Diagnosing and fixing Firebird performance problems

Alex Koviazin
ak@ib-aid.com

IBSurgeon, www.ib-aid.com

Slowness

Firebird performance problems can be caused by:

1. Bad transactions management
2. Problems in database structure
3. Wrong firebird.conf settings
4. Slow SQL queries
5. Highly concurrent access

Bad transactions management

2 main problems with transactions

1. Long-running [writeable] transactions
2. Rollbacked transactions

Gstat -h

Database header page information:

Flags	0
Checksum	12345
Generation	1564
Page size	4096
ODS version	10.1
Oldest transaction	**10009**
Oldest active	**20001**
Oldest snapshot	**20001**
Next transaction	**25007**
Bumped transaction	1

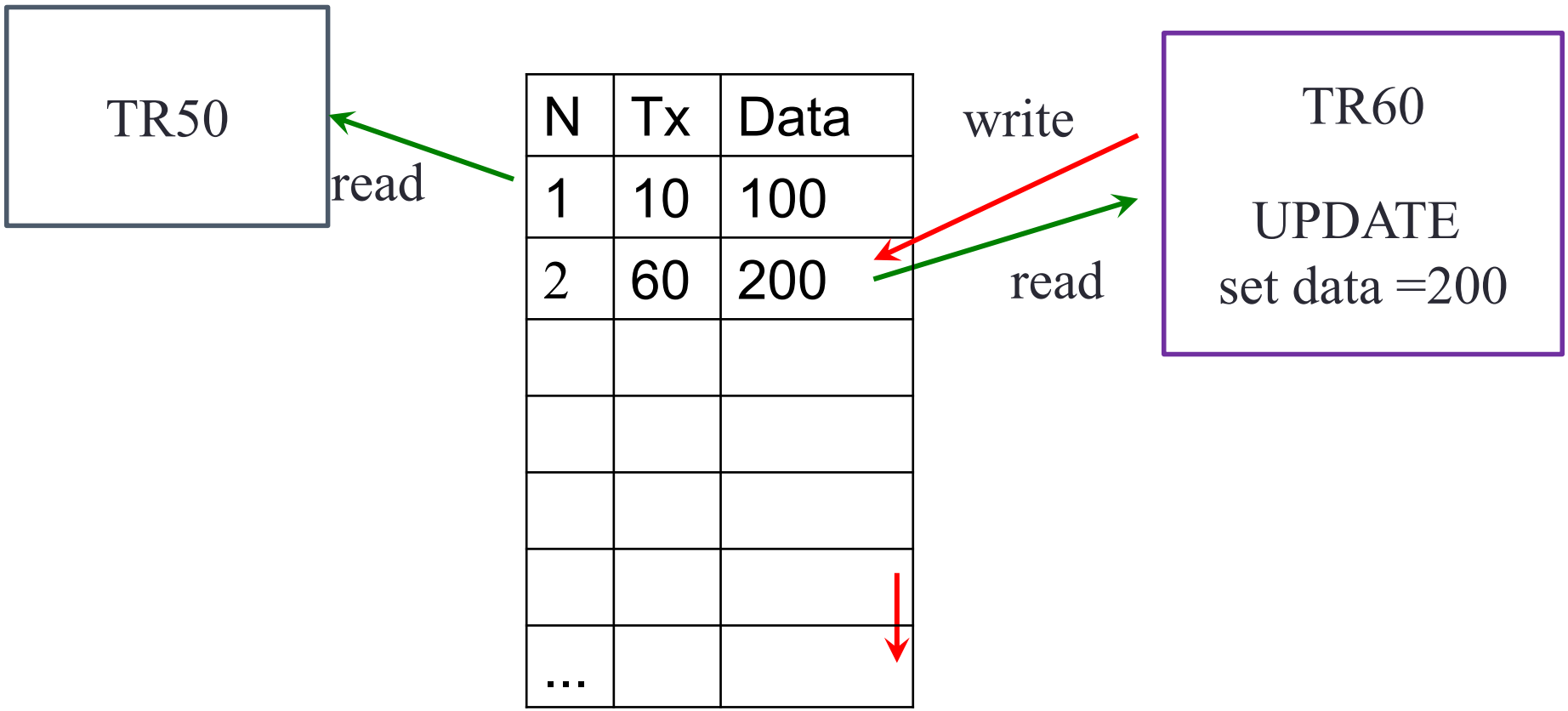
Long-running transactions

- All transactions have sequential numbers from 1 to…
- Oldest Active Transaction – currently active transaction with minimal number

Oldest transaction **10009**
Oldest active **20001**
Oldest snapshot **20001**
Next transaction **25007**

Interval NEXT – OAT shows number of *potentially* active transactions – server cannot clean **record versions** created by these transactions

What is a record version?



Different transactions can see different versions of the record.

Why many record versions are bad

After data load or restore – no versions

N	Transaction
1	1
2	1
3	1
4	1
5	1
6	1
7	1
...	

N	Transaction
1	1
2	1
3	1
4	1
5	1
6	1
7	1
...	

N	Transaction
1	1
2	1
3	1
4	1
5	1
6	1
7	1
...	

The more versions record has, the **more read operations** server does to find necessary version

N	Transaction
1	2
2	2
3	3
4	3
5	3
6	4
7	4
...	

N	Transaction
1	3
2	4
3	2
4	2
5	3
6	1
7	1
...	

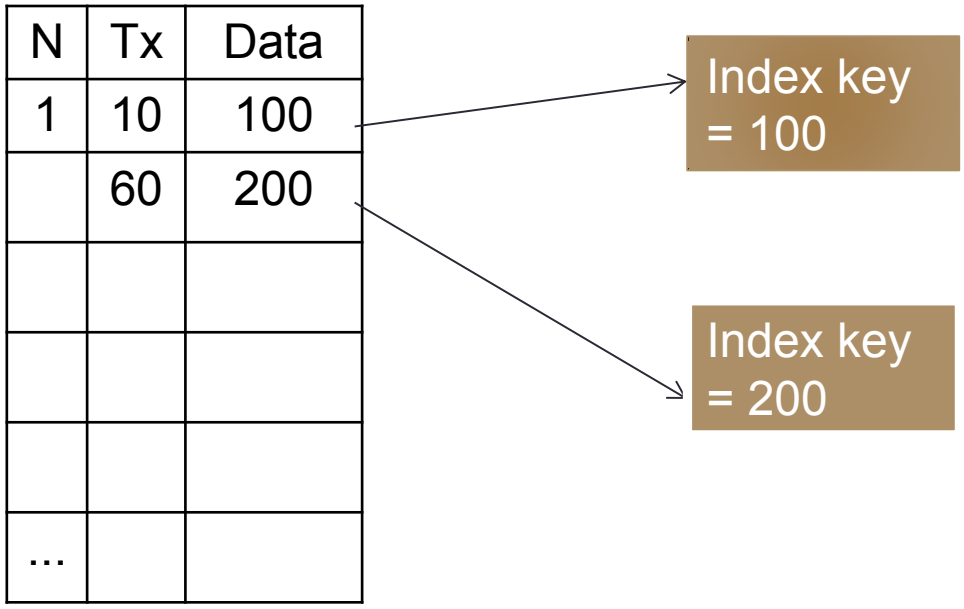
N	Transaction
1	5
2	5
3	5
4	2
5	2
6	2
7	2
...	

N	Transaction
1	1
2	1
3	1
4	1
5	1
6	10
7	10
...	

N	Transaction
1	1
2	10
3	1
4	10
5	1
6	1
7	1
...	

Why many record versions are bad

- When UPDATE changes indexed fields, indices also must be updated, and - UPDATE does not update keys in the index, it **adds new keys** for new record version!
- DELETE does not delete index keys!



How to identify long running active transactions in Firebird?

1. Manual query to MON$ tables
2. HQbird FBDataGuard & Firebird MON$Logger - demo

How to fix active long-running transactions in Firebird?

1. Don’t do it (i.e, fix the source code)
2. Restart connections
    1. stop/start client applications
    2. Restart Firebird

Rollbacked transactions

- When some transaction is marked in transaction inventory as **rollbacked**, it prevents record versions beeing cleaned by collective or background garbage collection

Oldest transaction	**10009**
Oldest active	**20001**
Oldest snapshot	**20001**
Next transaction	**25007**

Interval **Oldest Snapshot – Oldest Interesting** determines the need for sweep

How to fix OIT stuck?

1. Don’t do it (always commit ☺)
2. Sweep
   - Sweep is a process of cleaning database from garbage versions
   - Sweep is necessary when OST-OIT interval becomes big

SWEEP reads database from the beginning to the end and cleans obsolete versions

How to make sweep

- Autosweep
  - by default 20000
  - Starts immediately when interval > threshold
  - Slowness at unpredicted moments
- Manual sweep
  - Scheduled sweep during the inactivity period of time
  - Can be run manually with gfix –sweep or in HQbird FBDataGuard

Sweep must be controlled!

- If sweep did not succeed to align transaction markers, it can indicate a serious problem or corruption!

- HQbird FBDataGuard checks sweep status

Problems in database structure

2 main problems with database structure

- Deep indices
- Fragmented tables

Deep indices

- How to find
  - Gstat –r
  - HQbird IBAnalyst – demo

- How to fix – Increase Page size
- In case of 16Kb page size – consider another index

Fragmented tables

- Fragmented by BLOBs
- Fragmented by big records

- How to find – IBAnalyst only
- How to fix
  - Increase page or decrease page size
  - Change schema – move BLOBs to another table

Wrong firebird.conf settings

Wrong firebird.conf settings-1

1. These default settings are too small, must be increased
   - LockHashSlots
   - TempCacheLimit
   - DefaultDBCachePages
- Situation is better in 3.0

Wrong firebird.conf settings-2

2. Settings which do not correspond Firebird Architecture

- Too big DefaultDBCachePages for Classic/SuperClassic – recommended 256-2048
- Too small DefaultDBCachePages for SuperServer
  - 10000 for 2.5
  - 50k – 2M for 3.0

Wrong firebird.conf settings-3

2. Wrong combination of settings

If FileSystemCacheThreshold < DefaultDBCachePages, file system cache will be disabled
= disaster for Classic/SuperClassic
= not so good for SuperServer

How to fix

1. Read comments in firebird.conf

2. Use optimized Firebird configuration from IBSurgeon
    1. http://ib-aid.com/en/optimized-firebird-configuration/
    2. Bundled with HQbird (text files)

Slow SQL queries

Slow SQL queries

How to find slow queries

1. Trace API – 2.5, 3.0
2. MON$ tables – 2.1, 2.5, 3.0
3. FBScanner – 1.0, 1.5, 2.0, 2.1, 2.5
4. In-app SQL statistics

Trace API

- It catches everything
  - Queries, Transactions, Stored Procedures, Triggers
- It makes all operations slower
  - Can be improved with time threshold, less things to be monitored, etc

  - Demo

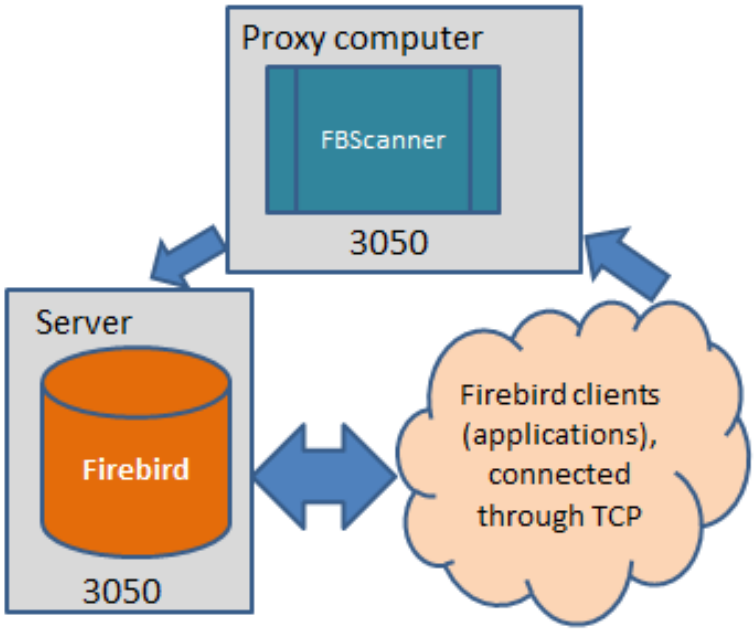
Mon$

- Show only current SQL queries (no sp/triggers)
  - Idle, Stalled, Active
- It shows reads and writes, not the time
  - Shows also fetches

  - Demo

FBScanner

- Works as a proxy between client application and Firebird (3.0 is not supported)
- Can be setup on remote server, and track queries for the selected subset (1 workstation)



How to fix slow SQL query?

- Sorry, it requires 1 day seminar!

Highly Concurrent SQLs

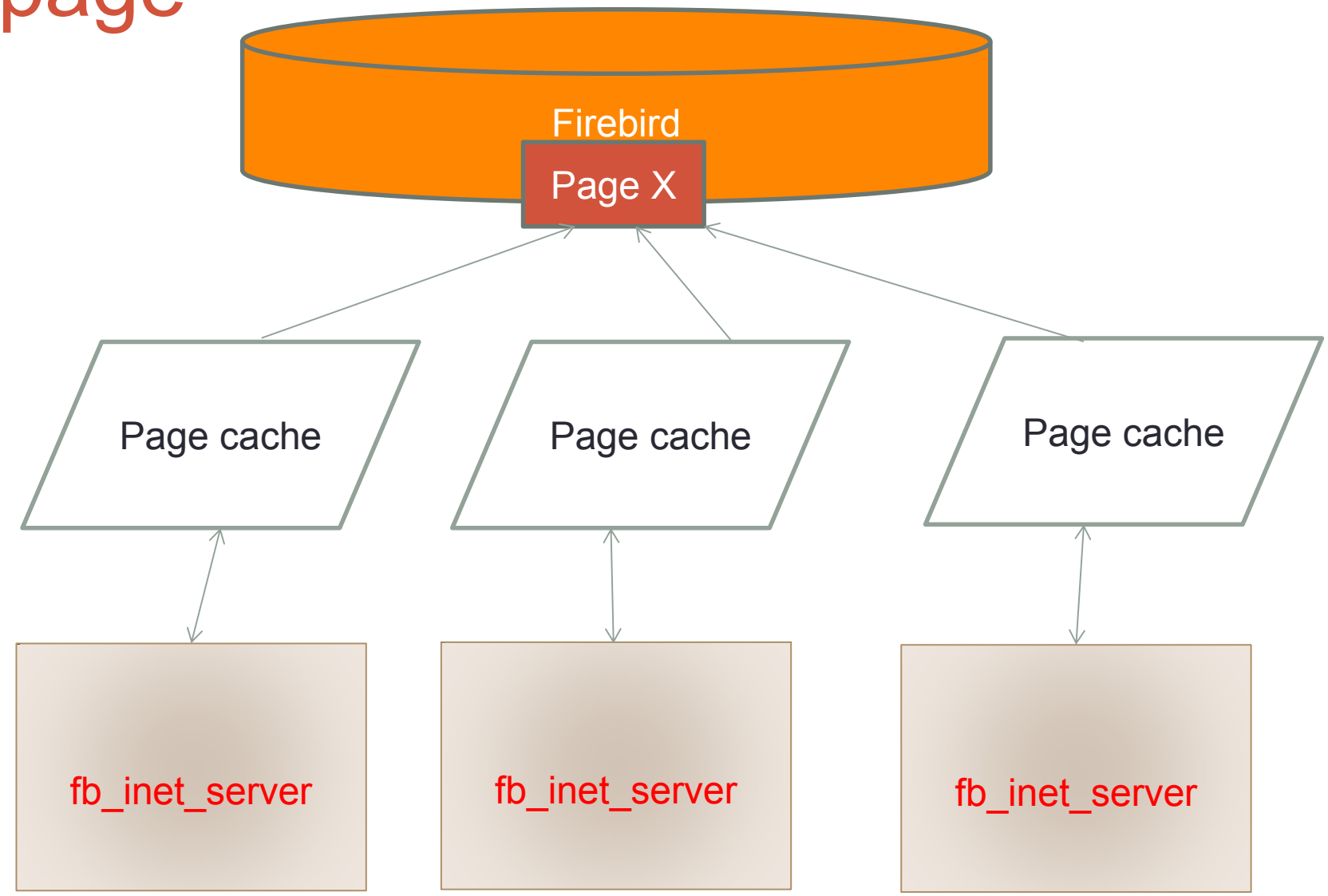
What is highly concurrent SQLs?

- When query which work fine at 1 computer, works 10x-100x slower with many connections
- Lock table -> **Mutex wait** values is more than 30%

Lock table – where to look

Fb_lock_print –d <database_name>
LOCK_HEADER BLOCK
Version: 17, Active owner: 0, **Length: 6291456, Used: 5517236**
Flags: 0x0001
Enqs: 10906251, Converts: 58907, Rejects: 22373, Blocks: 210859
Deadlock scans: 5841, Deadlocks: 0, Scan interval: 10
Acquires: 13636997, Acquire blocks: 558879, Spin count: 0
Mutex wait: 4.1%
Hash slots: **2003,** Hash lengths (min/avg/max): **2/ 11/ 26**
Remove node: 0, Insert queue: 0, Insert prior: 0
Owners (107): forward: 26696, backward: 5517140
Free owners: *empty*
Free locks (1630): forward: 3878196, backward: 2264580
Free requests (793): forward: 5412916, backward: 1906516
Lock Ordering: Enabled

Many SQLs compete for the same page



Examples of highly concurrent access

- Implementation of notifications through the table and SELECT
- Update of some flag table
- Getting GEN_ID values very often

How to catch it?

- Use MON$
- Demo

Thank you!
And don’t forget these links

Questions? ak@ib-aid.com

IBSurgeon optimized configuration files
https://ib-aid.com/en/optimized-firebird-configuration/

HQbird Standard - tools are compatible with 1.5-3.0, free trial for 14 days
https://www.ib-aid.com/en/hqbird

Why should we monitor lock table parameters?

- Lock table is a critical part of Classic and SuperClassic architectures
- Access to shared objects is implemented through locks in Lock table…

Analysis of lock table parameters is the easiest way to reveal problems with concurrent access

Why should we monitor lock table parameters?

- Lock table is a critical part of Classic and SuperClassic architectures
- Access to shared objects is implemented through locks in Lock table…

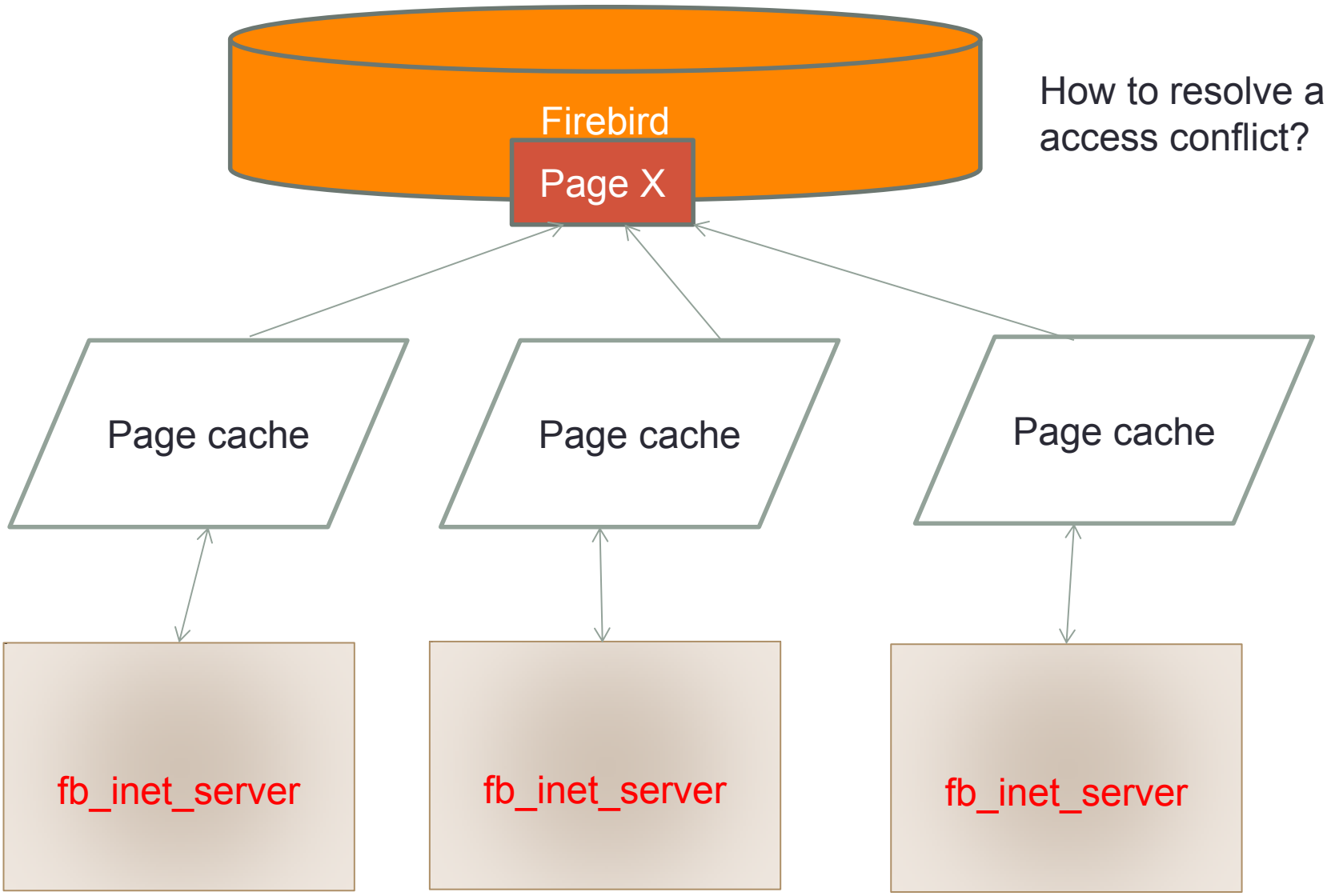
Analysis of lock table parameters is the easiest way to reveal problems with concurrent access

Why should we monitor lock table parameters?

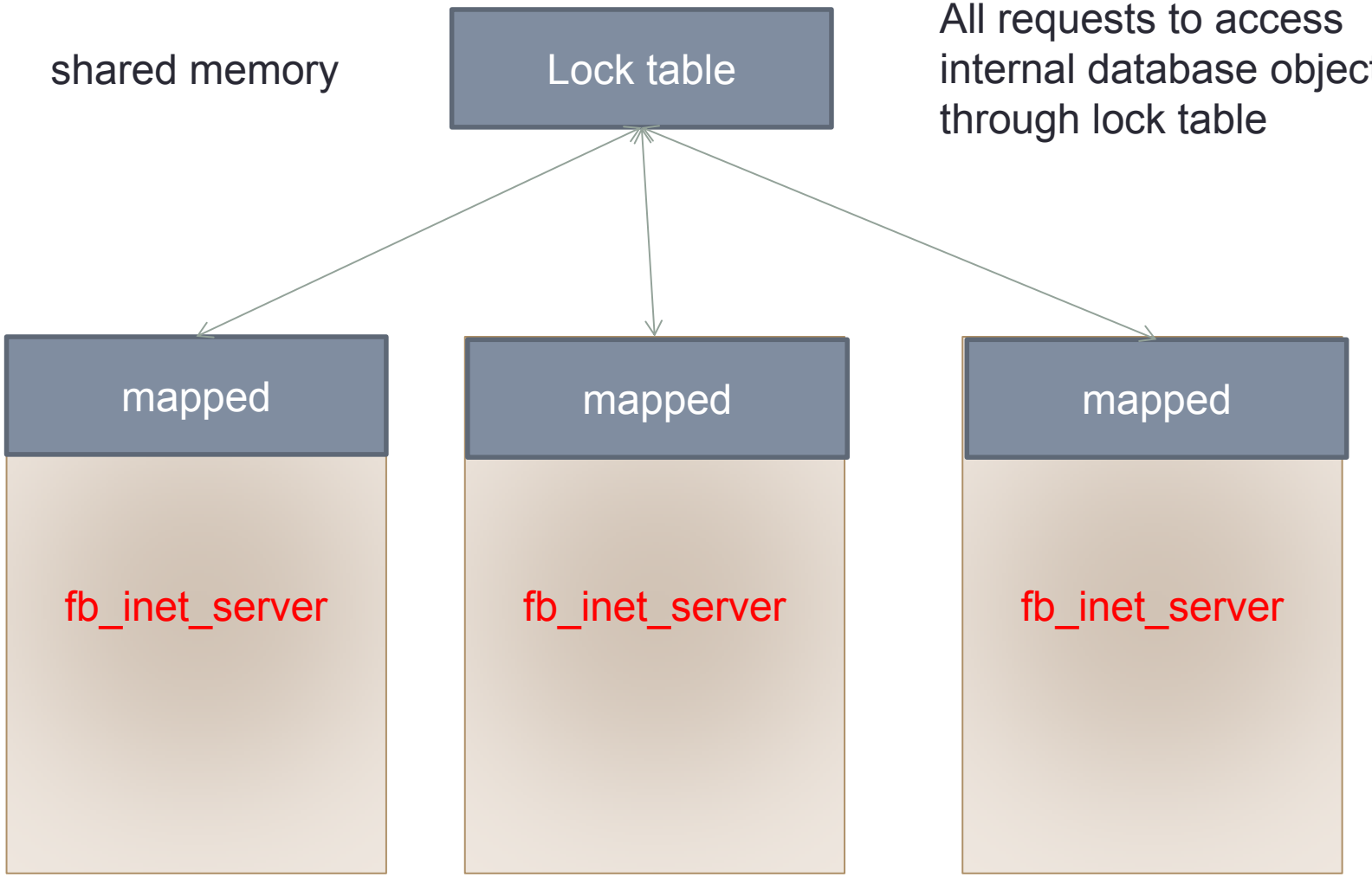
- Lock table is a critical part of Classic and SuperClassic architectures
- Access to shared objects is implemented through locks in Lock table…

Analysis of lock table parameters is the easiest way to reveal problems with concurrent access

How lock table works



How lock table works



Why should we monitor lock table parameters?

- Lock table is a critical part of Classic and SuperClassic architectures
- Access to shared objects is implemented through locks in Lock table…

Analysis of lock table parameters is the easiest way to reveal problems with concurrent access

Lock table analysis - raw

- Fb_lock_print –d <database_name>
- LOCK_HEADER BLOCK
  - Version: 17, Active owner: 0, Length: 6291456, Used: 5517236
  - Flags: 0x0001
  - Enqs: 10906251, Converts: 58907, Rejects: 22373, Blocks: 210859
  - Deadlock scans: 5841, Deadlocks: 0, Scan interval: 10
  - Acquires: 13636997, Acquire blocks: 558879, Spin count: 0
  - Mutex wait: 4.1%
  - Hash slots: 2003, Hash lengths (min/avg/max): 2/ 11/ 26
  - Remove node: 0, Insert queue: 0, Insert prior: 0
  - Owners (107): forward: 26696, backward: 5517140
  - Free owners: *empty*
  - Free locks (1630): forward: 3878196, backward: 2264580
  - Free requests (793): forward: 5412916, backward: 1906516
  - Lock Ordering: Enabled

Lock table analysis – where to look

- Fb_lock_print –d <database_name>
- LOCK_HEADER BLOCK
  - Version: 17, Active owner: 0, **Length: 6291456, Used: 5517236**
  - Flags: 0x0001
  - Enqs: 10906251, Converts: 58907, Rejects: 22373, Blocks: 210859
  - Deadlock scans: 5841, Deadlocks: 0, Scan interval: 10
  - Acquires: 13636997, Acquire blocks: 558879, Spin count: 0
  - **Mutex wait: 4.1%**
  - Hash slots: **2003,** Hash lengths (min/avg/max): **2/ 11/ 26**
  - Remove node: 0, Insert queue: 0, Insert prior: 0
  - **Owners (107):** forward: 26696, backward: 5517140
  - Free owners: *empty*
  - Free locks (1630): forward: 3878196, backward: 2264580
  - Free requests (793): forward: 5412916, backward: 1906516
  - Lock Ordering: Enabled

How to monitor lock table

- 1) Command prompt (on the server only), run every N min

fb_lock_print -d E:\OLTP-EMUL\oltp30.fdb

- 2)

Alerts and automatic recommendations

Lockprint

Lockprint: **IMPORTANT** [Last run: 9 min 26 sec ago]
Deadlock Scans=0, Deadlocks=0, Mutex Wait=0
Hash slots=1009, Hash Lengths: Min=0, Avg=0, Max=3
Owners=1, FreeOwners=0, Size: Length=1048576, Used=22144
<<less
Increase number of hash slots. (1009)
It is recommended to have increased number of hash slots to achieve better performance. Consider to increase LockHashSlots parameter at least to 2049, or use optimized IBSurgeon configuration files (http://ib-aid.com/en/optimized-firebird-configuration/).

Thresholds and recommendations

Essential values:	**Firebird.conf params to adjust locks:**
Length: 6291456	LockMemSize = 1048576 x10
Hash slots: **2003,** Hash lengths (min/avg/max): **2/ 11/ 26**	LockHashSlots = 1009 Must be prime number, x20
Mutex wait: 4.1%	Nothing to adjust, mutex is an indicator of concurrency
Owners (107)	Number of connections for server

Use optimized Firebird configuration files from IBSurgeon:
http://ib-aid.com/en/optimized-firebird-configuration/

What to monitor?

- Database statistics – Record Versions and Max Versions
- Indices keys

How to monitor

1. With gstat command line tool
2. With HQbird Firebird Database Analyst

HQBird
Advanced FirebirdSQL Distribution for Enterprises

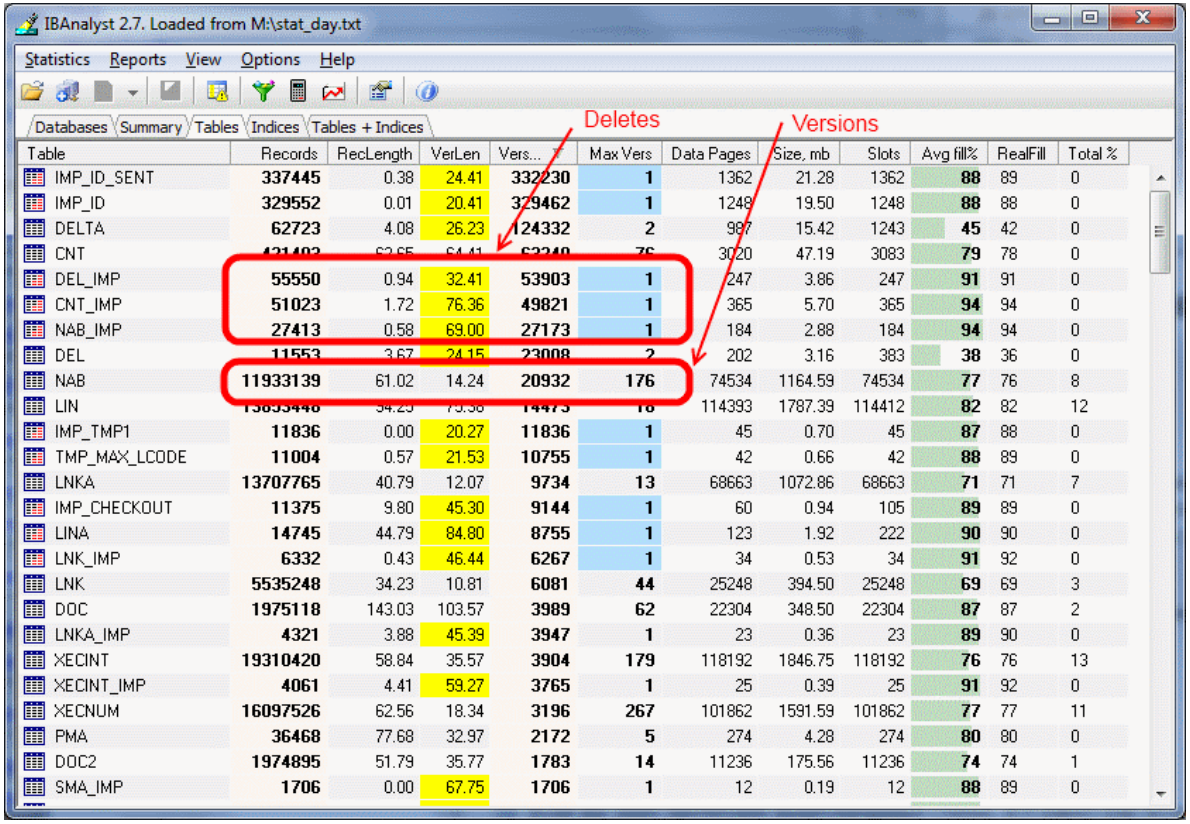


Table	Records	RecLength	VerLen	Vers...	Max Vers	Data Pages	Size, mb	Slots	Avg fill%	RealFill	Total %
IMP_ID_SENT	337445	0.38	24.41	332230	1	1362	21.28	1362	88	89	0
IMP_ID	329552	0.01	20.41	329462	1	1248	19.50	1248	88	88	0
DELTA	62723	4.08	26.23	124332	2	987	15.42	1243	45	42	0
CNT	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	3020	47.19	3083	79	78	0
DEL_IMP	55550	0.94	32.41	53903	1	247	3.86	247	91	91	0
CNT_IMP	51023	1.72	76.36	49821	1	365	5.70	365	94	94	0
NAB_IMP	27413	0.58	69.00	27173	1	184	2.88	184	94	94	0
DEL	11553	3.67	24.15	23008	2	202	3.16	383	38	36	0
NAB	11933139	61.02	14.24	20932	176	74534	1164.59	74534	77	76	8
LIN	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	114393	1787.39	114412	82	82	12
IMP_TMP1	11836	0.00	20.27	11836	1	45	0.70	45	87	88	0
TMP_MAX_LCODE	11004	0.57	21.53	10755	1	42	0.66	42	88	89	0
LNKA	13707765	40.79	12.07	9734	13	68663	1072.86	68663	71	71	7
IMP_CHECKOUT	11375	9.80	45.30	9144	1	60	0.94	105	89	89	0
LINA	14745	44.79	84.80	8755	1	123	1.92	222	90	90	0
LNK_IMP	6332	0.43	46.44	6267	1	34	0.53	34	91	92	0
LNK	5535248	34.23	10.81	6081	44	25248	394.50	25248	69	69	3
DOC	1975118	143.03	103.57	3989	62	22304	348.50	22304	87	87	2
LNKA_IMP	4321	3.88	45.39	3947	1	23	0.36	23	89	90	0
XECINT	19310420	58.84	35.57	3904	179	118192	1846.75	118192	76	76	13
XECINT_IMP	4061	4.41	59.27	3765	1	25	0.39	25	91	92	0
XECNUM	16097526	62.56	18.34	3196	267	101862	1591.59	101862	77	77	11
PMA	36468	77.68	32.97	2172	5	274	4.28	274	80	80	0
DOC2	1974895	51.79	35.77	1783	14	11236	175.56	11236	74	74	1
SMA_IMP	1706	0.00	67.75	1706	1	12	0.19	12	88	89	0

Goal of every Firebird developer is

to avoid creating versions for records!

Not only because server slow reads multiple record versions, but also because of

GARBAGE COLLECTION

What is garbage and how it is collected

- When record versions become obsolete and non-interested by any transaction, it is considered as garbage and need to be cleaned.

- Garbage collection is a process of removing unnecessary records versions
  - It can be cooperative (Classic or SuperClassic) or background (SuperServer)

Why should we monitor garbage collection?

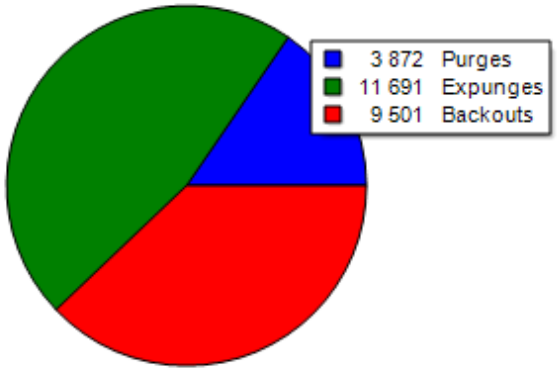
- It consumes resources.
- We should locate and fix parts of the applications which produce many record versions and provoke GC

How to monitor

HQBird
Advanced FirebirdSQL Distribution for Enterprises

- 1) Manual SQL queries to MON$ tables
- 2) HQbird Firebird MonLogger

Garbage collection details



Top attachments with max garbage collection (Purges+Expunges+Backouts)

Username	IP	Process name	Purges	Expunges	Backouts
SM5870	10.1.55.94	S_Market.exe	7	3195	13
SM5870	10.1.55.94	S_Market.exe	7	3195	13
SM1745	10.1.55.144	S_Market.exe	1407	1199	18
SM1745	10.1.55.144	S_Market.exe	1407	1199	18
SM5695	10.1.55.94	S_Market.exe	10	1347	9
SM5695	10.1.55.94	S_Market.exe	10	1347	9
SM1091	10.1.55.23	S_Market.exe	738	264	132

Memory usage

Monitoring transactions in Firebird

Temp files

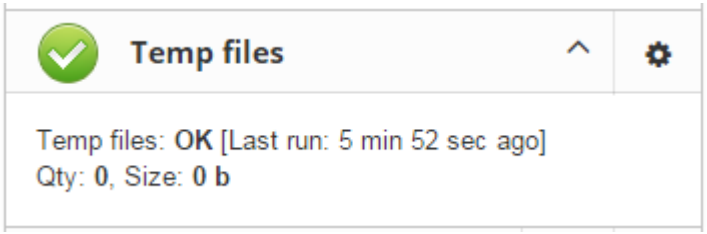
Temp files

- Temp files are created in default temp folder or in specified folders (TempDirectories in firebird.conf)
- Actually they are written to the disk only if size of all sort files is more than TempCacheLimit parameter
- ***It is better to have sorting in memory!***

How to track Temp files

HQBird
Advanced FirebirdSQL Distribution for Enterprises

1. Manually check size and quantity of temp files (fb_sortxxx) in all temp folders
2. HQbird FBDataGuard monitors temp files



How to move temp files to memory

- Increase **TempCacheLimit** parameter
- Warnings:
  - At Classic TempCacheLimit is allocated for each process
  - 32bit processes have 2Gb limitation in memory to address

Use optimized configuration files from IBSurgeon!

Monitor connections, SQL queries and transactions

Connections

- Connection peaks
  - Massive operations
  - Direct web connections
  - Several connection per client

  - Connections peaks are dangerous at Classic/SuperClassic — memory can be easily exhausted.

Transactions

- Typical mistakes with transactions
  - 1. Transactions are not closed
  - 2. Always using 1 transaction per operation (usually due to autocommit)
  - 3. Using writeable transactions for read-only operations

SQL statements

- 1. Slow queries
- 2. Inactive queries (consume memory)

- Demo

Hardware monitoring from Firebird point of view

Does your hardware work really good?

- 1. Why we need universal score for hardware
- 2. Firebird Hardware Guide

http://ib-aid.com/download/docs/Firebird_Hardware_Guide_IBSurgeon.pdf

Unversal scoring for Firebird

Official Firebird test!

Available in Firebird repository and in HQbird

Thank you!
And don’t forget these links

Questions? ak@ib-aid.com

IBSurgeon optimized configuration files
http://ib-aid.com/en/optimized-firebird-configuration/

Firebird Hardware Guide
http://ib-aid.com/download/docs/Firebird_Hardware_Guide_IBSurgeon.pdf