

部署文档

- Docusign测试服务器配置
 - 虚拟机创建
 - Docusign服务器所需服务安装
 - Docker 安装
 - 更新系统
 - 安装 Docker
 - 启动 Docker
 - Nginx 安装
 - 创建nginx.conf文件
 - 证书准备
 - 打包镜像
 - 创建Dockerfile
 - 运行Dockerfile
 - 中间件安装
 - 发布程序
 - 部署文件上传
 - 构建镜像
 - Postgres 安装
 - Docker Compose 配置
 - 启动程序

Docusign测试服务器配置

测试环境配置：

名称	规格
CPU	2核
内存	4G

硬盘	SSD 30G
静态IP	163.228.234.38

正式环境推荐配置：

名称	规格
CPU	2核
内存	4G
硬盘	SSD 500G
静态IP	是

虚拟机创建

1) 主页点击虚拟机



2) 点击创建



- 选择资源组，这里选择创建好的资源组，测试环境选择了docusign-test
- 配置虚拟机名称
- 选择镜像，这里选择了ubuntu20.04
- 选择虚拟机配置，测试环境选择了2H 4G
- 选择验证方式，测试环境选择了用户名密码

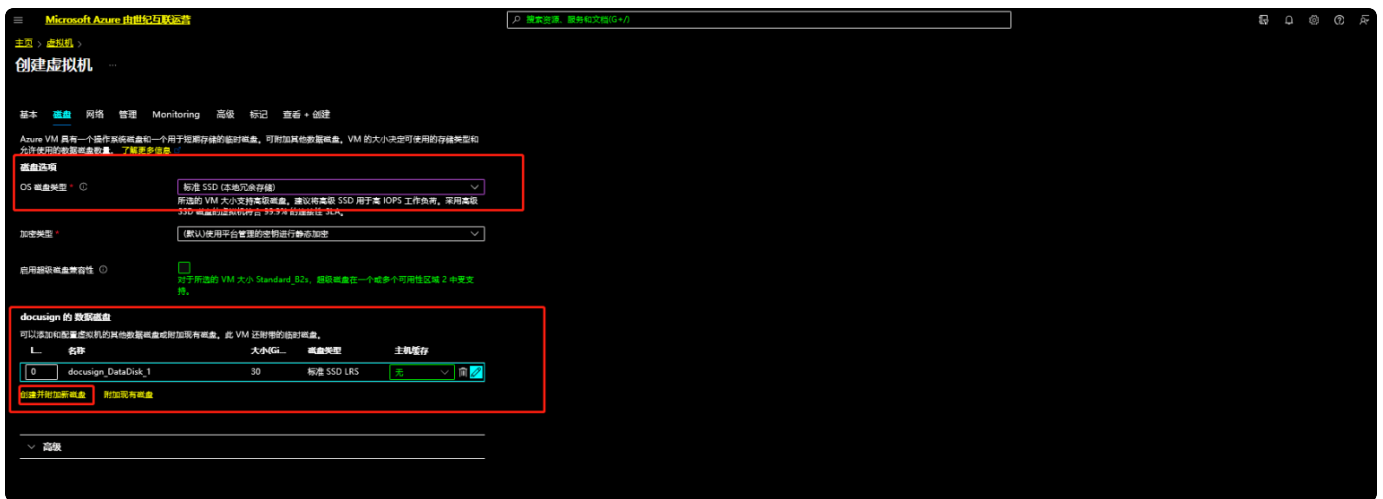
然后点击下一步配置磁盘



4) 选择磁盘类型是标准ssd

创建一个自定义的30G新标注ssd磁盘

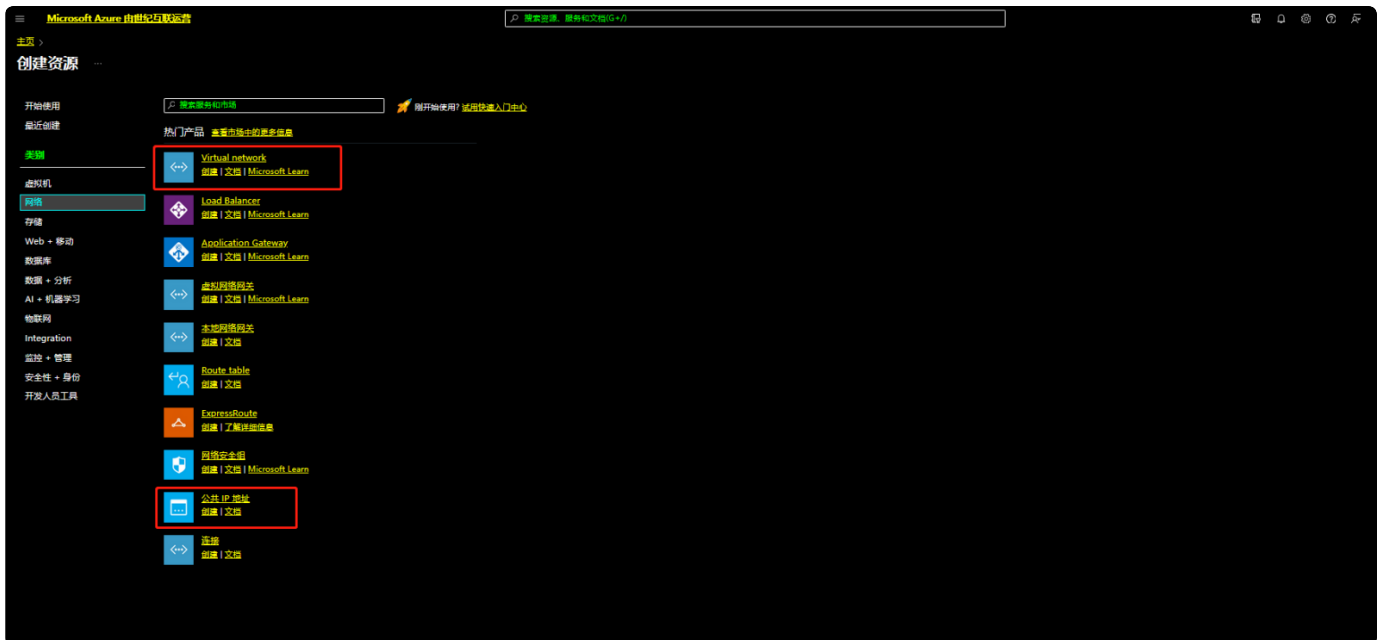
点击下一步网络



5) 在配置网络之前我们需要购买一个静态ip

回到主页点击创建资源

创建公共IP和子网



选择资源组

选择创建名字

选择SKU基本

下一步Tags

没有配置继续下一步

然后创建

Microsoft Azure 由世纪互联运营

创建公共 IP 地址

Basics Tags Review + create

resources through a public address. [了解详细信息](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

订阅: docuSign-TestSign-Middleware-NonProd

资源组: docuSign-test

Instance details

Region: (Asia Pacific) 中国北部 3

Configuration details

Name: xxxxxx

IP 版本: IPv4

SKU: 基本

IP 地址分配: 静态

空闲超时(分钟): 4

DNS 名称标签:

< 上一步 下一步: Tags 审核 + 创建

创建子网

选择资源组

选择名称

Microsoft Azure 由世纪互联运营

创建虚拟网络

基本 IP 地址 安全性 标记 查看 + 创建

Azure 虚拟网络(VNet)是 Azure 中专用网络的基本构建块。VNet 允许许多类型的 Azure 资源(如 Azure 虚拟机)安全地与来自 Internet 和本地网络进行通信。VNet 与您在自己的数据中心中操作的传统网络相似,但它有 Azure 基础设施的其他优点,例如规模、可用性和成本。 [详细了解 VNet 优势](#)

项目详细信息

订阅: docuSign-TestSign-Middleware-NonProd

资源组: docuSign-test

实例详细信息

名称: xxxxx

区域: (Asia Pacific) 中国北部 3

查看 + 创建 < 上一步 下一步: IP 地址 下载自动化脚本

下一步

Microsoft Azure 由世纪互联运营

主页 > 创建资源 >

创建虚拟网络

基本 IP 地址 安全性 标记 查看 + 创建

虚拟网络的地址空间。采用 CIDR 表示法(例如 192.168.1.0/24)指定为一个或多个地址前缀。

IPv4 地址空间

10.1.0.0/16 10.1.0.0 - 10.1.255.255 (65536 个地址)

☐ 添加 IPv6 地址空间

子网的地址范围采用 CIDR 表示法(例如 192.168.1.0/24)。它必须包含在虚拟网络的地址空间中。

+ 添加子网 删除子网

子网名称	子网地址范围	NAT 网关
default	10.1.0.0/24	-

建议使用 NAT 网关处理从子网出站的 Internet 访问。创建虚拟网络后，可以部署 NAT 网关并将其分配给子网。 [了解出站 NAT](#)

查看 + 创建 < 上一步 下一步: 安全性 > 下载自动化脚本

然后下一步直到创建

6) 选择刚刚创建好的子网和公网ip然后下一步直到创建完成

Microsoft Azure 由世纪互联运营

主页 > 虚拟机 >

创建虚拟机

基本 磁盘 网络 管理 Monitoring 高级 标记 查看 + 创建

通过配置网络接口和 NIC 设置来建立虚拟机和网络连接。你可通过安全组规则来控制端口。入站连接和出站连接。也可采用现有负载均衡解决方案。 [了解更多信息](#)

网络接口

创建虚拟机时，Azure 门户会创建一个网络接口。

虚拟网络 ☐ docusign-test-vn [新建](#)

子网 ☐ default (10.0.0.0/24) [新建子网](#)

公用 IP ☐ 无 [新建](#)

NIC 网络安全组 ☐ 无 ☒ 基本 ☐ 高级

公共入站端口 ☐ 无 ☒ 允许选定的端口

选择入站端口 22 (SSH)

启用加速网络 ☒

负载均衡

可将此虚拟机放在现有 Azure 负载均衡解决方案的池当中。 [了解更多信息](#)

是否将此虚拟机置于现有负载均衡解决方案之后? ☐

DocuSign服务器所需服务安装

通过创建服务器的用户名和密码连接到服务器

Docker 安装

更新系统

- ▼ 在开始安装之前，先更新 Ubuntu 系统。可以通过以下命令更新

Bash

🔗 复制代码

```
1 sudo apt-get update sudo apt-get upgrade
```

安装 Docker

- ▼ 执行以下命令，添加 Docker 的 GPG 密钥：

Bash

🔗 复制代码

```
1 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

- ▼ 添加完 GPG 密钥后，添加 Docker 软件包源：

Bash

🔗 复制代码

```
1 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

- ▼ 更新源信息：

Bash

🔗 复制代码

```
1 sudo apt-get update
```

- ▼ 最后，安装 Docker：

Bash

🔗 复制代码

```
1 sudo apt-get install docker-ce
```

启动 Docker

- ▼ 安装完成后，启动 Docker：

Bash

🔗 复制代码

```
1 sudo systemctl start docker
```

- ▼ 为了防止每次开机都需要手动启动 Docker，还可以设置 Docker 服务开机自启

Bash

🔗 复制代码

```
1 sudo systemctl enable docker
```

▼ 现在 Docker 已经安装完成，可以通过运行以下命令检查 Docker 是否安装成... Bash | 复制代码

```
1 docker --version
```

这将显示安装的 Docker 版本信息。

Nginx 安装 [9806095_www.iscn.com.cn_nginx.zip](#)

创建nginx.conf文件

▼ 创建nginx.conf文件 Bash | 复制代码

```
1 vim nginx.conf
```



```
1  ▼ events {
2      worker_connections 1024;
3  }
4
5  ▼ http {
6      include      mime.types;
7      default_type  application/octet-stream;
8      sendfile      on;
9      keepalive_timeout 65;
10
11     server {
12         listen      443 ssl;
13         server_name www.iscn.com.cn;
14
15         ssl_certificate      /etc/ssl/certs/cert.pem;
16         ssl_certificate_key  /etc/ssl/private/key.pem;
17         ssl_protocols        TLSv1.2 TLSv1.3;
18         ssl_ciphers          HIGH:!aNULL:!MD5;
19
20         location / {
21             # 这里将443端口上的流量代理到本机的80端口
22             proxy_pass http://127.0.0.1:80;
23             proxy_set_header Host $host;
24             proxy_set_header X-Real-IP $remote_addr;
25             proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
26             proxy_set_header X-Forwarded-Proto https;
27         }
28     ▼ location /mmjj.jpg {
29         # 这里将https://163.228.234.38:443/mmjj.jpg代理到本机端口443上
30         proxy_pass http://222.128.117.139:8868/mmjj.jpg;
31         proxy_set_header Host $host;
32         proxy_set_header X-Real-IP $remote_addr;
33         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
34         proxy_set_header X-Forwarded-Proto https;
35     }
36
37     error_page 502 /502.html;
38     ▼ location = /502.html {
39         root /var/www/errors/;
40         internal;
41     }
42 }
43 }
```

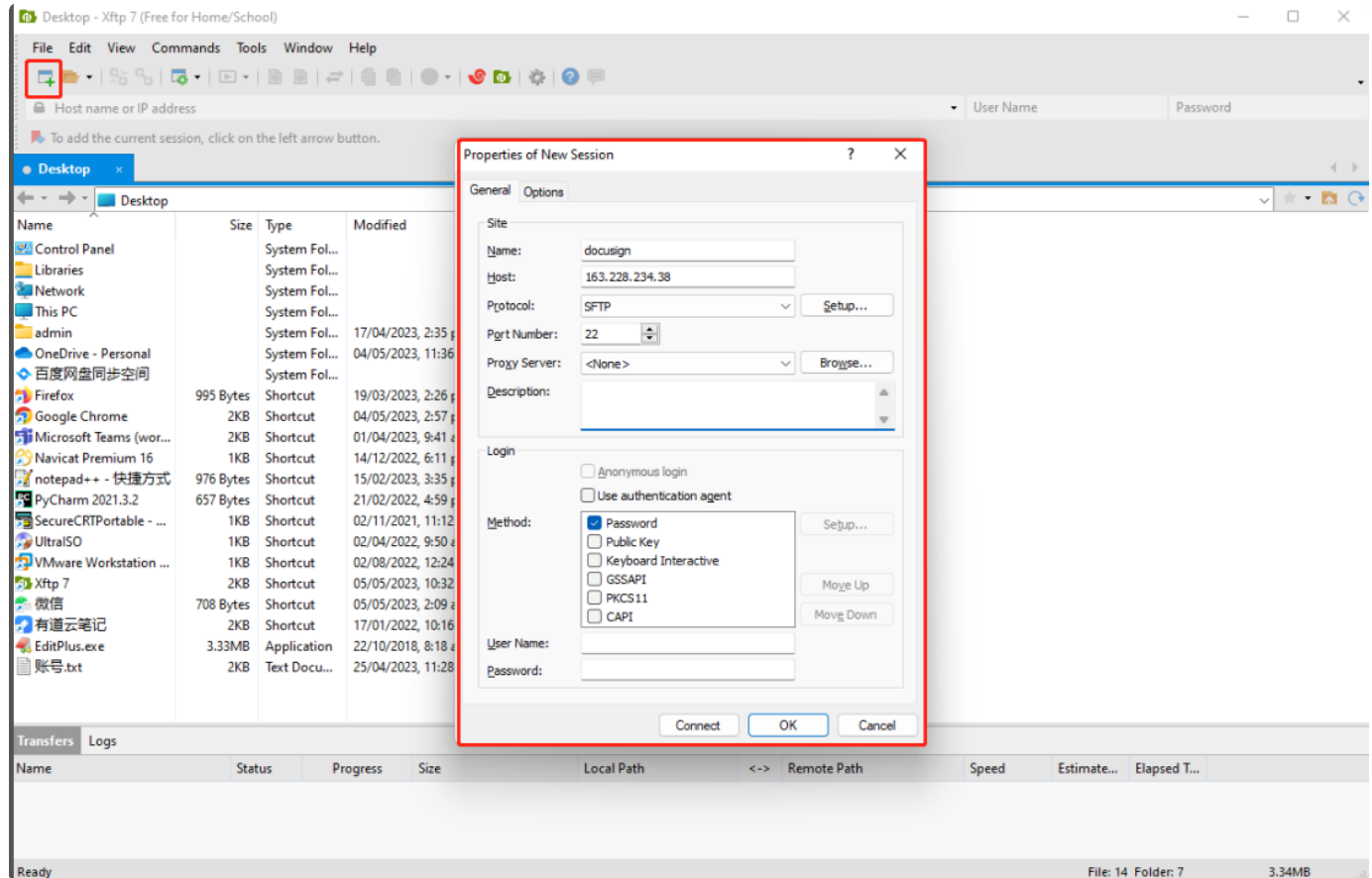
证书准备

将证书下载到本地服务器中在Dockerfile中打包后再本地服务器删除

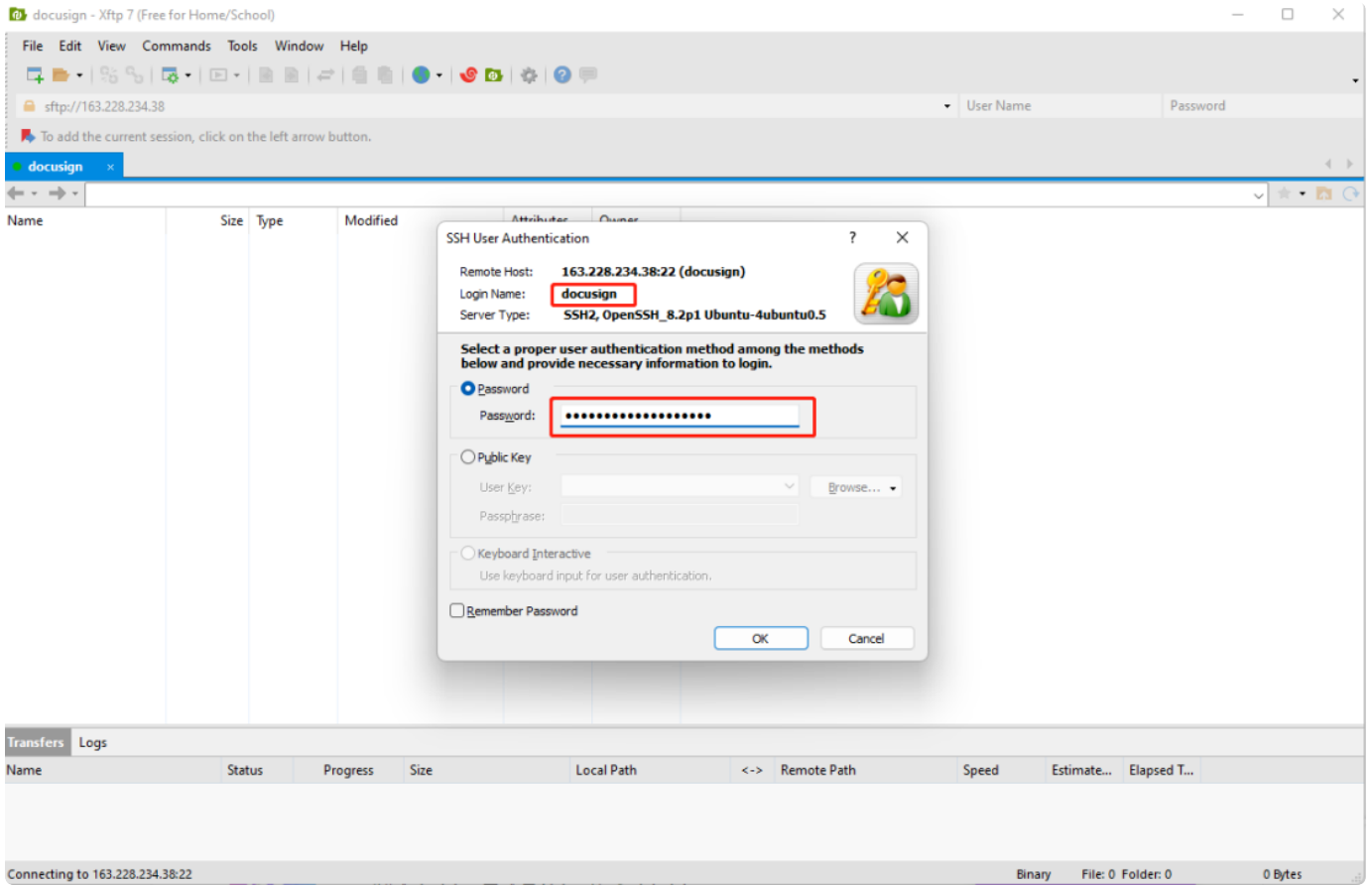
<https://www.xshell.com/zh/xshell/> (xshell 服务器ssh工具)

<https://www.xshell.com/zh/xftp/> (xftp ftp工具)

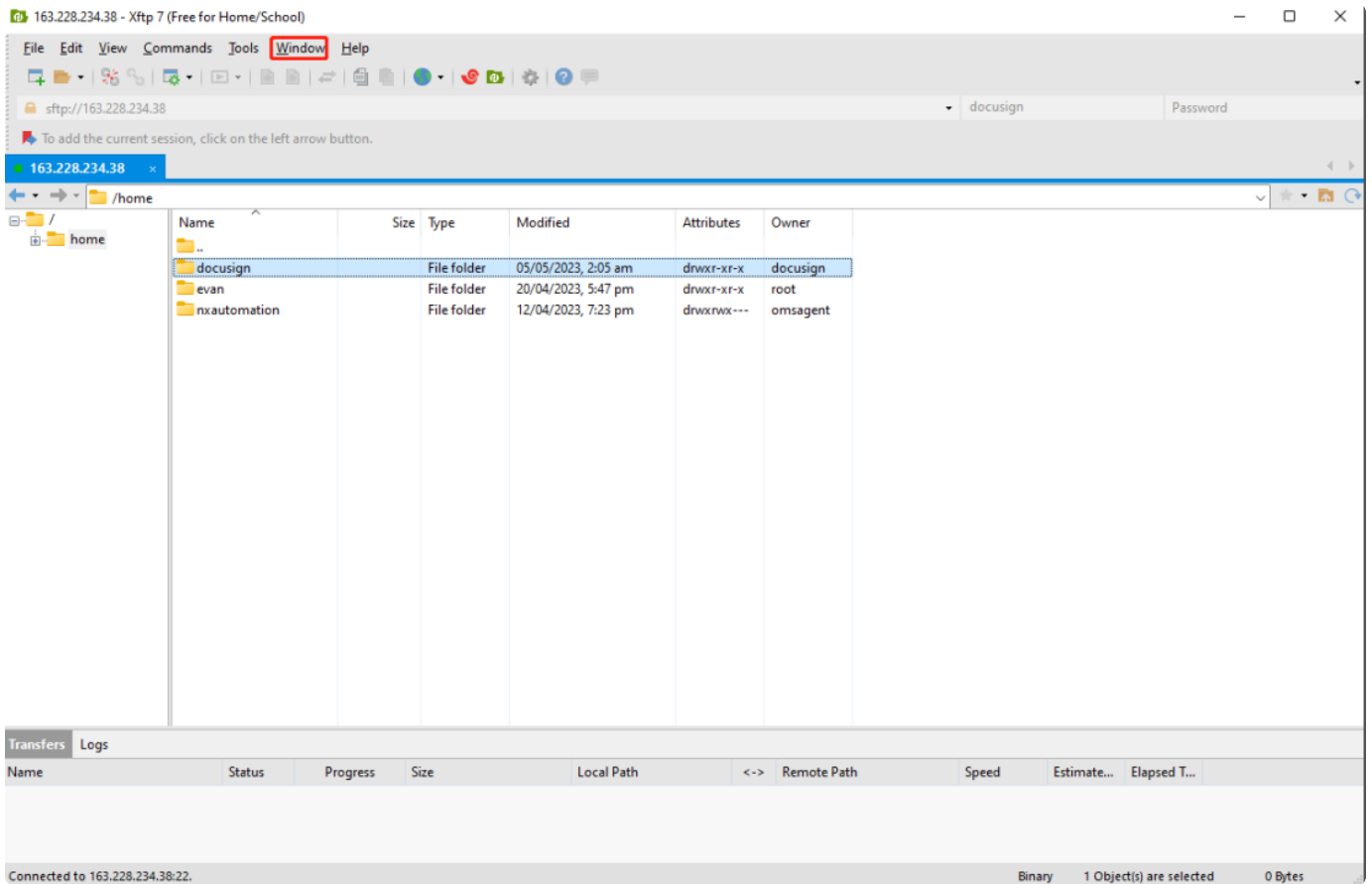
打开下载好的xftp上传本地的证书文件



输入用户名和密码（确保自己的ip在ssh白名单中）



点击window创建一个windows和xftp的链接用来上传文件



▼ 将证书文件上传到服务器然后改名为cert.pem 和key.pem

Bash

📄 复制代码

```
1 mv 9806095_www.iscn.com.cn.pem cert.pem
2 mv 9806095_www.iscn.com.cn.key key.pem
```

打包镜像

创建Dockerfile

▼ Dockerfile

Dockerfile

📄 复制代码

```
1 FROM nginx:latest
2 # Install required packages
3 RUN apt-get update && \
4     apt-get install -y curl jq && \
5     rm -rf /var/lib/apt/lists/*
6
7 # Copy the script and make it executable
8 COPY get_keyvault_cert.sh /get_keyvault_cert.sh
9 RUN chmod +x /get_keyvault_cert.sh
10
11 # Copy the Nginx configuration file
12 COPY nginx.conf /etc/nginx/nginx.conf
13 COPY cert.pem /etc/ssl/certs/cert.pem
14 COPY key.pem /etc/ssl/private/key.pem
15 COPY 5021.html /var/www/errors/502.html
16 COPY mmjj.jpg /var/www/errors/mmjj.jpg
17
18 # Change ownership and permissions of error page and image
19 RUN chown -R nginx:nginx /var/www/errors \
20     && chmod 644 /var/www/errors/*.html \
21     && chmod 644 /var/www/errors/*.jpg
22 CMD nginx -g "daemon off;"
```

运行Dockerfile

▼ 运行Dockerfile

Bash

📄 复制代码

```
1 docker build -t my-nginx .
```

▼ 运行成功后删除本地证书

Bash

📄 复制代码

```
1 rm -rf cert.pem key.pem
```

中间件安装 [app_publish.tar](#)

发布程序

1. 通过 terminal 进入系统源代码目录下 app 目录。
2. 发布程序

▼ 发布程序

Bash

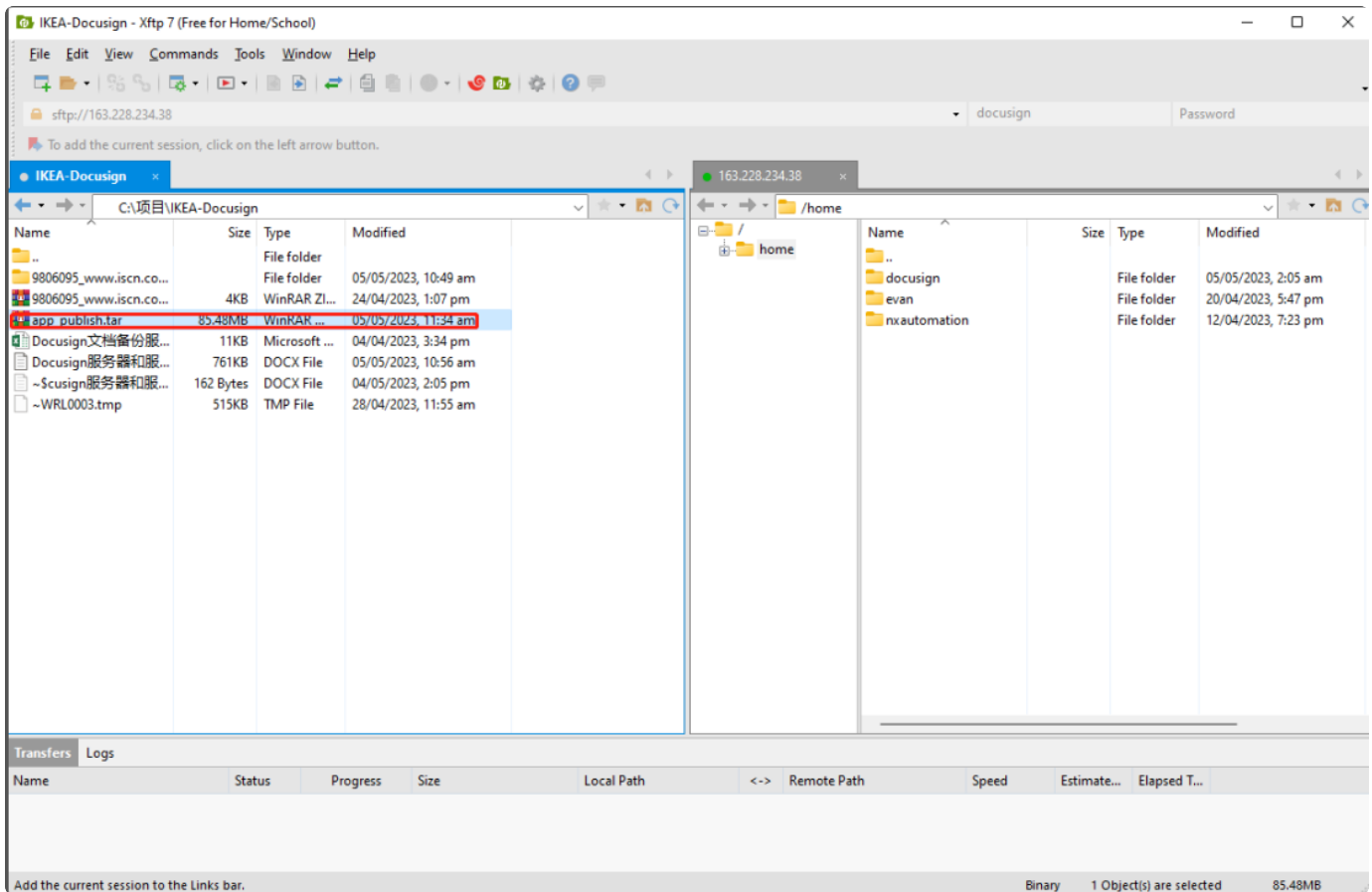
📄 复制代码

```
1 dotnet publish -c Release
```

3. 将文件打包上传至服务器

部署文件上传

使用xftp工具上传publish.tar 到服务器解压



构建镜像

构建镜像

Bash

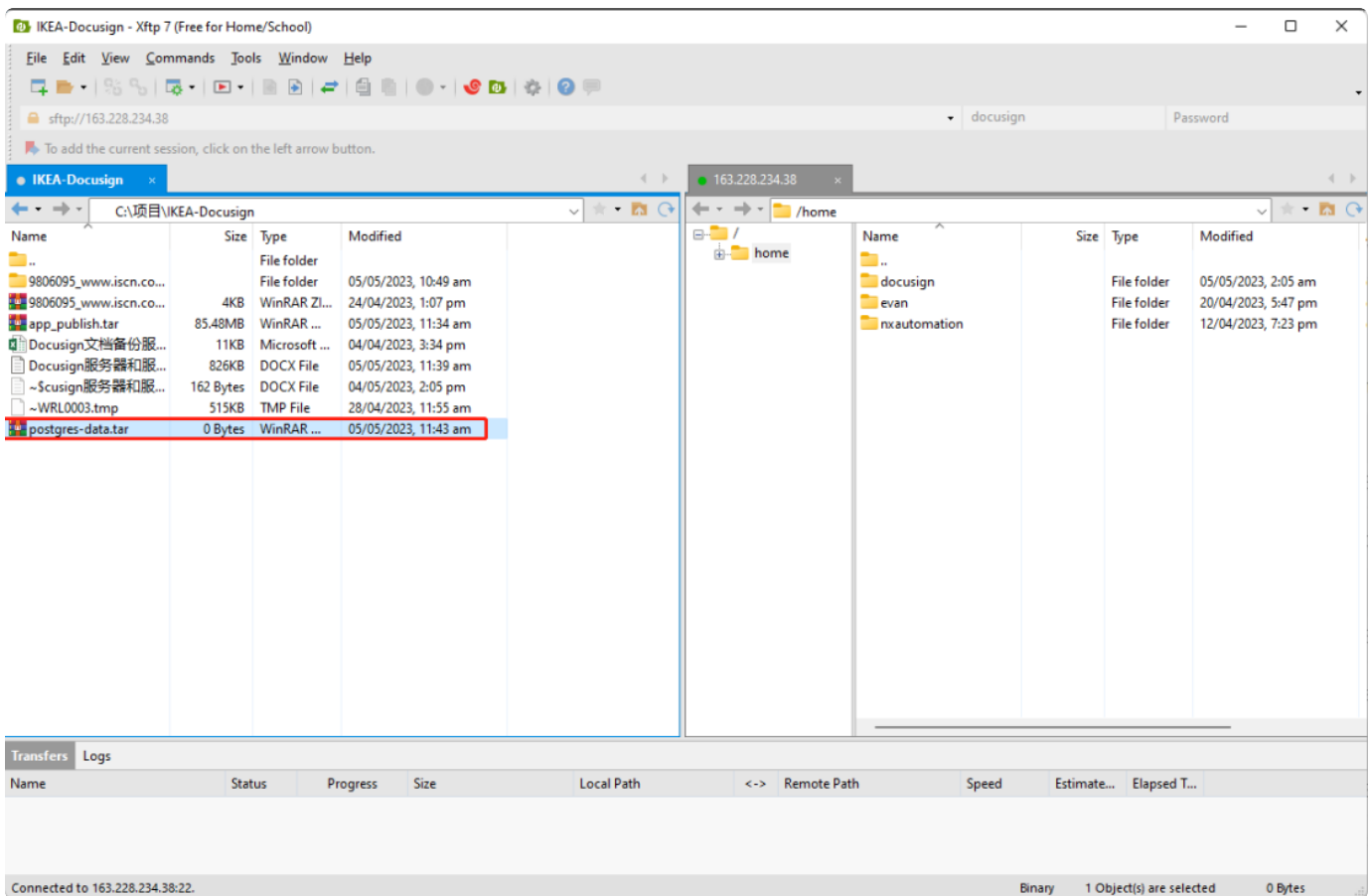
复制代码

```
1 tar xf app_publish.tar
2 cd app_publish
3 docker build -t app .
```

Postgres 安装 postgres-data.tar

上传静态文件postgres-data到服务器

使用xftp工具将postgres-data.tar上传到服务解压



▼ 解压postgres-data.tar

Bash

复制代码

```
1 tar xf postgres-data.tar
```

Docker Compose 配置

▼ 创建docker-compose

Bash

复制代码

```
1 vim docker-compose.yml
```

▼ docker-compose.yml

Plain Text

📄 复制代码

```
1  version: '3.7'
2  services:
3    db:
4      image: postgres
5      restart: always
6      environment:
7        POSTGRES_PASSWORD: MyP@ssw0rd123!
8        POSTGRES_PASSWORD_REQUIREMENTS: 'length:8,uppercase:1,lowercase:1,number:1,special:1'
9      volumes:
10       - ./postgres-data:/var/lib/postgresql/data
11     ports:
12       - "5432:5432"
13
14     nginx:
15       image: my-nginx
16       restart: always
17       volumes:
18         - ./nginx.conf:/etc/nginx/nginx.conf
19       # ports:
20       #   - "443:443"
21       network_mode: host
22
23     app:
24       image: app
25       environment:
26         TZ: Asia/Shanghai
27       restart: always
28       ports:
29         - "80:80"
30       # volumes:
31       #   - ./appsettings.json:/app/appsettings.json
```

启动程序

▼ 启动docker-compose

Bash

📄 复制代码

```
1  docker-compose up -d
2  docker-compose ps
```



```
root@docusign-test-vm:/home/docusign/cert# docker-compose ps

```

Name	Command	State	Ports
cert_app_1	dotnet App.dll	Up	0.0.0.0:80->80/tcp, :::80->80/tcp
cert_db_1	docker-entrypoint.sh postgres	Up	0.0.0.0:5432->5432/tcp, :::5432->5432/tcp
cert_nginx_1	/docker-entrypoint.sh /bin ...	Up	

```
root@docusign-test-vm:/home/docusign/cert#
```