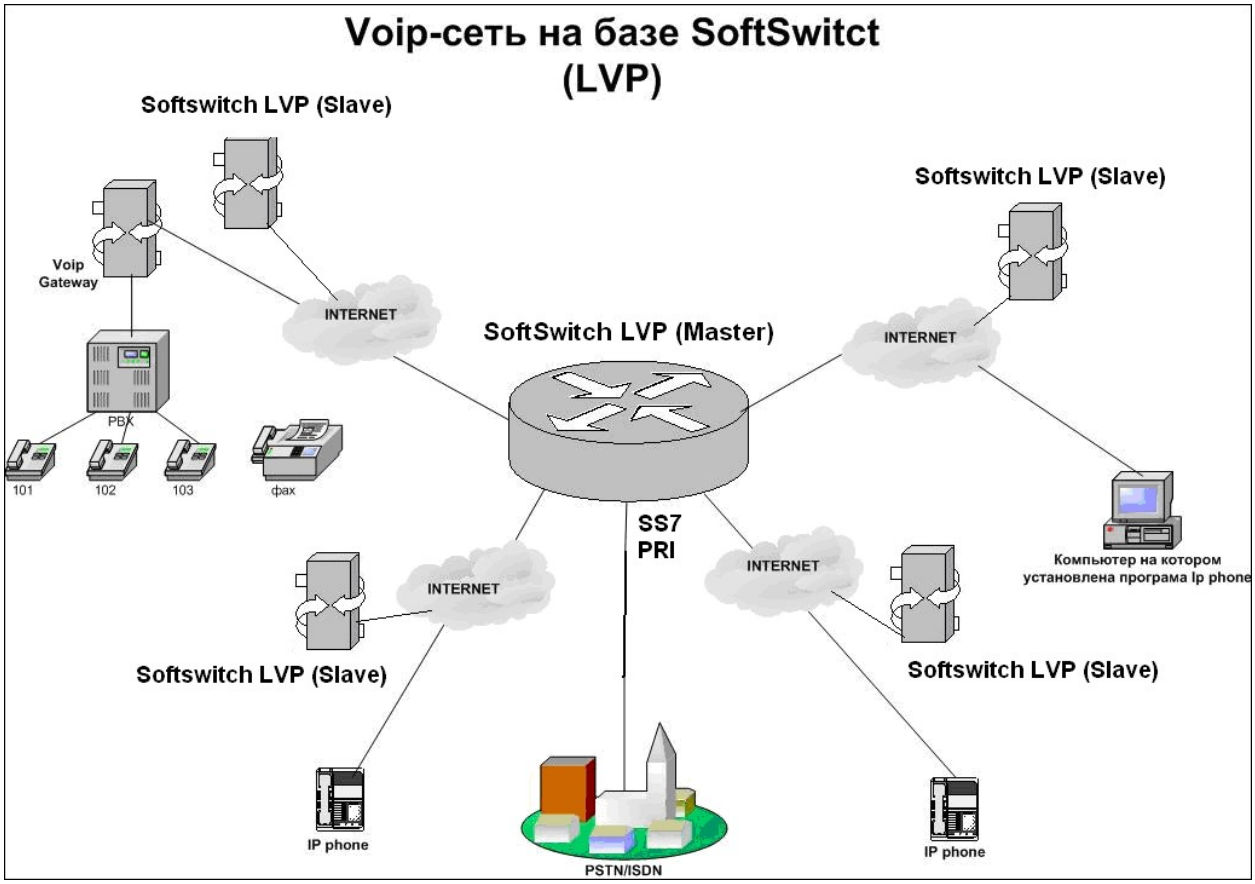
Программный коммутатор Voip трафика
Инструкция пользования LVP- ПРОСТОР Linux version.

General Networks

Введение

В настоящий момент IP технологии все больше внедряются в телекоммуникации. Данный программный комплекс позволяет решать весь комплекс задач по маршрутизации голосового трафика через IP. Рассмотрим основную схему построения сетей на основе LVP.



Описание продукта

Программа LVP разработана для маршрутизации VOIP трафика, работающая под Linux или FreeBSD.
Распространяется две версии софтсвича.

1) Софтсвич для маршрутизации трафика со скриптами и виртуальным офисом с интерфейсом для билинговой системы.
2) Версия для маршрутизации трафика с поддержкой RMTP протокола со скриптами и виртуальным офисом с интерфейсом для билинговой системы

Основные преимущества программы.

- Каждый звонок осуществляется как отдельная задача, т.е. если происходит ошибка при выполнении звонка, она никак не отражается на выполнении других звонков;

- Возможность горячей замена версии, при которой все звонки непрерываются;

- Кластеризация нескольких LVP в один софтсвич. Количество одновременных звонков при полном проксировании увеличивается путем сложения мощностей компьютеров. Появляется возможность распределять UDP трафик по разным Интернет каналам;

- Разработан новый протокол передачи RTP пакетов, который позволяет работать на каналах с потерями до 10% (только между двумя LVP);

- Встроенный язык разработки сценариев звонков (автоофисов) дальше по тексту скриптов;

- Возможность отладки скриптов одновременно со штатной работой программы по маршрутизации трафика, ошибка скрипта лишь приводит к завершению текущего звонка, остальные звонки продолжаются;

- Взаимодействие со многими SQL серверами одновременно, бекап билинговой информации в текстовый файл;

- Возможность запуска других приложений, команда типа run;

- Возможность передачи данных по http протоколу;

- Трансляция протоколов H323 в SIP и наоборот.

Дополнительные возможности есть по согласования различных реализаций H323-го протокола, интеллектуальная система перебора маршрутов, защита от атак, ограничение количества соединений, количества новых соединений в минуту, количества звонков на один номер, маршрутизация трафика по временным параметрам.

Программа поставляется как безлимитная версия так и с ограничением по количеству одновременных соединений 30, 60, 120. Язык скрипт позволяет согласовывать программу с любой билинговой системой при наличии драйвера SQL. Количество драйверов постоянно обновляется.

Поставляется slave версия, которая позволяет увеличивать мощность софтсвича, за счет распределения проксирования udp пакетов на дополнительные компьютеры, а также перераспределение маршрутов трафика по регионам.

Софтсвич может комплектоваться одной или двумя платами E1, каждая по 4 E1. Поддерживается протоколы OKC7 и ISDN PRI. Обеспечивается exo подавление, DTMF детекция, факс детекция. Для передечи факса протоколом T38 используется дополнительное устройтсво.

Возможности скрипт языка

- возможность выполняться, как высылая сообщение connect, так и высылая сообщение callproceeding или progress. Другими словами скрипт может работать когда соединение уже состоялось так и когда соединение только в прогрессе. Скрипт может открывать или закрывать звуковые потоки, давать свои звуковые сообщения, в сторону А так и в сторону Б, когда она поднимет трубку(удобно для колцентра). Есть команды, которые позволяют выполнять процедуры на SQL серверах и получать данные для их дальнейшей обработки. Есть возможность работать в режиме offline (когда сторона А положила трубку), делать колбек и соединять два плеча между собой. Встроены функции обработки строк, чисел, времени. Неограниченное количество вложенных команд IF else endif. До пяти вложений команд do while. Возможно запускать другие скрипты. У каждового скрипта есть возможность обработки события разрыва связи со стороной А при условии, что скрипт выполняется в режиме online. Здесь можно в режиме offline выполнять все команды и строить свой сценарий.

Инсталляция LVP

LVP инсталлируется техническим персоналом компании. Версия slave инсталлируется самостоятельно заказчиком.
Папка, где инсталирована программа, содержит сам бинарный модуль lvp и файл конфигурации lvp.conf, файл конфигурации автоофисов lvp_ao.conf, бинарный файл udp_sender. Здесь же создается файл lvp.log, в котором ведется журнал соединений, папки config_from_SQL, logs_fd, call_manager, logs, gk_routing, avoices.
Папка config_from_SQL служит для размещения конфигурационных файлов полученных из SQL.
Папка logs_fd служит для размещения бинарной информации tcp пакетов отладки.
Папка call_manager служит для размещения информации call_manager и протокола работы call_manager.
Папка logs служит для хранения отладочных протоколов соединений.
Папка gk_routing служит для хранения кеш информации для работы с гейткиперами.
Папка avoices служит для размещения голосовых сообщений в rtp формате. Поставляется программа конвертации wav файлов в rtp и наоборот. В папке создаются папки с названиями конкретных автоофисов для использования специальных версий сообщений. Сообщения ищутся в следующем порядке папка конкретного автоофиса, дальше в общей папке.

Начало конфигурирование LVP

1. Определяем интерфейсы, с которыми работает софтсвич.
2. **Для этого создаем секцию [Interfaces].**
3. Здесь мы определяем названия интерфейсов и IP адреса, которые им соответствуют, например
   eth0 10.10.10.10
   eth1 192.2.2.2
   out ifconfig:eth1:0
   Последняя команда определяет интерфейс out который определяется из назначенного системой интерфейсу eth1:0.
4. **Определяем основные параметры программы, для этого создаем секцию [Global].**
5. Устанавливаем уровень отладочной информации ddl=0-7. Значение 0 определяет, что отладочная информация не выводится, значение 7 – максимум информации даже записываются содержание udp пакетов.
   Рекомендуемое значение 5.
6. Определяем IP адрес и порт, который будет слушать LVP, используя комманду Default_Incoming=название интерфейса. По умолчанию, слушается порт 1720, если вы хотите изменить, используем команду Tcp_Listen_Port=новое значение.
7. Определяем IP адрес, с которого LVP будет делать соединения по умолчанию, используя команду Default_OutGoing=название интерфейса.
8. Определяем максимальное значение одновременных соединений, для блокировки зацикливания командой MaxPhoneConnections= количество.
9. Ограничиваем длительность звонков Max_Connection_Duration= длительность в секундах.
10. Определяем значения основных таймаутов. Для этого используем команду TimeOuts=TimeOut0, TimeOut1, TimeOut2, TimeOut3, TimeOut4, TimeOut5, TimeOut6
    TimeOut0 определяет время, которое ожидает LVP прихода сетап пакета, входящего плеча после открытия TCP соединения;
    TimeOut1 определяет время открытия TCP соединения к узлу, на который мы делаем соединение;
    TimeOut2 время, которое мы ждем для получения алертинга или прогресс сообщения;
    TimeOut3 время ожидания коннект сообщения после получения алертинга или прогресс сообщения;
    TimeOut4 зарезервированное поле должно быть 0, используется для отладочных целей;
    TimeOut5 время для закрытия TCP , за которое мы даем возможность пересылку сообщения RELEASECOMPLETE
    TimeOut6 время ожидания udp пакетов, если их нет, значит, соединение надо закрывать (нет звука) .
11. Определяем, куда пишем «экаунтинг» командой Calls_Log_File=bill/lvp_v3.txt. Здесь будет храниться билинг информация каждого плеча, состоявшихся звонков в формате
    sp_lvp_billing_second
    id_call,'ani','dnis','session_start','Connect_start','Disconnect_time',time_in_sec, Release_code,'IP_source','IP_destination',Type_call,'Account',Quality,Bytes_receive, Byte_Send, Is_it_user,'Listen_IP','Dest_IP'
    Здесь sp_lvp_billing_second – служебное слово;
    id_call – номер файла, где храниться отладочная информация;

ani – номер абонента А
dnis – номер абонента Б
session_start – время начала соединения
Connect_start – время начала разговора
Disconnect_time – время окончания разговора
time_in_sec – продолжительность разговора в секундах
Release_code – код завершения соединения
IP_source – IP адрес входящего плеча
IP_destination – IP терминирования разговора
Type_call – 0 мы терминировали, 1 мы оригинировали
Account – номер или название абонента, на которого мы относим этот звонок
Quality – индекс качества сервиса для этого звонка
Bytes_receive – количество байтов udp пакетов, которые мы приняли
Byte_Send – количество байтов udp пакетов, которые мы передали
Is_it_user – 0 звонок сервера, 1 звонок пользователя
Listen_IP – IP адрес, который слушает LVP
Dest_IP – IP адрес, на который мы инициировали звонок (в случае гейткипера он может быть другим, чем IP_destination).

12. Определяем глобальный "эккаунт", на который будем относить транзитный исходящий трафик, используя команду Account=name, где name название или номер счета пользователя.
13. Для ограничения возможности маршрутизации трафика на тот же узел, откуда пришел трафик, мы используем команду connect_to_friendly_ip_disabled. Эта команда делает невозможным посылку трафика даже на другой IP, если он прописан под тем же именем.
14. Для использования ряда команд управления количества звонков, нужен запуск КоллМенеджера, для этого используем call_manager.
15. Устанавливаем адрес на котором принимаем SIP запросы, например listen_sip 10.10.10.1 5060.
16. **Создаем секцию [Aliases]** . В этой секции мы связываем названия узлов с IP адресами. Например, gw_name 10.10.10.10,10.10.101.1. Количество IP адресов ограничивается только количеством символов в строке.
17. **Для определения узлов, которые имеют право посылать на нас трафик, создаем секцию [Friendly_Hosts]**. Здесь мы определим список узлов и правила обработки звонков согласно следующему формату
    Host translation_rule [maxconn=N] [maxconn_per_min=NN] [ani_allow=ani_allow] [account=aon | IP | name] [auto_progress|auto_proceeding|no_proceeding] [udpacct=host] [quality=n] [country=xx] [file_billing] [unique_dnis] a_o name
    -translation_rule правило преобразования dnis, которое может состоять из 8-ми пар типа -HH +HHH, например -001 +7 -002 +1 +96. Правила отрабатываются попарно слева на право до первого совпадения правила со знаком минус. Последнее правило может быть непарное со знаком +.
    -maxconn=N определяет, какое максимальное количество соединений мы разрешаем с данного узла
    -maxconn_per_min=NN ограничивает количество новых соединений в минуту (защита от атак)
    -aon_authorization_filename название файла, в котором находится список номеров ani, для которых разрешен доступ, если его нет, то всем разрешен
    -account=aon | IP | name определяем на кого "билить" входящее плечо aon на номер в ani, IP на IP адрес, | name на конкретный счет, По умолчанию на IP адрес;
    -auto_progress опция, которая указывает, что сигнализация непередается между двумя плечами, LVP самостоятельно отрабатывает каждое плечо;

- udpacct=hosts определяет, куда надсылать udp «экаунтинг»;
- quality=n определяем номер качества сервиса для этого узла;
-country=xx определяем страну происхождения, например, country=ru
-file_billing записываем билинг информацию только в файл, SQL не задействуется
-unique_dnis разрешаем только одно соединение на номер
-a_o name запускает скрипт автоофиса name с расширением + , если опция auto_progress|auto_proceeding|no_proceeding не включена, высылается сообщение connect.

18. **Следующий шаг – это создание секции [Bind_Routing]** для определения параметров соединения исходящего плеча. Здесь мы можем определить, какой IP адрес используется, на какой порт должно быть соединение, является ли узел гейткипером использовать ли кеш для гейткипера. Формат команды в секции [Bind_Routing] следующий
to_alias_or_ip send_from_alias_or_ip [port=N] [codec] [gk | gk_cache]
-to_alias_or_ip название или адрес узла, на которое идет соединение;
- send_from_alias_or_ip – название или адрес интерфейса с которого делаем соединение;
-port=N – определяем номер порта, на которое мы делаем соединение, по умолчанию 1720;
-codec – кодек g711a,g729r8,speex,iLBC который мы принудительно выбираем для этого соединения;
- gk | gk_cache – определяет, что узел есть гейткипер, и мы не используем | используем кеш (кеш удобно использовать в случае, когда гейткипер может быть недоступным).
19. Следующим шагом может быть определения правил транзитного доступа одних серверов к другим. **Для этого создаем секцию [Connect_Access]**, в которой используем две команды
host_from allow host_to_1, host_to_2, host_to_3,...
host_from deny ip_to_1 host_to_1 ip_to_2 ip_to_3 host_to2 ...
Команды разрешают транзит из host_from к host_to, или запрещают. Есть служебное обозначение всех узлов any. Например,
any deny 10.10.10.1 – команда блокирует все звонки на сервер 10.10.10.1.
Эти команды имеют приоритет выше глобальной команды connect_to_friendly_ip_disabled.
20. **Наконец мы можем приступить к секции маршрутизации трафика [Routing] .** Здесь мы описываем направления и перечисляем маршруты. Направления начинаются с «+». Формат команды
+[code] [hh:mm-hh:mm] [weekday] [+nnn] [quality=N] [country=CountryName] [account=number] [min_phone_len=nnnn]
- +code первые цифры телефонного номера, например +7095, может быть только +;
- hh:mm-hh:mm – значит, что направление работает только ограниченные часы;
- weekday – работает в конкретные дни недели { 0/sun = sunday, 1/mon = monday , ... 6/sat = saturday }, например, 0,1,sat;
+nnn (здесь nnn любые цифры) значит, что направление будет работать только для ani, которые начинаются на nnn. Например, +7095;
- quality=N значит, что направление работает только, когда качество сервиса равно N, например quality=4;
-country=CountryName значит, что направление работает, только, когда страна происхождения трафика равна CountryName (2 буквы). Например, country=ru;
account=number направление работает только для пользователя, определенного в number;
-min_phone_len=nnnn – параметр, определяющий минимальное количество цифр в

номере;
Примеры направлений
+ quality=5
+7095

21. В каждом направлении прописываются маршруты в порядке очередности. Маршруты, описываются командой формата
target_host_alias [+/-]prefix [TimeOut_N1,... TimeOut_N6] [max_line=n] [Send_From_Interface]
где
-target_host_alias узел на который отправляем трафик;
-[+/-]prefix правило преобразования префиксов (может быть парой) Например -7095 +9;
-TimeOut_N1,... TimeOut_N6 позволяют переопределить глобальные таймауты для конкретного маршрута, если мы хотим оставить какой-то таймаут глобальный мы там пишем «*» (нулевой таймаут в маршруте изменить нельзя) ;
- max_line=n определяет максимум соединений на данный узел для конкретного соединения (нужен колменеджер);
- Send_From_Interface позволяет указать название или IP адрес интерфейса, с которого мы будем посылать трафик (этот интерфейс имеет приоритет над интерфейсом в секции [BindRouting]
В качестве маршрутов могут быть служебные слова
auto_connect_lvp – запускает внутренний скрипт дебитной платформы;
mssql - запускает стандартную функцию выбора маршрута из SQL;
virtual_number - запускает стандартную функцию выбора маршрута для виртуального номера из SQL;
run_program name_program parameter - запускается внешняя программа name_program с параметрами account=account,aon=aon,called_phone=called_phone, parameter, например, run_program /bin/echo nc@yahoo.com;
pause N – делает паузу на N секунд.
a_o name [ext=n] [no_connect] – Запускает скрипт автоофиса записанного в файле lvp_ao.conf, или в SQL; ext=n определяет, какое расширение автоофиса запускаем; опция no_connect определяет, что мы вызываемой стороне connect не высылаем до начала исполнения скрипта, скрипт должен самостоятельно позаботиться о посылки «коннекта».

Использование SQL сервера для маршрутизации трафика

Для интеллектуальной маршрутизации трафика желательно использовать SQL сервер. На данный момент поддерживается MS SQL сервер. Смысл использования SQL состоит в том, чтобы дополнять существующие секции LVP [FriendlyHosts] – список узлов от которых мы принимаем вызовы , [BindRouting] – выбор параметров для инициирования вызовов, [ConnectAccess] – разрешение на транзит вызовов, [FriendlyUsers] – таблица пользователей для офлайновой аутентификации, [Routing] таблица направлений и маршрутов, [Outgoing_prefixes] правила преобразования префиксов для инициирования вызовов. Последняя конфигурация всегда сохраняется на LVP, таким образом, у нас всегда есть возможность продолжать работу даже в случае отсутствия связи с билингом. При записи билинговой информации, билинг возвращает сигнал, надо ли перегрузить, обновить информацию. При получении такого сигнала происходит обновление всей или части информации. Для того чтобы LVP начал взаимодействие с билингом надо включать в соответствующие секции команды.
Рассмотрим команды в секции [Global]

; ^^ serverIP ^^login ^^ encrypted_password ^connect_TimeOut ^query_TimeOut
Данная команда определяет, что сервер находится по адресу 10.10.1.6, пользователь lrt, зашифрованный пароль b2if234p, таймаут на соединение 5 секунд, таймаут на выполнение запроса 10 секунд.
auto_offices_from_sql – определяет, загружать ли тексты скриптов с SQL. Если мы хотим пользоваться текстами с SQL и текстами описанные в lvp_ao.conf, мы должны добавить команду auto_offices_include_static_file.
friendly_users_from_sql – разрешает загружать таблицу пользователей для офлайновой аутентификации.
connect_access_from_sql – загружает таблицу разрешения на транзитный трафик.
bind_routing_from_sql – загружает таблицу параметров для инициирования вызовов
routing_from_sql – загружает таблицу направлений и маршрутов.
Установлены правила приоритетов. Наивысший приоритет имеют маршруты из конфигурационного файла, далее маршруты из SQL.
Следует заметить, что маршруты выбираются согласно правилам, направление, качество, страна происхождения, время, ani, account. То есть качество сервиса имеет наибольший приоритет. Есть специальные номера качества 1-4. Направления без качества равносильны качеству 1. Направление 4 – самое высшее качество, пользователи качества 4 могут пользоваться направления для 1-4 качества, пользователи качества 3 могут пользоваться направления для 1-3 качества, пользователи качества 2 могут пользоваться направления для 1-2 качества, пользователи качества 1 могут пользоваться направления только для 1 качества. Для качеств от 5 и выше направления могут быть только с тем же качеством.
Для загрузки таблицы узлов, которые имеют право посылать на нас трафик в секции [FriendlyHosts] надо добавить служебное слово
FH_Sql_Server
Данное слово служит как обычное название узла и к нему можно добавлять параметры.

Рассмотрим конфигурационный файл .

```
# example of configuration file for linux voice proxy(lvp) program
# lines that begins with symbol "#" or ";" are ignored (comments)
# all symbols after ";" are ignored (comments)
;
;

[Aliases]
# format:
# host alias    ip_address [, ip_address ...]
test   100.16.45.114,111.16.44.4                    ; Здесь определяются
                                                    символьные обозначения серверов с
                                                    которыми мы работаем
test1 100.16.11.2
host_admin                                          10.10.10.10

[Friendly_Hosts]
#list of aliases that can establish a phone connection
#to our voice proxy program
#(other words - can open port 1720)
#all other hosts will be rejected during opening tcp port
# syntax: HOST  [[+/-]prefix] [aon_authorization_filename] [account=aon | IP]
                                                    [auto_progress]    [maxconn_per_min=N]
                                                    [quality=NN] [country=cc] [udpacct=host]
```

; aon_authorization_filename – файл со списком номеров ani, которые могут звонить
example: cisco5300 -6000 aon.txt ; account=aon опция которая позволяет
;
; записывать экаунтинг на пользователя с ani или IP (по умолчанию на сервер).

; maxconn_per_min определяет количество звонков, которым ограничивается данный сервер, quality=NN устанавливает качество сервиса для этого сервера, country=cc устанавливает страну происхождения трафика, udpacct=host определяет сервер, который получает udp аккаунтинг для мониторинга трафика.

cisco_definity aon_definity.txt
cisco_another
cisco_another2 +3000
note: prefix(if exist) MUST start with '+' or '-' character!

; Prinemaem zvonki ot vseh hostov!
unknown_host -1234 +14459751 maxconn=2 file_billing auto_progress maxconn_per_min=3; принимаем звонки от всех серверов (используем преопределенное название сервера)
; екаунтинг записываем в текстовый файл
; при приеме звонка выдаем прогресс сообщение и сетап и фасилити нетранзитем.
; транзитем только коннект и алертинг
; здесь в dnis мы убираем префикс и добавляем новый ; Максимальное количество одновременных соединений с этого сервера 2
; здесь в минуту сервер может инициировать до 3 соединения.

test1 udpacct=host_admin quality=2 country=ru ; Здесь мы разрешаем принимать трафик с test1 и посылаем udp екаунтинг на компьютер host_admin
; здесь мы принимаем, что качество сервиса

равно 2, страна происхождения звонка равна ru

```
[InterFaces]
```
; определяем обозначения интерфесов с которыми работает LVP

```
eth0 11.16.45.101
eth0:3 11.16.44.81
lo   127.0.0.1
[Global]
```
; обределяем глобальные параметры

```
ddl=5 ; default debug level. correct values range 0..7 (0-lower, 7-higher)
      ; recommended value = 4 (0=no any output&files debug only billing aviable)
      ; 6 - be careful, all tcp messages saved in logs_fd directory !!!
      ; 7 - be careful, full udp send/recv debug !!!
      ; level 6 and 7 is not recommended to set for a long time period
Calls_Log_File= bill/lvp_v3.txt ; billing information file
```
; определяем файл для текстовой записи екаунтинга

```
Account=5202 ; global user account
```
; определяем имя пользователя на которого записываем исходящий трафик

```

```
; входной трафик пишется на IP адрес.

```
TimeOuts=5,5,40,120,0,3,360 ; (global  TimeOut_N0,... TimeOut_N6)
```
; определяем таймауты

```
    ; TimeOut_N0 = receiving SETUP packet message from ORIGINATE side
    ; TimeOut_N1 = opening port 1720 on target host
    ; TimeOut_N2 = receiving ALERTING packet message from ANSWERING side
    ; TimeOut_N3 = receiving CONNECT packet message from ANSWERING side
    ; TimeOut_N4 = for debug purpose
    ; TimeOut_N5 = pause after end of the call
    ;    (with a hope that Originate client will initiate a break of tcp connection)
    ; TimeOut_N6 = timeout for udp data.
    ;   If no udp data available from OR or AN sides for a specified seconds
    ;    Then ReleaseMSG with Cause=Normal, unspecified(31) sent to ORiginate
    ;   Note: TimeOut_N6 start checking for udp data after ConnectMSG only.
; all TimeOuts values are in seconds
MaxPhoneConnections=160 ; max number of simultaneous phone connections
Default_Incoming=eth0:3 ; interface for listening tcp 1720 port
Default_OutGoing=eth0:3 ; interface from which program will connect to 1720
                        target/remote host
listen_sip 10.10.10.1 5060
```
; interface for listening SIP

```
Max_Connection_Duration= 3600; in seconds
user=proxy
```
; имя пользователя под каким запускается СофтСвич LVP

```
connect_to_friendly_ip_disabled
```
; Запрещает делать звонки на тот же сервер (даже если IP разные)

```
connect_access_from_sql                                    ; Загрузит список
                                                           разрешенных или запрещенных
                                                           серверов с SQL

[Routing]                                                  ;
                                                           секция, которая определяет правила
                                                           маршрутизации трафика
#format:
# +phone pattern [time of call hh:mm-hh:mm] [weekday]
# weekdays { 0/sun = sunday, 1/mon = monday , ... 6/sat = saturday }

# first  target_host_alias [+/-]prefix TimeOut_N1,... TimeOut_N6 [Send_From_Interface]
# second ......          TimeOut_N1,...N6 [Send_From_Interface]
# third  ......
; -------------------------
; local TimeOuts(if specified) for the server override global Timeouts values
; TimeOut_N1,... TimeOut_N6 = see global description of timeouts
; Note: * = means global value timeOut, 0 = No TimeOut
; Example: AS53_ia     +11##       15,20,*,0,3
;          ^^^^^^^^^^host ^^^^^prefix ^^^^^^^^^^timeOuts(N3,N6=global N3,N6;N4=absent)
;
;

+14459751                                                  ; Здесь
                                                           все звонки, что начинаются на 14459751
                                                           будут идти
test   -14459751 +3899                                     ; на сервер test, при
                                                           этом, префикс сначала убирается и
                                                           добавляется 3899
test1  -14459751 +3899                                     ; если на первом
                                                           сервере неуспешная попытка пробуем
                                                           второй

+       host=test                                          ; Здесь
                                                           все звонки от сервера test   посылаем по
                                                           следующим маршрутам
test1   +3899

+14459751  +7095                                           ; Этот
                                                           маршрут срабатывает если ani
                                                           начинается на 7095
test1  -14459751 +3899

+14459751  +7095 quality=2                  country=ru     ; This dial-
                                                           pear runs if ani begins on 7095 if quality
                                                           service 2 and country of origination is equal
                                                           to ua
test  -14459751 +3899                                      ; Последнее
                                                           направление будет выбираться только
                                                           тогда, когда качество сервиса сервера
                                                           больше или равно качеству
                                                           направлению
```

```
[Connect_Access]                                                    ; Секция,
                                                                    которая описывает разрешения и
                                                                    запрещения соединения между
                                                                    серверами
;host_from allow host_to_1, host_to_2, host_to_3,...
;host_from deny ip_to_1 host_to_1 ip_to_2 ip_to_3 host_to2 ...
; ANY  deny ip_to                                                   ; Зпрещает
                                                                    всем делать звонки на ip_to

;example:
10.10.128.9 deny 10.10.128.19
ANY deny 10.10.110.229
test allow test1, 10.10.128.10
10.10.128.9 allow 10.10.128.9
```

Последнее направление будет выбираться только тогда, когда качество сервиса сервера больше или равно качеству направлению.

Начальное конфигурирование довольно простое задание. Весь смысл состоит перечесть все сервера, с которыми вы работаете. Расставить префиксы. Дальше прописать все маршруты для терминации. Заметим, что экаунтинг состоит из двух звонков: - один входящий, второй исходящий. Входящее плечо, по – умолчанию, записывается на IP адрес, а исходящее на пользователя Account (в нашем примере 5202).

Кластеризация

Количество одновременных соединений ограничивается лицензионными условиями и мощностью компьютера на котором установлен софтсвич. При полном проксировании существенную нагрузку на компьютер дает проксирование UDP пакетов. Существенная нагрузка возникает на каналы связи и маршрутизаторы. Для этого разработан протокол кластеризации LVP. Для этого определяем мастер(master) и много «слейвов» (slave). Основная задача мастера отрабатывать сигнализацию и в случаи проблем со слейвами проксирование голоса. Мастер распределяет нагрузку между слейвами согласно направлению и времени соединении. Для работы кластеров на мастере должна быть включена глобальная опция Call_Manager.

Для начала определяем секцию

[Cluster]

Количество секций неограниченно, название каждого кластера должно быть уникальным, формат названия должен быть [ClusterN], где N любое число.

В секции определяем порядок распределения нагрузки формата

IP limit [direction] [period] [port]

IP – адрес слейва, limit – максимальное количество соединений, direction – направление, period – время суток, port – порт, который слушает слейв.

Например:

[Cluster]

127.2.0.1 200 +1 2:00-17:00 port=1720

127.2.3.1 100 +1 2:00-17:00 port=1720

localhost 1000 +1 2:00-17:00

127.2.0.2 200 port=1720

127.2.3.1 100 port=1720

localhost 1000

Здесь мастер для направления +1 для времени 2:00-17:00 будут работать два слейва, которые слушают порт 1720. В другое время суток и для всех других направлений будут работать слейвы 127.2.0.2 и 127.2.3.1. При исчерпании нагрузки слейвов будет работать сам мастер.

При работе со слейвами реализован следующий алгоритм. Слейв считается, что он рабочий если соединение с ним открывается меньше чем 3 секунды. Если 10 раз подряд слейв «тормозит» он считается перегружен и он отключается на 1 минуту. Если 10раз подряд время соединения превысило 10 секунд, тогда слейв считается «мертвым» и он отключается на 10 минут. Таймаут на соединение со слейвом установлен 7 секунд. Эти таймауты можно изменить используя глобальную команду например:

slave_timeout 3 7

Для того чтобы трафик friendly host использовал кластер надо в его праметрах включить cluster (название кластер) например:

[Friendly_Hosts]

10.10.10.1 cluster

Теперь нам осталось только прописать на слайве LVP, что он будет работать в режиме слайв с мастером. Для этого нужно указать секцию [masters] и дать список IP адресов.

Например

[masters]

10.10.10.4

10.10.11.2

Заметим, что мастер тоже должен быть прописан, как friendly host.

Протокол гарантированной доставки RTP

Основная проблема IP телефонии связана с качеством голоса. Качество голоса связано с потерями, неравномерностью прохождением пакетов. Для борьбы с этим разработан протокол передачи UDP пакетов. Здесь мы получаем задержку до 40 мс. Данный алгоритм гарантирует уверенную передачу голоса в каналах с потерями до 10%. Для этого нужно инсталлировать программу LVP_UDP (поставляется безплатно). В конфигурационном файле которой прописываются следующие параметры.

[interfaces]

local 127.0.0.1 ; обязательный интерфейс

eth0 10.10.10.1 ; интерфейс который слушает LVP , должны быть перечислены все

[global]

ddl=0

listen local eth0 ; слушаем все интерфейсы

udp_buffer ; включаем буфер для восстановления потерянных пакетов

start_duplicate ; включаем отсылку избыточной информации для восстановления потерянных пакетов.

Для использования протокола надо в секции [Bind_Routing] указать с какими IP мы будем использовать данный протокол.

Например

[Bind_Routing]

10.1.20.3 10.2.1.16 port=2020 lvp_udp

Рассмотрим Расширенные возможности.

Основными особенностями версии есть расширенные функции работы с SQL сервером и язык-интерпретатор для построения собственных скриптов.
Дополнительные функции работы с SQL связаны с возможностью управлением маршрутизацией трафика с SQL, Авторизацией серверов и пользователей. Работой с Автоофисами.
Интерпретатор позволяет добавить функциональность автоофисам. В наличии функции умеющие проговаривать числа, время. Команды if else endif (нет ограничений на вложения), do while (до 4 вложений).

Остановимся на SQL функциях. SQL сервер проводит статистику звонков и соответственно удобно дать ему возможность управлять маршрутизацией. Управлением SQL каждым звонком довольно сложная задача и при большом количестве звонков может быть непосильной. Для этого предлагается загружать таблицу маршрутизации на Софтсвич, а при изменении таблицы загружаем только изменении. Данное решение позволяет правильно маршрутизировать трафик даже при отсутствии связи с SQL. На софтсвиче всегда сохраняется копия последней таблицы. Удобно, чтобы корпоративные пользователи, у которых есть кредит, могли авторизоваться и делать звонки даже при отсутствии связи с SQL. Для этого происходит синхронизация таких пользователей с SQL.
Удобно, чтобы пользователи, могли изменять настройки своего Автоофиса через ВЕБ и дальше SQL. Для этого делается синхронизация с SQL.
Сами же Автоофисы расширены возможностью написанием скриптов.
Рассмотрим язык.
Основным критерием написания языка было его «легкопонятность» (байсик подобность).
В тексте Автоофиса встречаются, как команды Автоофиса, так команды интерпретатора.
Команды интерпретатора начинаются со слова operation. Рассмотрим команды настройки Автоофиса.

[main] ; Название Автоофиса

```
udp_accounting=10.10.2.1,10.10.2.6      ;       Посылаем акаунтинг для мониторинга
prompt_file=ru_say_hello_office         ;       Файл приветствия
information_file =ru_info               ;       Файл дополнительной информации (
                                                Сначала проигрывается prompt_file а потом
                                                information_file)
delay=2                                 ;       Задержка на ввод цифр для
                                                блокировки муссора
max_extension_length 4                  ;       Максимальное количество цифр в
                                                расширении (не надо ждать окончания
                                                набора)
access_international=9810       ;      ;       Код доступа к международной связи
                                                (для переадресации)
access_national=98            ; ...     ;       Код доступ к междугородней связи
                                                (для переадресации)
access_local=9          ; ...           ;       Код доступ к городу
                                                (для переадресации)
code_tech=810                           ;       Технический префикс
code_country=7          ; ...           ;       Код страны
code_city=7095          ; ...           ;       Код города
system                                  ;       если эта команда присутствует, можно
                                                менять параметры звонков такие,
;       как bani, baccount (счет на кого записывать билинг)

;
;       Когда наберем 92331122 получим -9, +7095, +810 =81070952331122  -
                                                местный
;       Когда наберем 9891612312322  -98, +7, +810=810791612312322      -
                                                междугородний
;       Когда наберем 981012122233445       -9810, +810=81012122233445  -
                                                международный
;
office_code=000111          ; ...       ;       Выбираем код Автоофиса для учета
                                                трафика на внутренние номера
        ;                               Например звонок на расширение +1 будет
                                                учитываться как на 0001111
redirect_code=*21;          ;           ;       Код для переадресации звонка
        ;                               Когда внутренний телефон наберет *21
                                                тогда он переведет звонок на любой номер,
;       используя преобразования описанные выше
pickup_access=13.18.12.21               ;       с какого сервера можно делать
                                                перехват звонков
voicemail_file leave_mail               ;       файл, который информирует, что
                                                оставьте сообщение
voicemail_pass prompt_pass              ;       введитие пароль для прослушивания
                                                почты
voicemail_prompt prompt_number          ;       введите номер сообщения для
                                                прослушивания

password 123672  +8                     ;       пароль доступа к голосовой почте
                                                расширения 8
password 123674  +1                     ;       пароль доступа к голосовой почте
                                                расширения 1
```

Рассмотрим команды Автоофиса
Расширения
Начинаются на +
Пример
+001
Дальше должны быть маршруты.
Например
10.10.10.3 +20 здесь +20 не префикс, а полный номер телефона. Здесь при успешном звонке работа Автоофиса закнчивается
Есть дополнительные команды
A_o main ext=5 info - это значит, что перейти на начало Автоофиса но проигрывать только информацию information_file и перейти сразу на расширение 5. Два последние параметры они опциональны.

pause=9 Устанавливает паузу в 9 секунд.
play_prompt file_name - проигрывает звуковой файл
voicemail - программа записи сообщений которая сначала проигрывает файл voicemail_file
voicemail listen - программа, которая сначала проигрывает файл voicemail_pass, а потом при правильном введении пароля voicemail_prompt и проигрывает файл, номер которого вы ввели. Если вы введете 0 то проигрывает все файлы, если *# то сотрет все записанные файлы.

Если обнаружена ошибка в команде то команда игрорируется.
При окончании списка команд в расширении Автоофис заканчивает свою работу.

Подключение Софтсвича к телефонной сети общего пользования по протоколу ISDN PRI

Создаем транк группы. Для чего изначально создаем секцию [trunk-group] .

В этой секции прописываем транк группы
например:

e1_out dchan 16 timeslots 1-15,17-31

e1_in dchan 47 cpe timeslots 32-46,48-62 echocancel=128 echotraining=70
echoconf=echo.conf buffer=buffers

Здесь, e1_out, e1_in произвольные названия групп, dchan 16 – Д канал (каналы номеруются попорядку, так 16-й канал второго потока будет 47), параметр cpe- обозначает, что наш поток будет клиент, timeslots 1-15,17-31 – определяет, какие каналы принадлежат группе, здесь установлено, что isdn switch-type primary-net5, echocancel=128 – время в миллисекундах для эхо подавления, echotraining=70 – время в миллисекундах для тренинга эхо подавления.

echoconf=echo.conf значит, что точная настройка параметров эхо компенсации находится в файле echo.conf. Пример файла

7916\11 0 0

74951\11 128 130

Здесь задается, что на входящие звонки с номером А и исходящие с номером Б, что начинаются на 7916 (количество знаков 11) не устанавливать эхо компенсацию,

что начинаются на 75951 (количество знаков 11) устанавливать эхо компенсацию с параметрами echocancel=128 echotraining=130.

Buffer=buffers значит, сто в файле buffers находится конфигурации управления буфером для номеров dnis, если звонок уходит на voip. Или номеров ani если звонок приходит с voip.

Пример

111111\6 5 7 ; все номера начинающиеся на 111111 будут иметь буфер 5 семплов(50мс), 7 преобразовываем в биты

00000111- правый бит – значит, что мы выключили эхо компенсацію, второй, что мы производим дтмф детекцию, третий, что мы выключили Т38 петлю.

Если мы хотим принимать звонки из транк группы, мы должны прописать название транк группы в секции [friendly_hosts]. Например

e1_out file_billing

e1_in -80 +380 -82 +38044 -810 +810 +38044
file_billing

Здесь e1_out, e1_in названия групп, file_billing – эккаунтинг записываем в файл, -80\11 +380 -82\9 +38044 -810 + -\7 +38044 - правила преобразования номеров Б слева направо (здесь\X количество знаков в номере). Примеры результатов преобразований:

80671111111 -> 380671111111

826123456 -> 380446123456

81012121234567 -> 12121234567

1234567 -> 380441234567

В списках секции [friendly_hosts], можно задать правила преобразования номеров А.

Например

e1_in1 A -8\11 A +7 A -\7 A +7495

Здесь, если номер А придет в формате 84951234567, мы его преобразуем в 74951234567.

Если придет только 7 знаков xxxxxxx, мы их преобразуем в 7495xxxxxxx.

Если у нас установлено дополнительное устройство для передачи и приема факса в протоколе Т38, мы добавляем запись в секции [global]

t38fax711 +000100 host

Здесь t38fax711 - название команды, +000100 произвольно выбранный номер, на котором будет происходить преобразование протокола, host название сервера или транк группы, через которые будет происходить конвертация. Вместо названия возможно использовать IP адрес сервера.

Дополнительно, можно прописать правила в секции [Bind_Routing]

Например

e1_in 127.0.0.1 ta=N tb=N na=N nb=N bani=4411111T ani_deny=deny.conf

Здесь, bani=4411111T будет проверять, чтобы номер А соответствовал маске 4411111, если номер отличается, будет он заменен к соответствию с маской. Маска может быть следующей

bani="1390(0,1,2,3,4).." количество знаков должно быть 7, 5-й знак должен быть из списка, два последних может быть любое число.

bani=4411111T здесь первые 7-мь знаков должны совпадать с маской, количество знаков может быть произвольным.

Ani_deny=deny.conf определяет файл со списков номеров A, для которых запрещено использование e1_in.

ta – тип вызывающего номера, tb - ...вызываемого, na номерной план вызывающего абонента, nb – вызываемого.

Типы номера: 0,1,2,3,4,6,7
PRI_TON_UNKNOWN 0
PRI_TON_INTERNATIONAL 1
PRI_TON_NATIONAL 2
PRI_TON_NET_SPECIFIC 3
PRI_TON_SUBSCRIBER 4
PRI_TON_ABBREVIATED 6
PRI_TON_RESERVED 7

Типы номерного плана: 0,1,3,4,8,9,15
PRI_NPI_UNKNOWN 0
PRI_NPI_E163_E164 1
PRI_NPI_X121 3
PRI_NPI_F69 4
PRI_NPI_NATIONAL 8
PRI_NPI_PRIVATE 9
PRI_NPI_RESERVED 15

Теперь мы готовы, чтобы использовать названия транк групп в секции [routing].

Подключение Софтсвича к телефонной сети общего пользования по протоколу ОКС7

Создаем транк группы. Для чего изначально создаем секцию [trunk-group] .

В этой секции прописываем транк группы
например:

ss7_test dchan 1 timeslots 2-5 ni 2 opc 0001 dpc 0002 echocancel=128 echotraining=130

Здесь opc 0001 код пункта сигнализации нашего софтсвича, dpc 0002 код пункта сигнализации противоположной стороны, ni 2 идентификатор сети. Остальные параметры аналогичны параметрам для протокола ISDN PRI.

Дальше мы производим настройки аналогично, как в предыдущем параграфе.

Рассмотрим синтаксис языка интерпретатора

Operation [command]
Здесь [command] - команда. Интерпретатор понимает определенные переменные и константы в виде текста или целого числа. Все переменные и константы в памяти сохраняются в виде текста. Есть переменные, которые могут интерпретироваться только, как текстовые, есть переменные, которые могут быть, как текстовые или целые. Различие между этими типами проявляются только при сравнении. Существуют следующие предопределенные константы

ani – номер абонента А
dnis – номер на котором работае Автоофис
ListenIP IP на котором работает сервер
originIP IP который инициирует вызов
originate_codec_id номер кодека integer constant
Null - пустая строка
result - результат выполнения последней операции
cm_account - id счета, который авторизирован в Call manager
false –‘0’
true –‘1’

Есть специальные пременные
baccount – системный счет для биллинга
bani - системный ani для команды PlaceCall и Call (их можно изменять только когда установлена опция Автоофиса system, после окончания звонков они восстанавливают значения по умолчанию)

Есть переменные только строки:
ip
account
baccount
bani
pin
dnis
ani
phone

Следующие переменные могут быть как строки, так и целыми
local
quality
book
country
time
money
num1
num2
..
num0
Есть обычные пременные, которые инициализируются в процессе выполнения скрипта. Например &text или $num. Здесь & значит, что переменная текстого типа, $ - переменная целого типа. Можно определити локальные переменные, которые будут видны только в конкретной процедуре и будут забываться при выходе. Например
@&text или @$num. @ - значит, что переменная локальная.
Last_Call_Release_Code
Введен массив Array, у которого 100 элементов, начиная с 0 и оканчивая 99.
Есть два альтернативных методы доступа к элементу массива. Первый – прямой, например, Array10; Второй – косвенный например
read num0 num1 прочитать значения Array[num1] в num0
put num0 num1 записать значения Array[num1] в num0

Начнем с команд set:
SET timeout 5 таймаут на ввод цифр

set local 1 устанавливается локальное приведение номера телефона к нормальному формату
set country ua|ru|en… устанавливается параметр страна происхождения трафика
set quality 2 - устанавливается качество сервиса
set clearinput 1 - устанавливается флаг очистки буфера ввода перед командой input
set release_code ## набор кнопок после которых отвечающая сторона может завершить два плеча звонка

set pause_after_connect 0 установка времени задержки на прием клавиш от клиента после коннекта (для того чтобы убрать улюлюканье) на Автоофис

set language [ua|ru|en|...] установка языка, действует для произношения чисел

set interrupt_say 1 [0] устанавливает режим прерывания звука при выполнений функций say_number say_phone

set signal_busy 1 [0] --- default устанавливаем режим для PlaceCall когда при неудачном завершении давать сигнал занято 5секунд

set cm_account num0 устанавливаем колМенеджеру id пользователя, если есть такой же id, то такие звонки перерываются.

set fh_file_billing 1|0 если 1 то будет писаться билинг в файл, если 0 то в SQL
set digit_input_pause_after_connect_msg integer_value Меняет значения задержки восприятия клавиш после коннекта при звонке с Автоофиса
set Last_Call_Release_Code code .

Функции обработки строк и целых
LEFT num1 num2 num3 num1=left(num2, num3) num3 левых символов num2 в num1
RIGHT num1 num2 num3 num1=right(num2, num3) num3 правых символов num2 в num1
SHIFT num1 num2 num num3 num1= strcpy(&num2+ num3) отбрасываем num3 символов спереди
INT Num1 Num2 Num1=Int(num2) целое знацение числа num2
Incr num0 - увеличиваем на единицу
Decr num0 - уменьшаем на единицу
ADD num0 num1 num2 строковое сложение num0=num1+num2 например 777+777=777777

Команды if else endif. Есть следующие сравнения GE GT LE LT EQ NE. Например:

IF GE money num1
…
ELSE
…
ENDIF – конец условия
Возможно неограниченное количество вложений.
Команда цикла
do
…
while GE| GT| LE| LT| EQ| NE num1 num0
Возможно только 5 вложений.

Рассмотрим служебные функции:
Input prompt prompt_err num1 n1 n2 n3 - Проигрываем файл prompt если какая-то ошибка prompt_err, ждем на ввод кнопок до n1 или до # , минимум n2, повторяем n3 раз и с таймаутом для скрипта, ясли успешно result='1'
play_till_connect prompt_file Проигрывает по кругу файл до получения коннекта
play_till_alerting prompt_file Проигрывает по кругу файл до получения алертинга
record_start [file_name] начинает записывать собщение
record_stop заканчивает записывать сообщение

play_file time time>=0 Если time =0 или отсутствует то проигруем полностью файл. Если Time >0 тогда играем только небольше времени определенным параметром.

PlaceCall phone time – делаем звонок на номер phone и максимальным временем разговора time
call phone time ip - делаем звонок на IP адрес с номером phone и максимальным временем разговора time
kill_calls num0 - завершаем все сессии с id равным num0
local_user_aut account pin аутентификация локального пользователя, используя account и pin (используются пользователи загруженные с sql)

Возможно вызвать процедуры из SQL используя две функции:

exec num0 "" например exec num1 procedure_t account pin – здесь num1 получает результат выполнения процедуры, при успешном выполнении result будет равен 1.
Следующая функция
execa 10 procedure account pin записывает результат в Array начиная с 10 элемента. То-есть результат будет в Array[10] Array[11] и дальше. При успешном выполнении result будет равен 1.

Примеры вызова SQL процедур

Operation execa 0 GetPhones account - возвращает телефонную книгу абонента начиная с 0 по 9 (нулевой номер это повтор последнег звонка) в
Array[0] - Array[9]
execa 10 GetAni account возвращает в Array[0] количество элементов, а список в Array[1] - Array[.]

execa 0 authenticate account pin
Возвращает
1 - код ошибки Error
2 – id_customer
3 - balance
4 – quality

Error
0 - Success
1 - Customer not found
2 - bad password
3 - Balance of customer below zero
4 - account disable
5 - account blocked
6 - account is locked out

7 - customer is calling now
8 - customer agent blocked
9 - dealer blocked
10 - Server not found
11 - Prefix not present
12 - Real phone number is zero length
13 - Destination region not present
14 - Rate for country not defined

exec money authentic account pin аутентифицируем абонента и в результате получаем деньги

exec num0 unblock_call account - разблокируем счет
Num0
0 - разблокировано
1 - ошибка

Operation exec time authr account phone listenIP - Авторизует клиента на сервере listenIP для звонка на телефон phone и в результате получаем время разговора
Operation execa 0 authorize account phone listenIP
0- Errors
1- time insec
2- time in min

operation execa 0 Virtual account listenIP

0 Error code
1 account
2 number
3 time
4 min
5 sec
6 multi_connections {false|true}
7 id_customer
8 IP address

operation exec phone check_ani ani ; if existed phone 1 else 0

operation exec phone add_ani account ani ; if OK phone 1 else 0

operation exec phone del_ani account ani ; if OK phone 1 else 0

operation exec number0 ChangePhone account phone number ; if OK number0= 1 else 0

exec num0 add_phone account phone num9 Додаем номер phone в ячейку num9, если все нормально тогда num0=0
exec num0 del_phone account num9 Удаляем номер в книжке для пользователя account в ячейке num9, если все нормально тогда num0=0

Пример скрипта для Дебет – платформы с аутентификацией по номеру А.
Для вызова скрипта используем команду
A_o debit_ani ext=5
Если мы хотим авторизоваться только по номеру счета и пин – коду
A_o debit_ ext=5

```
[debit_]                                ;автоофис входа (предполагается выбор
                                        языка) авторизации по номеру карточки и пин
                                        коду
[debit__proc]                           ;автоофис  авторизации по номеру карточки и
                                        пин коду
[debit_ani_proc]                        ;автоофис для авторизации по ani
[debit_PlaceCall]                       ;автоофис для осуществления звонков
;       сначала мы перечислили все автоофисы, которые будут использоваться
[debit_ani]                              ;автоофис входа для авторизации по ani
system                                   ;опция, которая позволяет изменять
                                        системные параметры билинга
account office                          ;название (номер) счета, на который
                                        записывается билинг по умолчанию

+5                                       ;Для перехода используем расширение 5
operation const num0 0                  ;инициируем переменные
operation const num3 3
Operation Const book 0                  ; Is downloaded Phonebook
operation set country ru                ;устанавливаем страну
operation set language ru               ;устанавливаем язык

operation set quality 2                  ;устанавливаем качество сервиса по
                                        умолчанию

Operation Set local 1                   ; To normalise number localy
operation set clearinput 1               ; Before input clear key buffer
operation set signal_busy 1             ;включаем сигнал занято в конце перебора
                                        маршрутов

operation set interrupt_say 1           ;разрешаем перерывать информационные
                                        сообщения
pause 2                                                 ;подождем 2секунды, чтобы
                                        отсеять мусор
a_o debit_ani_proc ext=5                ;переходим на Автоофис, который должен
                                        провести аутентификацию
```

```
[debit_ani_proc]                         ;Автоофис аутентификации по ani
system                                   ; системный
account office
```

```
+5                                          ;расширение на котором начинаем
operation Const local 0                     ;инициируем константу, которая несет
                                            информацию, авторизован локально

operation set digit_input_pause_after_connect_msg 0 ; устанавливаем блокировку
                                            кнопок после коннекта в 0
Operation Const num6 ani                    ;записываем текст ani в num6
Operation Get account ani                   ;значение ani в account

operation local_user_aut account num6 ; проводим локальную аутентификацию по
                                            ani

operation IF EQ result true                 ; если аутентифицировали
operation Const local 1                     ; устанавливаем влаг
A_o debit_PlaceCall ext=5                   ; переходим к звонкам
operation endif

                                            ; Мы локально не аутентифицированы

operation do                                ; Запускаем цикл для соединения с SQL

operation const num5 1                      ; инициируем

operation execa 0 authenticate account  num6 ;аутентифицируем пользователя в
                                            SQL

operation if EQ  result false               ; если нет связи
operation const num5 0                      ; будем возвращаться в начало цикла
pause 5                                     ; ждем 5 сек
operation endif

operation while EQ  num5 false              ; завершение цикла

operation IF NE Array3 false                ; Если надо изменить качество сервиса
operation set quality Array3                ; устанавливаем новое качество
operation endif

operation Get money Array2                  ; Устанавливаем количество денег на счету
operation Get num4 Array0
operation IF GT num4 false   ; if not OK 1 ; проверяем не заблокирован ли
                                            пользователь

operation const num5 7                      ; пользователь заблокирован

Operation IF EQ Array0 num5  ; if calling проверяем тип блокирования 7

Operation exec num4 unblock_call    account   ; разблокируем

operation IF EQ Array4 false                ; если нельзя одновременным звонкам
operation set cm_account  Array1            ; устанавливаем id колменеджеру
pause 12                                                    ; даем время
                                            завершить параллельным звонкам
```

```
operation endif

A_o debit_PlaceCall ext=5                ; идем делать звонок


Operation ENDIF                          ; end calling

a_o debit__proc  ext=5                   ; посколько счет неактивен или неверный
                                         идем авторизоваться по номеру счета

operation ENDIF                          ; end not OK 1


operation if ne local true              ; if not local     ;если мы авторизованы не
                                         локально

operation IF EQ Array4 false             ; если нельзя одновременным звонкам
operation set cm_account  Array1         ; устанавливаем id колменеджеру
operation endif

operation IF LE money 1                  ; если кончаются деньги говорим сколько
play_prompt Your_account_has
operation say_number money
operation endif


Operation ENDIF                          ; end not local


A_o debit_PlaceCall ext=5                ;идем звонить



[debit_PlaceCall]

system
+5
operation set fh_file_billing 0          ; устанавливаем,что пытаемся билинг писать
                                         сначало в SQL
operation Set timeout 5                  ; на окончание ввода ждем 5секунд

operation do                             ; можно звонить много раз
Operation Input ask_phone say_bad_digits_count  num9  16 1  3 ; вводим номер
                                         телефона
Operation IF EQ result false
Operation Shutdown say_bad_digits_count
Operation ENDIF
operation set clearinput 0

Operation Const time 001   ; we would like how much on account
operation IF EQ num9 time                ; узнаем хотим ли мы узнать сколько у нас
                                         денег на счету
```

```
Operation IF EQ local false              ; говорим сколько денег для нелокальных
                                         пользователей
play_prompt Your_account_has
operation say_number money
operation endif
A_o debit_PlaceCall ext=5                возвращаемся в начало для ввода нового
                                         телефона
operation endif                          ; end how much

Operation Const time 030   ; we would like how quality
operation left num5 num9 3               ; если введен префикс 030 мы меняем
                                         качество сервиса на 3
operation IF EQ num5 time
operation set quality 3
operation endif              ; end how quality



Operation Const time 000                 ; we would like if virtual

operation IF EQ num5 time                ; проверяем на виртуальный номер

operation do

operation const num5 1

operation execa 0 Virtual num9 listenIP

operation if EQ result false
operation const num5 0
pause 5
operation endif
operation while EQ num5 false

operation Get num4 Array0
operation IF EQ num4 false               ; if error false
operation Get baccount Array1            ; устанавливаем билинг счет Array1
operation Get phone Array2
operation Get time Array3

operation IF EQ Array6 false
operation if NE cm_account Array7
operation set cm_account Array7
operation endif
operation endif

operation Get ip Array8
operation IF EQ Array8 0                 ; if regular virtual phone
operation placecall phone time
operation else                           ; if virtual ip and phone
operation call phone time ip
```

```
operation endif

a_o debit_PlaceCall__ ext=5

operation else                          ; else error false

operation IF EQ num4 7                  ; if calling now

Operation exec num4 unblock_call account

operation IF EQ Array6 false
operation if NE cm_account Array7
operation set cm_account Array7
operation endif
operation endif

pause 12
operation endif

a_o debit_PlaceCall ext=5               ;после звонка на виртуальный возвращаемся
                                        на новый номер


operation endif                         ; endif calling now

operation endif                         ; endif error false

operation endif                         ; end if virtual




Operation Const time 00                 ; проверяем, хотим ли мы повторить
                                        последний номер
Operation If NE num9 time               ; check is last number

 Operation Const time 0

 Operation If EQ num9 time              ; check is number from phonebook
  Operation Input ask_phone__ say_bad_digits_count  num9  1 1 1
  Operation IF EQ result false
   Operation Shutdown say_bad_digits_count
  Operation ENDIF
  Operation IF EQ book 0

operation Const num8 8
operation do
operation Decr num8
operation IF LT num8 0
Operation Shutdown say_problem_connecting
operation Endif
```

```
operation const num5 1

   Operation execa 0 GetPhones account

operation if EQ  result false
operation const num5 0
pause 5
operation endif

operation while EQ  num5 false

     Operation Const book 1
  Operation ENDIF

  Operation read phone num9              ; from phonebook

 operation else
  operation get phone num9
 operation endif

operation else                           ; last number
 operation if EQ phone null ;num9
     Operation IF EQ book 0

operation Const num8 8
operation do
operation Decr num8
operation IF LT num8 0
Operation Shutdown say_problem_connecting
operation Endif

operation const num5 1

   Operation execa 0 GetPhones account

   operation if EQ  result false
   operation const num5 0
   pause 5
   operation endif

operation while EQ  num5 false

        Operation Const book 1
     Operation ENDIF

    Operation Get phone Array0           ; берем последний набранный номер

 operation endif
operation endif
```

```
Operation IF EQ local true                ; если клиент локальный, максимум
                                          разговора 6000секунд

 operation Const time 6000

operation else                            ; в противном случае проводим авторизацию

operation do

operation const num5 1

Operation execa 0 authorize account  phone listenIP

   operation if EQ  result false
   operation const num5 0
   pause 5
   operation endif

operation while EQ  num5 false

operation Get time Array1

 Operation IF LE time false               ; если нету времени заканчиваем сеанс
  Operation Shutdown say_problem_connecting
 Operation ENDIF


play_prompt you_have
operation say_number Array2 minutes seconds 0      ; говорим сколько времени

Operation ENDIF


Operation Const num2 6000

Operation IF GT time num2                 ; если время больше 6000 мы его
                                          ограничиваем
Operation GET time  num2
Operation ENDIF

OPeration GET baccount account            ; устанавливаем билинг счет

operation set signal_busy 1               ; при неуспешном звонке будем давать
                                          сигнал занято

operation PlaceCall phone time            ; делаем звонок по маршрутам
Operation Get Array0 phone                ; Устанавливаем номер как последний
                                          набранный
operation set clearinput 1                ; устанавливаем флаг очистки буфера ввода
operation set signal_busy 0               ; восстанавливаем значение
operation while EQ true true              ; завершаем цикл
```

```
[debit__proc]                                                    ; скрипт для авторизации
                                                  по номеру счета и пину
system
delay=1
account office
+5

operation set fh_file_billing 0
operation set digit_input_pause_after_connect_msg 0

operation incr num0

operation if GT num0 num3
operation Shutdown say_problem_connecting
operation endif

operation Set timeout 5
operation Input ask_account say_bad_digits_count account 6 3 3
operation IF EQ result false
a_o debit__proc  ext=5
operation EndIF
operation Const Array40 0

operation len num4 account
operation if EQ num4 6
operation left num4 account 3
operation IF EQ num4 0                    ; if virtual

operation Input ask_account_ say_bad_digits_count_ num5 16 0 1

operation len num4 num5
operation if GT num4 0
operation ADD account account num5
operation endif

operation Const num8 8
operation do
operation Decr num8
operation IF LT num8 0
Operation Shutdown say_problem_connecting
operation Endif

operation const num5 1

operation execa 0 Virtual account listenIP

operation if EQ  result false
operation const num5 0
pause 5
operation endif
```

```
operation while EQ  num5 false

operation Get num4 Array0
operation IF EQ num4 false              ; if error false
operation Get baccount Array1
operation Get phone Array2
operation Get time Array3

operation IF EQ Array6 false
operation set cm_account Array7
operation endif

operation else

operation IF EQ num4 7                  ; is calling now
operation Get baccount Array1
operation Get phone Array2
operation Get time Array3

Operation exec num4 unblock_call    account

operation IF EQ Array6 false            ; is multiconnection false
operation set cm_account Array7
pause 12
operation endif                         ; end if multiconnection false

operation else
a_o debit__proc ext=5

operation endif                         ; end if calling now

operation endif                         ; end if error false
operation Get ip Array8
operation IF EQ Array8 0
operation placecall phone time
operation else
operation call phone time ip
operation endif
a_o debit__proc ext=5

operation endif                         ; end if virtual
; to much in account and to transfer to pin
operation Const Array40 1
operation shift Array41 account 6

operation endif


 operation set clearinput 0
 operation Input ask_pincode say_bad_digits_count pin 6 6 3
 operation IF EQ result false
  a_o debit__proc ext=5
```

```
 operation EndIF

operation local_user_aut account pin
Operation IF NE result false                ; if local user

operation const local 1
Operation Set local 1
operation Const money 1000
A_o debit_PlaceCall ext=5

operation ENDif



operation Const num8 8
operation do
operation Decr num8
operation IF LT num8 0
Operation Shutdown say_problem_connecting
operation Endif

operation const num5 1



operation execa 0 authenticate account  pin

operation if EQ  result false
operation const num5 0
pause 5
operation endif
operation while EQ  num5 false

operation Get Array55 Array1

operation Get money Array2

operation IF NE Array3 false
operation set quality Array3
operation endif

Operation Get num4 Array0
Operation IF GT num4 false              ; If not OK
operation const num5 7

Operation IF NE Array0 num5             ; If not just calling

 play_prompt wrong_account              ; else  - is just calling
 operation else                 ; just calling now

 operation Get baccount account
 operation Get money Array2
```

```
operation set fh_file_billing 0

Operation exec num4 unblock_call    account

operation IF EQ Array4 false
operation set cm_account  Array1
pause 12
operation endif
A_o debit_PlaceCall ext=5


Operation ENDIF                          ; end just calling



operation set clearinput 1               ; Before input clear key buffer
  A_o debit__proc ext=5
Operation ENDIF                          ; end not OK


play_prompt Your_account_has
operation say_number money

operation IF EQ Array4 false
operation set cm_account Array1
operation endif

A_o debit_PlaceCall ext=5


[debit_]
system
account office


+5

operation const num0 0                   ; нулевая попытка
operation const num3 3                   ; устанавливаем максимальное число
                                           попыток
Operation Const book 0                   ; устанавливаем флаг, что записная книга,
                                           еще не загружена
operation set country ru                 ; устанавливаем страна ru
operation set quality 2
Operation Set local 1                    ; разрешаем локальное приведение номера к
                                           номальному виду( используем глобальные
                                           правила перобразования)
operation set signal_busy 0              ; устанавливаем не проирывать сигнал занято
                                           при неуспешной попытки звонка

operation set interrupt_say 1            ; разрешаем прерывать приглашения
```

```
operation set clearinput 1                ; устанавливаем флаг очистки буфер ввода
                                          цифр

a_o debit__proc ext=5
```

Дополнение

Инетерфейсы функций синхронизации Данных между LVP и SQL

Для задачи синхронизации конфигурации софтсвича, и записи билинговой информации используется ряд функций.
Первая функция используется для вненсения билинговой информации. Рассмотрим образец для MSSQL

```
CREATE  PROCEDURE sp_lvp_billing
--called by lvp (Voice Proxy for Linux)
--lvp version must be 72 or greater (try: lvp -v)
      @Acct_Session_Id int,
      @Calling_Station_Id char(64),   -- phone, ani
      @Called_Station_Id char(64),    -- phone, dnis
      @Setup_Time datetime,               -- время прихода setup
      @Connect_Time datetime,             -- время connectMSG
      @Disconnect_Time datetime,          -- время рассоединения
      @Acct_Session_Time int,             -- длительность разговора в секундах
      @Disconnect_Cause int,              -- код завершения звонка
      @IP_from char(16),                  -- IP откуда пришло соединение
      @IP_to char(16),                    -- IP куда пошло соединение
      @f_OR_AN int,                       -- флаг, который указывает направления плеча 0 =
ORIGINATING,
                                          -- 1 = ANSWERING
      @user_account char(64),             -- имя пользователя
      @quality int = 0,                   -- заявленное качество сервиса
      @received int = 0,                  -- количество байт
      @transmited int = 0,                -- количество байт
      @is_last_called_number int = 0,     -- номер @Called_Station_Id – надо ли его
                                          -- сохранять для пользователя
                                          -- как последний набранный пользователем
                                          -- (значение 1 = надо , 0 -  ненадо
     @Listen_IP char(16) = '',            -- IP адрес, которую слушает LVP )
      @IP_Dest_by_Gk char(16) = null,     -- IP адрес куда пошло соединение (IP, которое
получено  (IP_TO) gatekeeprom)
      @id_order INT = 0,                  -- ID заказа (для специонального использования)
      @id_order_src INT = 0,              -- ID заказа для покупки (для специонального
использования)
        @Allerting_Time datetime  ,       --  Время прихода алертинга, если небыло то оно
равно времени окончания звонка
        @WhoHandDown INT                  -- Кто положил трубку если 0 тогда сторона A
AS
begin tran

select 'ok' 'sp_lvp_billing_success' -- sp должна возвращать эту строку в случае успешного
внесения данных
commit tran

SET NOCOUNT OFF
--end
GO
```

Следующая по важности функция, дает возможность исполнять комманды execa, exec.

Образец функции следующий:

```
CREATE  PROCEDURE sp_lvp_exec @sp_name varchar(255),
@param1 varchar(255)=null,
@param2 varchar(255)=null,
@param3 varchar(255)=null,
@param4 varchar(255)=null,
@param5 varchar(255)=null,
@param6 varchar(255)=null,
@param7 varchar(255)=null,
@param8 varchar(255)=null,
@param9 varchar(255)=null,
@param10 varchar(255)=null
AS

DECLARE @error INT, @balance MONEY, @balance_int INT, @time_call INT,
@time_call_min INT, @time_call_sec INT,
      @id_customer INT, @id_language INT, @id_record INT, @user_name VARCHAR(16),
@phone_number VARCHAR(32), @ip_address VARCHAR(16),
      @quality INT, @count INT, @max_connect INT, @id_action INT, @ani VARCHAR(32),
      @ip VARCHAR(16), @pn VARCHAR(32), @ip_bk VARCHAR(16), @pn_bk
VARCHAR(32),
      @game_count VARCHAR(1),@kod_game_is_ok VARCHAR(1),
      @text VARCHAR(160), @agent_code VARCHAR(24)

SET NOCOUNT ON
SET @error = 0
SET @balance = 0
SET @balance_int = 0
SET @time_call = 0
SET @id_customer = null
SET @user_name = '0'
SET @id_record = null
SET @phone_number= '0'
SET @ip_address = null
SET @quality = 0
SET @count = 0
SET @max_connect = 0
SET @id_language = 0

IF ( @sp_name = 'authentic' )
BEGIN
      EXECUTE @error = sp_lvp_authenticate @param1, @param2, '', @balance OUTPUT
      IF ( @error <> 0 )      SET @balance = -1 * @error
      SELECT @balance 'lvp_exec_txt_result'
END

SET NOCOUNT OFF
RETURN
```

Данная функция возвращает каждый результат в новой строке.
Используются допольнительные функции

```
sp_lvp_active_customers                @listen_ip varchar(16)
```

должна возвращать список активных пользователей, в которых включен кредит например
102385 380441234567 ani 1
102385 23234 1231234 1
120233 12312 1231231 0
Здесь пользователь с ID 102385 имеет счет 23234 пин 1231234, номер телефона 380441234567 и ему разрешено делать несколько одновременных звонков. Пользователю с ID 120233 неразрешено делать несколько одновременных звонков.

Работа со временем и ДАТА типом

Дата тип — представляется в виде строки и все операции сравнения такие же как операции со строками.
Есть встроенные переменная date (текущее время), SetupTime (начало исходящего плеча), AlertingTime(время алертинга), startO -(начало поднятия трубки для оригинирующего плеча, startA (время ответа отвечающего плеча), durationO — длительность разговора для оригинируещего плеча, durationA — длительность разговора отвечающей стороны.
Пример использования
operation get &date date (Res,Reg,Cnt:0,1,0 - Result: date=2012-07-16 17:12:41, &date=2012-07-16 17:12:41,)
Заметим, что все выше перечисленные переменные имеют точность 1 с.
Часто надо иметь точность выше. Для этого используются аналогичные переменные datems (текущее время), SetupTimems (начало исходящего плеча), ProceedingTimems-время Proceeding, AlertingTimems (время алертинга), startOms -(начало поднятия трубки для оригинирующего плеча, startAms (время ответа отвечающего плеча), durationOms — длительность разговора для оригинируещего плеча, durationAms — длительность разговора отвечающей стороны. Если небыло алертинга тогда startAms = AlertingTimems. Если не было просидинга тогда startAms= ProceedingTimems.
Примеры использования
operation get &d1ms startOms (Res,Reg,Cnt:0,1,0 - Result: startOms=2012-07-16 17:12:51.755, &d1ms=2012-07-16 17:12:51.755,)
operation get &d1ms durationOms (Res,Reg,Cnt:0,1,0 - Result: durationOms=136738, &d1ms=136738,)
operation get &d1ms startAms (Res,Reg,Cnt:0,1,0 - Result: startAms=2012-07-16 17:12:51.754, &d1ms=2012-07-16 17:12:51.754,)
operation get &d1ms durationAms (Res,Reg,Cnt:0,1,0 - Result: durationAms=136738, &d1ms=136738,)
operation get &d1ms startO (Res,Reg,Cnt:0,1,0 - Result: startO=2012-07-16 17:12:51, &d1ms=2012-07-16 17:12:51,)
operation get &d1ms durationO (Res,Reg,Cnt:0,1,0 - Result: durationO=136, &d1ms=136,)
operation get &d1ms startA (Res,Reg,Cnt:0,1,0 - Result: startA=2012-07-16 17:12:51, &d1ms=2012-07-16 17:12:51,)
operation get &d1ms durationA (Res,Reg,Cnt:0,1,0 - Result: durationA=136, &d1ms=136,)
operation get &d1ms SetupTimems (Res,Reg,Cnt:0,1,0 - Result: SetupTimems=2012-07-16 17:12:41.430, &d1ms=2012-07-16 17:12:41.430,)
operation get &d2ms AlertingTimems (Res,Reg,Cnt:0,1,0 - Result: AlertingTimems=2012-07-16 17:12:41.445, &d2ms=2012-07-16 17:12:41.4
45,)
operation get &d1 SetupTime (Res,Reg,Cnt:0,1,0 - Result: SetupTime=2012-07-16 17:12:41, &d1=2012-07-16 17:12:41,)

operation get &d2 AlertingTime (Res,Reg,Cnt:0,1,0 - Result: AlertingTime=2012-07-16 17:12:41, &d2=2012-07-16 17:12:41,)

Для манипулирования временем существуют следующие функции
addday, addhour, addmin, addms, diffday, diffhour, Diffmin, diffms
Пример использования
operation get &d1ms SetupTimems(Res,Reg,Cnt:1,1,0 - Result: SetupTimems=2012-07-16 17:38:55.801, &d1ms=2012-07-16 17:38:55.801,)
operation addday &d2ms &d1ms $dd12 (Res,Reg,Cnt:1,1,0 - Result: &d1ms=2012-07-16 17:38:55.801, $dd12=0, &d2ms=2012-07-16 17:38:55.8
01,)
operation addms &d2ms &d1ms $dd12 (Res,Reg,Cnt:1,1,0 - Result: &d1ms=2012-07-16 17:38:55.801, $dd12=0, &d2ms=2012-07-16 17:38:55.80
1,)

operation diffms $w datems AlertingTimems (Res,Reg,Cnt:1,1,0 - Result: datems=2012-07-16 17:49:42.009, AlertingTimems=2012-07-16 17
:49:38.459, $w=3550,)
operation diffms $w ProceedingTimems AlertingTimems (Res,Reg,Cnt:1,1,0 - Result: ProceedingTimems=2012-07-16 17:49:38.459, Alerting
Timems=2012-07-16 17:49:38.459, $w=0,)
operation diffsec $w date AlertingTime (Res,Reg,Cnt:1,1,0 - Result: date=2012-07-16 17:49:42, AlertingTime=2012-07-16 17:49:38, $w=
4,)
operation diffmin $w date AlertingTime (Res,Reg,Cnt:1,1,0 - Result: date=2012-07-16 17:49:42, AlertingTime=2012-07-16 17:49:38, $w=
0,)

Для удобства введены переменные day, second, minute, month, year, weekday, yearday.
Пример использования
operation get $day day (Res,Reg,Cnt:1,1,0 - Result: $day=16,)
operation get $second second (Res,Reg,Cnt:1,1,0 - Result: $second=41,)
operation get $minute minute (Res,Reg,Cnt:1,1,0 - Result: $minute=47,)
operation get $month month (Res,Reg,Cnt:1,1,0 - Result: $month=7,)
operation get $year year (Res,Reg,Cnt:1,1,0 - Result: $year=2012,)

Управление сигнализацией

Рассматривается несколько этапов прохождения звонка:
+init
Здесь пришло соединение от оригинирующей стороны. Мы здесь планируем, что делать со звонком и куда его маршрутизировать. Событие происходит только раз. Здесь удобно инициализировать нужные переменные, можно отбить звонок. Все происходит в теле звонка.
+setup
Мы породили исходящее плечо. Процесс порождается дочерний. Здесь можно не спешить и обращаться в базу данных для ее звещения. Процессов может много, номер можно определить в переменной child.
+alerting
От отвечающей стороны пришел алертинг. Процесс порождается дочерний, все происходит аналогично как для сетапа.

+connect
От отвечающей стороны пришло поднятие трубки. Процесс порождается дочерний, все происходит аналогично как для сетапа
+voice
происходит только при условии включения детекции голоса. От отвечающей стороны пришел голос. Процесс порождается дочерний, все происходит аналогично как для сетапа
+release
Отвечающая или оригинирующая стороны положили трубку.Процесс порождается дочерний, все происходит аналогично как для сетапа
+disconnect
Звонок полностью завершен. Событие происходит в основном звонке. Здесь мы можем выполнить все действия связанные с биллингом и другими сервисами. Можно не спешить.

Для того чтобы полностью контролировать прохождения звонка следует во френдли хостах прописать параметры аналогично следующим

i-phone prefixB=ruleB a_o authorise no_proceeding;
Здесь i-phone алиас, prefixB=ruleB указывает, что будет использоваться правила преобразования префиксов со списка ruleB, при соединении запускается скрипт authorise, no_proceeding — указывает, что ни алертинга ни просидинга и коннекта мы автомтом не высылаем.
В дальнейшем в скрипте мы можем определить какие параметры мы хотим использовать для оригинирующей стороны, выставить их в биндроутинге и выслать call_proceeding
Пример
[authorise]
system
any_quality
delay=0
account 1111111
+init
operation get &num_or original_called_number
operation get &num_red redirecting_number
operation get &ani ani
operation const &ok ok
operation get $mypid pid
operation get $mysid session_id
operation get $sk is_skype
operation get $sip is_sip
;предыдушие комманды для демонстрации комманд
operation unique_id dnis
operation if gt result 1
operation del_unique_id dnis
operation shutdown
operation endif
; здесь мы проверяли, если на данный номер идет звонок то отбиваем
+alerting
operation parent transmit_udp 2 0 ; отключаем пересылку пакетов с 2 на 1 (с ансверинг на оригинирующую)
operation parent play_till_connect malinkiwav ; включаем музыку для оригинирующей стороны до поднятия трубки
+connect
operation parent transmit_udp 2 1 ; включаем пересылку звука с 2 на 1

```
+detect_voice
operation @ voice ; комманда выдает комментарий в логе


+detect_ring
operation @ ringing

+detect_aon
operation @ AON

+detect_fax
operation @ fax
+
operation set bind_routing "127.0.0.1  127.0.0.1 g711a samples=1"
; устанавливаем параметры к оригинирующему плечу
; можно парамметры узнать в SQL
operation EXECA 0 get_bind_ip ani &originip 0 0 0 0 0 0 0 0
;sql=g711a,   Array[1]=g711a
;sql=sip,   Array[2]=sip
;sql=5060 samples=1,   Array[3]=5060 samples=1
;sql=in,   Array[4]=in
operation get &ip_b Array4
operation get &codec Array1
operation get &prot Array2
operation get &port Array3
operation const &sport sip_port
operation set bind_routing  &originip &ip_b &codec &prot &sport &port
operation send_call_proc_msg
; высылаем колл просидинг сообщение. Здесь выставляется параметры.
operation set detect_voice 2 1000 200 3
; включаем детекцию голоса со стороны 2
; 1000 амплитуда голоса,
; 200 амплитуда шумов;
; 3 дополнительный параметр
operation call dnis 3600 10.1.1.1
;делаем исходящий звонок на 10.1.1.1 с номером, который пришел.

+disconnect
operation get &discon_side disconnect_sideo
; определяем с которой стороны произошел разрыв связи
operation get $noiseORI noise0
operation get $noiseANS noise1
; определяем уровень шума с каждой стороны
operation get $dur durationO
operation cm_asr nash_asr $dur 100 $asr
;определяем  asr за последних 100 звонков
operation cm_avg nash_awg durationO 10 $avgdur
; определяем среднюю длительность разговоров за последние 10 успешных разговоров
; данные сохраняются в колл менеджере и используются всеми звонками
```

Порядок пересылки сигнализации

Лира по умолчанию пересылает автоматически сообщения от отвечающей стороны на оригинирующую call proceeding, alerting(ringing), progress, connect (200 OK). В некоторых случаях удобно, чтобы сама лира решала, что послать. Например мы хотим переслать connect (200 OK) только тогда когда услышим голос отвечающей стороны. Для этого мы используем комманду
operation set no_transmit_messages 1
После чего в событиях +alerting +detect_voice мысамостоятельно высылаем необходимые комманды
Пример

```
[dir_w]
account 111
system
any_quality
+
operation send_call_proc_msg
; высылаем колл просидинг сообщение. Здесь выставляется параметры.
operation set detect_voice 2 1000 200 3
; включаем детекцию голоса со стороны 2
; 1000 амплитуда голоса,
; 200 амплитуда шумов;
; 3 дополнительный параметр
operation set no_transmit_messages 1
operation call dnis 3600 10.1.1.1
;делаем исходящий звонок на 10.1.1.1 с номером, который пришел.

+alerting
operation parent send_progress_Msg
;высылаем прогресс
+connect
operation parent checkpoint 5000
; устанавливаем таймер на получения голоса
+detect_voice
operation const $connect 1
operation parent variable $connect
operation parent send_connect_Msg
; получили голос, отослали коннект и выставили флаг
+checkpoint
operation if ne $connect 1
operation parent stop_call
operation endif
;сработал таймер если флаг не установлен заканчиваем звонок
```

Управление вызовами

В Лире представлен ряд возможностей управлением входящих так и исходящих вызовов. Для временного управления вызовов введено ряд таймаутов, которые изначально устанавливаются в глобальном конфиге.
Например

```
TimeOuts=5,5,40,120,0,0,360 ; (global  TimeOut_N0,... TimeOut_N6)                ;
                                                    определяем таймауты
    ; TimeOut_N0 = receiving SETUP packet message from ORIGINATE side
    ; TimeOut_N1 = opening port 1720 on target host
    ; TimeOut_N2 = receiving ALERTING packet message from ANSWERING side
    ; TimeOut_N3 = receiving CONNECT packet message from ANSWERING side
    ; TimeOut_N4 = for debug purpose
    ; TimeOut_N5 = pause after end of the call
    ;    (with a hope that Originate client will initiate a break of tcp connection)
    ; TimeOut_N6 = timeout for udp data.
    ;   If no udp data available from OR or AN sides for a specified seconds
    ;    Then ReleaseMSG with Cause=Normal, unspecified(31) sent to ORiginate
    ;   Note: TimeOut_N6 start checking for udp data after ConnectMSG only.
; all TimeOuts values are in seconds
```

Данные таймауты можно переустановить в процессе выполнения скрипта, как все, так и по отдельности командой(все начиная с первого, кроме нулевого)
Например
operation set call_TimeOuts *,*,40,0,0,0
В данном случае первый и второй остаются изначальные.
При попытке исходящего вызова в логе выводятся актуальные таймауты. Например
TimeOuts[0..9]=0 15 40 30 0 1 0 0 0 7200
Здесь последний таймаут обозначает максимальная длительность разгвора. Этот таймаут устанавливается в командах call, placecall. call_to_orignate, placecall_to_originate.
Пример

operation call dnis 3600 10.1.1.1
;делаем исходящий звонок на 10.1.1.1 с номером, который пришел.
Максимальная длительность разговора 3600 секунд.
Команды call, placecall различаются тем, что первая идет только на конкретный IP. Вторая использует таблицу маршрутизации для номера, качества сервиса, времени и номера А.
Команды с суффиксом _to_orignate аналогичны предыдущем, но отличаются тем что после поднятия трубки абонентом, эта сторона автоматически становится оригинирующей и если там потом положат трубку, звонок разрушится. Здесь перед выполнением команды надо установить флаг. Пример

operation set stay_offline 1
operation call_to_originate 1111 7200 10.0.0.1
Здесь 1111 номер вызываемого абонента, 7200 — максимальная длительность разговора, 10.0.0.1 — IP адрес вызываемого устройства. При поднятия трубки абонентом флаг stay_offline сбрасывается автоматически.

Базовая функция маршрутизации основана на таблицы маршрутизации. Ее преимущества:

- много маршрутов для направления;
  Пример

```
+7495
10.10.10.1
10.10.10.2
```

- Маршрутизация в зависимости от качества сервиса;

```
 +7495 quality=2
```

10.10.10.1
10.10.10.2
+7495 quality=3
10.10.10.4
10.10.10.5
Первое направление работает для качества сервиса 2, Второе для 3. Качество сервиса устанавливается во френдли хосте или например командой operation set quality 2

- Маршрутизация в зависимости от дня и времени суток;

+7495 Mon 09:00-18:00
10.10.10.4
10.10.10.5
+7495
10.10.10.4
10.10.10.5
Первое напраление выбирается только в понедельние с 9:00 до 18:00

- маршрутизация в зависимости от номера А

+7495 +7495
10.10.10.4
10.10.10.5
+7495 +7
10.10.10.4
10.10.10.5
Здесь первое направление выбирается если номер А начинается на 7495. Второе, если номер начинается на 7.

- запись в биллинг информации о выборе направления и маршрута, контроля длинны номера и тарифа. Например

+7495 quality=11 min_phone_len=11 id_order=1333606
127.0.0.1/+08012 id_order=734897 rate=0.0167
Здесь для направления 7495 и качества сервиса 11, длина номера должна быть не меньше 11. Заявка направления равна 1333606. Маршрут прописан на IP 127.0.0.1 с добавочным префиксом 08012 и заявка маршрута равна 734897. Маршрут заявки сработает если тариф абонента больше равно 0.0167.

- одновременные маршруты

+7495
10.10.10.4 OR
10.10.10.5
Здесь одновременно будут сделаны два параллельных вызовов, Если по какому то маршруту подымут трубку, второй вызов закончится.
10.10.10.4 OR5
10.10.10.5
Здесь будет выбран первый маршрут и с задержкой 5 секунд будет произведен параллельный вызов следующего маршрута

- балансировка нагрузки

10.10.10.4 AND
10.10.10.5
Здесь будет нагрузка балансировать 50% на 50%. Можно легко определить какой маршрут более качествен.

Перебор маршрутов
Используя таблицу маршрутов возникают проблемы с решением при каких условиях надо прекращать перебор. По умолчанию Лира перебирает все маршруты, пока не закончатся или кто то подымет трубку. Можно дополнительно настроить правила:

- количество отбоев з кодом занято. В Глобальном конфиге в секции Global комманда busy_count = 1 ;(только один отбой)
- Если вы хотите, что бы после прихода прогресса или ringing больше не перебирать маршруты используем комманду скрипта
  ```
  operation set progressStop 1
  ```
  для отключения опции
  ```
  operation set progressStop 0
  ```

Установка параметров входящего плеча

Установка входящего плеча имеется ввиду выбор rtp портов и адреса, с которых мы будем их слать. Также здесь следует установить каким кодеком мы будем пользоваться и размером пактов. Все эти параметры мы устанавливаем с секции [bind_routing] или коммандой operation set bind_routing.

Синтаксис комманды

operation set bind_routing 10.1.1.1 in g711a port 1720 samples 2

operation set bind_routing &originip &ip_b &codec &prot &sport &port

Здесь можно использовать переменные строковые & или просто текст.

Пример

operation get &o_udp_listen o_udp_listen; определяем адрес RTP оригинатора

operation set bind_routing &o_udp_listen &ip_b &codec &prot &sport &port

Здесь мы устанваливаем параметры биндроутинга.

Установка параметров исходящего плеча

Параметры для исходящего плеча аналогичны, как и для входящего, но есть Важные особенности:

- параметры Номера А
- параметры номера Б
- фильтры номера А
- фильтры номера Б

Параметры номера А — это каким образом будет представляться номер А. Например, абонент А имеет городской номер 1234567. При вызове междугороднего номера вы хотите чтобы номер представлялся в виде 04951234567. Здесь также нужно бы выставлять тип номер, номерной план, оригинальный вызываемый номер, переадресовываемый номер, количество переадресаций. Но в данном документе для простоты не рассматриваем. Необходимая информация устанавливается в параметре

bani_file=filename

Здесь filename название файла в котором мы пропишем что делать с номерами А.

Пример файла

495123450811040/-9/8
495123450811050/-9/8
49512345.../8
495123454../8
495123455../8
495123458../8
495123450../8
495123459../8
951234...
9512344..
9512345..
9512348..
9512340..

9512349..
951234432
Здесь номер A должен быть 8-ми значный. В первых двух строках, если номер совпадает с маской, мы берем 9 символов слева и из них берем 8 справа. Следующие 6-ть строк указывать что при совпадении маски мы в качестве номера A берем 8 правых символов. Последняя строка указывает номер, который будет использоваться в качестве пилотного, если ни одна из масок не сработает.
Для ограничение номеров A, которые могут воспользоваться сервером(маршрутом) устанавливаем фильтр номеров A. Пример
filter=fileName2
Здесь fileName2 название файла в котором прописываются маски номеров A, которым разрешено.

Дополнительно используются опции фильтра ani_allow, ani_deny, dnis_deny, которые указывают на списки а не названия файлов. Пример
dnis_deny=blist
где blist название списка, в котором находятся шаблоны номеров.

codec_pt в секции [global]
codec_pt — позволяет подстраиваться под кодеки и размеры пакетов с каждого плеча.

Списки

Списки удобнее чем файлы тем, что их можно загружать из SQL. Также их можно формировать из файла. Формат из SQL
[list_blist]
0000000000
000000T
Здесь blist название списка.
Формат из Файла подразумевает, что должна быть секция в глобальном конфиге [list], в котором мы указываем соотношение между названием списка и файлом. Например
[list]
blist black_file

Для работы со списками есть дополнительные комманды
operation list black_list_ip OriginIP $in_list &date
Здесь result указывает успешно ли выполнилась команда. В переменные $in_list,&date будут занесены значение полей. Пример списка
[list_black_list_ip]
1.1.1.1 1 “2012-07-07 19:01:01”

Списки могут содержать большое количество записей. Тестировалось несколько сотен тысяч. Списки индексируются поэтому скорость поиска очень высокая, время поиска несколько милисекунд.

Friendly_Hosts, управление входящими вызовами (продолжение)

Friendly_Hosts позволяют управлять кому можно делать входящие вызовы, а кому запрещено. Здесь можно использовать алиасы или IP адреса. Алиасы могут содержать множество IP адресов. Есть специальные алиасы такие, как Unknown_Host — любой

сервер, FH_Sql_Server — сервера, которые будут загружены с SQL, а также command_prompt, который должен быть прописан в секции
[Aliases]
command_prompt 127.0.0.1
Последний разрешает запускать скрипт с командной строки.
Как правило в строке Unknown_Host мы указываем ряд дополнительных параметров, которые должны обеспечивать безопасность. Например
Unknown_Host prefixB=dnis_list PrefixA=ani_list dnis_deny=blist ani_deny=blacklist ani_allow=alist a_o authorise nat no_proceeding sip_registration quality=2 force maxconn=10

Здесь prefixB указывает, что правила преобразования префиксов номеров Б используют список dnis_list, prefixA указывает, что правила преобразования префиксов номеров Б используют список ani_list, запрещены номера А со списка blacklist. Разрешены номера А со списка alist. Все входящие вызова запускают скрипт a_o authorise. nat - RTP пакеты будут отсылаться на порты с которых будут приходить пакеты не зависимо от портов которые присланы в Сетапе. no_proceeding - значит, что proceeding мы будем высылать командой. sip_registration — указывает, что все сип звонки должны быть с регистрации. Force - указывает, что сип регистрация происходит при каждом звонке. Quality=2 — значит, что все звонки с этого сервера будут иметь 2-е качество сервиса. Ограничение на количество звонков указывает опция maxconn=10.
Часто бывает, что надо для некоторых телефонов предоставлять другое качество сервис, чем для всех телефонов. Тогда используем опцию ani_file=best_p, где best_p название файла с масками номеров А. Для этого создаем копию хоста в котором, можно поменять a_o authorise, quality=2, maxconn=10 и добавляем опцию ani_file. Например
Unknown_Host prefixB=dnis_list PrefixA=ani_list dnis_deny=blist ani_deny=blacklist ani_allow=alist ani_file=best_p a_o authorise nat no_proceeding sip_registration quality=3 force maxconn=15
Unknown_Host prefixB=dnis_list PrefixA=ani_list dnis_deny=blist ani_deny=blacklist ani_allow=alist a_o authorise nat no_proceeding sip_registration quality=2 force maxconn=10

Сип регистрация

Для проведения сип регистрации используются скрипт [sip_register]. Сам текст скрипта пишется под нужды заказчика. Для успешной регистрации должна выполниться команда
operation sip_register pin time
где pin имеет значение call-id, time длительность регистрации в секундах. При неуспешной регистрации выполняем
operation sip_register pin 0.
Для злоумышленников которые подбирают пароль, удобно выполнять команду
operation sip_register_fake pin 600
Данная команда внешне ведет себя как успешная регистрация но звонить не дает.

Пример скрипта
[sip_register]
+
operation list black_list_ip Array4 $in_list
operation if EQ $in_list 1
operation sip_register_fake pin 600
operation const $abonent_not_pbx 1

```
operation return
operation endif
operation use MySql_test
operation get num0 array0  ;
operation get num1 array1  ;account
operation get num2 array2  ;ani
operation get pin array3   ;call-id
operation get ip array4
operation get &ext_ip array4                    ;
operation get &sip_phone array5
operation get $port array6
operation local_sip_request array1 array2
; здесь мы проверили на локальную аутентификацию
operation if eq result true
operation sip_register pin 600
operation return
operation endif                 ;
operation const Array10 1
operation execa 10 sip_auth num1 num2 0 0 0 0 0 0 0 0
; здесь мы сделали запрос  в  SQL
operation if eq Array10 0
operation md5auth array11
; проверили   md5 суму
operation get $result result
operation if eq $result true
operation sip_register pin 600
operation return
operation endif                 ;
operation endif                 ;
operation sip_register pin 0
 ; по умолчанию отбиваем регистрацию
```

Аналогично создаем

```
[sip_subscribe]
+
operation get num0 array0  ;
operation get num1 array1  ;account
operation get num2 array2  ;ani
operation get pin array3   ;call-id
operation get &call_id array3   ;call-id
operation get ip array4
operation get $port array6
operation get &phone array7
operation sip_subscribe &call_id 3600 &phone
;Здесь надо бы логику усложнить, но как демо может быть
operation return
```

В глобальном конфиге можно создать локальную таблицу пользователей
Пример

```
[Friendly_Users]
1001 5001 5001x 1 2
1001 5002 aon 1 2
```

1001 5003 ani 1 2
Здесь 1001 уникальный идентификатор пользователя. Все записи для оного пользователя должны иметь одинаковый идентификатор. 5001 — логин, 5001x пароль, 5002 основной номер A, который мы присваиваем пользователю, 5003 дополнительный номер A который может иметь абонент при регистрации. Команда operation local_sip_request array1 array2 ищет пользователя, если находит делает контрольную суму, если сошлась возвращает 1.
Для того, чтобы использовать сип регистрацию для маршрутизации трафика надо в глобальном конфиге в секции [global] добавить команду sip_routes.

sip_attackCount

sip_attackCount позволяет защищаться от массированной внешней атаки. Команда используется в глобальном конфиге в секции [global.
Пример
sip_attackCount 7

File_manager

file_manager — модуль, который помогает асинхронно проводить запись логов и звуковых файлов на диск и или удаленный файл сервер. file_manager включается в глобальном конфиге в секции [global]. Синтаксис команды
file_manager [ddl] [uWriteEveryNSec] [ttInActiveCloseSec] [N skip selects] [buffer, default 32000]
Пример
file_manager 5 3 30 100

file_manager позволяет проводить запись всех разговоров и делать логи с максмальным уровнем отладочной информации. Проводились тесты на шести ядерном процессоре 1500 одновременных звонков. Загрузка процессоров не превышала 12%, качество звука не страдало.

call_manager

call_manager модуль, который позволяет сохранять в памяти данные, полученные от звонков. По умолчанию при выключении Лиры, все данные теряются. Сруктрура данных состоит из ключа и его значении. Ключи индексируются и поиск данных не превышает нескольких миллисекунд. Команды доступа к данным следющии
operation cm_read &key &val &day_date
Здесь &key — ключ, &val — получаемое значение, &day_date — дата последней модификации.
operation cm_write &key &val
записываем значение ключа в call_manager.
operation cm_sum &key $val $delta
Добавляем к значению ключа &key величину $delta. Комманда возвращает новое значение ключа в переменную $val.

Следующие команды создают ключи с временем жизни пока идет звонок

operation unique_id &key
В одном процессе unique_id может быть много. Result содержит количество процессов в которых есть одинаковый ключ.
operation unique_dnis &key
В одном процессе unique_dnis может быть только один. При новой команде, предыдущий unique_dnis вытирается.
Часто надо только проверить на количество ключей без создания таковых. Для этого служат комманды
operation unique_id_lookup &key
operation unique_id_lookup &key
Result содержит количество процессов в которых есть одинаковый ключ +1.
Для того чтобы вытереть unique_id ключ, используем команду
operation del_unique_id &key

keep_call_manager_data

Часто надо сохранять данные даже после перезагрузки Лиры. Для этого используем в глобальном конфиге в секции [global] команду keep_call_manager_data.
Для того чтобы сделать дамп памяти, который при старте Лиры загрузится в память выполняем команду
operation cm_save

Уровень Отладочной информации в логах

В глобальном конфиге в секции [global] помещаются команду, которые определяют какой уровень отладочной информации будет в логах.
Существуют следующие команды (не полный список)
ddl=5 ; глобальный уровень для логов
sip_ddl=5 ; уровень для Сип Менеджера
sql_ddl=5 ; уровень для SQL менеджера
mm_ddl=7 ; уровень для Медиа Менеджера (для RMTP)
cm_ddl=5 ; уровень для Call Менеджера
dm_ddl=5 ; уровень Device Менеджера (для модемов)

Есть скриптовые команды
operation set ddl 7
operation set sip_ddl 7

Уровень отладочной информации может быть от 0 до 7. 0 — отладочная информация не пишется.

События

В лире отрабатываются следующие события:
+init

порождается в основном процессе, немного ограничено в функционале. Здесь нельзя проигрывать файлы, посылать сообщения connect_msg, progress_msg, alerting_msg. Основная цель этого события выставить флаги, если надо отбить звонок.
+setup
порождается в дочернем процессе и стартует параллельно при начале исходящего звонка. Все переменные наследуются с основного процесса. Таких событий в одном звонке может быть много, столько сколько было исходящих звонков. Переменная child равна номеру по порядку.

+alerting
порождается в дочернем процессе и стартует параллельно при получении alerting или progress сообщений. Все переменные наследуются с основного процесса. Таких событий в одном звонке может быть много, столько сколько было исходящих звонков. Переменная child равна номеру по порядку.

+connect
порождается в дочернем процессе и стартует параллельно при получении connect(200 OK) сообщения. Все переменные наследуются с основного процесса. Таких событий в одном звонке может быть много, столько сколько было исходящих звонков. Переменная child равна номеру по порядку.

+checkpoint
порождается в дочернем процессе и стартует параллельно при срабатывании таймера. Все переменные наследуются с основного процесса при помощи команды checkpoint. Примеры команды
operation parent checkpoint 5000
operation checkpoint 30000

+dtmfo
порождается в дочернем процессе и стартует параллельно при детекции dtmf сигнала с оригинирующей стороны, переменная dtmf_key содержит dtmf код. Если событие прописано то dtmf сигнал не пересылается, если его надо переслать, используем команду отсылки. Пример
+dtmfo
operation get &key dtmf_key
operation if eq originIP 10.10.1.1
operation const &k d
operation if eq &key *
operation const &k dz
operation else
operation add &k &k &key
operation endif
operation const &ext .wav
operation add &k &k &ext
operation parent play_file &k 1
; на отвечающую сторону проигрываем звук
operation return
operation endif
operation parent send_key &key 1
; на отвечающую сторону высылаем dtmf.

+dtmfa

порождается в дочернем процессе и стартует параллельно при детекции dtmf сигнала с отвечающей стороны, переменная dtmf_key содержит dtmf код. Если событие прописано то dtmf сигнал не пересылается, если его надо переслать, используем команду отсылки

+message
порождается в дочернем процессе и стартует параллельно при получении от Call_Manager сообщения. Переменная message содержит само сообщение. Сообщение отсылается командой
operation message &unique_dnis &message
Здесь &unique_dnis — ключ, который использовался в команде unique_dnis. Если мы хотим послать сообщение процессам, по ключу unique_id. Тогда мы должны к ключу спереди добавить id. Пример

```
operation unique_id_lookup @&d3
operation if gt result 1
operation add @&idd3 id  @&d3
operation message @&idd3 "stop_wait"
operation endif
```

```
+message
operation get &message message
operation if eq message "stop_wait"
operation parent cancel_wait
; предыдущая команда говорит основному процессу закончить команду wait и продолжить выполнение скрипта.
operation endif
operation return
```

+detect_voice
порождается в дочернем процессе и стартует параллельно при получении основным процессом информации, что пошел голос.

+detect_ring
порождается в дочернем процессе и стартует параллельно при получении основным процессом информации, что детектируется гудки вызова.

+detect_aon
порождается в дочернем процессе и стартует параллельно при получении основным процессом информации, что детектируется сигнал запроса аон.

+detect_fax
порождается в дочернем процессе и стартует параллельно при получении основным процессом информации, что детектируется факс.

Все четыре события происходят при выполнении команды

operation set detect_voice 1|2 1000 200 3
Здесь включаем детекцию голоса со стороны 1 или 2 , 1000 амплитуда голоса при которой идет срабатывание детекции, 200 амплитуда шумов, 3 дополнительный параметр для подстройки алгоритма

+release
порождается в дочернем процессе и стартует параллельно при завершении исходящего плеча. Все переменные наследуются с основного процесса. Таких событий в одном звонке может быть много, столько сколько было исходящих звонков. Переменная child равна номеру по порядку.

+disconnect
порождается в основном процессе, немного ограничено в функционале. Здесь нельзя проигрывать файлы, посылать сообщения connect_msg, progress_msg, alerting_msg. Событие происходит когда звонок разрушен и надо обработать статистику.

В дочерних процессах есть возможность взаимодействовать с основным процессом, отдавать ему команды, посылать сообщения. Примеры команд

operation parent play_file &name $side ; $side= {1,2}
operation parent variable $var ; устанавливает в основном процессе значени переменной
operation parent send_progress_Msg
operation parent send_connect_Msg
operation parent send_call_proc_Msg
operation parent checkpoint $time
operation parent cancel_wait
operation parent stop_call ; останавливаем исходящее соединение
operation parent finish_process ; заканчиваем весь звонок
operation parent transmit_udp 2 1 ;включает полностью пересылку звука
operation parent transmit_udp 0 1 ;включает пересылку со стороны 0
operation parent transmit_udp 1 1 ;включает пересылку со стороны 0

Автоматический запуск скрипта

Часто есть необходимость при старте Лиры запустить процесс, который должен отслеживать некоторые события.
Для этого в глобальном конфиге в секции [global] выполняем команду

autorun script_autorun extention
Здесь script_autorun названия скрипта, который мы хотим запустить, extention расширение этого скрипта. Если мы хотим, что эти скрипты выполнялись постоянно следует использовать пример
[http]
account 380442053031
system
+
operation cm_sum http_a $my_id_http 1; устанавливаем id для обеспечения уникальности
operation set MAX_SESSION_DURATION 0
operation set stay_offline 1 ; отвязываемся от оригинирующей стороны
operation do
operation cm_read http_a $my_id_check; проверяем не запущен ли новый процесс

```
operation if ne $my_id_http $my_id_check ; если да то мы закнчиваем
operation shutdown
operation endif
…..
operation wait 10; время ожидания перед следующим циклом
operation while eq 1 1 ; бескончный цикл
```

Размножение процессов

Часто есть необходимость размножить процессы. Причин может быть множество. Одни из них:

- при входящем звонке вы хотите известить SQL сервер, но также желаете не тратить время при прохождении звонка
- делаете предиктивный обзвон и желаете наростить количество исходящих соединений

Для этого используется команда
operation dublicate_task skip
Комманда делает копию процесса, В новом процессе закрывает все сокеты и продолжает скрипт пропустив одну строку. Пример

```
operation const $dubl 0
operation dublicate_task skip                    ;
operation incr $dubl                           ;
operation if eq $dubl 0                          ;
operation proc script_autorun extention                ;
operation endif                                ;
```

Работа с сокетом

Лира может выступать клиентом ВЕБ сервисов. Для этого существуют ряд команд

operation http_get &url @&dest_IP &parameters @$dest_port
Пример

```
;string1 = "?incom_call=380...1xx#session_id#AON"
operation http_get 10.1.2.3 10.1.2.3 string1 443
```

operation http_send null @&dest_IP @$dest_port
Пример

```
operation buffer 1 putf &mycheck_st0; записываем в буфер строку
operation http_send null @&dest_IP @$dest_port ; соединяемся
operation if eq result 0 ; если еще не соединились подождем
operation wait 2
operation endif
operation get $rr disconnect_sideo
operation wait 2 until_get    ; ждем ответа
operation get $asn_buf_len buffer1    ; проверяем не пустой ли буфер
operation get $sidet side  ; проверяем с какой стороны пришли данные
...
```

operation set zero_buffer 0 : зануливаем буфера
operation set zero_buffer 1
operation get $rr disconnect_sideo
operation if ne $rr 1
operation shutdown 1
operation endif

Команды управления буфером

operation buffer 1 puth hex 0 a1 b2 c3 ;
operation buffer 0 read &variable position length
operation buffer 1 readi &variable position
operation buffer 0 put &variable position length
operation buffer 0 puti &variable position
operation buffer 0 add &variable position length
operation buffer 0 addi &variable position
operation buffer 0 sum &variable position_start position_stop
operation get num1 buffer0
operation get num2 buffer1

Примеры
operation buffer 0 readf &fileName 0 $b1 ; read from buffer to file
operation buffer 1 putf &fileName ; read from file to buffer
operation const &hex_str "FD 1A 2A 3A FF F0"
operation buffer 1 putS &hex_str 0 ; put string to buffer
operation buffer 1 addS &hex_str ; add string to buffer
operation buffer 0 readh &bbb 0 $b1 ; read from buffer to string

Процедуры и функции

В лире предусмотрено построение процедур, которые вызываются командой

operation proc Name ext param1 parm2 …

Где Name название процедуры, ext — расширение; param1 parm2 параметы. Парметров может быть много.

Функция, это процедура с названием function. И выполняется командой

operation function ext param1 parm2 …

Каждая процедура и функция должна заканчивать командой return. Команда return может возвращать результаты. Пример

operation function cm_map tw14141234561 oq38094 800000 800003

operation get &aa @f1 ; получаем результат выполнения функции

operation get &bb @f2 ; получаем результат выполнения функции

[function]

system

any_quality

+cm_map

operation set ddl_return 0   ; скрываем лог до ретурна

operation get @&key param1

operation get @&pref param2

operation get @$n1 param3

operation get @$n2 param4

...

operation return @&res @&res1 @&res2

Хороший стиль использовать внутри функций и процедур локальные переменные с префиксом @.

Команда switch

В лире можно использовать довольно мощный инструмент команду switch. Конструкция выгдядит следующей

operation switch &key1 &key2 …

operation case &val1 &val2...

…

operation default

...

operation endswitch

Здесь &key1 &key2 … список ключей, которые мы проверяем на условия. &val1 &val2... -

значения, которым должны соответствовать ключи. Значений может быть меньше, тогда проверяются только те которые есть. Если ни одно из условий не сработает тогда срабатывает default, default может и небыть.

Пример

operation switch &codec &codec2

operation case g711a g711a

operation set mmcodec 2 2

operation case g711a

operation set mmcodec 2 2

operation case g729r8

operation set mmcodec 3 3

operation case wave

operation set mmcodec 1 2

operation default

operation set mmcodec 4 2

operation endswitch

Технические характеристики

Операционная система	Linux, ядро 2.4.19, FreeBSD
Требования к компьютеру	PC, Processor Pentium III-300MHz, Ram 256M, HDD 2Hb
Поддержка протоколов	H323, SIP
Максимальное Количество одновременных соединений для PC, Processor Pentium III-800MHz, Ram 512M, HDD 2Hb	Не меньше 500 (при отключении вывода отладочной информации)
Статическая маршрутизация	Неограниченно
"Экаунтинг" в текстовых файлах	Неограниченно
Возможность написания IVR-скриптов	Неограниченно
"Экаунтинг" в SQL-сервер	Неограниченно
Возможность осуществления UDP-экаунтинга	Неограниченно
Функция авторизации и динамической маршрутизации через SQL-сервер	Неограниченно