The bridge to possible

Speak Truth to DNS

Joe Clarke, Distinguished Engineer

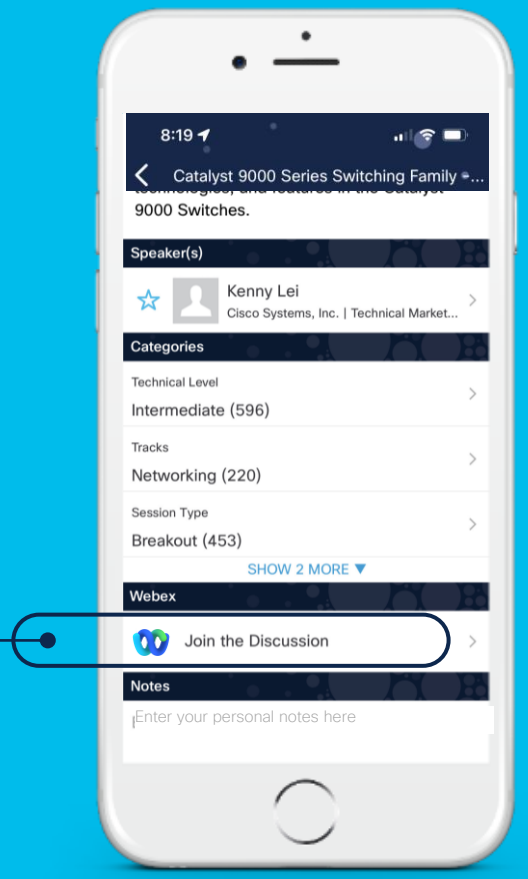
Cisco Webex App

Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space



Webex spaces will be moderated until February 24, 2023.

Abstract

Automation begins with a Single Source of Truth (SSoT) from which all the goodness flows. Once the SSoT is up and running, next up should come DNS. In this session, we'll show how we can automate the powerful REST API of Cisco Prime Network Registrar to seamlessly integrate SSoT and DNS. Once we've taken care of authoritative DNS, we will use the same programmability approach to keep our Umbrella data in sync with our source of truth. Both approaches can be folded into a Continuous Integration approach so that network visibility is always up-to-date and accurate. A demo will be shown that is being used in Cisco's Learning & Certification organization for managing the DNS responsible for the labs used in our training courses.

Abstract

Automation begins with a ~~Single~~ SourceS of Truth (SSoT) from which all the goodness flows. Once the SSoT is up and running, next up should come DNS. In this session, we'll show how we can automate the powerful REST API of Cisco Prime Network Registrar to seamlessly integrate SSoT and DNS. Once we've taken care of authoritative DNS, we will use the same programmability approach to keep our Umbrella data in sync with our source of truth. Both approaches can be folded into a Continuous Integration approach so that network visibility is always up-to-date and accurate. A demo will be shown that is being used in Cisco's Learning & Certification organization for managing the DNS responsible for the labs used in our training courses.

What Is Truth?

What Is Network Truth?

Authoritative

- Physical and virtual Inventory
- Addressing resources

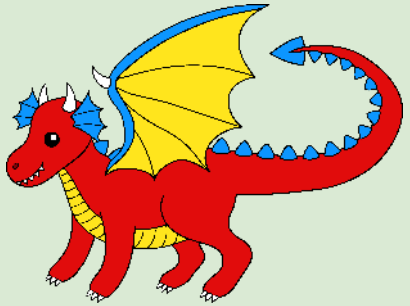
Extensible

- Flexible on organization structure
- Customizable data and metadata elements

Programmable

- Connect to multiple systems
- Cascade truth where it is needed

What Is Network Truth?

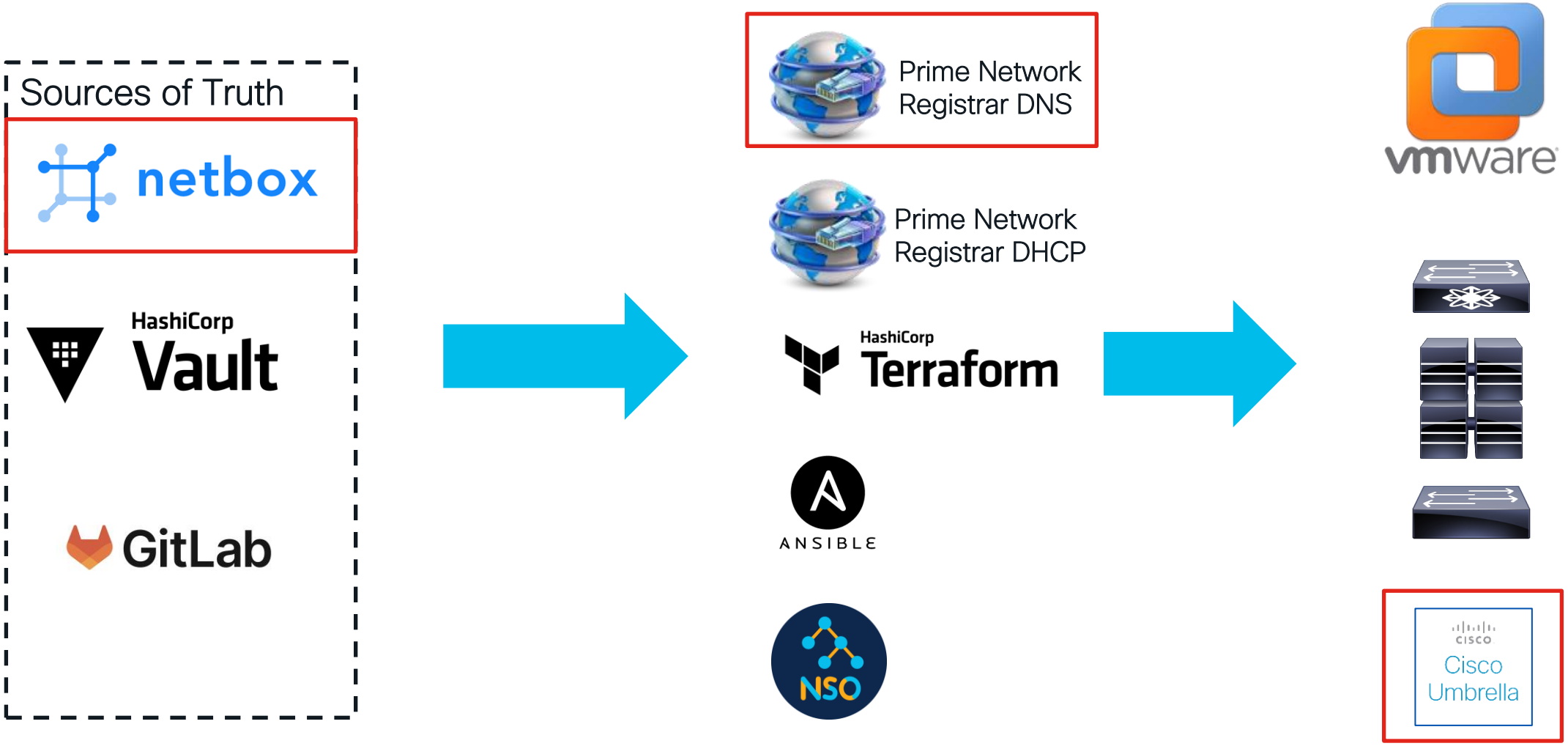
Dragon	Elf	Unicorn
		

Agenda

- Introduction
  - NetBox Overview
  - Prime Network Registrar Overview
  - Umbrella Overview
- Handling Authoritative DNS
- Making Umbrella Insights Nicer
- Wrap Up

Introduction

Learning & Certifications Data Centre Services



Build a Clear Naming Convention

- What gets named?
  - Locations
  - **Devices**
  - **Virtual Machines**
  - Storage
  - Network Objects
  - **Services**
  - **Domains**
- How are names constructed?
  - Include location, purpose, instance

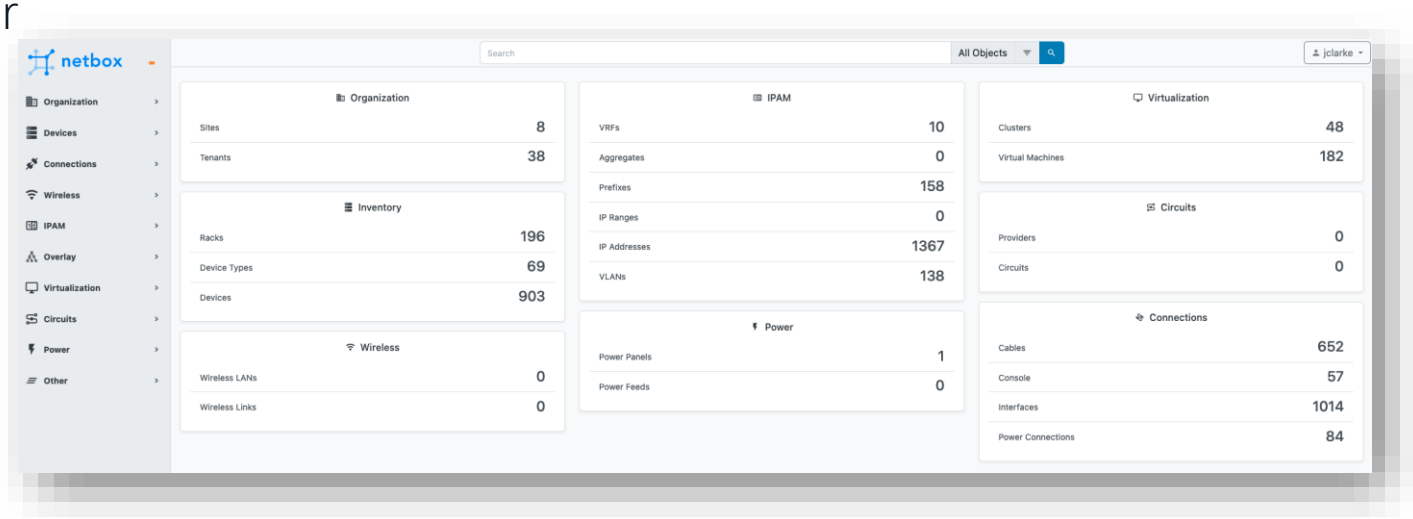
- Learning & Certifications example – Devices/VMs
  - [LOC-NAZ*-TYPE-PURPOSE-INSTANCE]
  - tst01-z0-vm-logging-01
  - sjc17-z0-vm-authdns-01
  - rcn03-z1-sw-tor03-02
- Domains
  - test01.infra.cll.cloud : Active Directory
  - systems.cll.cloud : Production systems
  - shared-pre01.cll.cloud : Pre-production shared services

* NAZ: Network Availability Zone

NetBox Overview

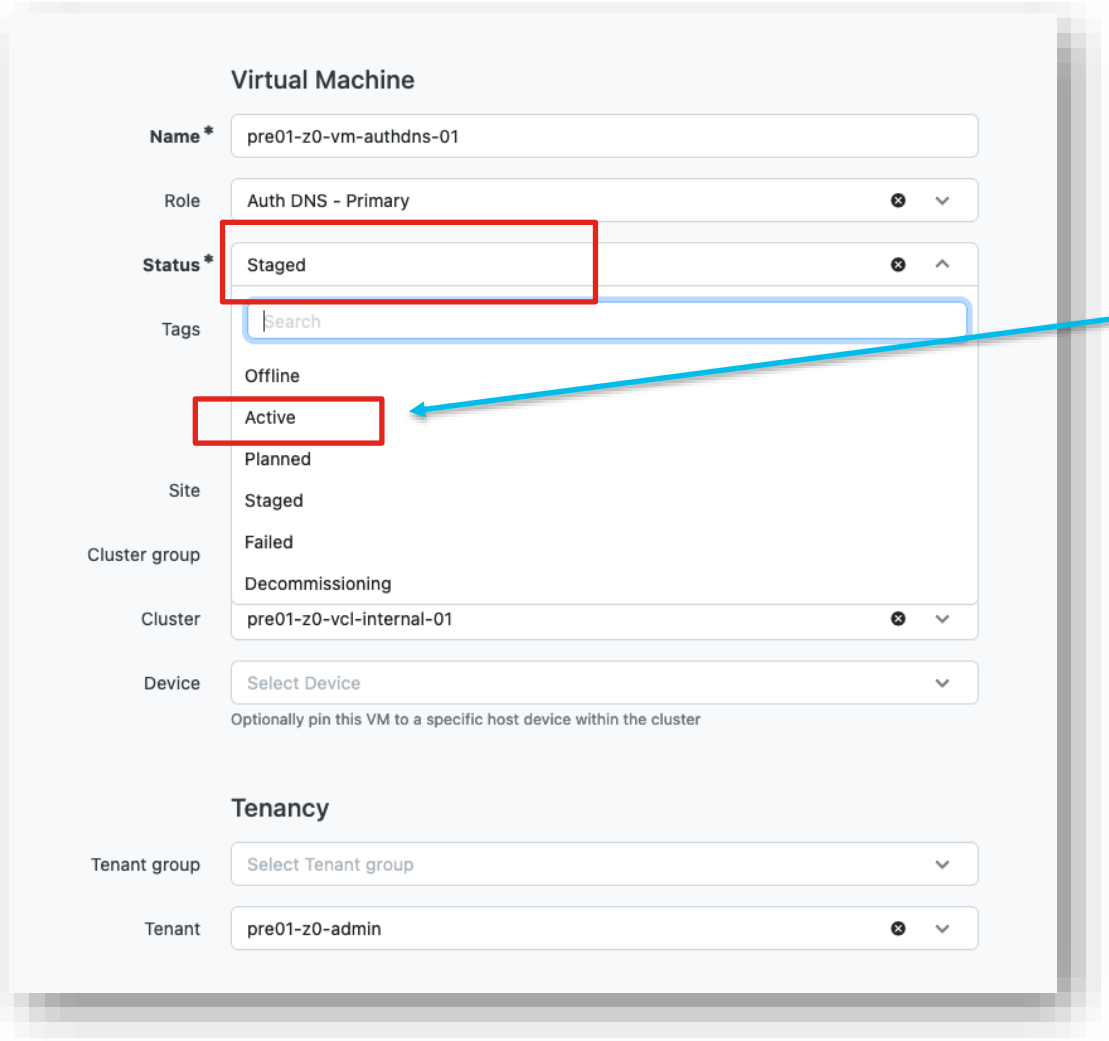
Truth Spoken Here

- Open Source, web-based
- Provides source of truth and documentation for common IT elements
  - Datacenter Infrastructure Management (DCIM)
  - IP Address Management (IPAM)
  - Virtualization inventory
  - Interconnection map
- Highly programmable with rich REST and GraphQL APIs



https://github.com/netbox-community/netbox

Supports Lifecycle Management



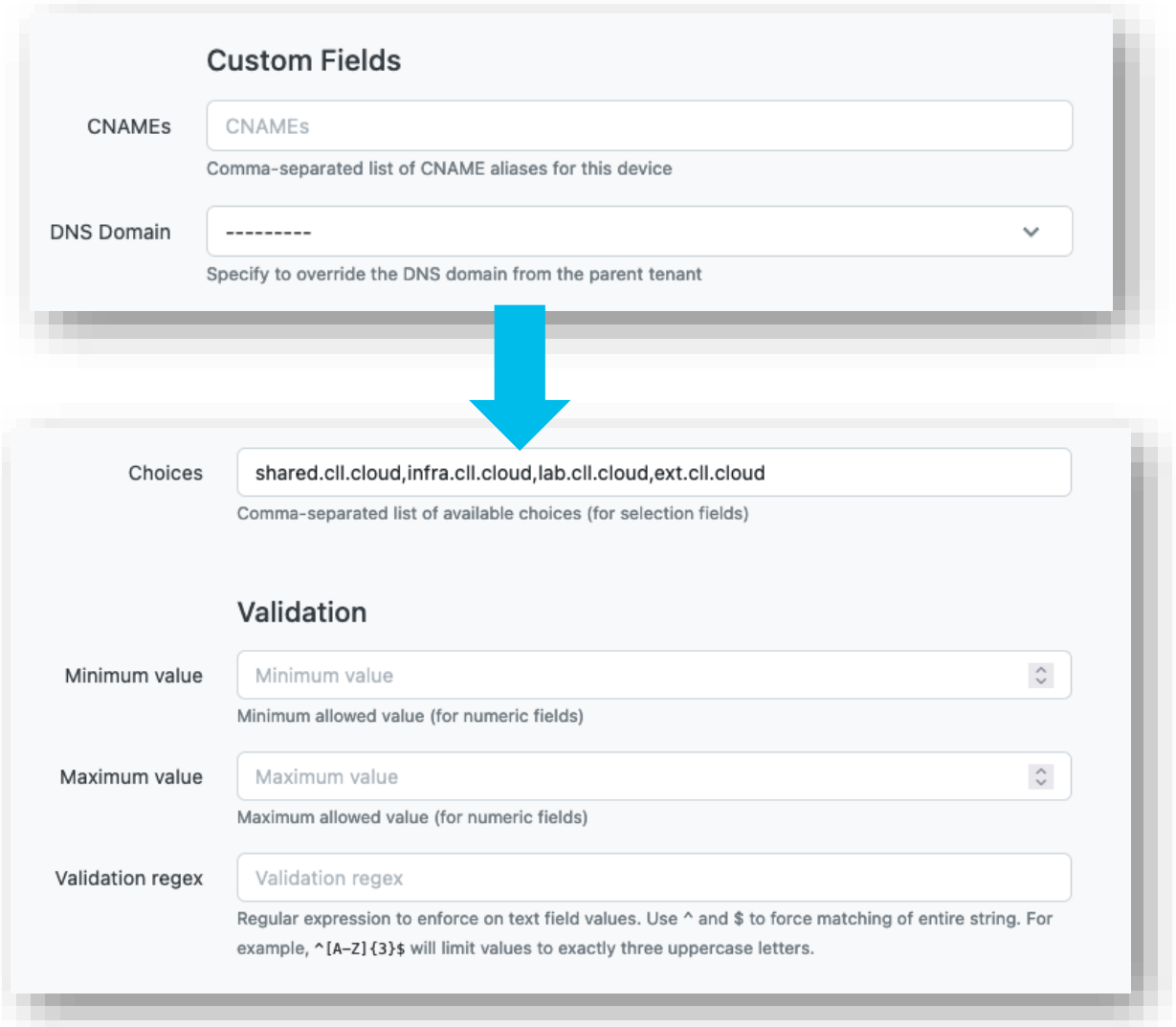
HashiCorp Terraform

ANSIBLE

ZABBIX

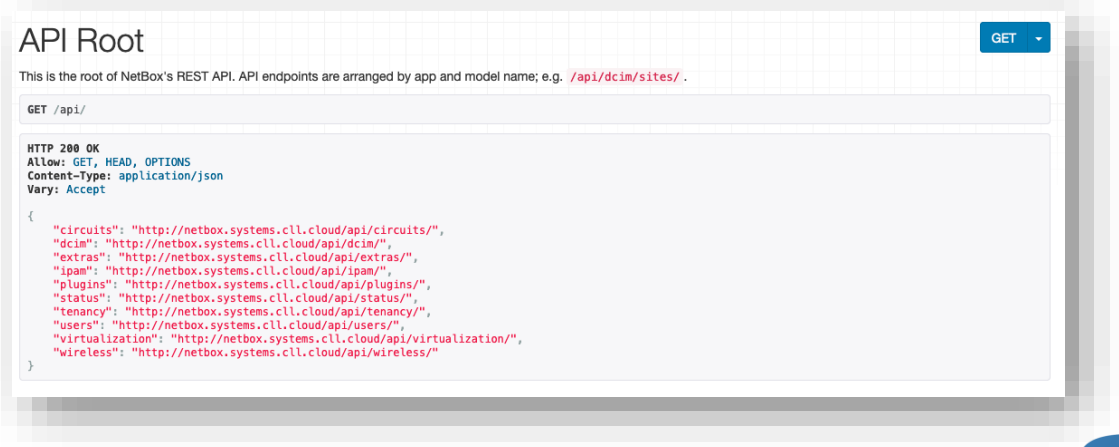
Provide Your Own Metadata

- Extensible with custom fields
- Support for types (text, integer, Boolean, date)
- Can use custom validation logic, including making them mandatory
- Searchable and reportable via the API



Workflows Driven With Automation

- REST API covering all aspects of the UI
- Support for webhooks to trigger workflows based on events
- An intuitive Python library is also provided for easier automation
  - *Extended for our use case*




```
vm_obj = enb.virtualization.virtual_machines.create(
name=name.lower(), platform=platform_obj.id, vcpus=vm["cpu"],
disk=vm["disk"], memory=vm["mem"], cluster=cluster_obj.id
)
vm["vm_obj"] = vm_obj

vm_intf =
enb.virtualization.interfaces.create(virtual_machine=vm_obj.id,
name=mgmt_intf)

ip_obj.assigned_object_id = vm_intf.id
ip_obj.assigned_object_type = "virtualization.vminterface"
ip_obj.save()
```

https://github.com/CiscoLearning/ciscolive-brkops-2039/tree/main/elemental-utils

Prime Network Registrar Overview

Putting The Authority in DNS

- Incredibly scalable DNS and DHCP solution
  - Though has a small initial footprint
- Designed for use in large cable networks, yet simple to setup and use
  - Web-based graphical interface
  - Wizard-driven approach to setting up DNS (and DHCP)
  - Multiple levels of options (Basic, Advanced, Expert)
- Support for simple-to-use high availability
- **Programmable(!)** via REST API

Deploying CPNR

- For maximum scalability, it is typically deployed in multiple tiers
- Tiers can be combined for ease of maintenance in smaller deployments
- HA provides resilience and simplifies management

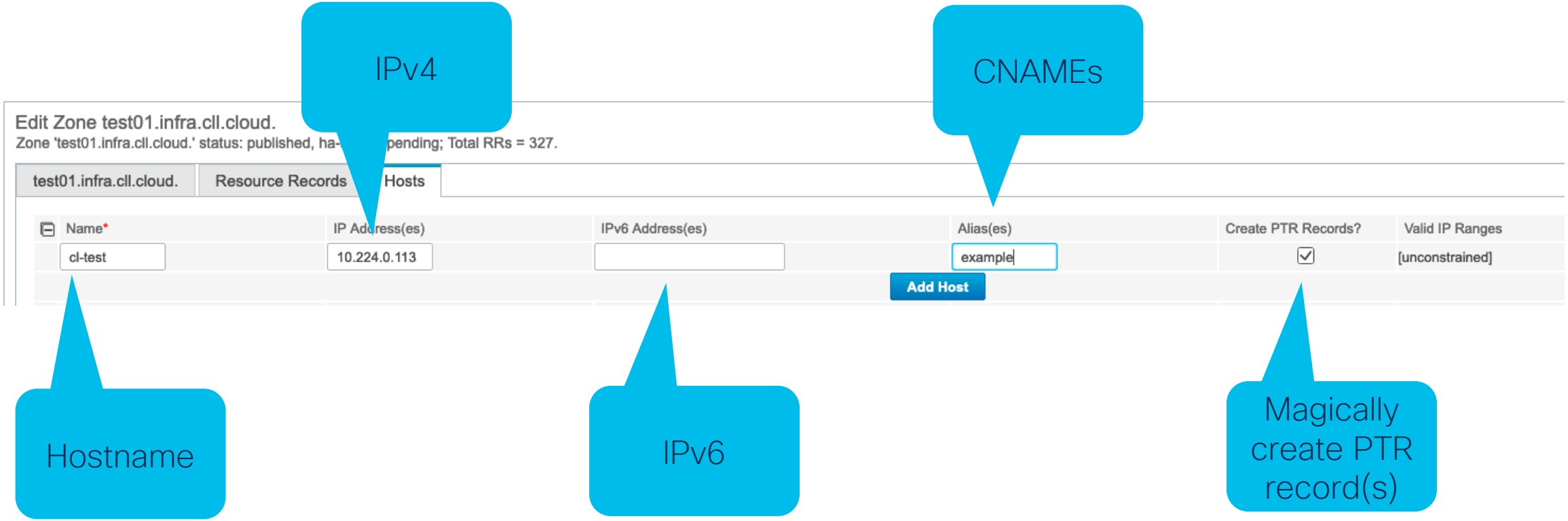
Regional server:
- Licensing
- Manager of managers


Authoritative DNS Layer
- Typically HA pair


Caching DNS Layer
- Recursive resolvers


DHCP Layer

Adding a Host (A and AAAA)



Comprehensive Resource Record Support

Edit Zone test01.infra.cll.cloud.
Zone 'test01.infra.cll.cloud.' status: published, ha-server-pending; Total RRs = 329.

Commands Edit RRs Delete RRs

test01.infra.cll.cloud. | Resource Records | Hosts

Name	TTL	Type	Data
	-1		

Record

Name

all

lear Filter

Name	TTL	Data
@	[24h]	ns01.test01.infra.cll.cloud. jclarke.cisco.com. 2021299421 10800 3600 604800 600
	[24h]	ns01.test01.infra.cll.cloud.
	[24h]	ns02.test01.infra.cll.cloud.
	[24h]	10.224.0.11
	[24h]	ns01.test01.infra.cll.cloud.
	[24h]	ns02.test01.infra.cll.cloud.
	10m	0 100 3268 tst01-z0-ad-01.test01.infra.cll.cloud.
	10m	0 100 88 tst01-z0-ad-01.test01.infra.cll.cloud.
_ldap._tcp.Default-First-Site-Name._sites	10m	0 100 389 tst01-z0-ad-01.test01.infra.cll.cloud.
_gc._tcp	10m	0 100 3268 tst01-z0-ad-01.test01.infra.cll.cloud.
_kerberos._tcp	10m	0 100 88 tst01-z0-ad-01.test01.infra.cll.cloud.
_kpasswd._tcp	10m	0 100 464 tst01-z0-ad-01.test01.infra.cll.cloud.
_ldap._tcp	10m	0 100 389 tst01-z0-ad-01.test01.infra.cll.cloud.
_kerberos._udp	10m	0 100 88 tst01-z0-ad-01.test01.infra.cll.cloud.
_kpasswd._udp	10m	0 100 464 tst01-z0-ad-01.test01.infra.cll.cloud.
adeansible-01	[24h]	tst01-z0-vm-adeansible-01.test01.infra.cll.cloud.
adeansible-02	[24h]	tst01-z0-vm-adeansible-02.test01.infra.cll.cloud.
adeansible-03	[24h]	tst01-z0-vm-adeansible-03.test01.infra.cll.cloud.
dev11-z0-vm-vcenter-01	[24h]	10.224.130.128
	[24h]	"v=_netbox url=http://netbox.systems.cll.cloud/api/virtualization/virtual-machines/147/ type=vm id=147 ip_id=860"
dev11-z1-vm-vcenter-01	[24h]	10.224.130.129
	[24h]	"v=_netbox url=http://netbox.systems.cll.cloud/api/virtualization/virtual-machines/144/ type=vm id=144 ip_id=844"
dev11-z2-vm-vcenter-01	[24h]	10.224.130.130
	[24h]	"v=_netbox url=http://netbox.systems.cll.cloud/api/virtualization/virtual-machines/145/ type=vm id=145 ip_id=845"
DomainDnsZones	10m	10.224.0.11
_ldap._tcp.Default-First-Site-Name._sites.DomainDnsZones	10m	0 100 389 tst01-z0-ad-01.test01.infra.cll.cloud.
_ldap._tcp.DomainDnsZones	10m	0 100 389 tst01-z0-ad-01.test01.infra.cll.cloud.

A
A6
AAAA
AFSDB
ANY
AXFR
CAA
CDNSKEY
CDS
CNAME
DHCID
DNSKEY
DS
GID
HINFO
ISDN
IXFR
MB
MD
MF
MG
MINFO
MR
MX
NAPTR
NS
NSAP
NSEC
NSEC3
NSEC3PARAM
OPT
PTR
RP
RRSIG
RT
SPF
SRV
TKEY
TSIG

Follows BIND-like syntax with a graphical frontend

Using the API

- REST API is powerful, but not well-documented ☹
- Support for adding, deleting, an updating
  - Zones (domains)
  - Resource records
  - Hosts
- Ability to perform limited commands
  - Sync HA pairs
  - Restart services
- I created a Python wrapper that is similar in style to NetBox
- https://github.com/CiscoLearning/ciscolive-brkops-2039/tree/main/elemental-utils

```
cpnr_record["name"] = record.hostname
cpnr_record["addrs"] = {"stringItem":
[record.ip]}
cpnr_record["zoneOrigin"] =
primary_domain
cpnr_record["createPtrRecords"] = True

edns.host.add(**cpnr_record)
```

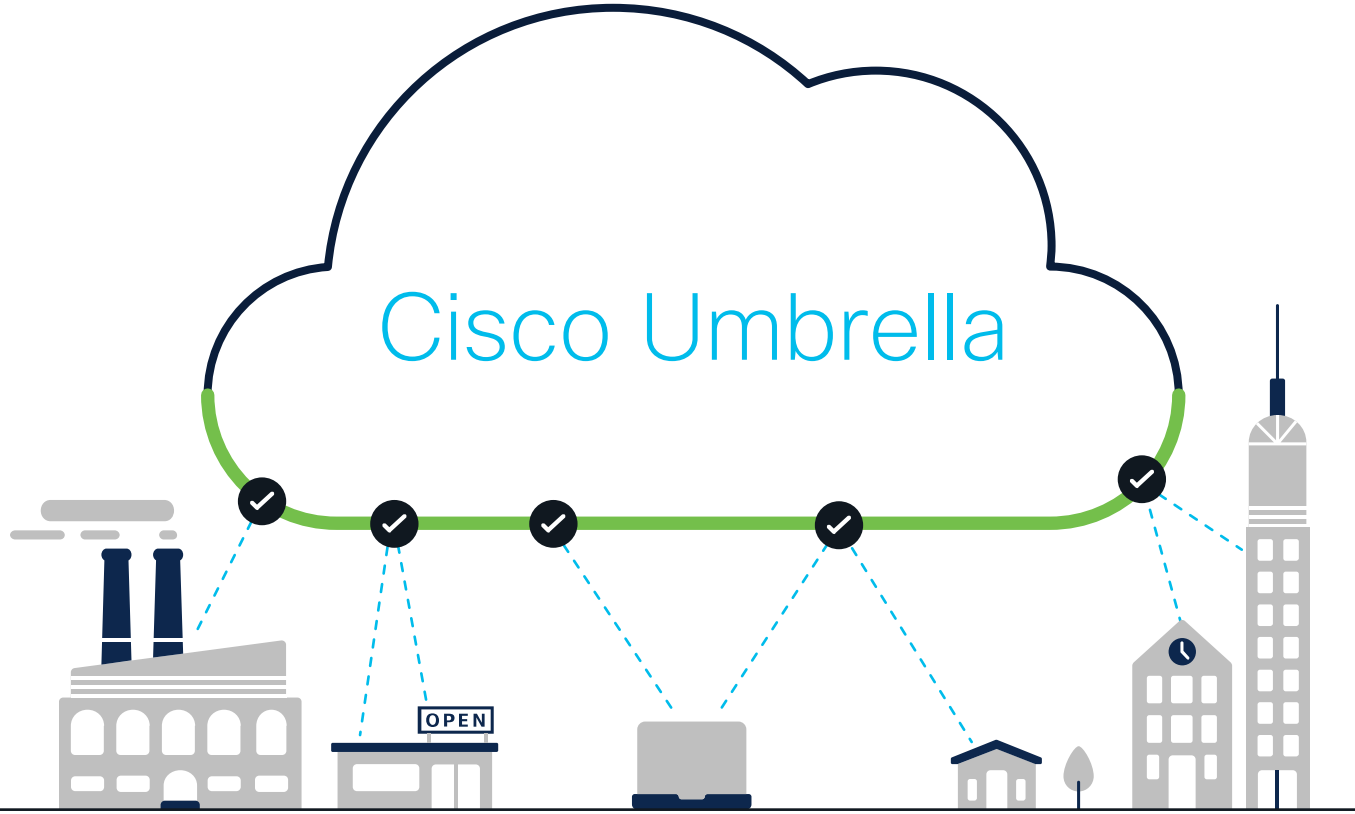
```
cpnr_record["name"] = record.alias
cpnr_record["zoneOrigin"] =
record.domain
target =
f"{record.host.hostname}.{record.host.
domain}"
cpnr_record["rrList"] = {"CCMRRItem":
[{"rdata": target, "rrClass": "IN",
"rrType": "CNAME"}]}

curr_edns.rrset.add(**cpnr_record)
```

Cisco Umbrella

Umbrella DNS-layer security

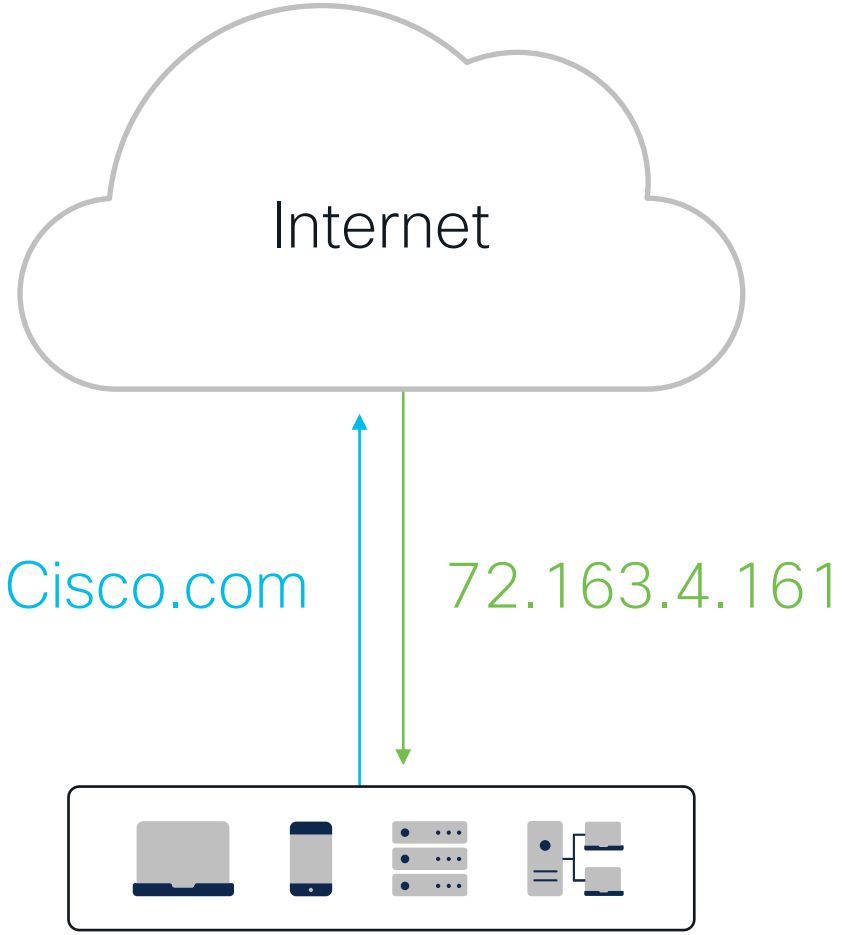
Visibility and protection for all activity, anywhere

- See all internet traffic
- Block attacks earlier
- Contain malware if already inside
- Easily enforce content web filtering
- Discover, manage or block cloud apps
- Gain context for faster investigation



Why is DNS useful for security?

- First step in connecting to the internet
- Precedes file execution and IP connection
- Used by nearly all devices



Built into the internet foundation

Destinations

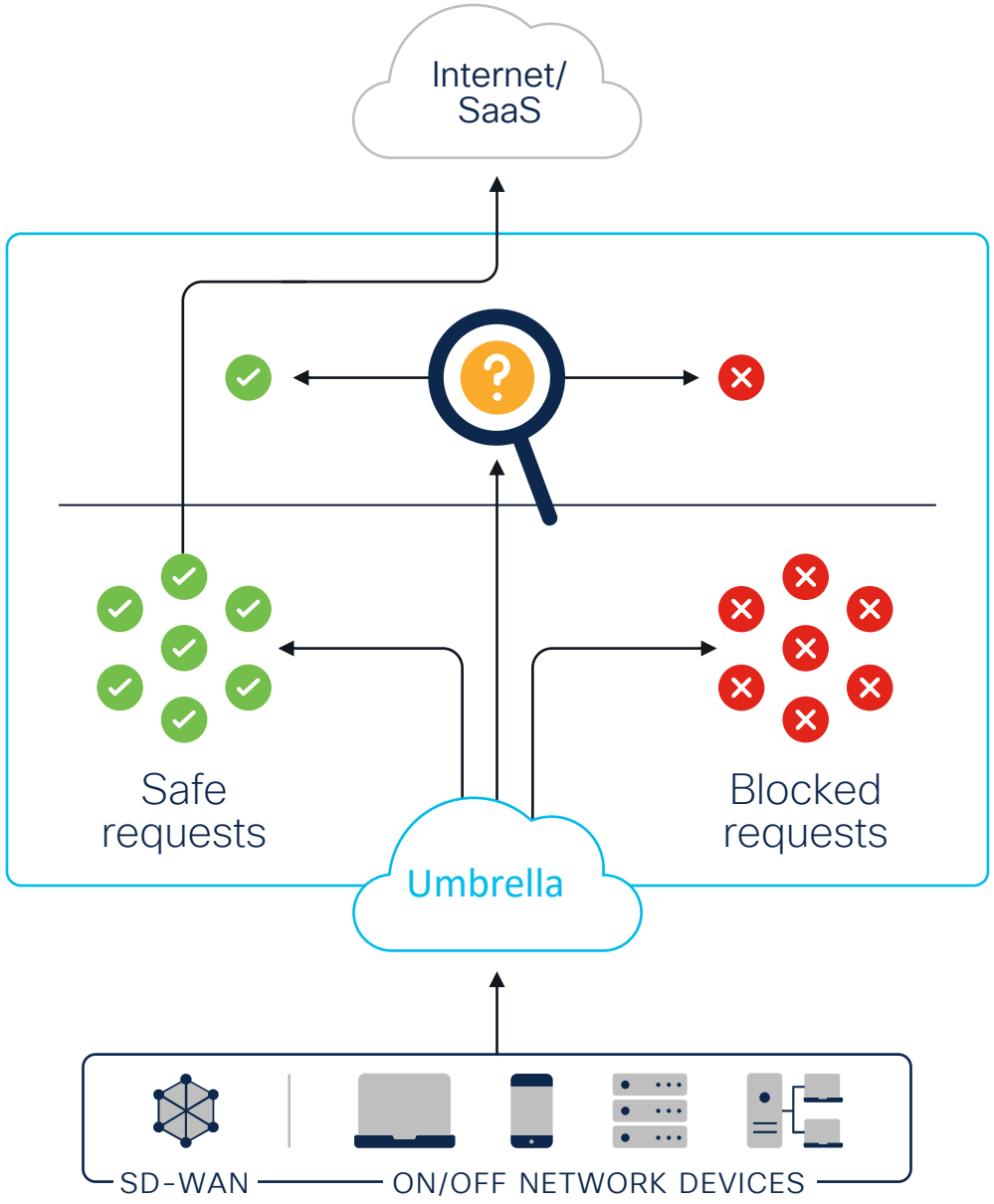
Original destination or block page

Security controls

- DNS enforcement
- Risky domain inspection through proxy
- SSL decryption available

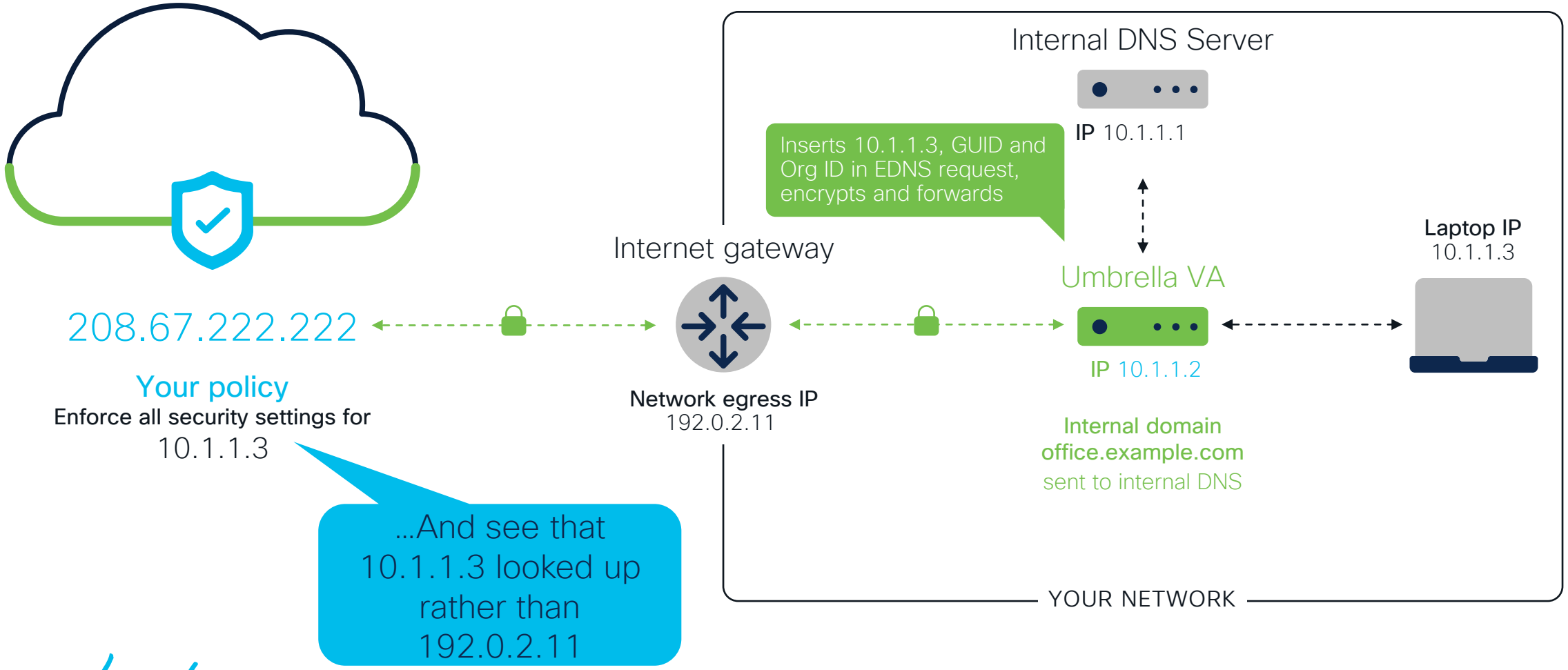
Internet traffic

On and off-network



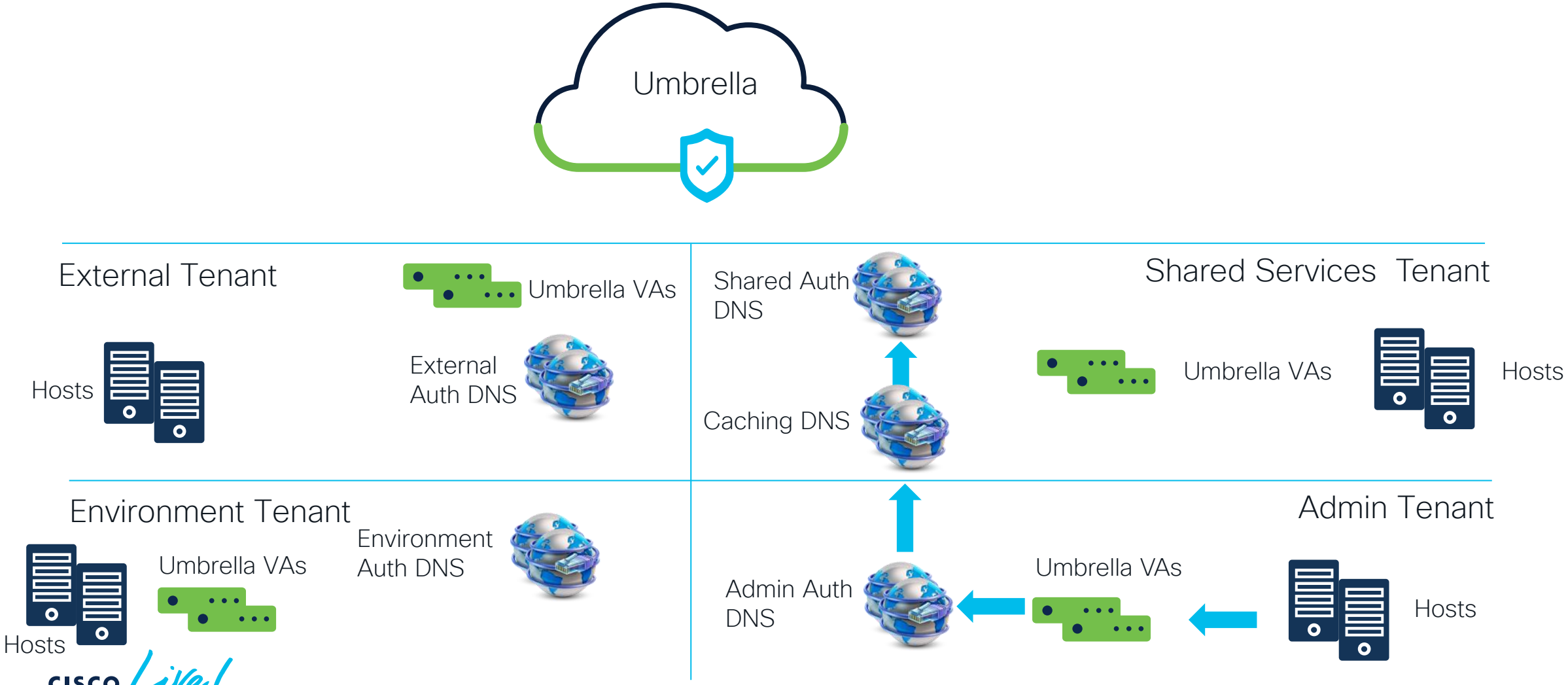
Protect internal networks

Via Umbrella virtual appliance

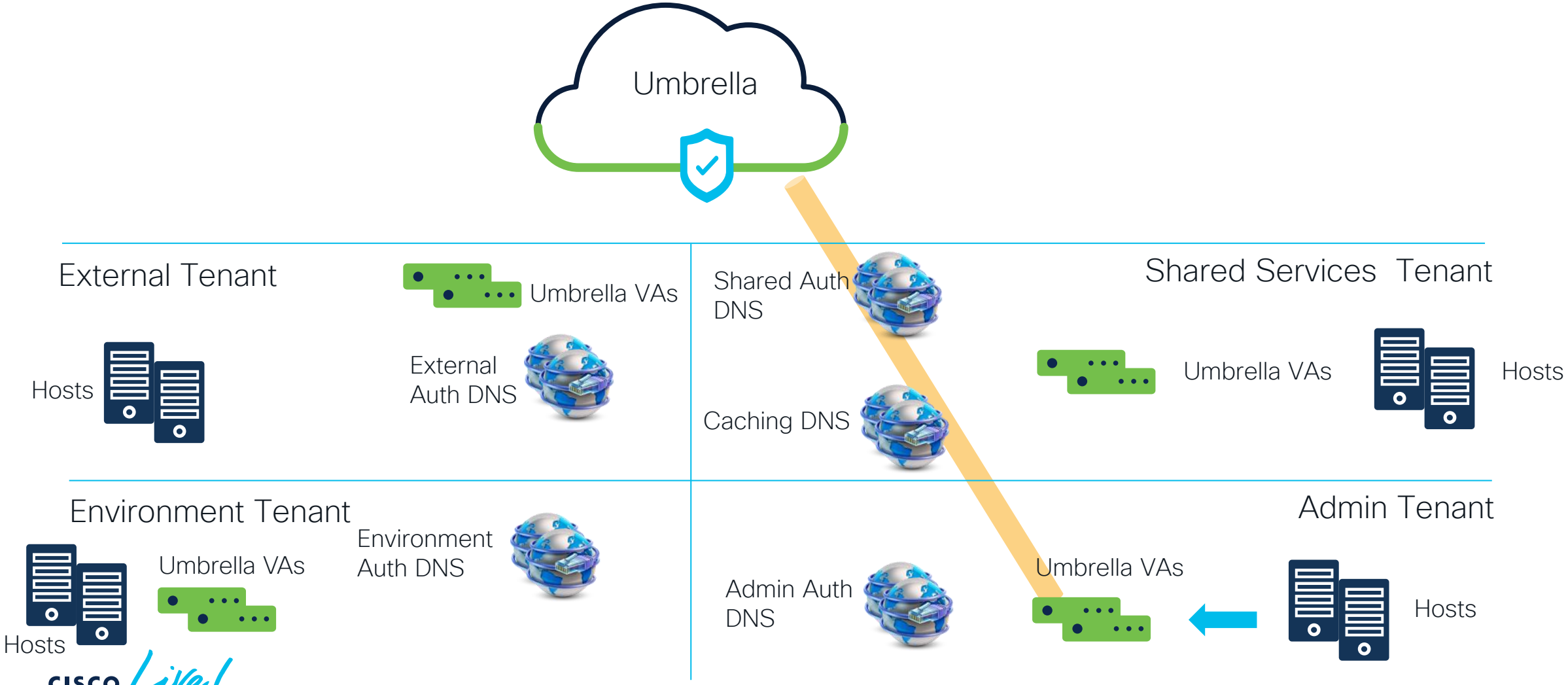


Handling Authoritative DNS

Learning & Certifications DNS Flow



Learning & Certifications DNS Flow



L&C DNS Design Considerations

- CPNR assumes the DNS responsibilities from Active Directory
- All authoritative DNS servers are distributed in HA pairs
  - Changes performed to primary server and synced to secondary via API

High-Level NetBox to CPNR Flow

Event

1. Device or Virtual Machine added to NetBox
2. Primary management address assigned to device or VM

Main Workflow

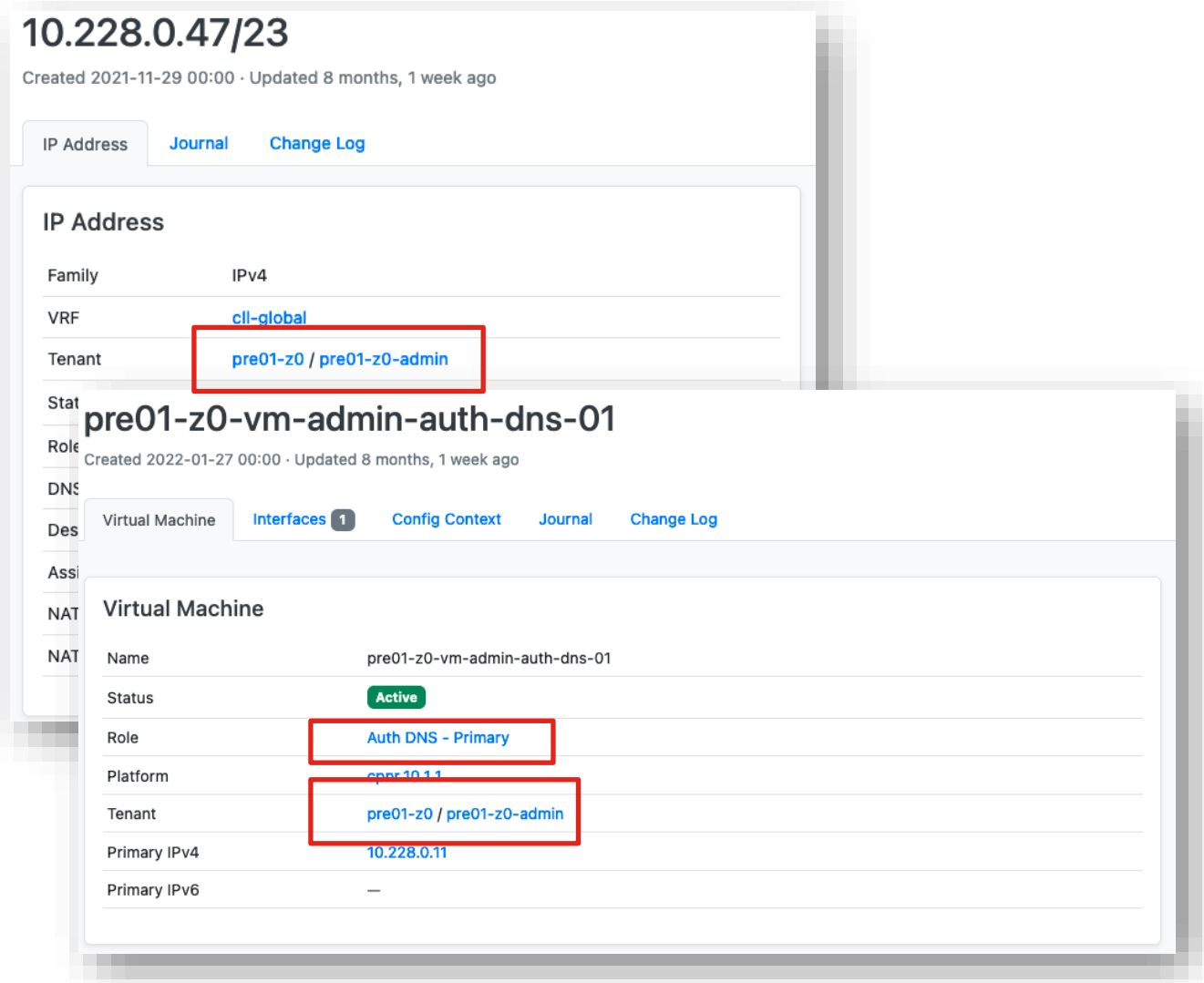
3. *Timed job kicks off*
4. DNS A, PTR, and optional CNAME records added to proper auth CPNR instance
   a. Incorrect records removed
   b. Correct records added
5. Sync primary to secondary
6. Flush the cache servers

Clean up Workflow

1. *Timed job kicks off*
2. CPNR records that are **not** associated with NetBox entries are removed
3. Sync primary to secondary
4. Flush the cache servers

Finding the Right Authoritative Server

- Each element (e.g., device or VM) in NetBox is associated with a tenant
- A Role is assigned to the auth DNS VMs
- A similar role is assigned to the caching DNS servers



Another Source of Truth

- Some aspects of the DNS config didn't feel *right* to put in NetBox
  - Root domain of our global data centre
  - Overall admin contact
  - Zones/domains that have special treatment
- Chose to use git for this metadata for now (YAML file in git)
  - Retrievable via raw URL link

```yaml
root_domain: cll.cloud # The ultimate root domain name.
# The servers that are authoritative for the root_domain.
# We list them here as they are not [yet] in NetBox, and we
need a simple way to identify them.
root_servers:
  - 172.17.0.253 # SJC root server
  - 172.27.0.253 # RTP root server
ns1_name: ns01 # Logical name of the primary authoritative
DNS
ns2_name: ns02 # Logical name of the secondary authoritative
DNS
admin_contact: jclarke@cisco.com # Email address of the
administrative contact for DNS
ad_prefixes:
  # List of prefixes that have AD semantics. These domains
will get additional treatment to support AD.
  - infra
```

How To Connect NetBox To CPNR

- Python is our chosen *lingua franca*
- Two Python scripts were created for each workflow
- Ideally, webhooks would trigger the first workflow to run, but we have a composite event

Scripts at https://github.com/CiscoLearning/ciscolive-brkops-2039/tree/main/elemental-scripts/cpnr

Adding New Entries

1. Get a list of all tenants for a given site (e.g., tst01, pre01, prod)
2. Get all the assigned IP addresses for each tenant
3. Determine a list of DNS records to add, update, delete (update is a delete then create)
4. Process records (allows for a dry-run test)
   - Deletions first
   - Then creations

```
# Get the current A record from DNS (if it exists)
current_host_record = edns.host.get(dns_name,
zoneOrigin=primary_domain)
# Get the current PTR record from DNS (if it
exists)
current_ptr_record = edns.rrset.get(ptr_name,
zoneOrigin=rzone_name)
```

- Change needed?
  - An A (host) record doesn't exist
  - A record exists, but pointing to a different IP
  - *Check if it's a NetBox entry*
  - PTR exists but pointing to a different host

How To Tie a CPNR Record To NetBox?

- In other systems, we provide NetBox *context* metadata with entries (e.g., in NSO we use the device context field)
- CPNR doesn't have a specific place for metadata
- DNS provides a TXT record where general character data can be stored

```
$ host -t TXT pre01-z0-vm-ad-01.pre01.infra.cll.cloud

… "v=_netbox url=http://netbox.systems.cll.cloud/api/virtualization/virtual-machines/170/ type=vm id=170 ip_id=1016"
```

Preamble (it is from NetBox)
URL to the specific NetBox entry
Type of NetBox entry
NetBox ID
ID of the associated IP address

Deleting Stale Entries

- Main workflow script removes stale *assigned* records from CPNR,
- It cannot find those that are no longer assigned
- A second script
  1. Gets all tenants for a site
  2. Gets all host records and all resource records from CPNR
  3. Finds the stale A and PTR records by looking at the TXT records
  4. Finds stale CNAMEs
  5. Deletes all stale records from CPNR

```
found_txt = None
for rr in host_rr.rrList["CCMRRItem"]:
    if rr["rrType"] == "TXT" and
(rr["rdata"].startswith('"v=_netbox') or
rr["rdata"].startswith('"v=_static')):
        found_txt = rr["rdata"]
        break
```

- If no TXT record, the resource name is stale
- If the NetBox IP ID is not assigned to the resource name, then the resource is stale

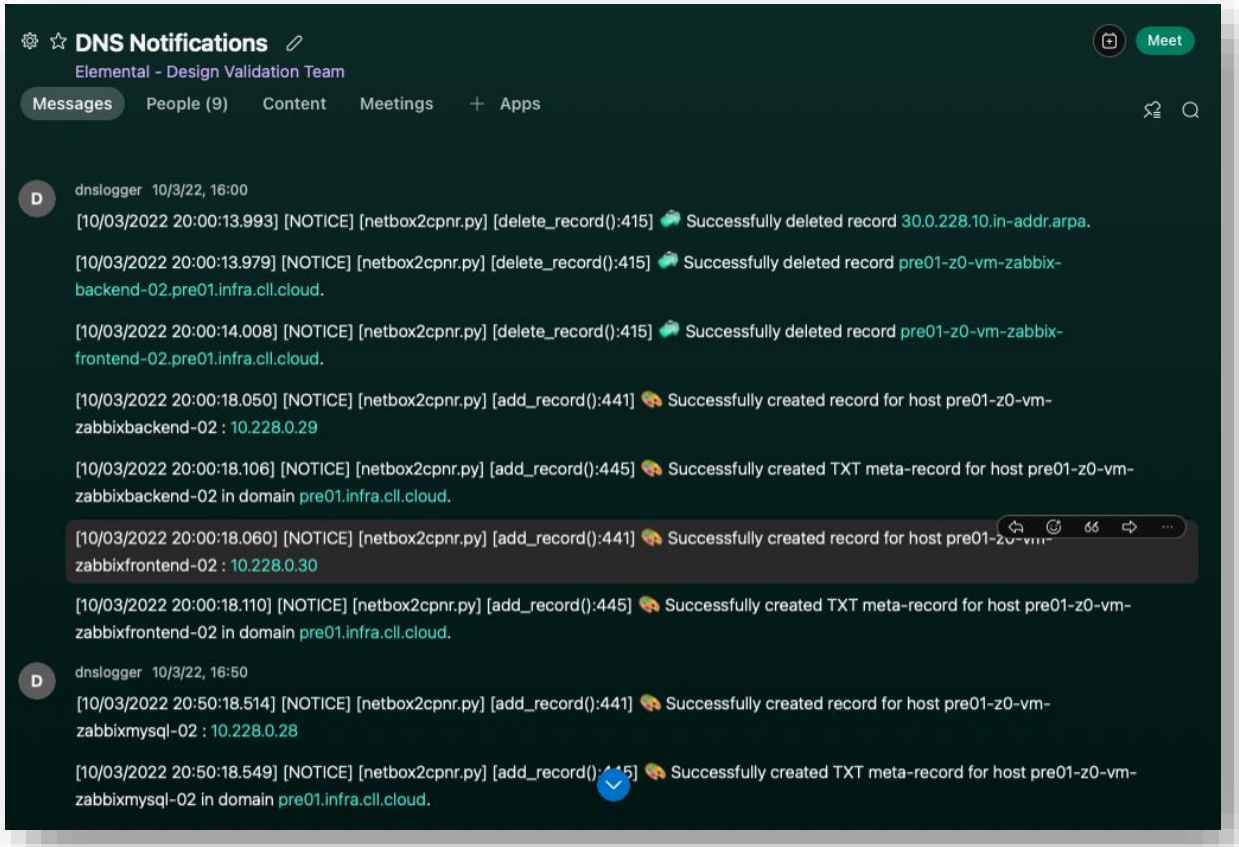
Scaling the Workflows

- Over time, NetBox (and CPNR) records grow
- We want the scripts to run in a reasonable amount of time (typically under five minutes)
- *Threadpools* are used to parallelize the work

```
with
concurrent.futures.ThreadPoolExecutor(max_workers=worke
rs) as executor:
    future_task = {executor.submit(task, item, *args):
item for item in iterator}
    for ft in
concurrent.futures.as_completed(future_task):
        item = future_task[ft]
        try:
            ft.result()
        except Exception as e:
            if not name_attribute:
                logger.exception(f"⛔ Failed to
{task_name} for {item}: {e}")
            else:
                logger.exception(f"⛔ Failed to
{task_name} for {getattr(item, name_attribute)}: {e}")
            result = False
            if stop_on_error:
                break
```

Letting Others Know What Was Done

- Logging is critical in order to have a record of the work done
- Don't automate in the blind (https://blogs.cisco.com/developer/avoidingsilentautomation01) 😉
- Since DNS record creation isn't real-time, created a chat-ops integration for Webex using the same Python logging system
  - pip install webex-handler

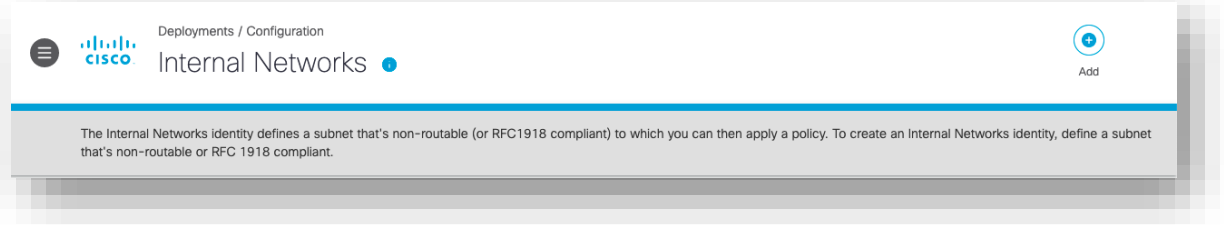


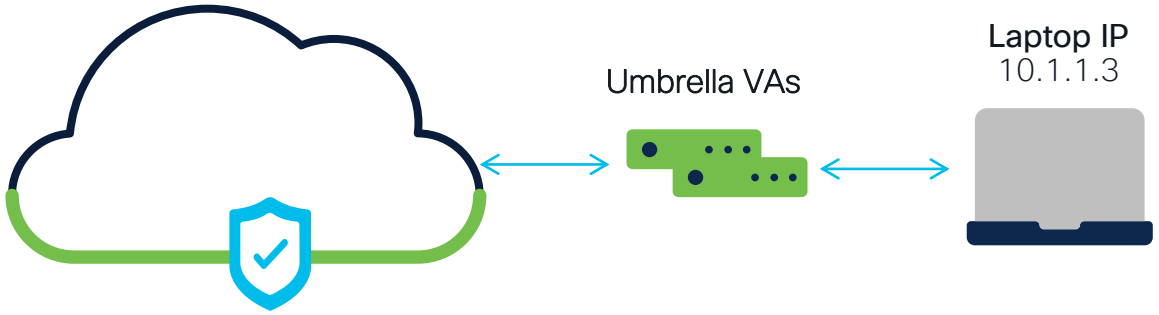
Making Umbrella Insights Nicer

cisco Live!

End To End Visibility of DNS Queries

- Umbrella allows definition of internal networks to provide finer-grain control over non-routable subnets
- Umbrella learns about queries from internal networks via the virtual appliances
- Even if all your networks will have the same policy, defining internal networks gives better visibility





Applying Our Naming Convention to Umbrella

- Umbrella has a few requirements for internal network names
  - Must be unique
  - Maximum of 50 characters
- Solution for Learning & Certifications
  - Truncate
  - Check for uniqueness
  - Add numeric suffix if needed
  - Truncate again

Keep the name under 50 characters

```
suffix = None
net_name = name
while True:
    net_name = truncate_network_name(name,
suffix)
    if not any(d["name"] == net_name for d in
int_networks):
        break

    if not suffix:
        suffix = 1
    else:
        suffix += 1
```

L&C's Internal Networks

Network name: [VLAN_NAME] : [SITE_NAME]

Default [NetBox] VRF: Umbrella Site corresponds to NetBox tenant

vcsa-ha : pre01-z0-private	pre01-z0-private	10.228.98.0/29	Default Policy	•••
VLAN101 : rtp07-c0-dcmds-eg00-core	rtp07-c0-dcmds-eg00-core	192.168.101.0/24	Default Policy	•••

Example of student lab environment

Non-default [NetBox] VRF: Umbrella Site corresponds to VRF name

Searching for Sites and Networks

Reporting / Core Reports

Activity Search

Schedule | Export CSV | LAST 24 HOURS

FILTERS | Search by domain, identity, or URL | Advanced | CLEAR | Customize Columns | All

IDENTITY tst01-z0-admin ×

Search filters

Response — Select All
- Allowed Advanced
- Blocked
- Selectively Proxied

Warn Page Behavior — Select All
- Warned
- Accessed After Warn

44,701 Total — Viewing activity from Jan 23, 2023 9:49 PM to Jan 24, 2023 9:49 PM

Results per page: 50 — 1 - 50

Request	Identity	Policy or Ruleset Identity	Destination	Destination IP	Internal IP
DNS	vmware-mgmt : tst01-z0-admin	vmware-mgmt : tst01-z0-admin	assets.contentstack.io		10.224.130.12
DNS	vmware-mgmt : tst01-z0-admin	vmware-mgmt : tst01-z0-admin	assets.contentstack.io		10.224.130.12
DNS	vmware-mgmt : tst01-z0-admin	vmware-mgmt : tst01-z0-admin	assets.contentstack.io		10.224.130.12
DNS	vmware-mgmt : tst01-z0-admin	vmware-mgmt : tst01-z0-admin	assets.contentstack.io		10.224.130.12

Getting the Code

- Like the CPNR sync, Umbrella sync is scheduled
- Could be done on webhook, but new tenants and environments aren't created often
- Python script makes use of a simple wrapper around the Umbrella REST API

Scripts at https://github.com/CiscoLearning/ciscolive-brkops-2039/tree/main/elemental-scripts/netbox

Putting It All Together

- Rather than use pure cron jobs, scheduled tasks were added to git
- Using GitLab-CI, tasks for each site were created
- DNS sync currently runs every 5 minutes with a cleanup once per day
- Umbrella sync runs once per day

.gitlab-ci.yml

```yaml
sync_to_umbrella:
  stage: sync
  script:
    - netbox/netbox-to-umbrella.py
  rules:
    - if: '$CI_PIPELINE_SOURCE == "schedule" && $SYNC_TASK == "umbrella"'
      when: on_success
    - when: manual

sync_to_cpnr_tst01:
  stage: sync
  script:
    - |
      cpnr/netbox2cpnr.py --site tst01
      cpnr/netbox2cpnr.py --site tst02
  rules:
    - if: '$CI_PIPELINE_SOURCE == "schedule" && $SYNC_TASK == "cpnr" && $SYNC_ACTION == "sync" && $SYNC_TARGET == "tst01"'
      when: on_success
    - when: manual
```

jclarke › elemental-scripts › **Schedules**

All 5 | Active 5 | Inactive 0 | New schedule

Description	Target	Last Pipeline	Next Run	Owner
Cleanup CPNR for PRE01	master	#39505	in 3 hours	jclarke
Sync PRE01 to CPNR	master	#39763	in 7 minutes	jclarke
Cleanup CPNR for TST01	master	#39506	in 3 hours	jclarke
Sync TST01 with CPNR	master	#39762	in 7 minutes	jclarke
Periodic Umbrella Sync	master	#39530	in 5 hours	jclarke

GitLab

Wrap Up

Speak Truth to DNS

- Keeping all aspects of DNS up-to-date is critical for proper network operations.
- Cisco DNS products like Umbrella and Cisco Prime Network Registrar provide rich APIs for automating aspects of DNS.
- Automating from Source(s) of Truth means less manual operations and less individual tools that people need to touch.

Complete your Session Survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (open from Thursday) to receive your Cisco Live t-shirt.

- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Session Catalog and clicking the "Attendee Dashboard" at https://www.ciscolive.com/emea/learn/sessions/session-catalog.html

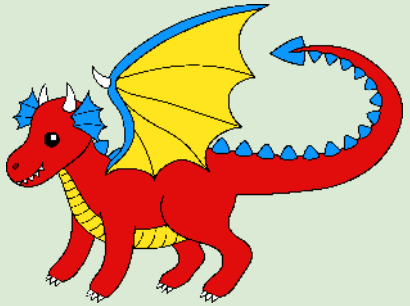
Continue Your Education

Visit the Cisco Showcase for related demos.

Book your one-on-one Meet the Engineer meeting.

Attend any of the related sessions at the DevNet, Capture the Flag, and Walk-in Labs zones.

Visit the On-Demand Library for more sessions at ciscolive.com/on-demand.

Dragon	Elf	Unicorn
		

Your Network Reality

Authoritative	Extensible	Programmable
• Physical and virtual Inventory • Addressing resources	• Flexible on organization structure • Customizable data and metadata elements	• Connect to multiple systems • Cascade truth where it is needed

The bridge to possible

Thank you

cisco Live!

cisco Live!

ALL IN