EDB
Postgres® for the AI Generation

sslpassword obfuscation with hook **PQsetSSLKeyPassHook_OpenSSL**

William Ivanski
Senior Principal Support Engineer
EnterpriseDB
December, 4th, 2024

Authentication Security

Importance of Security

- Different industry sectors use PostgreSQL to store **confidential data**:
    - Finance
    - Health
    - Intellectual Property, etc

- Inadequate or lack of security implies in **severe risks**, for example:
    - Unauthorized access to data
    - Data theft, exposition and kidnapping
    - Identity theft
    - Financial, industrial and marketing fraud

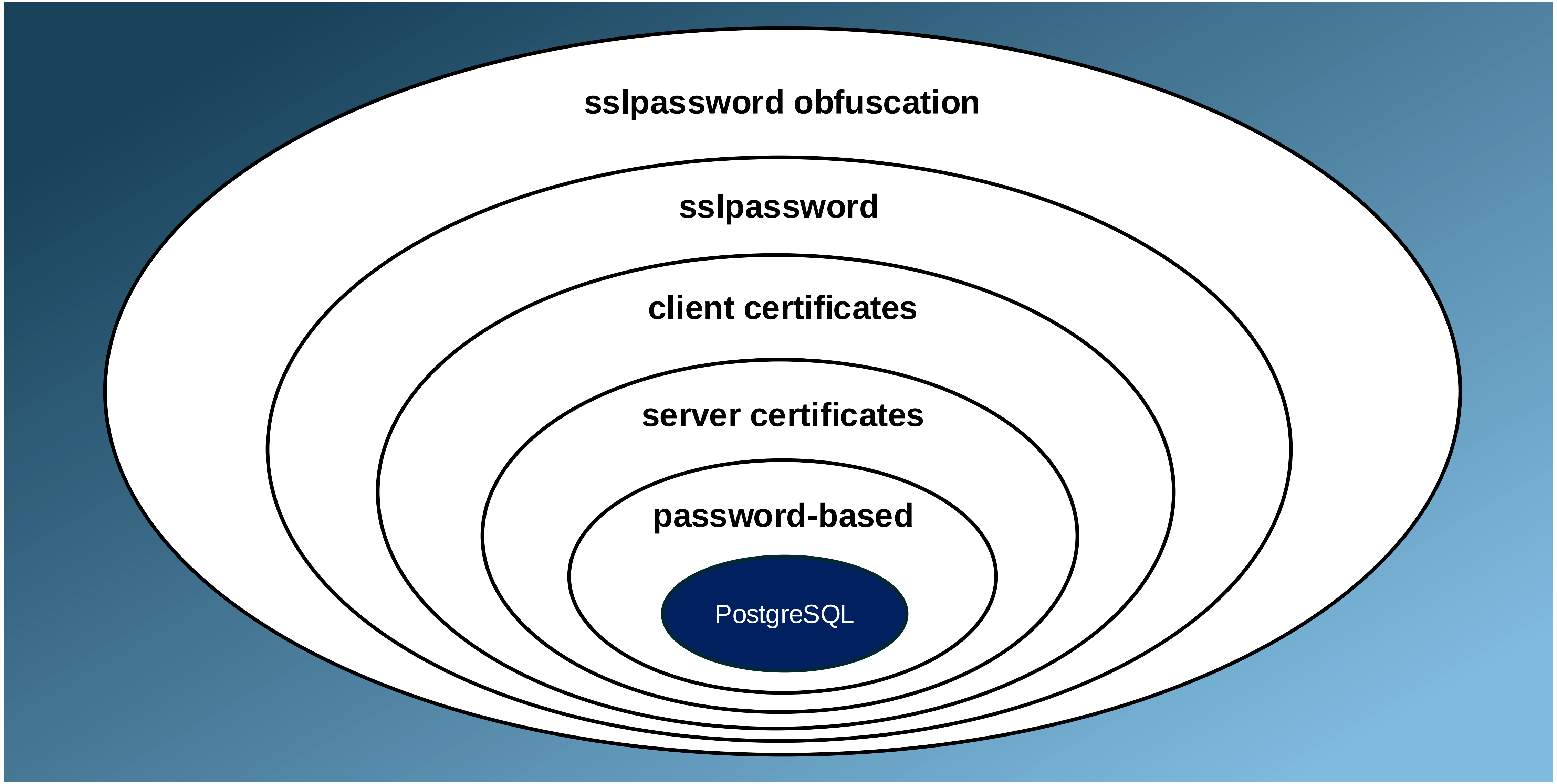
Authentication Security

- Authentication is one of the barriers in data security
- Requirements to log into the system, for example:
  - Origin address
  - User name
  - Database
  - Cryptography
  - Authentication method

Types of Authentication

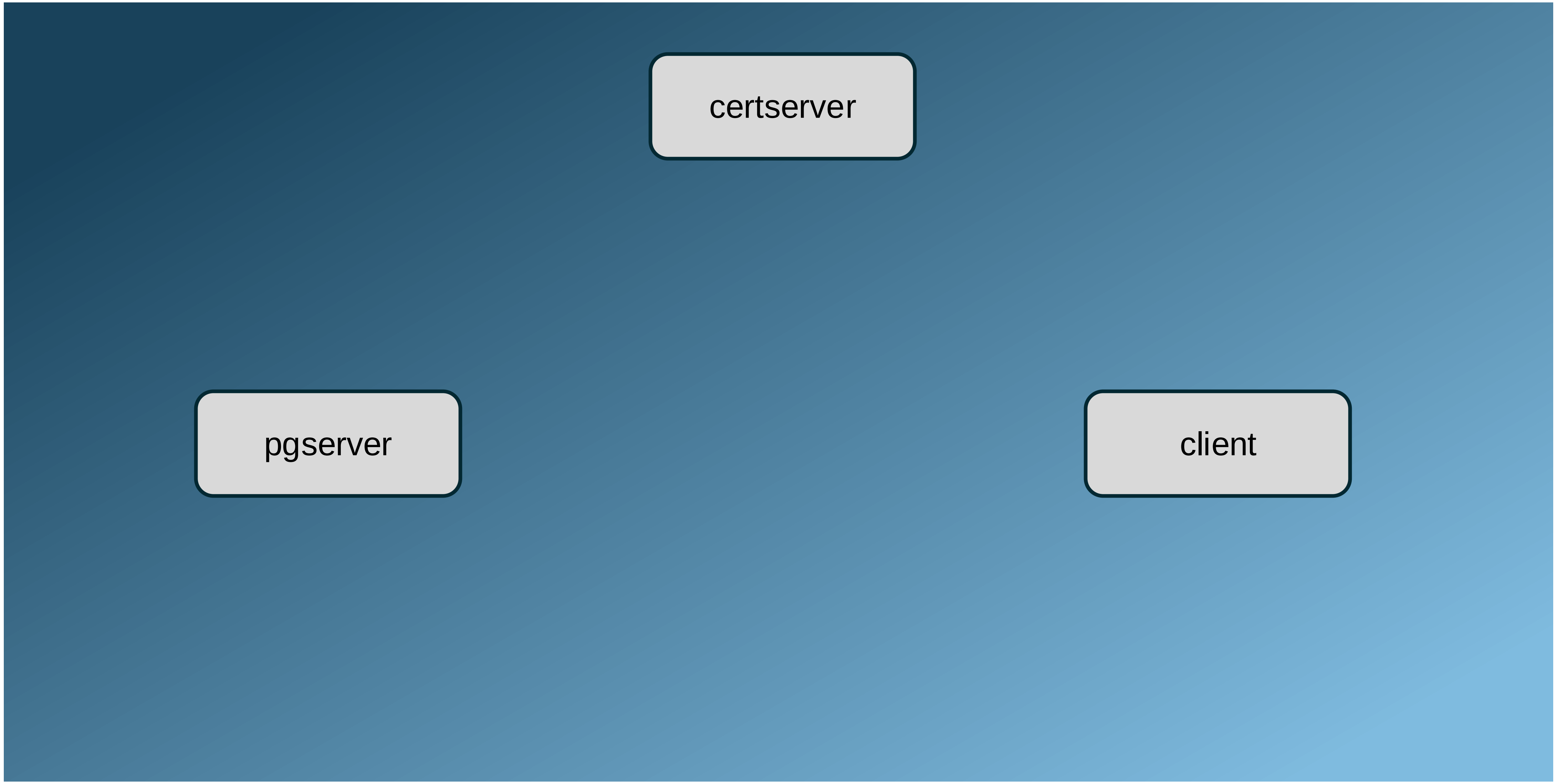
- External authentication
  - GSSAPI, SSPI, LDAP, RADIUS
- Operating system authentication
  - BSD, PAM, Peer, Ident
- Internal PostgreSQL authentication
  - Trust / Reject
  - Password-based: password, md5, SCRAM
  - SSL / TLS

INTERNAL AUTHENTICATION SECURITY (SIMPLIFIED)

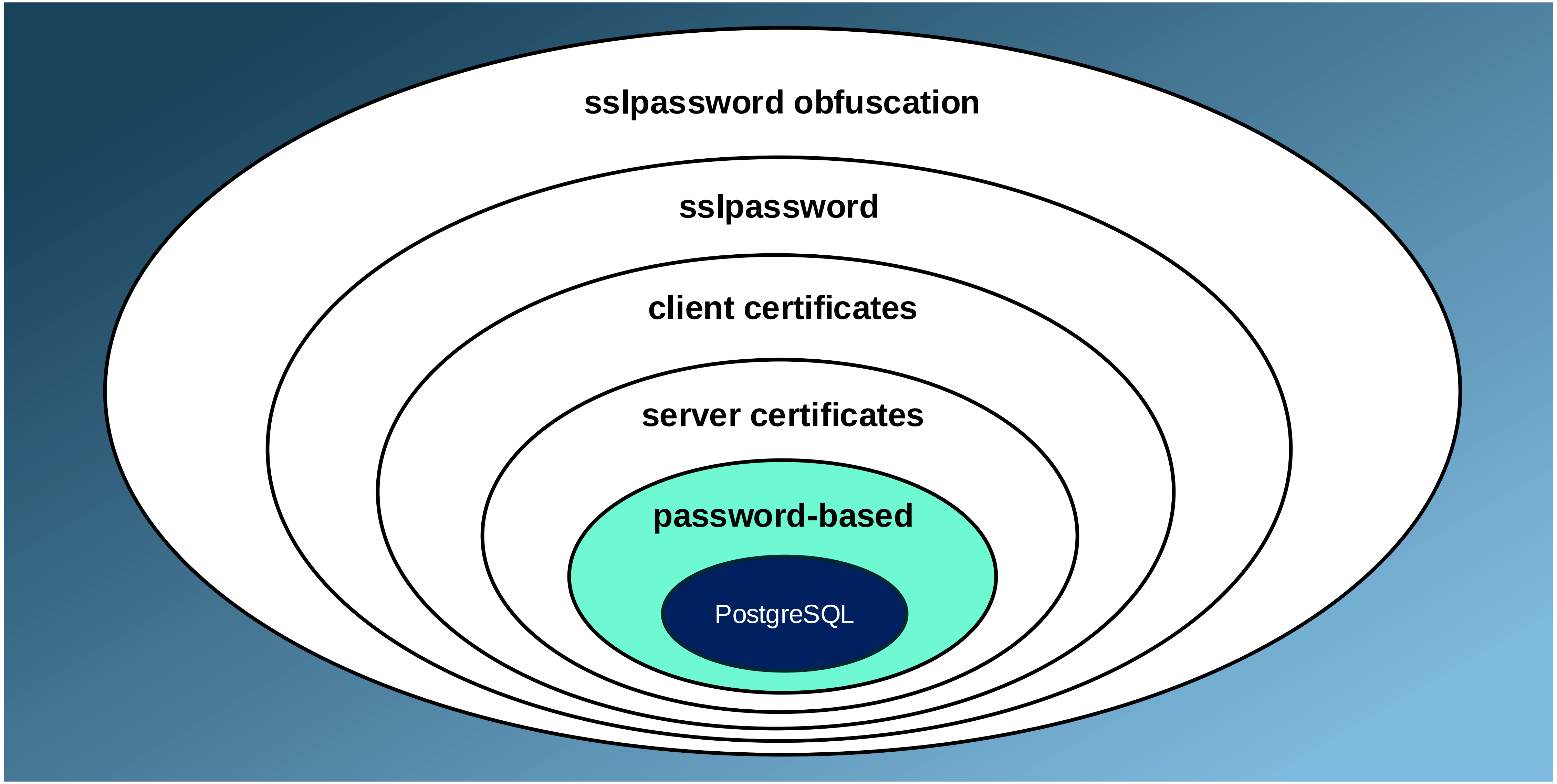


Internal Authentication Laboratory

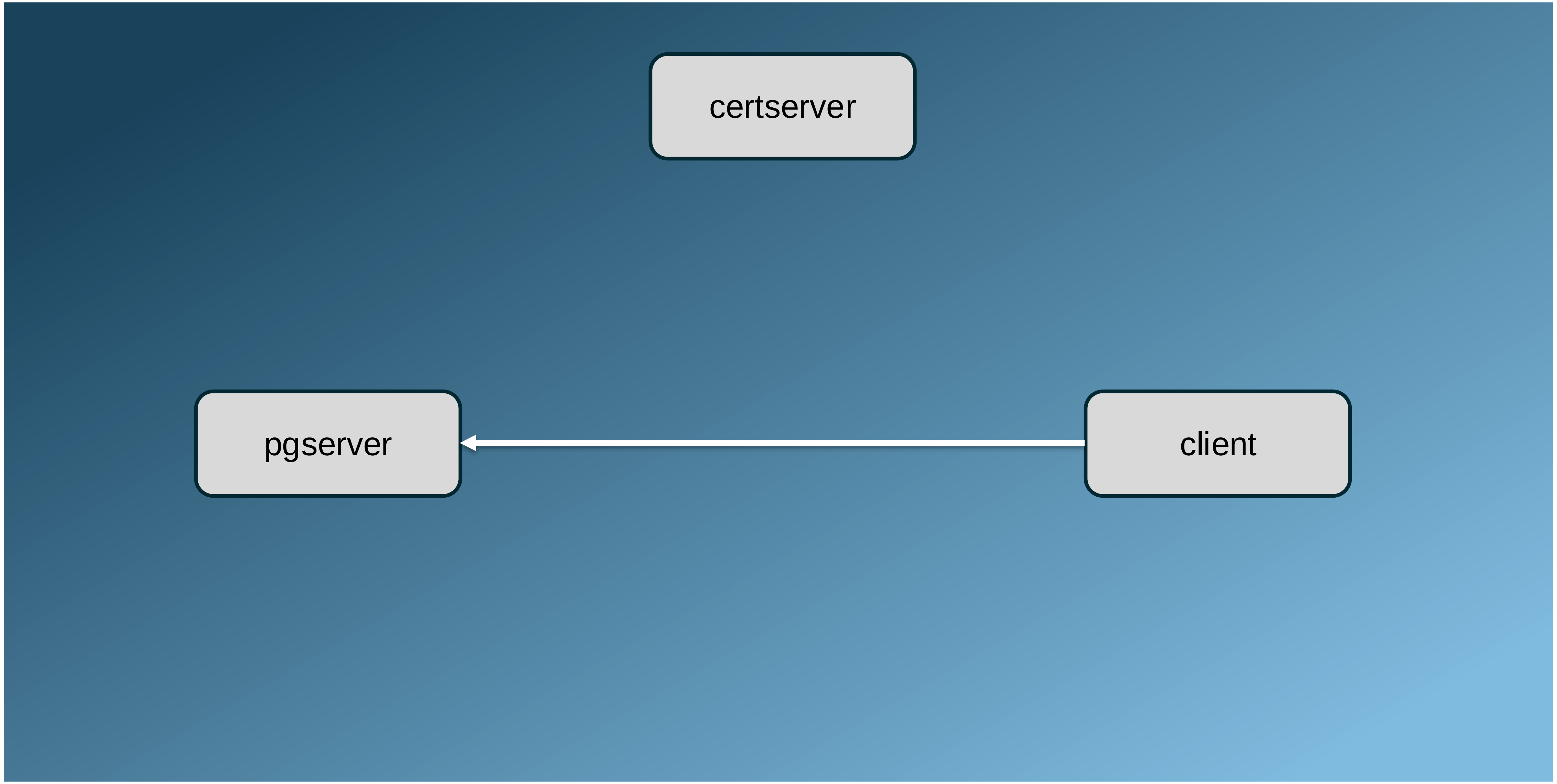
0- SERVERS



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
  - Original implementation by Netscape, now obsolete
- TLS: Transport Layer Security
  - Evolution of SSL
  - Obsolete versions: 1.0 and 1.1
  - Recommended versions: 1.2 and 1.3
- TCP socket-level cryptography
  - HTTPS
  - SSH
  - etc

SSL / TLS

- Asymmetric cryptography:
  - Public / private key pair
  - Public key to encrypt
  - Private key to decrypt

- Symmetric cryptography:
  - Use the same key to encrypt and decrypt

SSL Certificates

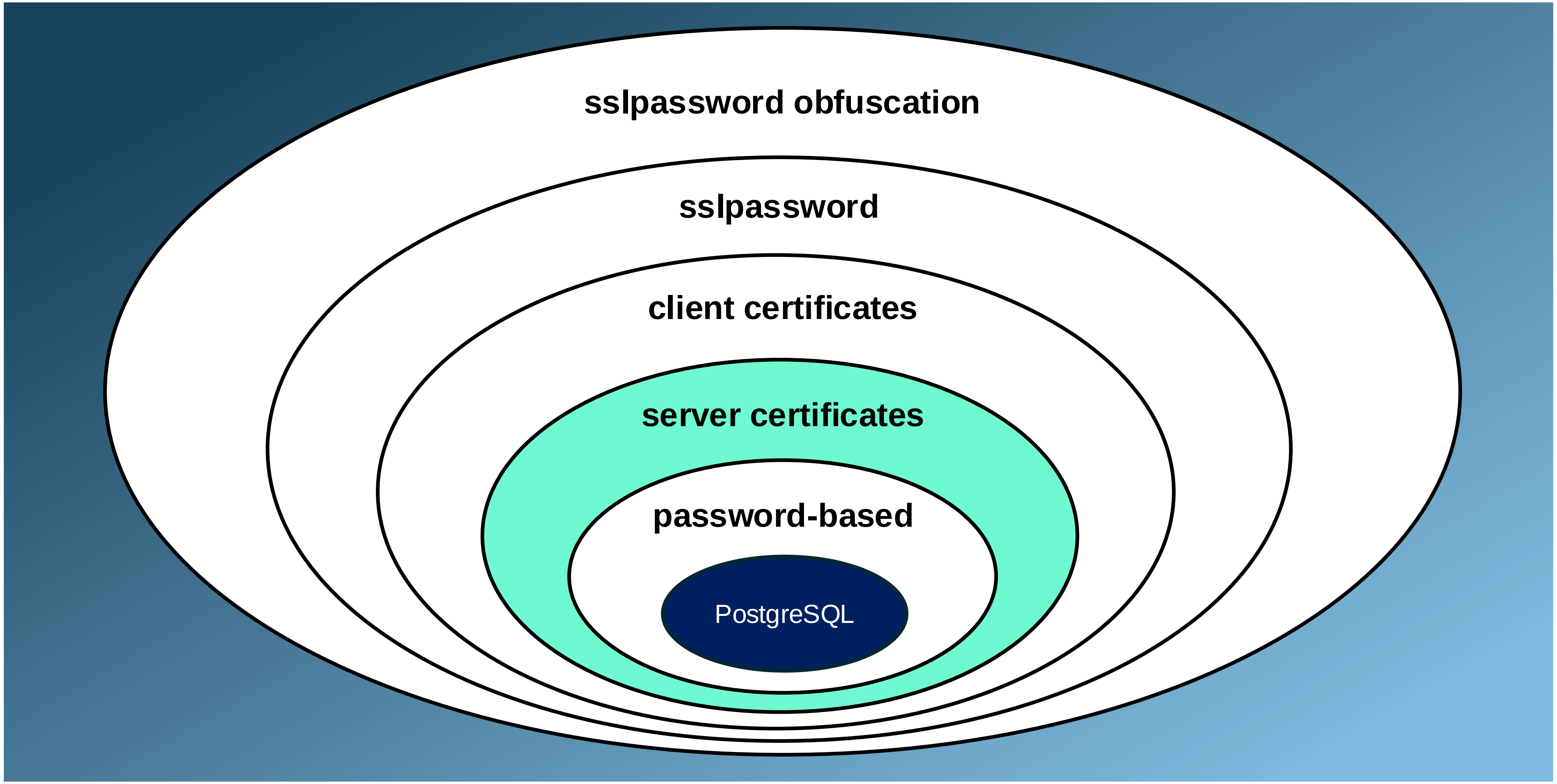
- A SSL certificate contains:
  - Public key
  - Information about the identity
  - Certificate authority (CA)
  - Among other information

- Private key needs to be stored safely and never shared!

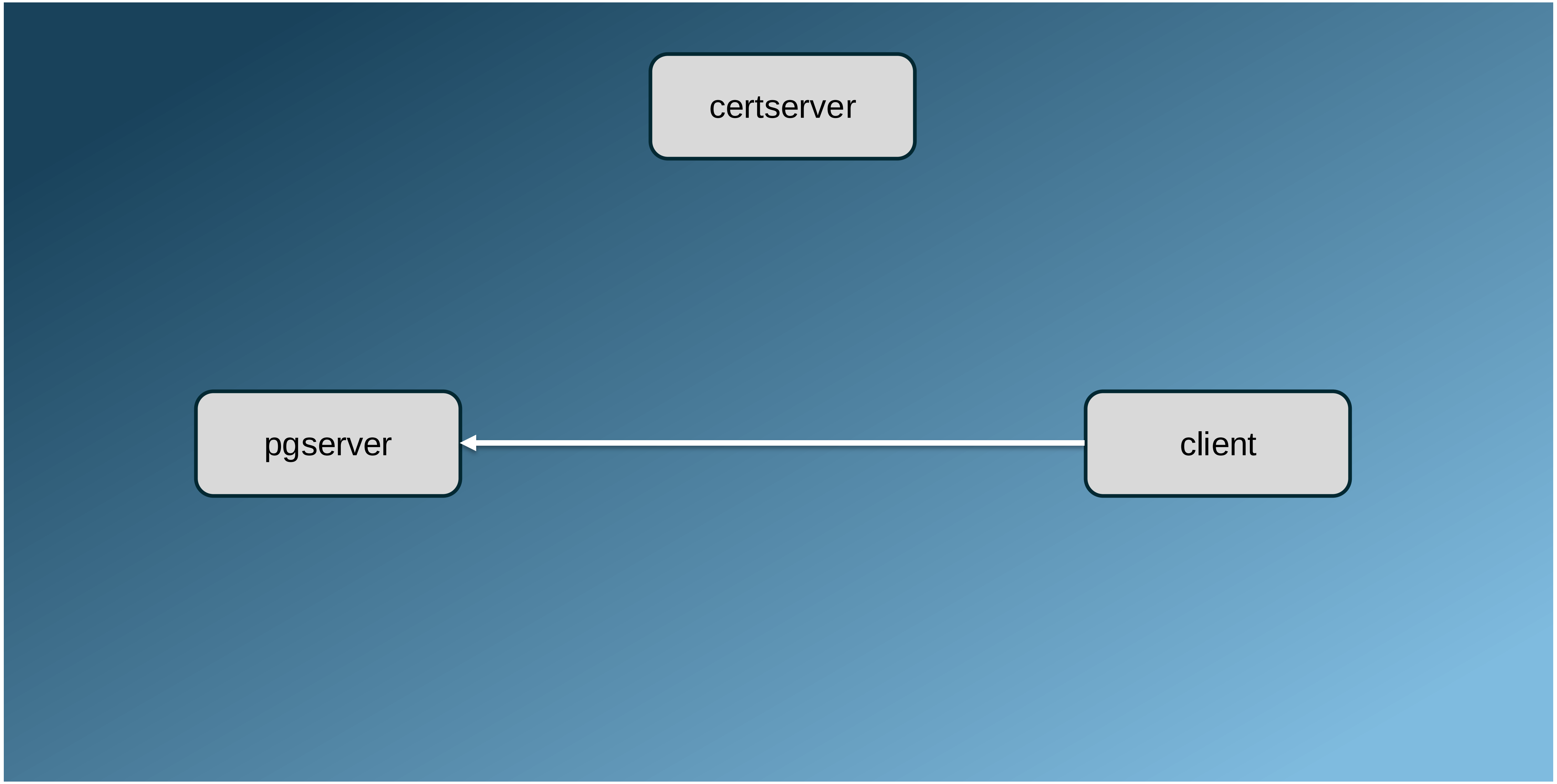
Encrypted communication using SSL certificates

- Uses both symmetric and assymmetric keys
- Asymmetric:
  - TLS handshake
  - Uses the key pair
  - Create and encrypt the new symmetric key or token that will be used for that communication channel
- Symmetric:
  - Once established the symmetric key during the TLS handshake, it's used to encrypt all communication in both directions

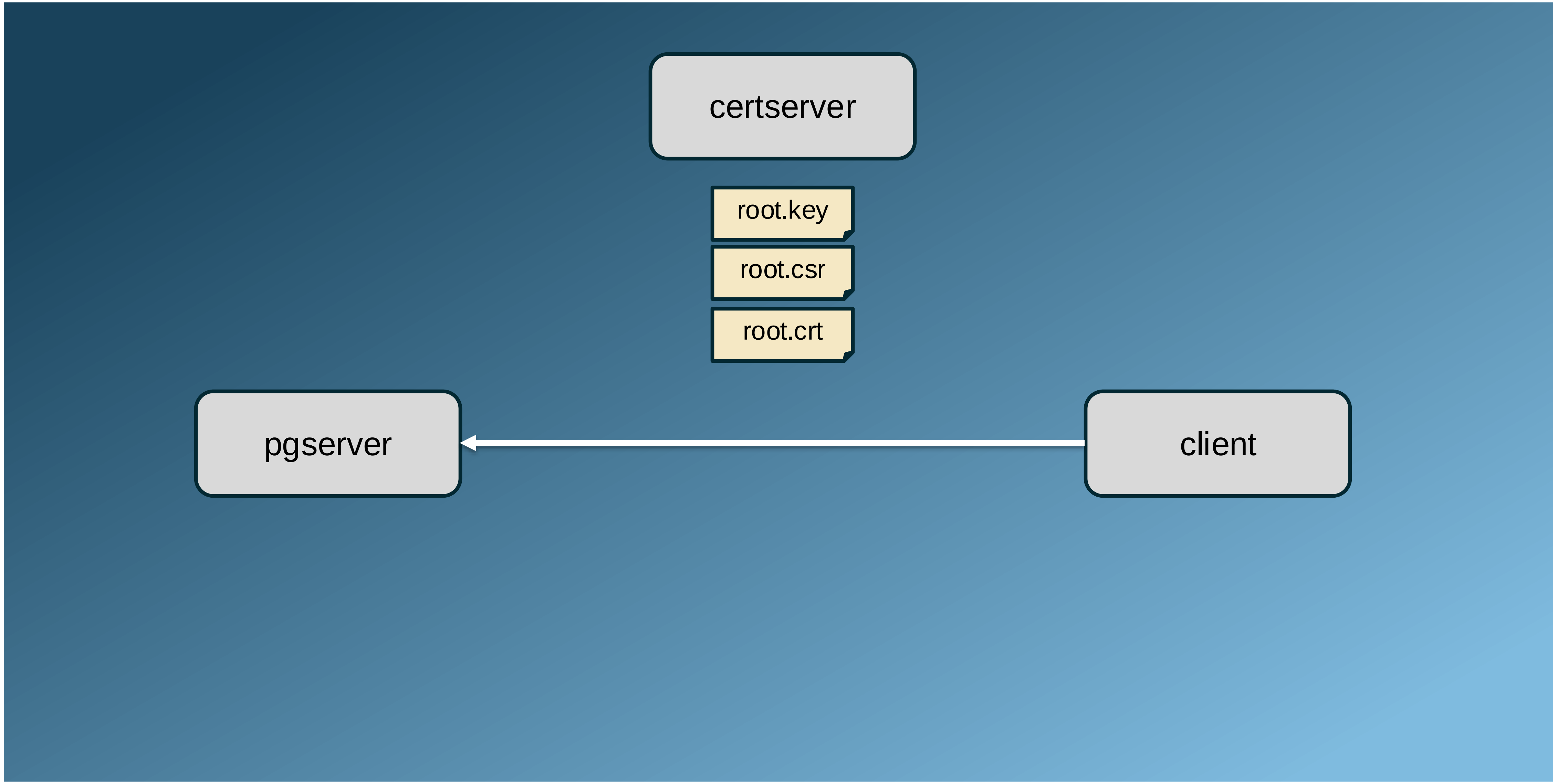
2- SERVER CERTIFICATES



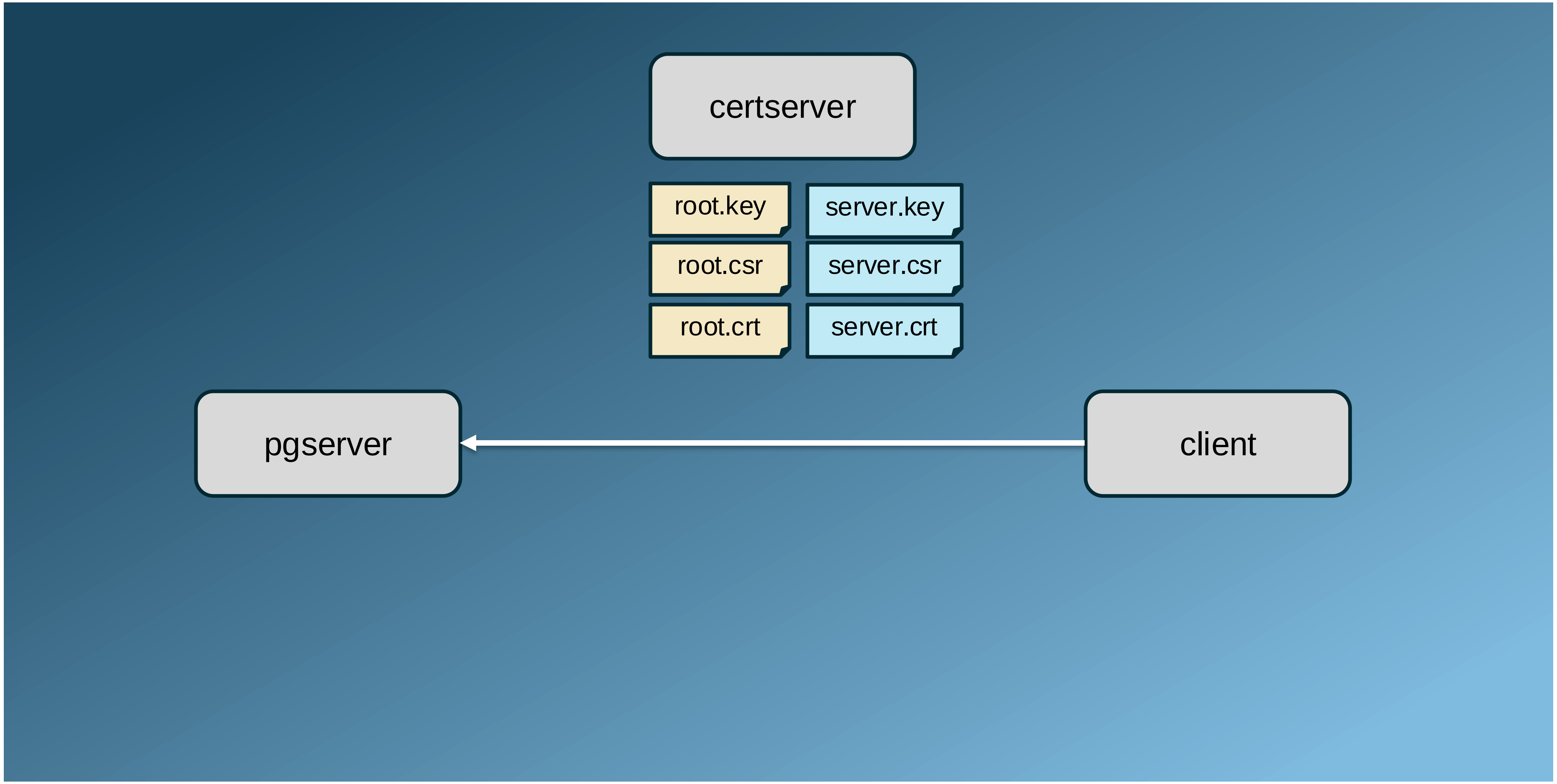
2- SERVER CERTIFICATES



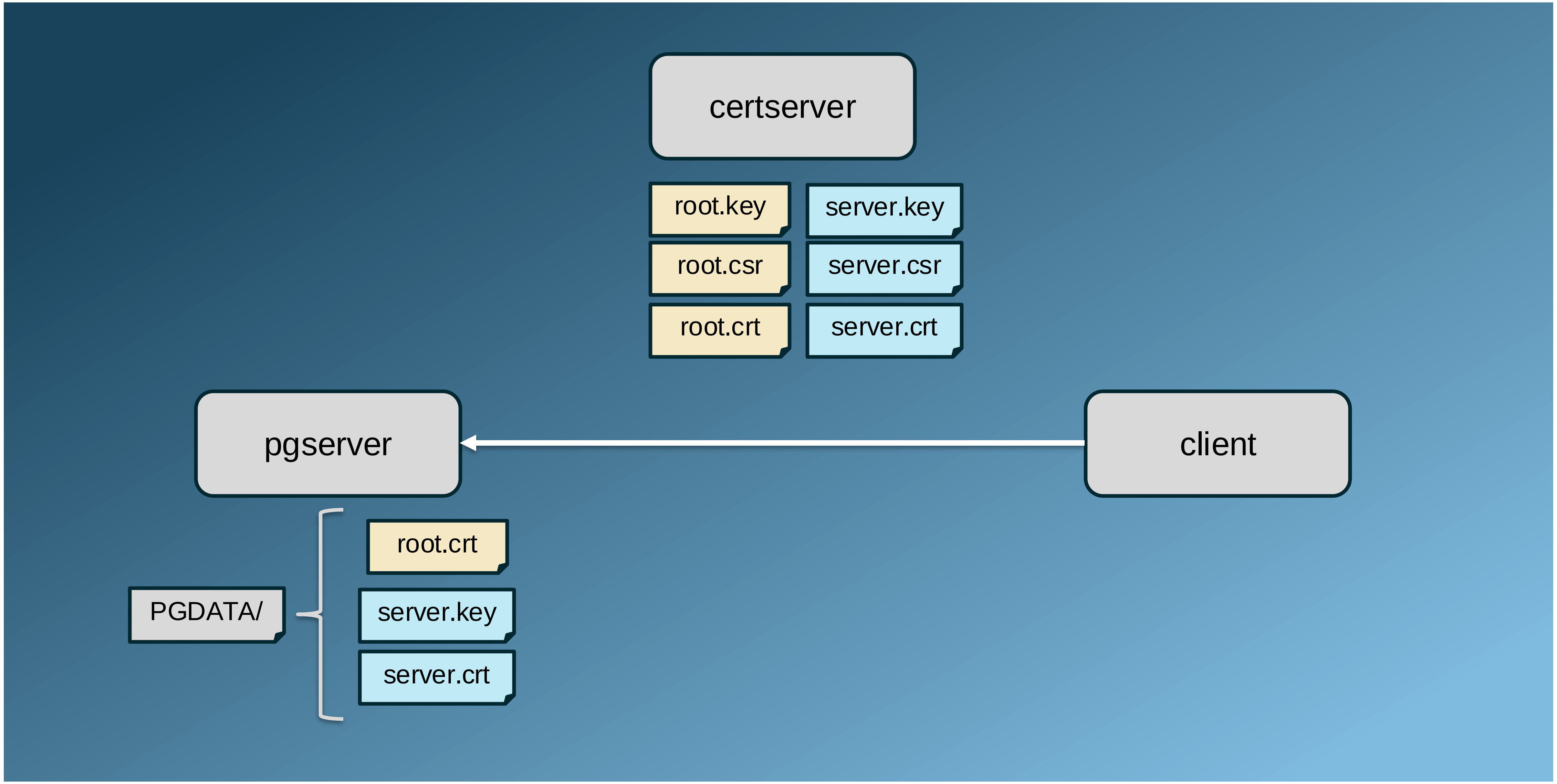
2- SERVER CERTIFICATES



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
```

Restart 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

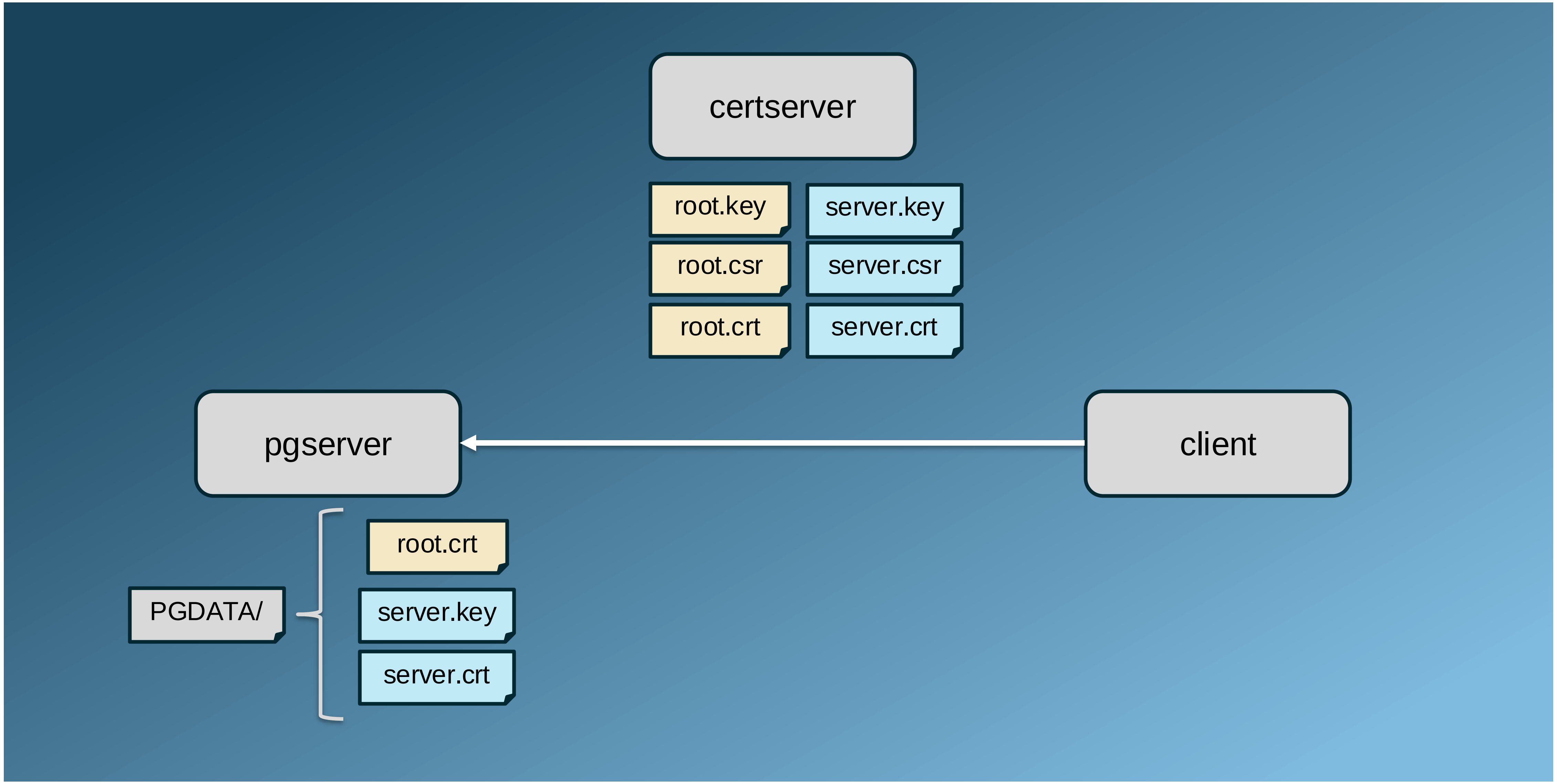
```
postgres=# SELECT
  a.client_addr, a.datname, a.usename,
  s.ssl, s.version, s.bits
FROM pg_stat_ssl s
JOIN pg_stat_activity a ON s.pid = a.pid;
 client_addr | datname  |  usename  | ssl | version | bits
-------------+----------+-----------+-----+---------+------
 172.18.0.22 | myappdb  | myappuser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

sslmode	Description
disable	Use only unencrypted connections.
allow	Try an unencrypted connection. If it fails, try an encrypted connection.
prefer	(**Default**) Try an encrypted connection. If it fails, try an unencrypted connection.
require	Use only encrypted connections. If a root certificate is available on the client, validate it against the server certificate.
verify-ca	Requires a root certificate available on the client, which will be validated against the server certificate. If the validation fails, the connection is not allowed.
verify-full	Same as **verify-ca**, but also validates the **host** attribute of the connection string against the **CN** (**Common Name**) of the server certificate.

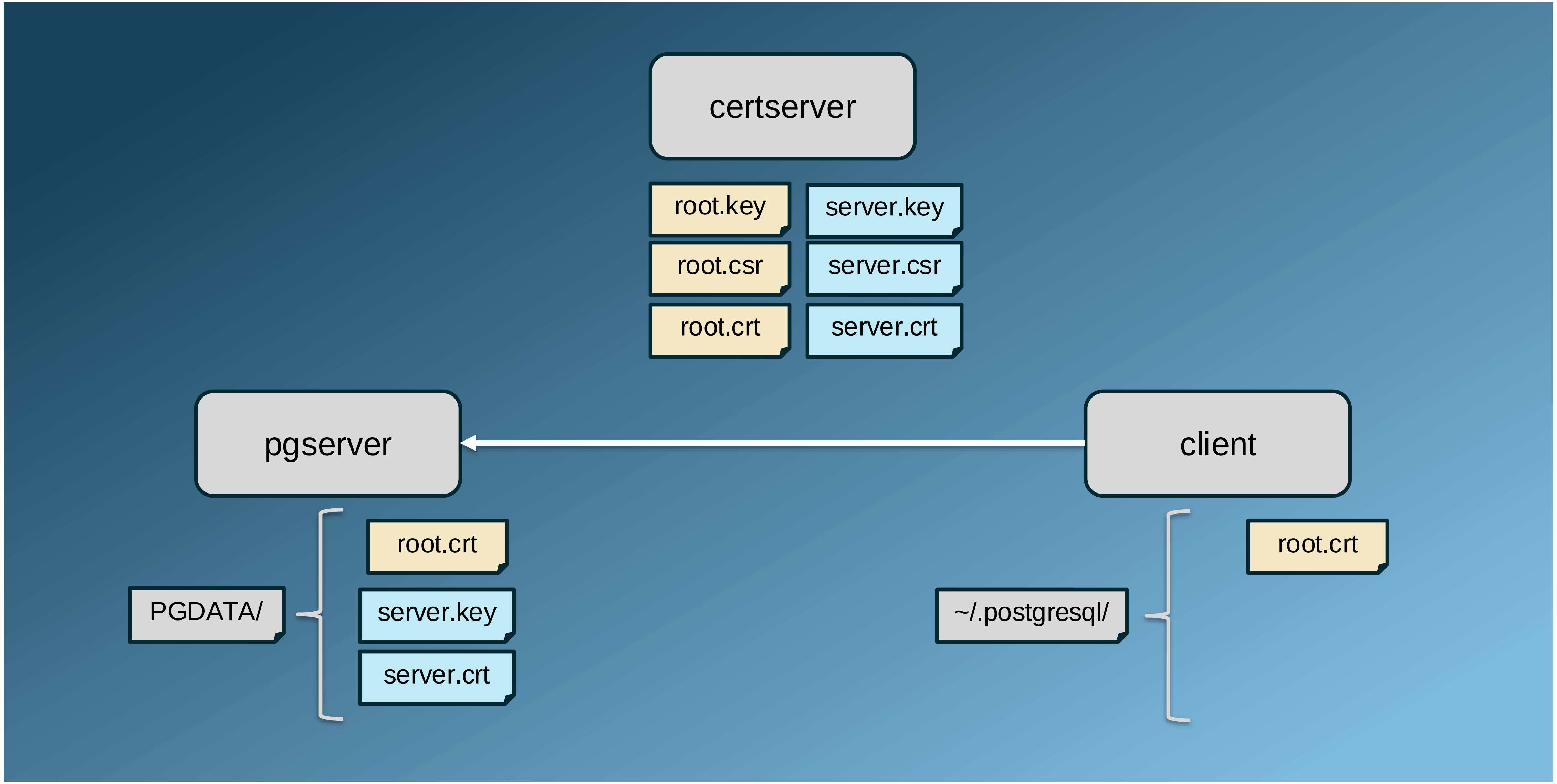
2- Server certificates (verify-ca)

```
[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)



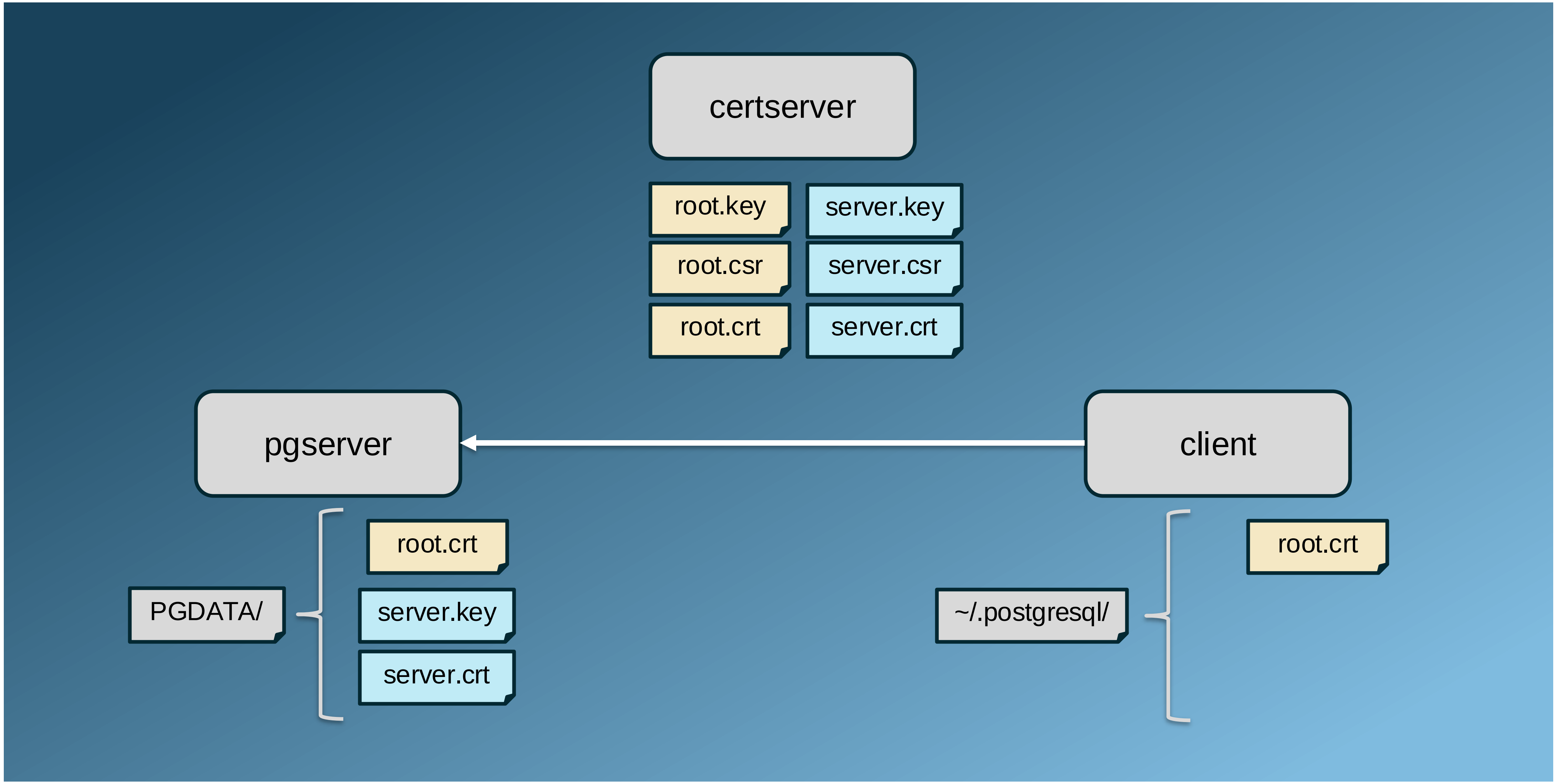
2- Server certificates (verify-ca)

```
[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=>
```

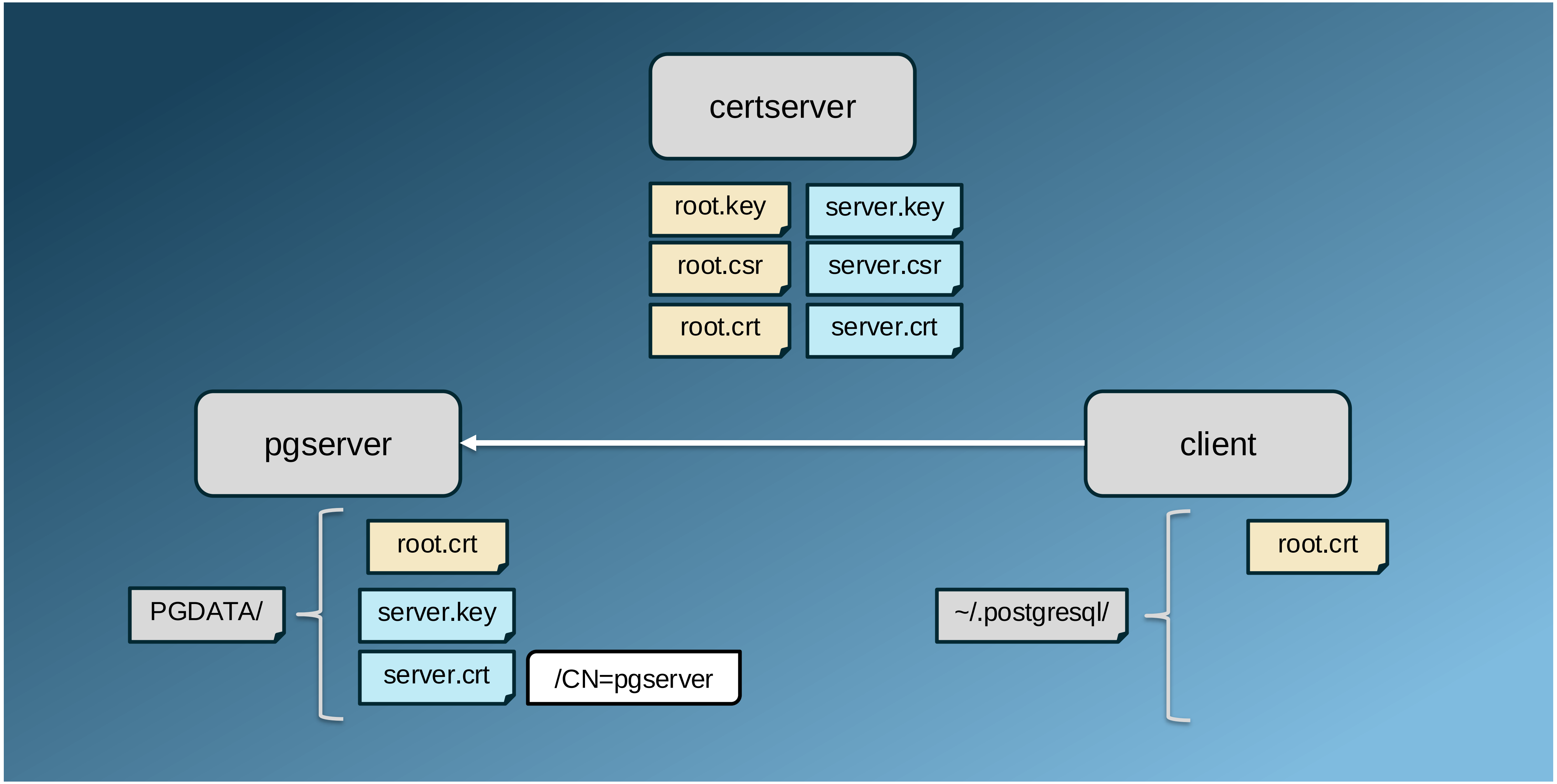
2- Server certificates (verify-ca)

```
[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)



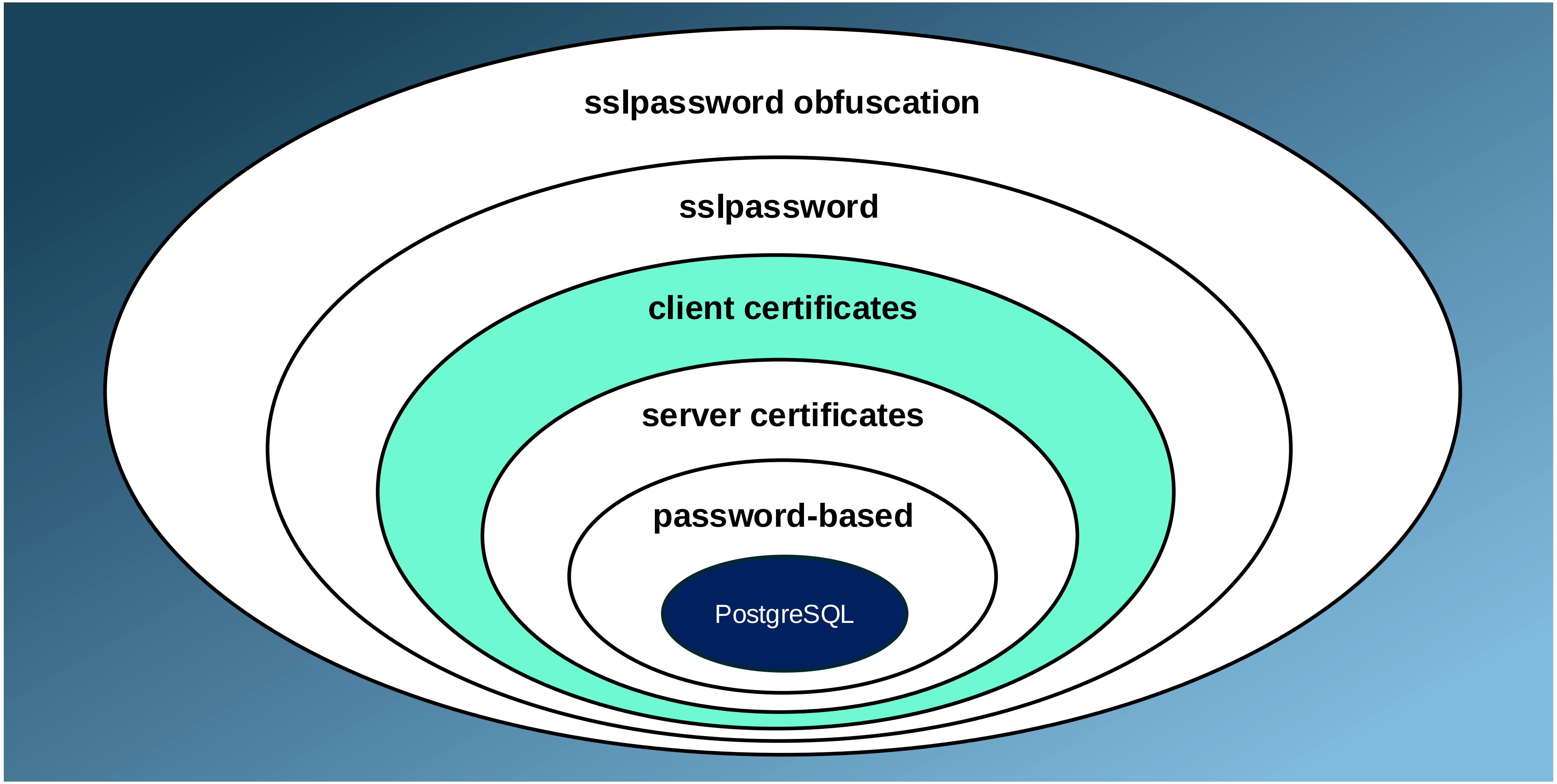
2- Server certificates (verify-full)

```
[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"
```

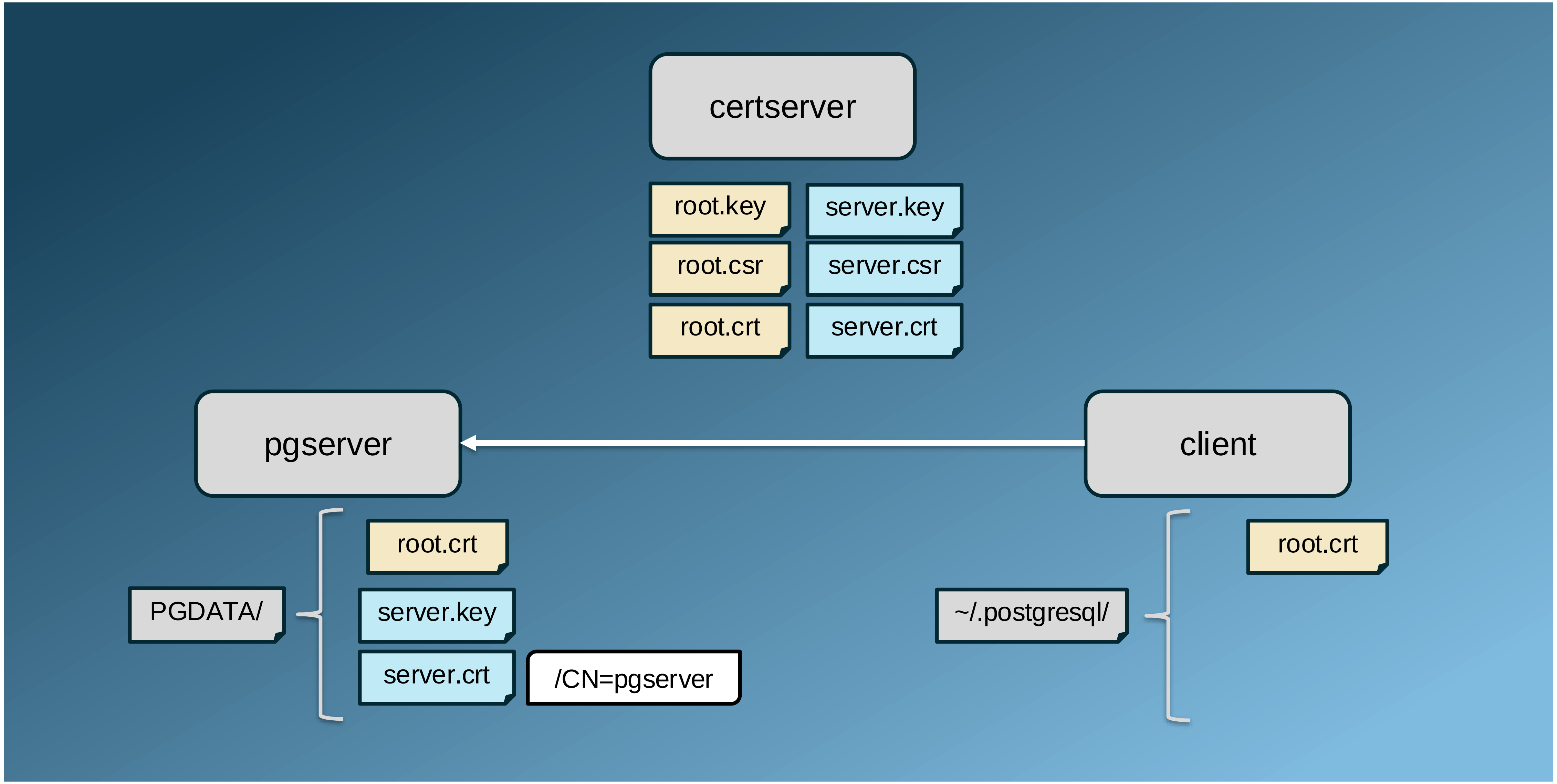
2- Server certificates (verify-full)

```
[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=>
```

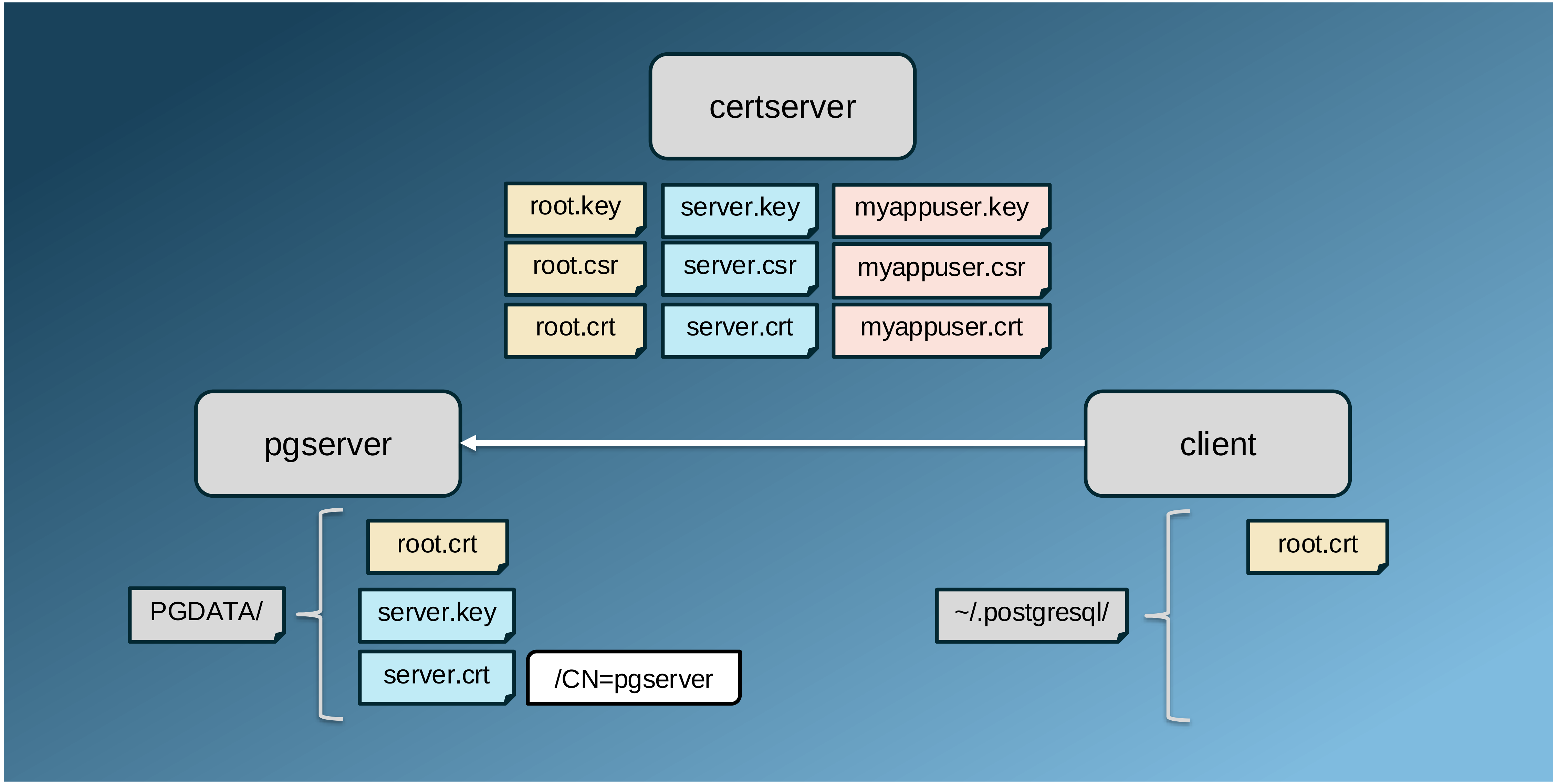
3- CLIENT CERTIFICATES



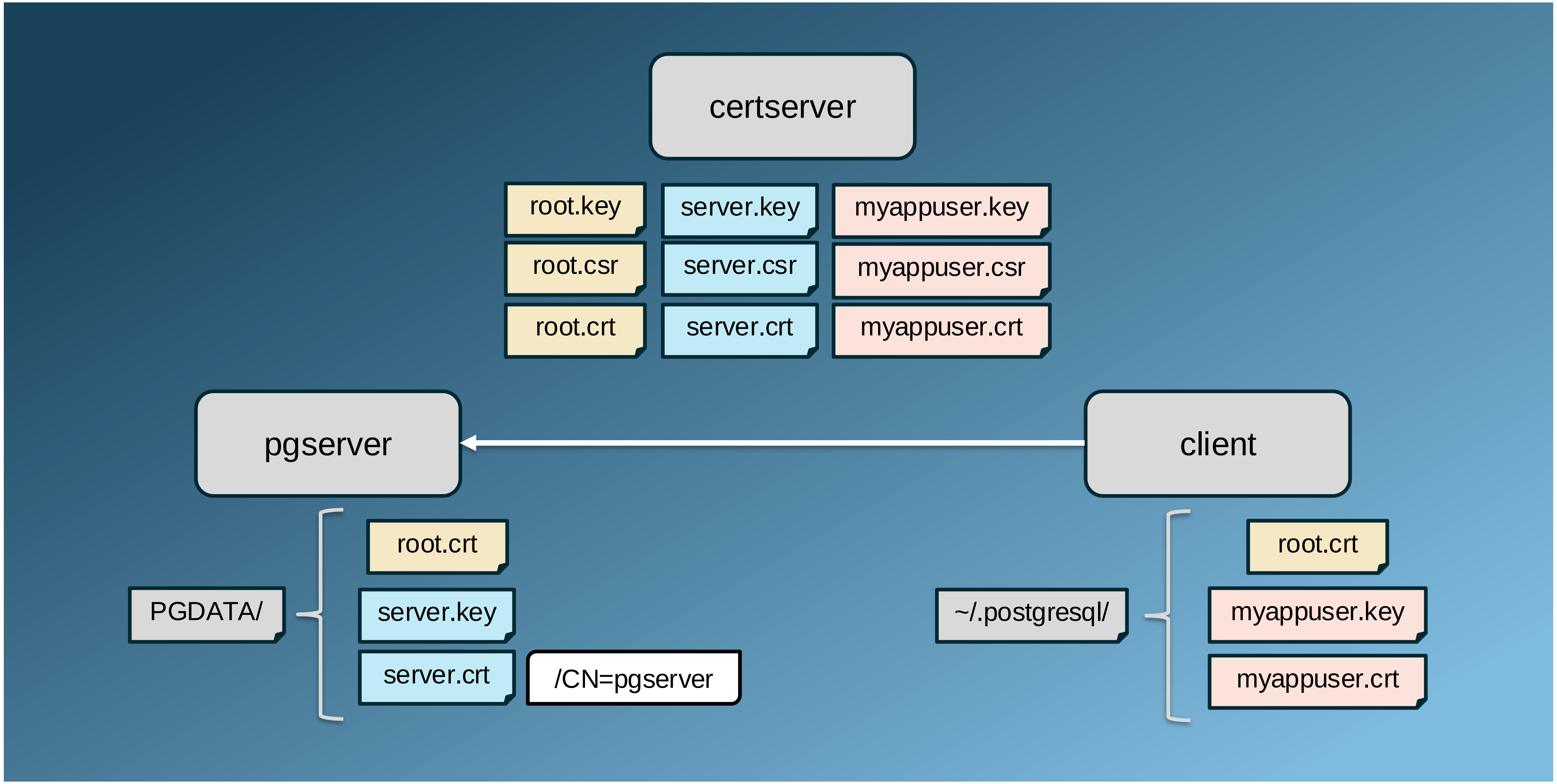
3- CLIENT CERTIFICATES



3- CLIENT CERTIFICATES



3- CLIENT CERTIFICATES

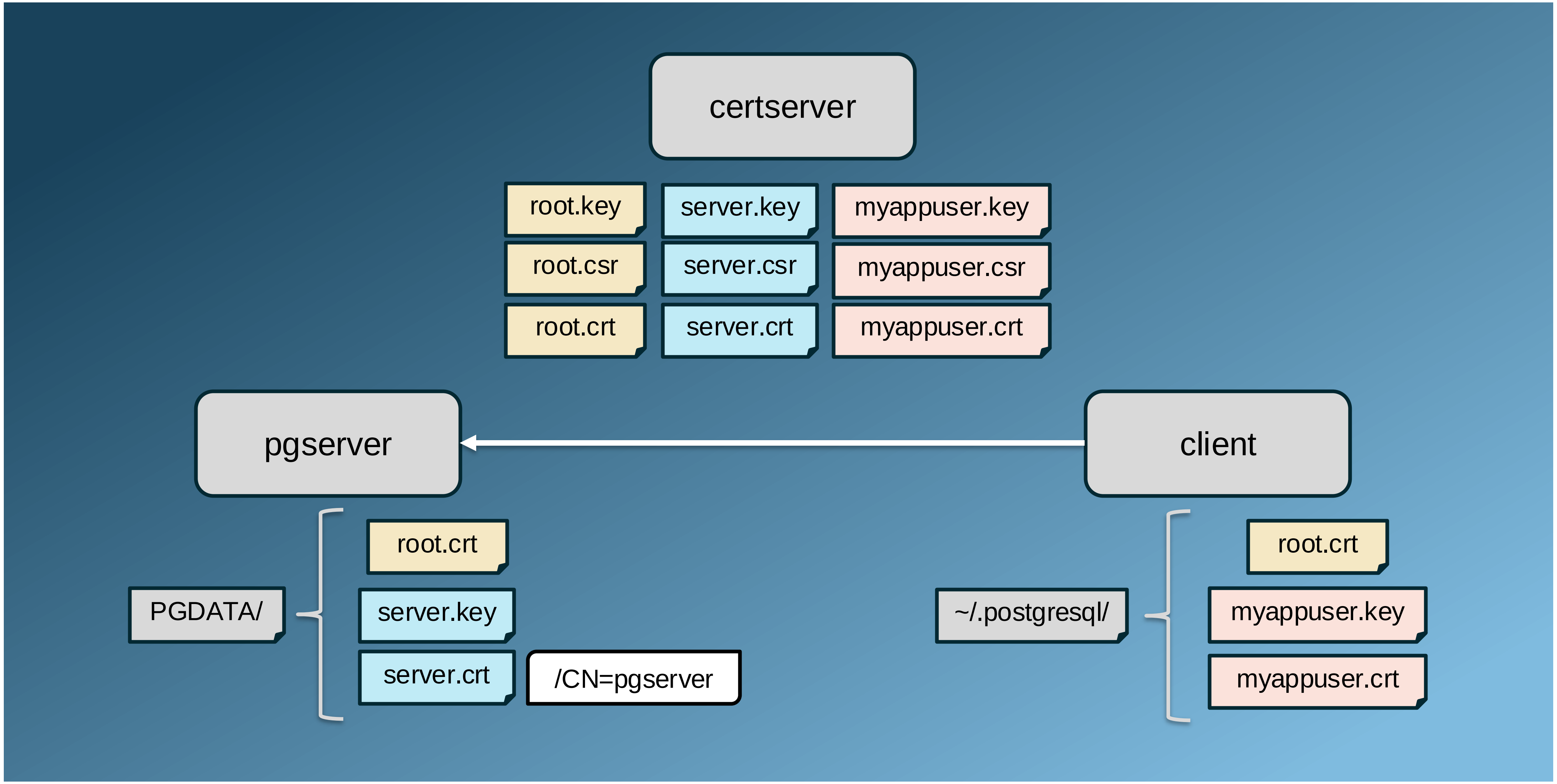


3- Client certificates

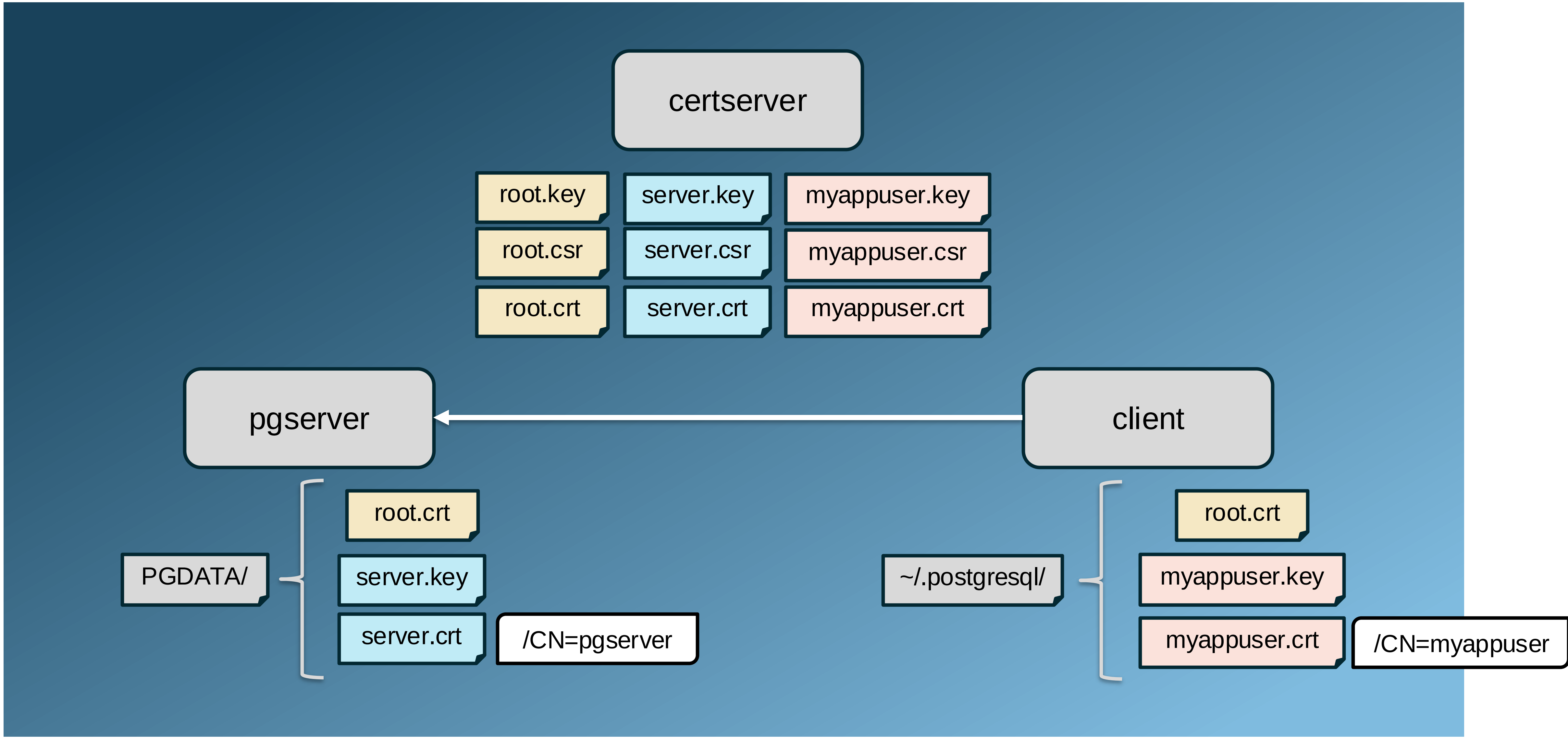
pg_hba.conf:

```
-- Postgres validates the ssl_ca_file against the root CA of the client certificate
hostssl myappdb myappuser 172.18.0.22/32 scram-sha-256 clientcert=verify-ca

-- Postgres also validates the CN of the client certificate against the user name
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=>
```

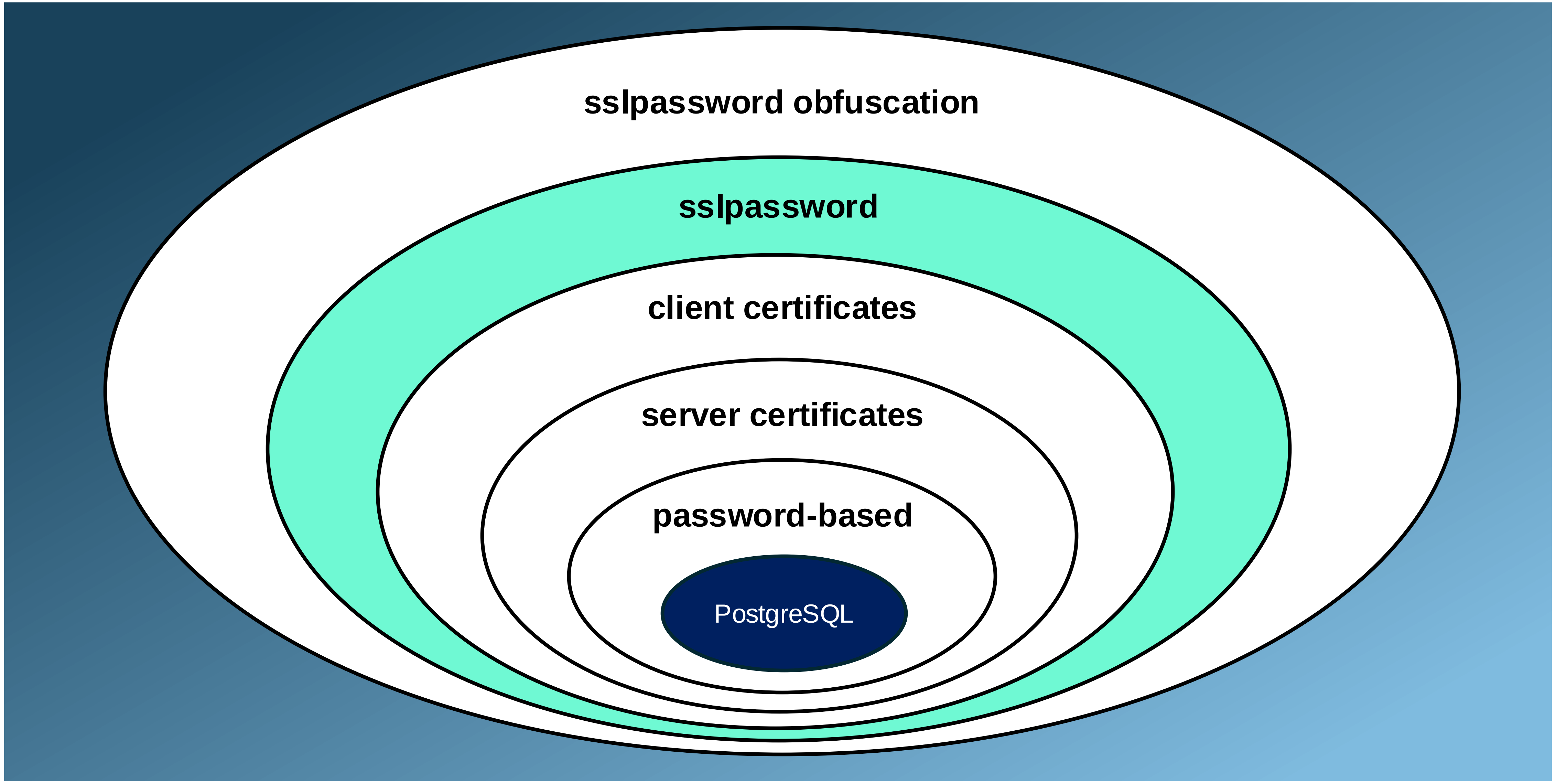
3- Client certificates

```
[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:

```
-- Authentication method "cert"
hostssl myappdb myappuser 172.18.0.22/32 cert

-- Actually is the same as:
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
```

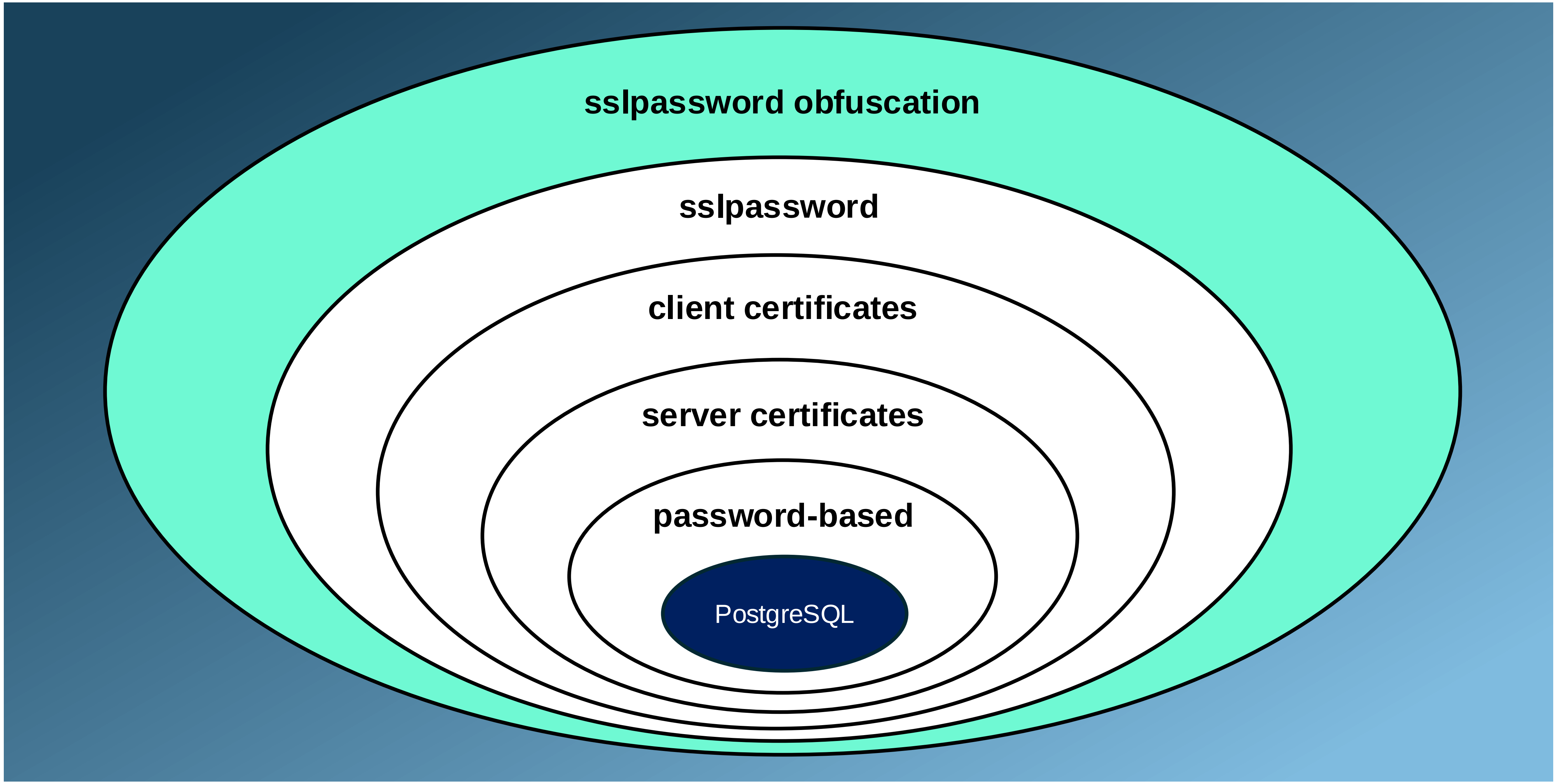
4- sslpassword

```
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
What’s this password for?	The database user.	The encrypted SSL key for connection using a client certificate.
Interactive mode	**Password for user XXX:**	**Enter PEM pass phrase:**
Environment variable	**PGPASSWORD**	N/A
Password file	**~/.pgpass**	N/A
Can it be used in ~/.pg_service.conf?	Yes	Yes

sslpassword obfuscation

5- SSLPASSWORD OBFUSCATION



5- sslpassword obfuscation

- There is no environment variable for **sslpassword**
- It's not always possible to use **~/.pg_service.conf** to hide the **sslpassword**
- **sslpassword** can be required in a shared connection string

- We already have security…
  - Rules in **pg_hba.conf**
  - Access to the client certificates

- … But we can make it even more difficult!

5- sslpassword obfuscation

- Strategy
  - Obfuscate the **sslpassword** in the connection string
  - Deobfuscate the **sslpassword** at the moment of the connection
- How
  - Hook **PQsetSSLKeyPassHook_OpenSSL**
  - Allows building a deobfuscation library to be loaded together with **libpq**
  - Customized function that runs at the moment of the connection
- Replace the **sslpassword** of the connection string with the real **sslpassword**

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) {
                // Real deobfuscation would happen here
                //
                free(obfuscated);
                strncpy(buf, deobfuscated, strlen(deobfuscated) + 1);
                return strlen(buf);
        }
        else {
                buf[0] = '\0';
                return 0;
        }
}
```

5- sslpassword obfuscation

- Never use a pure text password inside the library!

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

- Instead, implement or use a deobfuscation or decryption algorithm
- Other arguments of the connection string can be used for your algorithm to find the real password

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

Repository and Contact

- Slides of this presentation in PDF
- Step by step with all commands and explanations
- **openssl** commands to create the certificatesCódigo-fonte **libpqdeobfuscate.c**
- Rocky Linux 9 / PostgreSQL 17
- https://github.com/wind39/libpqdeobfuscate
- william.ivanski@enterprisedb.com