



乾颐堂 现任明教教主

# 18. NetDevOps2022



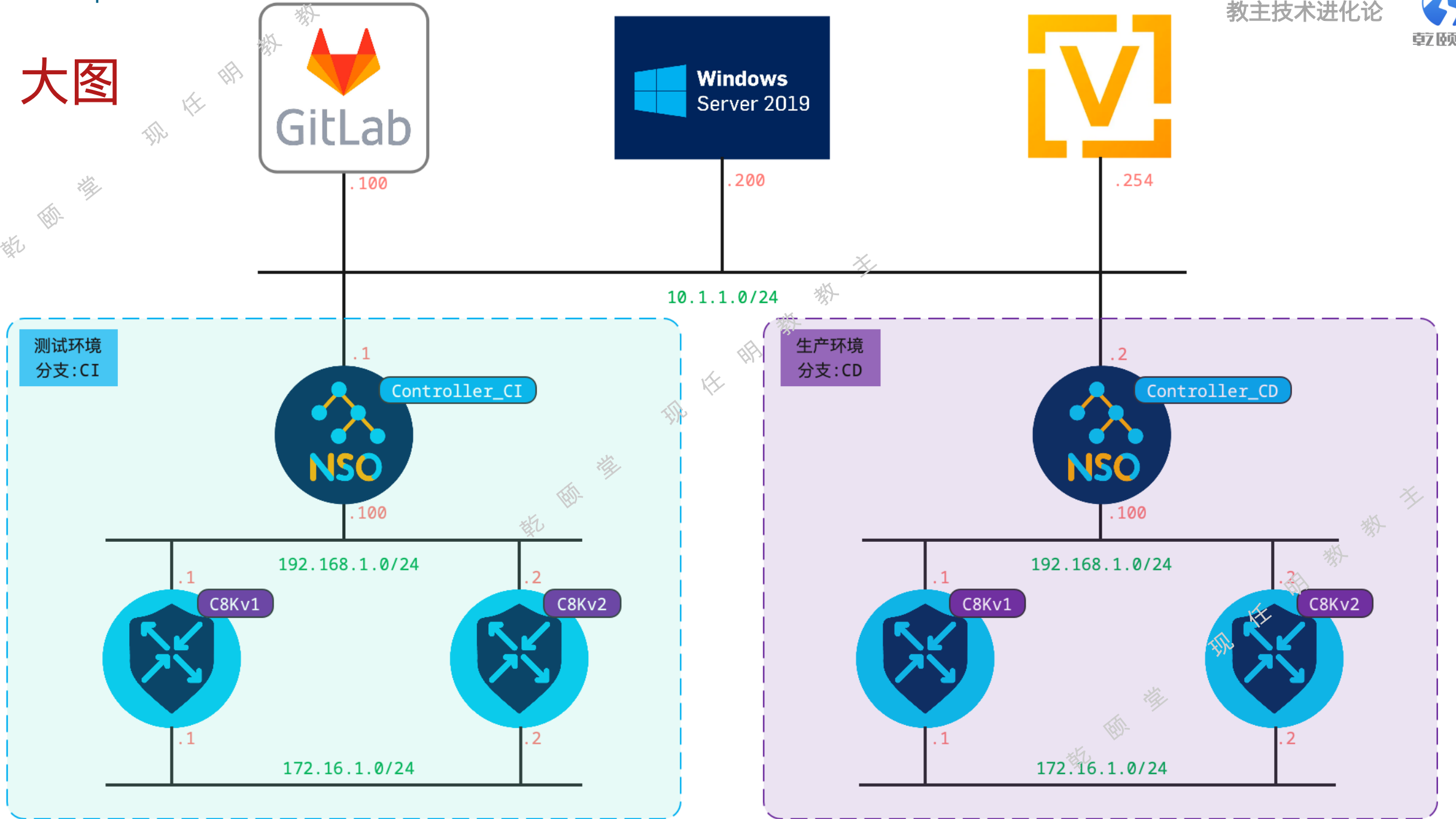
乾颐堂 现任明教教主

教主技术进化论 2022

教主VIP, 聊点高级的!

乾颐堂 现任明教教主

# 大图



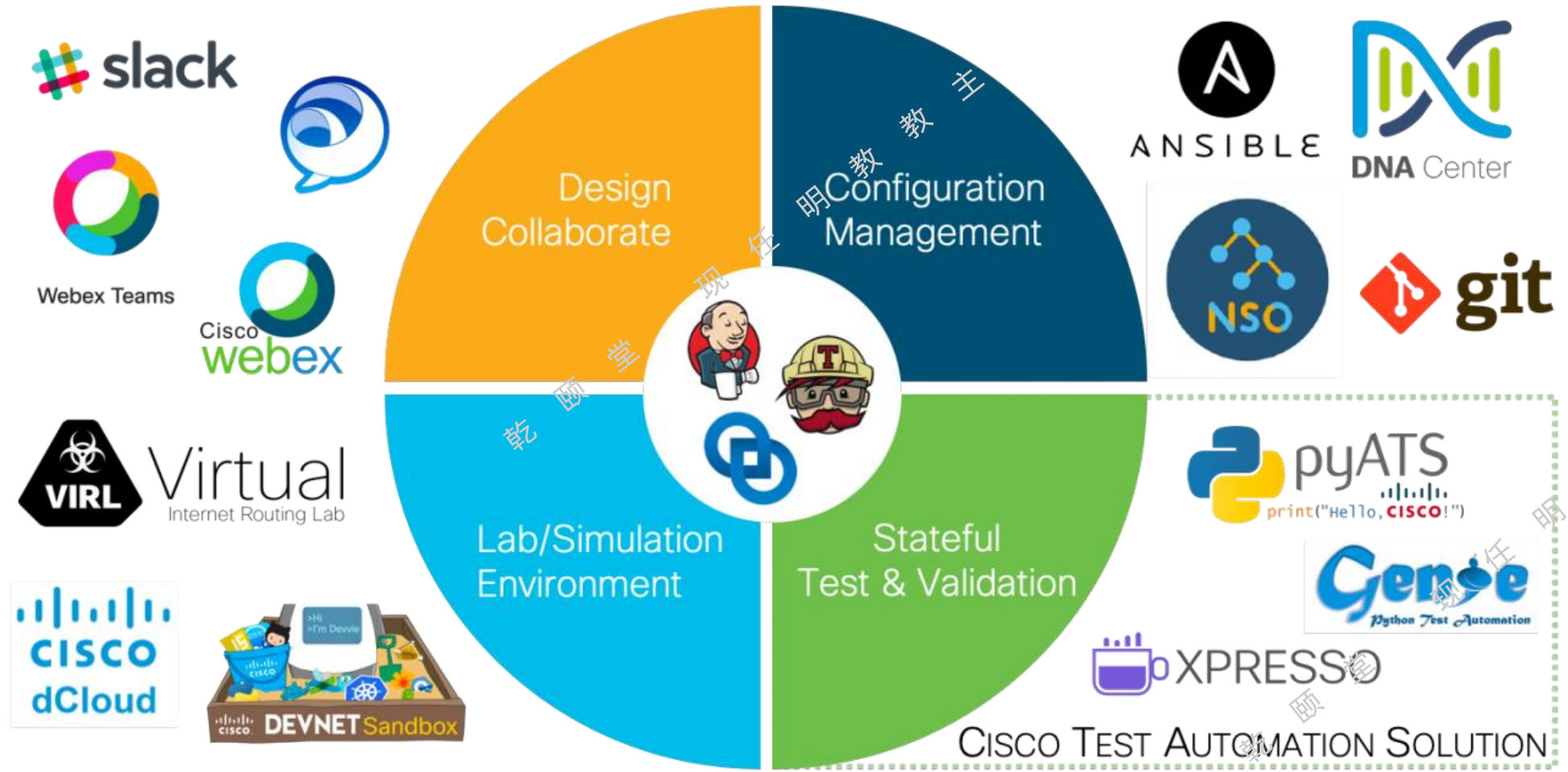
乾颐堂 现任明教教主

乾颐堂 现任明教教主

# 1. NetDevOps

乾颐堂 现任明教教主

# NetDevOps简介

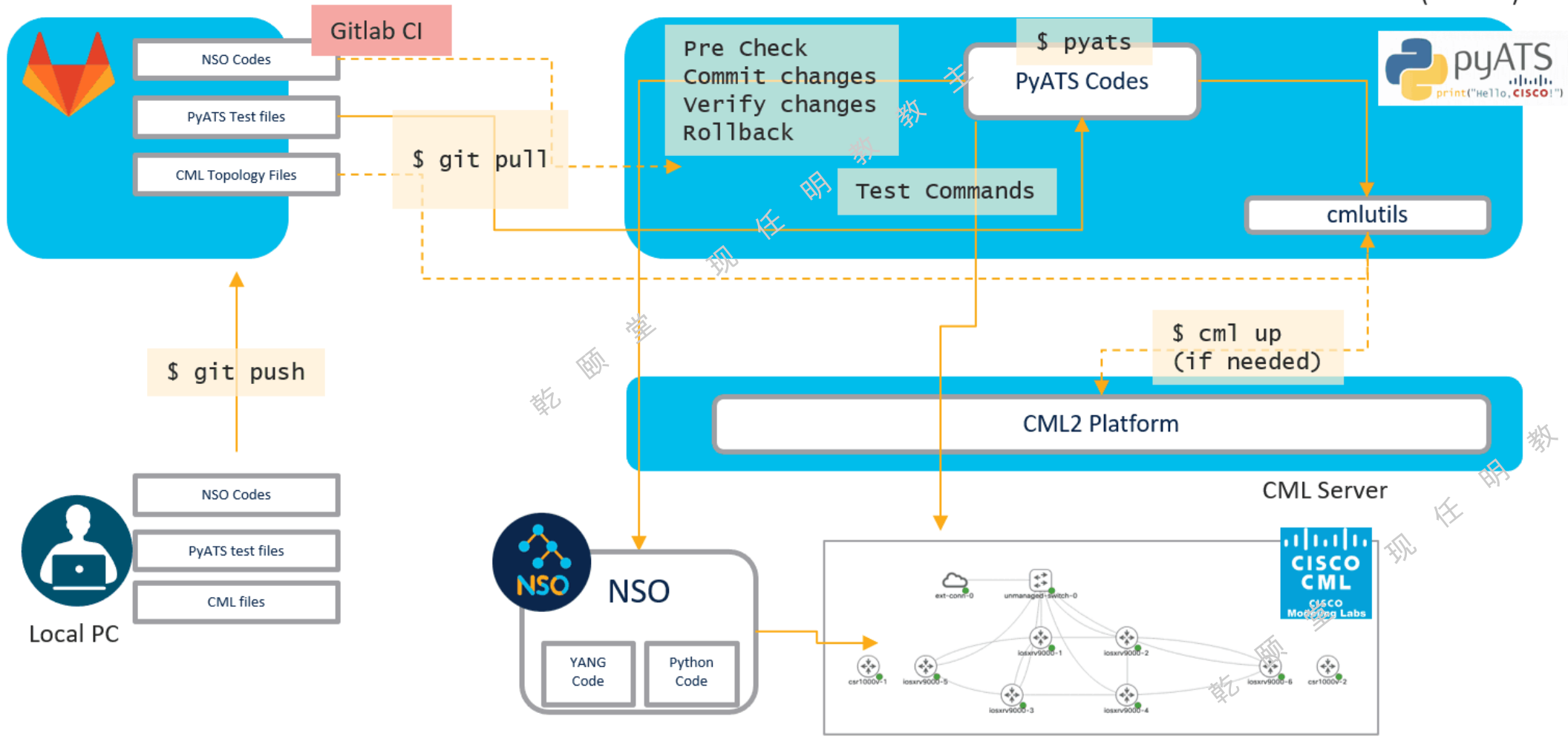


# NetDevOps简介

乾颐堂 现任明教教主

Gitlab Internal Cloud  
(gitlab-sjc.cisco.com)

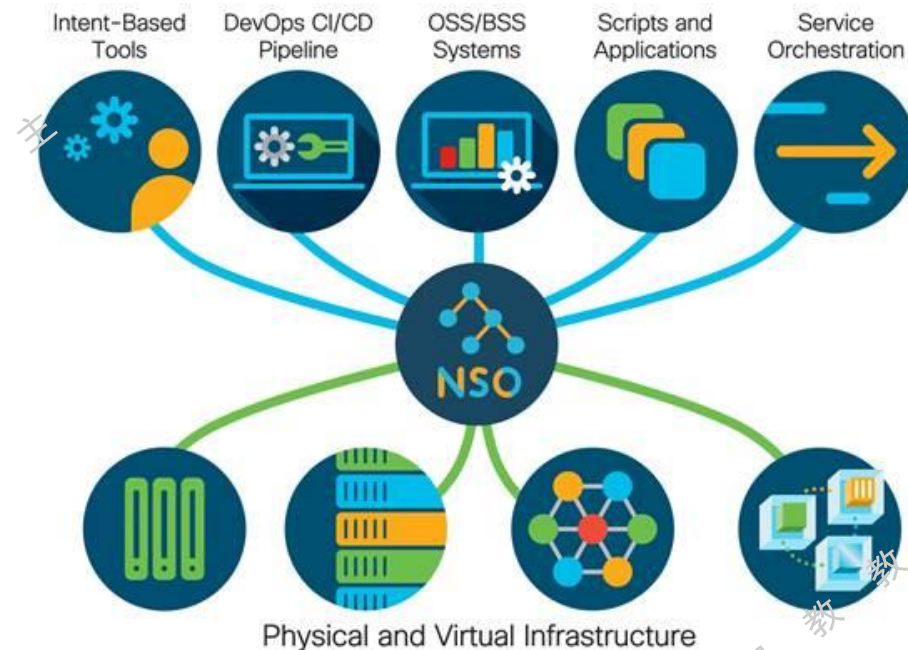
Gitlab Runner (Ubuntu)



乾颐堂 现任明教教主

# Cisco NSO

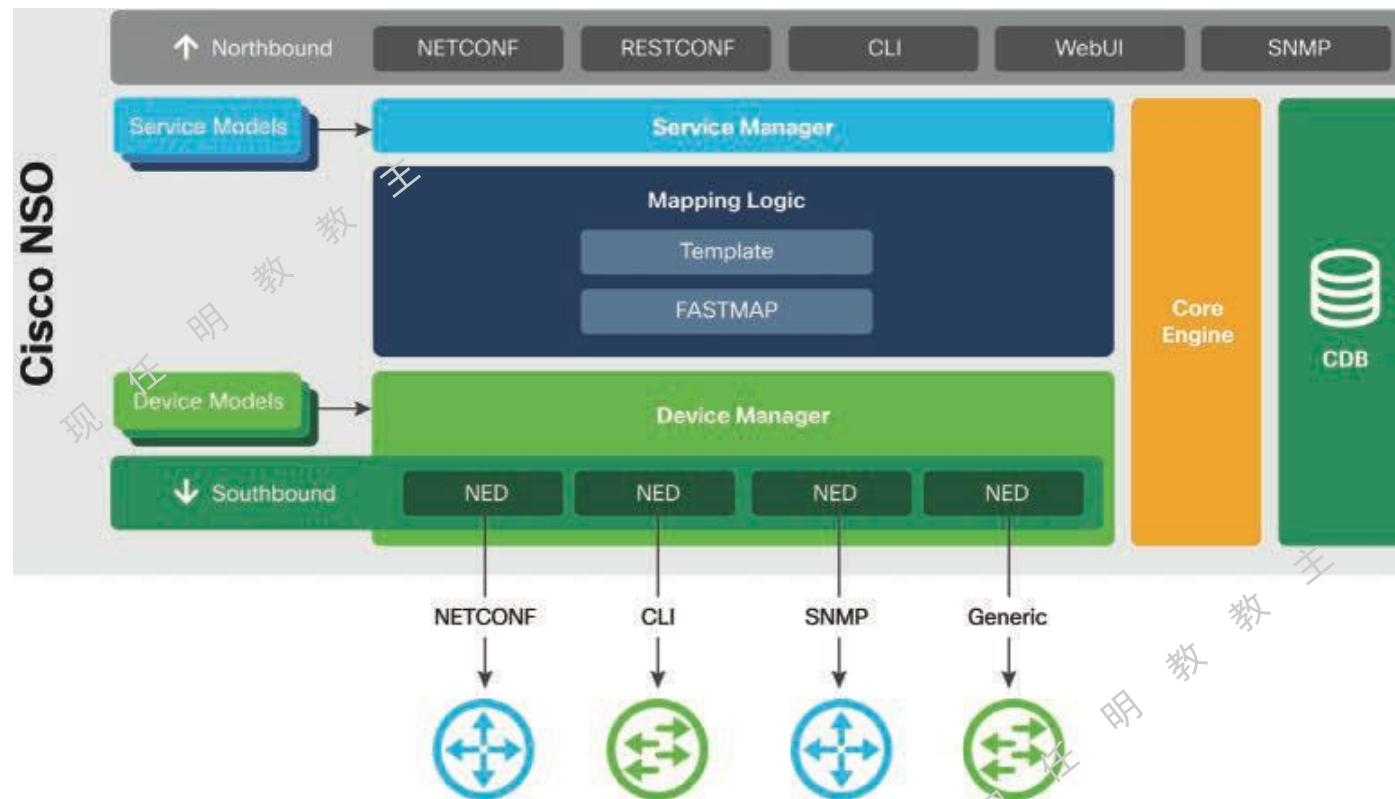
Cisco Network Services Orchestrator (NSO) is a **network services orchestration**[网络服务编排] platform that enables end-to-end service deployment across **multivendor physical and virtual infra-structure**[支持多厂商的虚拟和物理架构]. NSO can also be considered a **multivendor service-layer SDN controller**[多厂商服务层控制器] 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:

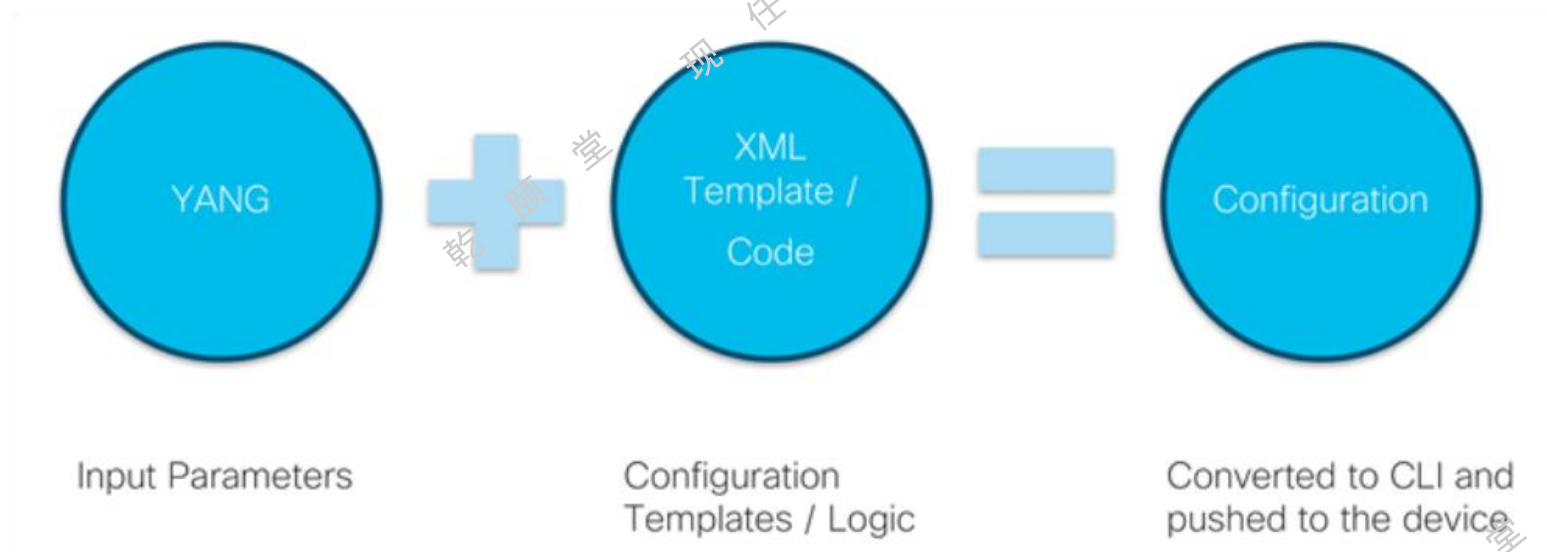
- Service manager
- Device manager
- Mapping logic
- Configuration database



## 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 configuration 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.

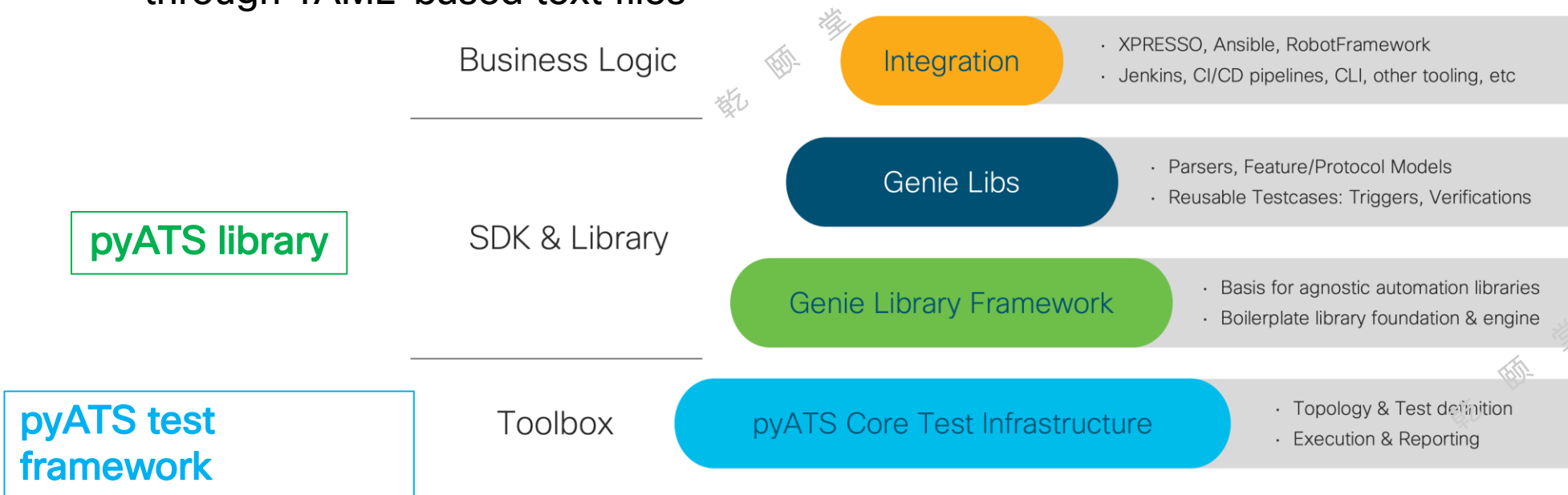


<https://developer.cisco.com/docs/pyats-getting-started/>

# 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



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

乾颐堂 现任明教教主

乾颐堂 现任明教教主

乾颐堂 现任明教教主

# Gitlab三个Runner

### Specific runners

These runners are specific to this project.

Set up a specific Runner for a project

1. Install GitLab Runner and ensure it's running.
2. Register the runner with this URL:  
`https://gitlab.qytangnetdevops.com/`

And this registration token:  
`UsJ4JHwnfxET1EWx4xBL`

[Reset registration token](#)

[Show Runner installation instructions](#)

---

### Available specific runners

<span style="color: green;">●</span> #3 (aW5yuZDb)	<a href="#">Remove runner</a>
controller_ci <a href="#">controller_ci</a>	
<span style="color: green;">●</span> #2 (FVgZrnBL)	<a href="#">Remove runner</a>
controller_cd <a href="#">controller_cd</a>	
<span style="color: green;">●</span> #1 (JeDwXoHR)	<a href="#">Remove runner</a>
netdevops <a href="#">netdevops</a>	

### Shared runners

These runners are shared across this GitLab instance.

The same shared runner executes code from multiple projects, unless you configure autoscaling with [MaxBuilds](#) set to 1 (which it is on GitLab.com).

Enable shared runners for this project

This GitLab instance does not provide any shared runners yet. Instance administrators can register shared runners in the admin area.

### Group runners

These runners are shared across projects in this group.

Group runners can be managed with the [Runner API](#).

This project does not belong to a group and cannot make use of group runners.

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

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. [Learn more.](#)
- Masked:** Hidden in job logs. Must match masking requirements. [Learn more.](#)

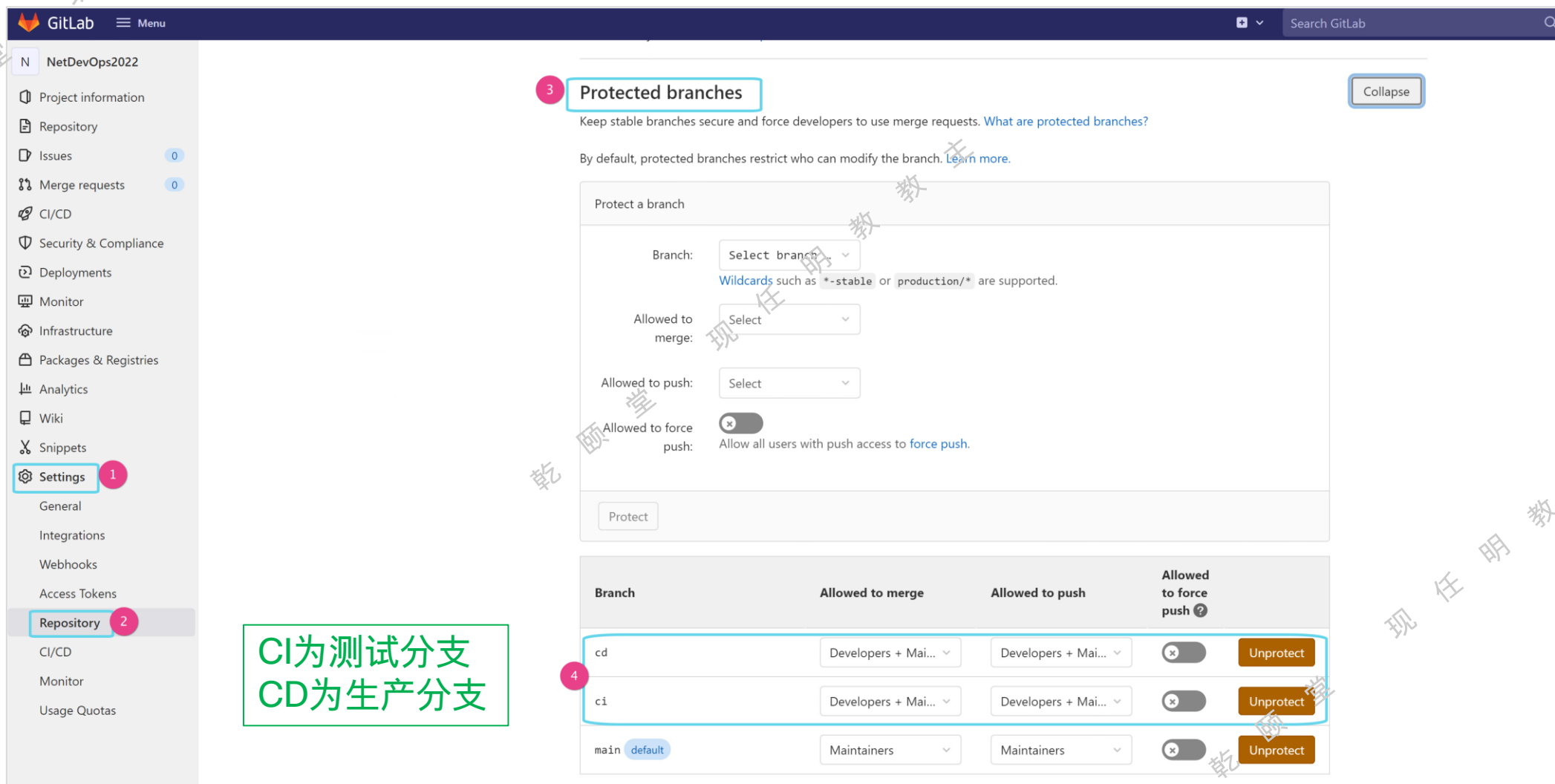
Environment variables are configured by your administrator to be **protected** by default.

Type	Key	Value	Protected	Masked	Environments
Variable	cd_nso_restconf_base_url	https://controller_cd.qytangnetdevops.com:8888/restconf/data/	✓	✗	All (default)
Variable	ci_nso_restconf_base_url	https://controller_ci.qytangnetdevops.com:8888/restconf/data/	✓	✗	All (default)
Variable	nso_password	Cisc0123	✓	✗	All (default)
Variable	nso_username	ncsuser	✓	✗	All (default)

[Add variable](#) [Reveal values](#)



# ci和cd两个分支都是保护分支



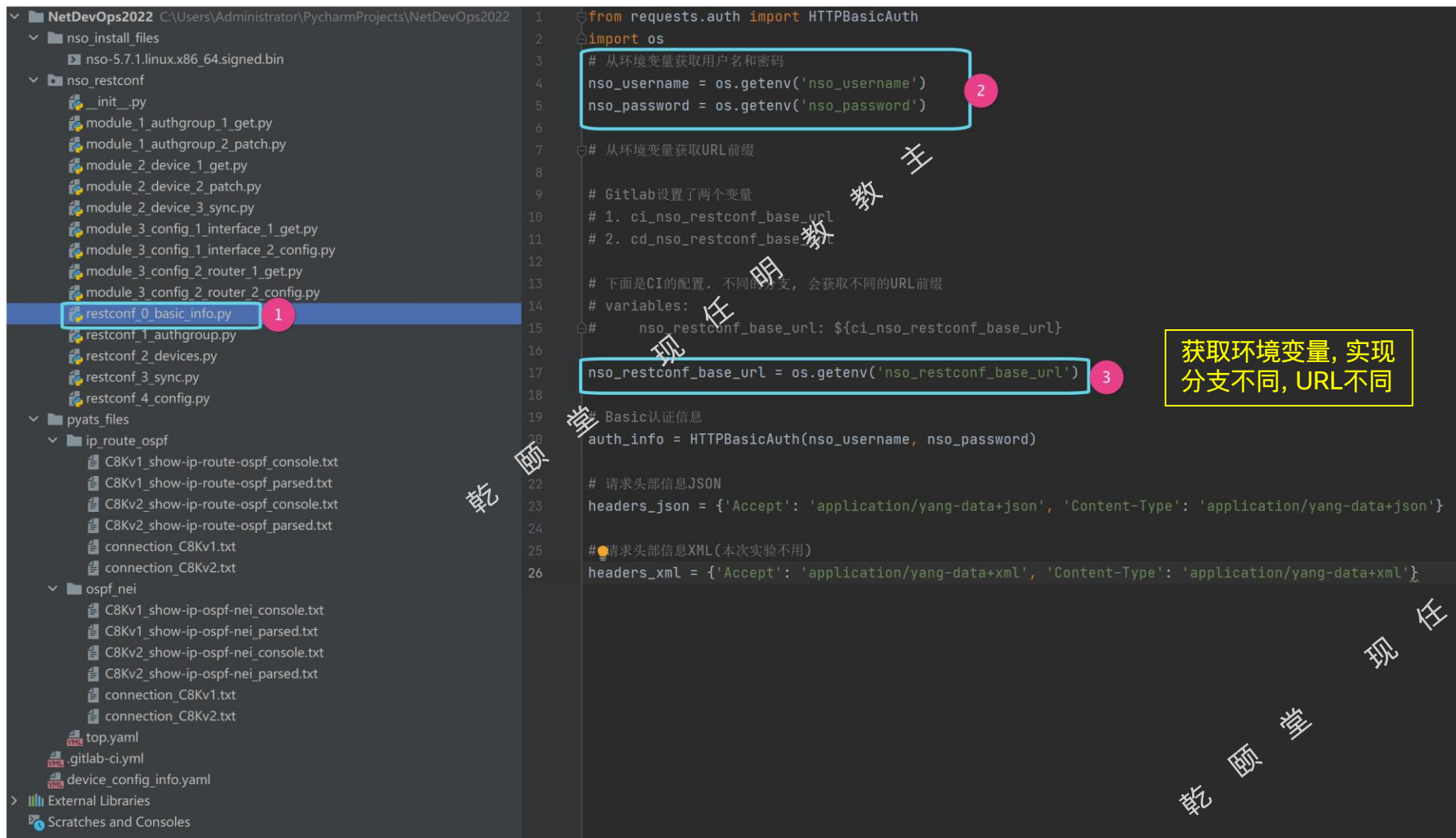
The screenshot shows the GitLab interface for configuring protected branches. The left sidebar contains navigation options, with 'Settings' (1) and 'Repository' (2) highlighted. The main content area is titled 'Protected branches' (3) and includes a 'Collapse' button. Below the title, there is a 'Protect a branch' form with fields for 'Branch', 'Allowed to merge', and 'Allowed to push', along with a toggle for 'Allowed to force push'. A 'Protect' button is at the bottom of the form. Below the form is a table of existing protected branches.

Branch	Allowed to merge	Allowed to push	Allowed to force push ?	
cd	Developers + Mai...	Developers + Mai...	<input type="checkbox"/>	Unprotect
ci	Developers + Mai...	Developers + Mai...	<input type="checkbox"/>	Unprotect
main <small>default</small>	Maintainers	Maintainers	<input type="checkbox"/>	Unprotect

4

CI为测试分支  
CD为生产分支

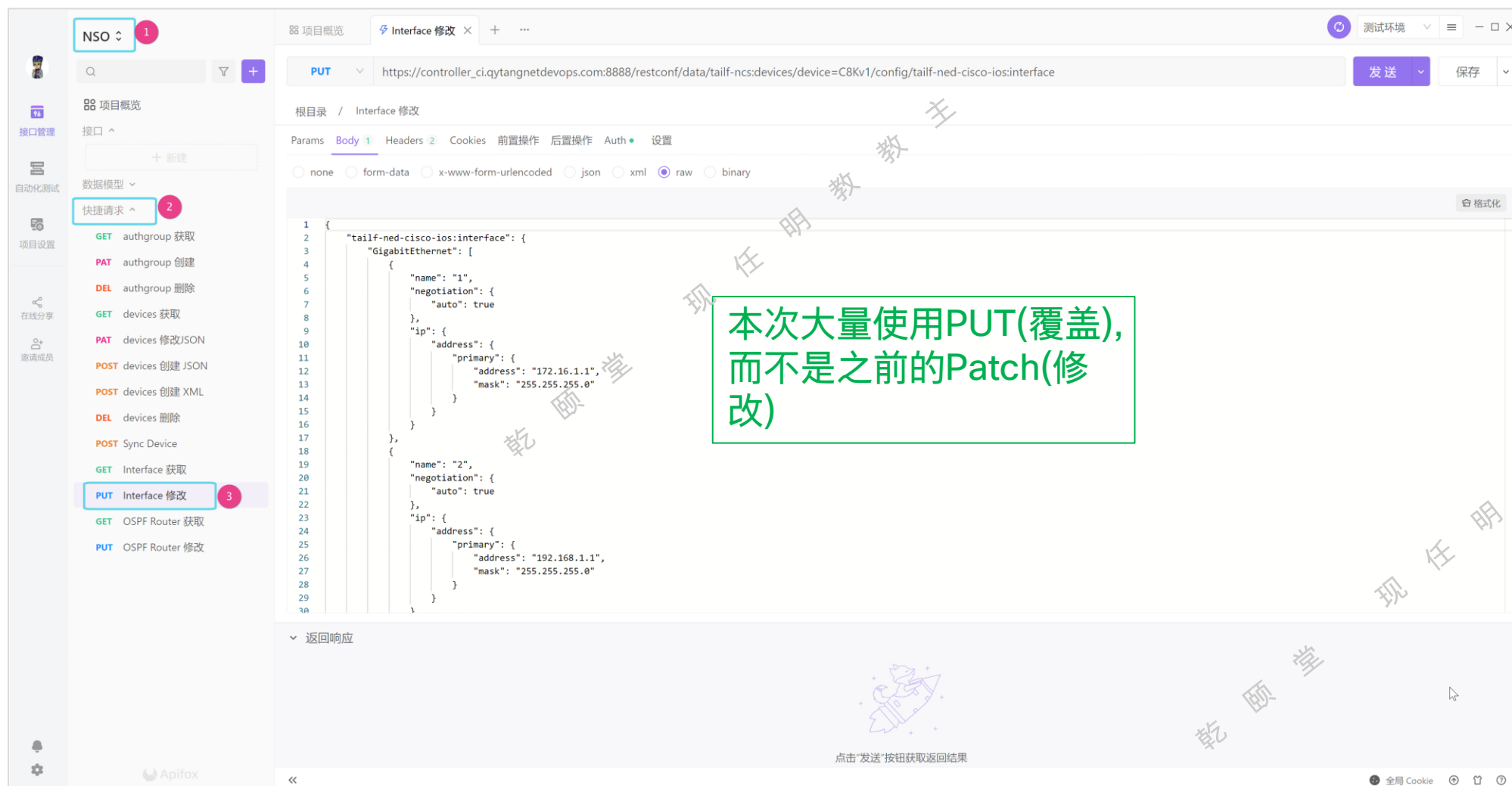
# restconf\_0\_basic\_info.py



```
1 from requests.auth import HTTPBasicAuth
2 import os
3 # 从环境变量获取用户名和密码
4 nso_username = os.getenv('nso_username')
5 nso_password = os.getenv('nso_password')
6
7 # 从环境变量获取URL前缀
8
9 # Gitlab设置了两个变量
10 # 1. ci_nso_restconf_base_url
11 # 2. cd_nso_restconf_base_url
12
13 # 下面是CI的配置。不同的分支，会获取不同的URL前缀
14 # variables:
15 #   nso_restconf_base_url: ${ci_nso_restconf_base_url}
16 nso_restconf_base_url = os.getenv('nso_restconf_base_url')
17
18 # Basic认证信息
19 auth_info = HTTPBasicAuth(nso_username, nso_password)
20
21 # 请求头部信息JSON
22 headers_json = {'Accept': 'application/yang-data+json', 'Content-Type': 'application/yang-data+json'}
23
24 # 请求头部信息XML(本次实验不用)
25 headers_xml = {'Accept': 'application/yang-data+xml', 'Content-Type': 'application/yang-data+xml'}
```

获取环境变量, 实现分支不同, URL不同

# 提前对所有的RESTCONF请求做了测试



NSO 1

项目概览

PUT https://controller\_ci.qytangnetdevops.com:8888/restconf/data/taif-ncs:devices/device=C8Kv1/config/taif-ned-cisco-ios:interface

发送 保存

根目录 / Interface 修改

Params Body 1 Headers 2 Cookies 前置操作 后置操作 Auth 设置

none form-data x-www-form-urlencoded json xml raw binary

```
1 {
2   "taif-ned-cisco-ios:interface": {
3     "GigabitEthernet": [
4       {
5         "name": "1",
6         "negotiation": {
7           "auto": true
8         },
9         "ip": {
10          "address": {
11            "primary": {
12              "address": "172.16.1.1",
13              "mask": "255.255.255.0"
14            }
15          }
16        }
17      },
18      {
19        "name": "2",
20        "negotiation": {
21          "auto": true
22        },
23        "ip": {
24          "address": {
25            "primary": {
26              "address": "192.168.1.1",
27              "mask": "255.255.255.0"
28            }
29          }
30        }
31      }
32    ]
33  }
34 }
```

格式化

快速请求 2

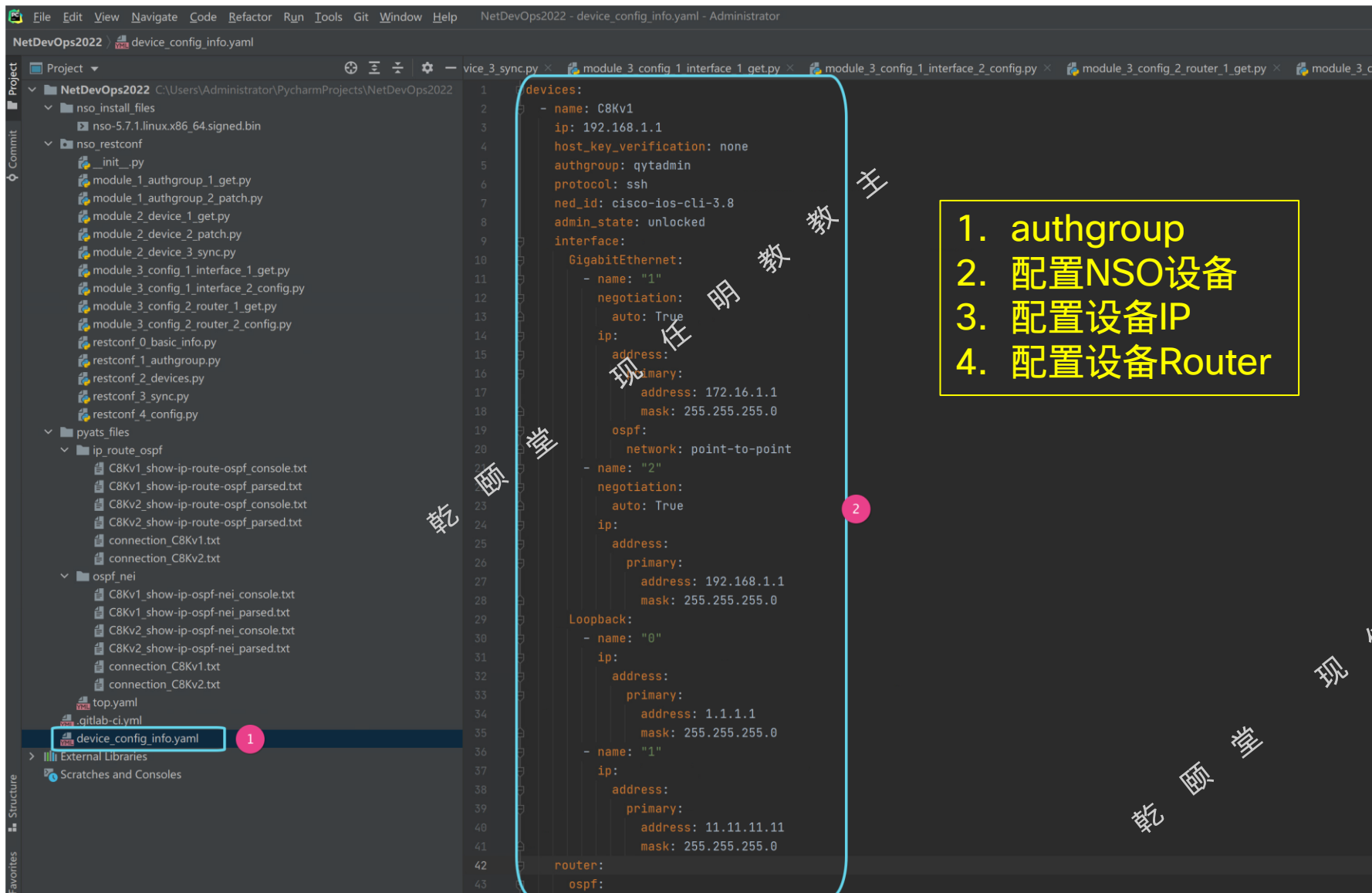
- GET authgroup 获取
- PAT authgroup 创建
- DEL authgroup 删除
- GET devices 获取
- PAT devices 修改JSON
- POST devices 创建 JSON
- POST devices 创建 XML
- DEL devices 删除
- POST Sync Device
- GET Interface 获取
- PUT Interface 修改 3
- GET OSPF Router 获取
- PUT OSPF Router 修改

返回响应

点击“发送”按钮获取返回结果

全局 Cookie

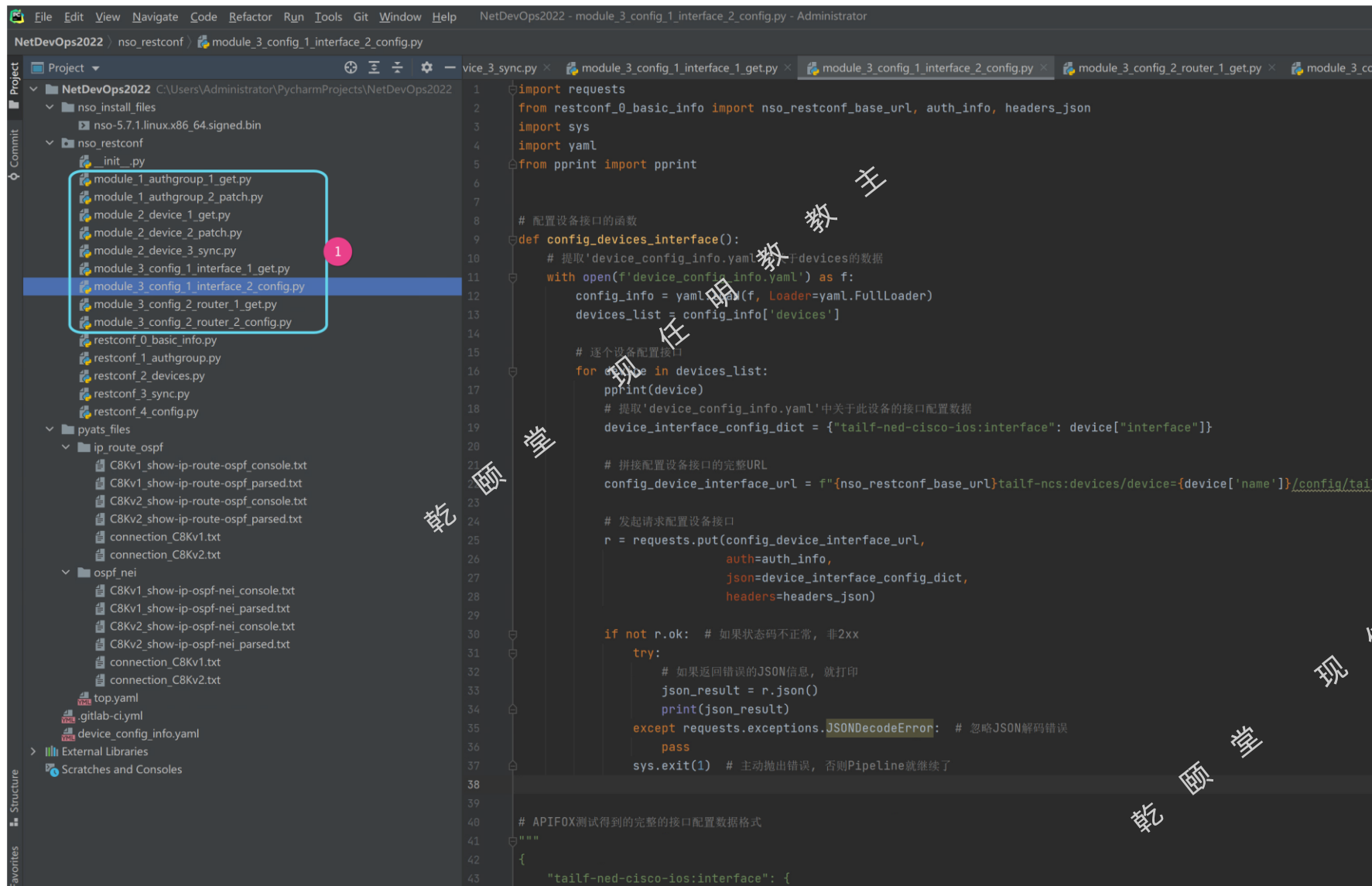
# 设备配置数据



```
1 devices:
2   - name: C8Kv1
3     ip: 192.168.1.1
4     host_key_verification: none
5     authgroup: qytadmin
6     protocol: ssh
7     ned_id: cisco-ios-cli-3.8
8     admin_state: unlocked
9     interface:
10      GigabitEthernet:
11       - name: "1"
12         negotiation:
13           auto: True
14         ip:
15           address:
16             primary:
17               address: 172.16.1.1
18               mask: 255.255.255.0
19         ospf:
20           network: point-to-point
21       - name: "2"
22         negotiation:
23           auto: True
24         ip:
25           address:
26             primary:
27               address: 192.168.1.1
28               mask: 255.255.255.0
29         Loopback:
30           - name: "0"
31             ip:
32               address:
33                 primary:
34                   address: 1.1.1.1
35                   mask: 255.255.255.0
36           - name: "1"
37             ip:
38               address:
39                 primary:
40                   address: 11.11.11.11
41                   mask: 255.255.255.0
42         router:
43         ospf:
```

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

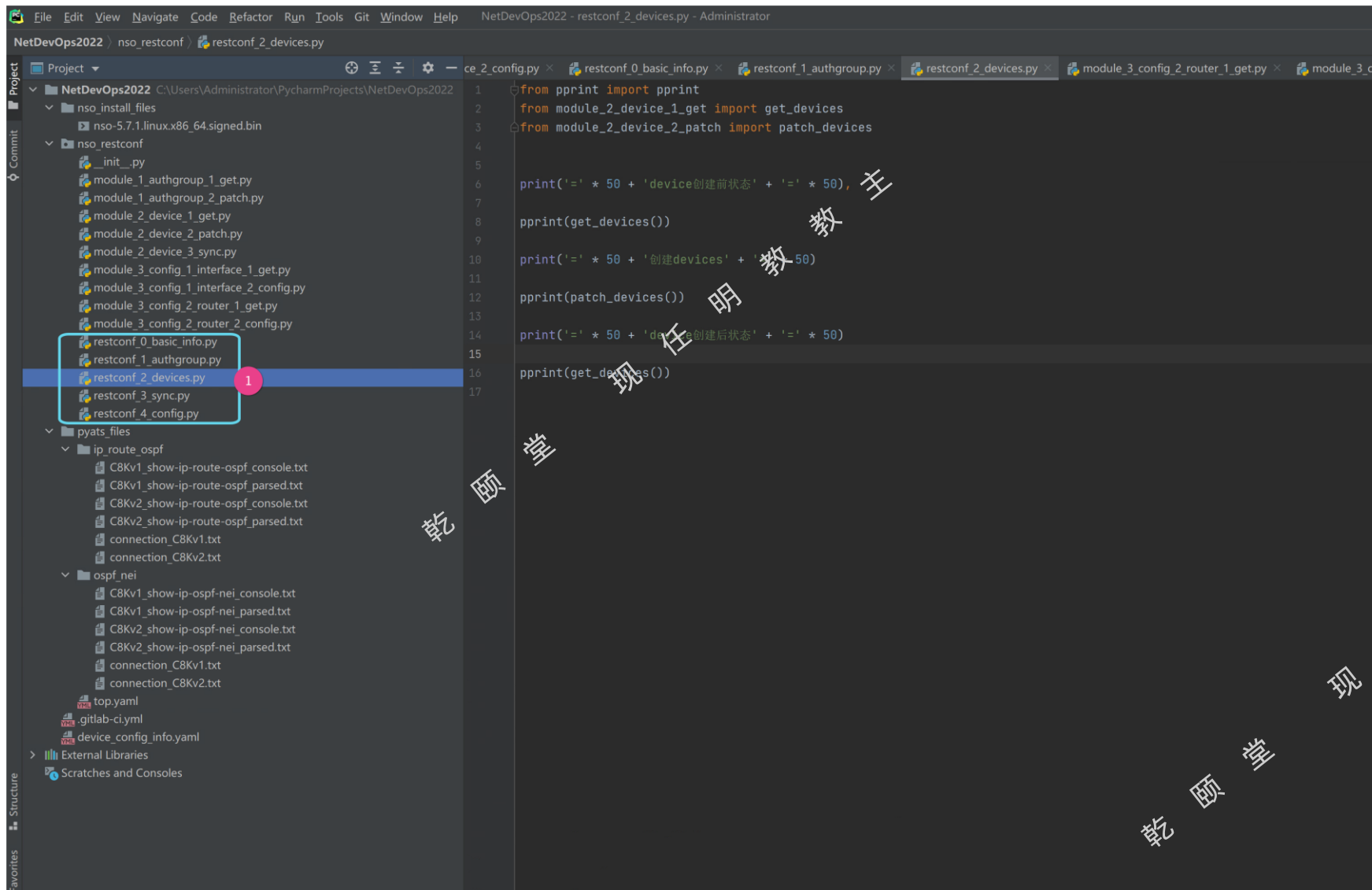
# 配置模块(功能拆分)



```
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
  nso_install_files
  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
    ospf_nei
  top.yaml
  .gitlab-ci.yml
  device_config_info.yaml
  External Libraries
  Scratches and Consoles
  vices_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_con...

1 import requests
2 from restconf_0_basic_info import nso_restconf_base_url, auth_info, headers_json
3 import sys
4 import yaml
5 from pprint import pprint
6
7
8 # 配置设备接口的函数
9 def config_devices_interface():
10     # 提取'device_config_info.yaml'中devices的数据
11     with open(f'device_config_info.yaml') as f:
12         config_info = yaml.safe_load(f, Loader=yaml.FullLoader)
13         devices_list = config_info['devices']
14
15     # 逐个设备配置接口
16     for device in devices_list:
17         pprint(device)
18         # 提取'device_config_info.yaml'中关于此设备的接口配置数据
19         device_interface_config_dict = {"tailf-ned-cisco-ios:interface": device["interface"]}
20
21     # 拼接配置设备接口的完整URL
22     config_device_interface_url = f"{nso_restconf_base_url}tailf-ncs:devices/device={device['name']}/config/tailf-ncs:interface={device['interface']}"
23
24     # 发起请求配置设备接口
25     r = requests.put(config_device_interface_url,
26                     auth=auth_info,
27                     json=device_interface_config_dict,
28                     headers=headers_json)
29
30     if not r.ok: # 如果状态码不正常, 非2xx
31         try:
32             # 如果返回错误的JSON信息, 就打印
33             json_result = r.json()
34             print(json_result)
35         except requests.exceptions.JSONDecodeError: # 忽略JSON解码错误
36             pass
37         sys.exit(1) # 主动抛出错误, 否则PipeLine就继续了
38
39
40 # APIFOX测试得到的完整的接口配置数据格式
41 """
42 {
43     "tailf-ned-cisco-ios:interface": {
```

# 任务汇总模块



# Gitlab Pipeline配置文件

The screenshot shows a code editor with a project named 'NetDevOps2022'. The file explorer on the left shows a directory structure including 'nso\_install\_files', 'nso\_restconf', and 'pyats\_files'. The main editor displays the content of '.gitlab-ci.yml'. A blue box highlights the 'nso\_restconf' directory in the file explorer, and a red circle with the number '1' is placed next to the '.gitlab-ci.yml' file. A yellow box highlights the 'nso\_authgroup' stage in the pipeline configuration, and a red circle with the number '2' is placed next to it. The pipeline configuration includes stages for 'nso\_authgroup' and 'nso\_devices', each with 'ci' and 'cd' jobs. A yellow callout box on the right contains the text: '任何一个步骤都有CI和CD两个'.

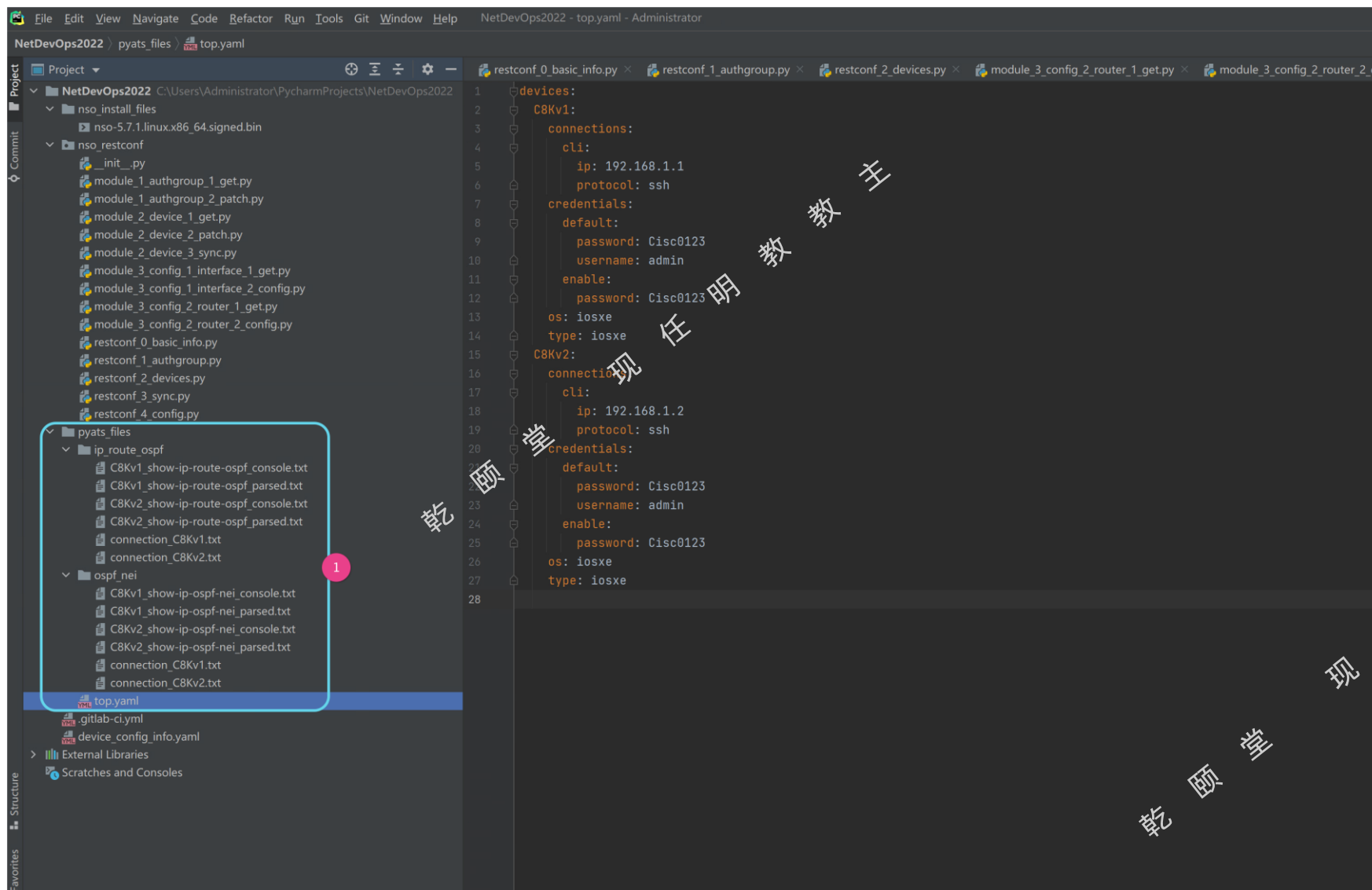
```

1 stages:
2   - nso_authgroup
3   - nso_devices
4   - nso_sync
5   - nso_config
6   - pyats_diff
7
8
9   ci_nso_authgroup: # 配置认证组添加设备 (CI)
10    stage: nso_authgroup
11    only:
12      - ci
13    variables: # 相同的环境变量nso_restconf_base_url, 不同分支不同的值
14      nso_restconf_base_url: ${ci_nso_restconf_base_url}
15    script:
16      - python3 ./nso_restconf/restconf_1_authgroup.py
17    tags:
18      - netdevops
19
20
21   cd_nso_authgroup: # 配置认证组添加设备 (CD)
22    stage: nso_authgroup
23    only:
24      - cd
25    variables: # 相同的环境变量nso_restconf_base_url, 不同分支不同的值
26      nso_restconf_base_url: ${cd_nso_restconf_base_url}
27    script:
28      - python3 ./nso_restconf/restconf_1_authgroup.py
29    tags:
30      - netdevops
31
32
33   ci_nso_devices: # 创建设备 (CI)
34    stage: nso_devices
35    only:
36      - ci
37    variables: # 相同的环境变量nso_restconf_base_url, 不同分支不同的值
38      nso_restconf_base_url: ${ci_nso_restconf_base_url}
39    script:
40      - python3 ./nso_restconf/restconf_2_devices.py
41    tags:
42      - netdevops
43

```

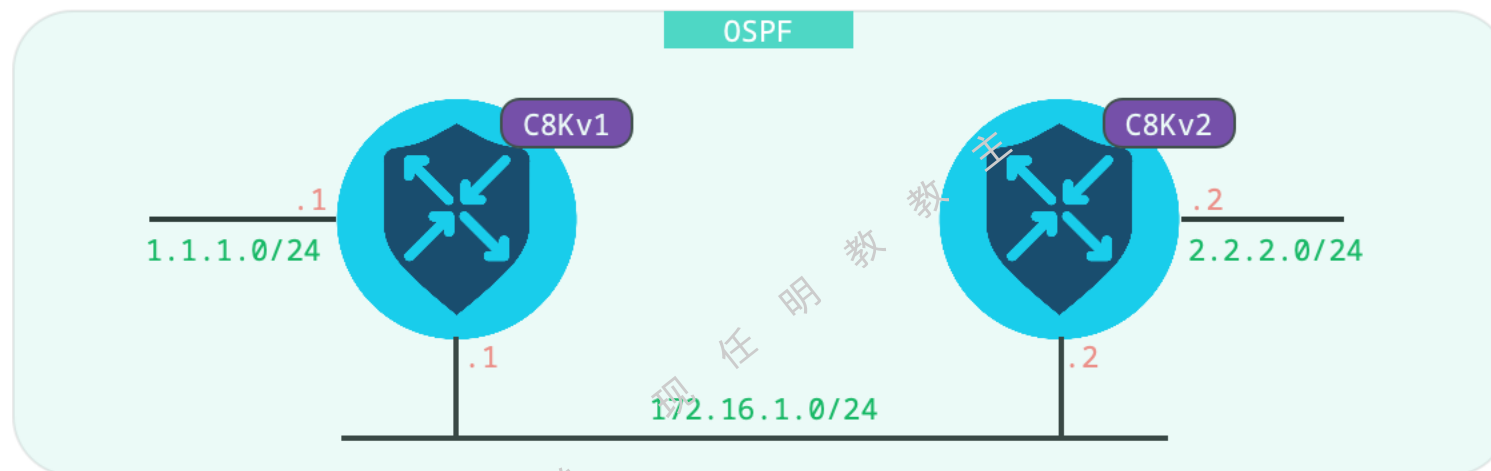
任何一个步骤都有CI和CD两个

# pyATS部分





# 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
  0       2.2.2.2 [110/2] via 172.16.1.2, 00:03:03, GigabitEthernet1

```

乾颐堂 现任明教教主

乾颐堂 现任明教教主

乾颐堂 现任明教教主

### 3. 测试

# CI分修改配置然后Push

The screenshot shows the PyCharm IDE interface with the following elements:

- File Editor:** Displays the `device_config_info.yaml` file. The `Loopback` section is highlighted with a blue box and labeled '3' (修改配置). The configuration includes:
 

```

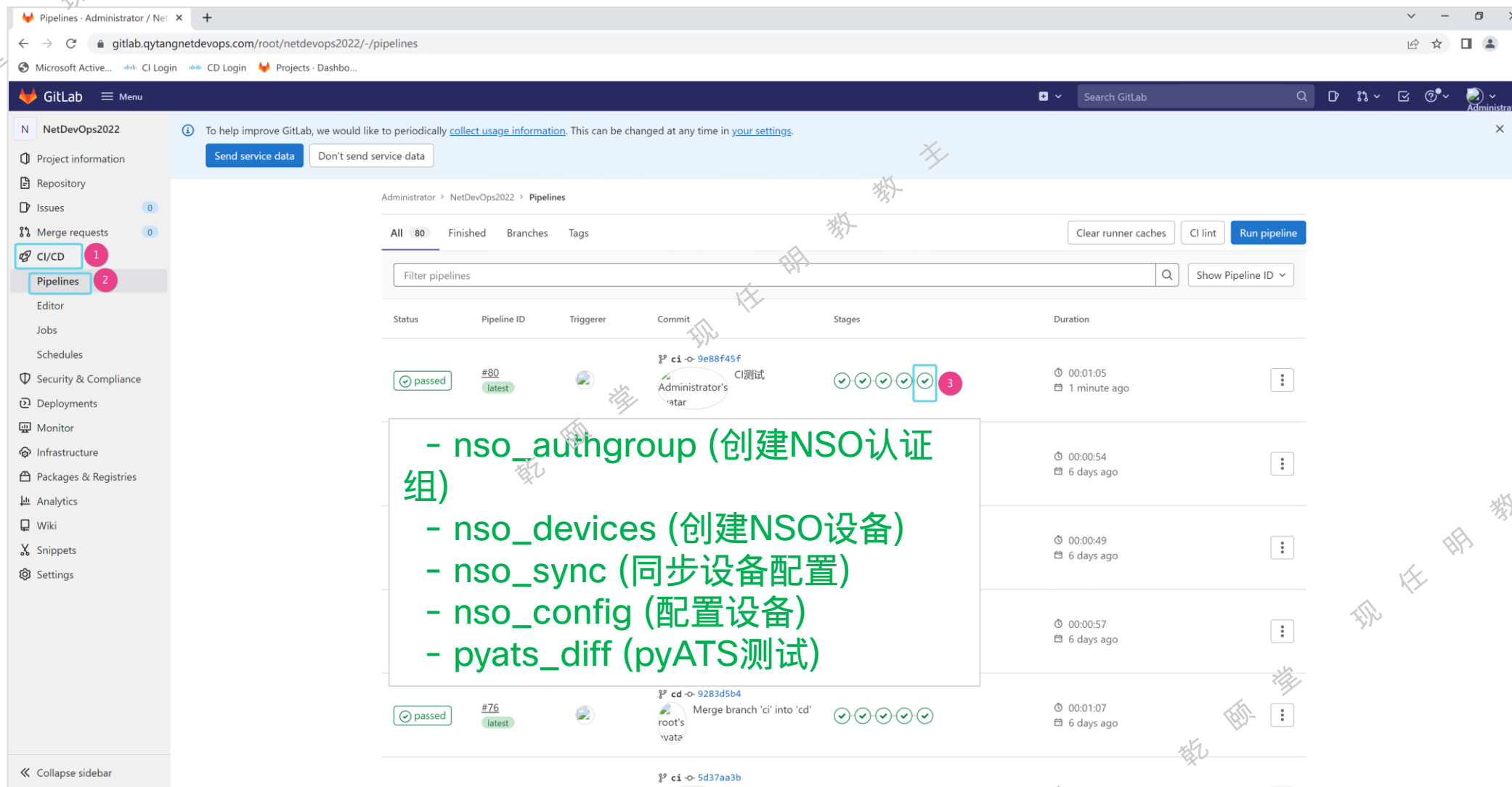
      Loopback:
      - name: "0"
      ip:
      address:
      primary:
      address: 1.1.1.1
      mask: 255.255.255.0
      - name: "1"
      ip:
      address:
      primary:
      address: 11.11.11.11
      mask: 255.255.255.0
      
```
- Commit Dialog:** A 'Push Commits to NetDevOps2022' dialog is open, showing the local branch 'ci' and the remote branch 'labgittab: ci'. The 'Push' button is highlighted with a yellow box and labeled '4' (Push).
- Terminal:** Shows the command `CI测试` and the output `1 file committed: CI测试`. A yellow box labeled '1' (CI分支) points to the terminal output.
- Annotations:**
  - '2' (Default Changelist) points to the commit dialog.
  - '3' (修改配置) points to the Loopback configuration.
  - '4' (Push) points to the push button.
  - '1' (CI分支) points to the terminal output.

修改配置

配置两个环回口, 并且宣告  
最好Lo1地址最后一个111

CI分支

# 查看Pipeline状态



Administrator > NetDevOps2022 > Pipelines

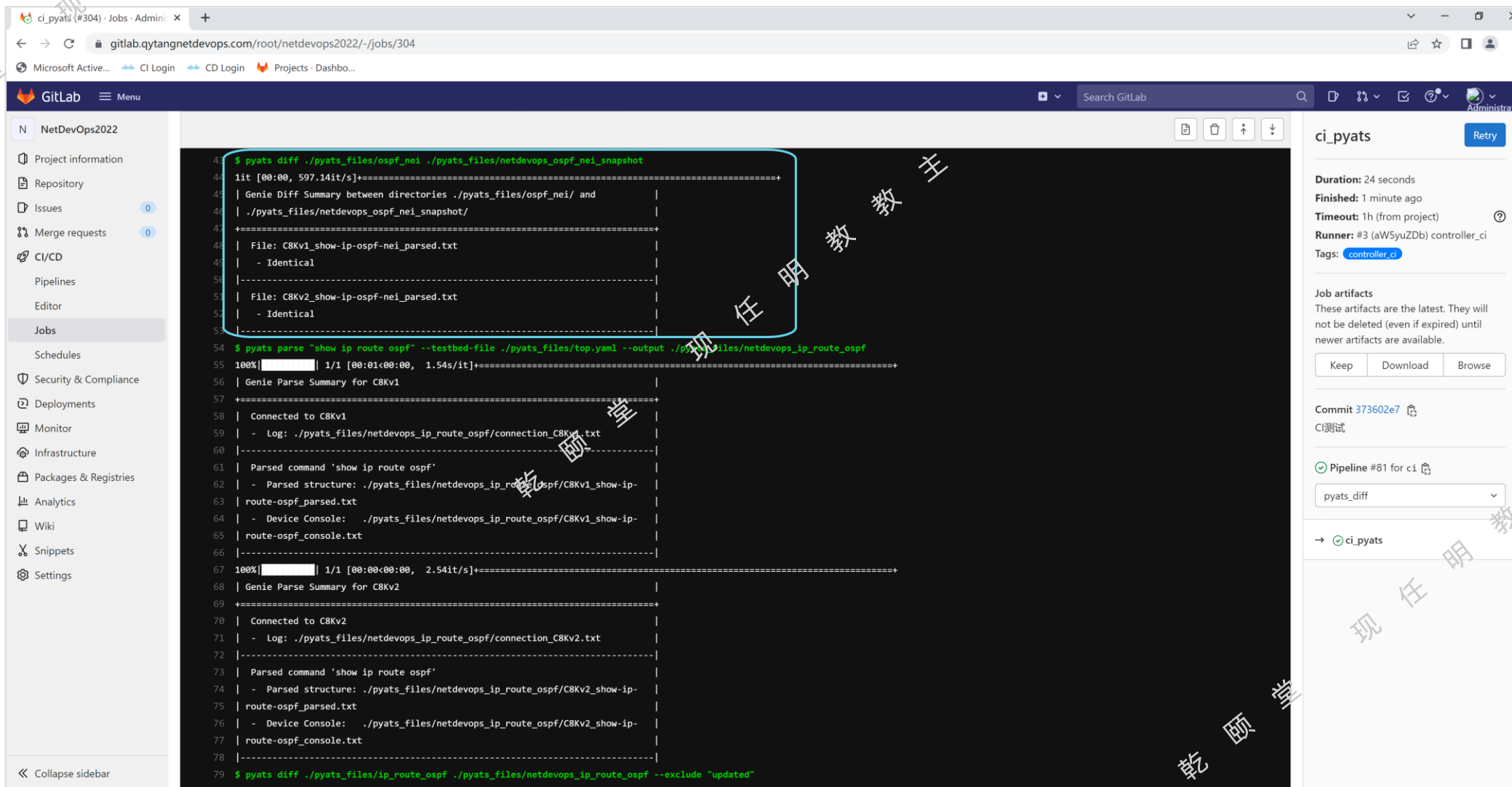
All 80 Finished Branches Tags

Filter pipelines

Status	Pipeline ID	Triggerer	Commit	Stages	Duration
passed	#80 latest	Administrator's 'atar	ci -> 9e88f45f Administrator's 'atar	CI测试	00:01:05 1 minute ago
passed	#76 latest	root's 'vata	cd -> 9283d5b4 Merge branch 'ci' into 'cd'		00:00:54 6 days ago
passed	#76 latest	root's 'vata	cd -> 9283d5b4 Merge branch 'ci' into 'cd'		00:00:49 6 days ago
passed	#76 latest	root's 'vata	cd -> 9283d5b4 Merge branch 'ci' into 'cd'		00:00:57 6 days ago
passed	#76 latest	root's 'vata	cd -> 9283d5b4 Merge branch 'ci' into 'cd'		00:01:07 6 days ago

- nso\_authgroup (创建NSO认证组)
- nso\_devices (创建NSO设备)
- nso\_sync (同步设备配置)
- nso\_config (配置设备)
- pyats\_diff (pyATS测试)

# 通过pyATS看到邻居是一致的



```
44 $ pyats diff ./pyats_files/ospf_nei ./pyats_files/netdevops_ospf_nei_snapshot
44 |it [00:00, 597.14it/s]+++++
45 | Genie Diff Summary between directories ./pyats_files/ospf_nei/ and
46 | ./pyats_files/netdevops_ospf_nei_snapshot/
47 |-----
48 | File: C8Kv1_show-ip-ospf-nei_parsed.txt
49 | - Identical
50 |-----
51 | File: C8Kv2_show-ip-ospf-nei_parsed.txt
52 | - Identical
53 |-----
54 $ pyats parse "show ip route ospf" --testbed-file ./pyats_files/top.yaml --output ./pyats_files/netdevops_ip_route_ospf
55 |100%| | 1/1 [00:01:00:00, 1.54s/it]+++++
56 | Genie Parse Summary for C8Kv1
57 |-----
58 | Connected to C8Kv1
59 | - Log: ./pyats_files/netdevops_ip_route_ospf/connection_C8Kv1.txt
60 |-----
61 | Parsed command 'show ip route ospf'
62 | - Parsed structure: ./pyats_files/netdevops_ip_route_ospf/C8Kv1_show-ip-
63 | route-ospf_parsed.txt
64 | - Device Console: ./pyats_files/netdevops_ip_route_ospf/C8Kv1_show-ip-
65 | route-ospf_console.txt
66 |-----
67 |100%| | 1/1 [00:00:00:00, 2.54it/s]+++++
68 | Genie Parse Summary for C8Kv2
69 |-----
70 | Connected to C8Kv2
71 | - Log: ./pyats_files/netdevops_ip_route_ospf/connection_C8Kv2.txt
72 |-----
73 | Parsed command 'show ip route ospf'
74 | - Parsed structure: ./pyats_files/netdevops_ip_route_ospf/C8Kv2_show-ip-
75 | route-ospf_parsed.txt
76 | - Device Console: ./pyats_files/netdevops_ip_route_ospf/C8Kv2_show-ip-
77 | route-ospf_console.txt
78 |-----
79 $ pyats diff ./pyats_files/ip_route_ospf ./pyats_files/netdevops_ip_route_ospf --exclude "updated"
```

**ci\_pyats** Retry

**Duration:** 24 seconds  
**Finished:** 1 minute ago  
**Timeout:** 1h (from project)  
**Runner:** #3 (aW5yuZDb) controller\_ci  
**Tags:** [controller\\_ci](#)

**Job artifacts**  
These artifacts are the latest. They will not be deleted (even if expired) until newer artifacts are available.

[Keep](#) [Download](#) [Browse](#)

**Commit** [373602e7](#)  
CI测试

Pipeline #81 for [c1](#)  
[pyats\\_diff](#)

→ [ci\\_pyats](#)

# 通过pyATS看到路由不一致

The screenshot shows a GitLab CI pipeline for a project named 'ci\_pyats'. The main terminal window displays the output of a test run. A blue box highlights the command used to generate diff files: `$ pyats diff ./pyats_files/ip_route_ospf ./pyats_files/netdevops_ip_route_ospf --exclude "updated"`. Below this, the terminal shows the paths to the generated diff files: `./diff_C8Kv1_show-ip-route-ospf_parsed.txt` and `./diff_C8Kv2_show-ip-route-ospf_parsed.txt`. A green box highlights the artifact upload step: `Uploading artifacts for successful job`. On the right side of the interface, the 'Job artifacts' section shows a 'Download' button, which is circled in blue and labeled with a red '3'. A green arrow points from this button to the diff files mentioned in the terminal. A green box with the text '可以下载 diff文件' (Can download diff files) is also present, with an arrow pointing to the diff file paths in the terminal. A red '2' is placed near the diff file paths, and a red '1' is placed near the command. The bottom right corner of the terminal shows the job status as 'Job succeeded'.

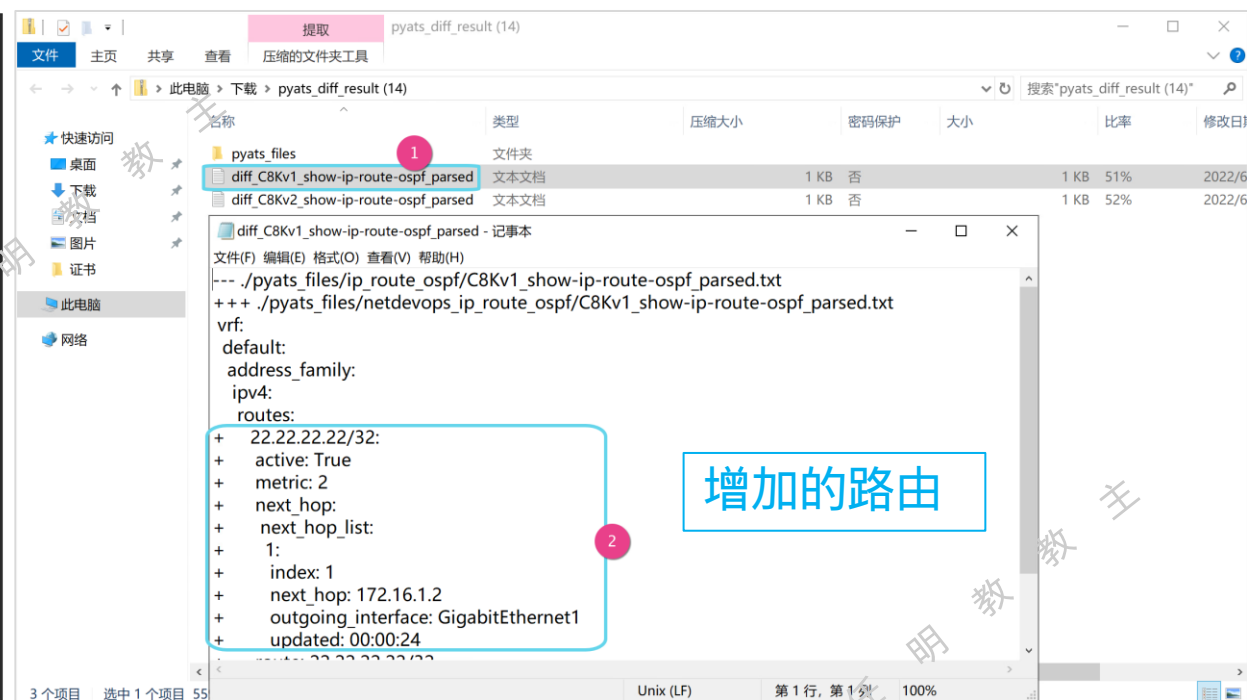
# 查看Diff文件

```

cd_pyats:
  stage: pyats_diff
  on:
    - 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

Administrator > NetDevOps2022 > Merge requests

Open 0 Merged 6 Closed 1 All 7

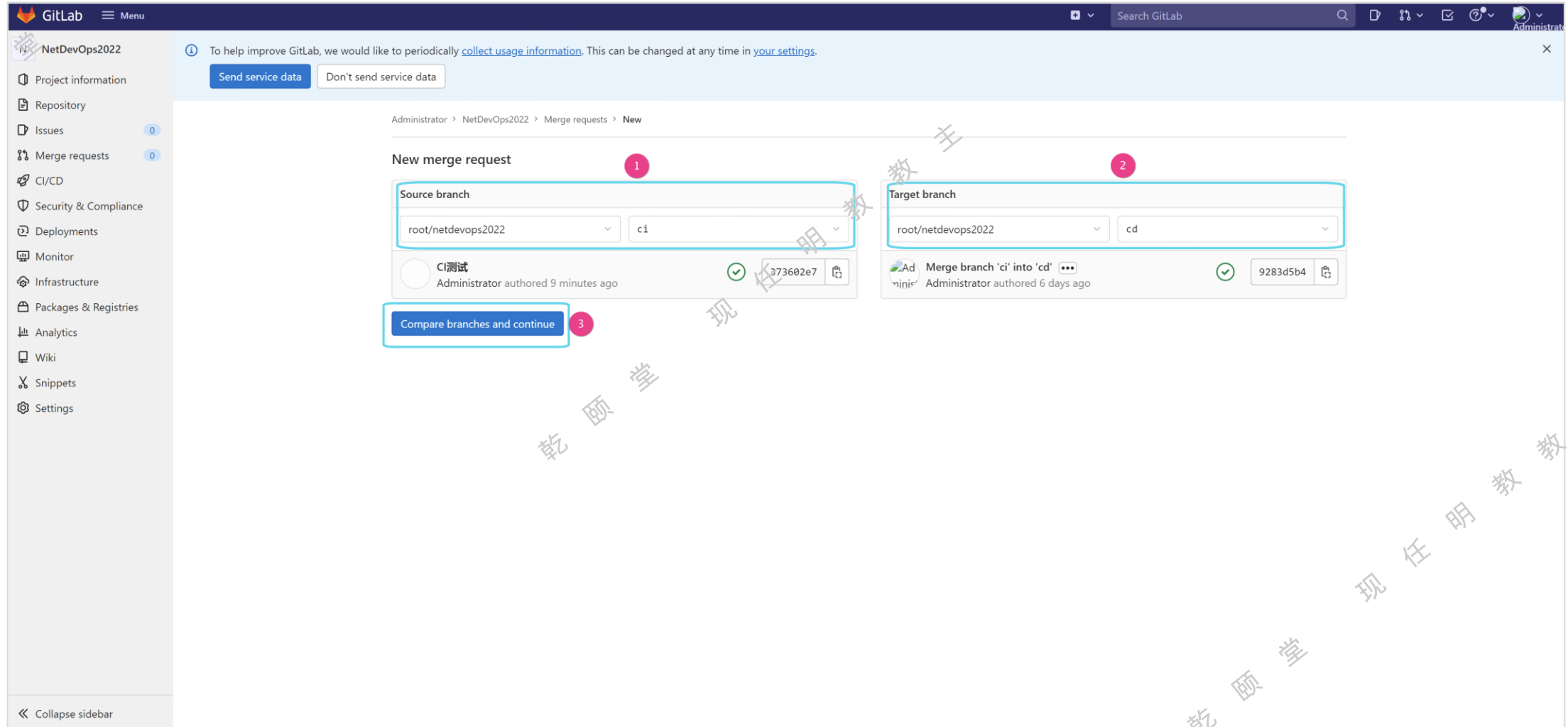
Recent searches Search or filter results... Created date

There are no open merge requests  
To keep this project going, create a new merge request

New merge request



# Merge Request



The screenshot shows the GitLab web interface for creating a new merge request. The page title is "New merge request". At the top, there is a notification banner about collecting usage information. Below that, the breadcrumb path is "Administrator > NetDevOps2022 > Merge requests > New".

The main content area is titled "New merge request" and contains two columns of input fields:

- Source branch:** A dropdown menu is set to "root/netdevops2022" and a text input field contains "ci".
- Target branch:** A dropdown menu is set to "root/netdevops2022" and a dropdown menu is set to "cd".

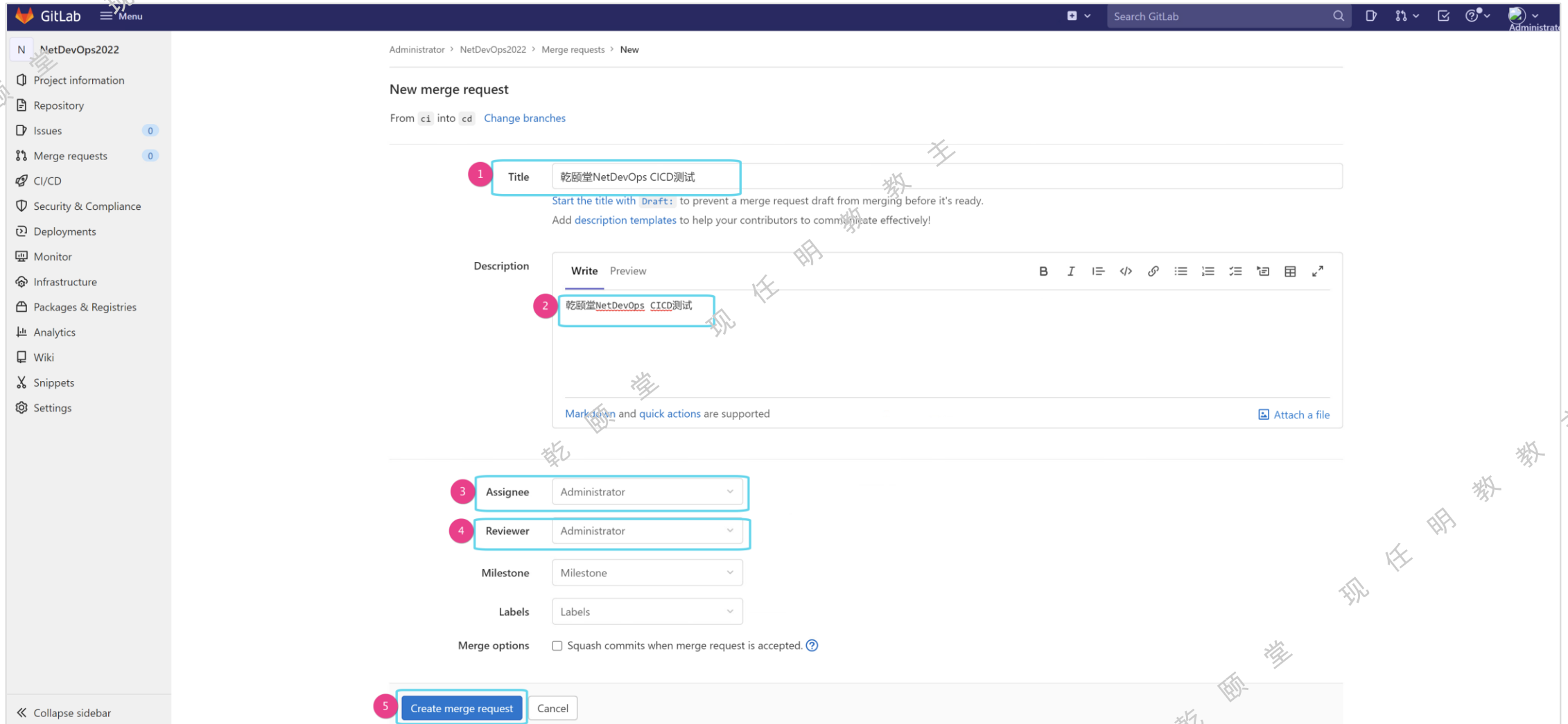
Below the source branch field, there is a preview of the merge request card. It shows a green checkmark, the author "CI测试", and the commit hash "373602e7".

Below the target branch field, there is a preview of the merge request card. It shows a green checkmark, the author "Administrator", and the commit hash "9283d5b4".

At the bottom of the form, there is a button labeled "Compare branches and continue".

Three red circles with numbers 1, 2, and 3 are overlaid on the image to highlight the source branch field, the target branch field, and the "Compare branches and continue" button, respectively.

# Merge Request



Administrator > NetDevOps2022 > Merge requests > New

## New merge request

From `ci1` into `cd` [Change branches](#)

1 **Title** 乾颐堂NetDevOps CICD测试

Start the title with **Draft:** to prevent a merge request draft from merging before it's ready.  
Add [description templates](#) to help your contributors to communicate effectively!

**Description**

Write Preview

2 乾颐堂NetDevOps CICD测试

Mark down and quick actions are supported [Attach a file](#)

3 **Assignee** Administrator

4 **Reviewer** Administrator

Milestone Milestone

Labels Labels

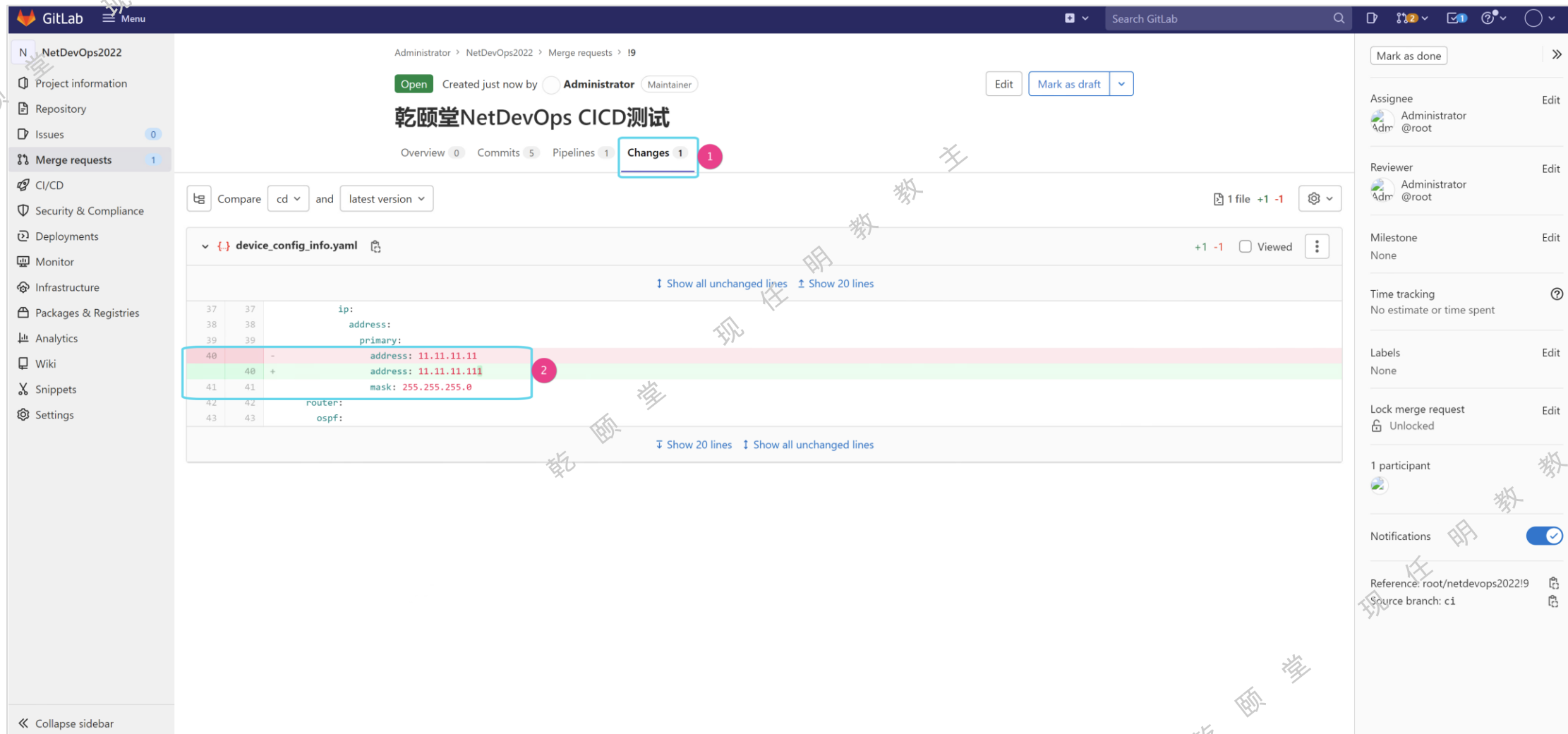
Merge options  Squash commits when merge request is accepted. ?

5 **Create merge request** Cancel

# 管理员查看change

The screenshot shows the GitLab Merge Request interface for a project named 'NetDevOps2022'. The main content area displays the merge request details for '乾颐堂NetDevOps CICD测试', created by 'Administrator'. The 'Changes' tab is selected, showing a request to merge the 'ci' branch into the 'cd' target branch. The source branch is 5 commits behind the target. A pipeline (#82) is shown as passed for commit 06a86610 on the 'ci' branch. The interface includes an 'Approve' button, a 'Merge' button, and a 'Squash commits' option. The right sidebar contains settings for assignees, reviewers, milestones, and labels. The bottom section shows a comment area with a rich text editor.

# 管理员查看change



The screenshot shows the GitLab Merge Request interface for a project named "NetDevOps2022". The merge request is titled "乾颐堂NetDevOps CICD测试" and is created by "Administrator". The "Changes" tab is selected, showing a diff for the file "device\_config\_info.yaml". The diff shows a change in the "address" field of the "ip" section, from "11.11.11.11" to "11.11.11.111". The change is highlighted in green, and a red box is drawn around it. A red circle with the number "2" is placed next to the change. A red circle with the number "1" is placed next to the "Changes" tab. The interface also shows a sidebar with navigation options, a top navigation bar, and a right sidebar with various settings and options.

Administrator > NetDevOps2022 > Merge requests > 19

Open Created just now by Administrator (Maintainer)

Edit Mark as draft

### 乾颐堂NetDevOps CICD测试

Overview 0 Commits 5 Pipelines 1 **Changes 1**

Compare cd and latest version 1 file +1 -1

device\_config\_info.yaml +1 -1 Viewed

Show all unchanged lines Show 20 lines

37	37		ip:
38	38		address:
39	39		primary:
40	-		address: 11.11.11.11
40	+		address: 11.11.11.111
41	41		mask: 255.255.255.0
42	42		router:
43	43		ospf:

Show 20 lines Show all unchanged lines

Mark as done

Assignee Administrator (Adm) @root Edit

Reviewer Administrator (Adm) @root Edit

Milestone None Edit

Time tracking No estimate or time spent

Labels None Edit

Lock merge request Unlocked Edit

1 participant

Notifications

Reference: root/netdevops2022:19

Source branch: ci

# 代码审核 Approval

The screenshot displays the GitLab Merge Request interface for a project named 'NetDevOps2022'. The main content area shows a merge request titled '乾颐堂NetDevOps CICD测试' (Qianruo Academy NetDevOps CICD Test). The status is 'Open', created by 'Administrator' (Maintainer). The merge request is for merging branch 'c1' into 'cd'. The source branch is 5 commits behind the target branch. A pipeline #82 passed for commit 06a86610 on branch c1 1 minute ago. The interface includes a 'Merge' button, a 'Squash commits' option, and a 'Revoke approval' button. The 'Approved by you' section shows the user's approval. The right sidebar contains settings for the merge request, including assignee, reviewer, milestone, time tracking, labels, lock merge request, and notifications. The bottom section shows a list of activities, including the user's approval.

# 管理员 Merge

The screenshot shows the GitLab Merge Request interface for a project named 'NetDevOps2022'. The interface includes a left sidebar with navigation options like 'Project information', 'Repository', 'Issues', 'Merge requests', 'CI/CD', 'Security & Compliance', 'Deployments', 'Monitor', 'Infrastructure', 'Packages & Registries', 'Analytics', 'Wiki', 'Snippets', and 'Settings'. The main content area displays the merge request details, including a pipeline status, approval information, and a list of activities. A 'Merge' button is highlighted with a red circle and the number '4'. Below it, a thumbs-up icon is highlighted with a red circle and the number '1'. A text input field for a comment is highlighted with a red circle and the number '2', containing the text 'good work'. At the bottom, a 'Comment' button is highlighted with a red circle and the number '3'. The right sidebar shows settings for the merge request, such as 'Assignee', 'Reviewer', 'Milestone', 'Time tracking', 'Labels', 'Lock merge request', and 'Notifications'.

# 查看Pipeline

The screenshot shows the GitLab interface for the 'NetDevOps2022' project. The 'Pipelines' section is active in the left sidebar. The main area displays a list of pipeline runs. The top row, representing pipeline #83, is highlighted with a red box and a red circle containing the number '3'. The pipeline is in a 'passed' state. The table below shows the details of several pipeline runs.

Status	Pipeline ID	Triggerer	Commit	Stages	Duration
passed	#83	root's vate	cd -> 62313140 Merge branch 'ci' into 'cd'	5 stages (all passed)	00:01:02 6 minutes ago
passed	#82	Administrator's vatar	ci -> 66a86610 CI测试	5 stages (all passed)	00:00:42 9 minutes ago
passed	#81	Administrator's vatar	ci -> 373602e7 CI测试	5 stages (all passed)	00:00:48 22 minutes ago
passed	#80	Administrator's vatar	ci -> 9e88f45f CI测试	5 stages (all passed)	00:01:05 31 minutes ago
passed	#79	root's vate	cd -> 9283d5b4 Merge branch 'ci' into 'cd'	5 stages (all passed)	00:00:54 6 days ago
passed	#78	Administrator's vatar	ci -> dbf64c2f 最终快照	5 stages (all passed)	00:00:49 6 days ago