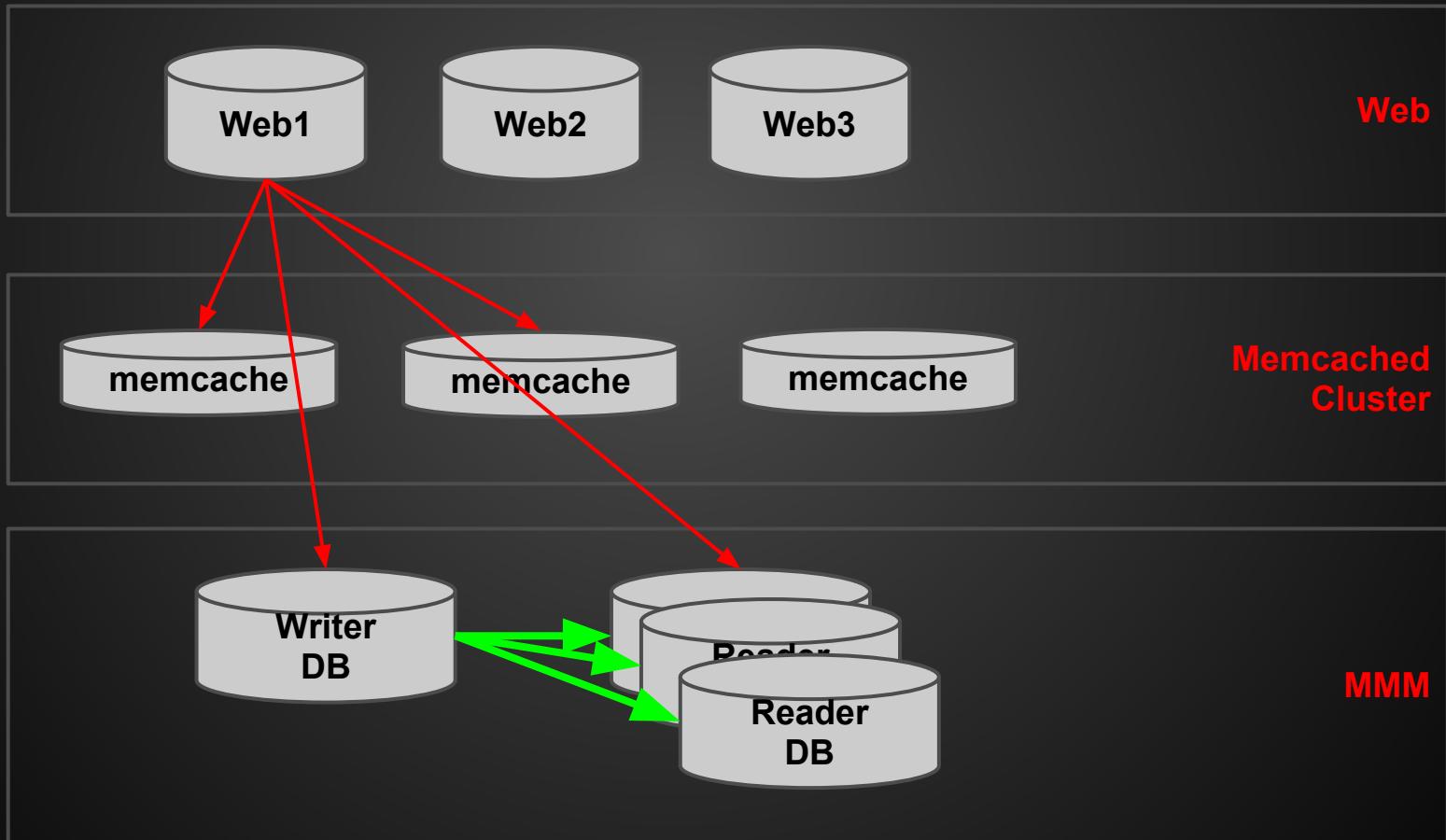


# HA Architecture in DP

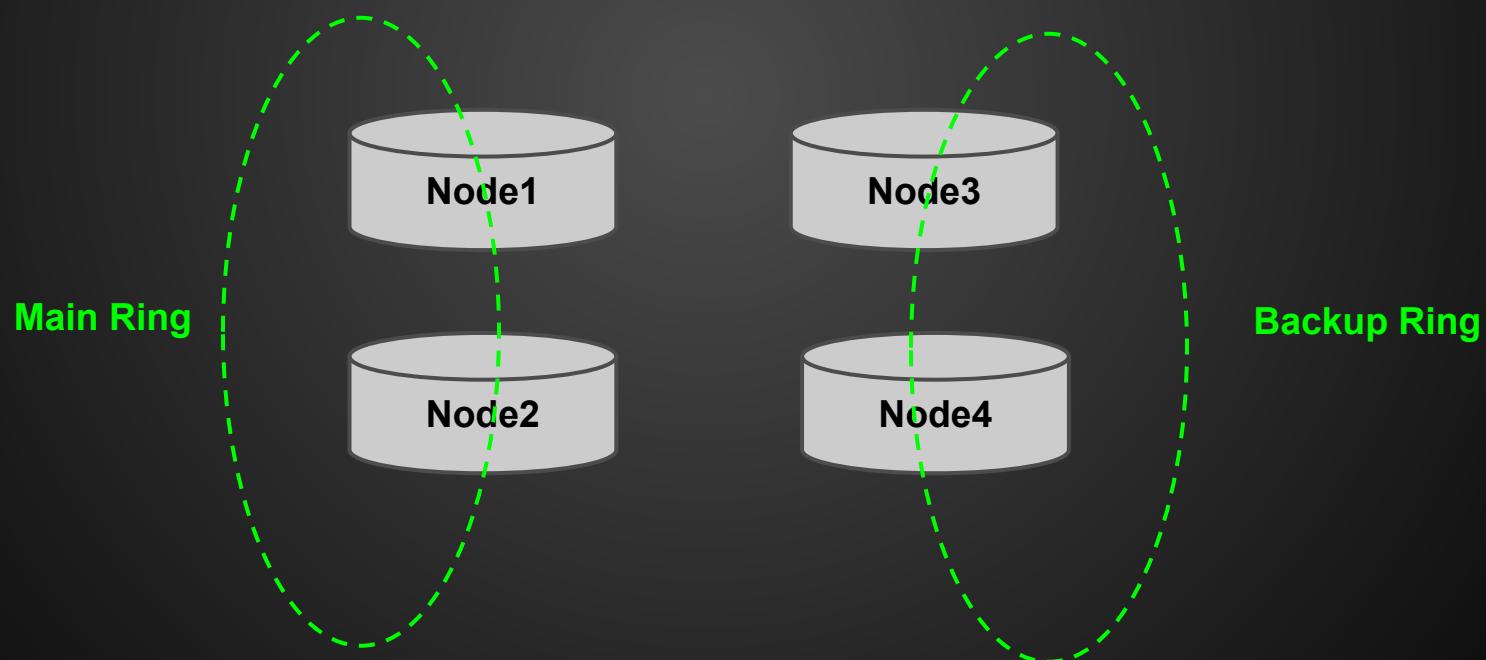
# HA in DP



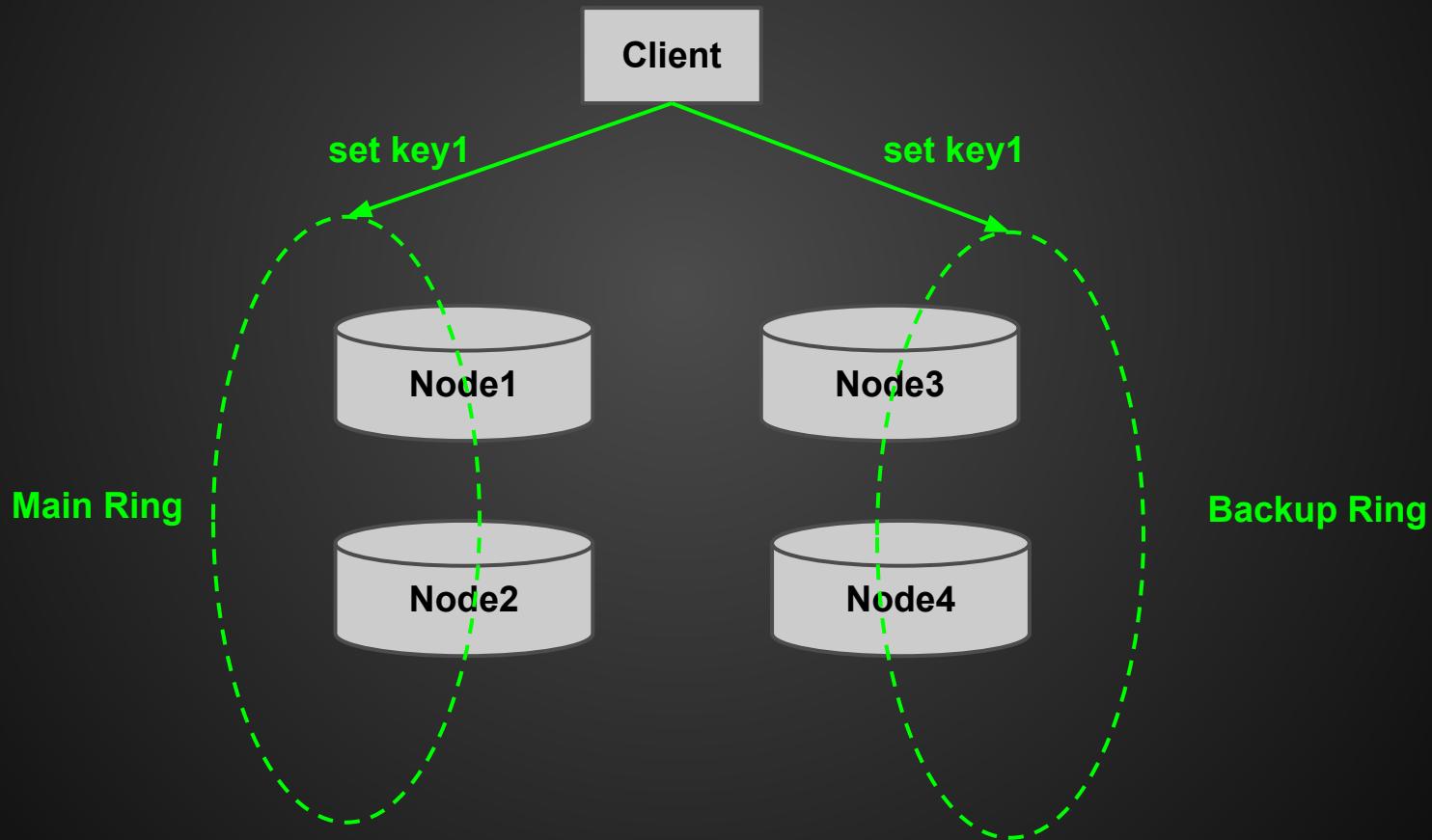
# Memcached

memcached in DP

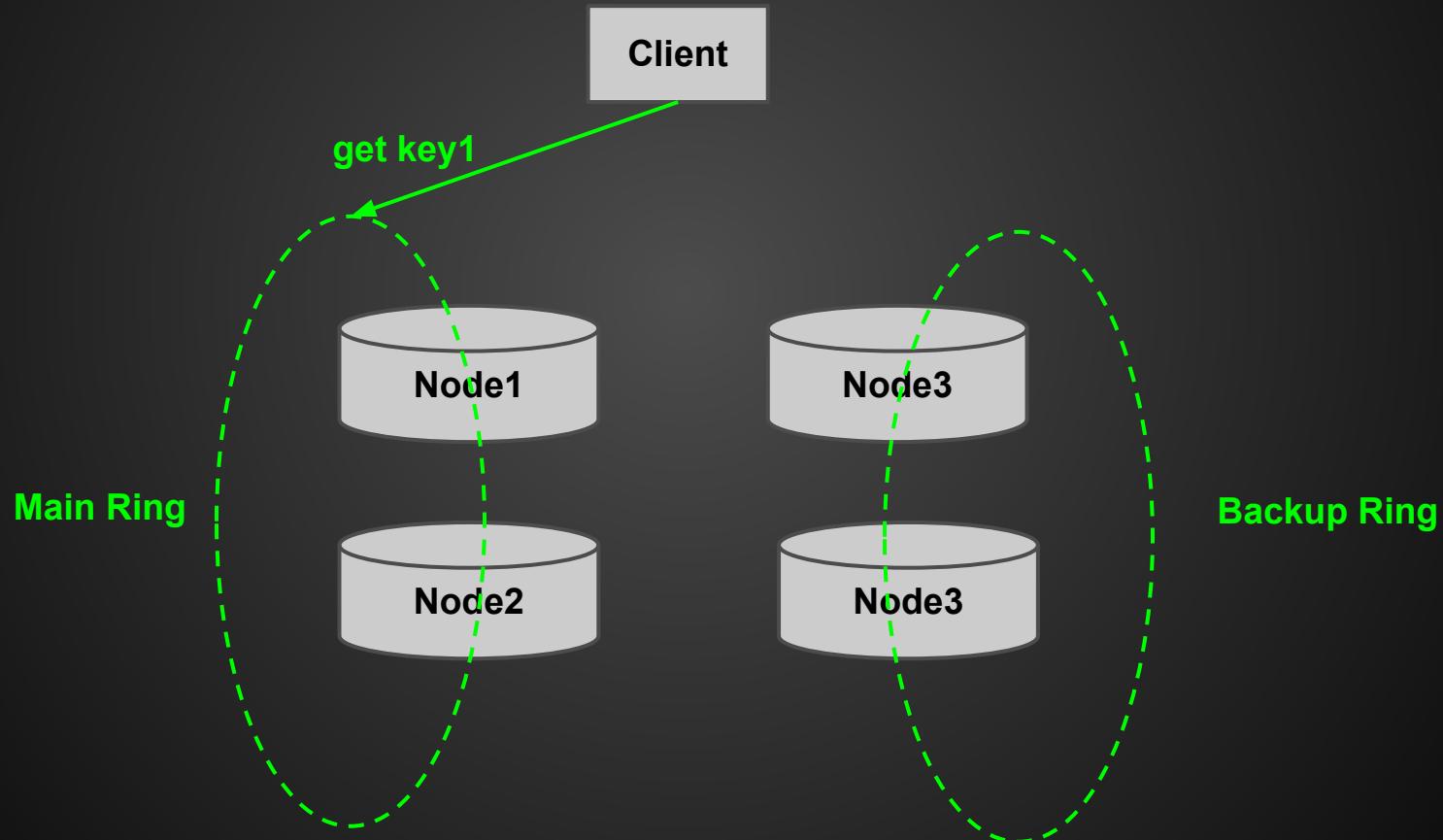
# Memcached in DP



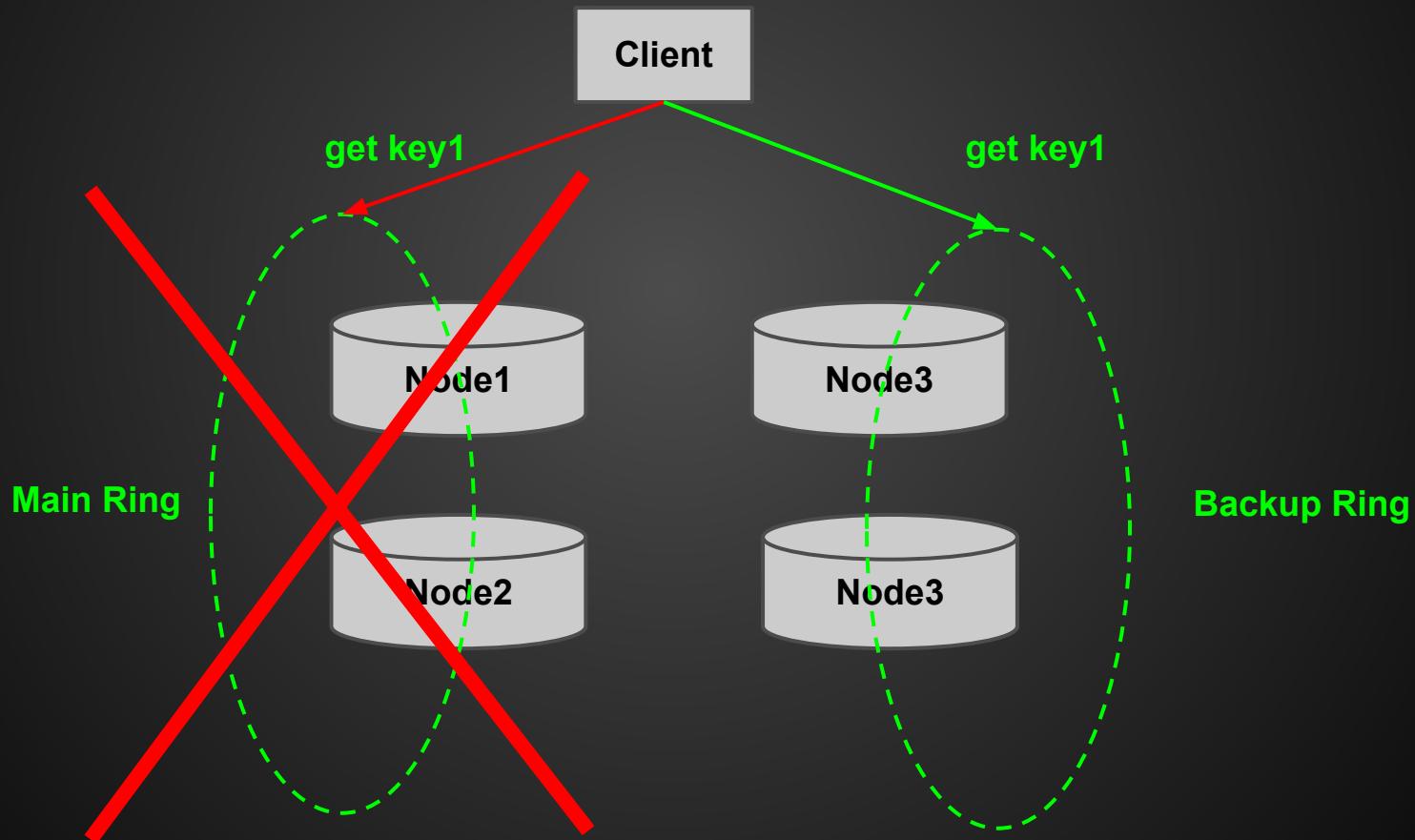
# Memcached in DP



# Memcached in DP



# Memcached in DP



# Memcached

## Problems We Met

# Problem 1

## Cache Miss Storm

# Cache Miss Storm

Happens when :

- Memcached failed (physical)
- Key expired (logical)

# Cache Miss Storm

## Ideal Cache Miss Procedure

1. get memcached miss
2. query MySQL
3. set value into memcached

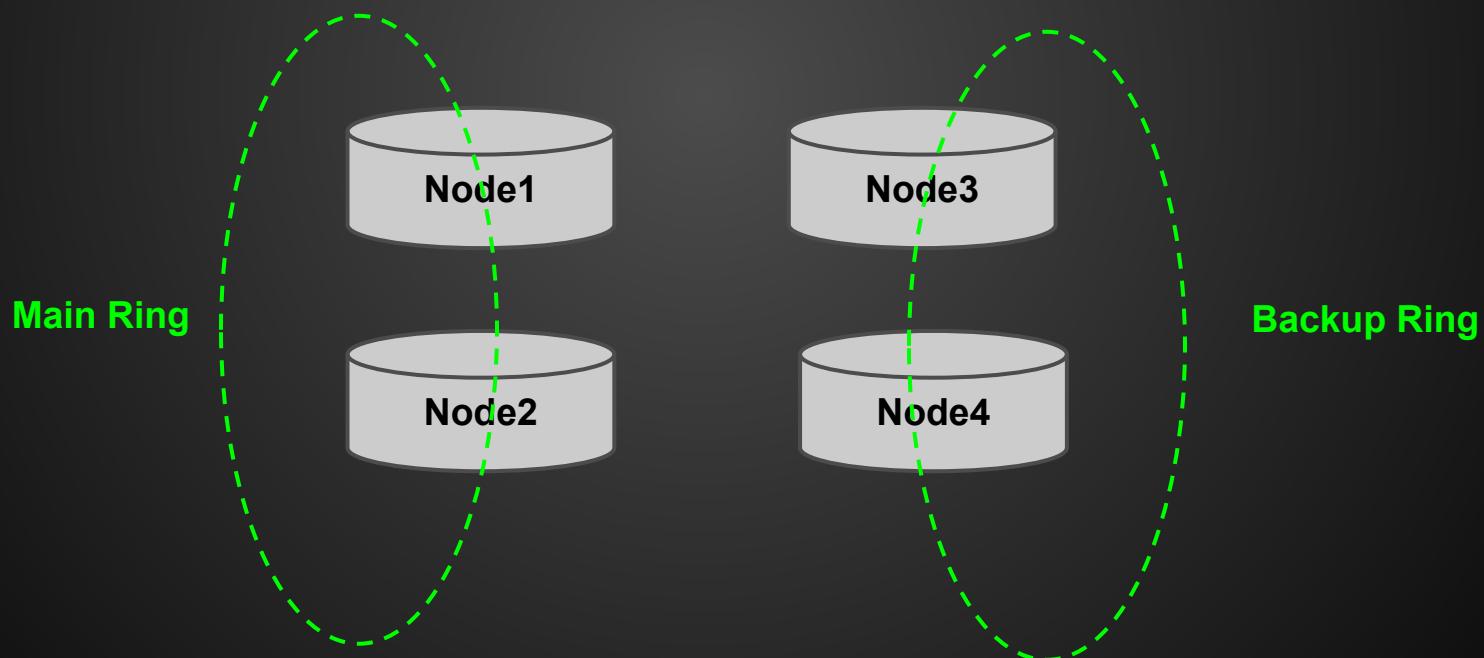
# Cache Miss Storm

In Fact !

1. get memcached miss
2. massive concurrent query on MySQL  
(timeout)
3. nothing can be set into memcached
4. cache miss forever....

# Cache Miss Storm -- Fix1

Fix for physical cache miss -- Double Write



# Cache Miss Storm -- Fix2

Fix for logical cache miss -- Hot Key

0. *get && set value into web local cache*
1. *get memcached miss*
2. try to *add a lock key into memcache*
  - a. if (success) query MySQL & *set memcache*
  - b. if (failed) return local cache

\* Only one web can query MySQL for missed key at the same time.

# Problem 2

Multi Get Hole

# MultiGet Hole

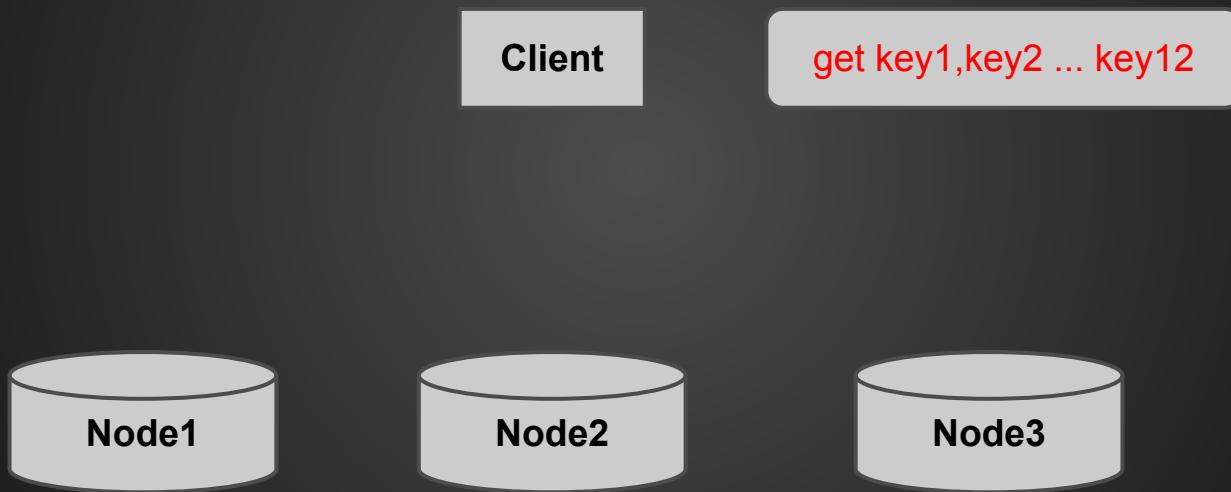
MultiGet / Gets: get command with multiple keys

**Purpose:** Omit the multiple network round-trips,

**Problem:** The *gets* command will be slower when we add more nodes into the cluster.

<http://highscalability.com/blog/2009/10/26/facebook-s-memcached-multiget-hole-more-machines-more-capacit.html>

# MultiGet Hole



# MultiGet Hole

Client

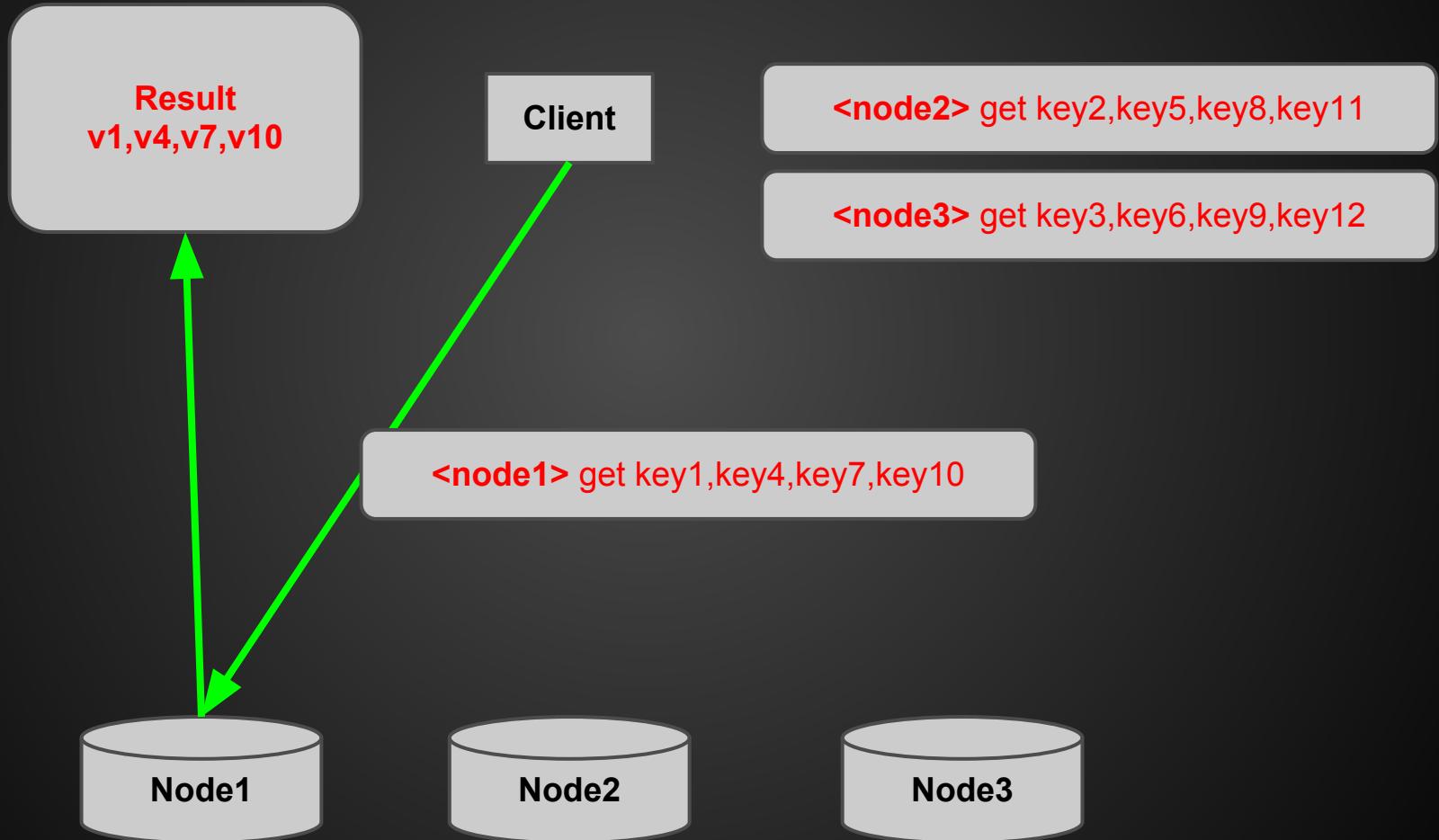
<node1> get key1,key4,key7,key10

<node2> get key2,key5,key8,key11

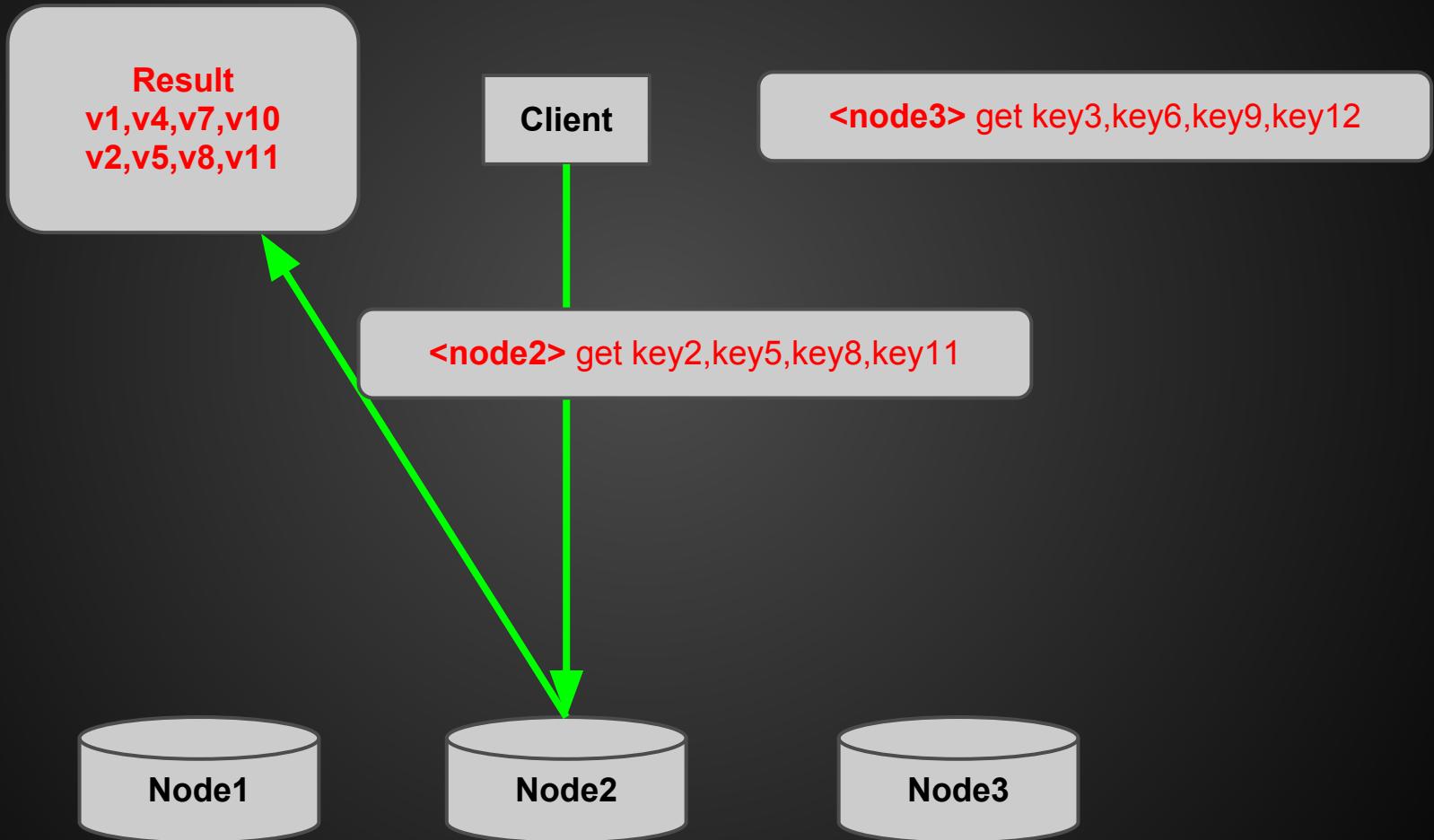
<node3> get key3,key6,key7,key12



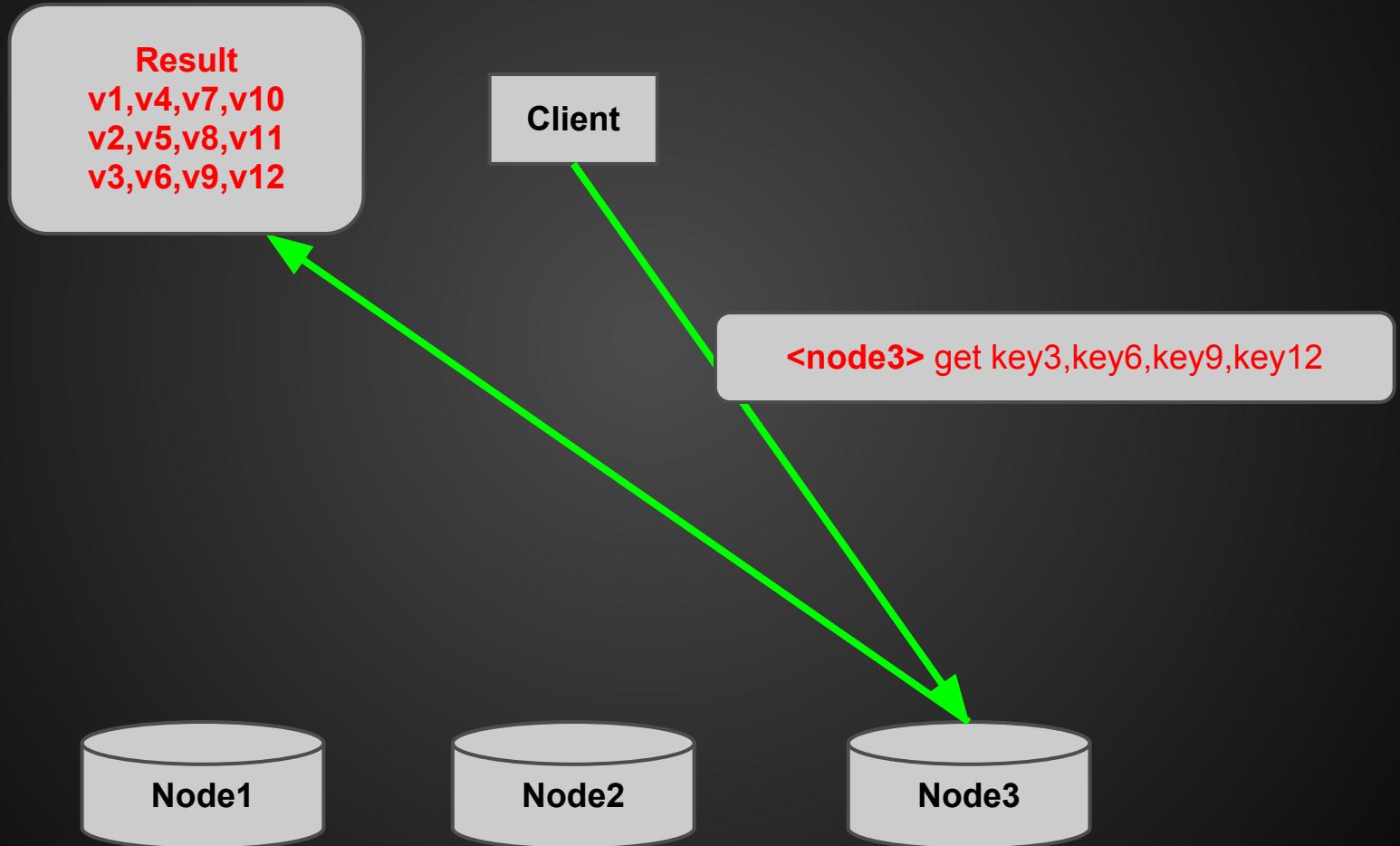
# MultiGet Hole



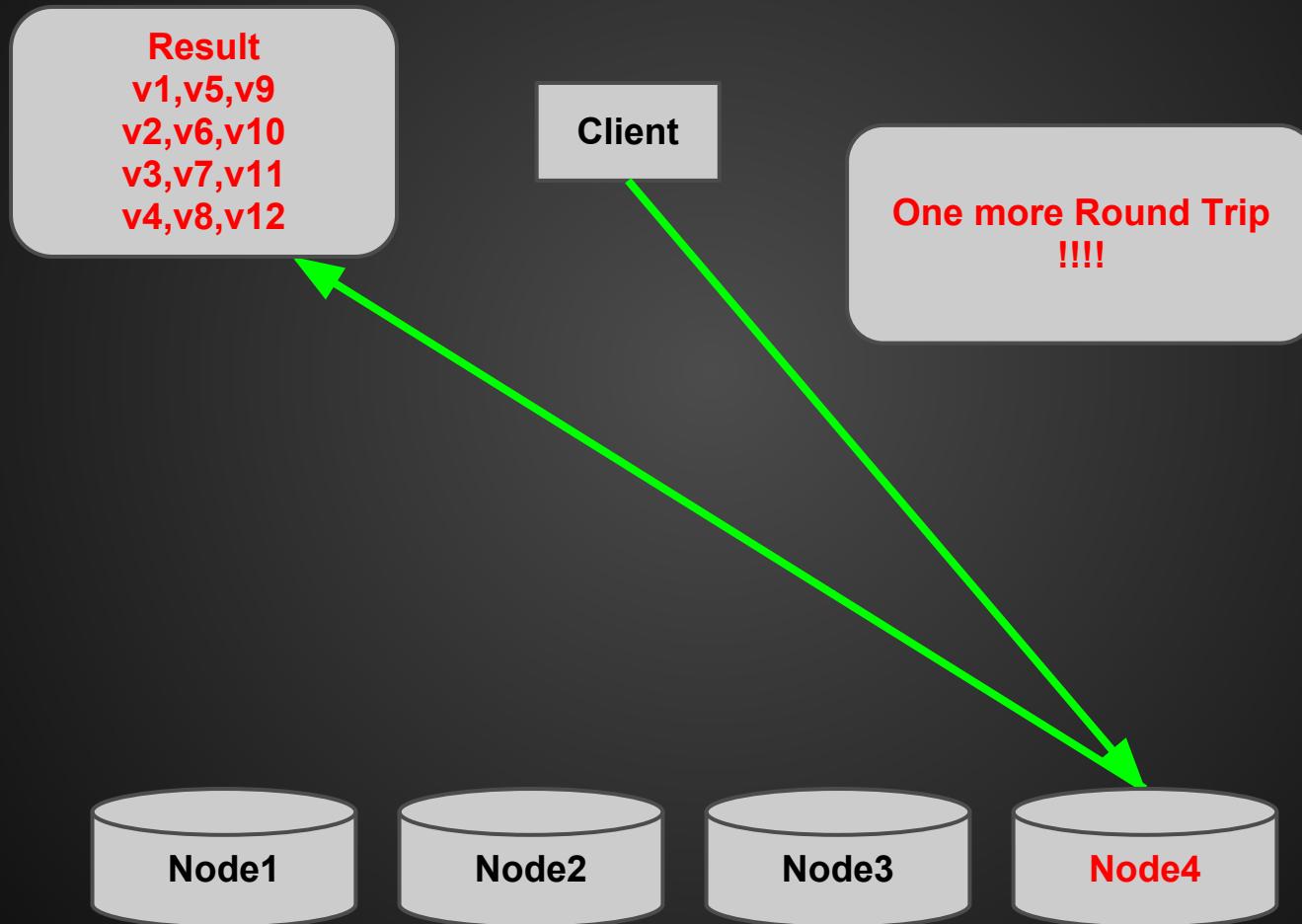
# MultiGet Hole



# MultiGet Hole



# MultiGet Hole



# MultiGet Hole -- Fix

**Before:** client use key name to

- decide which host to store
- store value

**Now:** client use master key name to

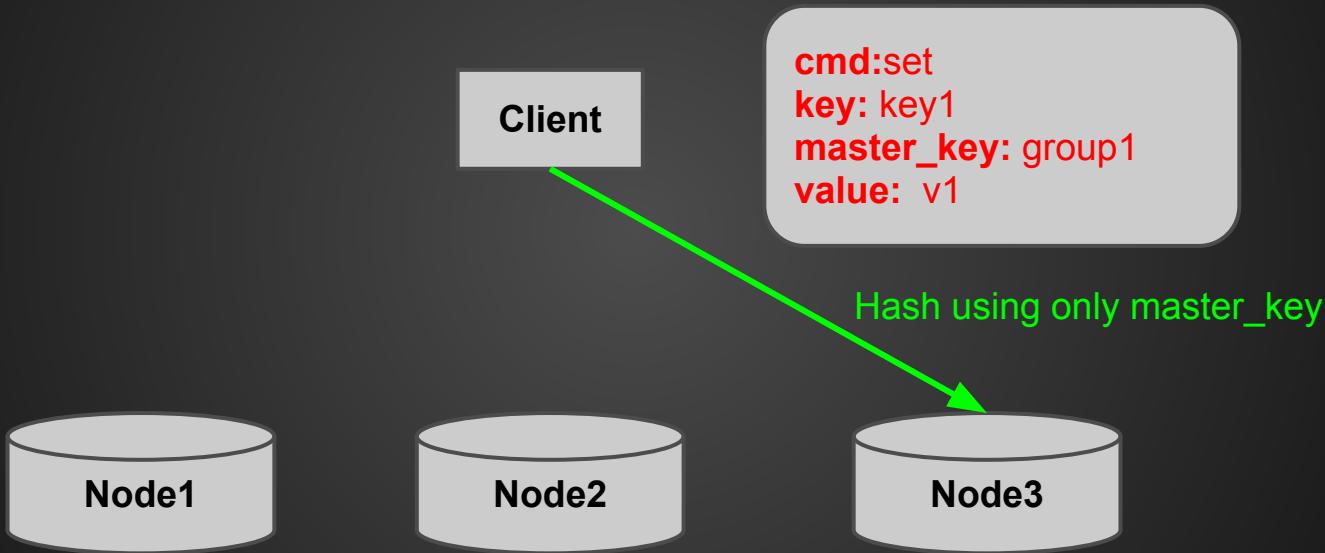
- decide which host to store

use key name to

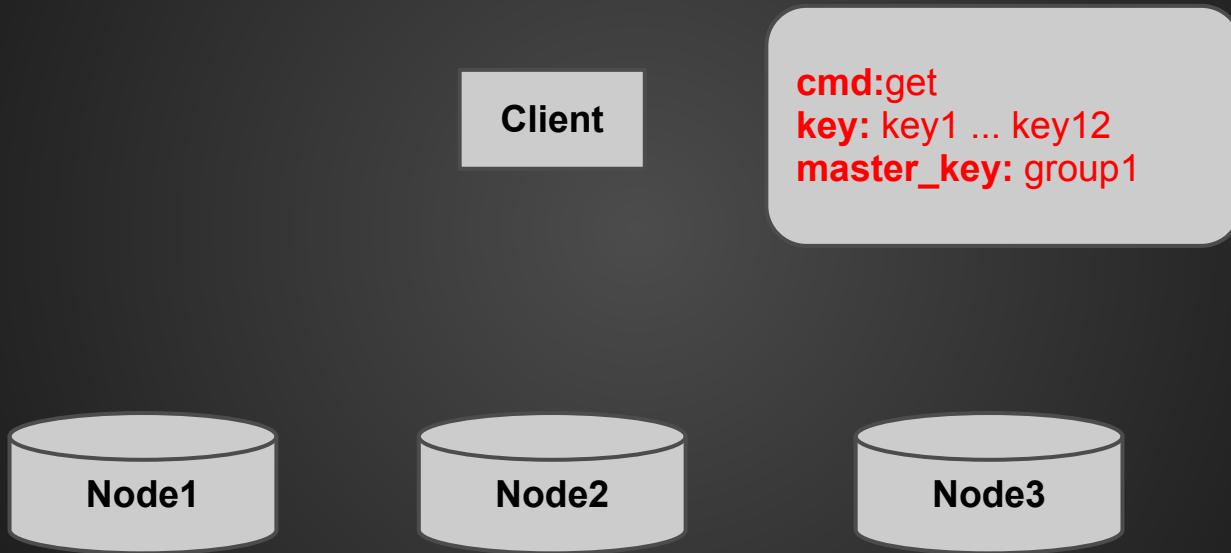
- store value

\* Put the keys of the same group into only one node

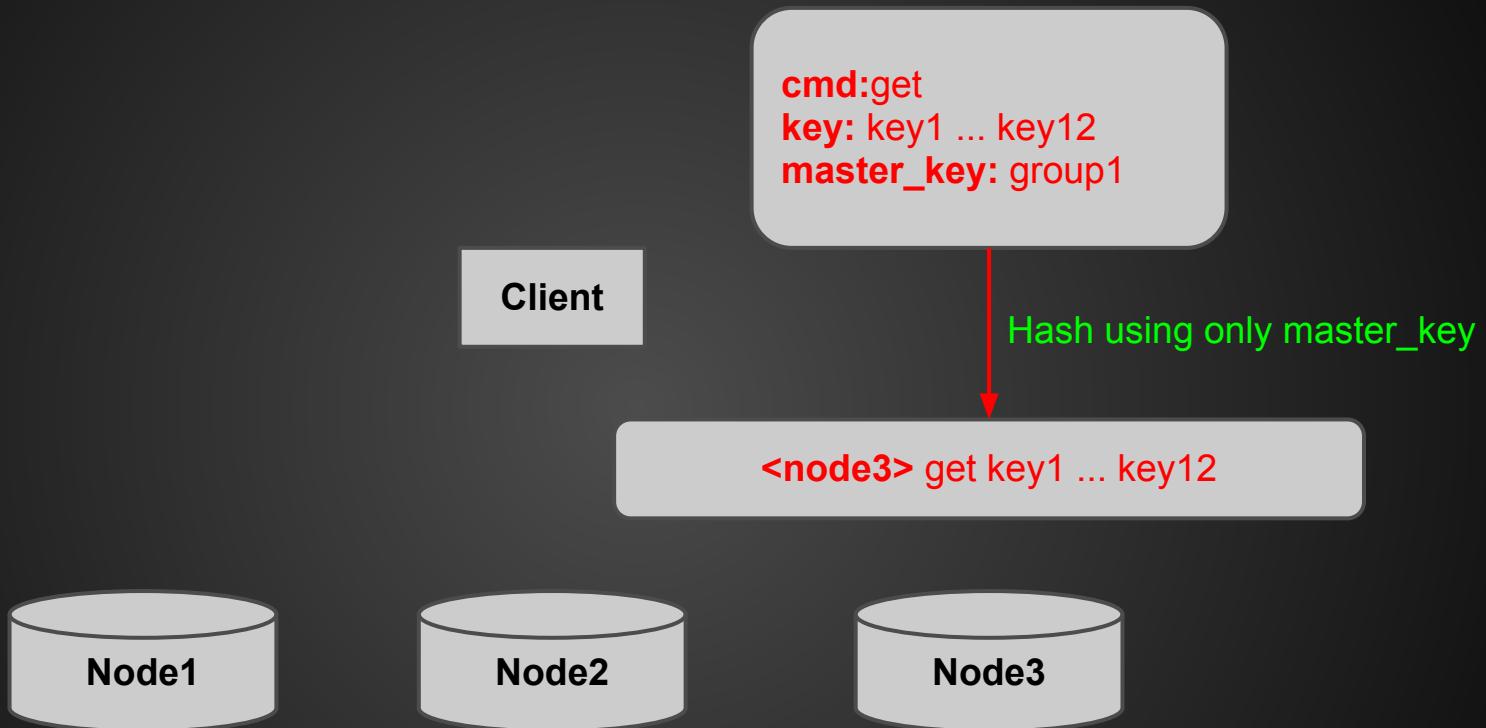
# MultiGet Hole -- Fix



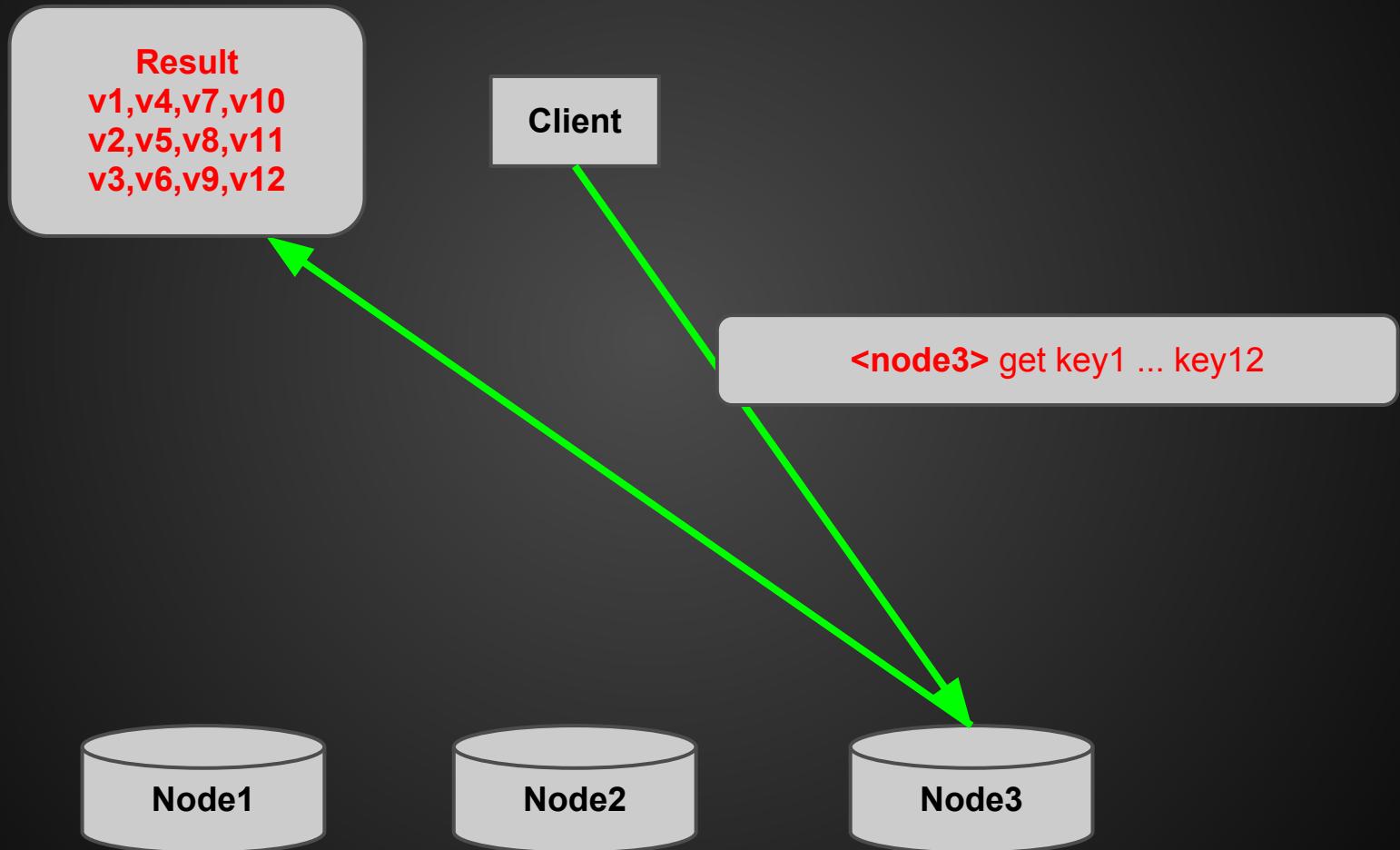
# MultiGet Hole -- Fix



# MultiGet Hole -- Fix



# MultiGet Hole -- Fix



# Problem 3

## Virtual Packet Loss

# VPL

VPL: virtual packet loss

No actual packet loss, but vm response time exceeds the retransmission timeout

275149 34.380647	10.1.6.84	10.1.7.194	MEMCACHE	VALUE TGNavitTagByCategoryServer.c160WEB0_17 4 1607
275151 34.380842	10.1.7.194	10.1.6.84	TCP	34668 > memcache [ACK] Seq=950199 Ack=6167562 Win=5884
275885 34.451498	10.1.7.194	10.1.6.84	MEMCACHE	[TCP Previous segment lost] get TGNavitTagByCategoryServer.c160WEB0_17 4 1607
275886 34.451506	10.1.6.84	10.1.7.194	TCP	[TCP Dup ACK 275149#1] memcache > 34668 [ACK] Seq=6167562 Ack=950199 Win=5884
276253 34.495090	10.1.7.194	10.1.6.84	MEMCACHE	get TGDealGroupIdsByShopGroupAndCity.32736671_0
276254 34.495096	10.1.6.84	10.1.7.194	TCP	[TCP Dup ACK 275149#2] memcache > 34668 [ACK] Seq=6167562 Ack=950199 Win=5884
276283 34.504971	10.1.7.194	10.1.6.84	MEMCACHE	get TGNavitTagByCategoryServer.c1MTUAN0_17
276284 34.504976	10.1.6.84	10.1.7.194	TCP	[TCP Dup ACK 275149#3] memcache > 34668 [ACK] Seq=6167562 Ack=950199 Win=5884
276285 34.505215	10.1.7.194	10.1.6.84	MEMCACHE	[TCP Fast Retransmission] get TGNavitTagByCategoryServer.c1MTUAN0_17
276286 34.505223	10.1.6.84	10.1.7.194	TCP	memcache > 34668 [ACK] Seq=6167562 Ack=950377 Win=15784

Two network-bounded virtual machine put together result in huge get timeout.

# VPL

A normal retransmission consume 50ms, which exceeds our Memcached timeout.

timeout == no result == cache miss

Result: another kind of cache miss storm

# VPL -- Fix

- Split Network-Bound biz on different real machine.
- Maybe UDP?
- Maybe fast retransmission?

# Short Questions?

# MMM

# What is MMM

- Perl
- Message between Monitor & Agent
- Auto Failover for M/S

but MMM is not:

- SQL router
- Load Balancer

# What's wrong with MMM

MMM is

- 1) fundamentally broken and unsuitable for use as a HA tool
- 2) absolutely cannot be fixed.

Baron Schwartz: <http://www.xaprb.com/blog/2011/05/04/whats-wrong-with-mmm/>

# MMM Internals

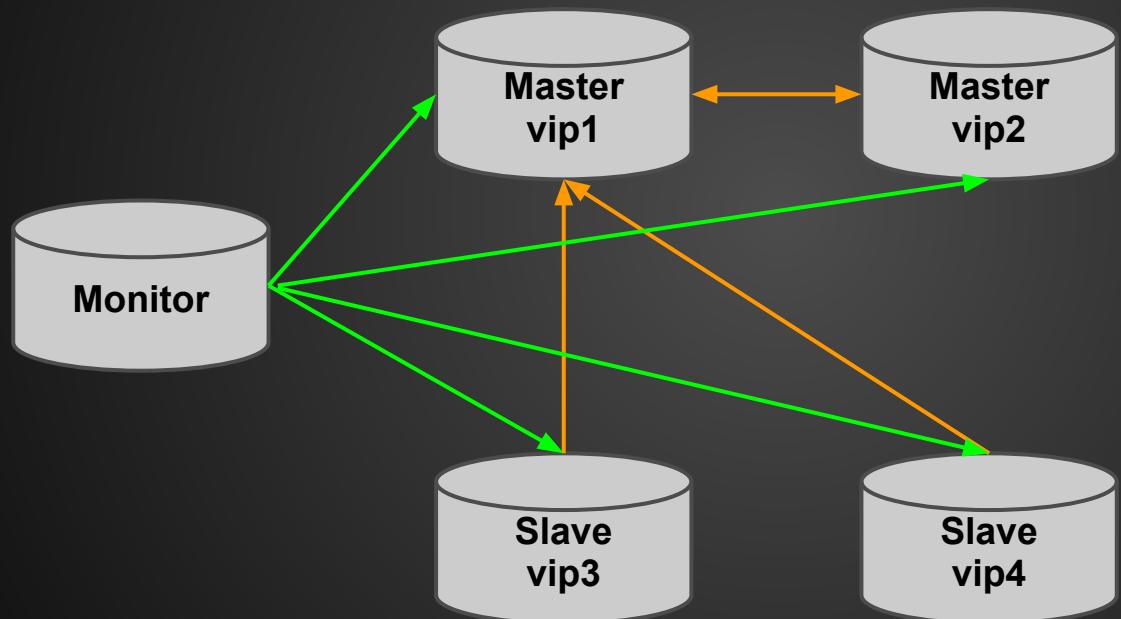
## Monitor

```
while(){  
    process_check_results  
    check_host_states  
    process_commands  
    distribute_role  
    send_status_to_agent  
    s  
}
```

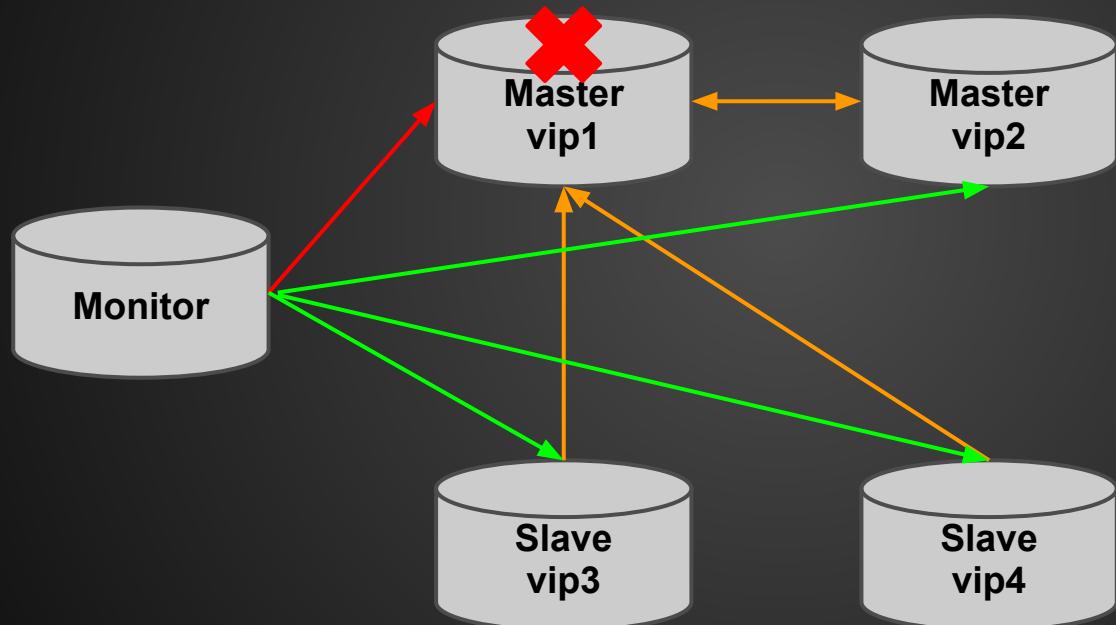
## Agent

```
while( read socket){  
    handle_command  
}
```

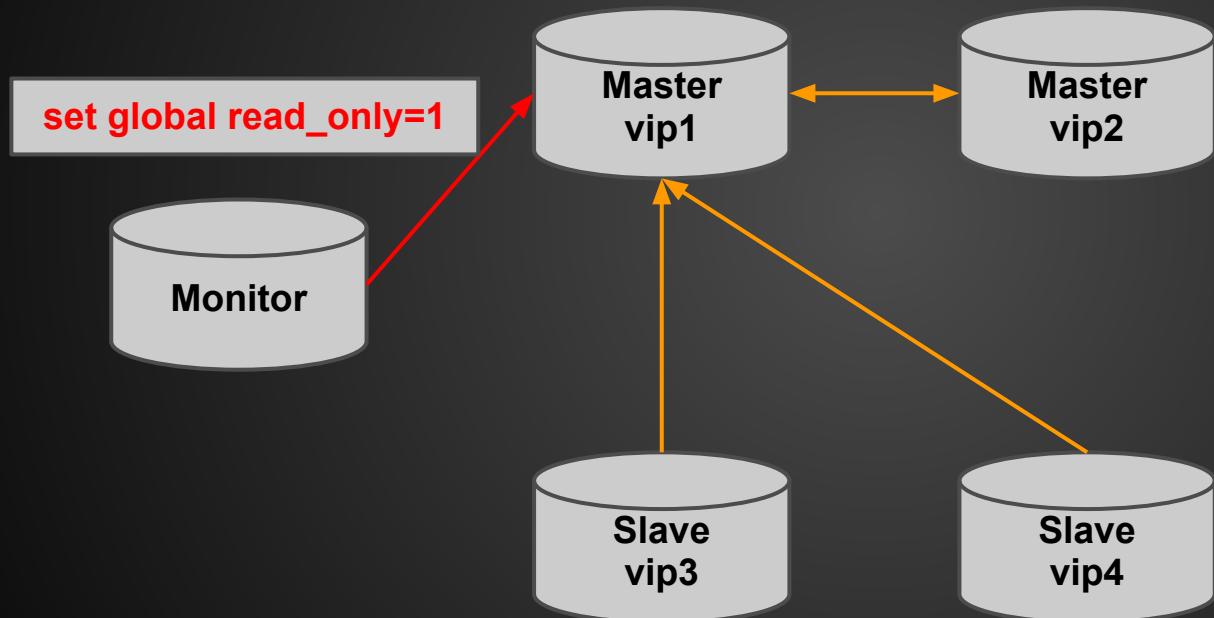
# MMM architecture



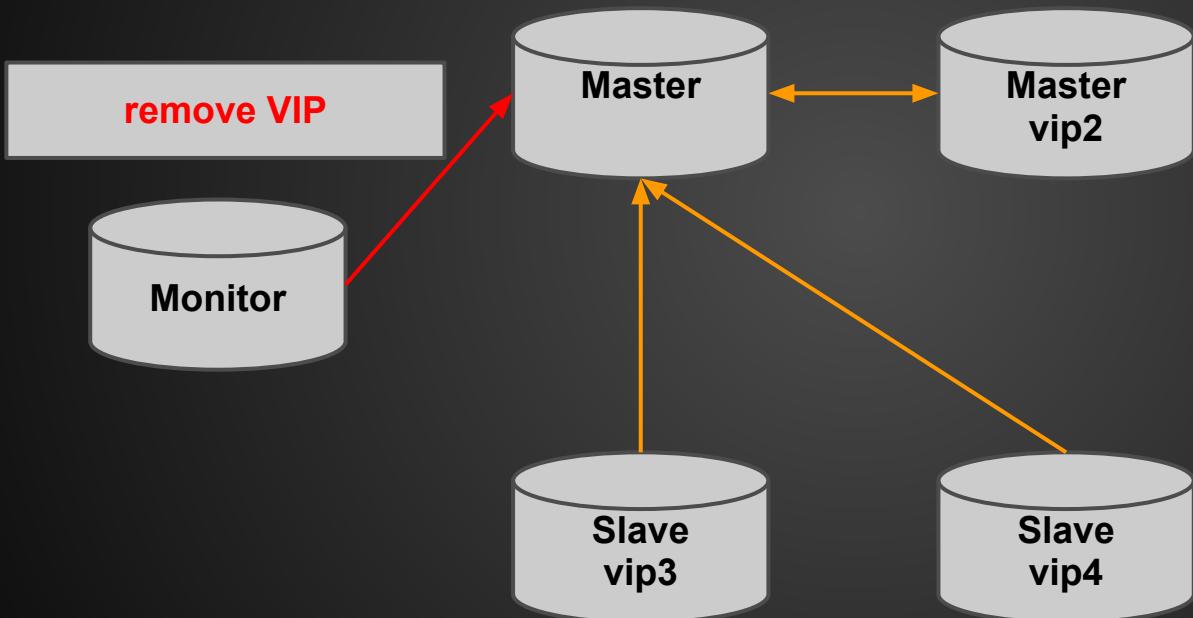
# How MMM Do Failover



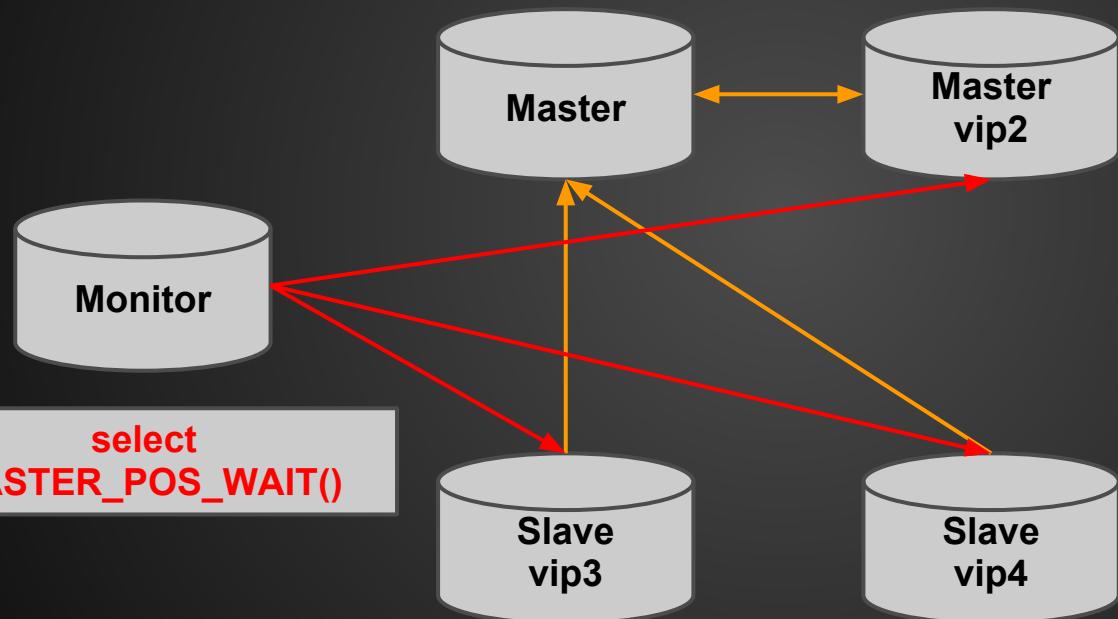
# How MMM Do Failover



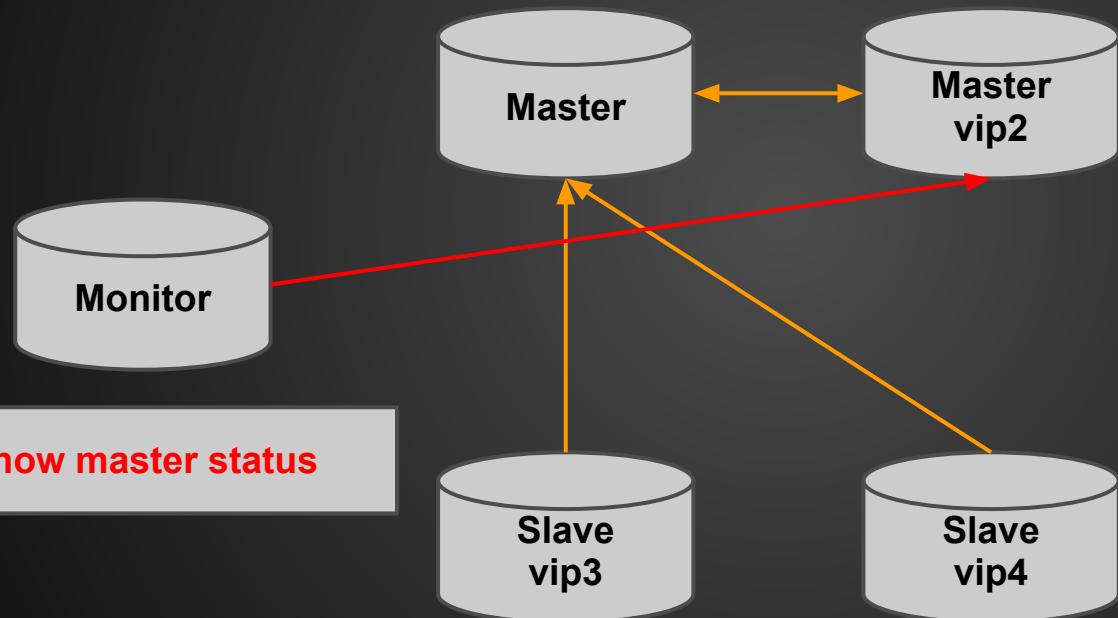
# How MMM Do Failover



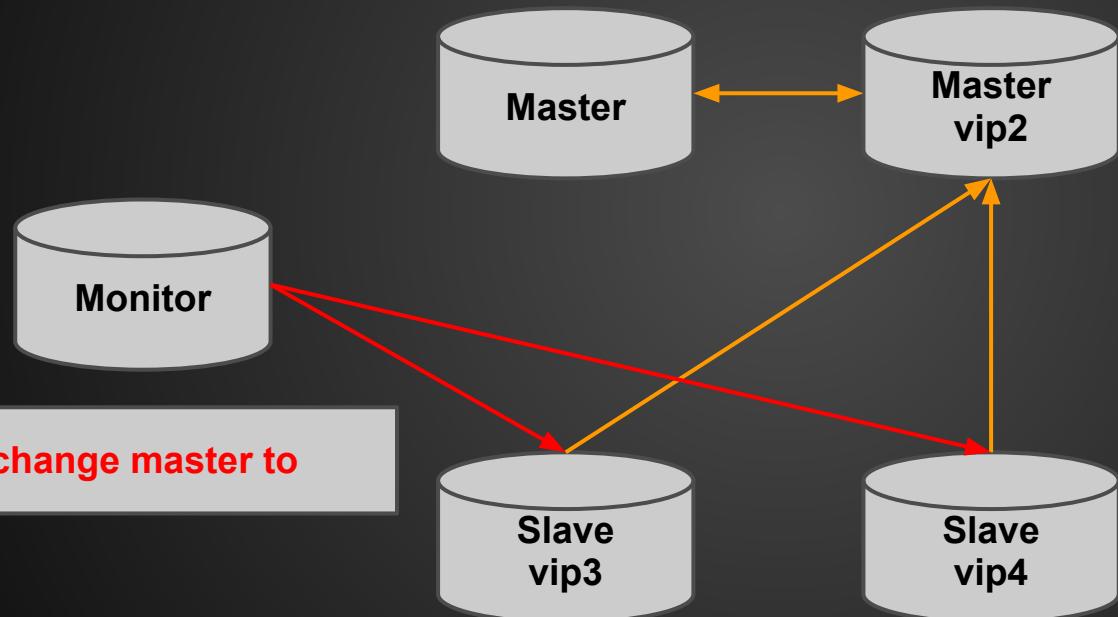
# How MMM Do Failover



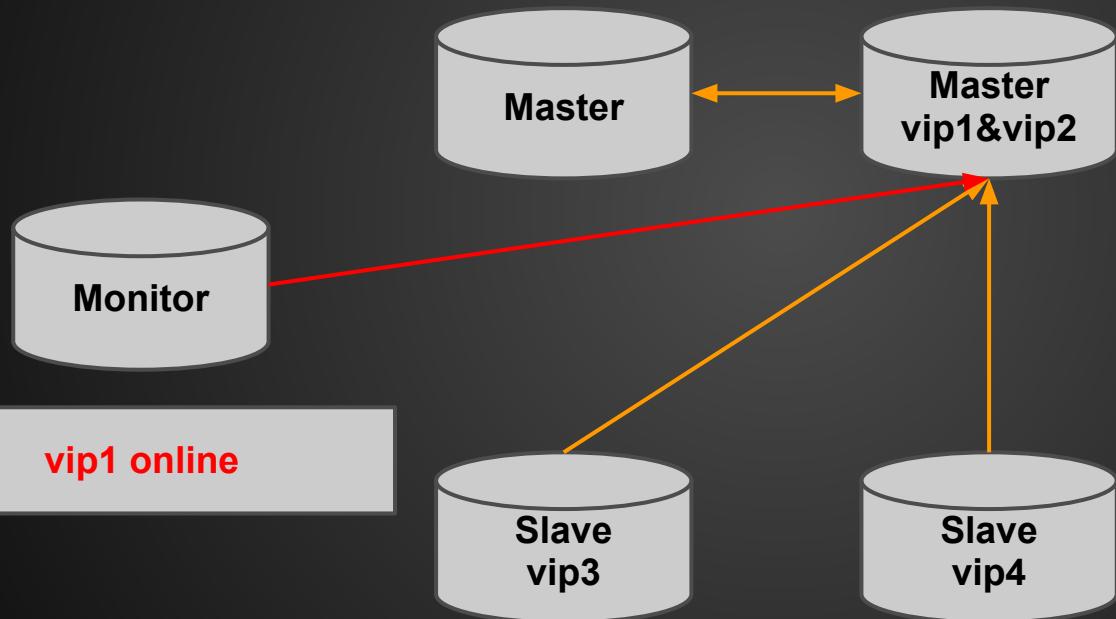
# How MMM Do Failover



# How MMM Do Failover



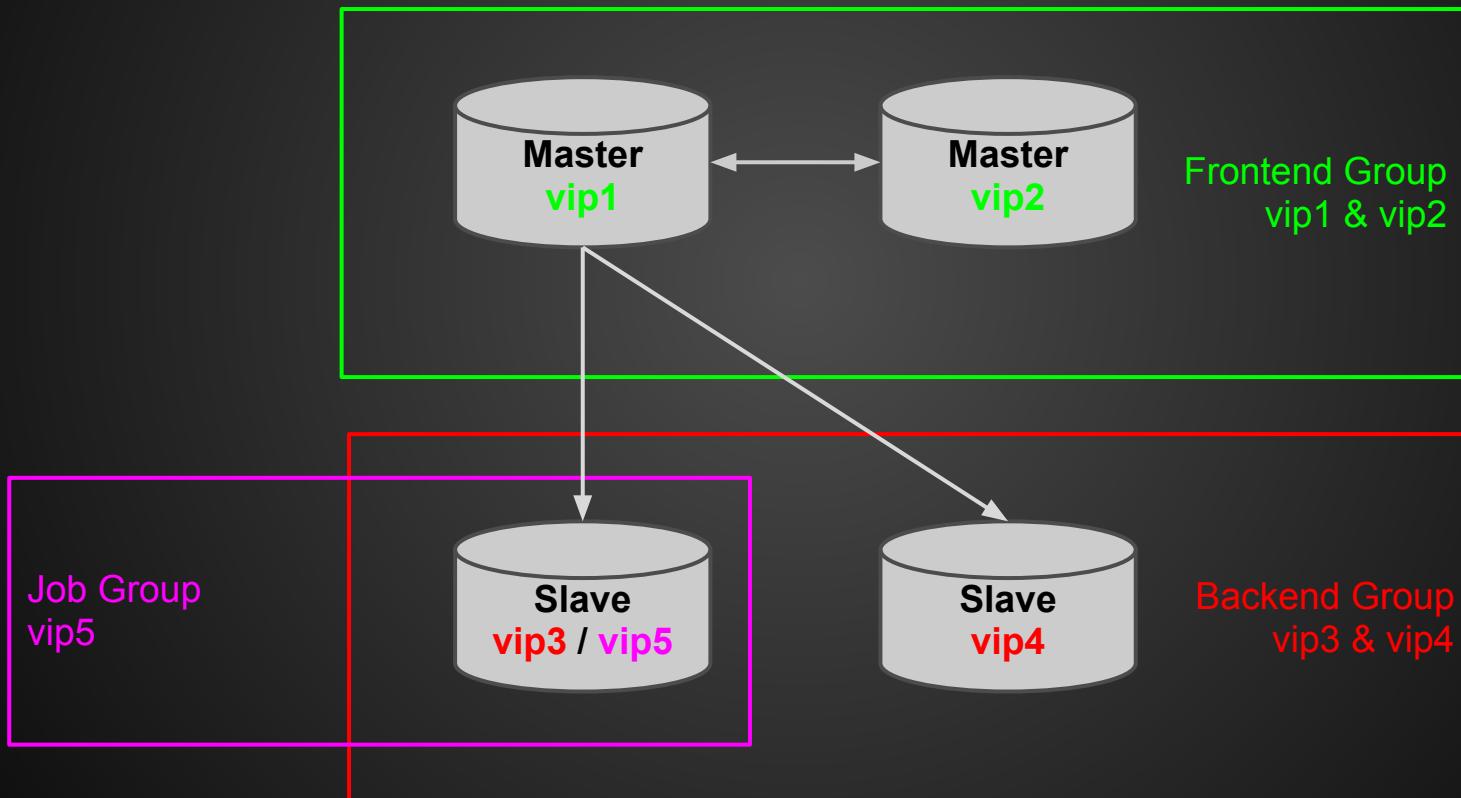
# How MMM Do Failover



# MMM

## MMM in DP

# MMM in DP



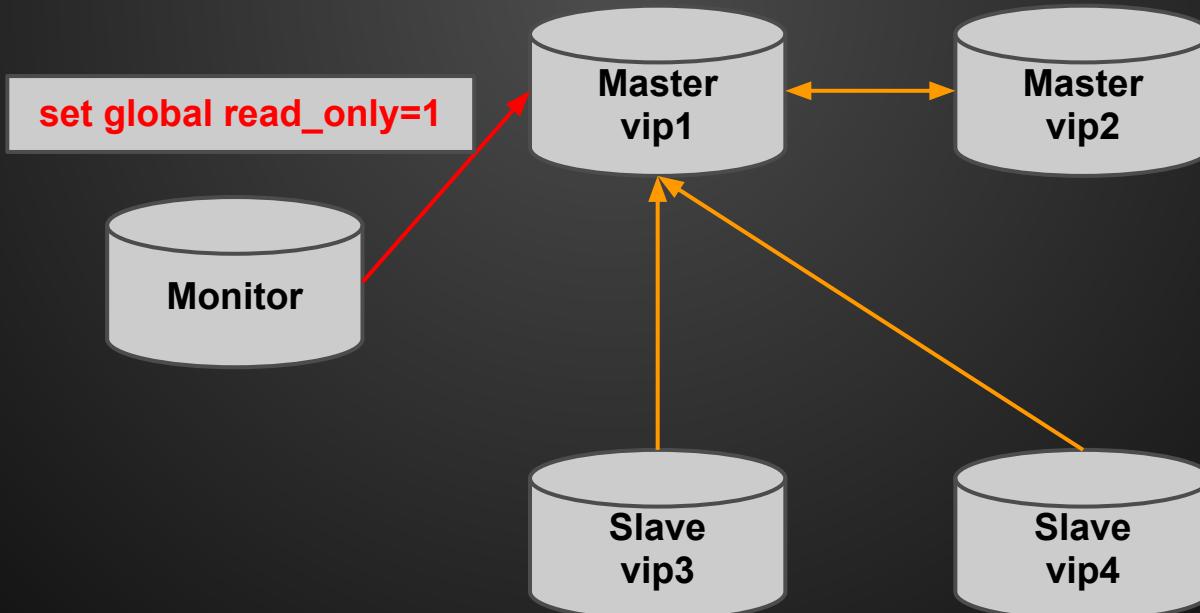
# MMM

## Problems in MMM

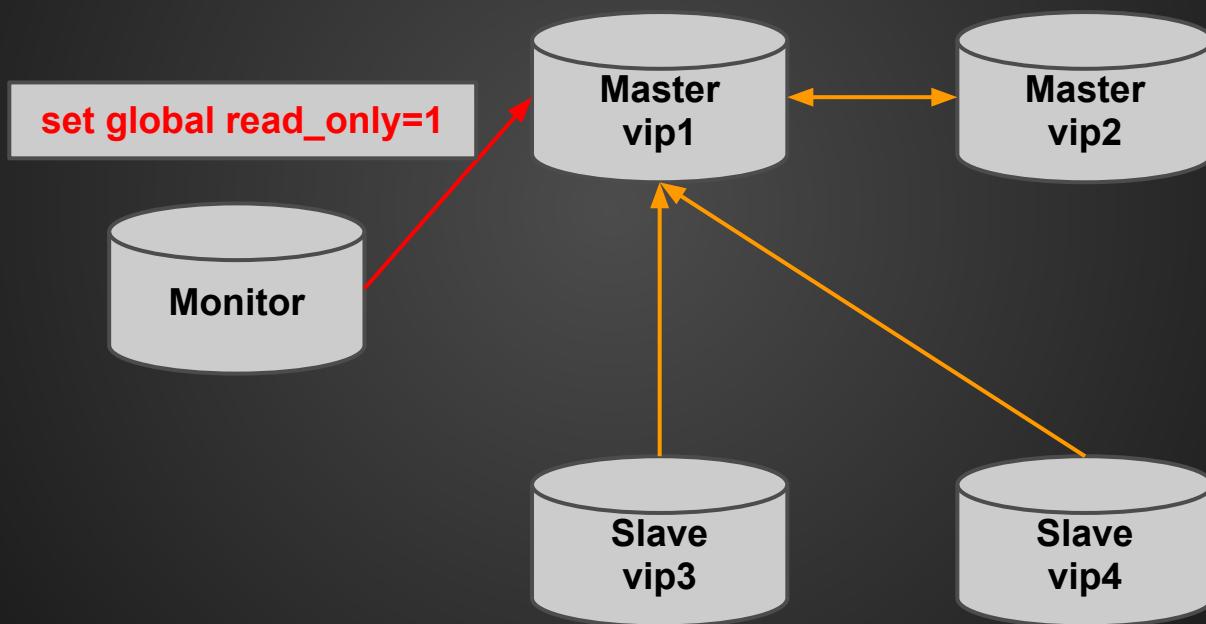
# MMM Problem 1

set read\_only is difficult on busy server

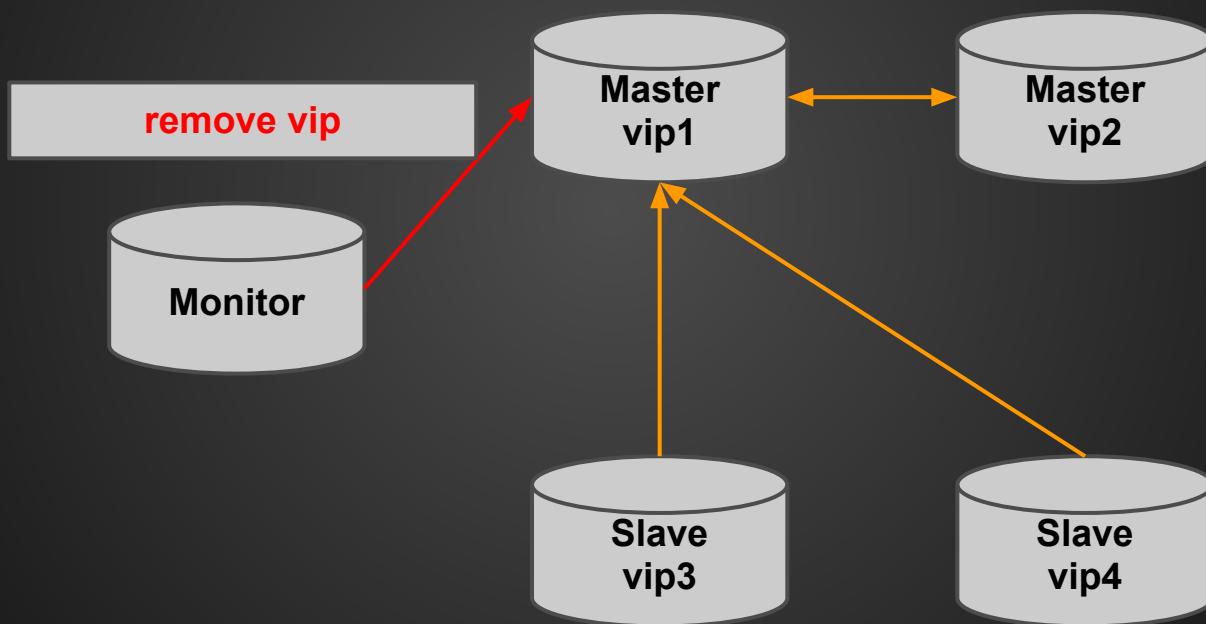
set read\_only will be blocked by long running SQL



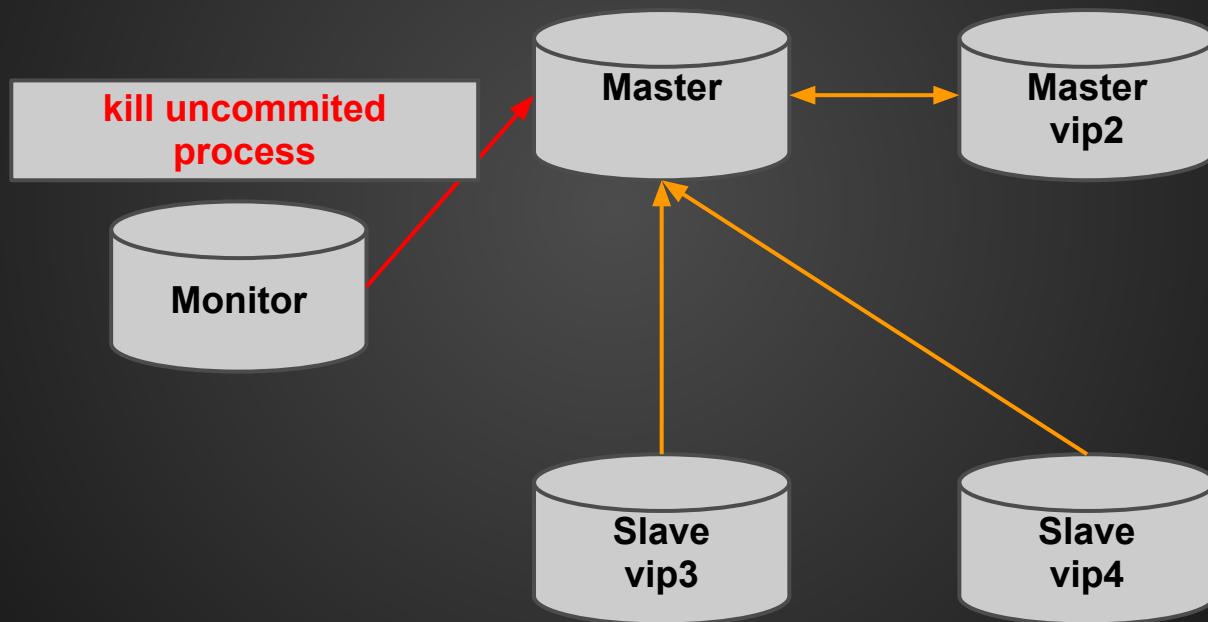
# MMM Problem 1



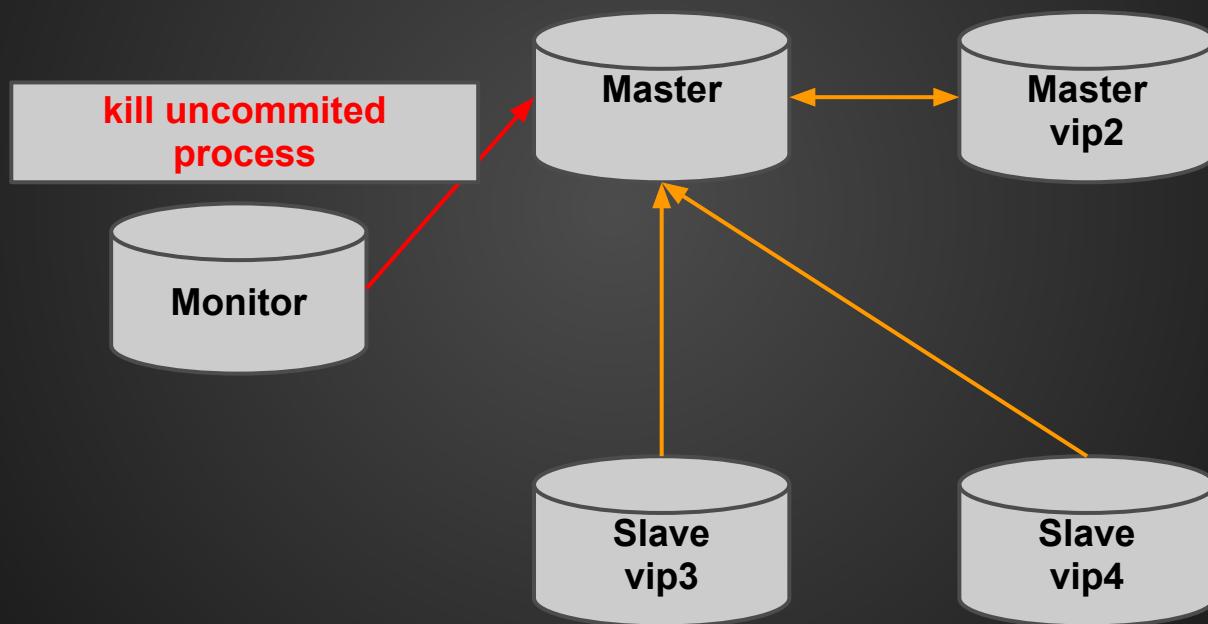
# MMM Problem 1 -- Fix



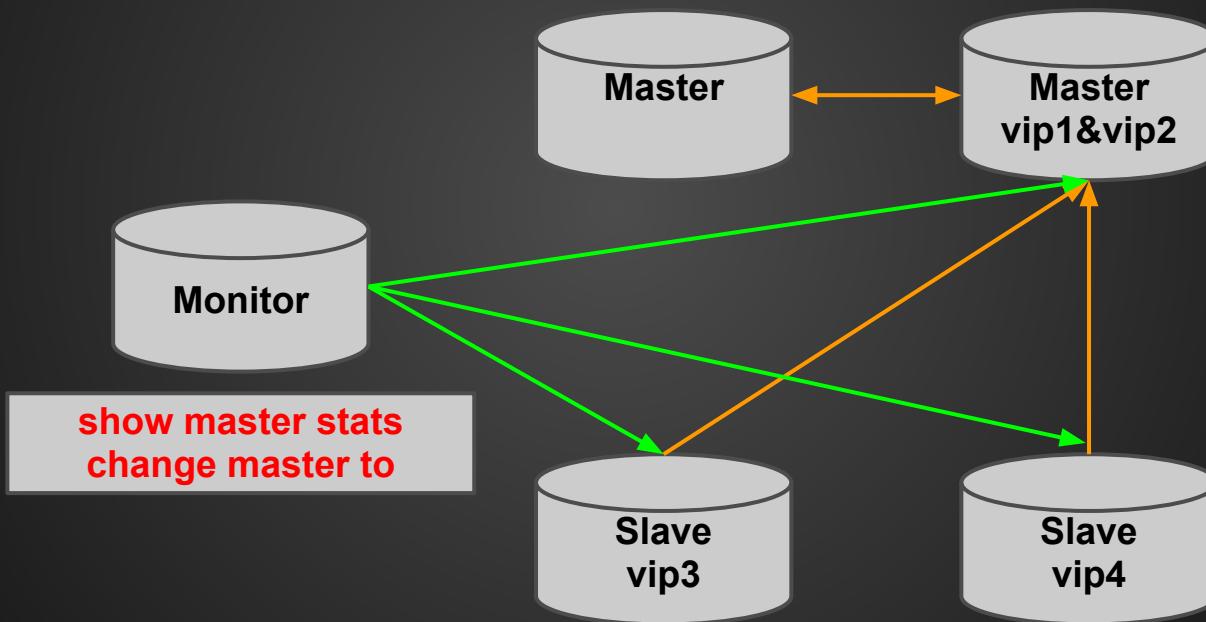
# MMM Problem 1 -- Fix



# MMM Problem 1 -- Fix

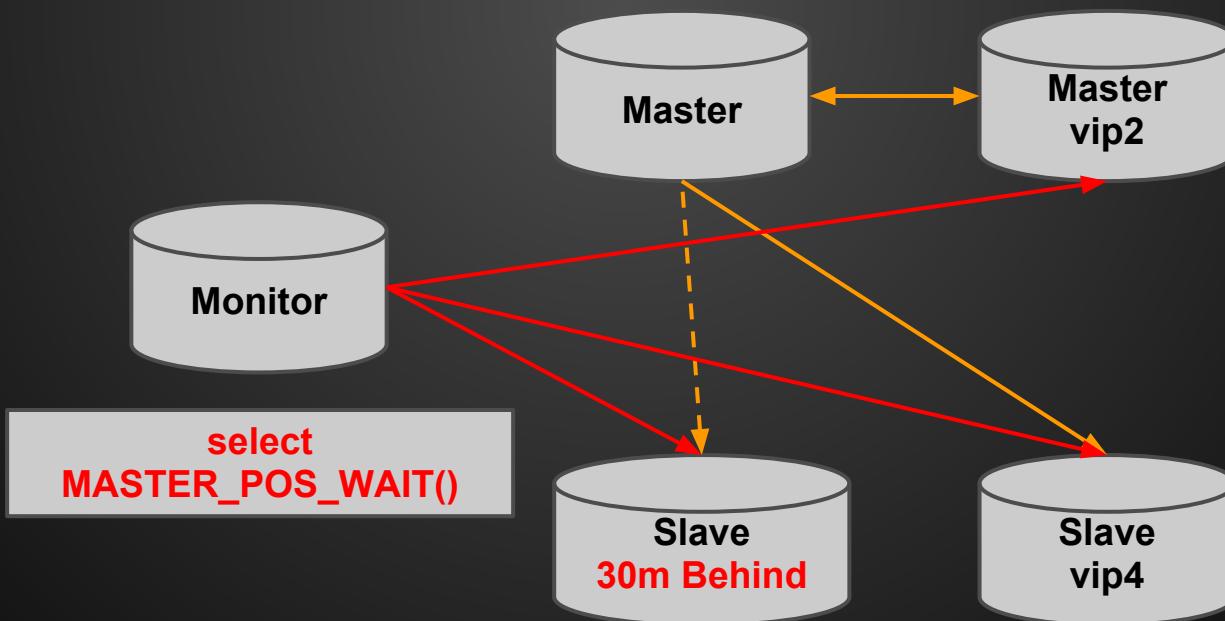


# MMM Problem 1 -- Fix



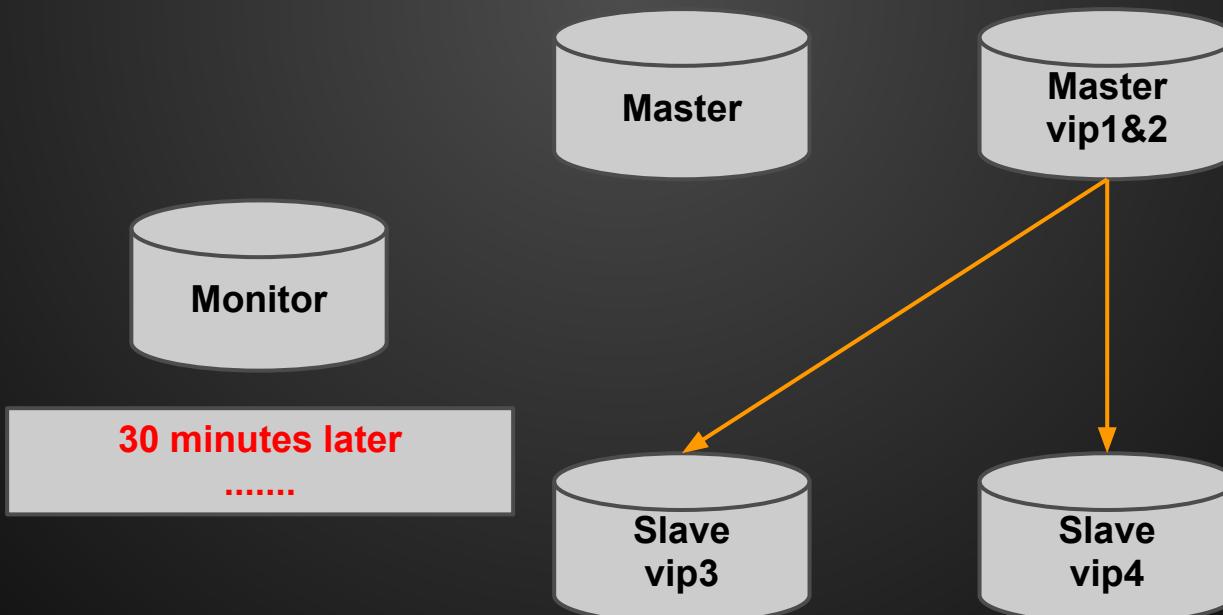
# MMM Problem 2

Writer VIP cannot be accessed when slave is far behind master



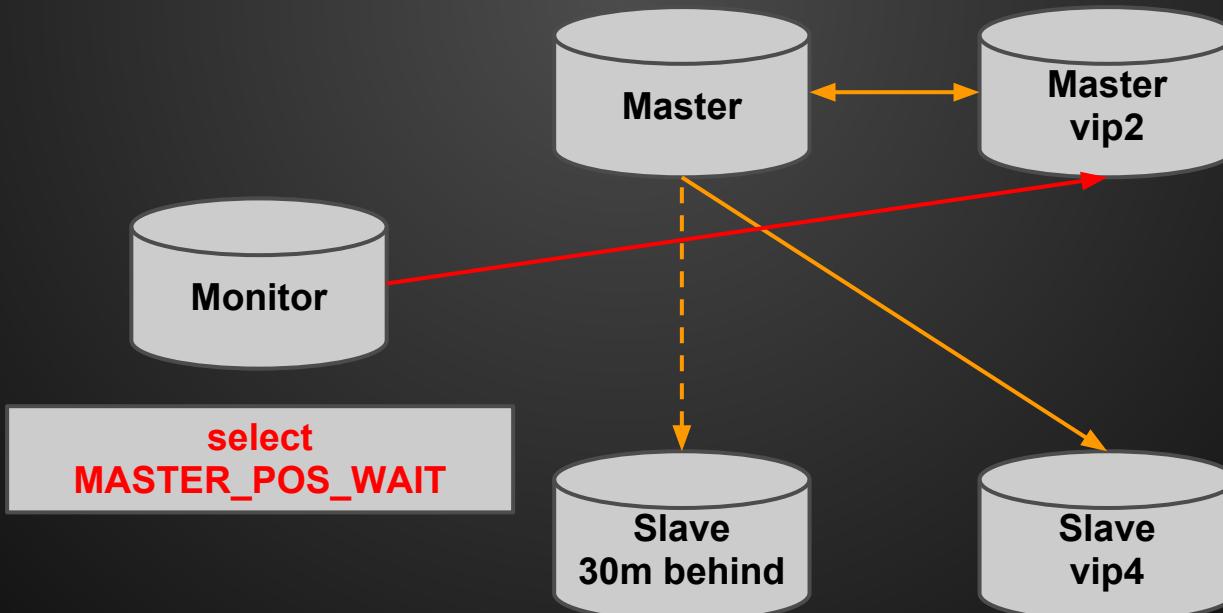
# MMM Problem 2

Writer VIP cannot be accessed when slave is far behind master



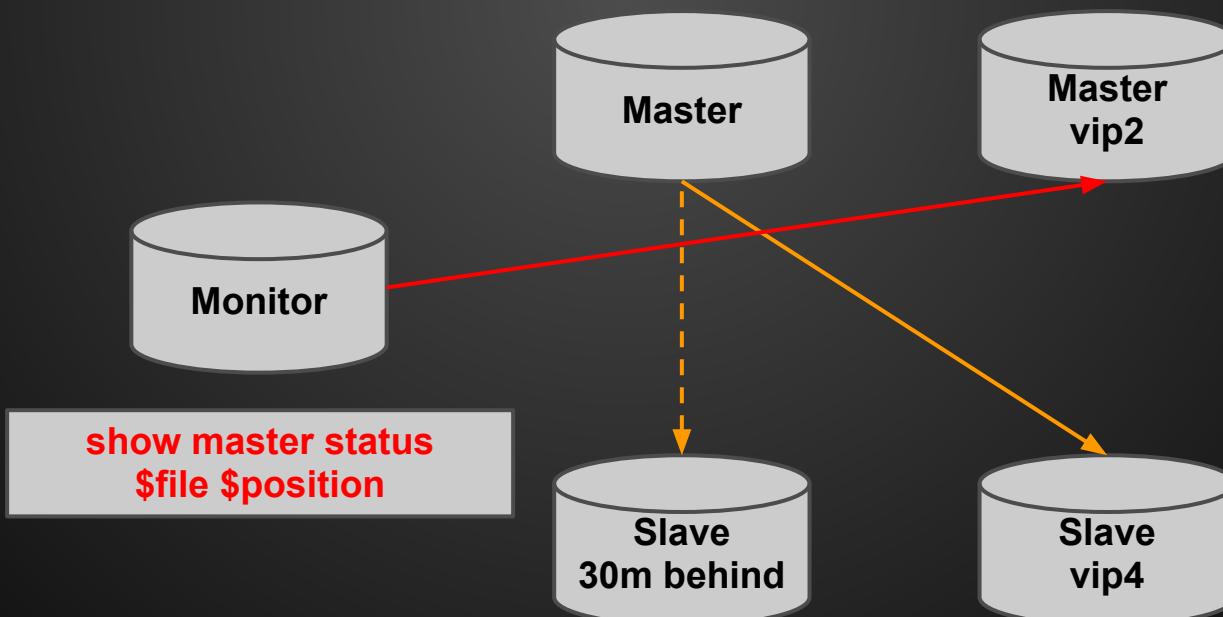
# MMM Problem 2 -- Fix

Record the position on M2 and Bring on VIP1 immediately



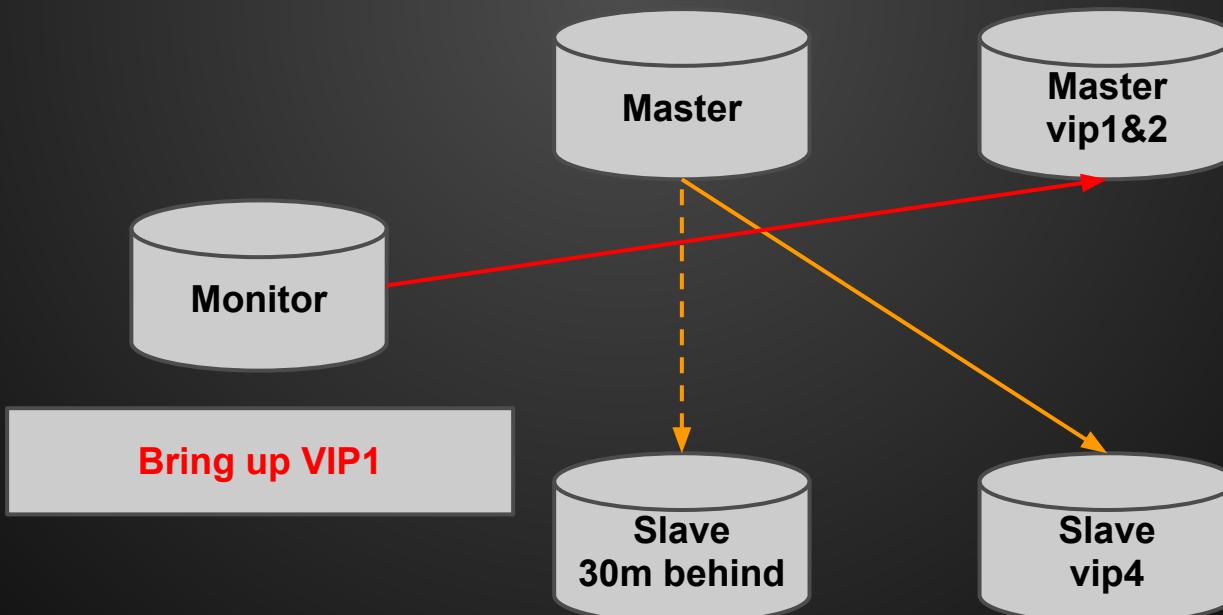
# MMM Problem 2 -- Fix

Record the position on M2 and Bring up VIP1 immediately



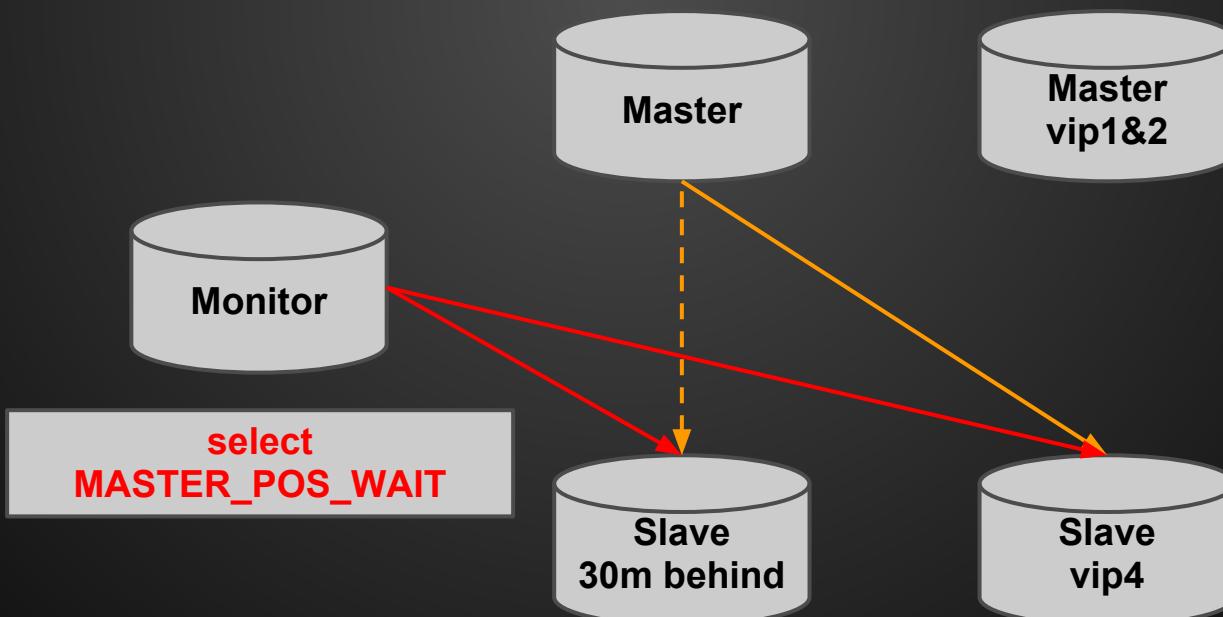
# MMM Problem 2 -- Fix

Record the position on M2 and Bring up VIP1 immediately



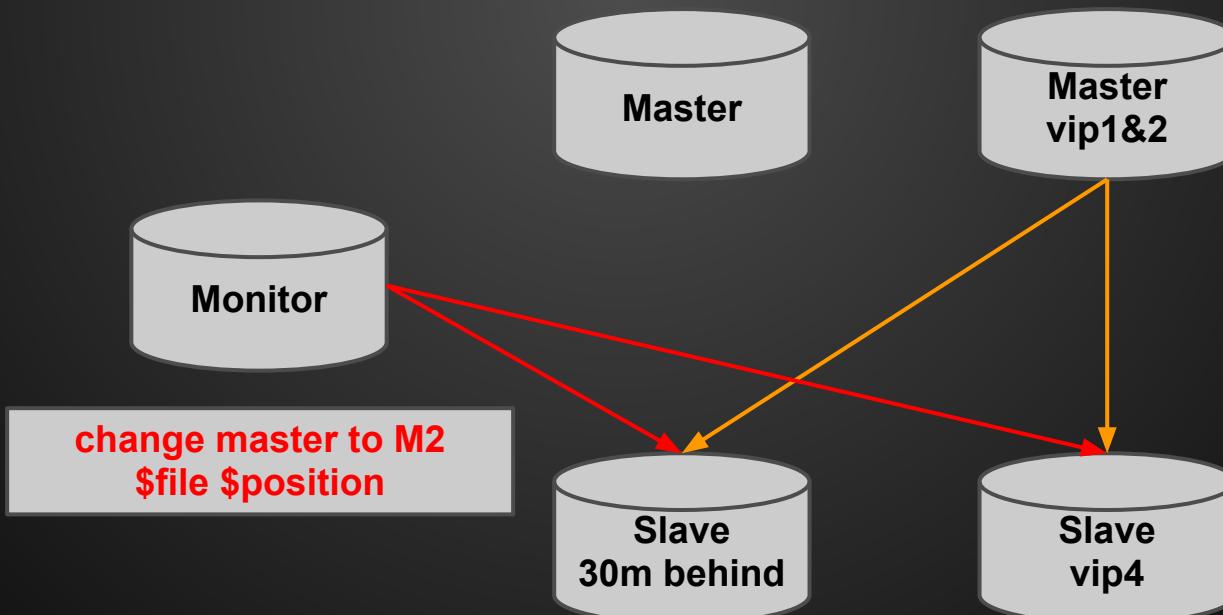
# MMM Problem 2 -- Fix

Record the position on M2 and Bring up VIP1 immediately



# MMM Problem 2 -- Fix

Record the position on M2 and Bring up VIP1 immediately



# Thanks!

## Q&A