Firebird Database Encryption Workshop

Alex Peshkoff, Firebird
Alexey Kovyazin, IBSurgeon

Agenda

1) Why Encryption?
2) How Encryption works
   1)On Server-Side
   2)On Client-Side
3) Installation and Configuration
4) Performance of encrypted databases
5) Real-world cases and real-world problems
   1)Windows CryptoAPI and in-place keys
   2)Multi-thread client applications

1. Why Firebird Encryption

1. Why Encryption

- Protect database from the physical stealing
- Protect database from the access from the applications without keys
- Protect databases with pre-filled data
- Protect metadata (stored procedures, triggers) with non-trivial logic
- Because government wants it

AES 256

When we don’t need encryption

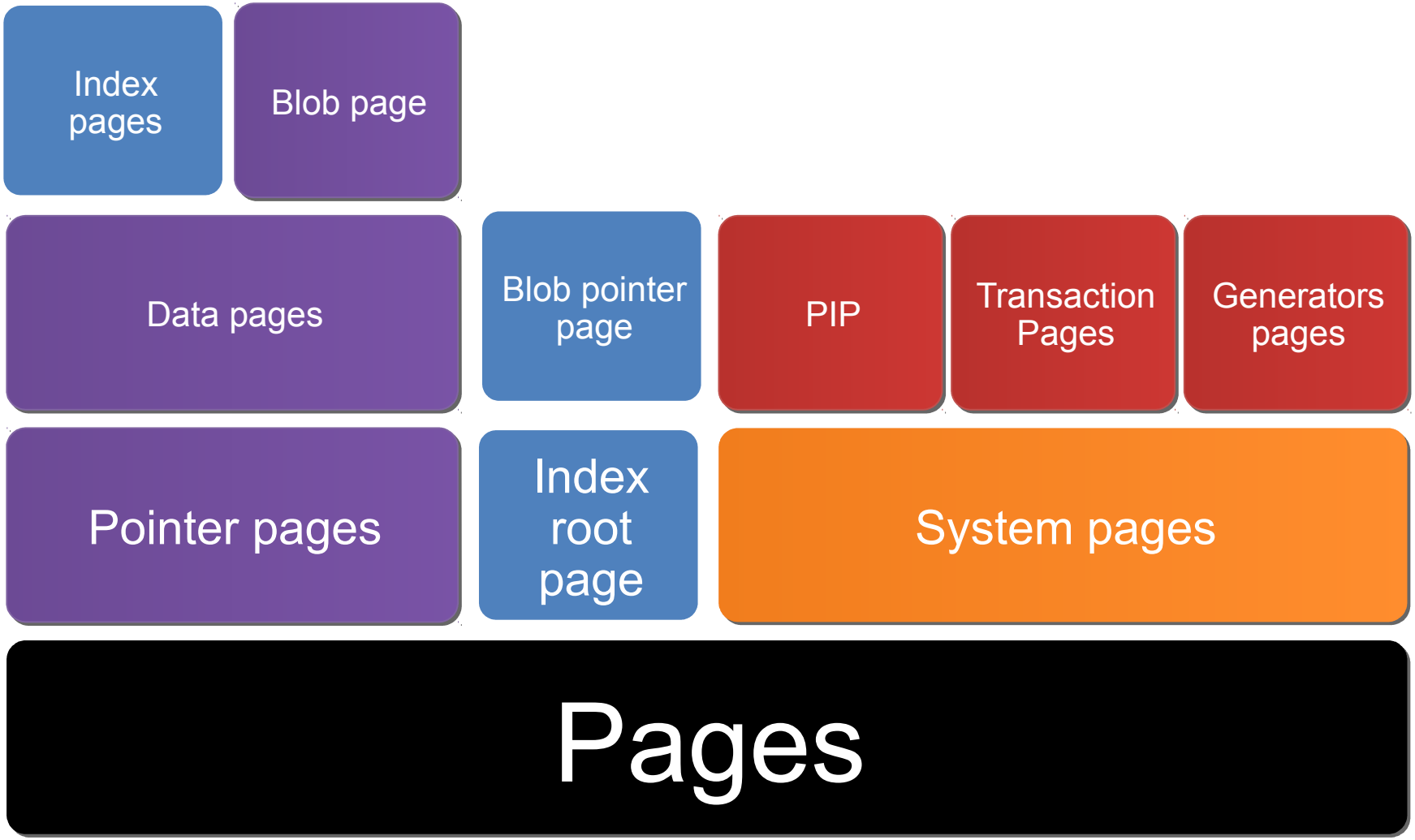
- Protect database from file copying
  - Adjust security settings on the server and in the network
- Restrict access to the specific database
  - Use separate security database, etc

2.1 How Firebird Encryption Works On the Server-Side

2.1. How Firebird Encryption Works On Server-Side

1)What part of a database is encrypted
2)When encryption happen?
3)How keys are transferred to plugin
4)DbCrypt plugin
5)DbCrypt and KeyHolder: key exchange details

What part of a database is encrypted?



Non-crypted DB: FirstAID data preview

irect Fix | Connect to DB | Create New DB | Run Script... | Disconnect | Export Structure | Extract All tables

External metadata | Exit

Quote Names | 32bit generators

Pages Summary | DDL | Preview Data | Indices and Constraints | Generators | DirectFix

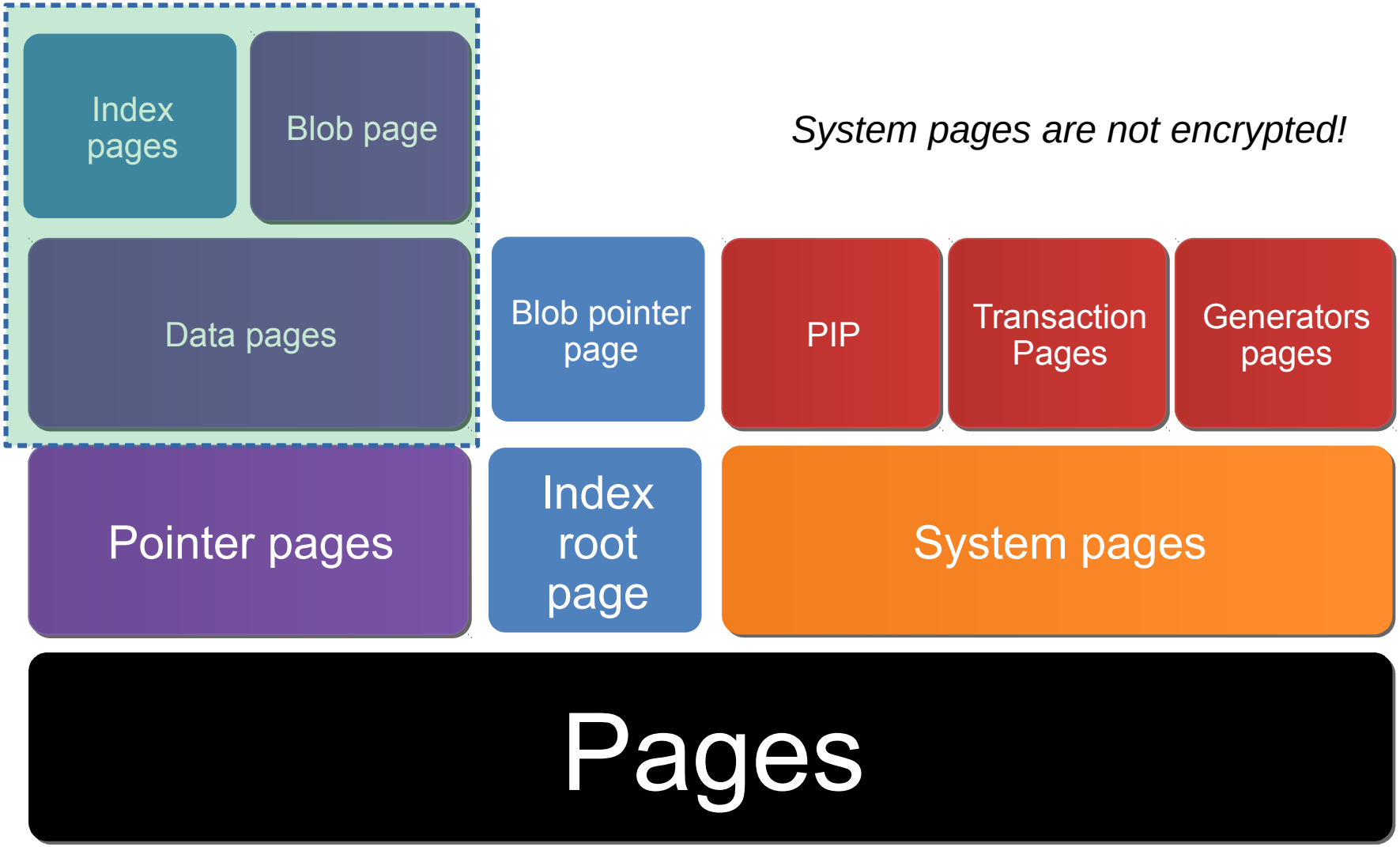
Data page pos #: 1 | Preview

Rec #	ID	UNIT	EX...	FB_GDS...	T...	...		INFO
1	255	v_all_customers	NULL	NULL	186	40	0	id_min=2.500000000000000
2	256	v_all_customers	NULL	NULL	186	40	0	id_max=50.50000000000000
3	257	sp_fill_shopping_cart	NULL	NULL	186	40	0	view=v_all_wares, rows=4, oper=1000
4	258	v_all_wares	NULL	NULL	186	40	0	id_min=0.5000000000000000
5	259	v_all_wares	NULL	NULL	186	40	0	id_max=400.5000000000000
6	260	doc_list_biud	NULL	NULL	186	40	0	dh=330, op=INS new=1000; new.id=330, acn_typ
7	261	doc_list_aiud	NULL	NULL	186	40	0	dh=330, op=INS new=1000
8	262	sp_multiply_rows_for_qdistr	NULL	NULL	186	40	0	dh=330, q_sum=25.000
9	263	doc_list_biud	NULL	NULL	186	40	0	dh=330, op=UPD old=1000 new=1000;
10	264	doc_list_aiud	NULL	NULL	186	40	0	dh=330, op=UPD old=1000 new=1000
11	265	srv_find_qd_qs_mism	NULL	NULL	186	40	0	ok, dh=330, op=1000, sum_qty=25, cnt_qds=25, r
12	266	t$perf-norm:sp_client_order	NULL	NULL	186	40	0	ok saved 12 rows
13	337	t$perf-abend:sp_reserve_write_off	NULL	NULL	187	40	0	gds=335544517, autonomous Tx: 3 rows
14	274	sp_reserve_write_off	NULL	335544517	187	40	0	

Charset: ANSI_CHARSET

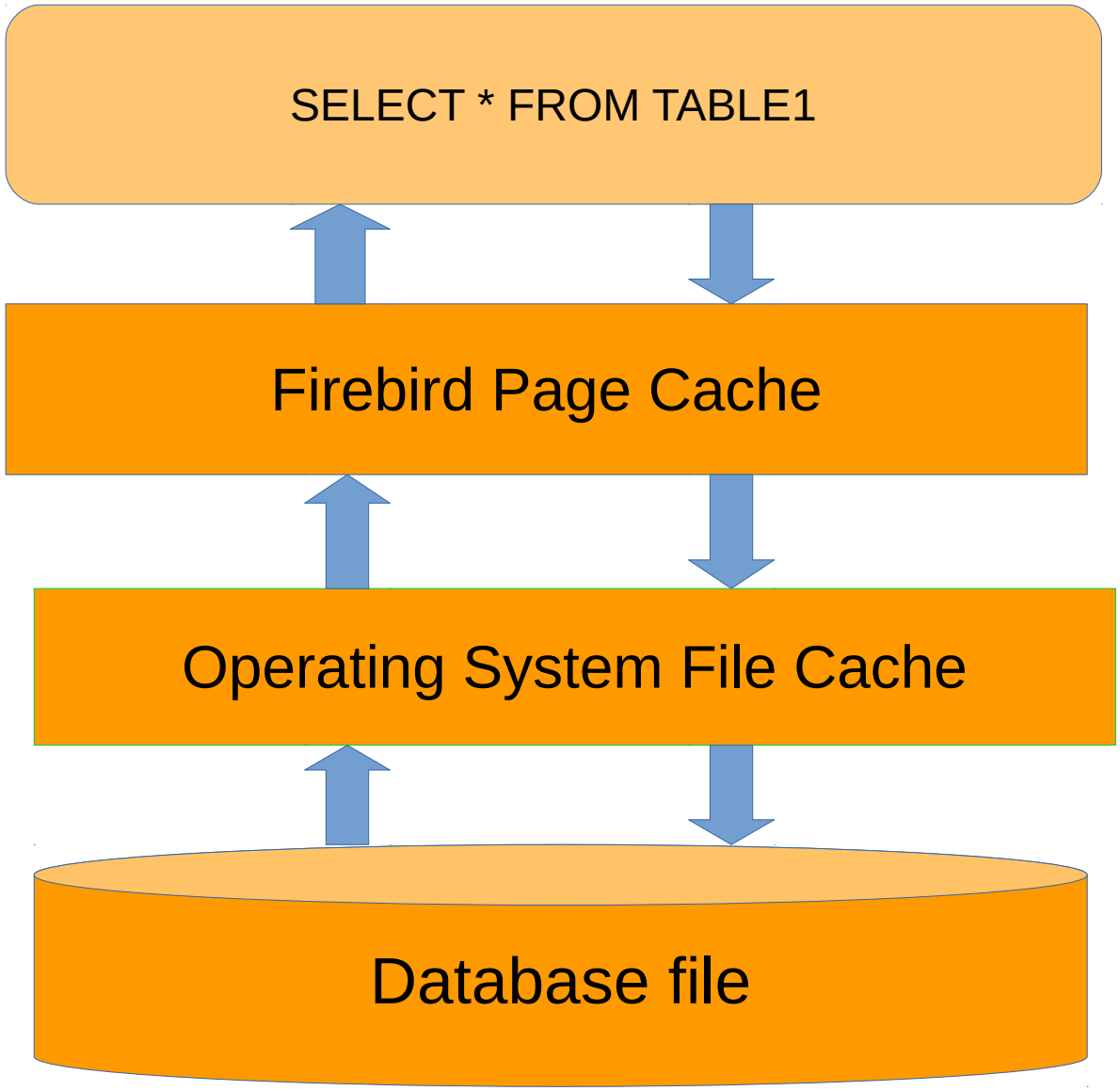
Relation 163: Datapage #291; Seq #1

Only pages with users data encrypted

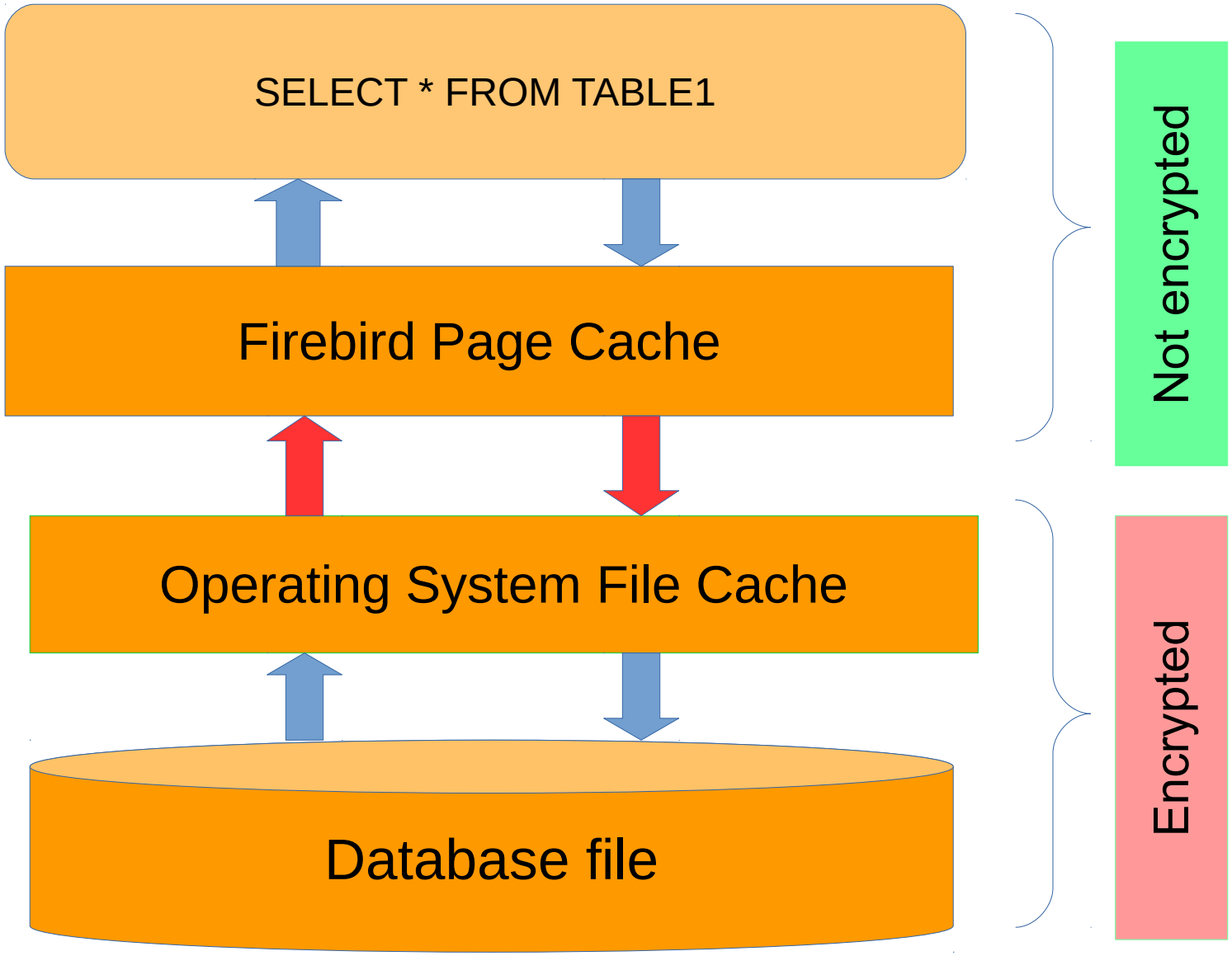


System pages are not encrypted!

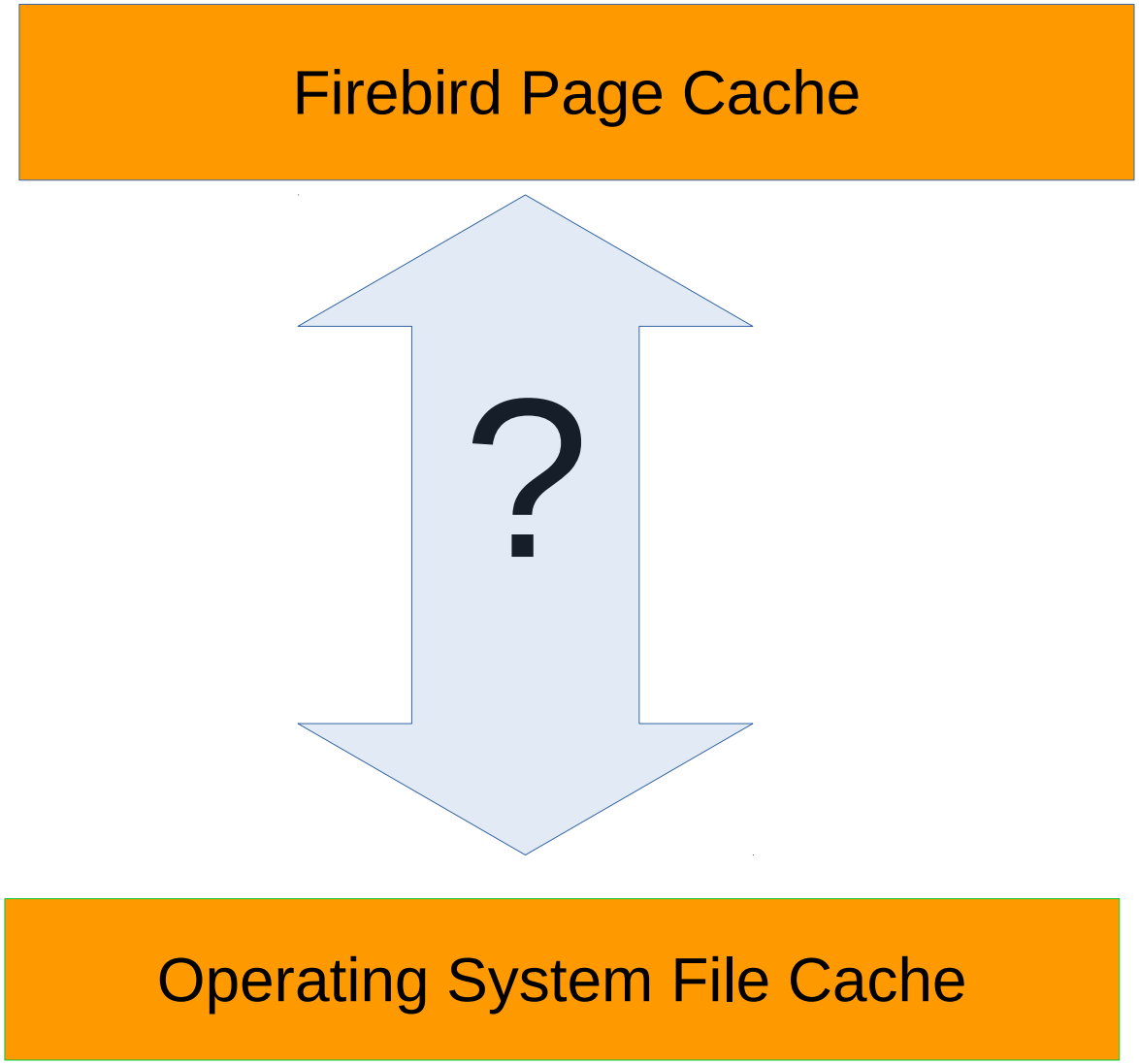
When data pages are being encrypted?



When data pages are being encrypted?



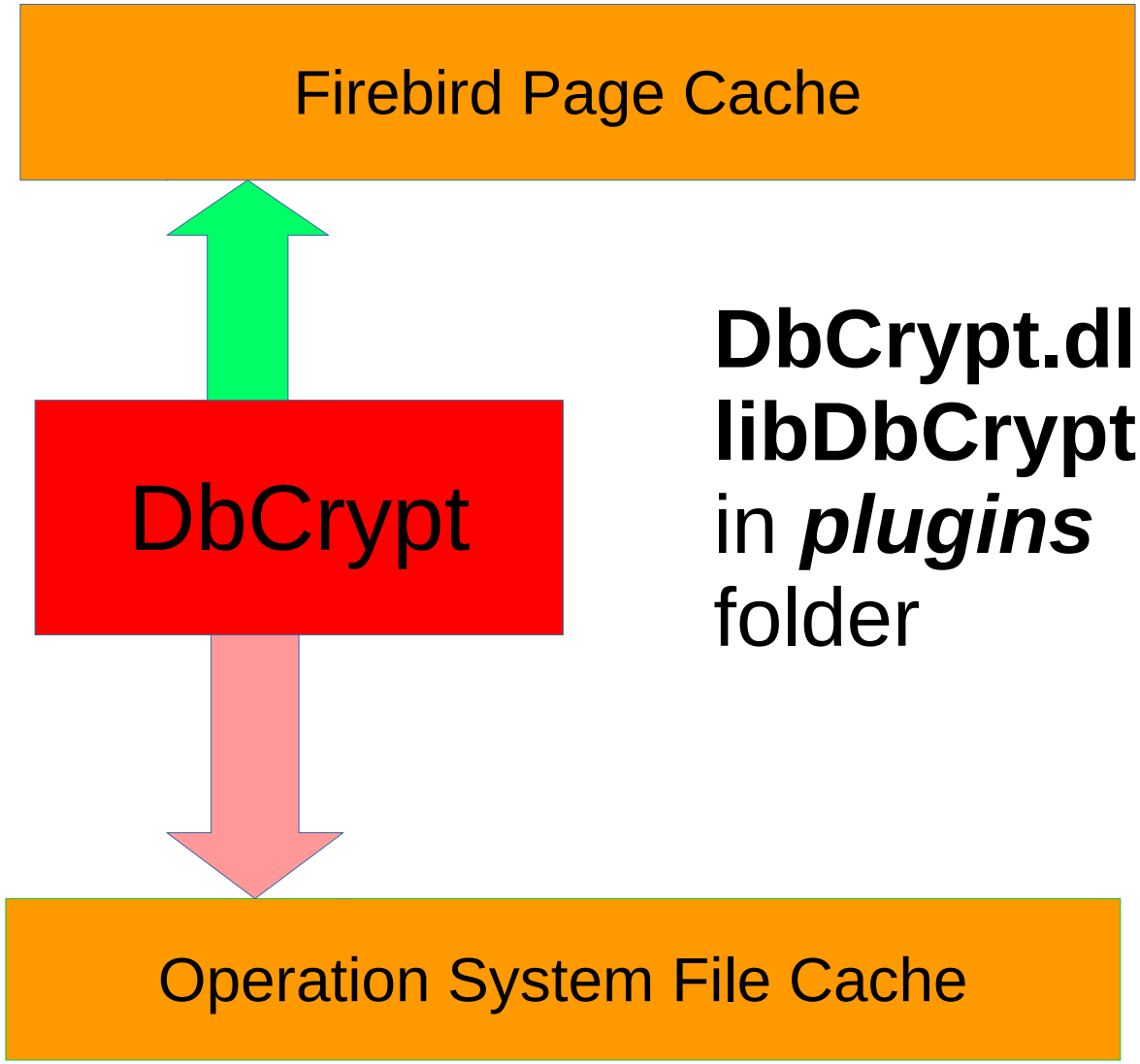
Let’s consider details



Not encrypted

Encrypted

Details: DbCrypt Plugin

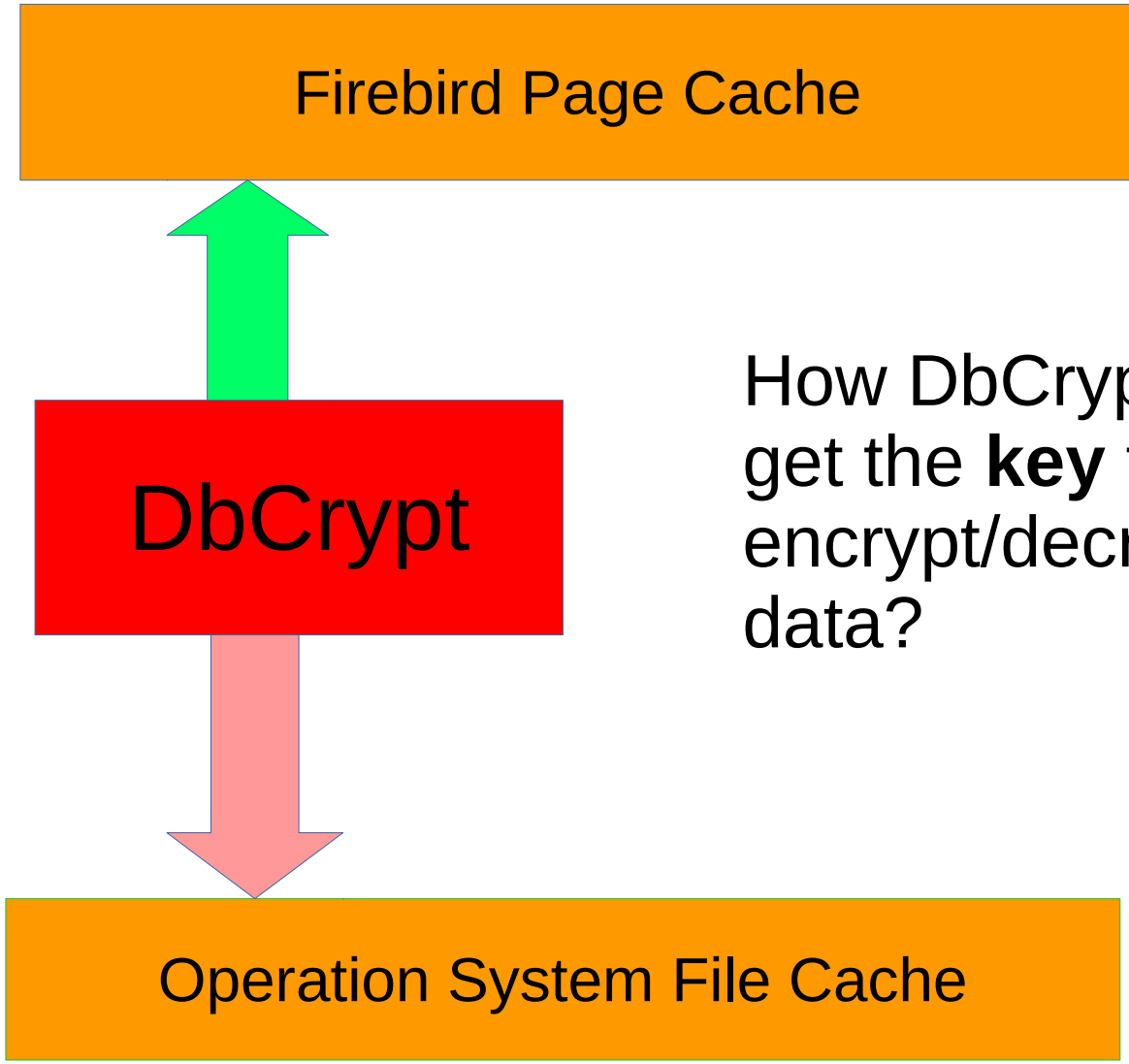


DbCrypt.dll or **libDbCrypt.so** in ***plugins*** folder

Not encrypted

Encrypted

Details: DbCrypt Plugin

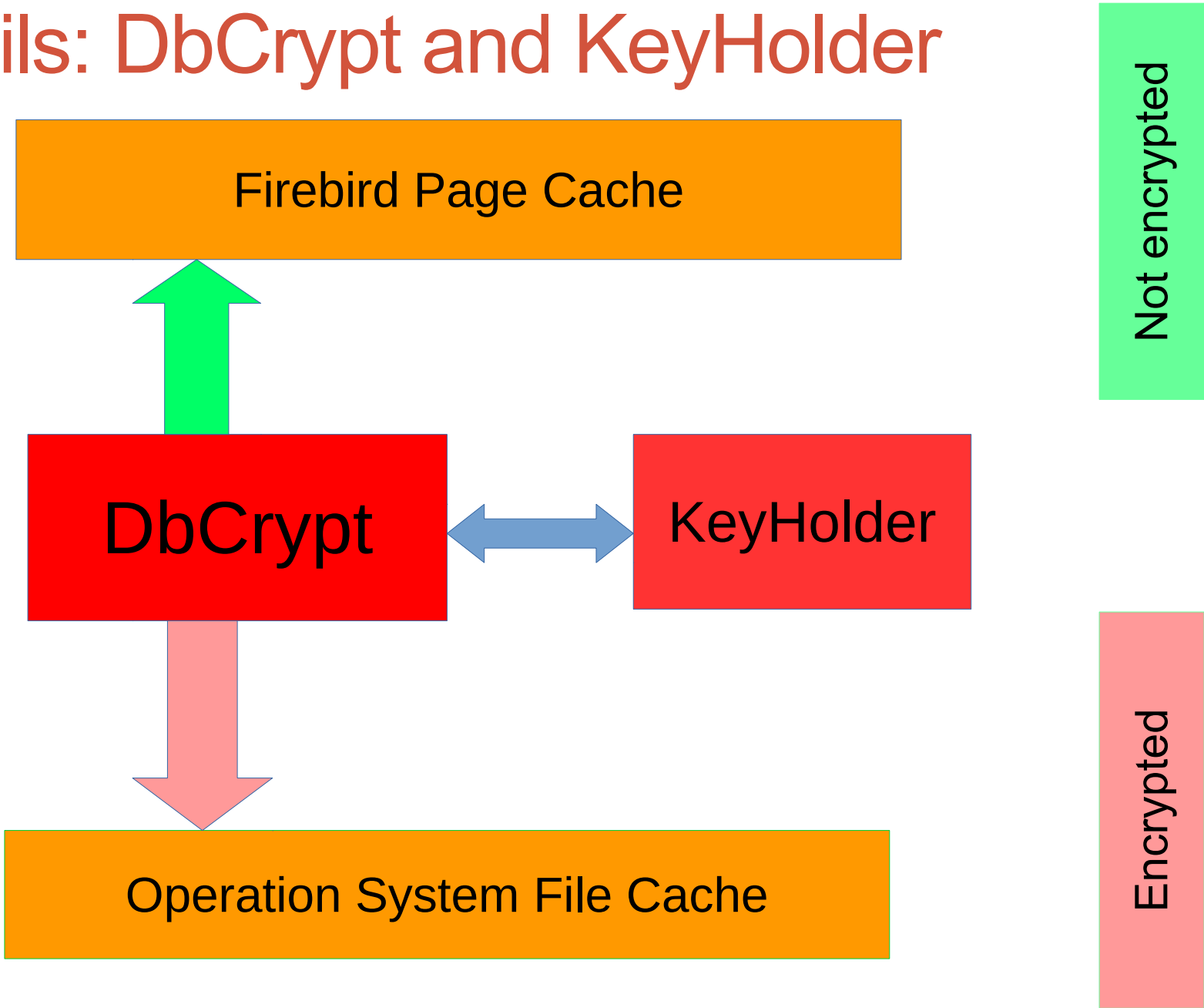


How DbCrypt get the **key** to encrypt/decrypt data?

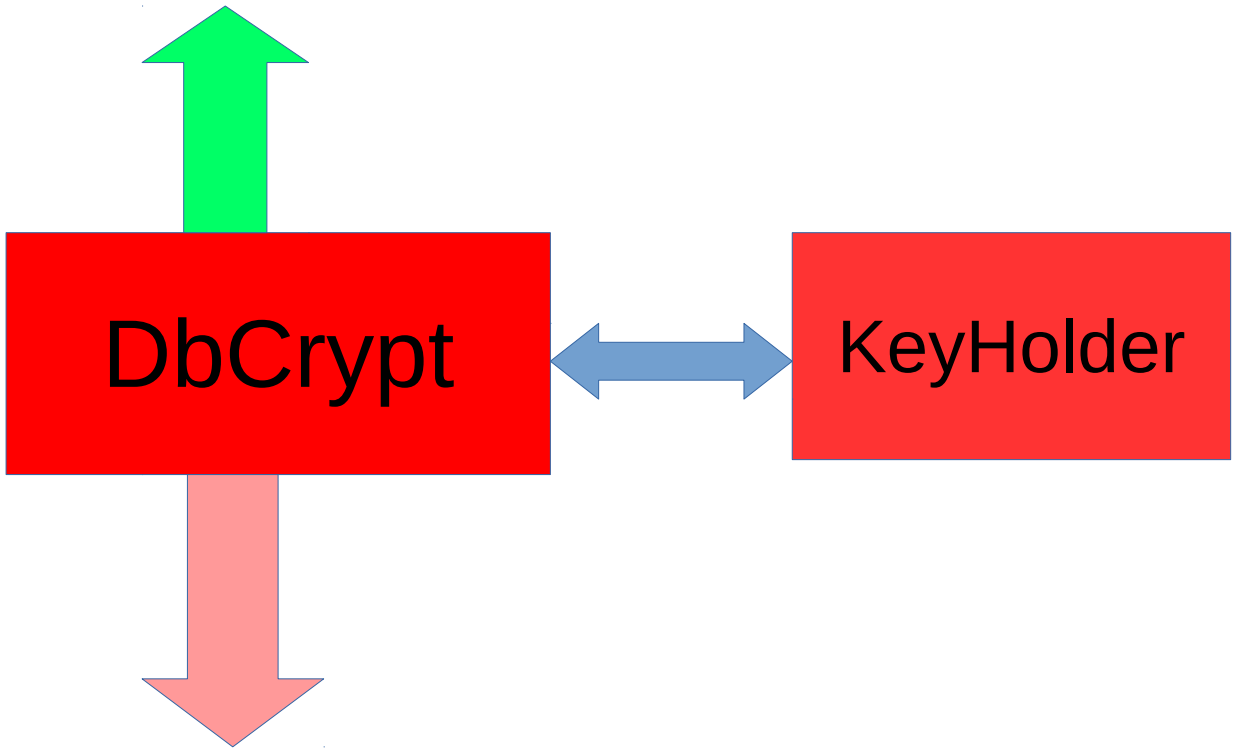
Not encrypted

Encrypted

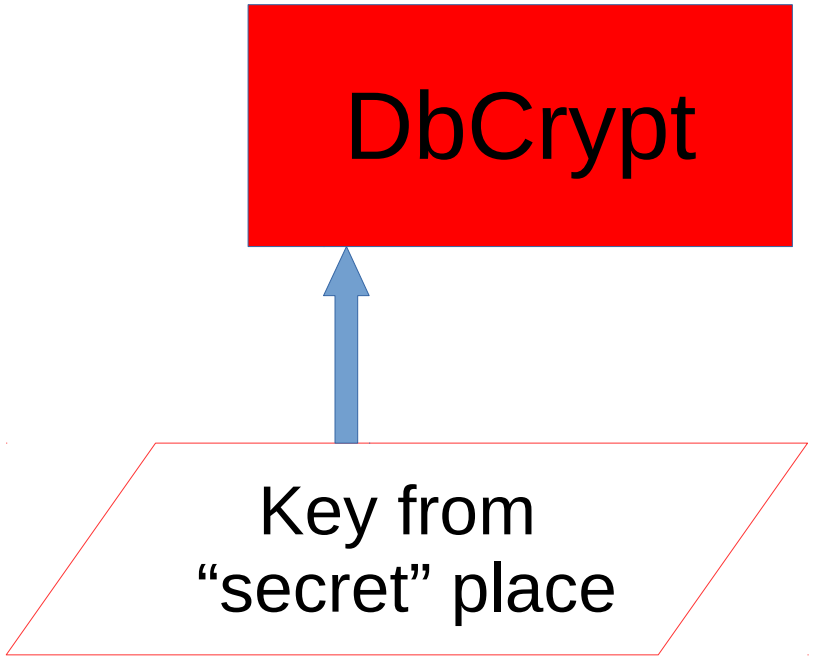
Details: DbCrypt and KeyHolder



Keys exchange details



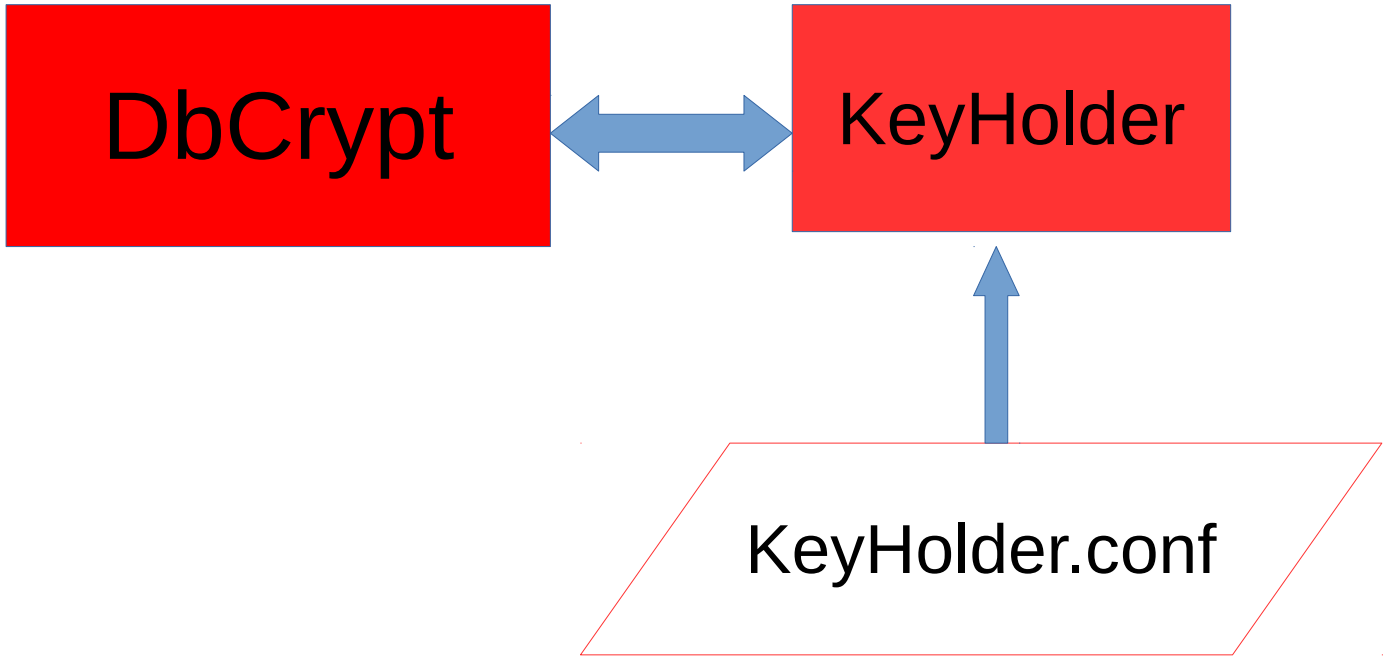
Read key from server-side file



- “Secret” place or USB stick
- Encrypted key file (with built-in or CryptoAPI)

Read key from server-side file

- Unified way to work with keys – KeyHolder
  - In case of file with keys – KeyHolder.conf



Example of KeyHolder.conf

Key=Red
0xec,0xa1,0x52,0xf6,0x4d,0x27,0xda,0x93,0x53,0xe5,0x48,0x86,0xb9,0x7d,0xe2,0x8f,0x3b,0xfa,0xb7,0x91,0x22,0x5b,0x59,0x15,0x82,0x35,0xf5,0x30,0x1f,0x04,0xdc,0x75,

Key=Green
0xab,0xd7,0x34,0x63,0xae,0x19,0x52,0x00,0xb8,0x84,0xa3,0x44,0xbd,0x11,0x9f,0x72,0xe0,0x04,0x68,0x4f,0xc4,0x89,0x3b,0x20,0x8d,0x2a,0xa7,0x07,0x32,0x3b,0x5e,0x74,

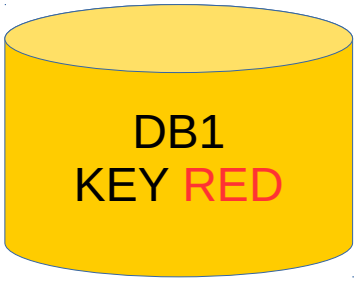
Database header of encrypted database (gstat -h databasename)

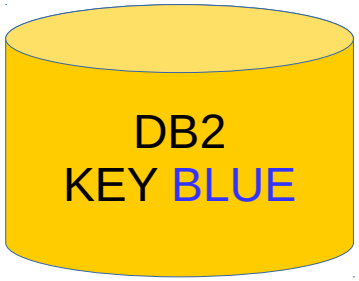
```
Database header page information:
....
Creation date     Jan 11, 2017 15:12:20
Attributes        force write, encrypted, plugin DBCRYPT

Variable header data:
 Crypt checksum: MUB2NTJqchh9RshmP6xFAiIc2iI=
 Key hash:       ask88tfWbinvC6b1JvS9Mfuh47c=
 Encryption key name:    RED
 Sweep interval:         0
*END*
```

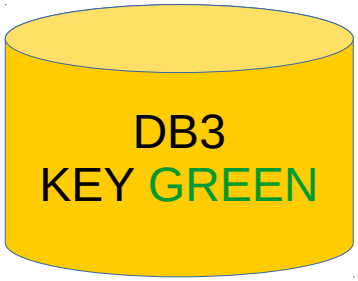
Multi-database access

DbCrypt

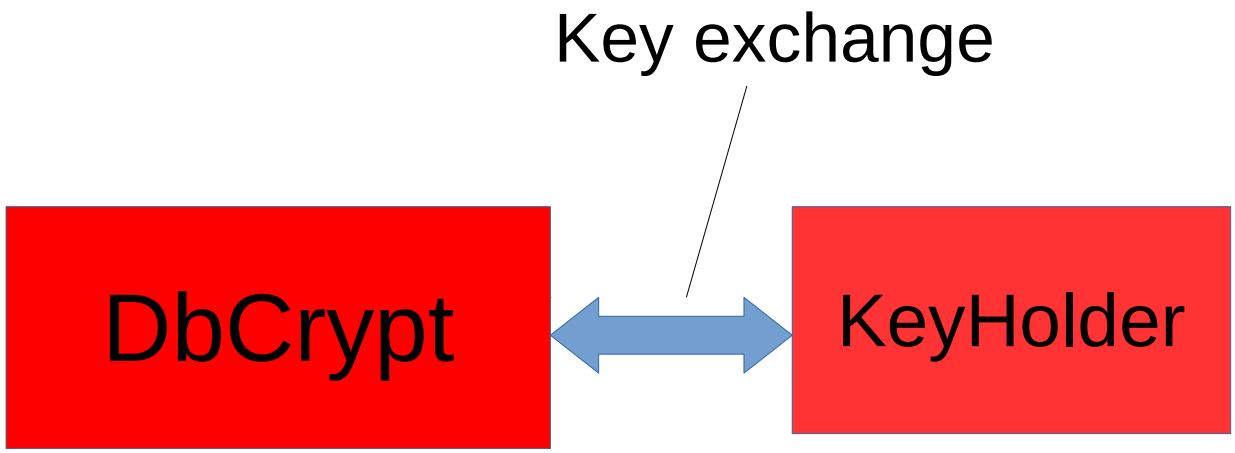
Array of Keys:
{ ‘RED’,
0xec,0xa1,0x52,0xf6,... }
{‘BLUE’,
0xab,0xd7,0x34,0x63,…}
{‘GREEN’, 0x32,…}





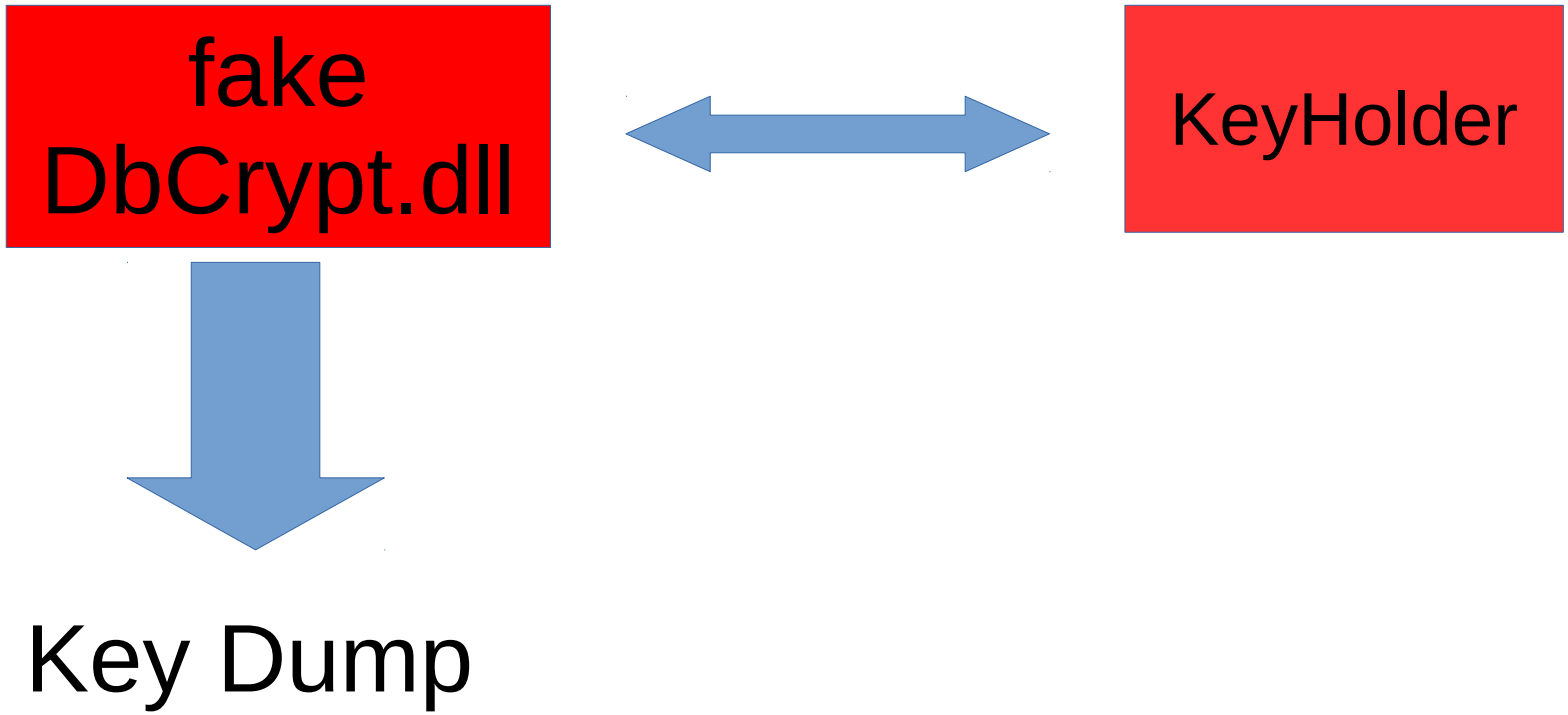


Keys management: KeyHolder

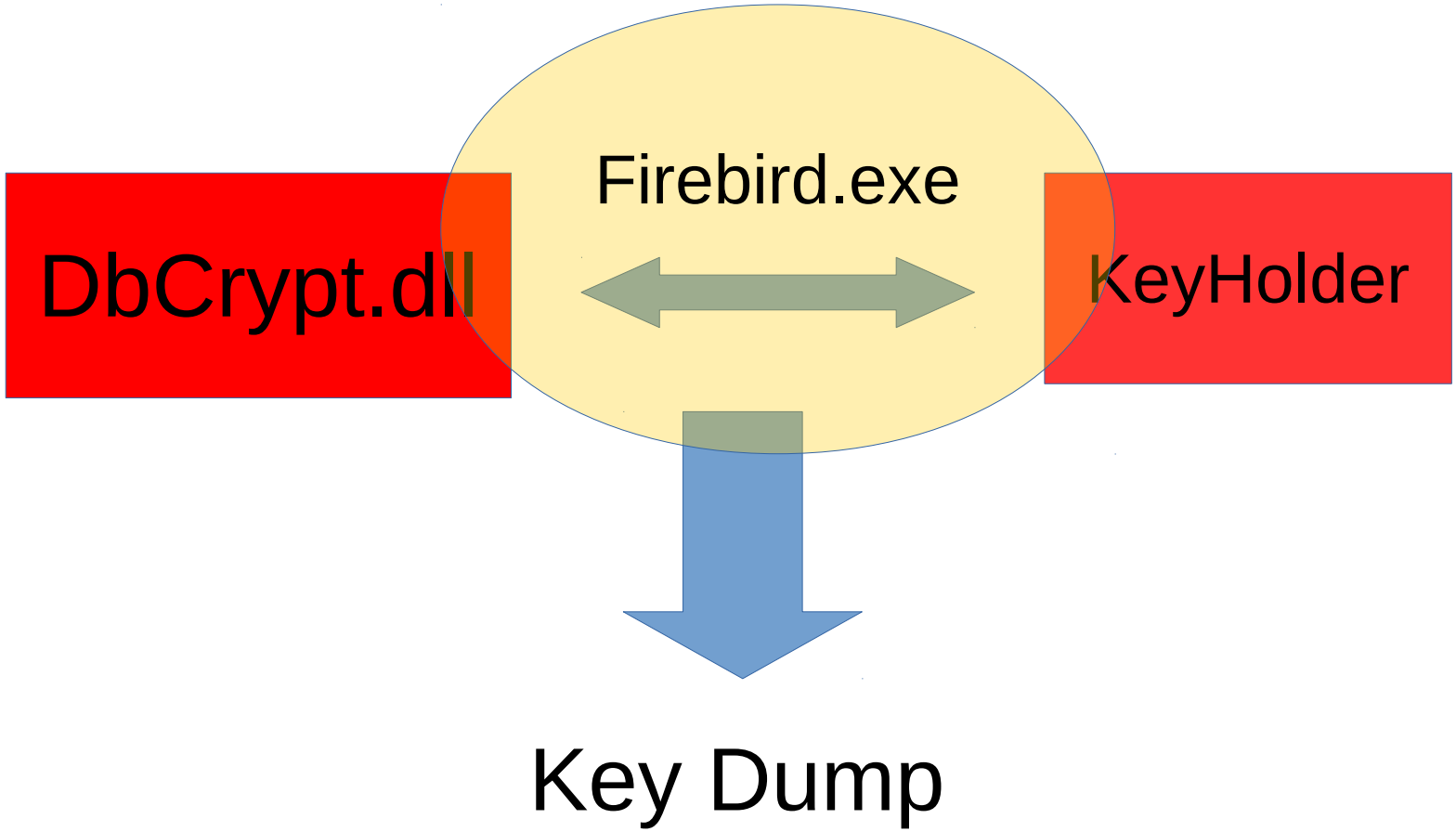


KeyHolder gets the keys from the client app, or from the safe storage

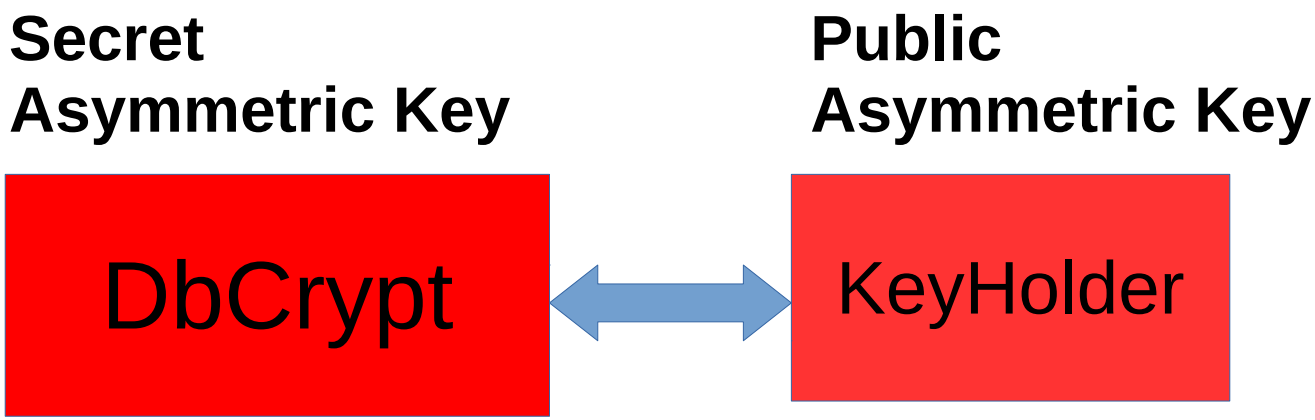
Attack scenarios
Option 1: Fake DbCrypt.dll



Attack scenarios:
Option 2: Fake Firebird.exe



Protection from fake modules



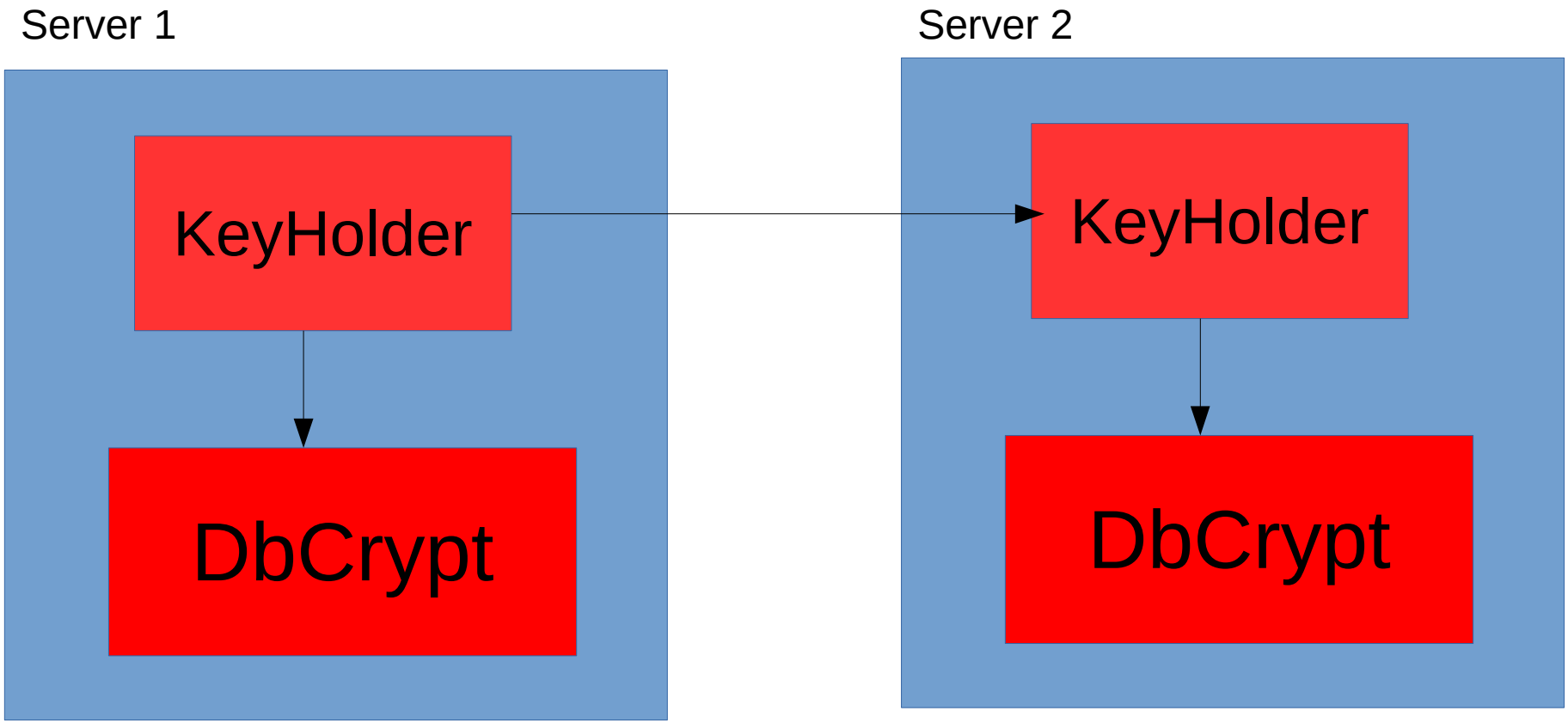
Key exchange is encrypted
with pair of public/private keys

Key exchange protocol (simplified)

- DbCrypt → KeyHolder:
  - Give Me The Key
- KeyHolder
  - Encrypt Key With Token From DbCrypt
  - Transfers Encrypted Key to DbCrypt
- DbCrypt
  - Decrypt Key
  - Ready To Work

Execute Statement On External

- In case of ES On External – how to transfer encryption key?



Summary for the server-side part of encryption

- Encryption/decryption is done by DbCrypt plugin, page by page, during the load/upload data from Firebird page cache
- Key management can be implemented in the simple way, when DbCrypt reads keys directly, but better with KeyHolder plugin

- Now let’s discover how client applications work the encrypted databases

2.2 How Firebird Encryption Works On the Client-Side

Regular Firebird connection process (simplified)

1) Client application loads client library
   1) fbclient.dll – native Windows apps
   2) libfbclient.so -native Linux apps
   3) Java, .NET - implements simplified version of protocol
2) Client app initiates connection, sending
   1) Username, e.g. SYSDBA
   2) Password, e.g., masterkey
   3) Path/alias to database

Connection in a case of encrypted database

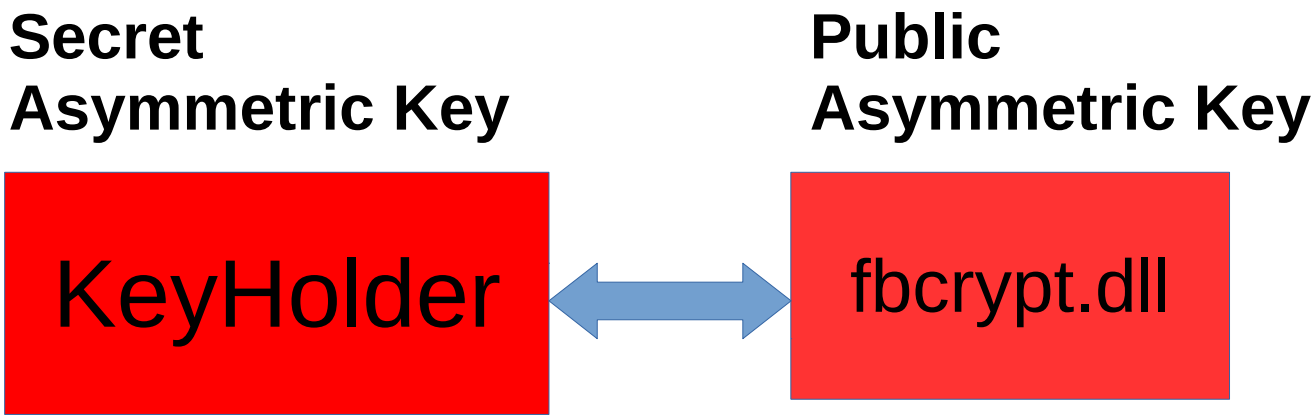
- It is necessary to pass the encryption name and key during the regular connection
  - Yes, additional network roundtrip(s) is done
  - To pass key, it is necessary to implement interface ICryptCallback

How to implement ICryptCallBack

- It can be done in any popular programming language
- To simplify creation of protected interface one can use at client side:

  fbcrypt.dll (HQbird)

Protection from key stealing



Key exchange is encrypted
with pair of public/private keys

Connecting native apps through fbcrypt.dll (Delphi, C++, PHP)

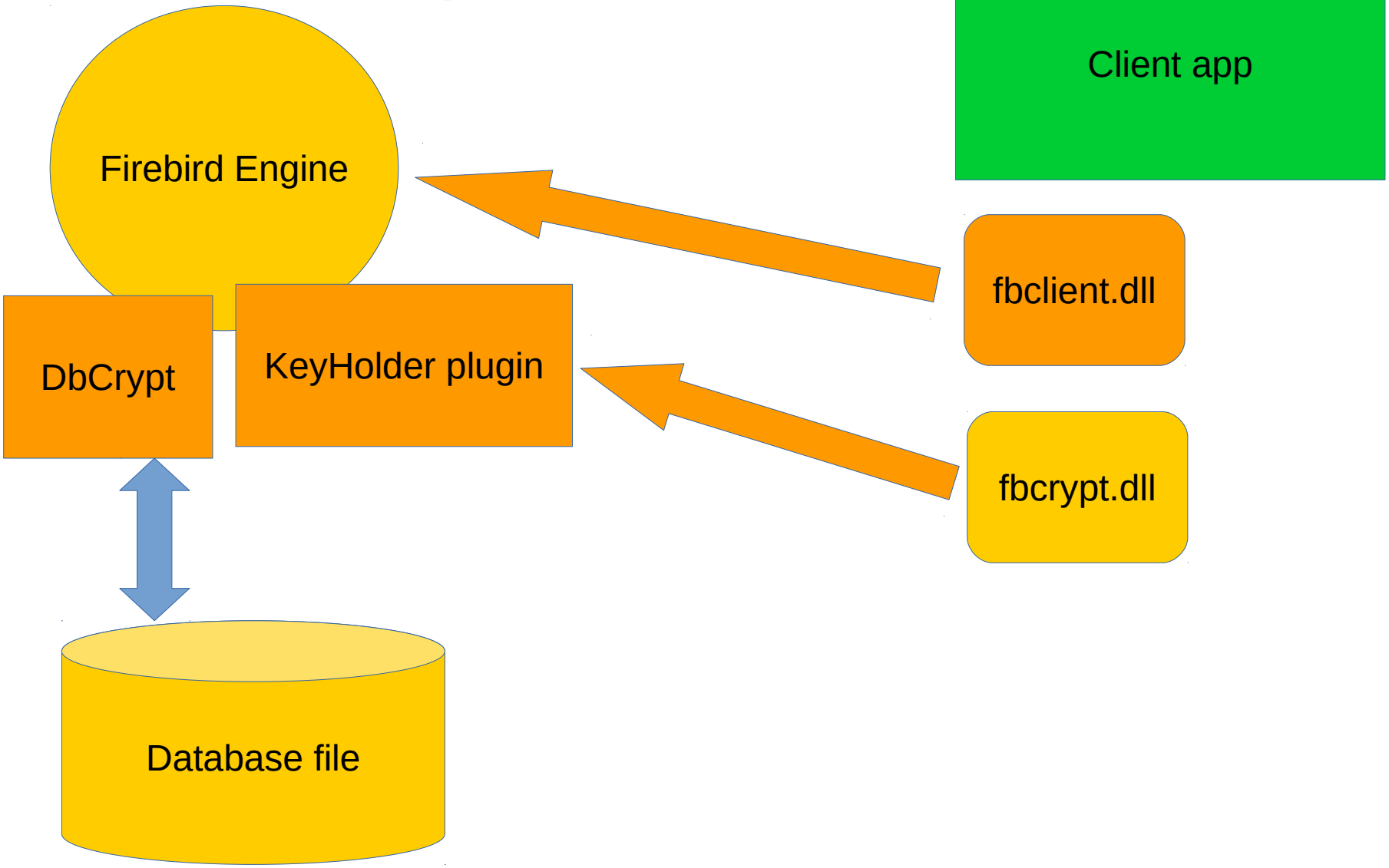
1)fbcrypt_init(pszClientPathName:Pointer) : integer;

2)fbcrypt_key(pszKeyName:Pointer;pKeyValue:Pointer;iKeyLength:Cardinal) : integer;

3)fbcrypt_callback(provider:Pointer) : integer;

- And after that establish connection as usual

Connection process



Delphi example (w/o error handling)

In BeforeConnect handler

```
fbcrypt_init(PAnsiChar('C:\Firebird30\fbclient.dll'));
fbcrypt_key('RED', '0xec,0xa1,0x52,0xf6,...'));
fbcrypt_callback();
```

Then connect as usual

```
Database1.Active:=True;
```

Thread safety

- fbcrypt calls must be done before the connection
- fbcrypt calls must be done in the same thread where the connection will be established
- Every thread requires own key transfer (as well as own connection)
  - The single call of fbcrypt_callback is enough

fbcrypt_key

- Invoking fbcrypt_key() adds key to internal key storage (array) in dll
- Keys by default are never deleted
  - To explicitly delete all keys from internal storage on the client side, use fbcrypt_init

Connecting to .NET and Java apps

- .NET and Java drivers have simplified implementations of the Firebird connection protocol
- The “ugly hack” is to send key through the connection string
  - Yes, it is unsafe! No protection from fake server module

It is necessary to set in HQbird

UnsafeClient=true in the file KeyHolder.conf

```
try
        {
            string connectionString =
            "User=SYSDBA;" +
            "Password=masterkey;" +
            "Database=G:\\Databases\\ODS12\\CRYPT.FDB;" +
            "DataSource=localhost;" +
            "Port=3053;" +
            "Dialect=3;" +
            "Charset=NONE;" +
            "Role=;" +
            "Connection lifetime=15;" +
            "Pooling=true;" +
            "MinPoolSize=0;" +
            "MaxPoolSize=50;" +
            "Packet Size=8192;" +
            "ServerType=0;" +
             "cryptkey = TXlLZXk6MHhlYywweG……;";
```

Why encryption key looks different?

//you need calculate base64 from string:

"MyKey:0xec,0xa1,0x52,0xf6,0x4d,0x27,0xda,0x93,0x53,0xe5,0x48,0x86,0xb9,0x7d,0xe2,0x8f,0x3b,0xfa,0xb7,0x91,0x22,0x5b,0x59,0x15,0x82,0x35,0xf5,0x30,0x1f,0x04,0xdc,0x75,"

// and use it as param for "cryptkey=xxx;" with ";" at the end

Detailed examples

- The ready-to-use sample client applications for Delphi, PHP, Java and .NET are here
- https://ib-aid.com/crypt

3. Installation and Configuration

Firebird.conf

- Put in firebird.conf

```
KeyHolderPlugin = KeyHolder
```

- Or, alternatively, in databases.conf, for alias crypt:

```
crypt =
C:\Temp\EMPLOYEE30\EMPLOYEE30.FDB
{
    KeyHolderPlugin = KeyHolder
}
```

Files on server

- %FirebirdFolder$\plugins
  - DbCrypt.dll
  - DbCrypt.conf
  - KeyHolder.dll
  - KeyHolder.conf – for development mode!
- %FirebirdFolder$
  - fbcrypt.dll
  - libcrypto-1_1-x64.dll

Test the encryption on server-side

isql localhost:C:\Temp\EMPLOYEE30\EMPLOYEE30.FDB -user SYSDBA -pass masterkey

SQL>**alter database encrypt with dbcrypt key red;**

SQL> show database;

Database: localhost:C:\Temp\EMPLOYEE30\EMPLOYEE30.FDB

….

ODS = 12.0

Database encrypted

Default Character set: NONE

Yes, CaSe is ImporTanT on LinuX

Please note - on Linux it is necessary to use quotes and case-sensitive plugin name:

alter database encrypt with "DbCrypt" key Red;

But keys names are always case-insensitive

Moving from Windows to Linux

"DbCrypt"

```
Database header page information:
....
Creation date     Jan 11, 2017 15:12:20
Attributes        force write, encrypted, plugin
DBCRYPT
```

In order to fix DBCRYPT → DbCrypt, make backup/restore

Or, better, add DBCRYPT to plugins.conf

Files on client side (Windows)

- Demo app – CryptTest.exe (32bit)
- Mandatory files:
  - plugins/keyholder.dll
  - fbcrypt.dll
- Optional files
  - Gbak.exe
  - plugins/dbcrypt.dll
  - Plugins/DbCrypt.conf

Background encryption thread

Encryption works only when at least 1 connection is established.
It is running in the separate parallel thread, and can take significant time!

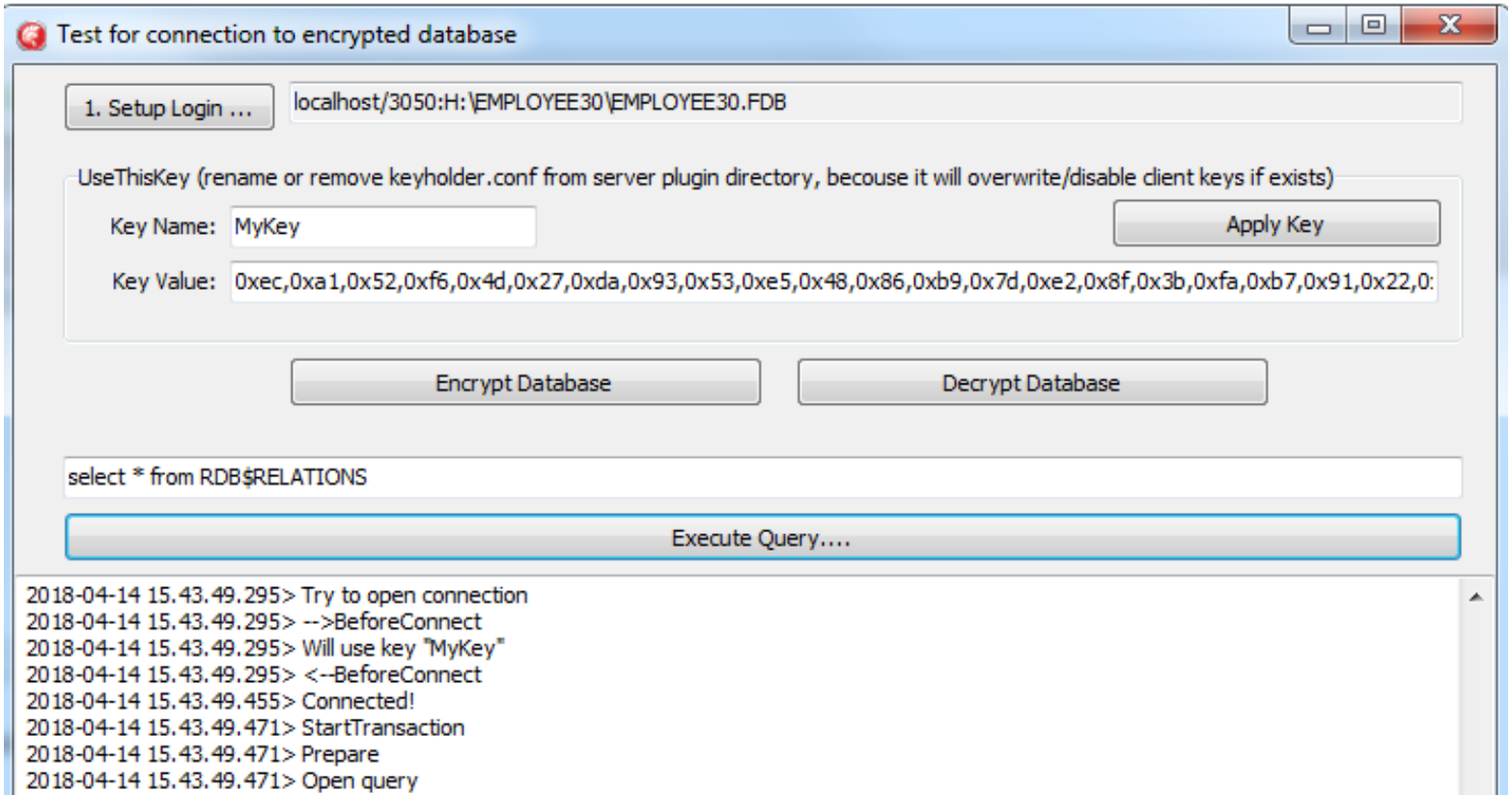
No need for downtime!

How to track encryption progress

```
select mon$crypt_page *
100.0 / mon$pages as Percent
from mon$database;
commit;
```

Test client application access

- Remove KeyHolder.conf (or comment out keys)
- Try demo app connection



Backup/restore operations

gbak support in HQbird (FB 3)

```
 -KEYFILE name of a file with DB and backup
crypt key(s)
 -KEYNAME name of a key to be used for
encryption
 -KEY     key value in "0x5A," notation
```

Backup/restore operations

gbak support in Firebird 4

```
 -KEYHOLDER  name of a key holder plugin
 -KEYNAME  name of a key to be used for encryption
```

Backup/restore operations

1) Backup copy will be created encrypted with the same key as in the database or as specified
2) Restore will be restored with the same key name or as specified
3) Multi-thread backup/restore is not supported for encrypted backups and databases (only 1 thread will be used)

By design it is impossible to create unencrypted backup of encrypted database!

The opposite is possible.

4. Performance of encrypted database

Encryption performance

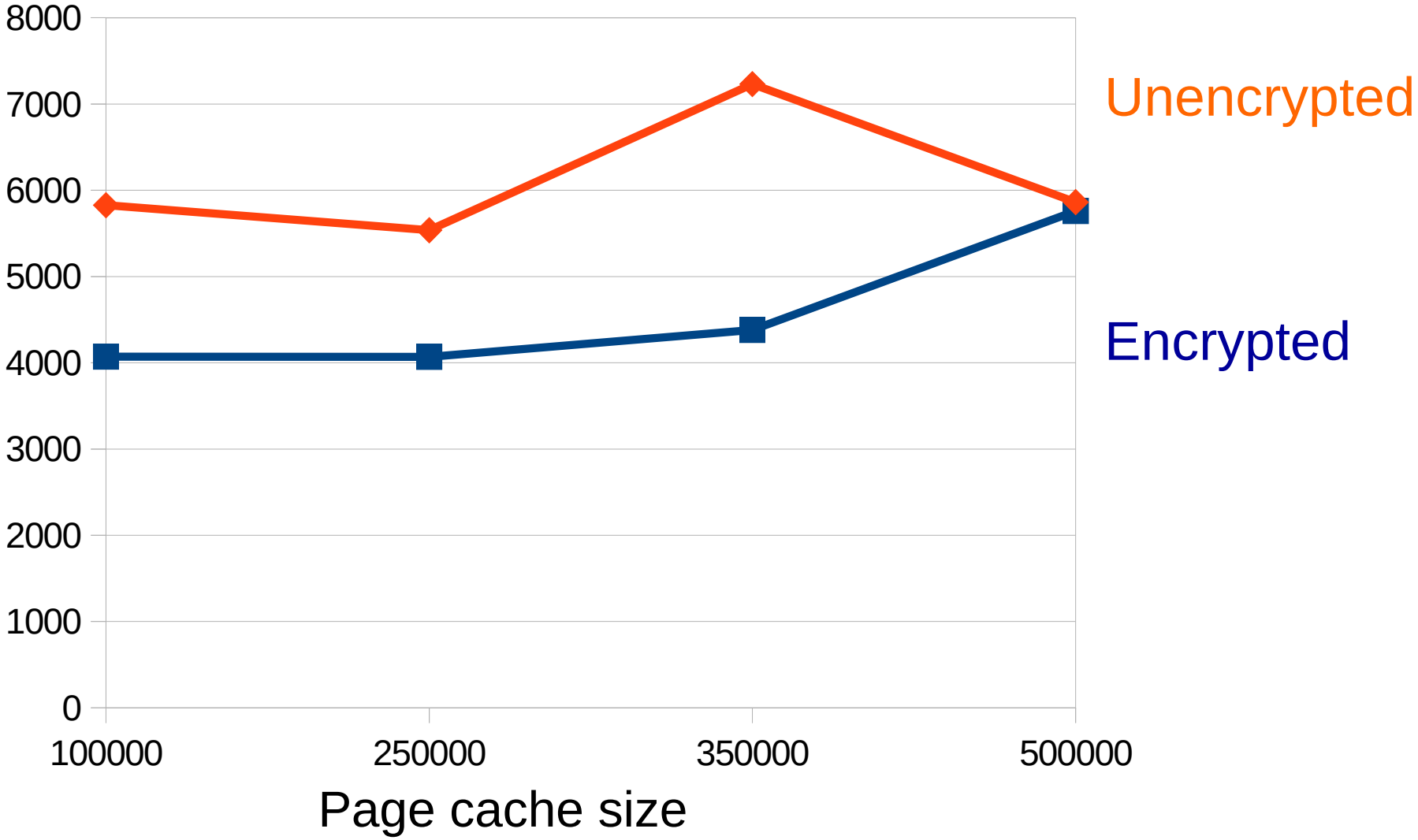
- There are 3 factors: CPU, RAM and Forced Writes
  - CPU: the faster CPU, the better results
  - RAM: the bigger part of the database is in page cache, the better results (because database pages in database cache are not encrypted)
  - Forced Writes Off - cache is flushed less frequently

Test (intensive IO), DB < RAM

- 24 (12 with HT) CPU Xeon
- RAM 32 Gb
- SSD
- 100 connections, 90 minutes
- AES256 (OpenSSL)
- Database size = 5Gb, Page Buffers 6Gb > DB

Forced writes	Not encrypted	Encrypted	Performance loss
On	4491	4152	**8%**
Off	4346	4183	**4%**

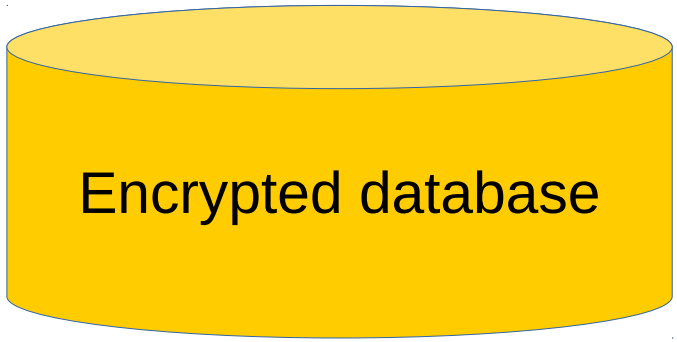
Test (Firebird OLTP-EMUL), 16Gb

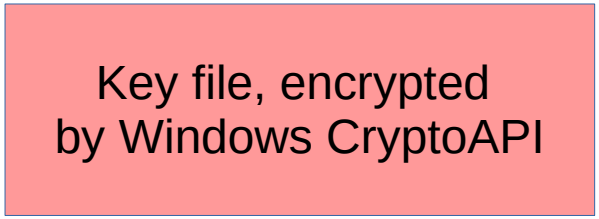


5. Real-world cases of Firebird encryption

1. Encryption with encrypted key in place

- Plugin uses Windows CryptoAPI to read key file, encrypted with CryptoAPI (custom plugin)
- Key is stored near the database
- End user every time is asked to enter Windows password

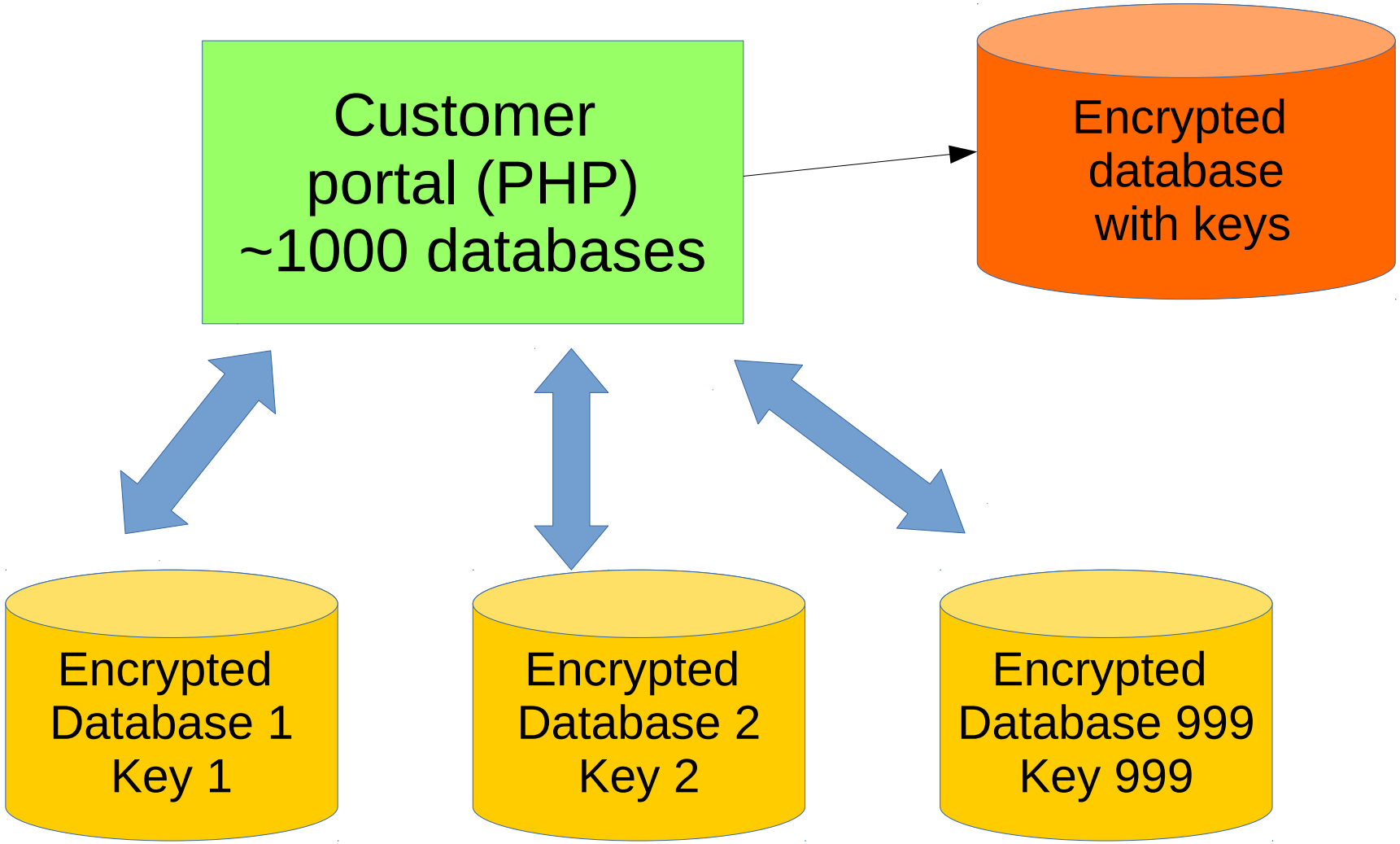




Pro & Cons

- Pro
  - Simple
  - The same level of protection of key as in Chrome, etc
- Cons
  - Protection is not related with the application

2. Multi-thread middleware



Pro & Cons

- Pro
  - Simple enough
- Cons
  - One ring rules them all (c)

Summary

Benefits of chosen encryption approach

- Online encryption/decryption – no downtime
  - Separate thread is launched
  - Only when database has other connection(s)!
- Connections to the several databases may be encrypted with different keys
  - Keys are selected according to DB header
  - Up to 2048 keys were tested
  - Support of execute statement on external
- Low performance penalty

Limitations and side-effects of encryption

- gbak – requires special version of gbak to create encrypted copies (HQbird)
- nbackup, gfix – do not work, will be fixed in FB4
- gstat – only gstat -h and gstat -e (encryption statistics)
- In case of a serious corruption, database requires decryption out of FB engine to use third-party recovery tools (like FirstAID)

Thank you

- Questions?
-
- support@ib-aid.com
-