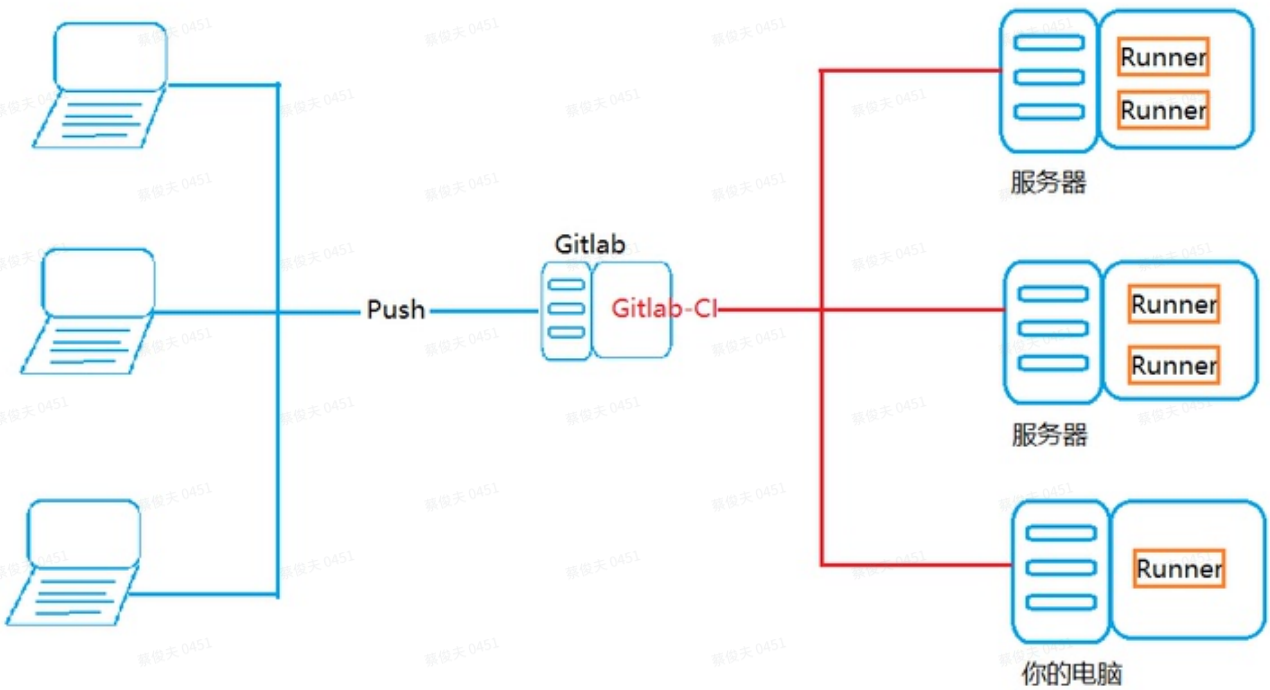
使用 GitLab CI/CD 实现自动化发布网站至IIS服务器

下面列出我本次使用的环境：

> 操作系统：Windows 11
> 项目版本：.NET Core 6.0
> 脚本执行环境：PowerShell 5.1.22621.963
> Web服务器：IIS 10.0
> Gitlab：git.local



基本步骤

1. 配置GitLab CI/CD Runner

首先，在本地或公司内网的机器上设置一个GitLab CI/CD Runner，该Runner将用于执行CI/CD任务。确保该Runner已经正确配置和注册到GitLab服务器。

注册流程

a. **下载 Gitlab Runner**

下载地址：https://docs.gitlab.com/runner/install/

本次下载的版本为 `Windows amd64`

下载好后将文件改名为 `gitlab-runner.exe`，使用cmd管理员模式进入到该文件目录

b. **注册 Gitlab Runner**

命令行输入 `.\gitlab-runner register`

出现提示 `Enter the GitLab instance URL (for example, https://gitlab.com/):`

此时需要输入协调器的地址，这个地址是在你 gitlab 项目的 `左侧导航栏` ， `设置` 下的 `CI/CD` 中，找到 `Runner` 点击展开，就会看到 `指定Runner` 和 `共享Runner` ,这里我们使用 `指定Runner` 做演示说明。

复制URL至刚才的命令中回车。

出现提示： `Enter the registration token:` ，此时输入 `CI/CD` 下的 注册令牌，然后需要输入描述，根据需求填。

之后提示： `Enter tags for the runner (comma-separated):` ，需要我们输入 Runner 的标签，这里我使用 deploy。输入后回车提示 `Enter optional maintenance note for the runner:` 也根据需求填。

回车后正常提示：（如报错请见**注册Gitlab Runner时报错**）

```
Registering runner... succeeded                     runner=GR1348941ZNHLJa9w
```

最后一个提示： `Enter an executor: parallels, virtualbox, docker+machine, instance, kubernetes, docker-windows, docker-ssh,`

`shell, ssh, docker-autoscaler, docker-ssh+machine, custom, docker:`，需要我们填入脚本的执行环境，这里先填 `shell`，这时你的 gitlab-runner.exe 下会生成一个 `config.toml` 文件，里面保存着我们刚刚输入的信息。

```
Runner registered successfully. Feel free to start it, but if it's running already the config should be automatically re
loaded!

Configuration (with the authentication token) was saved in "C:\\GitLab-Runner\\config.toml"
```

完整命令

```
PS C:\GitLab-Runner> .\gitlab-runner register
Runtime platform                                    arch=amd64 os=windows pid=10284 revision=436955cb version=15.11.0
Enter the GitLab instance URL (for example, https://gitlab.com/):
https://git.local/
Enter the registration token:
GR1348941ZNHLJa9wTLEMKwm-QsYj
Enter a description for the runner:
[tsdn301]: runner
Enter tags for the runner (comma-separated):
deploy
Enter optional maintenance note for the runner:
runner
WARNING: Support for registration tokens and runner parameters in the 'register' command has been deprecated in GitLab R
unner 15.6 and will be replaced with support for authentication tokens. For more information, see https://gitlab.com/git
lab-org/gitlab/-/issues/380872
Registering runner... succeeded                     runner=GR1348941ZNHLJa9w
Enter an executor: parallels, virtualbox, docker+machine, instance, kubernetes, docker-windows, docker-ssh, shell, ssh,
docker-autoscaler, docker-ssh+machine, custom, docker:
shell
Runner registered successfully. Feel free to start it, but if it's running already the config should be automatically re
loaded!

Configuration (with the authentication token) was saved in "C:\\GitLab-Runner\\config.toml"
```

也可尝试用参数一次性输入

```
./gitlab-runner.exe register --non-interactive --tls-ca-file=/etc/gitlab/ssl/git.local.crt --url "https://git.local/"  --registration-token "你的Runner注册令牌" --executor "shell"  --description "runner"   --tag-list "deploy"  --run-untagged  --locked="false"
```

c. 将 Shell 改成 PowerShell

上面选择脚本执行环境的时候我们选了 `shell`，但是本次我是在 Windows 环境下运行，所以需要将它改为 PowerShell，打开 `config.toml` 文件，在 `[[runners]]` 下改成 `shell = "powershell"`，然后保存文件即可。

```
concurrent = 1
check_interval = 0
shutdown_timeout = 0

[session_server]
  session_timeout = 1800

[[runners]]
shell = "powershell"
  name = "runner"
  url = "https://git.local/"
  id = 3
  token = "yC6CeTYgebZxyTbadb2F"
  token_obtained_at = 2023-05-25T02:29:46Z
  token_expires_at = 0001-01-01T00:00:00Z
  executor = "shell"
```

2. 创建CI/CD配置文件

在你的项目中创建一个名为 `.gitlab-ci.yml` 的文件，该文件定义了CI/CD的工作流程。在该文件中，需要定义一系列的阶段和任务，以完成自动化发布网站至本地IIS的过程。以下为示例代码：

```yaml
before_script:
  - cd src
stages:
  - build

# job
test:
  stage: build
  # 将会执行的脚本
  script:
    - dotnet restore
    - dotnet build
  # 哪个分支会执行
  only:
    - main
    #runner 注册时的 tag，这里指会触发的 runner
  tags:
    - deploy
```

以上代码在使用runner运行时会出现以下错误：

原因可能为项目未restore完成就开始build，所以还是需要在脚本上对每个步骤进行细分，以下为修改后的脚本：

```yaml
before_script:
  #解决powershell中文乱码问题
  - chcp 65001
  #进入项目工作目录
  - cd src

# 执行的 job
stages:
  - restore
  - build
  - deploy

# 校验代码
restore:
  stage: restore
  # 将会执行的脚本
  script:
    - dotnet restore
  # 哪个分支会执行
  only:
    - main
    #runner 注册时的 tag，这里指会触发的 runner
  tags:
    - deploy

build:
  stage: build
  # 将会执行的脚本
```

```
  script:
    - dotnet build
  # 哪个分支会执行
  only:
    - main
    #runner 注册时的 tag，这里指会触发的 runner
  tags:
    - deploy
```

`before_script` 在整个项目 clone 到 Runner 所处的服务器时会先执行这个里面的脚本，这里我是进入到了 src 目录，你还可以在这里面做一些包还原的操作。

`stages` 里放的是将会执行的 job。

`build` 是做作业（job），这个命名你可以根据自己的情况来。`build` 就是上面 `stages` 会执行的 job 的真正配置处。

a. `stage` 对应 `stages` 中的项，如果一个 job 没有指定 stage，那么这个任务会分配到test stage。

b. `script` 就是执行的脚本，构建自动化的核心也就是在此处。作为简单的演示，我就还原了包和生成项目。

c. `only` 是值该 job 会在被哪些分支 push 触发。

d. `tags` 上面在我们注册时有提到过，这个 `tags` 对应的是我们注册 `gitlab runner` 时所填的 `tags`，表示的是该 job 会触发哪些 Runner。

OK，我们此时已经将一个最简单的 `.gitlab-ci.yml` 构建好了，将配置好的文件 push（推送）到远端。回到 gitlab 中，我们点击 `CI/CD` 可以看到有一个流水线在运行。

3. 检出代码

在CI/CD配置文件中的第一个阶段，使用Git命令或GitLab提供的CI/CD变量来检出代码库。这将确保在CI/CD Runner上获取最新的代码。

会从gitlab库中将最新代码拉至本地

4. 构建和打包网站

根据项目需求，执行一些构建和打包的操作，例如编译源代码、安装依赖项等。这些步骤可以在CI/CD配置文件中定义为任务。

5. 自动化部署网站至本地IIS

编写脚本或命令来将打包好的网站部署至本地IIS。包括将文件复制到IIS的网站目录、配置IIS站点和应用程序池等。可以使用PowerShell脚本或其他适合的工具来执行这些操作。

在进行编写剩下的脚本之前需要定义几个变量：

`WebsiteName`：站点名称，用户关闭 IIS 站点和 IIS 对应进程池的，如果你的进程池和站点的名称不一致　　请再声明一个变量。
`WebsitePath`：站点的路径，用于备份、删除原有站点、新的项目复制到该路径下。

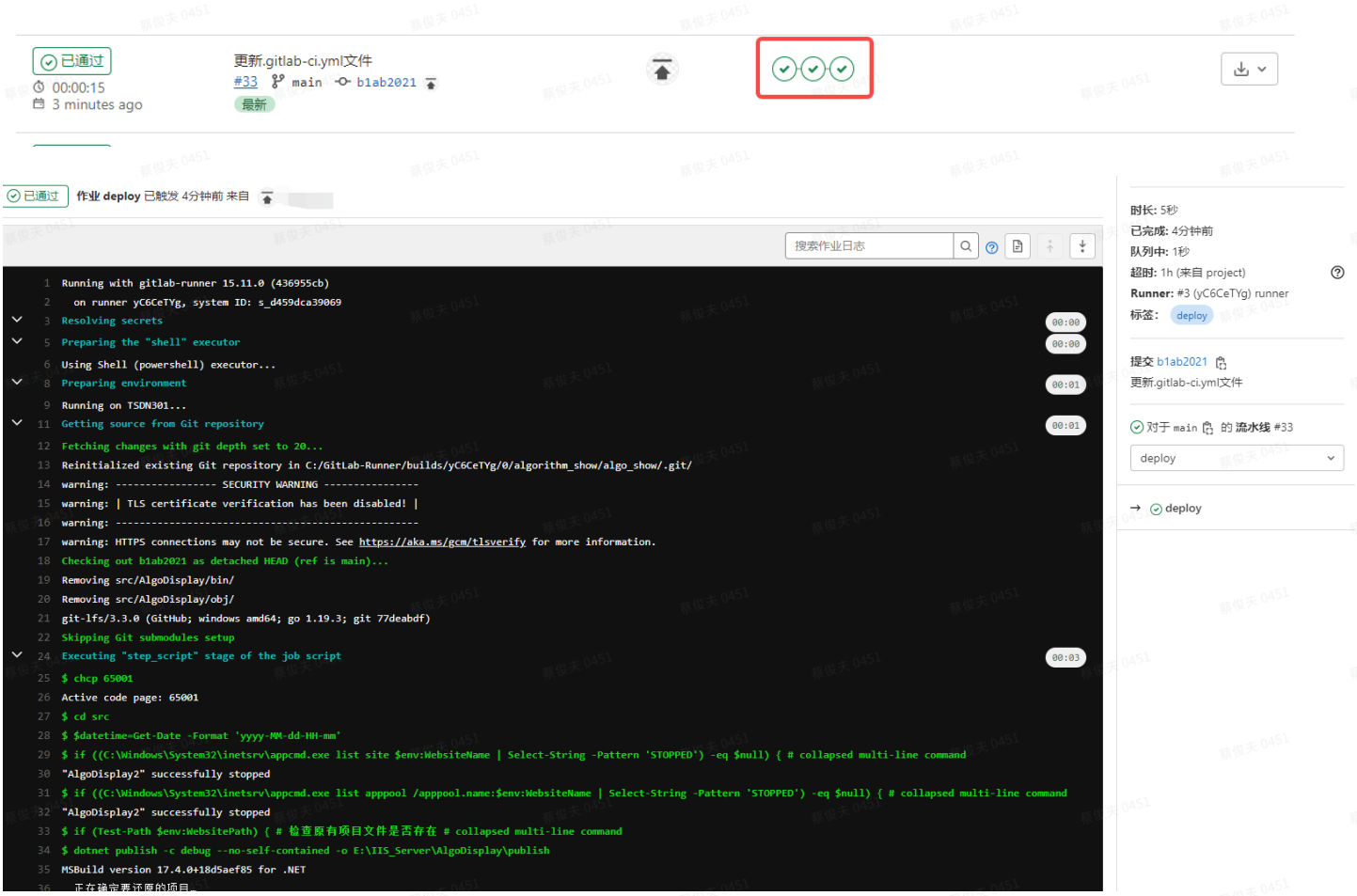
```
1
2  before_script:
3    #中文乱码问题
4    - chcp 65001
5    - cd src
6
7  # 执行的 job
8  stages:
9    - restore
10   - build
11   - deploy
12
13 # 校验代码
14 restore:
15   stage: restore
16   # 将会执行的脚本
17   script:
18     # 使用NuGet还原project.json文件中被指定的依赖项，以及特定于项目的工具。
19     - dotnet restore
```

```
  # 哪个分支会执行
  only:
    - main
    #runner 注册时的 tag，这里指会触发的 runner
  tags:
    - deploy

build:
  stage: build
  # 将会执行的脚本
  script:
    # 生成项目及其所有依赖项
    - dotnet build
  # 哪个分支会执行
  only:
   - main
    #runner 注册时的 tag，这里指会触发的 runner
  tags:
    - deploy

deploy:
  stage: deploy
  script:
    # 声明一个变量保存当前时间，用作备份数据文件夹名称
    - $datetime=Get-Date -Format 'yyyy-MM-dd-HH-mm'

    - |# 当站点未停止时才执行停止站点指令
      if ((C:\Windows\System32\inetsrv\appcmd.exe list site $env:WebsiteName | S
        C:\Windows\System32\inetsrv\appcmd.exe stop site $env:WebsiteName
      }
    - | # 当进程池未停止时才执行停止进程池指令
      if ((C:\Windows\System32\inetsrv\appcmd.exe list apppool /apppool.name:$en
        C:\Windows\System32\inetsrv\appcmd.exe stop apppool /apppool.name:$env:W
      }
    - |

      if (Test-Path $env:WebsitePath) { # 检查原有项目文件是否存在
        # 备份原有项目文件
        cp $env:WebsitePath "$env:WebsitePath$datetime" -Recurse
      }

    # 编译打包项目至iis目录
    - dotnet publish -c debug --no-self-contained -o E:\IIS_Server\AlgoDisplay\p
    # 启动进程池
    - C:\Windows\System32\inetsrv\appcmd.exe start apppool /apppool.name:"$env:W
    # 启动 IIS 站点
    - C:\Windows\System32\inetsrv\appcmd.exe start site $env:WebsiteName
```

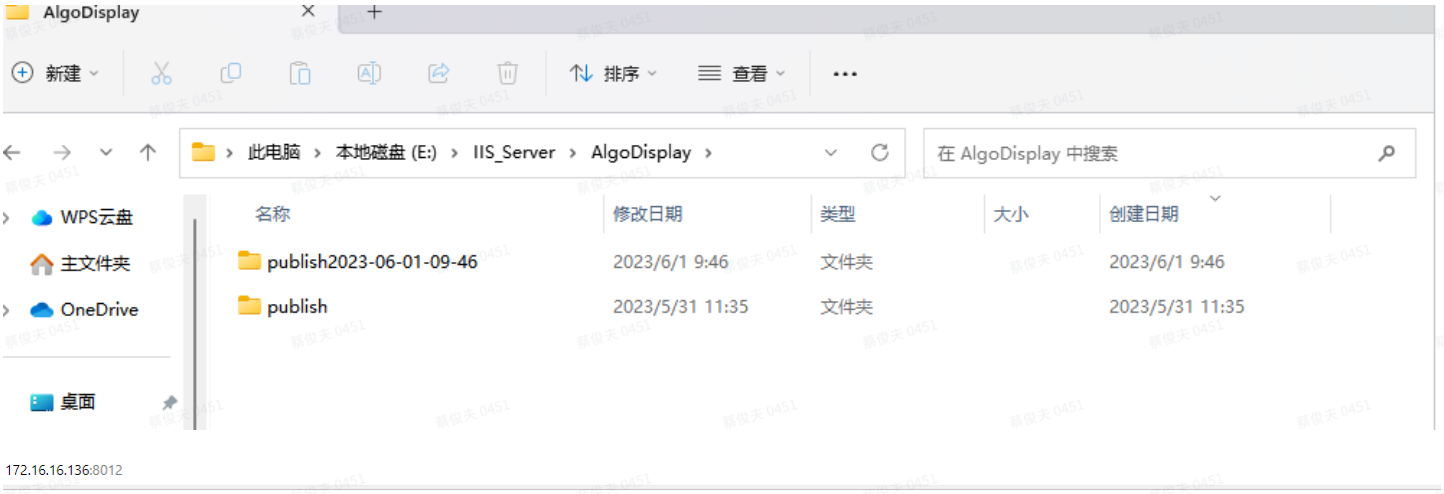
```
67
68    only:
69      - main
70    tags:
71      - deploy
```

到这一步整个自动化发布已经完成了，我们只要 push 代码到远端就会自动部署我们的项目到 IIS 中，但是需要注意必须确保你的 IIS 中已经配置好这个站点了。

回到 gitlab 中查看 `CI/CD` 可以看到这次我们的阶段有三个，因为我配置了三个作业（job），一个restore一个 build和一个 deploy。



IIS路径下文件已备份完毕，网站也可以正常访问了。



6. 清理和收尾

最后，在部署完成后，可以添加一些任务来清理临时文件、关闭连接等，以确保CI/CD环境保持干净和可靠。

遇到的坑

注册Gitlab Runner时报错

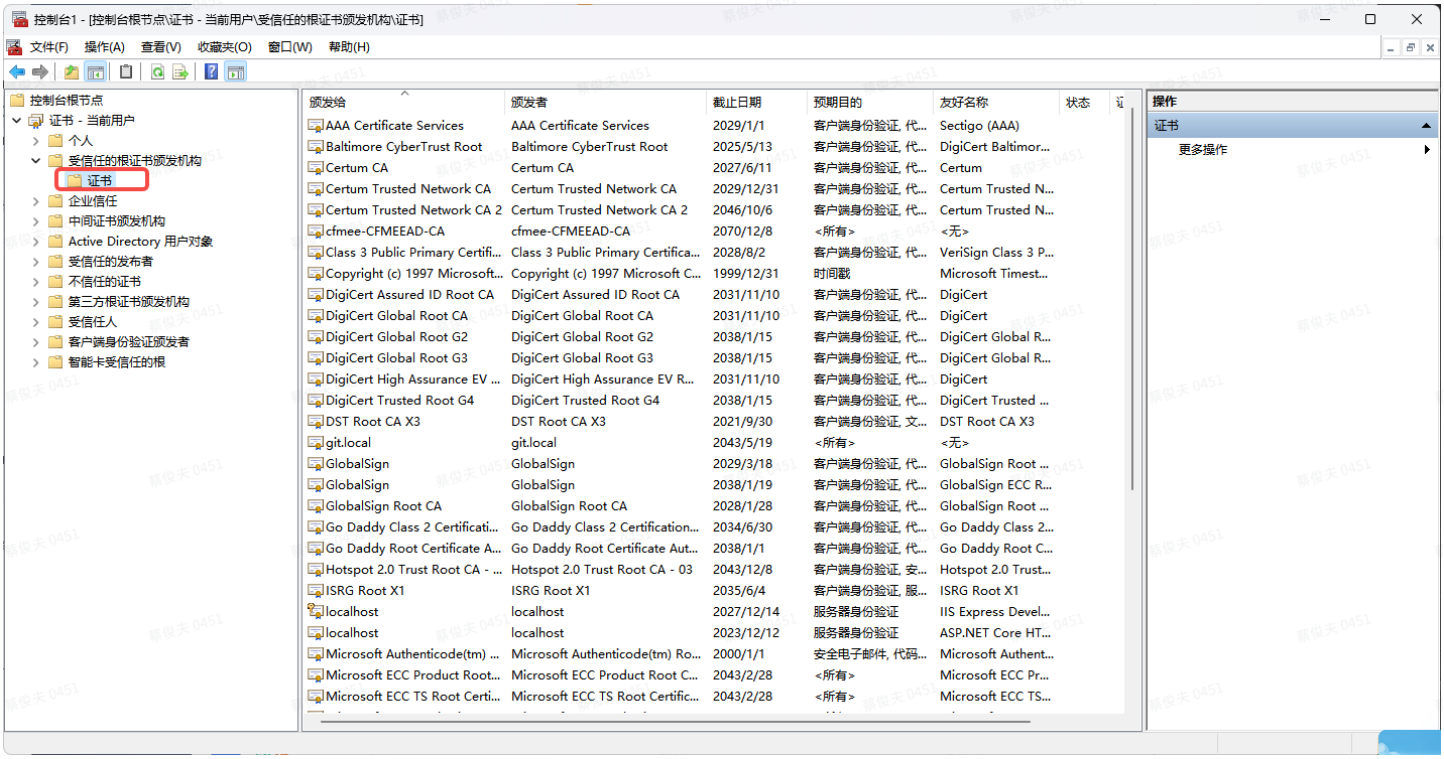
ERROR: Registering runner... failed runner=GR1348941ZNHLJa9w status=couldn't execute POST against https://git.local/api/v4/runners: Post "https://git.local/api/v4/runners": x509: certificate signed by unknown authority

PANIC: Failed to register the runner.

```
PS C:\GitLab-Runner> .\gitlab-runner.exe register --non-interactive --tls-ca-file=/etc/gitlab/ssl/git.local.crt --url "h
ttps://git.local/"  --registration-token "GR1348941ZNHLJa9wTLEMKwm-QsYj" --executor "shell"  --description "runner "   -
-tag-list "run"  --run-untagged  --locked="false"
Runtime platform                                    arch=amd64 os=windows pid=43540 revision=436955cb version=15.11.0
WARNING: Support for registration tokens and runner parameters in the 'register' command has been deprecated in GitLab R
unner 15.6 and will be replaced with support for authentication tokens. For more information, see https://gitlab.com/git
lab-org/gitlab/-/issues/380872
ERROR: Registering runner... failed                 runner=GR1348941ZNHLJa9w status=couldn't execute POST against https:
//git.local/api/v4/runners: Post "https://git.local/api/v4/runners": x509: certificate signed by unknown authority
PANIC: Failed to register the runner.
```

网上搜索原因为证书签名错误，需要reconfig重新生成自签名证书

将生成的 `git.local.crt` 按步骤：控制台1-收信人的根证书颁发机构-证书-右键-导入



作业挂起中，等待进入队列，没有执行流水线

先清除runner缓存

用管理员模式打开cmd，进入runner目录下，输入命令：

`./gitlab-runner stop`

`./gitlab-runner start`

`./gitlab-runner run` / `./gitlab-runner --debug run`

接着按上方向键?

runner开始运行

解决