Actor Bug: Concurrency - Recover

The actor model is built to allow actors that fail to recover and hopefully prevent further errors by having it's master recreate it. However, there can be unexpected variables or events involved during this recreation process and if not anticipated would lead to errors.

In Collossus, if a worker dies and recovered, there was a bug where it's connection attempts would all be refused. This was due to the recovery happening during a delicate portion of it's initialization. The server had not yet registered with the worker and then after the recovery, the worker would operate under the assumption that it already had the registrations it needed. This was solved by adding a connection registration attempt to the process.

The server has a ConnectionRefused case added to account for it’s connection attempts being rebutted by the worker. In this case, it sends a NewConnection message to the worker which has had this new type of message added to it’s API. This means when the situation of recovering of a worker before it’s registered a server occurs, the initial refusal will be caught and a connection made anyway. The server also now uses a MaxConnectionRegisterAttempts variable to limit the amount of retries for connecting to the faulty workers.

Link: https://github.com/tumblr/colossus/issues/132

Code:

colossus/core/Server.scala

```
+    case ConnectionRefused(sc, attempt) => if (attempt >= Server.MaxConnectionRegisterAttempts) {
+      log.error(s"Failed to register new connection $attempt times, closing")
+      sc.close()
+      self ! ConnectionClosed(0, DisconnectCause.Error(new Server.MaxConnectionRegisterException))
+    } else {
+      router ! Worker.NewConnection(sc, attempt + 1)
+    }
....
+
+  val MaxConnectionRegisterAttempts = 3
+  class MaxConnectionRegisterException extends Exception("Maximum number of connection register attempts
reached")
+
....
+  /** Sent from a worker to the server when the server is not registered with the worker.
+   *
+   * This generally happens when a worker has just been killed and restarted.
```

```
+   */
+  case class ConnectionRefused(channel: SocketChannel, attempt: Int)
+
```

colossus/core/Worker.scala

```
-    case NewConnection(sc) => delegators.get(sender()).map{delegator =>
+    case NewConnection(sc, attempt) => delegators.get(sender()).map{delegator =>
....
     }.getOrElse{
       //TODO: do something with the connection
+      sender ! Server.ConnectionRefused(sc, attempt)
-  private[core] case class NewConnection(sc: SocketChannel)
+
+  /** Sent from Servers
+   * @param sc the underlying socketchannel of the connection
+   * @param attempt used when a worker refuses a connection, which can happen if a worker has just restarted
and hasn't yet re-registered servers
+   */
+  private[core] case class NewConnection(sc: SocketChannel, attempt: Int = 1)
+
```