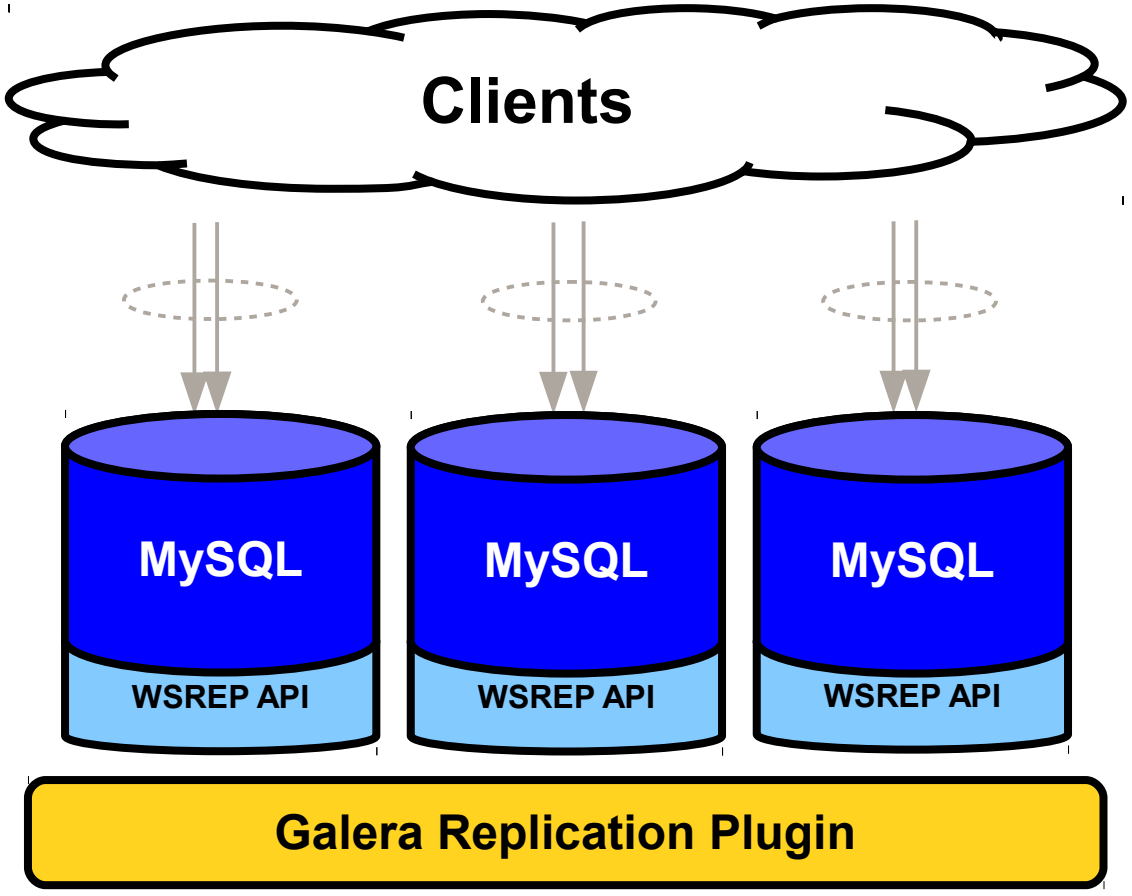
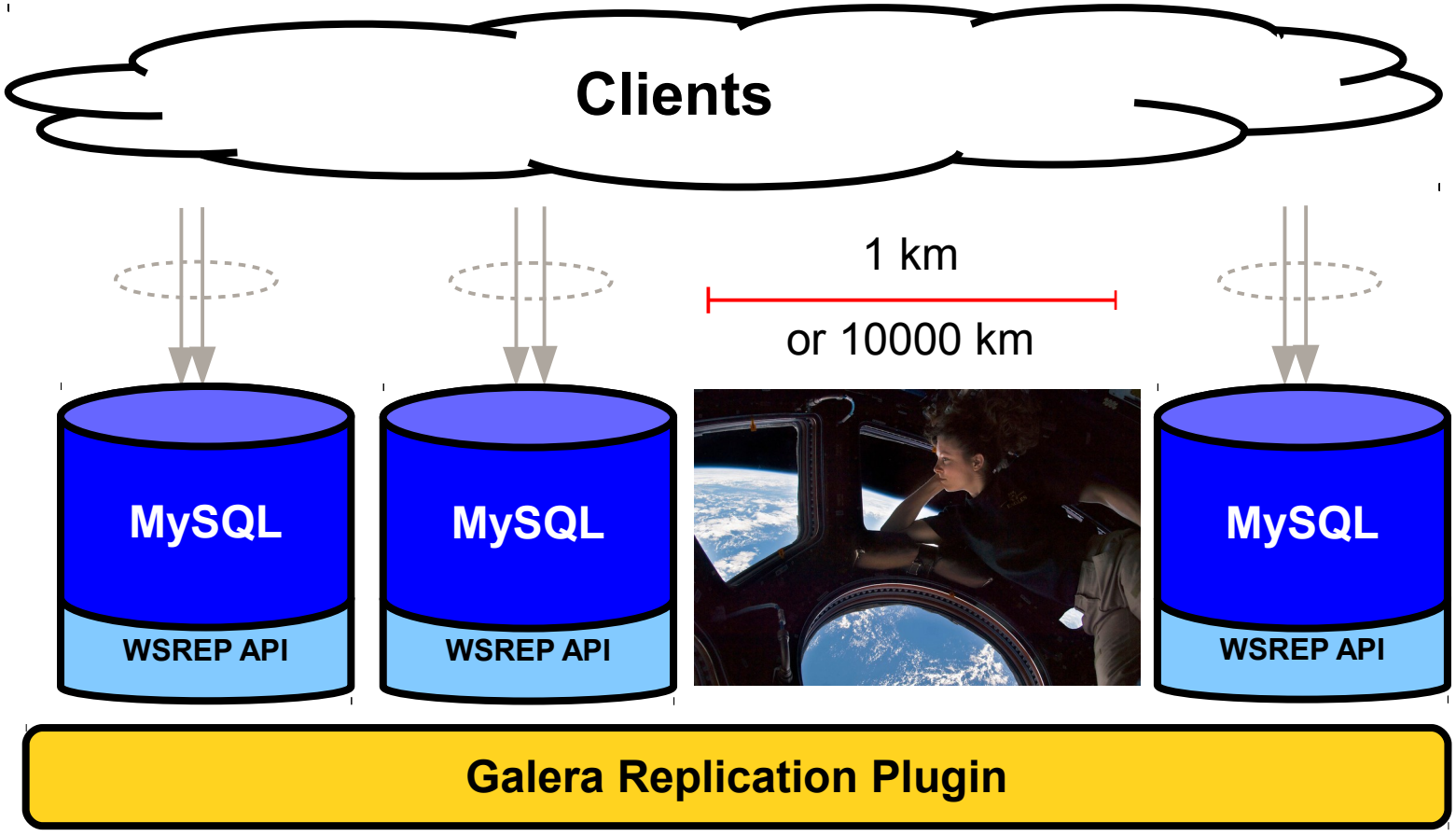
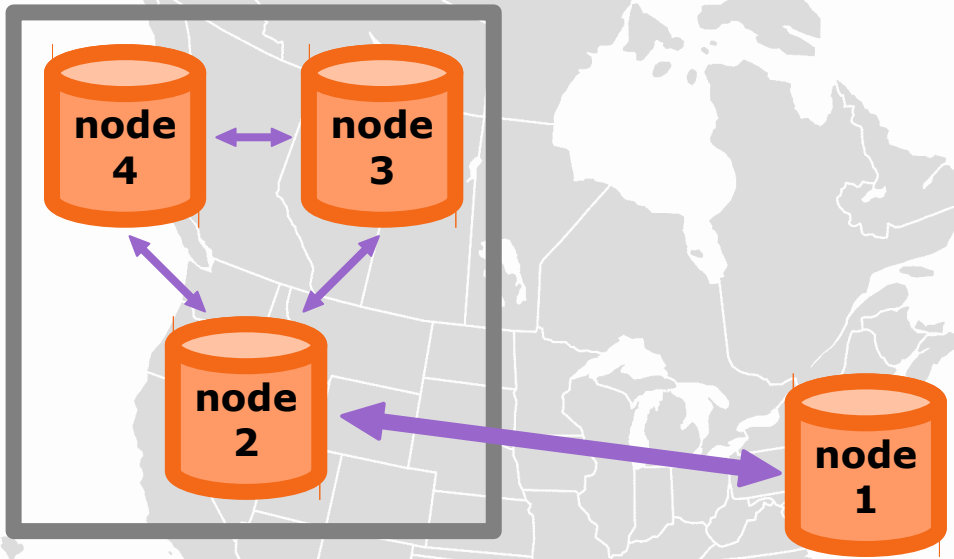
Using Galera Cluster to Power Geo-distributed Applications on the WAN

Philip Stoev
Codership


Clients
MySQL
WSREP API
MySQL
WSREP API
MySQL
WSREP API
Galera Replication Plugin


Clients
1 km
or 10000 km
MySQL
WSREP API
MySQL
WSREP API
MySQL
WSREP API
Galera Replication Plugin

Geo-distribution And Scaling



GALERA CLUSTER

Why WAN replication?

- Go beyond availability zones and achieve multi-data center redundancy
  - multiple availability zones can fail at the same time
- Span multiple cloud providers
- Bring data closer to application
- Distribute global data globally
  - OpenStack's Keystone and Glance databases

GALERA CLUSTER

Bring The Data Closer

- Most queries are read-only anyway
  - answer them from an up-to-date, local copy of the database
- Caching at its finest
  - InnoDB buffer pool takes care of the caching part
  - Galera takes care of “invalidation”, so data is always fresh
  - Single global “source-of-truth” database
- Most round-trip times are due to the MySQL client protocol
  - slash them all except at COMMIT

GALERA CLUSTER

Dedicated Features for WAN Replication

- Galera works across and between continents
  - minimal latency penalty / number of messages exchanged
- No or minimal slave lag
- Optimizations reduce cross-data center traffic
  - updates are sent only once per data center
  - new nodes get initial database from close neighbor
- Encryption
- Detection and automatic eviction of unreliable nodes

GALERA CLUSTER

Basic Configuration

- Specify network location for each node
  - set gmcast.segment=X in wsrep_provider_options
- Open Firewall
  - Galera uses ports 3306, 4567, 4568 and 4444
  - should be open both ways as any node can contact any other node
- Configure IPs
  - set wsrep_node_addres, wsrep_cluster_address with public IPs
  - or use a DNS name that resolves appropriately from any node

GALERA CLUSTER

Security First

- Securing Galera replication traffic and IST
  - set `socket.ssl_key`, `socket.ssl_cert`, `socket.ssl_ca` in wsrep_provider_options
- SST is secured separately
- VPN works too
  - but watch out for flow control and fragmentation

GALERA CLUSTER

Securing SST

SST must be secured separately depending on SST method

- rsync
  - add `CAfile`, `cert`, `key` to `/etc/stunnel/stunnel.conf`
- mysqldump
  - CREATE USER sst_user ... REQUIRE SSL
  - add ssl-ca, ssl-key, ssl-cert to [client] in my.cnf
- xtrabackup
  - add tkey, tcert, encrypt=3 to [SST] in my.cnf

GALERA CLUSTER

Configuring TCP for Performance

```
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216
net.core.rmem_default = 16777216
net.core.wmem_default = 16777216
net.ipv4.tcp_rmem = 4096 87380 16777216
net.ipv4.tcp_wmem = 4096 65536 16777216
net.ipv4.tcp_slow_start_after_idle = 0
```

GALERA CLUSTER

Configuring Galera for Performance

- Avoid fragmentation at binlog level:
  - binlog-row-event-max-size = 1048576
- Avoid flow control and fragmentation at Galera level:
  - wsrep_provider_options="
    gcs.max_packet_size=1048576;
    evs.send_window=512; evs.user_send_window=256"
- Disable InnoDB flush to disk:
  - set innodb_flush_log_at_trx_commit = 0
  - node failures are considered independent

GALERA CLUSTER

Achieving Reliability

How do I get the benefits of synchronous WAN replication but avoid blocking if there is a network problem?

Have  and almost eat it too

GALERA CLUSTER

Avoid Split-Brain

- Use an odd number of data centers
- If two data centers, one should be designated primary:
  - run a Galera arbitrator there
  - run a larger number of nodes
  - use pc.weight in wsrep_provider_options to affect quorum calculation

GALERA CLUSTER

Configure Timeouts

- Review default values for:
  - evs.inactive_timeout=PT15S
  - evs.suspect_timeout=PT5S
- Set up auto-eviction:
  - node will be evicted if it repeatedly suffers network issues
  - it will not be allowed to rejoin without a manual intervention

GALERA CLUSTER

A Latency Example

- EC2 in Nortern Virginia, Sydney and São Paulo (2 nodes)
- Latencies 238 ms, 119 ms and 316 ms

```
# Inserting from Sydney:
mysql> insert into t1 values (REPEAT('a',1000));
Query OK, 1 row affected (0.35 sec)

# Inserting from São Paulo #1
mysql> insert into t1 values (REPEAT('a',1000));
Query OK, 1 row affected (0.35 sec)
```

GALERA CLUSTER

Blobs

- 2.5 Mb worth of data in 5 blobs 512K each

```
mysql> INSERT INTO t2 VALUES (REPEAT('a', 512 * 1024)),(REPEAT('a',
512 * 1024)),(REPEAT('a', 512 * 1024)),(REPEAT('a', 512 * 1024)),
(REPEAT('a', 512 * 1024));
Query OK, 5 rows affected (0.67 sec)
```

GALERA CLUSTER

Thank you

Questions?

Weird setups you want to try out?

Have one node on the moon?

And another on a Raspberry PI?

philip.stoev@galeracluster.com

GALERA CLUSTER