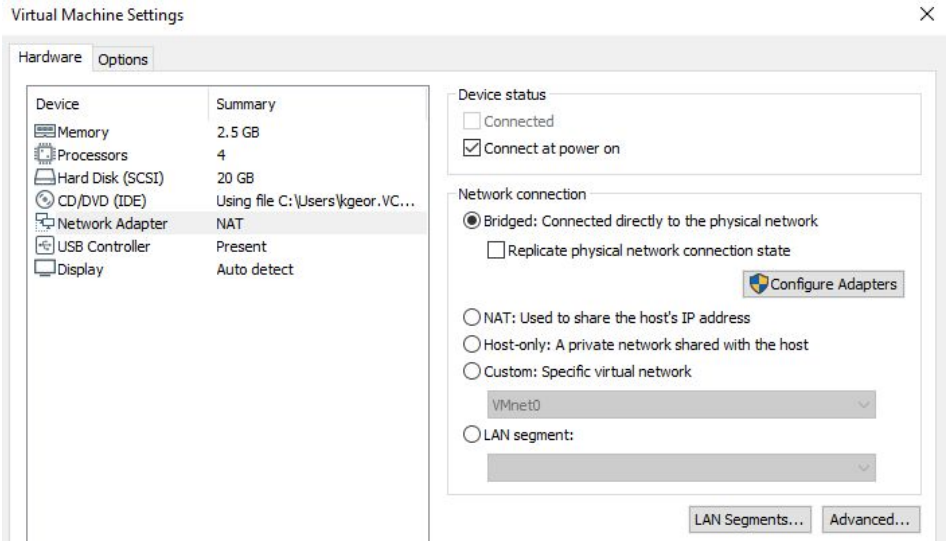
“Настройка канала между Asterisk. Realtime конфигурация.”

Часть 1.

Эту часть работы удобнее всего выполнять парами, поскольку потребуется два узла Asterisk. Для обеспечения сетевой связности между двумя ВМ, запущенными на разных ПК, потребуется внести некоторые изменения в конфигурацию каждой:

1) Откройте VMware Player, убедитесь, что ваша ВМ с CentOS выключена (State: **Powered off**), нажмите ***Edit virtual machine settings.***
2) Нажмите **Add...**, выберите Network adapter. Отредактируйте конфигурацию нового сетевого адаптера (***Network Adapter 2***), укажите ***Bridged*** вместо **NAT** в Network connection. Сохраните изменения.



3) После запуска ВМ настройте новый адаптер согласно инструкции ниже и убедитесь, что CentOS получает адрес из той же сети, в которой находится хостовой ПК.

Определите, какое имя получил новый адаптер в системе, выполните команду `ip address`, новый адаптер отобразится последним в списке.

Сгенерируйте уникальный идентификатор с помощью команды `uuidgen` *`имя_адаптера`* скопируйте полученное значение, его потребуется вставить в поле файла конфигурации на следующем шаге. Создайте файл конфигурации для нового адаптера в директории /etc/sysconfig/network-scripts с именем `ifcfg-`*`имя_адаптера`*.

В файле следует указать следующие параметры (пример актуален для подсети с `DHCP`)

```
TYPE=Ethernet

BOOTPROTO=dhcp

NAME=имя_адаптера

ONBOOT=yes

UUID=полученное_значение_от_uuidgen

NM_CONTROLLED=yes
```

Применение изменений в CentOS 8:

```
systemctl restart NetworkManager
```

Troubleshooting *Если новый сетевой адаптер (ens или eth) в CentOS не получил IP адрес (или получил вида 169.254.x.x), в настройках нового сетевого адаптера (Player - Removable Devices - Network Adapter 2 - Settings) выберите* ***Configure Adapters****, в появившемся окне снимите выделение со всех адаптеров, кроме того, через который хостовой ПК имеет доступ к локальной сети. Если потребуются права администратора, проконсультируйтесь с преподавателем на предмет, какие учетные данные использовать.*

На одном узле Asterisk (назовем его `node1`) требуется добавить клиента с номером в формате 4XXX, на втором (`node2`) в формате 5XXX (помните, клиенты описываются в файле `pjsip.conf` 3 секциями - `Endpoint`, `Auth`, `AOR`), контекст укажите тот, который создали в предыдущей работе для внутренних вызовов. Назначьте клиенту произвольный идентификатор CallerID (см. пример ниже как назначается идентификатор `first` клиенту с номером 4001).

Пример секции Endpoint из `/etc/asterisk/pjsip.conf` для `node1`

```
[4001]
type=endpoint
transport=udp-transport
context=internal
disallow=all
allow=alaw,opus
callerid=first<4001>
auth=4001
aors=4001
```

ВАЖНО! Убедитесь, что на первом Asterisk отсутствуют настроенные номера формата 5XXX, а на втором 4XXX.

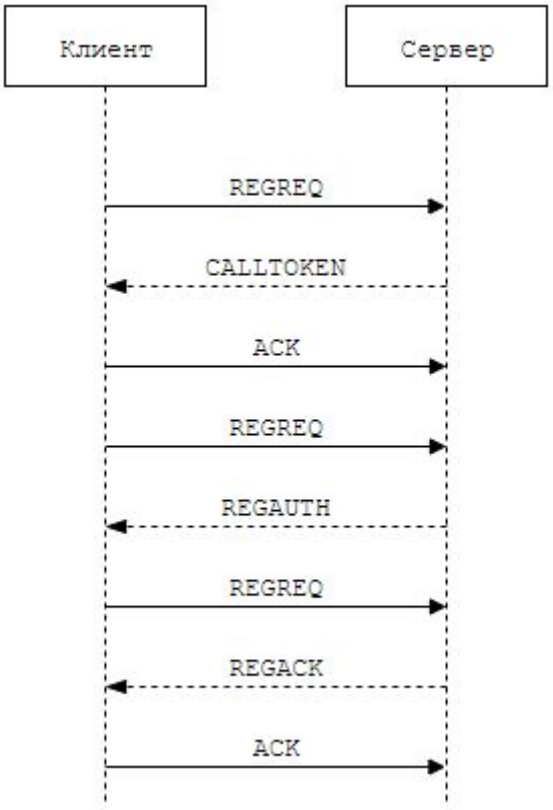
#Краткие сведения о протоколе IAX

Среди поддерживаемых Asterisk протоколов, особое место занимает протокол IAX. Это протокол, созданный разработчиками Asterisk (Digium Inc.), расшифровывается как Inter-Asterisk eXchange protocol - протокол обмена между

Asterisk. Хотя IAX и поддерживается некоторыми VoIP-телефонами, в первую очередь он предназначен для организации между узлами Asterisk каналов типа транк, в которых осуществляется одновременная передача данных нескольких вызовов. IAX первой версии в настоящее время не поддерживается ввиду наличия проблем в плане безопасности, а говоря IAX, обычно подразумевают вторую версию IAX2. Для работы протокола используется UDP-порт 4569, который предназначается не только для обмена сигнальной информацией о сеансе связи (аналогично SIP с SDP), но и для обмена голосовой информацией (аналогично RTP), таким образом, IAX берет на себя передачу и сигнальной и голосовой информации. Дополнительно следует сказать, что IAX - протокол, использующий бинарный формат для передачи данных, в отличие от протокола SIP, передающего сообщения в открытом текстовом формате, что позволяет IAX уменьшить объем передаваемых данных в случае использования его на транк-канале с одновременной передачей множества вызовов. Помимо этого, протокол использует агрегацию параллельных сеансов связи в рамках использования одного IAX-соединения между узлами Asterisk, так что в одном UDP-пакете может одновременно передаваться служебная и голосовая информация, принадлежащая нескольким вызовам.

Основные недостатки - сложная расширяемость (внедрение новых функций) и уязвимость старых реализаций к атакам типа DoS (версии Asterisk новее 2009 года менее подвержены этой уязвимости, поскольку есть возможность ограничить число подключений и вызовов с одного адреса, использовать дополнительно при соединении токены CallToken).

Процесс регистрации клиента IAX на сервере с настроенной аутентификацией с использованием алгоритма хэширования MD5 при поддержке CallToken:



1) Клиент отправляет серверу сообщение REGREQ с информационными элементами ‘username’ (содержит имя пользователя, под которым нужно зарегистрироваться), ‘refresh’ (период обновления регистрации в секундах) и пустым ‘CallToken’;

2) Сервер отвечает сообщением CALLTOKEN, содержащим элемент CallToken с идентификатором, который должен использовать клиент;

3) Клиент отправляет подтверждает получение токена сообщением ACK и посылает новое сообщение REGREQ (с элементами username, refresh и CallToken с полученным от сервера значением);

4) Сервер отвечает сообщением REGAUTH, содержащим элементы authentication methods, username и MD5 challenge (данные для вычисления хэша пароля);

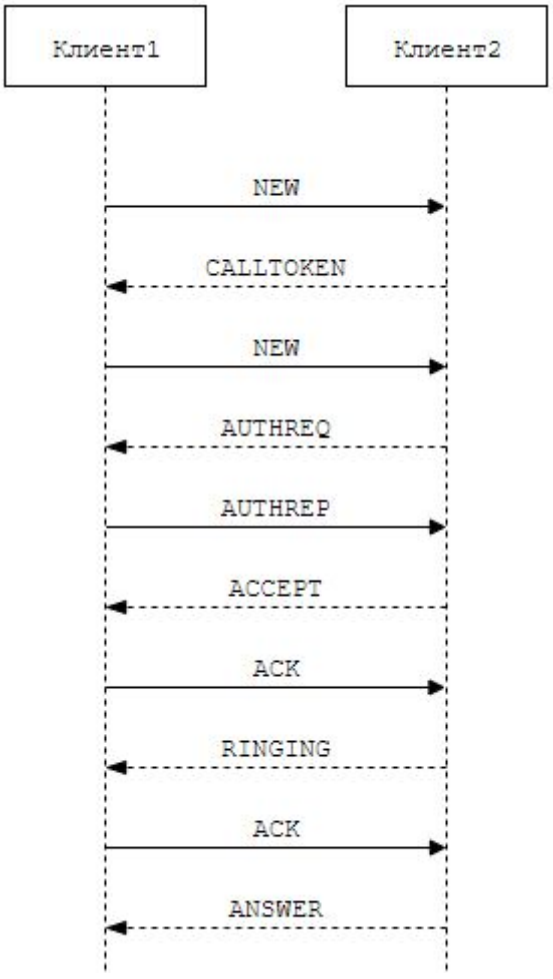
5) Клиент посылает финальное сообщение REGREQ с элементами username, refresh, CallToken и MD5 challenge с результатом выполнения хэш-преобразования по алгоритму MD5 над указанным в конфигурации паролем пользователя IAX;

6) Сервер, в случае, если полученный от клиента хэш совпадает с вычисленным от хранимого на сервере Asterisk пароля, отвечает клиенту сообщением REGACK с элементами username, date time (текущие дата/время), refresh, apparent address (содержит в себе адрес клиента);

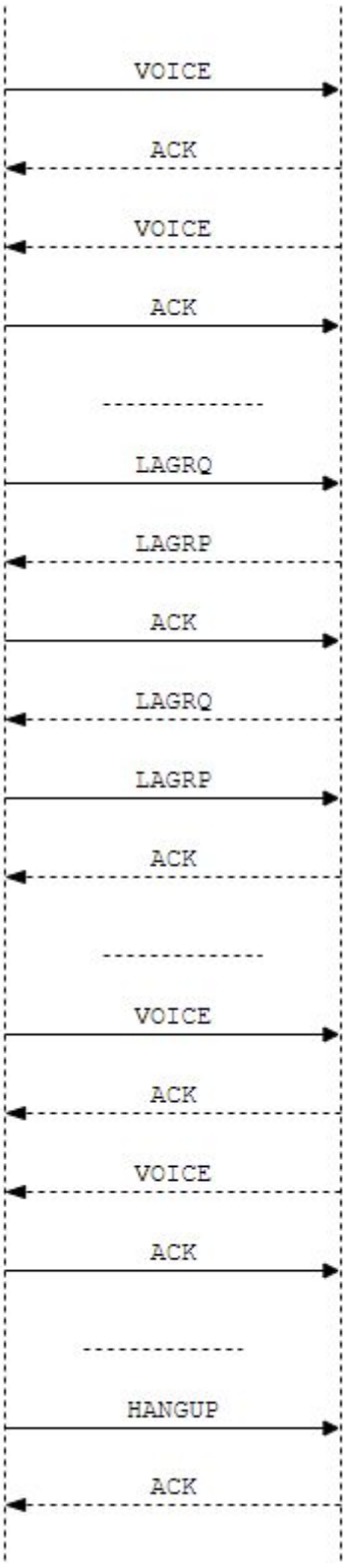
7) Клиент отсылает финальное сообщение ACK, если полученные данные корректны.

Для IAX регистрация означает отслеживание клиента (его адреса и порта прослушивания), все вызовы, проходящие через канал IAX проходят аутентификацию отдельно, впрочем, как и для SIP.

Ниже приведена диаграмма обмена сообщениями при совершении вызова: клиента 1 отправляет запрос NEW с параметрами вызова (номер, кодеки, пользователь IAX, дата/время) клиенту 2, клиент 2 отвечает сообщением, содержащим CallToken для этого вызова, клиент 1 повторно отправляет то же NEW с полученным CallToken. Клиент 2 высылает параметры для аутентификации (данные для MD5/RSA преобразования пароля) в AUTHREQ. Клиент в данном представлении может быть как Asterisk, так и конечным клиентом.



Происходит разговор клиента 2 с клиентом 1 (периодически посылаются сообщения LAGRQ (Request) LAGRP (Response) для определения задержки между узлами). Вызов завершается клиентом 1 (сообщение HANGUP).



#Настройка транка IAX в Asterisk

Приведенная ниже конфигурация предполагает, что оба клиента осуществляют взаимную регистрацию (схема peer-to-peer для равноправных узлов), в других ситуациях регистрация может быть односторонней (схема

клиент-сервер), как например при подключении корпоративной АТС к провайдерской.

Удалите сгенерированный по умолчанию файл конфигурации IAX
`sudo rm /etc/asterisk/iax.conf`

Конфигурация протокола IAX для данной работы должна производиться в новом файле `/etc/asterisk/iax.conf`, добавьте следующие строки

На `node1`:

```
[iaxuser1]
type=friend
qualify=yes
auth=md5
context=outcall
secret=secret1
host=dynamic
trunk=yes
```

На `node2`:

```
[iaxuser2]
type=friend
qualify=yes
auth=md5
context=outcall
secret=secret2
host=dynamic
trunk=yes
```

В приведенной конфигурации создается пользователь с именем, указанным в `[]` и паролем, заданным параметром `secret`, от имени которого удаленный узел сможет зарегистрироваться на данном Asterisk. Указанный тип `friend` дает пользователю право отправлять и принимать вызовы (существуют также типы `user` и `peer`, первый позволяет ТОЛЬКО принимать вызовы от удаленного узла, второй - ТОЛЬКО перенаправлять их на удаленный узел).

Параметр `qualify` включает опрос доступности удаленного узла (отправка сообщения POKE, ожидание в ответ PONG, отправка подтверждения получения ответа PONG сообщением ACK), параметр `auth` определяет тип аутентификации (по умолчанию происходит аутентификация с передачей учетных данных в открытом текстовом виде) md5 - передача хешированных данных. Параметр `context` определяет, в какой(ие) локальный(е) контекст(ы) должен быть передан пришедший из IAX-канала вызов.

Настройка взаимной регистрации IAX-серверов. Для этого добавьте в начало файла `/etc/asterisk/iax.conf`

на `node1`:

```
[general]
autokill=yes
language=ru
disallow=all
allow=opus,alaw
register => iaxuser2:secret2@A.B.C.D
```

где `A.B.C.D` *IP адрес* node1 (`IP_node1` далее)

На `node2` :

```
[general]
autokill=yes
language=ru
disallow=all
allow=opus,alaw
register => iaxuser1:secret1@A.B.C.D
```

где `A.B.C.D` *IP адрес* node2 (`IP_node2` далее)

Рассмотрим конфигурацию, проделанную на `node1`. Она предписывает узлу `node1` зарегистрироваться на узле `node2` с IP-адресом `IP_node2` под пользователем `iaxuser2` с паролем `secret2`. Параметр `autokill=yes` завершает неудачно установленные соединения по тайм-ауту.

Для принятия изменений перезагрузите службу Asterisk

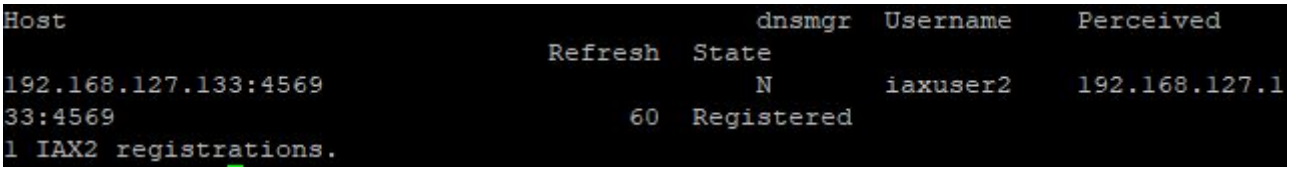
```
sudo systemctl restart asterisk
```

Проверка состояния регистрации данного узла на удаленном:

```
sudo asterisk -rx 'iax2 show registry'
```

**данная команда демонстрирует, как можно выполнять любые команды CLI Asterisk (и получать ответный вывод) напрямую из bash*

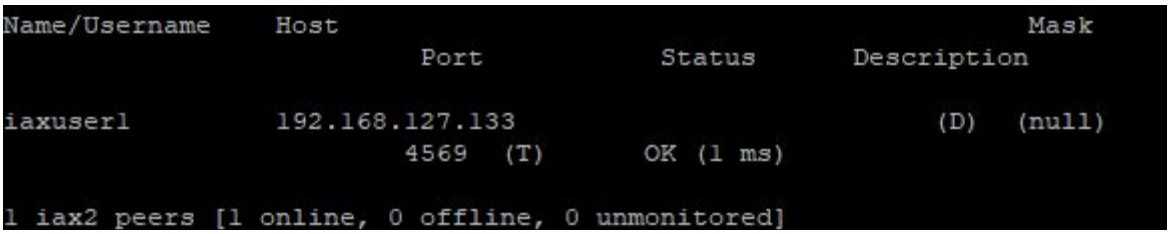
`В полученном выводе State` должно быть `Registered` (т.е. локальный Asterisk успешно зарегистрировался на удаленном). Состояние `Rejected` означает, что в регистрации было отказано, следует выяснить и устранить причину.



Состояние регистрации удаленного узла на данном:

```
sudo asterisk -rx 'iax2 show peers'
```

`Status` должен быть `OK`, а внизу указано, что есть `1 online`.



Вся необходимая конфигурация IAX выполнена, осталось только указать обоим узлам, вызовы на какие номера нужно передавать через IAX-подключение к другому Asterisk.

Для обработки приходящих из IAX-транка вызовов, на обоих узлах нужно добавить указанный ранее в конфигурации IAX контекст `outcall` в файл `/etc/asterisk/extensions.conf`

На `node1`:

```
[outcall]
exten => _4XXX,1,Dial(PJSIP/${EXTEN})
```

На `node2`:

```
[outcall]
exten => _5XXX,1,Dial(PJSIP/${EXTEN})
```

В файле `/etc/asterisk/extensions.conf` добавьте в начало контекста, отвечающего за обработку внутренних звонков экстеншн для обработки вызовов, уходящих в IAX-транк:

на узле `node1` :

```
exten => _5XXX,1,Dial(IAX2/iaxuser2:secret2@IP_node2/${EXTEN})
```

И аналогично для `node2`:

```
exten => _4XXX,1,Dial(IAX2/iaxuser1:secret1@IP_node1/${EXTEN})
```

Перезагрузите Asterisk:

```
sudo systemctl restart asterisk
```

Настройте клиенты PhonerLite на обеих сторонах для регистрации под соответствующим номером 4XXX или 5XXX. На обоих узлах Asterisk откройте командный интерфейс Asterisk (`sudo asterisk -vvvr`). Совершите звонок между клиентами. Убедитесь, что на обеих сторонах в консоли Asterisk отображаются служебные сообщения PJSIP, IAX об установлении соединения и информация о задействованных при обработке вызова экстеншенах. Если вышеуказанные условия выполнены и вызов успешно совершается, можно переходить ко 2 части работы.