INEX

IXP Automation

RIPE 75, Dubai, October 2017

Barry O’Donovan

Internet Neutral Exchange Association
Company Limited by Guarantee

IXP AUTOMATION

Background

- Original purpose of IXP Manager was to support route server config builds
- Designed with a structure capable of storing all participant switch configuration tokens
- Reticent about using database for network configuration
    - Cost / return ratio wasn’t right
    - Concerns about how to control configuration deployment
    - Poor tool support for interfacing with network devices

IXP AUTOMATION

Toolchain Problems

- “Traditional” server automation tools could not interface with network devices
- Tools of the era: RANCID, SSH, TFTP, bash + perl scripts
- No framework mechanisms available

IXP AUTOMATION

But now it's 2017

- Multiple automation approaches possible
- Server automation frameworks can interface with network devices
- Network Operating Systems now have APIs and / or API models
- Some NOSs support multiple APIs
- Rationale changes
  - Too much repetitive configuration: "Taking the operator out of operations"
  - Long term cost reduction

IXP AUTOMATION

Phase 1 Operational Goals

- Configure all IXP participant edge ports
  - Speed, dot1q framing, LAG ports, layer 2 filters
- Configure IXP core
  - Interfaces, BGP, VXLAN configuration
- Ready for service to handle INEX LAN1 forklift upgrade to Arista kit in 2017Q1

IXP AUTOMATION

Phase 2 Strategic Goals

- Use initial automation process to learn how to do this properly
- Build functionality into IXP Manager: user interface, database, export presentation
- Ensure that abstraction model is usable across different network devices and different organisations
- Release as open source

IXP AUTOMATION

Approaches

	Openflow	YANG	Vendor API
Abstraction Level	Low	High	Mid Range
Vendor Support	Version Dependent	In Development	Variable
Portability	High	High	Low
Cross-Platform	Low	Currently low	Needs Abstraction
Complexity	High	Mid	Low

IXP AUTOMATION

We Choose a Practical Approach

- YANG: only well supported on tiny number of NOSs
- Openflow: too low level, not loved by chipset manufacturers
- Decided to use NAPALM
  - Integrates with vendor APIs at the network device interface
  - Integrates with Ansible and SaltStack at control + provisioning DB interface
  - Long term support is likely to be good

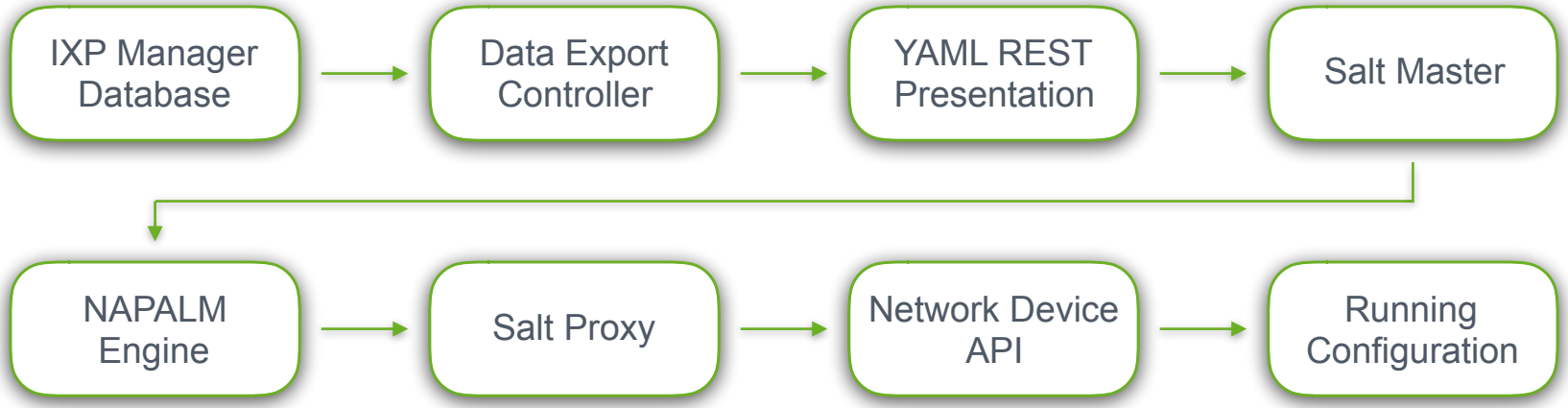
IXP AUTOMATION

INEX Kit Manifest

	Brocade FI	Brocade NI	Extreme	Arista EOS	Cumulus
INEX Lifecycle	EOL	EOL	Mid life	Early life	Pre-Deploy
API Support	None	Some YANG	XOS v21+	Excellent	Linux
Openflow	No	v1.3	v1.3	v1.3	No
NAPALM	No	No	*Not yet	Yes	No
Assessment	No plans	No plans	Partial support	Full Support	Full Support

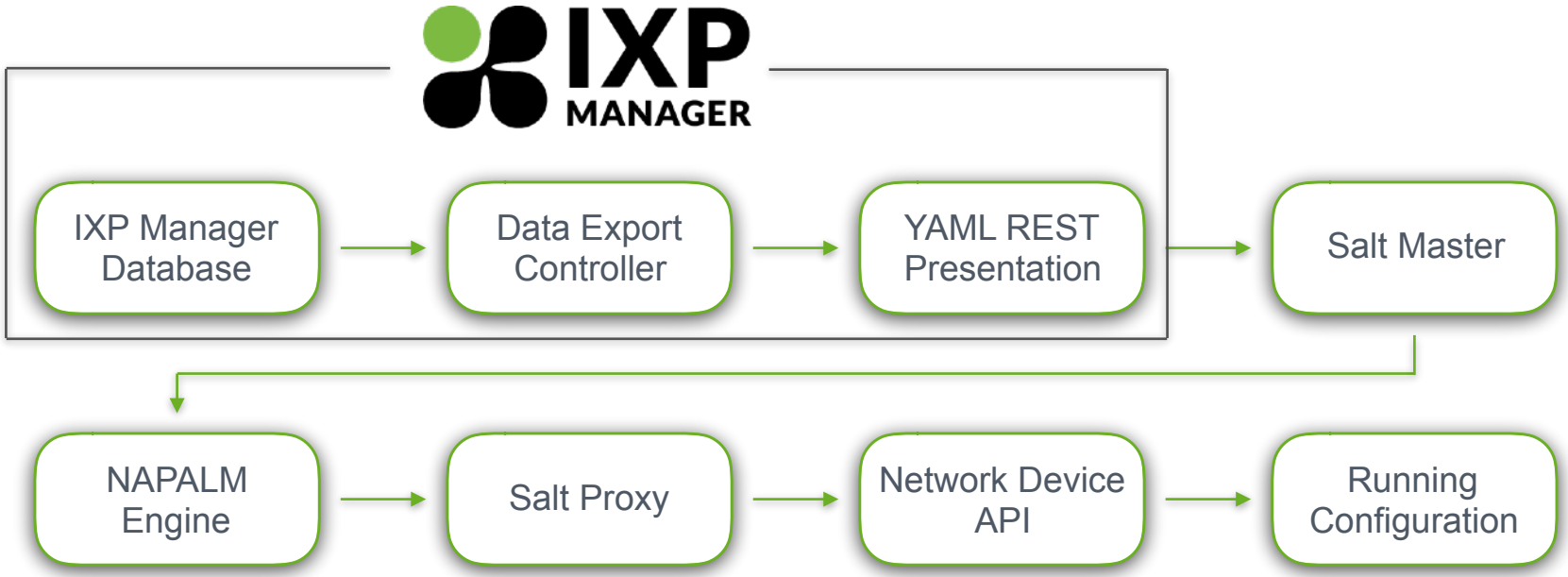
IXP AUTOMATION

Data Flow - Traditional NOS



IXP AUTOMATION

Data Flow - Traditional NOS



IXP AUTOMATION

IXP Manager Data Presentation

- API version 4 exports YAML via REST calls

- Exported data roughly breaks down as:
  - Vlans
  - Layer 2 interface information
  - Layer 3 interface information
  - Information required for routed core (bgp + vxlan)

IXP AUTOMATION

Sample YAML

```
layer2interfaces:
  - name: swp2
    type: edge
    description: "Packetloss Services Ltd"
    dot1q: yes
    shutdown: yes
    autoneg: yes
    speed: 10000
    lagindex: 1
    lagmaster: no
    fastlacp: yes
    virtualinterfaceid: 334
    vlans:
      - number: 12
        macaddress:
          - "54:1e:56:35:77:d0"
```

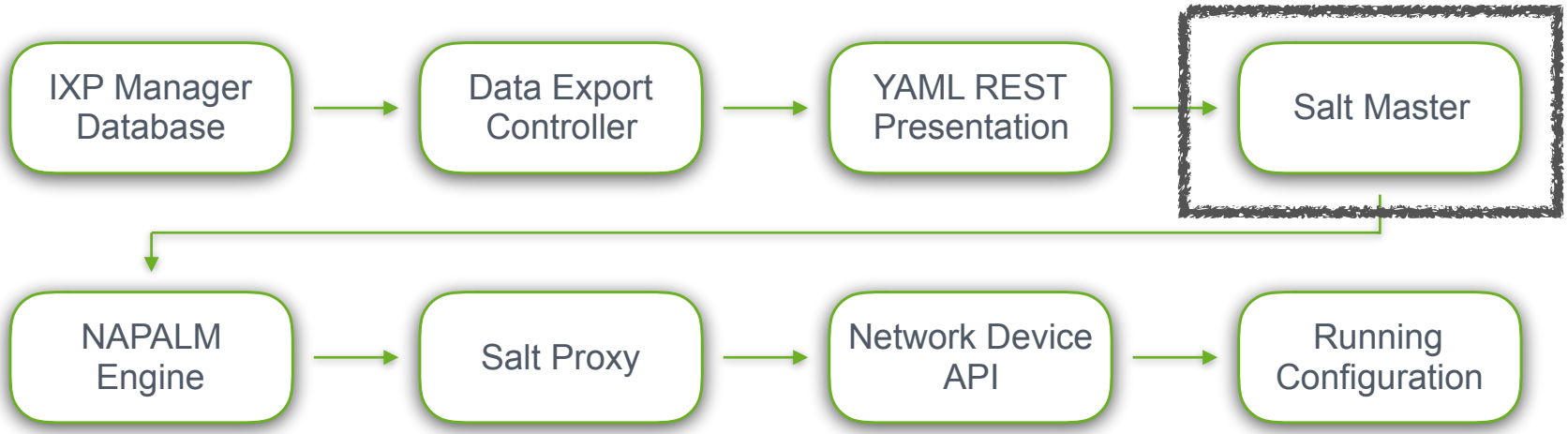
IXP AUTOMATION

Sample YAML

```
layer2interfaces:
  - name: swp49
    type: core
    description: "edge1-edge2"
    dot1q: yes
    shutdown: no
    stp: yes
    cost: 100
    autoneg: yes
    speed: 40000
    lagindex: 1010
    lagmaster: no
    virtualinterfaceid: 342
    vlans:
      - number: 12
      - number: 32
```

IXP AUTOMATION

Data Flow - Traditional NOS



IXP AUTOMATION

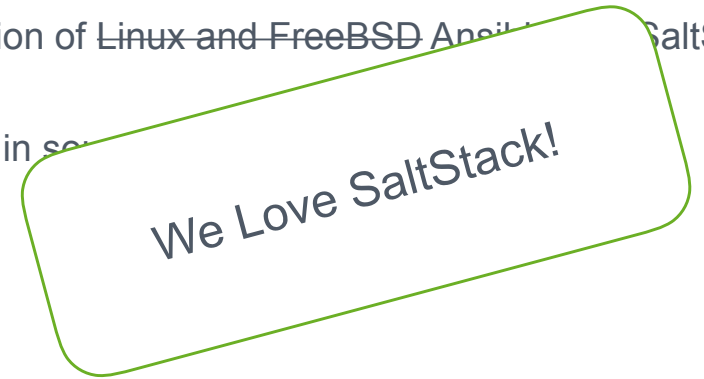
~~Vi vs Emacs~~ Ansible vs SaltStack

- Lengthy evaluation process
- Careful consideration of ~~Linux and FreeBSD~~ Ansible and SaltStack pros / cons.
- Rationale resulted in sound engineering decision:

IXP AUTOMATION

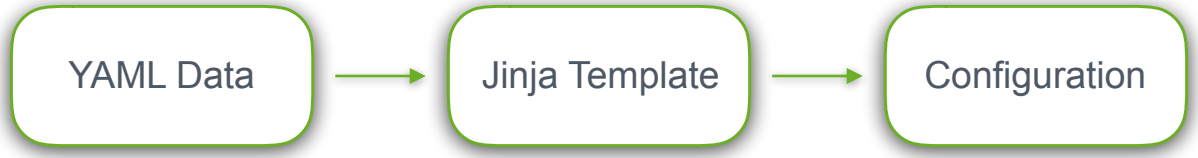
~~Vi vs Emacs~~ Ansible vs SaltStack

- Lengthy evaluation process
- Careful consideration of ~~Linux and FreeBSD~~ Ansi[illegible] SaltStack pros / cons.
- Rationale resulted in se[illegible]



IXP AUTOMATION

Data Templating



IXP AUTOMATION

Sample Jinja

```
{% if pillar.get('layer2interfaces') is iterable %}
{% for iface in pillar.get('layer2interfaces') %}

default interface {{ iface.name }}
interface {{ iface.name }}
 load-interval 30

{% if iface.description|default(false) %}
 description {{ iface.description }}
{% else %}
 no description
{% endif %}

[...]

{% endfor %}
{% endif %}
```

IXP AUTOMATION

Sample Jinja

```
{% if iface.speed == 100 %}
 speed forced 100full
{% elif iface.speed == 1000 %}
{% if not iface.autoneg|default(false) %}
 speed forced 1000full
{% else %}
 speed auto
{% endif %}
{% elif iface.speed == 10000 %}
{# speed auto #}
{% elif iface.speed == 40000 %}
 speed forced 40gfull
{% elif iface.speed == 100000 %}
 speed forced 100gfull
{% endif %}
```

IXP AUTOMATION

Idempotent Atomic Session-Based Configuration Merge

```
{% if bgp.local_as|default(false) %}
no router bgp {{ bgp.local_as }}
router bgp {{ bgp.local_as }}
   no bgp default ipv4-unicast
   bgp always-compare-med
[...]
{% endif %}


{% for iface in pillar.get('layer2interfaces') %}
default interface {{ iface.name }}
interface {{ iface.name }}
 load-interval 30
[...]
{% endfor %}
```

IXP AUTOMATION

Modelling Problems

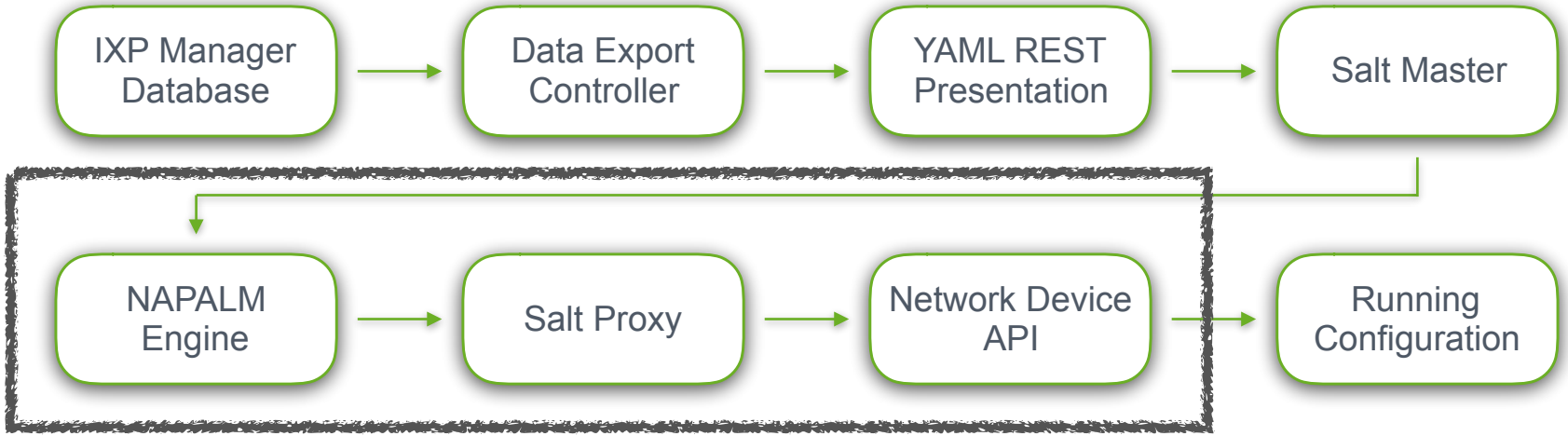
- Different switches use different data models for configuration

- E.g. Link Aggregation
  - Brocade uses physical interfaces and blocks changes on non-master after initial config
  - Extreme has a separate configuration item: “enable sharing XX”
  - Other devices use a virtual interface (Port-ChannelX, bondY, etc)
    - Even then, not all the semantics are the same (channel-group vs bond-slaves)

- Lessons learned: ensure your data model is flexible enough to support substantial semantic differences between device config models, and that it can be extended easily

IXP AUTOMATION

Data Flow - Traditional NOS



IXP AUTOMATION

NAPALM Support

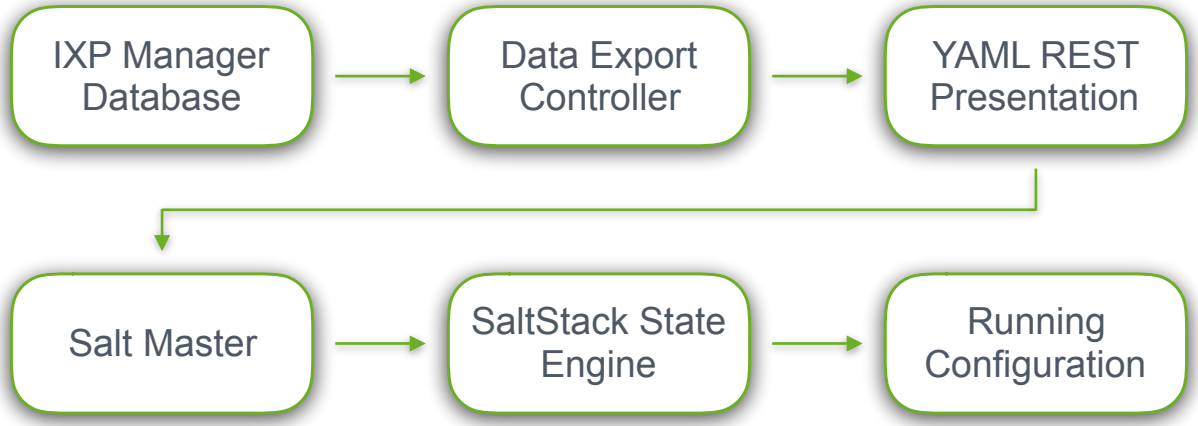
[2] Hand-crafted by the API as the device doesn't support the feature.
[3] Not supported but emulated. Check caveats.
[4] Check the caveats, this is a dangerous operation in this device.
[5] For merges, the diff is simply the merge config itself. See caveats.
[6] No for merges. See caveats.

	EOS	JunOS	IOS-XR	FortiOS	NXOS	IOS	Pluribus	PANOS	MikroTik	VyOS
Config Replace	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Config Merge	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	No	Yes
Config Compare	Yes	Yes	Yes[2]	Yes[2]	Yes[5]	Yes	No	Yes	No	Yes
Atomic Change	Yes	Yes	Yes	No[3]	Yes/No[6]	Yes	Yes	Yes/No[6]	No	Yes
Rollback	Yes[3]	Yes	Yes	Yes	Yes/No[6]	Yes	No	Yes	No	Yes

https://napalm.readthedocs.io/en/latest/support/index.html

IXP AUTOMATION

Data Flow - Cumulus Linux



IXP AUTOMATION

Deployment Workflow

- NAPALM supports config test, config load, commit and rollback
- SaltStack and Ansible support multiple deployment environments
  - e.g. lab / production, etc
- Good idea to use these mechanisms on production systems

IXP AUTOMATION

```
root@saltmaster:~ # salt swi1-pwt1-1 saltutil.refresh_pillar
[...]
root@saltmaster:~ # salt swi1-pwt1-1 net.load_template \
         /srv/napalm/templates/eos/configure_cust_interfaces.j2 saltenv=production test=true
swi1-pwt1-1:
    ----------
    already_configured:
        False
    comment:
        Configuration discarded.
    diff:
        @@ -500,7 +500,7 @@
            10 permit 30:b6:4f:e4:f8:f6 00:00:00:00:00:00 any
         !
         mac access-list l2acl-ixp-viid325
        -   10 deny any any
        +   10 permit 01:23:45:67:89:ab 00:00:00:00:00:00 any
         !
         mac access-list l2acl-ixp-viid326
            10 deny any any
    loaded_config:
    result:
        True
root@saltmaster:~ #
```

IXP AUTOMATION

```
root@saltmaster:~ # salt swt-cwt1-edge1 state.apply cumulus.configure_bgp saltenv=lab test=true
swt-cwt1-edge1:
----------
          ID: /etc/frr/frr.conf
    Function: file.managed
      Result: None
     Comment: The file /etc/frr/frr.conf is set to be changed
     Started: 08:51:09.367960
    Duration: 89.872 ms
     Changes:
              ----------
              diff:
                  ---
                  +++
                  @@ -16,12 +16,6 @@
                    neighbor pg-ebgp-ipv4-ixp description eBGP IXP session policy
                    neighbor pg-ebgp-ipv4-ixp timers 3 10
                    neighbor pg-ebgp-ipv4-ixp capability extended-nexthop
                  - neighbor 10.37.4.1 remote-as 65302
                  - neighbor 10.37.4.1 peer-group pg-ebgp-ipv4-ixp
                  - neighbor 10.37.4.1 description swt-cwt1-edge2
                  - neighbor 10.37.4.3 remote-as 65302
                  - neighbor 10.37.4.3 peer-group pg-ebgp-ipv4-ixp
                  - neighbor 10.37.4.3 description swt-cwt1-edge2
                    neighbor 10.37.2.2 remote-as 65311
                    neighbor 10.37.2.2 peer-group pg-ebgp-ipv4-ixp
                    neighbor 10.37.2.2 description swt-cwt1-mlnx1
----------
```

IXP AUTOMATION

```
[...]
----------
          ID: /etc/frr/frr.conf
    Function: service.running
        Name: frr
      Result: None
     Comment: Service is set to be reloaded
     Started: 08:51:09.803190
    Duration: 314.587 ms
     Changes:

Summary for swt-cwt1-edge1
------------
Succeeded: 5 (unchanged=2, changed=1)
Failed:    0
------------
Total states run:     5
Total run time:   1.872 s
root@saltmaster:~ #

root@saltmaster:~ # salt swt-cwt1-edge1 state.apply cumulus.configure_bgp saltenv=lab test=false
[...]
```

IXP AUTOMATION

Phase 1 Results

- Configure all IXP participant edge ports [in service using production DB]
- Configure IXP core [in service using pilot model data source]
- Handled INEX LAN1 forklift upgrade successfully
- Operations workflow changed to be safer, simpler and more reliable
- Single source of authoritative data about network configuration

IXP AUTOMATION

Phase 2 Progress

- Data abstraction model complete, refactoring complete, awaiting review
- IXP Manager: coding nearly complete, needs refactoring
- Templating for NAPALM / SaltStack: Arista: 100%, Cumulus: 97%
- Release candidate in production at INEX (still ironing out bugs!)
- Release as open source: planned in 2017Q4
- Documentation and creation of suggested operational workflow procedures

IXP AUTOMATION

Phase 2 Progress

- Documentation and creation of suggested operational workflow procedures

IXP AUTOMATION

Phase 2 Progress

- Documentation and creation of suggested operational workflow procedures

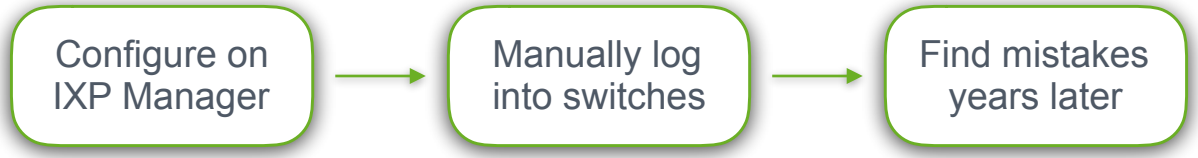
IXP AUTOMATION

Phase 2 Progress

- Documentation and creation of suggested operational workflow procedures

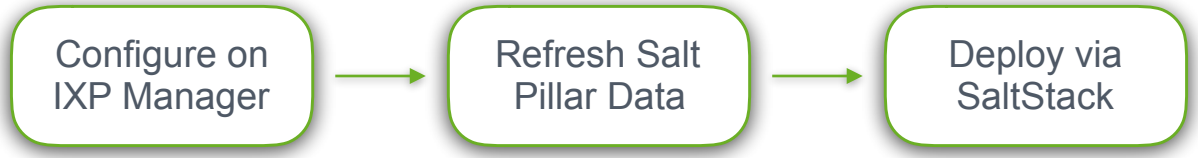
IXP AUTOMATION

Operations Then



IXP AUTOMATION

Operations Now



THANK YOU!

INEX

IXP MANAGER

Internet Society

NETFLIX

swissix
SWISS INTERNET EXCHANGE

GRIX

NIX
NORWEGIAN INTERNET EXCHANGE

STHIX
STOCKHOLM INTERNET EXCHANGE

github.com/inex/ixp-manager