数据安全演进之路

--从TDE到FDE

王秀敏
公司：瀚高基础软件股份有限公司

瀚高 HIGH GO
开源数据库国产化先行者

数据库中最重要的部分

数据文件

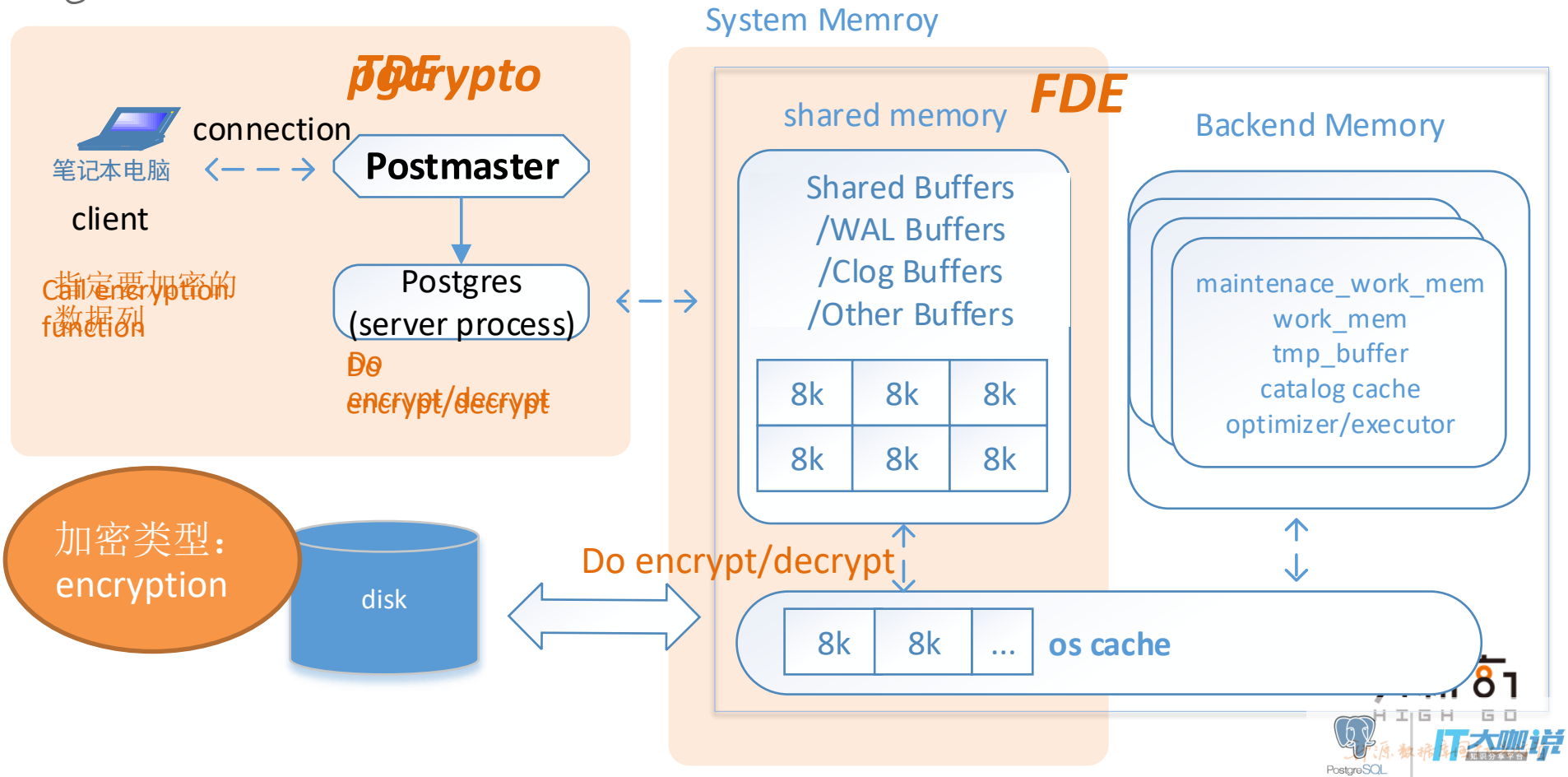
PostgreSQL数据库当前主要的安全措施

访问控制

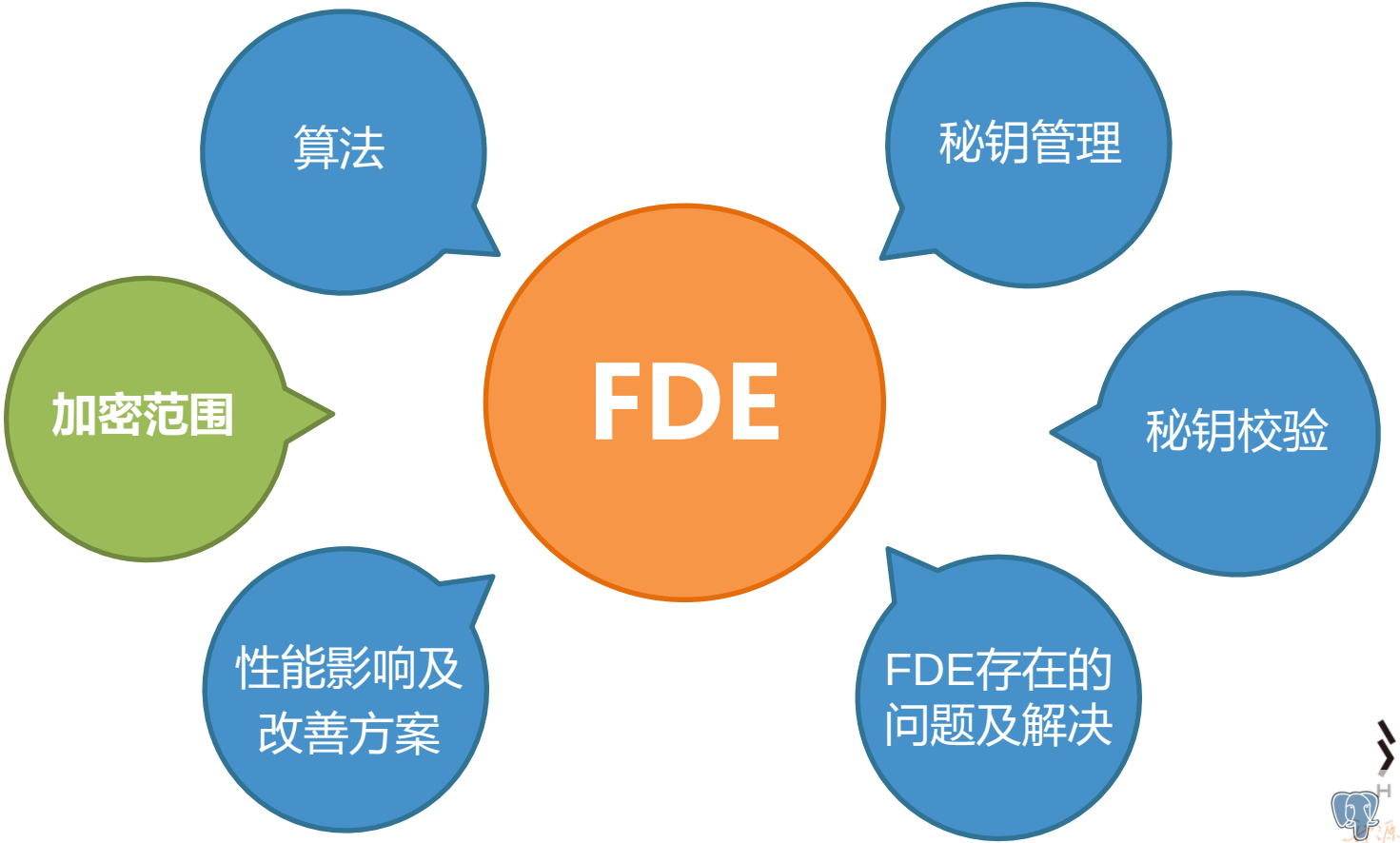
传统行业、政府部门
对数据安全很敏感

- 某政府项目

数据文件的保护
—— 数据加密

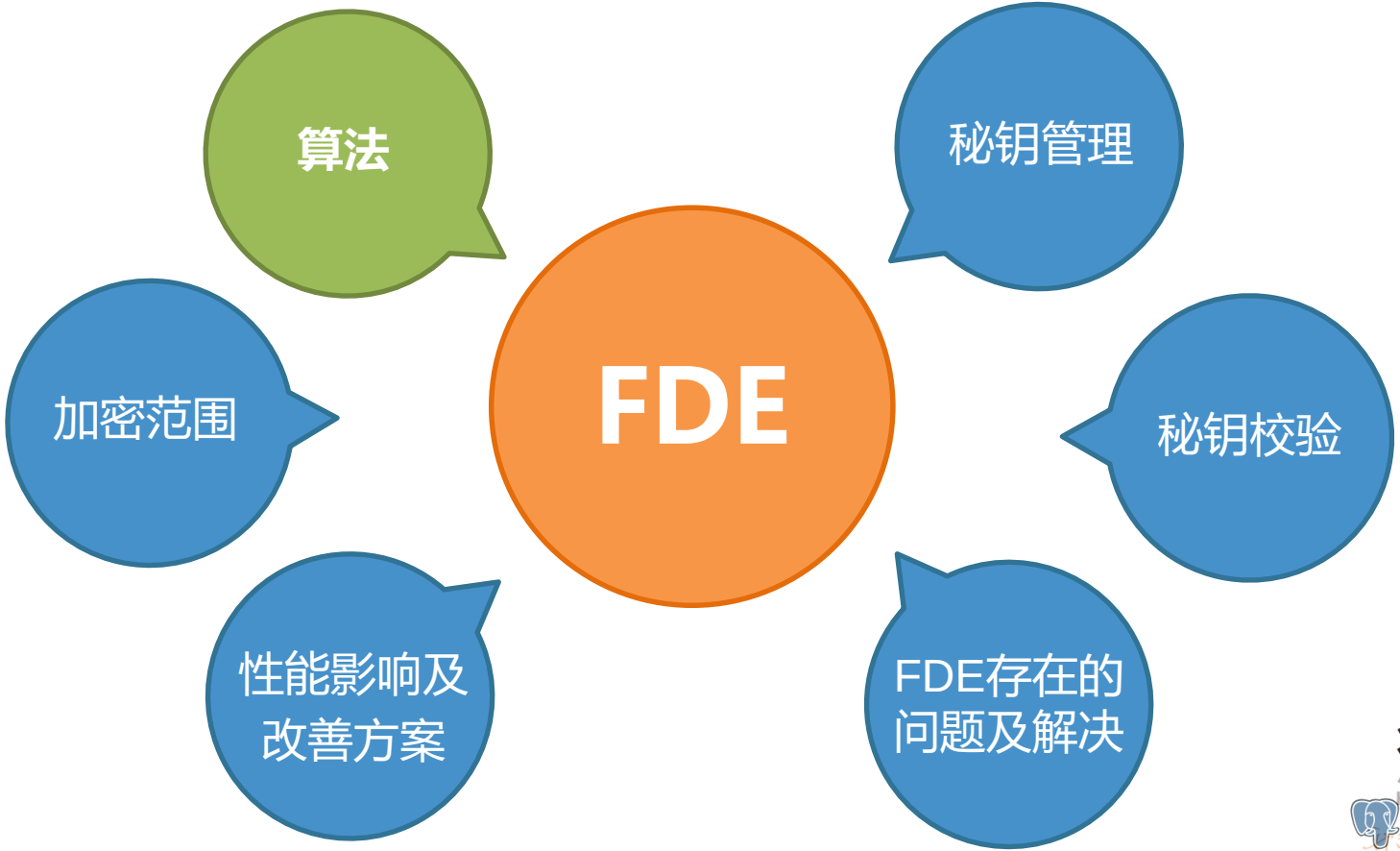

System Memroy
pgcrypto
TDE
connection
笔记本电脑
client
Postmaster
Postgres
(server process)
指定要加密的
数据列
Call encryption
function
Do
encrypt/decrypt
FDE
shared memory
Shared Buffers
/WAL Buffers
/Clog Buffers
/Other Buffers
8k 8k 8k
8k 8k 8k
Backend Memory
maintenace_work_mem
work_mem
tmp_buffer
catalog cache
optimizer/executor
加密类型:
encryption
disk
Do encrypt/decrypt
8k 8k ... os cache

	使用	加密内容	索引	性能影响
pgcrypto	需要手动调用加密函数	指定的数据	不支持	小
TDE	需要指定加密列	指定的列	不支持	小
FDE	初始化指定是否加密	磁盘文件	支持	较大


FDE
算法
秘钥管理
加密范围
秘钥校验
性能影响及
改善方案
FDE存在的
问题及解决

FDE加密范围

- Relation：table Index Sequences
- Visibility maps
  free space maps
- Write Ahead Log
- SLRUs: Clog commit_ts multixact subtrans notify
- temporary files for query execution

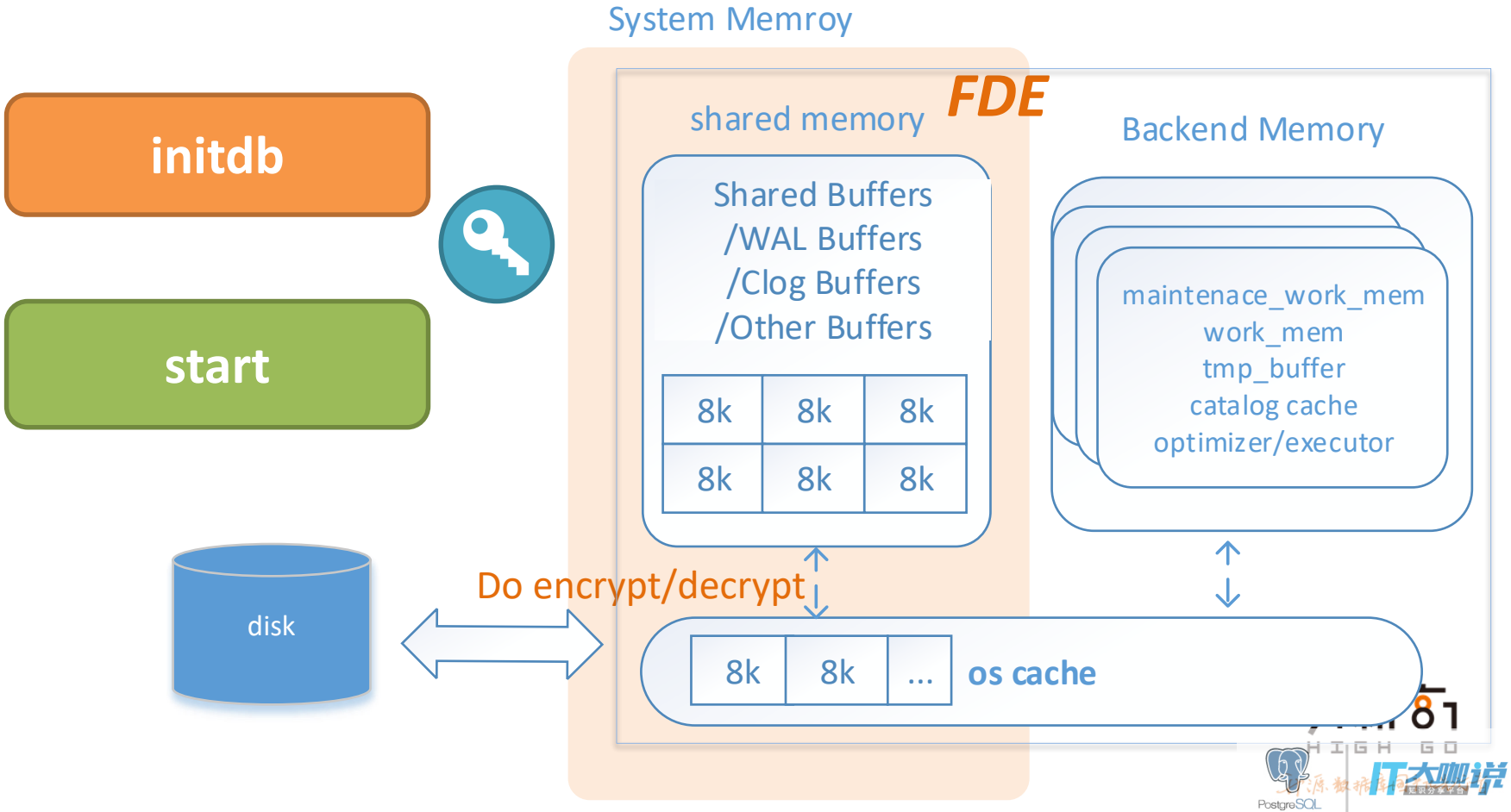

FDE
算法
秘钥管理
加密范围
秘钥校验
性能影响及
改善方案
FDE存在的
问题及解决

对称加密算法：
相对于非对称加密算法，速度快

Aes-128

aes-128、aes-192、aes-256、blowfish、des、3des、cast5、sm1、sm4


算法
秘钥管理
加密范围
FDE
秘钥校验
性能影响及
改善方案
FDE存在的
问题及解决


System Memroy
FDE
initdb
start
shared memory
Shared Buffers
/WAL Buffers
/Clog Buffers
/Other Buffers
8k 8k 8k
8k 8k 8k
Backend Memory
maintenace_work_mem
work_mem
tmp_buffer
catalog cache
optimizer/executor
disk
Do encrypt/decrypt
8k 8k ... os cache

秘钥管理

环境变量

秘钥设定在环境变量中，初始化/启动时从环境变量中取得

- ✓ 需要人工记忆
- ✓ 安全性差

外部命令

初始化后，设定取得秘钥的命令，存储于postgresql.conf

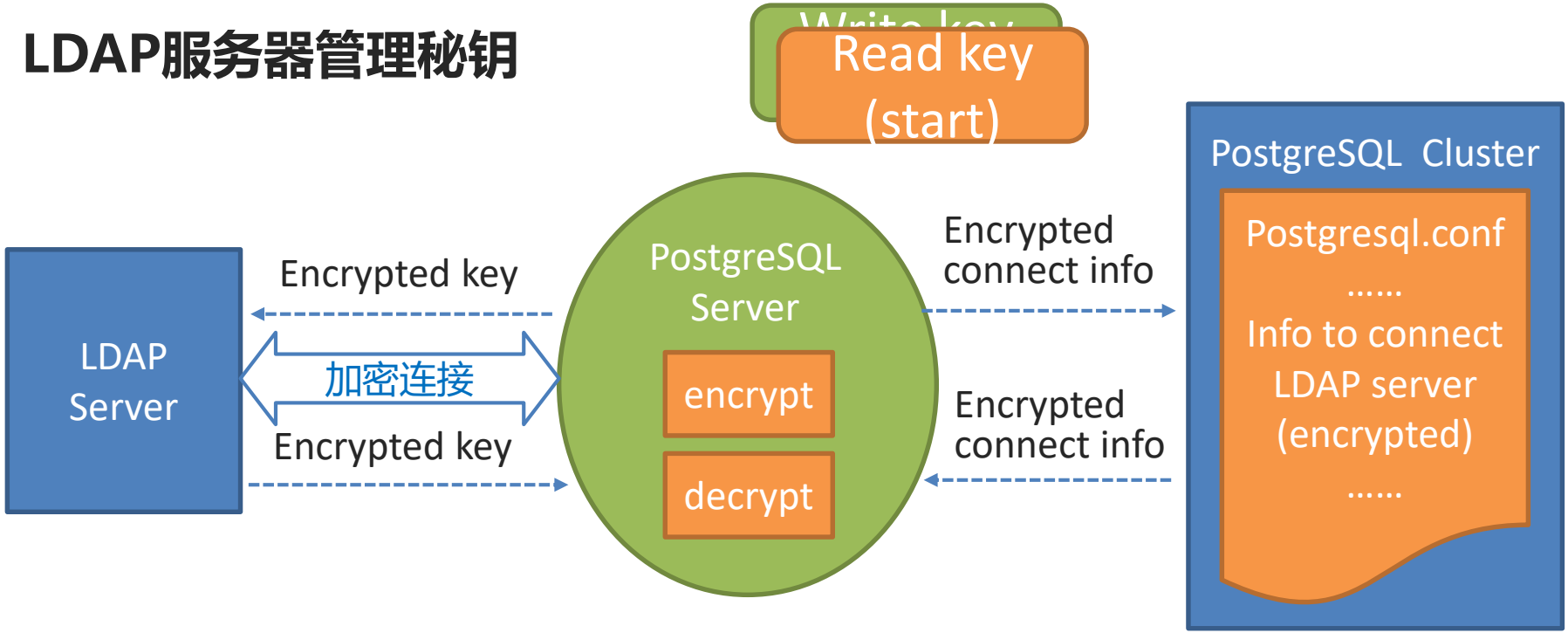
- ✓ 不需要人工记忆
- ✓ 命令存储于postgresql.conf秘钥易获得

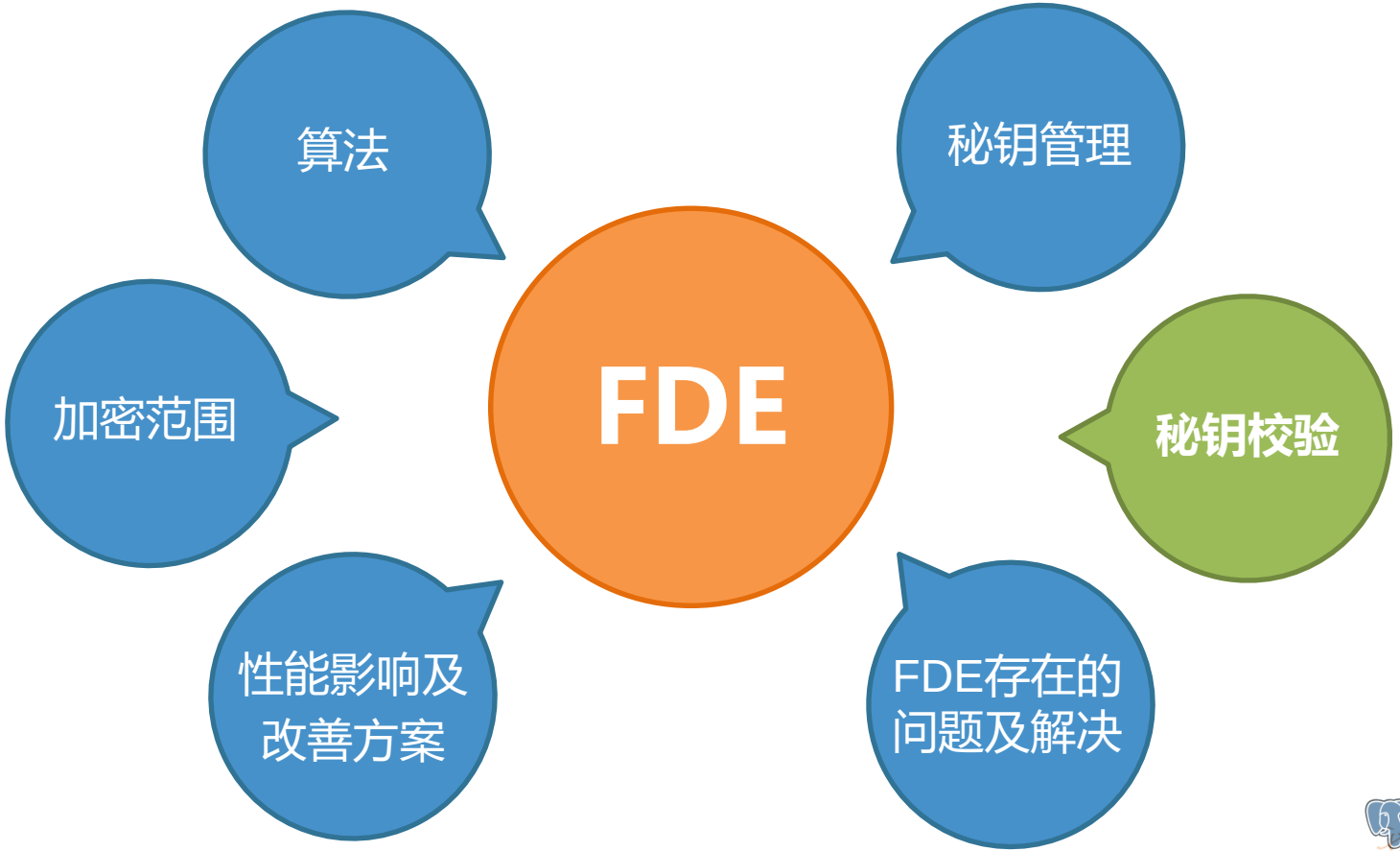
远程服务器

秘钥存储于远程服务器，访问服务器的信息加密存储于postgresql.conf

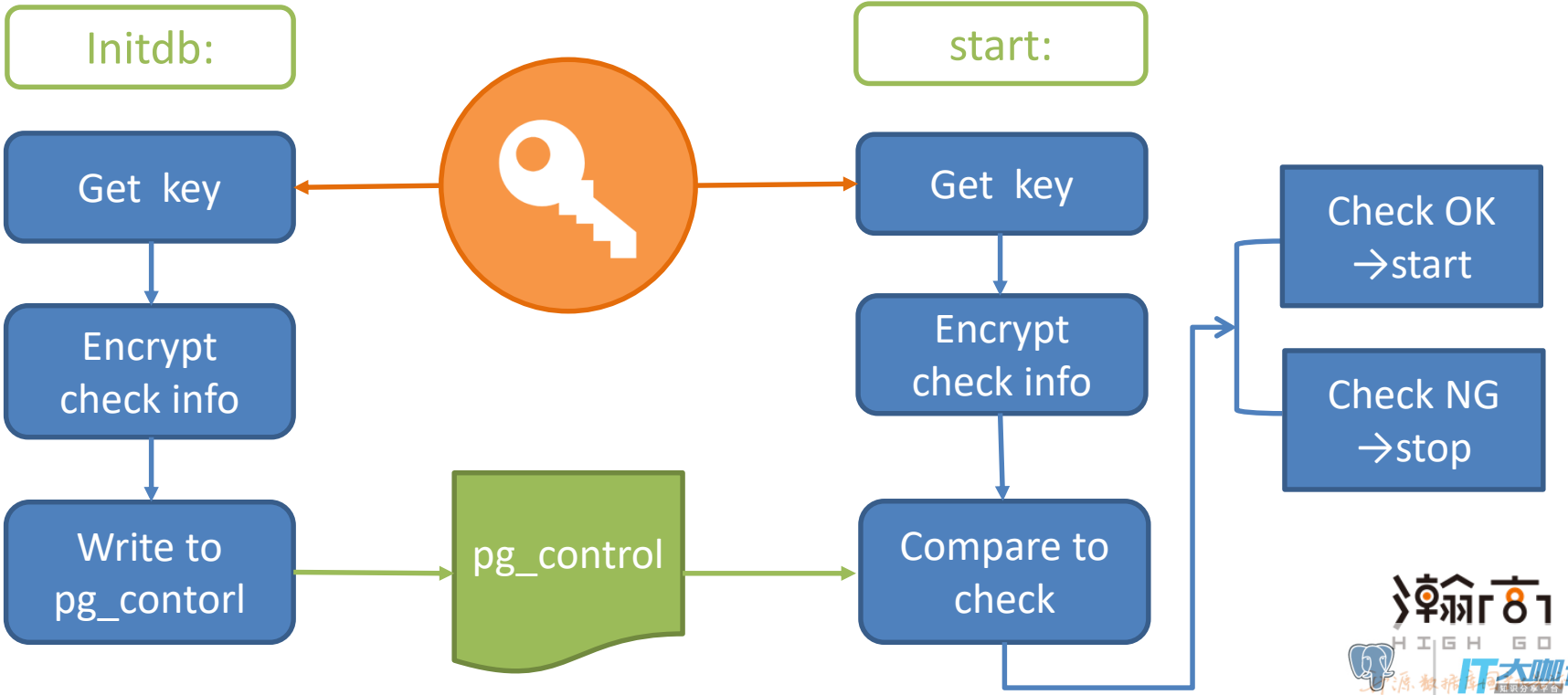
- ✓ 秘钥安全性高

LDAP服务器管理秘钥




算法
秘钥管理
加密范围
FDE
秘钥校验
性能影响及
改善方案
FDE存在的
问题及解决

秘钥校验


Initdb:
start:
Get key
Encrypt check info
Write to pg_contorl
pg_control
Get key
Encrypt check info
Compare to check
Check OK →start
Check NG →stop

流复制：

primary

record

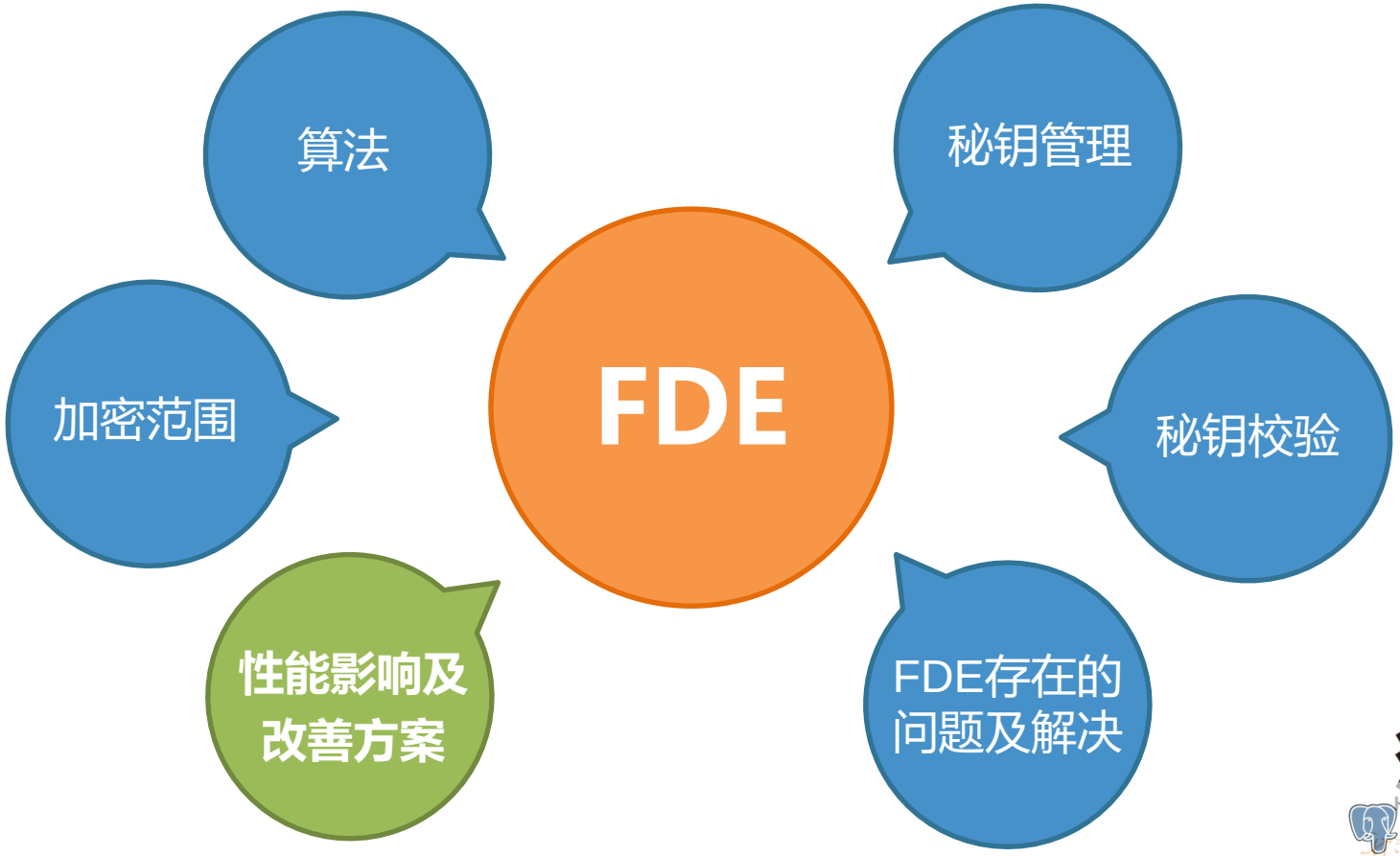
发送传输密文的密文

standby

FDE 以page(8k)为单位加密
流复制以record为单位传输

page：
Aes-128
加密分组

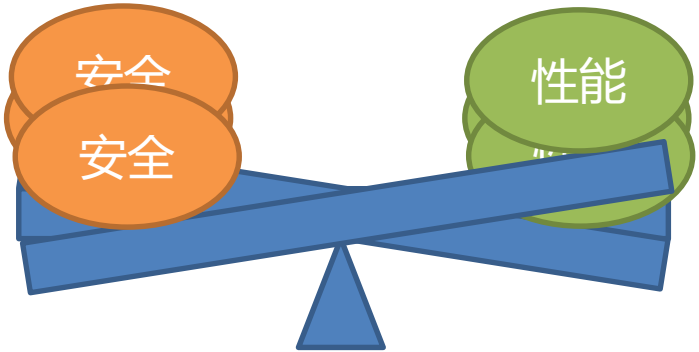
对称加密算法：
① 分组加密
② 各个分组独立
OR 依赖于前面的分组(加密模式)



FDE对性能的影响

算法	平均速度
不加密	440M/s
aes-128	89M/s
SM1	43M/s
SM4	35M/s

改善性能的方案 — 部分加密



- 选择加密部分文件，另外的文件不加密，提升数据库的性能
- 安全性下降

Relation&FSM&VM
Write Ahead Log
SLRU’s file
Temp file

改善性能的其他方案

硬件：

- ✓ 更大的内存，设定更大的sharebuffer,减少IO
- ✓ CPU 性能提高
- ✓ 使用GPU专门负责加解密计算
- ✓ 用PCIE加密卡进行加解密运算
- ✓ ……

软件：

- ✓ 使用性能好的加密算法
- ✓ 读写分离
- ✓ ……

Thanks!

瀚高 HIGH GO

开源数据库国产化先行者