Network Automation with Salt and NAPALM

Mircea Ulinic
Cloudflare, London

NANOG 68 Dallas, TX
October 2016
Cloudflare (a quick background)

- Once a website is part of the Cloudflare community, its web traffic is routed through our global network of 100+ locations
- How big?
  - Four+ million zones/domains
  - Authoritative for ~40% of Alexa top 1 million
  - 43+ billion DNS queries/day
    - Second only to Verisign
- 100+ anycast locations globally
  - 49 countries (and growing)
- Origin CA
Why automate?

- Deploy new PoPs
- Human error factor
- Replace equipment
- Monitor
- Much faster recovery
Automation framework requirements

- Very scalable
- Concurrency
- Easily configurable & customizable
- Config verification & enforcement
- Periodically collect statistics
- Native caching and drivers for useful tools
Why Salt?
Orchestration vs. Automation
<table>
<thead>
<tr>
<th>Salt</th>
<th>Ansible</th>
</tr>
</thead>
<tbody>
<tr>
<td>what best fits our needs</td>
<td>most used in network automation</td>
</tr>
<tr>
<td>● Long standing sessions</td>
<td>● open/close session per module</td>
</tr>
<tr>
<td>● Real-time job</td>
<td>● Real-time job (Tower: $$)</td>
</tr>
<tr>
<td>● Job scheduling</td>
<td>● Job Scheduling (Tower: $$)</td>
</tr>
<tr>
<td>● REST API</td>
<td>● REST API (Ansible Tower: $$)</td>
</tr>
<tr>
<td>● High Availability</td>
<td>● HA (Ansible Tower: $$)</td>
</tr>
<tr>
<td>● GPG encryption</td>
<td>● Security (Tower: $$)</td>
</tr>
<tr>
<td>● Pull from Git, SVN</td>
<td>● Pull from Git, SVN (Tower, $$)</td>
</tr>
</tbody>
</table>
Salt at Cloudflare: used for years

Multiple thousands of servers managed by Salt
Same tool for both servers and net devices
Why NAPALM?

(Network Automation and Programmability Abstraction Layer with Multivendor support)

https://github.com/napalm-automation
Open source recipe for network orchestration: napalm-salt

https://github.com/napalm-automation/napalm-salt
NAPALM-Salt for Public use

- NAPALM integrated in next release of Salt
- Execution Modules (selection)
  - NET
  - BGP
  - NTP
  - Probes
- States:
  - NTP
  - Probes
NAPALM-Salt (examples):

1. salt "edge*" net.traceroute 8.8.8.8
2. salt -N EU transit.disable telia  # disable Telia in EU
3. salt -G "os:junos" net.cli "show version"
4. salt -C "os:iosxr and version:6.0.2" net.arp
5. salt -G "model:MX480" probes.results
6. salt -l "type:router" ntp.set_peers 10.1.130.10 10.1.130.18 10.1.130.22

# salt --out=json edge01.dfw01 net.arp

```
[
    {
        "interface": "ae2.100",
        "ip": "10.0.0.1",
        "mac": "00:0f:53:36:e4:50",
        "age": 129.0
    },
    {
        "interface": "xe-0/0/3.0",
        "ip": "10.0.0.2",
        "mac": "00:1d:70:83:40:c0",
        "age": 1101.0
    },
    {
        "interface": "xe-0/0/3.0",
        "ip": "10.0.0.3",
        "mac": "10:0e:7e:de:84:07",
        "age": 1276.0
    }
]
```

# salt --out=yaml edge01.dfw01 net.arp

```
edge01.dfw01:
    comment: ''
    out:
        - age: 129.0
          interface: ae2.100
          ip: 10.0.0.1
          mac: 00:0f:53:36:e4:50
        - age: 1101.0
          interface: xe-0/0/3.0
          ip: 10.0.0.2
          mac: 00:1d:70:83:40:c0
        - age: 1276.0
          interface: xe-0/0/3.0
          ip: 10.0.0.3
          mac: 10:0e:7e:de:84:07
        - age: 1390.0
          interface: xe-0/0/3.0
          ip: 10.0.0.4
          mac: 00:24:38:97:39:06
```

Many others renderers: [https://docs.saltstack.com/en/latest/ref/renderers](https://docs.saltstack.com/en/latest/ref/renderers)
Abstracting configurations

protocols {
  bgp {
    group 4-PUBLIC-ANYCAST-PEERS {
      neighbor 192.168.0.1 {
        description "Amazon [WW HOSTING ANYCAST]";
        family inet {
          unicast {
            prefix-limit {
              maximum 500;
            }
          }
        }
      }
      peer-as 16509;
    }
  }
}

router bgp 13335
  neighbor 192.168.0.1
    remote-as 16509
    use neighbor-group 4-PUBLIC-ANYCAST-PEERS
    description "Amazon [WW HOSTING ANYCAST]"
    address-family ipv4 unicast
    maximum-prefix 500

bgp.neighbor:
  ip: 192.168.0.1
  group: 4-PUBLIC-ANYCAST-PEERS
  description: "Amazon [WW HOSTING ANYCAST]"
  remote_as: 16509
  prefix_limit: 500
Example

- Edge router with 1000 BGP peers
- Device is manufactured by VendorA
- Replaced by a device from VendorB
Most network engineers
vi /etc/salt/pillar/edge01_dfw01.sls

More about pillars: https://docs.saltstack.com/en/latest/topics/pillar/
Scheduled operations - all integrated!

# Redis details:
redis.host: localhost
redis.port: 6379

# Schedulers
schedule:
  traceroute_runner:
    function: traceroute.collect
    hours: 4

2071) "traceroute:edge01.sjc01-edge01.lhr01-Tata-4"
2072) "traceroute:edge01.iad02-edge01.sjc01-GTT-4"
2074) "traceroute:edge01.fra03-edge01.sea01-Cogent-4"
2075) "traceroute:edge01.yul01-edge01.lax01-Cogent-4"
2076) "traceroute:edge01.zrh01-edge01.fra03-GTT-4"
2077) "traceroute:edge01.mxp01-edge01.ams01-GTT-4"
2078) "traceroute:edge01.mia01-edge01.lhr01-GTT-4"
2079) "traceroute:edge01.msp01-edge01.scl01-Telefonica-4"
2080) "traceroute:edge01.fra03-edge01.mia01-Telia-4"
2081) "traceroute:edge01.lim01-edge01.scl01-Telefonica-4"
2082) "traceroute:edge01.arn01-edge01.mia01-GTT-4"
2083) "traceroute:edge01.prg01-edge01.lax01-GTT-4"
2084) "traceroute:edge01.osl01-edge01.lhr01-GTT-4"
Maintain configuration updated

Define NTP peers in the Pillar

```
ntp.peers:
- 10.1.130.22
- 10.1.130.18
- 10.1.128.10
- 10.1.131.10
- 10.1.132.10
- 10.2.52.10
- 10.2.48.10
- 10.2.55.10
- 10.2.50.10
- 10.2.56.10
```

Schedule config enforcement checks

```
schedule:
    ntp_config:
        function: state.sls
        args: router.ntp
        returner: smtp
        days: 1
    bgp_config:
        function: state.sls
        args: router.bgp
        hours: 2
    probes_config:
        function: state.sls
        args: router.probes
        days: 3
    users_config:
        function: state.sls
        args: router.users
        returner: hipchat
        weeks: 1
```

More about states: https://docs.saltstack.com/en/latest/topics/tutorials/starting_states.html
NTP state output example

edge01.jnb01:
----------
ID: ntp_config
Function: netntp.managed
Result: True
Started: 09:50:41.228728
Duration: 16813.319 ms
Changes:
----------
peers:
----------
removed:
- 10.10.1.1
servers:
----------
added:
- 17.xxx.xx.253
- 40.xxx.xxx.7
removed:
- 83.xxx.xxx.118
- 92.xx.xxx.58
- 91.xx.xxx.42

Summary for edge01.jnb01
----------
Succeeded: 1 (changed=1)
Failed: 0
----------
Total states run: 1
Total run time: 16.813 s
<table>
<thead>
<tr>
<th>Device</th>
<th>Interface</th>
<th>Description</th>
<th>IP</th>
<th>Enabled</th>
<th>Speed (Max)</th>
<th>MTU Address</th>
<th>IPv6 Address</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>null</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>null</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>null</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>null</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>null</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Remote "on 0/1/1" Name is one of the following UDF details:

<table>
<thead>
<tr>
<th>Device</th>
<th>Interface</th>
<th>Remote System Name</th>
<th>Remote System Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>

ARP MIB for interface 0/1/1:

<table>
<thead>
<tr>
<th>Device</th>
<th>Interface</th>
<th>Remote System Name</th>
<th>Remote System Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>2950-48</td>
<td>1/3/1</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>

22
A self-resilient network
Monitoring carriers (transit providers)

mircea@re0.edge01.dfw01> show configuration services rpm | display set | match 1299 | match probe-type
set services rpm probe transit test t-edge01.scl01-1299-12956-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.eze01-1299-6762-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.lax01-1299-1299-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.eze01-1299-12956-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.mia01-1299-1299-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.lhr01-1299-1299-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.ams01-1299-1299-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.fra03-1299-1299-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.iad02-1299-1299-4 probe-type icmp-ping
set services rpm probe transit test t-edge01.sea01-1299-1299-4 probe-type icmp-ping

JunOS: RPM
https://www.juniper.net/documentation/en_US/junos12.1x46/topics/concept/security-rpm-overview.html

IOS-XR: ISPLA
How many probes?

$ sudo salt-run transits.probes show_count=True

Generated 7248 probes.

Generated using:

- net.ipaddr
- net.interfaces
- bgp.neighbors
- bgp.config

All available in https://github.com/napalm-automation/napalm-salt
How are they installed?

```bash
$ cat /etc/salt/pillar/probes_edge01_dfw01.sls
probes.config:
  transit:
    t-edge01.sjc01-1299-1299-4:
      source: 1.2.3.4
      target: 5.6.7.8
    t-edge01.den01-1299-1299-4:
      source: 10.11.12.13
      target: 14.15.16.17
    t-edge01.den01-174-174-4:
      source: 18.19.20.21
      target: 22.23.24.25
    t-edge01.den01-4436-4436-4:
      source: 26.27.28.29
      target: 30.31.32.33
```

```bash
$ sudo salt 'edge*' state.sls router.probes
edge01.dfw01:
-------------
ID: cf_probes
Function: probes.managed
Result: True
Comment: Configuration updated
Started: 23:00:17.228171
Duration: 10.206 s
Changes:
-------------
added:
-------------
transit:
-------------
t-edge01.sjc01-1299-1299-4:
  probe_count: 15
  probe_type: icmp-ping
  source: 1.2.3.4
  target: 5.6.7.8
  test_interval: 3
removed:
updated:
-------------
```
Spaghetti
Retrieving probes results

$ sudo salt 'edge*' probes.results

dge01.dfw01:
    out:
    transit:
        t-edge01.sjc01-1299-1299-4:
            current_test_avg_delay: 24.023
            current_test_max_delay: 28.141
            current_test_min_delay: 23.278
            global_test_avg_delay: 23.936
            global_test_max_delay: 480.576
            global_test_min_delay: 23.105
The Internet (during a good day)
But usually the Internet looks like that
Self-resilient network
Self-resilient network: HipChat alerts

**event-action-script** · Sep-30 07:37
Cogent: Disabled in EU
Current alerts per router:
  Routers and their active alerts on transit:
  edge01.cdg01: 5
  edge01.otp01: 5
  edge01.man01: 5
  edge01.sof01: 5

**netperf** · Oct-5 10:36
[netperf] Anycast disabled on edge01.mde01

**event-action-script** · Oct-1 17:26
Comcast: Disabled in NA
Current alerts per router:
  Routers and their active alerts on transit:
  edge01.dfw01: 3
  edge01.bos01: 6
  edge01.den01: 4
  edge01.phl01: 4
  edge01.atl01: 2
How often?

1688 request-reply pairs during a random window of 7 days
~ 120 config changes / day in average
0 human intervention
How can you use it?

# apt-get install salt-master ([install guide](#))
# pip install napalm

Examples:
How can you contribute?

- NAPALM Automation: https://github.com/napalm-automation
- SaltStack https://github.com/saltstack/salt
Need help/advice?

Join https://networktocode.herokuapp.com/
rooms: #saltstack #napalm

By email:
● Mircea Ulinic: mircea@cloudflare.com
● Jerome Fleury: jf@cloudflare.com
Questions

By email:
- Mircea Ulinic: mircea@cloudflare.com
- Jerome Fleury: jf@cloudflare.com