PC软件 电脑技巧

# RustDesk通过API防止服务器被滥用 自动编译将服务器等信息内置客户端

2.04 k 25

## 资源下载

点击下载

**温馨提示**

点击下载按钮右侧下拉箭头，可以选择软件资源下载！用夸克移动端保存该资源，可以免费获取1TB存储空间！

## 引言

之前，鼠标出过rustdesk的搭建、编译等教程。近期，有同学反应，之前的教程，已经失效了！现在，Github无法通过设置变量，将服务器等信息编译进客户端！这可能是作者更新了源码，或者Github升级的缘故！同时，也有同学提出新诉求，问可以不可通过API，控制客户端的使用，防止服务器被滥用！今天，鼠标就一次性解决以上所有问题，话不多说，开始教程！

## 服务端及API安装

若想通过API，控制RustDesk客户端使用，我们用到Github大佬lejianwen修改的服务端源码，以及该大佬开源的API项目。

### SSH远程连接服务器

教程开始之前，我们需要准备一台云服务器，并用WindTerm SSH连接服务器。之前的视频，鼠标有详细讲过，还是不会的同学，可以看这期视频，这里就直接看我演示吧！

## 安装1Panel管理面板

接下来安装rustdesk服务端的时候，会用到docker compose，鼠标依旧选择1Panel，免费开源，界面简洁易用。同样，之前的视频也有详细说过，这里就不赘述了，还是不会的同学，翻翻这期视频，这里还是看我操作吧！

## 1Panel安装脚本：

```Bash
#RedHat / CentOS:
curl -sSL https://resource.fit2cloud.com/1panel/package/quick_start.sh -o quic
#Ubuntu:
curl -sSL https://resource.fit2cloud.com/1panel/package/quick_start.sh -o quic
#Debian:
curl -sSL https://resource.fit2cloud.com/1panel/package/quick_start.sh -o quic
```

安装过程注意：

- 你可以自定义面板访问端口，例：6666
- 也可以自定义用户名密码

## 服务器后台开放端口

1Panel安装好后，按住Ctrl点击这个地址，即可访问1Panel面板。多数情况，不能正常访问，我们还需要在服务器后台，开放该端口。同时，我们将rustdesk服务端及API用到的端口，也一并开启。这类教程，鼠标也讲了很多次，不会的，看这期视频，这里以阿里云为例，进行操作演示。

开放端口列表：

```
TCP: 21114-21119,6666(1Panel自定义端口)
UDP: 21116
```

## Compose一键部署RustDesk服务端及API

访问1Panel面板，左侧导航点击容器，编排-创建编排，名称随意，便于区分就好，粘贴如下代码：

```yaml
networks:
  rustdesk-net:
    external: false
services:
  rustdesk:
    ports:
      - 21114:21114
      - 21115:21115
      - 21116:21116
      - 21116:21116/udp
      - 21117:21117
      - 21118:21118
      - 21119:21119
    image: lejianwen/rustdesk-server-s6:latest
    environment:
      - RELAY=<relay_server[:port]>
      - ENCRYPTED_ONLY=1
      - MUST_LOGIN=Y
      - TZ=Asia/Shanghai
      - RUSTDESK_API_RUSTDESK_ID_SERVER=<id_server[:21116]>
      - RUSTDESK_API_RUSTDESK_RELAY_SERVER=<relay_server[:21117]>
      - RUSTDESK_API_RUSTDESK_API_SERVER=http://<api_server[:21114]>
      - RUSTDESK_API_KEY_FILE=/data/id_ed25519.pub
      - RUSTDESK_API_JWT_KEY=xxxxxx #
    volumes:
      - /data/rustdesk/server:/data
      - /data/rustdesk/api:/app/data #将数据库挂载
    networks:
      - rustdesk-net
    restart: unless-stopped
```

参数提示：

```Bash
RELAY=<relay_server[:port]> #中继服务器:21117
MUST_LOGIN=N #默认为N，设置为Y 则必须登录才能链接
RUSTDESK_API_RUSTDESK_ID_SERVER=<id_server[:21116]> #ID服务器
RUSTDESK_API_RUSTDESK_RELAY_SERVER=<relay_server[:21117]> #中继服务器
RUSTDESK_API_RUSTDESK_API_SERVER=http://<api_server[:21114]> #API服务器
RUSTDESK_API_JWT_KEY=xxxxxx #随便设置个字符串
/data/rustdesk/server #查看 key
```

点击确定，等待拉取docker镜像，出现这个提示，意味着rustdesk服务端及API部署完成。一个compose文件，将rustdesk服务端，以及API都安装好了，服务端是作者基于原版修改，目的是更兼容作者的第三方API，同时也添加了一些功能，譬如，只有登录客户端，才能发起远程连接，这就有效杜绝客户端被滥用的情况。

- 通过docker挂载的路径，可以查看客户端所用的key。
- API登录地址为服务器IP:21114,用户名默认是admin，密码可以通过刚部署的容器里日志查看，登录API后台后，切记及时修改密码。

```Bash
./apimain reset-admin-pwd <你的新密码>
```

### 测试部署的服务端及API

将服务器、API，以及Key等信息填入rustdesk客户端，进行远程连接测试，不出意外的情况，一切都正常，确定不登录客户端的情况，无法发起远程协助。

## :: 修改RustDesk源码

正如视频开头所说，之前的教程中讲到，利用Github的环境变量，将服务器等信息，编译到客户端，现在已经失效，无法将服务器等信息编译到rustdesk客户端。上一步，我们已经部署好了rustdesk的服务端及API，接下来，鼠标将通过修改源码的方式，将服务器等信息，保存到客户端源码，并用Github的Actions自动编译客户端。

### 修改源码准备工作

修改rustdesk源码，我们用到Git，用来克隆源码到本地修改，并且推送修改后的源码。因为rustdesk源码包含子模块，鼠标不会直接在Github上修改，所以只能采取git的方式。还有就是代码编辑器，本来工具想用Notepad++，奈何其作者是个垃圾，算了，使用更好用更美观的HBuilder X吧！最后一个，就是解决github的访问问题，推荐使用dev-sidecar。

## dev-sidecar安装及使用

鼠标博客下载并安装dev-sidecar，管理员运行dev-sidecar，根据引导安装证书，证书安装好后，首页开启代理服务、系统代理，以及Git代理。此时，再次打开github，我们就可以正常访问使用github了。还有一点需要注意，就是这段提示内容，忽略可能会引起电脑无法上网。

## git下载安装

鼠标博客或者软件官网下载git，全程选项保持默认下一步即可。安装好后，通过终端输入git命令，测试安装是否成功。

## HBuilder X下载安装

鼠标博客或者软件官网下载HBuilder X，该软件为绿色免安装版本，解压即用。

## Fork rustdesk项目

Github注册登录账号，Fork rustdesk及子模块项目到自己的账号下。rustdesk项目代码页，点击libs后再点击这个hbb，即可跳转到该子模块项目，直接fork到自己账号下即可。考虑到安全性的问题，fork的仓库无法设置为私有，你也可以自建私有仓库，分别导入rustdesk及子模块源码，这样可以有效防止服务器等信息泄露。

## 配置github SSHkey

想要正常克隆推送 github源码，我们需要配置github的sshkey，以此对接git和github。任务栏搜索框，搜索git bash并运行。在git bash代码框，依次输入如下代码并执行：

```bash
#配置 Git 的用户名和邮箱
git config --global user.name "用户名"
git config --global user.email "邮箱"

#生成密钥（一直回车直至这个界面）
ssh-keygen -t rsa -C "邮箱"

#查看公钥
cat ~/.ssh/id_rsa.pub
```

打开github个人中心-设置，左侧导航点击SSH and GPG keys，再点击New SSH key，标题随意，将git bash窗口公钥复制到key这里，最后点击 这里添加保存。

打开Windows终端或者git bash，运行如下代码，测试是否可以成功链接gtihub

```bash
#测试连接github
ssh -T git@github.com
```

终端询问是否继续连接，输入Yes回车确认，出现这个提示，就意味着我们通过SSH方式，可以正常连接gtihub。

## 通过Git克隆RuestDesk源码到本地

本地任意目录，运行Windows终端或者git bash，复制我们fork的rustdesk项目SSH地址，终端输入如下命令+SSH地址，将RustDesk源码仓库克隆到本地。

```bash
git clone --recurse-submodules <SSH地址> #--recurse-submodules参数，递归克隆子模块
```

## 修改子模块路径

输入如下命令，进入主仓库目

```Bash
cd RestDesk
```

修改主目录下.gitmodules文件，将原作者的子模块 URL 替换为我们 Fork的 地址，运行HBuilder X，将文件直接拖放到HBuilder X主窗口，修改替换如下内容：

```R
[submodule "libs/hbb_common"]
        path = libs/hbb_common
        url = <替换为我们fork的hbb_common项目地址>
```

同步子模块地址

```Bash
git submodule sync
```

提交修改后的代码，并推送到主仓库：

```SQL
git add .gitmodules
git commit -m "更新所有子模块地址至我的Fork（备注内容）"
git push origin <主仓库分支名，默认master>
```

## 修改ID/中继服务器

进入子模块文件夹

```Bash
cd libs/hbb_common
```

配置子模块并创建新分支

```Bash
git checkout -b <新建分支名>
```

修改子模块源码

```Bash
libs\hbb_common\src\config.rs #修改该文件的101-102行，分别是服务器及KEY
```

提交修改后的代码，并推送到子模块仓库：

```Bash
git add .
git commit -m "改为自己的ID/中继服务器"
git push origin <新建的分支名>  # 推送分支到我们子模块Fork
```

返回主仓库主目录：

```Bash
cd ../../..
```

更新子模块的 Commit ID：

```Bash
git add libs/hbb_common  # 提交子模块的新 Commit
git commit -m "更新修改后的子模块"
git push origin <主仓库分支名>
```

## 修改替换API地址

仓库主目录，修改如下文件:

```Bash
src\common.rs #947行替换API地址
```

提交修改并推送到主仓库：

```Bash
git add .
git commit -m "修改替换API地址"
git push origin <主仓库分支名，默认master>
```

### 删除客户端广告提示

仓库主目录，修改如下文件:

```R
flutter\lib\desktop\pages\connection_page.dart

#81-110行代码替换如下:
    Widget setupServerWidget() => Flexible(
       child: Offstage(
         offstage: !(!_svcStopped.value &&
             stateGlobal.svcStatus.value == SvcStatus.ready &&
             _svcIsUsingPublicServer.value),
         child: Row(
           crossAxisAlignment: CrossAxisAlignment.center,
           children: [],
         ),
       ),
     );
```

提交修改并推送到主仓库:

```Bash
git add .
git commit -m "修改删除客户端广告"
git push origin <主仓库分支名，默认master>
```

## Github编译客户端

通过以上操作，我们将服务器及API信息，写入到rustdesk的源码里。通过github的Actions进行编译 ，之前的视频鼠标有讲过，这里再简单演示下，不会的同学，可以翻看早期的视频，或者评论区留言。配置好自动编译后，接下来就是耐心等待编译。大概过了一个小时左右，Windows的客户

端编译好了，我们就可以下载使用了，客户端没有了那串广告提示，没有登录的情况下，是无法发起远程连接，说明API用户控制已经生效，登录用户后可以正常发起远程。

## 总结

以上，就是本期视频的全部内容，鼠标讲了rustdesk的服务端及API一键部署，同时通过修改代码，将服务器等信息写入到客户端。新版服务端，通过girhub大佬的修改，可以实现通过API用户，来控制客户端发起远程链接，这样有效防止自己的服务器被滥用。视频中涉及的软件工具，以及脚本代码等，都可以在鼠标博客免费获取！本期视频就到这里，谢谢您的观看！如果感觉有用，不妨关注下，点赞收藏一波，也不是不可以呀！

RustDesk

## 相关文章