服务器名称：

使用的系统：Red Hat 6.5 64 位

数量：7台(生产)

Cpu: 4/8

内存：16/64G

硬盘：磁盘1:64G，磁盘2：500G

软件：weblogic，Redis3.2.8，mongodb3.4.0，Oracle 11g

数据库Oracle11g

所在服务器的内存为64G,配置内存为25G

最大连接数缺省值为150，太小，调整为1500

集群服务器内存4g, 非堆内存2048

受管资源堆内存8G, 非堆内存2G, 线程数50-200自调整

weblogic的服务器：65和66上部署有

WebLogic是用于开发、集成、部署和管理大型分布式Web应用、网络应用和数据库应用的Java应用服务器

**GJYW项目在服务器上的分布情况**

生产环境下包含7台服务器，一台weblogic主服务器，两台代码服务器，一台Oracle数据库服务器，三台redis/mongodb服务器，发布项目时只需要weblogic主服务器中attachment目录下发布，他会自动更新到两台代码服务器中，redis采用的是一主两从三哨兵的方式排列在三台服务其中，MongoDB采用一主两从的方式排列。

**Redis**

Redis中主要缓存的有字典表，国家表，组织机构表，缓存可以提高速度在五六秒

**一主两从三哨兵的运行方式**

一主两从三哨兵，主redis具有读写查权限，从redis只有查权限，三个redis与三个哨兵分别分配到三个服务器上，由三个哨兵及时监听，当主redis挂了，30秒由两台哨兵确认确实挂了，然后制定从redis中任意取一个为主redis，另一个为从redis，继续执行任务

**网络存储使用的是直连式存储（SAS）**

**Shiro的使用及原理，好处**

Shiro是权限管理框架，实现用户认证、用户授权

Shiro的配置都在xml配置文件中配置，还有 realm（领域）用户信息验证，当用户登录或者访问时，shiro会生成一个token（用户令牌）缓存起来，每次访问都要进行验证。

**Shiro与spring security的区别**

1. spring security依赖于Spring
2. shiro属于轻量框架，使用简单
3. Shiro不仅可以在web项目中使用，还可以在javase中使用

**Solr服务器**

Solr非关系性数据库，国际业务使用他来实现全文检索

**MongoDB数据库**

MongoDB非关系性数据库，国际业务使用他来实现日志存放

日志分为登录日志和操作日志，使用的是自定义注解和Spring Aop面向切面

**磁盘分区、格式化、挂载**

磁盘分区后不能直接使用需要挂载，挂载前需要格式化，因为每个操作系统所设置的文件属性和权限并不相同，为了存放数据因此分区需要格式化后才能够放入需要的数据。

操作系统的文件系统格式并不相同。比如Windows系统的文件系统格式是FAT/FAT16/NTFS，而CentOS 就是Ext2/Ext3/Ext3 我这里是格式化成 ext4

**如何搭建svn分支**

**搭建svn分支的好处**

1. 生产上发布时可以选择性发布
2. 可以完全保存上一次发布的版本
3. 能够在及时查看发布版本是否报错

**虚拟路径virtual-directory-mapping**

虚拟路径主要是为了在我们请求服务器路径时进行映射用的。

SSO单点登录（基于HTTP header的单点登录）

1. 应用系统与代理服务器间交互采用SSL加密，应用服务器接受到参数需要对代理服务器的证书和IP进行校验，校验内容为检验证书的证书号和IP是否在受信范围

Webservice 使用http

所有与IAM系统集成实现帐号集中管理和单点登录的系统帐号均在IAM系统中集中管理

服务器之间使用ssl双向认证来保证服务期间的通信安全可靠

在身份认证系统的认证服务器与您的应用系统应用服务器（tomcat）之间建立SSL双向认证，从而保证双方服务器之间通信的安全可靠

**集群环境下如何保证spring定时器只执行一个**

项目中我们使用Spring定时器会去别的系统中定时去取数据，这里就遇到了一个问题，本来取一次数据就可以，由于集群环境，所以现在搭建了多少个项目就需要执行几次，造成了别人项目的压力。

解决方法：

1. 绑定ip固定某台服务器可以执行，但我们每台服务器有两个节点，方案不可行
2. 后来想到使用redis来保存一个状态位，1表示执行，0表示不执行，在这有个问题是操作这个状态有两步：第一步查看状态是否为1，第二步修改。当第一个节点查看到为1，准备去修改为0时，第二个节点已经看到为1，这时两个节点都会执行，这时想到了使用redis的事务加watch关键字，由于redis事务是命名打包执行，中间不执行其他操作，所以也实现不了
3. 既然redis的事务不可以，那数据库的应该可以吧。从而最终找到了现在的解决方案。

使用for update查询 然后判断为1时，去修改为0，别的查询到为0，就不执行了。

Multi 进入事务模式 将任务放进队列（所有的任务都返回queued（排队））

Exec 任务才开始执行（Redis 2.6.5之前，排队中的错误命令不会被执行，直接消除事务）

Discord 消除队列中的所有命令，退出事务

Watch

Redis不支持回滚