В репозиториях, которые включены в CentOS по умолчанию (Base, AppStream, Extras) доступны далеко не все пакеты, кроме того, большинство представлено устаревшими версиями (например, MariaDB, PHP). Некоторые из требуемых в данной работе пакетов доступны в репозитории EPEL (или Extra Packages for Enterprise Linux). Если один и тот же пакет находится в нескольких репозиториях, yum (если не указана желаемая версия и не сконфигурирован репозиторий по умолчанию для пакета) установит самую последнюю версию. Чтобы добавить репозиторий EPEL в список активных репозиториев, достаточно установить пакет `epel-release` из базового репозитория CentOS. Добавьте репозиторий EPEL:

```
sudo yum install epel-release
```

По умолчанию `yum` автоматически НЕ устанавливает пакеты, только проверяет их наличие в активных репозиториях, составляет список всех зависимых пакетов для установки, подсчитывает объем загружаемых файлов, требуемое место на диске для установки и спрашивает, следует ли установить их (см. рисунок ниже).* ***На вопросы об установке пакетов и обновлении GPG-ключей репозиториев (Is this ok?) следует ответить y.*

```
================================================================================
 Package              Arch           Version            Repository       Size
================================================================================
Installing:
 epel-release         noarch         8-5.el8            extras           22 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 22 k
Installed size: 30 k
Is this ok [y/N]: y
```

Если `yum` успешно установил запрашиваемые пакеты, вывод будет подобен приведенному на рисунке ниже (все установленные пакеты и зависимости (если существуют) приведены после строк `Installed:` и `Dependency Installed:` соответственно.

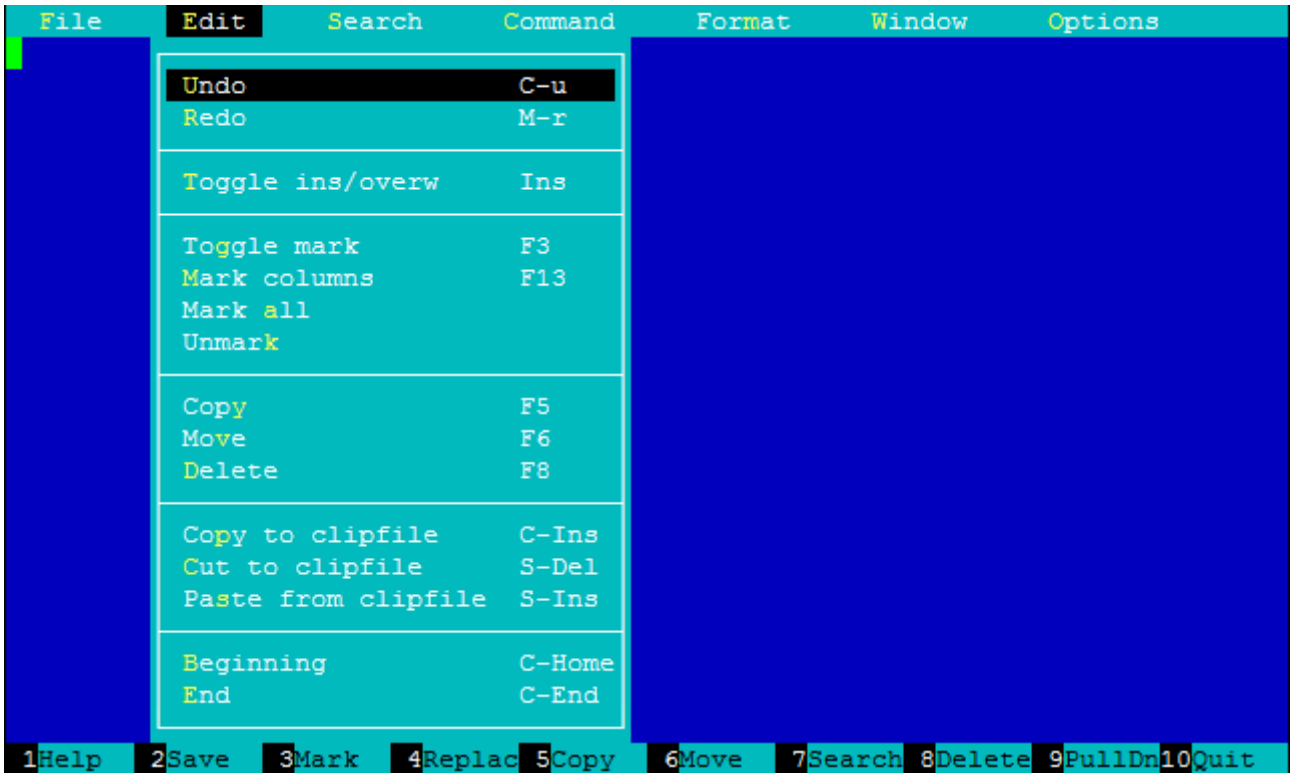
```
Running transaction
  Preparing        :                                                        1/1
  Installing       : epel-release-8-5.el8.noarch                            1/1
  Running scriptlet: epel-release-8-5.el8.noarch                            1/1
  Verifying        : epel-release-8-5.el8.noarch                            1/1

Installed:
  epel-release-8-5.el8.noarch

Complete!
```

Если `vi` Вам неудобен для редактирования, можно установить другой текстовый редактор, например редактор файлового менеджера `MidnightCommander`, который помимо возможности удобно просматривать файлы и директории, предоставляет свой собственный текстовый редактор.

Устанавливается через yum:

```
sudo yum install mc
```

Редактор mcedit использует ряд F-клавиш для управления, их назначение указано внизу окна на кнопках с соответствующим номером, кроме того, mc поддерживает управление мышью, можно кликнуть по верхней панели для отображения меню, а также по нижним кнопкам. Для копирования/вставки используются сочетания Ctrl-Insert/Shift-Insert соответственно (а также выделение мышью с зажатым Shift/ПКМ).



Чтобы открыть файловый менеджер в текущей директории выполните команду `mc`. Для редактирования файла замените vi на mcedit в вызове редактора:

```
sudo mcedit /etc/yum.conf
```

1. SELinux

Security-Enhanced Linux - дополнительная система безопасности Linux, реализующая мандатную систему управления доступом, позволяющая гибко разграничивать права пользователей и отдельных процессов в рамках контекстов безопасности. По умолчанию включена в CentOS в активном режиме, что без соответствующей настройки препятствует работе устанавливаемых пакетов и служб в фоновом режиме, поэтому ее требуется выключить.
Выключить SELinux немедленно до перезагрузки системы можно так:

```
sudo setenforce 0
```

Чтобы перевести SELinux в выключенный режим окончательно, требуется заменить `SELINUX=enforcing` на `SELINUX=disabled` в файле `/etc/selinux/config`:

```
sudo vi /etc/selinux/config
```

SELinux в некоторых случаях может оказаться крайне полезен, в дальнейшем будет полезно понять принцип его конфигурирования и работы.

2. Установка пакетов для скачивания и сборки исходного кода

Установите пакет `wget` для скачивания файлов по URL ссылке, клиенты систем управления версиями `git` и `svn`, компиляторы `gcc gcc-c++` и дополнительные пакеты для сборки проектов из исходного кода по Makefile.

```
sudo yum install wget git gcc gcc-c++ svn wget cmake make automake autoconf pkgconfig graphviz
```

Для управления репозиториями необходимо установить пакет утилит yum.

```
sudo yum install yum-utils -y
```

обратите внимание, что добавление ключа `-y` заставит `yum` установить пакет без подтверждения, это может быть полезно в плане экономии времени, однако иногда все же лучше сверять пакеты и версии перед установкой.

Включите репозиторий PowerTools (добавлен в CentOS 8 по умолчанию, но выключен) и установите криптографические пакеты и библиотеки для поддержки шифрования передаваемых голосовых данных

```
sudo yum-config-manager --enable PowerTools
sudo yum install openssl openssl-devel libsrtp libsrtp-devel doxygen -y
```

Установка требуемых для Asterisk зависимостей из репозиториев (набор пакетов приведён для конфигурации, необходимой в данной работе; некоторые ненужные в данном курсе модули Asterisk будут недоступны при сборке):

```
sudo yum install libedit-devel jansson-devel uuid libuuid-devel sqlite-devel libxml2-devel libcurl-devel xmlstarlet bison flex neon-devel lua-devel uriparser-devel libxslt-devel unixODBC
```

```
unixODBC-devel bluez-libs-devel radcli-devel freetds-devel newt-devel popt-devel libical-devel spandsp-devel codec2-devel fftw-devel libsndfile-devel unbound-devel binutils-devel gsm-devel zlib-devel libtool bzip2 patch sox redhat-rpm-config kernel-devel
```

3. Прокси для `wget` - пакета для загрузки файлов

Прописывается в начале файла `/etc/wgetrc`

```
sudo vi /etc/wgetrc
```

Вставьте следующие параметры:

```
http_proxy=http://прокси:порт
https_proxy=http://прокси:порт
proxy_user=логин
proxy_passwd=пароль
```

**также можно использовать переменные окружения для указания прокси-сервера, yum/dnf/apt и wget умеют работать с ними. Подробнее об определении прокси в `/etc/environment` или использовании команды `export` можете узнать самостоятельно.*

4. Настройка правил фаервола

В CentOS 8 в качестве фаервола используется `firewalld` - пакет, представляющий собой надстройку над встроенным в ядро Linux фаерволом `netfilter`. Является аналогом широко распространенного в Linux пакета `iptables`, но вместо настройки таблиц и цепочек реализует концепцию zone-based firewall (сетевые интерфейсы привязываются к зонам, правила фильтрации применяются к зоне, также при необходимости настраиваются правила между зонами). По умолчанию все доступные ОС сетевые интерфейсы принадлежат зоне public, все новые правила, если не указана зона, также автоматически применяются к этой зоне.

Согласно настроенным по умолчанию правилам, доступ по сети к CentOS разрешен по 22 порту для удаленного подключения по уже упомянутому ранее протоколу SSH. Для удобства управления правилами в firewalld присутствует возможность создать сервис и ассоциировать с ним группу правил.

Создание нового сервиса:

```
sudo firewall-cmd --permanent --new-service=asterisk
```

Добавление к сервису нужных портов:

```
sudo firewall-cmd --permanent --service=asterisk --add-port=5060/tcp
sudo firewall-cmd --permanent --service=asterisk --add-port=5060/udp
```

```
sudo firewall-cmd --permanent --service=asterisk --add-port=5061/tcp
sudo firewall-cmd --permanent --service=asterisk --add-port=5061/udp
sudo firewall-cmd --permanent --service=asterisk --add-port=4569/udp
sudo firewall-cmd --permanent --service=asterisk --add-port=10000-20000/udp
```

** где 5060 — SIP, 5061 — SIP over TLS, 4569 — IAX, 10000-20000 — диапазон для динамических портов RTP.*

Теперь добавьте созданный сервис в список активных правил:
```
sudo firewall-cmd --permanent --add-service=asterisk
```

Перечитать установленные правила firewalld (применить изменения без сброса текущих сетевых соединений):
```
sudo firewall-cmd --reload
```

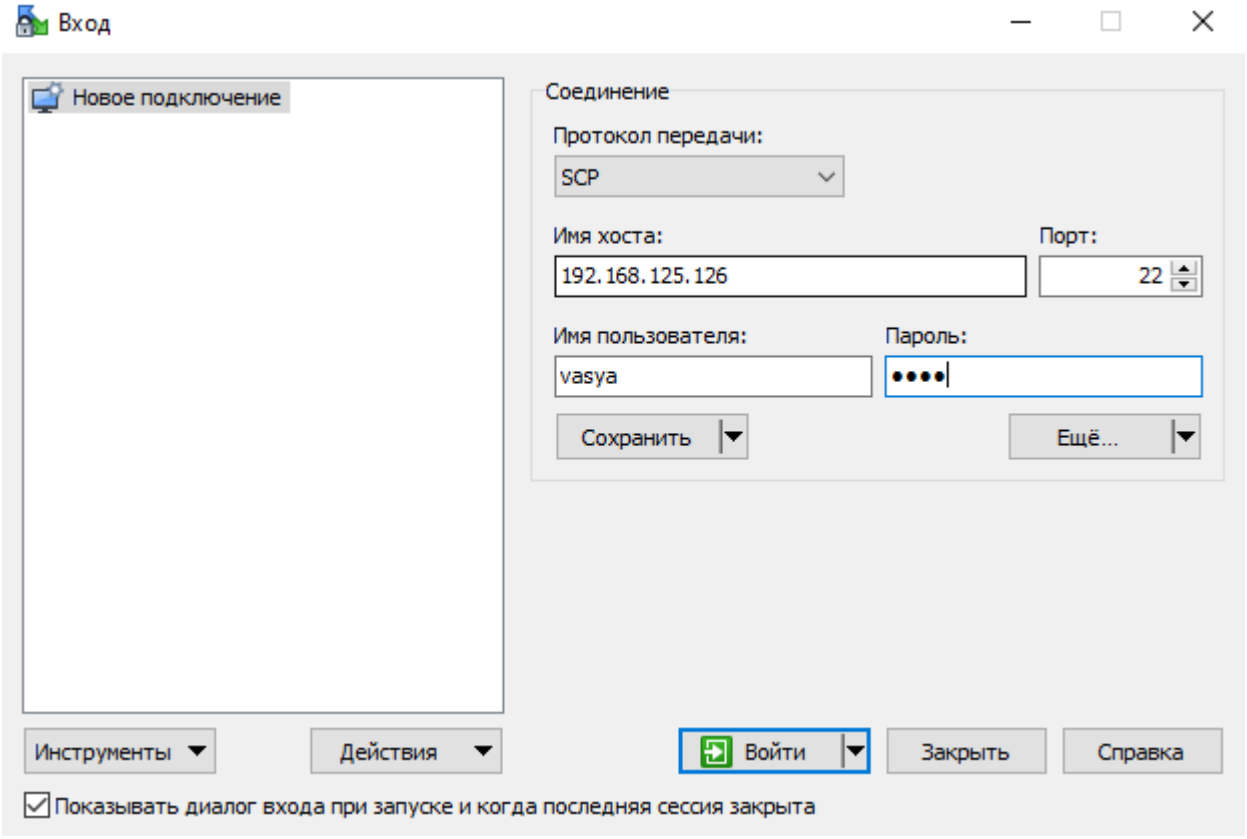
Так можно посмотреть правила, привязанные к сервису
```
sudo firewall-cmd --info-service=asterisk
```
Посмотреть всю конфигурацию firewalld
```
sudo firewall-cmd --list-all
```

Установка Asterisk

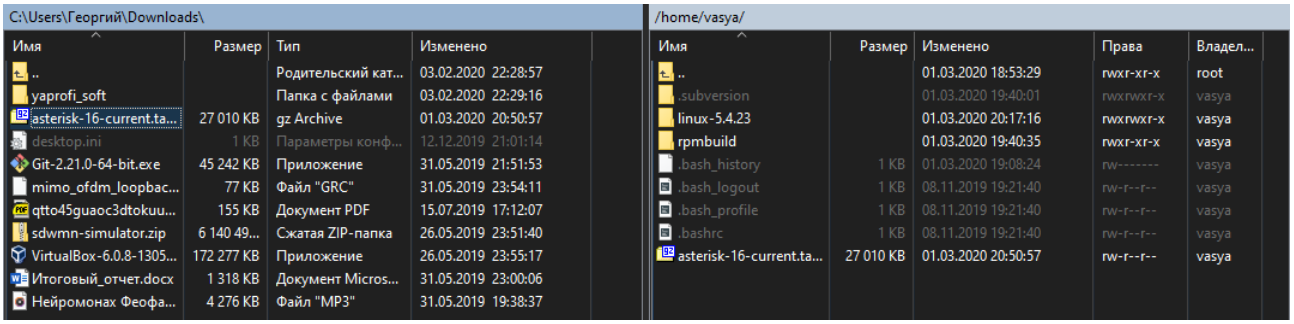
Asterisk устанавливается путем сборки из исходного кода. Полная установка проходит в 3 этапа:

1. (опц) Установка DAHDI (драйвера плат FXO интерфейсов);
2. (опц) Установка LibPRI (библиотека для работы с потоковыми TDM-интерфейсами);
3. Сборка и установка Asterisk.

При использовании тестового стенда на ВМ можно пропустить первые 2 пункта, поскольку использование спец оборудования не подразумевается, поэтому сразу приступим к сборке Asterisk. Архив с исходным кодом требуемой 16 версии Asterisk `asterisk-16-current.tar.gz` лежит в той же директории на сервере, где Вы брали образ CentOS; скачайте архив на свой ПК. Для того, чтобы перенести файл архива с хостового компьютера в CentOS, воспользуемся протоколом передачи файлов SCP (от англ. `secure copy`), который работает поверх SSH. Запустите на хостовом ПК программу WinSCP, которая представляет собой клиент таких протоколов передачи данных, как FTP SFTP SCP WebDAV и тд. В окне входа выберите протокол SCP, в поле Имя хоста укажите IP адрес CentOS, укажите имя пользователя и пароль в соответствующих полях (аналогично подключению по SSH). Нажмите Войти.



В режиме Коммандер WinSCP отображает слева папки Вашего локального пользователя на хостовом ПК, а справа домашнюю директорию пользователя, под которым Вы подключились к серверу (к CentOS). В левой части перейдите в папку, в которую Вы ранее скачали архив с исходным кодом Asterisk, выберите и перетащите нужный файл мышью на правую половину, убедитесь, что он появился в домашней директории Вашего пользователя на CentOS.



Распаковка архива и переход в папку с извлеченными файлами:

```
tar -xzf asterisk-*.tar.gz
cd asterisk-16.*
```

Следует отметить, что в скачанном архиве с исходным кодом Asterisk есть скрипт `(contrib/scripts/install_prereq)` для автоматической установки всех зависимостей, но он подразумевает установку множества пакетов для невостребованных в данной работе модулей, и в целях экономии времени,

дискового пространства и интернет-трафика, ранее была произведена “ручная” установка необходимых пакетов.

Для выполнения следующего шага потребуется прописать прокси для клиента системы управления версиями `subversion`. Создайте новый файл:
`sudo touch /etc/subversion/servers`

Или сразу укажите имя нового файла в вызове редактора, новый файл также будет создан.

В созданный файл `/etc/subversion/servers` необходимо добавить следующие строки:

```
[global]
http-proxy-host = прокси
http-proxy-port = порт
http-proxy-username = логин
http-proxy-password = пароль
```

Добавление пакетов для работы с mp3:
`./contrib/scripts/get_mp3_source.sh`

**если у вас эта команда не выполняется, проверьте, что вы находитесь в папке `asterisk-16.*` с распакованными файлами (проверка текущей директории командой `pwd`)*

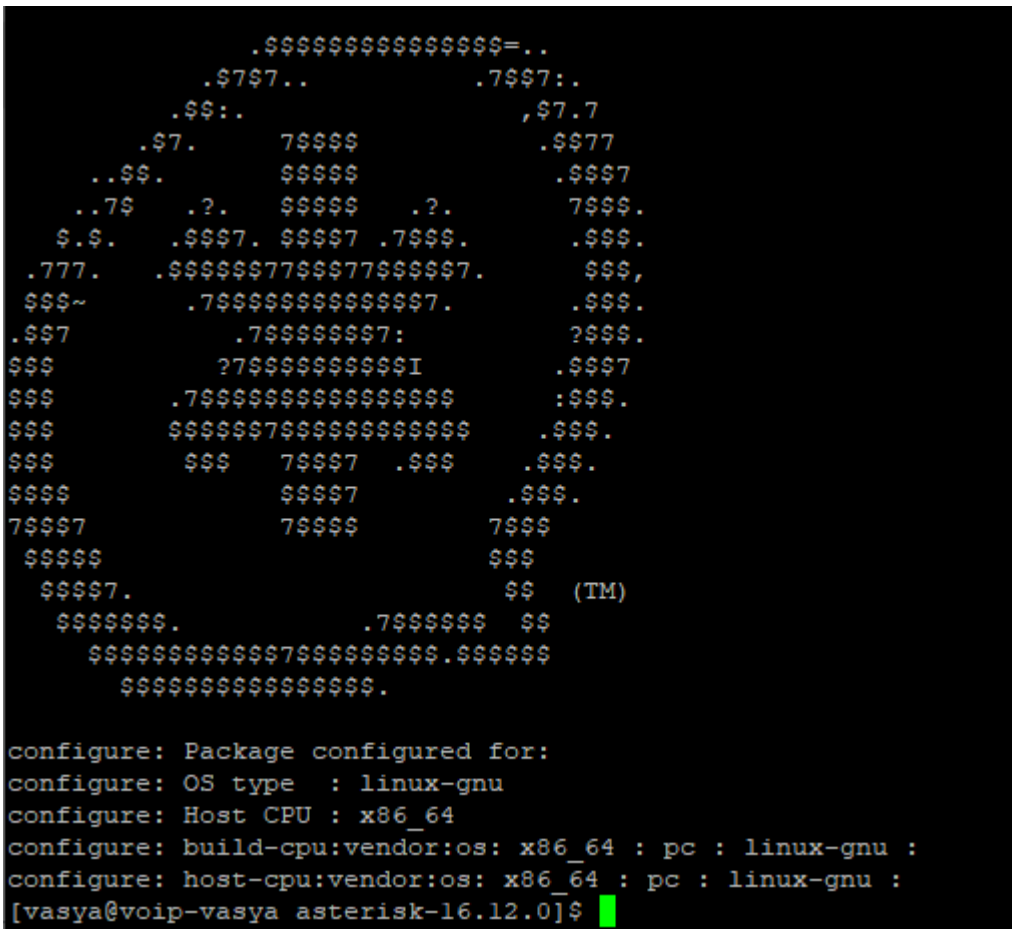
Очищение директории от существующих файлов конфигурации сборки:
`make distclean`

Конфигурация параметров сборки Asterisk (это одна команда, параметры, начинающиеся с - - разделяются пробелами):

```
./configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var --libdir=/usr/lib64 --with-iconv --with-libcurl --with-jansson-bundled --with-crypto --with-ssl=ssl --with-srtp
```

** полный перечень опций и что они означают можно посмотреть командой `./configure -h`*

В случае успешного конфигурирования в конце должна быть получена следующая картина



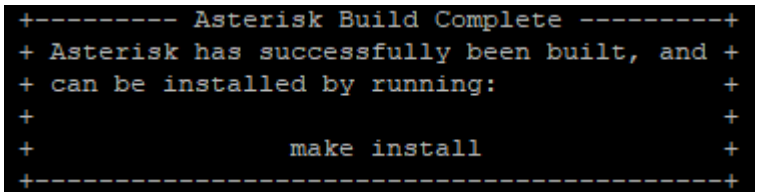
Меню выбора требуемых компонентов, модулей и приложений Asterisk:
`make menuselect`

Необходимые компоненты

- **Add-ons:** `format_mp3`
- **Call Detail Recording: убрать** `cdr_radius`, `cdr_sqlite3_custom`, `cdr_tds`
- **Channel Event Logging: убрать** `cel_radius`, `cel_sqlite3_custom`, `cel_tds`
- **Channel Drivers:** оставить только `chan_iax2`, `chan_pjsip`, `chan_rtp`
- **Codec Translators:** добавить `codec_opus`
- **Resource Modules: убрать** `res_agi`, все пункты с `res_ari`, `res_fax`, `res_phoneprov`, `res_smdi` (эти модули не нужны в данной установке и вызывают появление ошибок при запуске)
- **Compiler Flags:** `LOW_MEMORY`, `G711_NEW_ALGORITHM`, `G711_REDUCED_BRANCHING`
- **Core Sound Packages**: только `RU-WAV`.

После этого выйдите, нажав **Save & Exit.**
Запуск процесса сборки:
`make`

Если сборка прошла успешно, отобразится сообщение об успешном завершении и возможности установить собранный пакет:



Установка собранного пакета Asterisk:
`sudo make install`

Создание службы systemd для Asterisk и конфигурационных файлов по умолчанию:
`sudo make samples && sudo make config`

Установка завершена!

#Базовая конфигурация и запуск Asterisk

Откройте в редакторе основной конфигурационный файл Asterisk
`/etc/asterisk/asterisk.conf`

Необходимо раскомментировать (удалить ;) и редактировать следующие пункты:
```
runuser = asterisk
rungroup = asterisk
defaultlanguage = ru
```

Создайте служебного пользователя `asterisk`, от имени которого будет работать служба Asterisk (этот пользователь не сможет осуществлять вход в систему):
`sudo useradd -r -s /sbin/nologin asterisk`

Смените владельца следующих директорий на пользователя `asterisk` для предоставления ему полного доступа:
```
sudo chown -R asterisk:asterisk /var/run/asterisk
sudo chown -R asterisk:asterisk /etc/asterisk
sudo chown -R asterisk:asterisk /var/{lib,log,spool}/asterisk
sudo chown -R asterisk:asterisk /usr/lib64/asterisk
```

Запуск Asterisk для проверки корректности установки и первоначальной конфигурации:
`sudo asterisk -c`

Если Asterisk успешно запустится, в конце вывода служебных сообщений появится зеленая надпись **Asterisk Ready** и приглашение командной строки Asterisk ***CLI>** (*возможно появление предупреждений и ошибок - это обусловлено особенностями сгенерированной конфигурации по умолчанию, которая*

предполагает использование некоторых не сконфигурированных в текущей установке Asterisk модулей. На работоспособность в целом не влияет)
Выход обратно в bash: Ctrl+C

Запретите загрузку вызывающих ошибки модулей, которые не понадобятся в текущей работе:
`sudo vi /etc/asterisk/modules.conf`
вставить после строки `autoload=yes`:

```
noload => pbx_dundi
noload => res_config_ldap
noload => res_pjsip_phoneprov_provider
```

Если на предыдущем шаге Asterisk успешно запустился, можно запустить Asterisk как фоновую службу с автозапуском на старте ОС:
`sudo systemctl enable asterisk && sudo systemctl start asterisk`

Проверьте, что Asterisk корректно запущен в виде фоновой службы:
`sudo systemctl status asterisk`

Если служба корректно запущена и работает, в выводе должно быть указано `active (running)`.

Troubleshooting
Если получено иное, перезапустите службу и еще раз проверьте статус
`sudo systemctl restart asterisk`
`sudo systemctl status asterisk`

Проверьте возможность подключения к интерфейсу CLI Asterisk
`sudo asterisk -vvvr`
Подключаться к Asterisk Вам потребуется позже при отладке конфигурации, пока что выйдите обратно в bash.

**Если CLI Asterisk не открылся, проверьте состояние SELinux (`sudo sestatus`, `Current mode` не должен быть `enforcing`).*

4. Конфигурация Asterisk для обработки вызовов

Чтобы Asterisk начал обрабатывать телефонные вызовы, необходимо сконфигурировать параметры канала связи (драйвер протокола SIP в данном случае) и план набора (dialplan). Протокол SIP представлен в Asterisk двумя драйверами канала - старым, проприетарным `chan_sip` и новым, открытым кроссплатформенным `PJSIP`. Преимущества последнего помимо

кроссплатформенности в стабильности, активном развитии (исправлении багов и уязвимостей, оптимизации и обновлении), поддержке дополнительных протоколов (TURN и тд) для обеспечения лучшей проходимости трафика SIP через NAT, в возможности осуществлять множество регистраций на одну учетную запись клиента. Таким образом, для данной работы был выбран PJSIP как более функциональное современное решение. План набора конфигурируется как описание логики обработки вызовов практически одинаково для любого типа канала, отличия только в вызываемом приложении внутри записи экстеншена (пункт диалплана) .

Для конфигурации Asterisk в качестве IP АТС согласно изложенному ранее, необходимо отредактировать два основных конфигурационных файла:
1)`pjsip.conf`, в котором указываются сведения о транспортном протоколе, разрешенных кодеках, клиентах и регистрациях SIP, пользователи с соответствующими идентификаторами-номерами, параметры регистрации и аутентификации.
2)`extensions.conf`, описывающий план набора - `dialplan`, правила обработки и маршрутизации вызовов.

После внесения изменений в любой конфигурационный файл Asterisk, служба Asterisk должна быть перезапущена для применения изменений!!!
```
sudo systemctl restart asterisk
```

Удалите файлы конфигурации, созданные по умолчанию:
```
sudo rm /etc/asterisk/extensions.conf
sudo rm /etc/asterisk/pjsip.conf
```

5. Конфигурация плана набора

План набора в файле `extensions.conf` структурирован в секции, называемые контекстами. Контекст - это независимая от остальных часть внутри диалплана. Контексты используются для разделения функций, обеспечения безопасной обработки и фильтрации вызовов между различными частями, определения класса обслуживания разных пользователей и так далее.

План набора, как было сказано ранее, состоит из одного или нескольких контекстов. Контексты используются для реализации основных функций АТС, таких как:

- Безопасность: Можно разрешить вызовы на определенные номера только конкретным абонентам.
- Маршрутизация вызовов: Маршрутизация вызовов в зависимости от номера абонента.
- Многоуровневые голосовые меню: Голосовые меню для службы поддержки, отдела продаж и т.д.

- Авторизация: Запрос пароля для вызова на некоторые номера.
- Обратный вызов.
- Списки доступа: Занесение в черные списки нежелательных абонентов.
- Виртуальные АТС: Возможность создавать независимые виртуальные АТС в пределах Вашей основной АТС.
- Дневной/Ночной режим работы: Изменение поведения АТС в зависимости от времени.
- Макросы: Можно создавать скрипты для решения повторяющихся задач в плане набора.

Каждый контекст - это набор расширений (extension). Каждый экстеншен в контексте имеет уникальное имя, которое обычно является числовым идентификатором, присвоенным линии, идущей к конкретному телефону.

Синтаксис расширения начинается с выражения `exten =>` Далее указывается имя (или номер) экстеншена. В традиционных системах телефонной связи под номерами понимаются номера из цифр, которые надо набрать, чтобы позвонить определенному абоненту с этим номером. В Asterisk понятие имени (номера) намного шире, в качестве имени добавочного номера может использоваться любая комбинация цифр и букв. Полный экстеншен состоит из трех компонентов:

1. Имени (или номера).
2. Приоритета (каждый добавочный номер может включать множество шагов обработки вызова, порядковый номер шага называется его приоритетом).
3. Приложение (или команда), которое выполняет некоторое действие над вызовом.

Эти три компонента разделяются запятыми:
```
exten => имя,приоритет,приложение()
```

Приведём пример простейшего экстеншена:
```
exten => 123,1,Answer()
```

В этом примере имя добавочного номера – 123, приоритет – 1, а приложение – Answer().

В начале диалплана также можно разместить два специальных контекста, [general] и [globals].
[general] содержит список общих настроек диалплана
[globals] - глобальные переменные.

Ниже приведен пример простейшего плана набора (реализует обработку звонков на любые трехзначные номера) в контексте `default`, который должен быть, сохранен в файле `/etc/asterisk/extensions.conf`

```
[default]
exten => _XXX,1,Dial(PJSIP/${EXTEN})
```

Теперь рассмотрим реализацию простого голосового меню с подробным описанием процесса маршрутизации и обработки вызовов.

```
[internal]
exten => 1234,1,GoTo(ivr,s,1)

[ivr]
exten => s,1,Answer()
 same => n,Playback(hello)
 same => n,Background(basic-pbx-ivr-main)
 same => n,Playback(demo-thanks)
exten => 1,1,GoTo(1-otd,s,1)

[1-otd]
exten => s,1,Background(one-moment-please)
 same => n,GoTo(ivr,s,4)
```

При звонке клиента из контекста `internal` на номер 1234 происходит перенаправление вызова приложением `GoTo` в контекст `ivr` на соответствующий экстеншен с именем `s` и приоритетом 1. Далее пошагово идёт обработка вызова: ”поднимается трубка” приложением `Answer`, проигрывается звуковая запись `hello` с помощью приложения `Playback` (по умолчанию Asterisk ищет записи в `/var/lib/asterisk/sounds` для указанного в конфигурации языка), далее проигрывается запись `basic-pbx-ivr-main` с помощью приложения `Background`. Отличие между этими двумя приложениями состоит в том, что во втором случае Asterisk во время проигрывания звукового файла прослушивает линию на предмет ввода дополнительного номера в тональном режиме (DTMF). При вводе 1 обработка вызова передается в контекст `1-otd`. Если ввода не последовало, проигрывается `demo-thanks` и вызов завершается, поскольку больше нет настроенных экстеншенов для дальнейшей обработки вызова.

Задание. Создайте контекст с произвольным именем, настройте обработку внутренних 4-значных номеров, определите отдельно любой произвольный номер, при вызове на который Asterisk будет проигрывать запись `num-was-successfully` и принудительно завершать вызов с помощью приложения `Hangup()`.

6. Конфигурация PJSIP

Настройки PJSIP в Asterisk производятся через текстовый файл конфигурации `pjsip.conf`, состоящий из секций. Общий вид секций стандартен для всех конфигурационных файлов Asterisk - начинается с указания имени секции в скобках`[]`. Основное отличие в структуре конфигурационного файла PJSIP от классического драйвера chan_sip, в том что конфигурация SIP-клиентов разбивается на логические разделы:

– `Endpoint` — соответствует клиенту или транку SIP, содержит описание основных параметров клиента, его принадлежность к контексту и определяет связь с остальными обязательгымиг модулями, такими как Transport, Auth и AOR.
– `Transport` — данный раздел описывает тип транспортного протокола для подключаемых устройств, доступны TCP, UDP, TLS, а также WebSocket. Возможно использовать один раздел для многих конечных точек (разделов Endpoint), также можно при необходимости для раздела Endpoint создать свой собственный раздел Transport.
– `Auth` — раздел, содержащий параметры аутентификации для исходящей или входящей регистрации SIP, с данным разделом связаны разделы Endpoint и Registrations. Одна запись раздела Auth при необходимости может использоваться несколькими разделами Endpoint и Registration (т.е. имя пользователя для регистрации совсем необязательно соответствует номеру пользователя).
– `AOR (Address of Record)` — раздел по своей сути является указателем для Asterisk, каким образом связаться с точкой (Endpoint). Без соответствующей записи в AOR не будет возможности вызвать подключаем конечную точку (телефонный аппарат клиента или передать вызов в транк оператора).
– `Registration` — раздел, отвечающий за исходящую регистрацию, например, регистрация Asterisk на сервере оператора связи. Для корректной работы данного раздела обязательно должны присутствовать две опции в которых указываются имена используемых разделов: раздел Transport (опция transport) и раздел Auth (опция outbound_auth); также обязательна опция type.

Пример конфигурации транспорта SIP и создание клиента с 3-значным номером, описанные в файле `/etc/asterisk/pjsip.conf`

```
[udp-transport]
type=transport
protocol=udp
bind=0.0.0.0

[101]
type=endpoint
```

```
transport=udp-transport
context=default
disallow=all
allow=g726,gsm
auth=101
aors=101

[101]
type=auth
auth_type=userpass
password=verysecuresupercoolencryptedpassword
username=101

[101]
type=aor
max_contacts=1
```

Подсказка

Посмотреть, какие кодеки поддерживает Asterisk:

***CLI>** *core show codecs*

Задание. Создайте конфигурацию для 2-х клиентов, каждому из которых соответствует 4-значный номер (номера выберите сами), привяжите к ранее созданному в плане набора контексту, разрешите использование только кодеков OPUS, G.711 A-law (корректные имена кодеков узнайте самостоятельно).
Обратите внимание, PJSIP очень чувствителен к корректности синтаксиса файла конфигурации, даже отсутствие/ наличие пробела там где он должен/не должен быть приводит к полной неработоспособности. Перезапустите службу Asterisk после изменения файлов конфигурации.

Теперь проверим, что Asterisk готов принимать вызовы, приходящие по сети, для этого нужно убедиться, что прослушивается порт SIP - 5060. Получить статистику по открытым портам и соединения можно с помощью нескольких команд, одна из них - ss

```
ss - ulpn
```

```
[vasya@voip-vasya asterisk-16.12.0]$ ss -ulpn
State      Recv-Q     Send-Q          Local Address:Port           Peer Address:Port
UNCONN     0          0                     0.0.0.0:4569                0.0.0.0:*
UNCONN     0          0                     0.0.0.0:5060                0.0.0.0:*
```

Протоколы SIP и RTP

Современная IP-телефония строится на связке сигнальных протоколов и протоколов передачи данных реального времени. В качестве сигнального протокола, обеспечивающего установление сессий передачи данных наибольшую популярность обрел SIP в связке с SDP, непосредственно для

передачи данных используется протокол прикладного уровня RTP и тесно связанный с ним транспортный протокол RTCP.

Session Initiation Protocol (SIP) – это клиент-серверный протокол сигнализации прикладного уровня, предназначенный для установления, модификации и окончания сеансов связи с одним или несколькими участниками для обмена мультимедийным трафиком. SIP использует текстовые сообщения, в которых используется кодировка UTF-8, работает на порту 5060, как в случае использования транспортного протокола UDP, так и TCP. Описан в RFC 3261.

Основными функциями SIP являются:

- Определение местонахождения адресата.
- Определение готовности адресата установить контакт.
- Обмен данными о функциональных возможностях участников сеанса.
- Изменение параметров медиапотока уже установленного сеанса.
- Управление сеансом связи.

Основными функциональными элементами являются:

- ***Абонентский терминал (User Agent)***. SIP-клиент, реализованный как АО (VoIP телефон/VoIP-шлюз) или ПО (PhonerLite, SIPp), с помощью которого абонент совершает вызовы.
- ***Прокси-сервер***. Узел в сети, принимающий и обрабатывающий запросы от терминалов, выполняя соответствующие этим запросам действия и возвращая ответы. Прокси-сервер может принимать вызовы, переадресовывать их, вносить изменения в передаваемые сообщения SIP (например, для преодоления NAT), инициировать запросы к клиентам.
- ***Сервер переадресации***. Узел, хранящий записи о текущем местоположении всех зарегистрированных в сети клиентах (терминалах) и прокси-серверах. Сервер переадресации только переадресует вызовы и не генерирует собственные запросы.
- ***Сервер регистрации / определения местоположения пользователей (Register)***. Представляет собой базу данных адресной информации (IP-адресов). Необходим для обеспечения мобильности пользователей. Чаще всего совмещен с прокси-сервером

Asterisk в стандартной установке совмещает все 3 последние упомянутые роли.

В протоколе SIP определено 6 основных запросовов:

Метод	Описание
REGISTER	Запрос на регистрацию в сети, идентификация

	местоположения клиента в сети.
INVITE	Вызов другого участника для начала сеанса (установление соединения). Передаются виды поддерживаемых инициатором сервисов (сообщения SDP).
ACK	Используется для подтверждения готовности установить соединение (также могут быть переданы окончательные параметры сеанса).
CANCEL	Отмена ранее переданных запросов.
BYE	Запрос на завершение сеанса связи.
OPTIONS	Запрос информации о функциональных возможностях терминала адресата

Позднее SIP был расширен введением функций обмена мгновенными сообщениями и получения информации о статусе клиента. Дополнительные методы, описаны в RFC:

- info: Расширение протокола, описанное в RFC 2976
- notify: Расширение протокола, описанное в RFC 2848
- subscribe: Расширение протокола, описанное в RFC 2848
- unsubscribe: Расширение протокола, описанное в RFC 2848
- update: Запрос на изменение параметров сеанса, описан в RFC 3311
- message: Расширение протокола, описанное в RFC 3428
- refer: Расширение протокола, описанное в RFC 3515

Для SIP определено 6 кодов ответа, которыми прокси-сервер описывает состояние соединения, например: подтверждение установления соединения, передача запрошенной информации, сведения о неисправностях и пр. Все классы ответов, кроме 1 завершают выполнение запроса.

1. 1xx — Информационные ответы, сообщают о ходе выполнения запроса;
2. 2xx — Успешное окончание запроса;
3. 3xx — Информация об изменения местоположения вызываемого абонента;
4. 4xx — Информация об ошибке;
5. 5xx — Информация об ошибке сервера;
6. 6xx — Информация о невозможности вызова абонента (пользователь с таким адресом не зарегистрирован, или пользователь занят).

Непосредственным носителем голосовых или видеоданных является протокол RTP (Real-time Transport Protocol), SIP-сообщение же выполняет роль контейнера для сообщений протокола описания сеансов связи SDP (Session Description Protocol RFC 4566).

Протокол SDP используется для согласования параметров сессии передачи данных. Сообщение SDP описывает медиаданные в рамках сессии: тип медиаданных, транспортный протокол, кодек и тд. Последнее позволяет выполнять более совершенное управление голосовыми вывозами с переадресациями и продвинутой маршрутизацией. SIP поддерживает также приглашение участников к текущим сеансам наподобие многоточечных конференций, добавление к текущему сеансу или удаление из него мультимедийных данных, прозрачное распределение имён и перенаправление услуг, включая персональную мобильность пользователя.

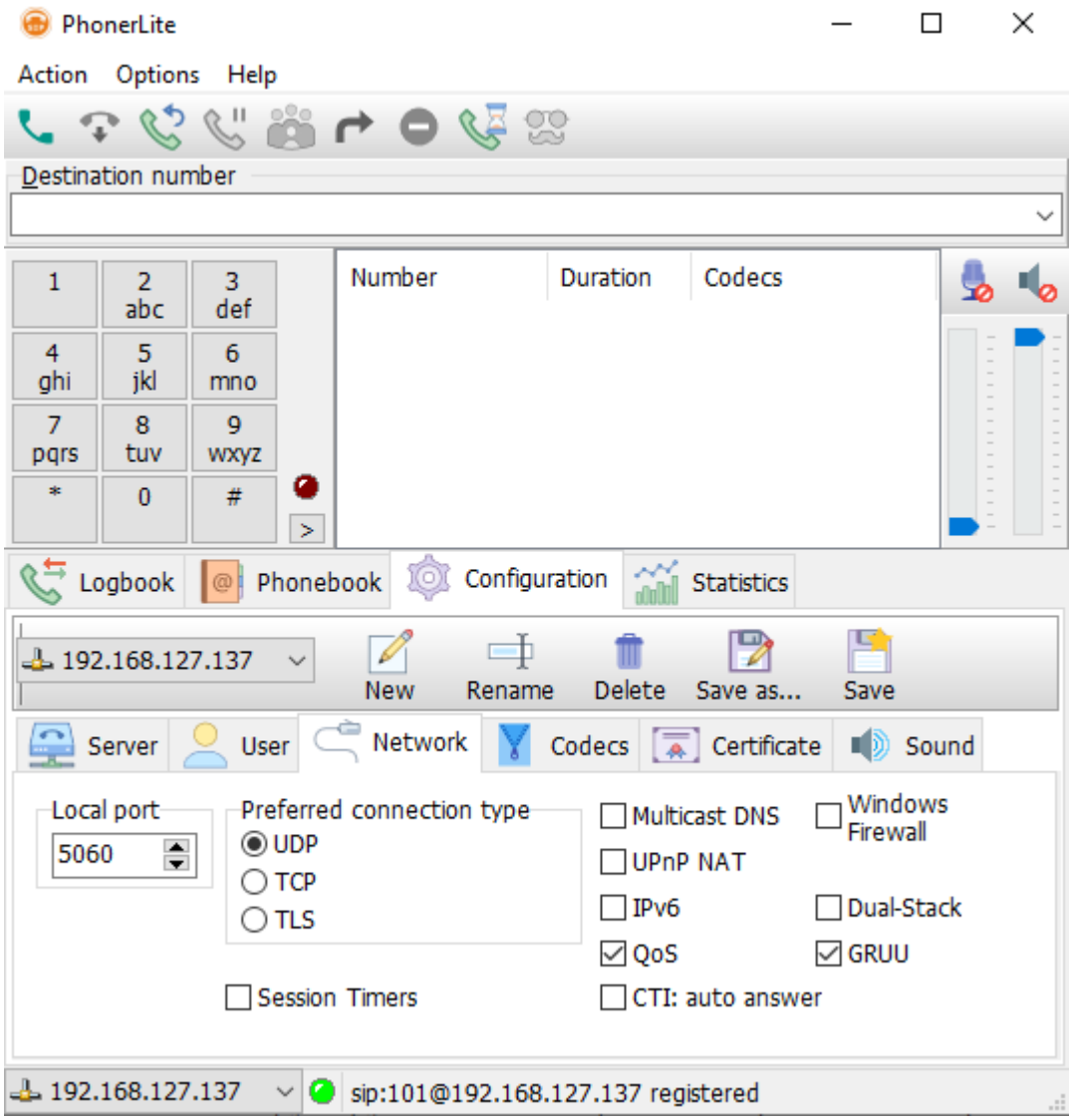
RTP - транспортный протокол для передачи трафика реального времени (в том числе потоковой передачи мультимедийного трафика). Для передачи и контроля параметров соединения используется протокол RTCP

Настройка клиентов

В терминале подключитесь к службе Asterisk (`sudo asterisk -vvvr`), оставьте открытой консоль Asterisk, чтобы там видеть информационные сообщения о регистрации клиентов и обработке вызовов. Из директории с руководством по данной работе на сервере скачайте на свой компьютер архив PhonerLite.zip, разархивируйте его в 2 разные папки для двух разных клиентов. Сначала проделайте все указанные ниже шаги для одного клиента, если клиент успешно зарегистрировался и может совершить звонок на номер Asterisk с проигрыванием записи `num-was-succesfully`, то сконфигурируйте и второго клиента для 2 номера.

Запустите приложение PhonerLite.exe, выберите manual configuration. В строке Proxy/Registrar укажите IP-адрес Asterisk, нажмите далее (кнопка со стрелкой влево), введите имя пользователя и пароль (из секции `Auth` pjsip.conf). Завершите оставшиеся этапы конфигурации клиента.

Если регистрация клиента на сервере прошла успешно, внизу окна появится соответствующая надпись и индикатор будет зелёным, а в интерфейсе Asterisk появятся сообщения о том, что прошла регистрация и клиент с номером … стал доступен.



Во вкладке Configuration - Network снимите галочку с пункта Multicast DNS, нажмите Save.

**иначе возможны некоторые нежелательные ситуации, поскольку клиенты умеют находить друг друга в сети посредством рассылки запросов и при вызове могут игнорировать Asterisk.*

Первому клиенту PhonerLite поставьте первым разрешенным приоритетом кодек Opus, второму - G.711 A-law, совершите звонок. Убедитесь, что в консоли Asterisk отображается информация об обработке вызова, если нет, проверьте, что неактивен пункт Multicast DNS, закройте клиенты, перезапустите Asterisk, откройте консоль Asterisk, запустите клиенты и попробуйте заново.

#Отладка SIP протокола

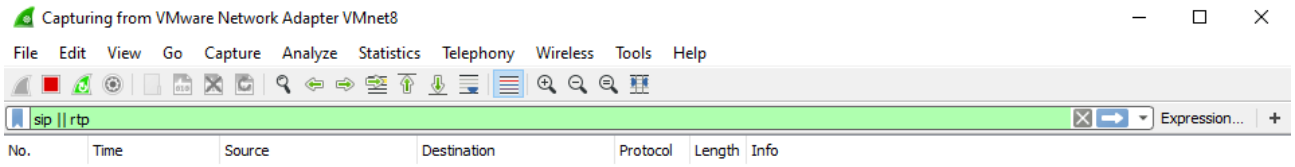
Модуль `res_pjsip_history` сохраняет в памяти историю всех отправленных и полученных SIP-сообщений, которые проходят через стек PJSIP.

Для того, чтоб начать захват, нужно выполнить следующую команду в CLI Asterisk:

```
pjsip set history on
```

Для наглядного отображения порядка проведения VoIP вызова с помощью протоколов SIP и RTP, можно использовать Wireshark - инструмента для захвата и анализа сетевого трафика. Запустите Wireshark и выберите соответствующий сетевой адаптер для захвата трафика из списка (VMnet8 для сети NAT в VMware Player).

В строке Apply a display filter введите `sip || rtp` для отфильтровывания в выводе пакетов соответствующих протоколов.



Совершите вызов из PhonerLite второму клиенту, не забудьте принять вызов на втором PhonerLite. После завершения вызова (достаточно принять вызов и можно сразу же нажать отбой) нажмите на красный квадратик слева сверху, чтобы остановить захват трафика и перейти к его анализу. В пункте Telephony выберите VoIP Calls, в открывшемся окне будут отображены все захваченные звонки (сеансы SIP). Выберите один из них и нажмите внизу Flow Sequence, чтобы наглядно отобразить этапы установления и завершения соединения с помощью протокола SIP.

Сравните с историей SIP сообщений на Asterisk:

```
pjsip show history
```

Также можно получить информацию о клиентах:

```
pjsip show endpoints
```

Вопрос. Почему некоторые сообщения SIP (в частности INVITE при инициализации сеанса) клиент отправляет по 2 раза?