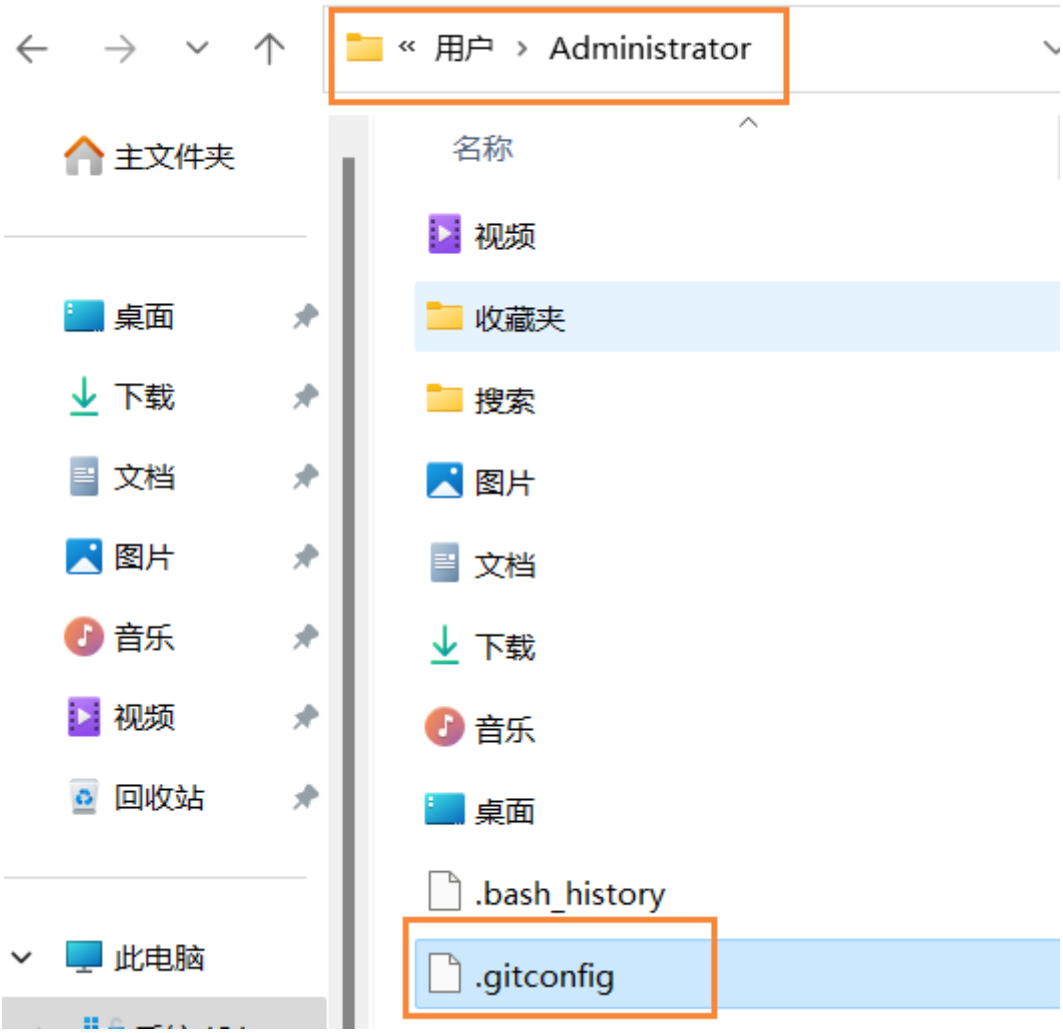
GitHub使用帮助

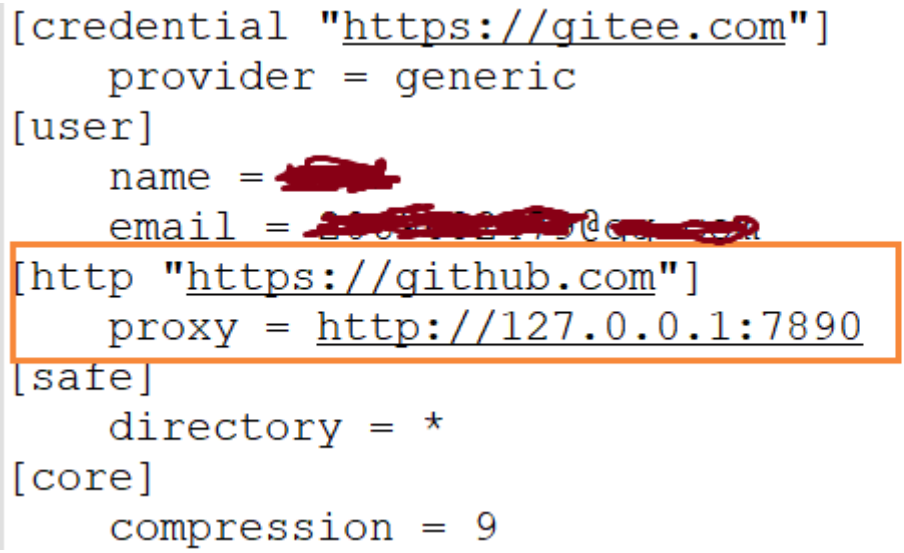
1 配置代理

由于网络原因，有的时候使用`https`的方式克隆、获取、拉取、推送代码会出现连接失败的问题，此时可以通过设置代理服务器的方式来解决问题。

使用代理的时候需要配置你的git，找到你的`.gitconfig`配置文件，一般在当前用户的根目录，如下图所示：



配置示例如下：




```
[http "https://github.com"]
    proxy = http://127.0.0.1:7890
```

将http://127.0.0.1:7890替换成你自己的代理服务器和端口

2 配置SSH

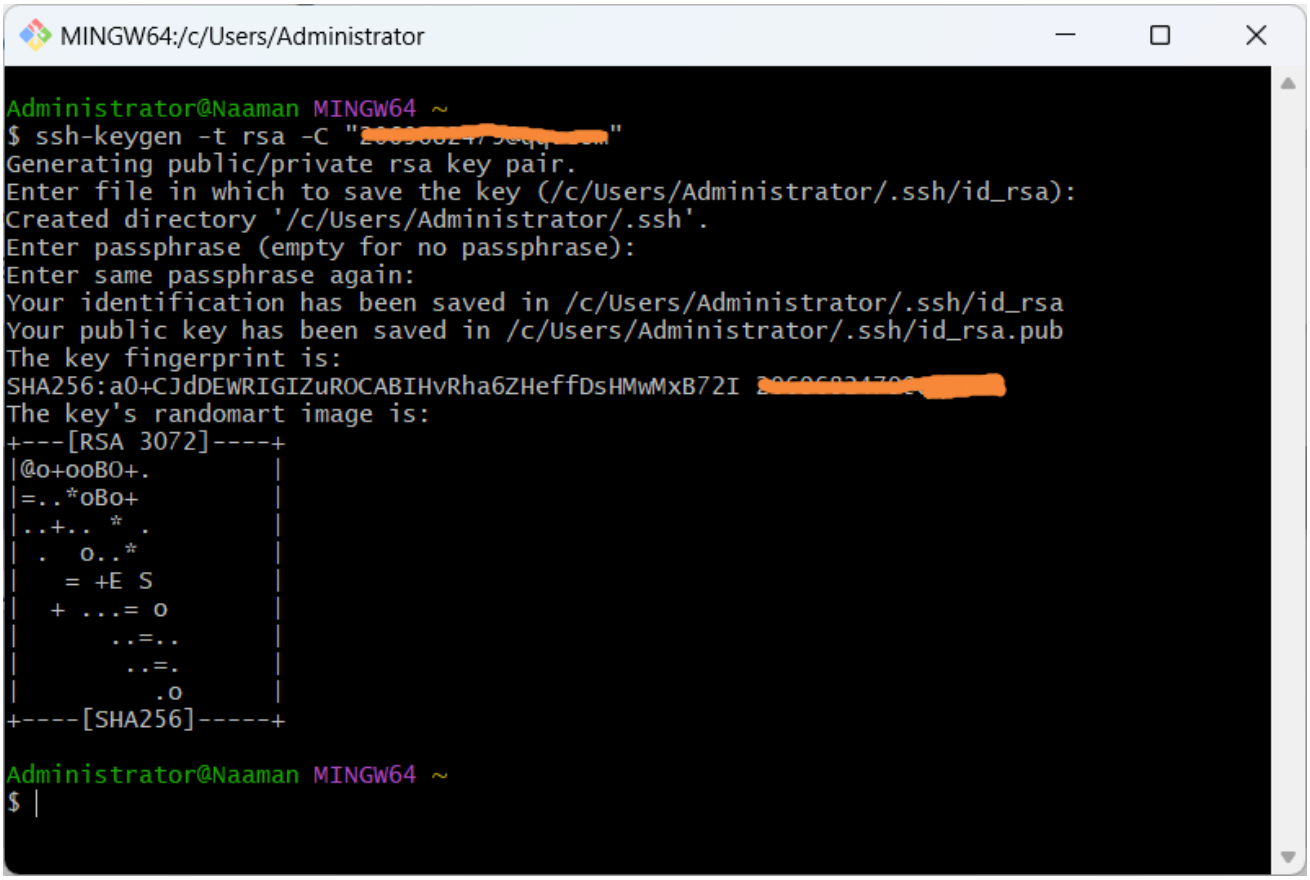
有时候通过SSH方式也能解决`GitHub`访问慢的问题。

2.1 生成秘钥对

打开Git bash工具，然后输入命令

```
ssh-keygen -t rsa -C "github邮箱"
```

遇到输入提示直接回车即可如下图所示



2.2 添加私钥到秘钥管理器

ssh-agent就是一个密钥管理器，运行ssh-agent以后，使用ssh-add将私钥交给ssh-agent保管，其他程序需要身份验证的时候可以将验证申请交给ssh-agent来完成整个认证过程。

确认一下秘钥管理器是否正常运行，执行`ssh-agent`命令，如下图所示。

```
Administrator@Naaman MINGW64 ~
$ ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-37i9fhpQZAKY/agent.1972; export SSH_AUTH_SOCK;
SSH_AGENT_PID=1973; export SSH_AGENT_PID;
echo Agent pid 1973;
```

添加私钥到秘钥管理器，执行下列命令

```
ssh-add ~/.ssh/id_rsa
```

执行结果如下图所示，表示出现了错误

```
Administrator@Naaman MINGW64 ~
$ ssh-add ~/.ssh/id_rsa
Could not open a connection to your authentication agent.
```

通过下面的方式解决

- 首先查询进程

```
ps aux | grep ssh
```

- 然后杀死进程

```
kill -9 进程号
```

- 进入用户名目录下的.ssh目录，执行如下命令

```
cd ~/.ssh
exec ssh-agent bash
eval ssh-agent -s
```

```
Administrator@Naaman MINGW64 ~
$ ps aux | grep ssh
    1973      1    1973      13784  ?         197108 15:48:39 /usr/bin/ssh-agent

Administrator@Naaman MINGW64 ~
$ kill -9 1973

Administrator@Naaman MINGW64 ~
$ cd ~/.ssh

Administrator@Naaman MINGW64 ~/.ssh
$ exec ssh-agent bash
Administrator@Naaman MINGW64 ~/.ssh
$ eval ssh-agent -s
SSH_AUTH_SOCK=/tmp/ssh-K7mYXG14ZAJY/agent.2010; export SSH_AUTH_SOCK;
SSH_AGENT_PID=2011; export SSH_AGENT_PID;
echo Agent pid 2011;
```

- 再次执行添加命令

```
ssh-add ~/.ssh/id_rsa
```

执行结果如下图所示表示添加成功

```
Administrator@Naaman MINGW64 ~/.ssh
$ ssh-add ~/.ssh/id_rsa
Identity added: /c/Users/Administrator/.ssh/id_rsa ([redacted])
```

2.3 配置GitHub

查看公钥

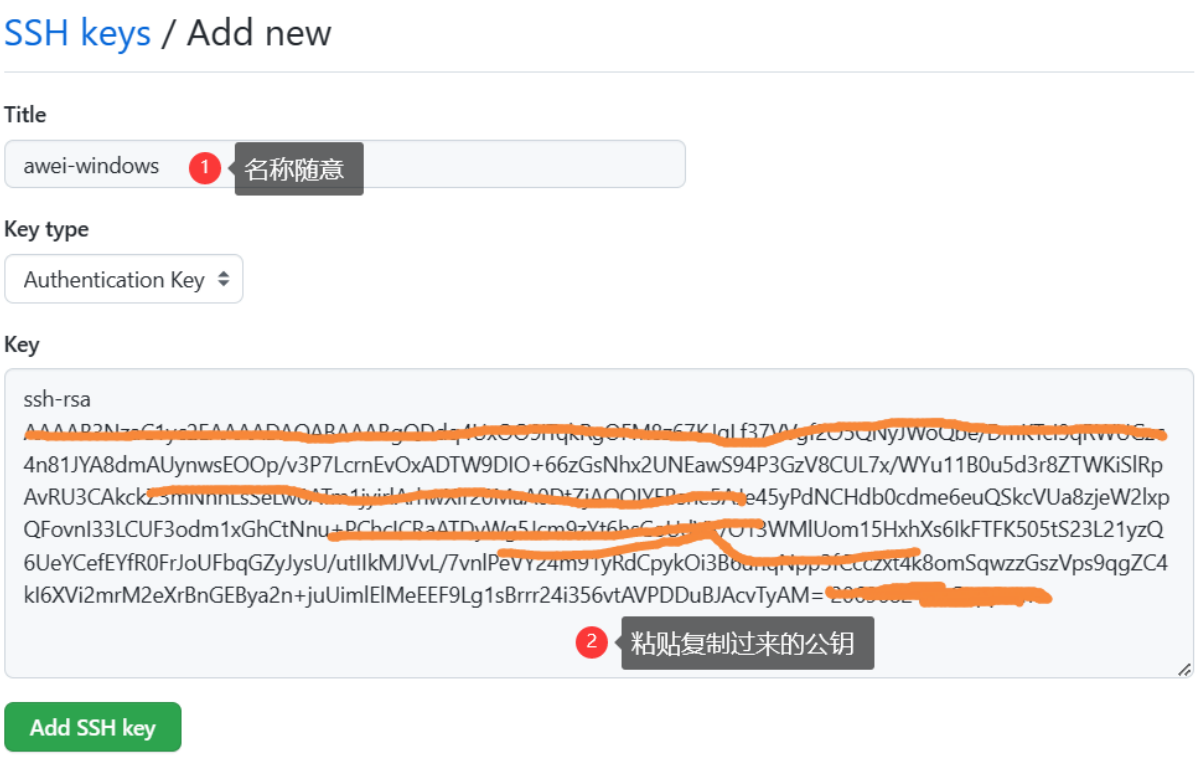
```
cat ~/.ssh/id_rsa.pub
```

```
Administrator@Naaman MINGW64 ~/.ssh
$ cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDdq4UxOO9iTqkRgOFM8z67KJgLf37VVgf2O5QNyJWoO
KTci9qRWUGzc4n81JYA8dmAUynwsEOOp/v3P7LcrnEvOxADTW9DIO+66zGsNhx2UNEawS94P3GzV8CUL7
11BOu5d3r8ZTWKiSlRpAvRU3CAkckZ3mNnnLsSeLwOATm1jyirlArhwXir2OMaAODtZjAQOIYFBsne5Aj
dNCHdbOcdme6euQSkcVUa8zjeW2lxpQFovnI33LCUF3odm1xGhCtNnu+PChcICRaATDyWg5Jcm9zYt6hs
X/O13WMlUom15HxhXs6IkFTFK5O5tS23L21yzQ6UeYCefEYfROFrJoUFbqGZyJysU/utIIkMJVvL/7vnl
4m91yRdCpykOi3B6unqNpp3fCcczxt4k8omSqwzzGszVps9qgZC4kI6XVi2mrM2eXrBnGEBya2n+juUin
```

复制公钥备用。

打开GitHub SSH Key新建链接：https://github.com/settings/ssh/new

然后在执行下图所示的操作

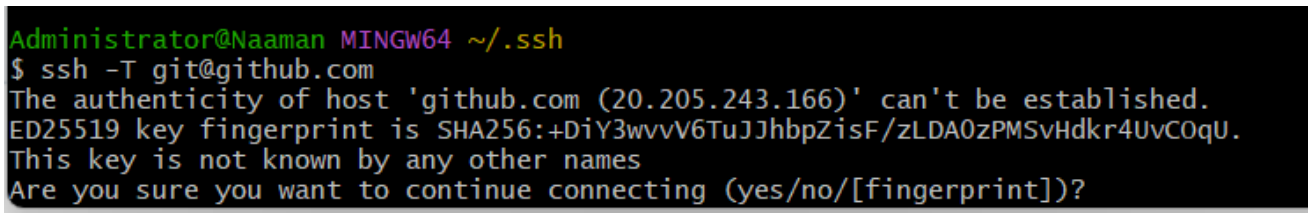


2.4 连接测试

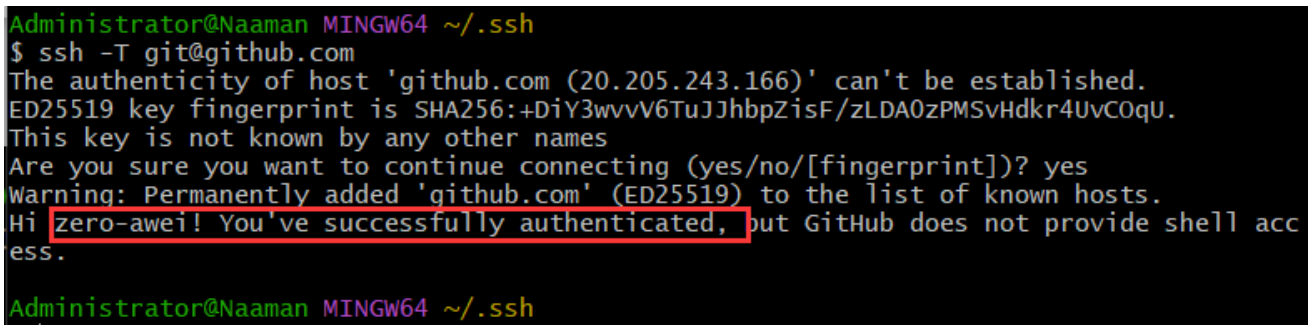
执行命令

```
ssh -T git@github.com
```

执行结果如如下图所示



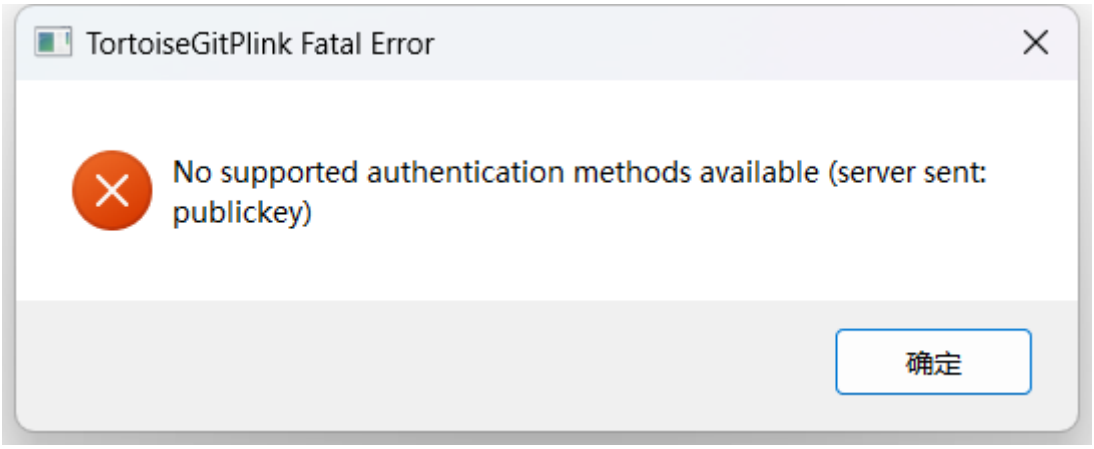
输入yes，执行结果如下图所示



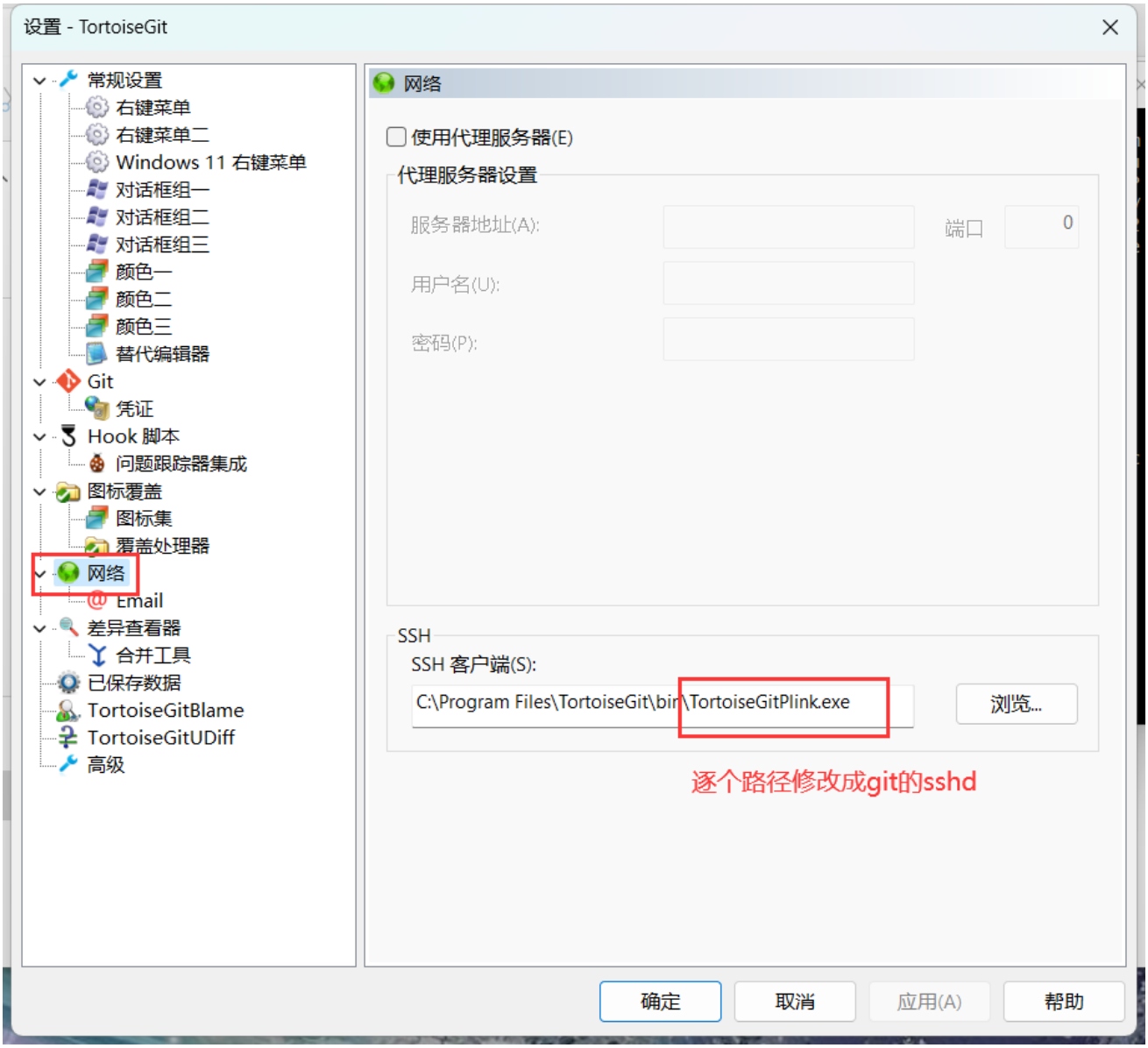
到此说明配置成功了，你就可以用ssh方式克隆和推送代码到`GitHub`了

提示：

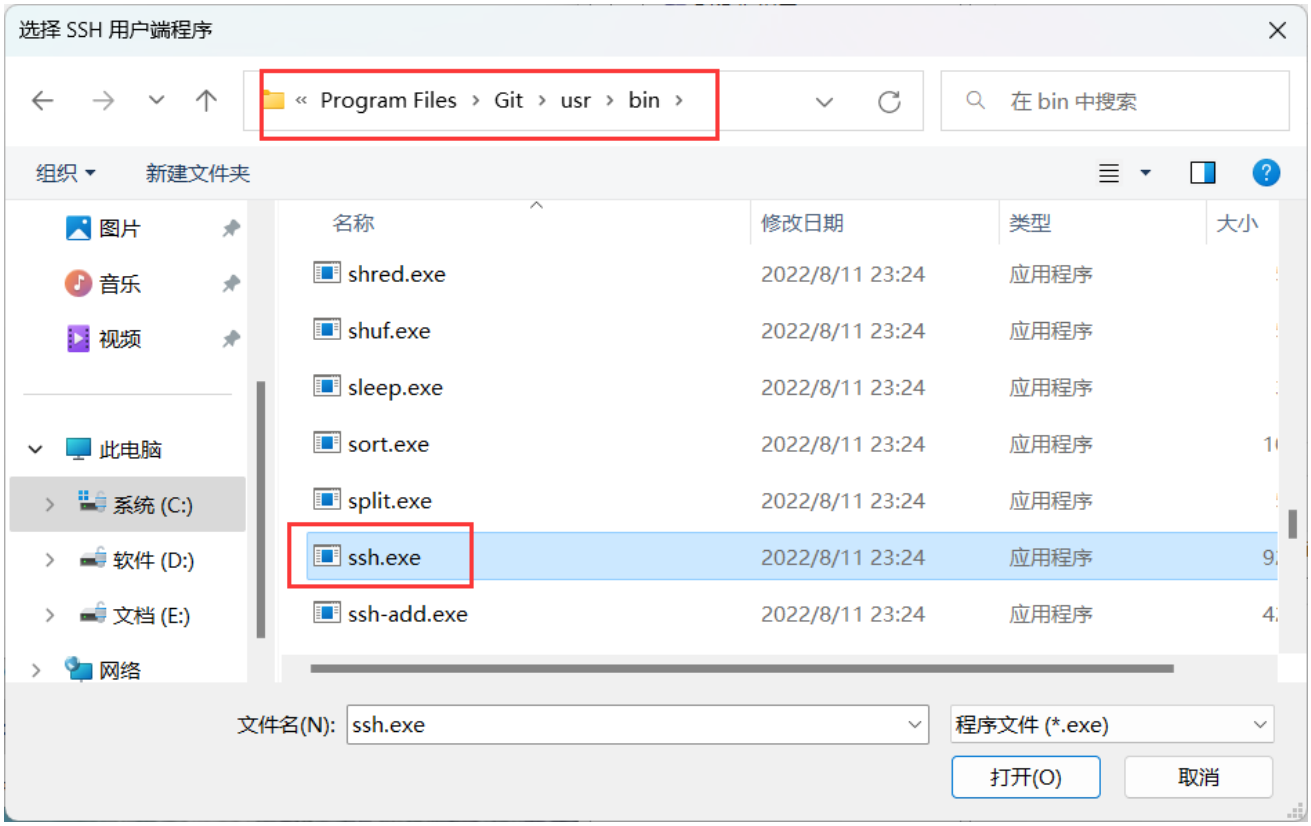
如果我们用乌龟clone项目的时候出现下图的错误提示，我们需要做以下处理。



我们需要修改`TortoiseGit`网络的SSH客户端为 git 服务器的`ssh.exe`



点击浏览，示例路径如下



再次尝试即可正常clone

2.5 扩展

如果需要配置多个凭证，可以在.ssh目录下面建立一个`config`文件书写配置来对应

下面是一个配置示例：

```
#github
Host github.com
HostName github.com
PreferredAuthentications publickey
IdentityFile ~/.ssh/id_rsa
User zero-awei
```

配置文件说明：

每个账号单独配置一个`Host`，`HostName`和`IdentityFile`三个属性即可。

- `Host`：别名，可以取为自己喜欢的名字，不过这个会影响git相关命令，例如：
  - `Host mygithub`这样定义的话，命令如下，即`git@`后面紧跟的名字改为`mygithub`，相应的`clone`命令为：
  - ```
    git clone git@mygithub:xxxx/xxxxx.git
    ```
- `HostName`：映射`Host`对应的真正的域名，如：`github.com`
- `IdentityFile`：配置私钥文件所在位置，如：`~/.ssh/id_rsa`
- `PreferredAuthentications`：配置登录时用什么权限认证
  - 可设置为：`publickey,password publickey,keyboard-interactive`等
    ```

- `User`：配置使用用户名