```java
entry.setAdmin("admins"); entry.setAdmin("users"); entry.setAdmin("guests,users");
. . .
```

activemq/activemq-broker/…/security/AuthorizationEntry.java

```java
1. public void setAdmin(String roles) throws Exception {
2.     adminRoles = roles;
3.     setAdminACLs(parseACLs(adminRoles));
4. }

1. protected Set<Object> parseACLs(String roles) throws Exception {
2.     Set<Object> answer = new HashSet<Object>();
3.     StringTokenizer iter = new StringTokenizer(roles, ",");
4.     while (iter.hasMoreTokens()) {
5.         String name = iter.nextToken().trim();
6.         String groupClass = (this.groupClass != null ?
               this.groupClass : DefaultAuthorizationMap.DEFAULT_GROUP_CLASS);
7.         answer.add(DefaultAuthorizationMap.createGroupPrincipal(name,
                                                    groupClass));
8.     }
9.     return answer;
10.}
```

activemq/activemq-broker/…/security/DefaultAuthorizationMap.java

```java
1.  public static Object createGroupPrincipal(
                String name, String groupClass) throws Exception {
2.      if (WILDCARD.equals(name)) {
        . . .
19.     }
20.     Object[] param = new Object[]{name};
21.
22.     Class<?> cls = Class.forName(groupClass);
23.
24.     Constructor<?>[] constructors = cls.getConstructors();
25.     int i;
26.     Object instance;
    . . .
33.     if (i < constructors.length) {
34.         instance = constructors[i].newInstance(param);
35.     } else {
    . . .
54.
```

constructors[i].newInstance(param) can lead to a **Code Injection** vulnerability if **'param' contains malicious data.**