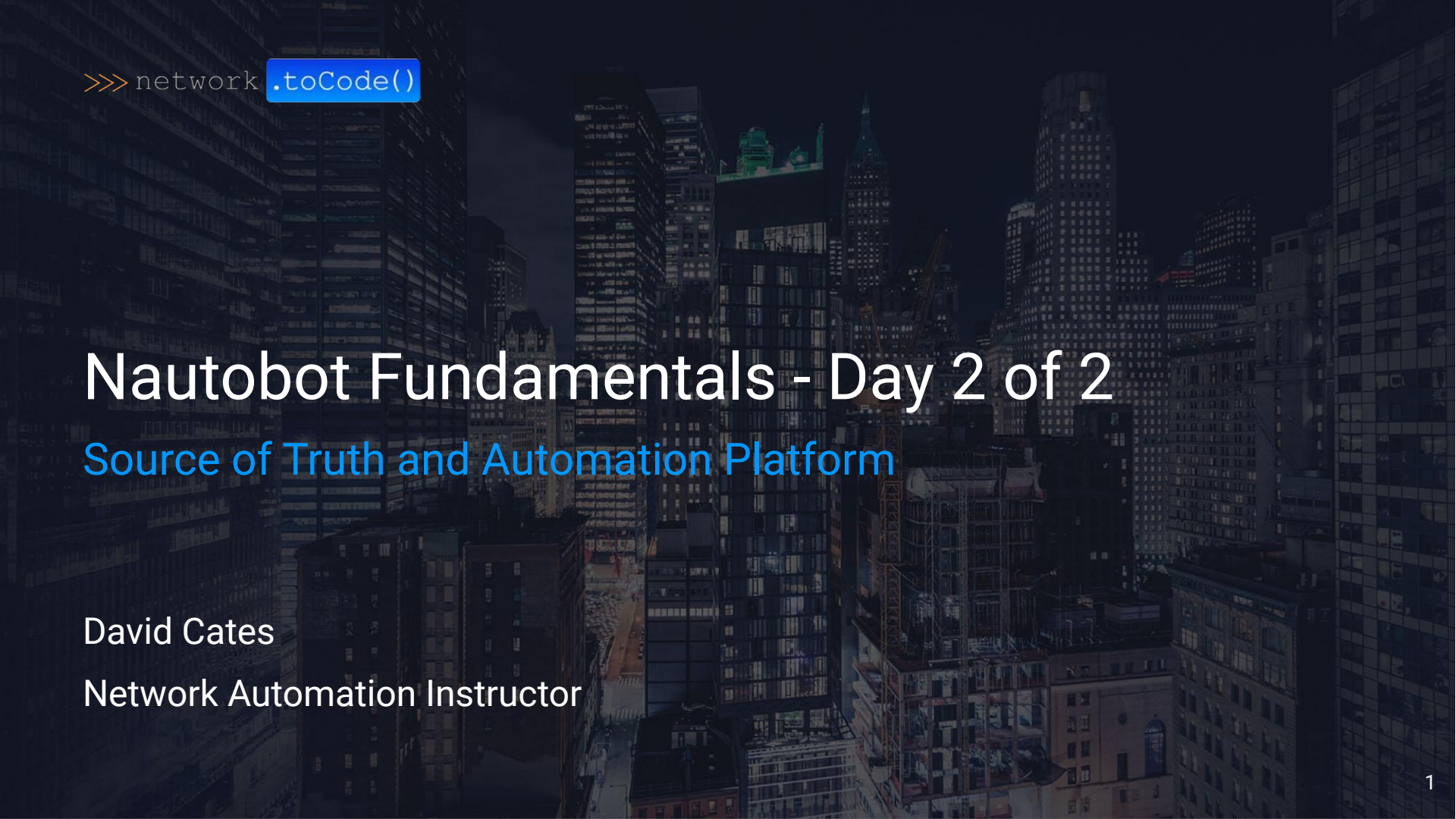


Nautobot Fundamentals - Day 2 of 2

Source of Truth and Automation Platform

David Cates

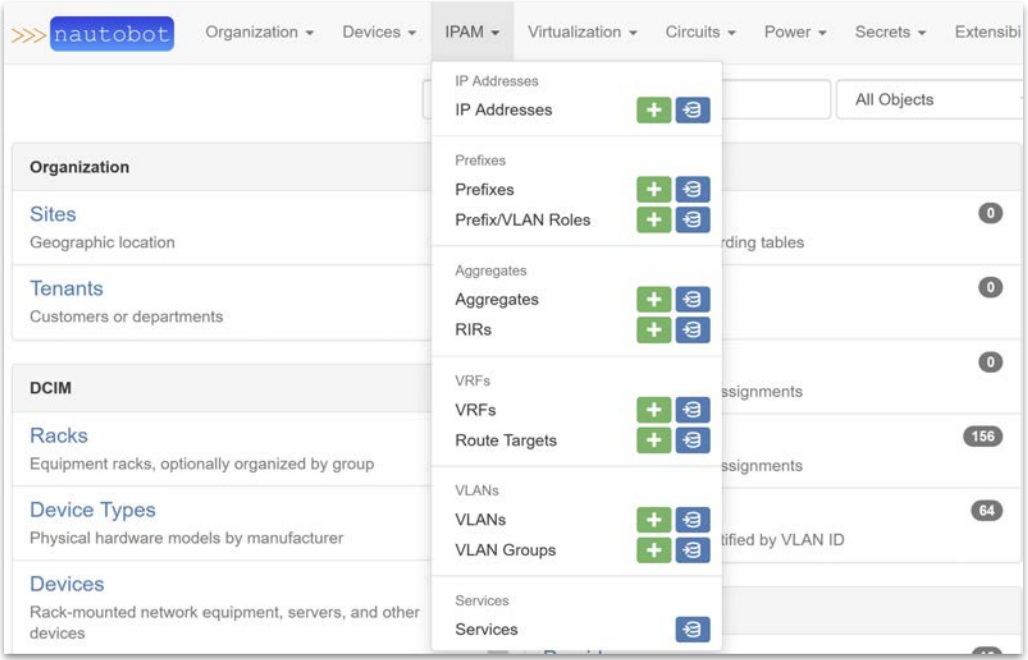
Network Automation Instructor

Course Agenda

1. Why Network Automation?
2. Nautobot Overview and Installation
3. Nautobot Organization and Devices
4. Nautobot IPAM to Circuits
5. Nautobot Apps
6. Nautobot Programming
7. Nautobot Extensibility

IPAM

>>> IPAM from the Browser



- IPAM component links are available:
  - On the Nautobot home page or
  - From the IPAM menu tab
- Major subsections:
  - IP Addresses
  - Prefixes
  - Aggregates
  - VRFs
  - VLANs
  - Services

>>> Prefixes, IP addresses, and VLANs

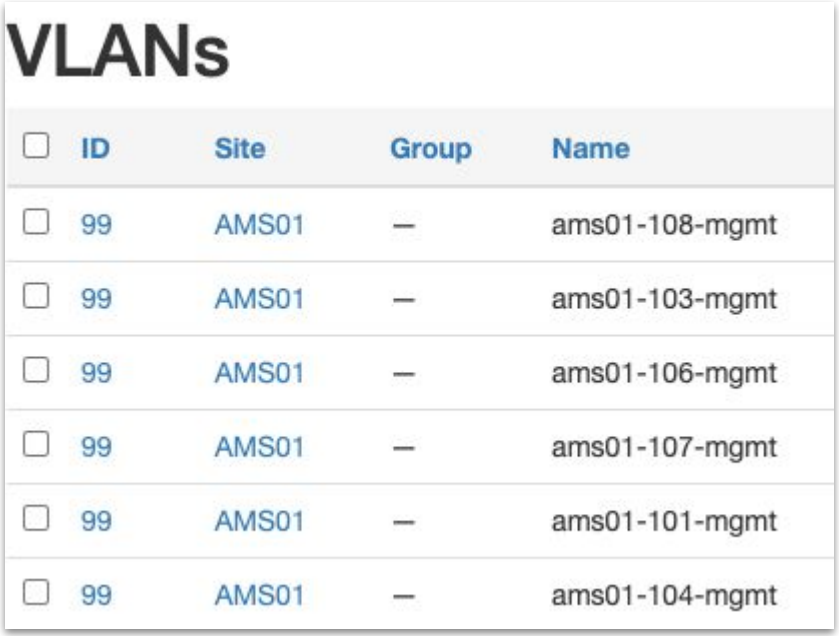
Prefixes

Prefix	Status	Children	VRF	Utilization	Tenant
10.0.0.0/8	Container	1309	Global	13%	Nautobot Baseball Stadiums
10.0.0.0/8	Container	1309	Global	13%	Nautobot Airports
• • 10.0.0.0/16	Container	38	Global	100%	Nautobot Airports
• • • 10.0.0.0/18	Container	8	Global	12%	Nautobot Airports
• • • • 10.0.0.0/24	Active	0	Global	0%	Nautobot Airports
• • • • 10.0.1.0/24	Active	0	Global	0%	Nautobot Airports
• • • • 10.0.2.0/24	Active	0	Global	0%	Nautobot Airports

- Prefixes can be containers or subnets
- Organization of prefixes and addresses is automatic
- IPs are assigned to interfaces which belong to devices

>>> Prefixes, IP addresses, and VLANs

Nautobot Core Models



VLANs

	ID	Site	Group	Name
☐	99	AMS01	—	ams01-108-mgmt
☐	99	AMS01	—	ams01-103-mgmt
☐	99	AMS01	—	ams01-106-mgmt
☐	99	AMS01	—	ams01-107-mgmt
☐	99	AMS01	—	ams01-101-mgmt
☐	99	AMS01	—	ams01-104-mgmt

- VLANs can be organized into VLAN Groups
- VLANs or VLAN groups can be assigned to sites

Demo 6

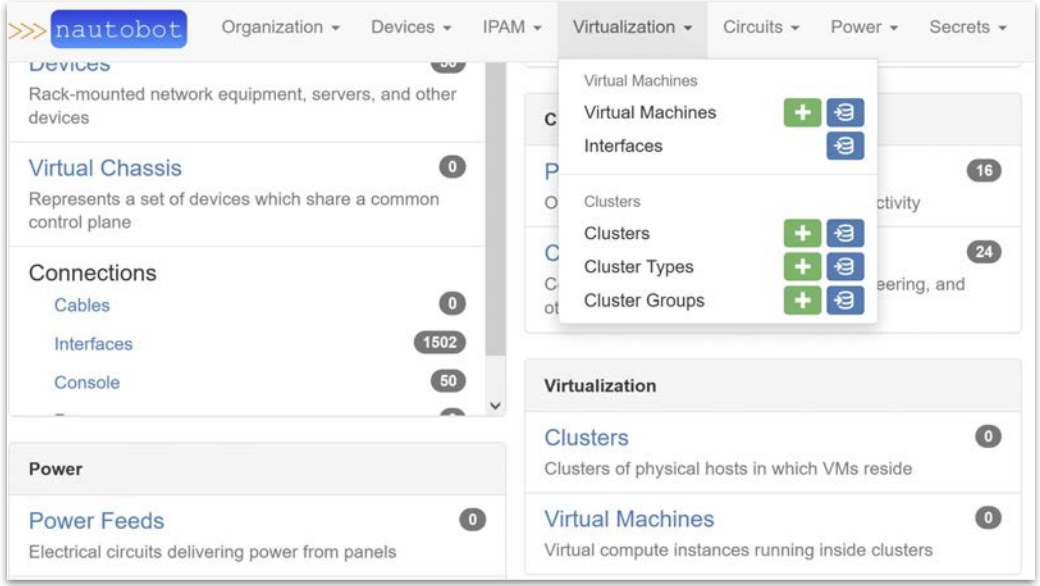
Modeling IPAM

Lab 4

Modeling IPAM

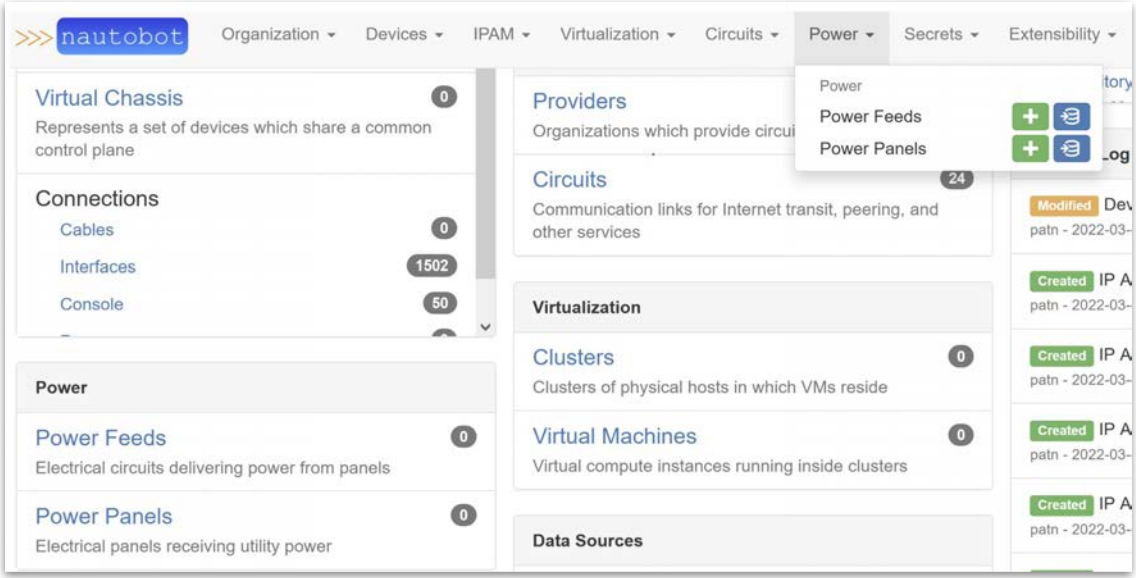
Virtualization, Power, Secrets, & Circuits

>>> Virtualization from the Browser



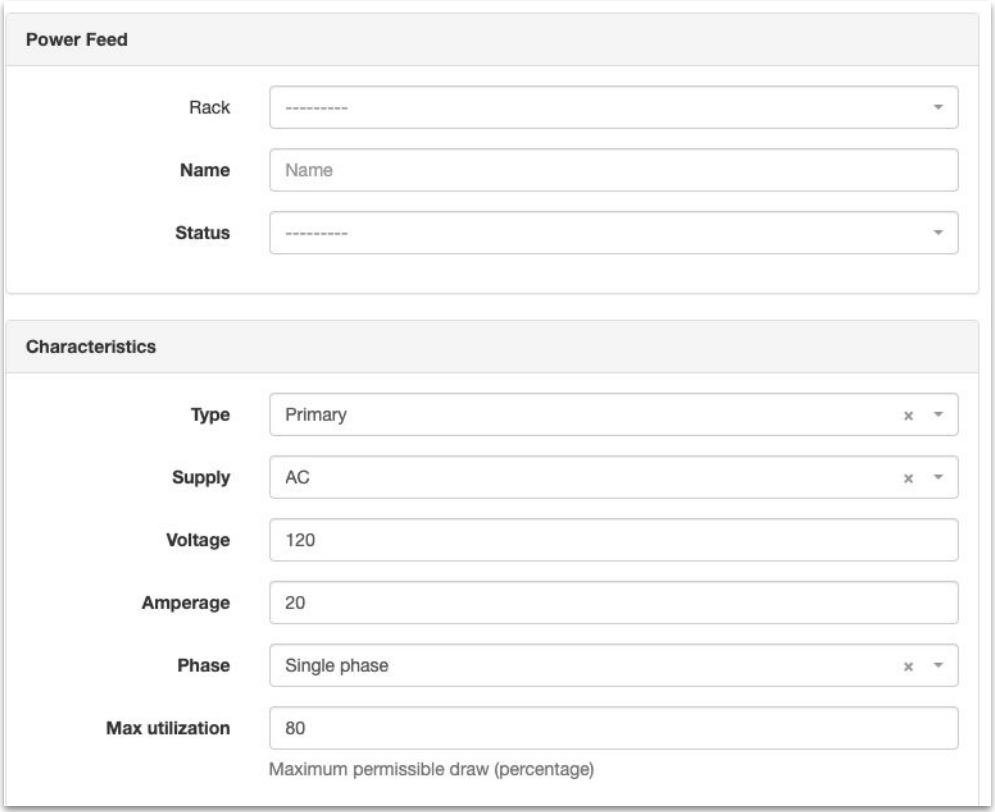
- Virtualization component links are available:
  - On the Nautobot home page or
  - From the Virtualization menu tab
- Major subsections:
  - Virtual Machines
  - Clusters

>>> Power from the Browser



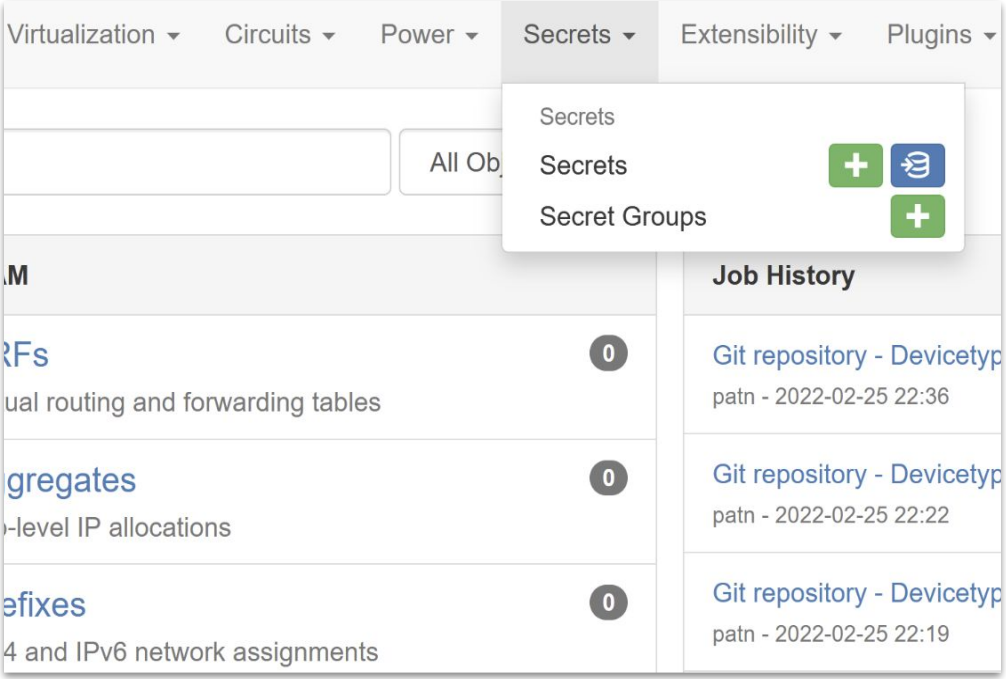
- Power component links are available:
  - On the Nautobot home page
  - From the Power menu tab
- Major subsection:
  - Power

>>> Power



- Power includes **Power Feeds** and **Power Panels**
- They are used to keep track of power utilization

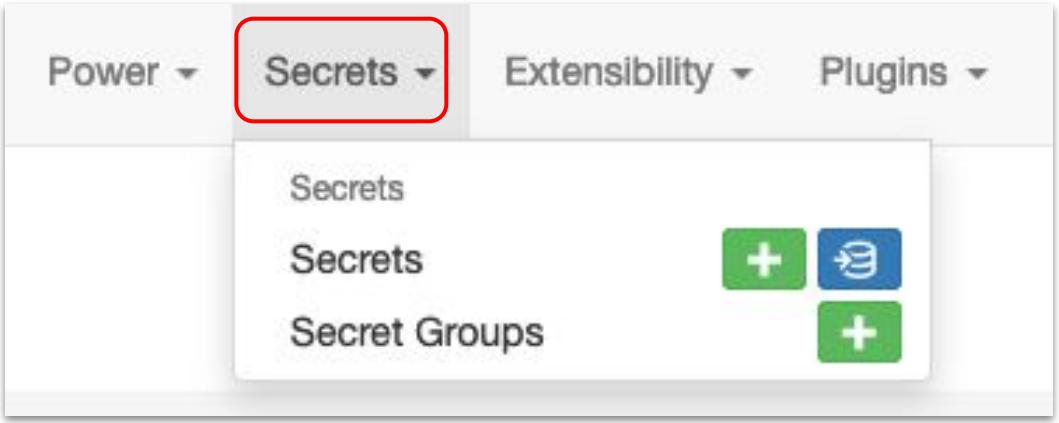
>>> Secrets from the Browser



- Secrets component links are only available from the Secrets menu tab
- Major subsections include:
  - Secrets
  - Secret Groups

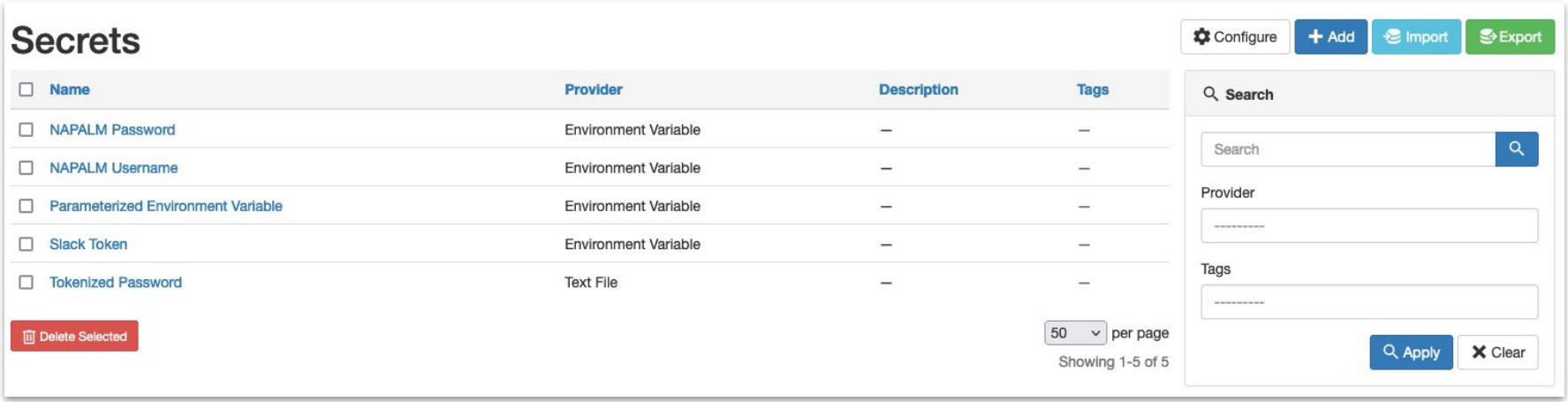
>>> Secrets Integration

- Secure retrieval and usage of arbitrary secrets
- Allows for industry standard secrets storage tooling
- Secrets are not stored directly in Nautobot



>>> Secrets Providers

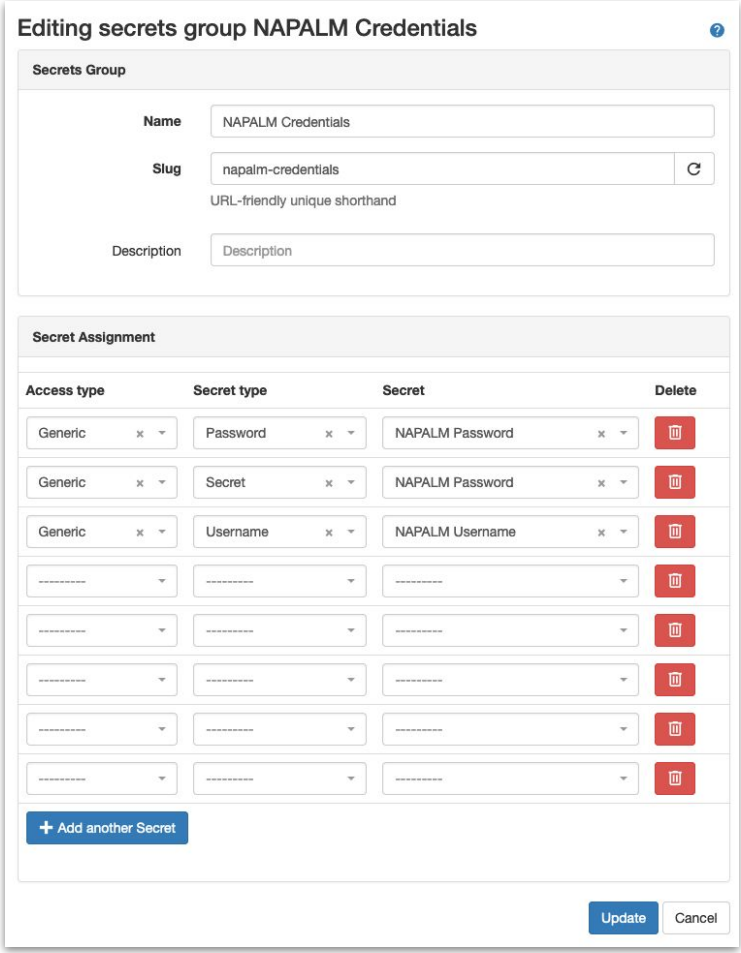
- Introduces *Secret* model
  - Defines how Nautobot can **retrieve** secret value and use when needed
  - Does **not** store secret value



>>> Secrets Integration

- Introduces *Secrets Group* model
- Collects and assigns meaning to secrets

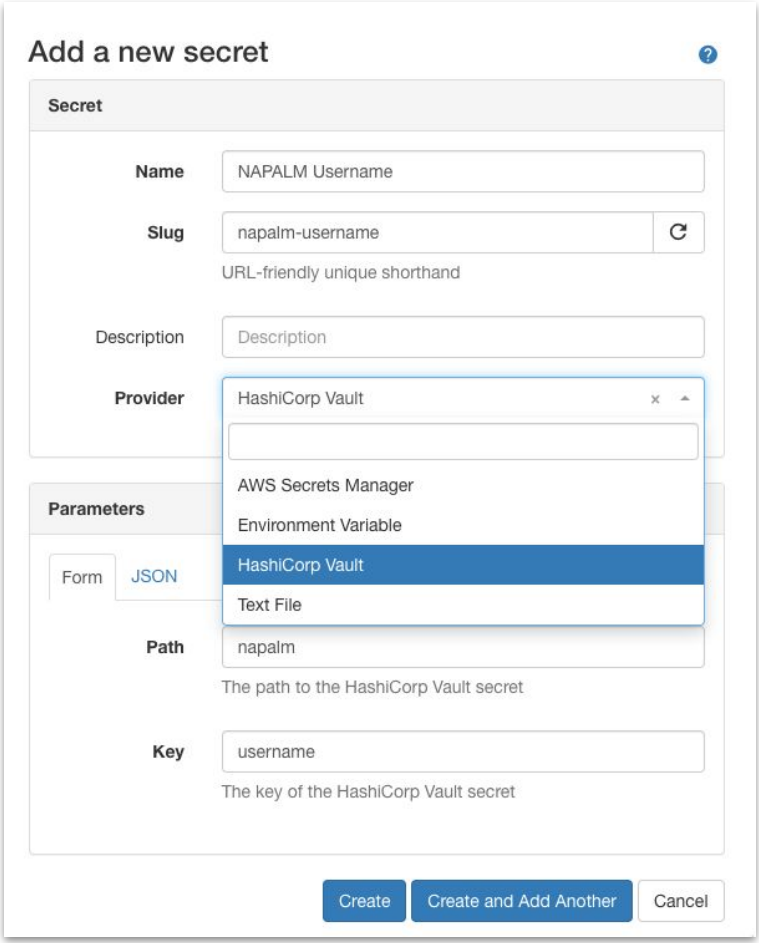
- The example to the right shows a Secrets Group consisting of three NAPALM credentials needed to access and configure a device
  - Username
  - Password
  - Secret



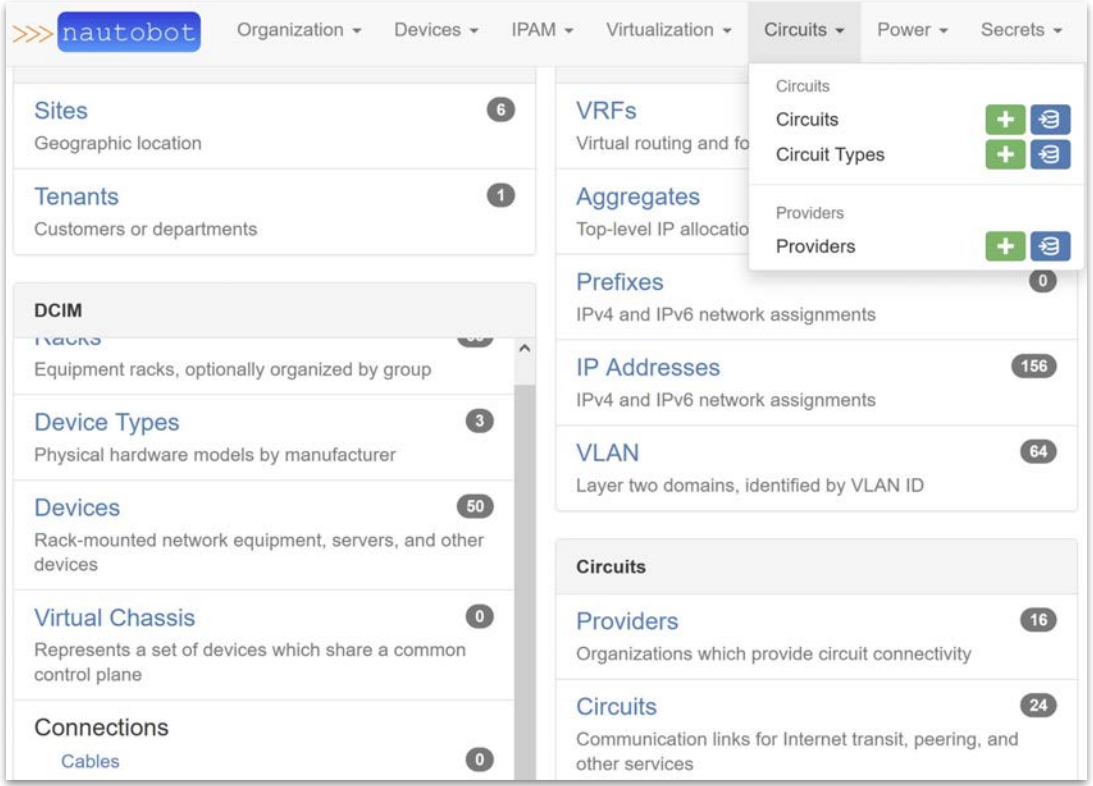
>>> Secrets Integration

- The feature's core code enables these two secret sources:
  - Environment variable
  - Text file

- This feature also enables the *Nautobot Secrets Providers* app:
  - Defines additional secret providers
  - This app initially includes AWS Secrets Manager and HashiCorp Vault

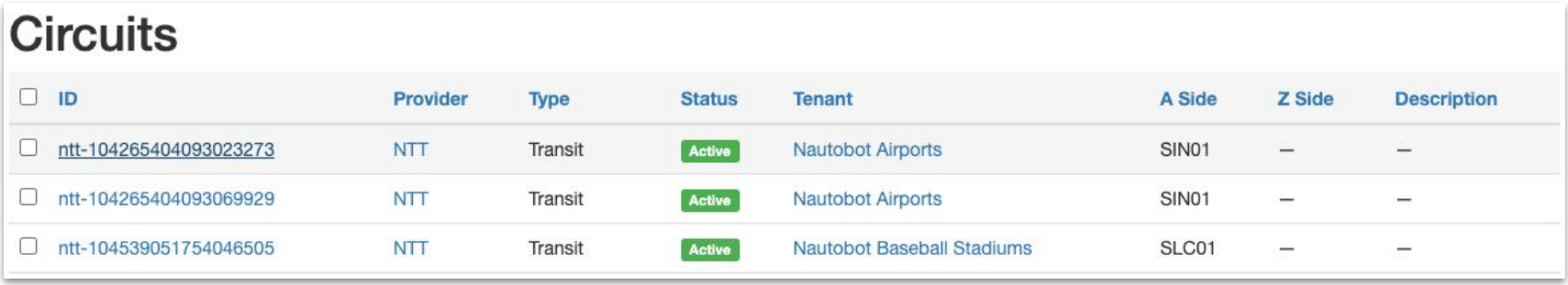


>>> Circuits from the Browser



- Circuits component links are available:
  - On the Nautobot home page
  - From the Circuits menu tab
- Major subsections:
  - Circuits
  - Providers

>>> Circuits



Circuits

	ID	Provider	Type	Status	Tenant	A Side	Z Side	Description
☐	ntt-104265404093023273	NTT	Transit	Active	Nautobot Airports	SIN01	—	—
☐	ntt-104265404093069929	NTT	Transit	Active	Nautobot Airports	SIN01	—	—
☐	ntt-104539051754046505	NTT	Transit	Active	Nautobot Baseball Stadiums	SLC01	—	—

- **Circuits** are objects representing some physical or virtual connectivity between two endpoints
- **Circuits** require a **Circuit Type** and a **Circuit Provider**

Demo 7

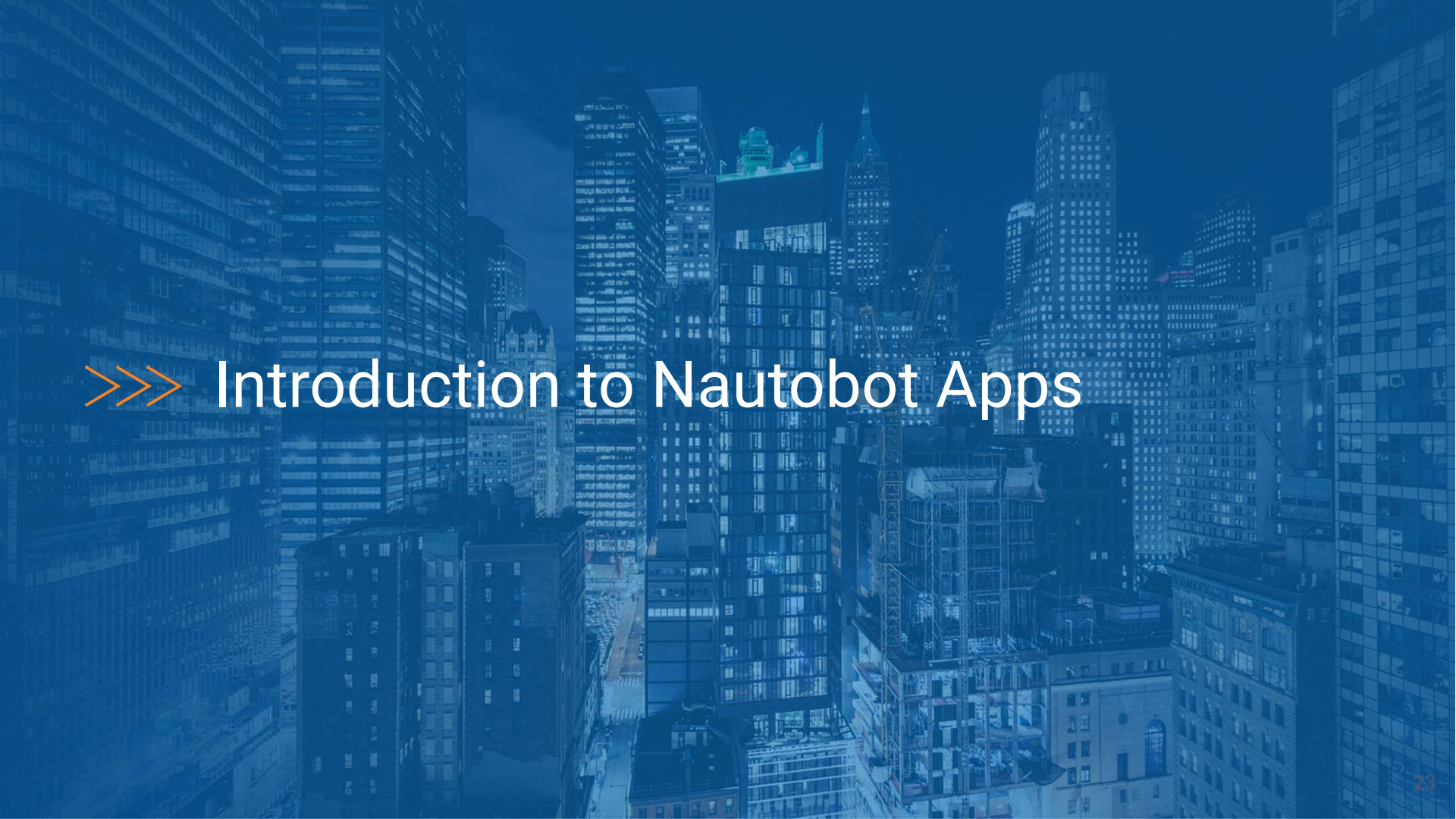
Learning the Circuits Model

Lab 5

Learning the Circuits Model

Course Agenda

1. Why Network Automation?
2. Nautobot Overview and Installation
3. Nautobot Organization and Devices
4. Nautobot IPAM to Circuits
5. Nautobot Apps
6. Nautobot Programming
7. Nautobot Extensibility



Introduction to Nautobot Apps

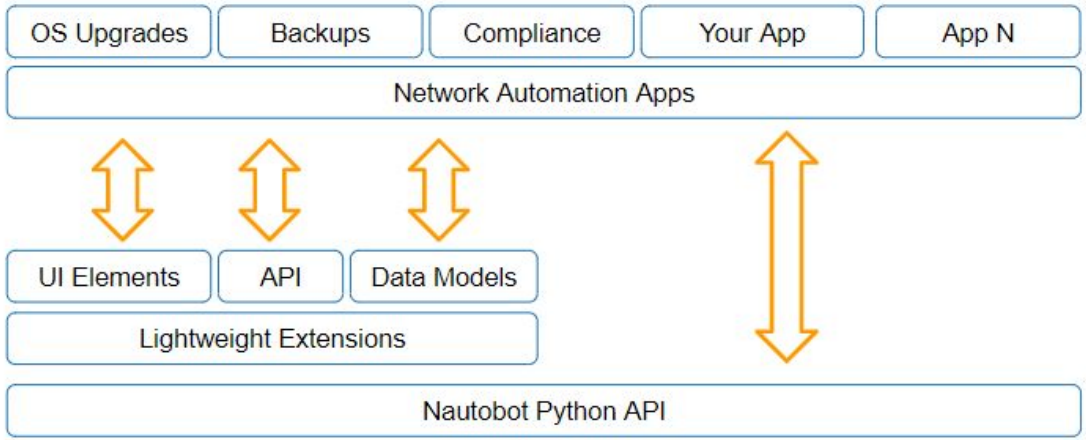
>>> What is an App?

- Python packages which extend Nautobot's functionality in some way
  - Packaged as reusable Django apps
- Apps facilitate the extension of Nautobot without needing to fork the code base
- Formerly (and still sometimes) referred to as plug-ins
- Can be installed/removed as needed
- Apps may be developed privately or published for public use
  - Ex: https://github.com/nautobot

>>> Where to Apps Fit Logically?

- Apps allow developers to invent and implement entirely new functionality
- Apps break into two levels of overall functionality, **extensions** and **apps**
- Apps can provide:
  - Models (and integration with core)
  - Inject content into core pages
  - Views
  - REST APIs
  - Data Validation

Apps are Python packages which are installed by the Nautobot user.



>>> What Can an App Do?

- Create custom Django models
  - Define new database tables
  - Generate and run SQL database migrations
- Provide custom “pages” within the UI
  - Implement custom business logic
  - Use Django Template Language(DTL) or Jinja2 to render templates
  - Register custom URLs under the /plugins path

>>> What Can an App Do?

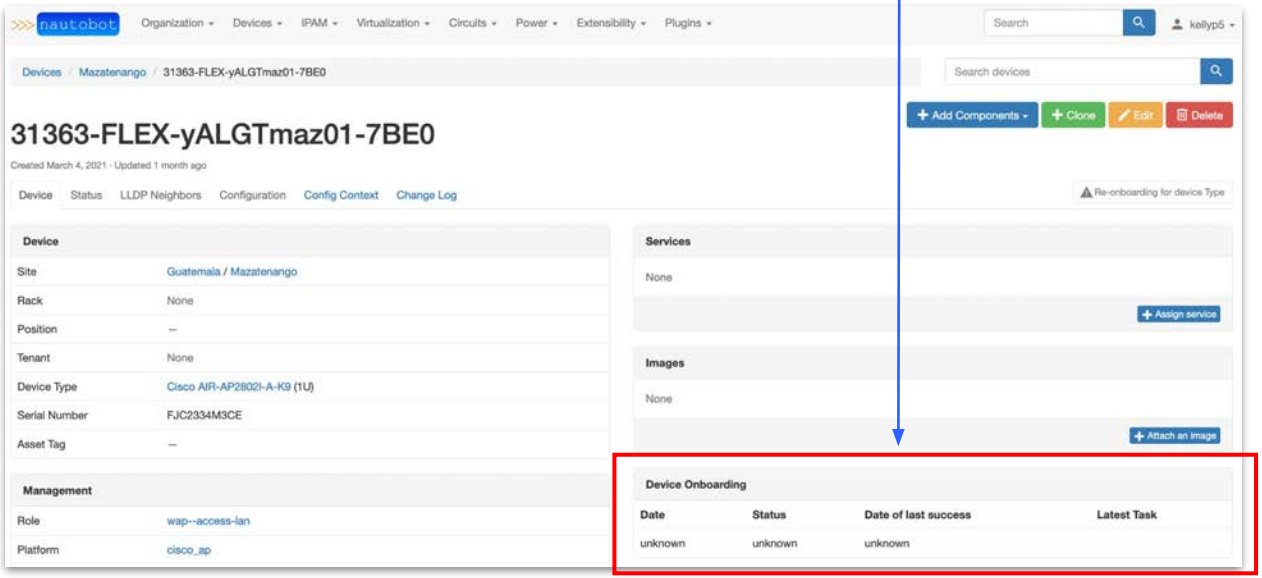
- Inject custom content within certain stock views
  - Embed custom content/buttons in prescribed locations within core object views
- Add REST API endpoints
  - Provide full CRUD support for your plugin's models
- Provide `Jobs`
- Leverage the complete Python environment and Django ORM

How to Inject Content in the UI?

- Add existing data to existing, but different UI pages
- Add new data to current UI pages
- Add new data to new UI pages
- Create wrapper APIs

Injected Content

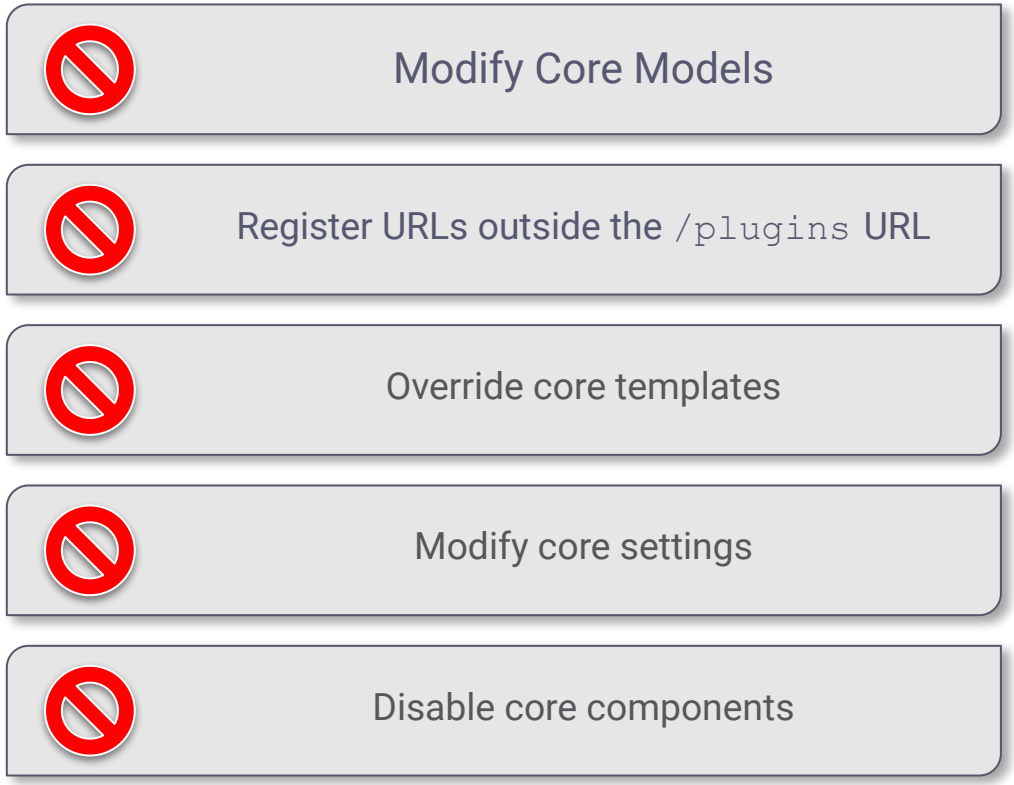
Existing Device Page



>>> App Examples

- **nautobot-plugin-chatops**
  - Framework to enable chatbots and nautobot chat commands
- **nautobot-plugin-device-onboarding**
  - Import data from live network devices to Nautobot
- **nautobot-plugin-capacity-metrics**
  - Expose prometheus metrics for nautobot server
- **nautobot-plugin-ssot**
  - Synchronize sources of record with Nautobot
- **nautobot-plugin-circuit-maintenance**
  - Manage circuit maintenance notifications

>>> App Limitations - What should an app NOT do?



>>> App Installation

1. Log into the `nautobot` user account
2. Install the Python package

```
$ sudo -iu nautobot

$ pip3 install nautobot-chatops
```

>>> App Installation

1. In your `nautobot_config.py`, add the app name to the PLUGINS list:
2. If the plugin requires any configuration, define it in `nautobot_config.py` under the PLUGINS_CONFIG parameter.
   - The available configuration parameters should be detailed in the plugin's README file.

```
PLUGINS = [
    “nautobot_chatops”,
]

PLUGINS_CONFIG = {
    “nautobot_chatops”: {
        “foo”: “bar”,
        “buzz”: “bazz”
    }
}
```

App Installation

1. Run `nautobot-server post_upgrade`

   This command will ensure any post-installation tasks are run, e.g.

   - Migrating the database to include any new or updated data models from the app
   - Collecting any static files provided by the app

```
# nautobot-server post_upgrade
Performing database migrations...
Operations to perform:
  Apply all migrations: admin, auth, circuits,
contenttypes, db, dcim, extras, ipam,
nautobot_plugin_example, sessions, social_django, taggit,
tenancy, users, virtualization
Running migrations:
  No migrations to apply.
Generating cable paths...
..
Finished.

Collecting static files...
0 static files copied to '/opt/nautobot/static', 972
unmodified.
Removing stale content types...
Removing expired sessions...
Invalidating cache...
```

Data Validation

>>> Data Validation Engine App

- The Data Validation Engine app allows users to define rules to **enforce business constraints** on the data formats in Nautobot

- Two types of rules are allowed:
  - Min/max for numerical values
  - Regex for text

- https://www.networktocode.com/apps/data-validation/

>>> Data Validation Engine Rules Are NOT Retroactive

- Validation rules only apply to new instances and edits of existing instances
- Existing attributes for an object that are not allowed by new validation rules are *grandfathered* in
  - Editing any attribute of an object that has another attribute that does not comply will require fixing the non-compliant attribute

- Example: New rule says VLAN **ID** must be 200 or greater
  - I want to edit existing VLAN with VLAN **ID** 100
  - I want to change the **Name** from *mgmt-01* to *mgmt-001*
  - This change will also require making the VLAN *ID* 200 or greater

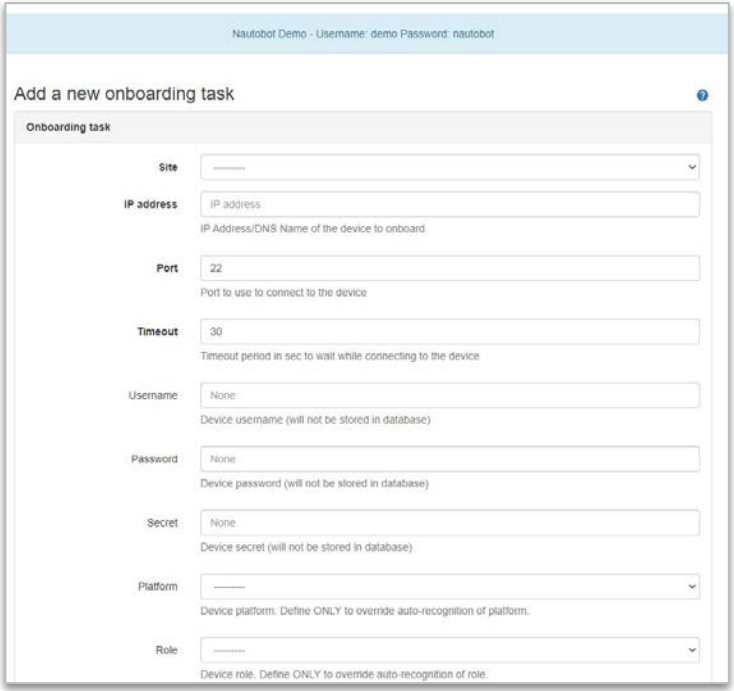
Demo 8

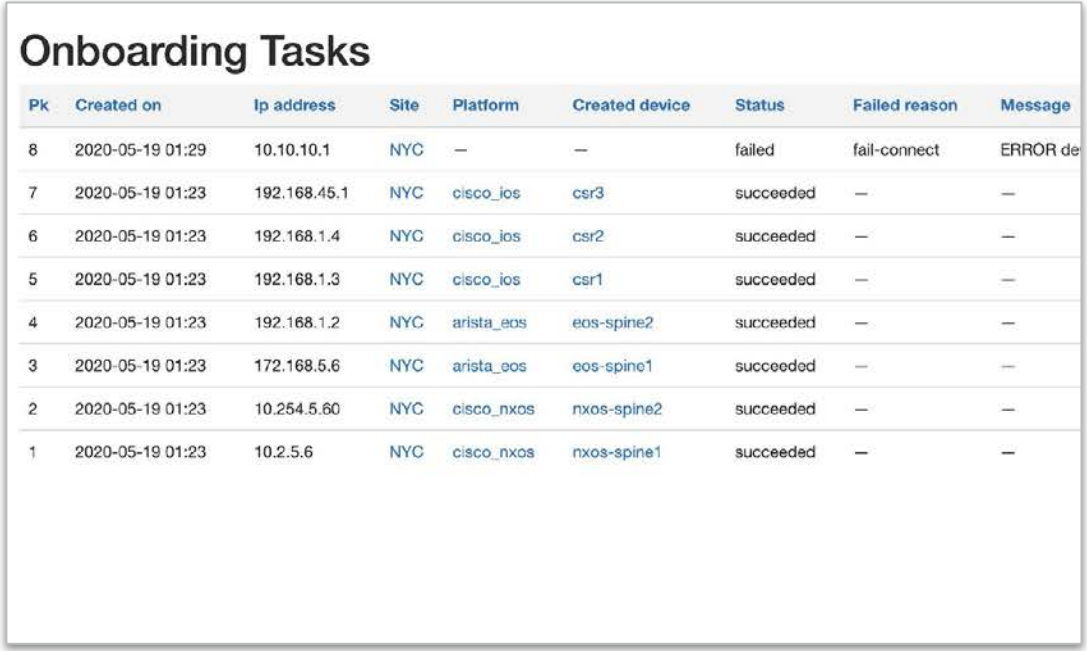
Discovering Apps

Device Onboarding

>>> Device Onboarding App

- Use the device onboarding app to onboard and discover details about devices by providing only an IP Address, credentials, and a site.





Pk	Created on	Ip address	Site	Platform	Created device	Status	Failed reason	Message
8	2020-05-19 01:29	10.10.10.1	NYC	—	—	failed	fail-connect	ERROR de
7	2020-05-19 01:23	192.168.45.1	NYC	cisco_ios	csr3	succeeded	—	—
6	2020-05-19 01:23	192.168.1.4	NYC	cisco_ios	csr2	succeeded	—	—
5	2020-05-19 01:23	192.168.1.3	NYC	cisco_ios	csr1	succeeded	—	—
4	2020-05-19 01:23	192.168.1.2	NYC	arista_eos	eos-spine2	succeeded	—	—
3	2020-05-19 01:23	172.168.5.6	NYC	arista_eos	eos-spine1	succeeded	—	—
2	2020-05-19 01:23	10.254.5.60	NYC	cisco_nxos	nxos-spine2	succeeded	—	—
1	2020-05-19 01:23	10.2.5.6	NYC	cisco_nxos	nxos-spine1	succeeded	—	—

>>> Device Onboarding App

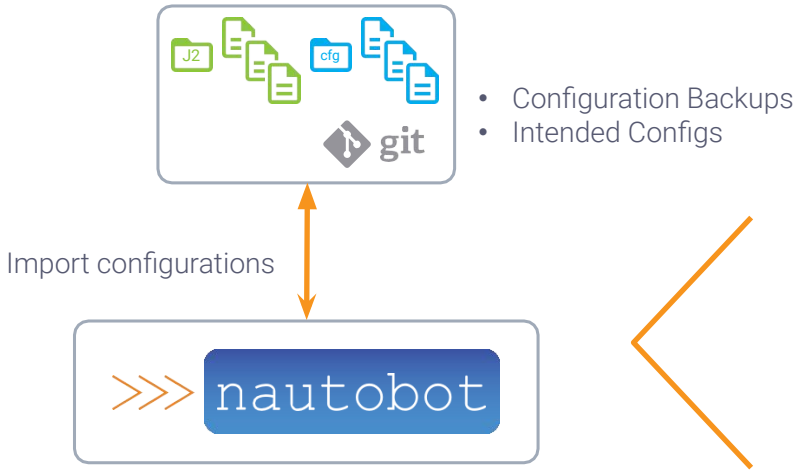
- Onboarding APIs
- **GET** `/api/plugins/device-onboarding/onboarding/`
  - Check status of all onboarding tasks.
- **POST** `/api/plugins/device-onboarding/onboarding/`
  - Onboard a new device
- **GET** `/api/plugins/device-onboarding/onboarding/{id}/`
  - Check the status of a specific onboarding task
- **DELETE** `/api/plugins/device-onboarding/onboarding/{id}/`
  - Delete a specific onboarding task

Golden Configuration

>>> Nautobot Golden Configuration

Using the rich data already in Nautobot, Golden Configuration turns Nautobot into a network automation solution that integrates into NetDevOps workflows.

- Performs network compliance per feature per device
- Natively integrates with Git repositories allowing user to store the backups, and intended configurations all in individual repositories
- Integrates into change pipelines allowing users to fetch configurations via the API or from Git repository



LOG	
Status	Non-Compliant
Intended Configuration	`logging server 10.125.1.171 6 port 7008`
Actual Configuration	`logging server 10.1.1.1 5 port 7008 use-vrf management`
Missing Configuration	`logging server 10.125.1.171 6 port 7008`
Extra Configuration	`logging server 10.1.1.1 5 port 7008 use-vrf management`

NTP	
Status	Compliant
Configuration	`ntp server 10.1.1.1 use-vrf default` `ntp server 10.2.2.2 prefer use-vrf default`

>>> Nautobot Golden Configuration

- Automate
  - Configuration backups
  - Generating intended configurations
  - Performing configuration compliance

Configuration Compliance

Report ▾ + Execute

Device	aaa	acl	bgp	dns	host	intf	log	ntp	snmp	svc
jcy-bb-01.infra.ntc.com	✗	✓	✓	✗	✓	✗	✗	✓	✓	✓
jcy-rtr-01.infra.ntc.com	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗
jcy-spine-01.infra.ntc.com	✗	✓	✓	✗	✓	✗	✓	✓	✓	–
jcy-spine-02.infra.ntc.com	✗	✓	✓	✗	✓	✗	✓	✓	✓	–
nyc-bb-01.infra.ntc.com	✓	–	✓	–	✓	✗	✓	–	–	✓
nyc-leaf-01.infra.ntc.com	✓	✓	✓	✓	✓	✗	✗	✓	✗	✓
nyc-leaf-02.infra.ntc.com	✗	✓	✓	✓	✓	✗	✗	✓	✗	✓
nyc-rtr-01.infra.ntc.com	✓	–	✓	–	✓	✗	✓	–	–	✓

Configuration Compliance

>>> Configuration Compliance Overview

Configuration Compliance

Device	aaa	dns	hostnam
gILUSded01	✗	–	✓
gILUSrlc09	✗	–	✓
gMLUSmdm01-P	✗	–	✓
hILAUolt01	✗	–	✓
hILDEfra01	✗	–	✓
hILDEfra02	✗	–	✓
hILDEfra03	✗	–	✓
hILSGsie01	✗	–	✓
hILSGsie02	✗	–	✓
hILSGsie03	✗	–	✓
hILUSchi01	✗	–	✓
hILUSchi02	✗	–	✓
hILUSchi03	✗	–	✓
hILUSchi06	✗	–	✓
Lab_Quimico_Te			
nALUSope100			
nALUSope101			
nALUSope102			

Overview Reports

Executive Summary

Type	Total	Compliant	Non-Compliant	Compliance (%)	Summary
Devices	1470	3	1467	0.2 %	
Features	6444	2079	4365	32.26 %	

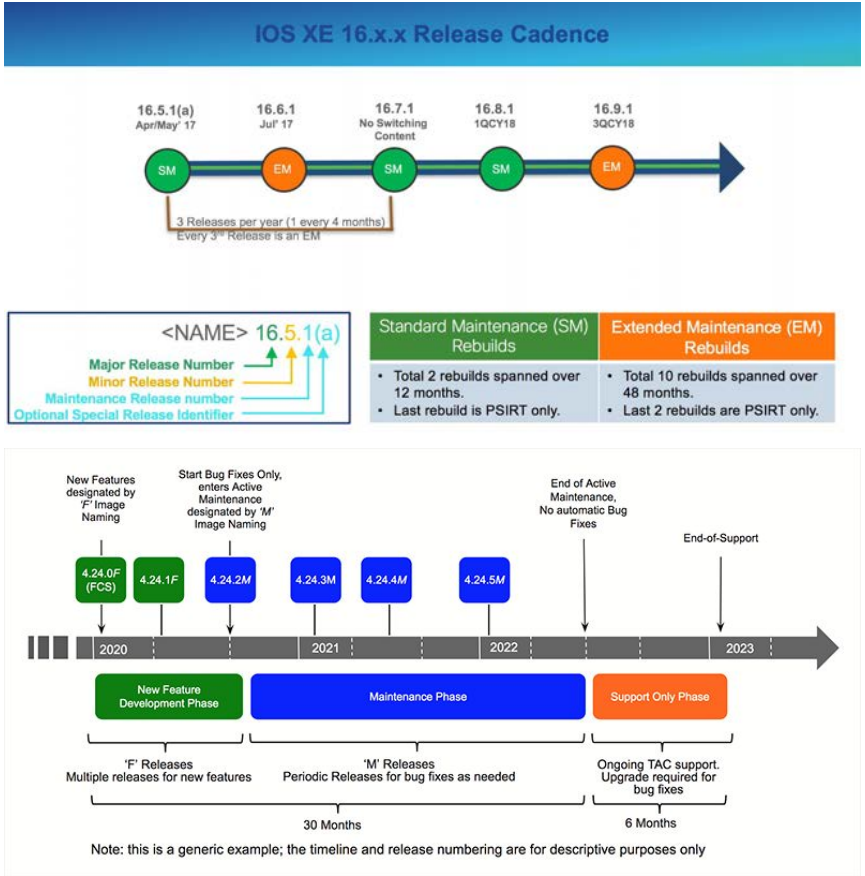
Nautobot_golden_config.jobs

Perform Configuration Compliance	Run configuration compliance on your network infrastructure.	Sept. 19, 2021 11:37 a.m. by zbrochm	Failed	4 0 0 5 Details ›
Execute All Golden Configuration Jobs - Single Device	Process to run all Golden Configuration jobs configured.	Never	N/A	–
Execute All Golden Configuration Jobs - Multiple Device	Process to run all Golden Configuration jobs configured against multiple devices.	Never	N/A	–

Device Lifecycle

>>> What problem are we trying to solve? (Software)

- How many software versions do we have?
- What software version is currently installed on my devices?
- What is the recommended/approved OS version for our campus switches?
- How can we automate software upgrades?
- How can we generate reporting of the OS version compliance?



>>> What problem are we trying to solve? (Hardware)

- Do we have a maintenance contract for our core switches?
- Is the hardware we use for our edge devices still supported?
- How do we manage End Of Life and End Of Support notices on a large estate?
- Can we automate the RMA process for the failed equipment?
- How can we plan for the hardware refresh?

End-of-life milestones

Table 1. End-of-life milestones and dates for the Cisco Catalyst 6506-E, Catalyst 6509-E, Catalyst 6509-V-E, Catalyst 6513-E, VS-S2T-10G, VS-S2T-10G-XL, Bundles & accessories

Milestone	Definition	Date
End-of-Life Announcement Date	The date the document that announces the end-of-sale and end-of-life of a product is distributed to the general public.	October 31, 2019
End-of-Sale Date: HW, Accessory	The last date to order the product through Cisco point-of-sale mechanisms. The product is no longer for sale after this date.	October 30, 2020
Last Ship Date: HW, Accessory	The last-possible ship date that can be requested of Cisco and/or its contract manufacturers. Actual ship date is dependent on lead time.	January 29, 2021
End of SW Maintenance Releases Date: HW	The last date that Cisco Engineering may release any final software maintenance releases or bug fixes. After this date, Cisco Engineering will no longer develop, repair, maintain, or test the product software.	October 30, 2021
End of Vulnerability/Security Support: HW	The last date that Cisco Engineering may release a planned maintenance release or scheduled software remedy for a security vulnerability issue.	October 30, 2023
End of Routine Failure Analysis Date: HW	The last-possible date a routine failure analysis may be performed to determine the cause of hardware product failure or defect.	October 30, 2021
End of New Service Attachment Date: HW	For equipment and software that is not covered by a service-and-support contract, this is the last date to order a new service-and-support contract or add the equipment and/or software to an existing service-and-support contract.	October 30, 2021

>>> What problem are we trying to solve? (Contracts)

- What contracts, hardware and software, do we have in place?
- What hardware is covered by contracts?
- Are any contracts up for renewal?
- Who owns the contract, what are the points of contact?
- What vendors provide each of the contracts?
- How much does it cost?

Contracts

Name	Contract Start Date	Contract End Date	Contract Cost	Support Level	Contract Type	Vendor
Arista EOS	2021-11-09	—	150000.00 GBP	Enterprise	Software	Arista
Catalyst 6500	2014-03-11	2022-06-14	2000000.00 USD	Next Day	Hardware	Cisco
Catalyst Supervisor 720	2017-12-06	2023-03-07	500000.00 USD	—	Hardware	Cisco

Vendors

Name	Physical address	Phone	E-mail	URL
Arista	5453 Great America Parkway, Santa Clara, CA 95054	+1-408-547-5500	tac@arista.com	
Cisco	Cisco Systems, Inc. 170 West Tasman Dr. San Jose, CA 95134 USA	(800) 553-6387 (NETS	tac@cisco.com	

Contract POCs

Name	Address	Phone	Contact E-mail	Comments	Priority	Contract
Huxley Rogala	Southfields, London	(44) 271828-314	rogalah@arista.com	—	100	Arista EOS
Marek Zbroch	Warsaw	256-655356	mzb@cisco.com	—	100	Catalyst 6500
Stan Podolski	Cisco, Diegem, Brussels, Belgium	553-6387-500	stan.p@cisco.com	—	100	Catalyst 6500

App Features

Device Lifecycle Management

Hardware LCM

Manage Hardware Support Notices

Maintain information about end of sale, support and security updates. Associated with device types and inventory items.

- Notice information on the device types and inventory item pages
- Reporting of expired notices

Software LCM

Manage Software Versioning with Nautobot

Components necessary to manage the OS installed on devices and inventory items. Define list of approved software versions.

- Current OS Running
- Valid OS version list
- Reporting of Compliance

Maintenance Contract

Manage related maintenance contracts

Maintain information about maintenance contracts, the associated devices, service levels, and expiration dates.

- Maintenance Contracts
- Device to Maintenance Contract relationship
- Service Level Information

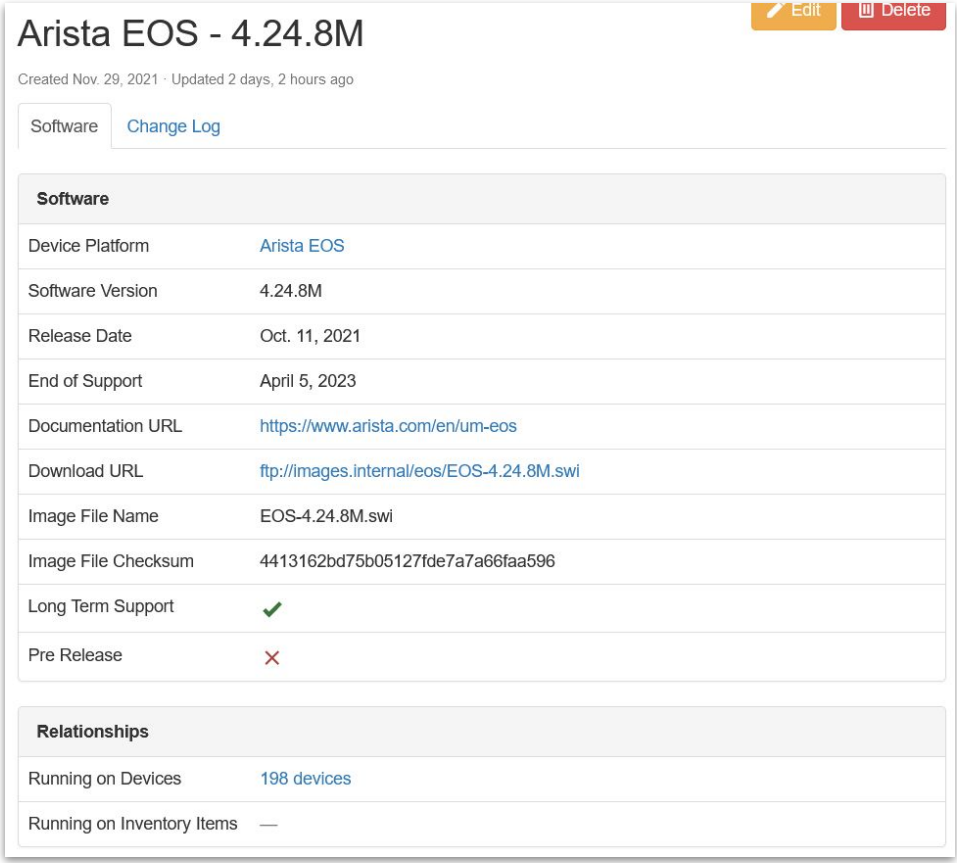
Components

>>> Software Versions

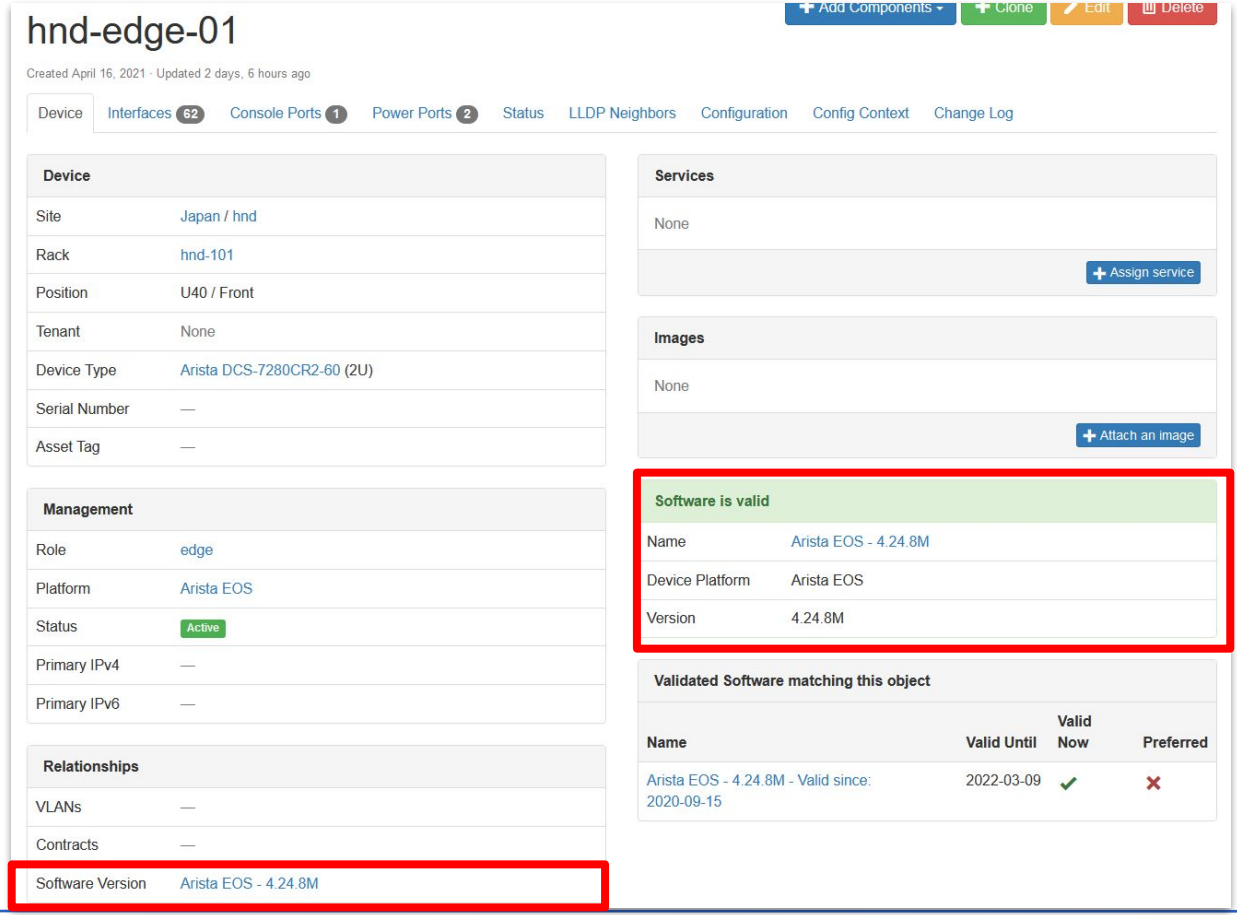
- Software objects record details of the given version
- Track current OS version installed on the Device/Inventory Item
- Initially populated as part of the onboarding process
- Full REST and GraphQL API support

- Recorded software information:
  - software image name and checksum
  - release and end of support dates
  - download and documentation URLs
  - and more

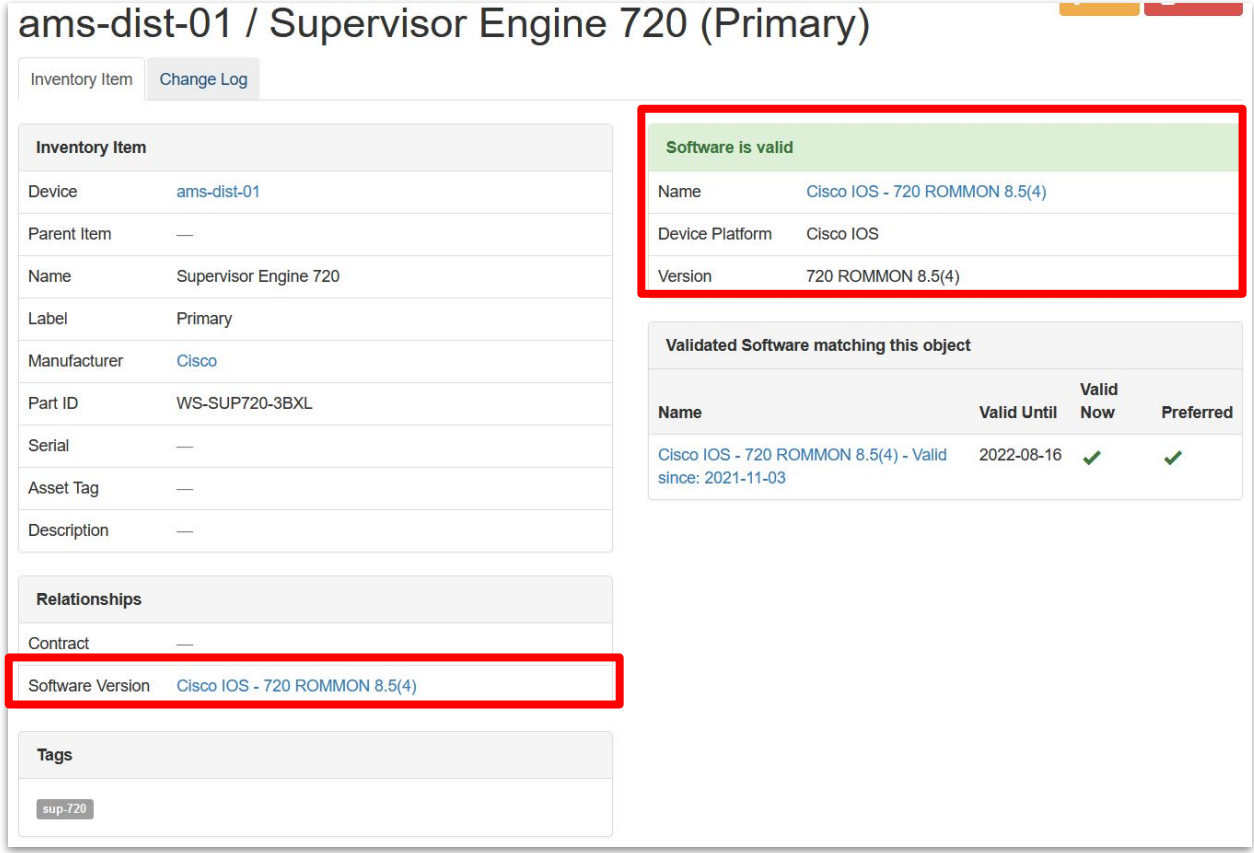
>>> Software Versions - Object Details



>>> Software Versions - Device view



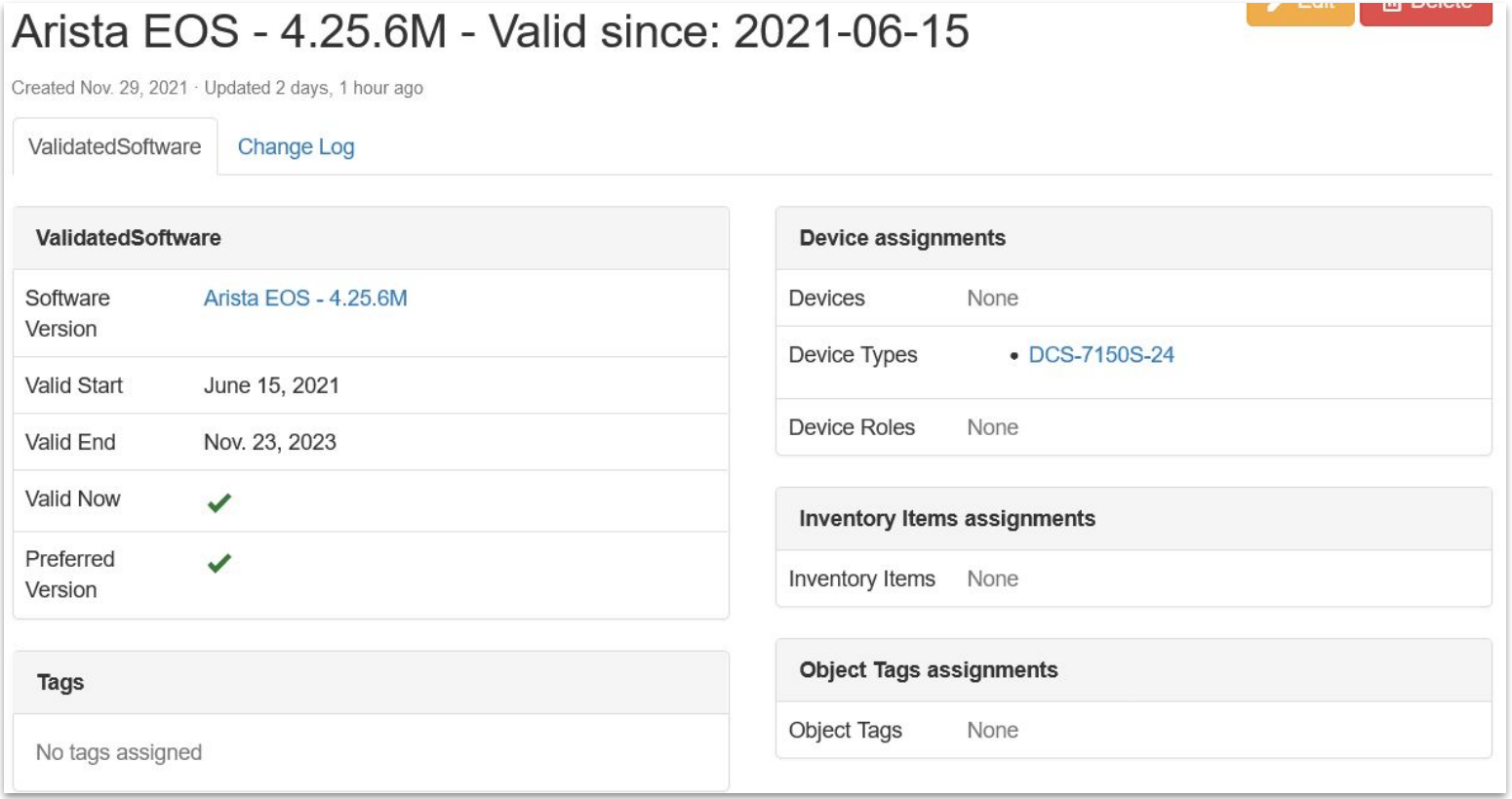
>>> Software Versions - Inventory Item View



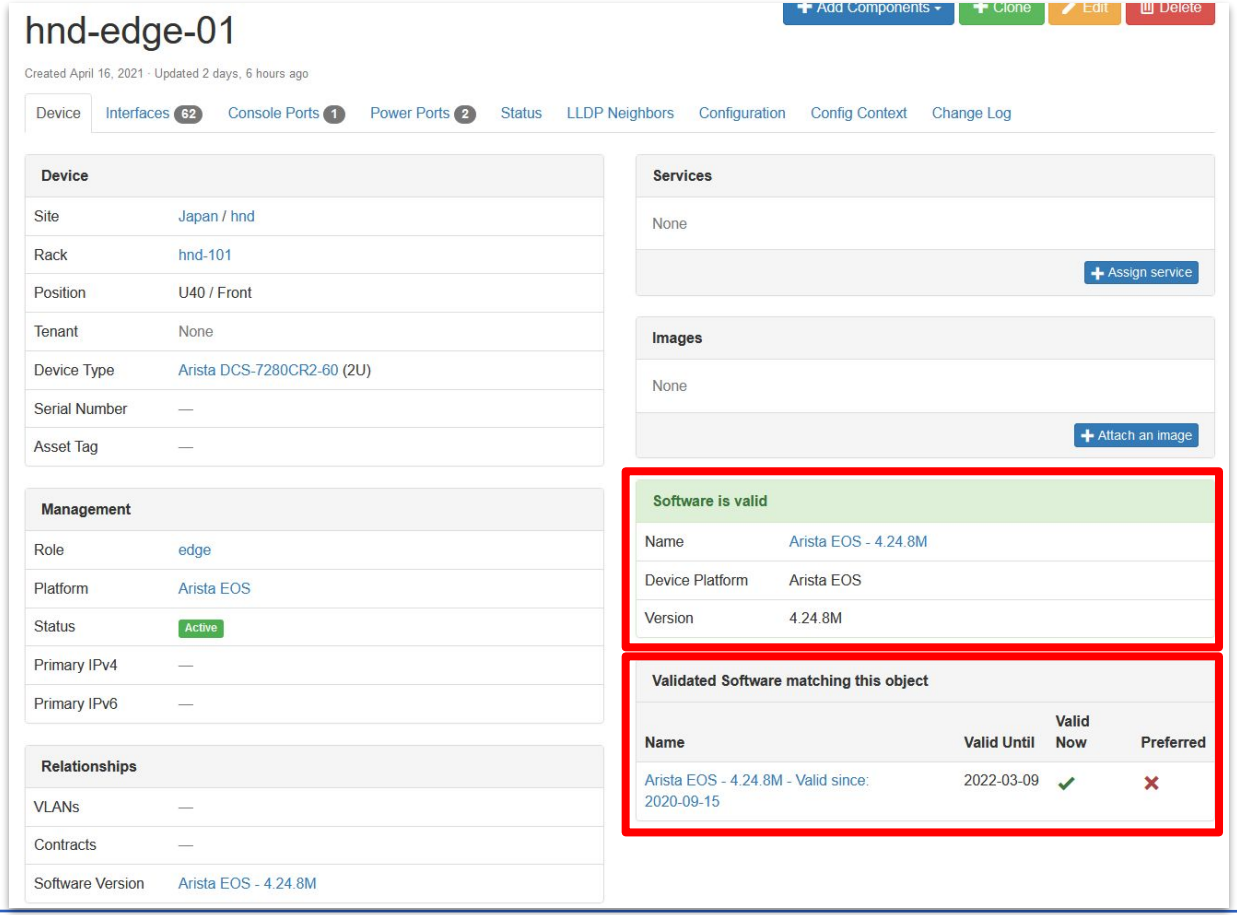
>>> Validated Software

- Validated Software defines which versions of software are valid/approved for given subset of devices and inventory items
- Can be applied to individual devices/inventory items or groups based on:
  - device model
  - device role
  - tags applied to devices/inventory items
- There can be many validated software objects for one software version.
  - Allows treating different appliance groups differently
- Used for compliance reporting.
- Can be used during upgrades to verify OS version is approved for given device.

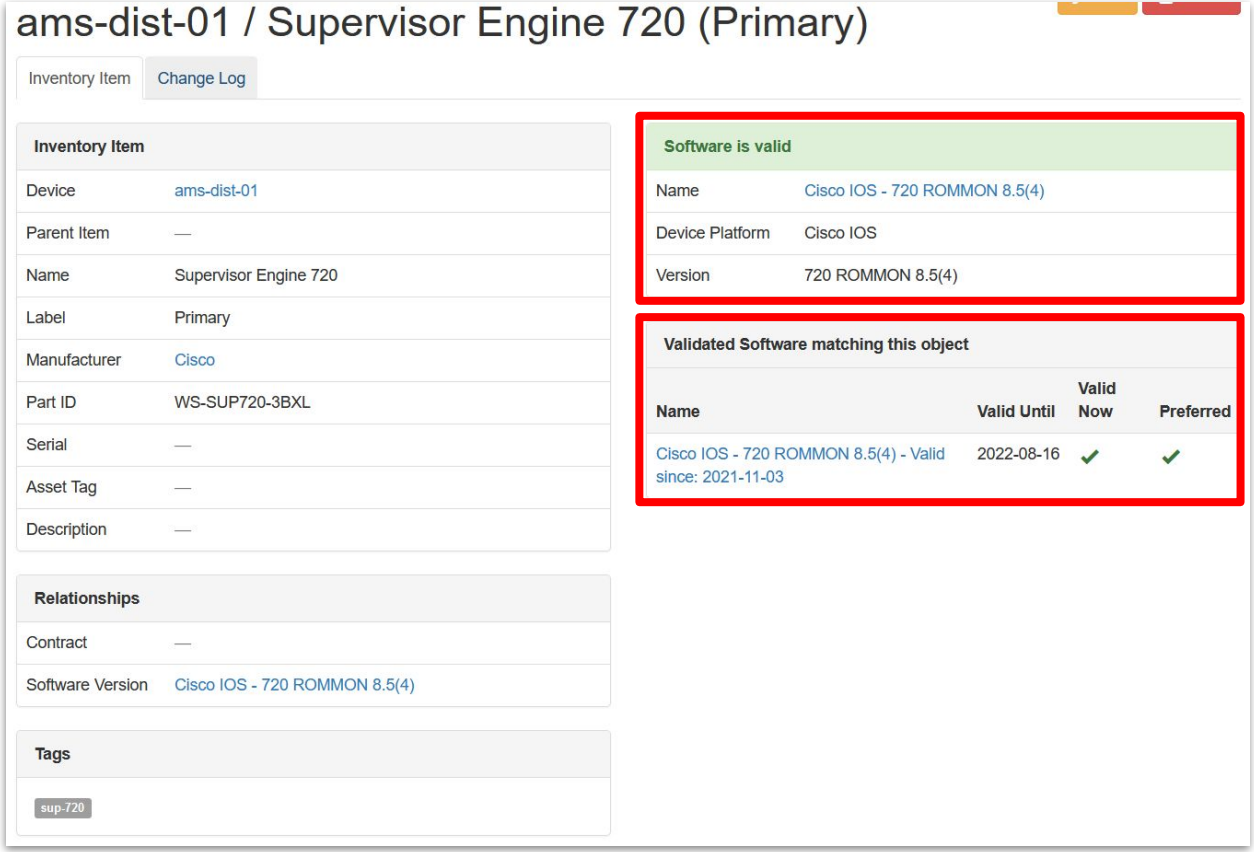
>>> Validated Software - Detail View



>>> Validated Software - Device view



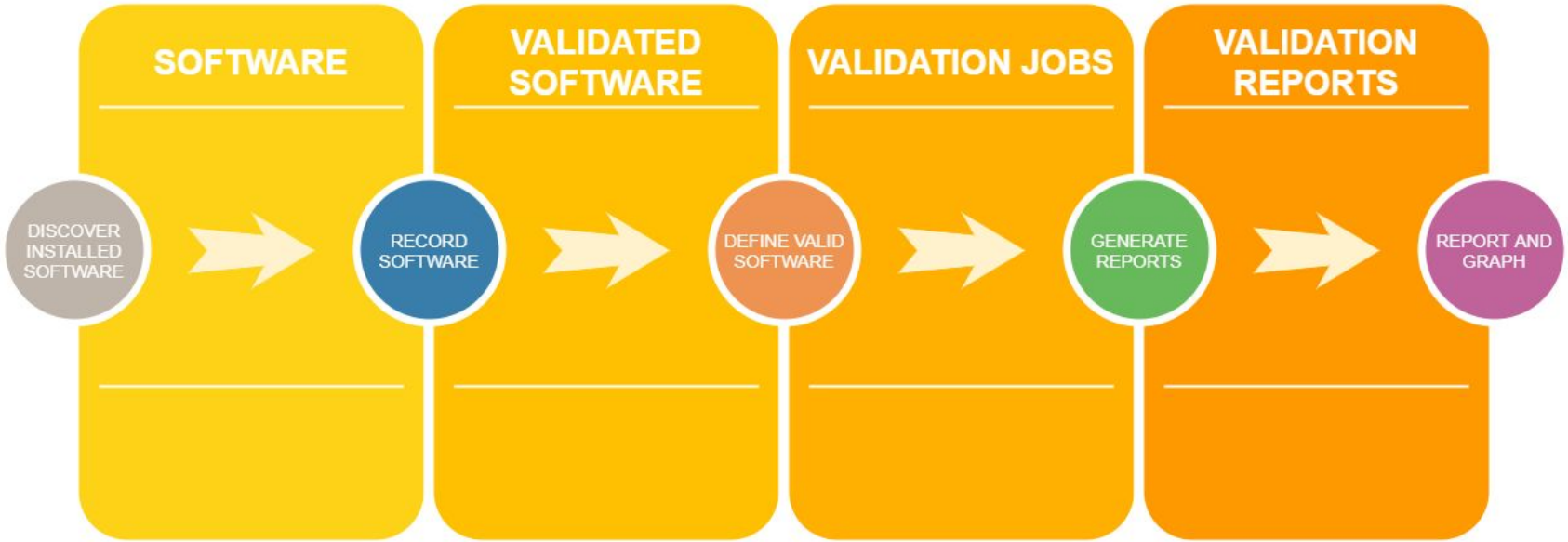
>>> Validated Software - Inventory item view



>>> Software Validation Reports

- Reporting jobs perform validity check of software installed on devices and inventory items
- Validated Software objects are used to match individual hardware items
- Results are presented in a graphical and tabular formats
- Rich set of filters allows analyzing results with graphs and tables adjusting dynamically
- Graphs can be saved and tables exported in CSV format

>>> Software Validation Workflow

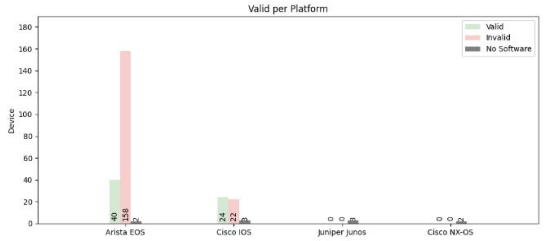


>>> Software Validation Report - Devices

Device Software Validation Reports

+Execute | Configure | Export

Last full run of the report: Dec. 2, 2021 10:23 a.m. - 1 day, 10 hours ago



Executive Summary

Type	Total	Valid	Invalid	No Software	Compliance (%)	Summary
Devices	254	64	180	10	25.2 %	

Device Type Summary

Device Type	Total	Valid	Invalid	No Software	Compliance (%)
DCS-7280CR2-60	40	40	0	0	100.0 %
Catalyst 6509-E	46	24	22	0	52.0 %
CSR1000V	3	0	0	3	0.0 %
DCS-7150S-24	156	0	154	2	0.0 %
Nexus 9Kv	2	0	0	2	0.0 %
vEOS	4	0	4	0	0.0 %
vMX	3	0	0	3	0.0 %

Search

Search

Software

Site

Region

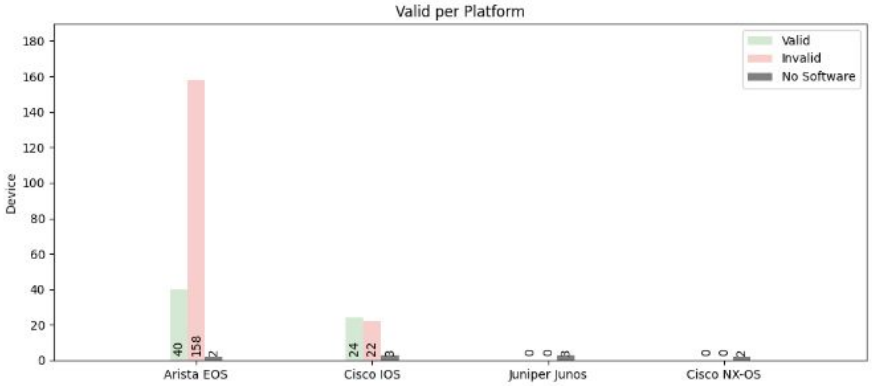
Device

Device type

Device role

Exclude missing software

Apply | Clear



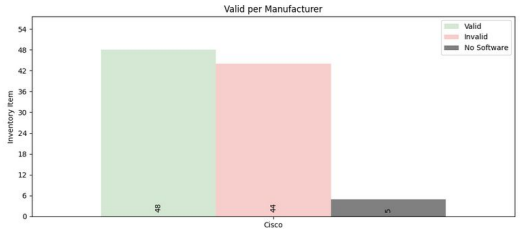
Device Type Summary

Device Type	Total	Valid	Invalid	No Software	Compliance (%)
DCS-7280CR2-60	40	40	0	0	100.0 %
Catalyst 6509-E	46	24	22	0	52.0 %
CSR1000V	3	0	0	3	0.0 %
DCS-7150S-24	156	0	154	2	0.0 %
Nexus 9Kv	2	0	0	2	0.0 %
vEOS	4	0	4	0	0.0 %
vMX	3	0	0	3	0.0 %

>>> Software Validation Report - Inventory Items

Inventory Item Software Validation Reports

+ Execute | Configure | Export



Search

Search

Inventory items

Devices

Device types

Device roles

Software

Exclude missing software

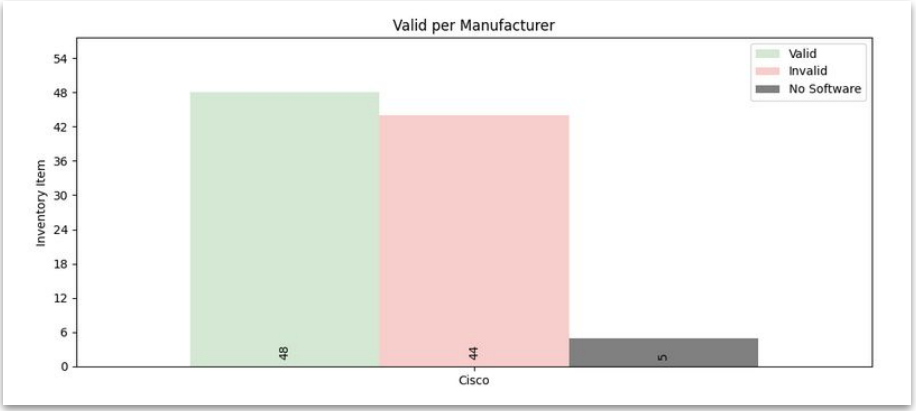
Apply | Clear

Executive Summary

Type	Total	Valid	Invalid	No Software	Validation (%)	Summary
Inventory Items	97	48	44	5	49.48 %	

Inventory Item Part ID Summary

Part ID	Total	Valid	Invalid	No Software	Compliance (%)
WS-SUP720-3BXL	92	48	44	0	52.0 %
WS-X6708-10G-3CXL	1	0	0	1	0.0 %
WS-X6748-GE-TX	2	0	0	2	0.0 %
X2-10GB-LR6	2	0	0	2	0.0 %



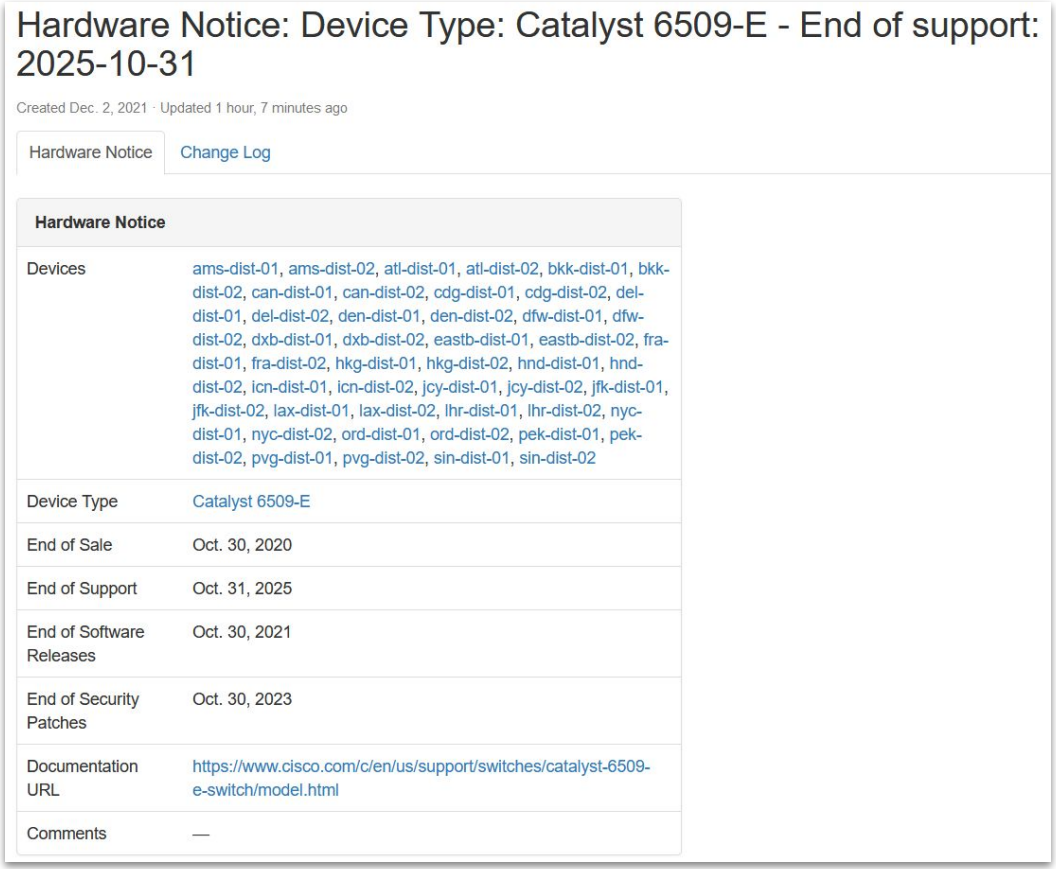
Inventory Item Part ID Summary

Part ID	Total	Valid	Invalid	No Software	Compliance (%)
WS-SUP720-3BXL	92	48	44	0	52.0 %
WS-X6708-10G-3CXL	1	0	0	1	0.0 %
WS-X6748-GE-TX	2	0	0	2	0.0 %
X2-10GB-LR6	2	0	0	2	0.0 %

>>> Hardware Notices

- Two associations
  - Associated to a Nautobot Device Type
  - Associated to an Inventory Item on a device
- End of Support Dates
- End of Security Updates
- End of Sale
- GraphQL interface
- Viewable at the Device/Inventory level
- Future integration with vendor API

>>> Hardware Notices - Detail View



Hardware Notice: Device Type: Catalyst 6509-E - End of support: 2025-10-31

Created Dec. 2, 2021 · Updated 1 hour, 7 minutes ago

Hardware Notice | Change Log

Hardware Notice	
Devices	ams-dist-01, ams-dist-02, atl-dist-01, atl-dist-02, bkk-dist-01, bkk-dist-02, can-dist-01, can-dist-02, cdg-dist-01, cdg-dist-02, del-dist-01, del-dist-02, den-dist-01, den-dist-02, dfw-dist-01, dfw-dist-02, dxb-dist-01, dxb-dist-02, eastb-dist-01, eastb-dist-02, fra-dist-01, fra-dist-02, hkg-dist-01, hkg-dist-02, hnd-dist-01, hnd-dist-02, icn-dist-01, icn-dist-02, jcy-dist-01, jcy-dist-02, jfk-dist-01, jfk-dist-02, lax-dist-01, lax-dist-02, lhr-dist-01, lhr-dist-02, nyc-dist-01, nyc-dist-02, ord-dist-01, ord-dist-02, pek-dist-01, pek-dist-02, pvg-dist-01, pvg-dist-02, sin-dist-01, sin-dist-02
Device Type	Catalyst 6509-E
End of Sale	Oct. 30, 2020
End of Support	Oct. 31, 2025
End of Software Releases	Oct. 30, 2021
End of Security Patches	Oct. 30, 2023
Documentation URL	https://www.cisco.com/c/en/us/support/switches/catalyst-6509-e-switch/model.html
Comments	—

>>> Hardware Notices - Device type view

Device Type | Change Log

Chassis

Manufacturer	Cisco
Model Name	Catalyst 6509-E catalyst-6509-e
Part Number	WS-C6509-E
Height (U)	1
Full Depth	✔
Parent/Child	—
Front Image	—
Rear Image	—
Instances	46

Tags

No tags assigned

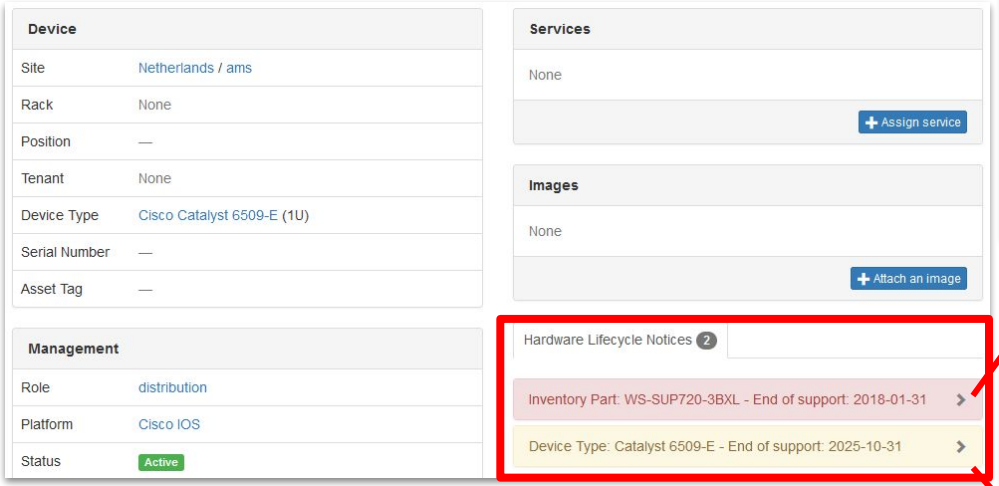
Comments

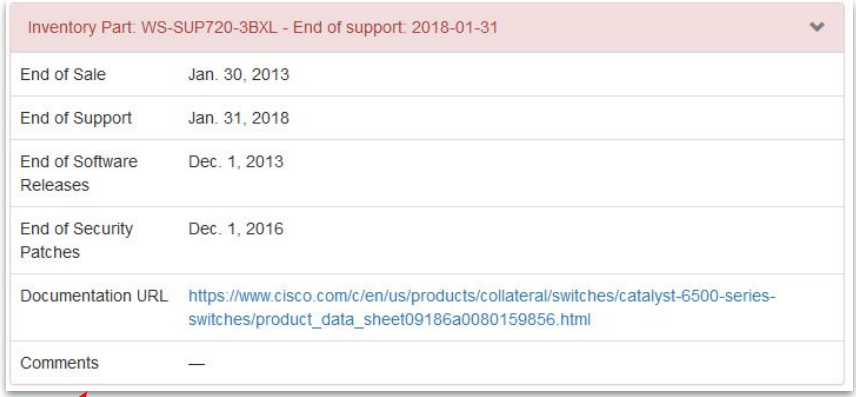
None

Hardware Notice

Name	Device Type: Catalyst 6509-E - End of support: 2025-10-31
End of Sale	Oct. 30, 2020
End of Support	Oct. 31, 2025
End of Software Releases	Oct. 30, 2021
End of Security Patches	Oct. 30, 2023
Documentation URL	https://www.cisco.com/c/en/us/support/switches/catalyst-6509-e-switch/model.html
Comments	—

>>> Hardware Notices - Device view



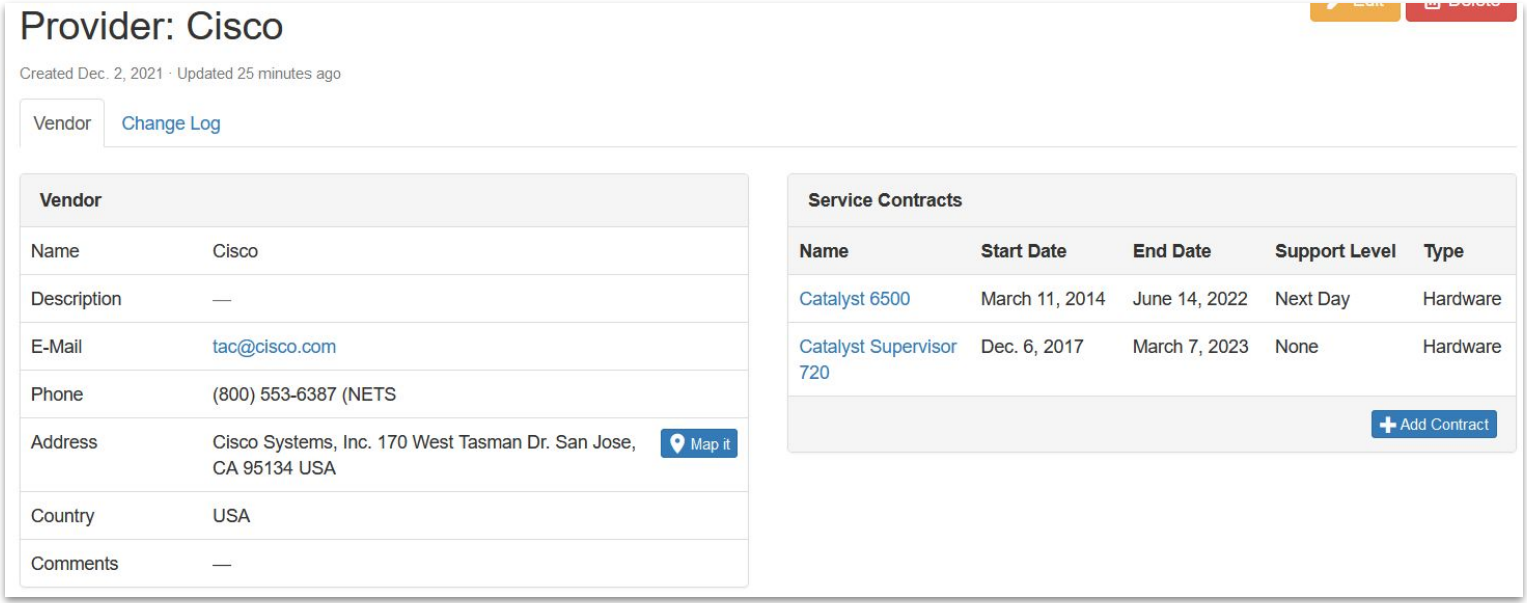


>>> Service Contract Management

- Two associations
  - Associated to a Nautobot Device Type
  - Associated to an Inventory Item on a device
- Track contract cost/support level
- Track start/end dates
- Provide escalation/owner contacts
- View relationships to devices/inventories

Contract	
Name	**Catalyst 6500**
Provider	Cisco
Start Date	March 11, 2014
End Date	June 14, 2022
Cost	2,000,000.00
Currency	USD
Support Level	Next Day
Contract Type	Hardware
Comments	—

>>> Contract Vendor Management



- Vendor contact details
- Shows linked contracts

>>> Contract Contacts Management

Contact: Huxley Rogala

Created Nov. 24, 2021 · Updated 48 minutes ago

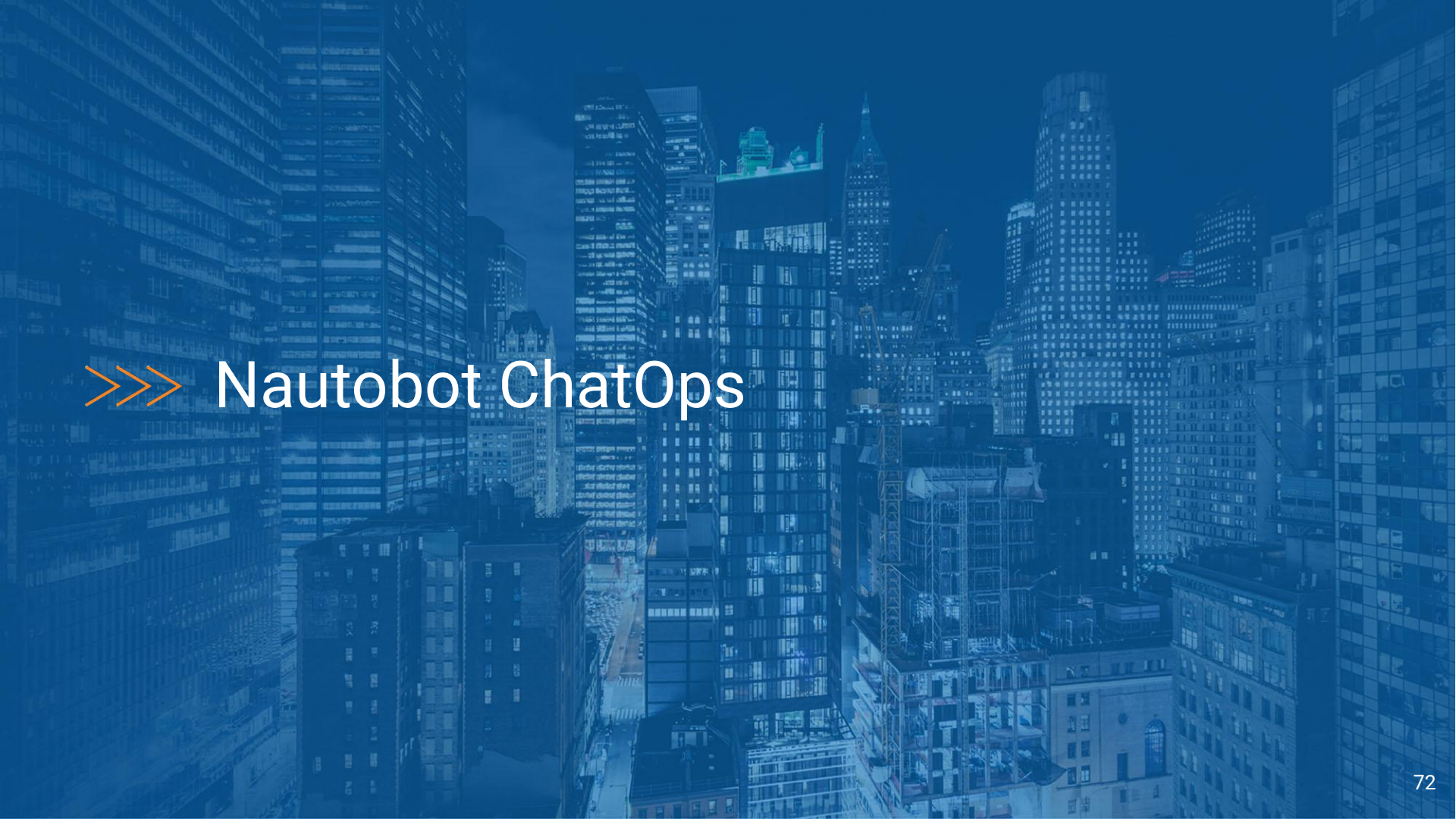
Contact | Change Log

Contact	
Name	Huxley Rogala
Address	Southfields, London
Phone	(44) 271828-314
E-Mail	rogalah@arista.com
PoC Type	Unassigned
Assign to Contract	Arista EOS
Priority	100
Comments	—

Contract POCs

Name	Address	Phone	Contact E-mail	Comments	Priority	Contract
Huxley Rogala	Southfields, London	(44) 271828-314	rogalah@arista.com	—	100	Arista EOS
Marek Zbroch	Warsaw	256-655356	mzb@cisco.com	—	100	Catalyst 6500
Stan Podolski	Cisco, Diegem, Brussels, Belgium	553-6387-500	stan.p@cisco.com	—	100	Catalyst 6500

- Contract point of contact details
- Shows contract assigned to

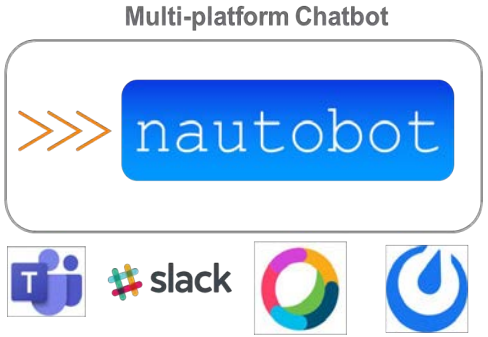


Nautobot ChatOps

>>> Nautobot ChatOps

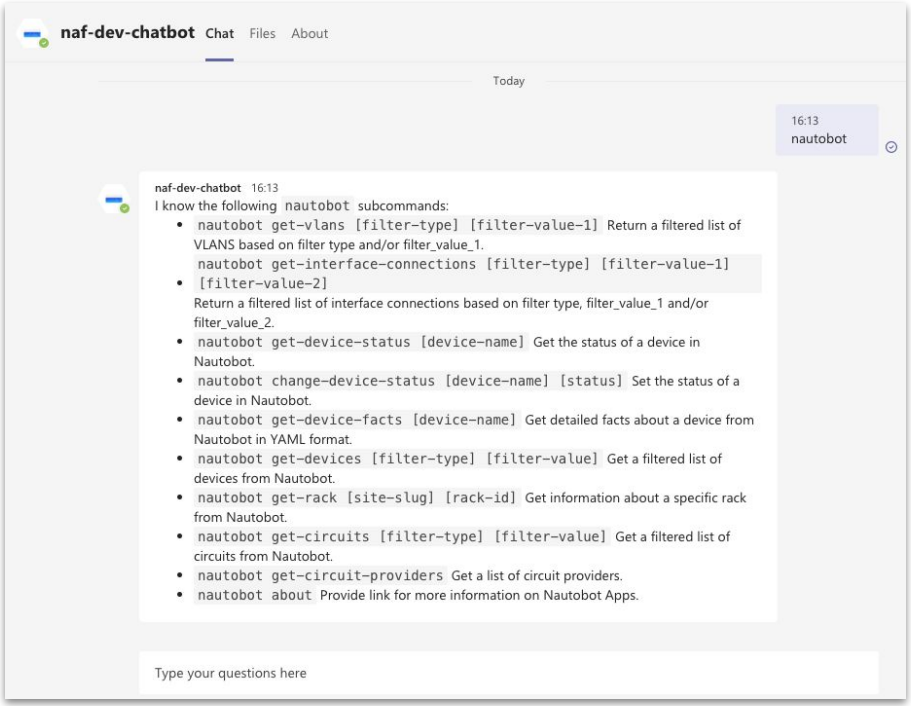
Runs as a chatbot inside Nautobot enabling teams to ask Nautobot questions and automate other IT systems

- Multi-platform chatbot that supports **Slack**, **MS Teams**, **Webex Teams**, and **Mattermost**
- Comes pre-built with commands to easily fetch data from Nautobot
- Enable other teams to easily verify interface connections, inventory, racks, circuits, or any other piece of Nautobot of data from chat
- Easily add new chat commands
- Reduces the amount of change tickets opened providing self-service to read-only data



Nautobot ChatOps Commands

Nautobot ChatOps includes many useful commands out of the box.



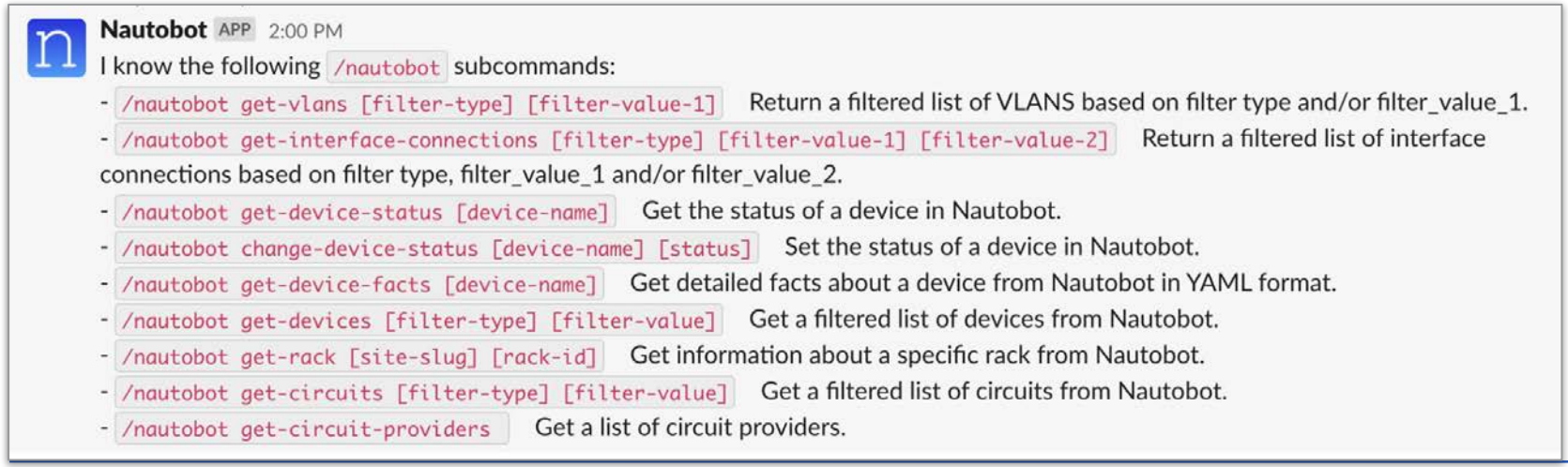
Extensible

Easily add additional *top-level* or *sub-level* commands to extend the functionality of ChatOps beyond the included options.

>>> `/nautobot` - Nautobot ChatOps App

- The *Nautobot* chatbot
  - Interacts with your Nautobot deployment
  - Is currently supported on the MS Teams, Slack, Mattermost, and Webex Teams chat applications
  - Comes with many sub-commands out of the box
  - Supports adding additional sub-commands

Users chat with the bot via the `/nautobot` command



>>> The Nautobot ChatOps Framework

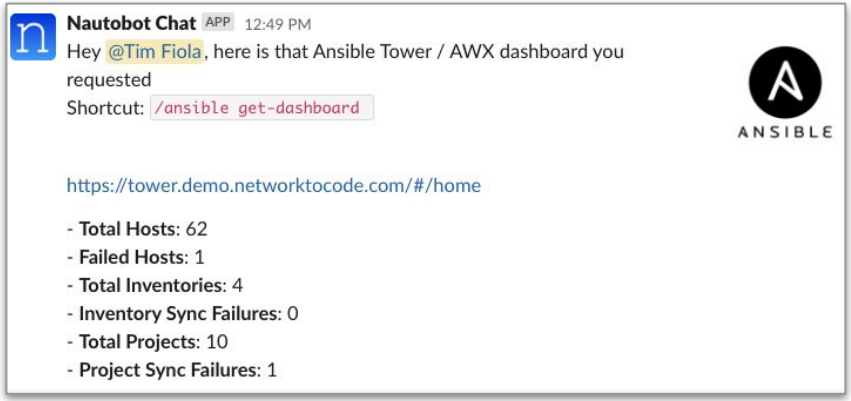
- The Nautobot ChatOps app does more than just let you interact with your Nautobot instance via chat:

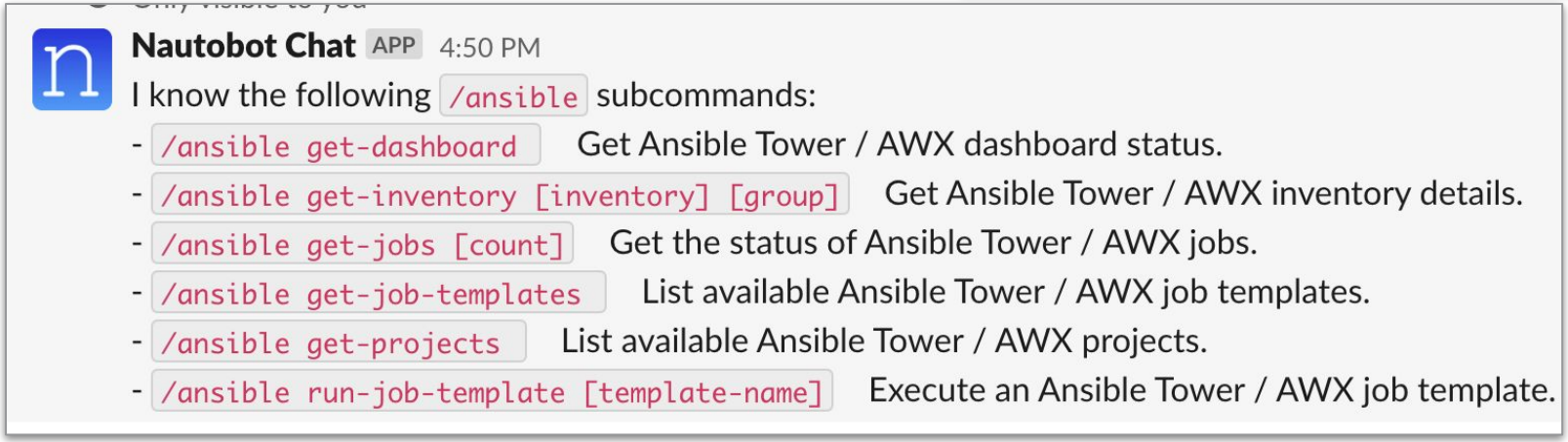
- The Nautobot ChatOps app also provides a **framework** that allows devs to easily craft additional chatbots for whatever API-enabled technology they wish . . .

>>> /ansible - Ansible ChatOps App

- The Ansible chatbot
  - Allows users to communicate directly with Ansible AWX/Tower from various chat platforms
  - Runs on the framework provided by the **Nautobot ChatOps App**
  - Comes with six sub-commands out of the box

Users chat with the bot via the `/ansible` command

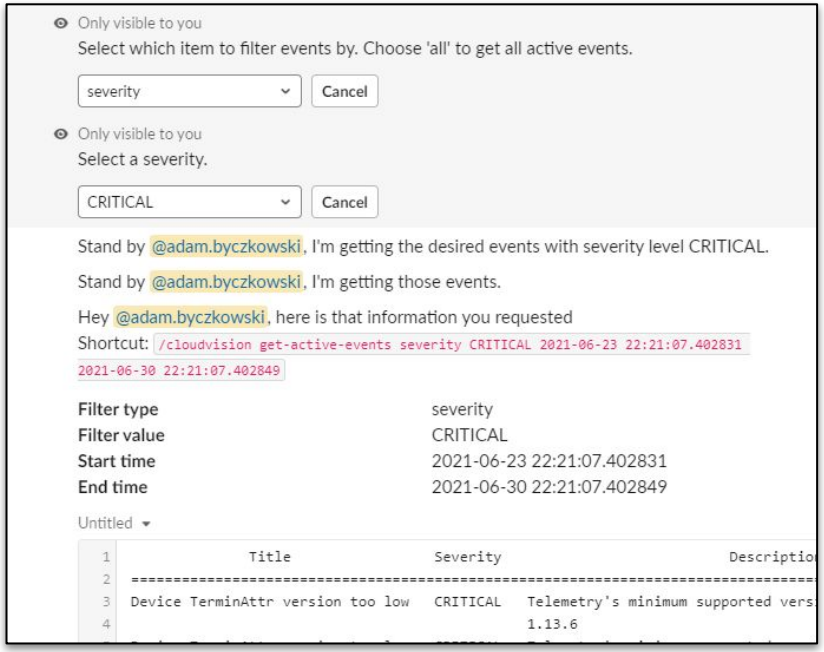




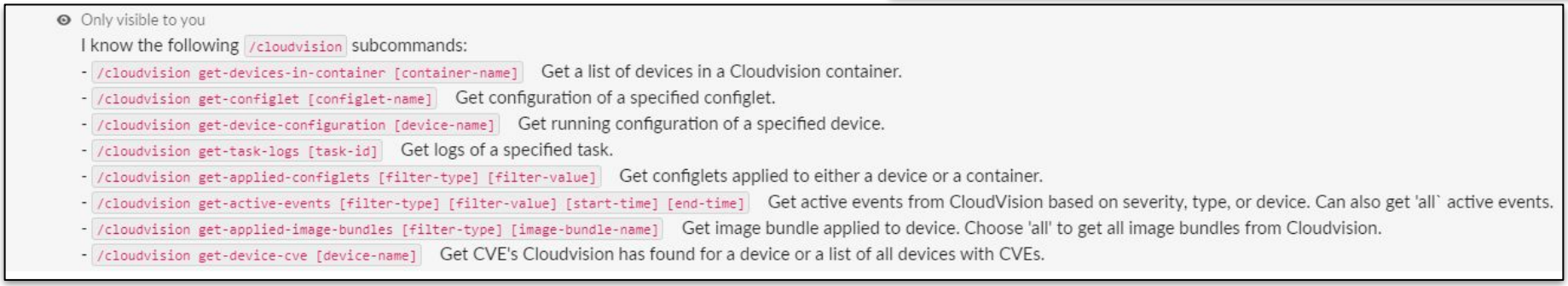
>>> /cloudvision - CloudVision ChatOps

The Arista CloudVision chatbot

- Allows users to gather various information directly from Arista CloudVision
- Runs on the framework provided by the **Nautobot ChatOps App**
- Comes with eight sub-commands out of the box



Users chat with the bot via the `/cloudvision` command

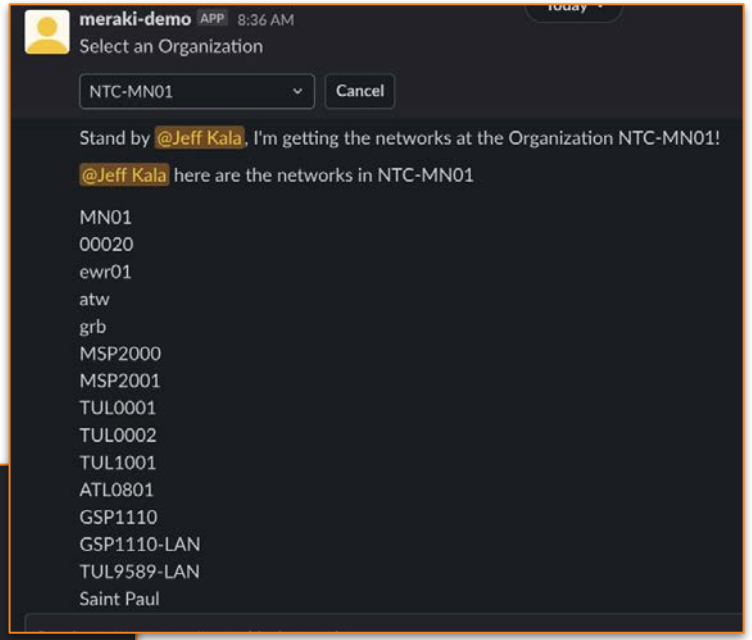


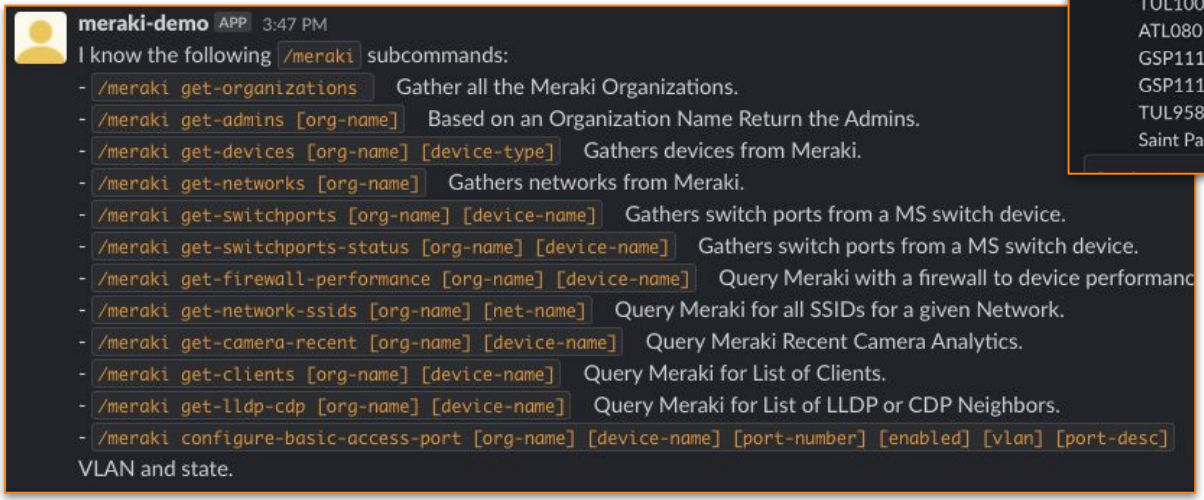
>>> /meraki - Meraki ChatOps App

The Meraki chatbot

- Allows users to chat with the Meraki controller
- Runs on the framework provided by the **Nautobot ChatOps App**
- Comes with 12 sub-command out of the box

Users chat with the bot via the `/meraki` command

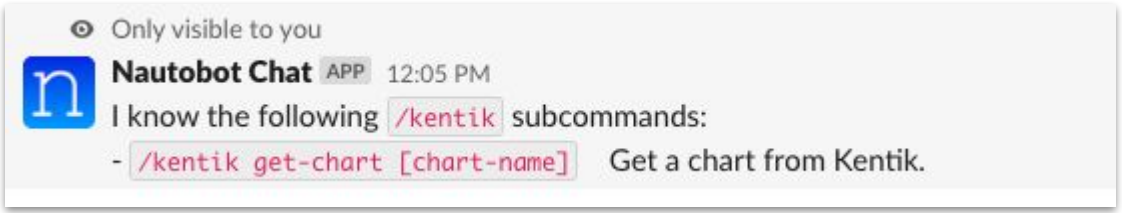


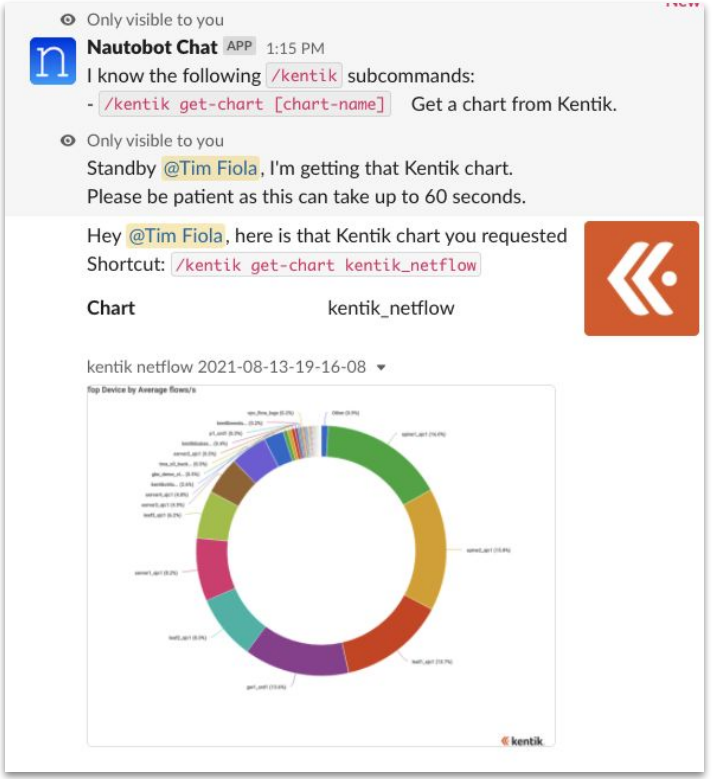


>>> /kentik

- The Kentik chatbot
  - Allows users to chat with Kentik
  - Runs on the framework provided by the **Nautobot ChatOps App**
  - Comes with the `get-chart` sub-command out of the box

Users chat with the bot via the `/kentik` command





Capacity Metrics

Capacity Metrics

- Exposes key data in Nautobot as Prometheus endpoints to be later consumed and visualized in tools like Grafana
- Exposes key data stored in Nautobot as a Prometheus HTTP endpoint so the data can be easily scraped, collected, stored, and visualized
- Exposing this data and visualizing it in a tool like Grafana will allow users to **track growth and trends** in interface capacity, site and rack utilization and capacity, and IP consumption, just to name a few of the common use cases

Capacity Planning Metrics

- Expose Nautobot data as Prometheus metrics
  - Any data that exists in Nautobot
- Used for capacity planning and gaining insights into data consumption
- Track IP Address utilization, rack capacity, device count, power, VLAN count, etc.
- Any app can also expose data using capacity metrics
  - Example: Chatbot exposes number of times a chat command is executed

`/api/plugins/metrics-ext/app-metrics`

Subset of metrics available

```
nautobot_model_count{app="dcim",name="Site"} 15.0
nautobot_model_count{app="dcim",name="Rack"} 18.0
nautobot_model_count{app="dcim",name="Device"} 66.0
nautobot_model_count{app="dcim",name="Interface"} 1004.0
nautobot_model_count{app="dcim",name="Cable"} 50.0
nautobot_model_count{app="ipam",name="IPAddress"} 90.0
nautobot_model_count{app="ipam",name="Prefix"} 45.0
# HELP nautobot_prefix_utilization percentage of utilization per container prefix
# TYPE nautobot_prefix_utilization gauge
nautobot_prefix_utilization{prefix="172.16.0.0/16",role="dcl",site="none"} 6.0
nautobot_prefix_utilization{prefix="172.16.0.0/20",role="dcl",site="infrastructure"} 56.0
nautobot_prefix_utilization{prefix="172.16.0.0/24",role="dcl",site="loopbacks"} 0.0
# HELP nautobot_command_library_total Count of command available per subcommand
# TYPE nautobot_command_library_total counter
nautobot_command_library_total{command="ansible",subcommand="get-jobs"} 1.0
nautobot_command_library_total{command="ansible",subcommand="get-projects"} 1.0
nautobot_command_library_total{command="ansible",subcommand="run-job-template"} 1.0
nautobot_command_library_total{command="clear"} 1.0
nautobot_command_library_total{command="nautobot",subcommand="change-device-status"} 1.0
nautobot_command_library_total{command="nautobot",subcommand="get-circuit-providers"} 1.0
nautobot_command_library_total{command="nautobot",subcommand="get-circuits"} 1.0
nautobot_command_library_total{command="nautobot",subcommand="get-device-facts"} 1.0
# HELP nautobot_circuit_bandwidth bandwidth per circuit
# TYPE nautobot_circuit_bandwidth gauge
nautobot_circuit_bandwidth{cid="GIZ00001-1",provider="att",site="hq",status="active"} 100000.0
nautobot_circuit_bandwidth{cid="GIZ00002-1",provider="att",site="hq",status="active"} 100000.0
# HELP nautobot_app_metrics_processing_ms Time in ms to generate the app metrics endpoint
```

add even more metrics

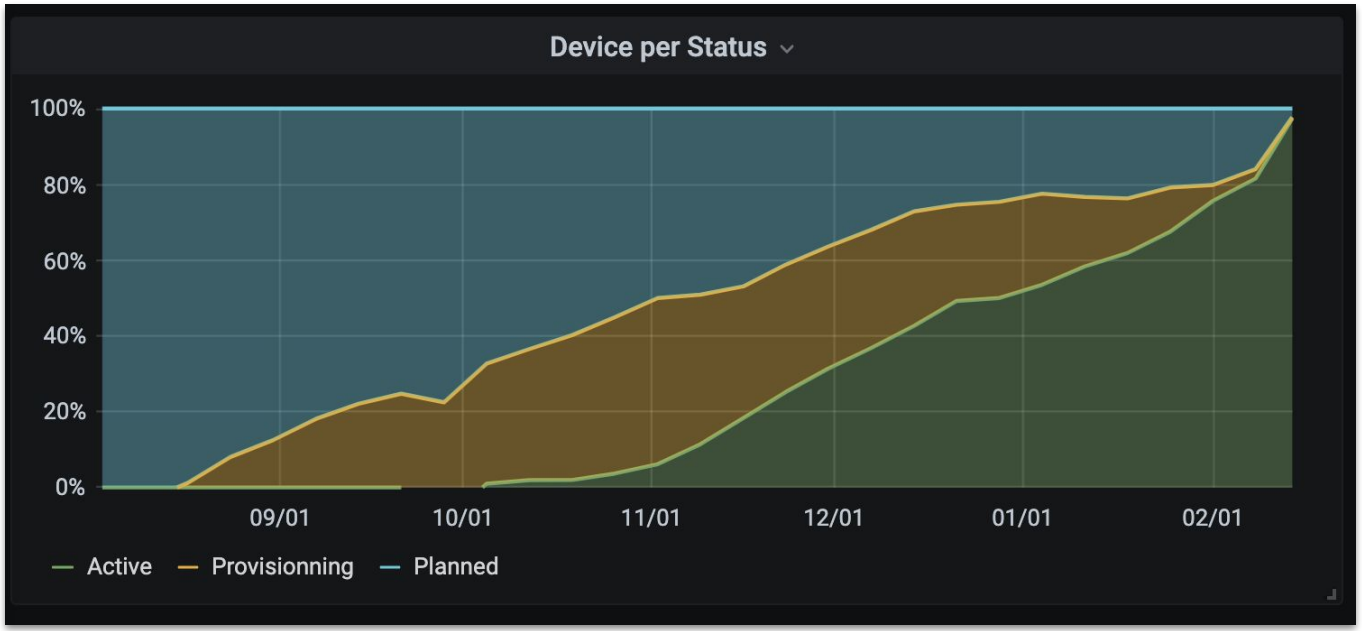
Nautobot data & metrics easily integrates into existing monitoring & telemetry pipelines

>>> Space Available per Rack

- Export rackspace availability
- Leverage data over time for capacity planning

>>> Device Count per Status of Device

- Track the deployment and the provisioning of all devices during a new site deployment
- Identify missing devices, cables …



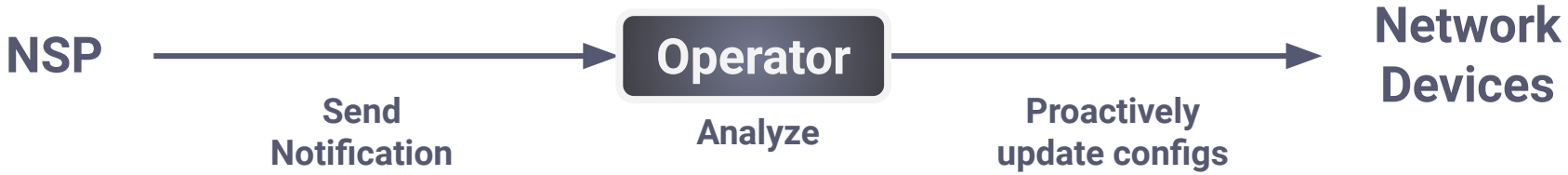
Circuit Maintenance

>>> Why do we need the Circuit Maintenance Plugin?

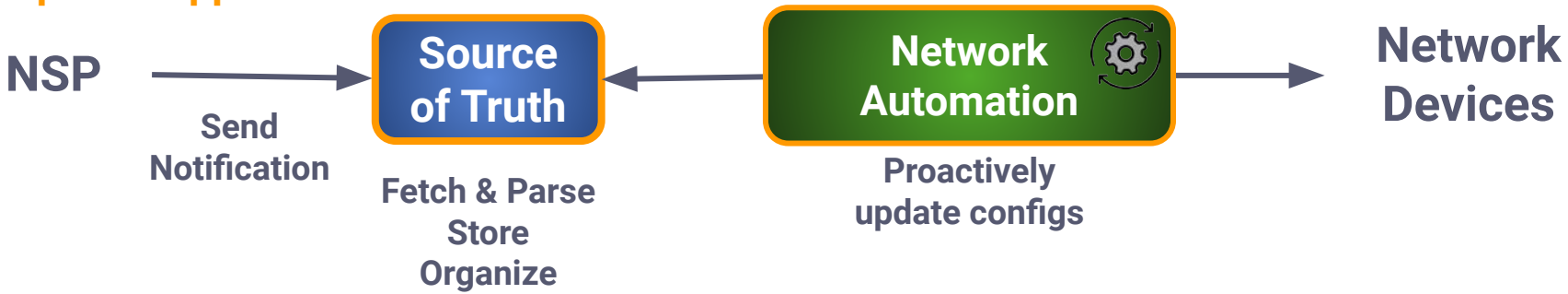
- Every network is built on top of myriad of circuits
- Every circuit will have periodic maintenance during its lifecycle
- Improperly handled circuit maintenance could impact your operations and, eventually, your business
- Handling circuit maintenance manually is time consuming,prone to errors, and doesn't scale well

>>> Changing the paradigm

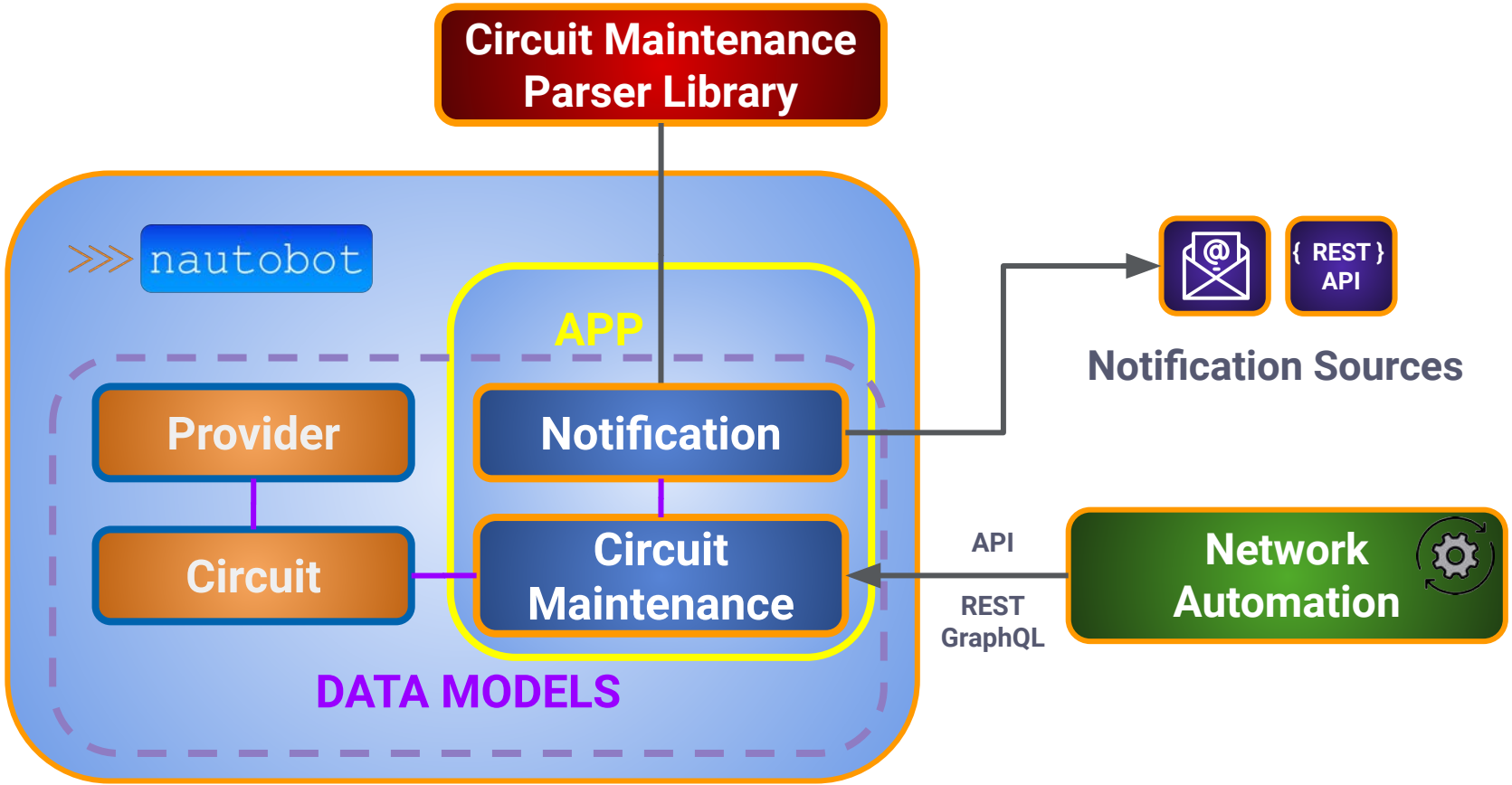
Current Situation



Proposed Approach



App Architecture



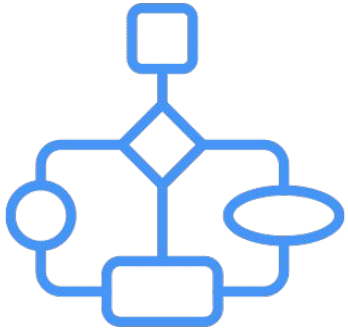
Version Control

>>> Nautobot's Version Control App

- Adds git-like version control to Nautobot's database to allow for GitHub-like workflows

- The Version Control app provides three main business/operational benefits
  - Adds safeguards to keep data clean by adding **explicit checkpoints** for human review
  - Allowing **partial/wholesale rollbacks** of changes to data if something breaks once it's in production
  - A **history of all changes** to production data

- **Requires use of a Dolt database**



Lab 6

Discovering Apps

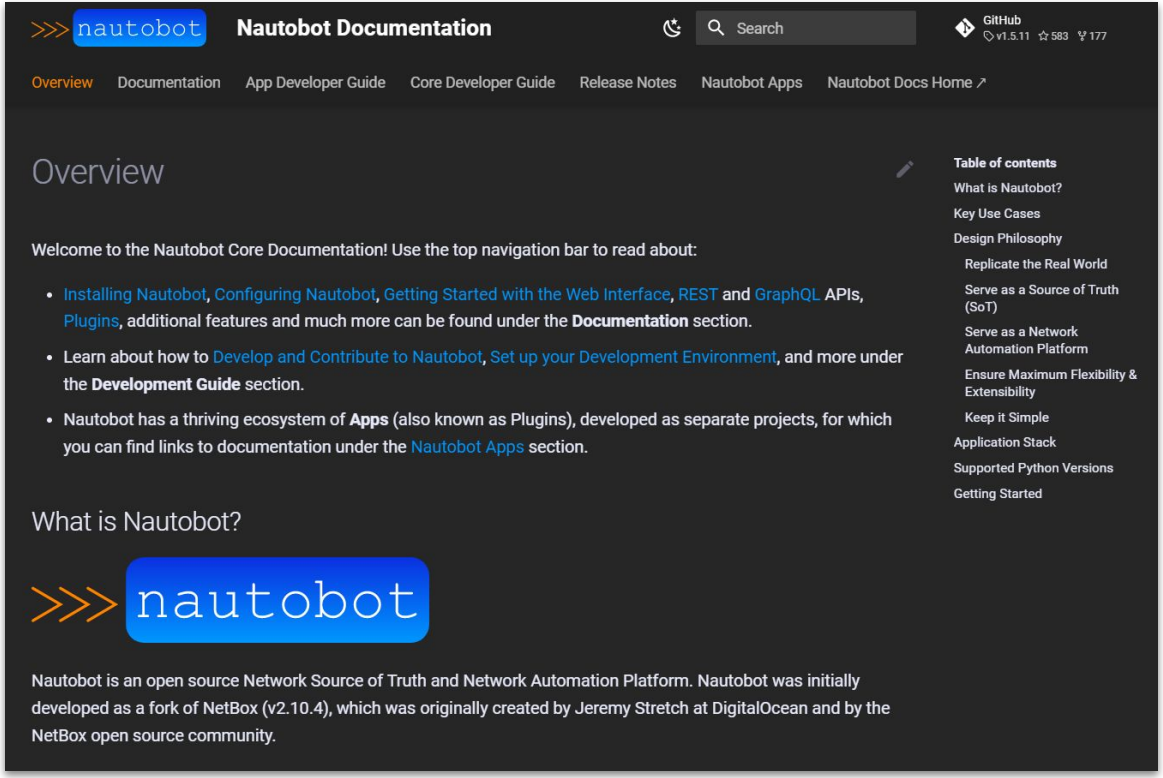
Course Agenda

1. Why Network Automation?
2. Nautobot Overview and Installation
3. Nautobot Organization and Devices
4. Nautobot IPAM to Circuits
5. Nautobot Apps
6. Nautobot Programming
7. Nautobot Extensibility


Docs

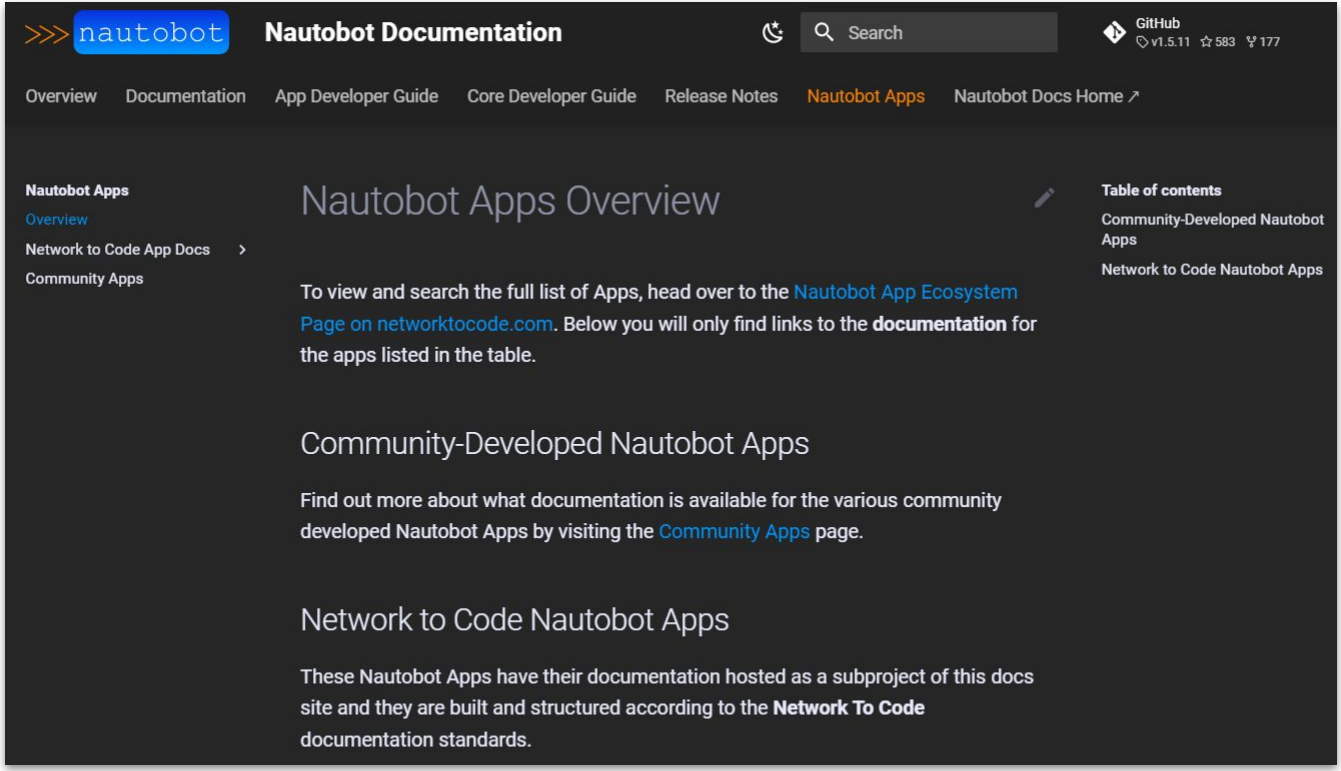
Nautobot Core Documentation

https://docs.nautobot.com/projects/core/en/stable/



Nautobot App Documentation

https://docs.nautobot.com/projects/core/en/stable/apps/



Code

>>> Nautobot Application Programming Interfaces

- **Web UI** - Primary interface for human interaction - form based workflows.
  - https://<nautobot>/<app>/<model>/<name>/
  - https://<nautobot>/dcim/sites/lax
- **Python SDK, pynautobot:** https://pynautobot.readthedocs.io/en/latest/#
- **API clients:** Postman, curl, Python requests etc.
  - Interactive REST API documentation built into Nautobot (/api/docs/)
  - Interactive GraphQL client built into Nautobot (/graphql/)
- **Ansible module: *nautobot-ansible***
  - Modules - e.g. nautobot_device, nautobot_interface, nautobot_prefix
- **Python shell (admin/dev only!)**
  - On-box Interactive shell pre-loaded with Nautobot models.
  - Direct and unrestricted access to the database, generally used for development.

>>> REST API and GraphQL

{ REST }

Expose Create/Read/Update and Delete Endpoints for most objects in the database.

List of endpoints documented with OpenAPI / Swagger. Native support for pagination

- Manipulate one object type at a time (device, interface ..)
- Bulk query, create, update
- The resource is indicated in the url
- Optimized for large volume

Standard query language to query and traverse multiple objects in the database.

Schema of the database generated dynamically and exposed externally.

- Select exactly what should be returned.
- Query multiple type of objects at the same time.
- Optimized for complex query
- Not optimized for large volume

ReST API

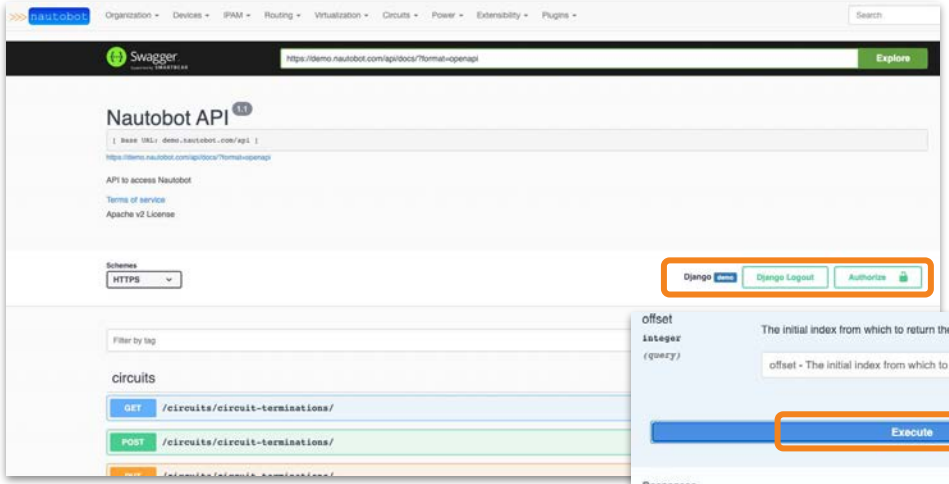
>>> REST API

Nautobot Application Interfaces

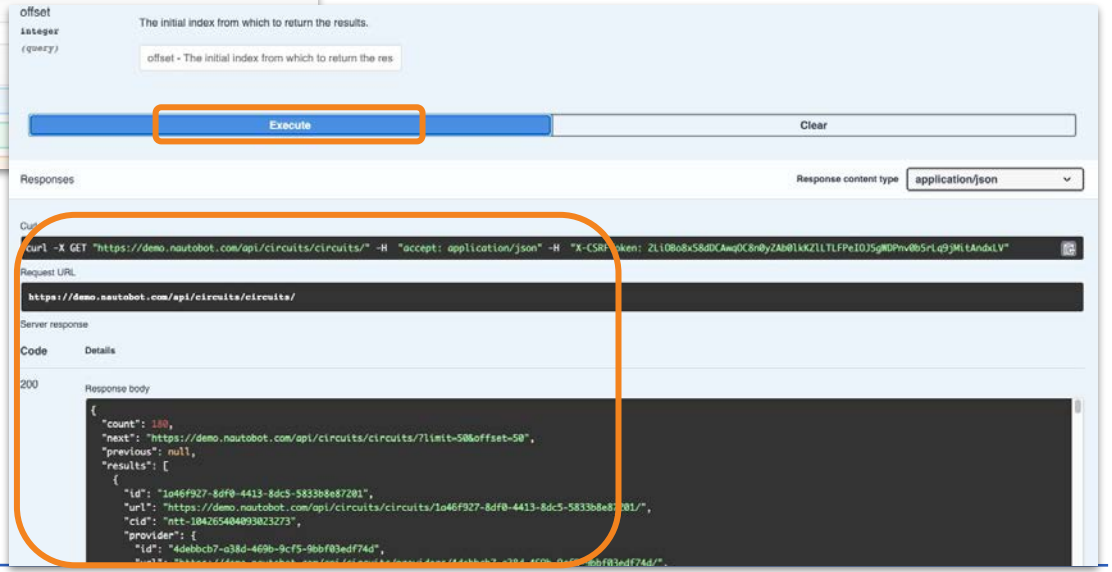
- **Nautobot REST API** - Primary interface for machine-to-machine interaction.
- Methods:
  - GET
  - POST
  - PUT
  - PATCH
  - DELETE

- Objects uniquely identified by ID (primary key)
  - …/api/dcim/devices/**9c30528e-a0e8-4577-9c91-fe61df57e0b2**/
  - https://<nautobot>/api/<app>/<model>/
  - https://<nautobot>/api/docs/
    - human friendly HTML interactive documentation

Interactive Nautobot API with Swagger



Login via API w/ User or API Token



Get example via curl, example URL, and responses.

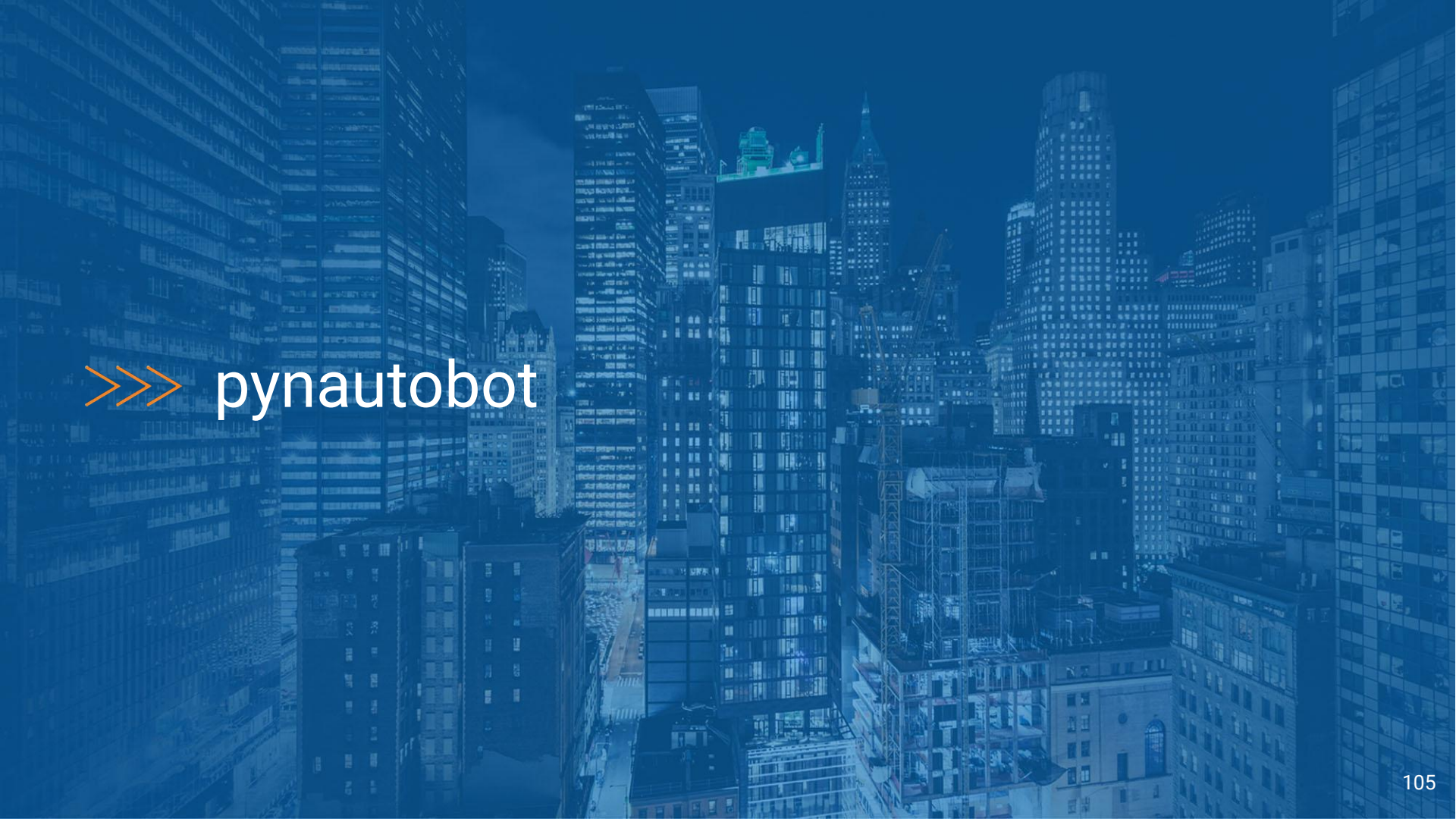
Possible filters also usable and viewable.

Demo 9

Discovering Swagger and Postman for REST APIs

Lab 8

Discovering Swagger for REST APIs


pynautobot

>>> pynautobot

- SDK for the Nautobot API
- Developed as a fork of pynetbox, in order to extend support for features unique to Nautobot
- Supports
  - Queries
  - Threading
  - API Versioning
  - Retries
- Github
  - https://github.com/nautobot/pynautobot

>>> pynautobot

```
import pynautobot
nautobot = pynautobot.api(
   url="https://demo.nautobot.com",
   token="aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa",)
# for all devices
devices = nautobot.dcim.devices.all()
for device in devices:
   print(device.name)
> aaa00-cwdm-01
> …(cut)
# for a single device
device = nautobot.dcim.devices.get(name='atl01-edge-02')
print(device.name)
> atl01-edge-02
```

Demo 10

Discovering Pynautobot


nautobot-ansible

>>> nautobot-ansible

```yaml
---
plugin: networktocode.nautobot.inventory
api_endpoint: https://demo.nautobot.com
token: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
validate_certs: True
config_context: False
group_by:
 - device_roles
device_query_filters:
 - has_primary_ip: 'true'
```

>>> nautobot-ansible (output)

```
{
    "_meta": {
        "hostvars": {
            "ams01-edge-01": {
                "ansible_host": "10.11.128.1",
                "custom_fields": {
                    "bgp_ecmp_maximum_paths": null
                },
                "device_roles": [
                    "edge"
                ],
                "device_types": [
                    "dcs-7280cr2-60"
                ],
                "is_virtual": false,
                "local_context_data": [
                    null
                ],
                "manufacturers": [
                    "arista"
[cut]
```

```
"device_roles_edge": {
        "hosts": [
            "ams01-edge-01",
            "ams01-edge-02",
            "ang01-edge-01",
            "ang01-edge-02",
            "atl01-edge-01",
            "atl01-edge-02",
            "atl02-edge-01",
            "atl02-edge-02",
            "azd01-edge-01",
            "azd01-edge-02",
[cut]
    "device_roles_leaf": {
        "hosts": [
            "ams01-leaf-01",
            "ams01-leaf-02",
            "ams01-leaf-03",
            "ams01-leaf-04",
            "ams01-leaf-05",
            "ams01-leaf-06",
```

Demo 11

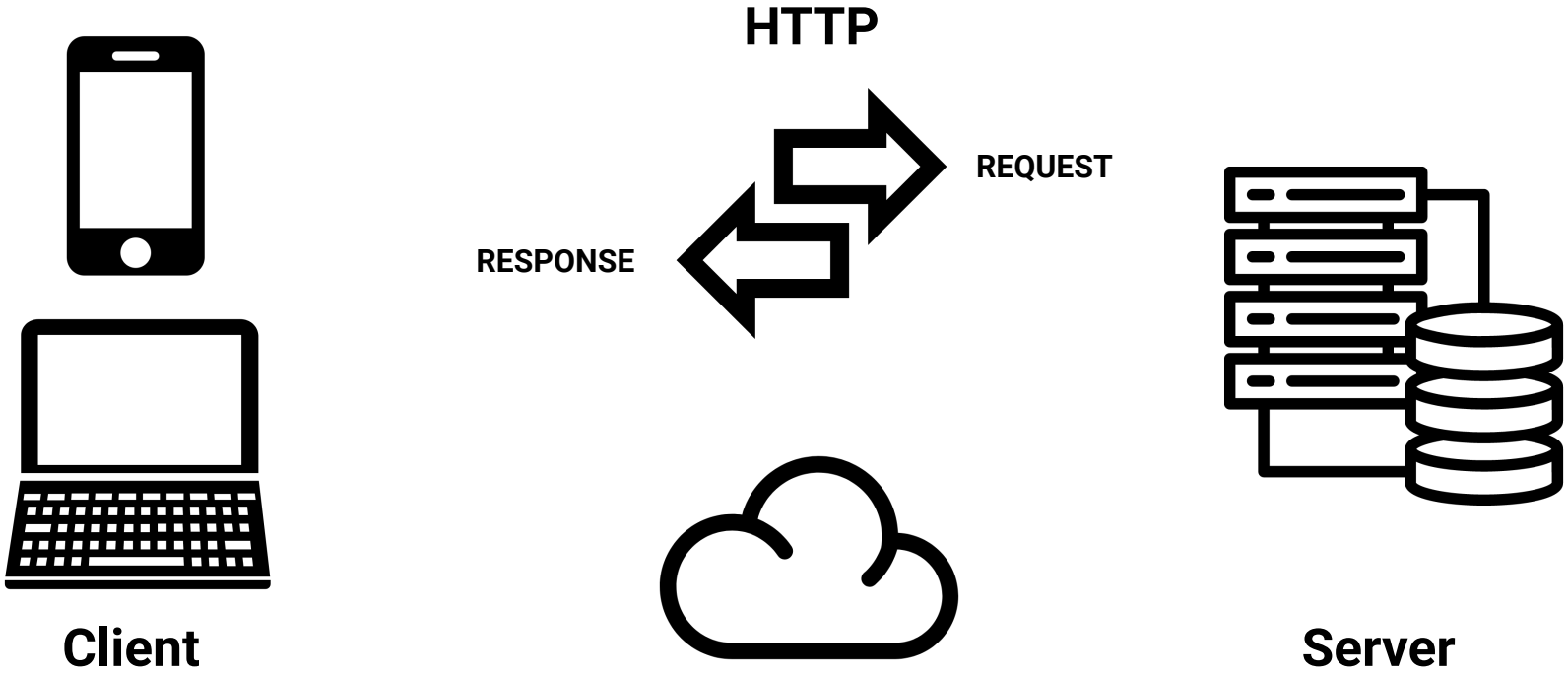
Applying Ansible with nautobot-ansible

GraphQL API

>>> What is GraphQL?

*"GraphQL is a **query language for APIs** and a **runtime for fulfilling those queries** with your existing data. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need and nothing more, makes it easier to evolve APIs over time, and enables powerful developer tools."*

>>> What is an API



>>> GraphQL Origin

- Developed by Facebook
  - Factors:
    - Increased mobile applications
    - Low powered devices
    - Variable network connections
- Open-source specification in 2015

>>> GraphQL Overview

- Hierarchical
  - Query is shaped like the response data
- Client specified queries
  - Declarative data fetching
- Application-Layer
  - Most commonly uses HTTP
  - Transport independent
  - String interpreted by server
- Strongly-Typed
  - Tools can validate syntax
- Introspective
  - Clients can query GraphQL server to understand the available type system


airbnb


GitHub


coursera


NBC

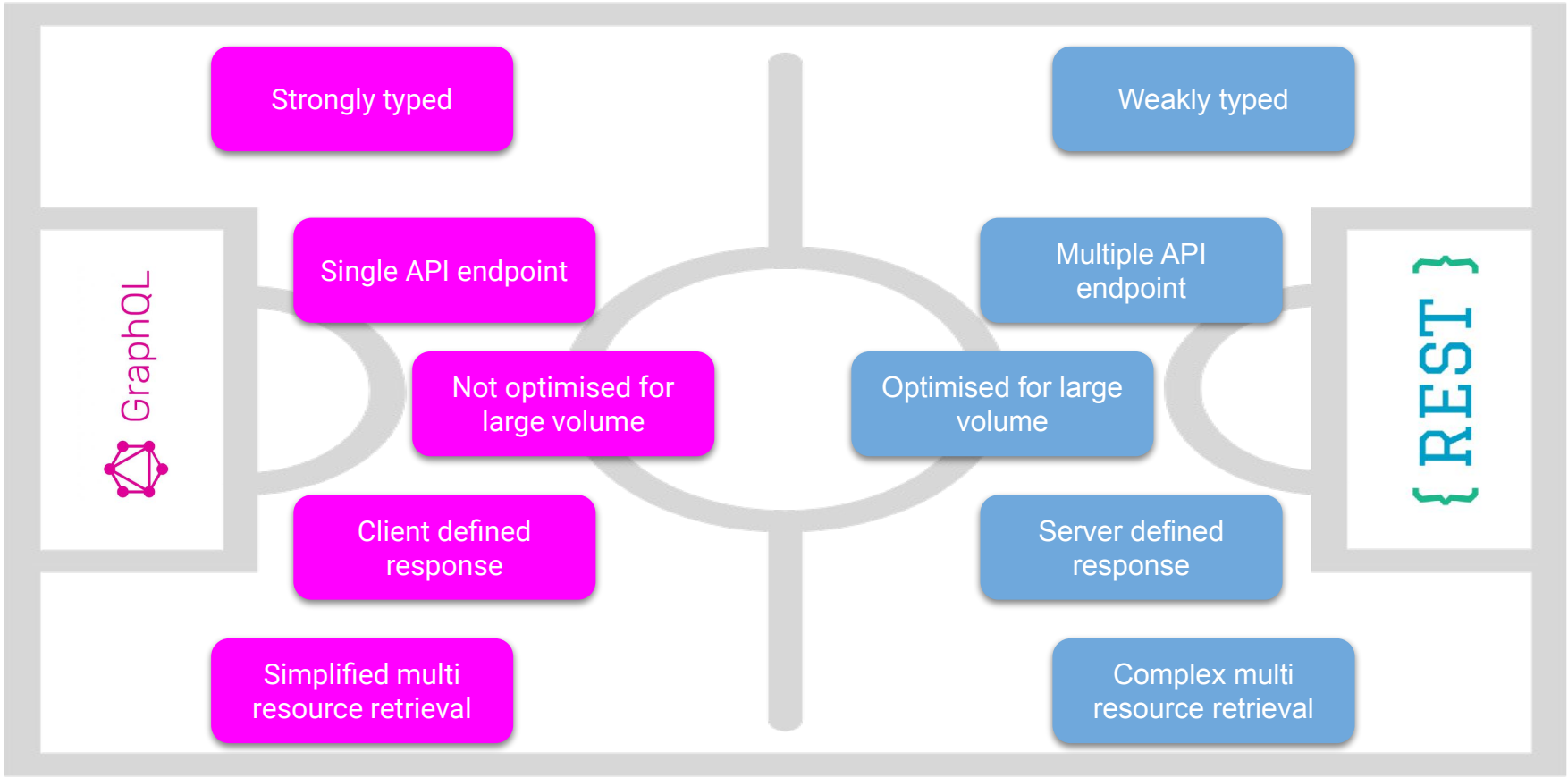

ATLASSIAN


FORWARD
NETWORKS


PayPal


shopify

>>> GraphQL vs REST



GraphQL vs. REST API Case Study

Tim Fiola

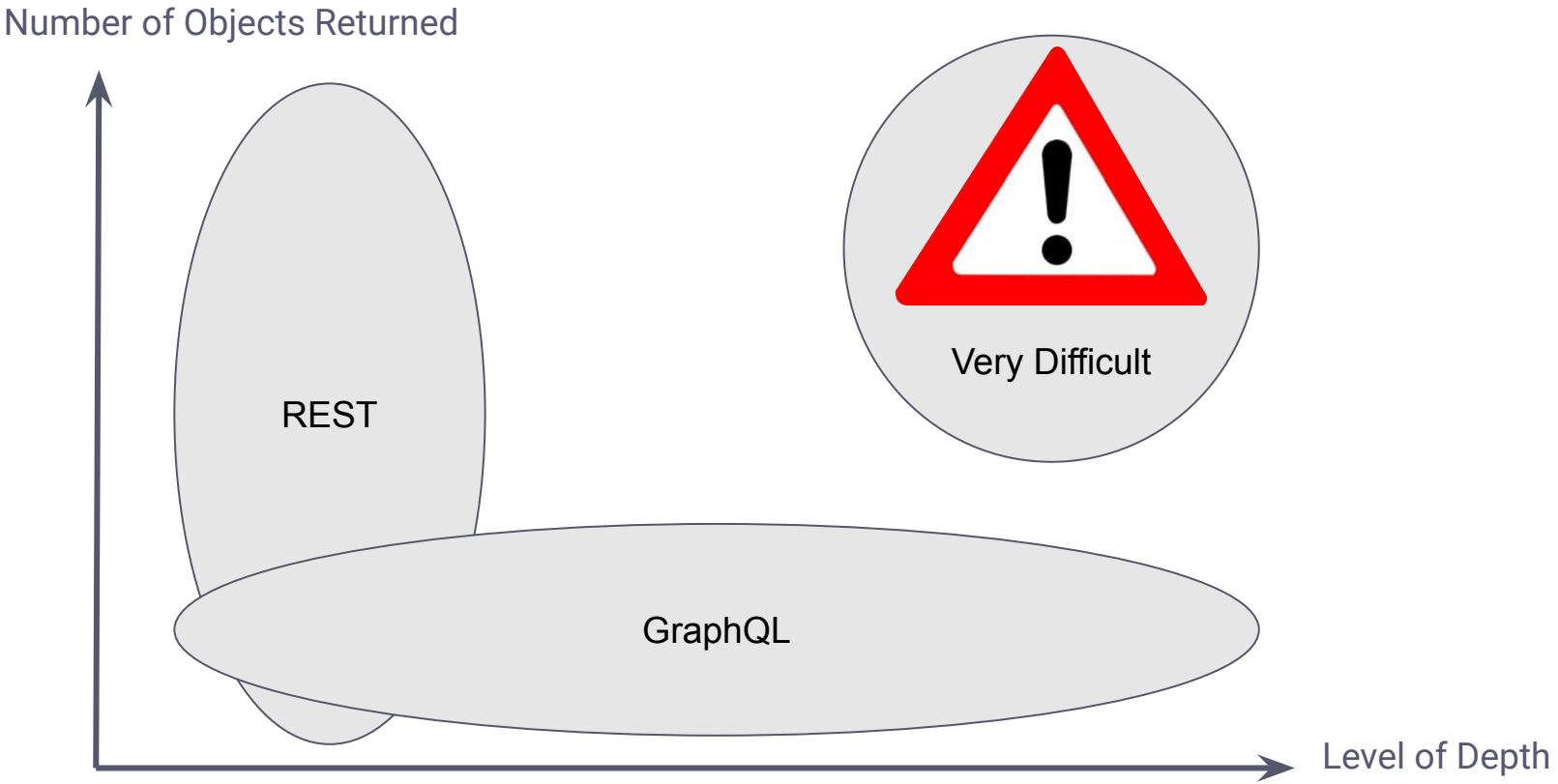
The Problem Statement

In this case study, the goal is to gather specific information for certain network elements from Nautobot. Specifically, we want a data structure with the following information:

- We want information from all devices within the ***ams*** site
- The data structure should organize information so that all the data is grouped on a per-device basis
- We want this specific data for each device:
    - Device name
    - Device role
    - All the interface names
    - The list of IP address(es) for each interface, even if the interface has no configured IP address(es)

Method	Average Run Time	# of Queries	Time to Create Script
GraphQL	2.2 seconds	1	~ 20 minutes
RESTful	14.9 seconds	17	~ 200 minutes+

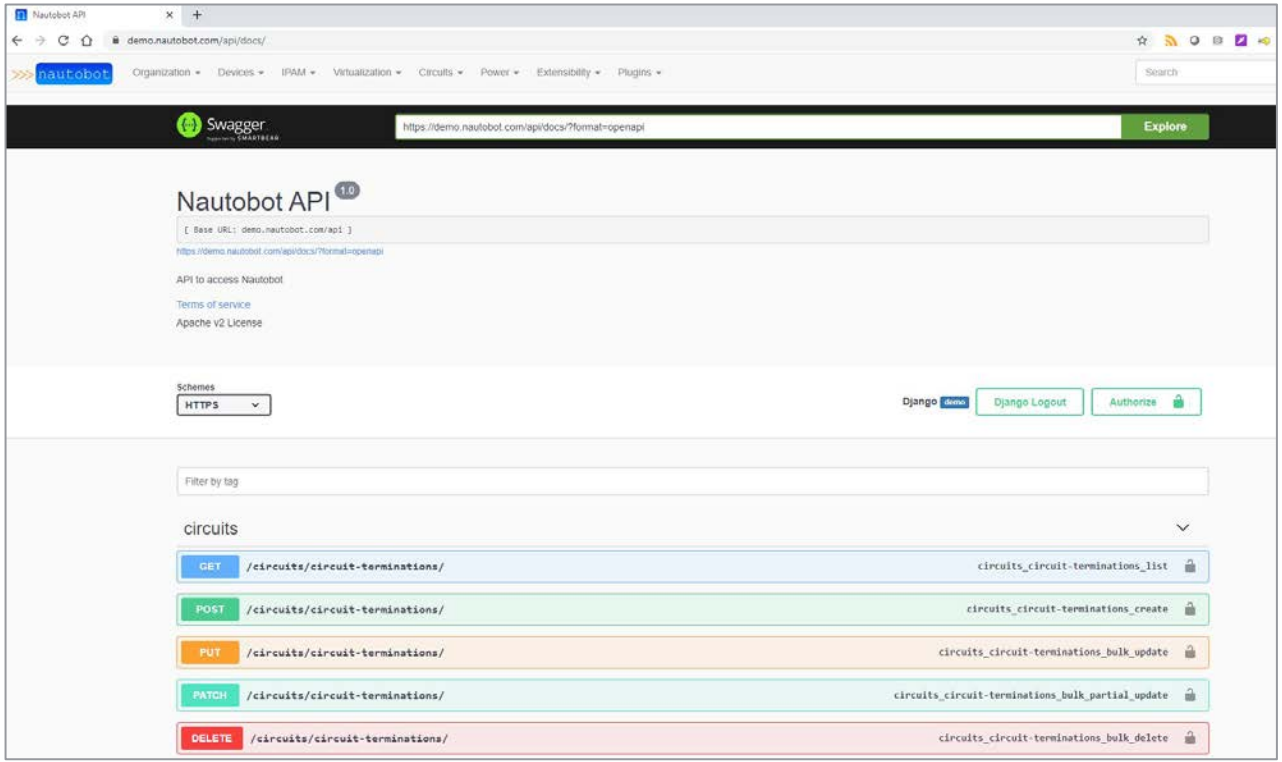
>>> Query Optimisation



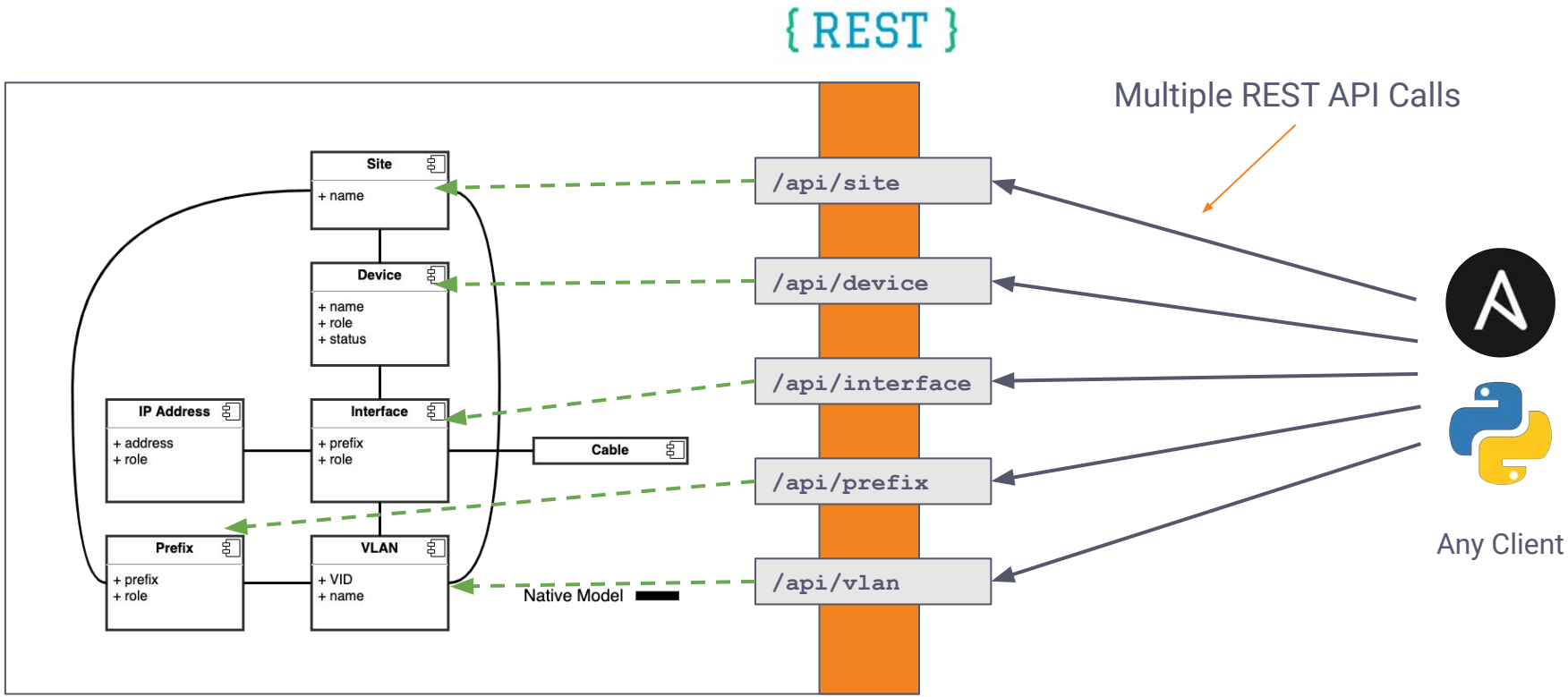
Nautobot API Documentation

Nautobot employs a REST API for CRUD operations and GraphQL for read only.

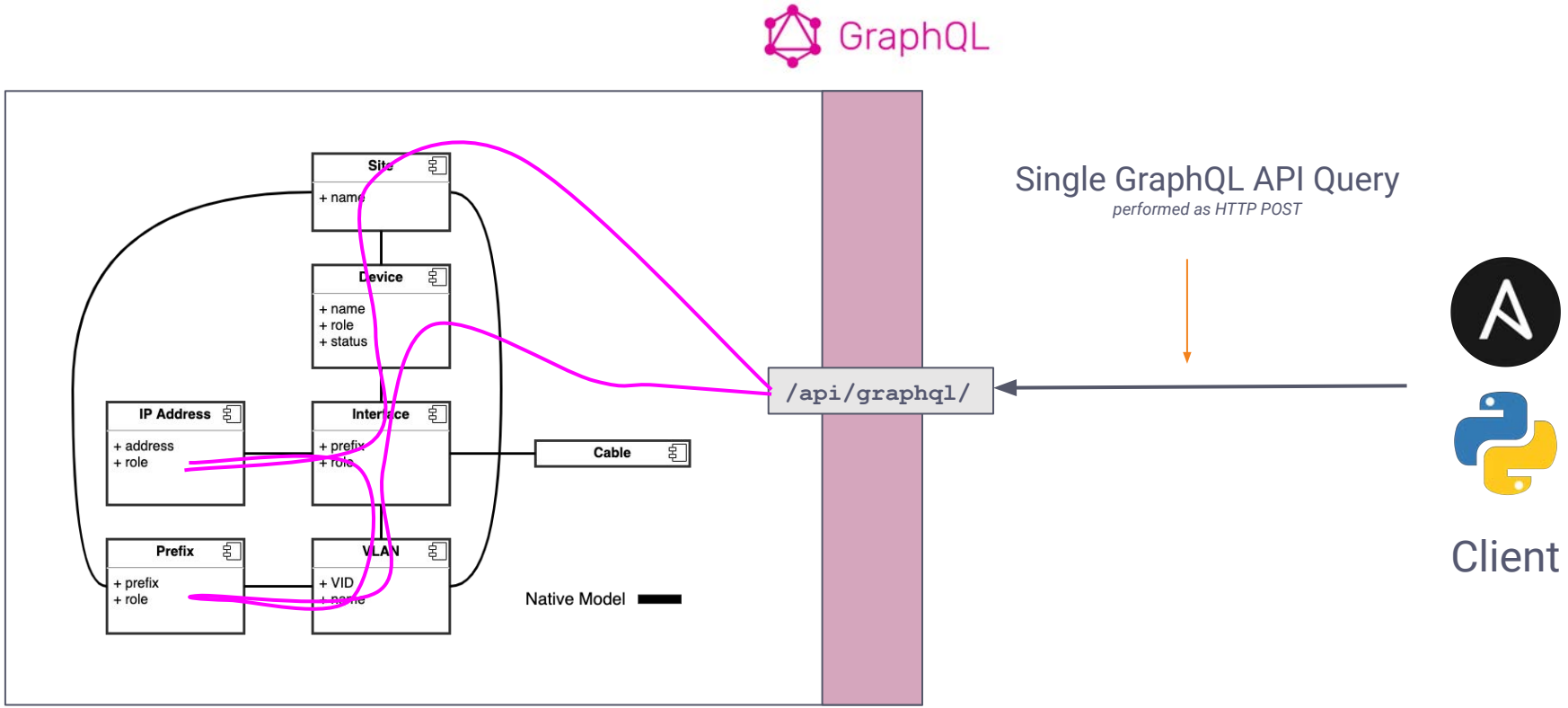
Interactive Documentation on each install



>>> REST API Endpoints



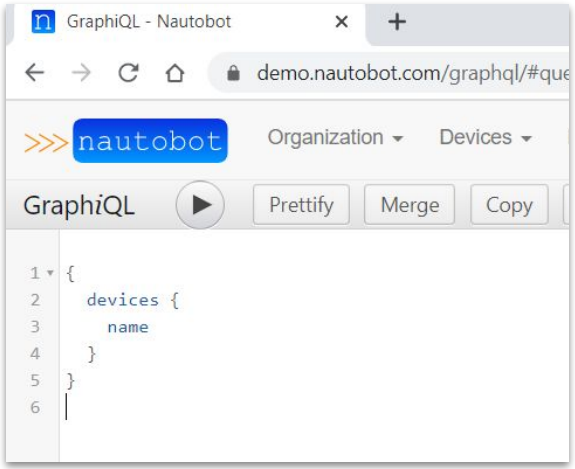
>>> GraphQL API Endpoint



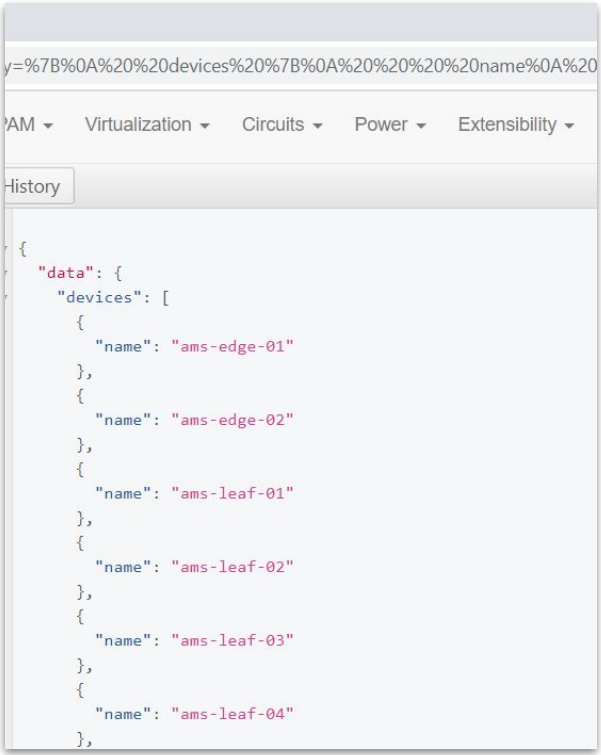
GraphQL in Nautobot

- Query Language for Relational Data
- Resources defined by a GraphQL Schema
- Client sends query
- Server orchestrates data
- Supports Read-only Operations

Nautobot supports an on-box GraphQL browser to test API queries



Query



Response

Demo 12

Discovering GraphiQL for GraphQL

Lab 8

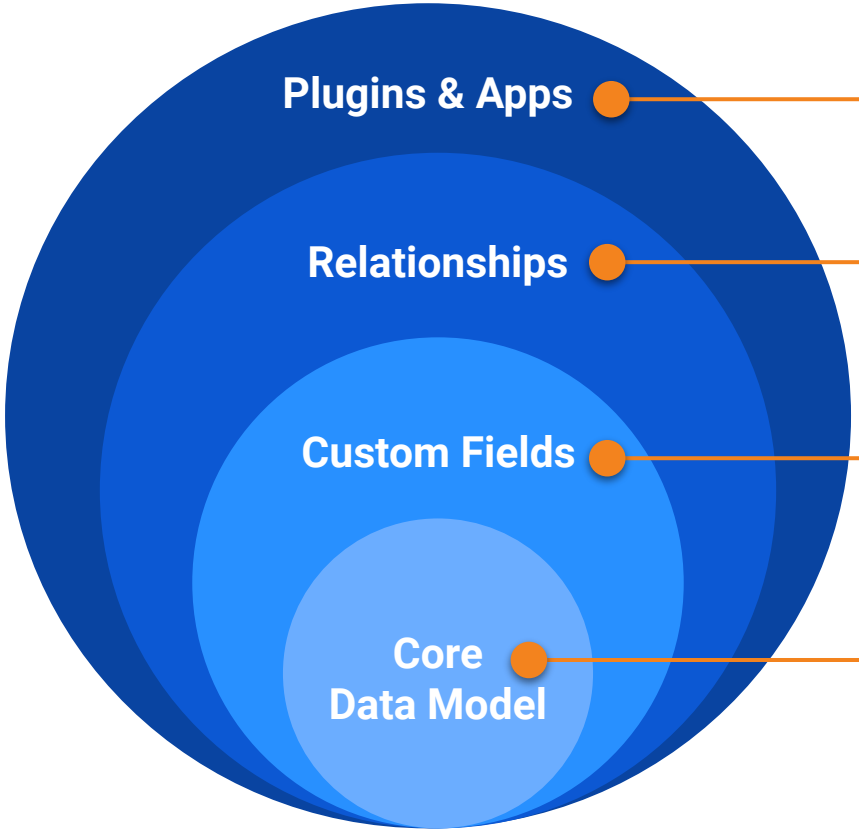
Discovering GraphiQL for GraphQL

Course Agenda

1. Why Network Automation?
2. Nautobot Overview and Installation
3. Nautobot Organization and Devices
4. Nautobot IPAM to Circuits
5. Nautobot Apps
6. Nautobot Programming
7. Nautobot Extensibility

Nautobot Extensibility

>>> Nautobot Data Model Layers



Larger bodies of functionality and reusable in the ecosystem. ChatOps, Circuit Maintenance, and many more

Defining custom associations between data. Ex: VLAN per rack, VLAN per circuit

Organizational specific attributes like interface role, site code, etc.

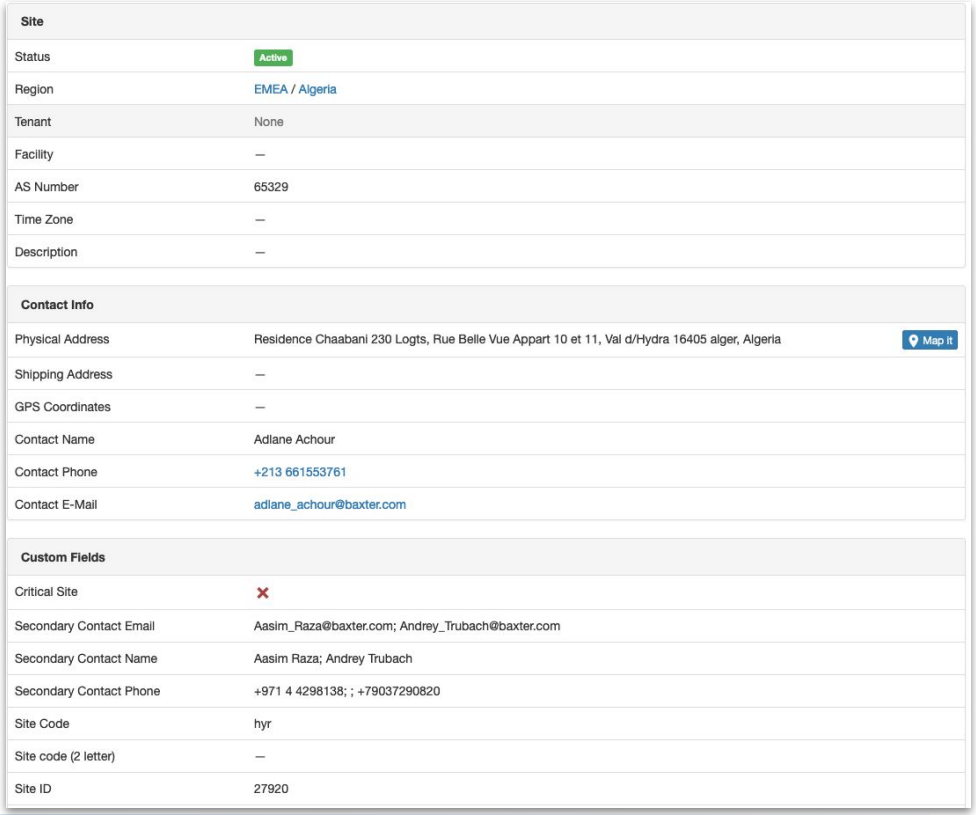
Devices, Sites, VLANs, Prefixes, Interfaces, Circuits, etc.

>>> Custom Fields

Custom fields enable attribute level use cases by augmenting existing models.

- Treated like normal database fields
- Native support in all interfaces
  - Web UI
  - REST API
  - GraphQL
  - ORM

Network Automation use cases like configuration parameters, role context, etc.



Site	
Status	Active
Region	EMEA / Algeria
Tenant	None
Facility	—
AS Number	65329
Time Zone	—
Description	—
Contact Info	
Physical Address	Residence Chaabani 230 Logts, Rue Belle Vue Appart 10 et 11, Val d/Hydra 16405 alger, Algeria Map it
Shipping Address	—
GPS Coordinates	—
Contact Name	Adlane Achour
Contact Phone	+213 661553761
Contact E-Mail	adlane_achour@baxter.com
Custom Fields	
Critical Site	✗
Secondary Contact Email	Aasim_Raza@baxter.com; Andrey_Trubach@baxter.com
Secondary Contact Name	Aasim Raza; Andrey Trubach
Secondary Contact Phone	+971 4 4298138; ; +79037290820
Site Code	hyr
Site code (2 letter)	—
Site ID	27920

>>> Relationships

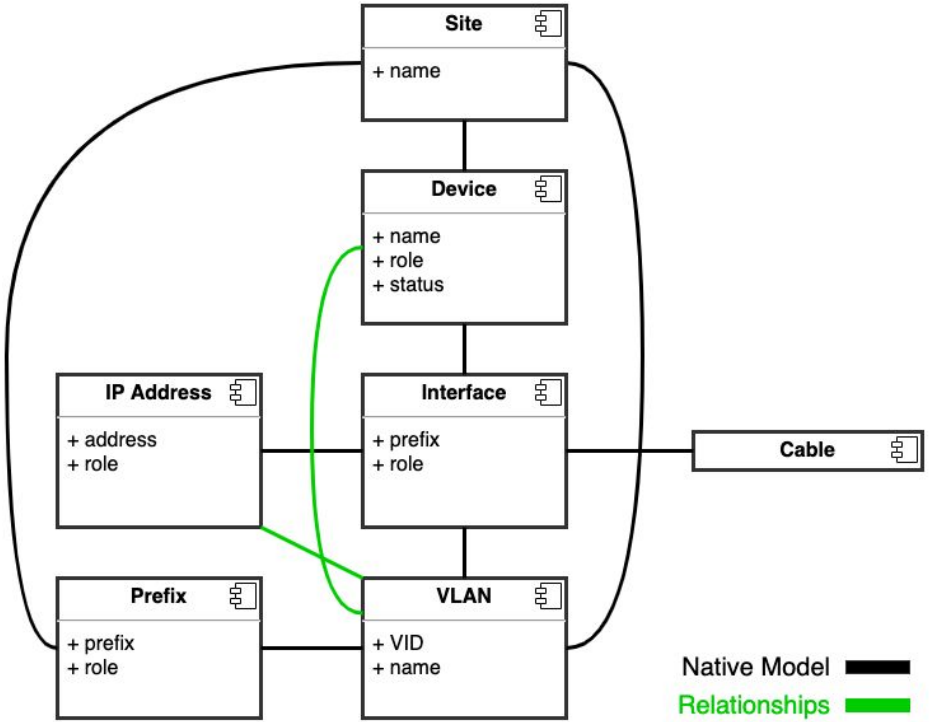
Relationships allow creating data associations that are meaningful to your use cases.

The network data model implements a series of relationships.

An organization may have needs that go beyond the existing model.

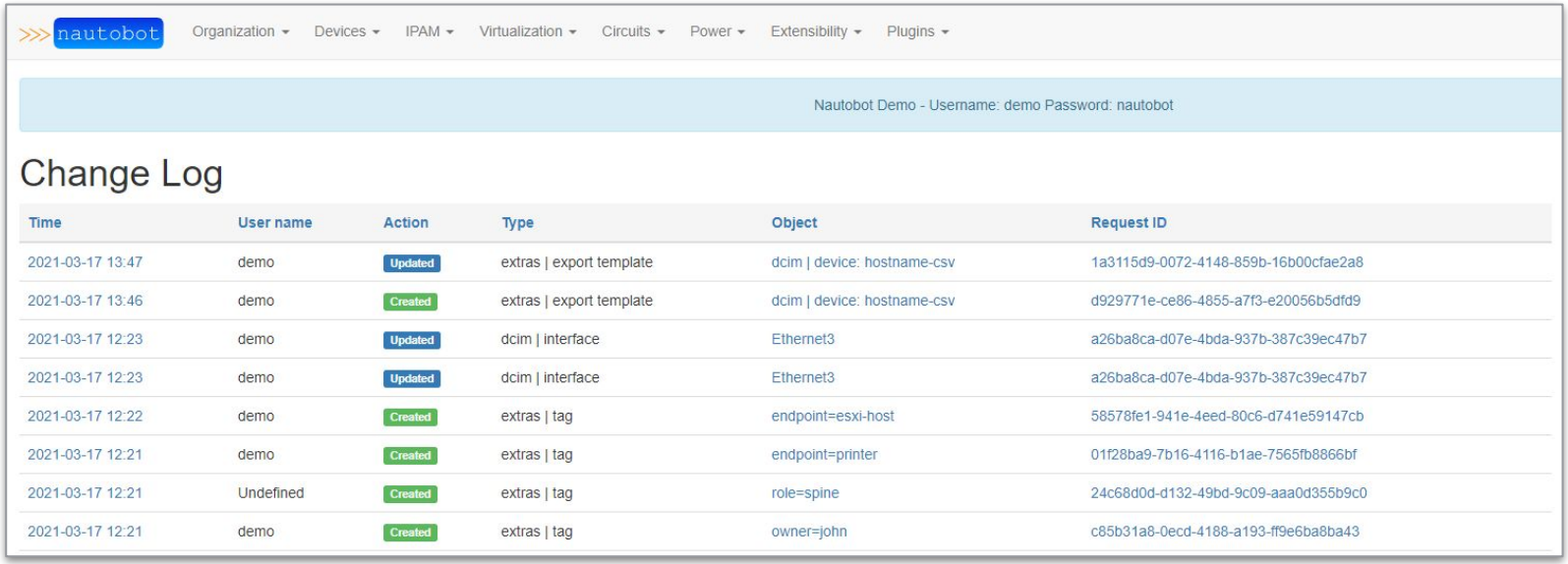
Some network automation use cases become easier with these custom relationships.

- VLANs to Device
- VLAN L3 IP Address
- Site primary Device



>>> Change Log

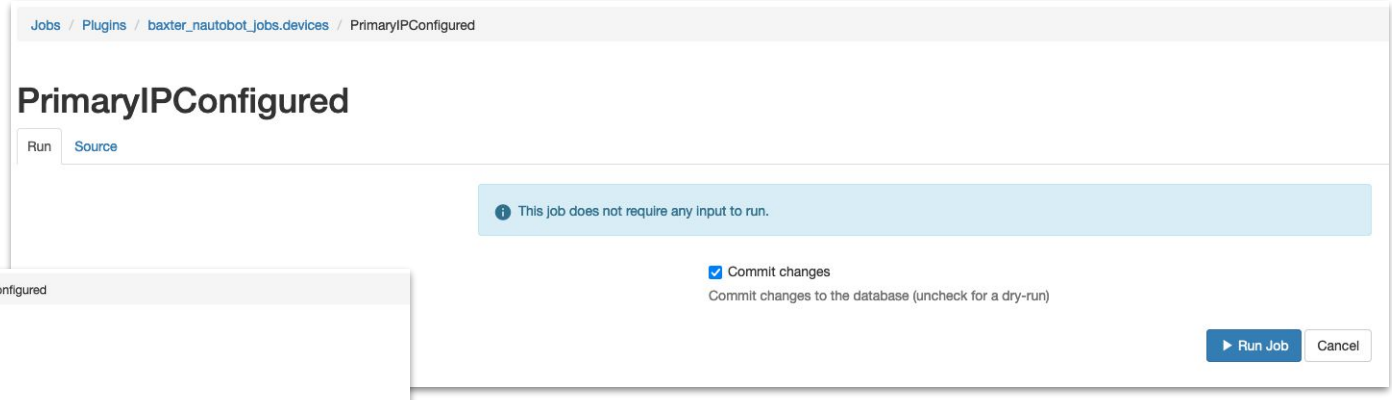
Nautobot's change log shows an audit trail of all user operations performed within Nautobot.

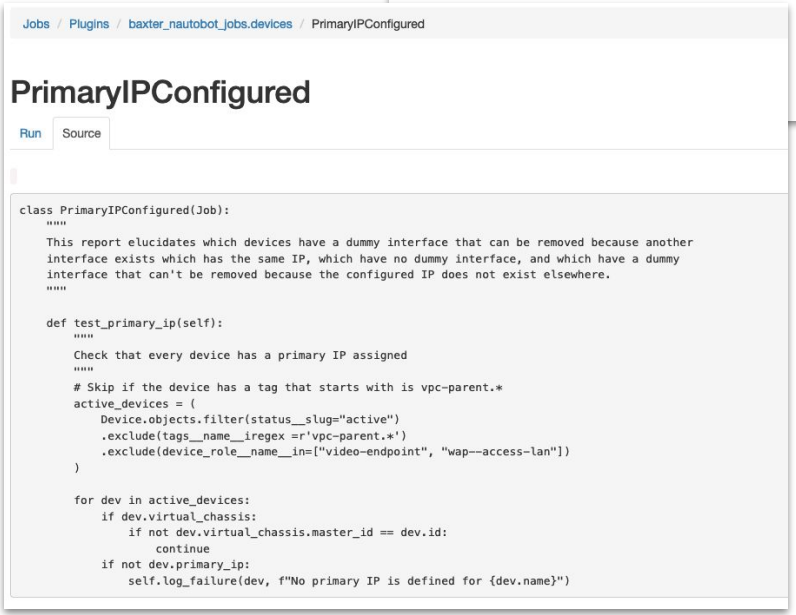


Jobs

>>> Jobs Examples

Run





Source code the job will execute

Demo 13

Discovering Extensibility

Lab 9

Discovering Extensibility

Network Automation Positive Outcomes

How to measure and predict value

>>> Positive Outcomes

How to measure the value of network automation

The following Key Performance Indicators (KPIs) provide the data needed to show the benefits of network automation including: increased reliability, operational efficiency, acceleration of delivering new products, reduced operational cost, risk mitigation, compliance improvements, and overall increased business agility.

% of total network changes that are automated	**% of emergency changes**	**% of backed out changes**	**% of incidents caused by changes**	**Number of Network devices per network administrator**
Average time per phase for new network service	**% of successful changes**	**Average time of roll-back**	**% of out-of-hours changes**	**Time to detect network failure**
Average time per phase for network change	**% of compliance errors**	**Number of changes in the backlog**	**Resources used and funds spent on changes**	**Time to repair network failure**

>>> Positive Outcomes

Predict these benefits from network automation

Lower Opex

- Simplify your underlying infrastructure
- Consume fewer hours configuring, provisioning, and managing network services

Improve Reliability and Security

- Reduce the chance for human errors
- Deliver a higher level of services
- Drive consistency across the network

Increase Innovation

- Increase productivity by allowing humans to do higher level work
- Allocate more time to drive business strategy and innovation

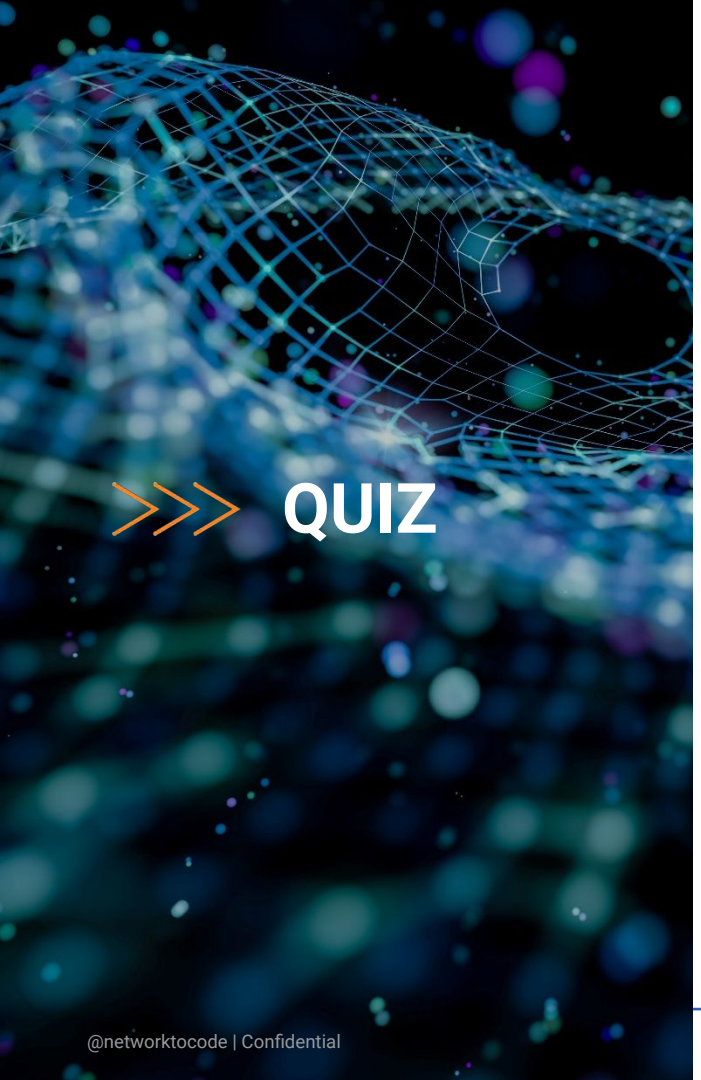
Greater Insight and Network Control

- Gain more visibility into the network
- Allow IT operations to become more responsive to change through analytics

Increase Business Agility

- Develop new network operational models
- Improve time-to-market
- Build, test, deploy, and optimize new network services quickly and continuously

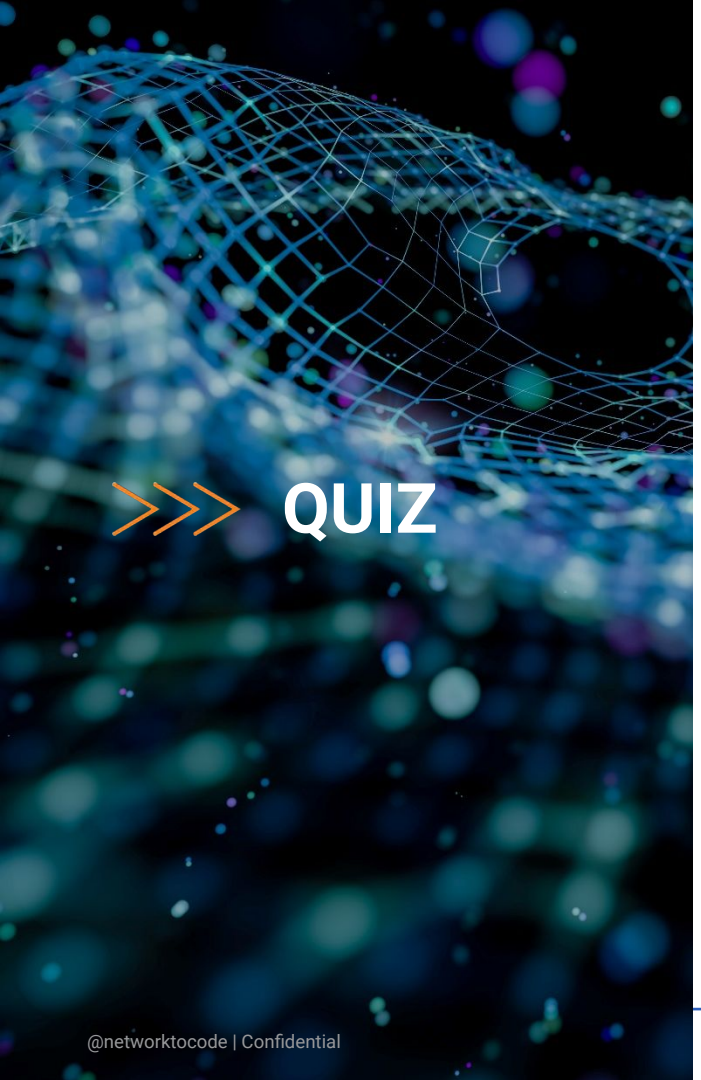
Summary and Next Steps



QUIZ

The Source of Truth is based on data that comes from:

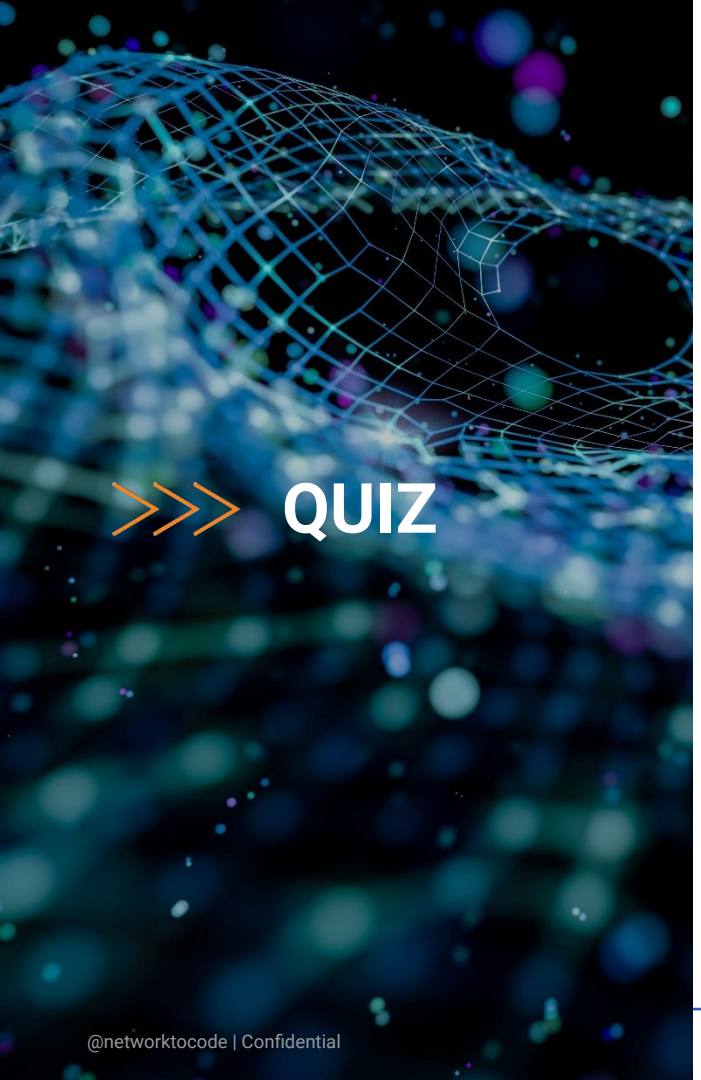
a. Intent
b. Reality



QUIZ

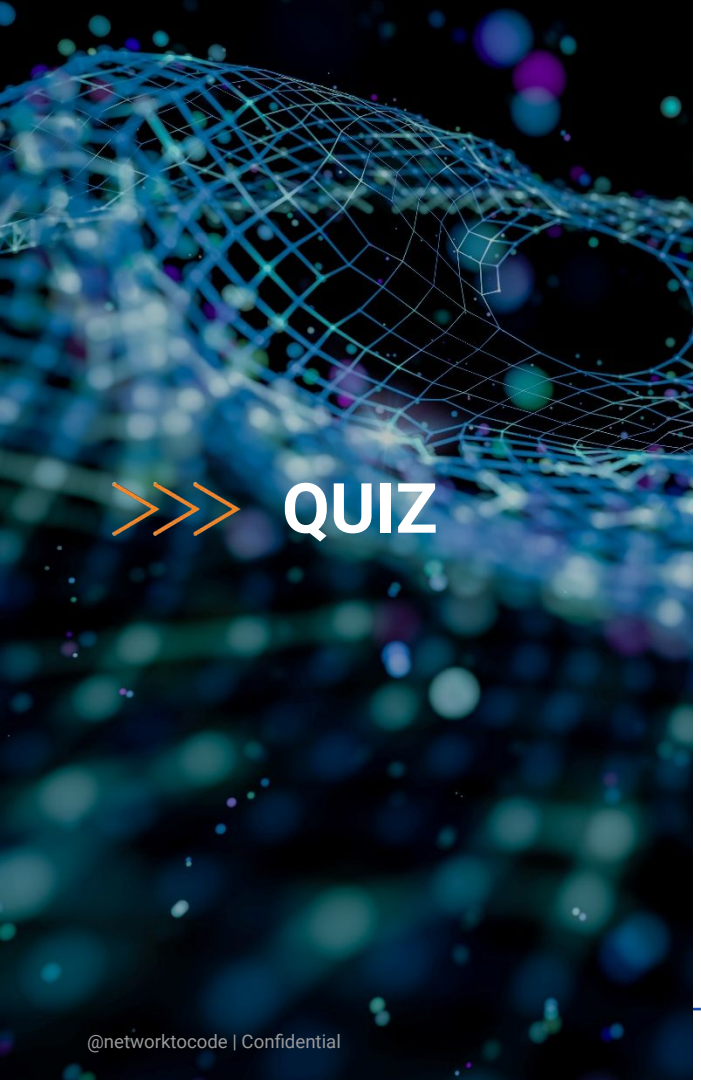
The Source of Truth is based on data that comes from:

a. Intent
b. Reality



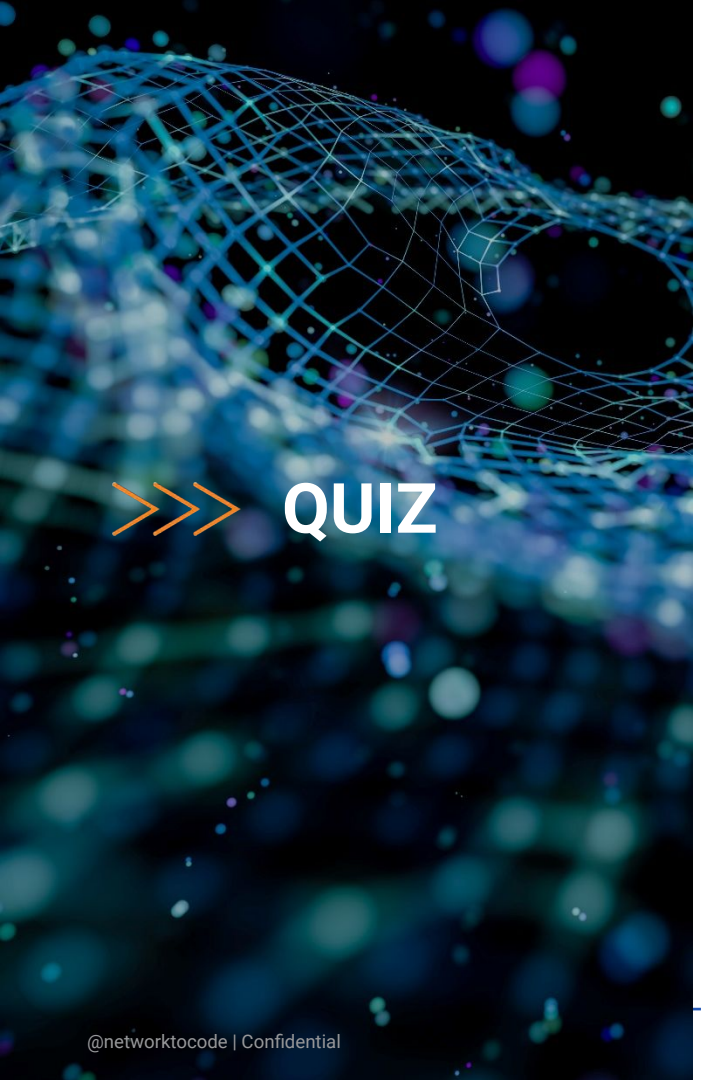
Which of the following are the main use cases for Nautobot?

a. Application Platform
b. Automation Platform
c. Distributed Platform
d. Security Platform
e. Source of Truth
f. Web UI



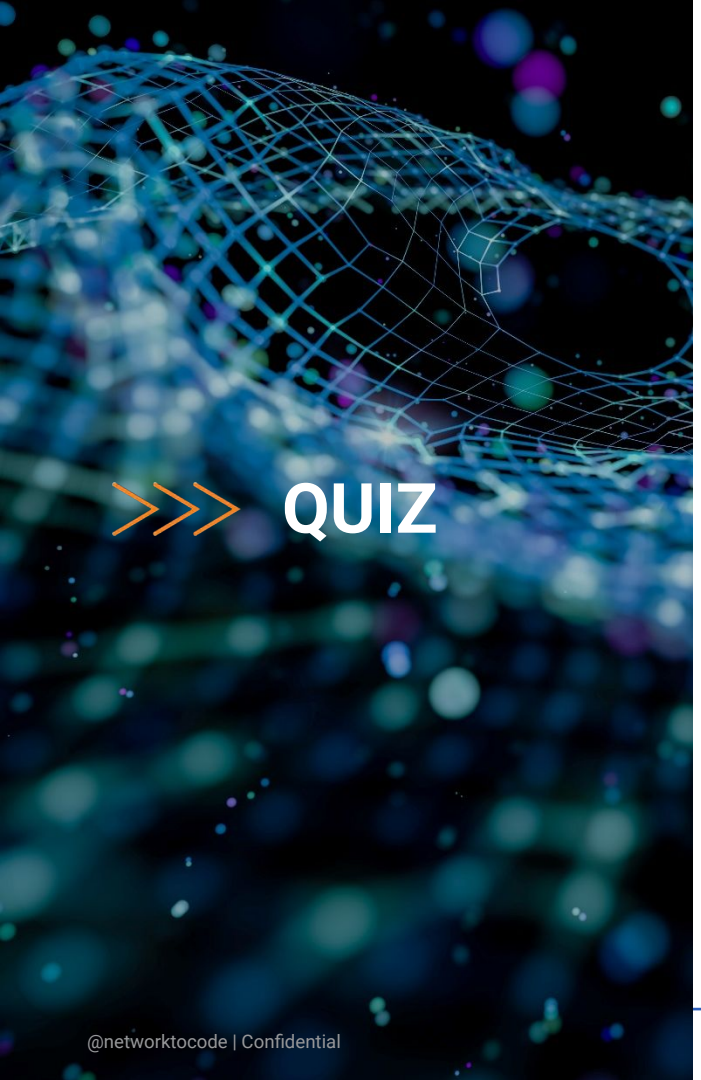
Which of the following are the main use cases for Nautobot?

a. Application Platform
b. Automation Platform
c. Distributed Platform
d. Security Platform
e. Source of Truth
f. Web UI



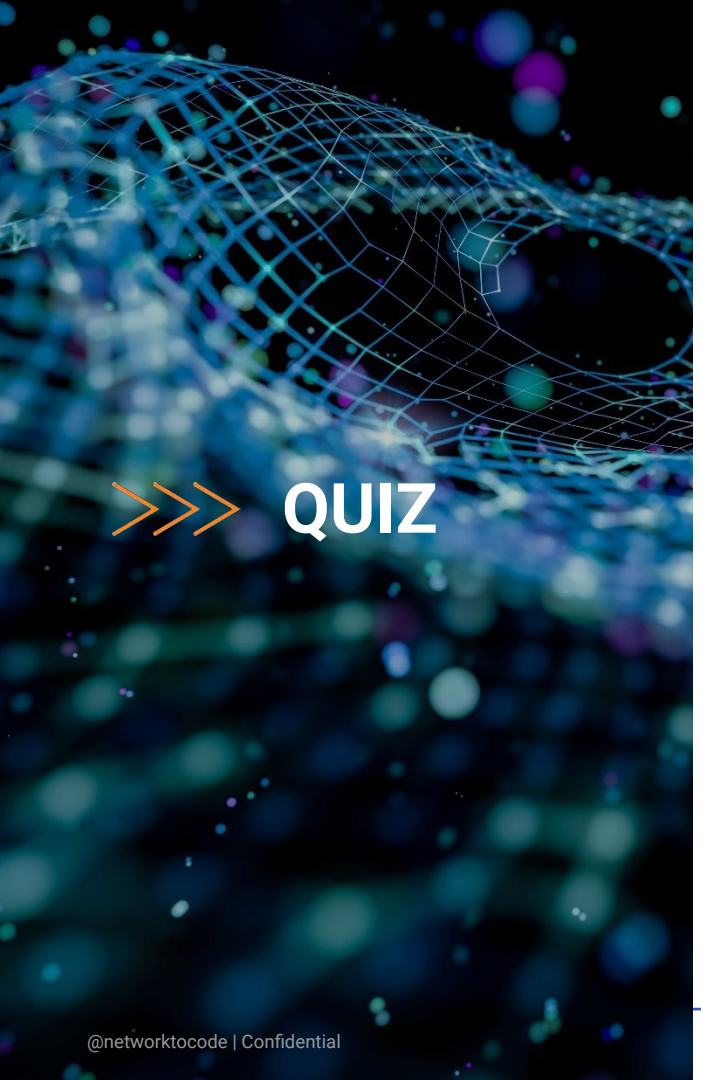
Encrypted secrets information is kept within Nautobot

a. True
b. False



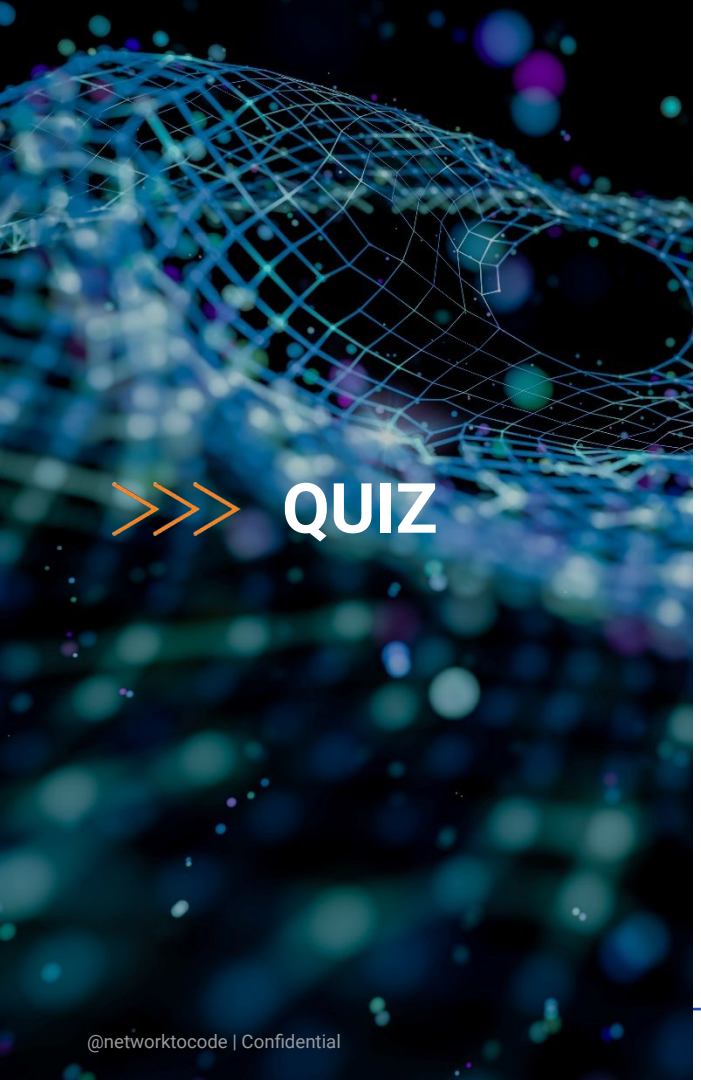
Encrypted secrets information is kept within Nautobot

a. True
b. False



Nautobot requires how much programming effort?

a. None
b. Little
c. Moderate
d. As much as you want
e. All of the above



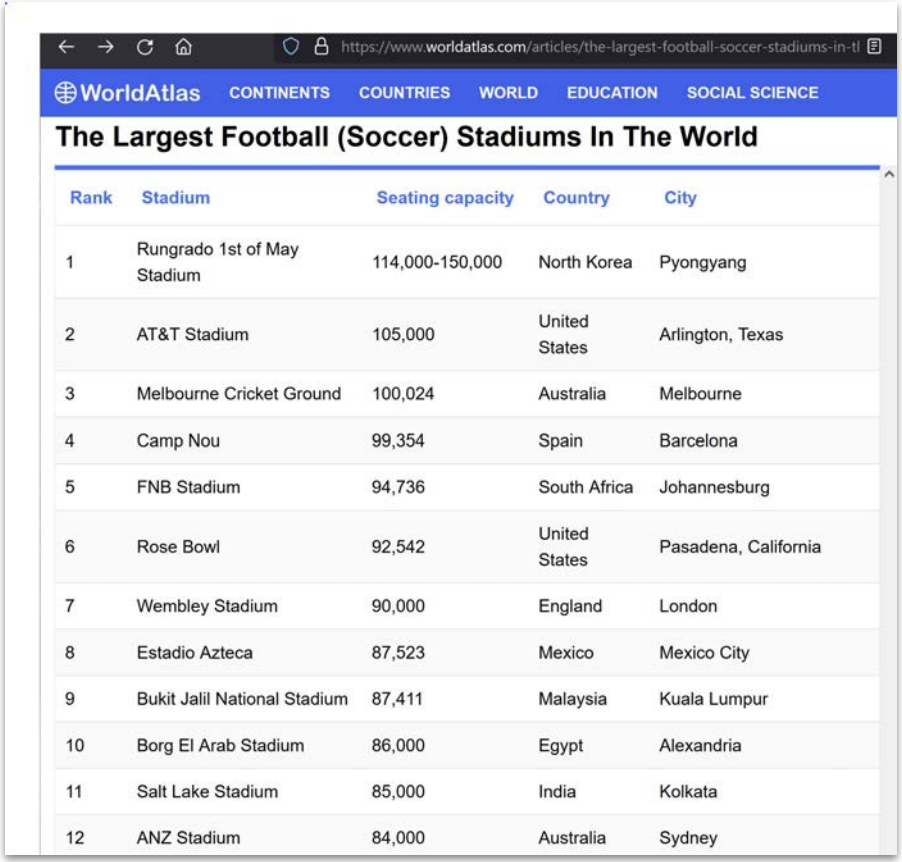
Nautobot requires how much programming effort?

a. None
b. Little
c. Moderate
d. As much as you want
e. All of the above

Demo 14

Summarizing the Experience

>>> Welcome to the Networked Football (Soccer) Stadiums Club! Congrats!



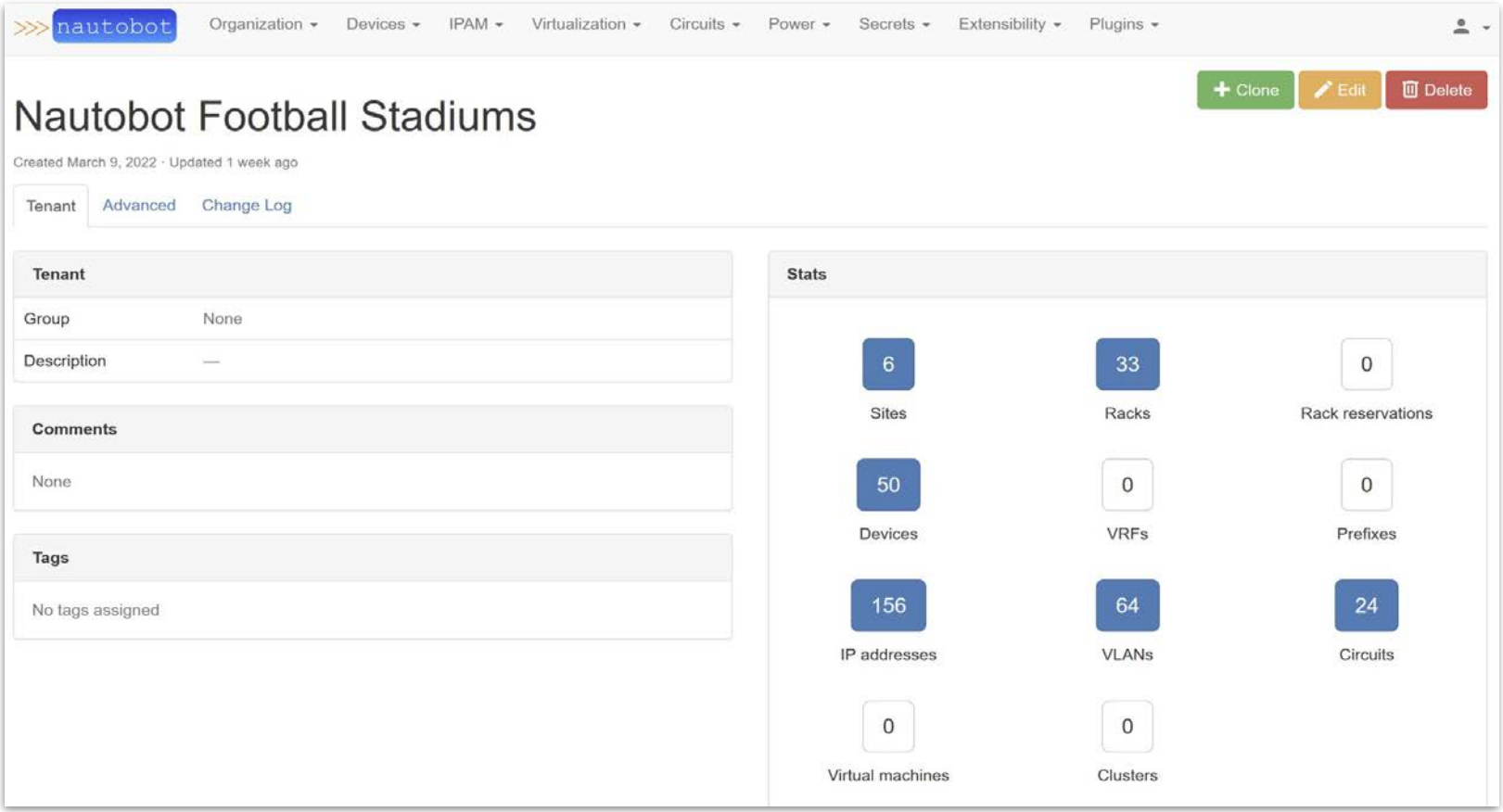
https://www.worldatlas.com/articles/the-largest-football-soccer-stadiums-in-tl

WorldAtlas CONTINENTS COUNTRIES WORLD EDUCATION SOCIAL SCIENCE

The Largest Football (Soccer) Stadiums In The World

Rank	Stadium	Seating capacity	Country	City
1	Rungrado 1st of May Stadium	114,000-150,000	North Korea	Pyongyang
2	AT&T Stadium	105,000	United States	Arlington, Texas
3	Melbourne Cricket Ground	100,024	Australia	Melbourne
4	Camp Nou	99,354	Spain	Barcelona
5	FNB Stadium	94,736	South Africa	Johannesburg
6	Rose Bowl	92,542	United States	Pasadena, California
7	Wembley Stadium	90,000	England	London
8	Estadio Azteca	87,523	Mexico	Mexico City
9	Bukit Jalil National Stadium	87,411	Malaysia	Kuala Lumpur
10	Borg El Arab Stadium	86,000	Egypt	Alexandria
11	Salt Lake Stadium	85,000	India	Kolkata
12	ANZ Stadium	84,000	Australia	Sydney

https://www.worldatlas.com/articles/the-largest-football-soccer-stadiums-in-the-world.html


nautobot
Organization
Devices
IPAM
Virtualization
Circuits
Power
Secrets
Extensibility
Plugins
Nautobot Football Stadiums
Clone
Edit
Delete
Created March 9, 2022 · Updated 1 week ago
Tenant
Advanced
Change Log
Tenant
Group
None
Description
—
Comments
None
Tags
No tags assigned
Stats
6
Sites
33
Racks
0
Rack reservations
50
Devices
0
VRFs
0
Prefixes
156
IP addresses
64
VLANs
24
Circuits
0
Virtual machines
0
Clusters


>>> network .toCode()
Thank you

Additional Nautobot Resources

>>> Nautobot Resources!

- Network to Code public Slack workspace -
  - http://slack.networktocode.com/
  - **#nautobot**
  - **#nautobot-chat**
  - Ask your questions, we want to help!
- Nautobot public sandboxes
  - https://www.networktocode.com/nautobot/sandbox-environments/
- Nautobot web page
  - https://www.networktocode.com/nautobot/
- *All things Nautobot* playlist at *Network to Code* channel on YouTube

>>> Nautobot Resources

Come join us on the journey and checkout all that Nautobot has to offer.

- GitHub: https://github.com/nautobot/nautobot
- Nautobot Projects & Apps: https://github.com/nautobot
- Slack: #nautobot on slack.networktocode.com
- Docs: https://nautobot.readthedocs.io
- Website: https://www.networktocode.com/nautobot
- Nautobot Sandbox: https://demo.nautobot.com
    - See project README on GitHub for more details using the sandbox environments
- For more information, email info@networktocode.com

>>> Chatbot Resources

Nautobot ChatOps development

- MS Teams Bots: https://docs.microsoft.com/en-us/microsoftteams/platform/bots/bot-features
- MS Graph API: https://docs.microsoft.com/en-us/graph/api/user-get?view=graph-rest-1.0&tabs=http
- Slack Block Kit: https://api.slack.com/reference/block-kit/block-elements
- Mattermost Docs: https://docs.mattermost.com/