RedDatabase 4.0 in Enterprise

Berlin, 2019

Firebird Conference 2019

Berlin, 17-19 October

IBSurgeon

Contents

- Description of the loaded system
- Tricks
- System improvements

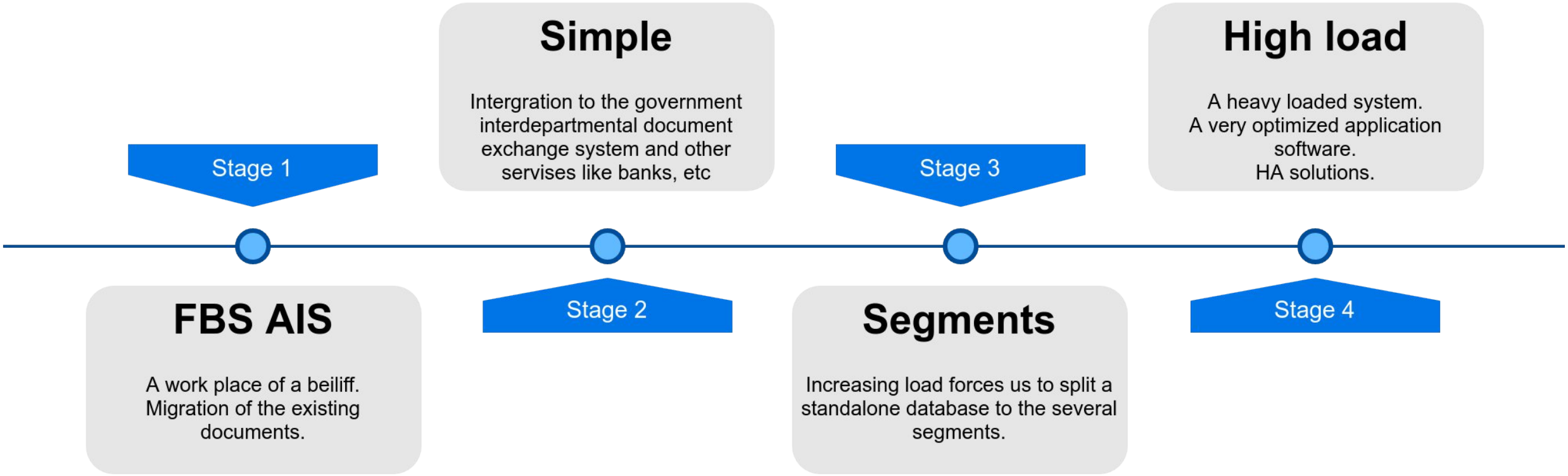
Description of the loaded system

Experience

- **Federal Bailiff Service (FBS)**
  - **~3000** installations of RedDatabase. One's in every Russian city
  - **10 billions** of documents annually
  - **24/7** mode
  - **100TB** in the main cluster of the central databases

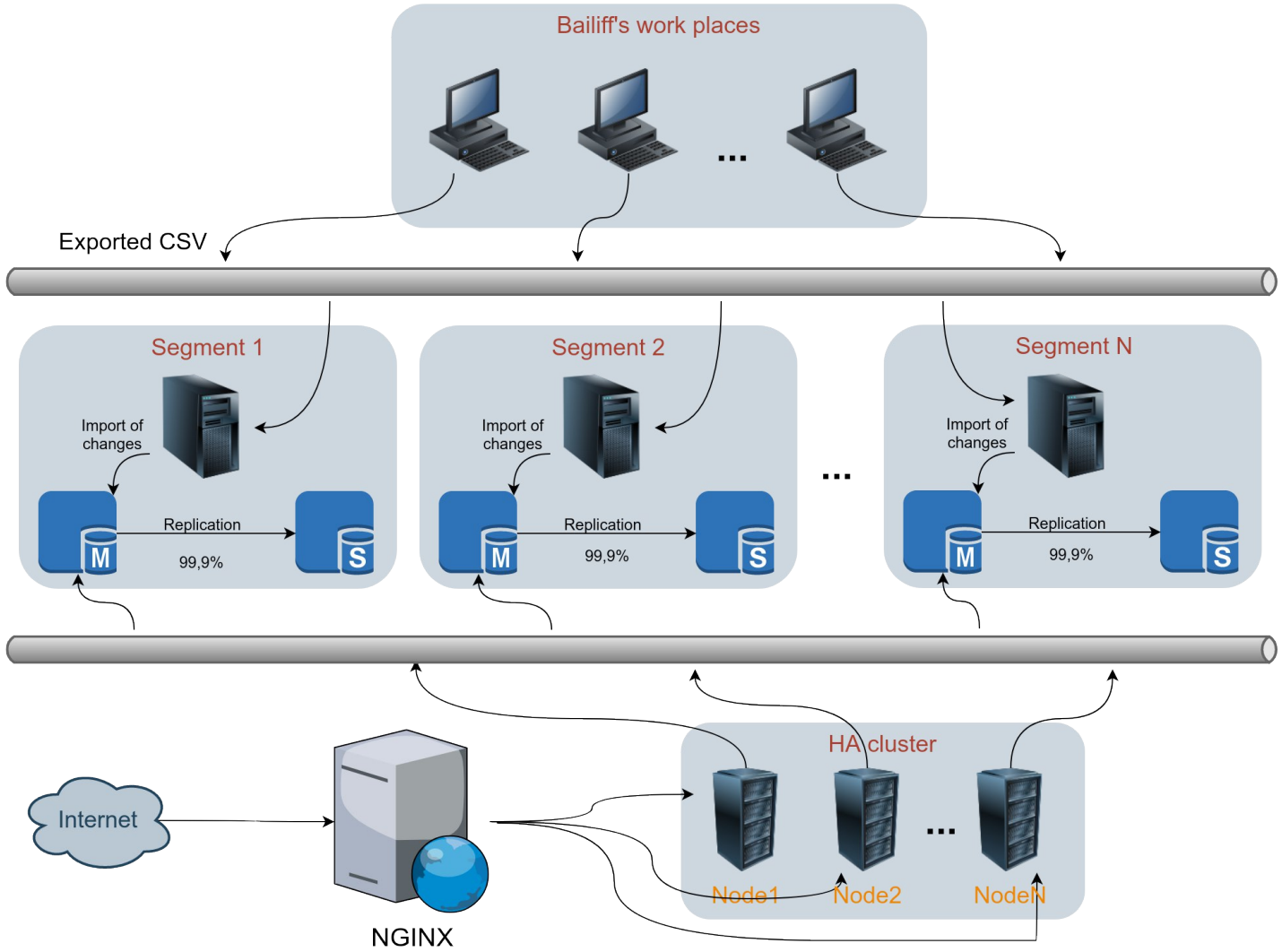
- **Interdepartmental document exchange system of FBS (IDES)**
  - the most loaded subsystem
  - typical database size is **3 TB**
  - a database has tables
    - with **12 columns** and
    - contains up to **1.5 billion records**.
  - every day about **3 million new records** are inserted or updated
  - A query with search takes **less than 100 ms**.
  - about **2000 clients concurrently** use the database.

Short history of IDES



Requirements to IDES

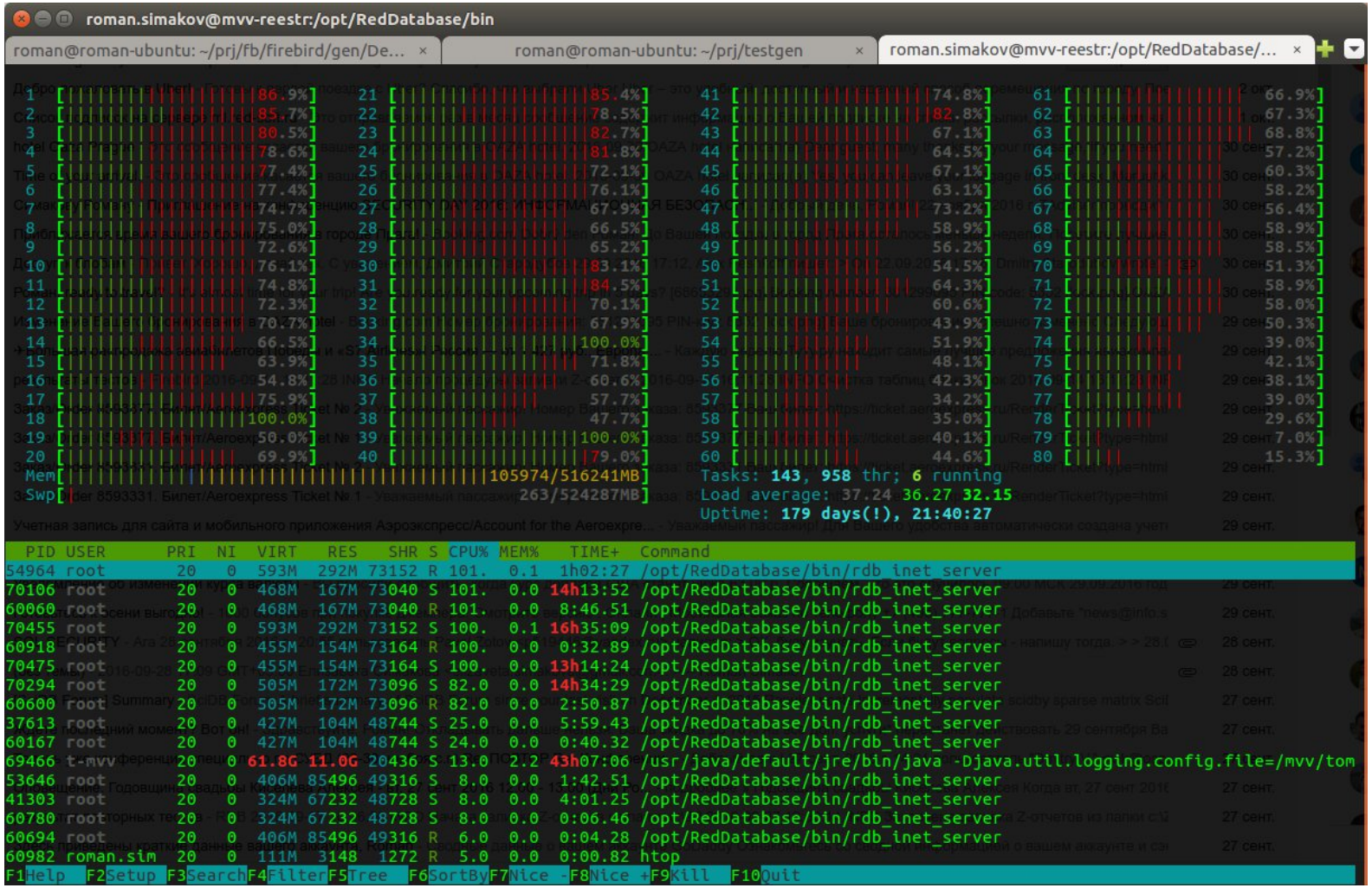
Architecture of IDES



A role of the segments

- Distribute load between several databases
- Collect necessary data to localize processing
- Every database is supported independent to other segments
- There is no a single point of failure
- It's more easy to maintain

Example of heavy loaded segment



Tricks

General tuning

- LCX containers
  - The fastest path between the engine and disk

- tmpfs - a special file system in Linux for temporary files
  - it's suitable for temporary files used in the sorting operations
  - can use swap if necessary
  - useful for big RAM volumes

- transparent hugepage
  - Slowdown memory allocation and usage
  - DB allocates small pages
  - It's better to disable

```
> echo 'madvise'> /sys/kernel/mm/redhat_transparent_hugepage/defrag
> echo 'madvise'> /sys/kernel/mm/redhat_transparent_hugepage/enabled
```

Limits

- Max process:
  - On RHEL/CentOS - 1024 by default
  - For classic - 1024 / 5 threads = ~200 connections
- Max open files:
  - On RHEL/CentOS - 1024/4096
  - Every classic process open ~20 files
  - Limit is ~200 connections
- To extend:
  - Carefully edit /etc/security/limits.conf
  - Can be changed online
  - OOM is much more destructive

Binding processes to physical CPU

Problems:

- OS can move a process to another NUMA nodes
- The process memory migrates too
- Only for very intensive modifying (not for selects!!!)

Solution 1:

- To tune an OS process scheduler CFS:
  - **sched_min_granularity_ns**
  - **sched_wakeup_granularity_ns**
- It didn't work for us

Solution 2:

- **To bind** RedDatabase processes to a physical CPU
- It works fine

How to find a problem with NUMA

- System time **>>** User time
- High **iowait** (uninterruptible sleep) in top **without** queues to I/O device
- Very fragmented memory (hit and miss are of **the same order of magnitude**)
- It's necessary to analyze **every thread** but not the process

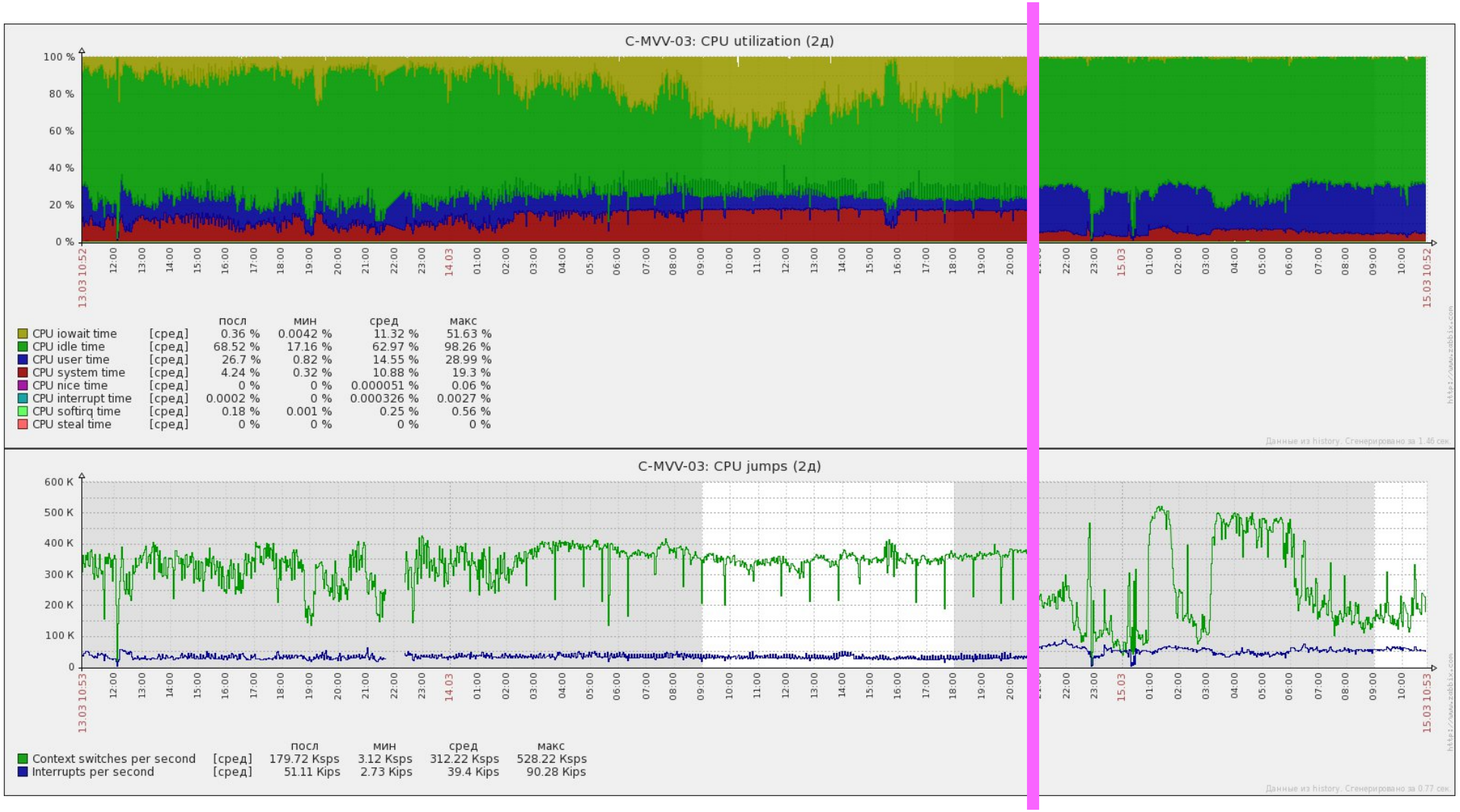
Example (numastat)

```
# numastat
                           node0           node1           node2           node3
numa_hit              35483991701     25907973778     43439858254     23399971622
numa_miss             13235884421     16606874744     16180699231     18803917332
numa_foreign          17220754371     11119698073     24867593917     11619329374
interleave_hit              53690           53720           53709           53739
local_node            35483981570     25907909025     43439790189     23399907916
other_node            13235894552     16606939497     16180767296     18803981038
```

```
# numastat
                          node0           node1           node2           node3
numa_hit             224300115716    293940768830    289461636335    296555251853
numa_miss               516913107       380035968       451756983       486206474
numa_foreign            371950817       457785062       488683152       516493501
interleave_hit              53615           53592           53603           53624
local_node           224299280174    293940256731    289461093452    296554599720
other_node              517748649       380548067       452299866       486858607
```

Example (graphics)

- System time is **less**
- User time is **more**
- iowait time almost **absent**



Fast backup/restore. Method.

- Run tmux and open 2 sessions
- In the first run:

```
gbak -v -y backup.log -g -b db stdout | lzop -1 -o db.fbk.lzo
```

- In the second run:

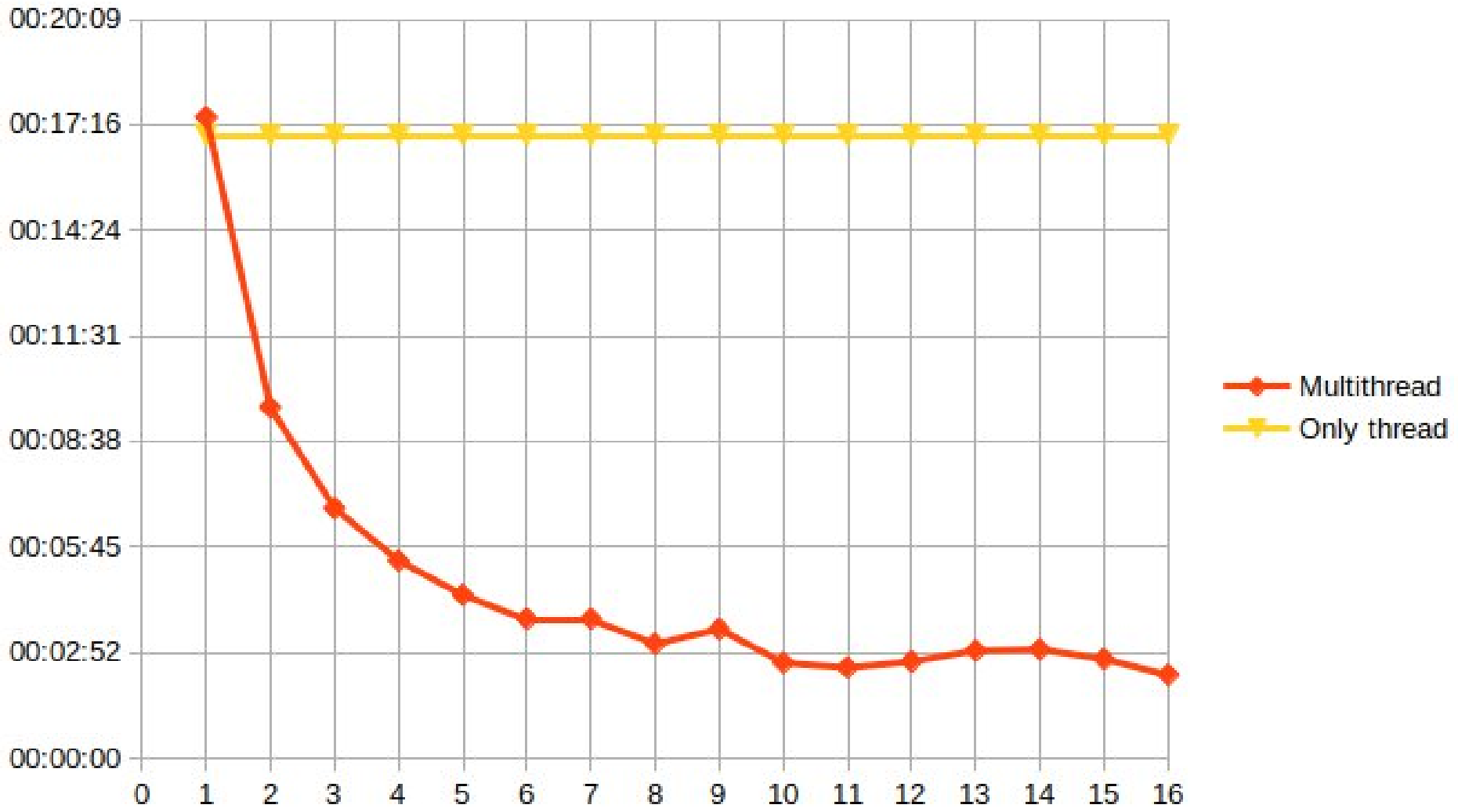
```
tail -n+0 -f db.fbk.lzo | lzop -dc | gbak -v -y restore.log -o -i -r stdin db-br
```

- Watch the end of process via restore.log and kill the second command
- Use **plum** tool (github.com/NeoZX/plume) to activate indecies concurrently

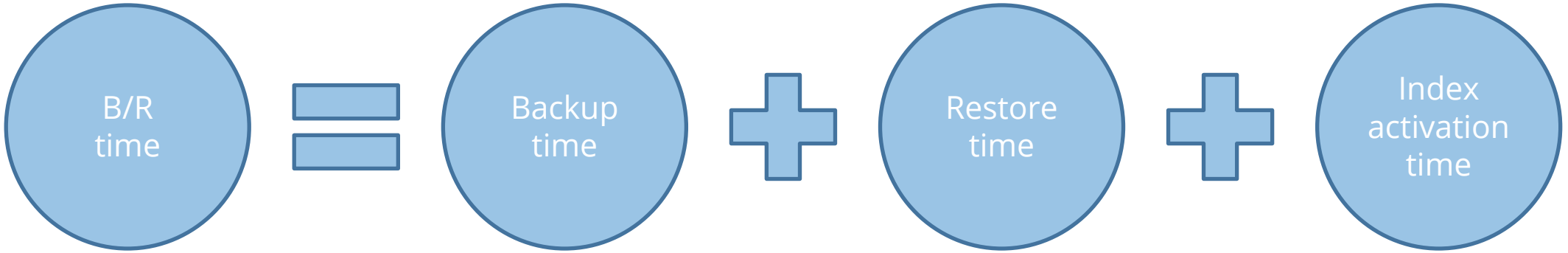
```
plume -u sysdba -p masterkey -t <N> -d localhost:db-br
```

- `N` - a number of threads
- `-ig, -n, etc` can be used like gbak

Fast backup/restore. Results.



Fast backup/restore. Conclusions.



10x faster index activation

3x faster B/R

gbak/nbackup via fifo

- Enterprise backup tools like `bacula` (https://www.bacula.org/)

- GBAK

```
>mkfifo -m 0660 /tmp/database.fbk && chgrp firebird /tmp/database.fbk && \
gbak -g -b localhost:database /tmp/database.fbk && rm /tmp/database.fbk
```

- NBACKUP

```
>mkfifo -m 0660 /tmp/database.${LEVEL}.nbk &&
chgrp firebird /tmp/database.${LEVEL}.nbk && \
nbackup -b ${LEVEL} localhost:database /tmp/database.level.nbk && \
rm /tmp/database.gbk
```

- Now `bacula` can backup `/tmp/database.*`

System improvements

Snapshot request & Garbage collection

- Patch
  - Originally implemented by Nikolay Samofatov (Red Soft)
  - Improved and merged by Vlad Khorsun (Firebird Foundation)

  - Keep only necessary record versions
  - Run request in snapshot

More details in Vlad's talks
https://firebirdsql.org/file/community/conference-2016/statement-level-read-consistency.pdf
New transactions' features and changes in garbage collection in Firebird 4

Improvements in restore via GBAK

- Errors in Restore:
  - is bad (many hours and oopps)
  - can be postfixed
  - must be warnings
  - like indicies

- How it works:
  - RDB$PROCEDURE_BLR is set to NULL for failed objects
  - It keeps dependencies
  - Later the failed should be recreated

Improvements in restore via GBAK. Example.

```
create procedure P1_ROUND (NUM float)
returns (RES float)
as
begin
  RES = round(NUM);
  suspend;
end^

create procedure P2
as
  declare variable VAR float;
begin
  VAR = (select RES from P1_ROUND(4.4564));
end^

alter procedure P1_ROUND (NUM float, SCALE
integer)
returns (RES float)
as
begin
  RES = round(NUM, SCALE);
  suspend;
end^
```

```
gbak:restoring stored procedure P1_ROUND
...
gbak:restoring stored procedure P2
...
gbak:committing metadata
gbak: ERROR:invalid request BLR at offset 50
gbak: ERROR:    Input parameter mismatch for procedure P1_ROUND
...
gbak:Exiting before completion due to errors
gbak:stopped at Thu Oct  3 11:00:11 2019
```

```
gbak:restoring stored procedure P1_ROUND
...
gbak:restoring stored procedure P2
...
gbak:committing metadata
gbak: ERROR:Error while parsing procedure P2's BLR
gbak: ERROR:    invalid request BLR at offset 50
gbak: ERROR:    Input parameter mismatch for procedure P1_ROUND
...
gbak: WARNING:Database is not online due to failure to restore one or
more objects.
gbak: WARNING:Run gfix -online to bring database online.
gbak:stopped at Tue Oct  8 16:48:00 2019
```

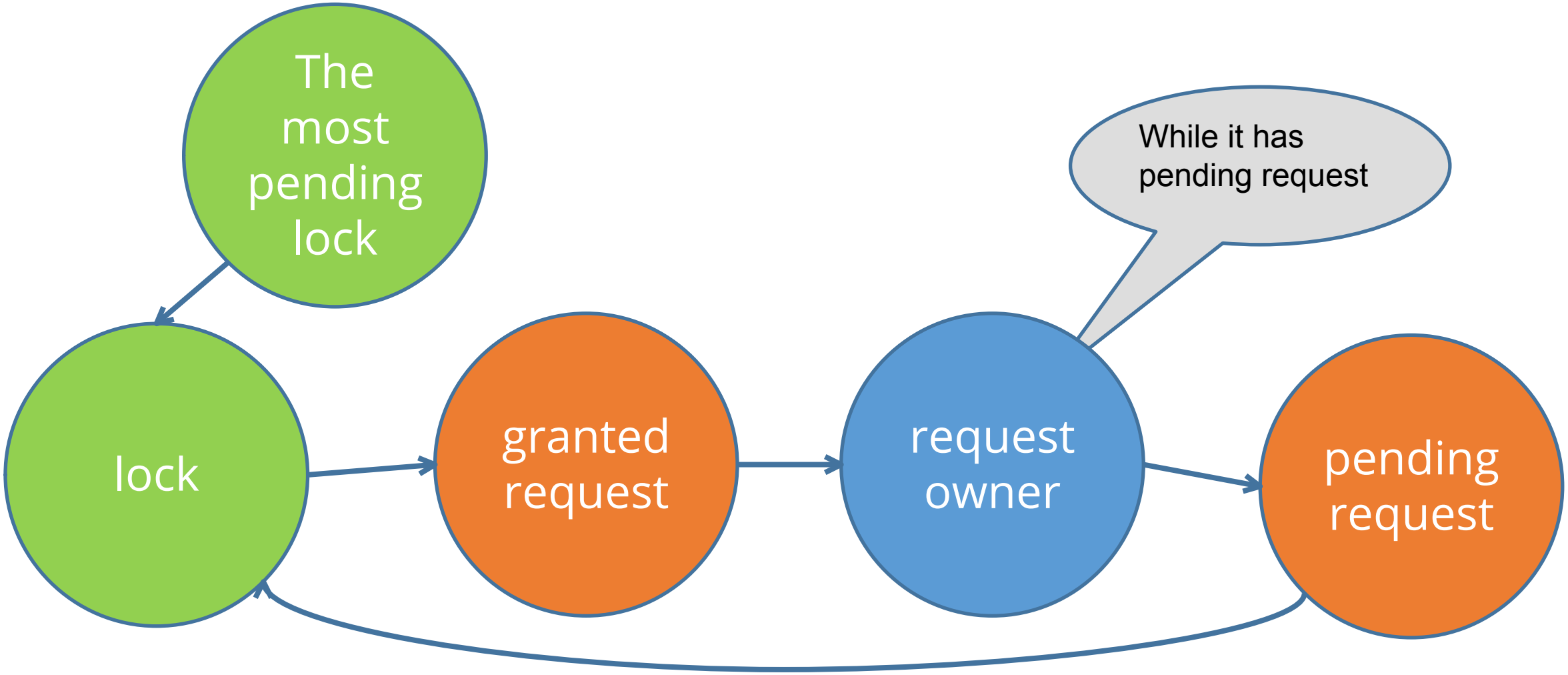
Improvements in restore via GBAK. Fixing.

```
select RDB$PROCEDURE_SOURCE from RDB$PROCEDURES where RDB$PROCEDURE_BLR is NULL;
select RDB$FUNCTION_SOURCE from RDB$FUNCTIONS where RDB$FUNCTION_BLR is NULL;
select RDB$TRIGGER_SOURCE from RDB$TRIGGERS where RDB$TRIGGER_BLR is NULL;
...
ALTER PROCEDURE/FUNCTION/TRIGGER ...
```

Improvements in fb_lock_print

- fb_lock_print -q <seconds>
  - find potential dead owner and request
  - wait the number of seconds
  - find potential dead owner and request again
  - equal requests very often mean the owner is hang
  - gdb helps.

Dead owner search algorithm



Binary trace file

- All events
- The fastest way
- Low overhead

- Big file

Binary trace file. Usage.

- fbtrace.conf

```
database
{
        # Do we trace database events or not
        enabled = 1
        # Log file format (0 - text, 1 - binary, 2 - system)
        format = 1
...
```

- In database

```
CREATE TABLE <table_name> EXTERNAL [FILE] <filespec> ADAPTER 'fbtrace'
[(<col_defs>)]
```

- If col_defs are absent a default set will be used

Binary trace file. Example.

- Select every SQL statement looking for PREPARE STATEMENT event type
- Join with every FINISH STATEMENT on statementID
- Filter only queries longer then 20 ms.
- Aggregate

```
SELECT LLL.SQL, SUM(L.PERF_TIME), AVG(L.PERF_TIME), COUNT(L.PERF_TIME)
FROM log L JOIN
        (SELECT cast (LL.STMT_SQL as varchar(32000)) as SQL, LL.STMT_ID FROM log LL
                        WHERE LL.EVENT_TYPE = 'PREPARE STATEMENT') LLL
        ON LLL.STMT_ID = L.STMT_ID
WHERE L.EVENT_TYPE = 'FINISH EXECUTE STATEMENT' and L.PERF_TIME > 20
GROUP BY 1
ORDER BY 2 desc
```

Binary trace file. Result.

SQL	SUM	AVG	COUNT
select sum(pl.delta_amt) amt, pl.submit_date carry_date, doc.doc_date, doc.doc_number, doc.opertype_id,	549207	61023	9
select sum(bh.day_credit) as Financing from expbudget exp join budgrest br on (br.budgetline_id=exp.budgetline_id) join account a on	443561	62	7042
select sum(bh.day_credit) as Financing from expbudget exp join budgrest br on (br.budgetline_id=exp.budgetline_id) join account a on	427189	71198	1

Load in a laboratory - big challenge

- Record API calls
- Emulate load profile based on trace
- Load applications which uses a database
- **Your ideas?**

Questions?

www.red-soft.ru
www.reddatabase.ru