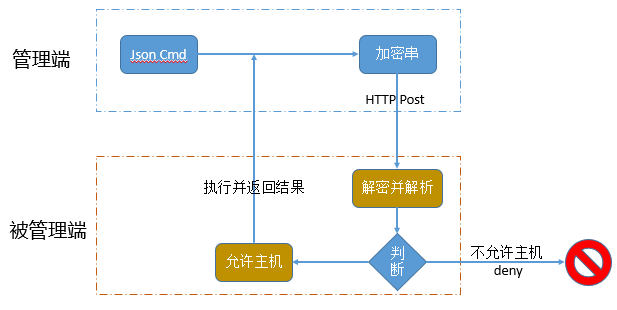
## 一、流程图



## 二、打包方法

**直接打包**：go build agent.go，打包完成后，会生成agent文件，在windows下会生成agent.exe文件。

**优化参数**：go build -ldflags "-s -w" agent.go ，建议加上这两个参数，会使生成的文件变小

**Windows下的特殊参数**：go build -ldflags "-s -w -H windowsgui" 启用windowsgui参数后，可以直接双击运行程序，不再出现dos窗口，但是启用windowsgui参数后，在DOS窗口就查看不到命令行参数了，只能直接运行，后面无法加参数。所以在windows下，参数可以直接编译进去，如遇到需要调试的，再使用不加windowsgui参数编译出的版本运行。

## 三、兼容性

目前在linux下的版本基本都是x64位的架构，所以兼容性更强些，基本在一个主流版本上编译出的，在其他版本上也可以运行。如遇到特别老的OS版本时，可以通过在相应的版本上安装golang程序及相关依赖包后，再编译一份即可。

Windows平台上很x86和x64两种OS版本，一些比较老的服务器版本，比如windows2003，还是存在很多32位的x86 OS的，可以在相应的版本上编译下，基本所有的32位系统，像win7 32 、win10 32位也都是通用的。

## 四、并发性能

在对比多种架构和压测结果来看，当前使用的代码最简单最高效，逻辑处理性能最好，而且便于源头控制。而底层基于协程，所以一台被控主机理论上可以支持到上万个任务同时执行（只要被控主机本身不存在性能瓶颈）。