Postgres PROFESSIONAL

Подключение и аутенти-фикация

Темы

Конфигурационные файлы для аутентификации

Основные настройки

Простые методы аутентификации

Аутентификация по паролю

Внешняя аутентификация и сопоставление имен

Задачи при подключении

Идентификация

определение имени пользователя БД

имя может отличаться от указанного (при внешней аутентификации)

Аутентификация

действительно ли пользователь тот, за кого себя выдает?

обычно требуется подтверждение (например, пароль)

Авторизация

имеет ли право данный пользователь подключаться к серверу?

частично пересекается с функционалом привилегий

Конфигурационные файлы

Расположение

обычно $PGDATA/*.conf

```
select name, setting
from   pg_settings
where  category = 'File Locations';
```

генерируются при создании кластера,
затем при необходимости могут изменяться администратором

Действия при изменении

файлы читается один раз при старте сервера,
поэтому при их изменении надо попросить сервер перечитать:

`$ pg_ctl reload` (или `kill -HUP`, или `select pg_reload_conf()`)

Основные настройки

pg_hba.conf (параметр hba_file)

Структура файла

- строка — набор полей, разделитель — пробел или табуляция
- пустые строки и текст после # игнорируются

Поля

- тип подключения (локальное, удаленное)
- имя базы данных
- имя пользователя
- адрес (имя или IP-адрес узла)
- метод аутентификации (пароль, сертификат и т. п.)
- необязательные параметры в виде *имя=значение* для аутентификации

Схема обработки

Записи просматриваются сверху вниз

Применяется первая запись, которой соответствуют параметры подключения (тип, база, пользователь и адрес)

- выполняется аутентификация и проверка привилегии CONNECT
- если результат отрицательный, доступ запрещается

Если ни одна запись не подошла, доступ запрещается

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD
# "local" is for Unix domain socket connections only
local   all             all                                     trust
# IPv4 local connections:
host    all             all             127.0.0.1/32            trust
# IPv6 local connections:
host    all             all             ::1/128                 trust
```

Тип подключения

local

локальное подключение через unix-domain socket

host

подключение по TCP/IP

(обычно требуется изменение параметра listen_addresses)

hostssl

шифрованное SSL-подключение по TCP/IP

(сервер должен быть собран с поддержкой SSL,
также требуется установить параметр ssl)

hostnossl

нешифрованное подключение по TCP/IP

Имя базы данных

all
: подключение к любой БД

sameuser
: БД, совпадающая по имени с пользователем

samerole
: БД, совпадающая по имени с ролью, в которую входит пользователь

БД
: БД с указанным именем (возможно, в кавычках)

имя[, *имя* ...]
: нескольких имен из вышеперечисленного

@файл
: прочитать имена баз данных из указанного файла

Адрес

all

любой IP-адрес

IP-адрес/длина_маски

указанный диапазон IP-адресов (например, 172.20.143.0/24)

или альтернативная форма в два поля (172.20.143.0 255.255.255.0)

samehost

IP-адрес сервера

samenet

любой IP-адрес из любой подсети, к которой подключен сервер

доменное_имя

IP-адрес, соответствующий указанному имени (например, domain.com)
допускается указание части имени, начиная с точки (.com)

Имя пользователя

all

любой пользователь

роль

пользователь (роль) с указанным именем (возможно, в кавычках)

+роль

пользователь, входящий в указанную роль

имя[, *имя* ...]

несколько имен из вышеперечисленного

@файл

прочитать имена пользователей из указанного файла

Простая аутентификация

trust

допустить без аутентификации

reject

отказать без аутентификации

Пример 1

```
# TYPE    DATABASE        USER            ADDRESS                 METHOD
hostnossl all             all             all                     reject
hostssl   sameuser        all             samenet                 trust
hostssl   pub             +reader         all                     trust
```

Пример 1 (объяснение)

```
# TYPE    DATABASE        USER            ADDRESS                 METHOD
hostnossl all             all             all                     reject
hostssl   sameuser        all             samenet                 trust
hostssl   pub             +reader         all                     trust
```

Запрещаются нешифрованные соединения

Разрешается доступ пользователям из своей подсети к одноименным базам данных

Разрешается доступ пользователям, входящим в роль reader, к базе данных pub

Аутентификация по паролю

password

запросить пароль открытым текстом
(опасно при удаленных не-SSL подключениях)

md5

запросить пароль в виде MD5-дайджеста

ldap *[параметры]*

проверка имени и пароля с помощью сервера LDAP

radius *[параметры]*

проверка имени и пароля с помощью сервера RADIUS

pam *[параметры]*

проверка имени и пароля с помощью подключаемого модуля

Пароли

Установить пароль пользователя

```
[create|alter] role ...
[encrypted|unencrypted] password 'пароль';
```

по умолчанию режим шифрования определяется параметром password_encryption

пользователю с пустым паролем будет отказано в доступе

Пароли хранятся в базе данных

```
select rolname, rolpassword from pg_authid;
```

encrypted — в зашифрованном виде (MD5)

unencrypted — в незашифрованном виде

Ввод пароля

Вручную

Установить переменную $PGPASSWORD

- неудобно при подключении к разным базам
- не рекомендуется из соображений безопасности

Файл с паролями

- расположение — ~/.pgpass (или $PGPASSFILE) на узле клиента
- строки в формате *узел*:*порт*:*база_данных*:*имя_пользователя*:*пароль*
- в качестве значения можно указать * (любое значение)
- строки просматриваются сверху вниз, выбирается первая подходящая
- файл должен иметь права доступа rw- --- ---

Внешняя аутентификация

ident *[map=...]*

получение имени пользователя у сервера клиента

peer *[map=...]*

запрос имени пользователя у ядра ОС (для локальных подключений)

cert *[map=...]*

аутентификация с использованием клиентского SSL-сертификата

gss *[map=... и другие параметры]*

аутентификация Kerberos по протоколу GSSAPI

sspi *[map=... и другие параметры]*

аутентификация Kerberos/NTLM для Windows

Соответствие имен

pg_ident.conf (параметр ident_file)

Структура файла

- строка — набор полей, разделитель — пробел или табуляция
- пустые строки и текст после # игнорируются

Поля

- название соответствия
  (указывается в параметре map в pg_hba.conf)
- внешнее имя
  (считается регулярным выражением, если начинается с косой черты)
- внутреннее имя пользователя БД

Пример 2

pg_hba.conf

```
# TYPE    DATABASE        USER        ADDRESS         METHOD
hostssl   sameuser        all         all             cert        map=m1
local     all             all                         md5
host      all             all         samehost        md5
```

pg_ident.conf

```
# MAPNAME         SYSTEM-USERNAME          PG-USERNAME
m1                /^(.*)@domain\.com$      \1
```

Пример 2 (объяснение)

pg_hba.conf

```
# TYPE    DATABASE        USER        ADDRESS         METHOD
hostssl   sameuser        all         all             cert        map=m1
local     all             all                         md5
host      all             all         samehost        md5
```

pg_ident.conf

```
# MAPNAME         SYSTEM-USERNAME          PG-USERNAME
m1                /^(.*)@domain\.com$      \1
```

SSL-соединения аутентифицируются по клиентскому сертификату

- предполагается, что имя в сертификате содержит название домена

Локальные соединения аутентифицируются по паролю

- пароль передается в зашифрованном виде

Итоги

Узнали про настройки в конфигурационных файлах

Рассмотрели основные настройки в pg_hba.conf

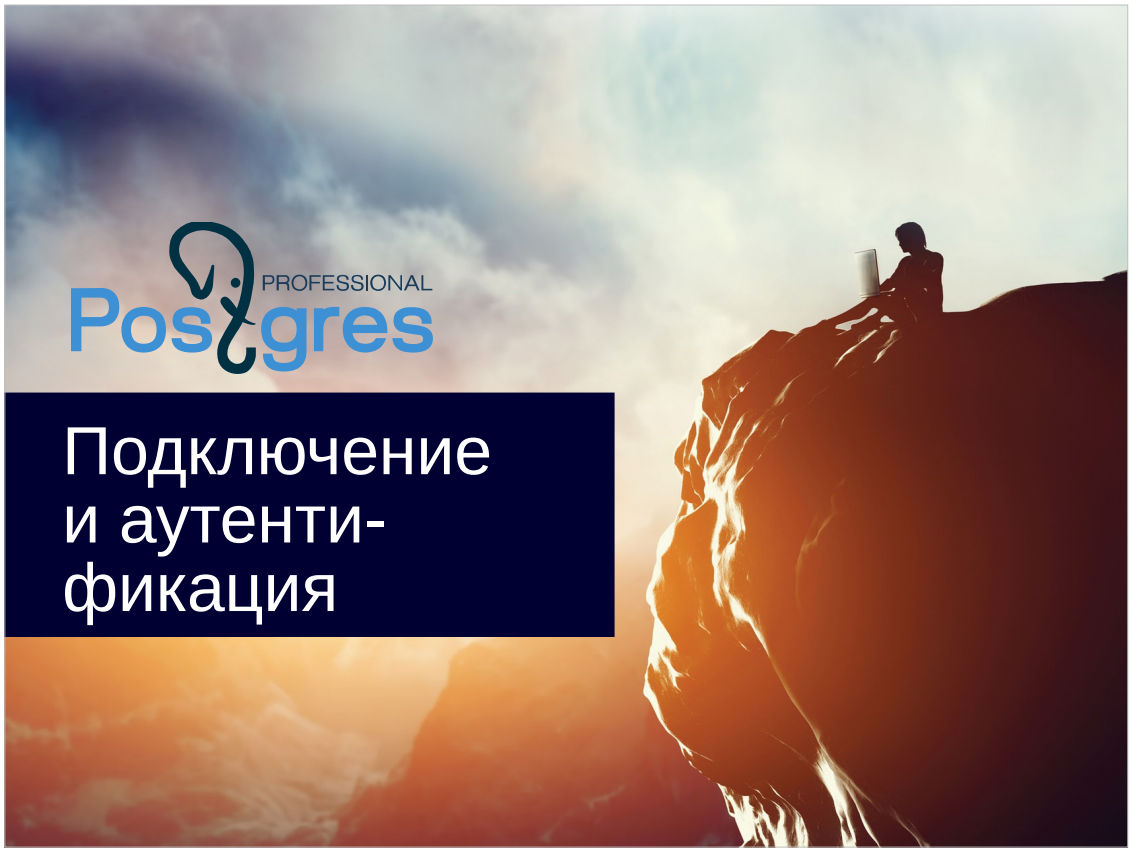
Познакомились с различными методами аутентификации

Узнали, как использовать пароль для аутентификации

Рассмотрели внешнюю аутентификацию и функционал сопоставления имен

Практика

1. Измените конфигурационные файлы (предварительно сохранив оригиналы) таким образом, чтобы:
   - безусловно разрешить локальное соединение пользователю postgres
   - разрешить сетевые подключения всем пользователям к любым базам данных с аутентификацией по паролю с шифрованием MD5
2. Создайте пользователя enc с зашифрованным паролем и пользователя unenc с незашифрованным.
3. Подключитесь к БД postgres под пользователем enc
4. Узнайте пароль пользователя unenc из таблицы pg_authid. Получилось? Почему?
5. Выйдите из psql и подключитесь как postgres. Посмотрите пароли пользователей enc и unenc.
6. Восстановите исходные конфигурационные файлы.



Обратная связь
Отзывы, замечания и предложения направляйте по адресу:
edu@postgrespro.ru

Темы

Конфигурационные файлы для аутентификации

Основные настройки

Простые методы аутентификации

Аутентификация по паролю

Внешняя аутентификация и сопоставление имен

Задачи при подключении

Идентификация

- определение имени пользователя БД
- имя может отличаться от указанного (при внешней аутентификации)

Аутентификация

- действительно ли пользователь тот, за кого себя выдает?
- обычно требуется подтверждение (например, пароль)

Авторизация

- имеет ли право данный пользователь подключаться к серверу?
- частично пересекается с функционалом привилегий

При подключении клиента, сервер должен выполнить несколько задач.

Во-первых, идентифицировать пользователя, то есть определить его имя. Для этого пользователь представляется, но указанное имя может отличаться от имени пользователя БД (например, если пользователь представляется своим именем в ОС).

Во-вторых, аутентифицировать пользователя, то есть проверить, что он действительно тот, за кого себя выдает. Простой пример — ввод пароля.

В-третьих, авторизовать пользователя, то есть определить, имеет ли он право подключения (эта задача частично пересекается с функционалом привилегий).

Иногда все три задачи называют общим словом «аутентификация». PostgreSQL позволяет гибко настроить этот процесс.

До сих пор мы подключались к серверу, никак не подтверждая свое имя. Дело в том, что настройки по умолчанию допускают любые локальные подключения без аутентификации.

Конфигурационные файлы

Расположение

обычно $PGDATA/*.conf

```
select name, setting
from   pg_settings
where  category = 'File Locations';
```

генерируются при создании кластера,
затем при необходимости могут изменяться администратором

Действия при изменении

файлы читается один раз при старте сервера,
поэтому при их изменении надо попросить сервер перечитать:

`$ pg_ctl reload` (или `kill -HUP`, или `select pg_reload_conf()`)

Настройки, связанные с подключением и аутентификацией, хранятся не в кластере, а в отдельных конфигурационных файлах. Обычно они имеют расширение conf и расположены в каталоге $PGDATA, хотя расположение можно изменить с помощью соответствующих параметров сервера.

Файлы генерируются при создании кластера и могут затем быть изменены администратором.

При изменении файлов конфигурации надо дать команду серверу перечитать настройки, так как обычно файлы читаются только один раз при старте сервера.

Основные настройки

pg_hba.conf (параметр hba_file)

Структура файла

- строка — набор полей, разделитель — пробел или табуляция
- пустые строки и текст после # игнорируются

Поля

- тип подключения (локальное, удаленное)
- имя базы данных
- имя пользователя
- адрес (имя или IP-адрес узла)
- метод аутентификации (пароль, сертификат и т. п.)
- необязательные параметры в виде *имя=значение* для аутентификации

Основной конфигурационный файл — pg_hba.conf (от «host-based authentication»).

Он состоит из строк, каждая строка является отдельной записью. Пустые строки и комментарии (все после символа #) игнорируются. Строка состоит из полей, разделенных пробелами или табуляциями.

Количество полей может различаться в зависимости от их содержимого, но общий список приведен на слайде.

Схема обработки

Записи просматриваются сверху вниз

Применяется первая запись, которой соответствуют параметры подключения (тип, база, пользователь и адрес)

- выполняется аутентификация и проверка привилегии CONNECT
- если результат отрицательный, доступ запрещается

Если ни одна запись не подошла, доступ запрещается

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD
# "local" is for Unix domain socket connections only
local   all             all                                     trust
# IPv4 local connections:
host    all             all             127.0.0.1/32            trust
# IPv6 local connections:
host    all             all             ::1/128                 trust
```


Конфигурационный файл обрабатывается сверху вниз. Для каждой строки определяется, подходит ли она к запрашиваемому клиентом подключению (по соответствию типа подключения, имени БД, имени пользователя и IP-адресу). Если подходит, то выполняется аутентификация указанным в строке методом. Если результат успешен, то подключение разрешается, иначе — запрещается (другие строки при этом уже не рассматриваются).

Если ни одна из строк не подошла, то доступ запрещается.

Таким образом, записи в файле должны идти сверху вниз от частного к общему.

Внизу слайда приведен фрагмент файла по умолчанию (убраны только комментарии). В нем видно три строки. Первая относится к локальным не-TCP подключениям (local) для любых баз (all) и пользователей (all). Вторая относится к удаленным подключениям (host) с адреса 127.0.0.1 (то есть localhost), третья — то же самое, но для IPv6.

Можно сделать вывод о том, что по умолчанию PostgreSQL допускает только локальные соединения (как сетевые, так и нет).

Далее возможные значения полей рассматриваются более подробно.

Тип подключения

local

локальное подключение через unix-domain socket

host

подключение по TCP/IP

(обычно требуется изменение параметра listen_addresses)

hostssl

шифрованное SSL-подключение по TCP/IP

(сервер должен быть собран с поддержкой SSL,
также требуется установить параметр ssl)

hostnossl

нешифрованное подключение по TCP/IP

В поле типа подключения можно задать одно из значений, перечисленных ниже.

Слово «local» — соответствует локальному подключению через доменный сокет (без использования сетевого подключения).

Слово «host» — соответствует любому подключению по TCP/IP. Поскольку по умолчанию PostgreSQL слушает соединения только с локального адреса (localhost), скорее всего потребуется задать другой адрес с помощью параметра сервера listen_address.

Слово «hostssl» — соответствует только шифрованному SSL-подключению по TCP/IP. Для таких соединений сервер должен быть собран с поддержкой SSL. Кроме того, требуется установить параметр ssl.

Слово «hostnossl» — соответствует только нешифрованному подключению по TCP/IP.

Имя базы данных

all
: подключение к любой БД

sameuser
: БД, совпадающая по имени с пользователем

samerole
: БД, совпадающая по имени с ролью, в которую входит пользователь

БД
: БД с указанным именем (возможно, в кавычках)

имя[,*имя*...]
: нескольких имен из вышеперечисленного

@файл
: прочитать имена баз данных из указанного файла

В поле базы данных можно задать одно из значений, перечисленных ниже, или несколько таких значений через запятую.

Слово «all» — соответствует любой базе данных.

Слово sameuser — соответствует базе данных, совпадающей по имени с пользователем.

Слово samerole — соответствует базе данных, совпадающей по имени с какой-либо ролью, в которую входит пользователь (в том числе совпадающей по имени с самим пользователем, поскольку пользователь — та же роль).

Наконец, можно указать имя конкретной базы данных.

Вместо перечисления имен можно сослаться на файл с помощью @. В файле имена могут быть разделены запятыми, пробелами, табуляциями или переводами строк. Допускаются вложенные подключения файлов (@) и комментарии (#).

Адрес

all
: любой IP-адрес

IP-адрес/длина_маски
: указанный диапазон IP-адресов (например, 172.20.143.0/24)

: или альтернативная форма в два поля (172.20.143.0 255.255.255.0)

samehost
: IP-адрес сервера

samenet
: любой IP-адрес из любой подсети, к которой подключен сервер

доменное_имя
: IP-адрес, соответствующий указанному имени (например, domain.com)
: допускается указание части имени, начиная с точки (.com)

В поле адреса может быть указано одно из следующих значений.

Слово «all» — соответствует любому IP-адресу клиента.

IP-адрес с указанием длины маски подсети (CIDR) — определяет диапазон допустимых IP-адресов. Альтернативно можно записать отдельно IP-адрес и в следующем поле маску подсети. Также поддерживаются IP-адреса в нотации IPv6.

Слово «samehost» — соответствует IP-адресу сервера (фактически, аналог 127.0.0.1 для систем, где такой адрес не разрешен).

Слово «samenet» — соответствует любому IP-адресу из любой подсети, к которой подключен сервер.

Наконец, адрес можно указать доменным именем (или частью доменного имени, начиная с точки). PostgreSQL определит принадлежность IP-адреса клиента указанному домену: для этого сначала по IP-адресу определяется доменное имя (reverse lookup), а затем проверяется, что такому домену действительно соответствует исходный IP-адрес (forward lookup). Таким образом проверяется соответствие владельца сети и владельца доменного имени для отсечения скомпрометированных адресов (см. FCrDNS).

Имя пользователя

all

любой пользователь

роль

пользователь (роль) с указанным именем (возможно, в кавычках)

+роль

пользователь, входящий в указанную роль

имя[,*имя*...]

несколько имен из вышеперечисленного

@файл

прочитать имена пользователей из указанного файла

В поле имени пользователя можно указать одно из значений, перечисленных ниже, или несколько таких значений через запятую.

Слово «all» — соответствует любому IP-адресу клиента.

Имя роли — соответствует пользователю (или роли, что то же самое) с указанным именем. Если перед именем роли стоит знак +, то имя соответствует любому пользователю, входящему в указанную роль.

Вместо перечисления имен можно сослаться на файл с помощью @. В файле имена могут быть разделены запятыми, пробелами, табуляциями или переводами строк. Допускаются вложенные подключения файлов (@) и комментарии (#).

Простая аутентификация

trust

допустить без аутентификации

reject

отказать без аутентификации

В поле метода аутентификации можно указать различные методы. Для начала познакомимся с двумя самыми простыми, а остальные рассмотрим ниже.

Метод «trust» безусловно доверяет пользователю и не выполняет аутентификацию. В реальной жизни имеет смысл применять разве что для локальных соединений.

Метод «reject» безусловно отказывает в доступе. Можно использовать, чтобы отсечь любые соединения определенного типа или с определенных адресов (например, запретить нешифрованные соединения).

Пример 1

```
# TYPE    DATABASE        USER            ADDRESS                 METHOD
hostnossl all             all             all                     reject
hostssl   sameuser        all             samenet                 trust
hostssl   pub             +reader         all                     trust
```


Пример настройки — объясняется на следующем слайде.

Пример 1 (объяснение)

```
# TYPE    DATABASE        USER            ADDRESS                 METHOD
hostnossl all             all             all                     reject
hostssl   sameuser        all             samenet                 trust
hostssl   pub             +reader         all                     trust
```

Запрещаются нешифрованные соединения

Разрешается доступ пользователям из своей подсети к одноименным базам данных

Разрешается доступ пользователям, входящим в роль reader, к базе данных pub

Первая строка запрещает нешифрованные соединения.

Вторая строка разрешает доступ пользователям к базам данных, совпадающих по имени, но только из подсетей, к которым подключен сервер.

Третья строка разрешает доступ пользователям, входящим в роль reader, к базе данных pub.

Аутентификация по паролю

password

запросить пароль открытым текстом
(опасно при удаленных не-SSL подключениях)

md5

запросить пароль в виде MD5-дайджеста

ldap *[параметры]*

проверка имени и пароля с помощью сервера LDAP

radius *[параметры]*

проверка имени и пароля с помощью сервера RADIUS

pam *[параметры]*

проверка имени и пароля с помощью подключаемого модуля

Приведенные здесь методы аутентификации запрашивают у пользователя пароль и каким-либо образом проверяют его.

Метод «password» сравнивает введенный незашифрованный пароль с паролем, хранящимся в базе. Имеет смысл применять только в сочетании с шифрованным соединением.

Метод «md5» сравнивает MD5-дайджест пароля с MD5-дайджестом пароля, хранящегося в базе. Алгоритм MD5 преобразует введенный пароль в последовательность символов, по которой практически невозможно восстановить исходный пароль (хотя алгоритм MD5 не считается криптостойким). Однако это не защищает от возможности подслушать MD5-дайджест и использовать его для аутентификации, так что и этот метод не следует использовать в ненадежных сетях.

Методы «ldap», «radius» и «pam» служат для выполнения внешней проверки имени и пароля с помощью сервера LDAP, сервера RADIUS или подключаемого модуля аутентификации PAM (Pluggable Authentication Module). Эти методы требуют указания дополнительных специфичных параметров. Подробно они не рассматриваются.

Пароли

Установить пароль пользователя

```
[create|alter] role ...
[encrypted|unencrypted] password 'пароль';
```

по умолчанию режим шифрования определяется параметром password_encryption

пользователю с пустым паролем будет отказано в доступе

Пароли хранятся в базе данных

```
select rolname, rolpassword from pg_authid;
```

encrypted — в зашифрованном виде (MD5)

unencrypted — в незашифрованном виде

До сих пор мы создавали роли без указания пароля. Если установить метод аутентификации по паролю, таким пользователям будет отказано в доступе.

Пароль хранится в базе данных в таблице pg_authid.

Чтобы установить пароль, надо указать его (при создании роли в команде create role или впоследствии в alter role). При этом надо определить, будет ли пароль храниться в открытом виде (unencrypted) или в зашифрованном (encrypted; используется дайджест MD5). Если не указать ни encrypted, ни unencrypted, то значение будет взято из параметра password_encryption.

Также можно указать время действия пароля.

Ввод пароля

Вручную

Установить переменную $PGPASSWORD

- неудобно при подключении к разным базам
- не рекомендуется из соображений безопасности

Файл с паролями

- расположение — ~/.pgpass (или $PGPASSFILE) на узле клиента
- строки в формате *узел*:*порт*:*база_данных*:*имя_пользователя*:*пароль*
- в качестве значения можно указать * (любое значение)
- строки просматриваются сверху вниз, выбирается первая подходящая
- файл должен иметь права доступа rw- --- ---

Пользователь может вводить пароль вручную, а может автоматизировать ввод. Для этого есть две возможности.

Во-первых, можно задать пароль в переменной окружения $PGPASSWORD. Однако это неудобно, если приходится подключаться к нескольким базам, и не рекомендуется из соображений безопасности.

Во-вторых, можно задать пароли в файле ~/.pgpass (его расположение можно изменить, задав переменную окружения $PGPASSFILE). К файлу должен иметь доступ только владелец, иначе PostgreSQL проигнорирует его.

Внешняя аутентификация

ident *[map=...]*

получение имени пользователя у сервера клиента

peer *[map=...]*

запрос имени пользователя у ядра ОС (для локальных подключений)

cert *[map=...]*

аутентификация с использованием клиентского SSL-сертификата

gss *[map=... и другие параметры]*

аутентификация Kerberos по протоколу GSSAPI

sspi *[map=... и другие параметры]*

аутентификация Kerberos/NTLM для Windows

Клиент подключается к базе, указывая имя пользователя СУБД.
В отличие от рассмотренных ранее методов (которые выполняют аутентификацию указанного имени), в данном случае и идентификация, и аутентификация происходит вне СУБД. В результате успешной аутентификации PostgreSQL получает:

1. имя, указанное при подключении (внутреннее имя СУБД),

2. имя, идентифицированное внешней системой (внешнее имя).

Поэтому все перечисленные методы позволяют указать как минимум один дополнительный параметр map. Он определяет правила сопоставления внутренних и внешних имен (подробнее об этом на следующем слайде).

Метод ident запрашивает имя пользователя для соединения TCP у клиентского сервера, используя протокол идентификации (RFC 1413).

Метод peer запрашивает имя пользователя у ядра ОС. Это аналог ident для локальных соединений (если для локального соединения указать ident, то будет использован peer).

Метод cert использует аутентификацию на основе клиентского сертификата и предназначен только для SSL-соединений.

Метод gss использует аутентификацию Kerberos по протоколу GSSAPI (RFC 1964). Поддерживается автоматическая аутентификация (single sign-on).

Метод sspi использует аутентификацию Kerberos или NTLM для систем на Windows. Поддерживается автоматическая аутентификация.

Соответствие имен

pg_ident.conf (параметр ident_file)

Структура файла

- строка — набор полей, разделитель — пробел или табуляция
- пустые строки и текст после # игнорируются

Поля

- название соответствия
  (указывается в параметре map в pg_hba.conf)
- внешнее имя
  (считается регулярным выражением, если начинается с косой черты)
- внутреннее имя пользователя БД

Правила сопоставления имен определяются в отдельном файле pg_ident.conf. Он имеет такую же структуру, как и pg_hba.conf. Записи состоят из трех полей: название соответствия, внешнее имя, внутреннее имя.

Название соответствия необходимо, чтобы разграничивать разные правила сопоставления внутри одного файла pg_ident.conf (которые могут потребоваться для разных записей в pg_hba.conf).

Внешнее имя должно совпадать с именем, возвращаемым внешней системой аутентификации или указанным в сертификате. Если это поле начинается с косой черты, то его значение считается регулярным выражением. Это позволяет обработать ситуации, когда внешнее и внутреннее имена отличаются только префиксами или суффиксами.

Внутреннее имя должно совпадать с именем пользователя базы данных.

Запись, сопоставляющая внутреннее и внешнее имена, означает, что данному внешнему пользователю разрешено подключаться к СУБД под данным внутренним пользователем (конечно, при условии успешной аутентификации).

Пример 2

pg_hba.conf

```
# TYPE    DATABASE       USER       ADDRESS       METHOD
hostssl   sameuser       all        all           cert       map=m1
local     all            all                      md5
host      all            all        samehost      md5
```

pg_ident.conf

```
# MAPNAME       SYSTEM-USERNAME          PG-USERNAME
m1              /^(.*)@domain\.com$      \1
```


Пример настройки — объясняется на следующем слайде.

Пример 2 (объяснение)

pg_hba.conf

```
# TYPE    DATABASE       USER       ADDRESS       METHOD
hostssl   sameuser       all        all           cert       map=m1
local     all            all                      md5
host      all            all        samehost      md5
```

pg_ident.conf

```
# MAPNAME       SYSTEM-USERNAME          PG-USERNAME
m1              /^(.*)@domain\.com$      \1
```

SSL-соединения аутентифицируются по клиентскому сертификату

- предполагается, что имя в сертификате содержит название домена

Локальные соединения аутентифицируются по паролю

- пароль передается в зашифрованном виде

Первая строка pg_hba.conf устанавливает метод аутентификации по клиентскому сертификату, причем к базам данных, совпадающих с пользователем по имени. Используется сопоставление имен m1.

Сопоставление использует регулярное выражение, «отрезающее» от имени в сертификате (CN) окончание «@domain.com». Поле внутреннего имени ссылается на выделенное скобками значение с помощью \1.

Вторая и третья строки pg_hba.conf устанавливают для любых локальных (сетевых и несетевых) соединений метод аутентификации по паролю. Пароль передается в зашифрованном виде (MD5). При этом сетевые подключения допускаются только с самого сервера.

Итоги

Узнали про настройки в конфигурационных файлах

Рассмотрели основные настройки в pg_hba.conf

Познакомились с различными методами аутентификации

Узнали, как использовать пароль для аутентификации

Рассмотрели внешнюю аутентификацию и функционал сопоставления имен

Практика

1. Измените конфигурационные файлы (предварительно сохранив оригиналы) таким образом, чтобы:
   - безусловно разрешить локальное соединение пользователю postgres
   - разрешить сетевые подключения всем пользователям к любым базам данных с аутентификацией по паролю с шифрованием MD5
2. Создайте пользователя enc с зашифрованным паролем и пользователя unenc с незашифрованным.
3. Подключитесь к БД postgres под пользователем enc
4. Узнайте пароль пользователя unenc из таблицы pg_authid. Получилось? Почему?
5. Выйдите из psql и подключитесь как postgres. Посмотрите пароли пользователей enc и unenc.
6. Восстановите исходные конфигурационные файлы.

Решение

```
$ cd $PGDATA
$ cp pg_hba.conf pg_hba.conf.default
-- редактируем файл pg_hba.conf в любом редакторе
$ cat pg_hba.conf
local  all  postgres       trust
host   all  all       all  md5
$ pg_ctl reload

$ psql
# create user enc encrypted password 'enc';
# create user unenc unencrypted password 'unenc';

$ psql postgres enc -h localhost
Password for user enc: enc

> select rolpassword from pg_authid where rolname='unenc';
-- permission denied
> \dp pg_catalog.pg_authid

$ psql
# select rolname, rolpassword
from pg_authid where rolname like '%enc';
 enc      | md59ad9487992ed433fd63349ea544f41d7
 unenc    | unenc

$ cp pg_hba.conf.default pg_hba.conf; pg_ctl reload
```