EDB
Postgres® for the AI Generation

Obfuscação de `sslpassword` com hook `PQsetSSLKeyPassHook_OpenSSL`

William Ivanski
Senior Principal Support Engineer
EnterpriseDB
08 de novembro de 2024

Segurança na Autenticação

Importância da Segurança

- Diferentes setores utilizam PostgreSQL para armazenar **dados confidenciais**:
  - Finanças
  - Saúde
  - Propriedade Intelectual, etc

- Segurança inadequada ou falta de segurança implica em **riscos graves**, por exemplo:
  - Acesso não autorizado aos dados
  - Roubo, exposição e sequestro de dados confidenciais
  - Roubo de identidade
  - Fraudes financeiras, industriais e de marketing

Segurança na Autenticação

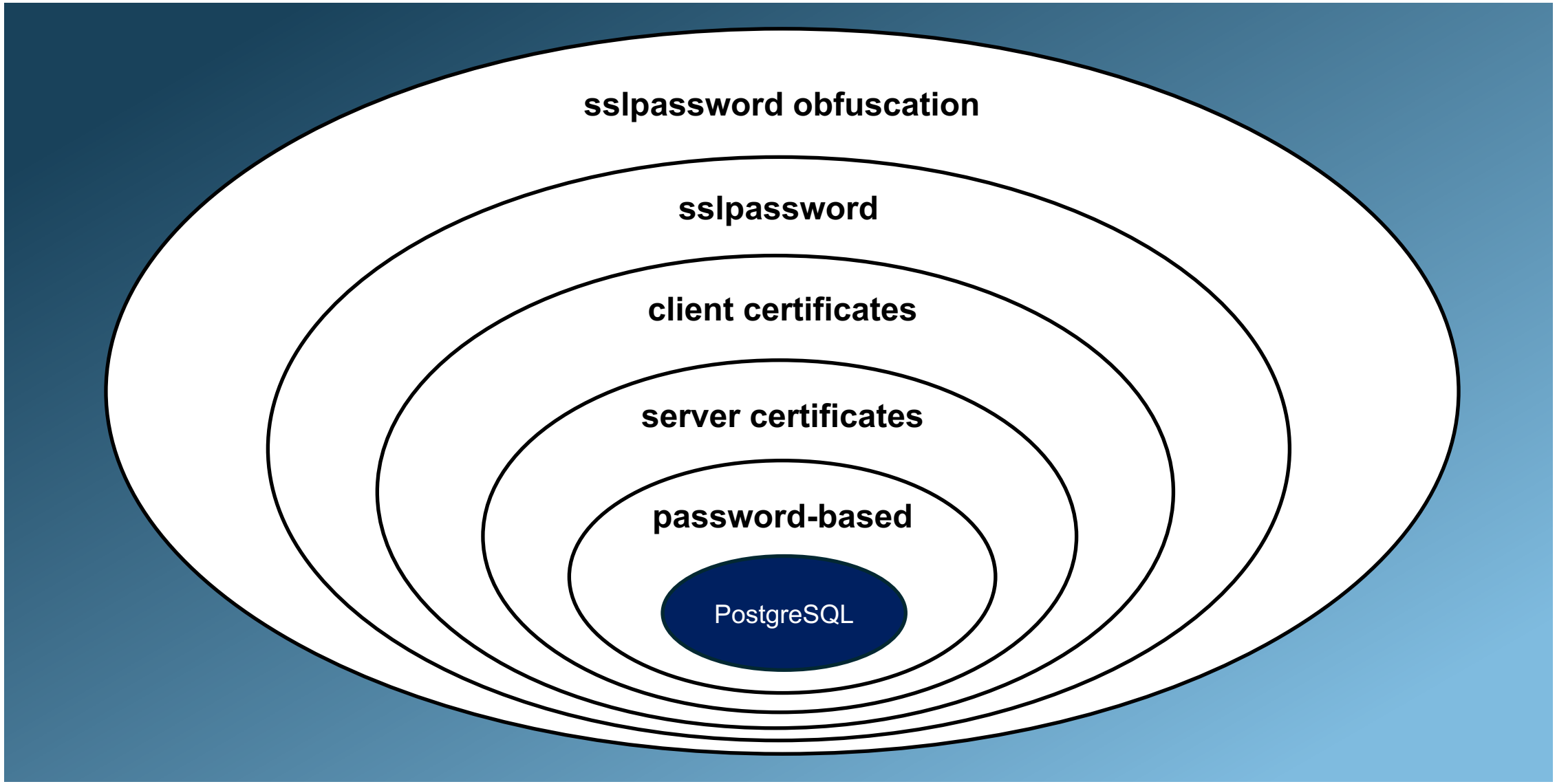
- Autenticação é uma das barreiras na segurança dos dados

- Requisitos para fazer “log in” no sistema, por exemplo:
  - Endereço de origem
  - Usuário
  - Banco
  - Criptografia
  - Método de autenticação

Tipos de Autenticação

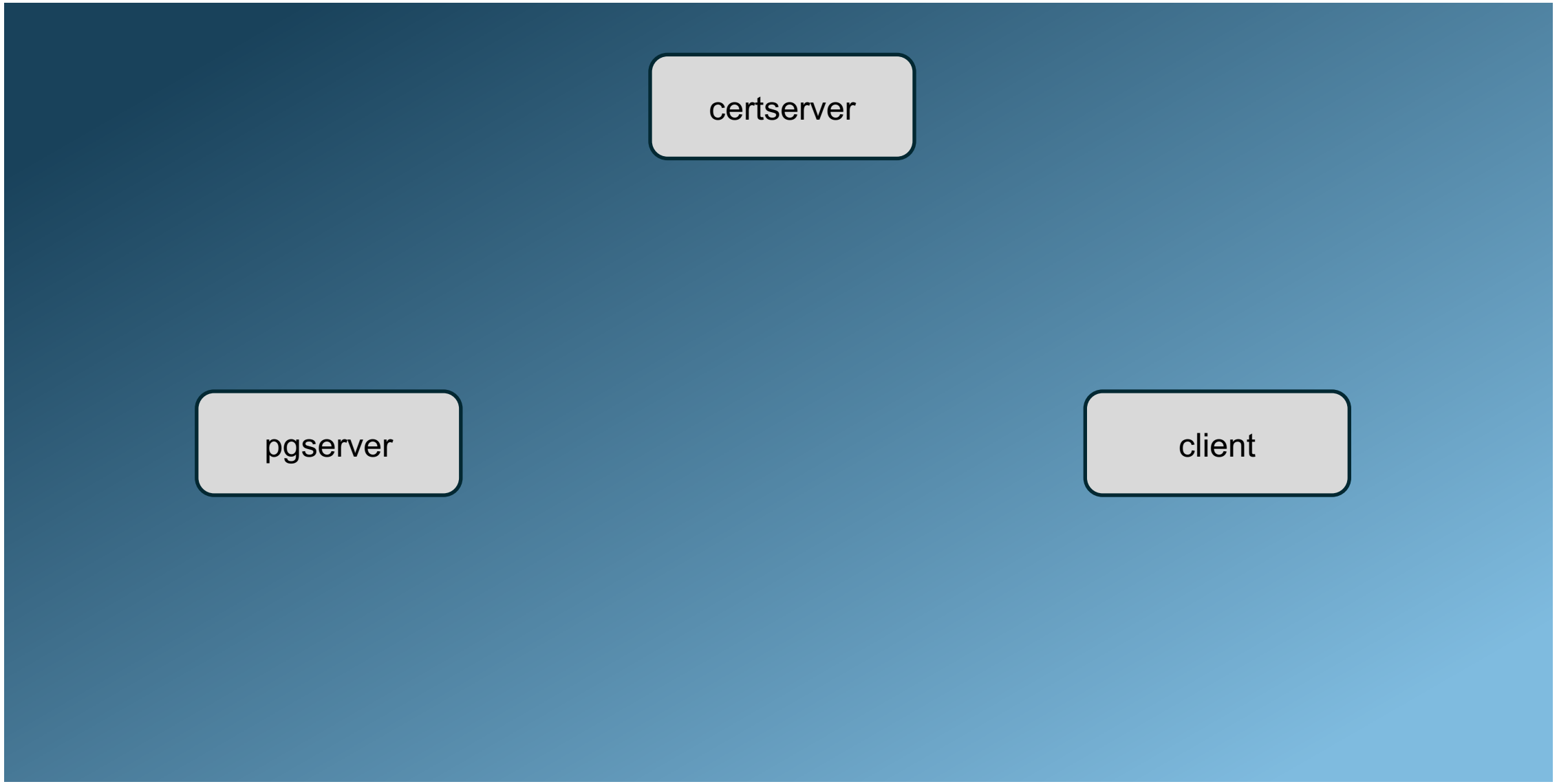
- Autenticação externa
  - GSSAPI, SSPI, LDAP, RADIUS
- Autenticação de sistema operacional
  - BSD, PAM, Peer, Ident
- Autenticação interna do PostgreSQL
  - Trust / Reject
  - Password-based: password, md5, SCRAM
  - SSL / TLS

SEGURANÇA NA AUTENTICAÇÃO INTERNA (SIMPLIFICADA)

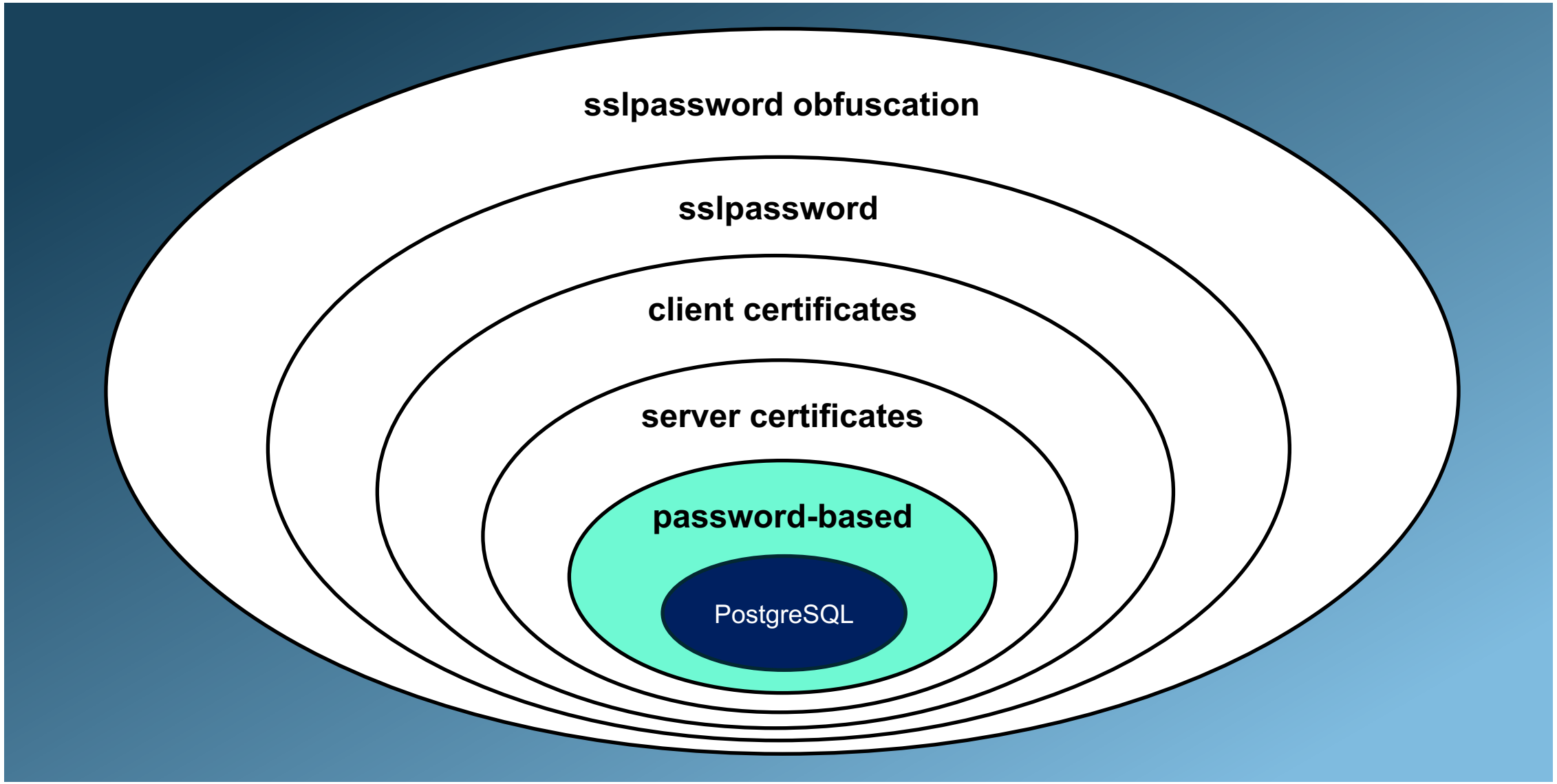


Autenticação Interna na Prática

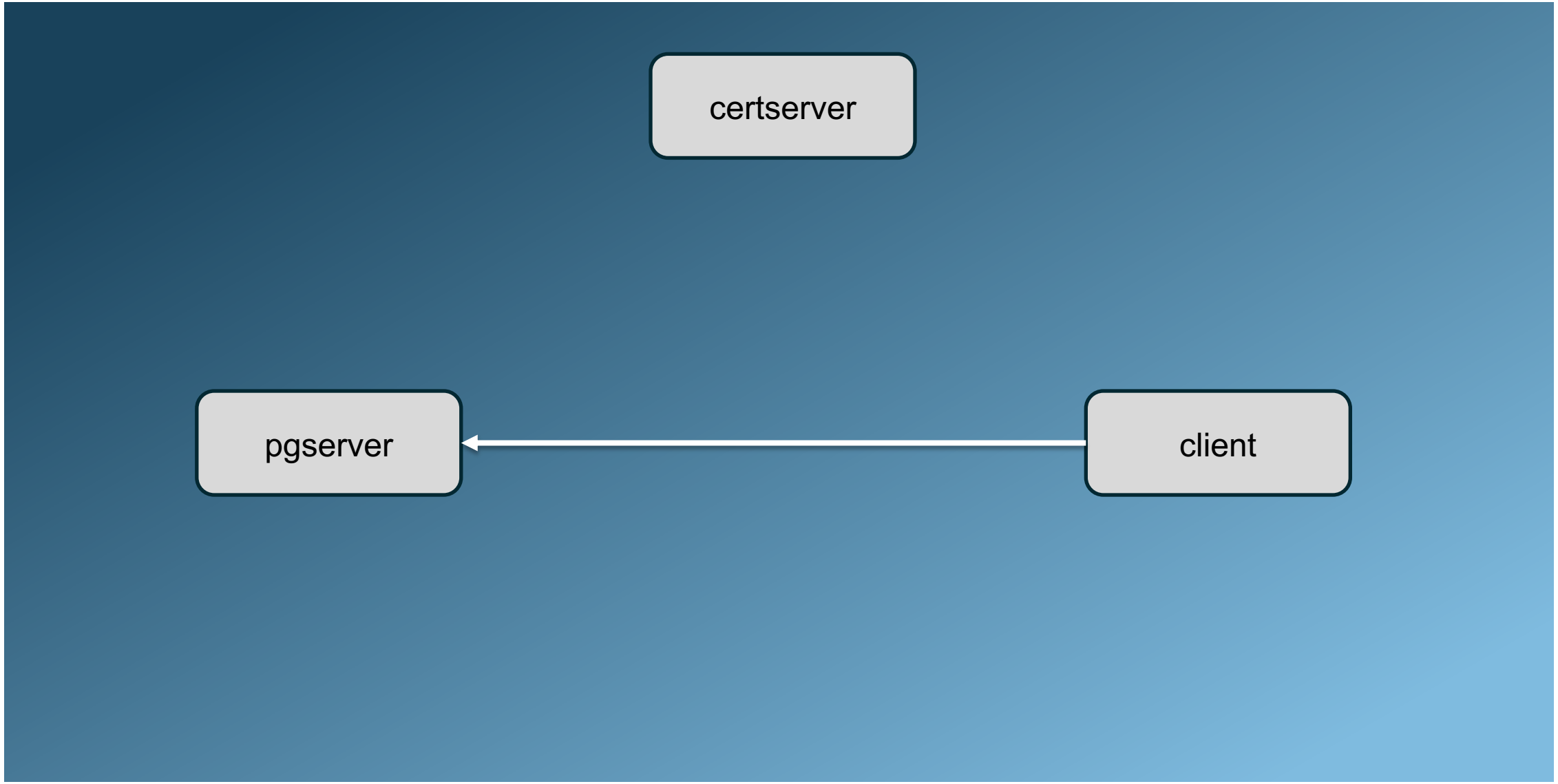
0- SERVIDORES UTILIZADOS



1- PASSWORD-BASED



1- PASSWORD-BASED



1- Password-based

pg_hba.conf:

```
host myappdb myappuser 172.18.0.22/32 scram-sha-256
```

```
psql -c “SELECT pg_reload_conf()”
```

1- Password-based

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser"
Password for user myappuser:
psql (17.0)
Type "help" for help.

myappdb=>
```

1- Password-based

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
password='ohsei7Ae'"
psql (17.0)
Type "help" for help.


myappdb=>
```

1- Password-based

```
[root@client ~]# export PGPASSWORD=ohsei7Ae

[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser"
psql (17.0)
Type "help" for help.

myappdb=>
```

1- Password-based

```
[root@client ~]# cat > ~/.pgpass << EOF
pgserver:5432:myappdb:myappuser:ohsei7Ae
EOF
[root@client ~]# chmod 600 ~/.pgpass

[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser"
psql (17.0)
Type "help" for help.

myappdb=>
```

SSL / TLS

SSL / TLS

- SSL: Secure Sockets Layer
  - Implementação original pela Netscape, agora obsoleto
- TLS: Transport Layer Security
  - Evolução do SSL
  - Versões obsoletas: 1.0 e 1.1
  - Versões recomendadas: 1.2 e 1.3
- Criptografia a nível de socket TCP
  - HTTPS
  - SSH
  - etc

SSL / TLS

- Criptografia assimétrica:
  - Par de chaves pública / privada
  - Chave pública para criptografar
  - Chave privada para descriptografar

- Criptografia simétrica:
  - Usa a mesma chave para criptografar e descriptografar

Certificados SSL

- Um certificado SSL contém:
  - Chave pública
  - Informações sobre a identidade
  - Autoridade certificadora (CA)
  - Entre outras informações

- Chave privada deve ser mantida protegida e nunca compartilhada!

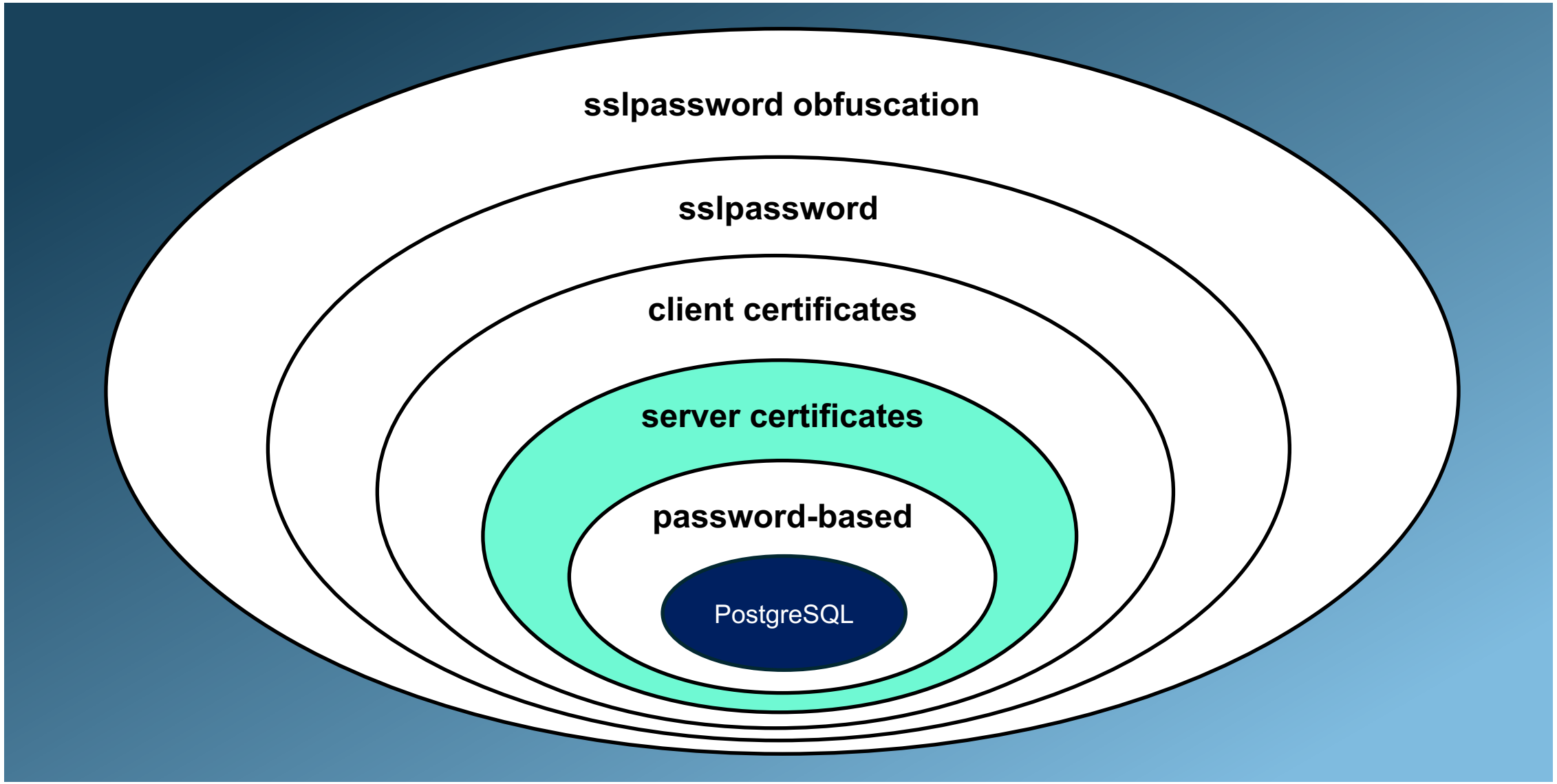
Comunicação criptografada utilizando certificados SSL

- Usa ambas criptografias assimétrica e simétrica

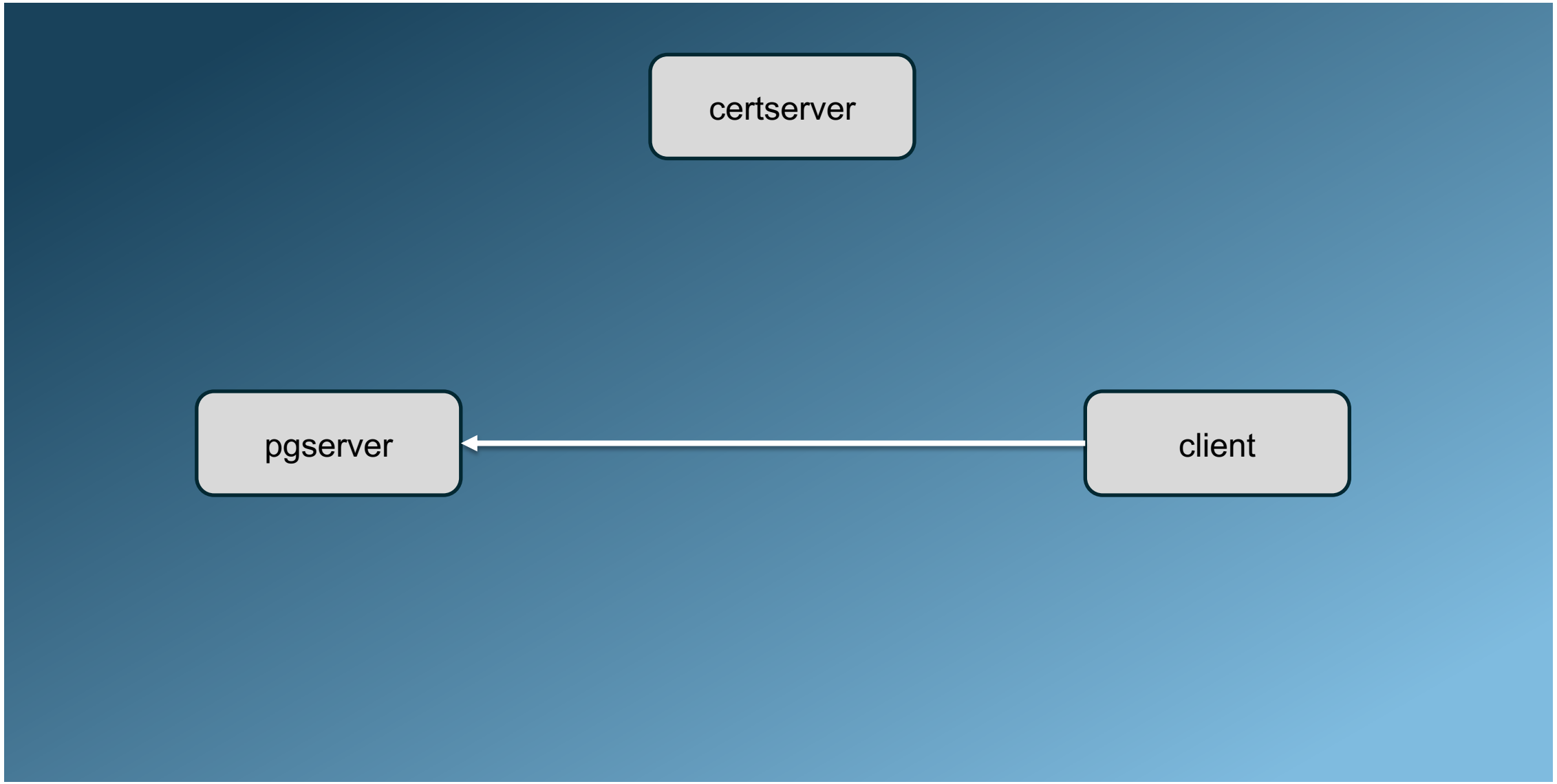
- Assimétrica:
  - TLS handshake
  - Utiliza o par de chaves
  - Criar e criptografar a nova chave simétrica ou token que será usado para aquele canal de comunicação

- Simétrica:
  - Uma vez estabelecida a chave simétrica durante o TLS handshake, é usada para criptografar toda a comunicação em ambas as direções

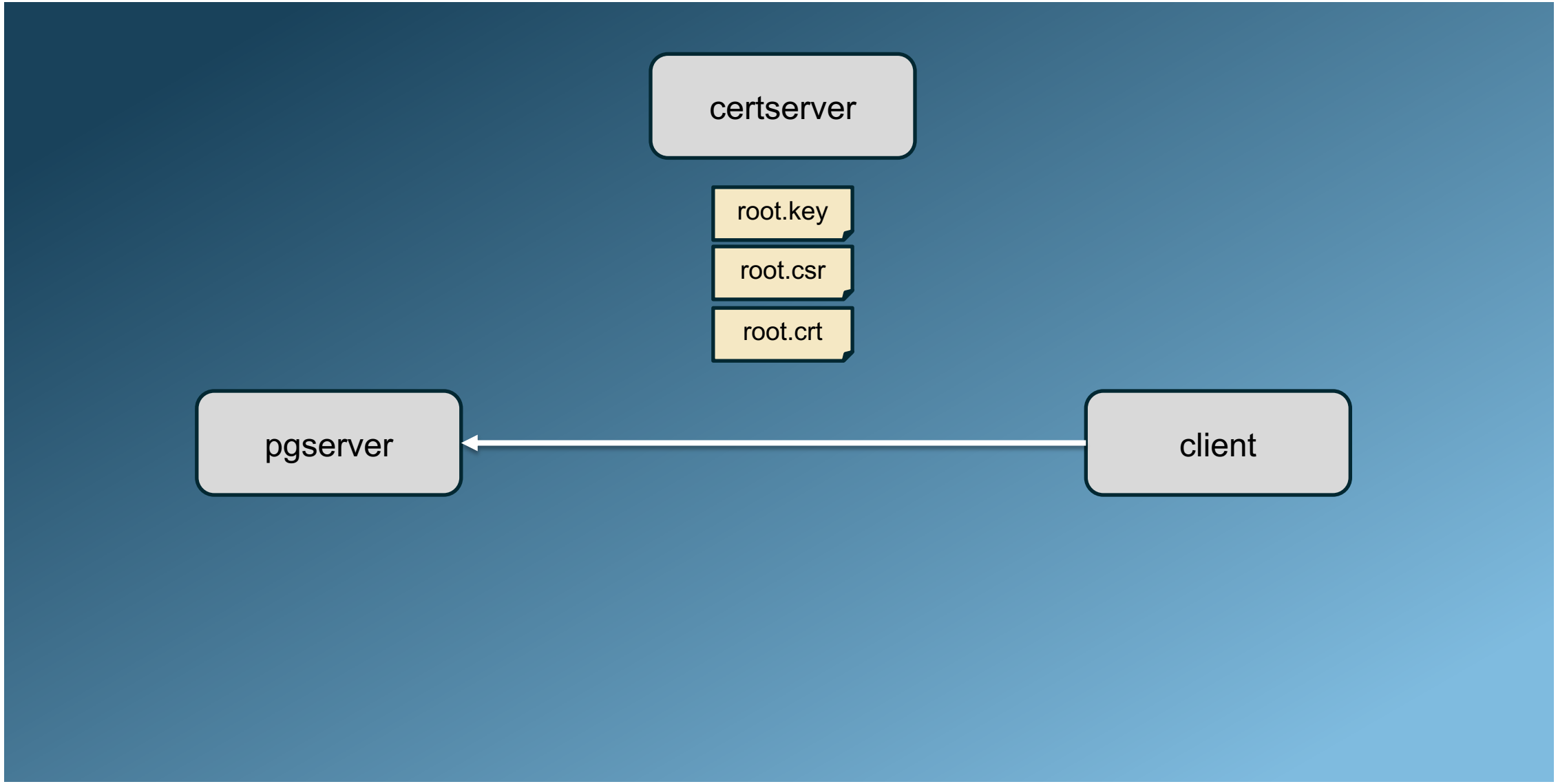
2- SERVER CERTIFICATES



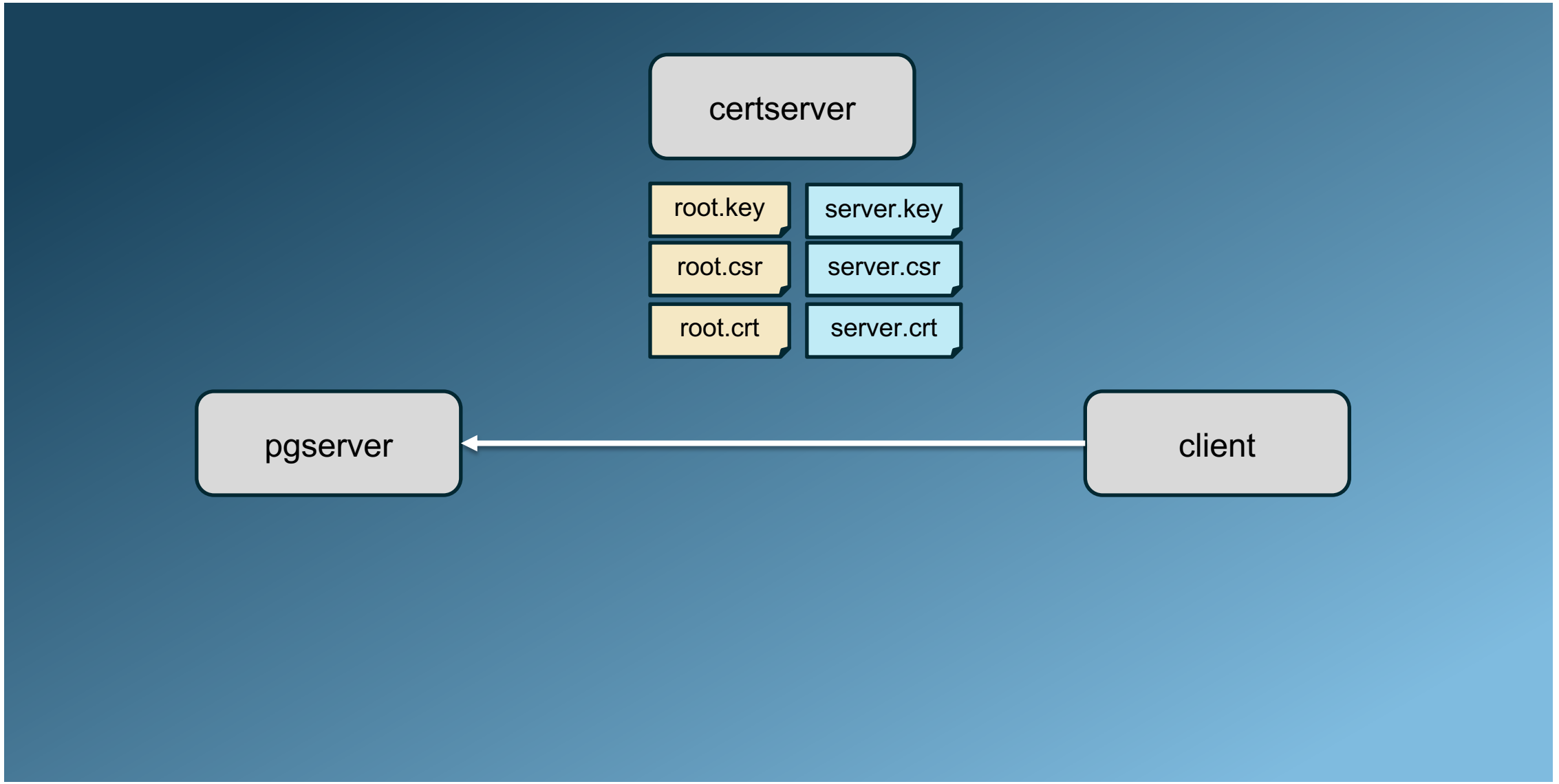
2- SERVER CERTIFICATES



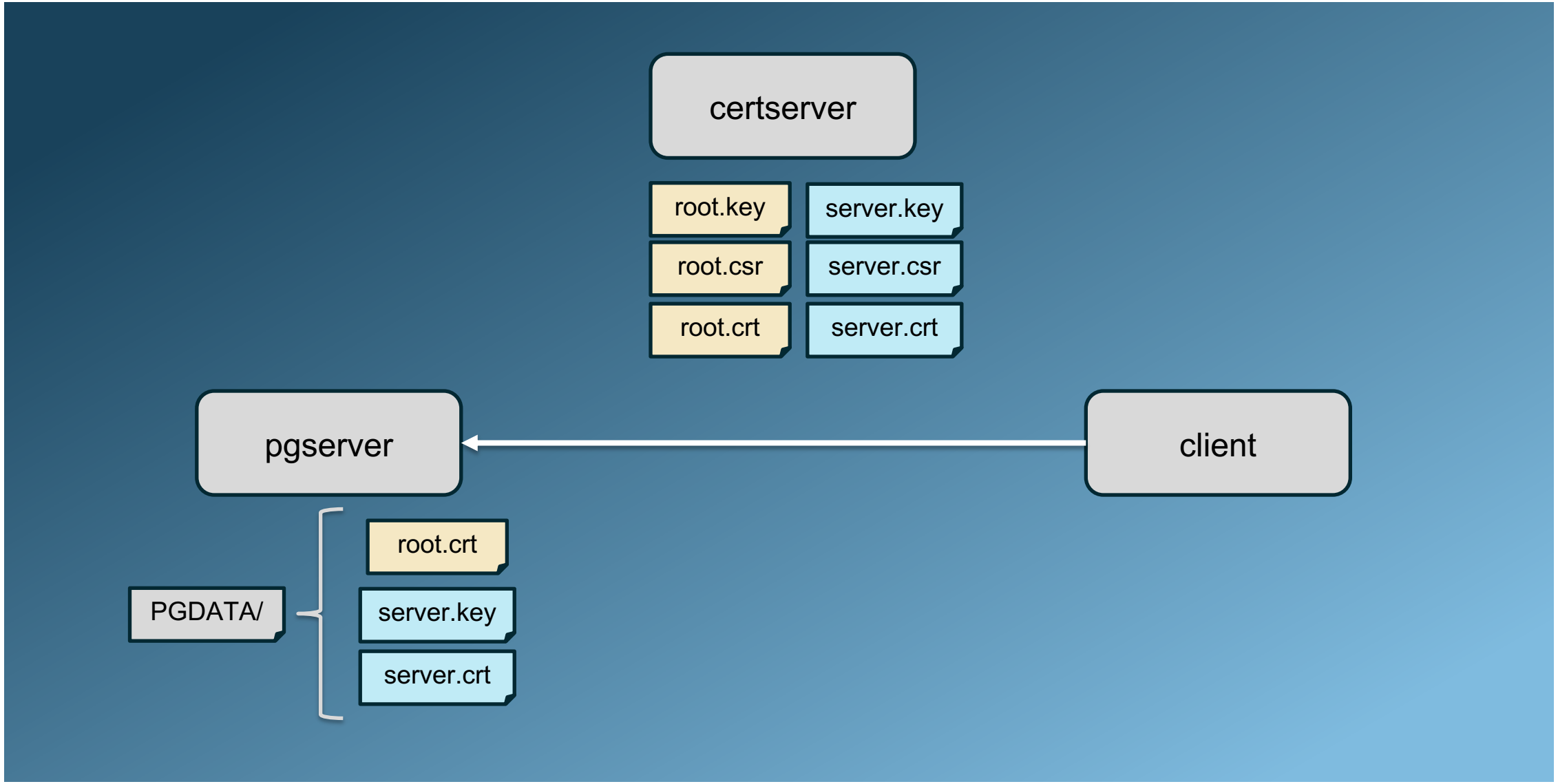
2- SERVER CERTIFICATES



2- SERVER CERTIFICATES



2- SERVER CERTIFICATES



2- Server certificates

postgresql.conf:

```
ssl = on
ssl_key_file = 'server.key'
ssl_cert_file = 'server.crt'
ssl_ca_file = 'root.crt’
```

pg_hba.conf:

```
hostssl myappdb myappuser 172.18.0.22/32 scram-sha-256
```

Reinicie o Postgres

2- Server certificates

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

2- Server certificates

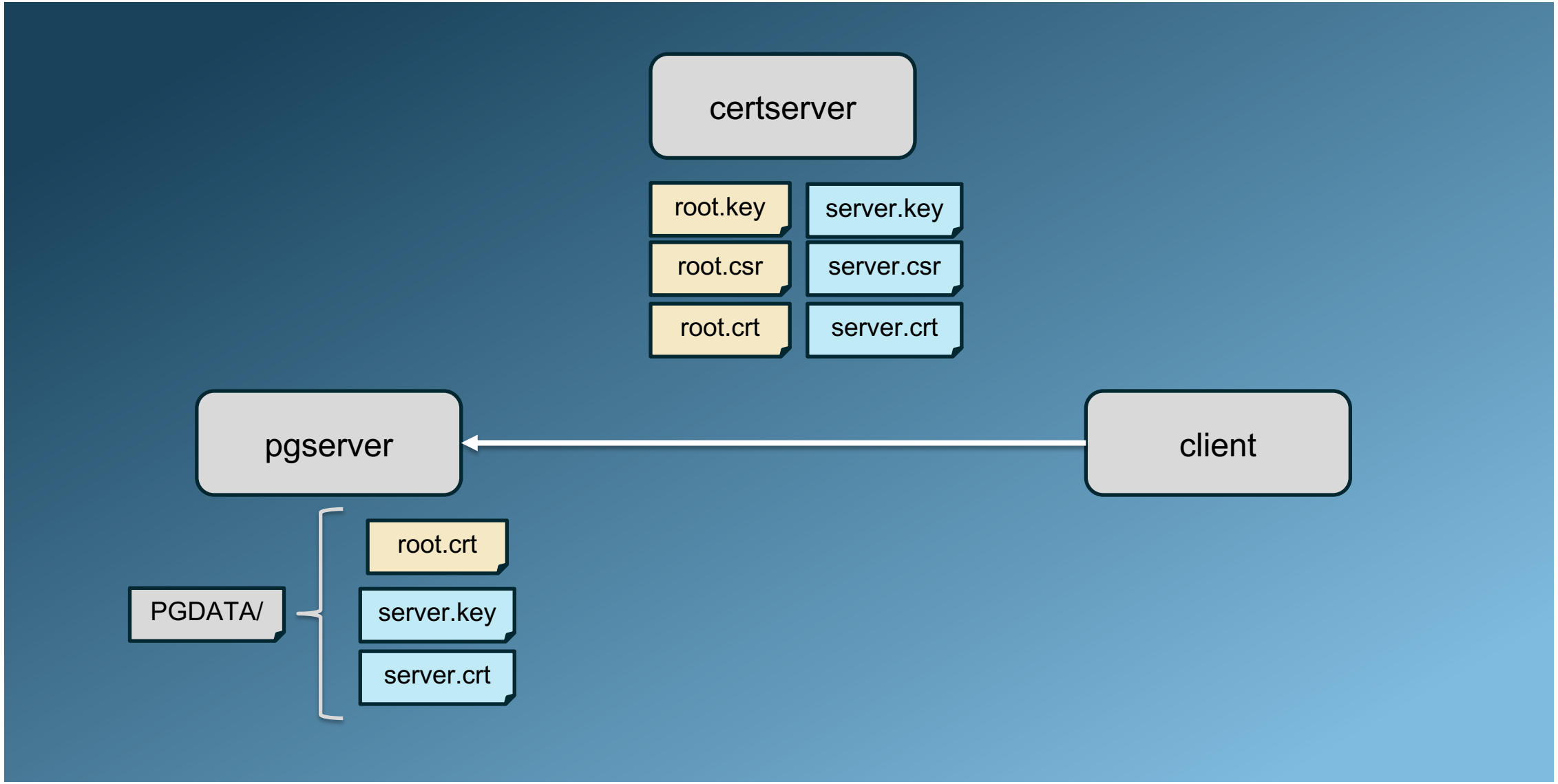
```
postgres=# SELECT
  a.client_addr, a.datname, a.usename,
  s.ssl, s.version, s.bits
FROM pg_stat_ssl s
JOIN pg_stat_activity a ON s.pid = a.pid;
 client_addr | datname  |  usename  | ssl | version | bits
-------------+----------+-----------+-----+---------+------
 172.18.0.22 | myappdb  | myappuser | t   | TLSv1.3 |  256
             | postgres | postgres  | f   |         |
(2 rows)
```

sslmode	Descrição
disable	Utilize apenas conexões sem criptografia.
allow	Tente uma conexão sem criptografia. Se falhar, tente com criptografia.
prefer	(**Padrão**) Tente uma conexão com criptografia. Se falhar, tente sem criptografia.
require	Utilize apenas conexões com criptografia. Se um certificado raiz estiver presente no cliente, valide-o contra o certificado apresentado pelo servidor.
verify-ca	Requer certificado raiz no cliente, que será validado contra o certificado apresentado pelo servidor. Se a validação falhar, a conexão não é permitida.
verify-full	Mesmo que **verify-ca**, mas também valida o atributo **host** da connection string contra o **CN** (**Common Name**) do certificado apresentado pelo servidor.

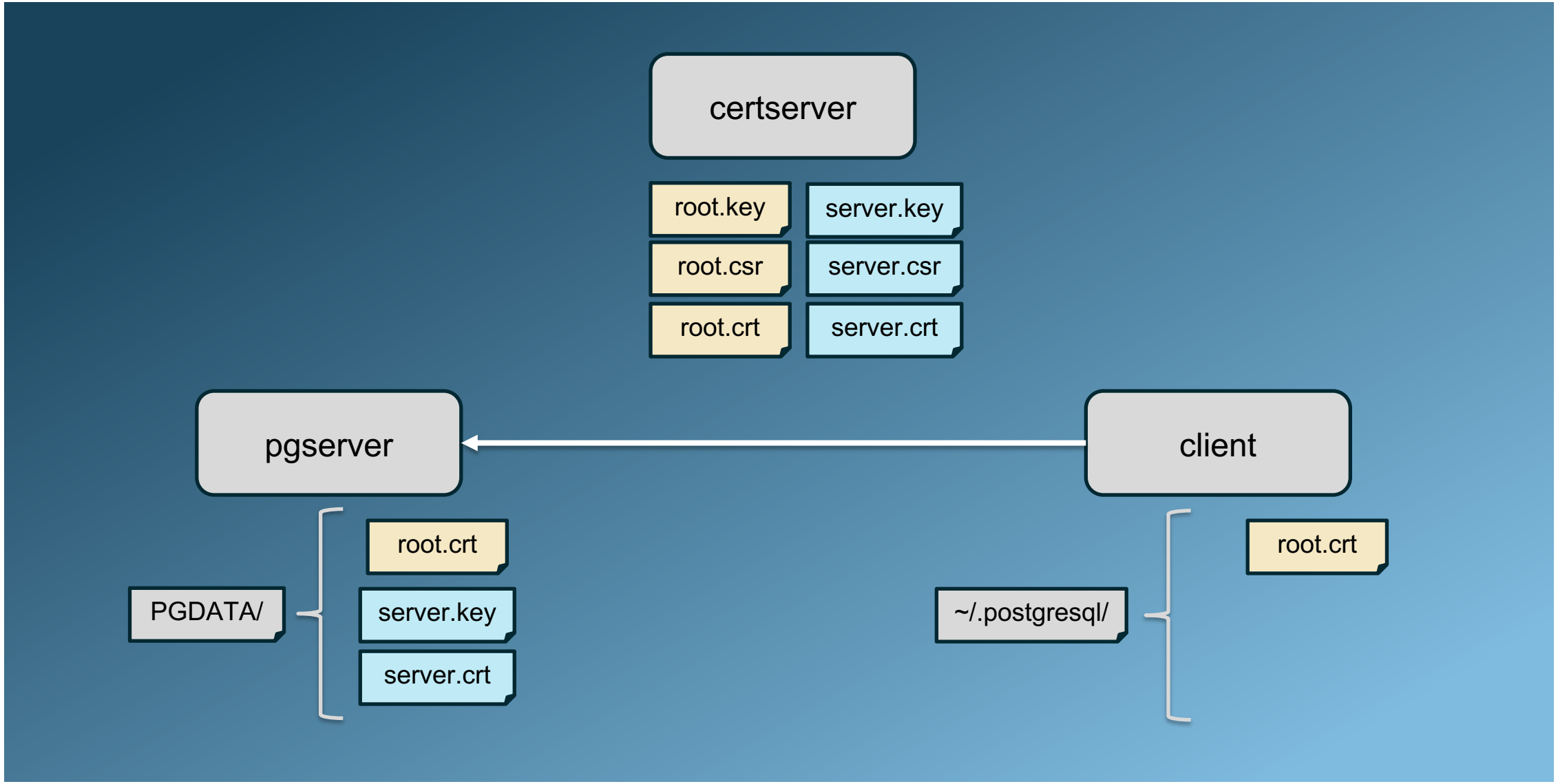
2- Server certificates (verify-ca)

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-ca"

psql: error: connection to server at "pgserver" (172.18.0.21), port 5432
failed: root certificate file "/root/.postgresql/root.crt" does not exist

Either provide the file, use the system's trusted roots with
sslrootcert=system, or change sslmode to disable server certificate
verification.
```

2- SERVER CERTIFICATES



2- SERVER CERTIFICATES (verify-ca)



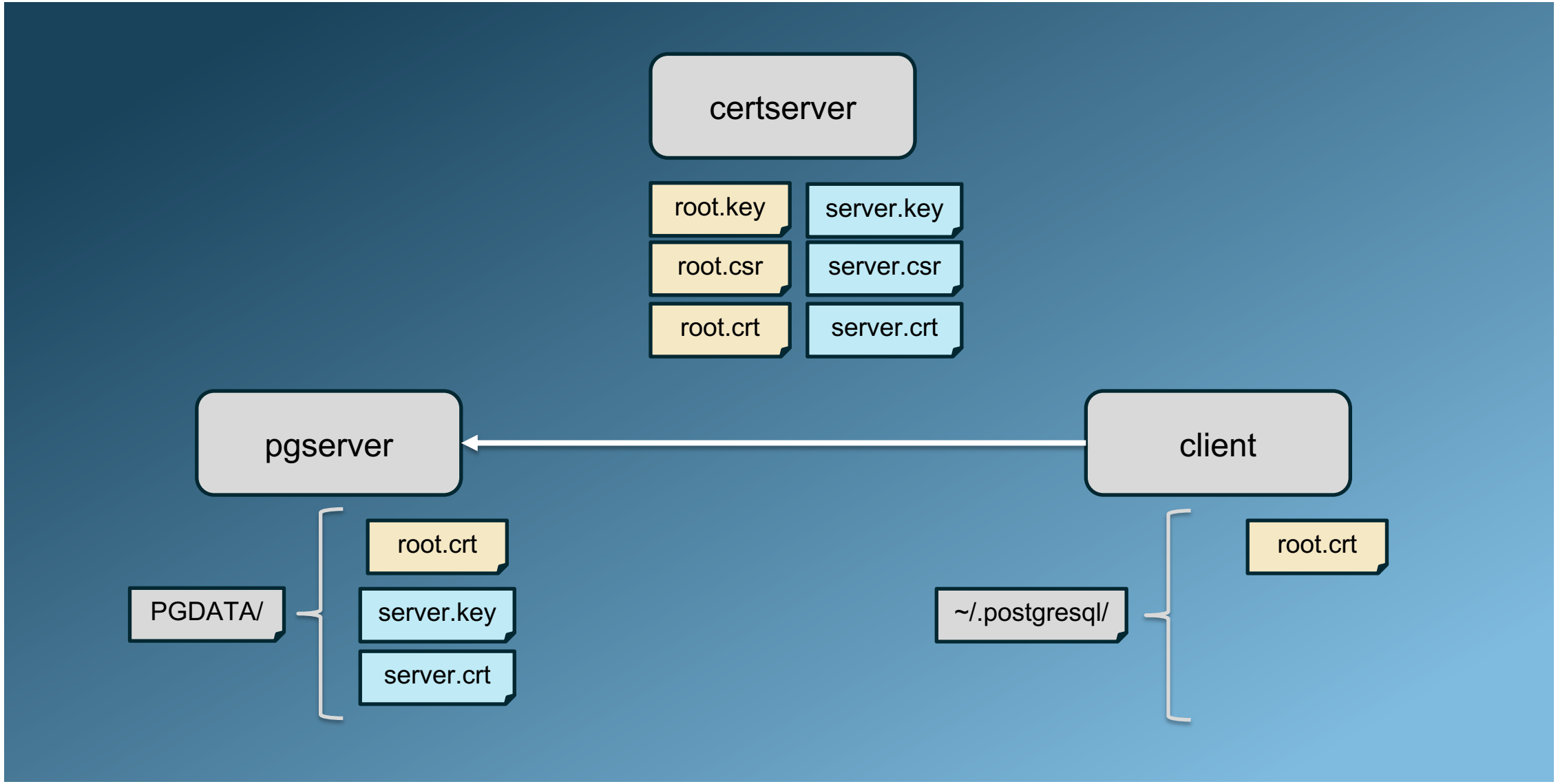
2- Server certificates (verify-ca)

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-ca"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

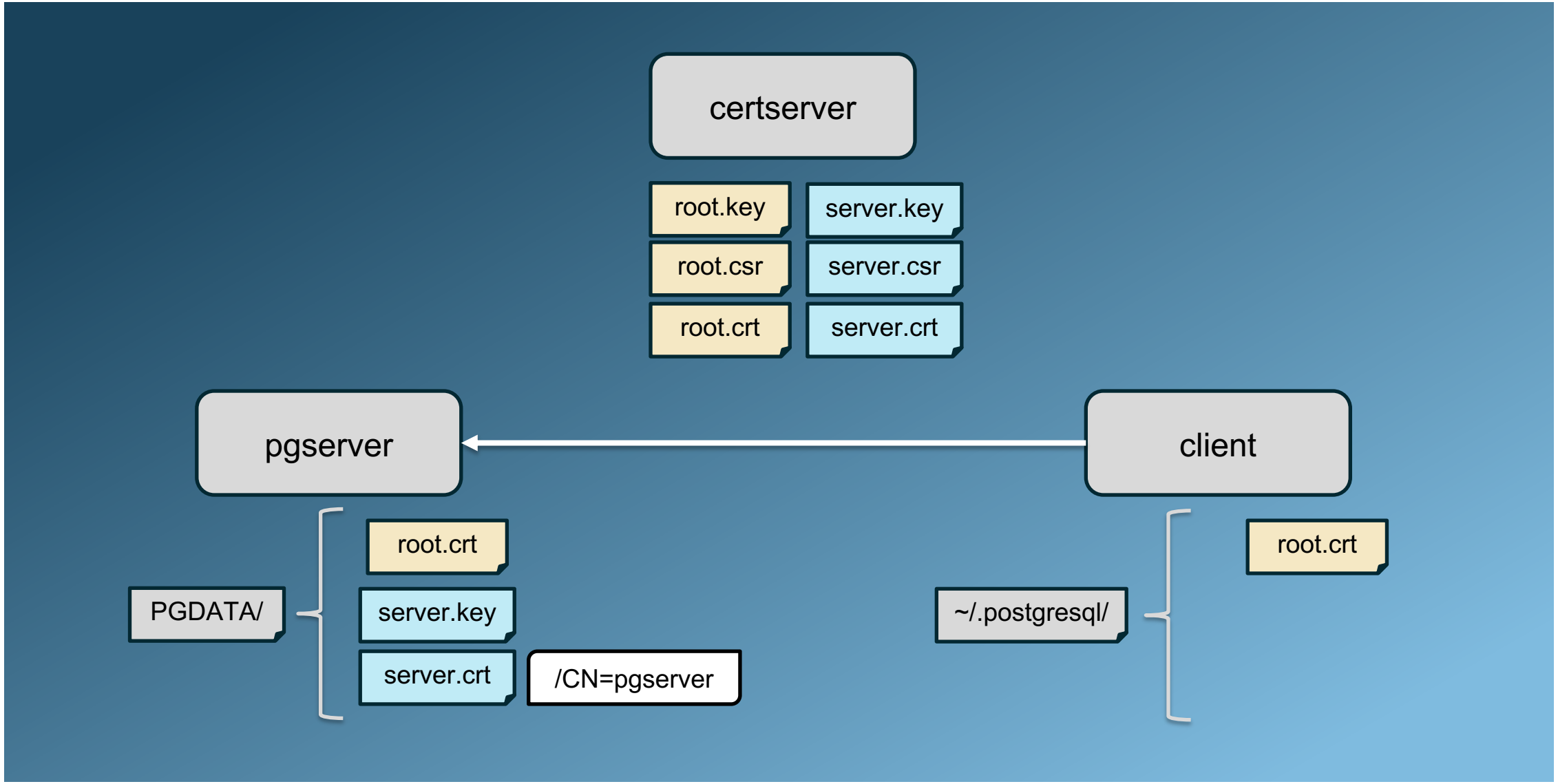
2- Server certificates (verify-ca)

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-ca sslrootcert='/root/.postgresql/root.crt'"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

2- SERVER CERTIFICATES (verify-ca)



2- SERVER CERTIFICATES (verify-full)



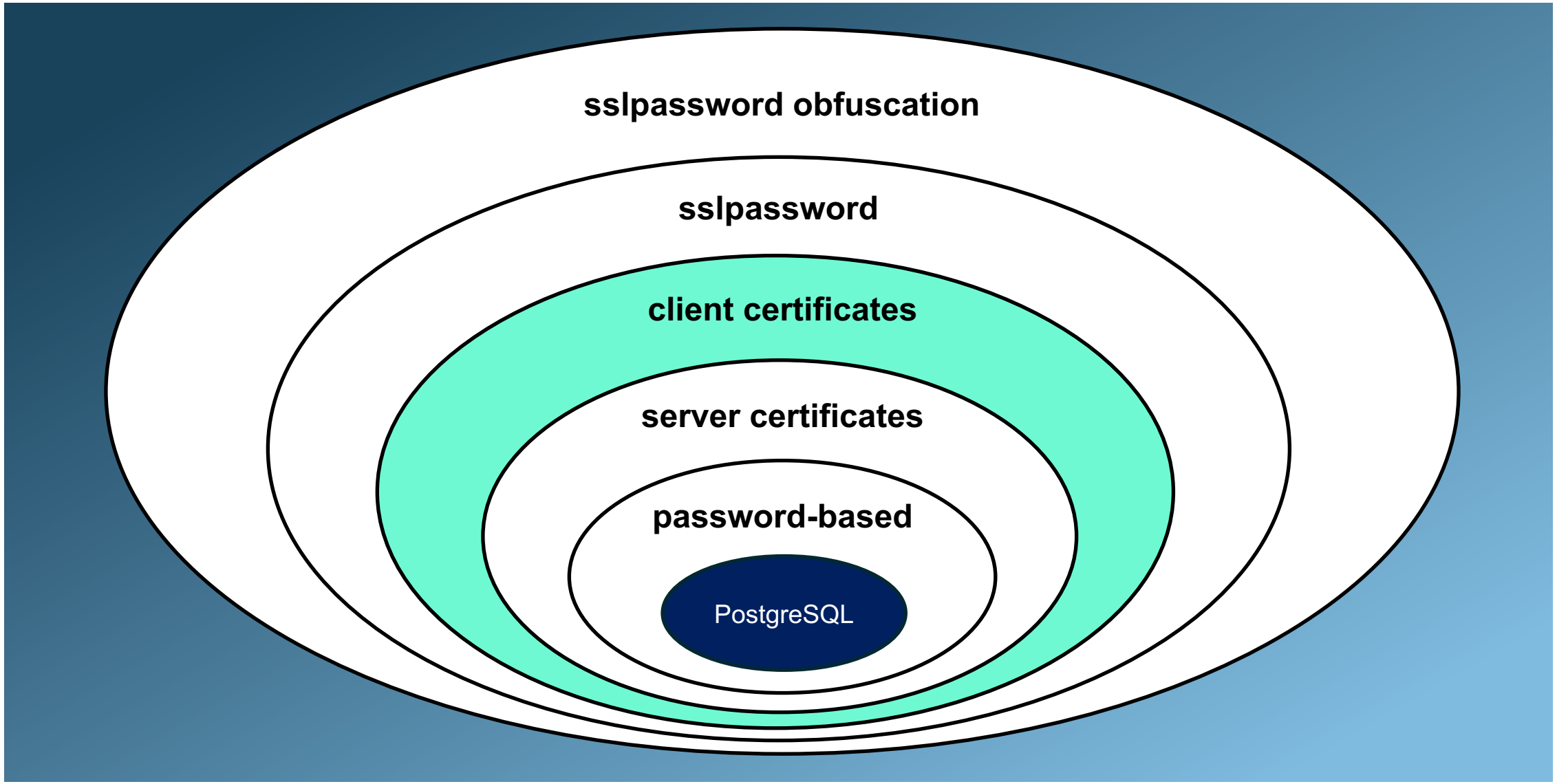
2- Server certificates (verify-full)

```
[root@client ~]# psql "host=172.18.0.21 port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
psql: error: connection to server at "172.18.0.21", port 5432 failed: server
certificate for "pgserver" does not match host name "172.18.0.21"
```

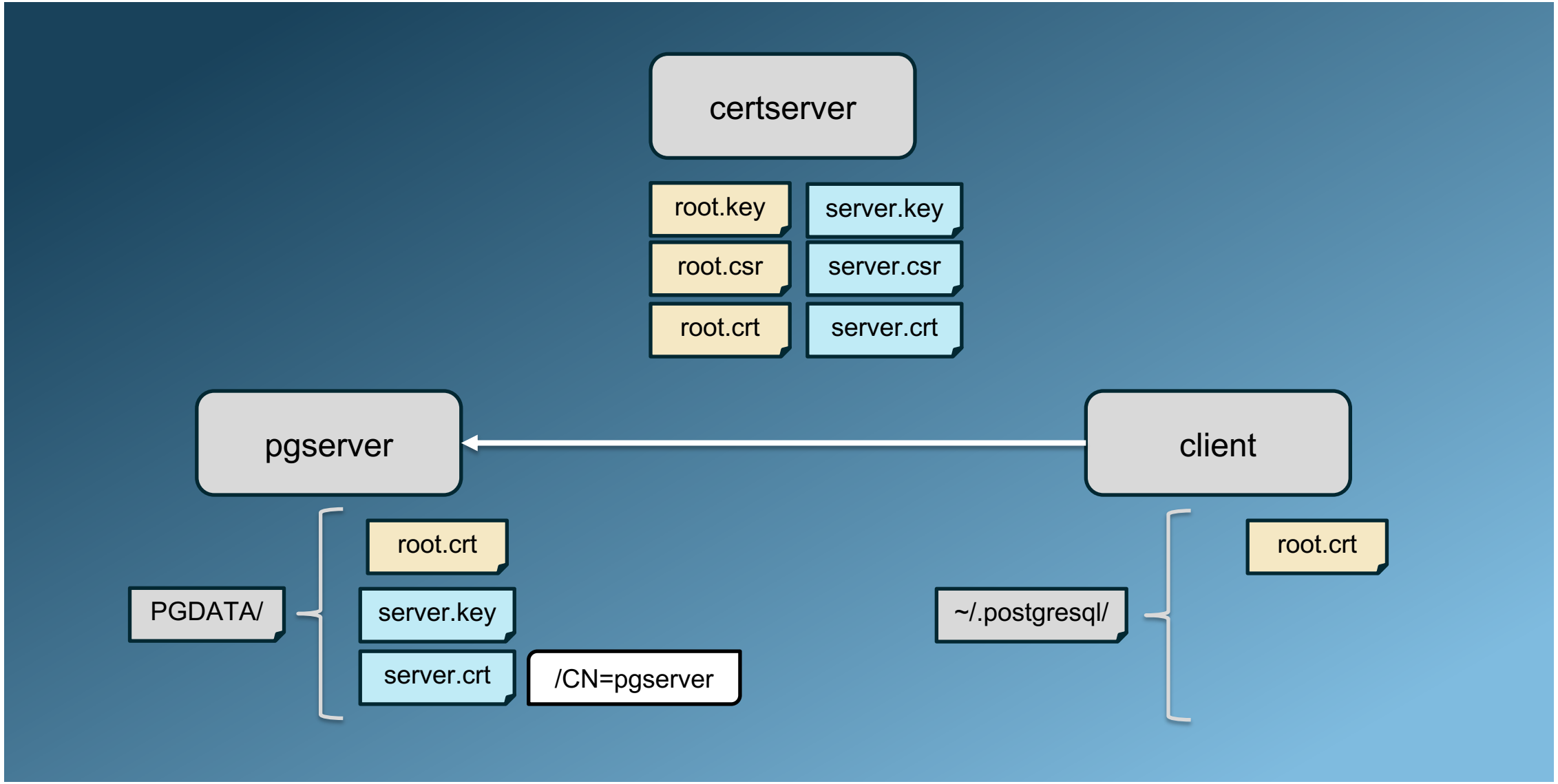
2- Server certificates (verify-full)

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

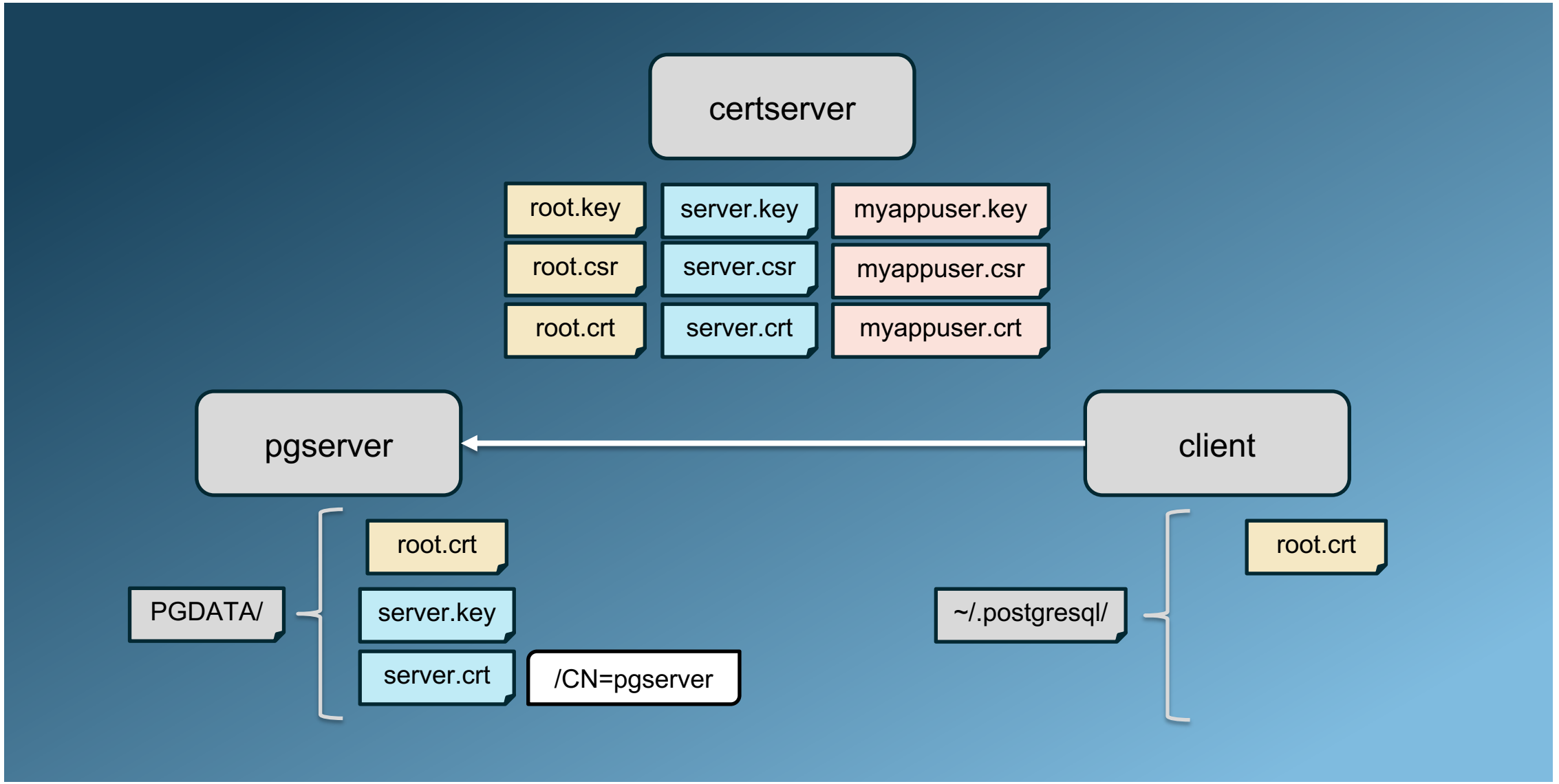
3- CLIENT CERTIFICATES



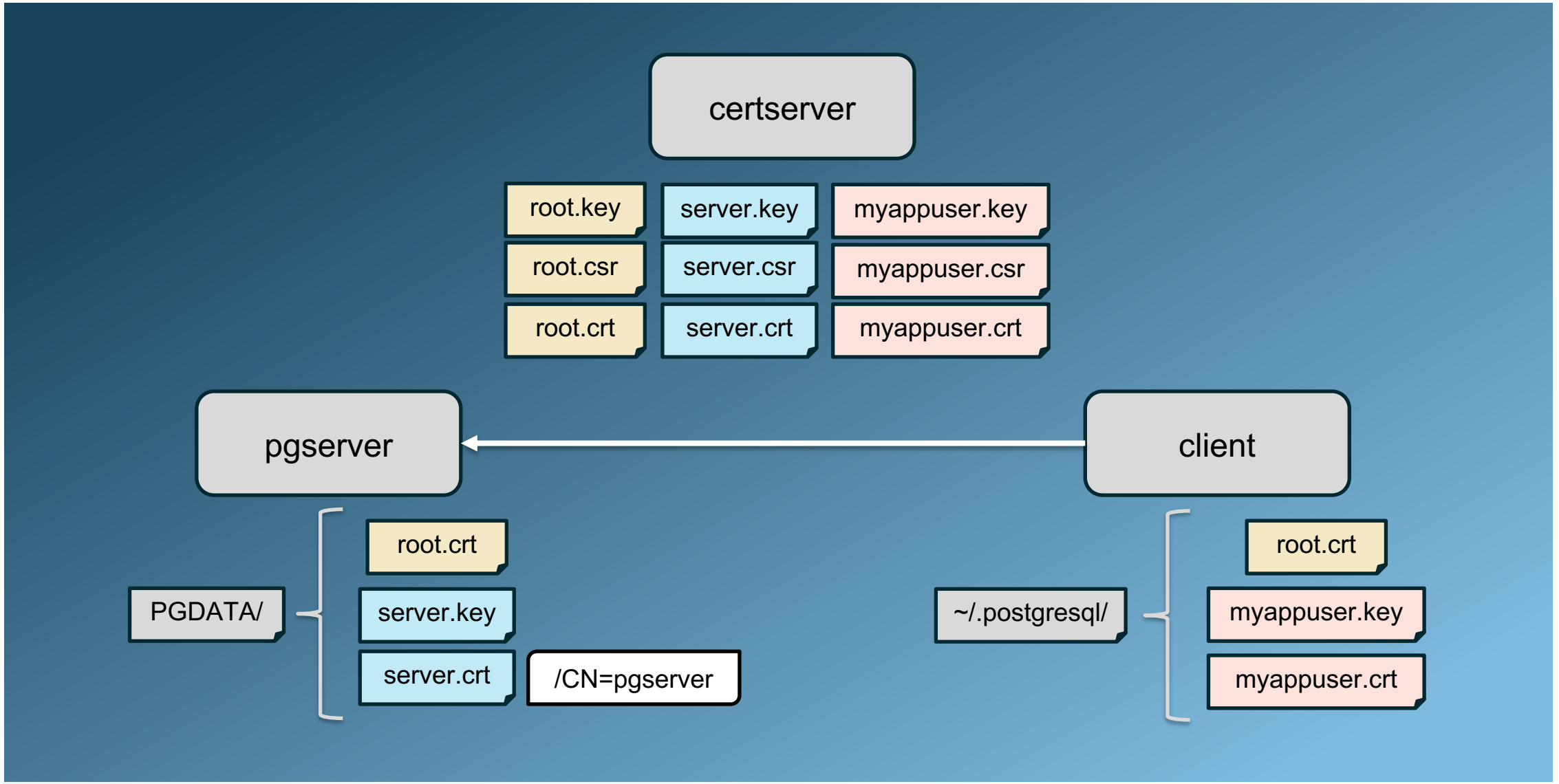
3- CLIENT CERTIFICATES



3- CLIENT CERTIFICATES



3- CLIENT CERTIFICATES

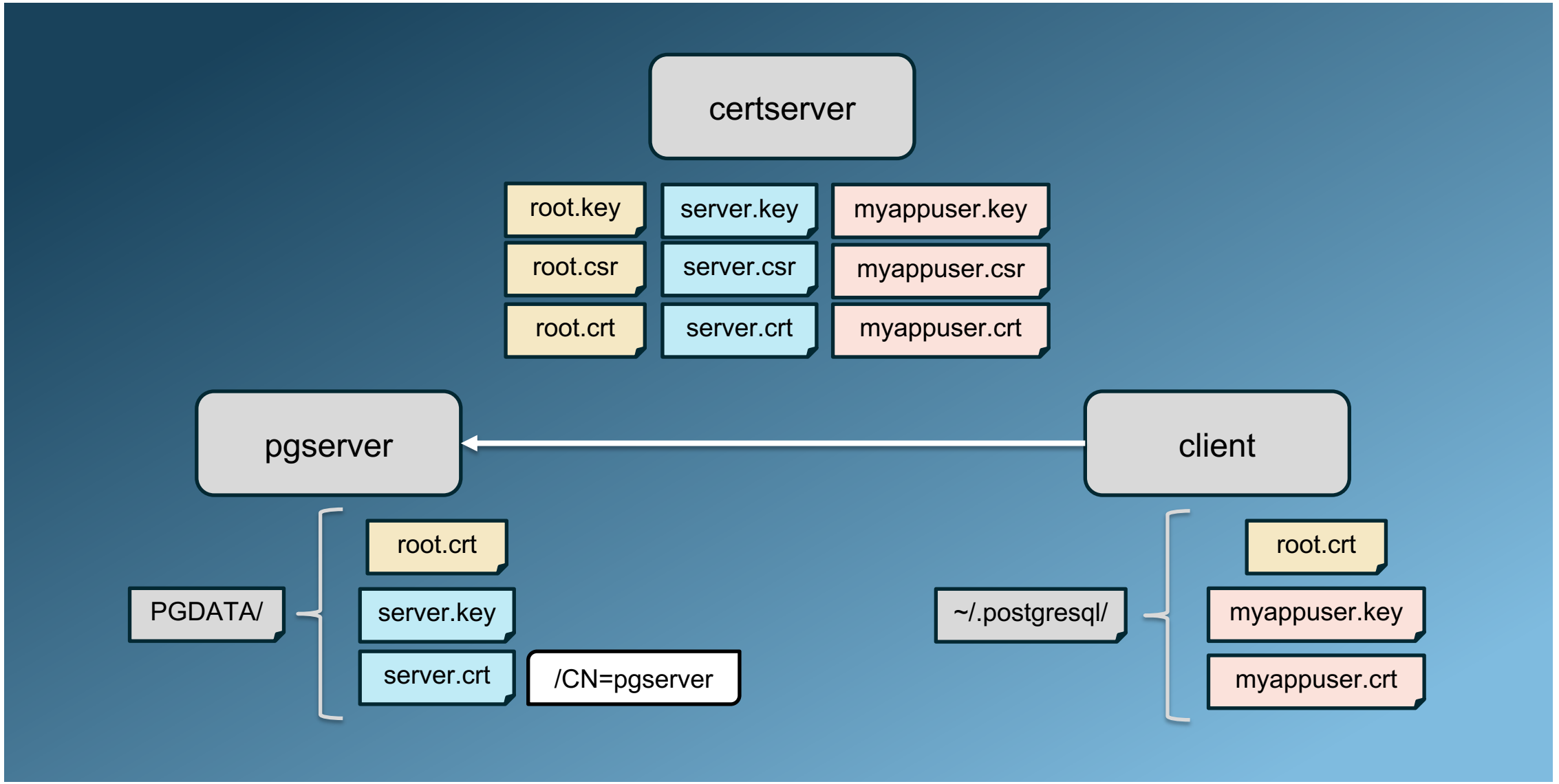


3- Client certificates

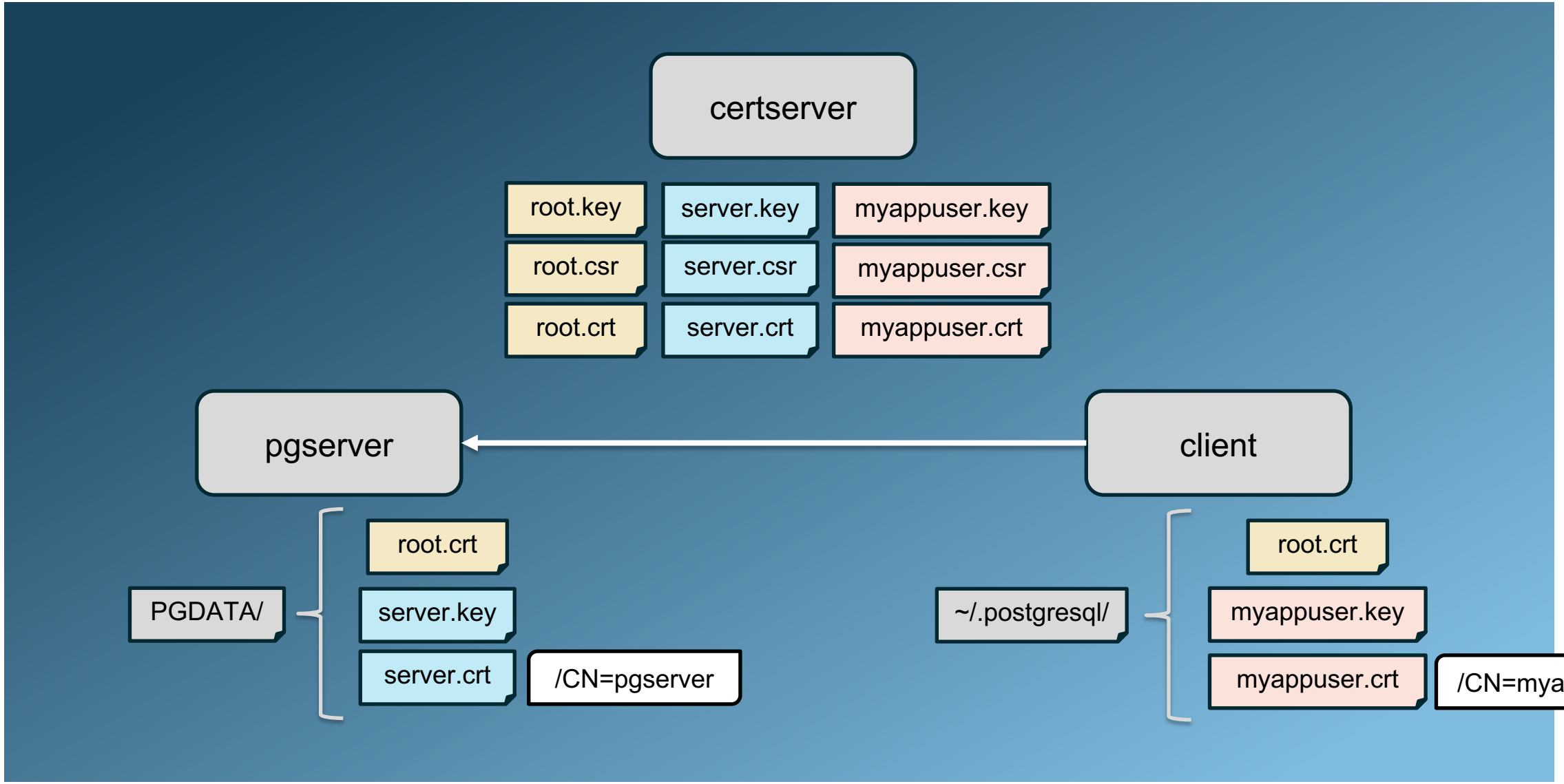
pg_hba.conf:

```
-- Postgres verifica o ssl_ca_file contra o root CA do client certificate
hostssl myappdb myappuser 172.18.0.22/32 scram-sha-256 clientcert=verify-ca

-- Postgres também verifica o CN do client certificate contra o nome de usuário
hostssl myappdb myappuser 172.18.0.22/32 scram-sha-256 clientcert=verify-full
```

3- CLIENT CERTIFICATES



3- CLIENT CERTIFICATES



3- Client certificates

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
psql: error: connection to server at "pgserver" (172.18.0.21), port 5432
failed: FATAL:  connection requires a valid client certificate
```

3- Client certificates

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full sslkey=/root/.postgresql/myappuser.key
sslcert=/root/.postgresql/myappuser.crt"

psql (17.0)

SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)

Type "help" for help.


myappdb=>
```

3- Client certificates

```
[root@client ~]# export PGSSLKEY=/root/.postgresql/myappuser.key
[root@client ~]# export PGSSLCERT=/root/.postgresql/myappuser.crt


[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

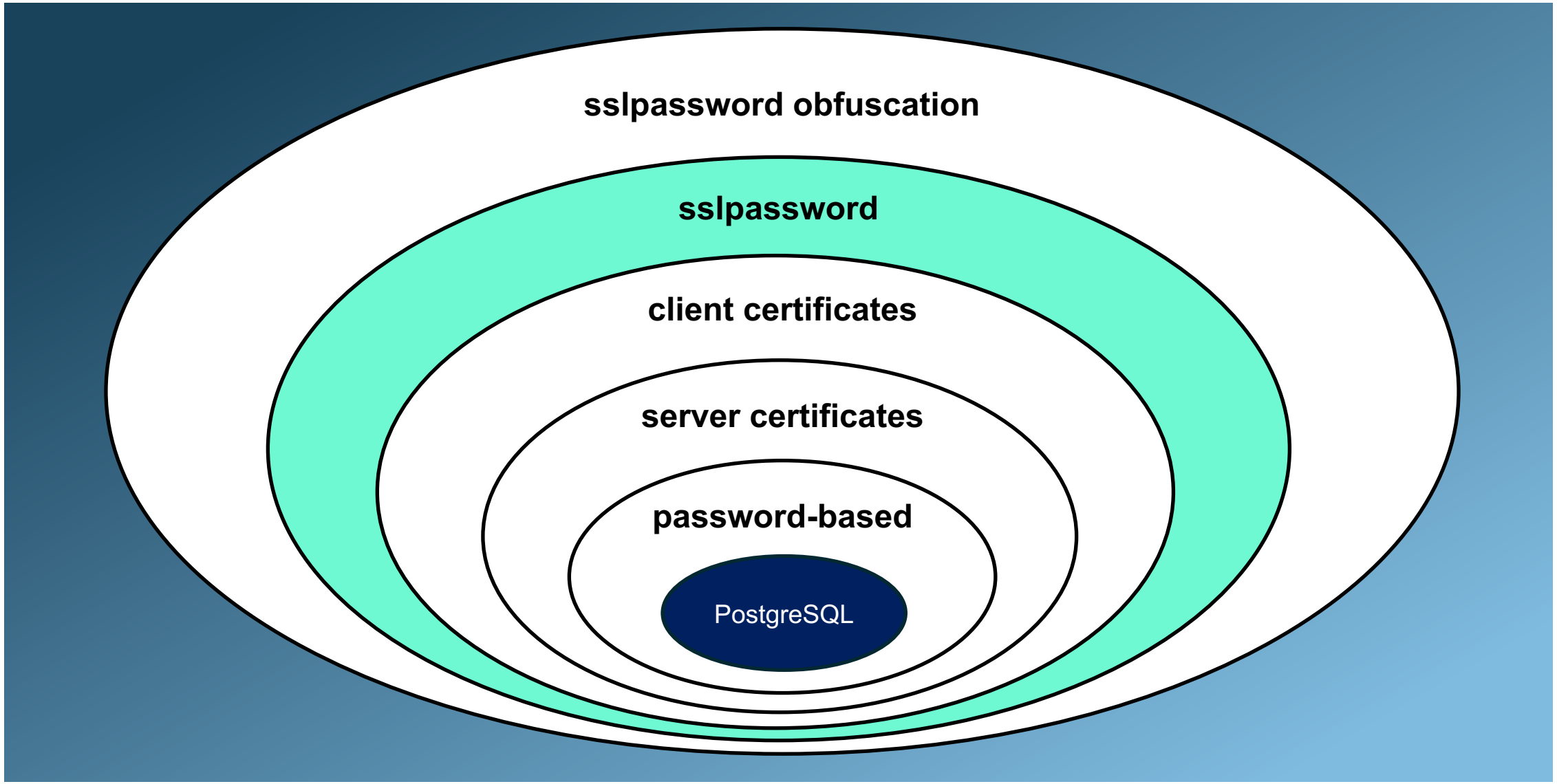
3- Client certificates

```
[root@client ~]# mv ~/.postgresql/myappuser.key ~/.postgresql/postgresql.key
[root@client ~]# mv ~/.postgresql/myappuser.crt ~/.postgresql/postgresql.crt


[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

3- Client certificates

```
postgres=# SELECT
  a.client_addr, a.datname, a.usename,
  s.ssl, s.version, s.bits, s.client_dn, s.issuer_dn
FROM pg_stat_ssl s
JOIN pg_stat_activity a ON s.pid = a.pid;
 client_addr | datname  |  usename  | ssl | version | bits |   client_dn   |
issuer_dn
-------------+----------+-----------+-----+---------+------+---------------+---
-------------
 172.18.0.22 | myappdb  | myappuser | t   | TLSv1.3 |  256 | /CN=myappuser |
/CN=certserver
             | postgres | postgres  | f   |         |      |               |
(2 rows)
```

4- SSLPASSWORD



4- sslpassword

pg_hba.conf:

```
-- O método de autenticação “cert”
hostssl myappdb myappuser 172.18.0.22/32 cert

-- Na verdade é o mesmo que:
hostssl myappdb myappuser 172.18.0.22/32 trust clientcert=verify-full
```

4- sslpassword

```
[root@client ~]# rm ~/.pgpass
[root@client ~]# export PGSSLKEY=/root/.postgresql/myappuser.key
[root@client ~]# export PGSSLCERT=/root/.postgresql/myappuser.crt


[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
Enter PEM pass phrase:
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

4- sslpassword

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full sslpassword='oe4keeP3'"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

4- sslpassword

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full"
Enter PEM pass phrase:
psql: error: connection to server at "pgserver" (172.18.0.21), port 5432
failed: could not load private key file "/root/.postgresql/myappuser.key": bad
decrypt
```

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full sslpassword='aaa'"
psql: error: connection to server at "pgserver" (172.18.0.21), port 5432
failed: could not load private key file "/root/.postgresql/myappuser.key": bad
decrypt
```

4- sslpassword

~/.pg_service.conf:

```
[myapp]
host=pgserver
port=5432
dbname=myappdb
user=myappuser
password=ohsei7Ae
sslmode=verify-full
sslrootcert=/root/.postgresql/root.crt
sslcert=/root/.postgresql/myappuser.crt
sslkey=/root/.postgresql/myappuser.key
sslpassword=oe4keeP3
```

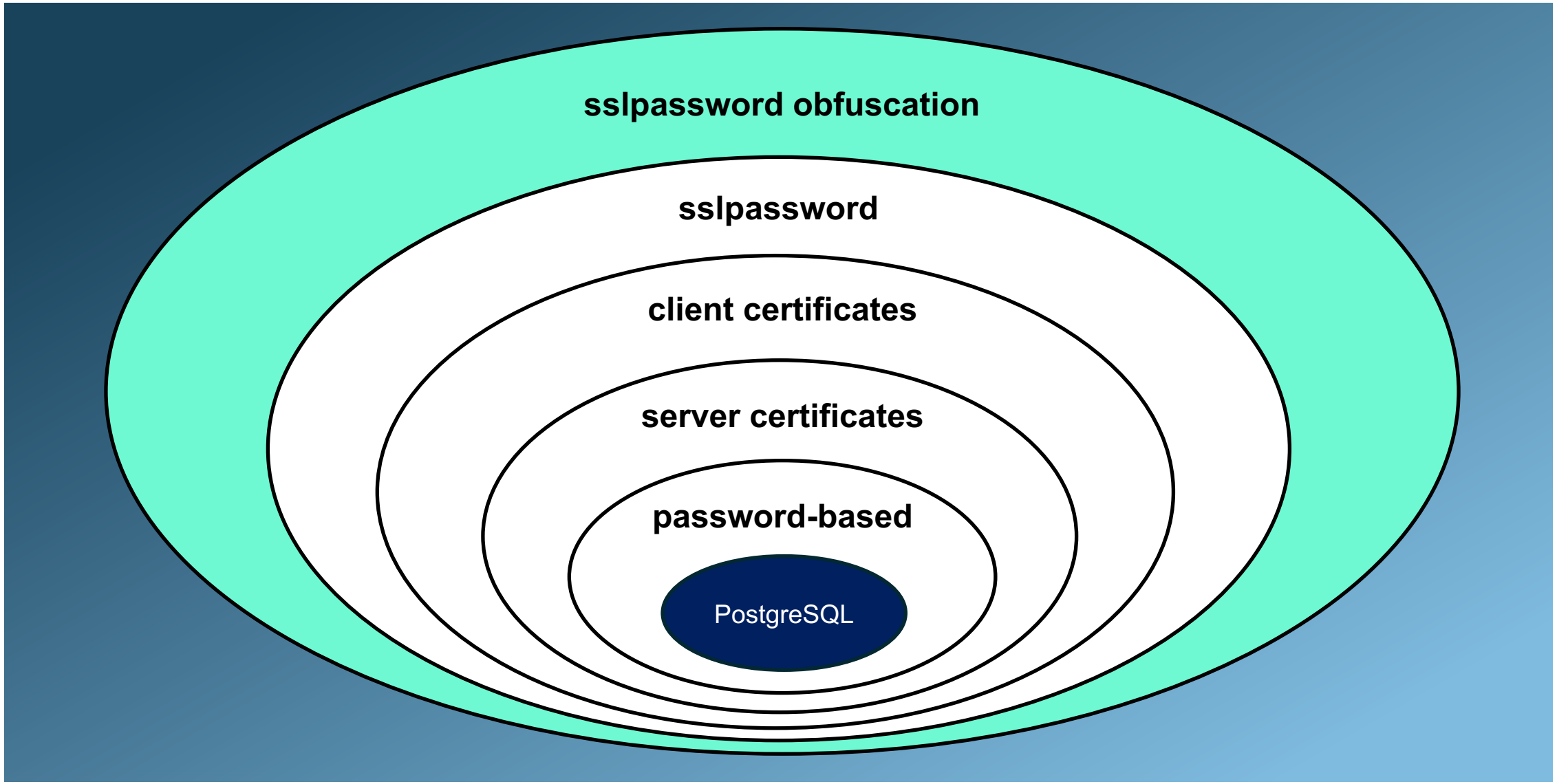
4- sslpassword

```
chmod 600 ~/.pg_service.conf


[root@client ~]# psql "service=myapp"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

	password	sslpassword
Senha de que?	Do usuário do banco de dados.	Da chave SSL criptografada para conexão via client certificate.
Modo interativo	`Password for user XXX:`	`Enter PEM pass phrase:`
Variável de ambiente	`PGPASSWORD`	N/A
Arquivo de senha	`~/.pgpass`	N/A
Pode ser usado no `~/.pg_service.conf`?	Sim	Sim

`sslpassword` obfuscation

5- SSLPASSWORD OBFUSCATION



5- sslpassword obfuscation

- Não há variável de ambiente para **`sslpassword`**
- Nem sempre é possível usar **`~/.pg_service.conf`** pra esconder a **`sslpassword`**
- **`sslpassword`** pode ser necessária em uma connection string compartilhada

- Já temos segurança...
  - Regras no **`pg_hba.conf`**
  - Acesso aos client certificates

- ... Mas podemos dificultar ainda mais!

5- sslpassword obfuscation

- Estratégia
  - Ofuscar a **`sslpassword`** na connection string
  - Desofuscar a **`sslpassword`** no momento da conexão
- Como
  - Hook **`PQsetSSLKeyPassHook_OpenSSL`**
  - Permite construir uma biblioteca de deofuscação a ser carregada junto com a **`libpq`**
  - Função customizada que roda no momento da conexão
- Substituir a **`sslpassword`** da connection string pela **`sslpassword`** real

5- sslpassword obfuscation

```
static char * get_sslpassword(PGconn * conn) {
        PQconninfoOption *conninfo = (*PQconninfo_func)(conn);
        char * result = NULL;
        PQconninfoOption *cursor;
        for (cursor = conninfo; cursor && cursor->keyword; cursor++) {
                if (strcmp(cursor->keyword, "sslpassword") == 0) {
                        if (cursor->val != NULL)
                                result = strdup(cursor->val);
                        break;
                }
        }
        PQconninfoFree(conninfo);
        return result;
}
```

5- sslpassword obfuscation

```
static int deobfuscate_pass(char *buf, int size, PGconn *conn) {
        char * obfuscated;
        char deobfuscated[] = "oe4keeP3";
        obfuscated = get_sslpassword(conn);
        if (obfuscated != NULL) {
                // Deofuscação real aconteceria aqui
                //
                free(obfuscated);
                strncpy(buf, deobfuscated, strlen(deobfuscated) + 1);
                return strlen(buf);
        }
        else {
                buf[0] = '\0';
                return 0;
        }
}
```

5- sslpassword obfuscation

- Não utilize senha em texto puro na biblioteca!

```
[root@client ~]# strings libpqdeobfuscate.so
...
oe4keeP3H
...
```

- Ao invés disso implemente ou utilize um algoritmo de deofuscação ou descriptografia
- Outros argumentos da connection string podem ser utilizados para o seu algoritmo encontrar a senha real

5- sslpassword obfuscation

```
gcc -DUSE_OPENSSL \
  -I/usr/pgsql-17/include/ \
  -I/usr/pgsql-17/include/server/ \
  -L/usr/pgsql-17/include/lib/ \
  -L/usr/lib64/ -lpq \
  libpqdeobfuscate.c \
  -shared -fPIC \
  -o libpqdeobfuscate.so
```

5- sslpassword obfuscation

```
[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser
sslmode=verify-full sslkey=/root/.postgresql/myappuser.key
sslcert=/root/.postgresql/myappuser.crt sslpassword=XXXXXX"

psql: error: connection to server at "pgserver" (172.18.0.21), port 5432
failed: could not load private key file "/root/.postgresql/myappuser.key": bad
decrypt
```

5- sslpassword obfuscation

```
[root@client ~]# export LD_PRELOAD=/usr/pgsql-17/lib/libpq.so.5:/root/libpqdeobfuscate.so

[root@client ~]# psql "host=pgserver port=5432 dbname=myappdb user=myappuser sslmode=verify-full sslkey=/root/.postgresql/myappuser.key sslcert=/root/.postgresql/myappuser.crt sslpassword=XXXXXX"

psql (17.0)

SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off, ALPN: postgresql)

Type "help" for help.


myappdb=>
```

5- sslpassword obfuscation

```
[root@client ~]# cat ~/.pg_service.conf
[myapp]
host=pgserver
port=5432
dbname=myappdb
user=myappuser
sslmode=verify-full
sslrootcert=/root/.postgresql/root.crt
sslcert=/root/.postgresql/myappuser.crt
sslkey=/root/.postgresql/myappuser.key
sslpassword=XXXXXX
```

5- sslpassword obfuscation

```
[root@client ~]# psql "service=myapp"
psql (17.0)
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression:
off, ALPN: postgresql)
Type "help" for help.


myappdb=>
```

EDB
Postgres® for the AI Generation

Repositório e Contato

- Slides dessa palestra em PDF
- Passo a passo com todos os comandos e explicações
- Comandos **`openssl`** pra criar os certificados
- Código-fonte **`libpqdeobfuscate.c`**
- Rocky Linux 9 / PostgreSQL 17
- `https://github.com/wind39/libpqdeobfuscate`
- `william.ivanski@enterprisedb.com`

EDB
Postgres® for the AI Generation

EDB is hiring

https://www.enterprisedb.com/careers