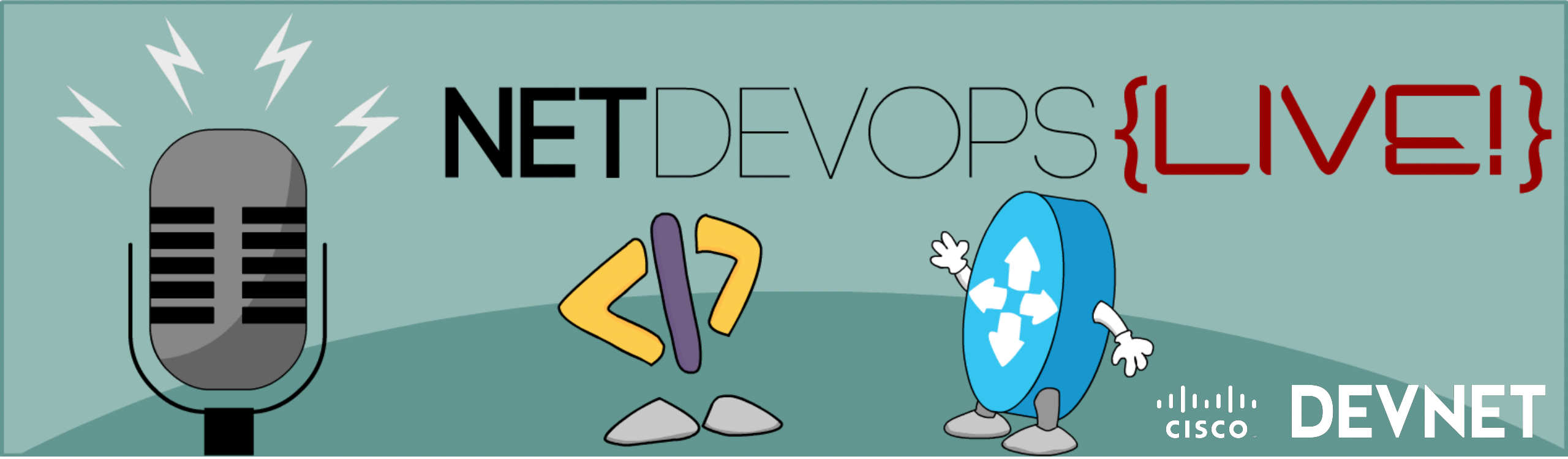


The Truth! Your Network WILL Handle the Truth with NetBox

Jeremy Stretch
Network to Code
Twitter: @jstretch85

Season 3, Talk 6
Hosted by Hank Preston, NetDevOps Engineer
Twitter: @hfpreston
https://developer.cisco.com/netdevops/live

>>> network .toCode()

NetBox as a Source of Truth

NetDevOps Live!

May 12, 2020

What is a Source of Truth (SoT)?

- Designated authority over a specific data domain
  - Where to find the “correct” value for a piece of data
- Traditional examples:
  - IPAM server
  - DNS server
  - Spreadsheets
- Crucial that sources of truth never overlap

SoT in Practice

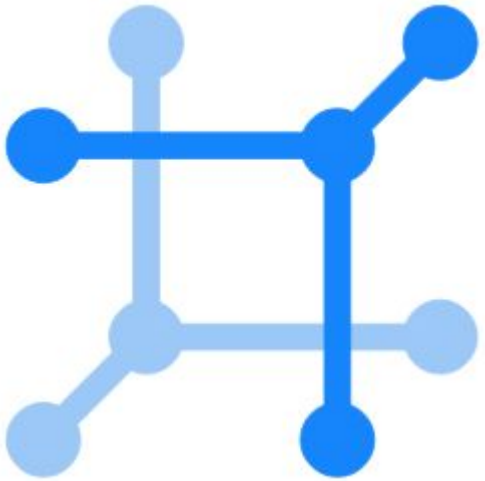
Legacy Approach

- Configure hosts in each tool individually
- Import from one tool to another depending on availability of information
- No checks in place to ensure consistent data across all tools

SoT Approach

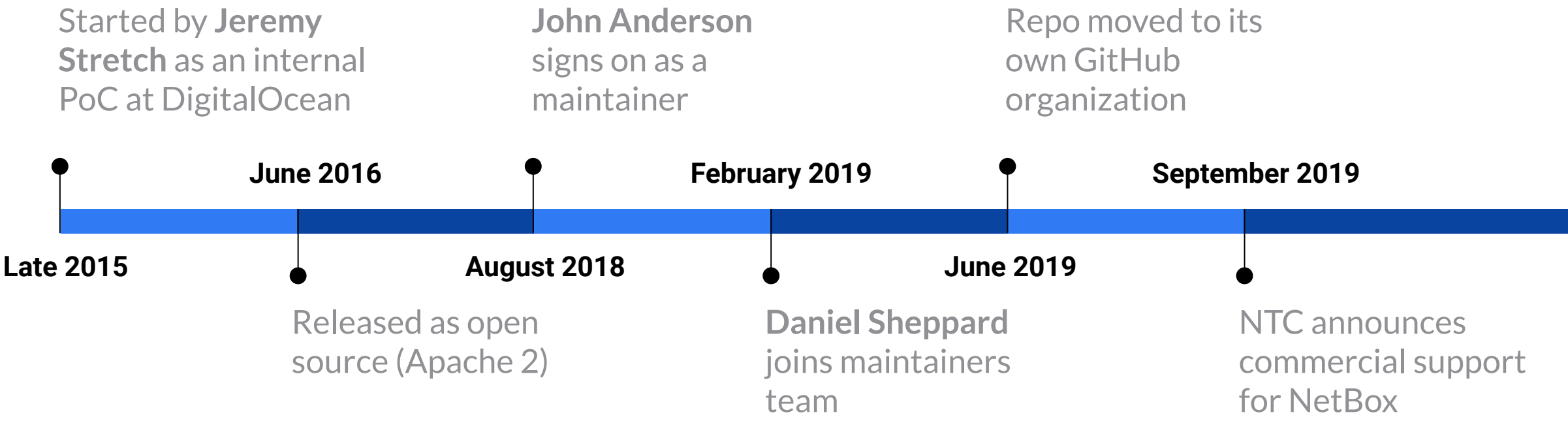
- Data is only entered into the SoT
- All tools populate directly from the SoT
  - Can be pull or push
  - Precise method of population may vary among tools
- The data within each tool can be validated against the SoT at any time

What is NetBox?



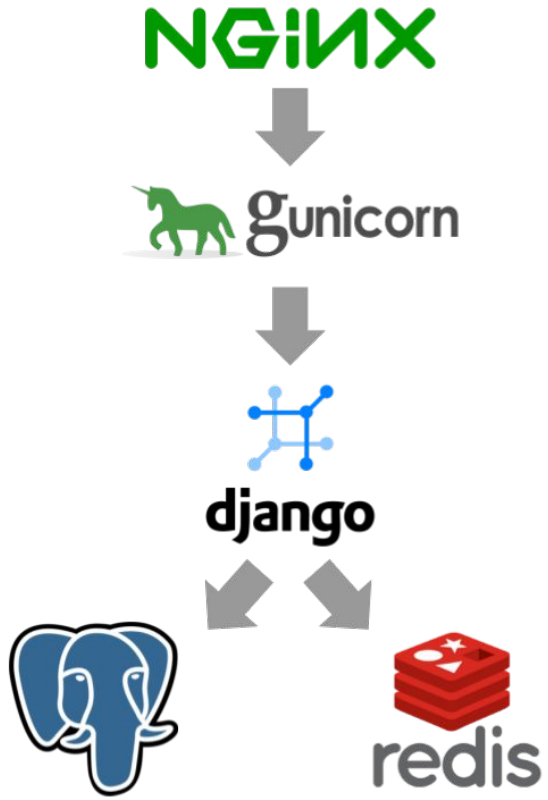
- A SoT for various domains of network infrastructure
  - IPAM, DCIM, circuits, tenancy, etc.
- Provides a structured data model with integration points
  - REST API, webhooks, plugins, etc.
- Written in Python using the Django web framework
- Open source and freely available
  - Apache 2.0 license

NetBox History



Application Architecture

HTTP daemon	**nginx or Apache** Handles HTTP requests
Application	**Django/Python** Framework for application logic
WSGI	**gunicorn or uWSGI** WSGI middleware
Database	**PostgreSQL** Relational database
Queuing/caching	**Redis** In-memory key-value store



Design Philosophy

- Replicate the real world
  - Data model is tightly coupled to real-world constraints
- Function as a source of truth for the network
  - Use NetBox to provision devices, not vice versa
- Keep things simple
  - High value and ease of maintenance are preferred over 100% complete solutions

Functional Scope

- Number resources (IPAM)
  - Prefixes, IPs, VRFs, VLANs, AS numbers
- Physical infrastructure
  - Sites, racks, devices, cables, power
- Virtual machines and clusters
- Data circuits
- Secrets (e.g. login credentials)

Some Things NetBox Doesn't Do

- DHCP server
- DNS server
- Asset tracking
- Authentication server (e.g. RADIUS/LDAP)
- Network monitoring
- Device orchestration

Why Use NetBox?

- Open source
  - Actively maintained and developed
- Emphasis on programmability and integration
- Strong support community
- Extensible

Integration with Other Tools

- REST API
  - Push and pull data in JSON format over HTTP
- Webhooks
  - Send HTTP requests to other tools in response to events
- Django ORM/Python shell
- Custom scripts
  - Execute arbitrary Python directly from the NetBox UI/API

REST API

```
# Send a POST request with the desired object attributes
$ curl -X POST \
-H "Authorization: Token $TOKEN" \
-H "Content-Type: application/json" \
-H "Accept: application/json; indent=4" \
https://netbox/api/ipam/prefixes/ \
--data '{"prefix": "192.0.2.0/24", "status": "active"}'

# Receive a HTTP 201 CREATED response
{
    "id": 18686,
    "family": {
        "value": 4,
        "label": "IPv4"
    },
    "prefix": "192.0.2.0/24",
    "status": {
        "value": "active",
        "label": "Active",
    }
}
```

API Root / IPAM / Prefix

Prefix

GET

« 1 2 3 ... 191 »

```
GET /api/ipam/prefixes/?within=10.0.0.0/8
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "count": 9506,
    "next": "http://localhost:8000/api/ipam/prefixes/?limit=50&offset=50&within=10.0.0.0%2F8",
    "previous": null,
    "results": [
        {
            "id": 16743,
            "family": {
                "value": 4,
                "label": "IPv4"
            },
            "prefix": "10.0.0.0/24",
            "site": null,
            "vrf": null,
```

Webhooks

- Created under NetBox admin UI for specific object types
- Fire in response to objects being created, updated, and/or deleted
- Request headers and body can be templated using Jinja2

Python Shell/Django ORM

```
# Enter the NetBox shell
$ ./manage.py nbshell
### NetBox interactive shell (demohost)
### Python 3.6.9 | Django 3.0.8 | NetBox 2.8.1
### lsmodels() will show available models. Use help(<model>) for more info.
>>>

# Retrieve a device
>>> router = Device.objects.get(name='dc1-edge1')
>>> router
<Device: dc1-edge1>
>>> router.device_type
<DeviceType: MX240>

# Work with related objects
>>> router.interfaces.count()
42

# Modify and save
>>> router.serial = 'FAC98210'
>>> router.save()
```

Custom Scripts

- Arbitrary Python scripts uploaded to NetBox to automate tasks
  - Ex: Populate base devices, prefixes, etc. for a new site
  - Ex: Detect and automatically correct data discrepancies
- Leverages the Django ORM to obviate direct database access
- Can be executed via web UI or REST API
- Custom reports
  - Similar to scripts but focused on validation

New Branch

Provision a new branch site

Run | Output | Source

Script Data

Prefix ---------

Select Parent Prefix to be assigned.

Site name

Name of the new site

Switch count

Number of access switches to create

Switch model ---------

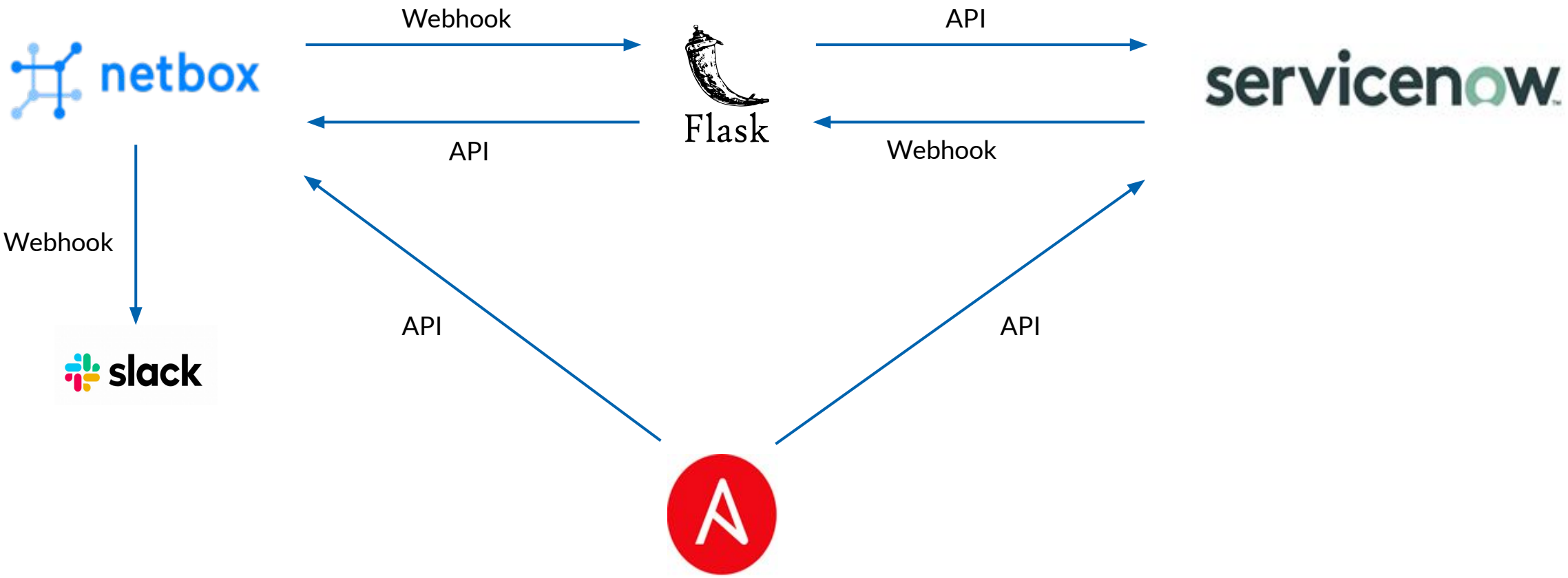
Access switch model

☑ Commit changes

Commit changes to the database (uncheck for a dry-run)

▶ Run Script | Cancel

Integration Examples



NetBox & Network to Code

- NTC offers commercial support for NetBox
  - Help with install, maintenance, custom development, etc
- We also offer customised training and professional services
  - Custom plugin development, custom scripts, reports, SoT integrations, etc.

Contact info@networktocode.com

Resources

github.com/netbox-community/netbox

Code

netbox.readthedocs.io/

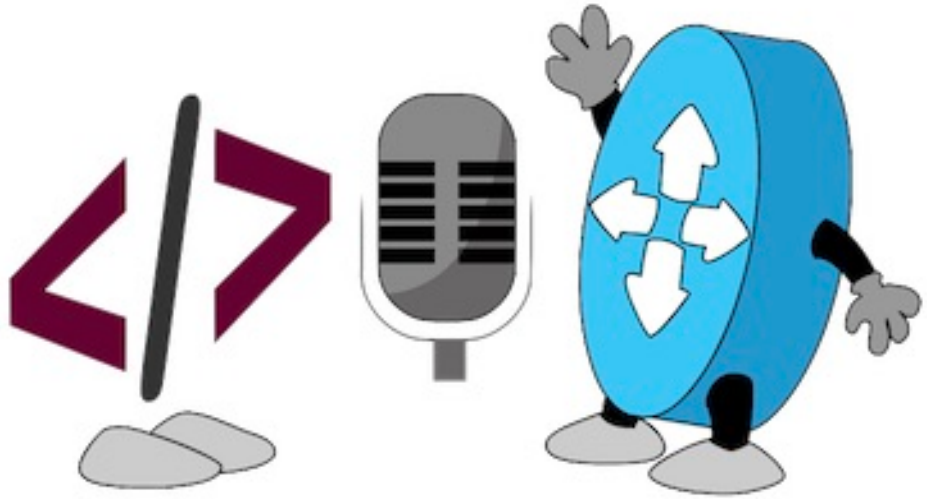
Documentation

#netbox on networktocode.slack.com

netbox-discuss@googlegroups.com

Community

NetDevOps Tech Chat

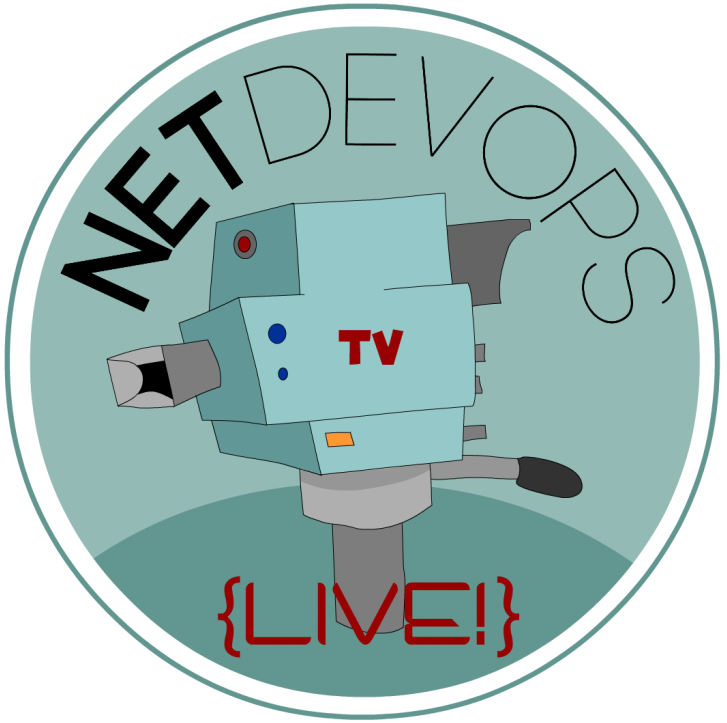


DEVNET
developer.cisco.com

</finish>

Webinar Resources on DevNet!

- Docs and Links
- Learning Labs
- DevNet Sandboxes
- Code Samples



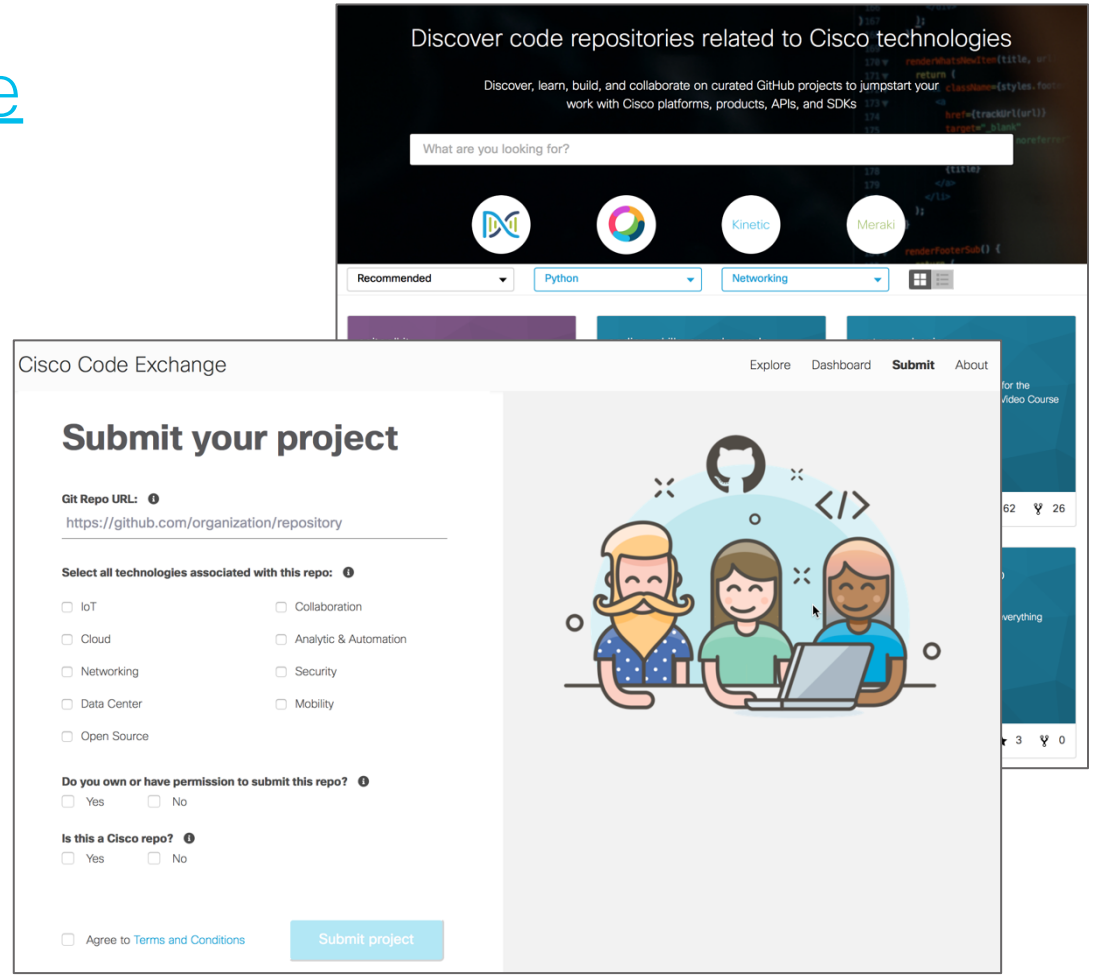
developer.cisco.com/netdevops/live/#s03t06

NetDevOps Live! Code Exchange Challenge

developer.cisco.com/codeexchange

Use NetBox as a Source of Truth in an automation project. Use it to drive configuration, or verify it. Your call!

Example: Check to see if interface descriptions configured on a switch match what's listed in NetBox



Looking for more about NetDevOps?

- NetDevOps on DevNet
  developer.cisco.com/netdevops
- NetDevOps Live!
  developer.cisco.com/netdevops/live
- NetDevOps Blogs
  blogs.cisco.com/tag/netdevops
- Network Programmability Basics Video Course
  developer.cisco.com/video/net-prog-basics/

Join us next week!

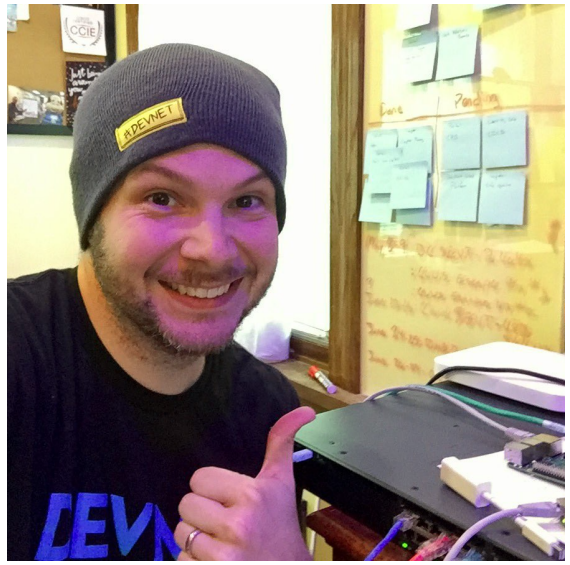
Logging and Back Again – A Network Engineers Journey with ELK

May 19, 2020 with George Kobar

https://developer.cisco.com/netdevops/live/#s03t07

DEVNET
developer.cisco.com

Got more questions? Stay in touch!



Hank Preston

hapresto@cisco.com

@hfpreston

http://github.com/hpreston

CISCO DEVNET

LEARN CODE INSPIRE CONNECT

developer.cisco.com

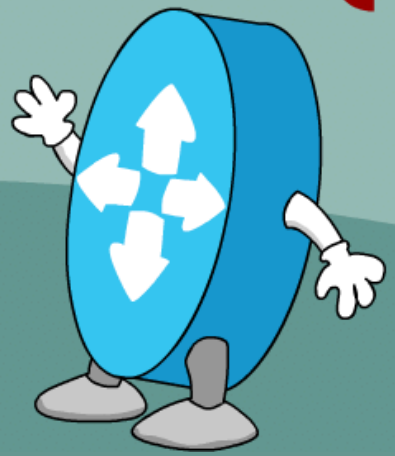
@CiscoDevNet

facebook.com/ciscodevnet/

http://github.com/CiscoDevNet

NETDEVOPS {LIVE!}

cisco DEVNET



https://developer.cisco.com/netdevops/live

@netdevopslive