Slow is Down and Down is Dead:
Using GridGain’s Multi Data Center Replication to build a bullet-proof application.

Rachel Pedreschi
Director of Solutions Architecture
GridGain Systems
rachel@gridgain.com
@rachelpedreschi
Tell ‘em what you are going to tell ‘em

**What** is Apache Ignite and GridGain?

**Who** needs multi-datacenter replication?

**Why** is using multi-datacenter replication essential for my modern data intensive application?

**Where** can I deploy it?

**How** does it work?

**When** should I do it?
But first... a word from our sponsor....
But first... a word from our sponsor....
Application

In-Memory
Scalable
No Rip & Replace

Always Available
SQL 99 / ACID / MapReduce
Application

In-Memory
Scalable
No Rip & Replace

Always Available

SQL 99 / ACID / MapReduce

© 2016 GridGain Systems, Inc.
Spark Application

In-Memory
Scalable
No Rip & Replace

Always Available
SQL 99 / ACID / MapReduce

Application
Data Grid

Apache Ignite / GridGain Professional
Data Grid
Compute Grid

Apache Ignite / GridGain Professional
OLTP Caching

OLAP Caching

GaaS

Services and Messaging

Streaming

Data Grid

Compute Grid

Apache Ignite / GridGain Professional
GridGain Enterprise

Security  High Availability  Monitoring and Management

OLTP Caching  OLAP Caching  GaaS  Services and Messaging  Streaming

Data Grid  Compute Grid

Apache Ignite / GridGain Professional
GridGain Enterprise

- Security
- High Availability
- Monitoring and Management

- OLTP Caching
- OLAP Caching
- GaaS
- Services and Messaging
- Streaming

- Data Grid
- Compute Grid

Apache Ignite / GridGain Professional
Heatmap of earthquake, tornado and hurricane hazard areas

- **Earthquake heatmap**
- **Tornado heatmap**
- **Hurricane heatmap**

Legend:
- **High**
- **Medium**
- **Low**
- **None**
• Batching
• Filtering
• Failover
• Complex Topologies
• Pause / Resume
• State Transfer
Server nodes holding cache

Entry Filter

SenderHub

DC1

Conflict Resolver

ReceiverHub

Server nodes holding cache

DC2

Client

Cache Operations

Replication

gridgain.com
Client

Cache
Operations

SenderHub

Server nodes
holding cache

Entry
Filter

Replication

Client

Cache
Operations

ReceiverHub

Conflict
Resolver

Server nodes
holding cache

DC1

DC2
Physics. It’s a thing.

Speed of Light.
186,000 mps.
Not just a good idea…
it’s the law
GridGainConfiguration contains methods for configuring Multi-DC replication. To define a datacenter ID:

```java
GridGainConfiguration ggCfg = new GridGainConfiguration();
ggCfg.setDataCenterId((byte)1);
```
Configure a Sender Cache using `CacheDrSenderConfiguration`. Be sure to set the **batch size** to determine the max entries before SenderHub sends data.

```java
CacheDrSenderConfiguration senderCfg = new CacheDrSenderConfiguration();

// Set batch size
senderCfg.setBatchSendSize(batchSize);

// Add sender cache config to ggCacheCfg
ggCacheCfg.setDrSenderConfiguration(senderCfg);

// Enable dr receiver for this cache
ggCacheCfg.setDrReceiverEnabled(true);
```
Configure a Sender Hub using DrSenderConfiguration.

```java
//create sender connection config
DrSenderConnectionConfiguration drSenderConnectionConfiguration = new DrSenderConnectionConfiguration();

//Set the remote DC to replicate to
drSenderConnectionConfiguration.setDataCenterId((byte)2);

//Set the addresses of remote DC's receiver hub
drSenderConnectionConfiguration.setReceiverAddresses("127.0.0.1:50002");
drSenderConnectionConfiguration.setLocalOutboundAddress("127.0.0.1");
drSenderCfg.setConnectionConfiguration(drSenderConnectionConfiguration);

//Add the sender configuration to the gridgain configuration
ggCfg.setDrSenderConfiguration(drSenderCfg);
```
Configure a Receiver Hub using **DrReceiverConfiguration**.

```java
// Set up the Receiver HUB
DrReceiverConfiguration drReceiverConfiguration = new DrReceiverConfiguration();

// Address receiver hub of this DC is bound to
drReceiverConfiguration.setLocalInboundHost("127.0.0.1");

// TCP port receiver HUB of this data center is bound to
drReceiverConfiguration.setLocalInboundPort(50001);

// Add the receiver configuration to the GridGain configuration
ggCfg.setDrReceiverConfiguration(drReceiverConfiguration);
```
Replication Filtering

```java
public class TextEntryFilter implements CacheDrEntryFilter {

    @Override
    public boolean accept(CacheDrEntry cacheDrEntry) {
        if((int)cacheDrEntry.value()>2) {
            return true;
        }
        else {return false;}
    }
}
```

Set using Sender Configuration

```java
senderCfg.setEntryFilter(new TextEntryFilter());
```
Conflict Resolution

```java
public class ChronologicalConflictResolver implements CacheConflictResolver {

    @Override public void resolve(CacheConflictContext ctx) {
        if (ctx.oldEntry().globalTime() > ctx.newEntry().globalTime())
            ctx.useOld();
        else
            ctx.useNew();
    }
}
```
☒ NOW
☐ LATER
Tell ‘em what you told ‘em

**What** is Apache Ignite and GridGain?

**Who** needs multi-datacenter replication?

**Why** is using multi-datacenter replication essential for my modern data intensive application?

**Where** can I deploy it?

**How** does it work?

**When** should I do it?
GOT QUESTIONS?

Thank you for joining us. Follow the conversation.

www.gridgain.com

@gridgain
#gridgain