və eynilə WEB serverinizdə də jurnalları analiz edin.

Səhvləri aşağıdakı fayllarda axtarırıq:

```
[root@web log]# cat /usr/local/freeswitch/log/freeswitch.log | grep -i err
[root@web log]# cat /var/log/fs_curl.debug
[root@web log]# tail -f /var/log/httpd/error_log
CHANGE
[ON LINE:88]
```

FROM:\$where_array [] = sprintf ("domain_id='%s'", \$domain ['id']);

```

TO:$where_array [] = sprintf ( "domain='%s'", $domain ['id'] );

[LINE:106]
FROM:$query = sprintf ( "SELECT * FROM directory d %s %s ORDER BY username",
$join_clause, $where_clause );
TO:$query = sprintf ( "SELECT * FROM directory %s %s ORDER BY username",
$join_clause, $where_clause );

```

Save and Exit.

FreeSWITCH vasitələ Mod_callcenter üzərində növbələmə

FreeSWITCH üzərində zənglərin növbələnməsinin **mod_callcenter** vasitəsilə qurulması heç də çətin deyil amma biraz qarışqlıq gətirib çıxara bilər. Bunun üçün **support**, **accounts** və **reception** növbələri qururuq. Bu növbələrin üzvləri isə **101**, **102** və **103** olacaq.

İlk işimiz növbələri
/usr/local/freeswitchconf/autoload_configs/callcenter.conf.xml faylında təyin etmək olacaq.

```

<configuration name="callcenter.conf" description="CallCenter">
  <settings>
  </settings>
  <queues>
    <queue name="opsosupport@default">
      <param name="strategy" value="top-down"/>
      <param name="moh-sound" value="$$ {hold_music} "/>
      <param name="time-base-score" value="system"/>
      <param name="max-wait-time" value="0"/>
      <param name="max-wait-time-with-no-agent" value="0"/>
      <param name="max-wait-time-with-no-agent-time-reached"
value="20"/>
      <param name="tier-rules-apply" value="false"/>
      <param name="tier-rule-wait-second" value="300"/>
      <param name="tier-rule-wait-multiply-level" value="true"/>
      <param name="tier-rule-no-agent-no-wait" value="false"/>
    
```

```

<param name="discard-abandoned-after" value="60"/>
<param name="abandoned-resume-allowed" value="false"/>
</queue>
<queue name="opsoreception@default">
<param name="strategy" value="top-down"/>
<param name="moh-sound" value="$$\{hold_music\}"/>
<param name="time-base-score" value="system"/>
<param name="max-wait-time" value="0"/>
<param name="max-wait-time-with-no-agent" value="0"/>
<param name="max-wait-time-with-no-agent-time-reached"
value="20"/>
<param name="tier-rules-apply" value="false"/>
<param name="tier-rule-wait-second" value="300"/>
<param name="tier-rule-wait-multiply-level" value="true"/>
<param name="tier-rule-no-agent-no-wait" value="false"/>
<param name="discard-abandoned-after" value="60"/>
<param name="abandoned-resume-allowed" value="false"/>
</queue>
<queue name="opsoaccounts@default">
<param name="strategy" value="top-down"/>
<param name="moh-sound" value="$$\{hold_music\}"/>
<param name="time-base-score" value="system"/>
<param name="max-wait-time" value="0"/>
<param name="max-wait-time-with-no-agent" value="0"/>
<param name="max-wait-time-with-no-agent-time-reached"
value="20"/>
<param name="tier-rules-apply" value="false"/>
<param name="tier-rule-wait-second" value="300"/>
<param name="tier-rule-wait-multiply-level" value="true"/>
<param name="tier-rule-no-agent-no-wait" value="false"/>
<param name="discard-abandoned-after" value="60"/>
<param name="abandoned-resume-allowed" value="false"/>
</queue>
</queues>
</configuration>

```

Gördüyümüz kimi, biz **default** kontekst istifadə edirik və bizim növbələrimiz **opsosupport**, **opso**reception**** və **opso**accounts**** adlanır.

Açar quraşdırımlar aşağıdakılardır:

name - Növbənin adı

strategy - Bu metoddur hansı ki, növbə növbəti agenti təyin edir ki, növbəni ona təqdim eləsin. Bu strategiyaların siyahısı və onların özlərinin necə aparmasını siz rəsmi

http://wiki.freeswitch.org/wiki/Mod_callcenter#Distribution_Strategy səhifəsindən eldə edə bilərsiniz.

moh-sound - Növbədə olan zəng edənlər üçün, gözləmədə öxunacaq musiqini təyin edir. Bu halda biz öz **/usr/local/freeswitch/conf/vars.xml** faylında öncədən təyin elədiyimiz **hold_music** dəyişənidən istifadə edirik. Bütün opsiyalar haqqında məlumatları rəsmi səhifədən

http://wiki.freeswitch.org/wiki/Mod_callcenter#Queue_options eldə edə bilərsiniz.

Sonra isə agentlərin quraşdırımları təyin edilməlidir:

```
<agents>
    <agent name="101" type="callback" contact="[call_timeout=60]user/101"
status="Logged Out" max-no-answer="3" wrap-up-time="10" reject-delay-
time="30" busy-delay-time="60" no-answer-delay-time="10" />
    <agent name="102" type="callback" contact="[call_timeout=60]user/102"
status="Logged Out" max-no-answer="3" wrap-up-time="10" reject-delay-
time="30" busy-delay-time="60" no-answer-delay-time="10" />
    <agent name="103" type="callback" contact="[call_timeout=60]user/103"
status="Logged Out" max-no-answer="3" wrap-up-time="10" reject-delay-
time="30" busy-delay-time="60" no-answer-delay-time="10"/>
</agents>
```

Agentlər üçün açar quraşdırma opsiyaları aşağıdakılardır:

name - Agentin adı. Bu istənilən ad ola bilər ki, sizin işinizdə daha rahat anlaşılan ad olsun. Asan olması agentlər üçün də, **login/logout** düymələrinin SPA telefonlarda istifadə elədikdə daha asan olacaq. Bizim halda sadəcə genişlənmə olaraq saxladıq.

contact - Agentə çatmaq üçün zəng etmək sətiri(Yəni agentə necə zəng etmək lazımdır). Nəzərə alın ki, **call_timeout** mənəsi sizin zəng üçün agentin nə qədər uzun zaman müddətində gözləməsini təyin edir. Bizim halda agent zəngə cavab vermesi üçün 60 saniyəsi var.

status - Bu agent yaradılması statusudur. Məsləhətdir ki, siz öz agentlərinizi **logged out** kimi qeyd edəsiniz və sonra onların özlərinə şərait yaradasınız ki, novbəyə **available** statusla gire bilsinlər.

max-no-answer - Agent buna bir neçə dəfə cəhd edəcək və bu saydan sonra kəsinti olaraq qeydə alınacaq.

wrap-up-time - Agentə yeni zəngə cavab verilməsindən əvvəl bu təqdim ediləcək. Bu halda agentə bir zəng ilə növbəti növbə arasında verilən zaman 10 saniyə olacaq.

reject-delay-time - Əgər agent zəngləri özü rədd edirsə onda, bu həmin vaxtdır hansı ki, növbəti zəng ona təqdim edilməzdən öncə istifadə ediləcək.

Bütün opsiyaları http://wiki.freeswitch.org/wiki/Mod_callcenter#Agent_options linkindən əldə edə bilərsiniz.

Sonra biz təqdim etməliyik ki, hansı agent hansı növbədə iştirak etməlidir:

```
<tiers>
    <tier agent="101" queue="opsosupport@default" level="1" position="1"/>
    <tier agent="101" queue="opsoaccounts@default" level="1" position="2"/>
    <tier agent="101" queue="opsoreception@default" level="1" position="1"/>

    <tier agent="102" queue="opsosupport@default" level="1" position="3"/>
    <tier agent="102" queue="opsoaccounts@default" level="1" position="3"/>
    <tier agent="102" queue="opsoreception@default" level="1" position="3"/>

    <tier agent="103" queue="opsosupport@default" level="1" position="2"/>
    <tier agent="103" queue="opsoaccounts@default" level="1" position="1"/>
    <tier agent="103" queue="opsoreception@default" level="1" position="2"/>
</tiers>
```

Açar quraşdirmalar aşağıdakılardır:

agent - Agentin adı

queue - Agent iştirakçı kimi, olacaq növbənin adı

position - Burda yuxarıdan aşağı gedən strategiya istifadə edilir.

Nəticədə ümumi quraşdırma faylı aşağıdakı kimi olacaq:

```
<configuration name="callcenter.conf" description="CallCenter">
    <settings>
    </settings>
    <queues>
        <queue name="opsosupport@default">
            <param name="strategy" value="top-down"/>
            <param name="moh-sound" value="$$\{hold_music\}"/>
            <param name="time-base-score" value="system"/>
            <param name="max-wait-time" value="0"/>
            <param name="max-wait-time-with-no-agent" value="0"/>
            <param name="max-wait-time-with-no-agent-time-reached"
value="20"/>
            <param name="tier-rules-apply" value="false"/>
            <param name="tier-rule-wait-second" value="300"/>
            <param name="tier-rule-wait-multiply-level" value="true"/>
            <param name="tier-rule-no-agent-no-wait" value="false"/>
            <param name="discard-abandoned-after" value="60"/>
            <param name="abandoned-resume-allowed" value="false"/>
        </queue>
        <queue name="opsoreception@default">
            <param name="strategy" value="top-down"/>
            <param name="moh-sound" value="$$\{hold_music\}"/>
            <param name="time-base-score" value="system"/>
            <param name="max-wait-time" value="0"/>
            <param name="max-wait-time-with-no-agent" value="0"/>
            <param name="max-wait-time-with-no-agent-time-reached"
value="20"/>
            <param name="tier-rules-apply" value="false"/>
            <param name="tier-rule-wait-second" value="300"/>
            <param name="tier-rule-wait-multiply-level" value="true"/>
            <param name="tier-rule-no-agent-no-wait" value="false"/>
            <param name="discard-abandoned-after" value="60"/>
            <param name="abandoned-resume-allowed" value="false"/>
        </queue>
        <queue name="opsoaccounts@default">
            <param name="strategy" value="top-down"/>
            <param name="moh-sound" value="$$\{hold_music\}"/>
            <param name="time-base-score" value="system"/>
            <param name="max-wait-time" value="0"/>
            <param name="max-wait-time-with-no-agent" value="0"/>
            <param name="max-wait-time-with-no-agent-time-reached"
value="20"/>
            <param name="tier-rules-apply" value="false"/>
            <param name="tier-rule-wait-second" value="300"/>
            <param name="tier-rule-wait-multiply-level" value="true"/>
            <param name="tier-rule-no-agent-no-wait" value="false"/>
            <param name="discard-abandoned-after" value="60"/>
            <param name="abandoned-resume-allowed" value="false"/>
        </queue>
    </queues>
</configuration>
```

```

    </queue>
</queues>
<agents>
    <agent name="101" type="callback" contact="[call_timeout=60]user/101"
status="Logged Out" max-no-answer="3" wrap-up-time="10" reject-delay-
time="30" busy-delay-time="60" no-answer-delay-time="10" />
    <agent name="102" type="callback" contact="[call_timeout=60]user/102"
status="Logged Out" max-no-answer="3" wrap-up-time="10" reject-delay-
time="30" busy-delay-time="60" no-answer-delay-time="10" />
    <agent name="103" type="callback" contact="[call_timeout=60]user/103"
status="Logged Out" max-no-answer="3" wrap-up-time="10" reject-delay-
time="30" busy-delay-time="60" no-answer-delay-time="10"/>
</agents>
<tiers>
    <tier agent="101" queue="opsosupport@default" level="1"
position="1"/>
    <tier agent="101" queue="opsoaccounts@default" level="1"
position="2"/>
    <tier agent="101" queue="opsoreception@default" level="1"
position="1"/>
    <tier agent="102" queue="opsosupport@default" level="1"
position="3"/>
    <tier agent="102" queue="opsoaccounts@default" level="1"
position="3"/>
    <tier agent="102" queue="opsoreception@default" level="1"
position="3"/>
    <tier agent="103" queue="opsosupport@default" level="1"
position="2"/>
    <tier agent="103" queue="opsoaccounts@default" level="1"
position="1"/>
    <tier agent="103" queue="opsoreception@default" level="1"
position="2"/>
</tiers>
</configuration>

```

Sonra bize lazımdır ki, zengləri növbənin içiñə salaq. Bizim İVR vardır hansı ki, mod_callcenter programını çağırın zengləri genişlənmələrə bölüsdürür. Bizim **/usr/local/freeswitch/conf/ivr_menus/Level1Opso.xml** İVR menyusu aşağıdakı kimi olacaq:

```

<include>
    <menu name="Level1Opso"
        greet-long="say:Welcome to One Metric. Press 1 for support, 2
for accounts, 3 for reception or 4 to join a conference call"
        greet-short="say:Welcome to One Metric. Press 1 for support,
2 for accounts, 3 for reception or 4 to join a conference call"
        invalid-sound="ivr/ivr-that_was_an_invalid_entry.wav"
        exit-sound="voicemail/vm-goodbye.wav"
        confirm-macro=""
        confirm-key=""
        tts-engine="flite"
        tts-voice="slt"
        confirm-attempts="3"
        timeout="3000"
        inter-digit-timeout="2000"
        max-failures="3"
        max-timeouts="3"
    </menu>
</include>

```

```

        digit-len="4">

    <entry action="menu-exec-app" digits="1" param="transfer 450
XML default"/>      <!-- Support -->
    <entry action="menu-exec-app" digits="2" param="transfer 451
XML default"/>      <!-- Accounts -->
    <entry action="menu-exec-app" digits="3" param="transfer 452
XML default"/>      <!-- Reception -->

    </menu>
</include>
```

Gördüyünüz kimi, zənglər seçilmiş opsiyaya əsaslanaraq **450**, **451** və **452** genişlənməlirnə təyin edilir. Bizim **/usr/local/freeswitch/conf/dialplan/default.xml** quraşdırma faylında aşağıdakı genişlənmə qurulması var.

```

<!-- One Metric Support -->
    <extension>
        <condition field="destination_number" expression="^(450)$">
            <action application="set" data="caller_id_name=One
Metric Support" />
            <action application="set" data="call_timeout=60" />
            <action application="set" data="originate_timeout=60" />
        </condition>
    </extension>

    <!-- One Metric Accounts -->
    <extension>
        <condition field="destination_number" expression="^(451)$">
            <action application="set" data="caller_id_name=One
Metric Accounts" />
            <action application="callcenter"
data="opsoaccounts@default"/>
        </condition>
    </extension>

    <!-- One Metric Reception -->
    <extension>
        <condition field="destination_number" expression="^(452)$">
            <action application="set" data="caller_id_name=One
Metric Reception" />
            <action application="callcenter"
data="opsoreception@default"/>
        </condition>
    </extension>
```

Biz faylinda **caller_id_name** dəyişənini dəyişirik. Bu telefona görə təyin edilir və beləliklə də agent zəngin hansı növbədən gəlməsini təyin edə biləcək.

Sonra biz agentlərə çatacaq ünvana sahib olmalıdır ki, növbəni qeydə ala bilək. Biz **/usr/local/freeswitch/conf/dialplan/01_Custom.xml** faylında

300 genişlənməsini təyin edirik ki, agentin girişini(log the agent in) təyin edək və **301** təyin edirik ki, agentin çıkışını(log the agent out) təyin edək.

```

<!-- Log Out of the Call Queues -->
<extension name="agent_login">
    <condition field="destination_number" expression="^300$">
        <action application="set" data="res=${callcenter_config(agent
set status ${caller_id_number} 'Available')}" />
        <action application="answer" data="" />
        <action application="sleep" data="500" />
        <action application="playback" data="ivr/ivr-
you_are_now_logged_in.wav"/>
        <action application="hangup" data="NORMAL_CLEARING" />
    </condition>
</extension>

<!-- Log Into the Call Queues -->
<extension name="agent_logoff">
    <condition field="destination_number" expression="^301$">
        <action application="set" data="res=${callcenter_config(agent
set status ${caller_id_number} 'Logged Out')}" />
        <action application="answer" data="" />
        <action application="sleep" data="500" />
        <action application="playback" data="ivr/ivr-
you_are_now_logged_out.wav"/>
        <action application="hangup" data="" />
    </condition>
</extension>
```

Artıq biz telefonlarımıza aşağıdaki genişlənmiş funksiyaları əlavə etməklə zəng növbələrində liniya düymələri olan **log in** və **logout**-a sahib olacaq.

Login - fnc=blf+sd+cp;sub=300@\$PROXY;ext=300@\$PROXY

Logout - fnc=blf+sd+cp;sub=301@\$PROXY;ext=301@\$PROXY

A Division of Cisco Systems, Inc.

Linksys Telephone Configuration

Phone Ext 1 Ext 2 Ext 3 Ext 4 Ext 5 Ext 6 User SPA932

User Login basic | advanced
Personal Directory Call History
SPA932 Status

General

Station Name: Peter Blackford Voice Mail Number:

Text Logo:

BMP Picture Download URL: tftp://192.168.1.20/PhoneBackground.bmp

Select Logo: BMP Picture Select Background Picture: BMP Picture

Screen Saver Enable: no Screen Saver Wait: 300

Screen Saver Icon: Power Save Screen

Line Key 1

Extension: 1 Short Name: Peter Blackford

Share Call Appearance: private

Extended Function:

Line Key 2

Extension: 1 Short Name: Peter Blackford

Share Call Appearance: private

Extended Function:

Line Key 3

Extension: Disabled Short Name: Login

Share Call Appearance: shared

Extended Function: fnc=blf+sd+cp;sub=300@\$PROXY;ext=300@\$PROXY

Line Key 4

Extension: Disabled Short Name: Logout

Share Call Appearance: shared

Extended Function: fnc=blf+sd+cp;sub=301@\$PROXY;ext=301@\$PROXY

Line Key 5

Extension: Disabled Short Name: \$USER

Share Call Appearance: private

Extended Function:

Line Key 6

Extension: Disabled Short Name: \$USER

Share Call Appearance: private

Extended Function:

Miscellaneous Line Key Settings

SCA Line ID Mapping: Vertical First SCA Barge-In Enable: no

Line Key LED Pattern

Idle LED: Remote Undefined LED:
 Local Seized LED: Remote Seized LED:

Dəyişiklikləri etdinən sonra işə, **fs_cli**-dan **reload mod_callcenter** əmrini işə salmalısınız ki, dəyişikliklər qüvvəyə minsin:
freeswitch@internal> reload mod_callcenter

Zəng növbələrini çap et:

```
freeswitch@internal> callcenter_config queue list
name|strategy|moh_sound|time_base_score|tier_rules_apply|tier_rule_wait_second|tier_rule_wait_multiply_level|tier_rule_no_agent_no_wait|discard_abandoned_after|abandoned_resume_allowed|max_wait_time|max_wait_time_with_no_agent|max_wait_time_with_no_agent_time_reached|record_template
support@default|longest-idle-
agent|/usr/local/freeswitch/sounds/music/custom1/ring.wav|system|false|300|true|false|600|true|0|0|5|/usr/local/freeswitch/recordings/archive/${strftime(%Y-%m-%d-%H-%M-%S)}.${destination_number}.${caller_id_number}.${uuid}.wav
+OK
```

Növbətə olan zəngləri çap et:

```
freeswitch@internal> callcenter_config queue list members opsosupport@default
```

```
queue|system|uuid|session_uuid|cid_number|cid_name|system_epoch|joined_epoch|
rejoined_epoch|bridge_epoch|abandoned_epoch|base_score|skill_score|serving_ag-
ent|serving_system|state|score
support@default|single_box|2e054006-204c-e611-887f-
005056802a6a||994222545852|994222545852|1468761097|1468761097|0|0|1468761099|
0|0|||Abandoned|569
support@default|single_box|6c73a614-204c-e611-887f-
005056802a6a||994516095851|994516095851|1468761122|1468761122|0|0|1468761123|
0|0|||Abandoned|544
support@default|single_box|43b50c33-214c-e611-887f-
005056802a6a||994508906353|994508906353|1468761602|1468761602|0|0|1468761613|
0|0|||Abandoned|64
support@default|single_box|b2044d57-214c-e611-887f-005056802a6a|82e43057-
214c-e611-887f-
005056802a6a|994553132466|994553132466|1468761663|1468761663|0|1468761666|0|0
|0|1053|single_box|Answered|3
+OK
```

Bütün agentləri çap elə:

```
freeswitch@internal> callcenter_config agent list
name|system|uuid|type|contact|status|state|max_no_answer|wrap_up_time|reject_
delay_time|busy_delay_time|no_answer_delay_time|last_bridge_start|last_bridge_
end|last_offered_call|last_status_change|no_answer_count|calls_answered|talk_
time|ready_time
1000|single_box||callback|[call_timeout=30]user/1000|Logged
Out|Waiting|3|10|30|60|0|0|0|0|1464697863|0|0|0|0
1001|single_box||callback|[call_timeout=30]user/1001|Logged
Out|Waiting|3|10|30|60|0|1465101299|1465101506|1465101296|1468655398|0|4|219|
1465100385
1002|single_box||callback|[call_timeout=30]user/1002|Logged
Out|Waiting|3|10|30|60|0|1465102796|1465102800|1465102796|1465102809|0|8|50|1
465101266
+OK
```

Növbədə olan bütün agentləri çap elə:

```
freeswitch@internal> callcenter_config tier list agents
queue|agent|state|level|position
support@default|1000|Ready|1|1
support@default|1067|Active Inbound|1|1
support@default|1068|Active Inbound|1|1
support@default|1075|No Answer|1|1
+OK
```

API vasitəsilə FreeSWITCH serverdən statusların alınması

Məqsədimiz FreeSWITCH serverdən API vasitəsilə statistikaların alınmasıdır. Bu işi Python kitabxanası **xmlrpclib** sayəsində edəcəyik. Bəllidir ki, biz artıq FreeSWITCH-in **mod_xml_rpc** modulu ilə tanışıq. Hal-hazırda mod_xml_rpc istifadə edərək Python vasitəsilə FreeSWITCH serverimizdən status məlumatları əldə edəcəyik.

mod_xml_rpc-ni işə salmaq üçün işə **/usr/local/freeswitch/conf/autoload_configs/xml_rpc.conf.xml** faylında aşağıdakı sətirlər uyğun olmalıdır. Göründüyü kimi istifadəçi adı **freeswitch** və şifrə **works** istəfadə edilir. Serverimiz bütün IP ünvanlarda **8080** portunda qulaq asacaq.

```
<configuration name="xml_rpc.conf" description="XML RPC">
  <settings>
    <param name="http-port" value="8080" />
    <param name="auth-realm" value="freeswitch"/>
    <param name="auth-user" value="freeswitch" />
    <param name="auth-pass" value="works" />
  </settings>
</configuration>
```

Öncə python kodunu istifadə edəcəyimiz serverə **python2.7**-ni və **git** yükleyirik. GIT bizi istifadə edəcəyimiz kodun endirilməsində lazım olacaq.

```
# pkg install -y python27
# python2.7 -m ensurepip
# pkg install -y git
```

Ardınca aşağıdakı anbarı özümüzə endiririk.

```
# git clone https://github.com/areski/freeswitch-telegraf-plugin.git
```

Endirdiyimiz ünvana daxil oluruz və python kodu yerinə yetirilən edirik.

```
# cd freeswitch-telegraf-plugin/
# chmod +x freeswitch_metrics.py
```

Kodun tərkibi aşağıdakı kimidir:

```
# cat freeswitch_metrics.py
#!/usr/bin/env python
"""

FreeSWITCH Metrics
```

This script collects the following FreeSWITCH metrics:

- total of active channels
- total of calls
- Current CPS

Usage:

```

export FS_HOST=localhost; export FS_PORT=8080; export
FS_USERNAME=freeswitch; export FS_PASSWORD=works
./freeswitch_metrics.py

"""

from xmlrpclib import ServerProxy
from datetime import datetime
import re
import os

# Connectiong settings for FreeSWITCH XML RPC
FS_HOST = os.getenv('FS_HOST', 'localhost')
FS_PORT = os.getenv('FS_PORT', 8080)
FS_USERNAME = os.getenv('FS_USERNAME', 'freeswitch')
FS_PASSWORD = os.getenv('FS_PASSWORD', 'works')

DEMO_MODE = False

server = ServerProxy("http://%s:%s@%s:%s" % (FS_USERNAME,
                                              FS_PASSWORD,
                                              FS_HOST,
                                              FS_PORT))

def get_calls():
    """
    Connect to FreeSWITCH server and get calls count

    fs_cli -x "show channels count"

    0 total.

    """
    calls = server.freeswitch.api("show", "calls count")
    num_calls = r'\n(?P<chans>\d+) total.\n'
    regexp = re.compile(num_calls)
    matches = regexp.search(calls)
    return int(matches.group("chans"))

def get_channels_cps():
    """
    Connect to FreeSWITCH server and get channels count & cps

    fs_cli -x "show status"
    UP 0 years, 9 days, 2 hours, 56 minutes, 28 seconds, 402 milliseconds,
    835 microseconds
    FreeSWITCH (Version 1.6.6 -13-d2d0b32 64bit) is ready
    5893 session(s) since startup
    0 session(s) - peak 7, last 5min 1
    0 session(s) per Sec out of max 60, peak 6, last 5min 1
    1200 session(s) max
    min idle cpu 0.00/97.17
  
```

```

Current Stack Size/Max 240K/8192K
"""
status = server.freeswitch.api("show", "status")
# lines = status.splitlines()
# lines[5]
num_channels = r'(?P<sessions>\d+) \w.* - peak'
regexp = re.compile(num_channels)
matches = regexp.search(status)
sessions = int(matches.group("sessions"))
num_cps = r'(?P<cps>\d+) \w.* per Sec'
regexp = re.compile(num_cps)
matches = regexp.search(status)
cps = int(matches.group("cps"))
return (sessions, cps)

def print_statics():
    try:
        n_calls = get_calls()
        (sessions, cps) = get_channels_cps()
    except:
        n_calls = 0
        sessions = 0
        cps = 0
    print("{\"active_channels\": %d, \"active_calls\": %d, \"cps\": %d}" %
(sessions, n_calls, cps))

if __name__ == "__main__":
    if DEMO_MODE:
        dt = datetime.now()
        n_calls = dt.minute
        sessions = dt.minute
        cps = dt.second
        print("{\"active_channels\": %d, \"active_calls\": %d, \"cps\": %d}" %
(sessions, n_calls, cps))
    else:
        print_statics()

```

Öncə kodu tam analiz edək.

Kod **FS_HOST**, **FS_PORT**, **FS_USERNAME**, **FS_PASSWORD** dəyişənləri istifadə edərək serverimizə qoşulur və **get_calls()**, **get_channels_cps()** funksiyalarını **print_statics()** funksiyası vasitəsilə çağırır. Və bunun sayəsində API-a qoşulub statusları əldə edir.

Qoşulma dəyişənlərini siz birbaşa faylin içində ya da istifadəçi adının ev qovluguñda istifadə edilən shell-də təyin edə bilərsiniz. Mənim halimdə aşağıdakı kimi idi:

```
# cat /root/.bashrc | grep FS
```

```
export FS_HOST="10.100.10.100"; export FS_PORT=8080; export
FS_USERNAME="freeswitch"; export FS_PASSWORD="works"
```

Sonda kodumuzu işə saldıqda aşağıdakı nəticəni əldə edirik:
fpyvenv freeswitch-call-metrics # **./freeswitch_metrics.py**
{"active_channels": **2**, "active_calls": **1**, "cps": 0}

Bu o deməkdir ki, iki istifadəçi bir-birlərinə zəng ediblər və burda **2** aktiv kanal və **1** aktiv zəng var.

Notice

Bu başlıqda biz çoxlu tərcübələr etdik hansı ki, öncəki başlıqlarda onlar haqqında nəzəri olaraq ətraflı danışmışdıq. Bunlara misal Multitenant, SIP sınaqdan kecirilmə, növbələnmə və FS HA təcrübədə olanı demək olar.

A

FreeSWITCH Onlayn cəmiyyəti

Açıq mənbə koduya program təminatının çox layihələri haqqında yaxşı şeylərdən biri odur ki, bütün dünyadan insanlar müntəzəm əsasda birləşirlər, Heç bir təmənna olmadan, şəxsi istəkləri ilə işləyirlər. FreeSWITCH bunlardan biridir.

Bu başlıqda biz onlayn cəmiyyətin əksər aspektlərini açıqlayacaq. Onlar aşağıdakılardır:

- FreeSWITCH mail siyahıları
- IRC vasitəsilə real vaxtda əlaqə
- Əsas FreeSWITCH saytı və wiki
- Hər il keçirilən ClueCon açıq qaynaqlı yaradıcı konfransı

Bu başlıq sizə dünya cəmiyyətinə qoşulmanın rahat üsullarını açıqlayır.

FreeSWITCH mail siyahıları

FreeSWITCH proektinin əsas mail siyahıları <http://lists.freeswitch.org/mailman/listinfo> ünvanındadır. Əksər istifadəçilərin əsas siyahısı uyğun olaraq, **freeswitch-users** adlanır. Əksər proektlərdə olduğu kimi siyahı GNU mail siyahısı idarəedicisi Mailman-dadır.

Siyahılara üzv olmaq üçün sadəcə **lists.freeswitch.org** ünvanına daxil olun və aşağıdakı şəkildəki göstərilən siyahıdan birinin üstünə **sixin**.



The screenshot shows a web browser window with the URL lists.freeswitch.org/mailman/listinfo in the address bar. The page title is "lists.freeswitch.org Mailing Lists". Below the title, there's a "Welcome!" message. The main content area displays a table of mailing lists with their descriptions. At the bottom of the page, there are logos for Mailman, Python, and Asterisk.

Welcome!

Below is a listing of all the public mailing lists on lists.freeswitch.org. Click on a list name to get more information about the list, or to subscribe, unsubscribe, and change the preferences on your subscription. To visit the general information page for an unadvertised list, open a URL similar to this one, but with a '/' and the list name appended.

List administrators, you can visit [the list admin overview page](#) to find the management interface for your list.

If you are having trouble using the lists, please contact mailman@tron.freeswitch.org.

List	Description
Freeswitch-announce	FreeSWITCH Announcements Mailing List
Freeswitch-biz	[no description available]
Freeswitch-branches	Freeswitch Branch Commit Logs
freeswitch-cluecon	Messages for ClueCon 2015
FreeSWITCH-dev	[no description available]
Freeswitch-docs	FreeSWITCH Docs Team
freeswitch-sec	FreeSWITCH Security List
Freeswitch-svn	[no description available]
Freeswitch-trunk	Freeswitch Trunk Commit Logs
FreeSWITCH-users	FreeSWITCH Users Help
Freeswitch-video	[no description available]
Openmrcp-users	[no description available]
OSTAG	OSTAG News and Information
TEST-LIST	[no description available]



Yeni istifadəçilər proektdə özlərini rahat hiss edənədək, yalnız FreeSWITCH-users siyahısına üzv olmalıdırlar. [Freeswitch-biz](#)(Kommersiya xarakterli müzakirələr burda aparılır)-dən başqa digər siyahılar təbiətən daha texnikidirlər.

Siyahiya üzv olmaq üçün istifadəçi adı və şifrəni daxil etməlisiniz. Bu məlumatları əl altında saxlayın çünki, lazım olan halda üzv olduğunuz yerlərdə elektron ünvanında dəyişiklik edə biləsiniz. Dəyişəbiləcək vacib qurulmalar "**digest**" emallərin alınmasından ibarətdir. Digest - tək ötürülmədə bir neçə elektron məktubun birləşməsidir. Digest metodu öz elektron poçtun trafikinin təsadüfi oxucusu üçün rahatdır. Ancaq, siz başqaları ilə qarşılıqlı əlaqəyə girmək istəyirsinizsə, digest-dən istifadə etmək yararsız olacaq, çünki müzakirənin müəyyən hissəsində iştirak etmək çətin olacaq.

Mail siyahilarının istifadə edilməsində bəzi nöqtələr aşağıdakılardır:

- Mail client istifadə edin çünki, belə üsulla mail axının izlənilməsi daha rahatdır.
- Axını "**hijack**" etmeyin. Hijack o halda baş verir ki, üzv email Subject-ni dəyişir və axına fərqli subject ilə yazar. Əgər yeni mövzu açırsınızsa, onda mütləq həmin mövzunun adını Subject-ə yazaraq yeni axın yaratmaq lazımdır.
- Yəni qoşulduğda təəccüblənməyin. Bir gün ərzində həddən artıq çoxlu email gələ bilər və diskinizdə də həddən artıq yer tutu bilər.
- Sizə lazım olan mövzunun axtarışı üçün saytin arxivindən istifadə edin. Misal üçün google-dan istifadə edə bilərsiniz. Google-da **site:lists.freeswitch.org "early media"** strukturunda axtarış edin ki, bütün "**early media**" Subject-li olanları tapa biləsiniz.

Mail siyahıları dünyada olan şəxslərlə əlaqəyə girmək üçün əla üsuldur. Ancaq elə hallar olur ki, sizin şəxsi dialoqa ehtiyac duyulur. Bu hallarda siz dünyada olan şəxslərlə birbaşa yazışa bilərsiniz.

IRC vasitəsilə real vaxtda əlaqə

IRC ya da Internet Relay Chat digər istifadəçilərlə danışmaq üçün həmişə seçiləndir.

irc.freenode.net-də FreeSWITCH komandasının çoxlu chat otaqları mövcuddur. Onlar aşağıdakılardır:

- #freeswitch
- #freeswitch-dev
- #freeswitch-social
- #freetdm

Dünyada olan fərqli cəmiyyət üzvlərinin həmçinin digər dillərdə chat otaqları olur. Onlardan bəziləri aşağıdakılardır:

- #freeswitch-de
- #freeswitch-es

- #freeswitch-fr

IRC-ni istifadə eləmək üçün sizin client programına ehtiyacınız olacaq. Onlardan seçə biləcəkləriniz çox var və aşağıda sadalanır:

- Chatzilla: A Firefox add-on
- IRSSI: A text-based IRC client
- Colloquy: An IRC client for Mac OS X
- mIRC: An IRC client for Windows

Siz həmçinin **#freeswitch** kanalına FreeSWITCH-in web saytında olan java applet istifadə edərək qoşula bilərsiniz.

IRC istifadə eləmək üçün mütləq **nickname** (qısa ad) seçməlisiniz. Çalışın qısa ad seçəsiniz və əgər mümkündürsə, bu ad üçün Freenode-da qeydiyyatdan keçin. <http://freenode.net/faq.shtml#userregistration> ünvanından adınızın qeydiyyatı və işə salınması haqqında ətraflı oxuya bilərsiniz.

Online gördüğünüz niklərdən bir neçəsiniz sadalayırıq:

- anthm: [Anthony Minessale](#)
- bkw_: Brian K West
- MikeJ: Michael Jerris
- mercutioviz: Michael S Collins
- pyite: Darren Schreiber
- intralanman: Raymond Chandler
- SwK: Ken Rice

Bunlar FreeSWITCH-də cəmiyyətində olan aktiv üzvlərdir. Digərləri də vardır ki, gün ərzində online olurlar(vaxt aralığı və gecə-gündüzdən asılı olaraq)

IRC istifadə elədikdə aşağıdakılari nəzərə almalısınız:

- Bu yaxşı niyyətli olan şəxslərin ünvanıdır.
- Digərləri kobud olsa da, siz nəvazişli olun.
- Kanalı çoxlu lazımsız məlumatla doldurmayın. Əgər sizin iki və ya 3 sətirdən çox məlumatınız varsa, onda <https://pastebin.freeswitch.org/> - dən istifadə edin.
- Əgər otağa üzv olursunuzsa, suali verməzdən önce soruşmağa ehtiyac yoxdur, sadəcə sualınızı verin. Misal üçün "Mən yeniyəm və çalışıram SIP trunk quram. Nə üçün təçizatçı IP tələb elədiyi halda, FreeSWITCH deyir ki, istifadəçi adı və şifrə daxil edin?"
- Səbirli olun! Adətən əksər insanlar bir neçə dəqiqə ərzində cavab verirlər amma, yadda saxlayın ki, kanalda olma ehtimalı Şimali Amerikanın biznes saatlarından sonradır.
- Müəyyən bir səviyyəyə sahib olan insanlar burda xoş qarşılanacaq. Əsas **#freeswitch** kanalı İngilis dilindədir amma, orda digər dillərdə danışan var.

FreeSWITCH kanalına qoşulmaqdə azad olun.

FreeSWITCH əsas WEB səhifə və WIKI

FreeSWITCH proekti üçün 2 əsas web səhifə mövcuddur:

- www.freeswitch.org: Proektin əsas səhifəsi
- wiki.freeswitch.org: Public wiki səhifəsi

Əsas FreeSWITCH səhifəsi - www.freeswitch.org

FreeSWITCH-in əsas web səhifəsi proekti üçün bütün şeylər üçün əsas nöqtədir.

Əsas səhifədən siz aşağıdakı işləri görə bilərsiniz:

- FreeSWITCH və VoIP xəbərlərin oxunması
- Mənbə kodlarını endirmək və baxmaq
- Bug və ya imkan üçün müraciət açmaq
- Sənədləşməyə baxmaq
- Java applet ilə **#freeswitch** IRC kanalına qoşulmaq

Səhifə üçün yeni mətn həftə yenilənir.

FreeSWITCH wiki səhifəsi - wiki.freebsd.org

FeeSWITCH Wikipedia üçün əsas mənbə FreeSWITCH rəsmi sənədləridir. FreeSWITCH wiki web səhifəsi istifadəçilərə məzmunu əlavə eləməyə, dəyişməyə ya da silməyə və digər məzmunu link etməyə izin verir. Classic Wiki səhifəsinin imkanlarıdır. FreeSWITCH özü üçün MediaWiki (<http://www.mediawiki.org>) istifadə edir. Eyni Wikipedia motoru Wikipedia tərəfindən istifadə edilir. FreeSWITCH wiki-si cəmiyyət resursudur. Baxmayaraq ki, Michael S. Collins əsas Administratordur, bütün istifadəçilər məzmun əlavə edə və deyişə bilər. Əksər wiki saytlarında olduğu kimi, orda çoxlu mətn var. Hərdən məlumatın axtarılması problemə çevrilə bilər. Biz məsləhət görürük ki, axtarış üçün Google-dan(site:wiki.freeswitch.org "Axtarış söyü") istifadə edəsiniz. Wiki yazarları aşağıdakı şeyləri yadda saxlamalıdır:

- Məzmun əlavə eləmədən öncə axtarmalıdır - Əlavə ediləcək məzmun orda artıq ola bilər və sadəcə yenilənməsinə ehtiyac var.
- Əmin olun ki, əlavə elədiyiniz bütün məzmun düzgün link edilmişdir.
- Əmin olun ki, əlavə elədiyiniz istənilən məzmun sayt kateqoriyasının hissəsidir.
- Səhv etməkdən çəkinməyin! Digərləri sizin səhvlərinizi düzəltməkdən həzz alacaqlar.

Açıq qaynaqlı program təminatının sənədləşməsi demək olar ki, həmişə problemdir ona görə də, əgər siz kömək edə bilirsinizsə, xahiş edirik Michael ilə msc@freeswitch.org əlaqəyə girin. Düzəltmə, faktların üzə çıxarılması, quraşdılmaların sınaqdan keçirilməsi və digər dillərdən sınaqdan keçirilməsi həmişə tələb edilir.

Hər il keçirilən ClueCon açıq qaynaqlı yaradıcı konfransı

Hər il 3 gün Chicago-da açıq qaynaqlı VoIP ilə məşğul olan professionallarla fanatlar görüş keçirir və vacib məqamları müzakirə edirlər. Konfrans Avqustun ilk həftəsində keçirilir. <http://www.cluecon.com/> səhifəsinə daxil olub növbəti konfrans vaxtını və ötən konfransların video ilə prezəntasiyalarına görə bilərsiniz.

Həmçinin ClueCon "by developers, for developers" yaradıcı olmayanlar üçün də genişlənməyə başlamışdır. Əksər təqdimatlar hələ də, texniki strukturda olur amma, orda çoxlu yaradıcılar öz məhsullarını göstərir və qarşılıqlı əlaqə haqqında da müzakirələr aparırlar. Konfransın məqsədi istifadəçilər, yaradıcılar və vendorlar arasında əlaqə yaratmaq üçün nəzərdə tutulur.

ClueCon əksər VoIP-lə məşğul olan OpenSource sevənləri öz təqdimatlarında iştirak etməyə çağırır. İllər ərzində Asterisk, FreeSWITCH, Kamalio və OpenSIPS haqqında çoxlu təqdimatlar olmuşdur. Avadanlıq vendorlarından işə Sangoma və Vestec-də bu təqdimatlarında iltifatlı olmuşlar.

B

Asterisk-dən FreeSWITCH-ə miqrasiya edilməsi

Bu bölmə üçün Stefan Wintermeyer xüsusi təşəkkur edirik.

Bölmə inzibatçılar və programçılar üçündür hansı ki, öz sistemlərini Asterisk-dən FreeSWITCH keçirmək istəyirlər. Bu kitab Asterisk-ə yox FreeSWITCH-ə aid olduğuna görə Asterisk-lə bağlı olan detalları biz bu başlıqda açıqlamırıq. Bu o anlama gəlir ki, oxuyucu artıq Asterisk konsepsiyaları, genişlənmələri, üstünlükleri və programları ilə artıq tanışdır.

Başlıq sizə Asterisk-də olan bütün quraşdılmaların tamlıqla FreeSWITCH-ə konvertasiya edilməsini nəzərdə tutmur. Sadəcə Asteriskdə görüləcək işlərin FreeSWITCH-də necə görülməsini açıqlayır. Başlıqlar aşağıdakılardan ibarətdir:

- FreeSWITCH və Asteriskin işə salınması və dayandırılması
- Jurnalların ətraflı araşdırılması səviyyələri

Asterisk haqqında biliklərinizi təkmilləşdirmək üçün <https://freeswitch.org/confluence/display/FREESWITCH/Rosetta+Stone> wiki səhifəsindən məlumat ala bilərsiniz.

İşə başlayaq

Bu sənəd üçün Asterisk və FreeSWITCH FreeBSD serverində işə salılmışdır və IP ünvanı 10.0.0.10-dur. Biz hər iki servisi ayrı-ayrılıqda fərqli maşınlarda işə sala bilərdik amma, daha asan misallar üçün növbə ilə bir servisi dayandırıb digərini işə salmaqla işlərimizi görəcəyik. Hər iki program təminatı üçün ən son versiyadan istifadə edəcəyik. SIP telefonlardan biri 10.0.0.20 və o biri işə 10.0.0.21-dir. SIP telefonların qurulması üçün 4-cü başlıqda SIP və istifadəçi qovluğu üsullarını oxuya bilərsiniz.

Qeyd: Siz Asterisk və SIP server eyni maşında eyni vaxtda işə sala bilməzsınız ona görə ki, onlar eyni port-da işləyirlər. Yalnız fərqli IP-lər və ya fərqli portlarda quraşdırıb eyni zamanda işə sala bilərsiniz. Misal üçün **/usr/local/etc/asterisk/sip.conf** faylında portu 5060-dan fərqli olana təyin edə bilərsiniz.

FreeSWITCH və Asteriskin işə salınması və dayandırılması

Asteriskin susmaya görə yüklenməsində siz onu **asterisk** əmri ilə də işə sala bilərsiniz. Bu əmr asteriski arxa fonda işə salacaq. Dayandırmaq üçün isə siz **asterisk -r** əmri ilə CLI-a daxil olub, **core stop now** əmrindən istifadə edə bilərsiniz. Adətən aşağıdakı kimi olur:

```
# asterisk -r
asterisk*CLI> core stop now
asterisk*CLI>
Disconnected from Asterisk server
Asterisk cleanly ending (0).
Executing last minute cleanups
root@asterisk:~ #
```

Adı FreeSWITCH yüklenməsində siz onu arxa fonda aşağıdakı əmrlə işə sala bilərsiniz:

```
# /usr/local/freeswitch/bin/freeswitch -nc
```

Dayandırmaq üçün isə eyni əmri **-stop** argументi ilə istifadə eləmək lazımdır:

```
# /usr/local/freeswitch/bin/freeswitch -stop
```

Jurnalların ətraflı araşdırılması səviyyələri

Aşağıdakı misalların müddətində siz detallı şəkildə nə baş verməsini görmək istəyəcəksiniz. Məsləhətdir ki, fərqli **debug** səviyyələrinə görə **debug** edib hər şeyi detallı araşdırırasınız.

Asterisk

Əgər sizin işlək Asterisk serveriniz varsa, **asterisk -r** əmri ilə CLI açırsınız. Jurnal səviyyəsini 3 təyin eləmək üçün **core set verbose 3** əmrini daxil etmək lazımdır. Əksər hallarda **core set debug 3** istifadə eləməyin çünki bu səviyyə programçılar üçün həddən artıq bol məlumat çap edir. Debug səviyyəsini sıfırlamaq üçün isə **core set debug 0** təyin edin. Adı sessiya aşağıdakı kimi olmalıdır:

```
debian*CLI> core set verbose 3
Set remote console verbosity to 3
debian*CLI>
```

FreeSWITCH

Əgər sizdə artıq işlək olan FreeSWITCH serveriniz varsa, **fs_cli** əmri ilə CLI-a daxil ola bilərsiniz. **/log info** əmri ilə debug səviyyəsini 6-cı təyin edə bilərsiniz. Aşağıdakı göstərildiyi kimi:

```
freeswitch@internal> /log info
+OK log level info [6]
```

Artıq debug edə bilirik. Gəlin həm Asterisk və həm də FreeSWITCH-in quraşdirmalarına baxaq.

Quraşdırma faylları

Asterisk və FreeSWITCH-in öz quraşdirmalarına görə geniş strukturları var:

tree /usr/local/etc/asterisk

FreeSWITCH üçün aşağıdakı əmrdir:

tree /usr/local/freeswitch/conf

Strukturdan göründüyü kimi, proektlər həddən artıq böyükdür və bəzi fayllar mövcuddur ki, onlara toxunmaq ümumiyyətlə lazım deyil.

Böyük fərq ondan ibarətdir ki, Asterisk üçün hər bir quraşdırma faylinin spesifik mənası vardır. Misal üçün, Asterisk dialplan üçün **extensions.conf** faylına baxacaq. FreeSWITCH quraşdırma faylini ayrıca seçilmiş faylda saxlamır. O böyük bir XML faylından oxuyur və onları daha da kiçiklərinə parçalayırlar. Susmaya görə olan fayllar nüsxələrdən ibarətdir. Siz öz FreeSWITCH quraşdırma fayllarınızı adlandırmak və düzənmək üçün azad qərar verə bilərsiniz. Faktiki olaraq əsas vacib tələb edilən quraşdırma faylı **freeswitch.xml**-dir(3-cü başlıqdə quraşdırma nüsxələrində sınaqların keçirilməsinə baxın).

Aşağıdakı seksiyalarda biz bəzi quraşdırma fayllarına ötəri gözlə baxacayıq.

Qeyd: Əgər biz nəzərdə tuturuqsa "replace" bu o deməkdir ki, tamlıqla faylı silib yenisiyi yaradaraq içiniə əlavə eləmək lazımdır. Çalışmayın ki, mövcud faylin içiniə kodları əlavə edəsiniz. Susmaya görə olan quraşdırma faylları əla nüsxələrdən ibarətdir yalnız, onlar daha dərindən öyrənmək üçün yaxşı deyil.

İki SIP telefonları

Kiçik PBX nüsxəsi **2** SIP telefondur. Biz birini **2000** genişlənməsi ilə və digərini **2001** genişlənməsi ilə yaradacayıq. Hər bir telefon qarşı telefon genişlənməsinə zəng edərək düşməlidir.

İlk SIP hesabı **2000**-nin də şifresi **1234** və ikincinin **2001**-in şifresi **1234-**dur. Xahiş olunur SIP telefonları bu hesablarları qurasınız.

Asterisikin qurulması

Asterisk öz SIP hesabları məlumatlarını **/usr/local/etc/asterisk/sip.conf** faylında saxlayır. Xahiş olunur susmaya görə olan **sip.conf** faylini silib yenidən yaradın və tərkibinə aşağıdakı sətirləri əlavə edin:

[general]

port=5060

```
bindaddr=0.0.0.0
```

```
[2000]
type=friend
secret=1234
context=default
host=dynamic

[2001]
type=friend
secret=1234
context=default
host=dynamic
```

Dialplan isə **/usr/local/etc/asterisk/extensions.conf** faylında saxlanılır. Xahiş olunur onun tərkibini aşağıdakı kod-la əvəz edin:

```
[default]
exten => _200[1-2],1,Dial(SIP/${EXTEN})
```

Artıq asteriski işə salsaq(**asterisk -c**) və jurnal səviyyəsini **3**-ə təyin eləsək(**core set verbose 3**), artıq telefonların necə qeydiyyatdan keçmələrinə görə biliərik:

```
*CLI> -- Registered SIP '2000' at 10.0.0.21:2048
-- Registered SIP '2001' at 10.0.0.20:3072
-- Unregistered SIP '2001'
-- Registered SIP '2001' at 10.0.0.20:3072
```

Biz telefon zəngini birindən digərinə edə bilərik. Zəng müddətində **sip show channels** əmri ilə mövcud zəng haqqında bəzi başlanğıc məlumatları əldə edə bilərik:

```
*CLI> sip show channels
Peer User/ANR Call ID Format Hold
Last Message Expiry Peer
10.0.0.21 2000 150b1e3879a2bff (ulaw) No
Tx: ACK 2000
10.0.0.20 2001 ea88263cebddd-la (ulaw) No
Tx: ACK 2001
2 active SIP dialogs
```

Bununlada biz Asteriskdə zəng prosesinin qurduq və yoxladıq. Gəlin eyni prosesi FreeSWITCH üçün edək.

FreeSWITCH qurulması

FreeSWITCH-in müəyyən olunmuş fayl və qovluq strukturu yoxdur. Sizin **/usr/local/freeswitch/conf** da gördünüz sadəcə bir nümunədir. Siz hər şeyi yalnızca bir XML fayla yerləşdirə və ya parçalayıb ayrı-ayrı XML fayllara yerləşdirə və onları da istədiyiniz kimi adlandıra bilərsiniz. Nümunə nizamlama faylları artıq misal kimi bir neçə nümunə SIP hesabları və genişlənmələri(extensions) özündə saxlayır. Buna baxmayaraq, biz yuxarıda

Asterisk üzerinde yaratdığımız aynı nümunəni FreeSWITCH üçün də yenidən yaratmaq istəyirik.

1-ci SIP istifadəçisi üçün
`/usr/local/freeswitch/conf/directory/default/2000.xml` faylını yaradıb
 aşağıdakı məzmunu əlavə edirik:

```
<include>
  <user id="2000">
    <params>
      <param name="password" value="1234"/>
    </params>
    <variables>
      <variable name="user_context" value="default"/>
    </variables>
  </user>
</include>
```

2-ci SIP istifadəçisi üçün
`/usr/local/freeswitch/conf/directory/default/2001.xml` faylını yaradıb
 aşağıdakı məzmunu əlavə edirik:

```
<include>
  <user id="2001">
    <params>
      <param name="password" value="1234"/>
    </params>
    <variables>
      <variable name="user_context" value="default"/>
    </variables>
  </user>
</include>
```

Son olaraq bu iki yeni SIP istifadəçiləri üçün Dialplan faylını yaradırıq.
`/usr/local/freeswitch/conf/dialplan/default/01_New.xml` adında dialplan
 faylını yaradırıq və məzmununa aşağıdakı sətirləri əlavə edirik:

```
<?xml version="1.0" encoding="utf-8"?>
<include>
  <context name="default">
    <extension name="Local_Extension">
      <condition field="destination_number"
        expression="^(200[1-2])$">
        <action application="export"
          data="dialed_extension=$1"/>
        <action application="bridge"
          data="user/${dialed_extension}@${domain_name}"/>
      </condition>
    </extension>
  </context>
</include>
```

Bu 3 faylı yaratıldıqdan və yaddaşda saxladıqdan sonra, FreeSWITCH işə salırıq.
`/usr/local/freeswitch/bin/freeswitch -nc`

FreeSWITCH prosesinin işə düşməsini bir qədər gözlədikdən sonra, **fs_cli** vasitəsi ilə qoşuluruq:
/usr/local/freeswitch/bin/fs_cli

Əmin olurraq ki, hər iki SIP telefon işləkdi və hər ikisi qeydiyyatdan keçmək üçün cəhd ediblər. (Əgər siz telefonu önce Asterisk serverə qoşmuşdunuzsa, siz telefonları yenidən qeydiyyatdan keçirməli və ya yenidən başlatmalısınız.) Telefonların qeydiyyatının vəziyyətini bu əmr vasitəsilə görə bilərsiniz : **sofia status profile internal reg**

Nəticə belə olacaq:

```
freeswitch@internal> sofia status profile internal reg
Registrations:
Call-ID: a270263caa23-uocan9j61z5y
User: 2000@127.0.0.1
Contact: "2000" <sip:2000@10.0.0.20:3072;line=0tqusdnm>
Agent: snom821/8.4.35
Status: Registered(UDP) (unknown) EXP(2013-01-13 06:46:31)
EXPSECS(3529)
Host: debian
IP: 10.0.0.20
Port: 3072
Auth-User: 2000
Auth-Realm: 10.0.0.10
MWI-Account: 2000@127.0.0.1
Call-ID: 3c26708e4d57-yzfzr61f7x41
User: 2001@127.0.0.1
Contact: "2001" <sip:2001@10.0.0.21:2048;line=9r6kyu0i>
Agent: snom360/8.4.35
Status: Registered(UDP) (unknown) EXP(2013-01-13 06:46:45)
EXPSECS(3543)
Host: debian
IP: 10.0.0.21
Port: 2048
Auth-User: 2001
Auth-Realm: 10.0.0.10
MWI-Account: 2001@127.0.0.1
Total items returned: 2
freeswitch@internal>
```

Indi biz 2000-dən 2001-ə və əksinə zənglər edə bilərik.

Zəng müddətində istifadə olunan kanalları **show channels** əmri ilə analiz edə bilərik. İki telefon arasında zəng edin və sonra **show channels** əmrini daxil edin. Aşağıdakına oxşar olani əldə ełəməlisiniz:

```
freeswitch@internal> show channels
uuid,direction,created,created_epoch,name,state,cid_name,cid_num,ip_
addr,dest,application,application_data,dialplan,context,read_
codec,read_rate,read_bit_rate,write_codec,write_rate,write_bit_
rate,secure,hostname,presence_id,presence_data,callstate,callee_
name,callee_num,callee_direction,call_uuid,sent_callee_name,sent_callee_
num
```

```

af6dc664-5cb3-11e2-ae64-41a8c0d6e735,inbound,2013-01-12
13:29:18,1357993758,sofia/internal/2000@10.0.0.10,CS_EXECUTE,2000
,2000,10.0.0.20,2001,bridge,user/2001@127.0.0.1,XML,default,PCMU,8
000,64000,PCMU,8000,64000,,debian,2000@10.0.0.10,,ACTIVE,Outbound
Call,2001,SEND,af6dc664-5cb3-11e2-ae64-41a8c0d6e735,Outbound Call,2001
af861bd8-5cb3-11e2-ae6d-41a8c0d6e735,outbound,2013-01-12
13:29:18,1357993758,sofia/internal/sip:2001@10.0.0.21:2048,CS_EXCHANGE_ME
DIA,2000,2000,10.0.0.20,2001,,,XML,default,PCMU,8000,64000,PCMU,8000,640
00,,debian,2001@127.0.0.1,,ACTIVE,Outbound Call,2001,SEND,af6dc664-5cb3-
11e2-ae64-41a8c0d6e735,2000,2000
2 total.
freeswitch@internal>

```

Beləliklə, biz Asteriskdə və FreeSWITCHdə SIP istifadəçilərini qurmuş olduq.

Oeyd: Əgər siz FreeSWITCH XML nizamlama fayllına FreeSWITCH işlək olduğu halda dəyişiklik etmişinizsə **reloadxml** əmrini icra etməlisiniz. Alternativ olaraq **F6** düyməsini də sıxa bilərsiniz.

Analiz

FreeSWITCH XML, Asterisk isə ənənəvi "**ini**" fayllarını istifadə edir. Sintaksis səhvlerinin tez tapılmasının mümkünüyünə görə XMLin böyük üstünlükləri var. Bu məsələdə Asteriskdə bir qədər boşluqlar var və bəzi hallarda sistem inzibatçıya kifayət edəcək qədər məlumat qaytarır. Bəzi hallarda bizim Asterisk Dialplan faylı düzgün görsənməyinə və əksər hallarda işləməsinə baxmayaraq bəzi vəziyyətlərdə işləmiyə bilir. Bir çox hallarda bu Asterisk Dialplanda olan sintaktik səhvrlə bağlı olur, hansı ki Asteriskin özü tərəfindən belə aşkar oluna bilinmir. Beləliklə sərt XML nizamlamalarının olması yaxşıdır, lakin buna öyrəşmək müəyyən qədər vaxt tələb edir. Bu keçid mərhələsində yaxşı bir XML editoru çox faydalı olardı. Sintaksis vurğulaması (**highlighting**) imkanı olan bir redaktördə FreeSWITCH nizamlama fayllarını gözdən keçirərkən və nizamlayarkən çox faydalıdır.

Bu iki program təminatında SIP istifadəçinin nizamlanması sintaksisi tamamilə fərqlənilir. Dialplan da çox fərqlənir. Biz Asterisk və FreeSWITCH üçün SIP istifadəçiləri **default** context ilə (hərfi olaraq "**default**") müəyyən etdik. Hər iki program təminatı öz qurulma faylında default konteksti axtarır. Asterisk bunun üçün və xüsusü nizamlamalar üçün **extensions.conf** faylini istifadə edir. **[default]** contextin içində uyğun gələn genişlənmələri axtarır. Müntəzəm ifadə **_200[1-2]** daxil edilmiş nömrəyə uyğun gəlir və Dial programını işə salır, hansı ki SIP protokolu vasitəsi ilə **\${EXTEN}**-ə zəng edir, hansıki Asterisk vasitəsi ilə avtomatik təyin olunan dəyişəndir və özündə yığılmış nömrəni saxlayır.

FreeSWITCH-də həmçinin **default** kontekstdə axtarış edir. (Çünki SIP istifadəçiləri bu kontekstdə təyin edilib. FreeSWITCH **condition** (şərt) bölməsinə uyğun gələn hissəni tapana kimi bu kontekstdə təyin edilmiş bütün extenstionlar-in üzərindən keçir. Şərt bölməsində bir çox hallar ola bilər, məs: vaxt və ya bizim misalda olduğu kimi zəng edilən nömrə hansı ki müntəzəm ifadə ilə uyğun gəldi **^(200[1-2])\$**. Bizim misalda bu aşağıdakı koddur:

```

<action application="export" data="dialed_extension=$1"/>
<action application="bridge" data="user/${dialed_extension}@${domain_name}" />

```

Bunu bir sətirdə də yaza bilərik:

```
<action application="bridge" data="user/${domain_name}"/>
```

Lakin bir çox hallarda **dialed_extension** dəyişənini təyin etmək çox faydalıdır. Bu Asterisk veteranına **\${EXTEN}**-i xatırlada bilər. Eyni zamanda biz burda FreeSWITCH-in **bridge** programını, Asteriskin **Dial** programının ekvivalenti olduğunu görürürük. FreeSWITCH-də biz **bridge** argumentində **@\${domain_name}**-ə istinadın olduğunu görürük. Nümunə üçün olan nizamlamada biz **\${domain_name}** kanal dəyişənin başqa yerdə təyin olunduğunu görürük.

Voicemail

Artıq diqqətimizi hər iki sistemdə mövcud olan imkana yönəldirik: voicemail

Asterisk

Asteriskin Dialplan programı olan **Dial**, bizə telefonun neçə saniyə zəng çalmasını təyin etmək imkanı verir. Zəng edilən tərəfin cavab verməsi üçün təyin olunmuş saniyələr(məsələn **10** saniyə) ərzində gözlədikdən sonra növbəti prioritətə keçəcək. Aşağıdakı nümunədə bu bizim Voicemail applikasiyamızdı. **/usr/local/etc/asterisk/extensions.conf** faylini aşağıdakı nizamlamarla əvəz edirik:

```
[default]
exten => _200[1-2],1,Dial(SIP/${EXTEN}, 10)
exten => _200[1-2],n,Voicemail(${EXTEN},u)
```

Voicemail üçün Asteriskdə bəzi əlavə nizamlamalara ehtiyac duyulur.

/usr/local/etc/asterisk/voicemail.conf faylini bu nizamlamalarla əvəz edirik:

```
[general]
format = wav
attach = yes
[default]
2000 => 1234,Mr. X
2001 => 1234,Mr. Y
```

Indi biz zəng edə bilərik. **10** saniyədən sonra **Dial** programı zəngi saxlayacaq. Asterisk prioriteti **1** vahid artıracaq və **\${EXTEN}** voicemail qutusu üçün **Voicemail** programını başladacaq ki, artıq zəng edən tərəf öz mesajını saxlaya bilər.

FreeSWITCH

FreeSWITCHdə voicemail nizamlanması tamam fərqlidi.

/usr/local/freeswitch/conf/dialplan/default/01_New.xml faylinə bu nizamlamaları əlavə edərək əvəzləyirik:

```
<?xml version="1.0" encoding="utf-8"?>
<include>
  <context name="default">
    <extension name="Local_Extension">
      <condition field="destination_number">
        <expression>^(200[1-2])$</expression>
        <action application="export"
          data="dialed_extension=$1"/>
        <action application="set" data="call_timeout=10"/>
```

```

<action application="set"
data="hangup_after_bridge=true"/>
<action application="set"
data="continue_on_fail=true"/>
<action application="bridge"
data="user/${dialed_extension}@${domain_name}"/>
<action application="answer"/>
<action application="sleep" data="1000"/>
<action application="bridge"
data="loopback/app=voicemail:default ${domain_name} ${dialed_
extension}"/>
</condition>
</extension>
</context>
</include>

```

FreeSWITCH dialplanı bir qədər mürəkkəbdir, lakin daha çox idarə etmə imkanı verir. **call_timeout=10** ilə **bridge** programının zəng edilən tərəfi saniyələrlə maksimum nə qədər gözləməli olduğunu göstəririk. **hangup_after_bridge=true** nizamlaması FreeSWITCH-ə **bridge** olunmuş zəngdən sonra kəsilmənin lazım olduğunu başa salır (Bu nizamlama həm də zəng edənin voicemailə getməsi və sonra dəstəyi aşması zamanı da vacibdir) **continue_on_fail=true** nizamlaması zəng olunan tərəfin məşğul olduğu halları idarə etmək üçün nəzərdə tutulub. Bridge programından sonra biz cavab əldə edirik, hansı ki sanki dəstəyi FreeSWITCH-in özü qaldırması anlamına gəlir. 1 saniyə gözlədikdən sonra FreeSWITCH zəngi voicemail programı üçün **loopback** ünvana bağlayır. Biz voicemail programını **loopback** kanal olmadan da istifadə edə bilərik, lakin bu xüsusi sintaksisi istifadə etməklə biz iştirak etmiş transferlərə istifadəçinin voicemail qutusuna izin veririik.

Voicemaili əldə etmək

Artıq sistemin bizim istifadəçilərin voicemail mesajlarını yazmasını tənzimləmişik, indi isə bu mesajları necə əldə edilməsinə baxaq. Hər bir halda, istifadəçiye zəngin edildiyinə və voicemail mesajının saxlandığına əmin olmaq lazımdı, aşağıdakı yolla voicemaili əldə etmək olar. Biz **4000** rəqəmini yüksaraq voicemaili əldə etməyə imkan verən dialplan hazırlamaq istəyirik.

Asterisk

```

/usr/local/etc/asterisk/extensions.conf faylinı aşağıdakı nizamlamalarla əvəz
edirik:
[default]
exten => _200[1-2],1,Dial(SIP/${EXTEN}, 10)
exten => _200[1-2],n,VoiceMail(${EXTEN},u)
exten => 4000,1,VoiceMailMain(${CALLERID(num)})

```

Asterisk **VoiceMailMain** programını zəng edənin şəxsi voicemail qutusuna çıxışını təmin edir. **\${CALLERID(num)}** funksiyası zəng edənin nömrəsini qaytarır. Bunu **\${EXTEN}** ilə qarışdırırmayaq, bu bizim yığdığımız nömrədi və indiki nümunədə **4000** olacaq.

FreeSWITCH

İlkin olaraq sahibi tərəfindən əldə oluna bilməsi üçün voicemail qutusuna şifrə təyin etməliyik. Sadə olması üçün biz **1234** təyin edirik.

/usr/local/freeswitch/conf/directory/default/2000.xml faylini açaq və bu sətiri tapaqq:

```
<param name="password" value="1234"/>
```

Yeni param sətri əlavə edək:

```
<param name="password" value="1234"/>
```

faylı yadda saxlayaqq. Eyni dəyişiklikləri

/usr/local/freeswitch/conf/directory/default/2001.xml üçün edək.

Artıq biz **4000** yığış voicemail mesajlarını əldə edə bilərik, FreeSWITCH dialplanı öncədən **4000**-i mesaj əldə etmək üçün genişlənmə kimi təyin edib. Alternativ olaraq ***98**-i də istifadə etmək olar. Diqqət edək ki, sistem bizdən həm ID nömrə həm də şifrə daxil etməyi tələb edir. Bəzi insanlar sistemdə əgər siz hansısa voicemail-ə hansısa extensiondan zəng etmisizsə, onda məhz həmin extensionun voicemailinə daxil olmağa cəhd olunmalıdır nizamlanmasının olunmasına üstünlük verirlər. Bu sadə yolla nizamlana bilər.

/usr/local/freeswitch/conf/dialplan/default.xml faylini açaq və bu extensionu tapaqq:

```
<extension name="vmain">
<condition field="destination_number" expression="^vma
in$|^4000$|^*98$">
<action application="answer"/>
<action application="sleep" data="1000"/>
<action application="voicemail" data="check default ${domain_
name}"/>
</condition>
</extension>
```

voicemail programı təyin olunmuş sətirə diqqət edək. Voicemail programına verilmiş argumenti dəyişək ki, o **caller ID** nömrəsinin zəng edənin əldə etmək istədiyi voicemail qutusu olduğunu zənn etsin.

```
data="check default ${domain_name} ${caller_id_number}"
```

Diqqət edin ki, biz argumentə **\${caller_id_number}** argumentini əlavə edirik. Bu voicemail programına caller ID nömrəsinin(**2000** ya da **2001**) **caller**-in əldə etmək istədiyi **voicemail** qutusu olmasını deyir. Faylı yadda saxlayaqq və **fs_cli**-dan **reloadxml** əmrini icra edək. **4000** yığsaq indi sistem yalnız şifrə tələb edəcək. FreeSWITCH global voicemail nizamlamaları **/usr/local/freeswitch/conf/autoload_configs/voicemail.conf.xml** faylından tapıla bilər.

Nəticə

Asteriskdən FreeSWITCH-ə keçid qorxulu görünə bilər, lakin bu mümkünkdür. Asteriksi yenice öyrəndiyin günləri xatırla və indi bildiklərinlə müqayisə et. Bu müəyyən qədər vaxt aldı, lakin sən çox şeylər öyrəndin. FreeSWITCH-i öyrənmək oxşar vaxt və zəhmət tələb edir. Asteriskin istifadəsindən əldə edilmiş təcrübələrdən istifadə edərək bu prosesi daha da sürətləndirmək mümkünkdür.

Vacib linklər:

<http://files.freeswitch.org/downloads/libs/>
<http://files.freeswitch.org/freeswitch-releases/>
<http://files.freeswitch.org/>

İstifadə olunmuş ədəbiyyat siyahısı

1. FreeSWITCH Cookbook - Anthony Minessale, Michael S Collins, Darren Schreiber, Raymond Chandler
2. FreeSWITCH 1.0.6 - Anthony Minessale, Michael S Collins, Darren Schreiber
3. FreeSWITCH 1.2 - Anthony Minessale, Michael S Collins, Darren Schreiber, Raymond Chandler
4. <https://freeswitch.org/confluence/display/FREESWITCH/FreeSWITCH+Explained>
5. https://wiki.freeswitch.org/wiki/Main_Page
6. <https://github.com/areski/freeswitch-telegraf-plugin>
7. <https://www.google.ru/>
8. <https://habrahabr.ru/>
9. <http://www.2600hz.org/>
10. <https://2600hz.atlassian.net/wiki>