WHY AUTOMATE?

MANUAL IS...

- Very error prone
  - Human error / typos
  - Copypasta propagation
- Time consuming
  - Customer must wait for a tech
  - The tech must make manual changes
- Hard to change
BENEFITS

- No human errors from the start
- Easy to change network design
  - Database / template driven
- Predefined, tested process for adding hardware
- Gives you something to do while waiting on vendors

AUTOMATE FIRST, CUSTOMERS LATER?
POTENTIAL CONCERNS

● PHP / PERL
  o Not our forte
  o Doesn’t integrate with existing code

● Lots of organic growth

● Larger scope than was needed
IXP MANAGER

- Well-thought-out schema from seasoned industry professionals at INEX
- Teaches, implements, and ensures best practices
- Automates most back-end processes
- Robust customer dashboard
  - Peering Manager
  - Peering Matrix
  - Peer-to-peer traffic graphing
Django integration / shared authentication uses the same database and config file

- **Python interface**
  - Quick and easy to build Python scripts for accessing and changing data

- **Lean**
  - Only overlays IXP Manager
  - New functionality belongs in other modules
  - Additional data as needed stays out of IXP Manager’s way
PROVISIONING

- Access data from DJANGO-IXPMGR
- Templates and Push
  - Servers abuse SSH force command
  - Current switches use netconf
  - Future—easy as template change plus new push command
- Provides
  - Port provisioning
  - ACL provisioning
  - Route server interaction
NEW CUSTOMER SIGNUP

- Imports Data from PeeringDB 2 (if available)
- Populates customer records in IXP Manager
- Sets port to quarantine mode
  - Netconf pushes config to their port on the respective switch
- Manual tasks
  - Check port for unwanted traffic
  - Check for single MAC address
  - Set mac address: manage.py ixpmgr_set_l2db <port> <mac>
PORT GOES LIVE

- IXP Manager provisions
  - Route server configurations
  - AS112
  - Graphing, etc.

- Our software provisions ACLs across all switches

- Provision the port
  - Allows access to public exchange fabric
L2 ACL with source or destination prefix
- No peer interaction required
- Only filters traffic going to requester’s port(s)

Filters traffic at VLAN ingress
- Keeps the entire fabric clean
- Scales well

Simple, well-known BGP communities
- Very easy to implement
- Most networks already do it for transit
- Does not require peering over the route servers
HOW IT WORKS

● To add:
  o Customer sends a BGP community to route server
  o 33713:666 for dest prefix; 33713:999 for src prefix
  o Route server sends a command back to a controller
  o Controller sends ACL to all switches

● To remove:
  o Controller polls all route servers for tagged routes
  o Compares routes from both route servers; removes from ACLs if necessary
IN REAL LIFE

- Any network that does blackholing, seamlessly blackholes traffic over the exchange
- Any DOS going to an exchange IP can be easily filtered at fabric ingress
FUTURE

- API
QUESTIONS / COMMENTS?

matt@unitedix.net
https://github.com/inex/IXP-Manager
https://github.com/20c/django-ixpmgr

SPECIAL THANKS

Barry O'Donovan <barry.odonovan@inex.ie>
Job Snijders <job@instituut.net>
Nick Hilliard <nick@inex.ie>