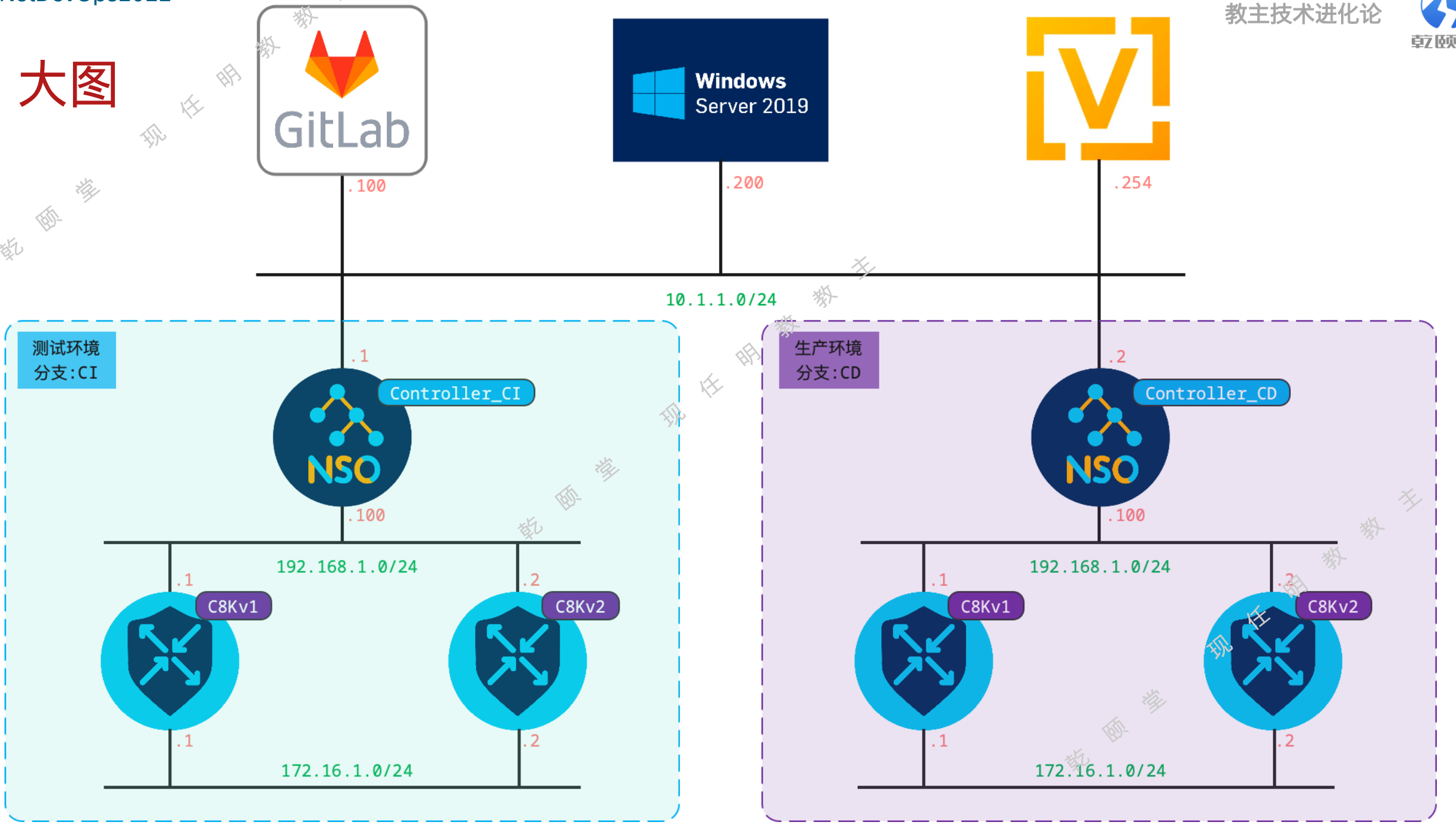
18. NetDevOps2022


教主技术进化论 2022

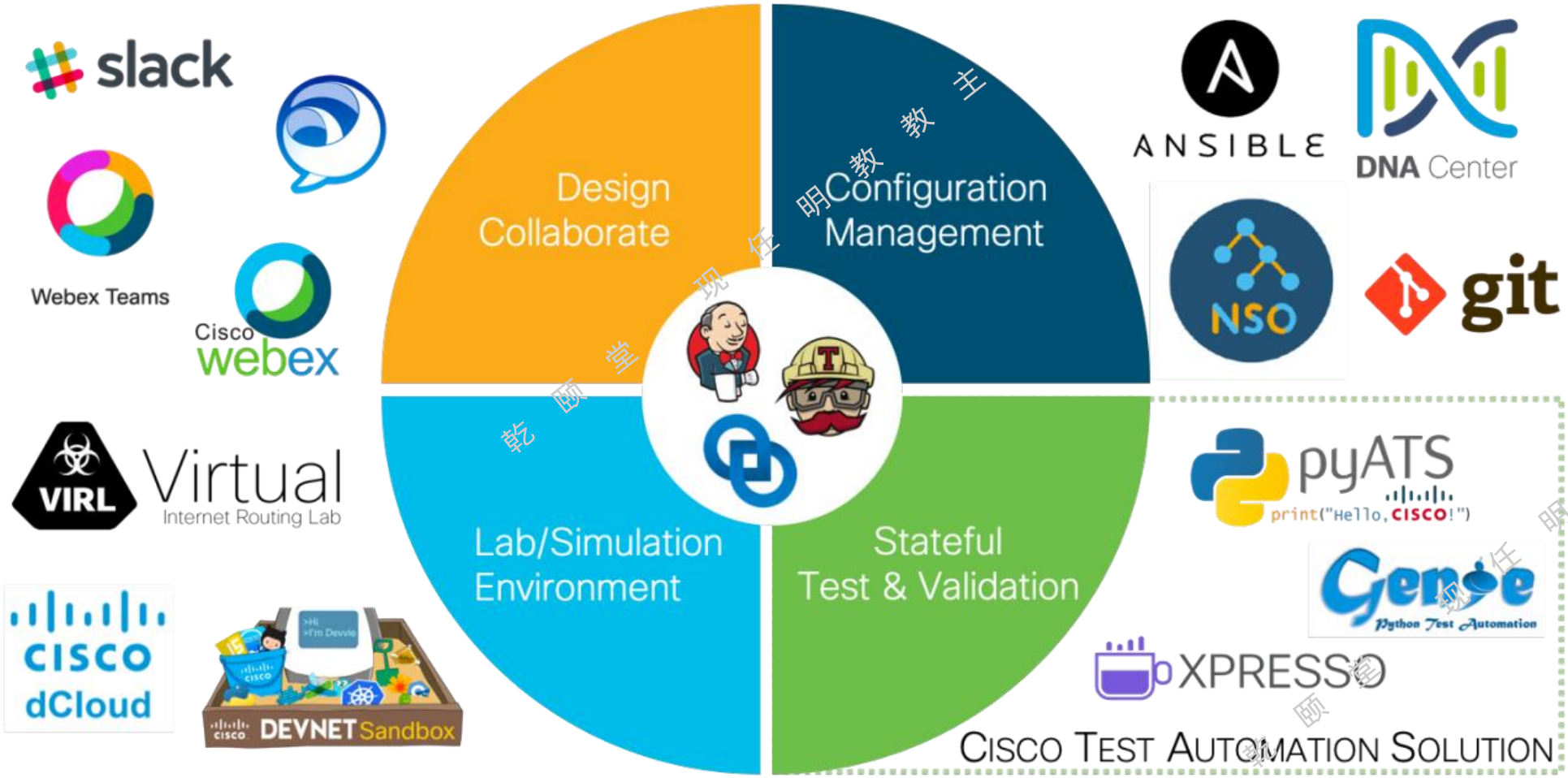
教主VIP，聊点高级的!

大图

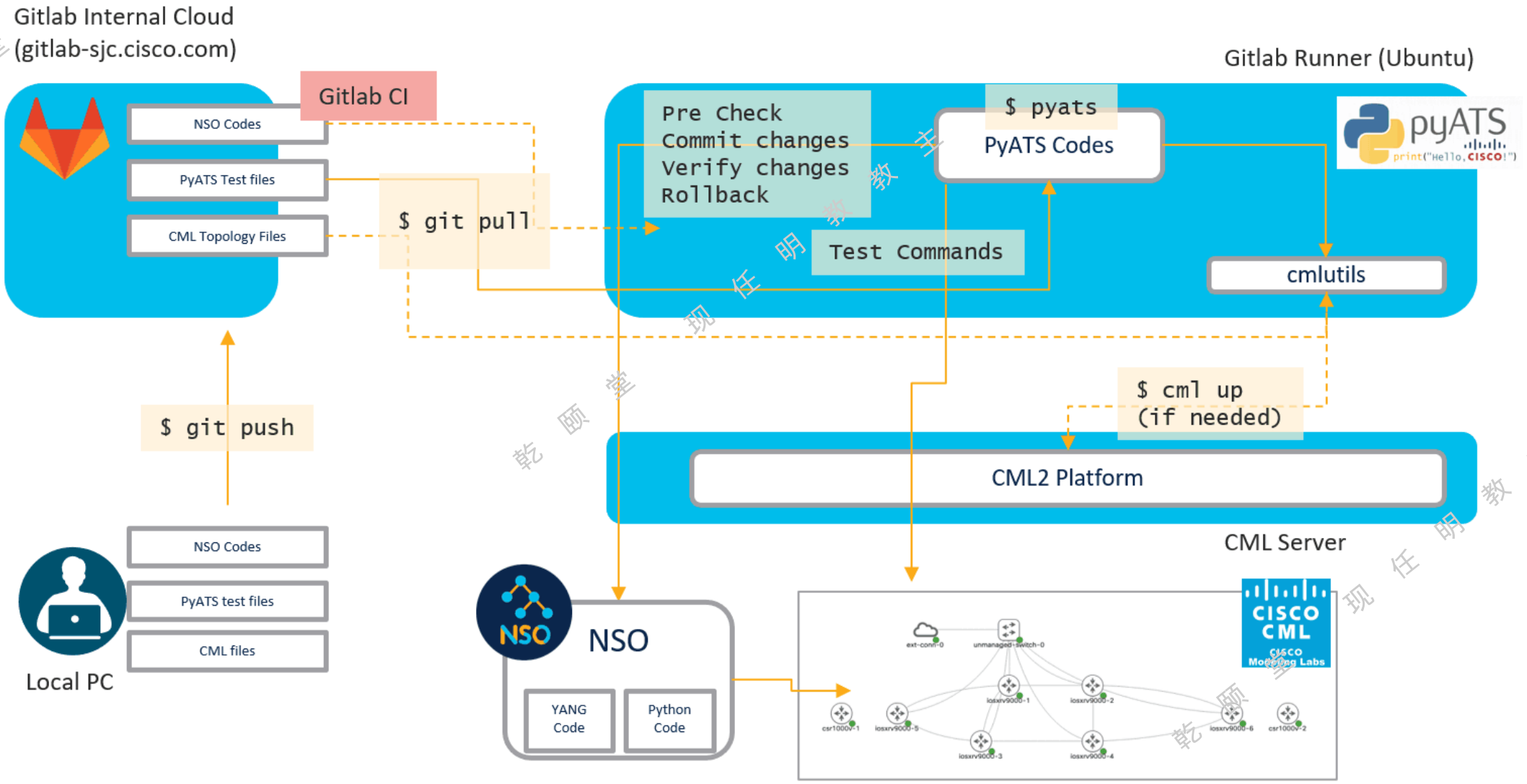

GitLab
.100
Windows Server 2019
.200
.254
10.1.1.0/24
测试环境
分支:CI
.1
Controller_CI
NSO
.100
192.168.1.0/24
.1
C8Kv1
.2
C8Kv2
.1
.2
172.16.1.0/24
生产环境
分支:CD
.2
Controller_CD
NSO
.100
192.168.1.0/24
.1
C8Kv1
.2
C8Kv2
.1
.2
172.16.1.0/24

1. NetDevOps

NetDevOps简介

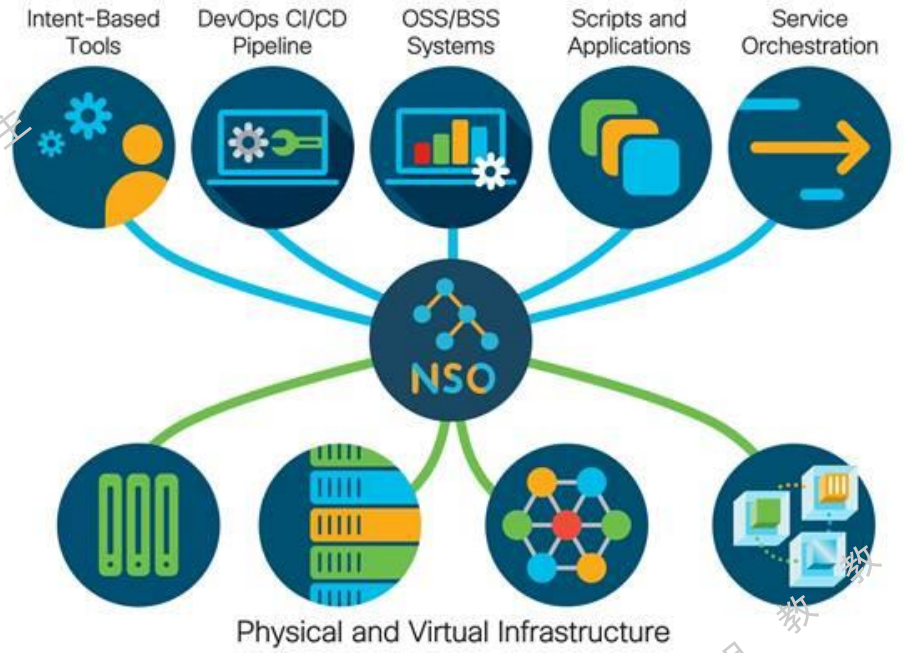


NetDevOps简介



Cisco NSO

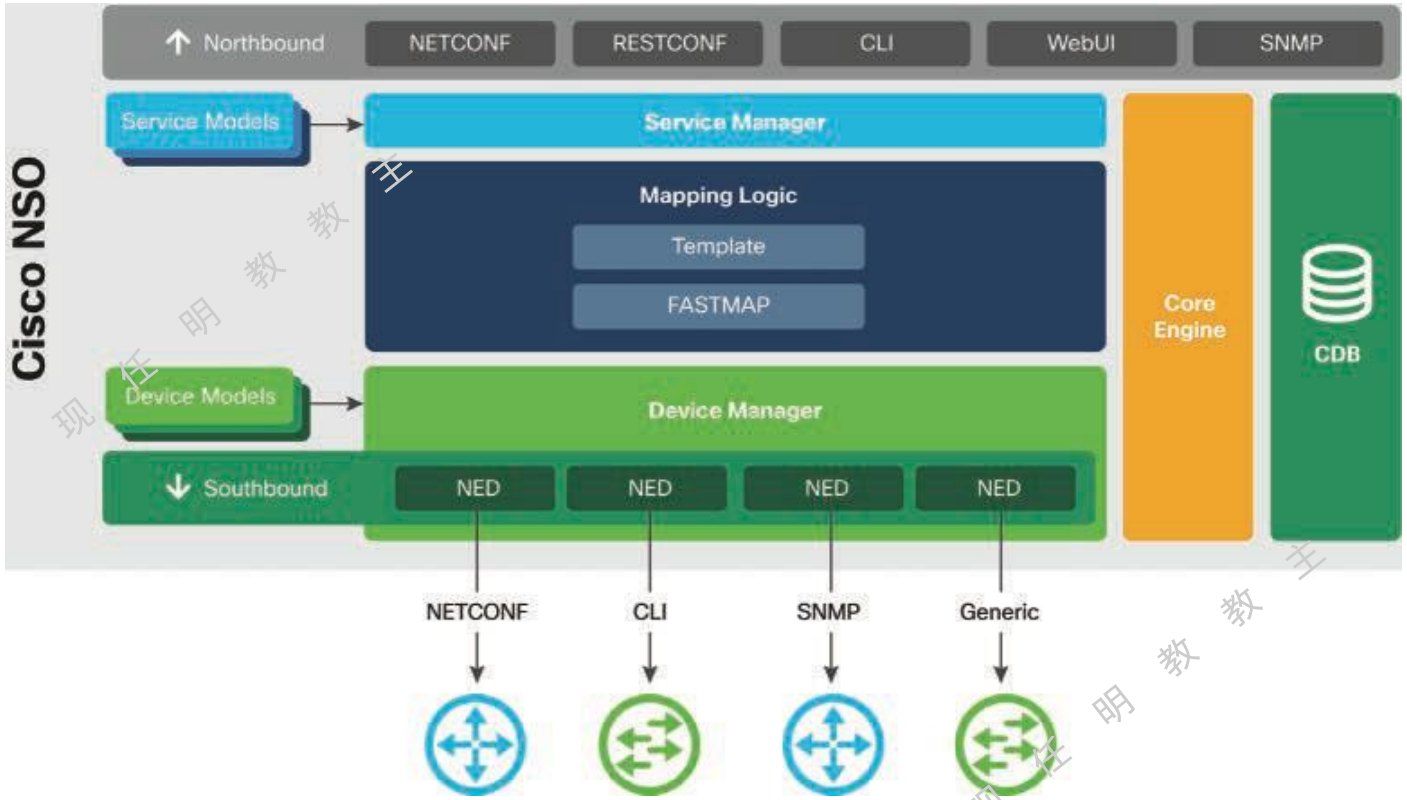
Cisco Network Services Orchestrator (NSO) is a network services orchestration[网络服务编排] platform that enables end-to-end service deployment across multivendor physical and virtual infra- structure[支持多厂商的虚拟和物理架构]. NSO can also be considered a multivendor service-layer SDN controller[多厂商服务层控制器] for data center, enterprise, and service provider networks. It takes full advantage of NETCONF and YANG data models to provide a single API[利用YANG数据模型提供单一API] and a single user interface to the network that it manages. Cisco NSO is the single source of truth in a network, constantly maintaining the current state of all the devices in its database. It ensures that the configuration database is synchronized[确保配置同步] with all the network devices at all times.



Cisco NSO主要组件

The main components of Cisco NSO:

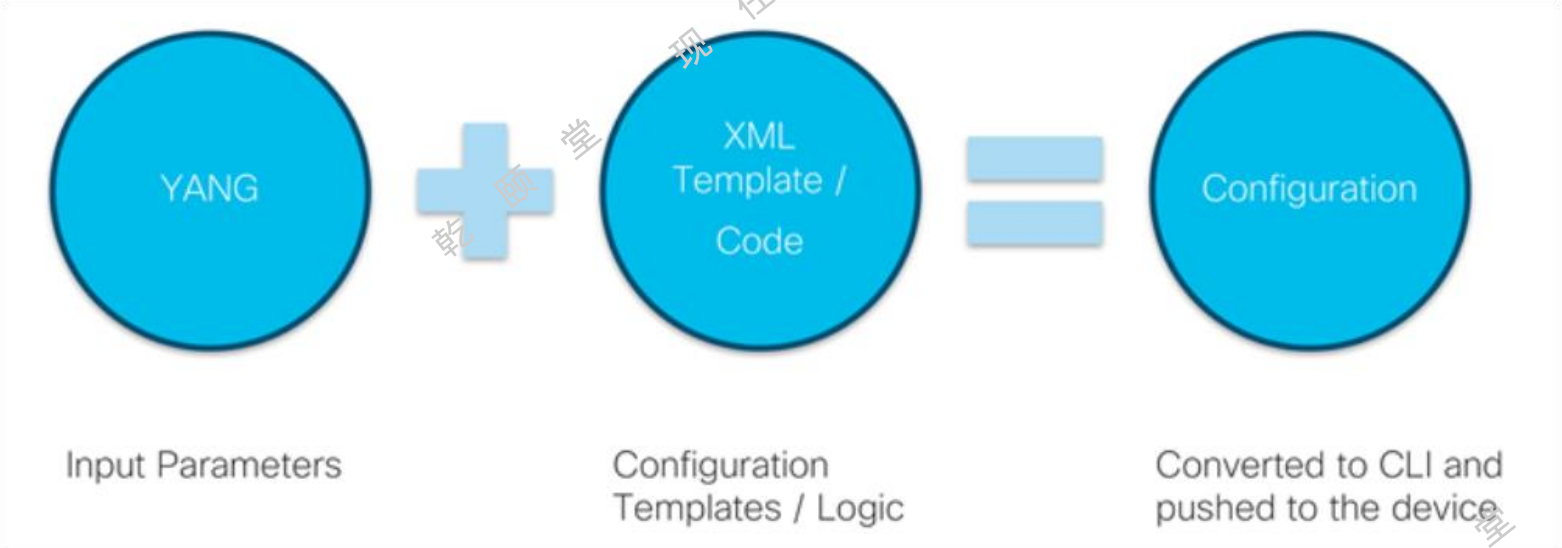
- Service manager
- Device manager
- Mapping logic
- Configuration database



Cisco NSO主要组件

The two primary technologies that are being used with Cisco NSO are the following:

- NETCONF: Used for standard and efficient automation of configuration
- YANG: Used for services and device configuration data modeling



Cisco NSO主要优势

1. 多厂商统一YANG数据结构统一API
2. 原子操作(Distributed atomic transactions)
3. 自动同步与备份配置

Cisco CML(Cisco Modeling Labs)

For a long time, building and maintaining infrastructure laboratories for testing and learning was time-consuming and required a lot of hardware, which needed space, power, and cooling; these labs were therefore very expensive and in reach of only a few companies. With the advent of Network Function Virtualization (NFV) and Cisco Modeling Labs/Cisco Virtual Internet Routing Laboratory (CML/VIRL), it has finally become easy and cost-effective to build infrastructure laboratories for all types of purposes.

[使用NFV技术轻松构建测试和学习环境]

一个CI/CD Pipeline

Downloading and uploading CML/VIRL network topology YAML files is easy and straight- forward, and CML/VIRL users can easily share topologies and also store them in version control systems such as Git. CML/VIRL can be part of a CI/CD pipeline for network con- figuration changes. An example of such a pipeline could contain the following components and processes:

- CML/VIRL topology files could be stored together with automation scripts, device configuration files, and possibly Ansible playbooks in a version control system.
  [CML拓扑文件可以和自动化脚本，设备配置文件，ansible剧本，一起存储在版本控制系统]
- Whenever a change is made to any of these files, a build automation tool such as Jenkins or Drone can be used to monitor and detect the change in version and initiate the automation processes.
  [不管任何文件修改，一个自动化工具，Jenkins或者Drone就会监控到，并且发现修改的版本，然后发起一个自动化流程]
- A complete automated process can create a test environment using the CML/VIRL REST APIs, build application servers, and deploy them in the test environment.
  [一个完整的自动化流程应该包含创建一个测试环境，使用CML REST API，构建应用服务器，部署他们在测试环境]
- The network configuration changes can be tested in this CML/VIRL virtual environment and a pass or fail criterion can be set based on the expectations of the administrator.
  [网络配置的修改应该在CML虚拟环境测试，根据管理员的预期来做出一个pass或者fail的标准]
- If the tests pass, a configuration automation solution like Ansible could use playbooks and automation scripts to apply the tested configuration changes in the production network.
  [如果测试通过，一个配置自动化解决方案，例如Ansible，使用剧本和自动化脚本，应用测试的配置修改到生产环境]

Python Automated Test System (pyATS)

pyATS and the pyATS library together form the **Cisco test and automation solution**, a vibrant ecosystem that aims to **standardize how automated network tests** are set up and run

pyATS was **developed by Cisco in 2014** and **is used extensively internally by thousands of engineers** to run unit tests, regression tests, and end-to-end and integration tests for a large number of Cisco products. The solution **is completely developed in Python3**, making it easy to work with, scalable, and extensible. Millions of pyATS tests are run every month internally at Cisco.

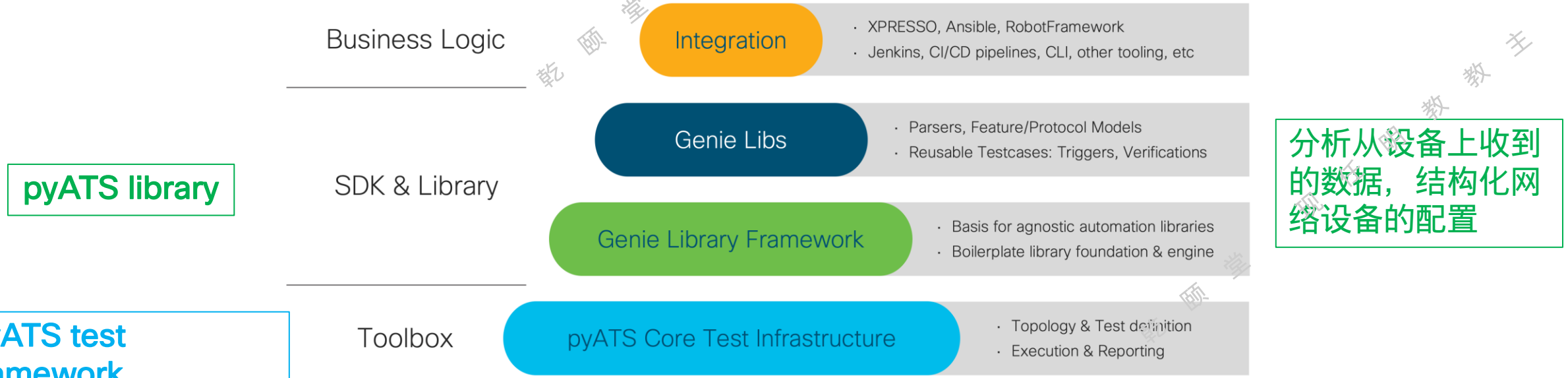
https://developer.cisco.com/docs/pyats-getting-started/

pyATS组件

The **pyATS test framework** provides ways to define **how the network topologies are created** and modeled, **how to connect to devices** through connection libraries, and **how to actually perform the tests and generate reports**

构建拓扑，连接设备，执行测试和产生报告

The **pyATS library** builds on this infrastructure framework and provides easy-to-use libraries that implement pyATS features, **parsers to interpret the data received from the devices**, a mechanism for **modeling network device configurations** for both Cisco and third-party vendors, reusable test cases in the form of triggers and verifications, and the ability to build test suites through YAML-based text files



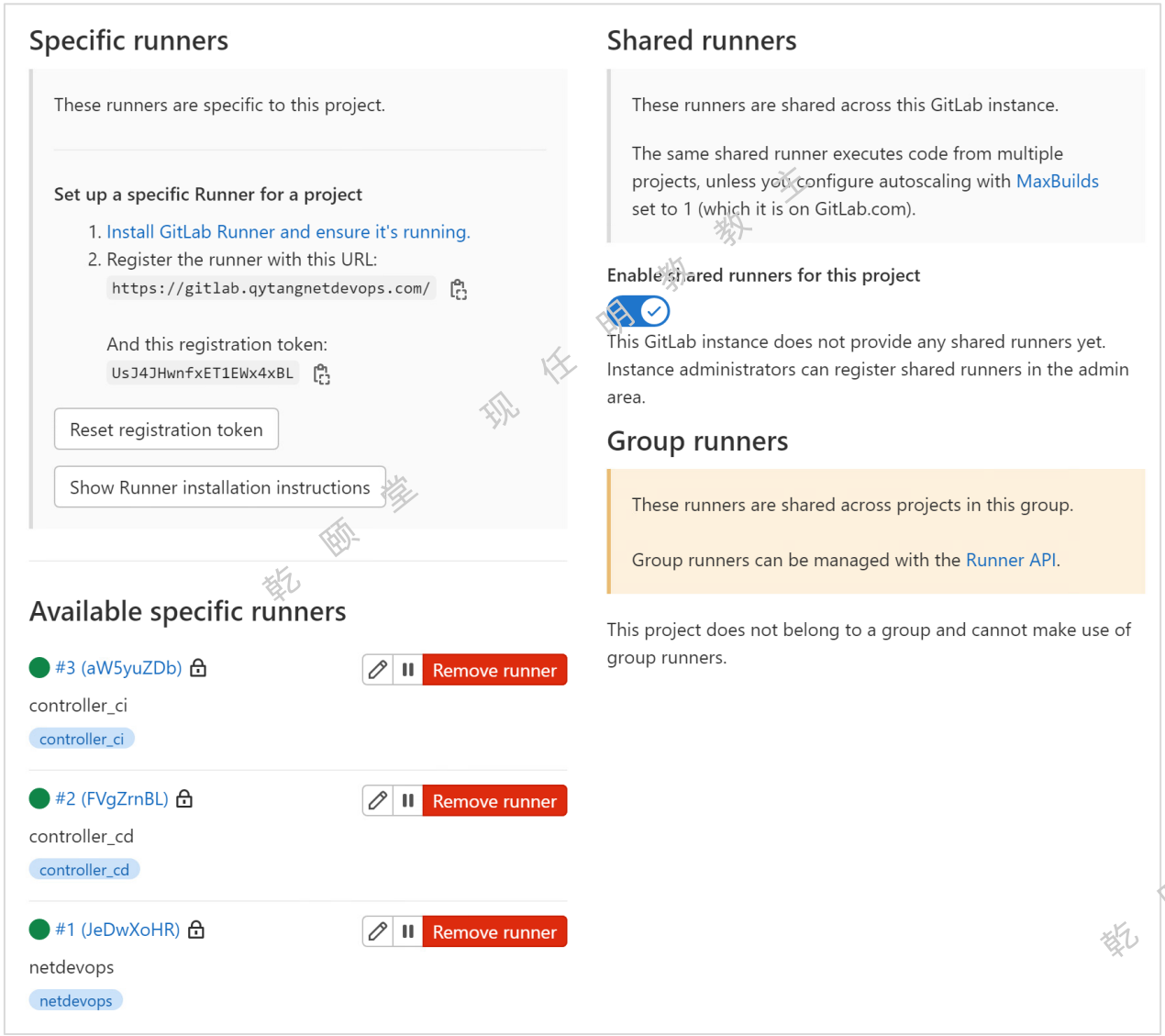
pyATS library

pyATS test framework

分析从设备上收到的数据，结构化网络设备的配置

2. 环境介绍

Gitlab三个Runner



位于Controller_CI
用于执行pyats在脚本

位于Controller_CD
用于执行pyats在脚本

位于Gitlab
用于执行RESTCONF请求

设置了变量

Variables

Collapse

Variables store information, like passwords and secret keys, that you can use in job scripts. Learn more.

Variables can be:

- `Protected:` Only exposed to protected branches or tags.
- `Masked:` Hidden in job logs. Must match masking requirements. Learn more.

Environment variables are configured by your administrator to be protected by default.

Type	Key	Value	Protected	Masked	Environments
Variable	cd_nso_restconf_base_url	**********	✓	✗	All (default)
Variable	ci_nso_restconf_base_url	**********	✓	✗	All (default)
Variable	nso_password	**********	✓	✗	All (default)
Variable	nso_username	**********	✓	✗	All (default)

https://controller_cd.qytangnetdevops.com:8888/restconf/data/

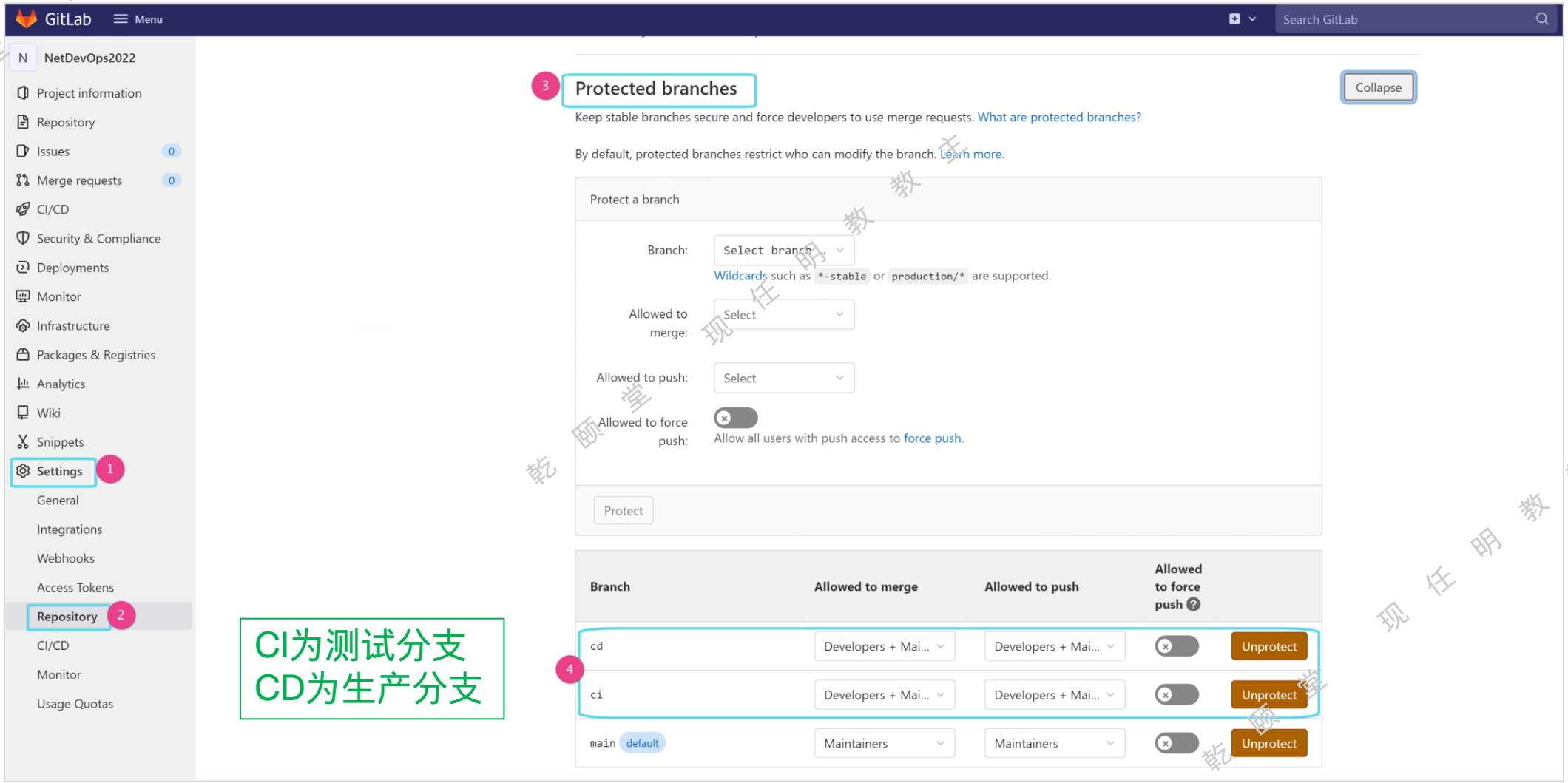
https://controller_ci.qytangnetdevops.com:8888/restconf/data/

Cisc0123

ncsuser

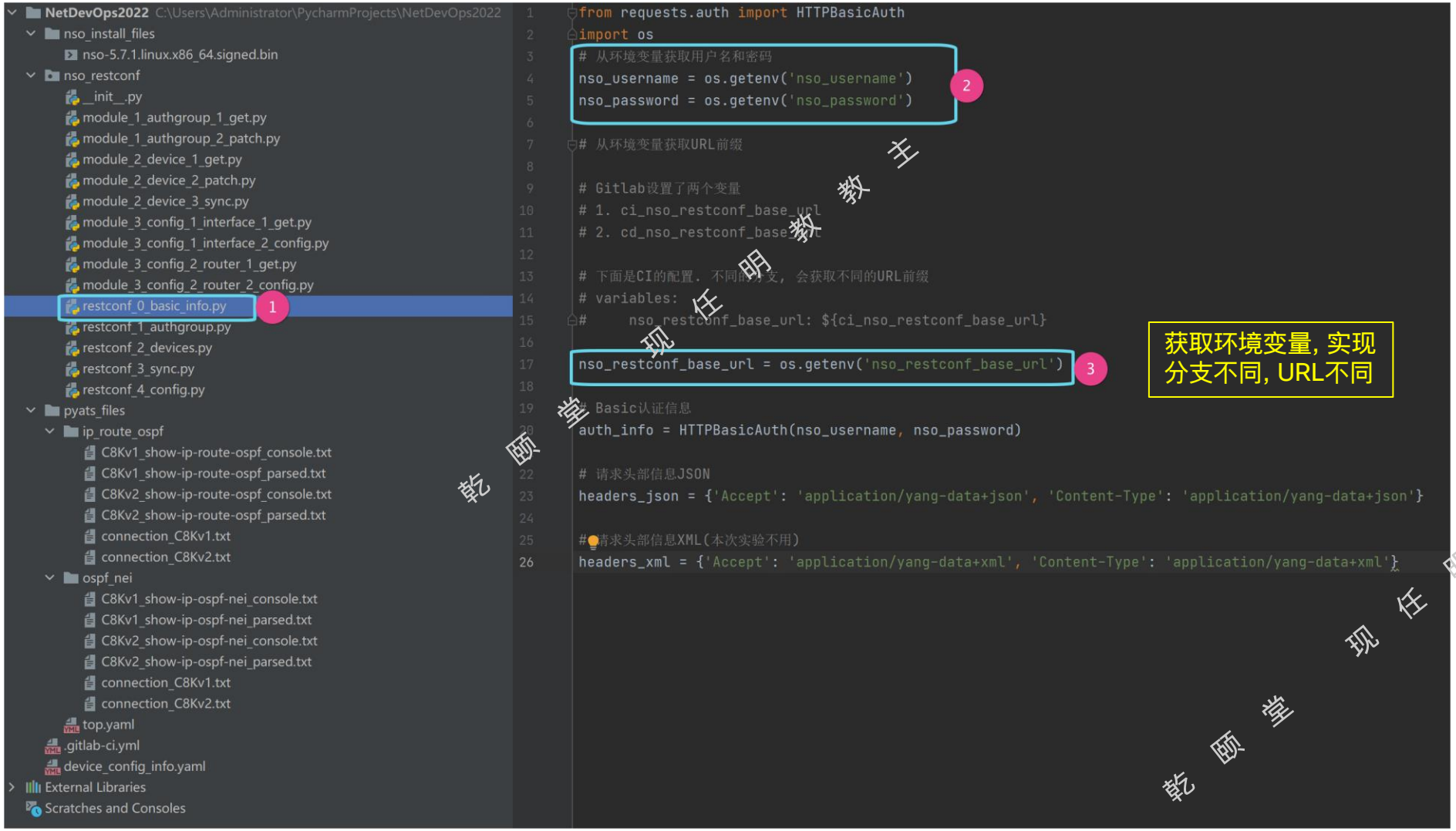
Add variable | Reveal values

ci和cd两个分支都是保护分支

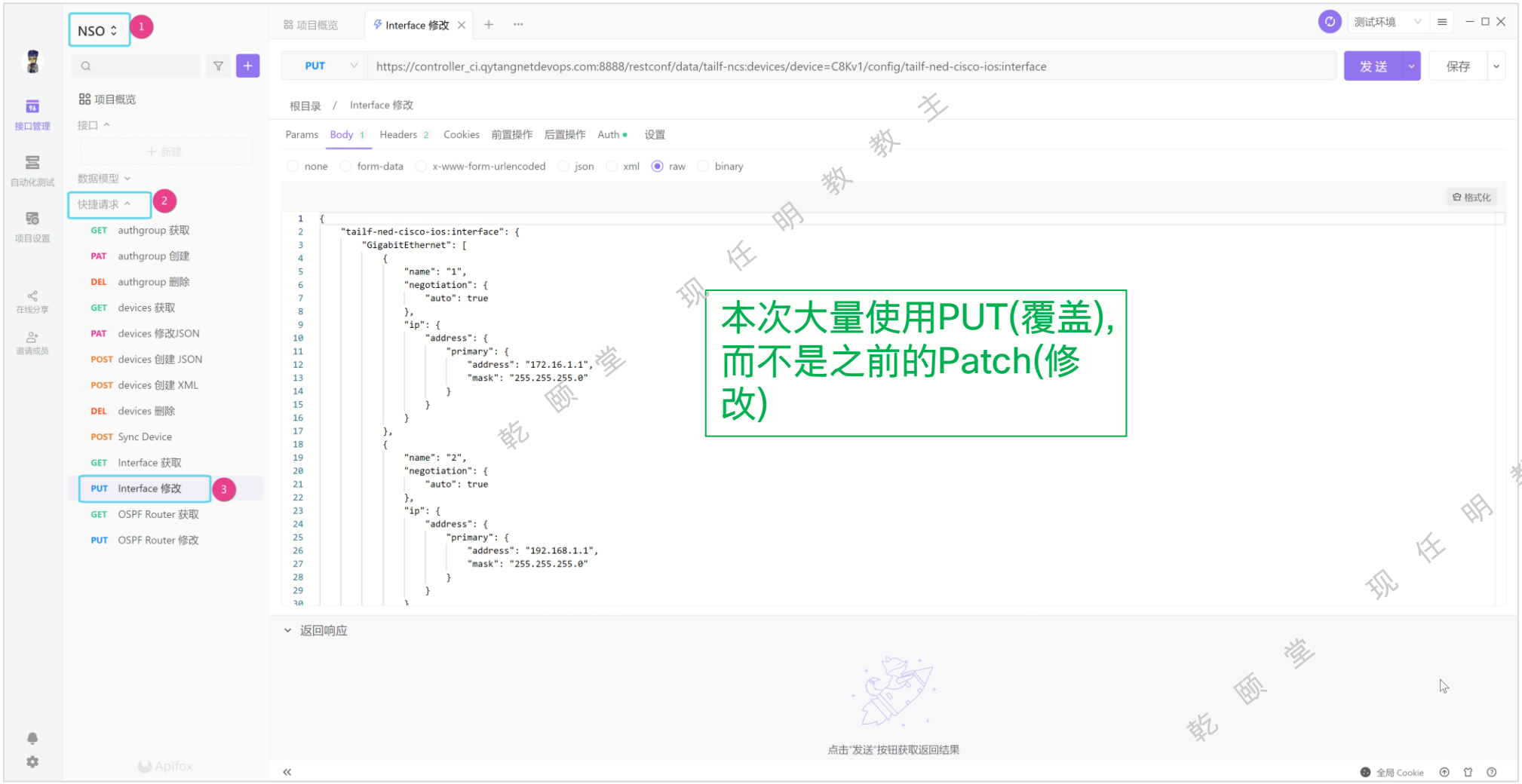


CI为测试分支
CD为生产分支

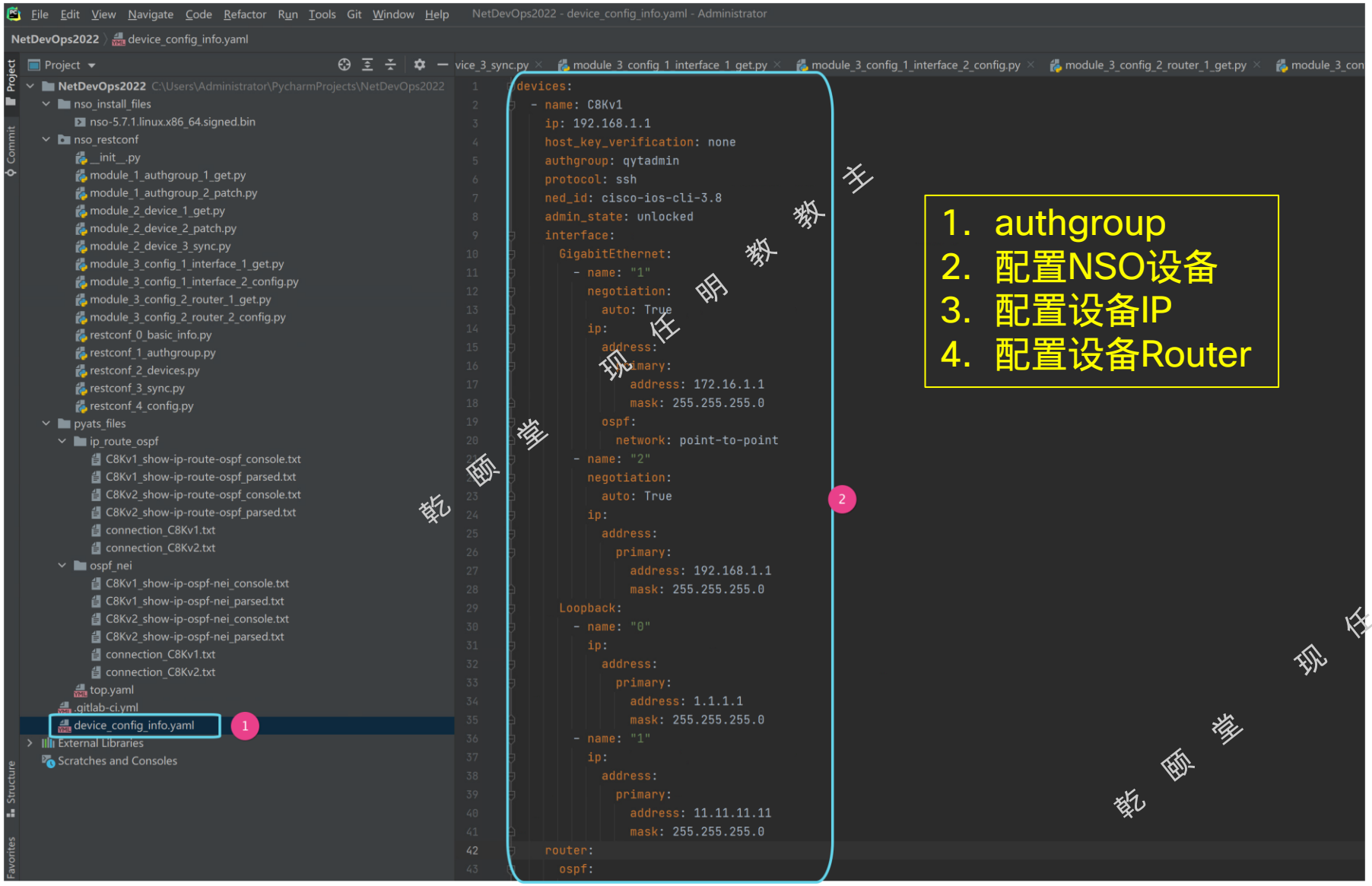
restconf_0_basic_info.py



提前对所有的RESTCONF请求做了测试

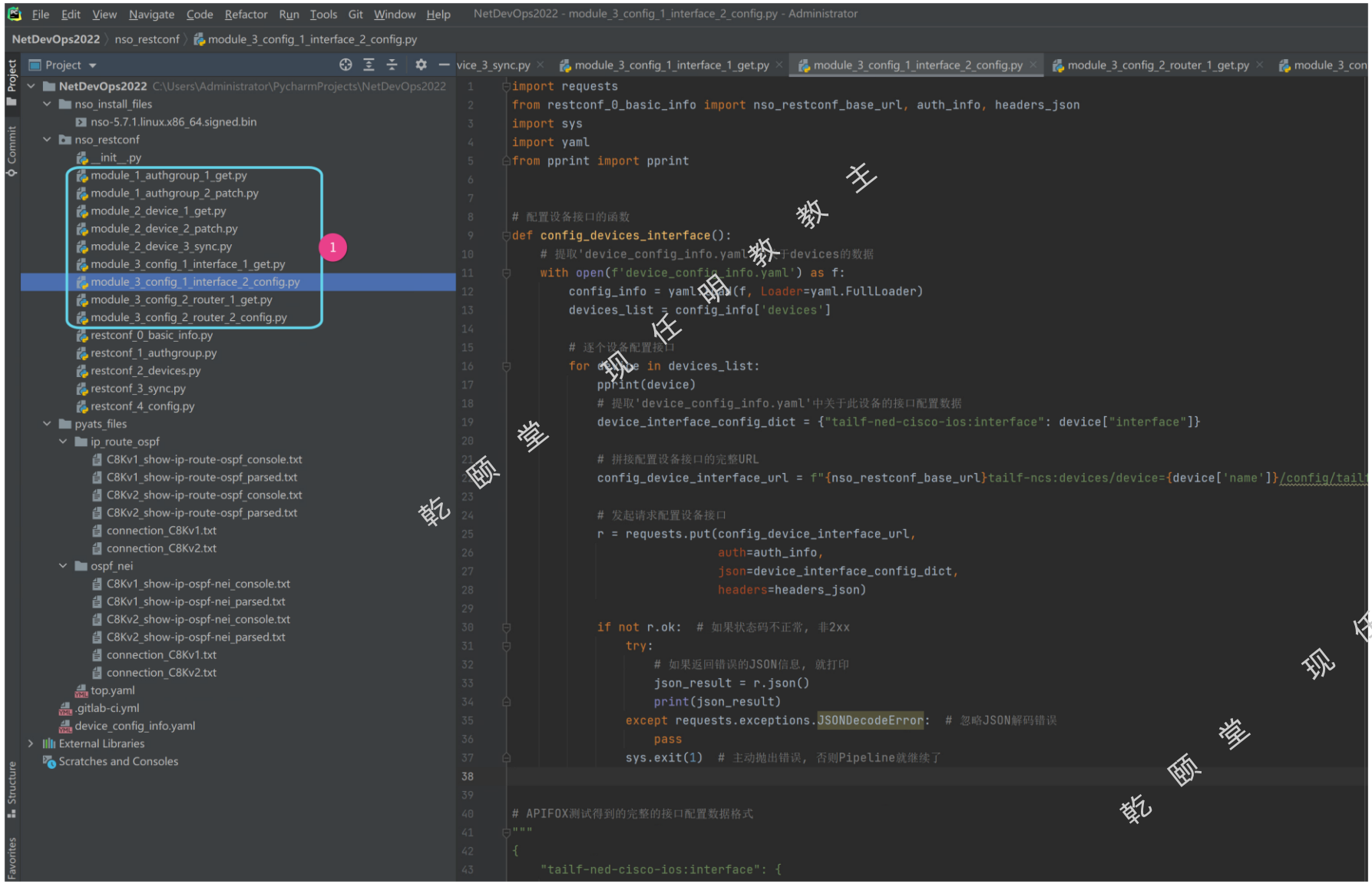


设备配置数据

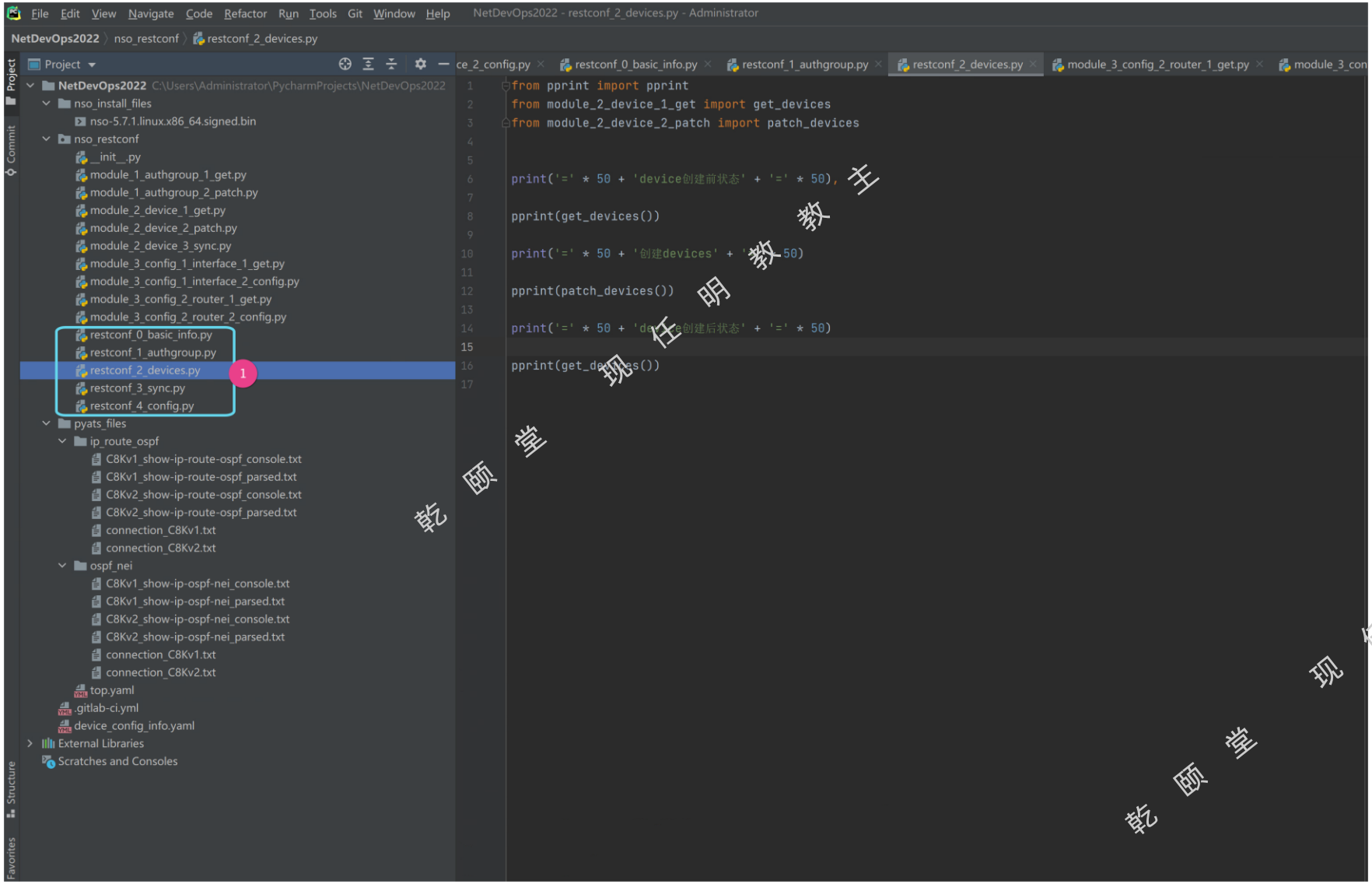


1. authgroup
2. 配置NSO设备
3. 配置设备IP
4. 配置设备Router

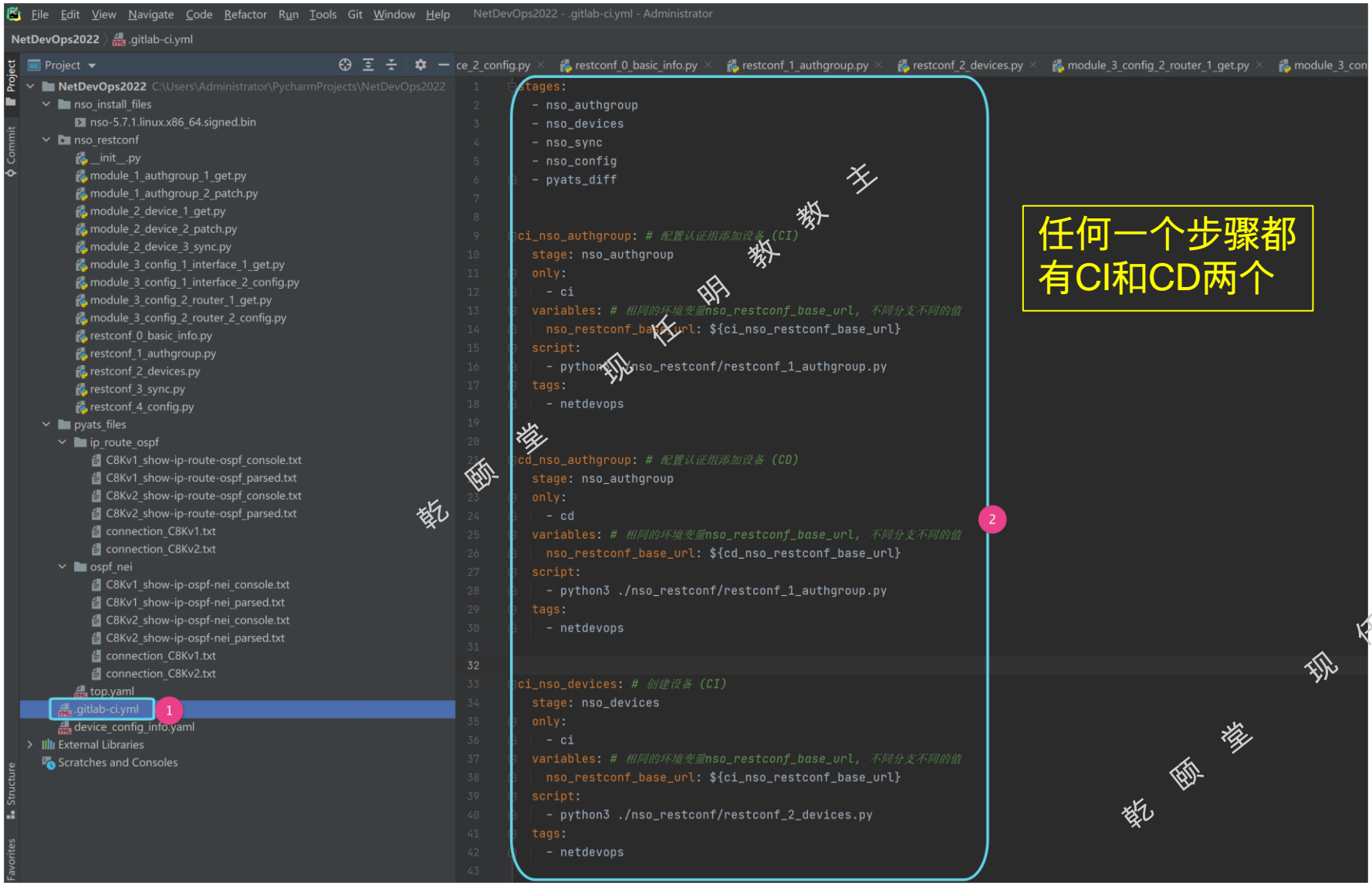
配置模块(功能拆分)


File Edit View Navigate Code Refactor Run Tools Git Window Help
NetDevOps2022 - module_3_config_1_interface_2_config.py - Administrator
NetDevOps2022 nso_restconf module_3_config_1_interface_2_config.py
Project
NetDevOps2022 C:\Users\Administrator\PycharmProjects\NetDevOps2022
nso_install_files
nso-5.7.1.linux.x86_64.signed.bin
nso_restconf
__init__.py
module_1_authgroup_1_get.py
module_1_authgroup_2_patch.py
module_2_device_1_get.py
module_2_device_2_patch.py
module_2_device_3_sync.py
module_3_config_1_interface_1_get.py
module_3_config_1_interface_2_config.py
module_3_config_2_router_1_get.py
module_3_config_2_router_2_config.py
restconf_0_basic_info.py
restconf_1_authgroup.py
restconf_2_devices.py
restconf_3_sync.py
restconf_4_config.py
pyats_files
ip_route_ospf
C8Kv1_show-ip-route-ospf_console.txt
C8Kv1_show-ip-route-ospf_parsed.txt
C8Kv2_show-ip-route-ospf_console.txt
C8Kv2_show-ip-route-ospf_parsed.txt
connection_C8Kv1.txt
connection_C8Kv2.txt
ospf_nei
C8Kv1_show-ip-ospf-nei_console.txt
C8Kv1_show-ip-ospf-nei_parsed.txt
C8Kv2_show-ip-ospf-nei_console.txt
C8Kv2_show-ip-ospf-nei_parsed.txt
connection_C8Kv1.txt
connection_C8Kv2.txt
top.yaml
.gitlab-ci.yml
device_config_info.yaml
External Libraries
Scratches and Consoles
1
vice_3_sync.py
module_3_config_1_interface_1_get.py
module_3_config_1_interface_2_config.py
module_3_config_2_router_1_get.py
import requests
from restconf_0_basic_info import nso_restconf_base_url, auth_info, headers_json
import sys
import yaml
from pprint import pprint
# 配置设备接口的函数
def config_devices_interface():
# 提取'device_config_info.yaml'中关于devices的数据
with open(f'device_config_info.yaml') as f:
config_info = yaml.load(f, Loader=yaml.FullLoader)
devices_list = config_info['devices']
# 逐个设备配置接口
for device in devices_list:
pprint(device)
# 提取'device_config_info.yaml'中关于此设备的接口配置数据
device_interface_config_dict = {"tailf-ned-cisco-ios:interface": device["interface"]}
# 拼接配置设备接口的完整URL
config_device_interface_url = f"{nso_restconf_base_url}tailf-ncs:devices/device={device['name']}/config/tail
# 发起请求配置设备接口
r = requests.put(config_device_interface_url,
auth=auth_info,
json=device_interface_config_dict,
headers=headers_json)
if not r.ok: # 如果状态码不正常, 非2xx
try:
# 如果返回错误的JSON信息, 就打印
json_result = r.json()
print(json_result)
except requests.exceptions.JSONDecodeError: # 忽略JSON解码错误
pass
sys.exit(1) # 主动抛出错误, 否则Pipeline就继续了
APIFOX测试得到的完整的接口配置数据格式
"""
{
"tailf-ned-cisco-ios:interface": {

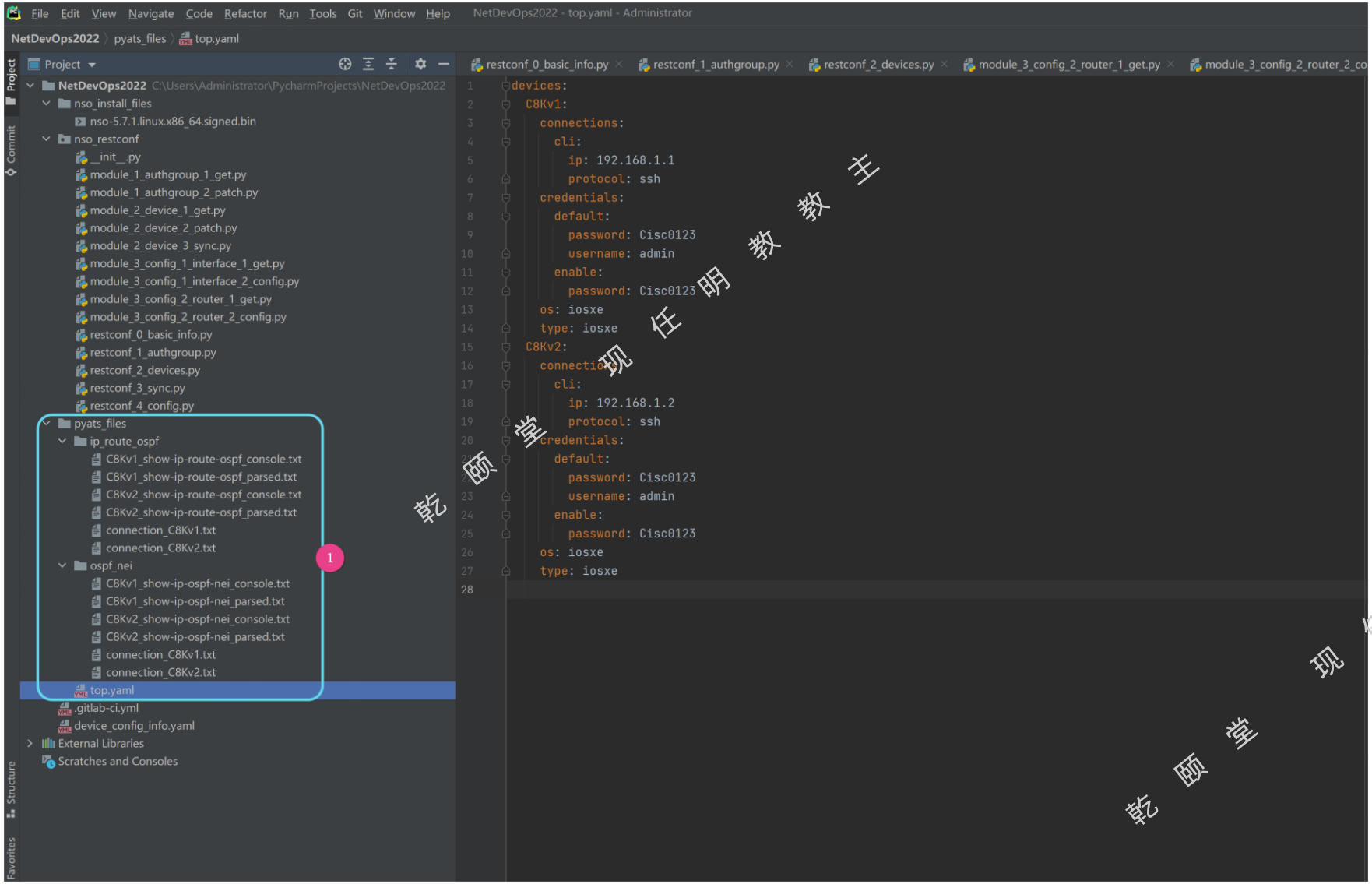
任务汇总模块


File Edit View Navigate Code Refactor Run Tools Git Window Help
NetDevOps2022 - restconf_2_devices.py - Administrator
NetDevOps2022 nso_restconf restconf_2_devices.py
Project
NetDevOps2022 C:\Users\Administrator\PycharmProjects\NetDevOps2022
nso_install_files
nso-5.7.1.linux.x86_64.signed.bin
nso_restconf
__init__.py
module_1_authgroup_1_get.py
module_1_authgroup_2_patch.py
module_2_device_1_get.py
module_2_device_2_patch.py
module_2_device_3_sync.py
module_3_config_1_interface_1_get.py
module_3_config_1_interface_2_config.py
module_3_config_2_router_1_get.py
module_3_config_2_router_2_config.py
restconf_0_basic_info.py
restconf_1_authgroup.py
restconf_2_devices.py
restconf_3_sync.py
restconf_4_config.py
pyats_files
ip_route_ospf
C8Kv1_show-ip-route-ospf_console.txt
C8Kv1_show-ip-route-ospf_parsed.txt
C8Kv2_show-ip-route-ospf_console.txt
C8Kv2_show-ip-route-ospf_parsed.txt
connection_C8Kv1.txt
connection_C8Kv2.txt
ospf_nei
C8Kv1_show-ip-ospf-nei_console.txt
C8Kv1_show-ip-ospf-nei_parsed.txt
C8Kv2_show-ip-ospf-nei_console.txt
C8Kv2_show-ip-ospf-nei_parsed.txt
connection_C8Kv1.txt
connection_C8Kv2.txt
top.yaml
.gitlab-ci.yml
device_config_info.yaml
External Libraries
Scratches and Consoles
1
restconf_0_basic_info.py
restconf_1_authgroup.py
restconf_2_devices.py
module_3_config_2_router_1_get.py
from pprint import pprint
from module_2_device_1_get import get_devices
from module_2_device_2_patch import patch_devices
print('=' * 50 + 'device创建前状态' + '=' * 50)
pprint(get_devices())
print('=' * 50 + '创建devices' + '=' * 50)
pprint(patch_devices())
print('=' * 50 + 'device创建后状态' + '=' * 50)
pprint(get_devices())

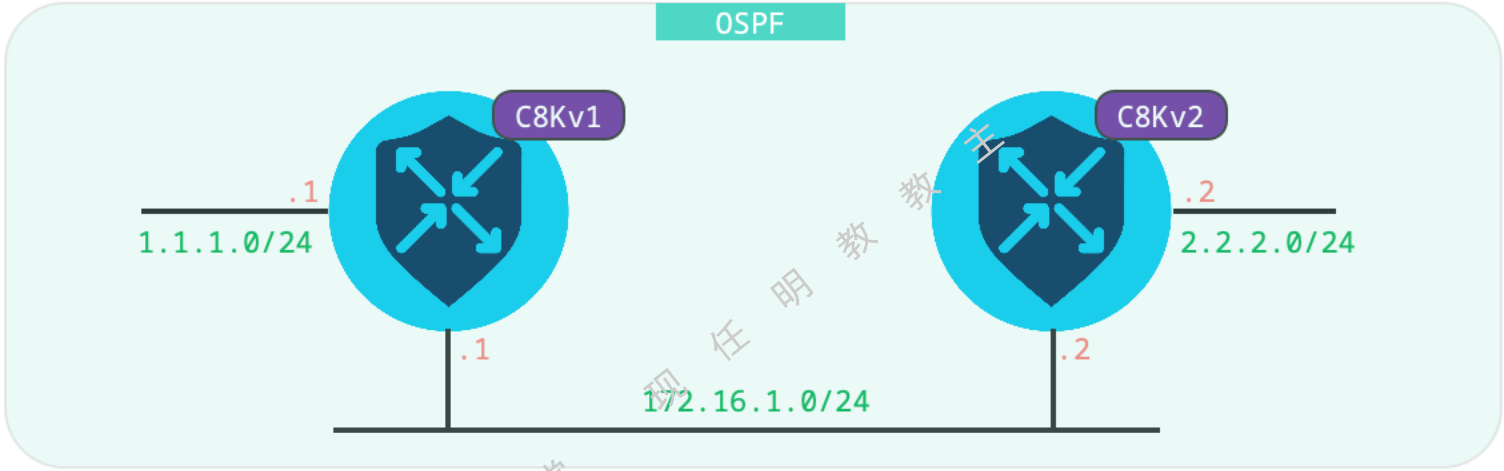
Gitlab Pipeline配置文件



pyATS部分


File Edit View Navigate Code Refactor Run Tools Git Window Help
NetDevOps2022 - top.yaml - Administrator
NetDevOps2022 pyats_files top.yaml
Project
NetDevOps2022 C:\Users\Administrator\PycharmProjects\NetDevOps2022
nso_install_files
nso-5.7.1.linux.x86_64.signed.bin
nso_restconf
__init__.py
module_1_authgroup_1_get.py
module_1_authgroup_2_patch.py
module_2_device_1_get.py
module_2_device_2_patch.py
module_2_device_3_sync.py
module_3_config_1_interface_1_get.py
module_3_config_1_interface_2_config.py
module_3_config_2_router_1_get.py
module_3_config_2_router_2_config.py
restconf_0_basic_info.py
restconf_1_authgroup.py
restconf_2_devices.py
restconf_3_sync.py
restconf_4_config.py
pyats_files
ip_route_ospf
C8Kv1_show-ip-route-ospf_console.txt
C8Kv1_show-ip-route-ospf_parsed.txt
C8Kv2_show-ip-route-ospf_console.txt
C8Kv2_show-ip-route-ospf_parsed.txt
connection_C8Kv1.txt
connection_C8Kv2.txt
ospf_nei
C8Kv1_show-ip-ospf-nei_console.txt
C8Kv1_show-ip-ospf-nei_parsed.txt
C8Kv2_show-ip-ospf-nei_console.txt
C8Kv2_show-ip-ospf-nei_parsed.txt
connection_C8Kv1.txt
connection_C8Kv2.txt
top.yaml
.gitlab-ci.yml
device_config_info.yaml
External Libraries
Scratches and Consoles
restconf_0_basic_info.py
restconf_1_authgroup.py
restconf_2_devices.py
module_3_config_2_router_1_get.py
devices:
C8Kv1:
connections:
cli:
ip: 192.168.1.1
protocol: ssh
credentials:
default:
password: Cisc0123
username: admin
enable:
password: Cisc0123
os: iosxe
type: iosxe
C8Kv2:
connections:
cli:
ip: 192.168.1.2
protocol: ssh
credentials:
default:
password: Cisc0123
username: admin
enable:
password: Cisc0123
os: iosxe
type: iosxe

pyATS 快照

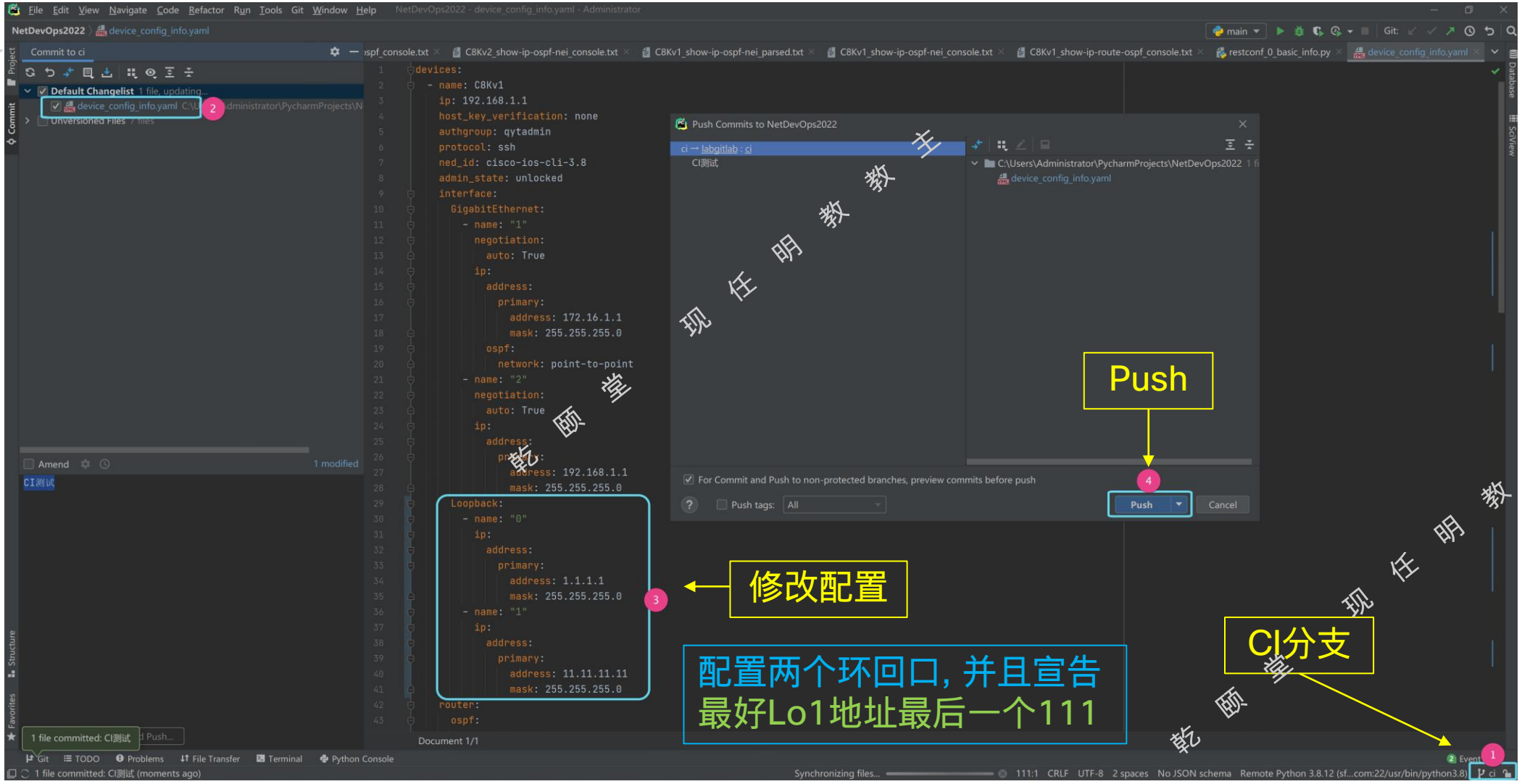



```
show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           0   FULL/  -        00:00:38    172.16.1.2      GigabitEthernet1
```

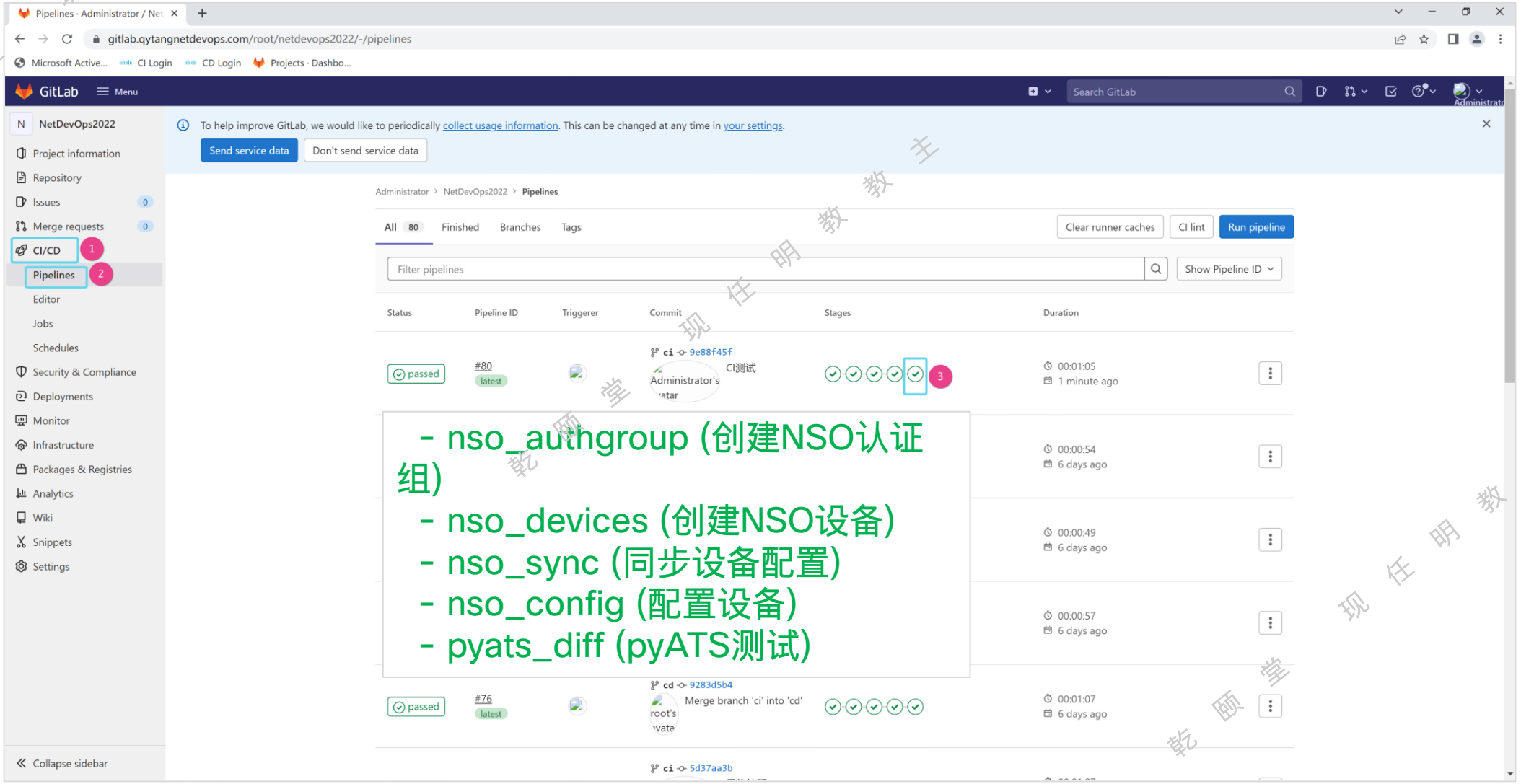
```
show ip route ospf
      2.0.0.0/32 is subnetted, 1 subnets
O        2.2.2.2 [110/2] via 172.16.1.2, 00:03:03, GigabitEthernet1
```

3. 测试

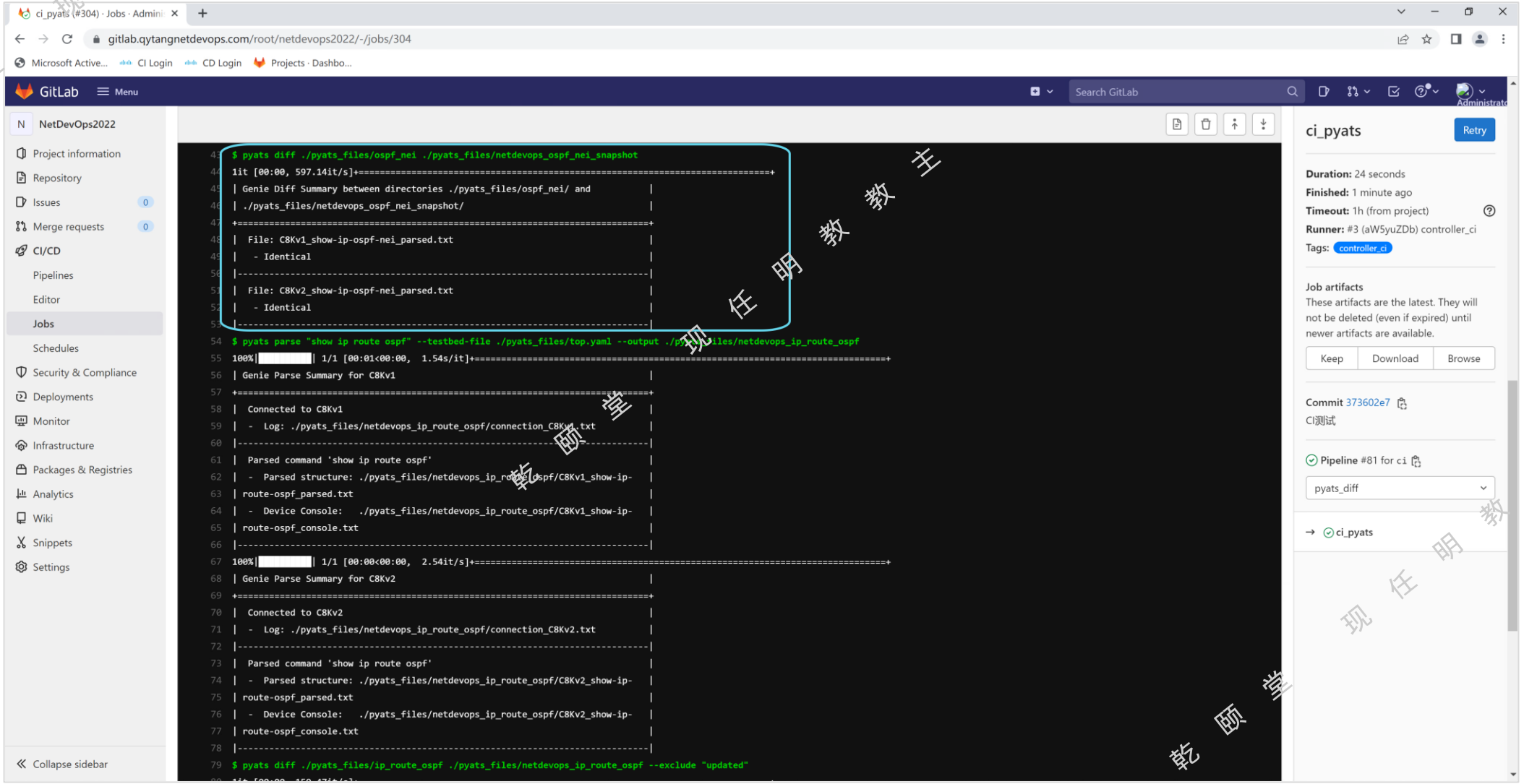
CI分修改配置然后Push



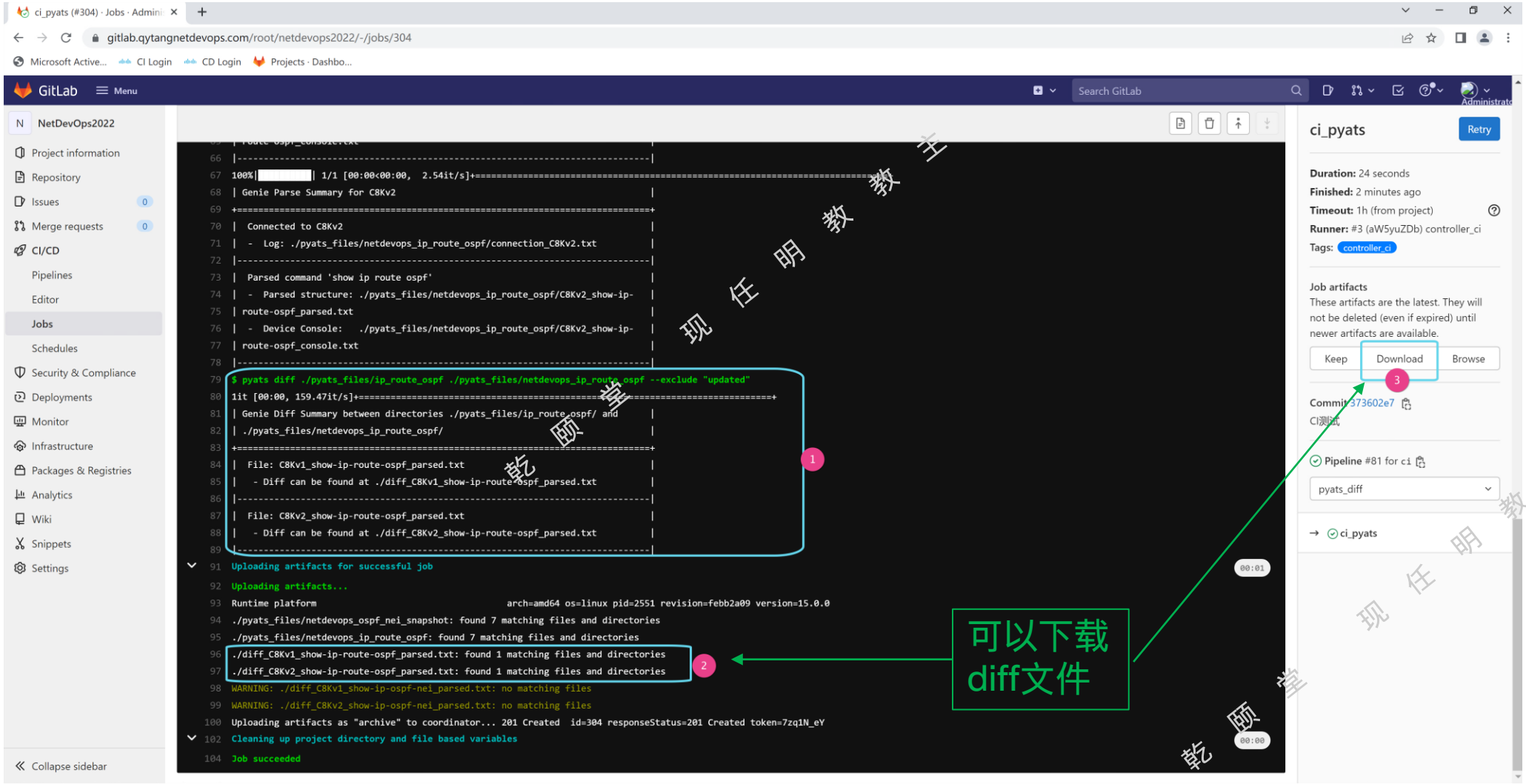
查看Pipeline状态



通过pyATS看到邻居是一致的



通过pyATS看到路由不一致

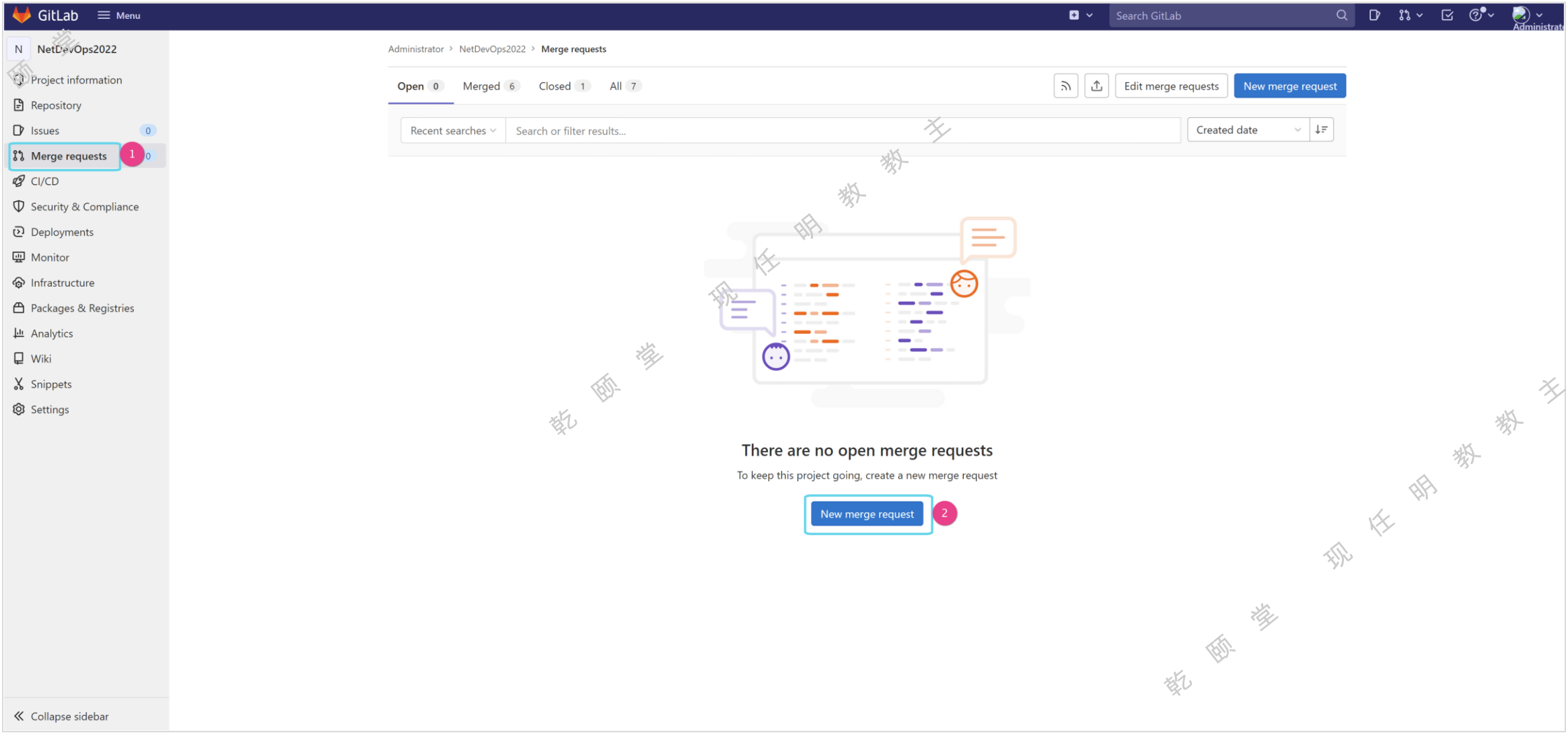


查看Diff文件

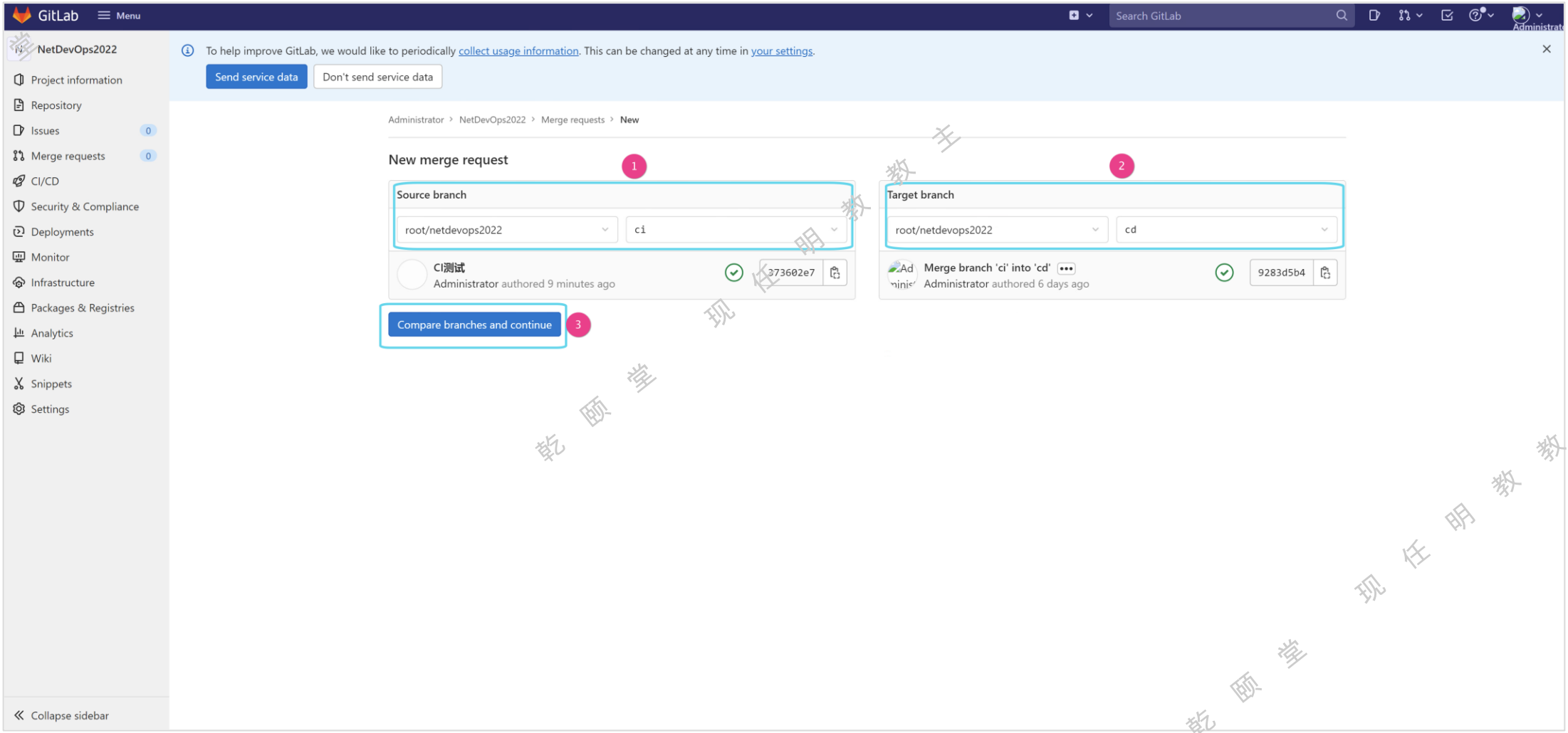
```yaml
cd_pyats:  # PYATS 比较 (CD)
  stage: pyats_diff
  only:
    - cd
  script:
    # 产生show ip ospf nei命令执行的快照
    - pyats parse "show ip ospf nei" --testbed-file ./pyats_files/top.yaml --output ./pyats_files/netdevops_ospf_nei_snapshot
    # 使用PYATS比较上一步产生的快照与项目中备份快照的区别
    - pyats diff ./pyats_files/ospf_nei ./pyats_files/netdevops_ospf_nei_snapshot
    # 产生show ip route ospf命令执行的快照
    - pyats parse "show ip route ospf" --testbed-file ./pyats_files/top.yaml --output ./pyats_files/netdevops_ip_route_ospf
    # 使用PYATS比较上一步产生的快照与项目中备份快照的区别，严重注意需要排除掉 "updated"
    - pyats diff ./pyats_files/ip_route_ospf ./pyats_files/netdevops_ip_route_ospf --exclude "updated"

  artifacts:  # 如果PYATS比较出差异，使用artifacts把比较产生的文件备份出来
    name: pyats_diff_result
    paths:
      - ./pyats_files/netdevops_ospf_nei_snapshot
      - ./pyats_files/netdevops_ip_route_ospf
      - ./diff_C8Kv1_show-ip-route-ospf_parsed.txt
      - ./diff_C8Kv2_show-ip-route-ospf_parsed.txt
      - ./diff_C8Kv1_show-ip-ospf-nei_parsed.txt
      - ./diff_C8Kv2_show-ip-ospf-nei_parsed.txt
  tags:
    - controller_cd
```

CI配置的artifacts

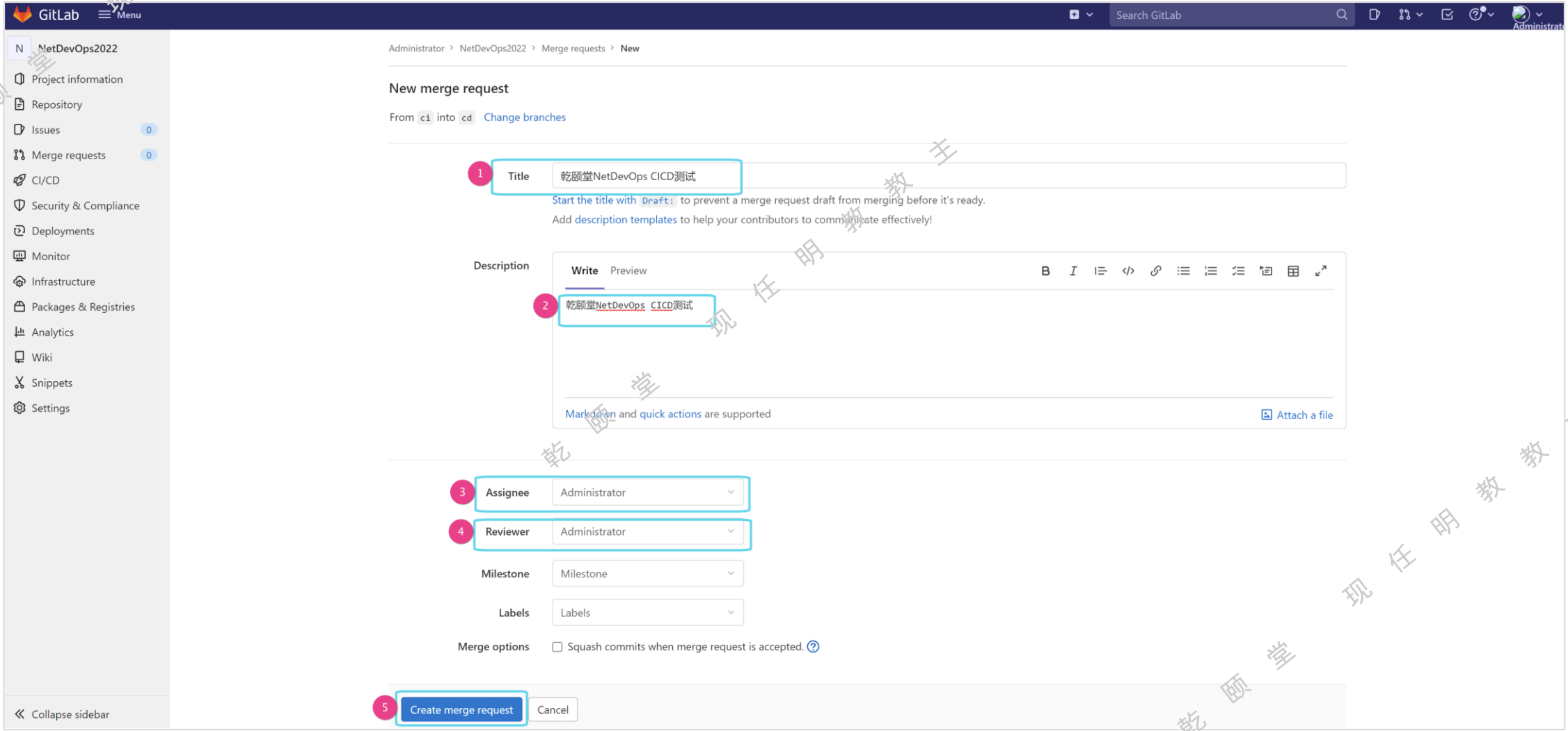
Merge Request



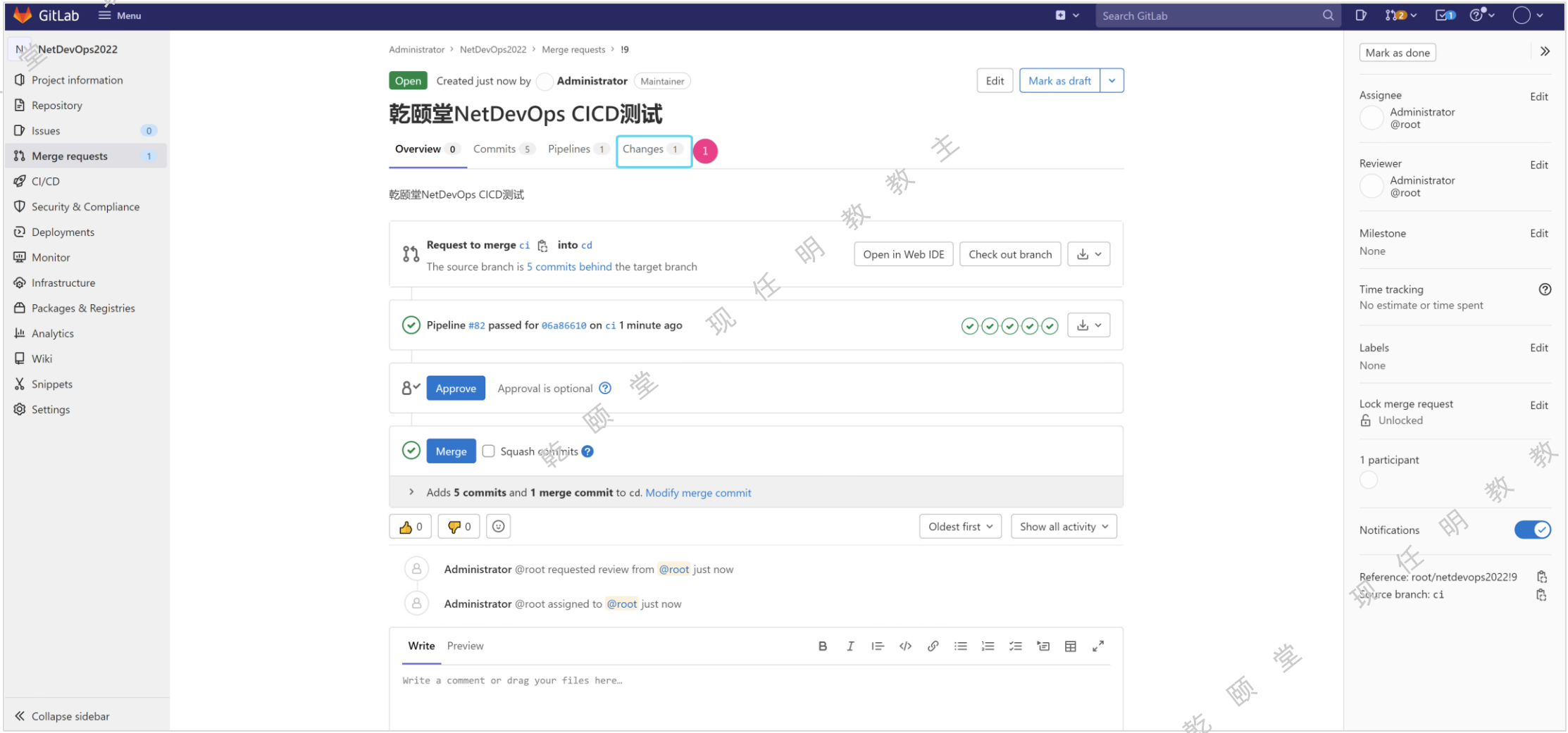
Merge Request



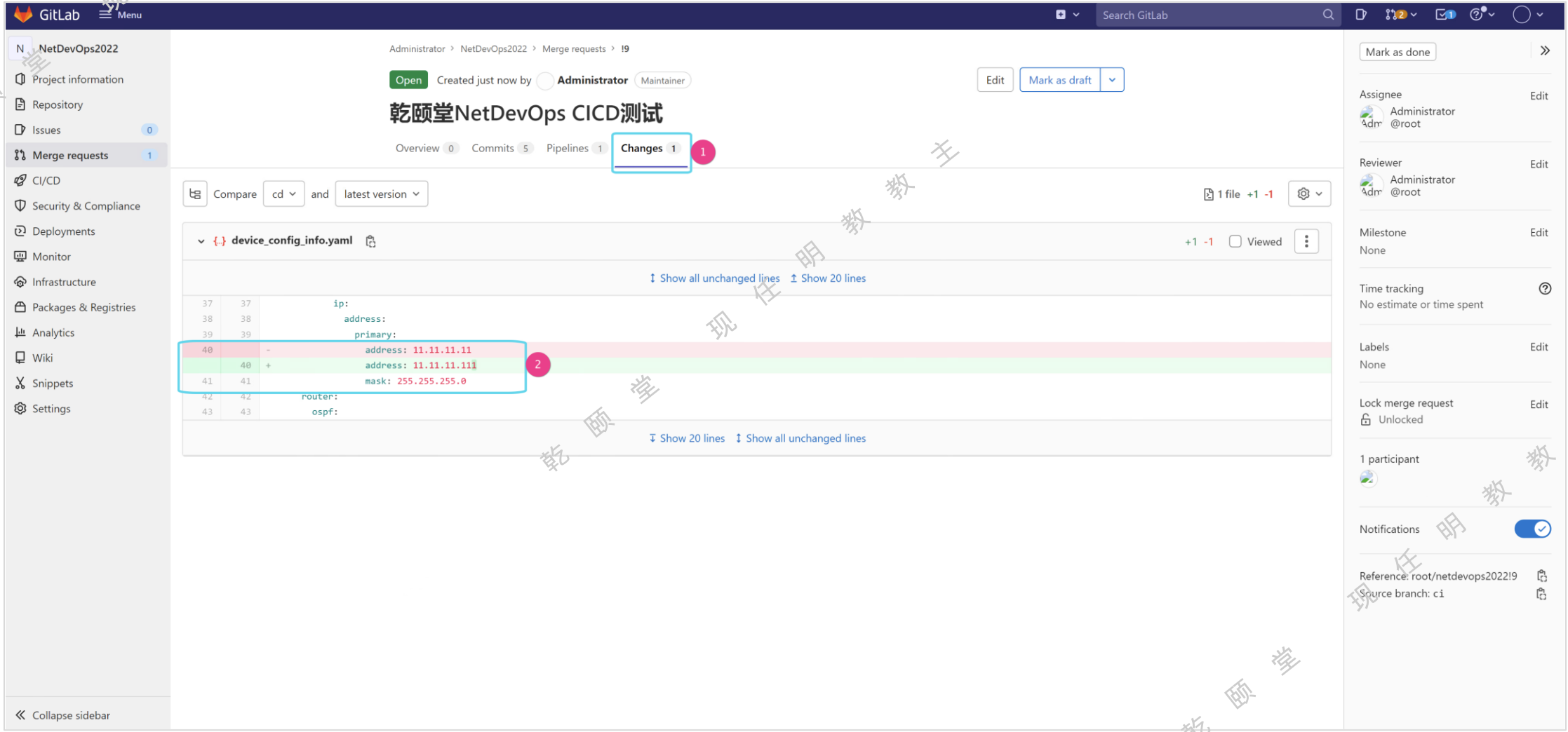
Merge Request



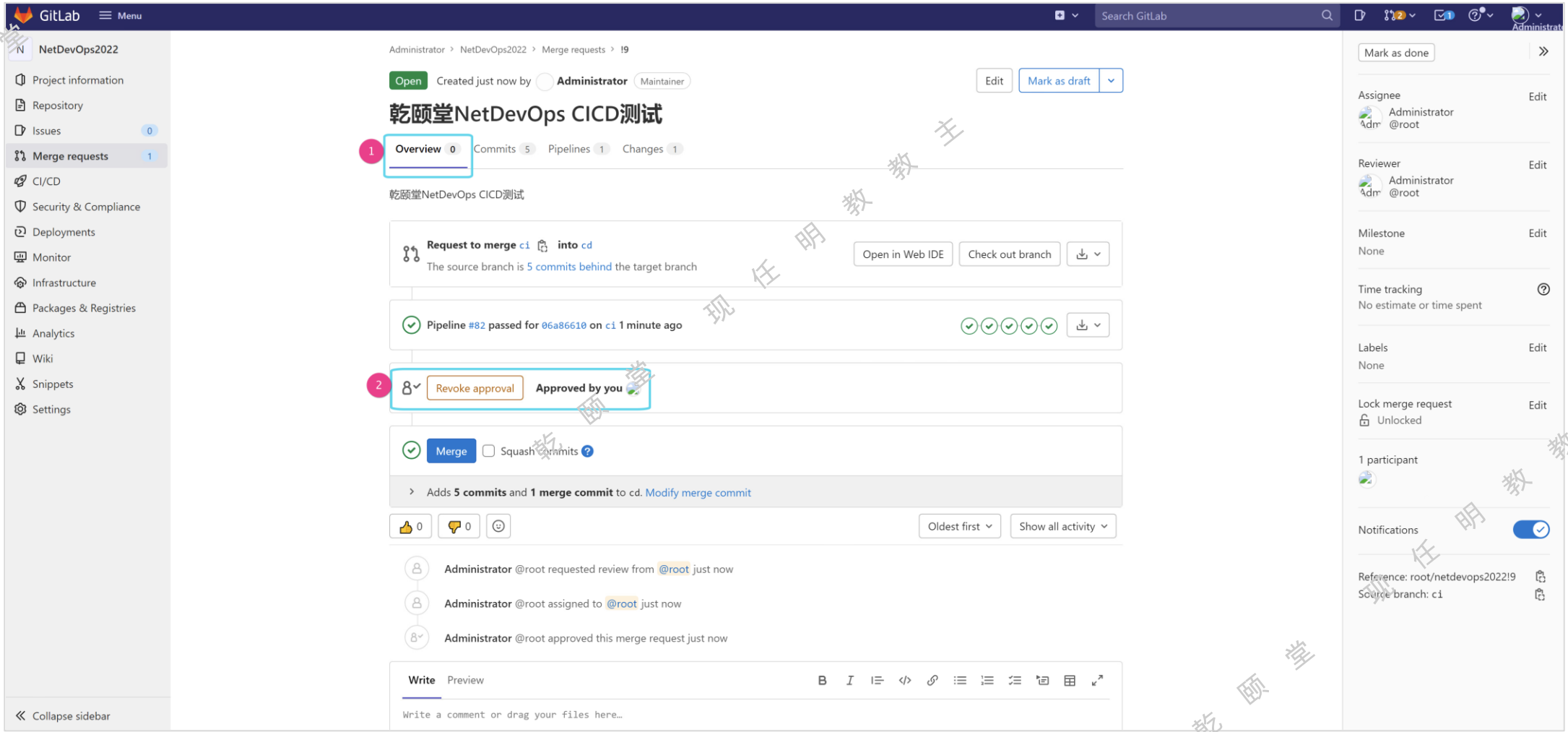
管理员查看change



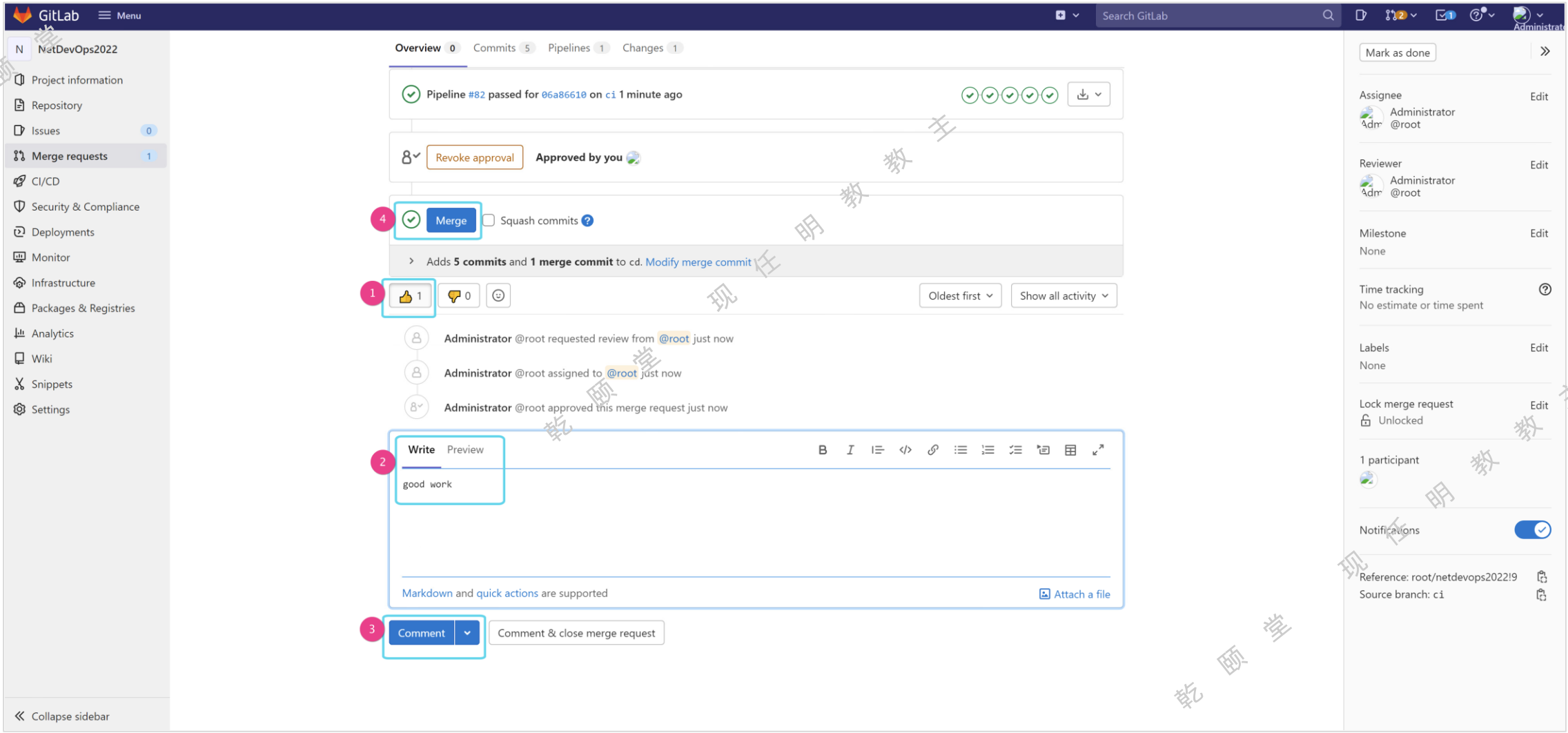
管理员查看change



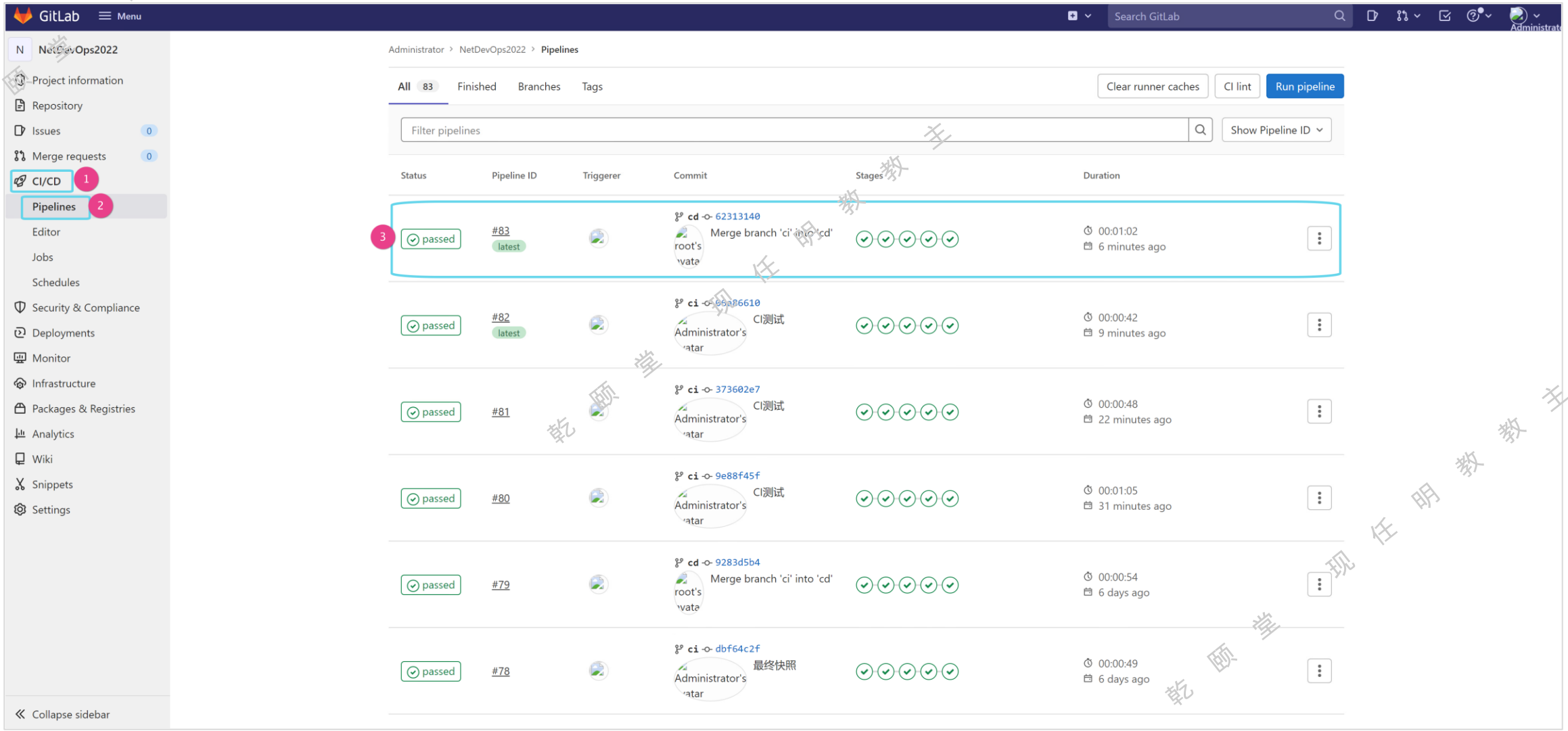
代码审核 Approval



管理员 Merge


GitLab
Menu
Search GitLab
NetDevOps2022
Project information
Repository
Issues 0
Merge requests 1
CI/CD
Security & Compliance
Deployments
Monitor
Infrastructure
Packages & Registries
Analytics
Wiki
Snippets
Settings
Collapse sidebar
Overview 0
Commits 5
Pipelines 1
Changes 1
Pipeline #82 passed for 06a86610 on ci 1 minute ago
Revoke approval
Approved by you
4
Merge
Squash commits
Adds 5 commits and 1 merge commit to cd. Modify merge commit
1
1
0
Oldest first
Show all activity
Administrator @root requested review from @root just now
Administrator @root assigned to @root just now
Administrator @root approved this merge request just now
2
Write
Preview
good work
Markdown and quick actions are supported
Attach a file
3
Comment
Comment & close merge request
Mark as done
Assignee
Edit
Administrator
@root
Reviewer
Edit
Administrator
@root
Milestone
Edit
None
Time tracking
No estimate or time spent
Labels
Edit
None
Lock merge request
Edit
Unlocked
1 participant
Notifications
Reference: root/netdevops2022!9
Source branch: ci

查看Pipeline


GitLab
Menu
Search GitLab
NetDevOps2022
Project information
Repository
Issues 0
Merge requests 0
CI/CD
Pipelines
Editor
Jobs
Schedules
Security & Compliance
Deployments
Monitor
Infrastructure
Packages & Registries
Analytics
Wiki
Snippets
Settings
Collapse sidebar
1
2
3
Administrator › NetDevOps2022 › Pipelines
All 83
Finished
Branches
Tags
Clear runner caches
CI lint
Run pipeline
Filter pipelines
Show Pipeline ID
Status
Pipeline ID
Triggerer
Commit
Stages
Duration
passed
#83
latest
cd 62313140
Merge branch 'ci' into 'cd'
00:01:02
6 minutes ago
passed
#82
latest
ci 66386610
CI测试
00:00:42
9 minutes ago
passed
#81
ci 373602e7
CI测试
00:00:48
22 minutes ago
passed
#80
ci 9e88f45f
CI测试
00:01:05
31 minutes ago
passed
#79
cd 9283d5b4
Merge branch 'ci' into 'cd'
00:00:54
6 days ago
passed
#78
ci dbf64c2f
最终快照
00:00:49
6 days ago