Scaling the Facebook backbone through Zero Touch Provisioning (ZTP)

David Swafford
Network Engineer
Building a new POP (Point of Presence)

- Build out-of-band network
- Build Optical Network
- Build IP network
- Provision compute
Building the out-of-band network

- Install Internet service
- Provision firewall
- Provision console servers
- Provision management switches
Bringing in optical connectivity

- Trench fiber
- Provision optical line system
- Provision client transponders
- Provision waves
Building the IP Network

Diagram showing a network with routers and switches connected.
Provisioning one of our edge routers

- Letting People Know
- Rack and Stack
- Cabling
- Management IP assignment
- Config Generation
- Software Upgrades
- Loading Config
- Validating Config
- Validating Hardware (Fans, Power Supplies, Linecards)
- Validating Physical Connectivity (LLDP and Light Levels)
- Validating Logical Connectivity (Protocols)
- Updating External Systems (Location Data, Status)
- Undraining Traffic
What was already solved?

- Letting People Know
- Rack and Stack
- Cabling
- Management IP assignment
- Config Generation
- Software Upgrades
- Loading Config
- Validating Config
- Validating Hardware (Fans, Power Supplies, Linecards)
- Validating Physical Connectivity (LLDP and Light Levels)
- Validating Logical Connectivity (Protocols)
- Updating External Systems (Location Data, Status)
- Undraining Traffic
30 steps involving 10+ tools...

MOPs?
We wanted push button!
Major Pieces Needed

A method to quickly and reliably:
- apply configuration to a blank device
- upgrade software

Software for:
- notifying people
- checking hardware
- updating our asset management system
- changing BGP policy to enable traffic

Empower and enable our engineers!
Options for loading configuration

```
CONSOLE >>  '\r'
CONSOLE <<  'login:'
CONSOLE >>  'root\r'
CONSOLE <<  'password:'
CONSOLE >>  '\r'
CONSOLE >>  'router>'
CONSOLE <<  'enable\r'
CONSOLE >>  'router#'
CONSOLE >>  'config t\r'
CONSOLE <<  'router(config)#'
...
```
Options for loading configuration

- OUT-OF-BAND MANAGEMENT
- ROUTER
- FIREWALL
- CONSOLE
- SWITCH

- \texttt{ETH0 \textgreater \textless DHCPDISCOVER}
- \texttt{ETH0 \textless \textgreater DHCPOFFER}
- \texttt{ETH0 \textgreater \textless DHCPREQUEST}
- \texttt{ETH0 \textless \textgreater DHCPACK}
- \texttt{ETH0 \textgreater \textless HTTP-REQUEST}
- \texttt{ETH0 \textless \textgreater HTTP-RESPONSE}
- \texttt{ETH0 \textgreater \textless HTTP-REQUEST}
- \texttt{ETH0 \textless \textgreater HTTP-RESPONSE}

...
Replacing MOPs with Vending Machine
Automating the MOPs?

- We needed to write a LOT of code.
- We needed a workflow automation system.
- We needed to replace the MOPs.
How? Divide and conquer!

- The system was built for the network engineer
- We removed the barriers
- We empowered our peer network engineering teams
Building for the network engineer

- Small, independent pieces of code written in any programming language
- Steps should do only one thing
- Knowledge of "the system" should not be required
How? Isolate "the system" from the workflow

- Units of work are called Steps
- A Step is a compiled piece of code that is executed as a binary
- Testing and development reduced to only your step
Giving the system a name

- We named it Vending Machine!
- Vending Machine is a purpose-built workflow automation system created around Zero Touch Provisioning
- Stability in step-level isolation
Provisioning redefined

- Letting People Know
- Rack and Stack
- Cabling
- Management IP assignment
- Config Generation
- Software Upgrades
- Loading Config
- Validating Config
- Validating Hardware (Fans, Power Supplies, Linecards)
- Validating Physical Connectivity (LLDP and Light Levels)
- Validating Logical Connectivity (Protocols)
- Updating External Systems (Location Data, Status)
- Undraining Traffic
Zero Touch Provisioning
Requesting a ZTP agent over DHCP

DHCP DISCOVER

OPTION 60, VENDOR-CLASS: "VENDORX;MODEL1001;ABCD1234"

DHCP OFFER

OPTION 67: BOOTFILE-NAME:
HTTP://VM/ABCD1234/AGENT.PY
Requesting a ZTP agent over DHCP

HTTP-GET

HTTP://VM/ABCD1234/AGENT.PY

HTTP-REPLY

OK:
<body>
...binary data of script
</body>

DHCP SERVER

VENDING MACHINE
Building a feedback loop

HTTP-GET /start/<SN>

/complete/<SN>

OK: "{'JOB_ID': '1'}"

OK
Delaying ZTP while running other Steps

HTTP-GET agent.py

404 Not Found

Router

Vending Machine

CONFIG
GENERATION

ZTP

SSH Check
ARE YOU UP YET?
BGP PEERS UP?
UNDRAIN TRAFFIC
DONE
YES!
YES!

ROUTER

VENDING MACHINE
Writing a Vending Machine Step
import json
import logging
import sys

def main():
    stdin = sys.stdin.read().strip()
    input = json.loads(stdin)
    hostname = input['hostname']

    logging.info(f'Generating configs for {hostname}

    build_configs(hostname)

    verify_configs(hostname)
Apache Thrift's client example:
```python
#!/usr/bin/python3

import urllib3

VM_VIP = '2a03:2880:f101:83:face:b00c:0:25de'

def verify_configs(self, hostname):
    with urllib3.PoolManager() as http:
        url = f'http://{VM_VIP}/{hostname}/config.conf'
        response = http.request('GET', url)

        if response.status == 200:
            logging.info(f'Successfully fetched config from {url}')
            sys.exit(0)
        else:
            logging.error(f'Failed to fetch config from {url}')
            sys.exit(1)
```

Config Generation
STDIN: 
'{"asset_id": "10001", "hostname": "router1", "serial": "AAEF0016", "job_id": "1", "attempt_id": "1"}"

STDERR: 
INFO: Generating configs for router1...
INFO: Generated new configs!
EXIT_SUCCESS
Vending Machine Internals
Design Goals

- Flexibility and Rapid Development
- Scalable
- Fast
- Resilient
- Predictable
Coordinating Jobs

- **MySQL DB**
- **Zookeeper Queue**
- **Controller**
- **Executor**

The diagram illustrates the relationships between different components, with arrows indicating the flow of coordination and data transfer.
Distributing the Work

ZOOKEEPER QUEUE

Job: 1
Step: are_we_up_yet

CONTROLLER

QUEUES STEP
Distributing the Work

ZOOKEEPER QUEUE

<table>
<thead>
<tr>
<th>Job</th>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>are_we_up_yet</td>
</tr>
<tr>
<td>2</td>
<td>erase_device</td>
</tr>
<tr>
<td>2</td>
<td>are_we_up_yet</td>
</tr>
</tbody>
</table>

CONTROLLER

0 1 2

QUEUE POSITION

QUEUES STEP
Distributing the Work

ZOOKEEPER QUEUE

CONTROLLER

EXECUTOR

Job: 1
Step: are_we_up_yet

Job: 2
Step: erase_device

Job: 2
Step: are_we_up_yet

NEW VERSION AVAILABLE?

ARE WE UP YET

REPO
Distributing the Work

**ZOOKEEPER QUEUE**

- Job: 1
  - Step: are_we_up_yet

- Job: 2
  - Step: erase_device

- Job: 2
  - Step: are_we_up_yet

**EXECUTOR**

- `{"asset_id": "10001", "hostname": "router1", "serial": "AAEF0016", "job_id": "1", "attempt_id": "1"}`

**STDIN:**

```
ARE WE UP YET
```

**CONTROLLER**

0 1 2

**QUEUES**

**STEP**

**QUEUE POSITION**
Distributing the Work

**ZOOKEEPER QUEUE**

- **Job**: 2
  - **Step**: erase_device
- **Job**: 2
  - **Step**: are_we_up_yet
- **Job**: 1
  - **Step**: are_we_up_yet

**ERROR:**
Device not up yet!

**EXECUTOR**

**ARE WE UP YET**

**EXIT_FAILURE**
Transient Failures

Traceback (most recent call last): File "<stdin>", line 1, in <module> File "/usr/lib64/python2.7/socket.py", line 224, in meth return getattr(self._sock,name)(*args) socket.error: [Errno 111] Connection refused
What to do?

Device

SN: ABCD1234
MAKE: WELLFLEET
MODEL: BNX
LOCATION: DEN

Target

SN: *
MAKE: FACEBOOK
MODEL: WEDGE
LOCATION: *

MATCH?
What to do?

Device

SN: ABCD1234
MAKE: WELLFLEET
MODEL: BNX
LOCATION: DEN

Target

SN: *
MAKE: FACEBOOK
MODEL: WEDGE
LOCATION: *

MATCH?
No Match
What to do?

Device

SN: ABCD1234
MAKE: WELLFLEET
MODEL: BNX
LOCATION: DEN

Target

SN: *
MAKE: FACEBOOK
MODEL: WEDGE
LOCATION: *

SN: *
MAKE: WELLFLEET
MODEL: *
LOCATION: *

SN: *
MAKE: WELLFLEET
MODEL: *
LOCATION: DEN

MATCH?

MOST SPECIFIC
Going Beyond the Device

- DRAIN PLANE 2
- UNDRAIN PLANE 2
vm configure router1

Warning: `vm configure` is meant to be used only for a factory-blank device.
If you are trying to (re)-configure an existing device, please use `vm reconfigure` instead.

Continue (y/N): y
Started CONFIGURE job 70110
run 'vm detail 70110' to see the job status
## vm detail

**Job 70101:** 100% ![progress bar]

**Device:** router1

<table>
<thead>
<tr>
<th>Name</th>
<th>Job</th>
<th>Status</th>
<th>Att</th>
<th>Started</th>
</tr>
</thead>
<tbody>
<tr>
<td>are_we_down_yet</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:27:33.809</td>
</tr>
<tr>
<td>bbe_edit_popbuilder_cfg</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:27:45.975</td>
</tr>
<tr>
<td>funet_mgmt_ip</td>
<td>---</td>
<td>DONE</td>
<td>6/30</td>
<td>2018-05-17 13:30:17.935</td>
</tr>
<tr>
<td>set_provisioning_status</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:27:46.176</td>
</tr>
<tr>
<td>set_self_provisioning_status</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:27:46.270</td>
</tr>
<tr>
<td>set_backbone_global_mesh_status</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:30:52.982</td>
</tr>
<tr>
<td>bbe_config_gen</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:31:01.391</td>
</tr>
<tr>
<td>bbe_remote_push</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:34:03.267</td>
</tr>
<tr>
<td>bbe_bgp_push</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:35:00.434</td>
</tr>
<tr>
<td>ztp</td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:46:55.521</td>
</tr>
<tr>
<td>bbe_evpn_nabndhechk</td>
<td>---</td>
<td>DONE</td>
<td>1/40</td>
<td>2018-05-17 13:48:55.133</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:49:38.280</td>
</tr>
<tr>
<td></td>
<td>---</td>
<td>DONE</td>
<td>1/30</td>
<td>2018-05-17 13:49:50.761</td>
</tr>
</tbody>
</table>
vm log tail

INFO [bb_connectivity_check_mpls#1] Checking "show mpls lsp egress" command output
INFO [bb_connectivity_check_mpls#1] Checking "show mpls lsp ingress" command output
INFO [bb_connectivity_check_mpls#1] LSPs Up: 363 egress and 317 ingress
INFO [bb_connectivity_check_mpls#1] LSPs Down: 0 egress and 0 ingress
INFO [bb_connectivity_check_mpls#1] LSPs Configured: 363 egress and 317 ingress
INFO [bb_connectivity_check_mpls#1] MPLS Health Checks Pass for router1
INFO [bb_connectivity_check_mpls#1] executor: attempt succeeded
INFO Job 70101 finished.
WHAT WOULD YOU DO IF YOU WEREN'T AFRAID?