

# 开源之夏项目计划书

## 项目概述

MCSManager 是一个用于管理 Minecraft 和大部分 Steam 游戏服务器的 Web 控制面板。

当前 MCSManager 对 Docker 的支持还未完全可靠，并且未支持 Docker for Windows。同时，文件上传不能断点续传，对大文件上传支持不友好。本项目将解决这些问题，同时修复一些其他问题。

## 项目分析

### Docker

当前 MCSManager 对 Docker 的支持用的是 Dockerode 库，对于 stdio 的接管用的是

`Dockerode.Container#attach`：

daemon/src/service/docker\_process\_service.ts@4d9b83e

typescript

```
226     const stream = await container.attach({
227       stream: true,
228       stdout: true,
229       stderr: true,
230       stdin: true
231     });
```

daemon/src/service/docker\_process\_service.ts@4d9b83e

typescript

```
268     this.stream = await this.container.attach({
269       stream: true,
270       stdout: true,
271       stderr: true,
272
273
```

```
    stdin: true
  });
```

这样只能接管未被接管的 stdio。另外，对于 Docker for Windows，这样的方案完全不能接管。

根据 [Docker API Docs](#)，可以使用 hijacking 的方式强制接管，根据 Dockerode 的实现，只需要把 attach 方法的调用改成这样即可：

```
container.attach({
  stream: true,
  stdout: true,
  stderr: true,
  stdin: true,
  hijack: true
});
```

ts

### 需要注意的是

When the TTY setting is enabled in POST /containers/create, the stream is not multiplexed. The data exchanged over the hijacked connection is simply the raw data from the process PTY and client's stdin.

正如 Docker API Docs 所说的那样，如果容器创建时设置了 TTY，那么这个流就不是多路复用的。如果有另一个程序也劫持了这个流，那么 MCSManager 就无法接管了。

MCSManager 创建容器时并没有设置 TTY。

在 MCSManager 启动时，检查 Docker 中是否有正在运行的、MCSManager 的容器，若有则接管其 stdio，就可以实现重启后自动接管。

另外，这样的操作也可以完成对 Docker for Windows 的支持。

另外一处需要修改的是：

```
daemon/src/service/docker_service.ts@4d9b83e
```

typescript

```
5   public static readonly defaultConfig: Docker.DockerOptions = {
6     socketPath: process.env.DOCKER_HOST ?? '/var/run/docker.sock'
7   };
```

Windows 下 Docker 的 socket 路径是 `///./pipe/docker_engine`。

相关 Issue:

- [#1071](#)
- [#1072](#)
- [#1460](#)

## 文件上传

允许选择多个文件/文件夹，在选择文件多于 10 个时，提示用户上传压缩包。

文件逐个上传。对于每个文件，记录哈希并切分成 1MiB 的块；上传时监听 onUploadProgress 事件，在进度条上实时反馈当前正在上传的文件和进度、总文件数量和已上传文件数量。一个文件上传完成后气泡通知。

文件名（当前文件序号/总文件数）

txt

-----（进度条） 已上传体积 / 总体积

file.txt (2/5)

txt

----- 1.5MiB / 5MiB

在 daemon 端，收到上传请求（来自 panel）时先检查缓存中是否有上传了一半的文件并校验哈希，若确认为续传则同时返回已上传的大小。收到的文件片段存储在临时目录中，文件上传完成后，移动到目标目录。临时目录文件有过期时间，若超过过期时间没有上传的片段则删除。

## SSL

支持配置 Panel 和 Daemon 的 SSL 证书，通过配置文件设置是否启用、证书路径、私钥路径。

如需开启，在 http.createServer 时传入包含 `key` 和 `cert` 的 option 即可。Daemon 端的 Socket.io 挂载在 http 服务器上，不需要额外处理。

## 其他问题

- [#1601](#) 在缺少运行库报错时提示库下载地址
- [#1606](#) rest api 查询实例支持 tag 过滤

- [#1305](#) 实例延时自启动
- 

## 项目规划

### 前半阶段 (7.1-8.15)

- Docker
  - Docker for Windows 支持
  - Docker 重新接管
  - 调整 Docker 连接逻辑，自动接管启动了的容器
- 文件上传
  - 文件切片上传
  - 支持断点续传
  - 文件上传进度条
  - 文件上传完成后气泡通知
  - 上传过程中阻止关闭窗口

### 后半阶段 (8.16-9.30)

- SSL
  - 支持配置 SSL 证书
  - 支持配置是否启用 SSL
- [#1601](#) 在缺少运行库报错时提示库下载地址
- [#1606](#) rest api 查询实例支持 tag 过滤
- [#1305](#) 实例延时自启动
- 解决在中期验收阶段中发现的问题
- 测试更多场景下第一阶段实现功能的可靠性
- 更新文档
- 思考可以改进和补充的地方