Recovery and high availability solutions for Firebird

Alex Koviazin, IBSurgeon
www.ib-aid.com

PLATINUM

IBSurgeon

HQBird

Advanced Firebird for Big Databases

- Replication, Recovery and Optimization for Firebird and InterBase since 2002
- Platinum Sponsor of Firebird Foundation
- Based in Moscow, Russia

www.ib-aid.com

What's the problem?

- Hardware fails
- Bugs
- Human factor

3 key things

1. Time to recover
2. % of saved data (since the last valid backup)
3. Chance of unsuccessful recovery

What are the options?

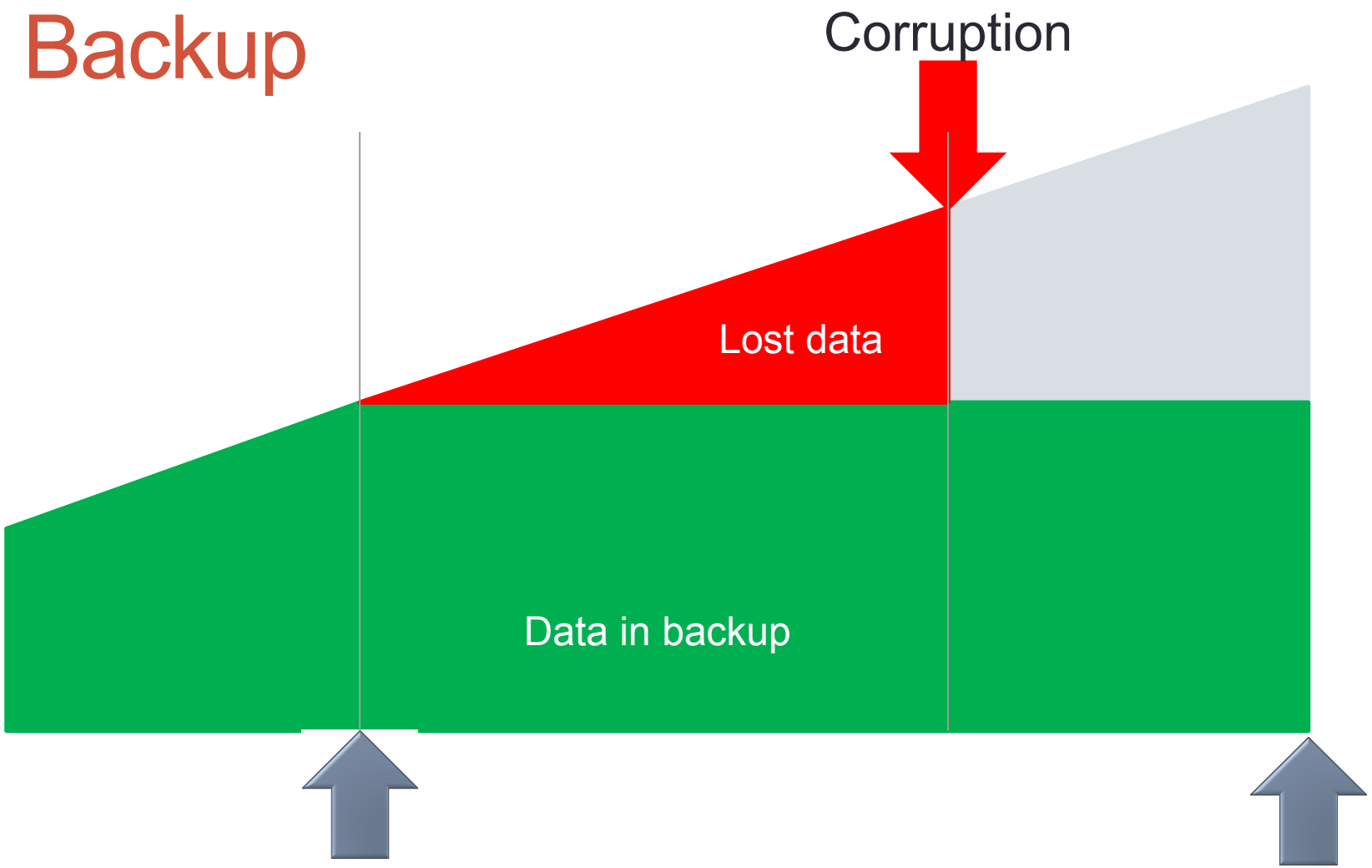
1. Return to the most recent backup
2. Recover corrupted database
3. Virtual Machines High Availability
4. DRBD/Shadow/etc
5. Fail-safe cluster
6. Warm-standby

What are the options?

	Time to recover	% of saved data	Chance of unsuccessful recovery
Return to most recent backup			
Recover database			
High availability solutions			
Virtual Machine Cluster			
DRBD/Shadow/etc			
Failover-cluster			
Warm standby			

1. Back to the backup

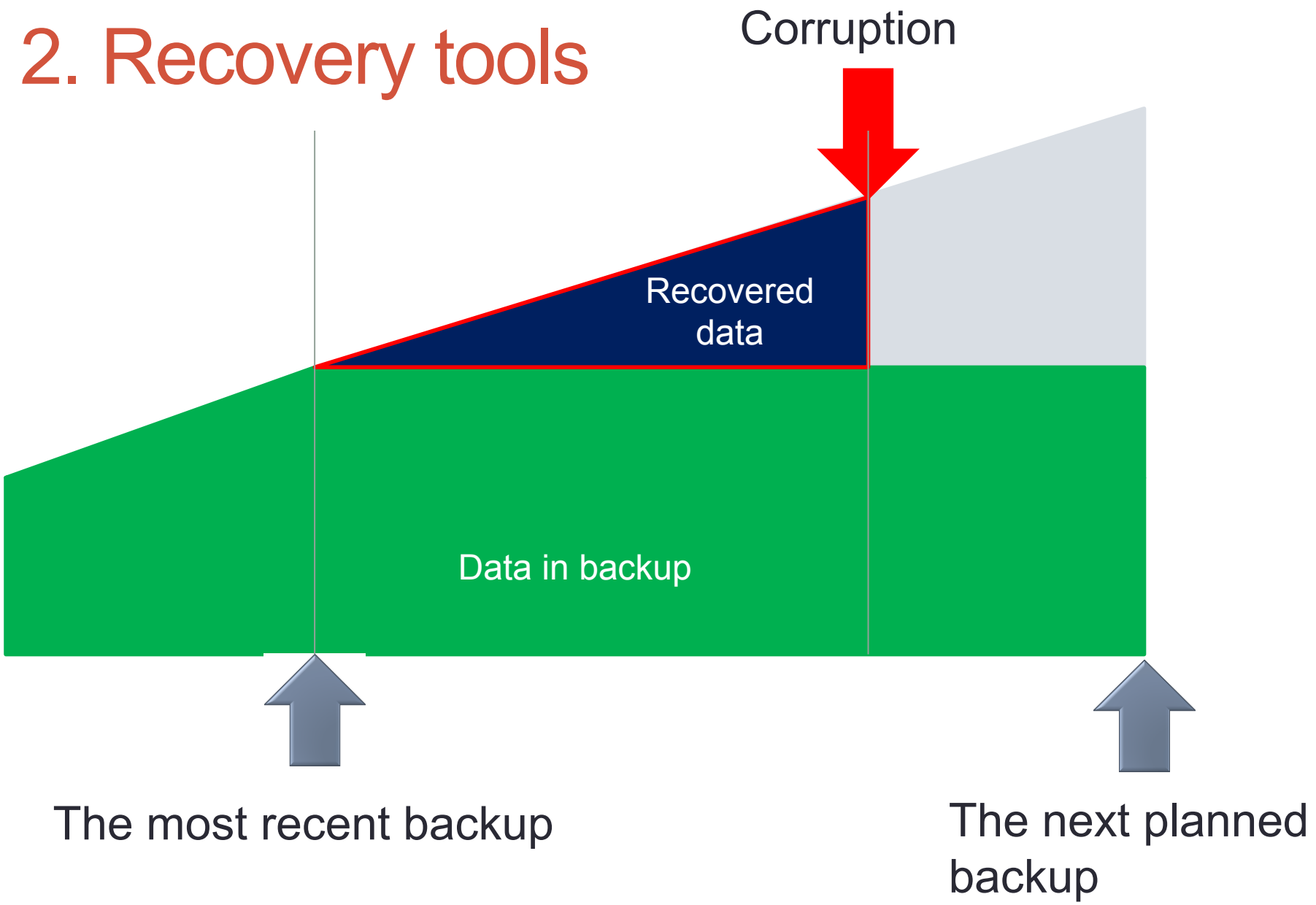
Backup


Corruption
Lost data
Data in backup
The most recent backup
The next planned backup

Backup summary

- Time to recover – several hours
  - Depends on the database size and IO speed
- % of saved data since the last backup – from 0 to 90%
  - Depends on the moment of corruption – the later is corruption, the more data will be lost
- Chance of unsuccessful recover – low
  - Check 12 Common Mistake in Databases Backups

2. Recovery tools



Repair corrupted Firebird database

- There are tools to recover corrupted databases:
  - Standard tools: gbak+ gfix
  - Advanced tools: **IBSurgeon FirstAID**
- Time to recover – from 1minute per Gb, several hours in average
  - Depends on IO and CPU
  - Often requires backup/restore or full data pump
- % of saved data – 70-99%
- Can be combined with the recent backup
  - Increases % of saved data to 90-99%

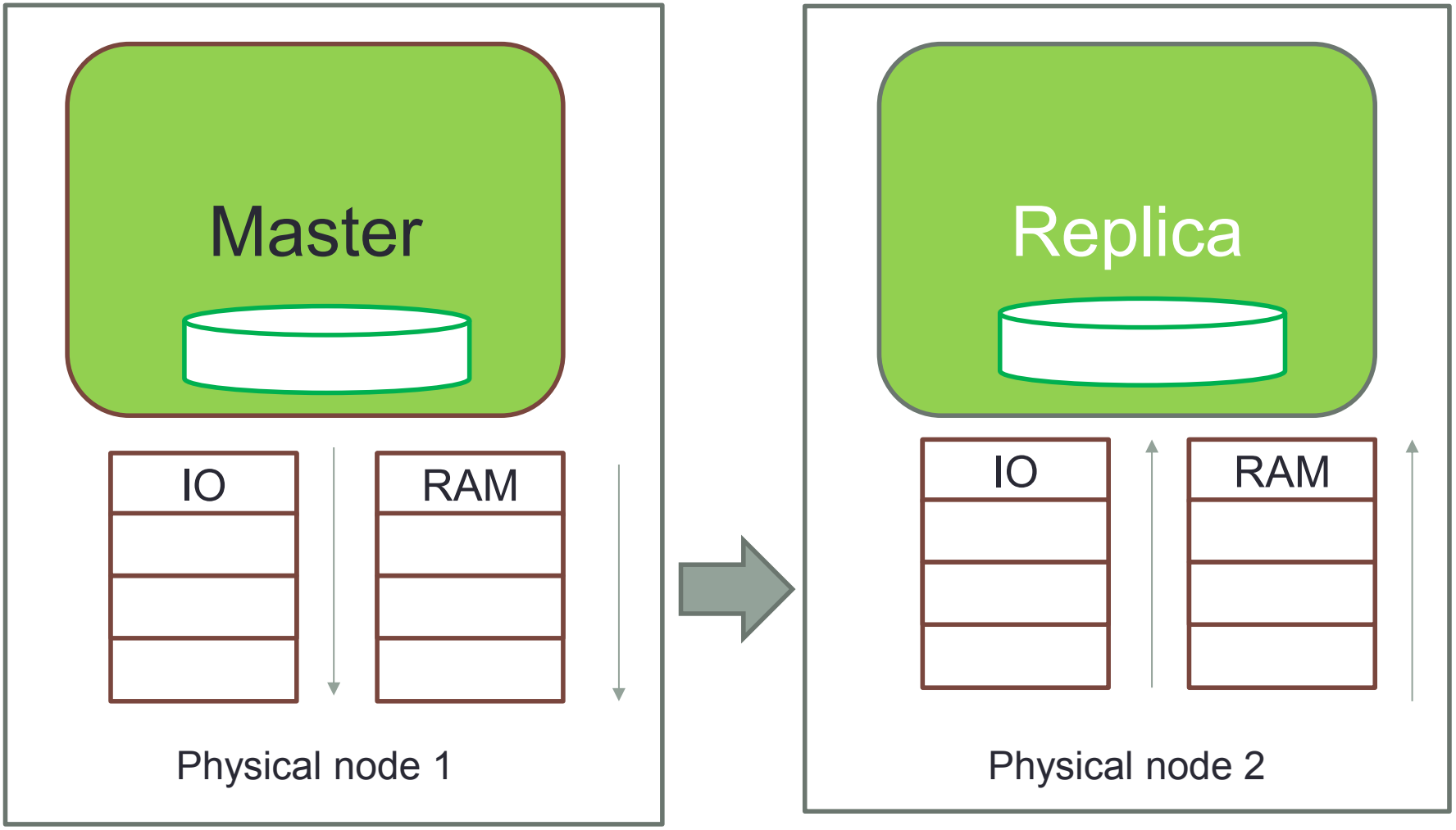
Recovery tools summary

- Time to recover – several hours
  - Usually it is longer than restore from backup
  - Depends on the database size and IO speed
- % of saved data since the last backup – from 70 to 90% in average
  - Depends on the difficulty
- Chance of unsuccessful recover – medium
  - Some corruption are unrepairable
  - Lower in combination with backup

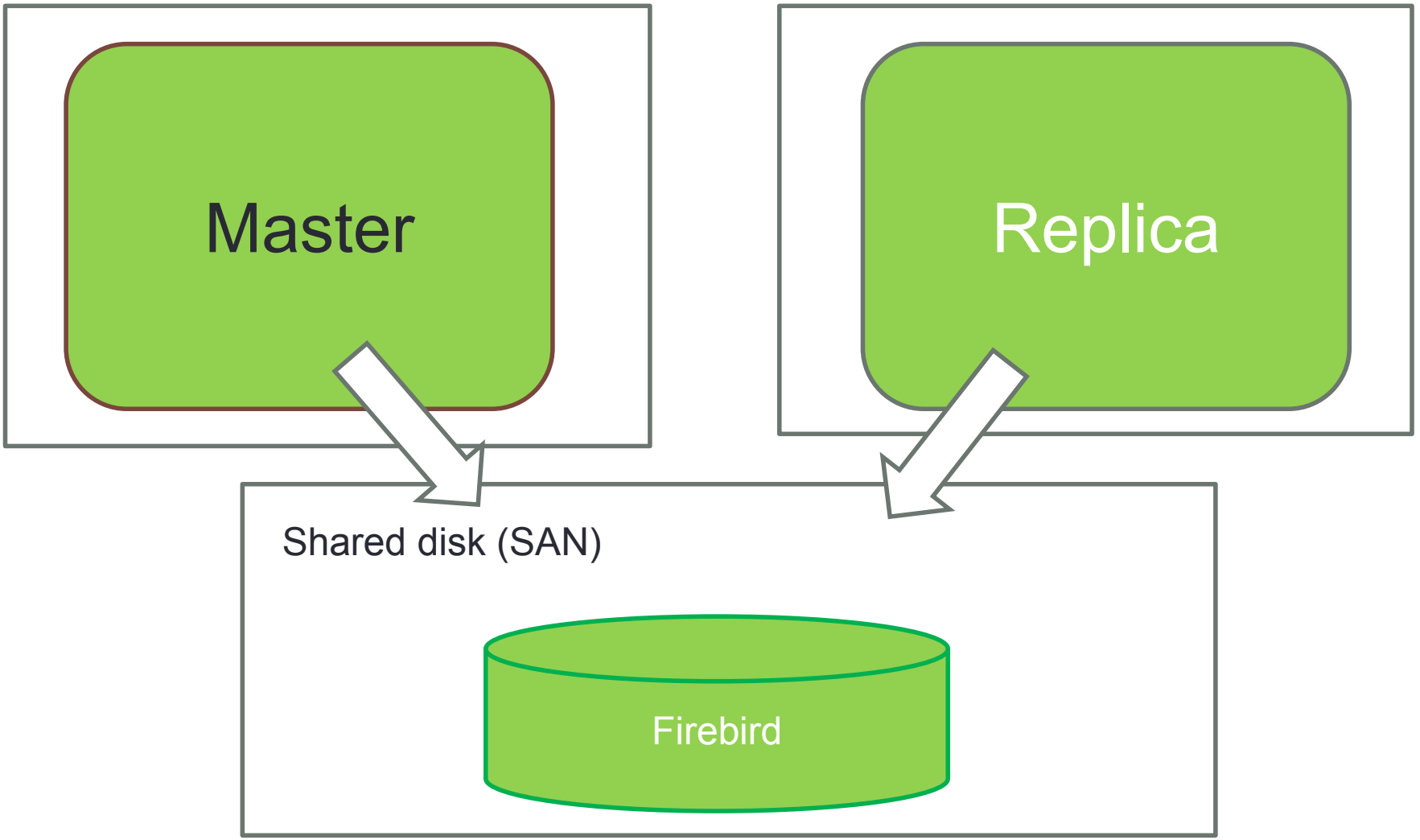
3. Virtual Machines

- High Availability with VMs
- Continuous backup

Virtual Machines High Availability-1



Virtual Machines High Availability-2



VM HA

- Designed to protect against VM failures, easy migration to more/less powerful VMs (Live Migration), easy reconfiguration of VMs
- No validity checks on database level
- All errors will be replicated
- Cannot be used as protection against corruption

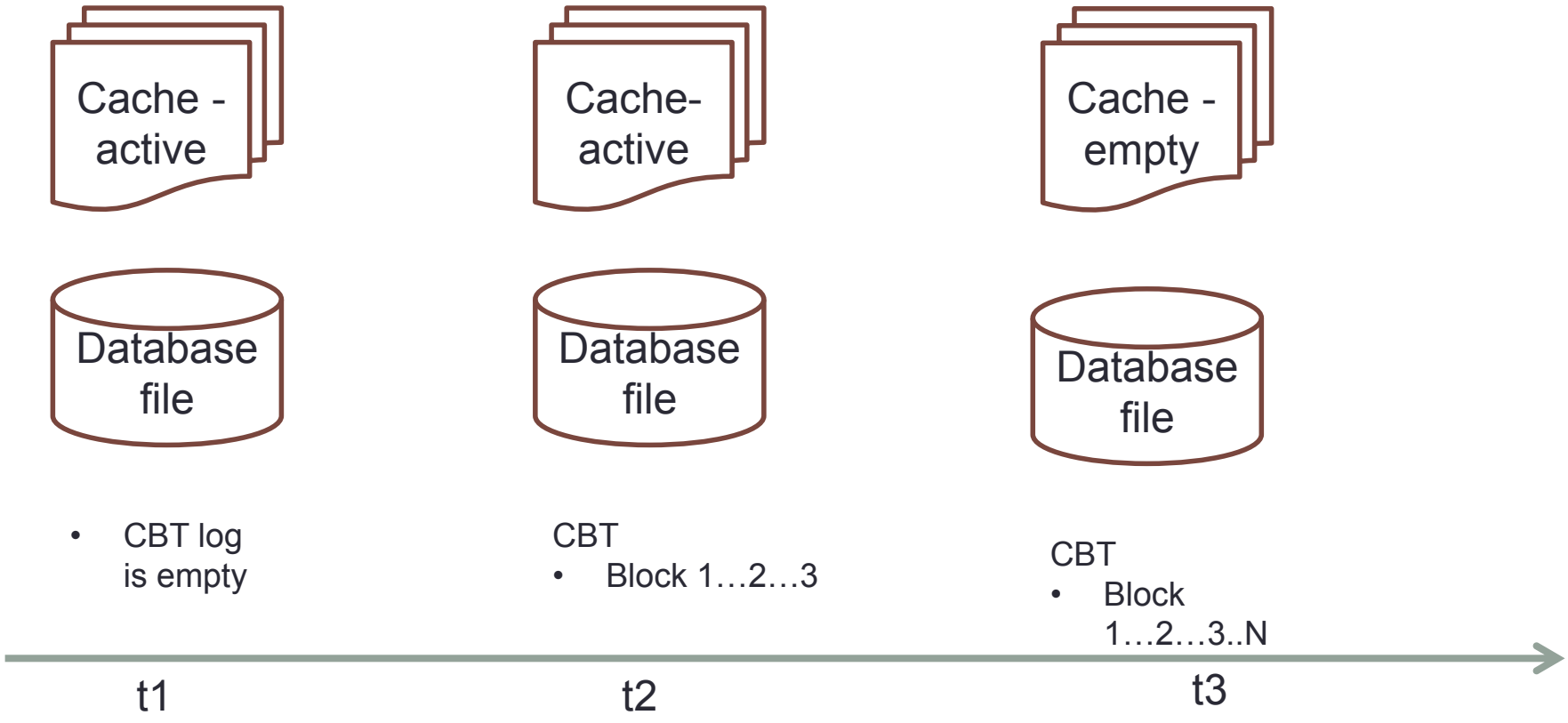
Continuous VM backup (based on Changed Block Tracking or similar)

- Virtualization vendors declare ability of “transparent backup”

- Administrators believe in Changed Block Tracking – tracking of file changes for “uninterruptable backup”

How Changed Block Tracking Works

- Change of virtual disks (VMDK) is being tracked. Files “-ctk.vmdk” contain numbers of changed blocks.
- CBT-filter is used not to cache all changes
- CBT **is not** streaming translation of changes
- Parsing of changed blocks – task for backup tool



1st snapshot. DB file is not consistent, backup copy is not consistent, CBT list is empty

Analyzing and applying changes, that accumulated in CBT. DB file is not consistent because of active changes in cache. Updated backup is not consistent, CBT list is not empty

DB file is consistent at the moment of CBT check, after merge of changes backup copy is consistent too, CBT list is not empty

Restrictions of CBT for databases

- 1st snapshot must be made from stable DB copy
  - Without active write cache, or
  - In special snapshot-friendly state (VSS or archivemode, or with nbackup for Firebird)
- Backup actualization must be made at the moment, when the original file is stable (no active cache)
  - Same as taking 1st snapshot
- If write cache is active, consistency of the copy is not guaranteed - result will be like state after sudden power off (hard reset)

Virtual Machine HA and Backups

- Virtual Machine vendors created a big mess in HA and Backup area
- VM backup is essentially nbackup, taken in very inconvenient way
- In essence, **VMs are useless** for the protection against databases failures

VM backup summary

- Time to recover – from minutes to hours
  - The same as in nbackup
- % of saved data since the last backup – from 0 to 100% in average
  - Depends on the time when corruption happened and intensity of IO
- Chance of unsuccessful recover – high
  - Some VM backups do not contain valid database – recovery scenario jumps in

4. Disk-level replication

1. DRBD
2. Shadow

DRBD

- Solution
  - Distributed replicated block device
  - Kind of remotely mirrored filesystem
  - Supports synchronous and asynchronous modes
  - Open-source (GPL) and free
- Issues
  - Replica cannot be validated online (not accessible)
  - Linux only
  - Writes are doubled

Shadow copy

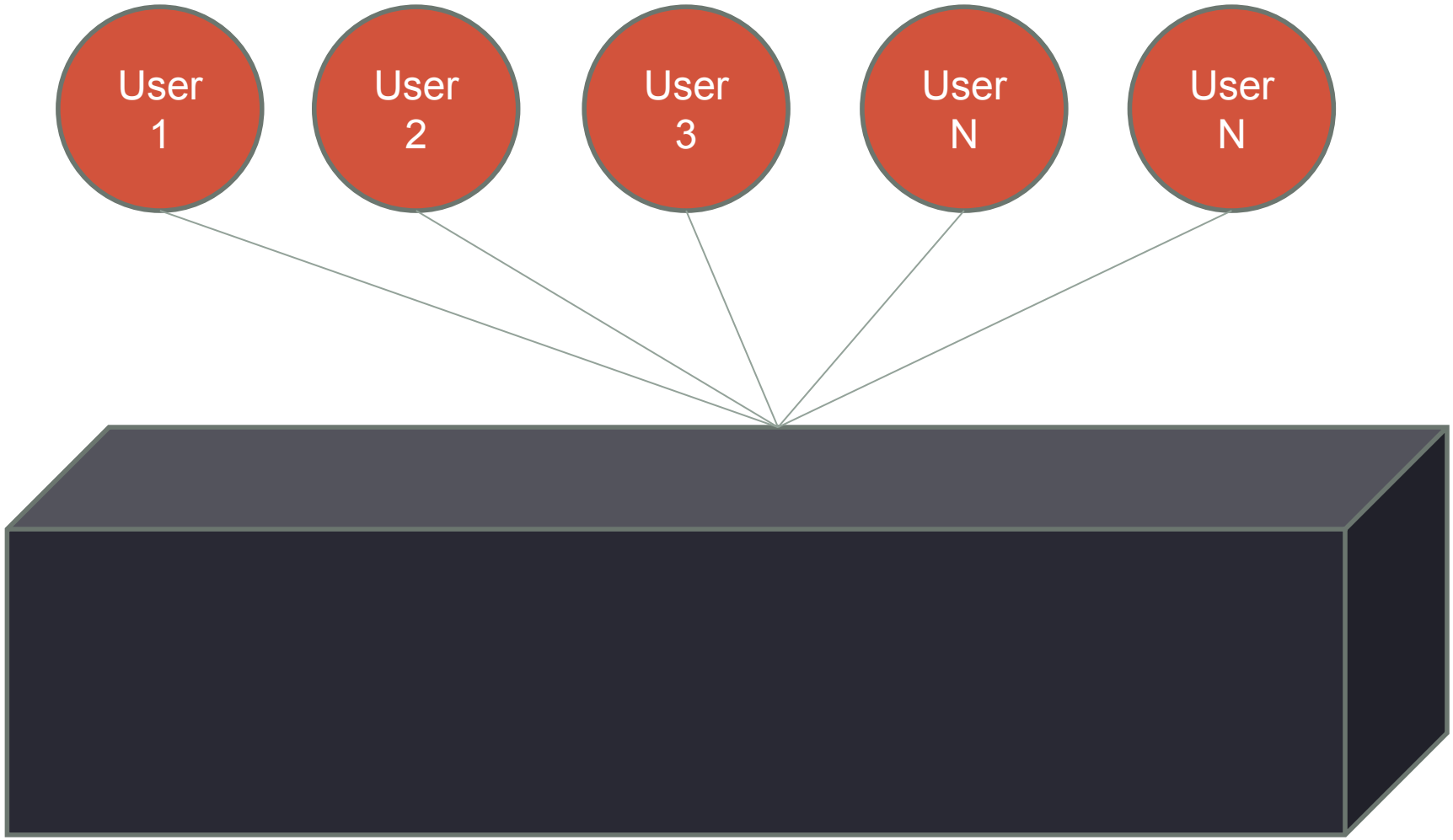
- Solution
  - CREATE SHADOW command
  - Shadow is mounted via NFS / SMB
  - RemoteFileOpenAbility → true
  - Synchronous
  - Platform independent
- Issues
  - Writes are doubled, errors will be distributed
  - Shadow is not accessible

DRBD/Shadow summary

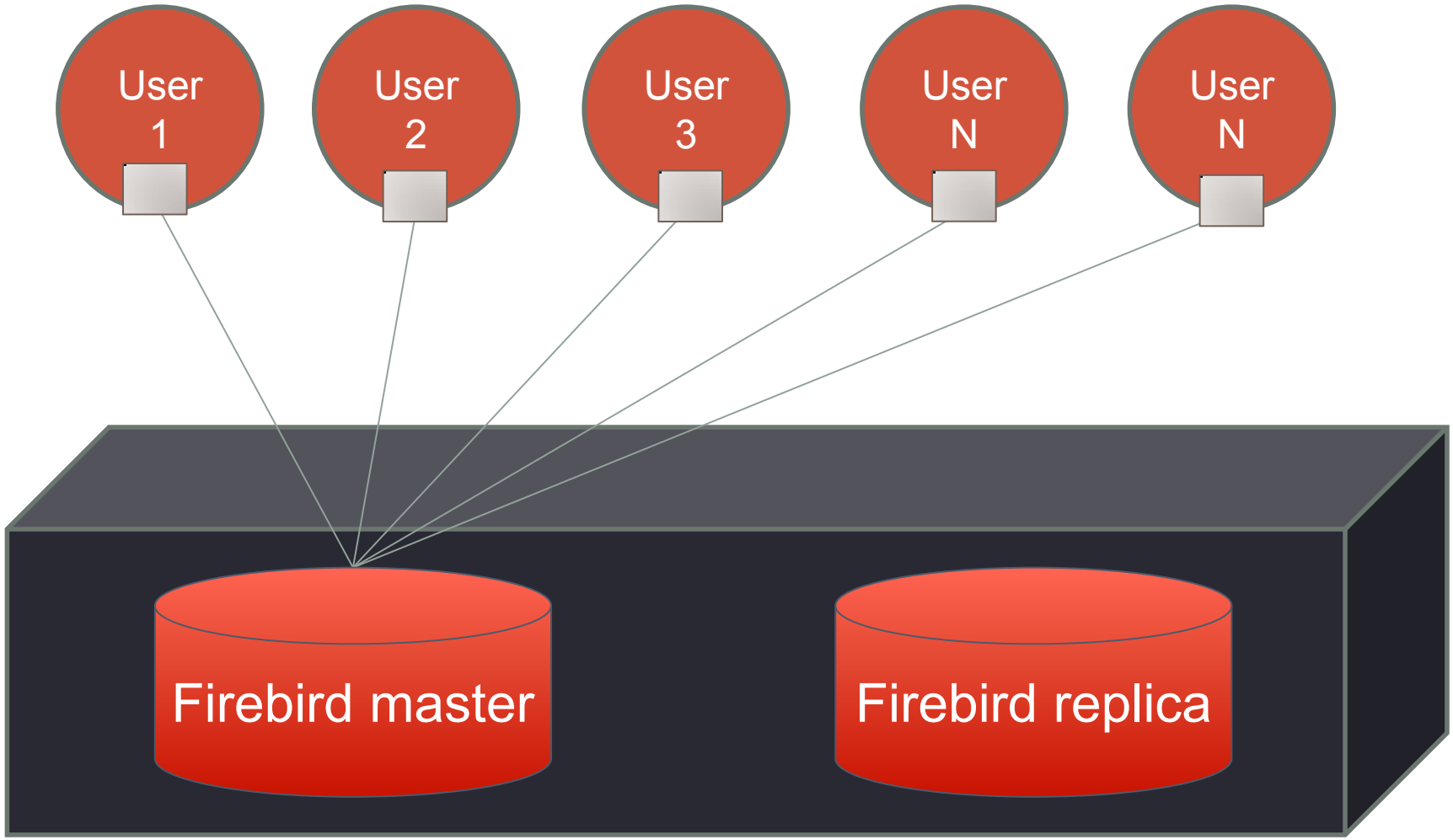
- Time to recover – from minutes to hours
  - Requires administrator assistance
- % of saved data since the last backup – from 0 to 100% in average
  - Depends on the corruption type
- Chance of unsuccessful recover – medium
  - Status is unknown till the restore

FAIL-SAFE CLUSTER AND SOMETHING MORE

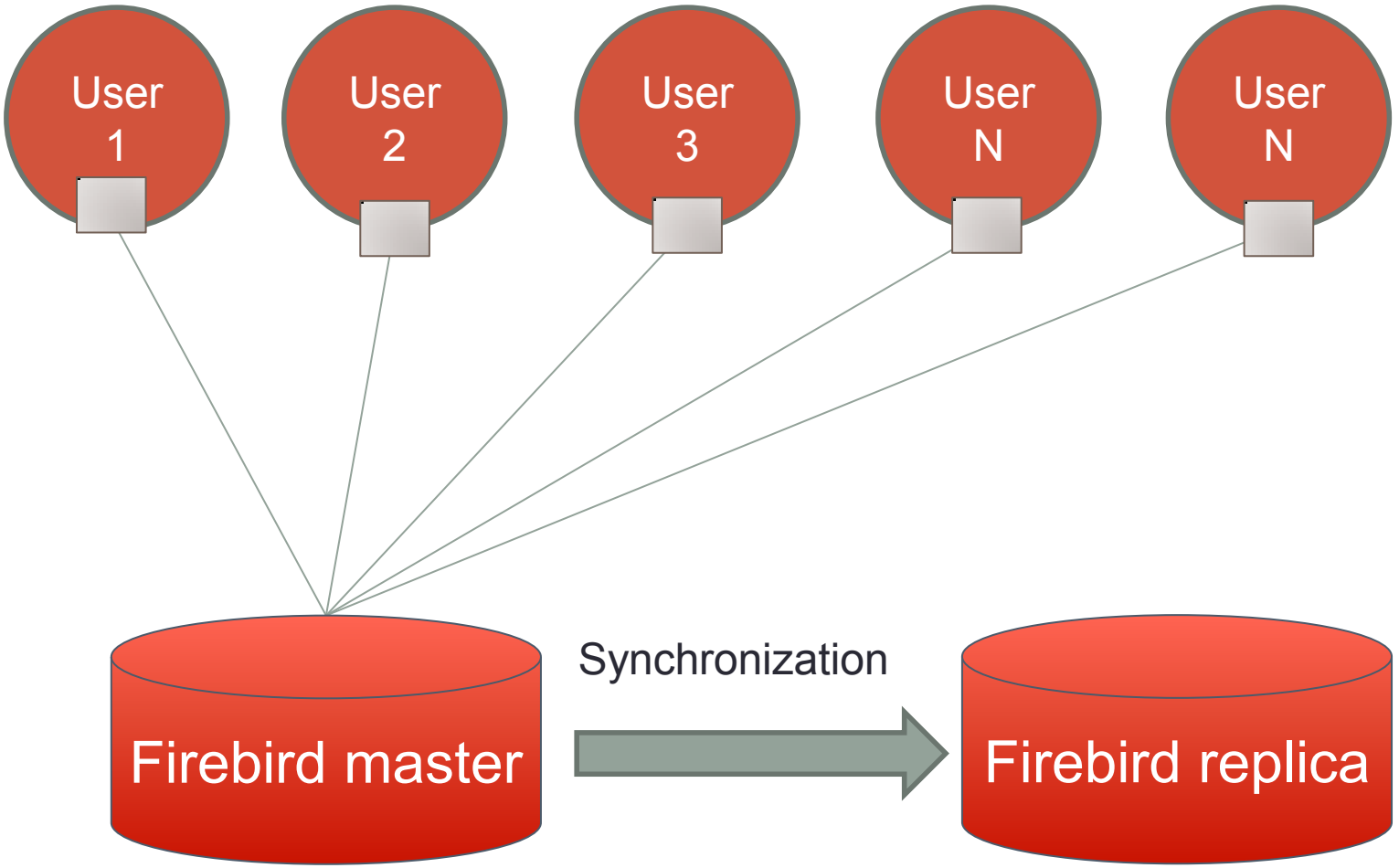
What is a fail-safe cluster?



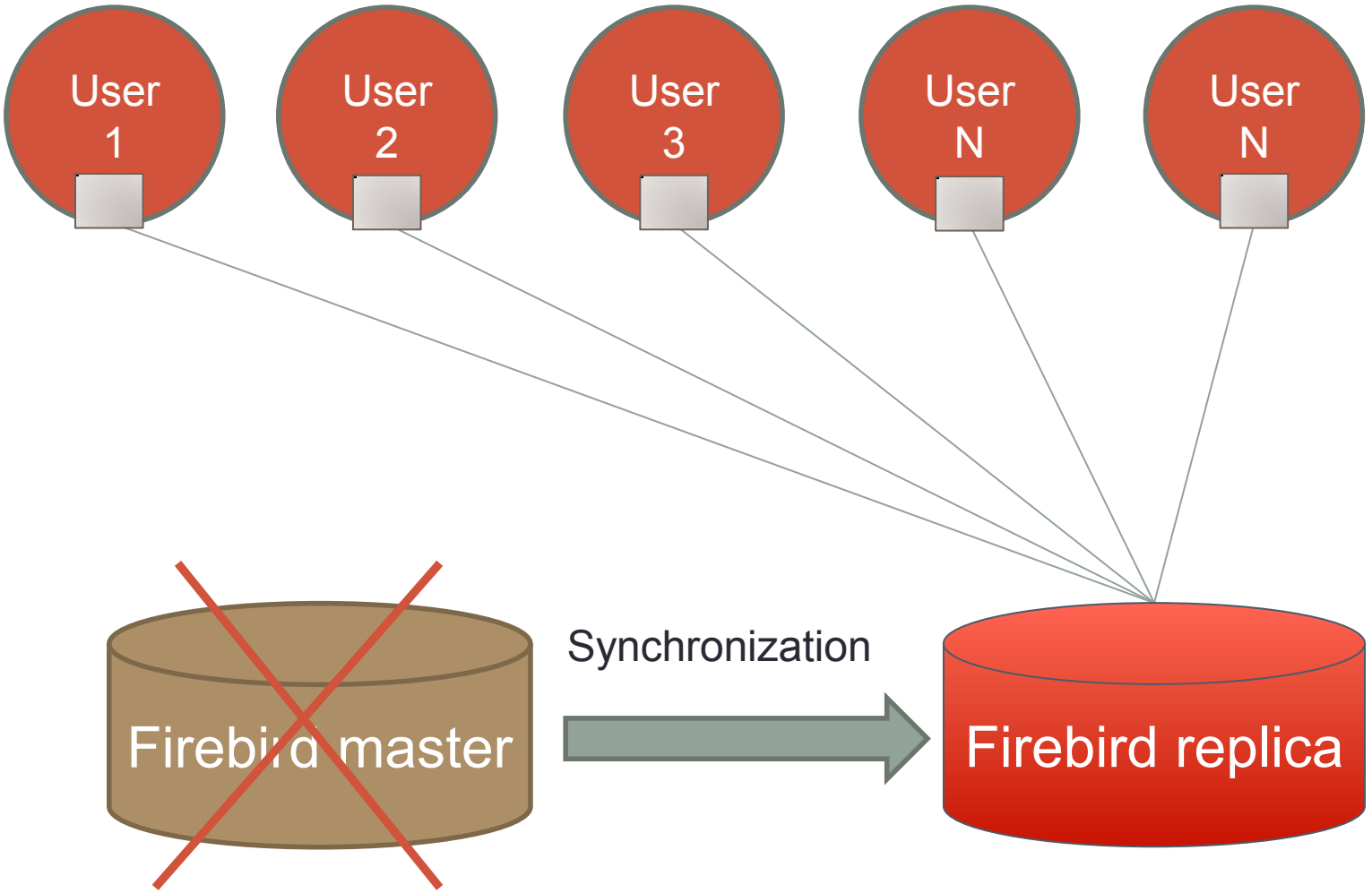
What is a fail-safe cluster internally?



What is a fail-safe cluster?



When master fails, users reconnect to new master



For Fail-Safe Cluster we need

1. At Server Side
   - Synchronization of master and replica
   - Monitoring and switching mechanism
2. At Application Side
   - Reconnection cycle in case of disconnects
     - web server is good example

Database fail-safe cluster is NOT:

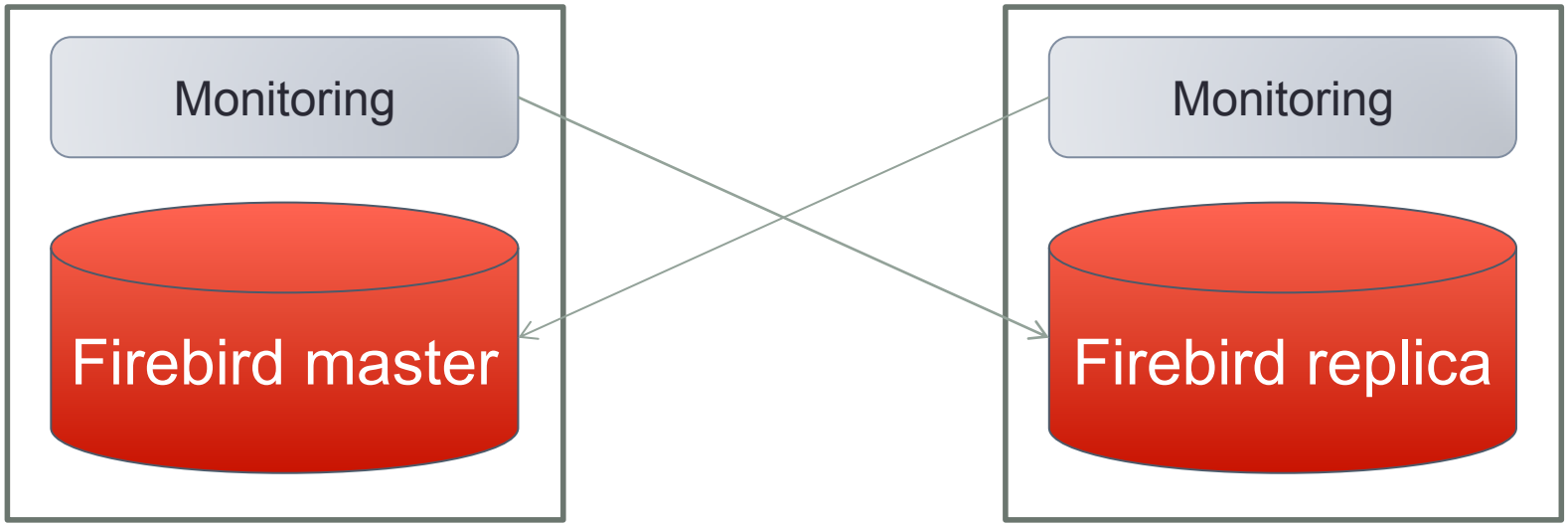
1) Automatically deployed solution
   - Complex setup (2-3 hours of work)
2) True scalable solution
3) Multi-master replication
   - Replicas are read only
4) Geographically distributed solution
   - Fast network connection required

How Fail-Safe Cluster Work: Applications

From the application point of view

1. End-user applications connect to the Firebird database, work as usual
2. If Firebird database becomes unavailable, end-user applications try to reconnect to the same host (correctly, no error screens)
3. End-user applications should better use caching (CachedUpdates) to avoid loss of uncommitted data

How Fail-Safe Cluster Work: Server

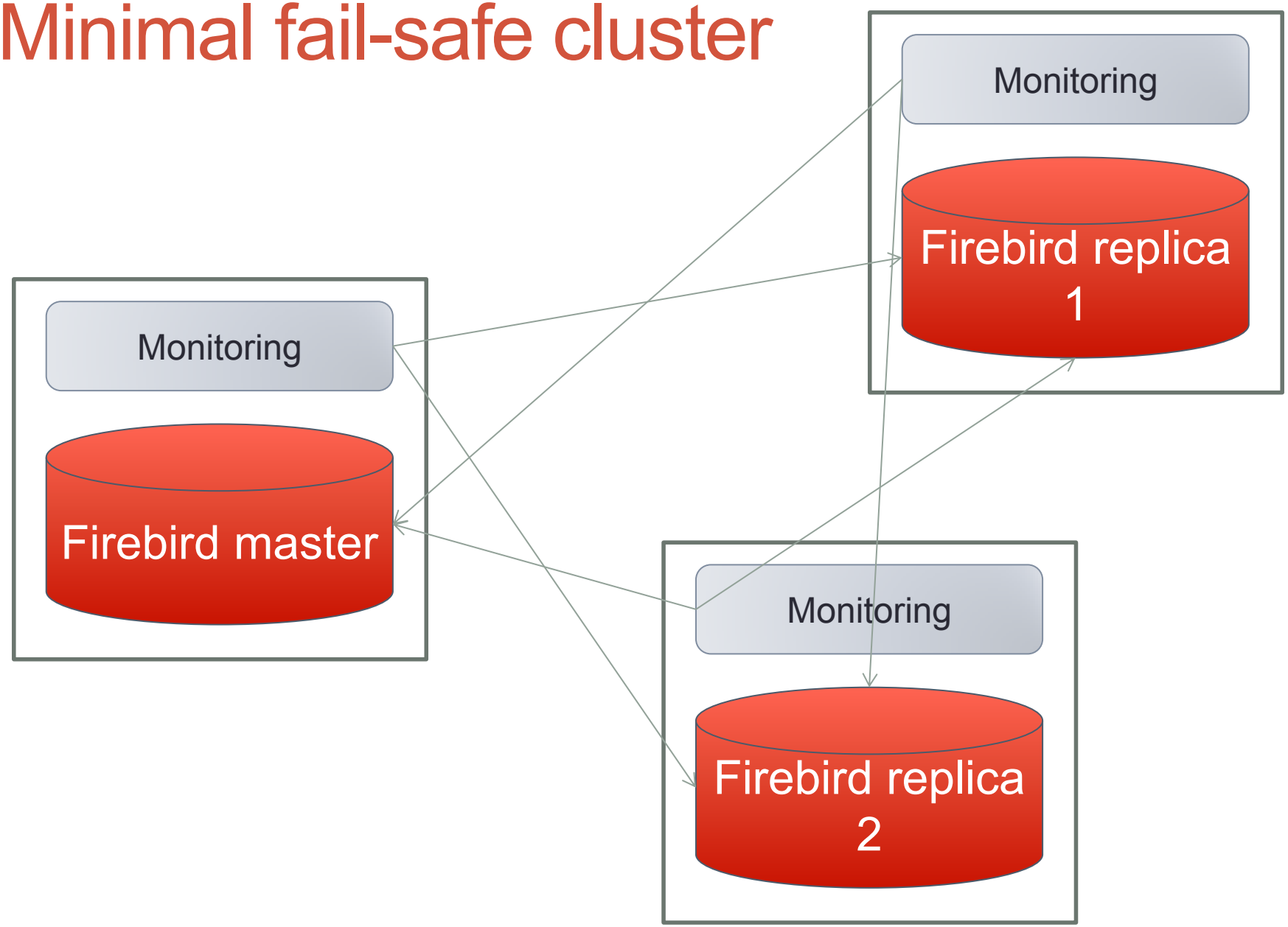


1. All nodes monitor each other: replica watches for master
2. If master fails, replica
   a) make “confirmation shoot” to master
   b) Changes itself as master – script + DNS

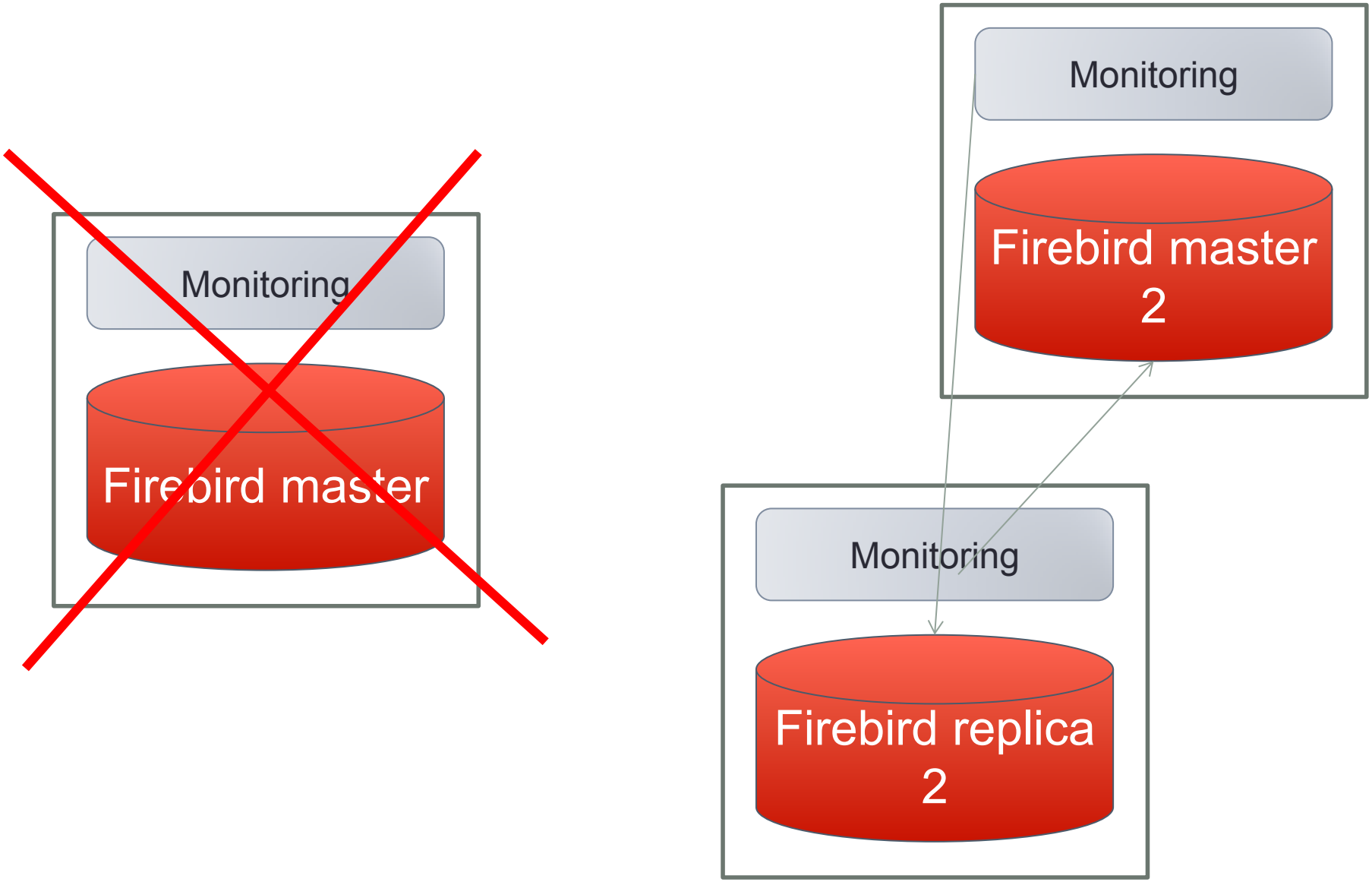
After replica fails, there is no cluster anymore!

Just a single database!

Minimal fail-safe cluster



Minimal fail-safe cluster



What is fail-safe cluster good for?

- For web-applications
  - Web caches data intensively
  - Tolerant to reconnects
  - Many small queries
- For complex database systems with administrator support

HOW TO CREATE FAIL-SAFE CLUSTER IN FIREBIRD

2 main things to implement fail-safe cluster in Firebird

1. Synchronization between master and replica
2. Monitoring and switching to the new server

Synchronization

Native

- Suitable for high-load
- Easy setup
- Strict master-slave, no conflicts
- DDL replication
- Very low delay in synchronization (seconds)

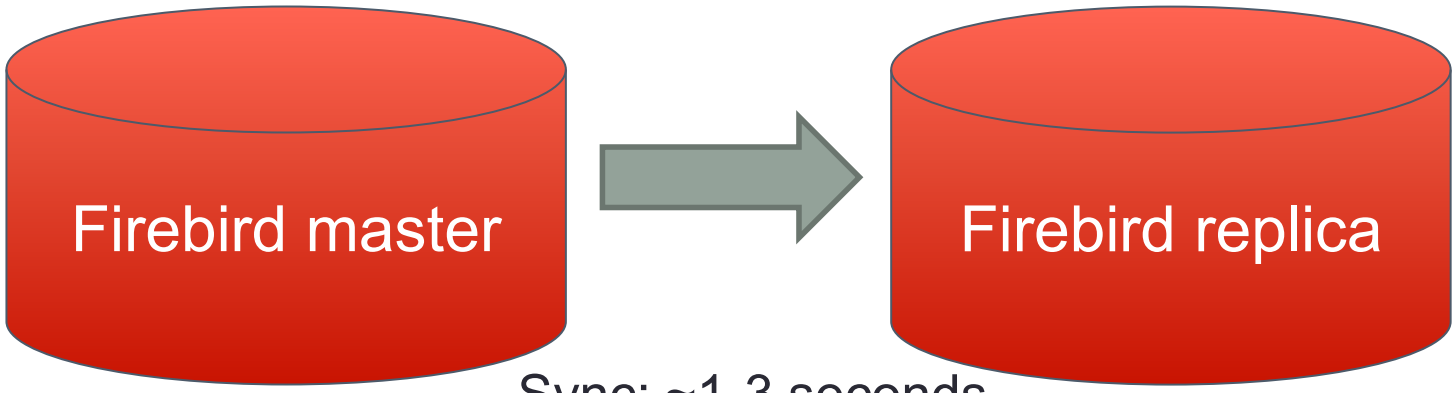
Trigger-based

- <200 users, log table is a bottleneck
- Difficult setup
- Mixed mode, conflicts are possible
- No DDL replication
- Delay is 1-2 minutes

Comparison of Firebird versions with replication

	HQbird	**RedDatabase**	**Avalerion**	**Firebird 4**
Status (October 7)	Production	Production	Production	Alpha
Supported Firebird version	2.5, 30	2.5	3.0	4.0
ODS	11.2, 12.0	11.3	13.1	13.0
How to migrate from Firebird 2.5 and 3.0	Zero efforts (same ODS)	Backup - Restore	ALTER statements	Backup/ Restore
Synchronous	Yes	Yes	No	Yes
Asynchronous	Yes	Yes	Yes	Yes

Native replication in HQbird



1. No Backup/restore needed
2. Sync and Async options
3. Easy setup (configuration only)

Steps to setup replication

SYNCHRONOUS

1. Stop Firebird
2. Copy database to replica(s)
3. Configure replication at master
4. Configure replication at replica
5. Start Firebird at master and replica

ASYNC

1. Stop Firebird
2. Copy database to the same computer
3. Configure replication at master
4. Start Firebird
5. Copy and configure at replica(s)

Configuration for replication

- GUI interface
- Demo

Downtime to setup replication

- Synchronous
  - Time to copy database between servers + Time to setup
- Asynchronous
  - Time to copy database at the same server
  - Suitable for geographically distributed systems

ASYNC is much faster to setup!

Monitoring and switching from master to replica

- HQbird (Windows, Linux)
- Pacemaker (Linux only)
- Other tools (self-developed)

Elections of the new master

- Replicas must select new master after old master failure

Priority list

MASTER1

REPLICA1

REPLICA2

REPLICA3

REPLICA4

Algorithm

1. We have priority list: master, replica1, replica 2
2. Each node watches for all other nodes
3. If master node fails, the first available replica takes over the
    1. Disables original master – stop Firebird server
    2. Changes DNS of replica to master
    3. Changes other servers configurations
    4. Restart Firebird server at new master to changes take effect
    5. Send notifications

After failure – re-initialization

- Downtime needed!
- Stop all nodes
- Copy database to the new/reinitialized node
- Reconfigure replication configuration
- Start Firebird at all nodes

Does it look too complex?

Fail-Safe Cluster is a custom solution

- HQbird gives the options, you decide how to implement them

- Requires careful planning and implementation
- Requires high-quality infrastructure
  - Requires 2+ replicas
  - Network outage in 10 seconds will cause all replicas to degrade!
  - High-speed network and high speed drives

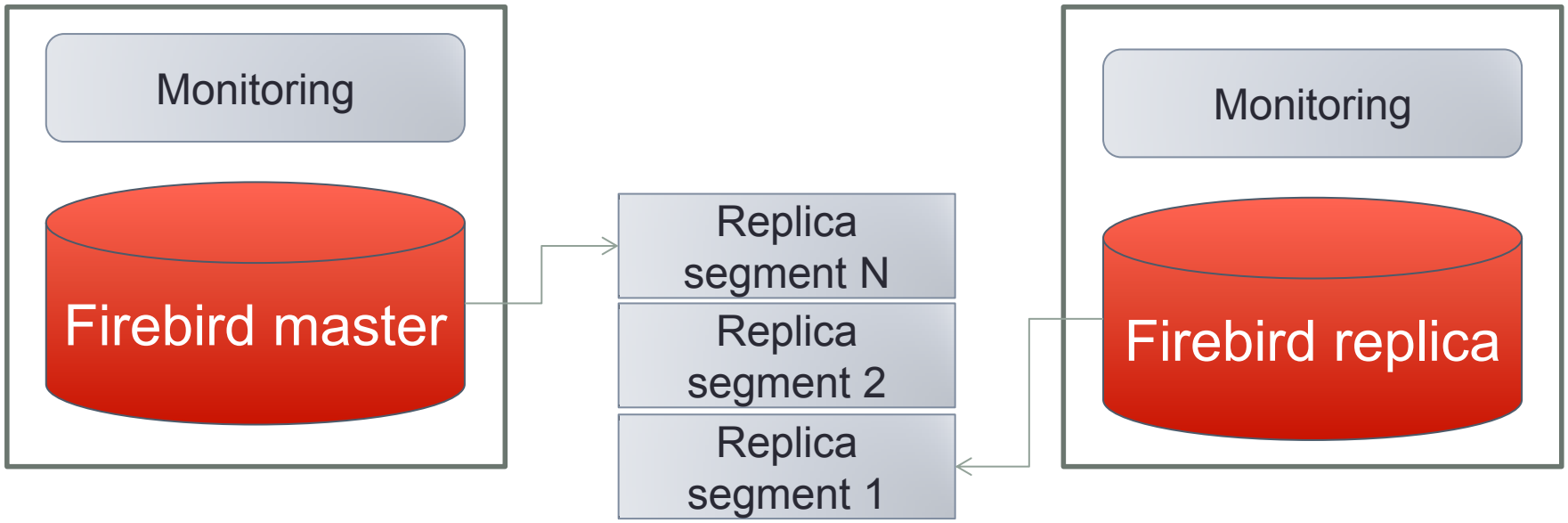
Fail-safe cluster summary

- Time to recover – 30-120 seconds
  - Client application must reconnect
- % of saved data since the last backup – 100%
  - Only uncommitted changes lost
- Chance of unsuccessful recover – low

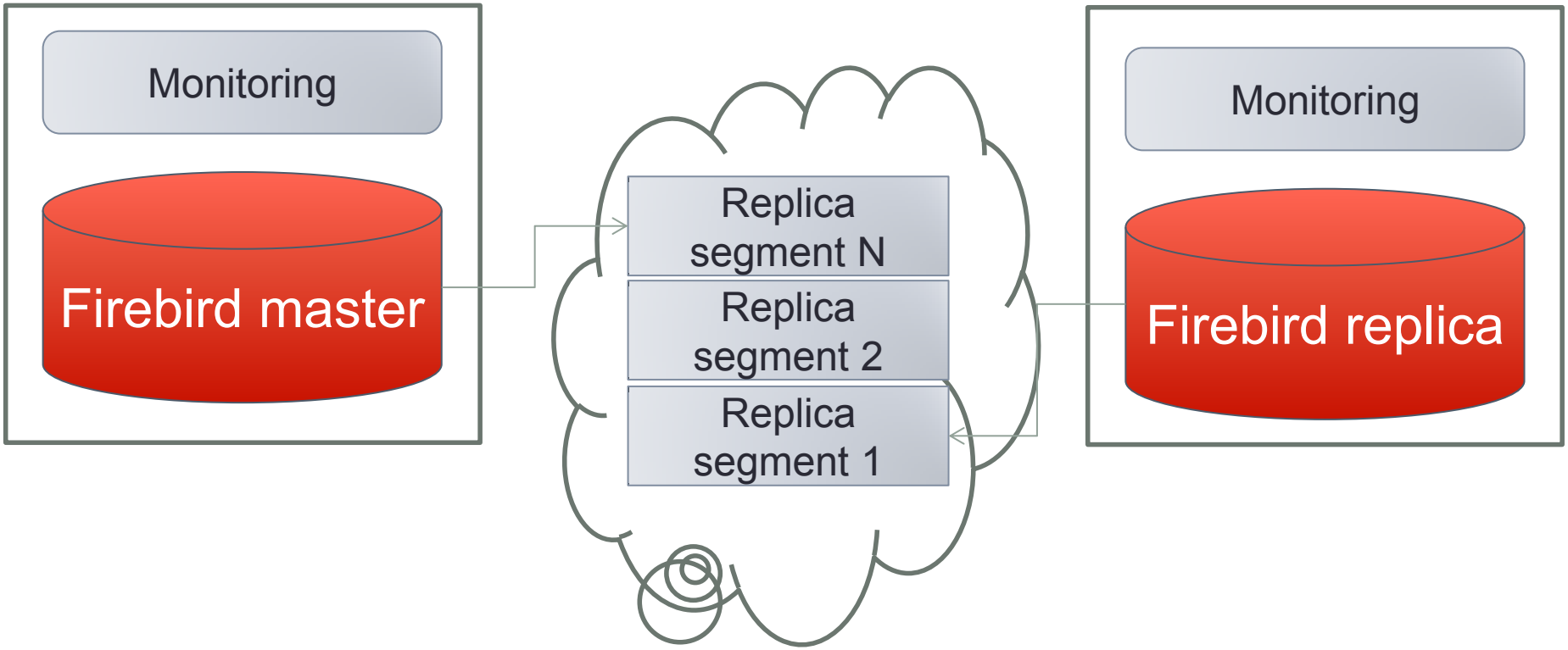
The main problem with fail-safe cluster

Warm-standby (mirror)

- Based on async replication
- Easy and faster to setup and configure



Can be geographically distributed (FTP, Amazon, cloud backup, VPS)



Warm-standby summary

- Time to recover – several minutes
  - Client application must reconnect
- % of saved data since the last backup – 100%
  - Only uncommitted changes lost
- Chance of unsuccessful recover – low

What are the options?

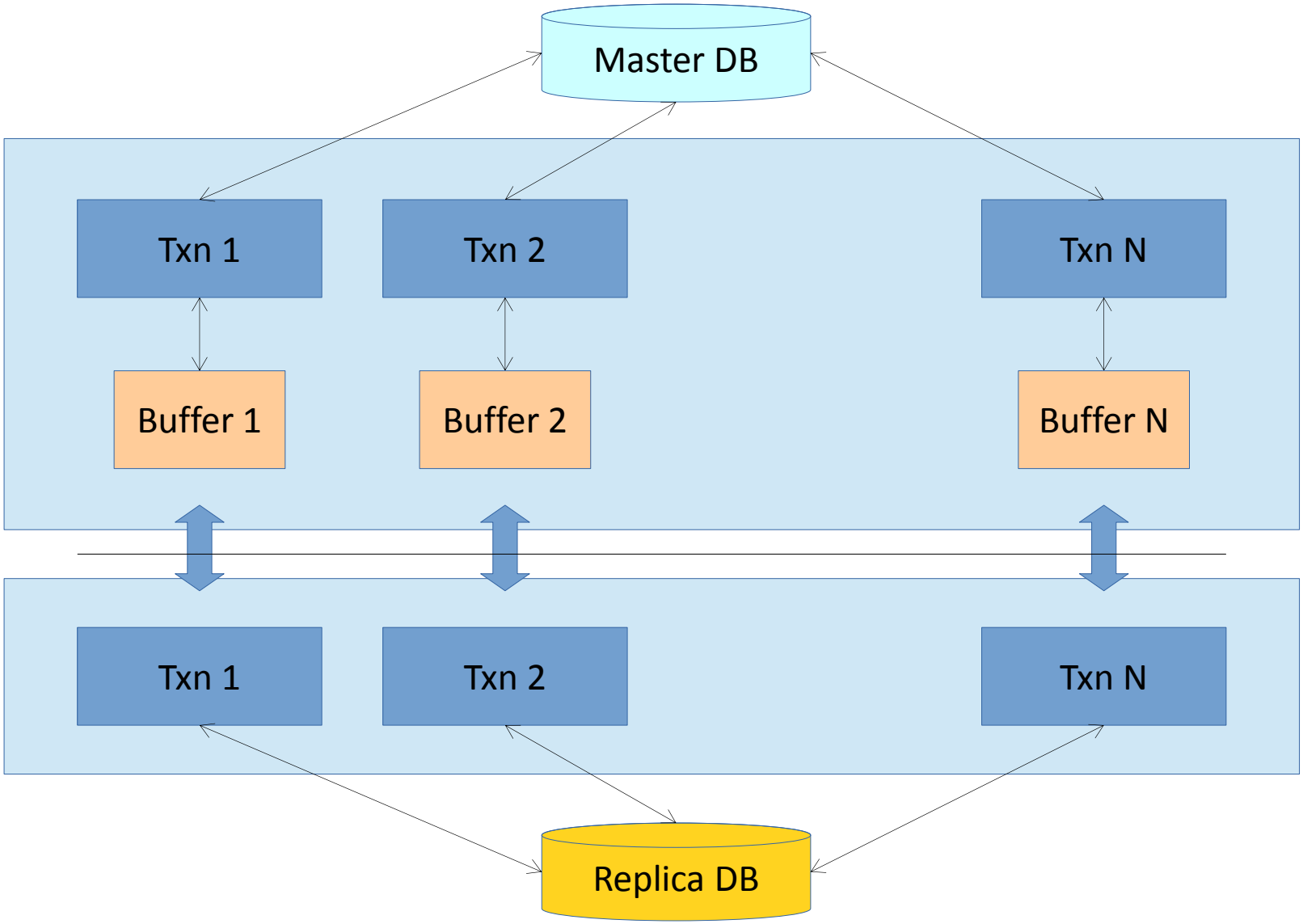
	Time to recover	% of saved daily data since last backup	Chance of unsuccessful recovery
Return to most recent backup	Hours	0-90%	Low
Recover database (in combination with backup)	Hours	50-99%	Medium
High availability solutions			
Virtual Machine High Availability	Minutes	0-99%	High
DRBD	Minutes	0-99%	Medium
Failover-cluster	Seconds	100%	Low
Warm standby	Minutes	100%	Low

Thank you!

Synchronous replication

- Solution
  - Changes are buffered per transaction, transferred in batches, synchronized at commit
  - Follows the master priority of locking
  - Replication errors can either interrupt operations or just detach replica
  - Replica is available for read-only queries (with caveats)
  - Instant takeover (Heartbeat, Pacemaker)
- Issues
  - Additional CPU and I/O load on the slave side
  - Replica cannot be recreated online

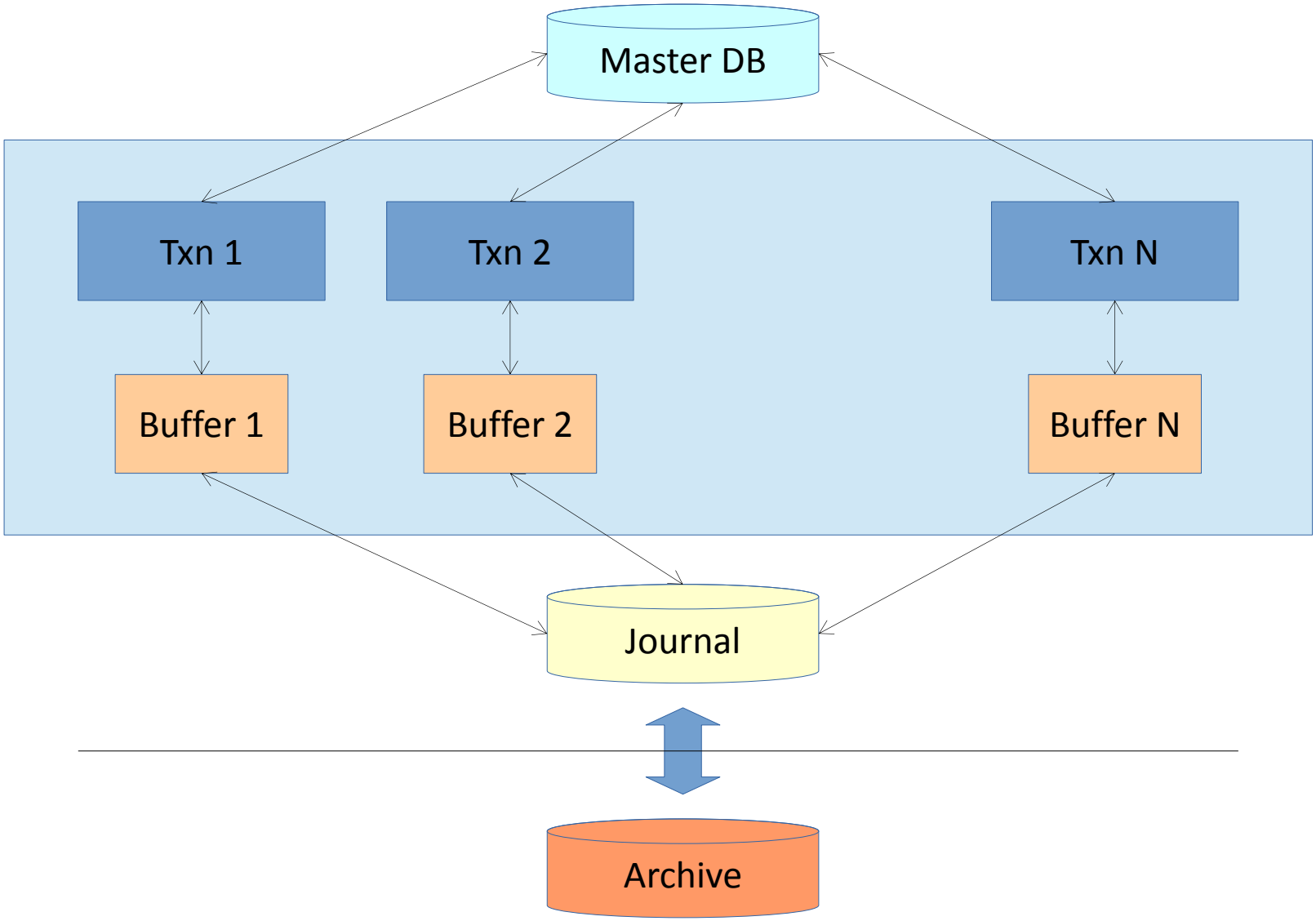
Synchronous replication



Asynchronous replication

- Solution
  - Changes are synchronously journalled on the master side
  - Journal better be placed on a separate disk
  - Journal consists of multiple segments (files) that are rotated
  - Filled segments are transfered to the slave and applied to the replica in background
  - Replica can be recreated online
- Issues
  - Replica always lags behind under load
  - Takeover is not immediate
  - Read-only queries work with historical data

Asynchronous replication



Asynchronous replication