Ruoyi Case

ruoyi分析

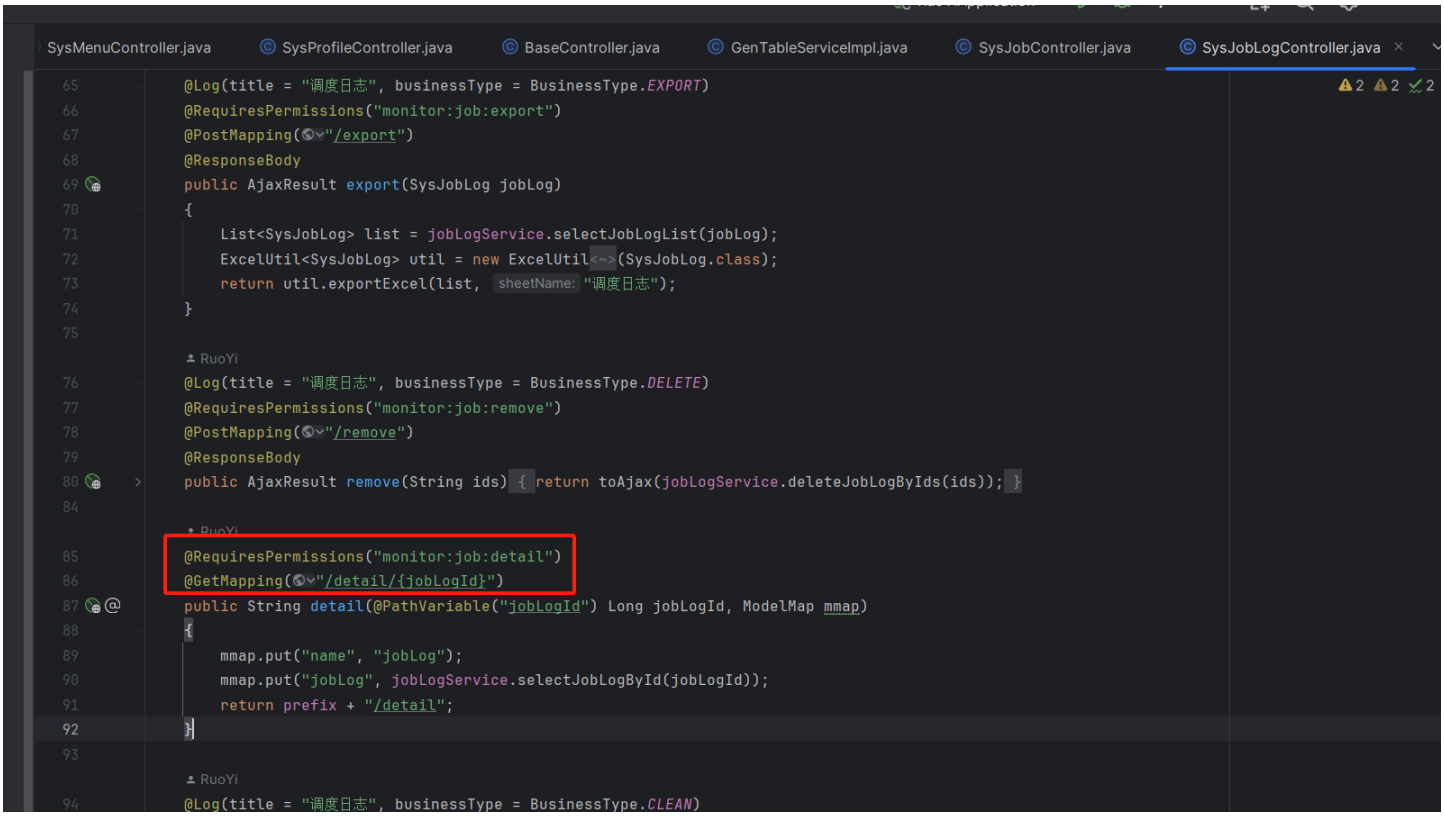
若以自带shior框架和spring框架，所以大多数是误报。我们使用seepseek进行过滤后来看一下结果。

在一共149个报出的case中，deepseek-r1认为其中19个存在漏洞。

Case 1 con: 6

```
@RequiresPermissions("monitor:job:detail")
    @GetMapping("/detail/{jobLogId}")
    public String detail(@PathVariable("jobLogId") Long jobLogId, ModelMap
mmap)
    {
        mmap.put("name", "jobLog");
        mmap.put("jobLog", jobLogService.selectJobLogById(jobLogId));
        return prefix + "/detail";
    }

    /**
     * 通过调度任务日志ID查询调度信息
     *
     * @param jobLogId 调度任务日志ID
     * @return 调度任务日志对象信息
     */
    @Override
    public SysJobLog selectJobLogById(Long jobLogId)
    {
        return jobLogMapper.selectJobLogById(jobLogId);
    }
```



这里可能存在水平越权，如果认为JobLog是敏感信息的话。

Case 2 con: 11

```java
@RequiresPermissions("monitor:job:detail")
    @GetMapping("/detail/{jobId}")
    public String detail(@PathVariable("jobId") Long jobId, ModelMap mmap)
    {
        mmap.put("name", "job");
        mmap.put("job", jobService.selectJobById(jobId));
        return prefix + "/detail";
    }

    /**
     * 通过调度任务ID查询调度信息
     *
     * @param jobId 调度任务ID
     * @return 调度任务对象信息
     */
    @Override
    public SysJob selectJobById(Long jobId)
    {
        return jobMapper.selectJobById(jobId);
    }
```

这个和刚刚那个差不多，刚刚那个是看log，这个是看job本身的信息，如果属于敏感信息的话存在水平越权

```
RuoYi +1
@Controller
@RequestMapping("/monitor/job")
public class SysJobController extends BaseController
{
```

SysProfileController.java | BaseController.java | GenTableServiceImpl.java | SysJobController.java

```
71
    RuoYi
72  @Log(title = "定时任务", businessType = BusinessType.DELETE)
73  @RequiresPermissions("monitor:job:remove")
74  @PostMapping("/remove")
75  @ResponseBody
76  public AjaxResult remove(String ids) throws SchedulerException
77  {
78      jobService.deleteJobByIds(ids);
79      return success();
80  }
81
    RuoYi
82  @RequiresPermissions("monitor:job:detail")
83  @GetMapping("/detail/{jobId}")
84  public String detail(@PathVariable("jobId") Long jobId, ModelMap mmap)
85  {
86      mmap.put("name", "job");
87      mmap.put("job", jobService.selectJobById(jobId));
88      return prefix + "/detail";
89  }
90
```

这个通过访问/monitor/job/detail/{id}就可以访问到。理论上一个用户应该不能去查看所有的定时任务信息吧？感觉这里缺少一个水平鉴权。那么请注意，**一个用户应该不能去查看所有的定时任务信息**这个逻辑是传统工具所不能检测的。

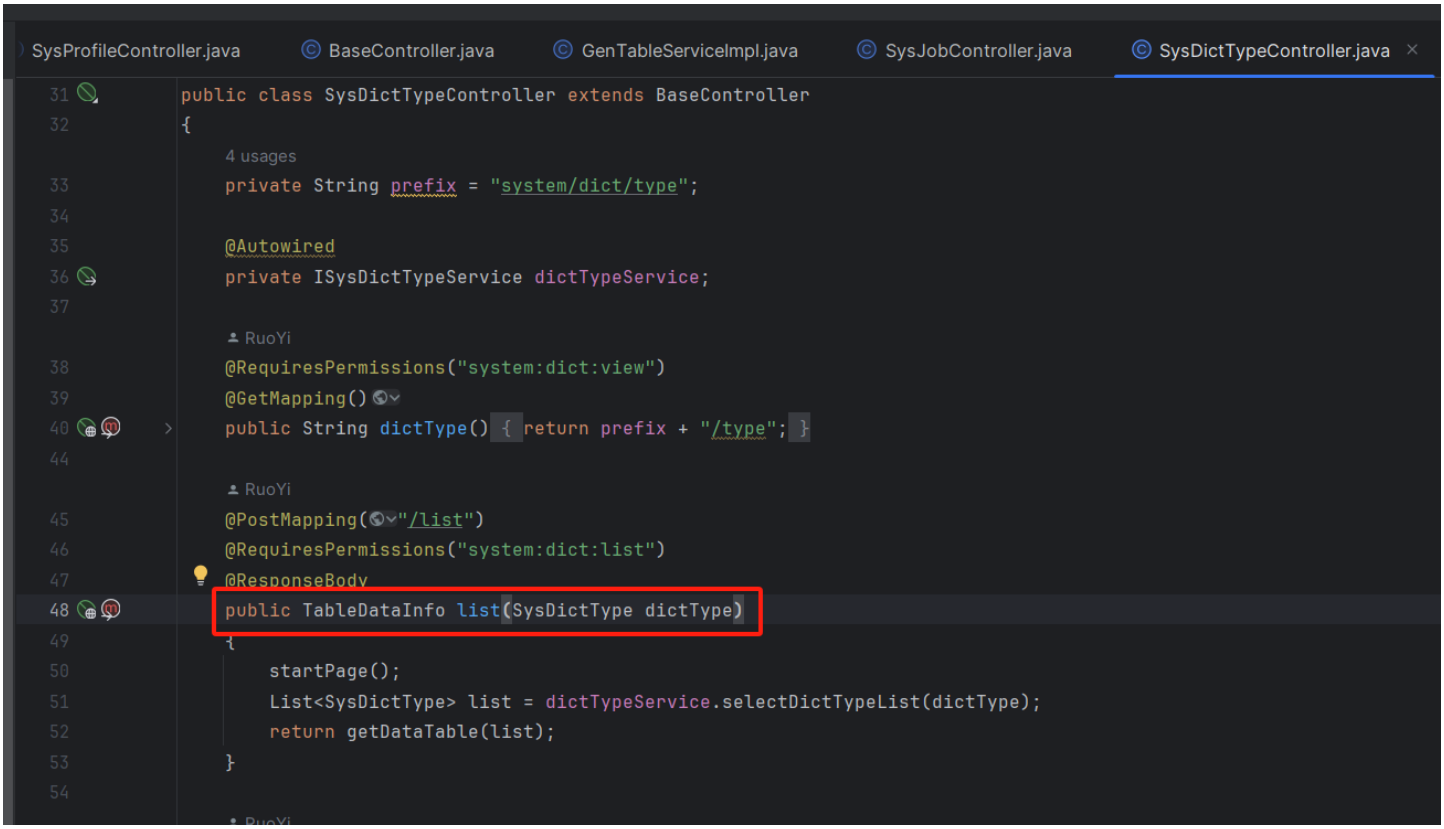
Case3 con: 18

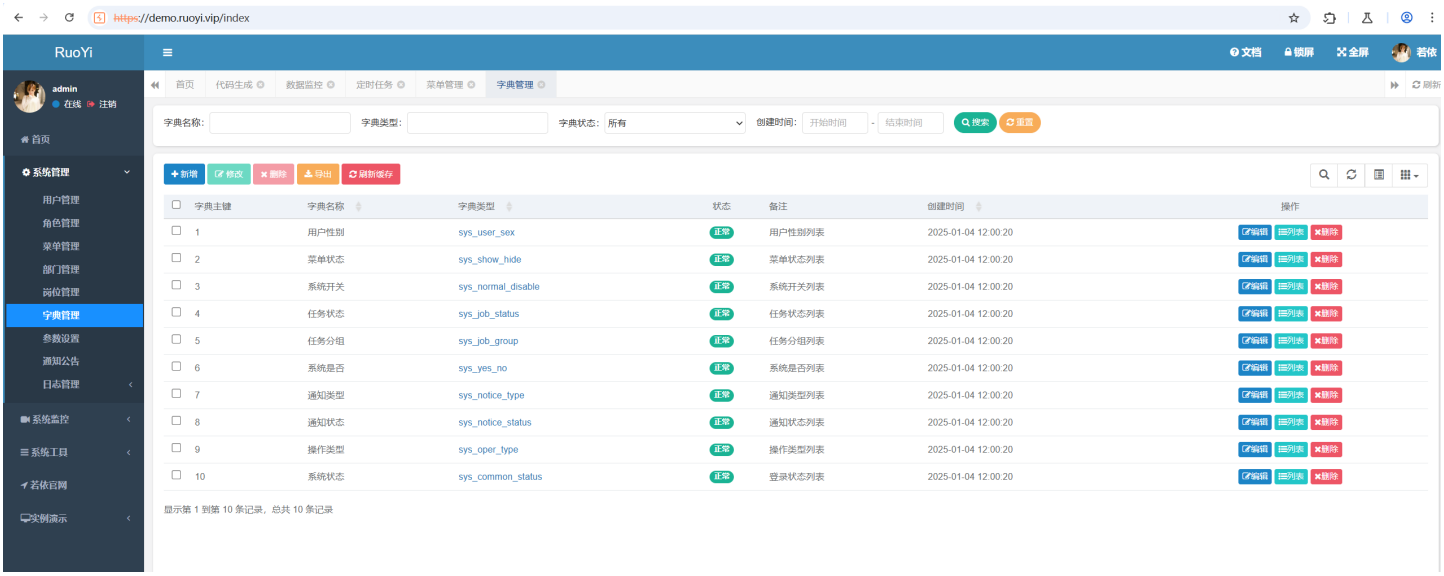
```
1       @PostMapping("/list")
2       @RequiresPermissions("system:dict:list")
3       @ResponseBody
4       public TableDataInfo list(SysDictType dictType)
5       {
6           startPage();
```

```
        List<SysDictType> list = dictTypeService.selectDictTypeList(dictType);
        return getDataTable(list);
    }

    /**
     * 根据条件分页查询字典类型
     *
     * @param dictType 字典类型信息
     * @return 字典类型集合信息
     */
    @Override
    public List<SysDictType> selectDictTypeList(SysDictType dictType)
    {
        return dictTypeMapper.selectDictTypeList(dictType);
    }

    /**
     * 响应请求分页数据
     */
    @SuppressWarnings({ "rawtypes", "unchecked" })
    protected TableDataInfo getDataTable(List<?> list)
    {
        TableDataInfo rspData = new TableDataInfo();
        rspData.setCode(0);
        rspData.setRows(list);
        rspData.setTotal(new PageInfo(list).getTotal());
        return rspData;
    }
```





这个感觉也存在水平越权问题呀，只是校验了以下是否有查看list的权限

Case4 con: 28

```
1       /**
2        * 查询数据表字段列表
3        */
4       @RequiresPermissions("tool:gen:list")
5       @PostMapping("/column/list")
```

```
6       @ResponseBody
7       public TableDataInfo columnList(GenTableColumn genTableColumn)
8       {
9           TableDataInfo dataInfo = new TableDataInfo();
10          List<GenTableColumn> list =
    genTableColumnService.selectGenTableColumnListByTableId(genTableColumn);
11          dataInfo.setRows(list);
12          dataInfo.setTotal(list.size());
13          return dataInfo;
14      }
15
16      /**
17       * 查询业务字段列表
18       *
19       * @param genTableColumn 业务字段信息
20       * @return 业务字段集合
21       */
22      @Override
23      public List<GenTableColumn>
    selectGenTableColumnListByTableId(GenTableColumn genTableColumn)
24      {
25          return
    genTableColumnMapper.selectGenTableColumnListByTableId(genTableColumn);
26      }
```

同样需要判断一些信息是否敏感。

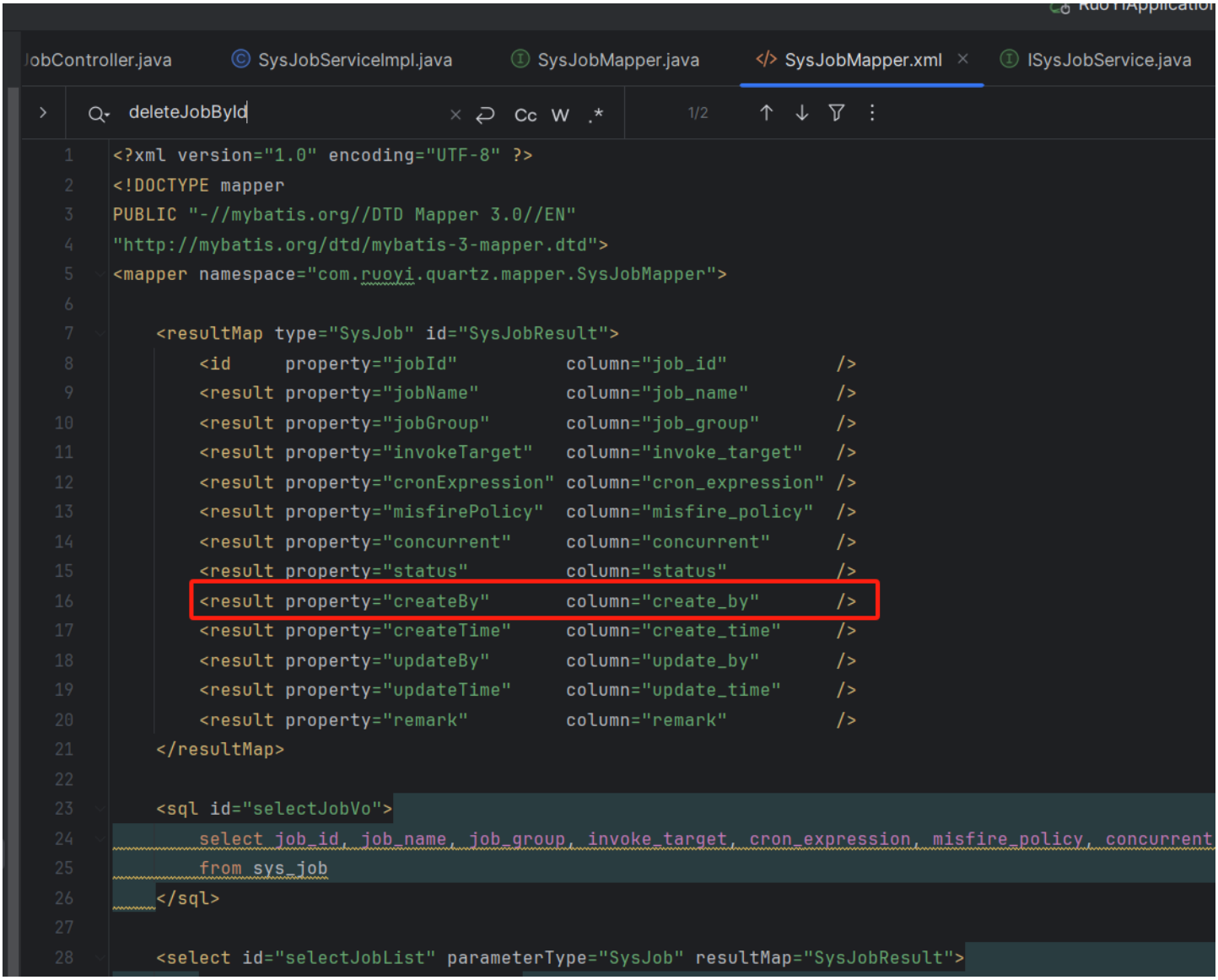
这个和上边那个是一样的

Case5 con:30

```
1       /**
2        * 修改调度
3        */
4       @RequiresPermissions("monitor:job:edit")
5       @GetMapping("/edit/{jobId}")
6       public String edit(@PathVariable("jobId") Long jobId, ModelMap mmap)
7       {
8           mmap.put("job", jobService.selectJobById(jobId));
9           return prefix + "/edit";
10      }
11
12      /**
13       * 通过调度任务ID查询调度信息
```

```java
     *
     * @param jobId 调度任务ID
     * @return 调度任务对象信息
     */
    @Override
    public SysJob selectJobById(Long jobId)
    {
        return jobMapper.selectJobById(jobId);
    }
```

如果job是每人一个的话，是有问题的。但是如果要判断job是不是每人一个，就是MoC那样的思路了，看userId。还是说job并不和用户绑定？但是我怎么看这玩意是和用户绑定的啊，至少按MoC的说法这里是不是应该算？



这个如果上边那个查询job是有问题的，那么这里这个修改也是有问题的，因为不能修改别人的

Case6 con: 34

```
    /**
     * 选择部门树
     *
     * @param deptId 部门ID
     */
    @RequiresPermissions("system:user:list")
    @GetMapping("/selectDeptTree/{deptId}")
    public String selectDeptTree(@PathVariable("deptId") Long deptId,
ModelMap mmap)
    {
        mmap.put("dept", deptService.selectDeptById(deptId));
        return prefix + "/deptTree";
    }

    /**
     * 根据部门ID查询信息
     *
     * @param deptId 部门ID
     * @return 部门信息
     */
    @Override
    public SysDept selectDeptById(Long deptId)
    {
        return deptMapper.selectDeptById(deptId);
    }
```

选择部门树算敏感信息吗?

case7 con:38

```
   /**
     * 修改字典类型
     */
    @RequiresPermissions("system:dict:edit")
    @GetMapping("/edit/{dictId}")
    public String edit(@PathVariable("dictId") Long dictId, ModelMap mmap)
    {
        mmap.put("dict", dictTypeService.selectDictTypeById(dictId));
        return prefix + "/edit";
    }

    /**
```

```
13          * 根据字典类型ID查询信息
14          *
15          * @param dictId 字典类型ID
16          * @return 字典类型
17          */
18         @Override
19         public SysDictType selectDictTypeById(Long dictId)
20         {
21             return dictTypeMapper.selectDictTypeById(dictId);
22         }
```

这个理论上不能被随意修改的吧。字典树是什么东西?

Case8 con:40

```
1          /**
2           * 新增
3           */
4          @RequiresPermissions("system:menu:add")
5          @GetMapping("/add/{parentId}")
6          public String add(@PathVariable("parentId") Long parentId, ModelMap mmap)
7          {
8              SysMenu menu = null;
9              if (0L != parentId)
10             {
11                 menu = menuService.selectMenuById(parentId);
12             }
13             else
14             {
15                 menu = new SysMenu();
16                 menu.setMenuId(0L);
17                 menu.setMenuName("主目录");
18             }
19             mmap.put("menu", menu);
20             return prefix + "/add";
21         }
22
23         /**
24          * 根据菜单ID查询信息
25          *
26          * @param menuId 菜单ID
27          * @return 菜单信息
28          */
```

```
29      @Override
30      public SysMenu selectMenuById(Long menuId)
31      {
32          return menuMapper.selectMenuById(menuId);
33      }
```

这个也感觉不像能随便改的

Case9 con: 50

```
1       /**
2        * 修改菜单
3        */
4       @RequiresPermissions("system:menu:edit")
5       @GetMapping("/edit/{menuId}")
6       public String edit(@PathVariable("menuId") Long menuId, ModelMap mmap)
7       {
8           mmap.put("menu", menuService.selectMenuById(menuId));
9           return prefix + "/edit";
10      }
11
12      /**
13       * 根据菜单ID查询信息
14       *
15       * @param menuId 菜单ID
16       * @return 菜单信息
17       */
18      @Override
19      public SysMenu selectMenuById(Long menuId)
20      {
21          return menuMapper.selectMenuById(menuId);
22      }
```

修改菜单信息

Case10 con:53

```
1       @RequiresPermissions("monitor:job:list")
```

```
    @PostMapping("/list")
    @ResponseBody
    public TableDataInfo list(SysJobLog jobLog)
    {
        startPage();
        List<SysJobLog> list = jobLogService.selectJobLogList(jobLog);
        return getDataTable(list);
    }

    /**
     * 获取quartz调度器日志的计划任务
     *
     * @param jobLog 调度日志信息
     * @return 调度任务日志集合
     */
    @Override
    public List<SysJobLog> selectJobLogList(SysJobLog jobLog)
    {
        return jobLogMapper.selectJobLogList(jobLog);
    }

    /**
     * 响应请求分页数据
     */
    @SuppressWarnings({ "rawtypes", "unchecked" })
    protected TableDataInfo getDataTable(List<?> list)
    {
        TableDataInfo rspData = new TableDataInfo();
        rspData.setCode(0);
        rspData.setRows(list);
        rspData.setTotal(new PageInfo(list).getTotal());
        return rspData;
    }
```

也是查看Job

Case11 con:54

```
    /**
     * 修改岗位
     */
    @RequiresPermissions("system:post:edit")
    @GetMapping("/edit/{postId}")
```

```
    public String edit(@PathVariable("postId") Long postId, ModelMap mmap)
    {
        mmap.put("post", postService.selectPostById(postId));
        return prefix + "/edit";
    }

    /**
     * 通过岗位ID查询岗位信息
     *
     * @param postId 岗位ID
     * @return 角色对象信息
     */
    @Override
    public SysPost selectPostById(Long postId)
    {
        return postMapper.selectPostById(postId);
    }
```

仍然只有垂直越权没有水平越权。这个东西MoC检测不出来吗，他和userId没有绑定吗?

Case12 con: 58

```
    /**
     * 修改保存参数配置
     */
    @RequiresPermissions("system:config:edit")
    @Log(title = "参数管理", businessType = BusinessType.UPDATE)
    @PostMapping("/edit")
    @ResponseBody
    public AjaxResult editSave(@Validated SysConfig config)
    {
        if (!configService.checkConfigKeyUnique(config))
        {
            return error("修改参数'" + config.getConfigName() + "'失败，参数键名已存在");
        }
        config.setUpdateBy(getLoginName());
        return toAjax(configService.updateConfig(config));
    }

    /**
     * 修改参数配置
```

```
20          *
21          * @param config 参数配置信息
22          * @return 结果
23          */
24         @Override
25         public int updateConfig(SysConfig config)
26         {
27             SysConfig temp = configMapper.selectConfigById(config.getConfigId());
28             if (!StringUtils.equals(temp.getConfigKey(), config.getConfigKey()))
29             {
30                 CacheUtils.remove(getCacheName(),
    getCacheKey(temp.getConfigKey()));
31             }
32
33             int row = configMapper.updateConfig(config);
34             if (row > 0)
35             {
36                 CacheUtils.put(getCacheName(),
    getCacheKey(config.getConfigKey()), config.getConfigValue());
37             }
38             return row;
39         }
```

同样的问题

Case13 con: 64

```
1          /**
2           * 任务调度状态修改
3           */
4          @Log(title = "定时任务", businessType = BusinessType.UPDATE)
5          @RequiresPermissions("monitor:job:changeStatus")
6          @PostMapping("/changeStatus")
7          @ResponseBody
8          public AjaxResult changeStatus(SysJob job) throws SchedulerException
9          {
10             SysJob newJob = jobService.selectJobById(job.getJobId());
11             newJob.setStatus(job.getStatus());
12             return toAjax(jobService.changeStatus(newJob));
13         }
14
15         /**
16          * 任务调度状态修改
```

```
17          *
18          * @param job 调度信息
19          */
20         @Override
21         @Transactional(rollbackFor = Exception.class)
22         public int changeStatus(SysJob job) throws SchedulerException
23         {
24             int rows = 0;
25             String status = job.getStatus();
26             if (ScheduleConstants.Status.NORMAL.getValue().equals(status))
27             {
28                 rows = resumeJob(job);
29             }
30             else if (ScheduleConstants.Status.PAUSE.getValue().equals(status))
31             {
32                 rows = pauseJob(job);
33             }
34             return rows;
35         }
36
37         /**
38          * 暂停任务
39          *
40          * @param job 调度信息
41          */
42         @Override
43         @Transactional(rollbackFor = Exception.class)
44         public int pauseJob(SysJob job) throws SchedulerException
45         {
46             Long jobId = job.getJobId();
47             String jobGroup = job.getJobGroup();
48             job.setStatus(ScheduleConstants.Status.PAUSE.getValue());
49             int rows = jobMapper.updateJob(job);
50             if (rows > 0)
51             {
52                 scheduler.pauseJob(ScheduleUtils.getJobKey(jobId, jobGroup));
53             }
54             return rows;
55         }
```

感觉也是存在问题的。

Case14 con: 65

```java
    /**
     * 任务调度状态修改
     */
    @Log(title = "定时任务", businessType = BusinessType.UPDATE)
    @RequiresPermissions("monitor:job:changeStatus")
    @PostMapping("/changeStatus")
    @ResponseBody
    public AjaxResult changeStatus(SysJob job) throws SchedulerException
    {
        SysJob newJob = jobService.selectJobById(job.getJobId());
        newJob.setStatus(job.getStatus());
        return toAjax(jobService.changeStatus(newJob));
    }

    /**
     * 任务调度状态修改
     *
     * @param job 调度信息
     */
    @Override
    @Transactional(rollbackFor = Exception.class)
    public int changeStatus(SysJob job) throws SchedulerException
    {
        int rows = 0;
        String status = job.getStatus();
        if (ScheduleConstants.Status.NORMAL.getValue().equals(status))
        {
            rows = resumeJob(job);
        }
        else if (ScheduleConstants.Status.PAUSE.getValue().equals(status))
        {
            rows = pauseJob(job);
        }
        return rows;
    }

    /**
     * 恢复任务
     *
     * @param job 调度信息
     */
    @Override
    @Transactional(rollbackFor = Exception.class)
    public int resumeJob(SysJob job) throws SchedulerException
    {
        Long jobId = job.getJobId();
        String jobGroup = job.getJobGroup();
```

```
48          job.setStatus(ScheduleConstants.Status.NORMAL.getValue());
49          int rows = jobMapper.updateJob(job);
50          if (rows > 0)
51          {
52              scheduler.resumeJob(ScheduleUtils.getJobKey(jobId, jobGroup));
53          }
54          return rows;
55      }
```

Case15 con: 71

```
1       /**
2        * 修改保存公告
3        */
4       @RequiresPermissions("system:notice:edit")
5       @Log(title = "通知公告", businessType = BusinessType.UPDATE)
6       @PostMapping("/edit")
7       @ResponseBody
8       public AjaxResult editSave(@Validated SysNotice notice)
9       {
10          notice.setUpdateBy(getLoginName());
11          return toAjax(noticeService.updateNotice(notice));
12      }
13
14      /**
15       * 修改公告
16       *
17       * @param notice 公告信息
18       * @return 结果
19       */
20      @Override
21      public int updateNotice(SysNotice notice)
22      {
23          return noticeMapper.updateNotice(notice);
24      }
```

Case16 con:138

```
    /**
     * 批量取消授权
     */
    @RequiresPermissions("system:role:edit")
    @Log(title = "角色管理", businessType = BusinessType.GRANT)
    @PostMapping("/authUser/cancelAll")
    @ResponseBody
    public AjaxResult cancelAuthUserAll(Long roleId, String userIds)
    {
        return toAjax(roleService.deleteAuthUsers(roleId, userIds));
    }

    /**
     * 批量取消授权用户角色
     *
     * @param roleId 角色ID
     * @param userIds 需要删除的用户数据ID
     * @return 结果
     */
    @Override
    public int deleteAuthUsers(Long roleId, String userIds)
    {
        return userRoleMapper.deleteUserRoleInfos(roleId,
Convert.toLongArray(userIds));
    }
```

Case17 con:144

```
    /**
     * 修改保存字典类型
     */
    @Log(title = "字典类型", businessType = BusinessType.UPDATE)
    @RequiresPermissions("system:dict:edit")
    @PostMapping("/edit")
    @ResponseBody
    public AjaxResult editSave(@Validated SysDictType dict)
    {
        if (!dictTypeService.checkDictTypeUnique(dict))
        {
            return error("修改字典'" + dict.getDictName() + "'失败，字典类型已存
在");
        }
        dict.setUpdateBy(getLoginName());
```

```
        return toAjax(dictTypeService.updateDictType(dict));
    }

    /**
     * 修改保存字典类型信息
     *
     * @param dict 字典类型信息
     * @return 结果
     */
    @Override
    @Transactional
    public int updateDictType(SysDictType dict)
    {
        SysDictType oldDict = dictTypeMapper.selectDictTypeById(dict.getDictId());
        dictDataMapper.updateDictDataType(oldDict.getDictType(), dict.getDictType());
        int row = dictTypeMapper.updateDictType(dict);
        if (row > 0)
        {
            List<SysDictData> dictDatas = dictDataMapper.selectDictDataByType(dict.getDictType());
            DictUtils.setDictCache(dict.getDictType(), dictDatas);
        }
        return row;
    }
```

Case18 con:144

```
/**
     * 修改保存代码生成业务
     */
    @RequiresPermissions("tool:gen:edit")
    @Log(title = "代码生成", businessType = BusinessType.UPDATE)
    @PostMapping("/edit")
    @ResponseBody
    public AjaxResult editSave(@Validated GenTable genTable)
    {
        genTableService.validateEdit(genTable);
        genTableService.updateGenTable(genTable);
        return AjaxResult.success();
```

```
    }

    /**
     * 修改业务
     *
     * @param genTable 业务信息
     * @return 结果
     */
    @Override
    @Transactional
    public void updateGenTable(GenTable genTable)
    {
        String options = JSON.toJSONString(genTable.getParams());
        genTable.setOptions(options);
        int row = genTableMapper.updateGenTable(genTable);
        if (row > 0)
        {
            for (GenTableColumn genTableColumn : genTable.getColumns())
            {
                genTableColumnMapper.updateGenTableColumn(genTableColumn);
            }
        }
    }
```