Distributed State Made Easy with MySQL

Phillip Morelock - Evite
Director of Operations
phillip@evite.com

Co-presented by O'Reilly Media, Inc. and MySQL AB
What is state?

Pagination

Customization

Cached stuff

Shopping carts

Nebraska
HTTP contains no state mechanism

Either you or your web server have to manage it!
Ways to track state
Tracking a user’s state

- Request parameters
Tracking a user’s state

- **Hidden form elements**

```
<input type="hidden" name="category" value="places" />
<input type="hidden" name="keywords" value="pizza" />
<input type="hidden" name="location" value="90048" />
<input type="hidden" name="price" value="" />
<input type="hidden" name="rating" value="" />
<input type="hidden" name="letter" value="" />
```
Tracking a user’s state

- **Cookies**

  ```
  Cookie: ab_test=prod; defaultEmail=phillip@evite.com;
  eviteAuth=DBBDBDBDBBBDBBDXXDDRR; gt_location=90048;
  category=places; keywords=pizza;
  ```
Tracking a user’s state

- Main transactional database
The `session` abstraction
Map of information

<table>
<thead>
<tr>
<th><strong>KEY</strong></th>
<th><strong>VALUE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>First name</td>
<td>Phillip</td>
</tr>
<tr>
<td>Last name</td>
<td>Morelock</td>
</tr>
<tr>
<td>Email</td>
<td><a href="mailto:phillip@evite.com">phillip@evite.com</a></td>
</tr>
<tr>
<td>Keywords</td>
<td>pizza</td>
</tr>
<tr>
<td>Location</td>
<td>90048</td>
</tr>
</tbody>
</table>
Web server provides `session` object

**Initial request**

`session.setFirstName(first_name)`

**Subsequent request**

`first_name = session.getFirstName()`
Approaches to sessions
One Server
Memory-based sessions

- Server stores table of sessions in memory

<table>
<thead>
<tr>
<th>ID</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ca8f7d8e</td>
<td>stuff</td>
</tr>
<tr>
<td>ae55f7ea</td>
<td>stuff</td>
</tr>
<tr>
<td>ee5fa55e</td>
<td>stuff</td>
</tr>
</tbody>
</table>
Handful of Servers
Lots of servers
(a load-balanced environment)
Behind the load balancer...

The Filesystem
The Filesystem Approach

- Store files containing user session data
Files suck!

- Locking mechanisms stink...slowly!
Files really suck!

- Individual I/O’s for each request become a serious bottleneck!
I/O Throughput of Session File Server

* effective session writes / second
Behind the load balancer...

Network / Multicast Cluster
Cluster / multicast approach

- Replicate memory-based sessions to other web servers across network

Cluster configuration

Multicast configuration
Cluster / multicast drawback #1:

Adding “Nth” server increases network traffic by “N”, not by 1
Network traffic for cluster / multicast
Cluster / multicast drawback #2:

Complexity and troubleshooting
Troubleshooting headaches* for cluster / multicast

* in headaches / minute
Cluster / multicast approach

Troubleshooting, anyone?
Whatcha gonna do?

What system or piece of software could possibly store volatile data in a central place on your network?
Might I Suggest...
Use a database!

<table>
<thead>
<tr>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Better I/O scalability than you</td>
<td>- Single point of failure</td>
</tr>
<tr>
<td>- Cross-platform availability</td>
<td>- As reliable as your database</td>
</tr>
<tr>
<td>- Ease of troubleshooting</td>
<td></td>
</tr>
<tr>
<td>- As reliable as your database</td>
<td></td>
</tr>
</tbody>
</table>
MySQL is fast

- 576.51 reads/s
- 104.31 updates/s
- 45.66 inserts/s
- 12.67 deletes/s
Disks kill

Friend #1: Buffer pool!

Buffer pool hit rate 995 / 1000
Disks kill

Friend #2: tmpfs

```
# mount | grep mysql
tmpfs on /mysql/mysql/data-tmpfs type tmpfs (rw)
```

```
# df -h | grep mysql
tmpfs 3.0G 1.0G 2.1G 34% /mysql/mysql/data-tmpfs
```
How does it work?
Shove binary data into BLOBs

Hash table map of values is serialized to a BLOB for insertion as binary data

Object graph → Serializer.java → Session DB → Deserializer.java → Object graph
The J2EE Request Cycle

Normally, the servlet container handles sessions behind the scenes.
The J2EE Request Cycle

At Evite, every request passes through our session handler.
CREATE TABLE sessionrows (  
  session_key char(20) NOT NULL,  
  session_blob mediumblob NOT NULL,  
  access_time timestamp(14) NOT NULL,  
  PRIMARY KEY (`session_key`),  
  KEY `at_index` (`access_time`)  
) TYPE = InnoDB;
Lifetime of Evite user sessions

1 minute  5  10  20  30  60 minutes
Evite Session API

**Initial request**

```java
HttpSession session =
    EviteHttpSessionFactory.getInstance(request);
...
userFirstName = userBean.getFirstName();
session.putValue("userFirstName", userFirstName);
```

**Subsequent request**

```java
HttpSession session =
    EviteHttpSessionFactory.getInstance(request);
...
Object oUFN = session.getValue("userFirstName");
if (oUFN != null) userFirstName = (String) oUFN;
```
CREATE TABLE sessions (  
id char(32) NOT NULL,  
a_session text,  
PRIMARY KEY (`id`)  
);
Apache::Session

Initial request

tie %session, 'Apache::Session::MySQL', undef, 
  { DataSource => 'dbi:mysql:sessions' };
$session_id = $session{_session_id};

$session{first_name} = "Chuck";
$session{an_array_ref} = [ $one, $two, $three ];
$session{an_object} = new Some::Class;

Subsequent request

tie %session, 'Apache::Session::MySQL', $id, 
  { DataSource => 'dbi:mysql:sessions' };

$first_name = $session{first_name};
A cross-platform solution
Multi-platform environments

J2EE

MySQL
session
database

mod_perl

Profiles

Online Catalog

Shared shopping cart

Checkout / fulfillment system
Migration and session-sharing

Custom serialization scheme allows cross-platform access

Object graph

Serializer.java

Deserialized.java

Serializer.pm

Deserialized.pm

Session DB

Object graph

Object graph

Object graph

Object graph
Thank you!

Phillip Morelock
phillip@evite.com